View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Infoscience - École polytechnique fédérale de Lausanne

From Individual Human Agents to Crowds

Daniel Thalmann, Soraia Raupp Musse, Marcelo Kallmann

In this paper, we first try to identify which mechanisms are to be simulated in order to implement truly virtual humans or actors. Starting from a structure linking perception, emotion, behaviour, and action, we emphasize the central concept of autonomy and introduce the concept of Levels of Autonomy. Finally, we propose a new abstraction for the specification of behaviours in complex virtual environment simulations involving human agents, groups of agents, and interactive objects endowed with different levels of autonomy.

1 Introduction

Virtual human agents (also referred to as *virtual humans*, *agents*, or *virtual actors*) are humanoids whose behaviour is inspired by that of humans [Meyer/Guillot 94]. They are equipped with sensors, memory, perception, and behavioural motor that enable them to act and react to events. They can also be much simpler, like guided by users in real time or interpreting predefined commands. We will use the term *group* to refer to a group of agents, and the term *object* for an interactive object of the environment. Agents, groups, and objects constitute the entities of the simulation.

Agents inhabit a dynamic and unpredictable world. To be autonomous, they must be able to perceive their environment and decide what to do to achieve the goal defined by their behaviour. The actions are then transformed into motor control actions. Hence, the design of a behavioural animation system raises questions about creating autonomous actors, endowing them with perception, selecting their actions, their motor control, and making their behaviour credible. They should seem spontaneous and unpredictable and give an illusion of life, making the audience believe that the actor is really alive and has its own will. Credibility of an actor is accomplished by emotions clearly expressed at the right moment [Bates 94]. It is the observable emotions of an actor and the way it reacts that makes it appear like a living being with needs and desires. Otherwise, an actor would look like an automaton. Moreover, emotions make actors, placed in the same context, react differently. By defining different emergence conditions on different actors for their emotions, the resulting emotions are ensured to be different and leading to different behaviours.

Daniel Thalmann, Swiss and Canadian, is Full Professor and Director of the Computer Graphics Lab in EPFL. He is a pioneer in research on Virtual Humans. His current research interests include real-time virtual humans in virtual reality, networked environments, artificial life, and multimedia.

Soraia Raupp Musse is just completing her Ph.D. thesis at EPFL Computer Graphics Lab. Her current research is in the area of Computer Animation. She has worked on human crowd simulation in her doctoral thesis. She has participated in the eRENA and COVEN European projects.

Marcello Kallmann is in the Computer Graphics Lab at EPFL since February 1997 as a Ph.D. candidate. His topics consists in enhancing the interaction with general objects by using a feature modelling approach.

2 Modelling the Properties of Virtual Humans

The ultimate objective of modelling actor behaviours is to build intelligent autonomous virtual humans with adaptation, perception and memory, capable to act freely and emotionally, and to be conscious and unpredictable.

2.1 Perception

Perception is defined as the awareness of the elements in the environment through physical sensation. This is achieved by equipping the agents with visual, tactile and auditory sensors so that they simulate everyday human behaviour such as visually directed locomotion, handling of objects, and responding to sounds and utterances. The most important perceptual subsystem is the visual system. A vision-based approach [Renault et al. 90] is ideal for modelling a behavioural animation, and offers a universal approach to passing information from the environment to the actor in the context of path searching, obstacle avoidance, and internal knowledge representation with learning and forgetting characteristics.

At a higher level, we may decompose perception as suggested by [Becheiraz/Thalmann 98]. An actor's perception may be restricted to the objects and other actors in the neighbourhood. But this limits the number of possible behaviours because only the presence and the characteristics of an object or an actor are implied in selecting a behaviour; the actions of the other actors are not taken into account. The perception module produces three types of perception: the perception of the presence of objects and actors, the perception of actions of actors, and the perception of actors performing actions on objects.

2.2 Emotion

Emotion may be defined as the affective aspect of consciousness: a state of feeling, a psychic and physical reaction (like anger or fear), subjectively experienced as a strong feeling, and physiologically involving changes that prepare the body for immediate vigorous action. Actors must be capable of responding emotionally to their situation and acting physically within it. Apart from making the actors more realistic, visible actors' emotions provide designers with a direct means for affecting the user's own emotional state. Actors are therefore equipped with a simple computational model of emotional behaviour, to which emotionally related behaviour such as facial expressions and posture can be coupled, and which can be used to influence their actions.

An emotion is a person's reaction to a perception. It leads him or her to respond by a facial expression, a gesture, or to select a specific behaviour. An emotion happens between a perception and a subsequent reaction. Two different persons can have different reactions to the same perception, depending on how they are affected by this perception.

[Ortony et al. 90] describe an emotional model. Emotions are caused in reaction to objects, agents' actions, and events. The class of emotions caused by events can be classed in three groups of emotion types:

- the emotions caused by potential events.
- events affecting the fate of others, and
- events affecting the well-being of the actor.

Each class is characterized by emergence conditions for each of its emotions and variables affecting its intensity. The emotions felt by an actor are caused by its perception. Although some perceived objects, actors or actions are necessary for the emergence of an emotion, they may not possess the required qualities with sufficient intensity to produce an emotion effectively pewrceived by the actor.

2.3 Behaviour

Behaviour is often defined as the way in which animals and humans act, and is usually described in natural language terms which have social, psychological or physiological meaning, but which are not necessarily easily reducible to the movement of one or two muscles, joints or end effectors. Behaviour is also the response of an individual, group, or species to its environment. Behaviour is not only reacting to the environment but also includes the flow of information by which the environment acts on the living creature as well as how the creature codes and uses this information.

Behaviour may be described in a hierarchical way. The behavioural model decomposes a behaviour into simpler behaviours which may themselves be decomposed further. Each level of this hierarchical decomposition contains one or more behaviours performed either sequentially, or concurrently. A level of the hierarchy containing several behaviours to be performed sequentially is called a *behaviour*. Each behaviour of a behaviour sequence is called a *behavioural cell*. A behavioural cell contains behaviours which are performed either concurrently, or exclusively when inhibition rules are specified. The behaviours contained in a behavioural cell are either behaviours or elementary behaviours. A behaviour allows recursive decomposition in the hierarchy. An elementary behaviour is situated at the bottom of the hierarchical decomposition and encapsulates a specialized behaviour which directly controls one or more actions. A behaviour is executed by recursively performing each of the behaviour, behavioural cell and elementary behaviour entities at each level of the hierarchical structure of the behaviour.

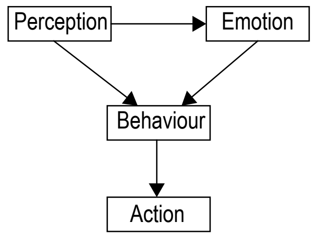

Fig. 1: Structure of the behavioral model

A high level behaviour reacts to sensorial input and uses special knowledge. A means of modelling behaviours is the use of an automaton. Each actor has an internal state which changes with each time step according to the currently active automaton and its sensorial input. To control the global behaviour of an actor we use a stack of behaviours. At the beginning of the animation the user pushes a sequence of behaviours (the script) into the actor's stack. At the end of the current behaviour the animation system pops the next behaviour from the stack and executes it. This process is repeated until the actor's behaviour stack is empty. Some of the behaviours use this stack too, to achieve subgoals by pushing itself with the current state on the stack and switching to the new behaviour. When this new behaviour has finished, the automaton pops the old interrupted behaviour and proceeds. With this behaviour control using a stack, an actor becomes more autonomous and creates its own subgoals while executing the original script.

2.4 Action

Based on perceptual information, an actor's behavioural mechanism determines the actions to perform. Actions may have several degrees of complexity. An actor may evolve in its environment, or it may interact with the environment, or communicate with other actors.

Actions are performed using a common motion architecture. The action module manages the execution of the actions used by a behaviour by animating a generic human model based on a node hierarchy. It allows the concurrent or sequential execution of actions by managing smooth transitions between terminating and initiating actions [Boulic et al. 97].

The animation is driven by a behavioural loop. The role of the behavioural loop is to update the state of the virtual world. At each iteration, the time is incremented by a discrete time step. To update the state of the virtual world, the loop updates the state of each object and actor. In the case of an actor, the perception is first carried through, then its emotions are generated, before its behaviour and its actions are performed:

```
repeat
    for each object and actor perception
    for each actor emotions generation
    for each object and actor behaviour execution
    for each object and actor actions execution
```

2.5 Memory

Memory is usually defined as the power or process of reproducing or recalling what has been learned and retained, especially through associative mechanisms. Memory is also the store for things learned and retained from an organism's activity or experience, as evidenced by modification of structure or

	guided	programmed	autonomous
Memory	Generally not provided	Generally not provided	Connected with others parameters internally to agents
Learning	Not provided	Not provided	Can be present
Autonomy	Low	Medium	High
Self-control	Not provided	Not provided	Result of a behavioural complex process using other internal parameters
Perception	Generally not provided	Should be provided	Can be vision, structure-oriented and connected to other parameters, e.g. action.
Behaviour	Driven-oriented	Program-oriented	Agent-oriented. Result of behavioural process using other parameters, e.g. perception.
Action	Driven-oriented	Program-oriented	Agent-oriented decision.
Motion	Driven-oriented	Program-oriented	Motion planning based.

Table 1: Comparison between different levels of autonomy

behaviour, or by recall and recognition.The implementation of memory in an actor is not very complex, as the memory is already a key concept in computer science. For example, [Noser et al. 95] propose a global 3D visual memory that allows an actor to memorize the environment he sees, and to adapt it to a changing environment.

3 Autonomy and Levels of Autonomy

3.1. Autonomy

Autonomy is generally defined as the quality or state of self-control. [Bourgine 94] defines an autonomous system as a system which has the capacity to guess viable actions. In cybernetics and in cognitive psychology, autonomy has always been strongly connected with self-organization [Courant et al. 94]. Hence, computer scientists sometimes prefer the following definition of autonomy: “the capacity of a system to maintain its viability in various and changing environments”.

The need for autonomous behaviour of actors arises from several considerations:

- in *virtual environments* where users are aided by actors for learning, equipment training etc.
- in *computer-generated films*, more autonomous behaviour built into the *virtual humans* saves work for the designer in creating complete scenarios
- in *simulation of real-life events*, like building evacuation in case of fire etc.
- in *interactive games*, autonomous behaviour creates the illusion that the actors are real humans.

Agent level of automomy	Agent goes to a specific location	Agent applies a specific action
Guided	Agent needs to receive during the simulation a list of collision-free positions	Agent needs to receive information about the action to be applied
Programmed	Agent is programmed to manage the information of a path to follow while avoiding collision with other agents and programmed obstacles	Agent is programmed to manage where and how the action can occur
Autonomous	Agent is able to perceive information in the environment and decide a path to follow to reach the goal, using the environment perception or the memory (past experiences)	Agent can decide about an action to be applied. This action can be programmed, imitated or existent in the memory (past experiences)

Table 2: Levels of autonomy present in different agent-oriented tasks.

Different parameters of agents’ simulation can be defined to achieve a compromise between different requirements: interactivity, complex behaviours, intelligent abilities and frame execution rate are directly related to the *level of autonomy* of each simulation entity. We distiguish three kinds of behavioural autonomy:

- *Guided autonomy* denotes the lower level of autonomy where the behaviours have to be informed by an external process (user, other system, etc.).
- *Programmed control* implies the use of a notation to define possible behaviours; the entity translates this information into internal behaviours.
- *Autonomous behaviour* is the capability of contacting independantly, exhibiting control over the internal state.

When less compromise with complex behaviours is necessary in the simulation, “less autonomous” agents can perform best in terms of frame execution rate and interactivity. The classification presented in Table 1 shows the main difference between the three levels of agent control. Different systems can be developed, blending the control in one single simulation. For instance, an agent can have a programmed or guided motion, but also memory with learning processes in order to achieve new behaviours, which can have more priority than the programmed or guided behaviours. Table 2 exemplifies the three level of actor autonomy, using two different agent tasks.

3.2. Levels of Autonomy Related to Groups of Agents

For the simulation of crowds of virtual actors we aim to avoid dealing with individual behaviours. Our goal is to describe methods to provide intelligence focused in a common group entity that controls its individuals. Figure 2 shows the correlation between these two parameters.

We call the crowd a *groups-based application,* and say *group applications* when individual complexity is less required. In this latter case, the intelligence abstraction is included in the groups. Although some crowd simulations seem to be composed of autonomous agents, the individuals are controlled by complex groups’ behaviours. However, some rule-based behavioural animation can be used in simulations formed by small groups.

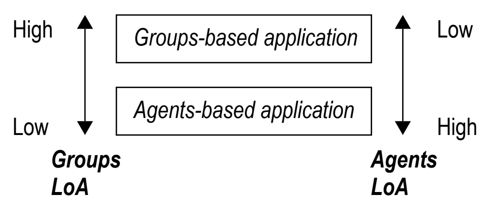

Fig. 2: Correlation between groups and agents level of autonomy (LoA).

Crowd LOA	*Group goes to a specific location*	*Group reacts to matched event*
Guided	During the simulation, the group needs to receive a list of positions "in-betweens" in order to reach the goal.	The group needs to receive information about the matched event and the reaction to be applied.
Programmed	The group is programmed to manage the information of a path to follow, avoiding collision with other agents and programmed obstacles.	The group can manage programmed events and reactions.
Autonomous	The group is able to perceive information in the environment, and decide a path to follow to reach the goal, using the environment perception or the memory (past experiences).	The group can perceive a matched event and decide about the reaction to be applied. This reaction can be also programmed or existent in the group memory (past experiences).

Table 3: Levels of autonomy present in different group-oriented tasks

With respect to levels of autonomy, we classify the crowd behaviours:

i) *Guided crowds*, with behaviours defined explicitly by the users;
ii) *Programmed crowds*, with behaviours programmed in a scripting language;
iii) *Autonomous crowds*, with behaviours specified by rules or other complex methods.

Table 3 exemplifies this classification of crowd autonomy using two different crowd tasks.

In the hierarchy found in crowds systems (crowds, groups and agents), complex structures like memory, decision etc. can be defined in the group level: the agents do not need this information.

3.3. Level of Autonomy related to Objects

When the simulation has to handle complex agent-object interactions, many difficult issues arise, because each object has its own movements, functionality and purposes. There is a range of growing complexity for possible agent-object interactions in a virtual environment. Examples are actions like grasping a fruit, automatic doors that open when agents are nearby, and also complex ones like entering a lift.

One could imagine that agents' perceptions can solve all the reasoning and planning processes necessary to achieve a simple task like, for instance, a single-hand automatic grasping of a small object. But this is no more possible for interactions with objects that have an intricate proper functionality like the lift example (Figure 4). Moreover, even for simpler interactions as our grasping example, we did not consider semantic aspects, e.g. recognising through sensors if a given fruit can be eaten or not.

A first approach to overcome these difficulties is to maintain a table with some semantic and grasping information for all graspable objects. Another approach models all possible object interaction features like its functionality and semantic information, containing also a complete description of all possible interactions it can offer to the agent. In fact, more information related to the object increases its level of autonomy. In the scope of simulations in virtual environments, increasing an object's level of autonomy moves it from a guided state to a programmed state, and a complete autonomous state. In the lowest level of autonomy, the object only knows possible movements it can apply to its parts. In the highest level, the object has all interaction information necessary, in a form of pre-defined plans, to take control over the agent and make it perform the interaction. In a mid-term, the programmed object controls its moveable parts, based on the agent decisions taken during the interaction.

Table 4 illustrates how an agent proceeds according to the different levels of autonomy for three different interactive objects in the environment: a door that opens with a simple lateral translation movement, a direction sign, and a two-stage lift. Depending on the level of autonomy of each object, different sensors (table 4) are required in the agent to perform an interaction. Such sensors can be difficult to control and expensive in terms of both computer memory and computer process time. To minimise such costs, and depending on the application, it can be interesting to use highly autonomous interactive objects. That means adopting the strategy of leaving inside each interactive object a list of available pre-defined plans that are automatically updated depending on objects' internal states.

3.4. A new Abstraction for Specification of Behaviours

With the abstraction levels guided, programmed and autonomous behaviours in mind, we present a schema that includes the entities group and object, as showed in Figure 3. We can so classify a simulation in terms of the autonomy distribution among its entities, i.e., a simulation S_i can be expressed as a function of three components: agents, groups and objects:

$$S_i = f\,(LOA(Agents),\ LOA(Groups),\ LOA(Objects)).$$

Object LOA	*Door*	*sign*	*Lift*
Guided	The agent has to move its arm to an reachable and meaningful location of the door, and control its movement, and open the door.	The agent recognises that the sign has an arrow and recognises the direction.	The agent recognises where the call button is, how and when the door opens, how and where to enter inside the lift, when and how to get off, etc.
Programmed	The agent has to move its arm to the right place, but the door opens by itself.	The agent recognises the sign, but the direction is given with no recognition.	The agent accesses the current lift state and decides its moves accordingly.
Autonomous	The door takes control of the agent, telling exactly the place where to put its hand and the complete movement of the door	The sign gives a new direction to go for each agent that passes nearby.	The lift takes control of the movements of the agent and gives him a complete plan, based on primitive actions, to perform the interaction.

Table 4: LOA present in different objects-oriented tasks

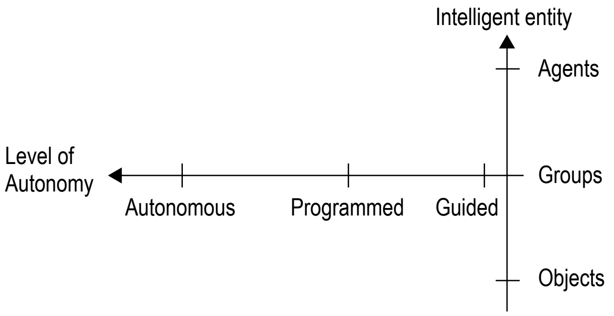

Fig 3: Level of autonomy vs. intelligent entity.

This way, depending on the application, we can choose the best distribution of autonomy to adopt. In general, if the simulation focusses in the behaviour of a given entity, this entity might have a maximum level of autonomy.

Interesting cases arise when we choose to have different levels of autonomy among the individuals of a same entity type. Consider, for example, the case of a simulation of autonomous agents, with a limited set of sensors, interacting with objects. The objects of the environment which are simple enough to be guided by such agents can be initialised as guided; while other more complex objects can be initialised with more autonomy. We must adopt a consistent strategy of priorities and negotiation in simulations where two entities with high LOA need to achieve a common task. One example is when an autonomous agent receives a complete plan from an autonomous object to achieve some interaction task. In this case, the agent will use its sensors to validate, or even improve, the plan. For this to happen, both entities must be capable of negotiating, with a common notion of priorities. A similar negotiation is needed when an autonomous agent is required to follow a behaviour that comes from its current autonomous group control.

We exemplify these simulations with the description of three simulations:

i) *Simulation of autonomous agents in a train station, with interaction between chairs, counters and a lift* (figure 4). Most are guided objects. The lift, which has a complex functionality, is autonomous. Thus, we regard the overall object autonomy as medium (programmed): the agent perceives and interacts with the different objects, sits on chairs, buys a ticket in a counter, and takes the lift. Because of the limited set of agent perceptions, the agent just accepts the autonomous behaviour of the lift, no negotiation is done .

ii) *Simulation of groups of agents involved in a political demonstration* (figure 5). The groups have their motion programmed, but are autonomous in their perception of other agents and their ability to react to the proximity of others. As in the example i), we chose to represent this mixed control as medium level of autonomy. The agents are programmed according to the groups' behaviours.

iii) *Simulation of a party populated by autonomous groups* (figure 6). The groups have two possible emotional states: *social* (representing groups that prefer to walk and meet others than eating) or *hungry* (the opposite idea). Through the meeting of groups, the emotional state can be dynamically changed as a function of sociological rules. If a group decides to eat something, the autonomous object table controls the interaction with the table of food.

4 Conclusions

In this paper, we propose a new abstraction to be considered in order to distribute the autonomy among the entities of the simulation. The idea we dealt herewith shows the possibility to improve the frame execution rate as well as to optimise the complexity required, by distributing some knowledge and

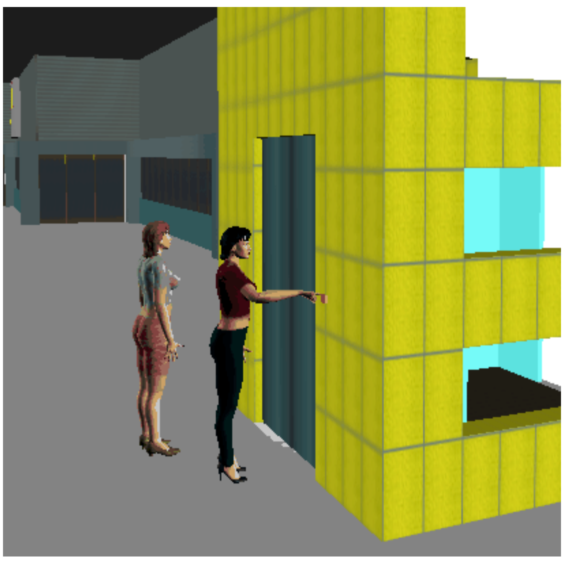

Fig. 4: Agents interacting with a lift. We used a lift programmed as a smart object to generate the Image.

Fig. 5: Political demonstration where autonomous and programmed groups are able to interact with others and react as a function of it. We used a crowd simulation framework to generate the image.

Fig. 6: Autonomous groups interacting with autonomous objects. Image generated mixing a smart object control inside the crowd simulation framework.

autonomy to other entities of the simulation: groups and objects. We tested this paradigm in the context of a *Virtual City* project [Farenc et al. 99] because we have to simulate several virtual human agents that can act in differently ways and apply different actions.

Acknowledgements

The research was sponsored by the Swiss National Research Foundation, FUNDEPE, CAPES and CNPq (Brazilian offices of Education and Science).The crowd simulation has been developed in the framework of the ESPRIT project eRENA financed by the Federal Office for Education and Science.

References

[Bates 94]
J. Bates: "The Role of Emotion in Believable Agents". In *Communication of the ACM* Vol. 37, No. 7, pp. 122–125, July 1994.
[Becheiraz/Thalmann 98]
P. Becheiraz, D. Thalmann: "A Behavioural Animation System for Autonomous Actors personified by Emotions", *Proc. First Workshop on Embodied Conversational Characters (WECC 98)*, Lake Tahoe, USA.
[Boulic et al. 97]
R. Boulic, P. Bécheiraz, L. Emering, D. Thalmann: Integration of Motion Control Techniques for Virtual Human and Avatar Real-Time Animation. In *Proc. VRST '97, ACM Press*, pp. 111–118, September 1997.
[Bourgine 94]
P. Bourgine: Autonomy, Abduction, Adaptation, in: (Magnenat Thalmann N, Thalmann D, eds), *Proc. Computer Animation '94*, IEEE Computer Society Press.
[Courant et al. 99]
M. Courant, B. Hirsbrunner, B. Stoffel: Managing Entities for an Autonomous Behaviour, in this issue.
[Farenc et al. 99]
N. Farenc et al.: "A Paradigm for Controlling Virtual Humans in Urban Environment Simulation". *Applied Artificial Intelligence Journal. Special issue on Artificial Intelligence. 1999* (to appear)
[Magnenat Thalmann/Thalmann 94]
N. Magnenat Thalmann, D. Thalmann: "Creating Artificial Life in Virtual Reality" in: (Magnenat Thalmann and Thalmann, eds) Artificial Life and Virtual Reality, John Wiley, Chichester, 1994, pp.1–10
[Meyer/Guillot 94]
J. A. Meyer, A. Guillot: "From SAB90 to SAB94: Four Years of Animat Research". In: *Proceedings of Third International Conference on Simulation of Adaptive Behaviour.* Brighton, England, 1994.
[Noser et al. 95]
H. Noser, O. Renault, D. Thalmann, N. Magnenat Thalmann: "Navigation for Digital Actors based on Synthetic Vision, Memory and Learning", *Computers and Graphics*, Pergamon Press, Vol.19 (1995), No 1, pp.7–19.
[Ortony et al. 90]
A. Ortony, G. L. Clore, A. Collins: The Cognitive Strucutre of Emotions. Cambridge University Press, 1990.
[Renault et al. 90]
O. Renault, N. Magnenat-Thalmann D. Thalmann: "A Vision-based Approach to Behavioural Animation", *The Journal of Visualization and Computer Animation*, Vol. 1 (1990), No. 1, pp. 18–21.

Further Readings

P. Becheiraz, D. Thalmann: "A Model of Nonverbal Communication and Interpersonal Relationship between Virtual Actors". In *Proceedings of Computer Animation 96*, Geneva, 1996 This paper presents a model of nonverbal communication and interpersonal relationship between virtual actors.

M. Cavazza, R. Earnshaw, N. Magnenat-Thalmann, D. Thalmann: "Motion Control of Virtual Humans". *IEEE CG&A*, Vol. 18, No. 5, Sept/Oct 1998, pp.24-31.
This paper proposes a new classification of synthetic actors according to the method of controlling motion, interaction and control of face and body.

M. Kallmann, D. Thalmann: "Modelling Objects for Interaction Tasks", *Proc. Eurographics Workshop on Animation and Simulation*, 1998.
This paper introduces the concept of smart-objects to specify entities containing interaction information of various kinds: intrinsic object properties, information on how-to-interact with it, objects functionality, and also expected agent behaviours.

P. Maes, T. Darell B. Blumberg, A. Pentland: "The ALIVE System: Full-Body Interaction with Autonomous Agents". *Proceedings of Computer Animation 95*, pp. 11–18 (Geneva, April 19–21, 1995).
The authors describe a full-body interaction system with a virtual dog.

S. R. Musse C. Babski, T. Capin D. Thalmann: "Crowd Modelling in Collaborative Virtual Environments". *ACM VRST /98, Taiwan*
In this paper, a crowd model has been introduced using different abstractions of behaviours, like the term guided crowd.

C. Reynolds: "Flocks, Herds, and Schools: A Distributed Behavioural Model", *Proc.SIGGRAPH '87, Computer Graphics*, Vol.21, No4, pp.25–34
This paper describes the first use of a behavioural model to produce a flocking behaviour.

X. Tu, D. Terzopoulos: "Artificial Fishes: Physics, Locomotion, Perception, Behaviour", *Proc. SIGGRAPH '94, Computer Graphics*, pp.42–48.
This paper describes a system where autonomous fishes living in a physically modelled virtual marine world.

D. Zeltzer: "Task-level Graphical Simulation: Abstraction, Representation and Control". Making them Move: Mechanics, Control and Animation of Articulated Figures. Edited by N. Badler, B. Barsky and D. Zeltzer. pp 3–33. 1991.
This paper presents a classification of levels of interaction and abstraction required in different applications.