IPT-EGVE Symposium (2007)
B. Fröhlich, R. Blach, and R. van Liere (Editors)

STORM: a generic interaction and behavioral model for 3D objects and humanoids in a virtual environment

N. Mollet[1,2], and S. Gerbaud[1,2,3], and B. Arnaldi[1,3]

[1]IRISA Rennes, France
[2]INRIA Rennes, France
[3]INSA Rennes, France

Abstract

We present in this paper research focused on interaction and behavior specification for 3D objects in general, including humanoids. This research is conducted in the context of a collaboration with Nexter-Group (a French military manufacturer) in order to introduce Virtual Reality (VR) in maintenance training. The use of VR environments for training is strongly stimulated by important needs of training on sensitive equipment, sometimes fragile, unavailable, costly or dangerous. Nevertheless, for the development of such applications, the re-use of existing developments is a major issue. Our research is focused on models that have been designed to achieve this re-usability and standardization for the efficient development of new virtual environments. In particular, we defined a new generic model named STORM, used to describe reusable behaviors for 3D objects in general, including humanoids, and reusable interactions between those objects.

Categories and Subject Descriptors (according to ACM CCS): I.6.5 [Model Development]: Modeling methodologies

1. Introduction

All about industrial equipments is complex and enormous: size, cost, maintenance operations, dangerous manipulations, etc. Virtual Reality (VR) appears to be a good solution for training on such equipments, due to costs, risks (for human people and materials) and new pedagogical actions which are not possible in reality. Creating VR environments for training requires the development of complex software, mixing multiple disciplines and actors: graphical and behavioral computer engineering, physics simulation, pedagogical approaches, specialized know-how in the field of training, etc. More generally, the efficient development of complex virtual environment is a major issue. The aim of our research is to propose a *full author-platform* for building VR training applications. Thus, we propose *generic and reusable approaches*, which can be applied in Virtual Environments in general. This paper presents one of the models developed for the platform: the model of behavioral objects and interactions.

The first part of this paper is a state of the art on interactions and behavioral objects in virtual environments. The second part presents our contribution in this area: how to describe complex behaviors and complex interactions with multiple complex objects and how we can design reusable objects and interactions for the efficient development of new virtual environment. We will see how this model can be used with a very complex behavioral object: a virtual humanoid. The third part presents the application and validation of our research in an industrial project named GVT. Then, we finish this paper by a conclusion.

2. State of the art - Interactions and behavioral objects in Virtual Reality

Our work relates to two main areas, presented in two separate parts. The first one deals with the way the behavior of 3D entities has been described and simulated in virtual environments. The second one deals with the description and simulation of interactions between such entities, and how human people can interact with. In those two parts we analyze the pertinence of approaches, and their potential re-usability in other context. Then we will conclude by underlining some interesting characteristics and problems.

2.1. Modeling behavioral 3D objects

Research focused on modeling the behavior of 3D objects belongs to the field of *behavioral animation*. Three main approaches [Lam03] exist: stimulus response systems, rule systems and state machine systems. Approaches based on stimulus response systems generally use grids of interconnected nodes which have different roles. A stimulus enters and evolves in the network to finally generate the response of the system. Thus, Van De Panne [vdP93] proposed a *SAN* network to control the evolution of 3D entities in an area. Granieri and Badler [GBR*95] used a *SCA* network to model the behavioral animation of a human agent. Those works are really interesting to obtain reactive, adaptive and reusable limited behaviors. But such approaches have a low-level of abstraction, and don't propose a fine control of the behavior obtained.

The Reynold's flock of birds simulation [Rey87] and the Tu's virtual fishes animation [TT94] relieved of the *rules systems* approaches. They have proposed to simulate and animate 3D virtual animals with simple rules given to each entity to model their own behavior. In those systems, a set of rules is defined, and is used to deduce a response to a set of input data which represents a vision of the environment. In comparison with stimulus response systems, this approach offers a higher level of abstraction. But, as the number of rules is strongly increasing, deductions could take much time, which is a problem in a real-time application.Furthermore, the re-usability of such approach is limited as it is necessary to re-use the whole rules-system.

The last family of systems used to describe the behavior of 3D objects is the state-machines approach. There is three main drawbacks to describe a complex behavioral animation with *simple* state machines: it is hard to construct, to understand and to maintain [Kal01]. But research has conducted to define new kinds of advanced state-machines, which can be used in this context:Badler [BW95] has, for example, proposed the *Pat-Nets* model of state-machines to control simple SCA network behavioral animation. Finally, simple state-machines have mainly evolved in this context to hierarchical and parallel state-machines. In this area, we can cite the work of Donikian [Don01] on *HPTS* language.Applications have been made in many areas like driving simulation or human behavior simulation. In comparison to other approach, rules-systems can appear easier to use, as this is a *declarative* description of the behavior. But the use of advanced state-machines allows a fine control without long deductions, and increases the potential re-use of several behaviors.

2.2. Modeling interactions with and between behavioral 3D objects

As we can define behavioral 3D objects in an environment, we are now going to analyze different approaches to define interactions between those objects. Complex interactions need a high-level of abstraction to be defined, and this issue has been mainly formalized in works on virtual humanoids. The first kind of interaction is the *direct manipulation* of 3D objects. There are many references [Han97] in this area which present different approaches to do such interactions: to take control of an object, to turn it, to move it, etc. Thus, Duval and Tenier [DT04] proposed an original solution for an easy re-use of behavioral objects in different virtual environments. Their approach consists in *increasing* a behavioral object with a generic adapter, which allows a direct manipulation.

Interactions of a virtual humanoid on behavioral 3D objects can also be recorded and replayed. That's a classic method, which has been massively used in video games. All the interactions with an object are recorded with tracking devices, and then a selection mechanism replays the interaction. The re-usability of such an approach is limited as it requires a large number of data acquisition. And the replay has to be *prepared* by pre-positioning the humanoid and objects in interaction.

Another approach is based on Artificial Intelligence techniques. This *top-down* view is based on real humans. For example, to interact with a door, a human being sees the door handle and then uses his memory and deduction to choose what to do and how.With the work of Hildebrand [HEHV03], a user can ask a humanoid to interact with objects in a virtual environment. The states and behaviors of the humanoid and the objects are managed by an expert-system. Although this approach is really interesting, due to real-time constraints it doesn't offer, for the moment, complex and various interactions with many objects and many configurations.

The last family of interactions are based on informed environments. In those approaches, behavioral 3D objects of the virtual environment contain a description of what they can propose as interaction or how to use them.Kallman's smart-object [Kal01] contains the description of the whole interaction. The humanoid is then able to be managed directly by the object to realize an interaction. The work of Badawi [BD04] is in the same idea, but here the humanoid uses the information offered by the object to realize the interaction by himself. Those works are really interesting from the point of view of re-usability and ability to offer complex interactions in real-time. But they are all limited to interactions between behavioral objects and virtual humanoid.Jorissen [JWL04] has proposed a generalization of this: in his model, virtual humanoids are objects like the others. This approach is really interesting for the generalization, but the authors are for the moment limited to simple interactions.

2.3. Conclusion of the state of the art

To conclude this state of the art, we can underline those few points. Concerning the behavioral animation, modeling be-

haviors of 3D objects with advanced state-machines appears to be an efficient solution which allows complex descriptions and re-use of behaviors and objects. For the description of interactions between behavioral objects, we can say that informed environments appear to offer more flexibility and re-usability than the other approaches. Nevertheless, there is for the moment a problem of particularization of the virtual humanoid, which doesn't allow to describe complex interactions between complex behavioral objects *in general*.

3. STORM: Simulation and Training Object-Relation Model

We tried to mix and to generalize different interesting properties found in our state of the art, in order to propose efficient and innovative solutions for the generalization and the re-usability. We define STORM in two parts. First, STORM contains a model of behavioral object. The main idea here is to propose reusable objects, which can interact easily with newest and potentially unknown objects. The second part of STORM is a model of interaction, which defines a general and standard process, valid for all the objects in the virtual environment. Then we present the STORM engine which animates all the objects. Afterwards, we describe the modeling of a humanoid in STORM. And finally, after giving a list of properties and fields of application of our model, we give some examples of use.

3.1. STORM behavioral object

A STORM behavioral object is composed of:

- a set of activities. A behavioral object is an object that has different activities, those ones completely define the behavior. A first part of them are important only for the object itself. That is, for example, related to a local adjustment or an internal calculation. Other activities are more important for the other objects and more generally for all the environment. That second part of the object activities is the public vision of the object behavior, and is important for interactions.
- a set of communication interfaces. A communication interface is a standard protocol of communication. Those ones represent the public part of the behavioral object. Each communication interface is related to a precise activity, as we can see in the next paragraph. Therefore, an interface is very interesting from the behavioral object point of view: the means of communication between objects is totally independent of the objects themself. There's two main advantages: from an implementation point of view, the communication protocols can change, objects are not concern. And, from a general point of view, when a new object is created, it just needs a standard communication interface to be able to communicate in the environment.
- a set of capacities. The public behavior and the linked communication interface is a couple named *capacity*. A behavioral object is a composition of different capacities and internal private activities. A capacity traduces a special public activity and a dedicated protocol of communication. For example, a plug has got a male-screwing capacity. It offers characteristics like the size and screwing commands in the communication *interface*, and screwing *activities* (screwing states and actions) from the behavioral point of view. This capacity can be cumulated with others capacities, for example general mechanical aspects: the capacity related to the gravity and the movable capacity.

As we can see on the figure 1, a behavioral object is composed with a set of cumulated capacities, and a set of internal activities. Creating a behavioral object with STORM consists in choosing the capacities of interaction we want to add to an object. This approach which consists in increasing object capacities, offers the possibility to easily *re-use* many standard and common capacities, and also to easily *adapt* existing objects. For a simple capacity re-use example, the screwing capacities are always the same, just a few characteristics are particular, like the size. So, when we want to have a new screwing object, a screwing capacity is simply added to 3D basic (or already advanced) object.

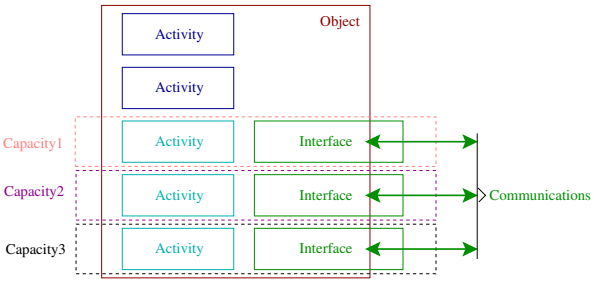

Figure 1: *STORM Object pattern*

3.2. STORM interaction : the relation

A relation represents the link that exists when objects enter in interaction (figure 2). It is a specific behavioral object. It inherits from the standard object pattern we have just described. A relation manipulates such standard objects. One problem we can identify through our state of the art is the place where to define the interaction between objects. The relation mechanism has to give a response. The interaction is not defined in one of the objects. It is not fully *distributed* between the objects either. It is located in what we call a relation. And it has to manage all the objects that are in interaction. A relation uses the capacities of the objects to create the interaction between them. For example, a screwing relation will use both male and female screwing capacity of a plug and a female support, to make the screwing link between them. The relation realizes the interaction between objects, and uses their standard communication interfaces to transfer

parameters and commands. As a relation is a standard object of the environment, a relation can manipulate other relations. So, a relation can divide works between sub-relations, more specific. It offers a standard process for all the interactions in the environment, based on capacities, and enables an easy re-use of the developments.

On the figure 2, we can see the mediator position of the relation, which is the center of the interaction between objects. Here, the interaction is between three objects: a plug, a female support and a screwing tool. Those objects have compatible capacities, which allows the relation to make a screwing interaction between them. One of the role of the screwing relation is to transfer movements of an appropriate tool to the plug, and to realize the screwing consequence in the female part. It also maintains the rigid link which could exist between the plug and the female support. Those objects do not communicate directly with one another, but they communicate with the relation via the communication interfaces.

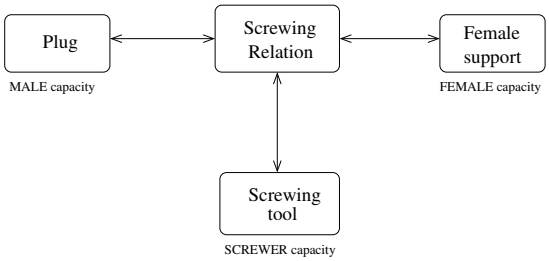

Figure 2: *STORM Relation: the link of interaction between behavioral objects.*

3.3. STORM Engine

The role of the STORM engine is firstly to combine capacities between objects to deduce potential interactions. So it can be consulted to propose a list of valid interactions between objects, which also depends on the various internal states of those objects (states of objects are represented in their interfaces) and relations already used with them. If a new interaction is wanted in the environment between the objects, the STORM engine will create a new relation between them. Its role is also to maintain the coherence between all the objects and the relations in the virtual environment. It has to animate the different relations, and to check the equity between concurrent relations that realize interactions between shared objects.

3.4. STORM example: modeling of a humanoid

In order to illustrate an advanced use of STORM, we present the modeling of a very complex behavioral object: a humanoid. This humanoid can be either the representation of a real user in the virtual world (an avatar) or a virtual human (controlled by the system). We focus on the interactions that can be triggered between this humanoid and the other STORM objects that composed the virtual environment. A humanoid has many other activities as a decision making process, an animation mechanism, etc, that are not described here because they are out of the scope of this paper.

A humanoid is composed of inner resources (body parts) like his hands, his legs, his gaze, etc, which are STORM objects with capacities: legs have the moving capacity, a hand has the grasping capacity, etc. A humanoid is actually an unbreakable STORM relation between its inner resources. The humanoid relation manages these resources and aggregates their capacities, and can additionally have global capacities. The humanoid relation receives all the requests for interaction and dispatch them to the corresponding resource if necessary, depending on these resources states and availability. A humanoid is also in charge of ensuring coherence between its resources, for example concerning their locations in the virtual environment. A resource has two states: free when it is engaged in no other relation than the humanoid relation, and busy when it is engaged in another relation. When a resource is busy, it can not be used by another relation. We treat the hand has a special resource since it has an additional state: holding an object (when engaged in a grasping relation). Indeed when a humanoid holds an object, his hand holding the object can be used by another relation: the grasping relation increments the capacities of the humanoid instead of blocking it.

In order to illustrate how the interactions append with a humanoid, we will exhibit an example of a humanoid who wants to drive a nail in a plank: Figure 3 shows the sequence of interactions required. At the beginning, the humanoid already holds a nail and wants to take a hammer. This humanoid has the grasping capacity and he wants to interact with a hammer which has the grasped capacity. The STORM engine dynamically detects that an interaction is possible thanks to the grasping relation (step 1). The humanoid relation transfers this interaction to his free hand (hand 2: the one with the grasping capacity). The relation between this humanoid hand and the hammer become then effective so that a nailing interaction between the humanoid and a plank is now possible (step 2). The humanoid transfers this interaction request to his grasped objects (the hammer and the nail), and the interaction appends (step 3).

The decomposition of a humanoid in resources depends on the interactions possible. When a body part can be monopolized by a given relation, we must particularize it as a resource that composed the humanoid. On the contrary, it is not relevant to particularize a body part that can be used by a relation but that needs not to be kept by this relation, we can simply add the corresponding capacity to the global humanoid relation. Let us take the ears as an example. If the only relation that needs them is the communication relation, no need to include this resource in the humanoid relation, it is sufficient to increase the humanoid relation with a lis-

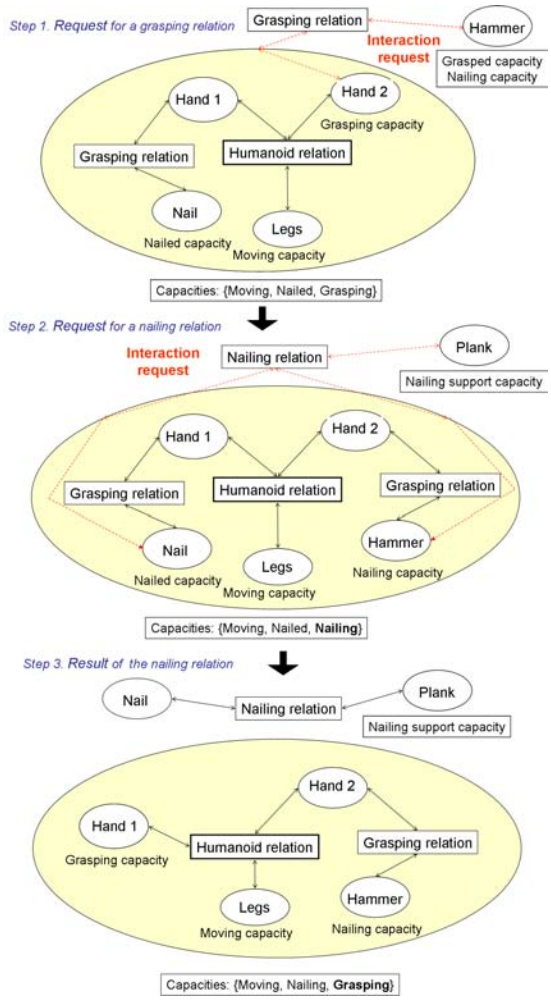

Figure 3: *nailing: sequence of interactions.*

tening capacity. The humanoid representation is thus simplified. However, we may want to set up a noise relation that connects the humanoid with a noisy object (e.g. a started engine) so that it prevents the humanoid from using its ears to communicate. In this case, the ears must be a resource of the humanoid with two possible states: free or busy.

To conclude, our STORM model of a humanoid is adaptable, depending on the interactions that the humanoid can enter with, thanks to its decomposition in resources. It takes advantage of the STORM model so that the humanoids can interact not only with any other STORM objects, but also with one another. This opportunity allows them to communicate or to collaborate for example, no matter if they are virtual humans or avatars of real users.

3.5. Main attributes of STORM

We can list some interesting properties of this model, which are solutions to the genericity and re-usability objectives:

- the bidirectional link of interaction represented by the *relation* offers flexibility and standardization: there is no particular object in the virtual environment. Humanoids are here objects with capacities of interaction, like any other behavioral objects.
- objects can easily be increased with reusable capacities of interaction. Furthermore, objects can directly be re-used.
- object behaviors can be very complex and can be described with any technique, as the model does not impose how the behavior has to be realized. So it can use a network of nodes, a hierarchical parallel state-machines or a declarative system: it can mix different techniques. A STORM object just has to offer interfaces of communication to define capacities, which allows other objects to interact with it. As a consequence, it is possible to re-use previous behavioral objects developed without STORM, those objects just need to be *adapted*.
- the local processing of the interaction is stable and dynamic. It is also easier to modify and maintain.

4. Industrial application and validation of STORM in the GVT project

The GVT project started in 2001, and is developed in a Research/Industry collaboration with three partners: IRISA and CERV laboratories, and Nexter-Group (Giat-Industries). This last partner is an important french industrialist, specialized in military equipments such as the Leclerc tank. The main current application of GVT is virtual training on Nexter's maintenance and diagnosis procedures. That is what can be found in the latest release of GVT, which is now an operational *industrial product*. GVT has been designed to be used on a network(figure 4) with one trainer and several trainees at different places, with different equipments, levels and procedures to learn. We are now interested in collaborative procedures where real users and virtual humans work together [GMA07].

A complete description of GVT, its kernel and its organization can be found in [MGA07]. GVT is not only a platform which offers several virtual training sessions, it is also a full-author platform, composed of authoring-tools which allow non-computer scientists to realize new virtual training environments with an efficient re-use of all the behavioral objects. Here is a short overview of the kernel of GVT. It mainly contains three engines: the STORM engine, presented in this paper, has to propose a reactive informed environment composed of complex interactive objects. The LORA [MA06] engine describes the procedure to realize, while the pedagogical engine [MGA07] supervises trainee's actions. Using the STORM model, we have realized several complete and operational applications. Complex and rich en-

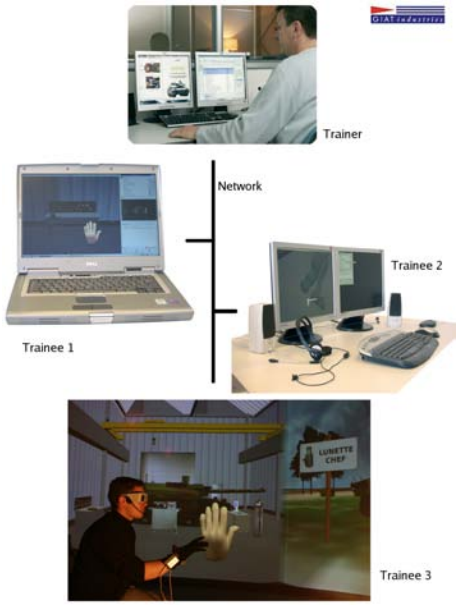

Figure 4: *GVT on a network: Virtual Training with one trainer and distant trainees on different hardware configurations (stereovision, voice synthesis, hand tracking, etc.)*

vironments have been defined, where all the objects and capacities can be reused for other environments thanks to the authoring-tools. As a last example of the use of STORM, we simulate the evolution of pressure between two complex hydraulic objects that trainees have to screw with dedicated tools. By using the objects capacities, STORM realizes the different 3D animations, but also the physical link, the local *relation* with the management of the pressure conducted in an hydraulic network composed in real-time by trainees.

5. Conclusion

STORM model is a generalization and a standardization of existing works. It proposed a complete solution to design complex behavioral objects and complex interactions between such objects in virtual environments. It also allows the author of such environments to re-use previous and future developments, and provide a standardization of humanoids as *objects* of the environment. As an application of our research, GVT is a very challenging project, involving two laboratories and an industrial company, and has conduct to a first *industrial product* release. Developed with our models there are now about thirty true industrial training scenarios on seven different equipments, such as the Leclerc tank. Our models give operational responses to GVT needs, and are turned towards the future. Special acknowledgments to all the participants of the GVT project.

References

[BD04] BADAWI M., DONIKIAN S.: Autonomous agents interacting with their virtual environment through synoptic objects. In *CASA2004* (2004), pp. 179–187. 2

[BW95] BADLER N., WEBBER B.: Planning and parallel transition networks: animation's new frontiers. *Pacific Graphics* (1995), 101–117. 2

[Don01] DONIKIAN S.: Hpts: a behaviour modelling language for autonomous agents. In *AGENTS '01* (2001), ACM Press, pp. 401–408. 2

[DT04] DUVAL T., TENIER C. L.: Interactions 3d cooperatives sur des objets techniques avec openmask. *Mecaniques et Industries* (Mai 2004). 2

[GBR*95] GRANIERI J. P., BECKET W., REICH B. D., CRABTREE J., BADLER N. I.: Behavioral control for real-time simulated human agents. In *SI3D '95* (1995), ACM Press, pp. 173–180. 2

[GMA07] GERBAUD S., MOLLET N., ARNALDI B.: Virtual environments for training: from individual learning to collaboration with humanoids. In *Edutainment - Hong Kong* (2007). 5

[Han97] HAND C.: A survey of 3d interaction techniques. *Comput. Graph. Forum 16*, 5 (1997), 269–281. 2

[HEHV03] HILDEBRAND M., ELIENS A., HUANG Z., VISSER C.: Interactive agents learning their environment. In *Intelligent virtual agents* (2003), Springer, pp. 13–17. 2

[JWL04] JORISSEN P., WIJNANTS M., LAMOTTE W.: Using collaborative interactive objects and animation to enable dynamic interactions in collaborative virtual environments. In *CASA2004* (2004), pp. 223–230. 2

[Kal01] KALLMANN M.: *Object Interaction in Real-Time Virtual Environnement*. PhD thesis, Polytechnique Lausanne, 2001. 2

[Lam03] LAMARCHE F.: *Humanoides virtuels, reaction et cognition: une architecture pour leur autonomie*. PhD thesis, IRISA Universite Rennes1, 2003. 2

[MA06] MOLLET N., ARNALDI B.: Storytelling in virtual reality for training. In *Edutainment* (2006), pp. 334–347. 5

[MGA07] MOLLET N., GERBAUD S., ARNALDI B.: An operational vr platform for building virtual training environments. In *Edutainment - Hong Kong* (2007). 5

[Rey87] REYNOLDS C. W.: Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics 21*, 4 (1987), 25–34. 2

[TT94] TU X., TERZOPOULOS D.: Artificial fishes: physics, locomotion, perception, behavior. In *SIGGRAPH '94* (New York, USA, 1994), ACM Press, pp. 43–50. 2

[vdP93] VAN DE PANNE M.: Sensor-actuator networks. In *SIGGRAPH '93* (1993), ACM Press, pp. 335–342. 2