

Using Navigation Meshes to Improve Storytelling Dramatization

Vinicius da Costa de Azevedo

Eduardo Ceretta Dalla Favera

Cesar Tadeu Pozzer

UFSM, Departamento de Eletrônica e Computação, Brazil

Abstract

In this paper, we focus on techniques to enlarge dramatization time in an Interactive Storytelling system, allowing, at the same time, the user to give tips defining preferences to be taken into account during the dramatization. The environment where the plot is to be dramatized was expanded and divided in regions and sub-regions, being all levels represented by graphs. We adopted a weak user intervention approach during dramatization, by means of which users can specify actors or kinds of scenes they prefer to watch.

Keywords: Camera techniques, agents, storytelling, Interactive TV

Authors' contact:

{pozzer, favera, azevedo}@inf.ufsm.br

1. Introduction

Interactive Storytelling is a research area in digital entertainment that has received growing interest recently. For entertainment, the generated stories serve to guide the dramatization of events inside games, and also for replacing hand-coded pre-established scripts, with a better chance to introduce interesting and surprising variations. Another use is the application of interactive storytelling in the context of digital TV, where users would play, in principle, a more passive role than in games, but may still want to interact with the story in diverse ways.

When designing techniques for applications where the user essentially behaves as an interactive spectator, like in an interactive TV setting, we must pay special attention to both interaction and dramatization. It is expected that the story has a structure similar to conventional TV programs (in respect to duration) and, at the same time, the user should be able to interact with during dramatization.

In this paper we focus especially on alternatives of weak intervention during dramatization, as well as means to enlarge dramatization time. These alternatives have been applied in a new version of the Interactive Storytelling Logtell system, which has been developed to run in an interactive TV environment.

Section 2 presents some related work and interactive storytelling models. Section 3 gives an overview of the Logtell approach. Section 4 explains how events are now detailed in order to enhance their dramatization. Section 5 describes the specification of

the scenario and new routing strategies that have been incorporated. User interaction at the dramatization level is presented in Section 6. Section 7 describes the director architecture. Section 8 contains concluding remarks.

2. Related Work

Different approaches to digital storytelling have been proposed. The suitability of each approach depends on the goal of each application. There are two main approaches: plot- and character-based.

In a *character-based* approach [Cavazza 2002] the storyline usually results from the real-time interaction among virtual autonomous agents that usually incorporates a deliberative behavior. The main advantage of a character-based model is the ability of anytime user intervention, which means that the user may interfere with the ongoing action of any character in the story, thereby altering the plot as it unfolds. For digital TV purposes, such a strong intervention requires an excessive user interaction.

By contrast, in *plot-based* models [Ciarlini 2005], characters that usually incorporate a reactive behavior should follow rigid rules specified by a plot. The plot is usually built in a stage that comes before dramatization. In a pure *plot-based* approach, user intervention is usually more limited. Such approach ensures that actors follow a predefined script of actions which are known beforehand. This approach best fulfills our requirements for light user intervention as well for planning dramatization in advance.

3. Story Generation

Logtell [Ciarlini et al. 2005] is a system that encapsulates story generation and visualization. Story generation is based on the logic specification of a model of the chosen story genre, where possible actions and goals of the characters are described.

In the context of fairy tales that was the example used to validate the Logtell approach, possible events generated by IPG are: Reduce_Protection, Go, Get_Stronger, Kidnap, Attack, Fight, Kill, Free and Marry. Possible characters are the knights Brian and Hoel (the heroes), Princess Marian (the victim) and Draco (the villain).

The following classical story can be generated by IPG: "The protection of Marian's castle is reduced. Draco regards that as an opportunity to kidnap her. Draco then goes to Marian's Castle, attacks the castle and kidnaps Marian. As a noble knight, Brian feels compelled to save her. He goes to Draco's Castle, attacks Draco's castle twice and then fights Draco. Finally, Brian kills Draco and frees Marian, who starts loving Brian as a result. Motivated by their mutual affection, Brian and Marian go to the church and marry each other."

4. Detailing Actions for Dramatization

In the new version of the system, we adopted the overall architecture of Logtell but we decided to implement the graphical engine again and change the way a scene is represented. Events are broken into atomic actions, that provide a higher level of detail and controlled by a nondeterministic automata specified for each kind of event [Doria et al. 2008].

Atomic actions are the real actions that are performed by the actors. Walk, kiss, talk, kill, are examples of this type of action. Each one of them is performed by the actors by means of minor operations, such as movement behaviors, animations, and so on. Not all possible atomic actions specified for an event are performed. Just a few of them are really necessary to represent a specific event. These **key actions** keep the essence of the operation and cannot be omitted.

5. Scenario Architecture

One major purpose of this paper is to find means of enlarging dramatization time and enriching graphical representation. The scenario organization is our key element. A 3D representation of the modeled world was developed using the Ogre 3D engine and is compounded by various scenarios. Each one exhibits part of a fairy-tale medieval-like world.

In contrast with the original 3D scenario representation presented in Logtell, the new conception distances were strongly increased to represent real traveling distances. This approach required changes in the way scenes are filmed. Filming continuously while an actor goes from one place to another, without cuts, would result in a tedious experience, since now it might take several hours to perform an event of type "Go".

To avoid this dreary experience, it was developed a logical structure connecting scenarios: a treeph (tree + graph) (Figure 1). A treeph is a connected graph, where each node has children that form a treeph. The structure takes advantage of both data structures: the tree hierarchical organization, which provides an organized way for dealing with levels of detail; and the graph suitability in path-finding algorithms.

Each node at Level 0 represents a scenario (continuous 3D area). Alternatively, each node, at any level of the treeph structure can be denoted as a **region**. If a region offers a number of possibilities to increase actor interaction, it must be detailed and divided into **sub-regions**, which would correspond to lower levels in the tree hierarchy. For example, a desert, when compared with a city, would not require so many refinements.

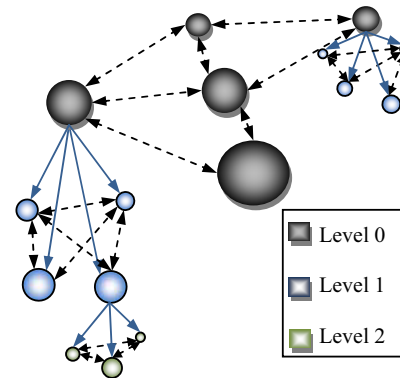


Figure 1: 3D perspective of the treeph

The regions and their sub-regions take advantage of the tree hierarchy. Children nodes inherit their parent's features since they belong to the same physical region.

5.1 Regions and Attributes

Each region has some associated attributes that characterize it and are used to select routes to represent events of type "Go", according to user's preferences, as presented in the following sections. Each attribute can be set to values within a numerical range. The lower the value, the greater is the region's level for this attribute. With this approach, path-finding algorithms can search for shortest (lightest) routes that respect user preferences.

In the current implementation there are four attributes:

- **Beauty:** How visually pleasant that region is. A waterfall or a green forest can be considered beautiful regions (lower values), in contrast with abandoned cities or garbage dumps.
- **Terror:** How creepy that region is. A dark forest should have a low value.
- **Safety:** How safe the region is. Safe places can be the princess's castle, in contrast with Draco's cave.
- **Population density:** How populated the region is.

5.2 Navigation Meshes

Each leaf node of the treeph was assigned to a set of waypoints that represent possible places actors can reach. With this approach, geographical path

information is going to be located only at the bottom of the treeph.

Besides being absolute reference position, waypoints provide useful information to the dramatization - actor's placement and camera takes - as they can provide information that different transitions between places require. Transitions between points that are physically connected, as frontiers from neighboring sub-regions, should be more subtle than transitions from different regions that are not connected. Additionally, in the last case, we need to deal with the loading time of scenarios. Wherefore, the waypoints are divided in three types.

- Path waypoint – The most common type. They are located inside leaf nodes and just provide information that guides actors to cross through the terrain.
- Short-transition waypoint – They are located in the border of the sub-regions and provide information about transitions between them. When the actor comes to a short transition waypoint, the virtual director (see Section 7) knows that is needed to perform a visual transition, which indicates that the actor is leaving one region and arriving into another. These waypoints are always pair-wise.
- Long-transition waypoint – They are located in the border between the regions of level 0. These regions require a different transition behavior with some parameters that enable the system to represent a transition between different contexts or places that are distant.

5.3 Selecting Routes

The aforementioned hierarchy of regions and attributes was designed in particular to support actions related to the movement of actors across the scene. Whenever an actor has to go from one place to another, it needs a path to be followed. We adopted an approach based on Dijkstra and A* [Buckland 2005] algorithms that finds a path with the following features: it must be connected, without cycles and the shortest in respect to the selected user attributes.

5.3.3 General Traveling Algorithm

Algorithm 1 creates the list of waypoints that connect the StartNode (SN) to the TargetNode (TG) specified by an event of type "Go". It has to take into account possible specified attributes, regions and their sub-regions and available waypoints. It presents two functionalities:

- Perform a top-down search level, starting from level 0. In each level it uses Dijkstra path-finding algorithm, searching for routes that fulfill specified attributes.

- For each leaf-node, using the A* algorithm, it finds a connected, shortest route among all the waypoints of the region. It doesn't take into account node attributes, but just real distances.

Functions `SelectStartWp()` and `SelectTargetWp()` find waypoints that are on the frontier of neighboring regions selected by Dijkstra algorithm, performed previously. `SelectStartNode()` and `SelectTargetNode()` select neighboring sub-regions that are connected by an graph edge of a subsequent lower level region.

Algorithm 1: Path-finding algorithm

```
WaypointList FindPath(SN,TN)
  Run Dijkstra(SN,TN) to find a route
    according user preferences
  For each node
    If node is a leaf
      wpS = SelectStartWp()
      wpT = SelectTargetWp()
      Run AStar(wpS, wpT)
      For each waypoint
        Add to the current path
        If waypoint is the target
          Return path
        EndIf
      EndFor
    Else
      SN = SelectStartNode()
      TN = SelectTargetNode()
      FindPath(SN, TN)
    EndIf
  EndFor
End
```

6. User Interaction

Once the route through regions can be selected in accordance with the values of their attributes, a weak intervention is granted and the generation of customizable digital content is possible. With just a few interactions, the user can change completely the dynamic of the plot representation. The user can intervene in two different (but complementary ways):

- By selecting the attributes to be considered; and
- By selecting the main actor, whose character should be emphasized.

With this approach, the same plot can be displayed in various different ways. The user could, for instance, witness the drama lived by the Hero to rescue the Princess being presented in beautiful scenarios that the protagonist would pass through. Alternatively, the malefic plans of the villain being thwarted by the Hero could be demonstrated in a terrifying world outlook.

7. The Director

In traditional film production the director is responsible for the overall decision making process. Automatic cinematography faces however two difficulties not found in the real world: the informal description of the rules of film-making are not explicit enough to be

directly encoded as a formal language. Moreover, most filmmakers work from a script that are read in advance and thus they have the opportunity to edit all the raw footage as a post process, not having to worry about user interaction like we propose here.

Our director is the agent that is responsible for the overall management of the plot dramatization, as shown in Figure 2. It communicates with the plot manager to receive the list of events that the user selected to dramatize. From the user, it obtains parameters such as preferable actor and desirable regions' attributes that should be considered to guide choices during the dramatization. The path-planner (see Section 5) is invoked to generate a set of waypoints that form a route an actor should follow, according the specified plot. Each actor receives a detailed list of atomic actions to be performed and finally, the camera module reasons about the better camera placement for each situation [Halper et al. 2001].

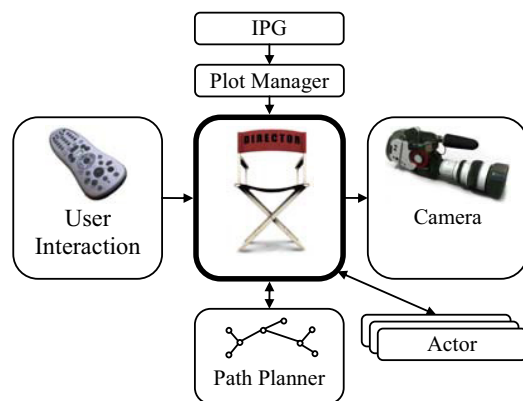


Figure 2: Director's architecture

The following algorithm summarizes the role of the virtual director proposed.

Algorithm 2: The director

```

1  Acquire the list of events from Plot Mngr
2  For each Event
3    If Event is of type "Go"
4      Invoke Path Planner to build the
        waypoints list
5      Collapse waypoints if necessary
6  EndIf
7  Define a set of atomic actions
8  For each atomic action
9    Place involved actors in the scene
10   Delegate them appropriate actions
11   Give camera parameters
12   Temporize events
13 EndFor

```

We adopt two strategies to remove unimportant and repetitive takes: we introduce the concept of Short- and Long-Transition waypoints, as presented in Section 5, and assign a level of importance to each event according to user specifications and the logic of the plot. Both are implemented in the line 5 of Alg. 2.

Moreover, the director can highlight events of type "Go" of any character in any context if the user selects this actor as the main one to be emphasized.

8. Concluding Remarks

The use of distinct regions to represent the scenario favors the diversity of the scenes that are graphically represented and provides the means to move actors across the environment extending the duration of an event according to the convenience of the dramatization.

The use of the path-finding strategy, based on weighted-graph search, also brings a good solution to highlight features according to user preferences. Our test scenario is still small, composed of just a few places. However, it was enough to test the concepts of regions with associated attributes, refinements of events of type "Go" and ways of user intervention prior and during the dramatization.

The weak user intervention proposed proved to be appropriate and effective to be applied in contexts where user is not expected to interact continuously.

9. References

- AZEVEDO, V. C., POZZER, C. T., Creating a Director for an Interactive Storytelling System. In: VI Brazilian Symposium on Computer Games and Digital Entertainment, November 2007.
- BUCKLAND, M. Programming Game AI by Example. Wordware publishing Inc, 2005.
- CAVAZZA, M., CHARLES, F. and MEAD, S., 2002. Character-based interactive storytelling. IEEE Intelligent Systems, special issue on AI in Interactive Entertainment, 17(4):17-24.
- CIARLINI, A., VELOSO, P., and FURTADO, A., 2000. A Formal Framework for Modelling at the Behavioural Level. In Proc. The Tenth European-Japanese Conference on Information Modelling and Knowledge Bases, Saariselkä, Finland.
- DORIA, T. R., CIARLINI, A. E. M., ANDREATTA, A. A Nondeterministic Model for Controlling the Dramatization of Interactive Stories. In: Proc. ACM MM2008 - 2nd ACM Workshop on Story Representation, Mechanism and Context - SRMC08, 2008, Vancouver, Canada, 2008.
- CIARLINI, A., POZZER, C. T., FURTADO, A. L. AND FEIJÓ, B., 2005. A logic-based tool for interactive generation and dramatization of stories, ACM SIGCHI International Conference on Advances in Computer Entertainment Technology - ACE 2005, Valencia, Spain, 133-140.
- HALPER, N., HELBING, R., STROTHOTTE, T., 2001. A camera trade-off between constraint satisfaction and frame coherence. in Eurographics, volume 20.