Simulation Level of Detail for Virtual Humans

Cyril Brom, Ondřej Šerý, and Tomáš Poch

Charles University in Prague, Faculty of Mathematics and Physics
Malostranské nám. 2/25, Prague, Czech Republic
brom@ksvi.mff.cuni.cz

Abstract. Graphical level of detail (LOD) is a set of techniques for coping with the issue of limited computational resources by reducing the graphical detail of the scene far from the observer. Simulation LOD reduces quality of the simulation at the places unseen. Contrary to graphical LOD, simulation LOD has been almost unstudied. As a part of our on-going effort on a large virtual-storytelling game populated by tens of complex virtual humans, we have developed and implemented a set of simulation LOD algorithms for simplifying virtual space and behaviour of virtual humans. The main feature of our technique is that it allows for several degrees of detail, i.e. for *gradual* varying of simulation quality. In this paper, we summarise the main lessons learned, introduce the prototype implementation called IVE and discuss the possibility of scaling our technique to other applications featuring virtual humans.

1 Introduction

Virtual humans are typically thought of as software components *imitating* behaviour of a human in a virtual world in a *believable* manner. Virtual humans are equipped with a *virtual body* which is graphically visualised. As the research and development in this field is maturing, increasingly large virtual environments are starting to be populated by virtual humans (v-humans), e.g. [2, 15, 17]. A lot of effort has been invested to examining graphical level of detail (LOD) to capitalise from the scene-believability/computational-complexity trade-off. However, almost nothing has been done *outside frustum* to cope with the problem of limited memory and processor resources, despite the growing need for such solutions, at least in the fields of computer and serious games.

One possible approach to the problem with limited resources is to optimise the algorithms needed, both graphical and others, as demonstrated to a large extent e.g. in [17]. However, obviously, we can not optimise beyond some limit. After we reach it, we are forced to commit ourselves to the *simplification* approach. This line of thinking compels us to gain computational resources on the expense of quality of the simulation at the places that are unimportant (and unseen) at a given instant, leading us to *simulation LOD*. As simulation quality can be degraded to nothingness, this method can be used for environments of almost any size.

Degradation of simulation quality makes this approach inapplicable when the simulation is aimed to *compute* a result, for example for large social or economic simulations, e.g. [12]. In such a case, the simulation typically must run distributively on more PCs instead. Fortunately, in the domain of v-humans, quality of the simulation out of the scene is not the issue, *provided a user at the scene experiences the right thing* [11]. Hence, the question behind simulation LOD is how to save resources by degrading quality, but keeping the *illusion* of quality, i.e. *believability*.

We have addressed this question as a part of our effort at a large educational storytelling game populated by tens of complex virtual humans [7] by:

1. Proposing an abstract framework for simulation LOD. We intended this framework to be implemented in our storytelling application, and also possibly scaled for other applications. It had to allow for simplifying space (hence, *space LOD*) and behaviour of v-humans (*LOD AI*) at the unimportant places. We demanded varying the detail of the simulation at all places *gradually*, i.e. we required several levels of detail, which has two advantages discussed later.
2. Implementing a prototype.
3. Implementing the technique in our storytelling application.

As our scope is games, the technique is aimed at RPG-like *large* environments, i.e. worlds including tens of different areas, e.g. a region with villages, fields, and meadows. These worlds are inhabited by tens of *complex* v-humans, i.e. v-humans performing in real time several, at least two, complex tasks that are possibly conflicting (adopted according to [9]). Typically, these tasks last minutes, include manipulation with several objects, and apparently require human-level cognitive abilities. An example of such a task is harvesting or watering the garden (as opposed to obstacle avoidance or crowding). The framework is further required to be universal in the sense that it can handle different complex tasks provided as data.

We stress the scope, since recently, applications featuring different kinds of large environments have appeared—e.g. towns where all the actors "only" walk along the streets avoiding each other and not entering the houses (not that these are simple!). Perhaps, these applications can also benefit from a simulation LOD, but one concerning steering, crowding etc. Our LOD is focused on complex cognitive tasks instead.

Recently, we have completed Point 2, i.e. prototyped the simulation LOD in our toolkit for v-humans investigation called IVE [8], and our on-going work is concerned with Point 3. In the course of prototyping, we have focused on the gist of the issue, i.e. on the simulation LOD itself, and abstracted from "other v-humans issues" we would have faced when developing a full application. This, above all, means that we have intentionally avoided 3D implementation. The LOD model is implemented as a sort of abstract machine with a grid-world environment (but still with embodied v-humans). To use a 3D world would only have distracted us from the problems that are simulation LOD relevant to the problems that are 3D world relevant, e.g. to the obstacle avoidance problem, the visibility problem etc. These issues are not of much relevance when degrading simulation detail, at least in typical cases. Similarly, our v-humans have presently only limited emotions and no social relationship to the others. Scaling the technique to a 3D world and equipping our agents with relationships and emotions, using our emotional model developed in parallel [4], is a part of our on-going work on Point 3.

Goal of the Paper. During prototyping, we have stumbled on several inherent simulation LOD issues stemming from the gradualness requirement, some of which we have solved. They are mostly representational problems (both space and behavioural), and issues concerning retaining information when simulation is degraded to be able to reconstruct (to some extent) the full simulation when the detail elevates. We believe that the most important contribution of our work is not the set of particular methods we have developed, but *uncovering* of the very issues.

In this paper, our aim is to provide the reader with insight into the problematic, helping the reader to adopt the gradual simulation LOD technique for his or her application if needed. Hence, we present our technique as a general framework for thinking about the topic rather than a set of formal algorithms addressing particular issues. However, some of our solutions are also presented.

Section 2 introduces simulation LOD in general. Simulation LOD is contrasted with graphical LOD, various simulation LODs that have been already proposed in literature are discussed, an example scenario on *gradual* simulation LOD introduced, and issues of this technique are detailed. Section 3 is devoted to explanation of our framework. In Section 4, our prototype is introduced. Section 5 discusses strengths and weaknesses of our technique, shedding light on its scalability and viability.

2 Simulation LOD Overview

When speaking about LOD techniques, it is good to start with the distinction between graphical and simulation LOD. Basically, while graphical LOD concerns the simplification of portrayal of what happens in the world model, simulation LOD concerns AI of v-humans and virtual world dynamic, i.e. what happens *per se*. This (simplified) distinction between graphical and simulation LOD is mirrored in the explanatory (and again simplified) “two-layer” metaphor regarding applications featuring v-humans as being comprised of: (1) a simulator of a world model and AI of v-humans, and (2) a GUI depicting the outcome of the simulator.

Basically, graphical LOD helps to reduce the detail of the scene out of the scope of the observer. This technique can be to great advantage applied not only to the walls and objects, but to v-humans bodies as well. For example, inspired by the authors of ALOHA system [15], we can speak about a geometrical, animation, and gesture and conversational LOD used for the bodies (apparently, we do not need to care much about facial expression and non-verbal gestures for background characters, saving the resources for the foreground characters). Even gesture and conversation LOD, however, deals only with the situation, when the v-humans are *at* the scene. Once we are aiming at a simulation of a large world with tens of complex v-humans, we will surely have v-humans that are also *out of* the frustum at some particular moments. At this point, simulation LOD comes into play (Fig. 1). The purpose of this branch of techniques is to further capitalise from simplifying the simulation out of frustum.[1]

In the rest of this section, we aim at providing the reader with a general intuition behind simulation LOD. We detail several relatively simple simulation LOD approaches, presenting a ground for thinking about limitations and possible improve-ments, and then, we explain requirements on our *gradual* method using an example scenario.

[1] It may be argued that simulation LOD may also concern what happens at the scene. E.g., the mentioned gesture and conversation LOD may be regarded as an “at the border” technique. See [14] for another example. Here, we will disregard this terminological issue for brevity.

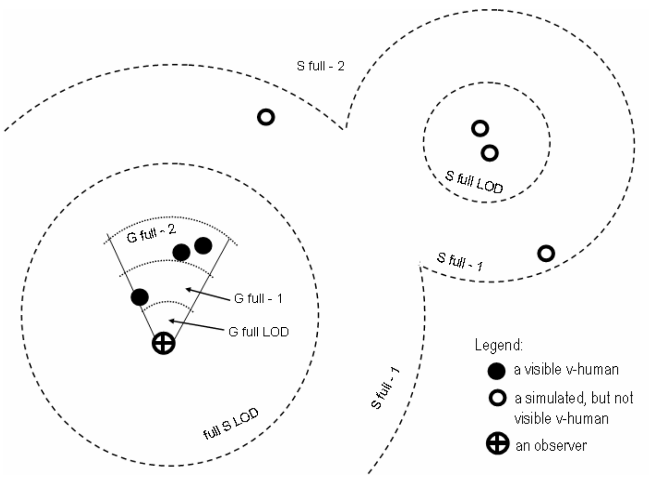


Fig. 1. The relation between graphical levels of detail (G LOD) and simulation levels of detail (S LOD) is depicted from the bird-eye view. "Full" denotes the full level of detail, "full-1" denotes the detail full minus 1 etc. As it can be seen, a place out of the frustum simulated in the full simulation detail can exist. This place may be important e.g. from the plot point of view.

Related Work. Computer games with large environments typically use a simulation LOD in addition to a graphical LOD, however, only to a limited extent. Either an *ad hoc* LOD is used, or, more often, the places in the frustum and next to it are simulated in the full detailed, while the places out of the sight not at all, e.g. [1, 13] (Fig. 2_1). Though this "all—nothing" method may fit for some games, it causes scenic inconsistencies in games emphasising a story, e.g. when a user expect a v-human to return to the scene. Moreover, this approach allows for only one place simulated in "full" detail: the place observed by the user. However, once we have a story, a drama manager may need to determine another place of importance due to course of the story reasons. Finally, "all—nothing" approach typically brings a high overhead of changing the focus of attention, since the simulation at the new centre must start from scratch, as opposed to gradual approach, where something is already simulated.

We may imagine an approach allowing for more important places (Fig. 2_4), or gradual simulation around the scene (2_5), but what we are looking for is actually a combination of both (Fig. 1, 2_3). This will help us to diminish the story inconsistencies and to keep the overhead in control. Such an approach has been presented in [5], however, it covers only combat and interaction behaviour of creatures from a real-time strategy game. Similarly, a gradual simplification has been already introduced for traffic simulation [11]. As opposed to these, we are interested in a general technique focused on complex v-humans.

In this domain, the ALOHA system has presented a robust simulation LOD using a role-passing technique [15]. When a v-human is far from the observer, it performs only a basic behaviour (staying, walking etc.). When the user approaches, each influenced v-human overtakes a role describing a domain specific behaviour, e.g. a bar-patron role. Additionally, path-finding for v-humans out of the frustum is simplified. Though this approach clearly brings new possibilities, the issue of partial behaviour is challenged only to limited extent. It likely would not be a problem to overcome the "see—not see" aspect of this method, but it is not clear how to scale the method to cope with performing a role only *partially*. Basically, ALOHA allows just for three simulation details: "full" (a v-human has taken a role), "medium" (it performs only basic behaviour), and "nothing". Additionally, ALOHA is concerned with LOD AI, but we are interested also in simplifying virtual space.

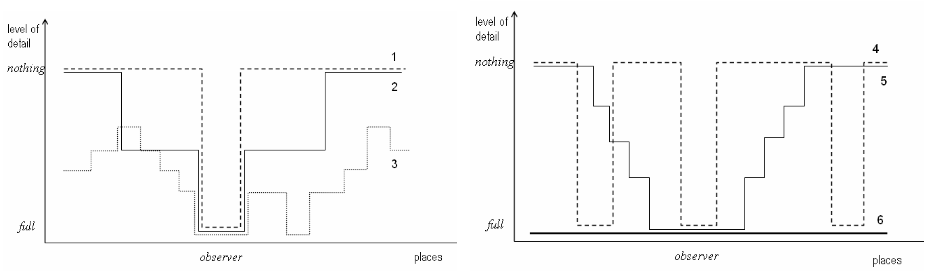


Fig. 2. Illustration of different simulation LOD approaches. 1) "see—not see" + "all—nothing", 2) ALOHA LOD, 3) gradual LOD with more partially important places, 4) "all—nothing" with more fully important places, 5) "see—not see" gradual, 6) full simulation.

Gradual Simulation LOD Example. To allow for comparison with ALOHA [15], we now demonstrate on a pub-scenario how we intended our gradual LOD to work. (We have implemented this scenario in our prototype with some minor distinctions).

Imagine a pub with a waiter and miners enjoying by drinking, chatting, watching TV, dancing etc. We will concentrate on "drinking at a table" behaviour (Fig. 3). If the pub is simulated in the detail "full", each miner should sit at a table, chat with his colleagues, sip of his beer from time to time, ask the waiter for a new beer when the old one is emptied, go to the toilet when nature calls, etc.

Assume that when the user leaves the pub and starts roaming around it, the detail in the pub is decreased to "full-1". What should happen? The miners should stop chatting and sipping of beer, instead, each should empty his glass at once every half an hour or so. After that, he should order a new beer, and the waiter should give it to him instantly, which means that he should not walk to the table, but move from the pipe to the miner in a one "giant step", and swap the glasses at once. Similarly, when nature calls, the miner in question should move to the toilet instantly.

Assume that when the user is in the farther part of the village, the detail in the pub is decreased to "full-2". At this detail, every room in the pub should be abstracted to a single point, and the glasses should cease to exist (i.e. simplifying space). However, the miners should still stay at their tables, and the waiter should decrease a bit beer level in the barrel every hour or so (instead of serving beer). Additionally, each miner still should be allowed to decide to change his activity—e.g. to start to play billiard, or to go to the toilet. At "full-3" detail, the whole pub should become a single point, and the tables and the chairs should cease to exist. Still, the miners and the waiter should stay in the pub and, of course, they should be allowed to go home, or a new person should be allowed to enter the pub, each using "giant steps" when moving.

Notice three important things. First, as the miners' behaviour is simplified, the processor load should decrease (which is the case, see Sec. 4). Second, the result of the simulation in "full" detail may differ from the simulation in a partial detail. However, third, believability is retained to large extent: when the user returns to the pub, he or she will find the miners in a consistent situation (more or less). Consider two scenarios. 1) If the user roams at the second part of the village, the detail is only "full-2", which means that when the user returns, the miners will still sit behind the

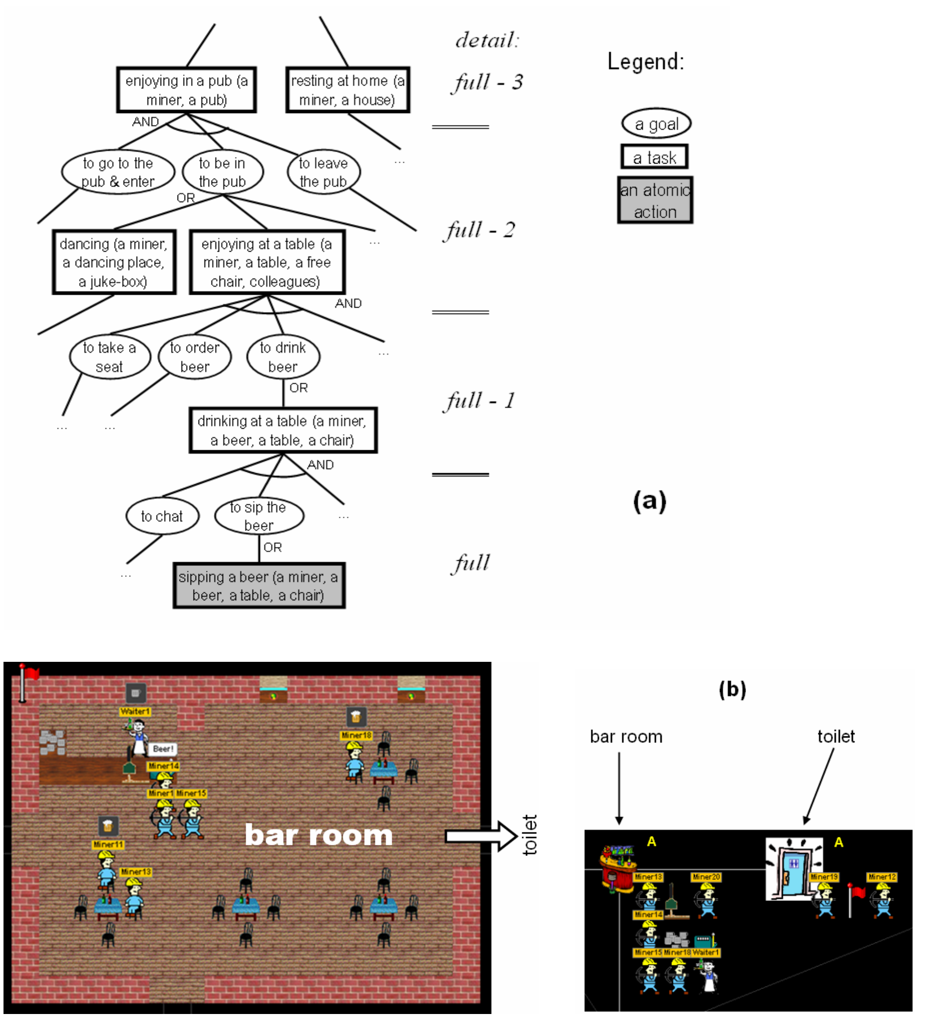

detail:
full - 3
full - 2
full - 1
full
Legend:
a goal
a task
an atomic action
enjoying in a pub (a miner, a pub)
resting at home (a miner, a house)
AND
to go to the pub & enter
to be in the pub
to leave the pub
OR
dancing (a miner, a dancing place, a juke-box)
enjoying at a table (a miner, a table, a free chair, colleagues)
AND
to take a seat
to order beer
to drink beer
OR
drinking at a table (a miner, a beer, a table, a chair)
AND
to chat
to sip the beer
OR
sipping a beer (a miner, a beer, a table, a chair)
(a)
bar room
toilet
(b)
bar room
toilet
A
A

Fig. 3. (a) A part of pub behaviour. Notice the AND-OR tree structure of behaviour. (b) The pub scenario in IVE. In this particular scenario, there is no user avatar, but the user can set the LOD using a "red flag". Left: the bar room, "full" detail. Right: the same situation depicted, but the detail is lowered: both the bar room and the toilet are abstracted to a single point (note the **A** mark).

same tables as before (or they will perform another activity, but the same as they would perform in the case of the full simulation during the user's absence). 2) If the user leaves the village at all ("full-3"), the sitting positions are not remembered and must be generated randomly: but sooner than s/he enters the pub again; actually, when s/he enters the village. Because of this, it is guaranteed that the simulation will have been running for a while with the new positions before s/he enters the pub. Moreover, the user will still find in the pub some of the old guys, who have not gone home.

Gradual Simulation LOD Problem Detailed. The sketched gradual LOD raises several questions, among which the most striking are:

1. How many levels of detail shall we have and how to simplify the space?
2. If we simplify a location, what shall we do with the objects in there?
3. How to vary behavioural complexity of v-humans?
4. How to assign importance to the places and what to do when the importance is being changed?
5. How to retain information to be able to reconstruct the full simulation?

These issues must be addressed when one aims at a gradual simulation LOD.

3 Proposed Simulation LOD Framework

This section introduces our framework, being structured according to Question (1) – (5). We remark that following our "GUI—simulator" explanatory metaphor for applications featuring v-humans, what we introduce here can be seen as a proposal of an abstract world simulator that presents a lower layer for a 3D GUI.

Questions (1): Space Representation. In our framework, space is represented hierarchically, each layer presenting an additional level of a logically coherent space abstraction. The layers typically mirror the following line: places/parts of rooms/ rooms/houses/suburbs/villages etc. Each layer also presents one possible simulation detail. A bar room from the example above is an area from the layer "full-2", while the pub is from "full-3". The number of details depends on the complexity and size of the world. Our example scenario has five levels, with four villages at the top.

We call the leaves of the hierarchy, i.e. the discrete ground constituting a world, *way-places* (or *tiles*). A non-leaf node is called an *area*, and the root represents the whole world. In addition to the tree structure, each layer is organised in a graph, whose edges afford passage to a neighbouring way-place/area (Fig. 4a). Upon this structure, hierarchical A* can be used, and a pseudo-continuous 3D world developed.

To describe how to simplify space we use a *membrane metaphor*—imagine an elastic membrane cutting through the hierarchy of way-places and areas (Fig. 4b) that can be reshaped in every time step. Each area or place that is at the membrane at a particular instant is simulated as an abstract point, and nothing "below" it exists. This means that the whole world is simulated in "full" detail when the membrane crosses just way-places. If it crosses an area, the detail is decreased there.

For the purposes of coherence, there is need to settle the following *shaping rules*:

- If an area or a way-place x is at the membrane, every neighbouring area or way-place with the same parent as x is at the membrane too.
- If an area y is above the membrane, every neighbouring area with the same parent as y is above or at least at the membrane too.

For example, if a way-place from the bar room is at the membrane, all its way-places will be at the membrane too. This ensures that when the simulation runs in "full" detail somewhere in the bar room, it will run in "full" detail everywhere here. As the bar room itself is above the membrane in this case, all rooms from the pub must be simulated at least as abstract points (because the pub is their common parent). Similarly, if there is a suburb that includes this pub and near houses, all the houses must be simulated at least as abstract points. This ensures that at least something happens in the areas near to the centre of attention, which helps with the issues of story consistency and overhead during reshaping, as discussed later.

Question (2): Objects. *Objects* are represented as abstract entities (which cannot be decomposed to sub-objects presently). Bodies of v-humans are represented as objects. Each v-human is driven by its autonomous action selection mechanism (ASM). The problem with the objects is: what shall we do with them when the detail is low?

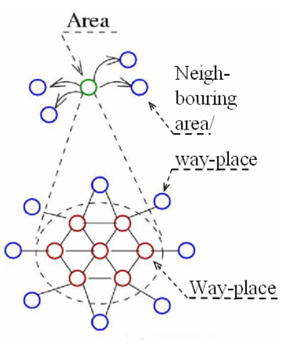
In our framework, if an area is fully simulated, the object is located at a way-place or at/on another object, e.g. at the table. If the detail is decreased, the object can either cease to exist, or it can be positioned at the area (or at/on another object that exists) (Fig. 4b). In the example from Sec. 2, with the detail "full-2", the chairs are placed at the tables; the tables, the barrel, and the actors are in the pub; and the glasses do not exist. To settle which of these possibilities apply for a particular object, we assign an *existence level* and a *view level* to every object. If the detail decreases below existence level, the object ceases to exist, otherwise, it is simulated. *View level* then determines detail required by the object after it starts to be simulated (hence, it is always equal to or higher than existence level). For example, if a waiter has "full-1" view level and "full-3" existence level, it will not exist if the detail is lower then "full-3", but if the detail elevates to "full-3", it must be further increased to "full-1" (which includes increasing the detail in the neighbouring areas according to the shaping rules). The algorithm for determining exact detail after an increase is detailed in [18].

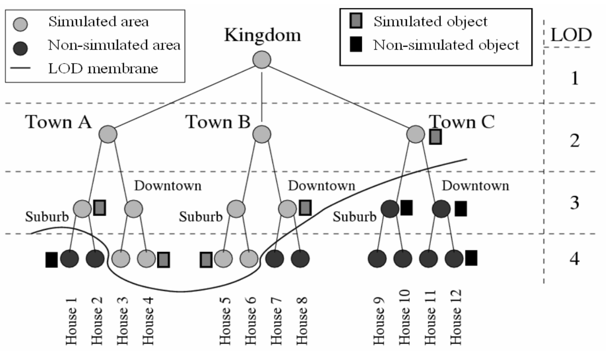
Though this seems simple, three problems rise: A) how to remember detailed positions of objects being removed? B) how to place the objects during elevating the detail? C) what to do when an object is moving between areas with different details?

We have adopted following solutions (the description is simplified for brevity). A) Every object existing at a low detail remembers its own positions at all possible higher details. Concerning the non-simulated objects, we distinguish between *area-native* and *area-foreign* objects (a glass or a table is area-native in a pub, as opposed to a watering can). Area-native objects are destroyed in the sense of object oriented programming and only their total number is remembered by the area. The position and state information of area-foreign objects are stored by the area.

B) In this situation, we have an area being expanded. Hence, we must find exact positions for (a) already simulated objects that remembers their detailed position, (b) already simulated objects that does not remember their detailed position (for they came into the area when the detail was low), (c) presently not-simulated area-foreign objects that have to become simulated, (d) like (c) but area-native. The objects from categories (a, c) are placed according to their remembered positions, while (b, d) using a "room-specific" placing algorithm implemented for each area. Generally, this algorithm uses the strategy of generating objects randomly around fixed places (e.g. glasses somewhere at the bar). At the extreme, objects might be placed to exact preset positions (e.g. a table that never move to the way-place <12, 14>). Note, that for some objects of the category (d), it may be beneficial to remember their positions as well (e.g. for tables that can be moved, which are however moved only rarely). This last point is however unimplemented presently.

C) When an object is moving between areas with different details, following three cases may occur. (a) If the target area has sufficient detail – higher than object's *view level* – the object is just placed at the area. (b) If the area detail is too low – lower than object's *existence level* – the object ceases to exist. Finally (c), if the area detail is between object's *existence level* and *view level*, the detail in the target area is increased. The area is thus expanded to subareas and the object is placed into one of them. The exact algorithm is detailed in [6].





(a) (b)

Fig. 4a (left). A part of a space hierarchy. The area comprises a set of way-places. Both the area and the way-places point to neighbouring areas or way-places respectively. **4b (right).** Space representation with the LOD membrane, and several simulated and non-simulated objects.

Question (3): Representation of Behaviour and ASM. Perhaps the most important issue is varying the complexity of behaviour. We need a hierarchical representation, with additional layers representing additional levels of behavioural complexity. We have adopted AND-OR trees as the representational concept, and reactive planning as the action selection mechanism (which is a relatively typical present day solution).

Basically, the AND-OR tree metaphor works with abstract goals representing *what* shall be achieved, and tasks representing *how* to achieve it. Typically, every goal can be accomplished by several tasks, while every task can be achieved by adopting some sub-goals. A v-human needs to perform only one task to achieve a goal (provided there is no failure), but to fulfil all sub-goals to solve a task. Hence, goals are represented as OR nodes, while tasks as AND nodes (Fig. 3a). The tasks that cannot be further decomposed are *atomic actions*, i.e. primitives changing the world-state.

In a typical simulation employing AND-OR trees, the ASM adopts a top-level goal, and recursively finds an atomic action for it. This is performed reactively in every time step, assuring that important external and internal events are taken into consideration. In our model, this is performed as well, but only when the simulation runs in "full" detail. If the detail is lowered, the ASM does not find sub-goals below a particular layer of the AND-OR tree (dependent on the detail), and instead, it executes the task at the lowest visited layer as it is atomic (Fig. 5a). E.g., when "drinking at a table" is executed atomically, the miner will empty the glass after half an hour instantly, and its bladder need will increase.

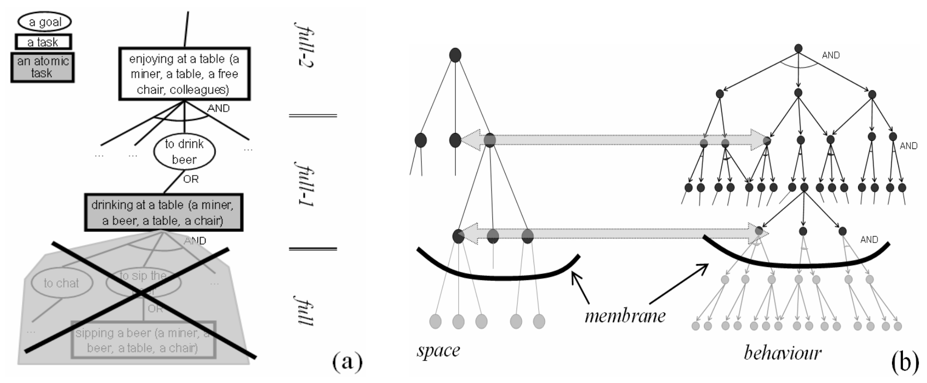

a goal
a task
an atomic task
enjoying at a table (a miner, a table, a free chair, colleagues)
AND
to drink beer
OR
drinking at a table (a miner, a beer, a table, a chair)
AND
to chat
to sip the beer
OR
sipping a beer (a miner, a beer, a table, a chair)
full-2
full-1
full
(a)
AND
AND
AND
membrane
space
behaviour
(b)

Fig. 5a. Performing a task atomically. **5b.** Layers of space hierarchy correspond with layers of behavioural representation (in reality, the correspondence can be a little more complicated).

How to specify which task shall be atomic at which detail? This can be settled simply by coupling the layers from the AND-OR tree with the layers of the hierarchy of areas and way-places (Fig. 5b). This projection can be read as "ability of the areas to afford what can be done in there" (a pub affords "enjoying in the pub", a bar room affords "enjoying at a table" etc.). If an area is not simulated, the respective tasks are not afforded. In our example, if the pub is shrunk to a point, which is a detail "full-3", "enjoying at a table" will not be possible to execute, instead, atomic execution of "enjoying in a pub" will be allowed. All what is needed is to carefully design existence level of every object so that the object is simulated when a task it is involved in can be performed. Note, however, that this is not a simple task.

Questions (4): Reshaping. We now turn our attention to changing detail. Basically, the importance of places originates in movement of important objects, which reshape the membrane. The mechanism for annotating important objects is view level. An example of an object always having "full" view level is the user avatar. Additionally, the drama manager is supposed to vary the detail (this is currently imitated manually). When detail in a particular area is elevated, objects must be placed and already executed atomic tasks must be "broken down" as described below. When detail decreases, we could adopt either of two mechanisms. First, we might immediately shrink the area, relocate objects in there or stop simulating them (depending on their existence level), interrupt currently running atomic tasks/actions and start the tasks from the higher layer of the AND-OR tree atomically. Second, we might adopt a "garbage collector" approach, which means to perform exactly the same, but not until processor resources are need. We have employed the latter mechanism for it helps us to keep down overhead in the case of a user roaming at the border of two areas.

From the implementation point of view, a discrete simulation paradigm was used. When an atomic action or an atomic task is to be started, its *end time* is estimated and hooked to a calendar as an event (e.g. end of "drinking at a table" is hooked at about 30 minutes from start of the task, Fig. 6). When the event is invoked, the result of the action/task is committed. If a conflict between two actions/tasks occurs, it must be handled in the time of occurrence, which may lead to unhooking the previously hooked ends of these two actions/tasks, and to hooking the end of a new action/task.

Consider now a task T that is executed atomically, but the simulation detail is increased before its end time (the end time is denoted as (2) and the time of expansion as (1) in Fig. 6). Two things now happen. First, T is *partially evaluated.* Second, simulation is started at the higher level of complexity, which means a sub-goal G for T is chosen, and a sub-task/action for G found and its end hooked ((3) in Fig. 6). Since T is likely half-finished after the partial evaluation, this sub-goal must be found carefully, otherwise believability may sustain a loss (e.g. chatting of the miners at a table should start in the middle). The case of decreasing the detail is handled similarly.

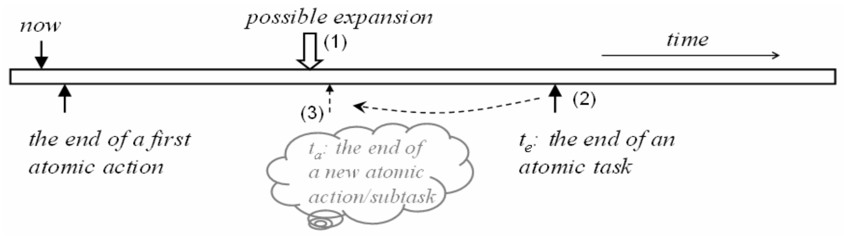

now
possible expansion
(1)
time
(3)
(2)
the end of a first atomic action
t_a: the end of a new atomic action/subtask
t_e: the end of an atomic task

Fig. 6. Every end of an action or a task being performed is hooked in the calendar. Let an end of the task T be hooked at t_e. If detail elevates at moment (1), this event (i.e. end of "T") must be unhooked (2), T must be partially committed, its subgoals must be found, one of them chosen, and an atomic action or a task for the chosen subgoal must be found. The end of this newly found action/task is them hooked at time t_a (3).

Question (5): Information Retention. Perhaps the most important problem of simulation LOD is how to keep information about the situation in an area for the period of the simulation being stopped, or how to keep illusion that the situation is evolving. Our framework diminishes this issue, since it allows for partial simulation. Though in a coarse-grained way, the original situation can still be evolved, and some of the objects are still placed or their positions are remembered. When we decrease the detail in the pub from "full" to "full-1", miners will continue what they were doing before the decrease, skipping only the atomic actions. Some detail may be lost, but this may not matter as the user is far away. We remark that this "retain of information" is very problematic to achieve using "all—nothing" approach.

4 Prototype Implementation

We have implemented a simulation LOD prototype scenario using our Java toolkit for v-humans ASMs investigation called IVE [8] (Fig. 3b), which is also being used for the storytelling experiments presently [7]. IVE can be downloaded freely.

The scenario comprises of about 100 v-humans (miners and waiters) acting in four villages; each with a pub, five mines, and nine houses. Five levels of detail/layers of the space hierarchy have been used (a place, a room, a house or a mine, a village, the world). Way-places are square tiles. For experimental purposes, the detail is currently elevated only on the supervisor's demands and in accord with view levels of objects.

Several experiments measuring the speed and memory requirements have been carried out [6]. The results have not only revealed that the simulation runs faster at lower details, but also that when the world is simulated in "full" detail, where the v-humans are driven by about 5 000 reactive rules altogether, the simulation takes only about 5-10% of the maximum processor load at a 3GHz PC (the whole world at "full-1" detail takes about 0.1-0.5% with about 2 500 rules, and less then a third of all objects). The results of reshaping overhead measuring clearly showed that our approach keeps overhead more in control comparing to "all—nothing" approach, owing to the fact that it allows for allocating the overhead to several time steps (the detail from "nothing" to "full" can be increased through the intermediate details).

The increase of scripting-time of behaviour due to simulation LOD is about 50% (a rough estimate), which is acceptable. After we develop a 3D world, we plan to conduct an elaborate study of believability.

5 Discussion and Future Work

In this paper, we presented our simulation LOD technique as a general framework for thinking about how to save computational resources in a large application featuring complex virtual humans on the expense of decreasing quality of the simulation out of the frustum. The most remarkable feature of the technique is that it allows for *gradual* simplification of the simulation, which helps to keep reshaping overhead in control, and to avoid story inconsistencies. These issues are almost unsolvable when one commits herself to "all—nothing" approach.

The results we gained from our grid-world prototype allow us to conclude that our technique performs well to be scaled for our on-going storytelling game [7]. However, the question of scaling the technique to *other* applications remains to be answered, together with highlighting limits of the technique.

1. The technique relies on a tree-based space representation. This fits well for closed areas, i.e. rooms or corridors, but may bring troubles for open areas. We have an idea how to address this issue using fuzzy borders between areas, however, this is out of scope of this paper.
2. Representation of behaviour is based on AND-OR trees. We think that other strictly hierarchical representations, e.g. hierarchical state machines (HFSMs), can be used, provided the hierarchy has enough layers. Actually, a LOD AI for a first-person shooter agent that uses HFSMs has been already sketched in [10] (but to our knowledge, not further explored). However, combining our technique with heterarchies, neural networks or free-flow hierarchies, e.g. [16], may be problematic.
3. The technique is focused on general behaviour. For some specific behaviours, e.g. for conversation behaviour, a "domain-specific" simulation LOD may be vital. We have also not employed emotions and social relationships; exploring how these affect simulation LOD is a part of our on-going work. We plan to use our emotional model develop previously [4].
4. Our prototype uses a grid world, but not a 3D world. This was certainly beneficial for prototyping purposes, but surely, we cannot live with a grid world for ever. This scaling may bring new problems, e.g. visibility ones, which remains to be tackled.

Some advanced “optimisation” issues inherent to our technique also exist. For example, a procedure for atomic execution of each task must be written manually. It is intriguing to think about whether it would be possible to generate these scripts automatically based on the results of many simulations performed in the full detail. Apparently, this presents a tough problem, currently not addressed. However, some advanced issues related to better membrane shaping are under investigation [6].

To conclude, we think that the technique is mature enough to be scaled to any application featuring a world complex in size and behavioural possibilities, provided the areas are (more or less) close, and behaviour can be represented in a strictly hierarchical fashion. Some particular issues may arise during scaling to 3D, and during equipping virtual humans with emotions and relationships, but we think these can be overcome relatively easily.

Acknowledgments. This work was partially supported by the Ministry of Education of the Czech Republic (grant MSM0021620838), and by the Program "Information Society" under project 1ET100300517, and GA UK 351/2006/A-INF/MFF. Authors would like to thank to Brian Mac Namee for his comments.

References

1. Adzima, J.: AI Madness: Using AI to Bring Open-City Racing to Life. In: Gamasutra Online (January 24, 2001) [6.3.2007] (2001)
2. Alelo Inc.: Tactical Iraqi: a learning program for Iraqi Arabic [18.11.2006] (2005), http://www.tacticallanguage.com
3. Aylett, R.S., Louchart, S., Dias, J., Paiva, A., Vala, M.: FearNot! – An Experiment in Emergent Narrative. In: Panayiotopoulos, T., Gratch, J., Aylett, R., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, pp. 305–316. Springer, Heidelberg (2005)
4. Bída, M.: Emotion bots in Unreal Tournament (in Czech) Bachelor thesis. Faculty of Mathematics-Physics. Charles University in Prague (2006)
5. Brockington, M.: Level-Of-Detail AI for a Large Role-Playing Game. In: AI Game Programming Wisdom I, pp. 419–425. Charles River Media, Inc, Hingham, Mas (2002)
6. Brom, C.: Action Selection for Virtual Humans in Large Environments (in Czech) PhD thesis. Faculty of Mathematics-Physics. Charles University in Prague (2007)
7. Brom, C., Abonyi, A.: Petri-Nets for Game Plot. In: Proceedings of AISB Artificial Intelligence and Simulation Behaviour Convention, Bristol, vol. 3, pp. 6–13 (2006)
8. Brom, C., Lukavský, J., Šerý, O., Poch, T., Šafrata, P.: Affordances and level-of-detail AI for virtual humans. In: Proceedings of Game Set and Match 2, The Netherlands, Delft (2006), http://urtax.ms.mff.cuni.cz/ive/public/about.php [6-3-2007]
9. Bryson, J.J.: Intelligence by Design: Principles of Modularity and Coordination for Engineering Complex Adaptive Agents. PhD thesis, Mas. Institute of Technology (2001)
10. Champandard, A.J.: AI Game Development: Synthetic Creatures with learning and Reactive Behaviors. New Riders, USA (2003)
11. Chenney, S.: Simulation Level-Of-Detail. In: GDC (2001), http://www.cs.wisc.edu/~schenney/research/culling/chenney-gdc2001.pdf [6-3-2007]
12. Gilbert, N., den Besten, M., et al.: Emerging Artificial Societies Through Learning. The Journal of Artificial Societies and Social Simulation, JASSS, 9(2) (2006)

13. Grinke, S.: Minimizing Agent Processing in “Conflict: Desert Strom”. In: AI Game Programming Wisdom II, pp. 373–378. Charles River Media, Inc, Hingham, Mas (2004)
14. Jan, D., Traum, D.R.: Dialog Simulation for Background Characters. In: Panayiotopoulos, T., Gratch, J., Aylett, R., Ballin, D., Olivier, P., Rist, T. (eds.) IVA 2005. LNCS (LNAI), vol. 3661, Springer, Heidelberg (2005)
15. McNamee, B., Dobbyn, S., Cunningham, P., O´Sullivan, C.: Men Behaving Appropriately: Integrating the Role Passing Technique into the ALOHA system. In: Proceedings of the Animating Expressive Characters for Social Interactions (2002)
16. de Sevin, E., Thalmann, D.: A motivational Model of Action Selection for Virtual Humans. In: Computer Graphics International, IEEE Computer Society Press, New York (2005)
17. Shao, W., Terzopoulos, D.: Environmental modeling for autonomous virtual pedestrians. In: Proceedings of 2005 SAE Symposium on Digital Human Modeling for Design and Engineering, Iowa City, Iowa (June 2005)
18. Šerý, O., Poch, T., Šafrata, P., Brom, C.: Level-Of-Detail in Behaviour of Virtual Humans. In: Wiedermann, J., Tel, G., Pokorný, J., Bieliková, M., Štuller, J. (eds.) SOFSEM 2006. LNCS, vol. 3831, pp. 565–574. Springer, Heidelberg (2006)