What Does Your Actor Remember? Towards Characters with a Full Episodic Memory

Cyril Brom[1], Klára Pešková[1], and Jiří Lukavský[2]

[1] Charles University, Faculty of Mathematics and Physics, Prague, Czech Republic
[2] Institute of Psychology, Academy of Sciences, Prague, Czech Republic
brom@ksvi.mff.cuni.cz

Abstract. A typical present-day virtual actor is able to store episodes in an *ad hoc* manner, which does not allow for reconstructing the actor's personal stories. This paper proposes a virtual RPG actor with a *full* episodic memory, which allows for this reconstruction. The paper presents the memory architecture, overviews the prototype implementation, presents a benchmark for the efficiency of the memory measurement, and details the conducted tests.

1 Introduction

Computer role-playing games (RPGs) typically feature virtual worlds large in size, lifetime and behavioural possibilities. These worlds, both in off-line and on-line RPGs, are usually inhabited by tens of virtual actors ("non-player characters", NPCs). An innate challenge of these games is to provide a player with a believable story. Though there is likely more narration in RPGs then in any other kind of commercial games, storytelling in RPGs is still far from novels or films.

The story generating in RPGs is a multi-dimensional problem—one cannot tackle it in a Façade-like style [11] for example, as the world is much more complex. One facet of this problem is unfolding the story and keeping it consistent when a player speaks with NPCs, i.e. letting the actors to tell believable stories about themselves. Imagine that while you are playing an RPG, you come to a medieval village, find an NPC shaman in her den and ask her (selecting a question from a template list):

> *Hey, I am the representative of the king Hromburac Pekelny, and you please tell me, what were you doing during the last week? And, please, summarise it in two sentences.*
>
> *Well* – she answers – *I was here every day, meditating and heeling toes, you know. Nothing interesting happened, except of Wednesday when filthy brigands came to rob me.*

To be able to answer such questions according to their personal histories, and not using a text written in advance by a designer, the NPCs must be simulated even if no player is around (at least to some extent), and they must be equipped with a *full episodic memory* and a *linguistic module* transferring the outcome of the memory to syntactically correct sentences. By "full", we mean a generic memory that stores more or less everything happening around the actor tagged with the actor's own relevance estimation, as opposed to an *ad hoc episodic memory* storing only the events specified in a hardwired fashion inside the actor's script or reactive plan (this kind of episodic memory is almost always present in current intelligent actors).

M. Cavazza and S. Donikian (Eds.): ICVS 2007, LNCS 4871, pp. 89–101, 2007.

A full episodic memory shall even allow the player to ask further, for example:

Ok, that sounds interesting. *Please, summarise the same now in 15 sentences.* or *Please, focus on the filthy brigands.* or *Tell me more about Saturday evening.*

We have been developing actors with this kind of episodic memory as a part of our on-going work on a large educational storytelling game [1]. The issue of episodic memory agents in general has been recently tackled in literature, e.g. [14]. However, we are interested in a *full* episodic memory that is *special-purpose* in the context of RPGs emphasising a story—and this issue has not been addressed sufficiently yet.

Two aspects of the issue must be stressed. First, it must be assumed that the virtual world lasts for days or weeks and features tens of different actors (this contrasts with many applications of intelligent actors). Consequently, the highest attention must be paid to the memory size, to storage/retrieval speeds, and to the balance between them, as they present a sort of trade-off. Second, because of the novelty of this research field, an issue how to measure efficiency of the memory arises. Standards for comparable testing, i.e. benchmarks, have not been defined yet.

The goal of this paper is to present a new model of a full episodic memory for an NPC. To implement an entire actor presents a several man-years effort. Presently, about a half is done (especially, the linguistic module has not been addressed yet); however, we propose the whole architecture to facilitate thinking about the complete picture. A subgoal is to present a benchmark for measuring efficiency of the memory, detailing results of tests concerning size of the memory. Other tests are detailed in [5].

Section 2 analyses requirements of the memory, and Section 3 details related work. Section 4 presents our architecture contrasting what has already been done with what remains. Section 5 presents benchmark scenario, and discusses its construction. Section 6 describes the prototype implementation, reveals the experimental results and discusses them, and also introduces future work. Section 7 concludes.

2 Problem Analysis and Requirements

From the psychological point of view, *episodic memory* [17] represents personal history of an entity. Episodic memories are related to particular places and moments, and are connected to subjective feelings and current goals. Some psychological studies focus on so called *autobiographic memory*, which is similar to episodic memory but usually used in larger, i.e. lifetime, scope.

As our goal is to *imitate* human-like episodic memory, we need the actors to remember only what real humans would remember, and forget in similar way and extend as real humans would do. Unfortunately, there is no thorough functional description of human episodic memory from which we could derive our model. Thus, we are forced to derive the requirements only from case-studies of forensic psychology (e.g. [13]) and from our own phenomenological experience:

1. The memory should cope with complex tasks that include manipulation with several objects and apparently require human-level cognitive abilities, like cooking or merchandising, for they can be performed by an NPC. These tasks typically have hierarchical nature—they can be logically decomposed to sub-tasks, which can be divided to yet smaller tasks, until some atomic actions are reached. This point is fully implemented by the presented work, as detailed in Sec. 4.

2. The memory has to store and reconstruct personal situations: a) what an actor performed with which objects and why, and b) who was seen by the actor and what did the other actor perform (we remark, that presently only (a) is implemented, (b) is a work-in-progress). Time information, at least approximate, shall be also stored. The memory should be able to provide information like "where is something?", "when did the actor see *x*?", "what did the actor do from *y* to *z*?", "why did you do *a*?" and reconstruct stories like the abovementioned. The memory is expected to reconstruct information *on demand*, i.e. when a user asks, rather than automatically based on an environmental clue. This point is implemented, though the actors' answers are grammatically poor, see Sec. 6 for an example.
3. The memory should not store all available information; either external, or internal. In particular (a) neither all objects located around the actor, nor all activities of other actors should be stored. The objects used by the actor should be stored more often then the objects not used, but only seen. (b) Only important actions of other actors should be stored. (c) Generally, the way of choosing activities/objects to remember should be based on their relevance to the actor, on their attractiveness, and on the actor's attentional and emotional state. Presently, storing of actions of other actors is not implemented. Point (a) is achieved concerning the objects. Concerning Point (c), attractiveness (or saliency) of the objects and attention are implemented (Sec. 4). The emotional module has been implemented separately.
4. The memory operates in large time scale. As time passes, the unimportant details should be forgotten, until only a "gist" of what happened is kept. Different episodes should be forgotten in different rates based on their importance and emotional relevance. Several similar episodes can be eventually merged together. Forgetting is partially implemented, but not the merging of episodes (Sec. 4, p. 7).
5. Coherence shall be maintained, in particular if there are two contradictory records in the memory, i.e. an object *x* has been seen both at *a* and *b*, one of them must be marked as more trustful. This issue is a work-in-progress described in [4].

3 Related Work

In classical agent research, the issue of generic episodic memory is almost untouched, since the agents typically do not need to store more than a few episodes in an *ad hoc* memory for action selection purposes. This is also the case of most intelligent actors, though exceptions exist—see bellow. In robotics, Dodd [3] has recently developed a general memory system for a humanoid robot ISAC, which included also episodic memory working with emotions. An a-life example is the work of Ho et al. [7], who developed agents with various episodic memories aiming to investigate how different types of memories improve survival of the agents. Though these memories fit well for their domain, they are relatively low-level from the point of view of human-like actors, especially those from RPGs. For example, they are not designed to cope with complex, hierarchical tasks. This is also the limitation of most memories studied in the context of navigation in robotics. In the field of virtual actors, an implemented memory model was presented on ALOHA system [15]. It exploited to a great advantage division of the memory to short-term one and long-term one, but

unfortunately stored records only about objects and groups of objects, but not actions. Another example is Steve [16], who employs an episodic memory [9] to explain himself after a given lesson, which lasts, however, only a couple of minutes. FearNot! actors are also equipped with a episodic memory [8], which is, however, similarly to Steve, relatively short term one.

In agent research, perhaps the most elaborate model of episodic memory has been developed by Nuxoll [14]. This model is intended to be a robust, general-purpose architectural extension of Soar with broader scope than the model of ours. This, however, means that our model may benefit from some domain-specific tricks, which may finally increase efficiency of the memory in our domain. In all cases, it will be interesting in future to compare this model with ours using our benchmark scenario.

4 Proposed Full Episodic Memory

This section outlines the architecture of our actor, and details its episodic memory. The memory has two parts: a short-term (STM), and a long-term memory (LTM). We start with presenting the behavioural representation of our actor, proceeding through the STM to perception-action loop and to the LTM.

Our actor employs classical cognitive architecture (Fig. 1). The dashed parts are not implemented (linguistic module, "other memories"), or implemented separately but not included in the prototype described in this paper (emotional module, drives).

The goal structure. The actor is driven by hierarchical reactive planning with behaviour represented by AND-OR trees. In general, in the context of action selection, the AND-OR tree metaphor deals with abstract ***goals*** representing what should be achieved, and ***tasks*** representing how to achieve the goals. Typically, every goal can be accomplished by several tasks, while every task can be achieved by adopting some sub-goals. An actor needs to perform only one task to achieve a goal, provided there is no failure (hence, OR nodes), but he needs to fulfill all sub-goals to solve a task (hence, AND nodes—Fig. 2). Usually, these sub-goals are being adopted depending on some conditions or priorities, which allows both for sequential as well as reactive behavior. The tasks that cannot be further decomposed are ***atomic actions***, i.e. primitives changing world-state. In our case, every top-level goal has its ***activity level*** based on drives, external events, and a schedule. The most active top-level goal is called ***active***, several nearly "the most active" are ***preactive***. To be performed, every task may need several ***resources***, i.e. objects (e.g. hammering is possible only with a nail and a hammer).

The important tenet of our approach is that behaviour of all actors in the simulation is represented in this way. Violation of this principle, e.g. in the case of the user's avatar, whose behaviour is not represented in this sense at all, is discussed later.

The STM. The STM has four parts: the perceptional field, the memory field, the own-tasks field, and the others-tasks field. The ***perceptional field*** (PF) can entail several ***phantoms***, which are index-functional entities representing perceived externa objects, including their presumed state. The ***memory field*** is similar, but the phantoms in there refer to the LTM memory instead of the external world. The ***own-tasks field***

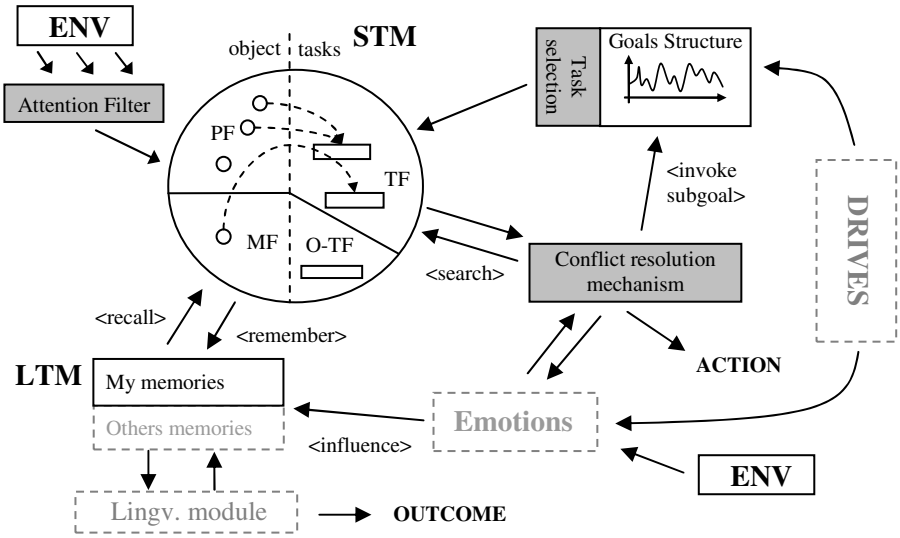


Fig. 1. The overall architecture of our actor. PF – phantoms of the STM. TF – own tasks of the actor. MF – records retrieved from the LTM. O-TF – others-tasks field. The dashed parts are not implemented, or they are implemented separately. See the text for further description.

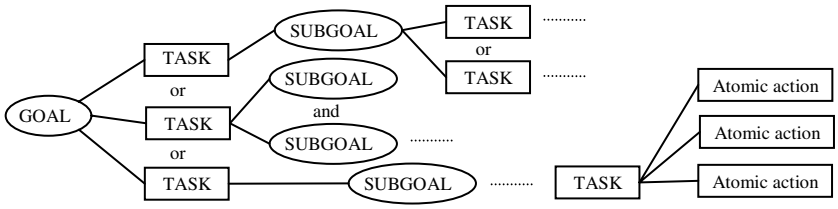


Fig. 2. Behavioural representation is based on AND-OR trees

(TF) entails tasks chosen to be performed or being performed (this structure is similar to so-called intentional stack from BDI architectures). Each phantom points at a parameter of a task *iff* it fits as a resource of the task. Each phantom can point at more tasks and more parameters of one task. The ***others-tasks*** field is not implemented yet. It is intended to represent tasks being seen as being performed by another actor.

The action selection mechanism (ASM). The action selection works as follows:

1. The activity level for each goal is calculated.
2. The task to be performed is chosen for the active goal (based on priorities) and passed to the TF. Sometimes, a task for a preactive goal is chosen as well.
3. The phantoms in the STM start to point to the added task.
4. If there is a task with a parameter at which no phantom points, a new *search task* on this type of resource is added to the TF.

5. The conflict resolution mechanism chooses a task to be performed from the tasks in the TF. If the task can be decomposed to a sequence of atomic actions (e.g. the search task), it is executed. If it can be decomposed only to subgoals, one of these subgoals is activated and the loop continues at step 1.

A search task works as follows: The actor looks around at the first place. If the required resource is not seen, the LTM is queried. If there is a record about the needed object, the most trustful record is put the memory field as a phantom, and the actor gets off to the presumed position of the object. When the object is seen, the phantom is moved to the PF. In case of no record in the LTM, a random search is initiated.

Perception. For the prototyping purposes, we have used a grid-world, where every object from the room the actor is located in can be seen (i.e. there are no obstacles). At every time step, just one object may pass through the attention filter and "become" a phantom in the PF. This is either an object looked for by a search task, or an object that has just captured the actor's attention because of its high ***saliency***. Saliency is computed from a preset base saliency and from current activity of both active and preactive goals (e.g. a preactive "eating" goal increase saliency of eatable objects). If a new phantom of an attractive object appears in the PF, all tasks that can be performed with this object are put to the TF to compete with the tasks placed there by active or preactive goals. (The whole mechanism is based on the capacity model of attention [**10**]. Its thorough description is, however, out of scope of this paper.)

This mechanism helps us to achieve several things. First, the amount of objects passing to the PF is limited (as in the case of a real human). Second, an external object can attract the actor's attention, and invoke a new task, or activate a preactive goal via its task in the TF (step 3, the ASM)—e.g. an actor having active "playing" goal and preactive "feeding" goal may stop playing and start feeding when noticing food.

Concerning the others-tasks field, basically, the actor is intended to directly see tasks of an observed actor, e.g. that he is cooking (i.e. performing "cooking" task), and right now, he is stirring a soup ("stirring ›soup‹" subtask), provided the second actor declared his tasks as public. This is to be achieved either through the mechanism of attention in case of an attractive/dangerous task, or by intentional observation.

Decay. A phantom is decaying both in the PF and the MF until it eventually vanishes. The decay can be stopped or slowed down by looking at the object repeatedly (i.e. "rehearsal") or by using the object. Similarly, tasks decay in the TF if not being performed. Typically, a task from the TF disappears a moment after it has been accomplished. When a phantom or a task vanishes from the PF/TF, it **is** recorded into the LTM, as detailed later. Tasks from the others-tasks field are intended to be handled similarly. There is no exact limit on the size of the STM, but typically, there is up to 10 phantoms and 5 tasks, which is roughly consistent with human data [12].

The LTM. The fixed arrangement of the LTM is a tree-like structure comprising tasks (Fig. 3a). During remembering, two types of entities are added into this structure: phantoms, and so-called time pointers. The basic principle of remembering

is to store everything that has just vanished from the PF or the TF (in future, we plan to add *activation* to each entity and to store only the entities above a threshold). Specifically:

1. If a phantom of an object that was seen by the actor, but not used has just vanished from the STM, its actual state, including the position, and the time of the decay are stored in the LTM, and the phantom is linked to all tasks where it can be used as a resource (Fig. 3b). Storage is done in constant time due to a hash map.

2. As soon as a task that is performed vanishes from the TF, a record is made about it using a time pointer. In the LTM, there is a set of *time pointers* that orders the tasks at every task layer of the LTM. When a new task is stored, a new pointer from the previous task to this new task is added at the appropriate layer (Fig. 3c). This pointer includes information about the time of the start and the end of the performance of the task, and a goal denoting why the task was carried out (if such a goal existed—a task could be generated without explicit active or preactive goal by an attractive object). Additionally, phantoms of the objects used as resources of this task are linked into the LTM. In case of a sequence of identical atomic actions, only their number and the pointer to the first different action are stored. In future, also a part of the motivational state of the actor that contributed most to the activation of the goal (e.g. hunger drive was high in the case of feeding goal activation), and the average emotional relevance over the time of task performance will be stored with the time pointers. Presently, the emotional relevance is determined manually for the experimental purposes.

3. The tasks performed by other actors are intended to be stored in the similar way as the own tasks (Point 2), except there will be no time pointers, but time-stamps, and no goal and motivational state will be stored (as they are not known for other actors).

Forgetting. Perhaps the most important feature of the LTM is that less emotionally interesting records are being deleted. Specifically, the emotionally less important episodes are being "bitten out" from the bottom of the LTM (Fig. 4). A *normalised importance* is computed first as: $i = e \cdot 1/h \cdot 1/d^{1/2}$, where h is the layer of the LTM structure, d number of days passed from the episode, and e is either the emotional relevance of a task (the cases 2, 3 above), or the saliency of an object (the case 1). i is low for the far-off, emotionally uninteresting, and fine-grained tasks. The time-pointers and phantoms of tasks with low i are then removed when the NPC "sleeps".

Retrieval. To increase memory retrieval speed, phantoms are (apart from the above-mentioned structure) hashed, and time pointers are indexed by days. Consequently, searching for an object, i.e. filling the memory field in the STM, is $O(1)$. The question "what did the actor performed at time t" is answered using time pointers in $O(n)$, where n is the average number of top-level tasks performed during a day.

The story reconstruction mentioned in the introduction is not implemented yet, but it will be built upon the already implemented question "what was the actor doing from t_1 to t_2". The answer to this question returns all tasks from the required layer of the LTM from t_1 to t_2. All what is needed additionally is to return the tasks from different layers to fit to the expected size of the story. Which tasks are to be chosen can be determined by their emotional importance similarly to determining which tasks should be forgotten (i.e. concrete tasks will be chosen only for highly emotional situations).

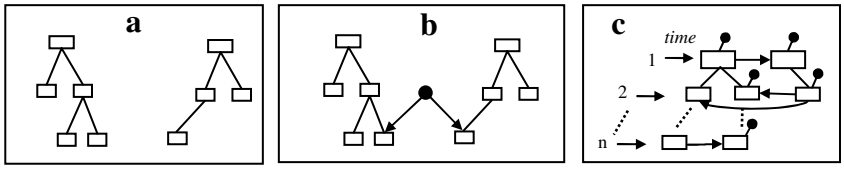


Fig. 3a. A fixed arrangement of the LTM, each box represents a task. 3b. Storage of a phantom, which was used in two tasks. 3c. The tasks are sorted by time pointers during storing.

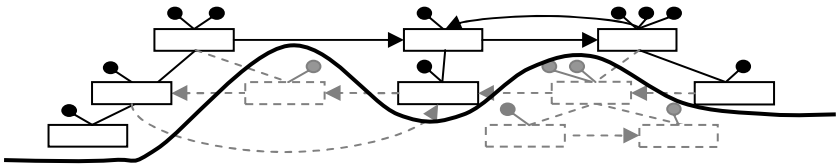


Fig. 4. Forgetting is schematically depicted. Phantoms and time pointers below a particular layer are removed with all information they included.

5 Benchmark Scenario

The trouble with measuring efficiency of a full episodic memory is that neither a rigorous testing methodology, nor a benchmark is available because of the novelty of the field. Hence, our subgoal was to create such a benchmark and to conduct preliminary tests to reveal basic characteristics of our memory and to make a first step towards laying a methodological ground. This section details the benchmark.

Any valuable benchmark scenario for an RPG must be "ecological plausible" with respect to this domain. There is no need to run a real RPG, but to model its most important features *just* to the extent that would allow for carrying out the experiments. At the first place, the scenario must feature situations in which a typical NPC can take place. Second, the scenario should last for several days. Third, it must seize a dynamic and a complexity of a typical RPG.

We have decided to exploit a five-day scenario world inhabited by a shaman NPC. The dynamic to be seized could emerge a) from the changing internal needs of the shaman, b) from her interaction with other NPCs or a player, and c) from the drama manager reasons. To model (a), during a day, the shaman is to carry out several *homeostatic* tasks like eating, sleeping, tidying up etc., one of which is a "parking" task performed when nothing more important is urging. To model (b, c), the shaman is to carry on *typical tasks*, e.g. voodoo doing, teeth extracting, as well as some *unexpected tasks* like extinguishing a sudden fire. In the abstract sense, these three kinds of tasks differ only in their timing. Every homeostatic can be performed several times a day, usually at least once, and more or less in the same time every day. A typical task can be performed anytime during a day. Unexpected tasks are carried less often, in case of this benchmark one per day. The concrete numbers used are given in Tab. 2.

For simplicity, we decided not to model other NPCs, but to imitate their presence. This is being done in two ways: first, by random changing positions of objects (i.e. "moving" them by an "NPC"), second, by invoking an NPC related goals operating with objects instead of NPCs (e.g. extracting teeth uses the object "teeth").

The former brings a problem with measuring the world *dynamic*. We decided to specify dynamic as *n*/*m*, where *n* is the number of changes of objects (because they were either taken by the shaman, or moved by an imitated NPC), and *m* is the total number of objects in the environment (i.e. dynamic can be greater than 1). Then, we measured performance of the memory in different dynamics.

To determine the exact setting and to allow for reproducibility, we decided to specify the course of activation of homeostatic goals in advance as a function of time, modelling presumed changing of drives (Tab. 1). Three typical tasks were performed once a day, and each day, one unexpected task was carried out; in both cases time being determined randomly (uniform distribution). This randomness models a random event, or an interference of a drama manager, or an interaction between the shaman and an imitated NPC. Average duration of each task was specified in advance as well, covering the intervals from minutes to hours (Tab. 1).

Complexity of RPGs further demands following:

1. The tasks have to be hierarchical, i.e. they must entail subtasks, and they may need more than one resource (e.g., extracting teeth needs both teeth and tongs).
2. There should be a plethora of different objects the shaman can use (Tab. 2).

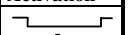
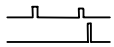
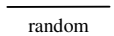
Name	Type	Depth	Durat.	Activation
Sleep	hom.	2+2	7 hours	
Eat	hom.	2+2	1+1+1	
Wash	hom.	2+2	1+1	
Smoke	hom.	2+2	0.5	
Tidy Up	hom.	3+3	1.5	
Herborize	hom.	2+2	3	
Meditate	park.	2+2	24	
Make Voodoo	typical	2+2	0.5-2	random
Predict Future	typical	2+2	0.5-2	random
Cure	typical	2+2	0.5-2	random
Heal Tooth	typical	2+2	0.5-2	random
Smoke Signals	typical	3+3	0.5-2	random
Fight	unexp.	2+2	0.5-1.5	random
Resist Wind	unexp.	2+2	0.5-1.5	random
Extinguish Fire	unexp.	2+2	0.5-1.5	random
Kill Predator	unexp.	2+2	0.5-1.5	random

Table 1. The top-level goals of the shaman. The depth is given in the number of goals+tasks. The x-scale of the activation is a one day scale.

Nr. of individual objects/classes of objects	50/18
Nr. of top-level tasks in total/ performed in one day:	
homeostatic	7/10
typical	5/3
unexpected	4/1
Avg. nr. of objects for a task parameter:	4.5
Avg. nr. of tasks an object is a resource for:	2.17

Table 2. The basic parameters of the scenario

6 Implementation and Tests

We have prototyped the memory model and the benchmark scenario in Python. We are also implementing the memory into our large-scale project [3] for the purposes of

our storytelling game [2]; but this is out of scope of this paper. We carried out two tests revealing the size of the memory and the efficiency of the actor (measured in time steps spent by searching for an object). For brevity, we detail in this paper only the benchmark and the first test. The second test is detailed in [5].

To abstract an RPG environment, we implemented a coarse-grained space model: a 3x3 grid of rooms, between which the can actor "jump" if there is a "door" (Fig. 5). Time was discretised: one time-step is 12 seconds, i.e. a day lasts 7200 time-steps. Every atomic action, including "jump", takes one time-step.

The shaman is equipped with a memory, a perception, and an action selection mechanism. The STM comprises perceptional field, memory field, and own-tasks field; the others-task field is unimplemented. The LTM allows for asking the following questions: "Where is a resource for task *T*?" (complexity of $O(1)$); "When did you do task *T*?" ($O(1)$); "Where was object *X* between t_1 and t_2?" ($O(m)$, where m are records concerning object *X*); "What did you do between t_1 and t_2?" ($O(n+m)$, where n is number of actions between t_1 and t_2 and m is time needed to find the first record from the interval, i.e. average number of top-level tasks performed during a day). The example of answer to the last question in the current implementation is:

> *"I was doing SearchRandom for smokeability because of Smoke. I was doing go from room 1 to room 2 because of SearchRandom. I was doing look in environment because of SearchRandom. I was doing go from room 2 to room 5 because of SearchRandom. I was doing pick up Calumet1 because of Smoke. I was doing Smoke."*

If there are two different records concerning the same object, the newer record is considered as more trustful. As already mentioned, the suggested linguistic module is out of our scope presently, hence unimplemented. The emotional module is implemented separately [1], i.e. not included in the prototype. Hence, LTM forgetting is implemented in an *ad hoc* manner; during a night (when it is "sleeping") the actor forgets the bottom level of time pointers and phantoms for the previous day for all tasks except of the unexpected one (which is considered as emotionally salient).

Setting. Two experiments were carried out on a 2 GHz Win-XP PC; the first one measuring the size of the memory without forgetting, and the second one with forgetting. We carried out 3 variants of each test; each with a different world dynamic (0.1, 0.5, 1). Ten runs were carried for each variant; each simulating 5 consecutive days. The position of all objects was randomised for each run. The saliency of the objects was switched off to reduce the amount of parameters of the simulation, i.e. only the objects the shaman used were remembered.

Results. The size of the fixed LTM structure was 140 items (tasks and atomic actions). Average number of phantoms and time pointers respectively in the LTM without forgetting for one day in dynamic 0.5 (over all runs and days) was: 90/277 comparing to 90/136 of the memory with forgetting. For comparison, *naïve memory* storing at every time step all objects in the room the actor was located in, stored 40.000 phantoms in one day on average. Fig. 6 depicts the change of the number of time-pointers stored in the memory with and without forgetting for dynamic 0.5 (average over 10 runs). We do not show the results for dynamics 0.1 and 1 for the tests showed the numbers are almost the same.

Discussion. A real-time RPG is a complex system with a vast amount of parameters. Any aspect of behaviour of memory can be understood only as a multi-dimensional, likely non-linear function over these parameters. To study complex, non-linear systems, scientists typically follow reductionism. We adopted this approach as well. We left the saliency effect of objects on attention out of account, we implemented forgetting without a real emotional importance, we avoided NPCs, and randomised objects' positions at the start of each trial (they should have been clustered by rooms instead—e.g. a spade typically can not be found in a bedroom). The world dynamic was calculated as if the probability that an object can be moved was same for all objects, disregarding that some objects are moving objects, some can change their state without changing the position, and some can be moved more often than others.

Still, after the reduction, the tests revealed several important observations:

1. Contrary to time pointers, phantoms are not being forgotten, since all of them are used as resources not only for atomic actions, but also higher-level tasks. Only if forgetting touches higher levels of the LTM structure, phantoms can be forgotten.
2. Our memory is far better than the naïve one, but still, given 200 B for an item, after 5 days, the shaman's LTM will possess 395 kB (200·(140+5·(90+277))) without, and 254 kB (200·(140+5·(90+136))) with forgetting. This fact and Point 1 suggest that forgetting, even deeper, is essential.
3. The memory size is independent on the world dynamic. The reason is that only tasks performed and their resources were stored, and their amount does not depend on the dynamic. Will the size become dependent when storing also salient objects?

Apart from this issue (Point 3), we plan to include real NPCs into the tests. It would be also interesting to compare other models of episodic memories, e.g. [14], using our benchmark. Besides directions of future trialling, there are some open questions concerning the model itself. First, trustfulness of the objects must be determined more plausibly. In worlds with high dynamics, the actor tends to search on wrong places, because the object in question has been already moved. The trustfulness issue is our work-in-progress [4]. This work also includes a better spatial representation of the environment for it is essential in determining the trustfulness. Second, actions of other NPCs are to be stored as well so that the reconstruction of *their* stories is possible. This is another work-in-progress. Third, equipping the actor with emotions and drives is needed to allow for implementing real forgetting. This will also elevate overall believability of the actor. Fourth, forgetting should be also augmented by a process of merging several similar or repeated unimportant episodes together (see the example from Introduction). Fifth, neither our actor currently can carry out two atomic actions in parallel (e.g. go and eat), nor is the memory optimised for task interleaving. These last two points presents future work. Finally, there is a sort of "holy-grail": guessing the goals of a user avatar. Contrary to the foreign NPCs, there is no available information about goals of the user. The NPC can only guess them from the sequences of atomic actions, hence create a sort of "theory of mind". We have an idea how to address this problem by bayesian learning, this is, however, a pie in the sky.

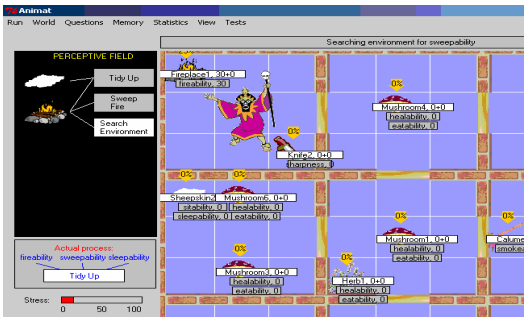


Fig. 5. A part of a screen-shot from the prototype. Left: the short term memory. Right: the grid world of the shaman actor (only 4 rooms from 9 are shown). The shaman is in the upper left room.

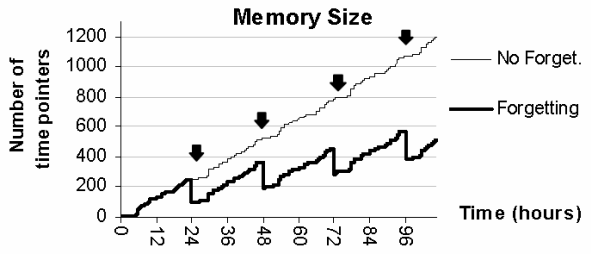


Fig. 6. Average nr. of time pointers stored in the memory during 5 days (over 10 runs; dynamic =0.5). The arrows denotes nights.

7 Conclusion

We have presented a full episodic memory for a non-player character from an RPG to allow for reconstruction of the character's personal history. The memory is widely optimised on effective storing and retrieval. Additionally, it enables to store episodes for a long time. These are the most notable features that make the memory applicable for RPGs. The model is portable to any other application featuring human-like actors provided it employs behavioural representations based on AND-OR trees.

We have also introduced the "5-day" shaman scenario aiming as a general benchmark for full episodic memory actors. Using this benchmark, we demonstrated that memory performs well concerning its size; however, forgetting was essential.

So far, the memory only stores actions the actor performed, and objects it used, and the emotional salience of the episodes must be determined manually. We are now connecting the memory to the emotional model [1], and implementing the storage of actions of *other* actors. Additionally, the issue of determining trustfulness of memory records is addressed [4]. The final actor will be ported to our large scale project [3].

Acknowledgments. This research was partially supported by the Project of the Ministry of Education of the Czech Republic No. MSM0021620838, and GA UK 351/2006/A-INF/MFF. Authors would like to thank to Andrew Nuxoll for answering questions on his work, and Tomáš Holan for general comments.

References

1. Bída, M., Kadlec, R., Brom, C.: Relevance of emotions for artificial beings (in Czech). In: Mind, intelligence and life, Vydavatel'stvo STU, Bratislava, pp. 158–172 (2007)
2. 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)
3. 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) (6.3.2007), http://urtax.ms.mff.cuni.cz/ive/public/about.php
4. Brom, C., Pešková, K., Lukavský, J.: Determining trustfulness of records in episodic memory by means of an associative network. In: Proc. of ECAL 2007, Lisbon, Portugal, Springer, Heidelberg (in press)
5. Brom, C., Pešková, K., Lukavský, J.: Modelling human-like RPG agents with a full episodic memory. Technical Report No. 2007/4 of the Department of Software and Computer Science Education, Charles University in Prague, Czech Republic (2007)
6. Dodd, W.: The design of procedural, semantic, and episodic memory systems for a cognitive robot. Master thesis. Vanderbilt University, Nashville, Tennessee (2005)
7. Ho, W., Dautenhahn, K., Nehaniv, C.: Autobiographic Agents in Dynamic Virtual Environments - Performance Comparision for Different Memory Control Architectures. In: Proc. IEEE CEC, pp. 573–580 (2005)
8. Ho, W., Dias, J., Figueiredo, J., Paiva, A.: Agents that remember can tell stories: Integrating Autobiographic Memory into Emotional Agents. In: Proc. of AAMAS, ACM Press, New York (2007)
9. Johnson, W.L.: Agents that learn to explain themselves. In: Proc. of the 12th Nat. Conf. on Artificial Intelligence, pp. 1257–1263. AAAI Press (1994)
10. Khaneman, D.: Attention and Effort. Prentice-Hall, New Jersey (1973)
11. Mateas, M.: Interactive Drama, Art and Artificial Intelligence. PhD thesis. Department of Computer Science, Carnegie Mellon University (2002)
12. Miller, T.: The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological review 63(2), 81–97 (1956)
13. Neisser, U.: John Dean's Memory: A case-study. Cognition 9, 1–22 (1981)
14. Nuxoll, A.: Enhancing Intelligent Agents with Episodic Memory. PhD thesis, The University of Michigan (2007)
15. Peters, C., O'Sullivan, C.: A Memory Model for Autonomous Virtual Humans. In: Proceedings of Eurographics, Ireland, pp. 21–26 (2002)
16. Rickel, J., Johnson, W.L.: Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control. In: App. Artificial Intelligence, vol. 13 (1999)
17. Tulving, E.: Elements of Episodic Memory. Claredon Press, Oxford (1983