Research Report

A Cinematographic Approach in Interactive Storytelling

Presented by
Alejandro Ramírez

Director
Dr. Josep Blat

Doctorate Program in Computer Science and Digital Communication
Universitat Pompeu Fabra
Barcelona
September 2002
# Table of Contents

1. Presentation .............................................................................................................. 1

2. Introduction .............................................................................................................. 2

3. Interactive storytelling ............................................................................................. 3
   3.1. The research field .............................................................................................. 3
       3.1.1. Approaching from a humanistic perspective ............................................. 6
   3.2. Characters .......................................................................................................... 8
       3.2.1. Intelligence .............................................................................................. 9
       3.2.2. Motivations .......................................................................................... 10
       3.2.3. Emotional components ....................................................................... 10
   3.3. More on the narrative elements ....................................................................... 12
       3.3.1. Emotions and personality ................................................................... 12
       3.3.2. Drama .............................................................................................. 14

4. Current methodologies and projects ..................................................................... 17
   4.1. Computer games and AI .................................................................................. 17
       4.1.1. NPC behavior ...................................................................................... 17
       4.1.2. Path planning and search algorithms ................................................... 21
       4.1.3. Storytelling in computer games ........................................................... 23
       4.1.4. Learning and adaptive behavior ............................................................ 24
       4.1.5. Natural language interfaces ................................................................. 27
       4.1.6. AI languages in game development ...................................................... 28
   4.2. Game engines and research ............................................................................. 29
       4.2.1. Game engines ...................................................................................... 29
       4.2.2. UnrealEd™ and Unrealscript™ ............................................................... 32
   4.3. Teeside University research ............................................................................. 36
       4.3.1. The storytelling system ......................................................................... 37
       4.3.2. AI and planning .................................................................................... 38
       4.3.2. HTNs and planning .............................................................................. 39
       4.3.3. Planning and execution ....................................................................... 41
       4.3.4. Generating stories ............................................................................... 42
       4.3.5. The user and the plot .......................................................................... 43
       4.3.6. Overview of the project ....................................................................... 44
   4.4. Other institutions and projects ......................................................................... 45
       4.4.1. CMU – Oz ............................................................................................ 45
       4.4.2. NCSU – Liquid Narrative Group ............................................................ 46
       4.4.3. SU – Virtual Theater Project ................................................................. 47
       4.4.4. UM – AI and computer games ............................................................... 49
5. Our proposal: a cinematic approach ........................................... 50
   5.1. The need for cinematographic systems .................................. 50
   5.2. Cinematic areas ........................................................................ 51
       5.2.1. Camera control ................................................................. 52
       5.2.2. Editing ............................................................................ 54
       5.2.3. Lighting control ............................................................... 55
       5.2.4. Audio ............................................................................. 56
   5.3. The project proposal ................................................................. 57
       5.3.1. Screenwriter ................................................................. 58
       5.3.2. Sound designer/composer ............................................... 58
       5.3.3. Cinematographer .......................................................... 59
       5.3.4. Editor ............................................................................. 59
       5.3.5. Director ........................................................................... 60
       5.3.6. Overall roles ................................................................. 60

6. Conclusions and perspectives ....................................................... 62
   6.1. Some personal reflections ..................................................... 62
   6.2. Conclusions ........................................................................... 63
   6.3. Research perspectives and future work .................................. 64
       6.3.1. Development ................................................................. 64
       6.3.2. Interactive narrative structure ........................................ 65
       6.3.3. Cinematic control .......................................................... 65
       6.3.4. IA for the cinematic agents ............................................. 65
       6.3.5. Definition of a cinematic ontology ................................. 66
       6.3.6. Multi-user environments ............................................... 67
       6.3.7. A closing remark .......................................................... 67

7. References ............................................................................... 68
1. Presentation

This report focuses on the research possibilities of Interactive Storytelling, concretely in the potential of developing different cinematic tools within a storytelling system, with the intention to provide strong, advanced and coherent narrative capabilities in a virtual environment where stories are generated with virtual actors. The context of the proposal is a virtual scenario, developed with the Unreal™ game engine, where virtual actors and virtual agents manage to generate a story that results from AI formalisms, such as planning techniques. The possible applications include computer-games and improved storytelling systems based in game engines.

The basic idea behind the proposal is to use and explore the cinematic language, and its elements, since not only the language has different formalisms that make it adequate to be translated to a computer system, but it is successful among both users and spectators when they experience a story that is being told. Even though users need not to be familiar with the formalisms of such a language in terms of a knowledge of its principles, they are already familiar with its results and effects in the overall storytelling process.

The exploration of the cinematic language, here with focus in the narrative elements and cinematic support of a virtual storytelling system developed with a game engine, is in conformity with the intentions of the doctoral program this document is linked to. The research and ideas proposed want to concentrate on the study and experimentation of digital media and the use of film language, with a combined approximation between engineering –our formal background– and humanities.
2. Introduction

This research report concentrates in the potential advantages and challenges that developing a cinematic system represent, in the context of interactive storytelling. The aim and proposal is, roughly, to define a set of tools, or system, that can provide cinematic capabilities to a storytelling scenario, where virtual actors generate a story and where a strong narrative is needed, in order to maintain the appeal and the coherence of any given story virtually shown and generated.

The proposal involves the development of a system capable of adding -or providing- cinematic capabilities within a storytelling system, supporting the narrative, being it a development made with the Unreal™ game engine and its extensions, and having virtual actors and virtual agents capable of creating a story (or a set of stories) based on AI formalisms, such as could be planning techniques or other decision-taking mechanisms.

Regarding its structure, the rest of the document includes different topics and presents some of the current interactive developments dealing with storytelling, together with our proposal and future perspectives. The subsequent part of this report, Section 3, provides a general overview of the research field of Interactive Storytelling, emphasizing its basic requirements, and mentioning some supplementary elements needed in a storytelling system in terms of narrative relevance. Section 4 gives an overview of the methodologies related to the subject, and includes the projects that represent the current leading developments in the area of virtual storytelling, while, in Section 5, we expose our ideas and the potential for a cinematic set of tools, and the development of cinematic improvements in the narrative components of a story which is virtually acted and generated. Some conclusions, ideas and future work and perspectives appear in Section 6.
3. Interactive storytelling

Interactive Storytelling promises to become a very important element in the development and evolution of computer games and entertainment systems, by allowing the convergence of traditional and interactive media, and by providing the capabilities to introduce improved narrative contents into this junction.

In order to present a better understanding of the research field, the following sections provide a general overview of the topic and its characteristics.

3.1. The research field

In the context of Computer Science, Interactive Storytelling involves all the projects and research that deal with the creation of dynamic narratives and narrative structures a user can interact with, in order to create some (or several) changes in the story narrated, hence affecting the way the story evolves and the way the characters interact as the result of these changes. Most of the current projects in the area [19, 36, 38, 56] make use of 3D scenarios to represent the virtual world where the stories are generated and developed, and this is just the natural response to the graphical evolution in the areas of videogames and virtual reality. Some current projects also aim to offer the opportunity to narrate a story while allowing different levels of user-intervention that can drastically interfere with the storytelling process in order to, eventually, alter its ending and overall progress [18].

Several approaches to this interaction in the stories have already been described by different researchers and institutions [17, 62, 72], specially focusing in the fact that Interactive Storytelling offers a challenging area where intelligent agents and their technology can be applied or developed. Looking at the name of the field, we can present some comments and ideas related to those terms.

Interactivity is, per se, quite a used word in several areas of research, and in the context of storytelling its implications must be clearly understood. Here, interactivity or, more precisely, the degree of interactivity needed, must include different methods that provide the capabilities to change and affect whatever is being experienced, which then incorporates and responds to different choices, the responses to these choices, and so on.

As pointed by Craven [22], interactivity has become “the Holy Grail of computer-based entertainment, although it also has proved problematic to define”. Besides the basic elements discussed by Laurel [44]: frequency (how often one can interact in the game), range (how many choices are available while playing), and significance (the impact of the choices), is important to recall the basic need she demands: “the importance of the extent to which participants feel themselves to be engaged in an ongoing interaction”.

Alejandro Ramírez
In projects like M. Cavazza’s [18], the interactivity implies direct physical interaction with virtual objects and interaction with synthetic characters through speech understanding. Physical intervention exists for the user in on-stage interaction, enabling him to remove or relocate objects (that have narrative significance) which are resources for character’s actions, and hence causing these actions to fail or vary.

In this case, the linguistic intervention allows the user to influence the characters in different ways, by providing them with information that will aim to solve their narrative goals (or some of them), by instructing them to take an specific action, or by giving advice on what may be an appropriate behavior.

The paradigm of interactive storytelling hence faces the fact of providing ways to change the story, as the opposite conditions experienced, for example, at the conventional cinema, where, whatever our responses to a movie may be, the film is itself a completed whole, where we cannot actively change the story. In plain words: it is what it is.

The story in the new interactive media (and research) is then radically different from what traditional media is used to: the new media demands users to think and re-conceive the act of story creation, and somehow learn to tell the interactive stories from its innermost parts to the external-visible- ones. In the ideal case, the new games or interactive entertainment systems can allow changes that will impact a story and, probably, also its ending.

The new paradigm can allow the development of a virtually acted story of Casablanca where Rick must not necessarily let Ilsa leave, just to mention a concrete example we find descriptive enough.

![Figure 1: The story in Casablanca cannot be changed.... Or can it?](image)

This paradigm, where a story cannot exist separately from the whole piece, raises many questions and issues, specially if we consider the whole set of possibilities
that arise from this emergent set of research fields that deal with different ways to satisfy one of our most human descriptors: the creation of a story, and now the capabilities to experience this story in a different way compared to the conventional methods.

In the context of the interactive storytelling, the story is not fully told until the final moment of the story is played, and here is where an active viewer (or, more precisely within our context, a player) comes in. Playing is the key concept, as it implies actions (and reactions) that were not necessary in conventional storytelling. The process of creating a story then changes the role of the author, that now must replace the usual conventions and aim to maintain the ability to create emotional, immersive and enjoyable stories.

As mentioned, the traditional model of storytelling implies a story that is already complete and has no potential for changing. Here, the viewer receives a completed story and extracts the meaning and emotions from it. The current paradigm is the exact opposite: the story cannot be conceived without the active participation of a player, that now takes an active role and helps to shape the story. The author now must set up a potential story (this is, potential story elements) in order to create a story that emerges from the interaction of a user.

The idea behind interactive storytelling demands then a whole set (or subsets) of potential storytelling elements, among which we can consider some elements included in the following list:

- Characters with qualities, goals and motivations;
- Settings with dynamic characteristics, possibilities and storytelling potentials;
- Objects imbued within the history, settings or characters
- Controls to shape the manipulation of object
- Controls or responses that affect the motivations, goals or attitudes of the characters;
- Facts, personality or ideas that can constantly modify themselves, possibly according to the player's interactions, or to the actions or responses of created characters, etc.

Tentatively, a defined story could have a specific subject, and all the objects in that context would relate, act or counteract with or against that specific subject. The job of the player (or, as usually mentioned, interactor) is then to manipulate the objects, interact with the characters, perhaps explore the settings, and realize the inherent goals and characteristics of each, trying to accomplish a task (or series of tasks) motivated by their own goals and ambitions, all within the general topic of the story.

We can then consider the player to be the protagonist of this new media, with a story arc, different characteristics, motivations and thoughts.
The new element for dynamism, the interactive protagonist, can then affect everything he gets in touch with, and this then will shape the relationships of characters and objects and, finally, create that which we may assertively call a story.

3.1.1. Approaching from a humanistic perspective

Why the need for further research in storytelling? At its core, conventional storytelling could be defined as the art of using a language (written, vocal, musical, gesture-based, physical movement, etc) with the aim to reveal the elements and images of a story to a specific audience. A central –and unique- aspect of conventional storytelling is its reliance on the audience to develop specific imagery and detail to complete and co-create the story.

The story implies the existence of a specific structure of narrative (with specific style, and set of characters) that includes a sense of completeness.

As human beings, stories are a sort of need. We use stories to share our experiences, to pass on our accumulated wisdom, beliefs, and values and, as pointed by Littlejohn “Humans are innately interested in conflict”[46]. Through stories we explain how things are, why they are, and our role and purpose. In other cases, stories allow us to believe something we know cannot be done, to be or become someone we are not, and to experience a set of things we cannot otherwise do, such as when we become heroes, warriors, knights or spies. Stories are a building block of knowledge, in some sense they are part of the foundation of memory, collective experience, and learning.

Stories connect us with our humanness and link the time (past, present or future) by teaching us, and by even allowing us to know the possible consequences of a set of actions.

The original (or now meaning conventional) telling part implied a live scenario, person-to-person oral and physical presentation of a story to an audience, or the
live-mental-aspect of the story created while reading. Telling involves and has involved a way of direct contact between the teller and the receiver (the listener, the viewer). It implies a direct presentation of the story by the teller, and a role: the teller's role is to prepare and present the necessary language (tone, vocalization, written style, or physicality) to effectively and efficiently communicate the images of a story.

The listener's role also exists, obviously, and it is to actively create the vivid, multi-sensory images, actions, characters, and events (the reality) of the story in their mind based on the performance by the teller, and on their past experiences, beliefs, and understandings.

The final completed story happens actually only in the mind of the listener or reader, unique and personal for each individual.

Storytelling then is, even in its conventional sense, already an interactive performance art form. Direct interaction between the teller (or the story) and the audience has always been an essential element of the whole storytelling experience. An audience must respond to the teller's words or actions, and the teller, in a live scenario, generally used some feedback to immediately, spontaneously, or with some improvisation, adjust the tone, emotions, or pace of the story to better meet the needs of the audience.

The same principle applies to the interactive new paradigm of this storytelling.

A rationalization for researching in storytelling from both a scientific and a humanistic point of view exists, and is quite straightforward. Storytelling is, by design, a co-creative process. Storytelling audiences do not passively receive a story from the teller. In the conventional sense of listening, the teller provides no visual images, and it's the listener who creates these images based on the performer's telling and on his own experiences and beliefs. In a movies or film sense of storytelling, the receiver uses the images shown and the story told to further develop an experience based on the idea expressed, on the sequences of takes, and the sequences of actions.

Storytelling is, by its nature, completely personal, interpretive, and uniquely human. Stories are a good vehicle for assessing and interpreting events, experiences, and concepts (in many fields, in many senses), and it is an intrinsic (and basic) form of human communication. In a strongly humanistic sense, the telling (and receiving) of stories is an integral and essential part of the human experience [30].

Storytelling is itself a process, a way for sharing, interpreting, offering the content and meaning of a story to an audience. Storytelling is experiential, dynamic, a way of interaction between beings, and a challenging area in all senses.
3.2. Characters

*Just because you are a character doesn't mean that you have character.*

Winston Wolf, in *Pulp Fiction*

Most of the research projects on virtual environments, interactivity, storytelling and games have helped to create a large number of research sub-fields, as well as a large number of questions regarding the true capabilities and true applications of the so-called interactive storytelling. Once we have presented the research area, we can include the characteristics of the characters in it, within the context of virtual actors in a storytelling system or project. Characters, in the context of a story, are the basic element to provide and support its narrative.

Historically, plot continuity and dramatic interaction was needed in most games and virtual environments in order to overcome the limitations of the first and basic levels of interactive games and environments, where the player was basically responsible for the whole progress of the plot, and problems arose whenever a puzzle or situation could not be solved by him, since the game or environment itself was not able to provide means to change this condition.

With this in mind, the key element of a virtual actor comes to the surface: we need to produce the illusion of an intelligent, emotional, motivated character.

In the first efforts to change the original situation, the main idea and attempts were oriented towards taking the player out of the role of being fully responsible for the flow of the plot. In her early works and book, Laurel [43, 44] explained how the user of an interactive system may indeed make some contributions related to the level of plot, but clearly stated that the responsibility for the integration of such contributions into the story, and the abilities to create and add other plot elements needed to maintain the necessary dramatic aspects, belong to the computer system. In this scenario, the user is liberated from the formal control of the action, and can immerse in the game experience.

If we go further on the idea of the first characters in computer games, we find they were too simple and had no real personality or even no personality at all. In these cases, the characters a player could find in any storytelling scenario, had only the meaning of posing a puzzle, and caused no emotional involvement or were a very important part of the overall storytelling process.

If we consider the general approach, the basic attempts to generate behavior in computer-controlled characters have followed a simple stimulus-response model. For each statement we may have given to a character, there was one response it may have displayed. Giving the same stimulus several times in a row, the character would simply repeat the same response.
On this scenario, there was clearly enough space for different ways of improvement in creating life-like characters, and here is where the field of Artificial Intelligence had a huge universe of unlimited possibilities, in the search to produce the illusion of intelligent, emotional, motivated characters.

In terms of AI, simple models of plans, emotions, or knowledge can totally be defined, and a character's plans, emotions, or knowledge can affect its behavior in a game or environment. A plan can be understood as a sequence of actions to be carried out, with the purpose of accomplishing some goal. An emotion could, for instance, be defined as a state which affects what goals are chosen, with for example, fear causing the need to escape, or hunger as the cause to begin a fight. Knowledge can be understood as a list of objects external to the character, and their relationships and attributes.

The following paragraphs provide a basic introduction to the aspects needed to produce the desired intelligent, emotional, motivated characters. The next sections address specific aspects on these considerations.

### 3.2.1. Intelligence

In order for the characters to act or appear as intelligent, they must be able to interpret the state of their environment and apply the appropriate behaviors or reaction. A certain degree of intelligent behavior and actions is required in any simulated character. The main goal is to obtain believable characters, this is, characters that convey their role in the story in a believable way [55].

A general problem domain, and extracting the principles in [57], hence contains:

- Representational primitives
- Problems (expressed in terms of those primitives)
- Problem solving procedures

In the context of storytelling, characters can apply their knowledge to solve problems, creating a story by their actions. The general term for the characters here is having *problem-solving agents*, which decide what to do by finding sequences of actions that lead to desirable state [60].

While applying a problem-solving procedure to attain a goal, however, complications may arise, and hence the need for sub-goals as the actions needed to accomplish other goals.

When a character is able to handle low level actions without being given explicit instructions, he appears much more intelligent. This method also provides a method for managing logistical details on behalf of the main player, who is then liberated to act or react at higher levels.
3.2.2. Motivations

*I'm sorry. It's written into my character to do it, so I do it.*

Tom Baxter, in *The Purple Rose of Cairo*

If we look at the world, we can see that all living creatures behave due to some motivations, whether conscious or not. Therefore, in order for characters to exhibit behaviors that appear reasonable (and believable), they must have their own set of motivations. These motivations can greatly help to stimulate the generation of a plot, though however, without any guidance for the plot, the results cannot always be of narrative relevance [15].

To ensure that the generated plot is interesting, the system needs to have some concept of drama, and apply it to the currently unfolding story. Understanding a story in its totality is a task that integrates the understanding of a character’s goals, plans, qualities, and emotions. As it will presented in this document, several projects have been successful in creating systems that focus on character behavior and interaction.

If we consider that a plot unfolds from the goals of the characters, their relationships to each other, and their individual traits, the user (or author) can indirectly influence the plot, through the definition of the characters, such as changing the initial state in order to generate a different story.

In some point, an author could also define additional goals or feelings that may be attached to a character at certain times critical to plot development, or in response to some external stimulus. The main goal is to influence an actor's behavior by giving him additional goals or by modifying his emotional state.

Using this approach, each character's emotional state and current goals drive the selection of a specific behavior from a defined set of possible behaviors. The intensity of the appropriate emotion values is then used to determine the intensity of the expression of the behavior.

Even when performing simple actions, a character's hidden emotional state may appear, in such a way that one character’s behavior can affect other character’s responses.

3.2.3. Emotional components

While creating a model of personality and relationships, it is obvious that a manageable set of emotion variables is needed. The magnitude of these variables will define how each character relates to the others.
Selecting the appropriate set of emotions is a surprisingly difficult task. Further, different sets of emotions are needed for different types of stories. In this context, a basic identified sets of emotions must exist, addressing the most appropriate conditions for a dramatic genre, for example: fear/pity/love for the tragic form, laughter/ridicule for comedy, and fear/hate/love for melodramas.

A strict ‘orthogonal’ approach, such as good/bad, strong/weak is difficult because real emotions overlap, and hence a real orthogonal emotions set can be a difficult task. Emotion values can tell us how one character feels or reacts about another. The asymmetry of interpersonal relationships (one character loves another, but it is not loved by him/her) is clearly a great source of narrative dynamics. From a storyteller's point of view, such aspects keep (or tend to keep) a story lively, giving rise to new goals all the time.

In addition to these two dimensional emotions (directed towards other characters), some other dimensional emotion variables can be created, to indicate a character's internal emotional state or mood (or personality attributes that remain constant or vary through time, or according to some actions).

The development of emotional, life-like agents, as a metaphor, aims to provide for highly personalized human-machine communication, such as in [3], where they present characters in interfaces to make the overall HCI more enjoyable.

### 3.2.3.1 Some considerations

With the previous ideas, we can suggest that, in order to produce interaction on more human terms, and in concordance with [11], a character-oriented system must have: beliefs

- A rich representation for emotions, knowledge, beliefs, etc. (An abstract narrative)
- A rich set of behaviors, driven by these representations (A collection of story pieces)
- A rich grammar for communication of knowledge, events, beliefs, and emotions. (A navigational strategy or reasoning)

At the center of any system dealing with this artificial personality issues, there must be an emulation of human emotions. Besides providing new motivation for any believable behavior, emotions give the characters a new domain for discourse. They may interact on the levels of physical state, information state, and emotional state.

While designing any interactive story system, we must keep in mind the interconnected dimensions of physical state, emotions, character beliefs, behavior, and communication capabilities. We must also keep sight of the main objective we pursue: the characters must display believable [8] and original behavior, while keeping an engaging, interesting and appealing dramatic interaction.
3.3. More on the narrative elements

While the previous subsection presented a brief overview of the characters and the characteristics they must include within a storytelling system, this part aims to provide an accompanying set of ideas and background on what we believe are two of the most important elements related to the narrative in a story: emotions and drama. A strong narrative must consider these two elements as the vital components needed to provide the correctly narrated rich, appealing, immersive story all researchers want to be able to produce.

3.3.1. Emotions and personality

I wouldn't go so far as to call a dog filthy but they're definitely dirty. But, a dog's got personality. Personality goes a long way. Jules, in Pulp Fiction

Characters that display emotions are critical to a rich and believable simulated environment, especially when those characters will interact with real people (which actually possess real emotions)[8]. Emotion is a very essential element that creates the difference between robotic behavior and life-like engaging behavior. Traditionally, in videogames, animators have carefully created these behaviors for pre-rendered intros and animations, called cutscenes, but this approach, however, is not possible (or good enough) when we wish to use autonomous, interactive characters that have their own distinctive individuality and mood.

Truly interactive characters must generate their behavior autonomously through further detailed and developed techniques.

As human beings, we have an innate understanding of what emotions are, and beyond the aspects of debating the how emotions are produced or why we have emotions, it is clear that characters in a virtual storytelling environment must provide a strong sense of emotional capabilities. Emotions are by all means an integral part of any decision-making system that appeals to be catching and to offer a correct sense of immersion.

Emotions adjust our decisions according to our personalities, moods, and momentary emotions to give us unique responses to situations presented by our environment. In some sense, and as Bryan Wilson proposes [69] “Personality has evolved as a problem-solving mechanism”. According to this idea, personality has evolved to provide tools to attack problems from different angles risky solutions vs. cautious and precise incremental ones; deep thought and reflection vs. the fact of gaining knowledge from others, etc.
Wilson also proposes that a system handling the emotions of artificial characters (AE, Artificial Emotion) needs to comprise three layers of behavior: momentary emotions, moods and personality.

The top level of emotions keeps the momentary feelings, and must consider behaviors displayed briefly in reaction to concrete events. As an example, we can think of laughing at a joke, or showing surprise when we see something unusual. Regarding moods, we can understand them as prolonged or long term emotional states caused by different issues, such as could be the cumulative effect of momentary emotions, the first layer. Personality, as the last layer, must hence include all the remaining and permanent issues, the emotions displayed when no mood or momentary feelings exist.

The possibility to include the three levels in a narrative system could greatly improve the overall drama of a story, by improving the characters as detonator of that goal.

The levels described keep an order of priority: momentary emotions have priority over mood when determining which behavior to display, and mood, in turn, has priority over the personality. This notion allows to use the layers to define, for example, a full set of possible reactions, since momentary emotions can assume the highest priority when selecting a behavior, but they are short and decay rapidly. Moods, for instance, can be used as produced by momentary emotions, for instance, by the cumulative affects of a series of momentary emotions, and they can be used as gradually increasing their prominence even after the momentary emotions have subdued, since their development can be understood as depending on whether the momentary emotions are positive or negative (punishment or reward). Here, if a character were to receive a large stream of negative momentary feelings, the mood would obviously change and would decay slowly. The personality layer, however, is always present and has a consistent level of prominence.

Figure 3: Mood, gestures (Shenmue™)
The use of notions like AE within a narrative system is straightforward, specially in those cases that are not dominated by a genre based in fixed emotions (war games, conquer games, etc.), such as can be the social environments or the storytelling systems that make use of social characters in the story. As Wilson claims, “Emotion primarily serves a social function in interactive entertainment. Emotional responses are used to make the characters that we encounter believable and engaging”[69]. Different personalities in characters help to provide scenes with immersive and believable social situations. “Believable agents” [8] are mentioned in other sections of this report.

AE can be used to produce gestures and actions as its output. Actions are a general category, and are dependent upon the context of the situation in which the character exists, selecting appropriate actions in relation to a character’s personality or mood. While an outgoing, extroverted and active character might perform an action enthusiastically, an extreme introvert would need controls in a different way to produce the same results, for example. Gestures, namely hand, body, and facial gestures, are a more complicated topic, but still are the way in which we can communicate our emotions.

Computer characters with no support for gestures, must make use of alternative means to show a tone, such as the cinematic approach we suggest in this report.

3.3.2. Drama

“This story's gonna grab people! It's about this guy, he's crazy about this girl, but he likes to wear dresses. Should he tell her? Should he not tell her? He's torn, Georgie! This is drama!" Edward D. Wood, Jr. in Ed Wood

“Game developers are beginning to realize that mindless, violent action, and fantastic special effects supported by ever advancing hardware will not hold the interest of core gamers forever”[46]. Littlejohn also claims that after that certain point where the increase in sales will not be due to the advances in resolution and sound, those “mindless games without good characters and narratives” will never be able to remain too attractive. The key issue stated is that we are facing a point where –finally- we have begun to reconsider the importance of story and character development.

From this perspective, the questions that emerge are the following[46]:

?? If story means linear, how do we make an effective interactive story?
?? what does effective mean in terms of interactive storytelling?

The goal is the development of compelling interactive storytelling, where the story keeps the attention of the audience, has emotions involved and, at the end, makes the audience feel the experience is worthwhile at the overall conclusion.
According to Littlejohn [46] or Paiva [55], *drama* is the answer: an art that has evolved to deal with issues related to being engaged with a story, to have a compelling, satisfying experience.

A brief description of drama and its characteristics, hence seems a natural part of this review, even though we are aware that the description cannot be complete since there are already many generalized attempts of description of drama, which cannot be easily presented, and which may be understood within the body of a narrative knowledge compiled over thousands of years by different cultures in the entire planet.

The main points of agreement are that drama deals with human conflict communicated, by means of speech and action, to an audience. Furthermore, depicting human conflict will command attention and interest and for that reason drama uses an innate human interest in conflict to engage an audience for the purpose of communicating a theme. The subject must be then something that we can all relate to.

In a dramatic presentation, the conflict is expressed by means of visible actions (something obvious in the design of some genres of computer games, for instance) but taking this notion beyond the basics, the goal is to make a project compelling, and then the reason for an action is far more important than the action itself. “The forces that cause the action are what excite the audience, making the action believable, and holding the audience in rapt attention”[46].

If we were aiming to find the reasons for action, we may find that in life and in drama, the study of the human being resolves itself into an evaluation of the motivation that provokes action: if all is in equilibrium, we prefer not to act, but when there is an imbalance of forces, and the motivation to restore the balance is strong enough to overcome the inertia, an action is taken. Considering this, and according to Littlejohn, the motivation to act lies in the wishes, needs, and desires. When any obstacle stands in the path of the resolution of these motivations, a conflict occurs. In its simplest form, all the drama comes down to a character (or a group of characters) we empathize with because they want something that we can all relate to desire, and to the hostile forces that oppose to the fulfillment of the original longing. Is the clash of these opposing forces what results in a dramatic action.

The human motivation can (questionably) be divided into four basic drives, following Littlejohn’s approach: desire for response, desire for recognition, desire for adventure, and desire for security. These are the motivating forces that control the actions of all humans. In a dramatic presentation the pattern of human conduct is developed within the framework of a particular structure or dramatic form, which, despite passing innovations, has persisted over thousands of years.

The study of this structure is the next step in understanding the principles of drama.
The whole dramatic structure can then be understood as the destruction and restoration of the balance of forces. Simply put “it is the process of getting into, and then back out of, trouble”. By examining any compelling story we can find that at the beginning an equilibrium exists, then the potentials of struggle may be present and even boiling under the surface, but the trigger has not been pulled. During the presentation or interaction the stability is destroyed, while at the termination the balance has been regained. It certainly may be a balance of forces wholly different from the one we had in the beginning, but the steadiness is present.

Figure 4: Balance recovered (final scene in Star Wars)

This balance-imbalance-balance structure is generally divided into different parts, including exposition, complication, climax, resolution and a final conclusion. This approach is generally understood in a three act structure, and is related to conventional narrative (film, theatre) and traditional drama structures, an issue which goes out of the scope of this report.

Stories, analyzed as sequences of causally-related events and reactions to those events by the characters, has been studied from different points of view, such as for browsing multimedia stories [1].

In all cases, a cinematic approach to show the narrative in a coherent manner is possible and a tool and mean to support and present dramatic content in a proved and descriptive manner.
4. Current methodologies and projects

Different institutions have an ongoing number of projects dealing with interactive storytelling. The following sections provide a general background and state-of-the-art of the elements related, and present what we believe are the most important projects that can be related to our ideas and goals on the subject.

4.1. Computer games and AI

The subsequent review is included as a basic component of a report dealing with interactive storytelling, and is related to computer games and AI. The main sources of information for this section were a technical survey written by M. Cavazza [14], and supplementary texts and overviews on the topic [37, 41, 63, 70].

It is known that computer games have traditionally implemented empirical solutions to many AI problems and are now turning to those more traditional AI algorithms, being AI a very important part in the overall gameplay [14]. The main techniques used in current computer games include finite-state transition networks, rule-based systems and search algorithms, and this relationship is not transitory or small: computer gaming and AI research have a lot to offer each other.

Although many commercially successful computer games have been rather visceral and violent, AI techniques offer the promise of creating engaging and dynamic interactive entertainment that has the addition of strong narrative components. In terms of classical AI research, being able to work in the context of computer games means challenges that are as complex (and convincing) as many real-world problem areas; gaming environments—besides the AI challenge—offer also a large base of users and graphical elements as added value.

The following subsections describe the implementation of AI in different commercial computer games, as well as some academic research in AI targeting computer games applications. The subsections have left aside the topics on neural networks, genetic algorithms, and related “A-life” techniques, but this is intentional, and the selection is influenced towards the symbolic AI approach, which reflects the current overall situation of game AI [14].

4.1.1. NPC behavior

If we leave to the side the games that rely heavily on physical simulation (such as the driving games or the flight simulators) most computer games require artificial opponents against which the player is competing. These artificial players should be credible as opponents and should also demonstrate an intelligent behavior during the course of the game. As Cavazza points, “the role of AI is often said to
be ‘loosing the game in a credible fashion’, for there would be no real fun if the system was unbeatable'[14]. In this section techniques for implementing the behavior of Non-Player Characters (NPC), essentially reviewing rule-based formalisms and Finite-State Transitions Networks (FSTN), are discussed.

### 4.1.1.1. Rule-based systems

Rule-based systems have been a very popular technique in AI. While these systems are normally associated with expert systems, the programming paradigm is used in games where declarations, modules and exploratory programming are required. Some generic cognitive architectures, such as Soar -used by Laird and his team [40]- have adopted a rule-based framework.

Rule-based systems in computer games have been used in two different paradigms. The first one is to implement decision-taking processes for intelligent NPC and the second is to simulate the behavior of a large set of actors, each one only using a few rules. Rule-based systems have been employed in many recent games, such as Civilization: Call to Power™, Age of Kings™ or Rainbow Six™.

As mentioned by Cavazza [14], rule-based systems comprise different elements:

- a rule formalism/syntax,
- a unification algorithm, and
- an inference engine.

In some formalisms, rules are triggered in a data-driven manner and the control mechanisms differ are not the same as those used in the inference engine. Behavior-oriented rule-based systems (such as those used in simulation) tend to adopt forward chaining where traditional decision support systems would use either backward or forward chaining.

Unification algorithms are known to pose difficult efficiency problems, due to the combinatorial nature of the pattern-matching problem. The trade-off for the use of rule-based systems appears to be between declarativity/modularity and the computational efficiency.

The Soar Quakebots developed by Laird [41] allow to illustrate the use of rule-based formalisms, being artificial opponents for the Quake II™ game.

The shooting games allow plenty of opportunity for the NPC to display intelligence in their tactics (even when the shooting characters seems not to be too attractive due to their limited action), and the tactics can be seen in the case of "death matches" between human players in Doom™, Quake™ or Unreal™.

The intelligence in these characters is mostly tactical, and involves planning strategies to ambush the opponents, collect weapons (and ammunition) without exposing oneself, etc. In some shooting games, the relative low intelligence of the opponents is compensated by their number, though this sometimes requires group
coordination (which is actually another topic on which lack of proper AI can reflect poorly on the game [14]). Laird’s Quakebots aim to achieve “human-level AI” in this specific context.

The Soar Quakebots essentially implement real-time expert systems with extensive knowledge of the game encoded in several hundreds of rules, and they rely on an internal model of the world on which they apply tactical rules to perform the main game actions (collect weapons and health bonuses, fight against other players). The tactics are dedicated to all the main phases of the game (collecting power-ups, attacking, retreating, chasing the enemy, ambushing, etc.), and though the system is mainly based on hierarchical goals, it is essentially reactive and does not incorporate any planning.

Soar, as an engine, is constantly making and executing decisions (see figure, from [14]), by proposing and evaluating operators (there are about a hundred in a Quakebot) that correspond to either primitive game actions (shooting, turning) or more abstract goals to be achieved (collecting an item). The architecture continuously proposes and applies operators by using rules that match the current state of the game.

As an example, the following rule proposes a top-level operator, in the basic mode for the Quakebot.

IF enemy is visible AND my health is < very-low-health-value(20%) OR his weapon is much better than mine
THEN propose to retreat

The system has been tested [40] looking for skill levels that use metrics similar to those applicable to human players, and the important aspect is that one of the main goals is to maximize "humanness", rather than try to implement unbeatable opponents, for instance by tuning the decision time to be similar to that of human players.
4.1.1.2. Finite-state machines

Finite-State Machines are a dominant technique in computer games, and this is due to several factors; as mentioned by Cavazza [14]:

- FSTN can be implemented efficiently.
- The size of FSTN in computer games is moderate, so the description and maintenance does not pose large problems.
- The formalism is used in animation and hence is familiar to the computer graphics community.

FSTN (and related formalisms) have several advantages if we think about the behaviors in computer games, since they can be used as an exchange format, and as a representation to discuss behaviors between designers and programmers.

Many games have implemented FSTN for their NPC behaviors: for instance, the overall behavior for the original Quakebots (in the "standard" Quake™) is described by a single 5-state FSTN based on possible states \{Idle, Search, Attack, Hurt, Dead\} [14].

The Team Buddies™ game is another example that uses an extension of FSTN techniques to implement more sophisticated NPC behavior [68].

In the Team Buddy, teams of agents compete against each other. One main task to be performed by the agents is to collect crates that appear at selected locations and deliver these to stacking pads. Depending on the size of the stacks assembled, crates generate new weapons, vehicles or even team members. Agents fight as a team using various weapons. The fighting involves destroying other player's stacks as well as directly eliminating other agents. The player is in control of one of the teams, which comprises up to four agents. He does so by directly controlling one agent while the others remain autonomous though responsive to commands.

NPC behavior is controlled by an Augmented Transition Network (ATN), in which state transitions are driven by percepts from the embodied agents. Percepts are calculations made within the game world from the agent's perspective. An example percept is EnemyNear, which checks the distance between the agent and the nearest enemy agent.

The actions to be carried by the NPC in the game are associated with these state transitions. There are approximately 160 axiomatic actions that can be assembled into some 1000 composite actions. There are four kinds of axiomatic actions: effectors, register operators, communicators and "state changers". Effectors physically control the agents: they include agent movement, attacking, and handling crates. Communicators send messages between agents to ensure group co-ordination. Both register operators and "state changers" can bypass the normal operating mode, to trigger immediate action or passing orders between agents.
Finally, the use of ATN was compatible with constraints on CPU resources, as each agent in the game requires approximately 1.4% of the processing time [68]. This issue is relevant, since historically, computer games have allocated less than 10 percent of their processing cycles to AI routines, though the current AI techniques have been making their way into commercial systems while the low-cost, high-end graphics cards (and increased CPU processing power) free system resources [63].

4.1.2. Path planning and search algorithms

The majority of the NPC in games progress in some sort of spatial layout (rooms, mazes, platforms, terrains, maps, etc.), and hence path planning is an ever-present problem in computer games.

Planning a creative path, avoiding obstacles for instance, is relevant not only in real-time strategy games but also in the NPC of computer games such as Quake™. In this context, path planning has become a test case for the use of AI in computer games, and it has also been one of the first documented examples of the use of traditional AI algorithms (such as A* [60]) in computer games.

Path planning is a well-described problem in areas such as robotics, where robots have –for instance- to reach a goal location by avoiding obstacles. In computer games, a NPC can have access to the whole graphic database (and therefore not being limited in its perception by the range of its sensors), and as a consequence, it is possible for a NPC to find its way in a specific labyrinth using A*. “This ‘omniscient’ approach somehow compensates the relatively poor intelligence of NPCs (largely due to implementation constraints)”[14], as Cavazza says, but, however, it does not drastically affect the overall believability of artificial actors.

The A* algorithm is a heuristic search algorithm that finds the best possible path from a starting point or node to a pre-defined goal node, in either an explicit or implicit graph. In the case of path planning, the ‘best path’ is defined as the shortest one, hence heuristics are distance-based (Euclidean distance or ‘Manhattan’ distance).

If we focus on computer games, path planning generally encompasses a discretization step, which associates to the environment a fixed data structure such as a uniform grid. For instance, by discretizing a game level through a grid it can creates a tree of branching factor 8 (since cell has 8 neighbors and the path can go through any neighboring cell) that can be directly searched for a path. As of now, different techniques for discretization have been described and the discretization step itself is itself a target for optimization, such as by creating variable-size grids [23].
A* is based on a heuristic function that assists on the selection of the best candidate node in order to calculate the shortest path. The formal properties of A* guarantee that, if an optimal solution exists, this will be provided by the algorithm. This formal property is called admissibility and is guaranteed for A*, provided the heuristic function is always a lower bound of the actual function. This is the reason why the most often used functions are straight-line Euclidean distance or ‘Manhattan’ distance to the goal destination. The figure (from [14]) shows the use of A* to plan a path for an agent in a graphic 3D environment.

A* has been widely used in recent computer games; for instance Start Trek: Armada™ [23] used it with an optimization variant (varying the size of grid cells during the discretization process). The game Half-life™ appears to use a similar algorithm as well [14].

There are, nevertheless, some limitations to the direct use of A* in the context of computer games. The first one is that, in many cases, the real speed of computation is far more important than actually finding a good path. Trading admissibility for speed of computation is actually a well-known problem but takes a particular significance in the context of computer games [14]. The other is that A* normally operates in a static environment, and hence has limitations when reaching for moving targets, where variants must be used. In general, dynamic environments need variants of A*, such as D*, that enables the ability of re-planning [64].

If the overall performance of A* resides in the discretization algorithm and the proper choice of its data structures in the implementation, the decision to use this algorithm should basically be based on the complexity of the environment (as measured, for instance, by some obstacle density). As a rule of thumb, the benefit of using A* is only clear for suitably complex environments and, when the obstacle density is low, simpler algorithms with local obstacle avoidance can be implemented more efficiently than a search-based path planning [14].
4.1.3. Storytelling in computer games

Computer Games differentiate themselves according to their storyline. Some do not have any storyline at all, some others even get to consider it damaging to gameplay in certain game genres, while other games, such as Myst™, try to develop a real and proper plot.

Some games are based on explicit plot representations that unfold as a function of the player's performance in some levels while, in other games, the story is emerging as a result of one or many user actions. The Sims™ is one of the best examples of this occurrence, even though the narrative aspects are mostly episodic, as the game is intended to mainly be a simulation.

The Sims™ offers the possibility for the user to create characters with specific personalities and observe their artificial life within their neighborhood. The game’s emergent narrative can then make sense at several levels, from the episodic incident of one character fighting against another, to the overall evolution of the family.

The Sims™ game does not use a specific narrative drive, however, and it is based only on a set of fixed mechanisms that define the relationships between the actors, and between the actors and the objects in their environment [14]. The Sims, the characters, are connected to a set of motives ("hunger", "social", "hygiene") that constrain their behavior, and each motive has some rules, called advertisements, that define how interacting with an object satisfies a corresponding motive. For instance, the presence of another Sim can satisfy the "social" motive.

While a Sim considers its following actions, it analyses the existing objects in the world and checks each advertisement, and computes a score for each interaction. During the process, advertisements are weighted to consider priorities in motives, and, in addition, the score can be influenced by a spatial component or by a personality component.

The spatial component determines whether a Sim will consider closest objects first, regardless of the magnitude of the associated advertisements. Personality variables explain for instance that a book triggers an augmentation in the entertainment motive for an ‘intellectual’ Sim.

This condition is shown in the figures (from [14]), where attraction is represented as altitude, and how it changes after an interaction is complete. In the first frame shown, a character’s loneliness is dominant, and then in the next figure, after interacting with the friend, the loneliness is gone and he is mostly bored.
As Cavazza points [14], “Though not claiming itself to be based on any specific AI theory or technique, its principles can be related to some traditional AI paradigms”. Here, the overall emergence of behavior from the interaction of a set of agents (and the setting of initial conditions) evokes distributed (symbolic) AI techniques.

**4.1.4. Learning and adaptive behavior**

One of the main features of intelligence is adaptive behavior. Intelligent agents are those who can learn from their errors or recognise the individual style of their opponent and therefore adapt to it. “It is also a natural way of accompanying the player’s learning curve than having NPC characters learn themselves, keeping up the interest in the game” [14].
While this feature is not often present in computer games, there have been recent developments (both in industry and in academia) on this specific topic. The following paragraphs mention learning of tactical knowledge, which is more sophisticated than simple ‘reactive tuning’ and, unlike full-scale planning, remains tractable within state-of-the-art technology.

The fighting games are an interesting test case for adaptive behavior. Trusoft™, a game company, has developed a technology, the artificial contender, which makes possible for NPC to learn tactical moves while facing a human player.

![Figure 10: Processing in Trusoft™’s Artificial Contender™.](image)

This approach is adapted to all games in which there is a two-player competition and offers very interesting perspectives for modelling opponent behavior that can be formalised as depth-bound plans. It has been extensively demonstrated on two-player fighting games (for example, Sega’s Virtua Fighter 2™). It is based on the representation of player moves and game states in the form of multi-layered semantic graphs on which pattern-matching operations are carried.

The system uses a planning-like formalism, in which actions are represented together with the state transitions they trigger. This formalism is used for game states and action recognition as well as for knowledge representation within the multi-layered graphs.

The systems processes information through a pipeline [14] (shown in the figure):

1. The first step consists in recognising the game state and formalising it.
2. Representations are produced at different levels of description, which speeds up pattern-matching in the final representation and makes possible for the system to reason at various levels of description.
The action and the state are subsequently stored in the knowledge structure and compared with the incrementally-built knowledge graph.

Finally, based on this comparison, the best action for the Artificial Contender™ is selected.

The overall process also depends on some personality attributes for a character that influence the overall learning process at various levels, from state recognition to situation assessment and decision taking.

The size of the semantic networks is compatible with their dynamic storage in memory. For example, the network obtained after playing 10 rounds in a fighting game is approximately 60 kB, while the largest network (called "saturation network") is 500 kB. The system also optimizes its use of CPU resources by processing some of the data in the knowledge graph off-line, for instance between the various rounds of a fighting game [14].

This approach to machine learning, which relies on an explicit symbolic representation also makes possible the generation of user explanations and should easily interface with other symbolic techniques, such as rule-based systems.

Another example of adaptive and learning behavior is represented by the Soar Quakebots introduced in a previous section. The Soar architecture incorporates its own learning procedure, called ‘chunking’. This mechanism was initially developed to ‘compile’ complex problem solving into simple rules (and then speeding up reasoning by bypassing complex inference steps). This form of learning has been incorporated into an anticipating Quakebot.

As mentioned earlier, the objective was to reproduce the behavior of human expert players, who apply tactical schemes that anticipate their opponent’s location and trajectories and ambush them accordingly. The implementation of the anticipating Quakebot and the results obtained is detailed in [40].

A sample rule learned by the Quakebot is:

```
IF  predict-enemy is the current operator
    AND
    there is an enemy with health 100, using the blaster, in
    room #11
    AND
    I am distance 2 from room #3
THEN
    Predict that the enemy will go to room #3 through door #7
```

It can be noted that the rules learned are highly specific: this is because the predictions are only valid in very specific contexts, due for instance to the spatial configuration of the rooms in which action takes place.
4.1.5. Natural language interfaces

Being able to control computer games through speech and Natural Language Processing (NLP) is becoming a large goal for the next generation of computer games. A natural language interface would support more complex user input (for instance, like that required by strategy and simulation games).

Besides that, it would greatly enhance the believability of any game featuring characters in third-person mode (adventure games, for example) and it could greatly improve gameplay by supporting interactive narratives that would open the way for new game genres [14].

While language technologies have progressed considerably over the past decade, there are still no universally accepted formalisms or techniques. Cavazza has already developed a prototype of NLC using a emulator for the Doom™ computer game [20], that was developed mainly to experiment the overall integration and the specific NLP problems faced when designing language-enabled agents in computer games. The system is based on an integrated parser that can accept input from keyboard or a speech recognition system.

In NLP, parsing consists in using linguistic knowledge (syntactic and semantic) to extract semantic information from the linguistic form. The parsing methodology depends strongly on the speech recognition paradigm employed. For instance, with multi-keyword spotting, simple template-based of finite-state systems can derive appropriate game commands from the keyword sequence. However, there are cases where more complex commands are required, for instance those involving actions parameters and spatial expressions (for instance, from a DOOM™ spoiler [14], shoot the toxic waste barrel next to the window).

In these cases more accurate parsing is required to correctly attribute the various relations between actions, objects, instruments and locations. The first step taken by Cavazza was to take a number of on-line DOOM™ spoilers, under the assumption that similar sentences could be used for giving instructions as well. This made possible to determine the most relevant syntactic constructions (for example, for action parameters and spatial expressions) and to design the corresponding sublanguage grammar.

The parser, based on a simplified variant of Tree-Adjoining Grammars produces semantic structures by aggregating semantic content in parallel with syntactic tree derivation. These structures are further translated into unambiguous command messages that can be interpreted by a game engine [20]. As Cavazza remarks [14], “here are strong dependencies, yet to be explored, between gameplay and the use of NLP”.

First of all, natural language input is only compatible with games whose timescales match the response time of spoken input and processing. Secondly, NL input assumes game characters with a certain level of intelligence and autonomy.
As spoken commands tend to be more abstract, game characters need to generate a fine-grained set of actions from them. On the contrary, more intelligent artificial actors that rely on explicit representations of goals and internal states would be best controlled through natural language instructions.

Another important aspect is the use of Natural Language Generation (NLG). There are currently several restrictions that prevent its use, though. The first one is that natural language descriptions assume scene interpretation, which can be quite complex. The other one is that, ideally, NLG is best used with Text-To-Speech (TTS) systems to provide in-game feedback to the user. However, good quality TTS systems still require more RAM than would be available in current game consoles [14].

### 4.1.6. AI languages in game development

When addressing the AI techniques, it is difficult to avoid discussing the choice of programming languages in the game development. In the specific context of video games this would not appear relevant in the first instance, as low-level but run-time efficient languages are the rule [14].

However, the choice of a programming language does not only concern run-time performance, but also development time and ease of debugging, particularly if large knowledge structures tend to be involved. Classic algorithms such as A* can be straightforwardly implemented in procedural languages like C.
Nonetheless, with the increasing CPU power available and the embedding of high-performance graphic algorithms in hardware there might be some space left for more traditional AI languages.

Some successful commercial games, such as Crash Bandicoot™ or Super Mario™ 64, have in fact reported the use of Lisp to describe the behavior of their characters. This remains somehow anecdotic so far [14].

However, the foundation for this is that modern Lisp compilers are highly efficient, while at the same time CLOS (Common Lisp Object System) supports the most sophisticated object-oriented features currently available. Also, many computer games tend to develop their own internal scripting language on top of procedural languages to serve as a programming tool specifying behaviors. This issue is further discussed in the following sections of this document. The use of Lisp or other high-level languages could probably provide a more consistent alternative, but in terms of research projects, the scripting language serves as a programming tool with good results.

It should be noted that, while AI techniques with a significant algorithmic component are in practice often implemented in procedural languages (search, inference engines, constraint propagation), this is not the case for extremely complex techniques that require sophisticated knowledge representations (like case-based reasoning). The number of variables and the specific game need to be considered for any decision on this.

### 4.2. Game engines and research

While it may be natural to consider the use of game engines to create computer games, its operation can be extended to include research areas and topics. Here, we provide some concepts related to the use of game engines in a research project and environment. A full set of notes and document related to this very specific issue can be found in [41, 45].

#### 4.2.1. Game engines

Some years ago CG required expensive hardware and terminals, but now, the most sophisticated, responsive interactive simulations can be found in the engines built to power computer games. Several reasons exist for the subject, but a very important one is related to money. The cost of developing has grown high, and game developers can no longer rely on recovering their investment from a single game, what has led to game engines: modular code written for a specific game, but general enough to be used for a family of similar games.

Here, the separation of function from content allows game code to be repurposed, even in projects related to scientific research [45], such as the ones that are included in this report.
While the very early computer games consisted of little more than one event-loop, some state tables, and some 2D graphic routines, ID software™ marked a new era of game design and play in 1993 with the release of Doom™. Although it was not the first game to offer an immersive first-person perspective, Doom™ was the first to do it successfully.

While Doom™ provided strong realism (for the time being) by making use of texture maps, 2½D and 4DF animation, and also provided multiplayer play over a network and some degree of user/third-party programmability, as en engine it had strong limitations: peer-to-peer and limited programmability were not a true option for most games [45].

The Quake™ game, by the same company, learned the lessons, and it was truly a 3D with environment and actors made from texture-mapped polygons. Besides, a client-server architecture (as opposite to peer-to-peer) provided benefits: keeping the server handling game state and the client handling the 3D engine. Quake™ also extended its programmability to provide the first game-independent game engine, and provided both a level editor, for changing layouts, and QuakeC, a byte-compiled scripting language for changing behaviors.

The following version, Quake II™, added support for hardware-accelerated graphics (which were just beginning to appear on the PCs).

With the release of Quake III Arena™ and Unreal Tournament™ (by Epic Games™), in 1999, the game engines reached their present (and mature) state. Both games are strictly multiplayer with single-user play simulated through play against a synthetic opponent, and offer an extensive support for hardware
acceleration, providing also the options for user modifications through level editors and the use of scripting languages.

Figure 13: Quake III Arena™

Quake III Arena™ made its C game source code available, keeping only the graphics engine code proprietary. Unreal Tournament™ is quite similar, but its open-source game code is written in UnrealScript™, a bytecode-compiled scripting language (similar to Java).

A very important issue related to these elements is that, although there are more than 600 commercial game engines, Quake III Arena™ and Unreal Tournament™ are acknowledged to provide the most flexible, developed and usable engines for research purposes [45].

Nowadays, in the modular structure of constructed games, the game’s engine refers to the collection of modules of simulation code that do not directly specify the game’s behavior (the game logic) or the game’s environment (the level data), and hence the engine includes modules that handle input, output (3D rendering, 2D drawing, sound) and generic physics/dynamics for game worlds described in the game or games [45]. The next figure shows the overall structure of a game engine.

Game engines, as a software tool, provide tools fully applicable to scientific research. If a research project requires to keep control of the research, needing support with a faithful maintenance of players, objects, and terrain locations, then the fidelity that can be obtained with a game engine is completely adequate.
While it is true that a precise multi-modal VE simulation (for instance, the kind of simulations required in accurate environment such as those needed in remote surgery) would require custom solutions, for most of the VE applications a game engine can add value by performing 3D control, providing all the networking and synchronization, and driving the high-fidelity hardware-accelerated graphics found in most current systems.

4.2.2. UnrealEd™ and Unrealscript™

Among researchers, the choice between the Quake™ and Unreal™ engines is quite balanced, and there seems not to be a choice better than the other since both are more than adequate for their intended purpose. Looking at the current projects, we can find an even division, with projects like Kaminka’s [38] or Jacobson’s [36] being done with Unreal™ while Bylund and Espinoza [12] use Quake™. People like Laird and colleagues avoids the decision by working with both [41].

Figure 14: Game Engine Structure

Figure 15: UnrealEd™ interface
We have chosen to use the Unreal Engine, the first we used in our approach to game engines, the system used by Cavazza and his team [13, 19, 21]. This section presents, hence, a brief introduction of its capabilities, by providing an introduction to programming for Unreal™ using Unrealscript™ and the basic class structure that Unreal™ uses, from the perspective of a beginning stage in the creation and use of Unreal™ modifications.

UnrealEd™ is the level editor of Unreal™, and it can be understood as a 3D level editor, with capabilities for the 3D components (graphics, characters, levels, textures), the game’s design (images, sounds, lights) and the overall modules and extensions that can be used to create the modifications for the projects. The editor provides not only a graphical interface that is similar to that of 3D modeling and design software, but several dialogs that allow to change the scripts and programs of the engine so that the modifications can be developed.

Unrealscript is a language developed for the creation of game content for Unreal™. It is based on Java and C++, and its syntax has the same style common to both of them, sharing many keywords found in Java. Unrealscript™ is an Object-Oriented-Programming language, keeping the notions of classes and inheritance, but it does not have all the features of a more complete OOP language such as Java (for example, it does not support multiple-inheritance). Nevertheless, the missing features are not necessary to make a fully-working modification of the engine.

A characteristic of Unrealscript™ that cannot be found in either Java or C++ is states. States are a feature that allows you to define different implementations of functions within the same class (in a similar manner as function overloading in OOP), and they are generally used for time-based programming, and in those cases where an action must be accomplished differently while executing a specific action.

An example can be this: a jump is a standard jump, while a jump while running must be transformed into a flip.
The level editor, UnrealEd™, has a built-in class browser, editor, and compiler for Unrealscript. The browser allows to use trees with collapsible/expandable branches, showing child classes, and the possibility of editing the class code, creating subclasses, etc. The code can then be compiled and saved, as in any standard development environment.

![Figure 17: Objects editing (properties and trees)](image)

### 4.2.2.1. Classes and class structure

The class structure allows to modify and make use of all the capabilities of the engine. The following information was obtained through experimentation and from different materials and tutorials available online.

**Object and Actor**

Everything that is coded in Unrealscript derives from a very abstract and basic **Object** class, from which we get many internal classes that deal with the engine (mainly sound, graphics, etc) and —more importantly— the **Actor** class, which encompasses most of what we will want to modify for partial modifications.

The **Actor** class defines the majority of standard variables and functions that are necessary for all classes, and all game-content related material is contained in classes derived from **Actor**. Players, weapons, artificial-intelligence, and even the game itself is derived from the base class of **Actor**. Important functions to note in **Actor** are **BeginPlay()**, **Spawn()**, **Touch()**, **Tick()** and **Destroy()**.

The **BeginPlay()** function is called whenever an instance of the class is put into play, into the game. At the point of creation, it allows us to initialize variables and do other actions at the point of creation. The **Spawn()** function is used to create an instance of a class and put it into play. **Touch()** is called whenever an actor is touched by another actor, and is useful for causing changes (altering variables) in the other actor. The **Tick()** function is called every game tick, in a context of time, generally as much as possible and hence is used to perform continuous actions (or to check for certain conditions).
Pawns

The Pawn class contains the ScriptedPawn (AI), FlockPawn (Insect/Flock AI), PlayerPawn (Players), and more recently the Bot classes, which are the "living entities" in a defined level. The Pawn class defines basic variables, movement and other miscellaneous functions which are common to all Pawns, such as its Health, Groundspeed/Airspeed, and JumpZ, among other useful variables found.

PlayerPawn includes all players (also known as clients) that are available in a specific game, and it has several child classes that represent the various players (UnrealPlayer, Human/Skaarj, male/female players, etc.). PlayerPawn also contains all of the functionality of the players (movement, input handling) while the rest of the children classes handle all of the animations and sounds specific to the different players. The PlayerInput() and PlayerCalcView() are two functions that can be altered for modifications; PlayerInput() handles all input from the player, calls pertinent functions, and modifies data, and PlayerCalcView() handles the calculations of the view of the player, and can be modified to create a third person view, for instance.

The ScriptedPawn is the class that relates to most of the AI-related game content, such as the creature’s behaviors in the game.

Info

The Info class is a generalized class that encompasses all the classes that deal with different game-related information, such as the current rules (Deathmatch VS SinglePlayer VS TeamGame), ZoneInfo (used to create things like lava, water, or just plain empty space) and PlayerReplicationInfo which contains special networking information for players.

Triggers

The Trigger class handles the operation of lifts, doors, and other events by responding to specific actions in the game, and then by performing another action, either by modifying an actor or by setting off other triggers (or both). Triggers are invisible in the game and are generally triggered by touch, proximity, or by another trigger. It is also to call the Trigger() function from another class to set off a trigger, for instance to open several doors at once.

Subclassing allows to create special case triggers, or other event-driven gameplay situations.

Effects

Effects covers all the special effects in Unreal™, like can be the explosions, laser blasts, etc. Subclassing allows to create some new special effects, and combining existing ones.
Inventory

The Inventory class, a child class of Actor, contains both the Pickup classes and Weapon classes. Pickup is all the items, other than weapons, that a Pawn will pick up and use in a game (in the original game, this is health, armor, ammo) but can be used to create unique items. The Weapon class handles all the functions for firing and acquiring weapons in a game. All of the individual weapons are defined as subclasses of the Weapon class, and handle the specific issues of the weapons, such as the animations and the creation of the original projectiles/weapon effects.

Other classes

There are obviously many classes that cannot be included in this concise overview, and that are used to create the remaining things you see in Unreal, such as the HUD (menu display system), lights, moving brushes, menus, etc.

Vectors and Angles

Unreal uses the concept of vectors and angles to handle all of the movement and rotation of the actors in the 3D environment of the defined world. A vector is simply a structure consisting of 3 components (X, Y, and Z values) where the X axis is forward/back, Y is left/right, and Z is up/down. The values can be modified to directly alter an actor’s position in the world, using functions such as setLocation() function with a new vector for the new location.

Rotation is handled by an Angle structure which also consists of 3 components, Pitch, Yaw, and Roll, where Pitch will control the up/down rotation, Yaw the left/right, and Roll will control roll (left/right tilting). Using SetRotation() we can alter the direction an actor is facing, or convert a rotation to a vector to alter the direction of an actor’s velocity, etc.

4.3. Teesside University research

Once the previous sections have provided a minimal background in the area covered, we present the work done at the University of Teesside by Mark Cavazza and his team, since we consider it not only one of the most valuable sources of information regarding the subject, but one of the most committed and devoted efforts to fully develop and propose ideas in interactive storytelling in a manner we find appealing and challenging.

The work done by Cavazza’s group can be thought of as the most active and leading collection of research dealing with storytelling in a sense we consider close to our goals and close to the essence of the research field, and considering they have already described several paradigms of interactive storytelling [15-19], comprising various dimensions such as user involvement or the relationship between the character and the plot.
This group’s approach follows Michael Young’s ideas [71] in the sense of having autonomous actors whose roles are implemented using real-time planning mechanisms and systems, with the intention to dynamically interact to generate a story and a storyline.

Among all the possibilities, Cavazza targets a specific kind of application: letting users interfere, at any time, with a predefined storyline and its progression [18]. In addition, rather than giving instructions, users can alter the environment by, for instance, stealing an object, or influence other characters by offering advice. The consequences of this intervention then affect the characters’ behavior and alter the course of action, creating new dramatic situations and eventually leading to different story endings.

### 4.3.1. The storytelling system

The system developed uses the Unreal Tournament™ game engine as a development environment. The interactive story appears as a real-time 3D interactive animation with subtitles that correspond to a characters’ dialogue (or important events), and in which users can physically interact with the characters and navigate through their environment using normal game controls (in the latest versions, they can also verbally interact with them using a speech recognition system).

![Figure 18: The storytelling system](image)

The test scenario used so far was inspired by the popular US television sitcom Friends, due to the fact that the sitcom genre offers some specific conditions, such as having the story ending and the intermediate situations as equally relevant, an issue that provides an appropriate test bed for the story generation.
Additionally, when developing the system, Cavazza’s team defined various roles for each character and formalized these roles as plans. When the system executes a plan, it generates character behavior at runtime.

By decomposing a plan into different sub-goals, an action’s different stages are reflected, while the lower layers of the plan decomposition correspond to various ways to achieve these goals. For example, if the character ‘Ross’ wants to ask ‘Rachel’ out, he then first must acquire information about her, gain her friendship, find a way to talk to her in private, and that kind of aspects. In this situation, he deals with several possibilities at each stage of the story (for example, to obtain information about the girl he could steal her diary, talk to one of her friends, or try something different). The different possibilities correspond to sub-goals in the description of the character’s plan, which can be further refined in the main plan representation until they can be described in terms of some terminal actions (which are the elementary actions carried out by the characters). The system then plays the actions in the virtual environment using standard Unreal™ animation sequences or additional animations that have been imported into the system [16].

A very important distinctiveness of this character-based approach is how it uses the same basic mechanisms to support both the story variability and the interaction. Plan-based roles for the various characters are dynamically combined to generate multiple variants of an initial storyline [17]. In the absence of any user intervention, the mechanism will produce a variety of plot instantiations. At the same time, user interaction can interfere with the characters’ plans (for example, causing an action’s failure [18]) and then activate a re-planning that causes the plot to vary.

In their prototype, the graphic environment was modeled using the game’s level editor, and they incorporated additional objects modeled using 3d Studio Max, and textures from online resources. The characters were imported from online repositories, and the AI layer was implemented in C++ and was integrated in Unreal as a set of DLLs. UnrealScript™, the game’s scripting language, defines all the functions that interface with Unreal™ events (the functions that deal with object interaction). The latest version also integrates communications into Unreal™ by using a speech-recognition system [19].

4.3.2. AI and planning

As acknowledged by the research group, a broad collection of AI techniques has been proposed to support interactive storytelling systems, including planning techniques and techniques for narrative-related issues [19].

Frequently, the technique used actually depends on the interactive storytelling paradigm that is being implemented but, nevertheless, there is not always a direct relation between a given AI technique and a storytelling paradigm. For example, Young [71] has used planning to control the narrative, rather than controlling the behavior of individual autonomous characters, while Cavazza has been mainly interested in the emergence of story variants from the interaction of autonomous
actors [18], this is, the emphasis has been put on the actors’ behavior and not on the explicit plot representation or the narrative control.

This character-based system [19] provides a unified principle for story generation and interactivity. As such, the system allows anytime interaction (opposite to plot-based systems, that tend to restrict the user intervention to selected key points in the plot representation).

Since the project still needed some planning formalism able to contain the authoring aspects of the baseline narrative, the knowledge-representation requirements led them to use planning techniques to be used in a complex environment, and they opted for Hierarchical Task Networks (HTN) planning [21]. The selection of HTN planning was due to the fact that it is an appropriate method for knowledge-rich domains, that can provide all the knowledge needed to support the planning process.

Regarding characters’ roles, it also was shown that these roles could be represented with the HTN (they serve as a basis for the narrative descriptions), since the main characters’ goals are decomposed into alternative actions.

### 4.3.2. HTNs and planning

Since a single HTN corresponds to several possible decompositions for a main task, HTNs can be thought of as an implicit representation for the set of possible solutions. In the project’s context, each ordered decomposition constitutes the basis for a character’s plan, and each HTN that is associated to a character contains the set of all possible roles the actor can have across the different story instantiations.

Even though the set of all possible roles is sufficient, the set of story instantiations is larger, because the story is composed of situations that are the product of the actors’ different roles. This also provides a style for authoring the story variants, because that goal node in the network can have several ways of solving a narrative goal. As an example, if Ross wants to talk to Rachel in private, he can take her from her friends by calling her, attract her attention, asking her friends to leave, etc. This option allows them to quite easy be able to improve or refine potential variants by adding extra options at authoring time [17].

As representations, HTNs can include essential properties of a character’s role, through the actions the agent takes toward its goals and the choices it faces [15].

Since the goals in the project are related to characters, there is a further need to categorize these actions according to a narrative criteria. The categories should represent properties having a certain relevance for inter-character relationships, which can be then matched to the different personalities of the characters [21].
For instance, the actions that target other characters can be classified, in terms of behavior, as “friendly,” “rude” and so on. If, for example, Ross interrupts Rachel’s conversation when he’s accomplishing the task of talking to her in private, and sends her friends away, the system tags the corresponding option in the HTN as “rude.” In a similar approach, the occupations of the characters can be categorized as well, according, for instance, to some degree of sociability, for example as “lonely” or “sociable” [19].

To certain degree, the categories are part of an ontology of inter-character relationships, and can help to determine how the other characters will react to the actions taken. These inter-character relationships, even though are obviously important in the Friends™ sitcom context, are a generic problem in overall interactive storytelling.

In the system, the contents of the HTN are determined by considering an actors’ role in the baseline story. The roles can then be refined by providing additional options (this process is naturally supported by the HTN formalism), and the search mechanisms associated with HTN planning also makes them a useful tool for debugging.

Since HTNs are searched from the root node, which is also the main goal, it is easier to gain access to the corresponding state of the world. One additional reason for the selection of HTNs as a formalism is the fact that their graphic nature seems naturally helpful in the authoring phase of the story.

Figure 19: Plan-based representation of an agent’s behavior
The previous figure shows a typical HTN for a character. Pre- and post-conditions for the various tasks (that are not explicitly represented in the graphic) are associated with each task node. The preconditions for the lowest-level operators are constituted by the combination of executability conditions for their associated terminal actions (which are actually ‘acted’ in the 3D environment). Some of the conditions are subject to change in a dynamic environment, and then they become a main channel for interaction. The system directly implements post-conditions through the effects of terminal actions, which are rolled back to the highest-level task node subsuming these actions [17].

The proposed scheme allows to relate HTNs to other forms of knowledge representation in storytelling. Specifically, there is a formal equivalence between subtasks of the HTN and narrative functions that stand for key narrative actions, seen from a given character’s perspective. The difference lies in the fact that the agentive (or predicative) structure for the equivalent narrative functions is situated out of the corresponding portion of the HTN, in the interaction with narrative objects and other characters filling up the roles for that narrative function. For instance, when seeking information about Rachel, Ross could talk to her friend Phoebe. If he talks to Phoebe, she will complement the agentive role of the corresponding narrative function. Also, every time that multiple characters interact, they potentially instantiate narrative functions “bottom-up” through the conjunction of activities from their own HTNs.

### 4.3.3. Planning and execution

Interactive storytelling requires to alternate planning and execution [72], and Cavazza and his group have developed a search algorithm that produces a plan from the HTN, by searching the HTN depth-first and left-to-right and executing any primitive action it encounters in the process (they use a total ordering assumption and subtask independence in the process). The algorithm allows backtracking when primitive actions fail (due, for example, to user intervention or because of competition for action resources by other agents) and it also attaches heuristic values to the different subtasks, so forward search can use them to select a sub-task decomposition.

An essential aspect of HTN planning is that it is based on forward search while being goal-directed at the same time, because the top-level task is the main goal. (Other recent forward-search planning systems, such as the Heuristic Search [10] search forward from the initial state to the goal.)

As a result, since the system is planning forward from the initial state expanding the subtasks left-to-right, the current state of the world is always known (in this case, this means the current state achieved by the plot). The initial description of the roles is done by the total ordering of the subtasks. The total-order HTN planning prevents the possibility of interleaving subtasks from different primitive tasks, and then it eliminates task interaction significantly.
In the case of storytelling, the subtasks are mainly independent because they represent the different story’s stages. The decomposition of the problem space derives from the natural breakdown of the story into different stages or scenes—a classical representation of stories.

The current use of the HTN associated with important simplifications of the associated planning problems (such as sub-goal independence, empty delete lists, and total ordering of subtasks at AND nodes), and according to Cavazza and their results, this approach to planning seems consistent with the knowledge-intensive nature of interactive storytelling [15] and some of its inherent properties, such as the temporal ordering of various scenes.

Other planning techniques – more oriented towards a problem-solving approach, for example—could be used, such as one that manages resources and orders actions, but it seems still unclear under which conditions a more generic approach would benefit the storytelling.

4.3.4. Generating stories

A very important challenge in the generation of a story using this character-based approach is achieving the variability needed in the while maintaining a well-defined story genre [15, 21]. This means that, in different plot instantiations, singular situations can occur, hence having different endings in the story. These situations, however, should generally keep the genre, here being it the sitcom.

Keeping a consistent genre has the benefit of helping the user to understand the course of events, and decide whether or not to interfere and how to do it [18]. The generation of a story, then, is the result of the dynamic interaction between the main characters’ plans [16] which correspond to a top-down approach, because the characters’ behavior is generated from their predefined HTNs. Nevertheless, while the story told is developing, some situations might appear, and they may not be part of the initial plans.

The interaction between the different characters’ plans results in random possibilities (and encounters) between agents that have a potential to create situations of narrative relevance [17].

These interactions constitute a bottom-up approach (because plan-based behaviors do not apply for these situations) and hence need some specific additional mechanisms: situated reasoning and action repair.

The situated reasoning [67] in the behavior of the plan-based actors is needed due to the fact that there exists an inconsistency between the agent’s expectations and the action preconditions. Situated reasoning is oriented toward obtaining a specific resulting state in a given situation, and hence becomes useful when it also considers avoiding undesirable results, such as being able to react to situations that appear from the spatial interactions of artificial actors [19].
If, for instance, the system randomly allocates the characters on the scenario before the story begins, although the characters will aim to follow their independent plans, they might get to a point where they are in situations that are not (and cannot be) explicitly represented as part of their plan - and the system cannot simply ignore these situations. The actions needed are a kind of “repair” actions.

Here, different positions of the actors can be related to different steps or reactions, but keeping a narrative perspective, the plans or sub-plans cannot simply fail or be extremely interrupted, since the overall plot’s relevance and coherence must be preserved, no matter the changes. A proper dramatization is part of the story.

In the system proposed, there are some cases where re-planning is needed (for instance, when a “repair” action cannot be executed due to the actor’s non-deterministic behavior, or even due to some timing issues, such as different action times for the characters, forcing the others to do the “repair” action or trying to influence the other characters so that the original action can be done.

These actions are related to the situated reasoning, but are not the same, even though the difference is subtle: while the action repair is dedicated to the restoration of the desired executability, or reaching the same final state as the original action, the situated reasoning consists in the interruption of the current plan and dealing with the specific situation. It therefore does so more from a dramatization perspective than from a planning perspective.

Regarding the actors, some key element must be presented. Even though the behavior of each actor is deterministic, there is a number of factors that contribute to make the overall action become unpredictable from a player’s perspective [18]: the initial position in the stages cause different beginnings for a story, the competition between the actors (aiming to get resources for action, in a dramatic sense), a random output of some actions, user intervention (object moving, hiding) and the mood status of each actor.

The initial position in the scenario, for instance, can greatly influence the emerging situations, and user intervention will also cause having conditions that do not end in the same results.

### 4.3.5. The user and the plot

The player of the system can watch the story as a mere spectator, and he or she can also observe the story from a particular character’s perspective, or navigate through the scenario, while the actions are being executed, acting as an omniscient observer. In the system, the user can choose to interfere with the goals of the characters, whose actions are dramatized by the use of animations which are appropriate to the story [16].

Here, an important issue must be taken into consideration: the characters are not improvising but are playing a role within the game and story, so their actions are
meaningful in a narrative context, and provide significant keys for user intervention (such as the case where the player chooses to hide an object the character is trying to get).

Even when the players can interact at will, in any time, there’s the need for them to be aware of the overall game intentions (in the game, Ross being interested in Rachel), so that they can see and react to the dramatic conditions that appear on the screen. The current version of the system allows to ways of interaction: by acting with physical objects that have narrative relevance, or by spoken advises using speech recognition.

The objects have narrative relevance since they are objects needed in key narrative functions (for example, chocolates or flowers, as gifts to Rachel, have important relevance when Ross is trying to get her attention), and can force the character to re-plan or change actions (such as when Ross wants to give chocolates but the box is taken off). Here, the user is making use of the basic “player” features of the Game Engine, navigating through the scenario and moving, hiding or stealing those narrative objects.

Regarding the speech recognition, speech intervention is used as a basic (and natural) way to influence the actions. The interaction is done through speech, but not in the sense of conventional user-agent commands, but more related to narrative conditions. The idea is not using the speech in direct commands (which would make the user to become a director of the story) but rather giving advises (such as helping with a “Rachel is in the room” when Ross wants to enter to read her diary). The system also makes use of Webber’s statements, with the description of generic rules of behavior that become relevant only when some specific situations happen.

### 4.3.6. Overview of the project

The system is capable of generating short stories (few minutes of length) that have Ross character’s perspective and goals as the main narrative element, and that end with Rachel’s answer when he asks her out, but has no defined method to measure the stories and their narrative relevance, though the results prove the fact that a character-based approach has a tremendous potential for the generation of stories.

The key subject is the consideration of deterministic techniques that provide unpredictability. The system has potential for the development of more complex stories (or narrative lines), increasing the number of characters of plans per character, for instance, but that will force the planning approach to face several limitations, and here different planning techniques (such as Bonet’s, [10]) can be taken into consideration, exploring issues such as the narrative control mechanisms and character’s improvisation.
4.4. Other institutions and projects

While the previous example is one of the possible approaches to interactive storytelling, and specifically covers a character-oriented approach, the following subsections provide some notes on additional researchers, projects and institutions that have other interesting results on the area. The selection includes the projects we find more relevant in relation to our ideas, but is obviously not intended to be a complete or detailed listing.

4.4.1. CMU – Oz

The Oz project, at Carnegie Mellon University, focused in approaching interactive drama within a context of immersive worlds with exciting characters and the possibility of many options, keeping a master interactive-storyteller that subtly controls the way things are presented.

In their system, interactive drama means the presentation (by computers) of highly interactive worlds, with dynamic and complex characters with pleasing stories [50], and they acknowledge a strong influence of Laurel’s [44] ideas in their motivations.

The project, done mainly in the early 1990's, developed a game engine that incorporated relatively simple, but believable character agents, and a drama planning mechanism that aimed for a more "believable" and entertaining game play. Overall realism in that gameplay was not a main objective, tough.

The project was one of the first to present results about providing all possible options to the interactor (as opposite to fixed choices, like in hypertext) with the aim of having him as the main protagonist, the one determining the curse of actions. Here, however, the existence of a destiny, created by the author of the interactive drama, aims to guarantee the drama component, embodying dramatic theory and principle, in order to create a cathartic experience.

The Oz system architecture included a simulated physical world, several characters, an interactor, a theory of presentation, and a drama manager. In the system, a model of each character's body and of the interactor’s body were kept in a physical world, while out of it a model of mind controlled each character's actions, with the interactor handling his own actions. The goal of the drama manager was to influence the characters' minds, hence affecting the physical world and the responses.

The Oz system had characters, presentation, and drama as the main research areas, considering them all equally important for creating a rich dramatic experience, with studies related to the creation of agents that appeared as reactive, goal directed, emotional, moderately intelligent [9, 50].
The project kept two different presentation models aiming to make it possible for artists to create interactive, dramatic systems: animated and textual. The animated system, a real-time animation version built on top of C, presented a graphical version of the world and the characters, with interactors using with the system physically, through sonar sensors and a mouse. The system produced a project named “The Edge of Intention” (or “The Woggles”).

![Image](image.png)

*Figure 20: The Edge of Intention*

The textual version uses text as the input and output vehicle, with the world and characters being described through text, and the interactor's actions being entered to the computer the same way. This system is built on top of Lisp, and is similar to the classic interactive fiction titles (like “Adventure” or “Zork”). Using this system they built “Lyotard”, a simulated cat that lives in a simulated apartment.

Bates’s work related to the project [8, 9] consisted in providing means to show the emotional state of the characters from a traditional Disney’s perspective: the state is clearly defined, the feelings are shown, and the emotions are emphasized.

The current developments related to the technology are done through Zoesis, a private company formed by J. Bates, the initiator of the project. M. Mateas keeps exploring AI, using it as a vehicle for the artistic expression [51, 52].

### 4.4.2. NCSU – Liquid Narrative Group

The Liquid Narrative research group at North Carolina State University uses different techniques from AI, Computer Gaming, Human-Computer Interaction, Virtual Reality and Cognitive Psychology, aiming to model narrative aspects of human interaction with computer systems.
They make use of the fundamental ideas from narrative theory, and aspire to provide computational models of interaction.

The basic idea in the approach of M. Young’s group, is understanding narrative as one of the fundamental means by which we organize, explain and understand our very own experiences. Young’s approach sustains that the aspects of narrative play a central role in our learning, our communication, our social interaction, our arts and, obviously, our recreation, and hence proposes to exploit this common orientation towards narrative in the structures of HCI.

In relation to our interests, the current focus and work done by M. Young related to the Mimesis system is one of their most relevant themes. The Mimesis system, still under development, is an implementation of an intelligent controller for virtual worlds that generates and aims to maintain a coherent, narrative-based storyline, involves AI planning and natural language discourse generation with the real-time control of the Unreal game engine in a full architecture. Mimesis is composed of different modules, like different Unreal clients, an extended Unreal server, and their own Mimesis Controller, handling the AI. While the server provides access to the Unreal environment, the controller is the source and control of the narrative-based interaction over time. The system uses HTNs (like Cavazza’s projects) for action representations.

The liquid narrative group has also some other projects which can be related to our areas of interest, such as Amerson’s camera control or Saretto’s supplementary work regarding narratives that are rewritten according to interaction and story progress.

### 4.4.3. SU – Virtual Theater Project

The Virtual Theater project, at Stanford University, aims to offer a multimedia environment in which users can play all of the creative roles associated with the production and performance of plays and stories in an improvisational theater company. The roles presented include: producer, playwright, casting director, set designer, music director, real-time director, and actor, and the system offers additional help on this matter: intelligent agents fill roles that are not assumed by the user.

In the system, animated actors perform a play in a set, under the supervision of an agent that manages stage and story, and they aim to offer “life-like” qualities: variability, idiosyncrasy in their behavior and affective expressiveness, at the same time they improvise. The player has then his own work, together with the improvisational performances of other actors. The main interest in the project is on building individual characters that can take directions from either the user or the environment, and act according to these directions in a consistent way that includes their emotions, moods, and personalities.
In addition to its primary merits as a test bed for research in AI, the project allows interdisciplinary collaboration, being a part of the Adaptive Intelligent Systems (AIS) project at Stanford.

The most relevant work found at Stanford, in terms of our interests, is that of P. Doyle, whose approach to agents takes a different starting point [24-26]: instead of aiming to provide smart agents in an environment, Doyle argues that simple intelligences that exhibit sophisticated behaviors can be obtained from the environment itself. The proposal separates the structures and leaves all abstract competencies in the agent, keeping the concrete domain knowledge in the environment. The justification for the efforts is that, as the “as the nature of online gaming changes to accommodate vast, persistent virtual worlds, there is a growing need to populate them with inhabitants capable of dynamic, adaptable, fruitful interactions with players”[24].

A very important use of the concepts provided is related to online games and multi-user experiences, since they are potentially much more difficult to have a strong dynamism and clear atmospheres and goals. While single-player experiences are most of the times correctly designed, taking care of the immersive characteristics the game needs, in multi-user environments the immersion into the story can easily be broken by the other players, hence braking the experience and causing, in the worst case, a lack of direction or a definite end.

Additionally, the work done aims to improve the believability of intelligent agents, particularly agents embodied in online worlds. Currently, as it has been mentioned in other sections of this report, the goal is to avoid creatures or characters with limited reasoning abilities, and improving the NPCs, which are, obviously, in need for this improvements. As it was presented in the overview of AI and computer games, NPCs are generally a stable population of agents that look and act like other players, performing ‘mundane’ tasks, and providing an overall sense of continuity. NPCs are hence enormously needed (to support narratives, to play the roles no player wants to take, playing supporting roles) and the goals from this perspective aim to offer improved believability in such NPCs.

In Doyle’s approach, virtual worlds offer great potential as environments for education, entertainment, and collaborative work, and the idea is seeking for the ability to acquire new behaviors and useful semantic information from the context, the environment allowing agents to quickly become intelligent actors in smart spaces [27].

The Virtual Theater offers another relevant concept that can be connected to our ideas: directed improvisation, a term coined by B. Hayes-Roth [32, 33], that proposed a new paradigm in HCI: while a user is in control of the characters, and ‘directs’ them with abstract instructions and constraints, the characters improvised, following the directions, but expressing a personal style while meeting the desired objectives together with new ones from their inspiration, but close to the original requirements.
4.4.4. UM – AI and computer games

John Laird and his work can be found in all citations related to the AI in computer games research areas. Laird’s research interest related to games center in an architecture for basic intelligence as a precursor to develop autonomous agents, the ultimate goal. The Soar architecture has already been mentioned in the Section, together with Laird’s projects [40-42].

Regarding the work done at the University of Michigan, the Haunt 2 system is very relevant to our ideas, since it develop an interactive adventure game that illustrates how human-level AI characters can make a real difference in the overall gaming experience.

![Figure 21: Haunt 2](image)

Haunt 2, the evolution from a text-based project, is implemented as a mod to Unreal™ and will eventually be publicly available. The gameplay focuses on interaction with intelligent NPCs (controlled by the Soar architecture), where, as the players, we control a ghost that tries to find a way back into his home world. The ghost character is strictly limited in its ability to manipulate the environment, since it can move or pick up light objects (such as a match or a piece of paper), but it can't move or manipulate heavy ones. Besides, the game proposes that metal drains the ghost ‘energy’, so that he must avoid metal objects. The constraints force the player deal with the AI characters, which have distinctive personalities in terms of their goals and reaction to the environment.

The game aims to integrate the knowledge-based, goal-oriented reasoning the group already has developed in older projects, with emotions, personality, and physical drives. As in other projects of Laird [40], the goal is to inspire the development of human-level AI characters and of new types of games these characters can make possible.
5. Our proposal: a cinematic approach

Once we have presented in the previous section some elements of the current works and projects related to interactive storytelling, we include what we believe are some areas where improvements can be further studied, and that can be translated in future research topics and a full research project.

5.1. The need for cinematographic systems

A strong narrative can make use of cinematic elements brought from conventional media, this is, make use of cinematography and its theory to support a story and its development. The need for cinematographic-like systems has already been studied and proposed for 3D environments, specifically in those projects that represent a virtual world with agents and users as actors [34, 65], but they have been implemented on reactive systems rather than making use of planning. As Tomlinson claims, emotion and motivations (as the central part of a cinematographic system) “are difficult to integrate into a planning approach” [65], but we believe this is not a solid reason to avoid the planning issues as an options of such a system or research area.

It is true that, while deciding the paradigm to use, a reaction-based system may be an alternative for a behavior-based autonomous cinematographer, but this applies in the case where the actors have full freedom to act as they want (such as in theatre-like sets, or theatre-like virtual worlds), but not in the case where the actors actually follow some rules when taking their own decisions, such as the storytelling projects, where the behaviors can be from the beginning the result of a planning approach [18, 21]. A mixed approach, combining improvisation with direction, can be obtained from the experience presented in those cases with actors being ‘directed’ but also having some freedom to improvise [32, 33].

The last concept is in some way also related to Funge’s [28] cognitive modeling, that empowers characters which, through some reasoning, work out a detailed sequence of actions satisfying an specification, when the animator only provided a behavior outline and a ‘sketch’ plan. In a related manner, these ideas have been considered in projects like [48], where the discussion centers around different types of control over synthetic characters in interactive stories, arguing that, to attain a deeper and more engaging control, in certain conditions, users should be able to inspect, disclose, and modify the characters minds. Tomlinson’s latest work [66] makes also use of partial autonomy.

The justification for further study and research in order to develop and provide cinematographic elements in a virtual storytelling environment arises from the fact that the acknowledged works that do cover the topic, fail to address the full needs of a complete cinematographic-like storytelling system.
The Virtual Cinematographer [34], for instance, uses a concept called *idiom*, a sub-unit of cinematographic expertise that captures the essence of a scene, and is actually able to shoot in an autonomous way different kinds of scenes, but by encoding a scenario in a very detailed manner, limits itself by not providing any true continuity among scenes (the transitions between the *idioms* force, or tend to this breaking), and by creating effective shots only in a scenario it is familiar with. This second aspect, related to the scenario, cannot be fully understood as a limitation (since in any storytelling environment, the scenario can be understood as pre-defined, as has been done with fixed ‘sets’, a *sitcom* scenario, for example [15]), but nevertheless does not provide any help with the topics of lighting, lighting design, or even interactivity in the sense of storytelling issues.

On the other hand, advanced projects that do cover lighting or expressiveness as topics [65], handle camera and lights within a 3D virtual world, but lack the planning or controlled approach, and are directed towards performance scenarios and not to virtual scenarios within a computer game or a narrative system. Currently existing systems such as [5, 6, 53], actually include constraints and problem-solving as key elements, but are aimed as helper-applications for virtual artists, and are not narrative systems but tools to create shots in virtual worlds.

These systems are apparatus that, even though actually create correct shots and transitions in a cinematographic sense, are merely static and aimed to help on issues such as the development of a short animated film with the help and creation of a storyboard. Tomlinson [65] efforts do provide a good set of knowledge regarding the importance of a cinematic system, but the work is aimed to a context with live-action characters mixed with virtual ones, and not related to a fully virtual storytelling environment.

Considering the previous elements, it makes sense to us to begin the development of tools, or a system, that can actually present the cinematographic elements needed in order to both improve the currently achieved projects, and provide a tool or accompanying development instrument in the field of virtual interactive storytelling.

The main objective of a cinematic approach is driving a story in a better way, heightening its impact by making use of a language that has been proved successful engaging spectators, and that has several formalisms that make it adequate to be translated to a system. A large set of notions and references for cinematographic language can be found in different texts [4, 39, 49, 54, 58, 59] dealing with film language, film structure, editing and cinematic development.

### 5.2. Cinematic areas

As mentioned, the possible areas that must be studied, in terms of the cinematographic impact, can improve the currently existing systems, or provide a new background for development that considers the narrative elements within a cinematic approach, by making use of the fact that users are used to this kind of language, and to the stories told in such a way.
Interactive media, from its beginnings, has a true like for adaptive stories. In terms of story creation, the possibility to have a setting and a character (or characters) that experience emotional changes and face a series of problems and actions has already provided several results, but if we expand this field to include a cinematic approach, the possibilities grow and the ability to provide results that are much more appealing to a standard user (say, a moviegoer) seems like a reality not to be dismissed. The cinematic approach is somehow under-studied, and its importance must be clearly taken into consideration.

As a language, film has a vocabulary made up of close-ups, establishing shots, reverse shots and the like, background music, lighting, and editing. The following areas are hence some possibilities for this cinematic approach within the context of storytelling systems.

### 5.2.1. Camera control

The camera is not only the eye within the virtual world, but the subtle element that can improve or change the way a story is being told. Here, the camera refers closely to the director element, the eye in front of the action, the eye that chooses the correct angle and the correct position in order to increase a desired effect.

A correct shot can greatly improve the dramatic experience of a story. A basic example would be the position of the camera in a particular scene that deals with a related emotion. If an actor is experiencing ‘fear’ when dealing with another actor, the camera might be lowered, and pulled-back, in order to provide an intimidating, threatening appearance. Generally speaking, a low camera means that the character is in charge; a high camera means that fate is in charge.

![Figure 22: Camera and actor (Tomb Raider™)](image)
The ability to extend the camera to a determined number of agents that can actually provide the camera capabilities in a multiple level will be absolutely important, considering the need for this knowledge when relating it to an editing phase of a cinematic experience.

Regarding the orchestration of the camera and the actor's movement, and closely related to editing, the following section, Barwood offers some basic but important considerations when making use of cinematic language in the computer games development [7):

- Close-up is important to cue important information in the gameplay.
- The focal length of lenses dictate perspective. Wide angles exaggerate motion and depth perception, while narrow angles reduce both.
- Scenes can be stages to look good on camera. Moving actors within a shot for emphasis is a mean; actors tend to stand closer together in a movie than they would in a real social situation.
- Larger image changes carry more power than small ones. Pace and rhythm is important.
- Shots have a life of their own. Something should happen in each one.

Even though the suggestions are aimed towards the making of the “cutscenes” (the introductions, and generally speaking "any non-interactive storytelling or scene-setting element of a game” [31]), the main ideas can be extended to the interactive part of the cinematic language used to present the stories.

The main goal of camera control is oriented to the dramatic emphasis: important things should be present if the shot needs them, a close-up is important to read an actor's mood or intentions, an actor shown alone can be used to emphasize his emotional state. Considering the stage, an over-the-shoulder shot can stress an enclosing social situation, and an insert (which can be defined as a close-up of hands or props or anything that doesn't include the actor's face) is often as expressive as anything else (for example, a match lighting a candle, feet rapidly using brakes to support the fact that a lead actor has just seen the girl he chases, etc.).

Generally, a film does not tell story from a particular person, and the camera moves wherever the best shot can be found, going out of buildings, falling, entering, following, etc. The movement is understood as natural, even when it will not represent a character’s point of view, since the audience is interested in the expression and not into the mechanics [7]. When needed, the point of view from an actor can be adopted, obviously, and the time this is kept depends on the story.
Camera control oriented to the film language sense of the term is a basic need in any narrative system that pretends to show convincing and believable stories.

5.2.2. Editing

Going far beyond the initial considerations of shot and cut, the ability to provide a series of techniques related to traditional editing or montage, can dramatically improve the cinematic potential of a system.

The possibilities for an editor agent within a cinematic environment can be understood in two main areas: an agent capable of influencing a shot based on editing theory, or an agent capable to determine the actual cuts and changes among takes.

The first option implies a previous knowledge or predictions about what will happen next (and hence suggest camera positions that will improve the edition), while the second is related only to the editing part, with angle and position changes related to a current event, such as the possibility of having two single shots of each speaker in a conversation supported by a two-shot.

Suspense is a key related to the editing, and a very important part of a good narrative: “when the audience is informed about the nature of a situation, that's suspense”[7]. The theory related this to pitfalls and goals: a couple of persons locked with cut to an insert of a bomb inside the room is suspense, as can be suspense a traffic jam supported by a correct proof that we are in a hurry but kept in the traffic.

Editing is obviously closely related to camera movement and position, since the overall sense of ‘navigation’ in a story is defined with both elements.
Editing principles relate to a correct narrative and storytelling can be found in specific bibliography [59], and include terms like the following:

?? Establishing shot, as the shot showing everything in place in a scene, so the spectator can see important spatial relationships and geographical locations.

?? Stage line, the imaginary (and elastic) line in the establishing shot.

?? Coverage, being the need for actors not to look at the camera, unless needed, and the correct directions in the shots establishing a conversation or continuous presentation.

?? Camera progress, meaning in chases the movement must generally proceed in the same direction from shot to shot, unless different elements are presented.

Regarding gameplay, and from a strictly oriented approach to computer games, different editing techniques can be adapted to the game creation while preserving this gameplay.

From [47]:

?? Inserting enough sequences to sustain the viewers' attention. In a game, cut scenes make a perfect addition to a high-paced game level. (This can be extended to storytelling with out the cutscenes)

?? Changes of camera shots allow visual diversity and stimulate the audience's attention.

With a good camera agent or system, the possibilities for editing seem quite reasonable, considering that every actor can have a virtual camera or director selecting the next shots.

The editing process must also be related to the overall atmosphere of a cinematic experience, since emotional or subjective tones can deeply be supported by a correct pace.

A basic example can be the case for a tense condition among a couple of actors, where faster cuts help to show the dramatic rhythm needed in, for instance, a tense-tone dialogue, or the required editing in order to provide a correct flow of takes in any given dialog, strictly aiming to support it in a cinematic way.

### 5.2.3. Lighting control

As of now, the lighting systems within a virtual scenario are quite simplistic, and the area of interactive lighting is quite forgotten or left aside. The main reason rests obviously in the fact that hardware limitations prevented the creation of elaborated dynamic light scenarios, but since that is no longer the case, within certain limitations, the lighting design can help improve the overall cinematic conditions.
In a photographic context, lighting is a powerful tool for developing and creating an atmosphere (which helps the creation of appealing characters and studies), but taking this into a dynamic context (where photos become frames), the light not only exists to support a scene (by providing some tone or emotional component) but can also improve the previous element, the camera, since the ability to control light, shadows and depth of field can greatly improve the dramatic effect of a correct camera positioning.

Furthermore, varying light intensity in a scene may lead the participant to focus his attention on more relevant artifacts (a technique already used in Myst™).

Lighting is also important for a sense of three-dimensionality and for emotional color [7]. The use of harsh lighting can make an actor to look angry or distressed, while soft lighting makes them seem romantic, just to mention a small example of possibilities.

### 5.2.4. Audio

Music and sound are a very important part of the narrative in a film. Sound design and composition is a well-known element of successful games, but in terms of a cinematic system, some ideas can be exploited, such as having score and sound-effects agents that can improve the expressiveness, together with the previously mentioned improvements, such as light or editing.

Luban [47], from a game development point of view, defends context-linked music themes and special sound effects, in a close relation with the overall editing. The suggestion is “never underestimate the descriptive power of sound”.

Projects like Tomlinson’s [65] already include sound effect changes related to the camera position, but they are not focused towards narrative, and are just used to show the existence of the “camera”, by adding simple ‘whooshes’ to make it seem much more alive.

The suggested approach should not only do the opposite (the camera agent should be transparent to the user, since it is intended to be of narrative help and not become an actor) but should be oriented towards the ability to change the score and the effects according to the pace, the emotional conditions of the characters, or the directing decisions that support the narrative in an overall sense.

The possibility of a musical director, and a sound editor, can be also part of a system with the goal of providing a complete film language set of tools to correctly narrate a story.
5.3. The project proposal

If we acknowledge that a coherent narrative is an obvious requisite of all virtual storytelling systems, the development of cinematic tools to make use of film language, supporting the narrative, becomes an important component to develop and explore.

The proposal consists in the development of an Unreal™ extension, a mod, in order to provide cinematic tools in a virtual storytelling environment. As it will be commented in the next section, the system can be thought in two different possibilities:

?? Development of a full interactive storytelling prototype that incorporates cinematic capabilities
?? Adaptation of an existing prototype to be able to incorporate cinematic components and capabilities

The prototype will be developed with the Unreal™ game engine having virtual actors and virtual agents capable of creating a story (or a set of stories) based on AI formalisms, such as could be planning techniques or other decision-taking mechanisms. The stories will present cinematic narrative, meaning the overall development of the storytelling will incorporate and make use of film language while telling the story. A possible structure for such a system is shown in the next figure, with the modules being part of a cinematic scheme that is part of an overall interactive storytelling system.

![Figure 24: Cinematic structure](image-url)
The structure shown provides a basic picture of the components and the sections of a cinematic arrangement, with agents controlling the visual presentation of a story within an interactive storytelling system. In relation to this structure, we present the relevant issues and aspects.

### 5.3.1. Screenwriter

The screenwriter or writer agent will be in charge of keeping the story information in such a way that is available to the directing and the editing processes, and obtaining information from the overall narrative/story status and the overall emotional elements within the story.

The screenplay component would then keep the information of the narrative elements of the story, such as the tone, the dramatic components, the story in general, the story goals, the values, the motivations of a character, the relationship among characters, etc. It can be understood as a narrative or screenplay agent that can be of help in terms of the screenplay proposal of the story, providing information that can aid the director agent, who will be the final responsible of the decision on the way a specific shot or sequence will be presented.

This section is closely related to the dramatic conditions of a scene, and the overall drama and emotional elements within the story, since it would be dealing with the dramatic values of both scene (story) and character or characters, the conflict being experienced, etc. Besides the current status of a scene or take, the agent must be aware of pre- and post-conditions of the story and the characters.

The emotional context defined in a specific moment of the story has an obvious and strong relation with audio and light.

### 5.3.2. Sound designer/composer

The sound agent will provide the audio conditions for the sequence or the mood, in relationship to the atmosphere and the overall story.

The agent must be aware of the emotional conditions of the story as well of the set or location being kept in the story told, in order to provide an adequate tone in terms of the music being played and the audible effects that are being experienced.

The relation with the dramatic conditions is straightforward, since audio and music will be able to set a tone that will help to establish the needed sequence after emotions have changed, locations have been presented, etc.

A basic separation between sound effects (or sound design) and music (as score) can be further studied, but we believe that, in terms of the initial conditions and possibilities of a system, both functions can be related.
5.3.3. **Cinematographer**

The cinematographer agent will be in control of issues related to both camera and light, helping the director agent by providing information on the suggested take under some conditions, or helping in the determination of the correct camera placement after considering light conditions, or location characteristics.

Together with the audio, the lighting information and decision will support the atmosphere and mood within the story. A lighting direction element will support moods, will aid on the remark of important cues (possibly together with the camera agent), and will help to provide the needed overall illumination condition in each take. Lighting information will provide overall cinematography conditions to help lighting decisions being taken.

If in the final structure of the system it is kept within the cinematic expertise of the directing agent, it will be a knowledge-base in terms of cinematography as an area within the film language.

The camera (which will be an agent too but merely acting to provide information on the scene) will provide basic information to support the director’s decision on specific shots, but is also in close relationship with the cinematographer.

Together, camera and cinematographer will help to support the tone (establishing, for instance, a correct sequence of two-shots and one-shots in a dialog), remarks (such as providing adequate close-ups) and overall cinematic tone. It provides the cinematic aids to support the overall expertise on camera-related matters.

5.3.4. **Editor**

The role of the editing agent will be to keep pace, rhythm, and an adequate sequence of shots to support the decisions of the director, in concordance to the story needs (screenplay, character development, mood, emotional states).

Besides the basic editing activities, the editor agent needs awareness about the next actions to be taken, to be able to produce adequate suggestions, such as the suggestion for a camera position that will improve the editing, or, in an opposite direction, to produce the adequate cutting scene after the camera moves.

Editing is closely related to the camera position (changing angles in a conversation, inserting two-shots and close-ups, etc.), and hence the communication with the directing agent is permanent to obtain an editing through the time of the story.
5.3.5. Director

The directing agent will be in charge of everything related to the scene, from character acting (suggesting behavior, allowing improvisation if applicable, suggesting movements to get a better shot, forcing movement to get that shot, moving camera, determining then next actions according to the screenplay, etc).

All the control, following of the key elements in the story, elections and final decision are the sole responsibility of the director, and hence he will be the overall responsible of a correct (and correctly executed) support to the narrative with the cinematic aids and language available.

A correct set of decisions in a cinematic context can greatly improve the dramatic experience of a story, be it a basic position of the camera in a particular scene that deals with an emotional condition, or the overall story being narrated from the first to the last frame.

The director will be in contact with all the main entities, in order to take his decision based on his expertise or the expertise he can obtain at that specific moment (annotated worlds will be included later in the document).

5.3.6. Overall roles

It is obvious that the previous descriptions are related to the conventional roles of a cinematic crew, and this is the result of the desired translation of the cinematic elements into the storytelling system. Further work and research will help to determine if this is the correct way to begin. The following image shows the structure together with the basic characteristics described above.
All of the above agents will be responsible of taking decisions, following the narrative and the story, and then will need to be agents capable of detailed decision-taking, and changing environments.

On the subject of the AI techniques to be used, advanced planning may be a possible solution [10], even using context sensitivity [29] to provide a richer way to interact, but this issue is further expressed in the last section of this report. An approach making use of annotated environments [26] will be further studied as well. As of now, no final decisions have been made on the matter considering this is a work in progress.
6. Conclusions and perspectives

This section provides some conclusions, ideas, and future research perspectives on the subject covered in the report, together with some personal thoughts on the subject and the project proposed.

6.1. Some personal reflections

Film language is something moviegoers and spectators are familiar with. As a language, it had to be learnt before but, once we had the basics, our ability to read it is part of the way we approach stories with our eyes. In film, aspects such as camera placement or editing are as important as the events that are being described with them, and a translation of these elements into a virtual scenario used for storytelling should be obvious from a developer’s approach with interests in the visual presentation.

Just by reminding the good experiences we can have, as moviegoers, after seeing a correctly produced picture, even when that certainty may be sort of inexplicable for those unfamiliar with film language having that certitude, it seems natural to propose to develop and support a project that deals with this cinematic language in the context of a storytelling environment.

First of all, a good cinematic system could not only generate correctly told (and good-looking) stories, but can certainly provide a huge amount of knowledge on what to do (and what not to do) when translating and adapting the cinematic standards and language to a system that aims to improve a storytelling context. Besides that, cinematography helps to establish relations among different characters, and considering the current conditions in the virtual characters that appear on any given story as of now, we find the area not only attractive but completely in need for further work.

In real movies and TV shows, it is known that some really good actors can sometimes appear in a film that is not considered to be ‘in the level’ of their standards. In some cases, the one to blame is the director of such films, since an actor is—in some cases—as good as the director asks (or ‘forces’) him to be, or as good as the director becomes. In other words, an actor is sometimes just as good as his director. Taking this concept to a virtual storytelling context, why not considering the need for a director within the story being told, even in those cases where improvisation takes the lead? why not providing a full set of directing and editing components to improve the narrative in a storytelling system?

Now that interactive systems, such as all those that have been mentioned throughout the report, are focusing in providing personality in the actors (whether it be to build relationships, or to solve a problem), and personality in the story being told, a cinematic system (and a cinematic approach in the development of the tools needed for such a system or concept) becomes what we believe is the
next stage in order to help improve what the actors are currently doing, to improve the current stages of a dynamic area. A cinematic system able to adapt, and to cover a wide range of conditions and elements that will likely be to happen and exist, is quite a challenge.

In the context of virtual actors and environments that have behaviors and responses that change dynamically, a system capable of balancing the internal and the external contexts of a story (the actions happening in the screen, and the way they are presented), can really help the creation of a better experience for the users of an interactive storytelling system.

If we are to tell a story, then why not telling it in a proper way, with a proved language and –quite importantly- a proved set of rules?

6.2. Conclusions

The current research on interactive storytelling focuses mostly on character and plot development. Both are completely important, but we believe that interactive stories need also to include investigation dealing with the visual presentation of the story and the way it is shown and told, in the context of applying a formal language to the visual components, in order to give a better shape to the story, to the whole dramatic experience. Cinematic language, translated from a movie to a virtual environment, should be a valid option, with different proved formalism that can be translated to a storytelling system.

While conventional media (theatre, cinema, performing arts) keeps as categorical the use of a correct expressive language to support the narrative desired, there are not many projects that can be publicly related to the development of a system or tools to change this condition. We believe an approach oriented to the application of film into virtual storytelling systems is a challenging approach and an open field to raise questions and produce interesting projects, aiming to provide better dramatic experiences and supporting the narrative relevance of the stories being narrated.

It is clear that recent advances in graphic rendering leave more ‘room’ for the development of AI in games and this kind of proposals. With the level of realism currently achieved in the graphic sense, further improvements in ‘realism’ will mostly come from both game physics and AI. The development of AI, here in the form of cinematic aids to narrative in a storytelling system, should also help the development of better or new game genres based on more sophisticated visual presentations.

Interactive Storytelling, as an area of application of AI, promises to have a major impact not only in the gameplay of the future of computer games, but to radically change the entertainment systems. We believe a cinematic-oriented system can become a very important and effective tool in the creation of appealing an catching storytelling projects, and can become a part of the steps towards a true cinematic experience in the virtual storytelling universe.
6.3. Research perspectives and future work

This document has been focused in the most up-to-date projects and information available, considering mainly a time-span of three years, due to the fact that the area has been just widely active enough in the last few years, and is until this span that we believe can totally be considered an area of its own in terms of having a larger set of information available.

While it has presented different references and ideas is, in no way, a closed report. Further analysis and overview of the needs of such a system and tools are needed, and additional references must also be obtained. The definite paths to follow will emerge from a deeper analysis of all the elements related to our idea, since there are several areas we have not wanted to include (or examine) due to the fact that the field is already becoming large and, while a lot of publications are being put available, some others are still vaguely presented but may exist.

6.3.1. Development

Regarding the election of a beginning point to the overall development we propose, the development possibilities can, for example, be roughly presented within this frame:

- Development of a full system
- Development of tools that complement an existing system

The final decision will be based in the progress obtained in the short term related to the setup of a fully working virtual environment, and probably in the support obtained from other researchers and institutions (that have now several working systems already finished).

The benefit of an approach based in working with extensions of a game engine have already been presented in the report, but for AI-oriented research, it seems immediate: the use of systems like Unreal™ provides readily accessible, completely stable, high-quality graphics, networking, database and process execution support for virtual environments, eliminating the need to begin from zero in many senses. In our case, a full development will take far more time than just developing tools and adapting an existing Unreal™ test-bed system, to begin deeper test and analysis but, since many of our ideas are not already been incorporated in current systems, we may find that creating a new environment may be the only way to begin. A final decision must be based in the final structure defined for our tools and extensions.

There are, obviously, a number of ideas and questions to explore and further contemplate, among which we can mention the following:
6.3.2. Interactive narrative structure

What is the correct choice of the narrative structure to be used in the system?

?? Authored vs. improvised

Even though our proposal is oriented towards the use of cinematographic techniques in the narrative, to support it, different approaches to the story told may need different ways to approach the use of film language in each case or conditions. While a mixed condition (for example an authored but character-improvised story) may be also a possibility, further study on the different needs for each condition is still needed.

In general, character-based AI systems have been recently validated to draw from and contribute to the state of the art in such areas as pattern recognition and planning [35], and hence a character-oriented approach, like that used by M. Cavazza [19] is a valid and tested option to work with.

6.3.3. Cinematic control

Election of cinematic control.

?? Will the player take over the visual control, forcing the director to react to his changes of camera (for instance) or the visual component will be changed back to the director’s election.

?? Will the ‘correct’ shot be determined after or before the player makes a change in the gameplay?

The election must be related to the real degree of interactivity given to the game, but must be taken into account in the general planning of the system and its possibilities.

6.3.4. IA for the cinematic agents

?? Can advanced planning techniques (10) be used for all the cinematic variables?

?? Can alternate approaches, such as an annotated environment [26] be used to provide intelligence to the cinematic agents instead of helping the characters?

?? Can all the film language parts be used with the election of a single technique?

Even though the film language and its formalism can be translated to a virtual environment, there is no much information available on a tested election for translating it to a virtual scenario with directing and editing agents.
If smart environments can be used to help the directing and the editing, for example, there is a downside, from a design perspective, of incorporating annotations, increasing the burden of the designer of the ‘sets’. However, since the sets are not to be changed dynamically, the approach can be explored.

As of now, annotations [27] have been used in emotions, reactive situations, problem-solving, role performance, and game playing. Annotated contexts with cinematic expertise for specific sets (dark environments, narrow corridors environments, open fields) seem like a good election for additional exploration. Besides a correct take in terms of emotions and character’s trajectory, for example, specific knowledge in scenes in a narrow corridor may force most of the takes to be, for instance, over-the-shoulder, with alternate inserts or close-ups. Here, the environment would be providing this kind of information to help the directing agent.

The development of such an architecture of virtual sets with well-designed environments and smart annotations is a topic in need of further study.

**6.3.5. Definition of a cinematic ontology**

Cinematic agents will need a film ontology to provide the expressive responses desired, and hence the definition of a growing ontology must be considered. Such a film description must be planned to be extended, but also be complete enough as to produce useful results even from the beginning stages.

Among the needed concepts to be included in a complete description, we can mention some basic ideas, with some basic descriptors. The list is provided merely as a basic guide of the kind of information to further detail and study:

**Lighting control**

- Back-light
- Environmental light
- Focused light/spotlights

**Audio control**

- Audio effects
- Environmental music
- Emotional/mood effects (character/tone)

**Camera control**

- Long-shot
- Full-shot
- Medium-shot
- Two-shot
Close-up
Panoramic shot
Tilting shot
Panning shot
Tracking/dolly shot
Crane shot
Over-the-shoulder shot

Editing control

Emotional status
Pace
Narrative context
Camera election/shot election

It is natural to consider that not all shots or elements must be included in an initial stage, but a complete understanding of all the possibilities must be taken into consideration when planning the overall system or tools. The final ontology will help to determine the final cinematic agents and their limits within the whole 'production'.

6.3.6. Multi-user environments

A very important challenge for a general cinematic agent (as the set of all the needed agents) is the context of a multiplayer context, where each of the players can act as a character, and the needing a completely different approach in terms of a generic director.

One director for each player?
One director for all the 'play' and story?

This concept must be further studied as well, but the multiplayer context with a cinematic agents seems like a larger challenge, since the ideal condition would need all possible players (all possible points of view) to keep a director for their specific condition, and hence the issue could become a matter of multiple stories, multiple POVs, that can be reduced to a pre-defined set of constraints and characteristics, being merely a variation for the single player condition with constraints like being in the opposite side of the room, for example. This topic is closely related to previously mentioned election of cinematic control.

6.3.7. A closing remark

While it is still early to define if it is possible to handle all of the above issues in a full sense, the project can certainly be used in a thesis, focused in all, some or even one of the elements mentioned, in a deep exploration. The decision will be based in further study, analysis, context evaluations, and the development process related to the ideas presented.
7. References


[71] R. M. Young, "Notes on the Use of Plan Structures in the Creation of Interactive Plot" presented at the AAAI Fall Symposium on Narrative Intelligence, 1999.
