Comparing Different Control Architectures for Autobiographic Agents in Static Virtual Environments

Wan Ching Ho, Kerstin Dautenhahn, Chrystopher L. Nehaniv

Adaptive Systems Research Group
Department of Computer Science, University of Hertfordshire
College Lane, Hatfield, Hertfordshire, AL10 9AB, UK
[W.C.Ho | K.Dautenhahn | C.L.Nehaniv] @herts.ac.uk

Abstract
This paper highlights the effectiveness of autobiographic memory applied to an autonomous agent from an Artificial Life perspective. A virtual experimental based approach deals with different implementations of control architectures for autobiographic agents, including detailed measurements of the agent's lifetimes compared with a Purely Reactive agent, running in two distinct static environments. Experimental results produce evidence which confirms the research hypothesis that autobiographic memory can prove beneficial, indicating increases in the lifetime of an autonomous, autobiographic, minimal agent. In the conclusion, reference is made to future research geared towards improving the current framework.

Introduction
This paper studies autobiographic memory in virtual agents. Autobiographic memory is a specific kind of episodic memory, which develops in human childhood [1]. Autobiographic agents are agents which are embodied and situated in a particular environment (including other agents), and which dynamically reconstruct their individual history (autobiography) during their lifetimes, as defined in [2]. Autobiographic memory also can be used for synthesizing agents that can behave adaptively [3] and also in socially intelligent ways [4], and for designing agents that appear believable and acceptable to humans [5].

In the area of Intelligent Virtual Agents [6], virtually embodied agents should be able to achieve a certain level of autonomy, which means they need to maintain their own internal states at an adequate level, besides responding to and reacting to different types of stimuli from their environments by their virtual sensors and virtual actuators. To achieve these goals, a robust control system is necessary to control the agents. The subsumption architecture [7] is one of the solutions dedicated to behaviour-based agents, since it decomposes the problem by defining layers of competing behaviours and defining relationships between those behaviours. This leads to a control architecture that can select an action to execute immediately based on the current value of the sensors and internal states of the agent, even though the agent does not possess any complex internal knowledge representations, such as an overall map of the environment. Furthermore, we implemented principles of the agent programming language PDL [8] which updates sensory quantities using the latest reading from sensors and which then sends new values of the action parameter quantities to the actuators. PDL was previously designed to specifically support dynamical, life-like behaviour for autonomous agents.

Related Work
Earlier studies suggested autobiographic memory is a social function of memory, underlying all of our story telling and history-making narrative activities. Thus, it was suggested that the primary function of autobiographic memory is sharing memory with other people [1]. Later, research [3] discussed that historical grounding of autobiographical agents helps developing the individualized social relationships and forms of communication, which are characteristic of social intelligence. It may also lead to more appealing and 'human-like' engaging interactions with artificial agents, making them more pleasant and acceptable to humans.

The first set of experiments of autobiographic agents used Situation-Action-Situation triplets as the core of the agents' autobiography. [9] Results showed that autobiographic memory which is embedded in the control architecture of the agent can effectively extend an agent's lifetime. This previous work compared trajectories and lifetimes of purely reactive (sensory-driven) and post-reactive (memory-driven) control agents.

Research Hypothesis

An experimental framework has been used in this paper for studying different hypotheses important to develop autonomous autobiographic agents from an agent-centered viewpoint. We investigate the hypotheses that, for appropriate contexts, different control architectures of minimal autobiographic agents [9] with a finite-length memory can be preferential to an autonomous, reactive agent. At the outset, control architectures for autobiographic agents are categorized as *Trace-back* and *Locality*. The main characteristic of memory functioning in the *Trace-back* architecture is that a memory entry is continuously and directly made based on sensory information while the agent is wandering around in its environment. This is similar to a short-term memory since earlier entries will be replaced when all memory entries are filled. Comparatively, a long-term memory approach is implemented in the *Locality* architecture. Here, only information which is meaningful to the agent will be remembered as long-term memory entries. Moreover, *Event-based* and *Time-based* are two inherently different mechanisms used for making memory entries with the *Trace-back* control architecture. The Event-based mechanism is based on the agent's encounters with various objects in the environment, in contrast to the Time-based mechanism, where agents make memory entries at regular, fixed periods of time (periodically after a certain number of time steps).

The lack of unique object identifiers in the virtual environment produces a limiting factor. As a consequence, this will inhibit the agent's ability to recognize specific targets in unfamiliar environments. Similarly, the unavailability of global coordinate information of distinct objects restricts the agents' resource tracking system. An important factor which must be taken into account in our experiments is the noise generated from the environment which influences the precision of the agents' actuators. Noise is likely to decrease the usefulness of the autobiographic memory in the process of remembering. We therefore hypothesize that noise will impair the agent's survival in the environment.

Experiments

The web-based 3D technology, VRML provides a set of comprehensive definitions of geometric shapes for building a suitable wide range of virtual environments and 3D objects involved in the related experimental simulations. To achieve a certain level of autonomy for virtual agents, control architectures for agent behaviours are implemented by using JavaScript and applying timed constraint programming techniques [10]. In each time step, the script node which acts as the brain of the agent, routes necessary information to modify the shapes of the agent's body and the objects in the environment, so that autonomous behaviour and continuous movement can be carried out. Figure 1 illustrates the simple design of using VRML to implement virtually autonomous agents, which can dynamically change the body (geometry) in terms of position, shape, or movement by calculating the agent's own status. Figure 1 also shows that duplicating the original prototype of agents can lead to multi-agent systems, in which agents of comparatively similar architecture facilitate the communication phase with routing information between them for cooperative goals. Future experiments will take advantage of this multi-agent option. In the present work only one virtual agent in the environment is being studied at a time.

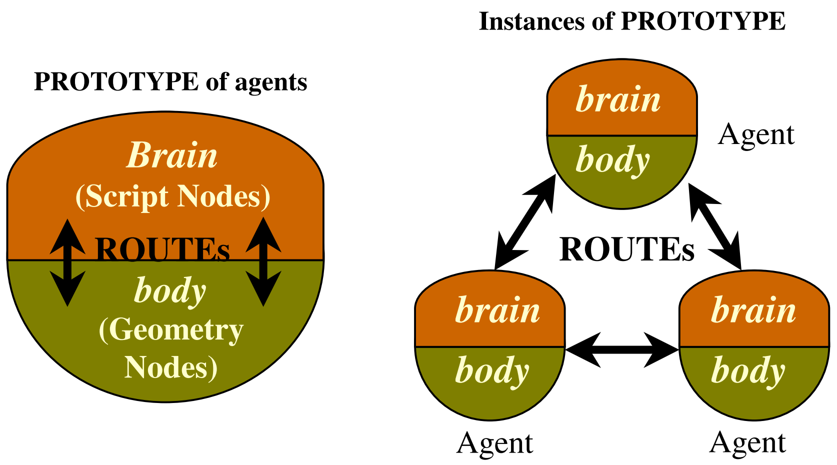


Figure 1. Simple design of a virtual autonomous agents using VRML (left), the design for multi-agent systems is shown on the right.

Virtual Environment & Virtual Agent

As the experimental test-bed, we designed large size and static environments with different object distributions with basic geometric shapes which are generated in real-time from VRML. In addition to being able to study the consequences of autobiographic memory influencing an agent's behaviours, we are also concerned with system performance since VRML is essentially a descriptive based language for 3D scenes and objects distributed on the web. The virtual environment consists of various types of resources including food, water and nests, as well as obstacles and walls which serve as the boundaries of the environments. Experimental virtual environments can be categorized according to the distribution of objects. For the first type of autobiographic agents using the *Trace-back* control architecture objects are uniformly and symmetrically distributed in the environment. For the second type of autobiographic agents using the *Locality* control architecture, similar types of objects are grouped together in certain areas which can be easily found by the agent in the initial stage of making memory entries. These two types of environments are illustrated in Figures 2.1 and 2.2.

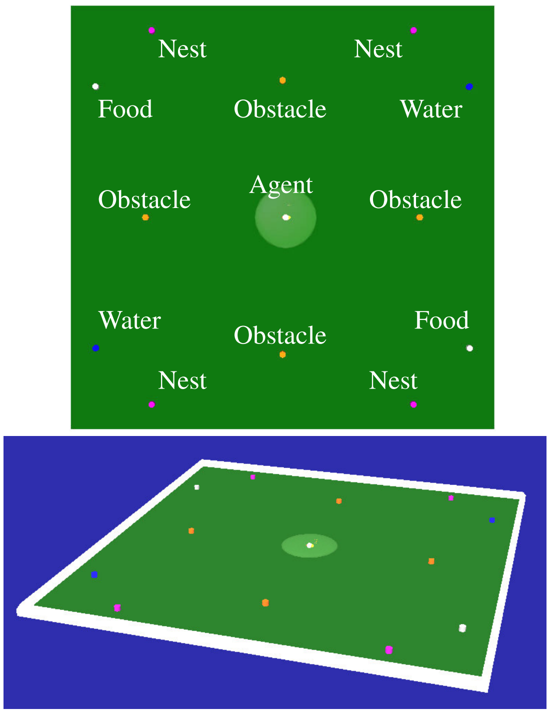


Figure 2.1 View of looking down from above and 45-degrees, showing the object distribution in the first type of virtual environment where the Trace-back control architecture is used.

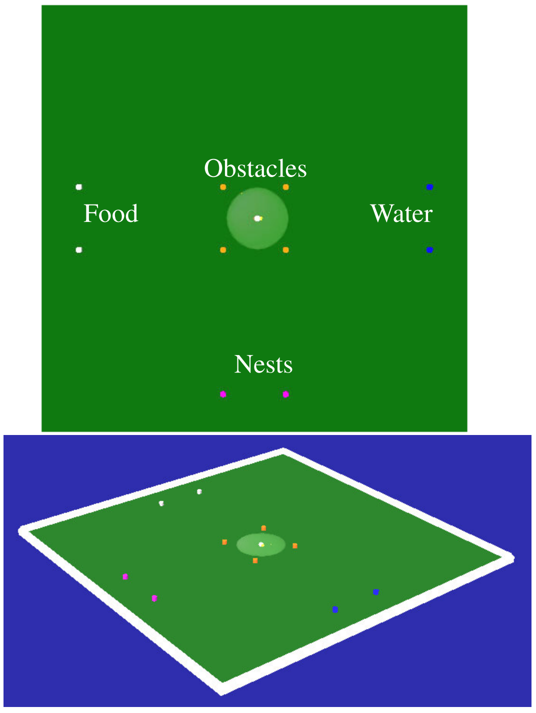


Figure 2.2 Two different views of object distributions in the second type of virtual environment using the Locality control architecture.

The virtual agent has a highly abstracted body, which includes the head, and the tail for indicating the agent's current direction. A partly transparent area of circular shape shows the effective range provided by the virtual sensor of the agent. The appearance of the virtual agent is shown in Figure 3.

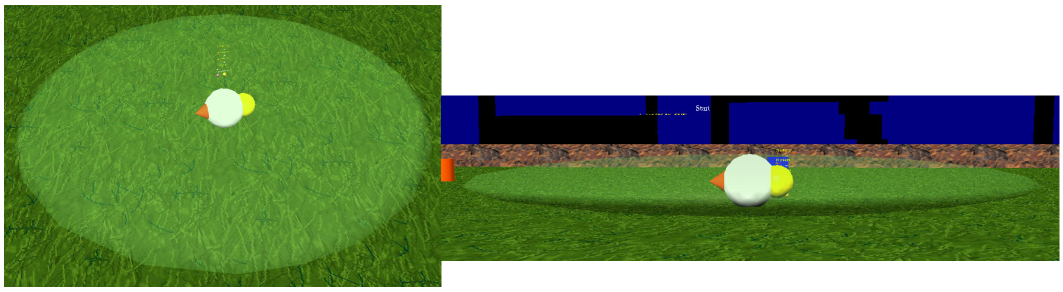


Figure 3. An abstract model of a virtual agent with a round- shaped sensing area. The yellow sphere is the head, and the red cone is the tail of the agent.

Memory Architectures

We study three types of agents, which are based on a subsumption control architecture, namely *Purely Reactive*, *Trace-back* and *Locality*. All agents have a finite lifetime. The survival of an agent depends on maintaining homeostasis for three internal variables: Energy, Moisture and Glucose. Each internal variable is initialised with a maximum value at the start of each experimental simulation run. Each translation or rotation of the agent will reduce each internal variable by a certain value. When one of the internal variables drops below a threshold, which is half of the maximum value, then the agent will be searching around for a specific resource located in the environment. If the agent is not able to detect the resource which it needs, and if the value of a particular internal variable is less than a particular minimum value, then the agent will die. The experimental parameters (thresholds etc.) that allow the agents to live in the virtual environment, but eventually die, were determined in initial tests.

Purely Reactive. The architecture of the Purely Reactive agent includes 5 layers and higher-level behaviours, which inhibit or override lower-level behaviours. The agent usually wanders around in the environment by executing the bottom layer in the architecture. At the same time, PDL control for increasing or decreasing the velocity of the agent is employed to achieve a certain level of continuous, natural movement; for example, the agent would slowly increase its velocity after it finished an obstacle avoidance behaviour and when it started again to explore the environment. When the agent encounters an object, which can be any kind of resource, obstacle or one of the boundaries of the environment (walls), then the agent avoids the obstacle or the wall by generating a random direction rotating its body. This behaviour will also be triggered in case the agent encounters a resource object, but in that case the internal state which needs that particular resource will be higher than the threshold. Figure 4 shows the control architecture for the Purely Reactive agent.

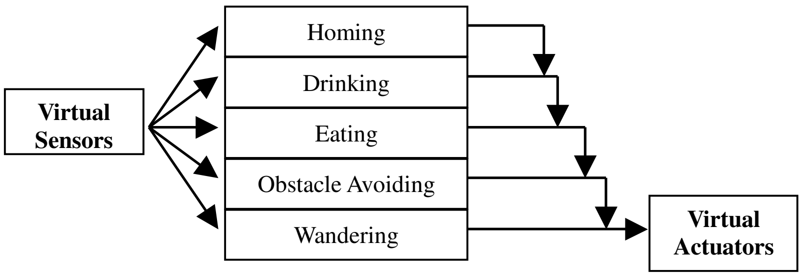


Figure 4. Behaviour hierarchy which is based on the subsumption architecture for a Purely Reactive agent.

Trace-back. On the basis of the design of a Purely Reactive architecture, autobiographic agents with memory *Trace-back* or *Locality* possess an autobiographic memory module on top of the subsumption architecture. Each type of autobiographic agent has a unique mechanism for making memory entries as the remembering process, and *using* the memory as a tracing process. In the case of the *Trace-back* mechanism, the agent has a finite number of circularly-ordered memory entries. Introduction of new entries depends on the way of making memory entries for the agent (Event-based or Time-based). In the Event-based mode, making new memory entries occurs when the agent encounters a different object and changes its current behaviour; in the Time-Based mode, a new memory entry is made every fixed number of time steps. Information of each memory entry includes the current time step of the simulation, the behaviour which is being executed, the direction the agent is facing, the object encountered by the agent (if any), how far the agent has walked (Distance) since last encountering an object, as well as the current internal state. This information is inserted at the current position of the index into the finite circular memory. Figure 5.1 illustrates the memory architecture of the *Trace-back* mechanism, Figure 5.2 shows samples of memory entries made by executing the Event-based mode and the Time-based mode.

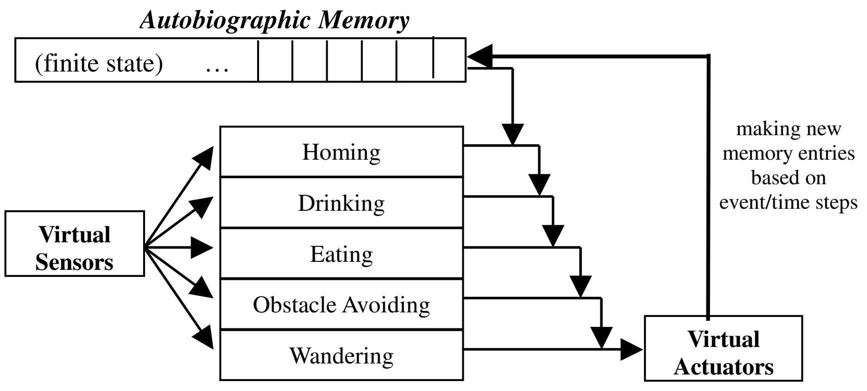
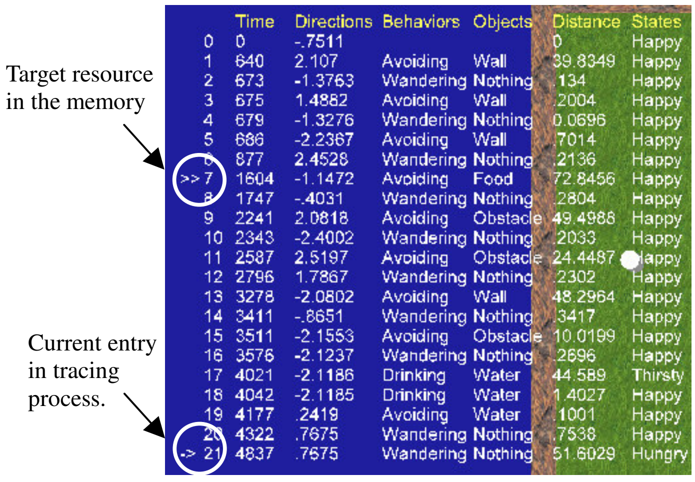
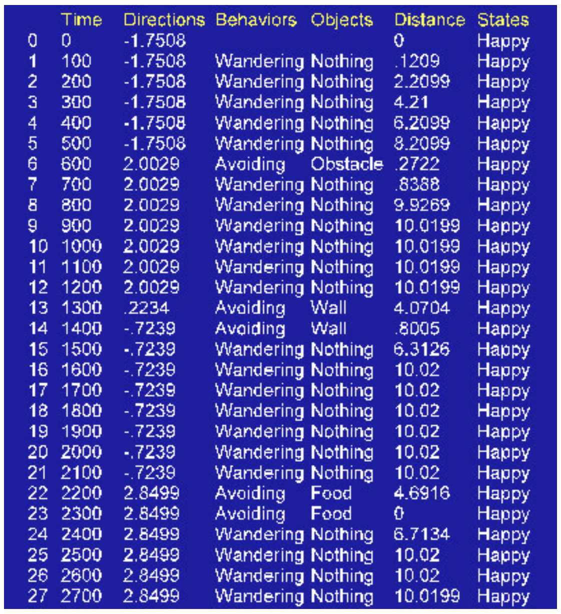


Figure 5.1 Memory architecture of Trace-back mechanism.


Target resource in the memory
Current entry in tracing process.

	Time	Directions	Behaviors	Objects	Distance	States
0	0	-.7511			0	Happy
1	640	2.107	Avoiding	Wall	39.8349	Happy
2	673	-1.3763	Wandering	Nothing	.134	Happy
3	675	1.4882	Avoiding	Wall	.2004	Happy
4	679	-1.3276	Wandering	Nothing	0.0696	Happy
5	686	-2.2367	Avoiding	Wall	.7014	Happy
6	877	2.4528	Wandering	Nothing	.2136	Happy
>> 7	1604	-1.1472	Avoiding	Food	72.8456	Happy
8	1747	-.4031	Wandering	Nothing	.2804	Happy
9	2241	2.0818	Avoiding	Obstacle	49.4988	Happy
10	2343	-2.4002	Wandering	Nothing	.2033	Happy
11	2587	2.5197	Avoiding	Obstacle	24.4487	Happy
12	2796	1.7867	Wandering	Nothing	.2302	Happy
13	3278	-2.0802	Avoiding	Wall	48.2964	Happy
14	3411	-.8651	Wandering	Nothing	.3417	Happy
15	3511	-2.1553	Avoiding	Obstacle	10.0199	Happy
16	3576	-2.1237	Wandering	Nothing	.2696	Happy
17	4021	-2.1186	Drinking	Water	44.589	Thirsty
18	4042	-2.1185	Drinking	Water	1.4027	Happy
19	4177	.2419	Avoiding	Water	.1001	Happy
20	4322	.7675	Wandering	Nothing	.7538	Happy
-> 21	4837	.7675	Wandering	Nothing	51.6029	Hungry



	Time	Directions	Behaviors	Objects	Distance	States
0	0	-1.7508			0	Happy
1	100	-1.7508	Wandering	Nothing	.1209	Happy
2	200	-1.7508	Wandering	Nothing	2.2099	Happy
3	300	-1.7508	Wandering	Nothing	4.21	Happy
4	400	-1.7508	Wandering	Nothing	6.2099	Happy
5	500	-1.7508	Wandering	Nothing	8.2099	Happy
6	600	2.0029	Avoiding	Obstacle	.2722	Happy
7	700	2.0029	Wandering	Nothing	.8388	Happy
8	800	2.0029	Wandering	Nothing	9.9269	Happy
9	900	2.0029	Wandering	Nothing	10.0199	Happy
10	1000	2.0029	Wandering	Nothing	10.0199	Happy
11	1100	2.0029	Wandering	Nothing	10.0199	Happy
12	1200	2.0029	Wandering	Nothing	10.0199	Happy
13	1300	.2234	Avoiding	Wall	4.0704	Happy
14	1400	-.7239	Avoiding	Wall	.8005	Happy
15	1500	-.7239	Wandering	Nothing	6.3126	Happy
16	1600	-.7239	Wandering	Nothing	10.02	Happy
17	1700	-.7239	Wandering	Nothing	10.02	Happy
18	1800	-.7239	Wandering	Nothing	10.02	Happy
19	1900	-.7239	Wandering	Nothing	10.02	Happy
20	2000	-.7239	Wandering	Nothing	10.02	Happy
21	2100	-.7239	Wandering	Nothing	10.02	Happy
22	2200	2.8499	Avoiding	Food	4.6916	Happy
23	2300	2.8499	Avoiding	Food	0	Happy
24	2400	2.8499	Wandering	Nothing	6.7134	Happy
25	2500	2.8499	Wandering	Nothing	10.02	Happy
26	2600	2.8499	Wandering	Nothing	10.02	Happy
27	2700	2.8499	Wandering	Nothing	10.0199	Happy

Figure 5.2 Memory entries made by executing Event-based mode (left) and Time-based mode (right).

The memory trace back process will be triggered if one of the internal states of the agent is lower than the threshold, and if the resource can be found at any entry, which indicates that the agent has previously encountered the resource. Once trace back has started, the agents will simply "undo" all previous behaviours. As indicated in Figure 5.2, the agent will execute in reverse order, the action to undo each step starting from the current entry to the target resource. This mechanism has a close connection to the group-theoretic notion of inverse in mathematics [3]. Thus, the agent will execute the reverse of each action step-by-step using the information specified in Direction and Distance. The trace back process will be completed once the agent has executed all memories entries and has reached the target resource. At this moment, the agent will start sensing around for the resource. Note, there is a possibility that the resource is not available at this location since the actual rotation value in each entry might have been slightly distorted by noise. Also, the trace back process will be terminated if the agent collides with any of the objects in the environment (e.g. due to accumulated errors caused by noise).

Locality. In contrast to the *Trace-back* mechanism, the *Locality* memory agent has only four memory entries for remembering the most recent behaviours using the Event-based mechanism. In addition, the agent maintains information relevant for traveling from one type of object to any other given type of object (*Locality* memory). Making or replacing entries of long-term *Locality* memory occurs exactly when a particular pair of objects has been encountered. Three 3x2 tables represent all relevant information for the possible specific pair of objects; each row of the table contains information of Direction and Distance, specifying what the agent did while traveling between two objects (Figure 6.2). The tracing memory process occurs when the agent is looking for a specific resource and the entry of the object leading the agent to that resource contains the required information. The *Locality* memory architecture of the agent is shown in Figure 6.1, and a sample of memory entries is shown in Figure 6.2.

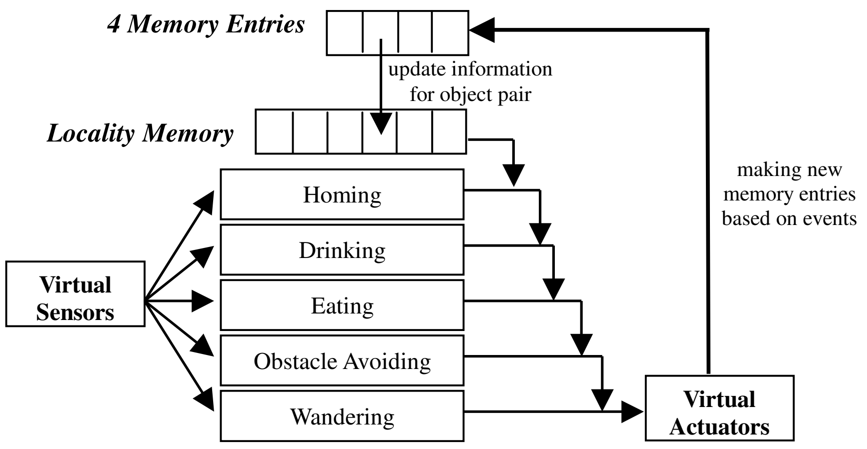
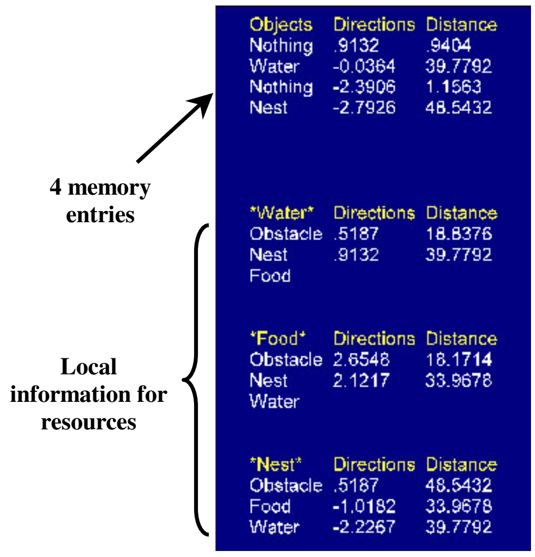


Figure 6.1 Memory architecture of Locality mechanism.



Objects	Directions	Distance
Nothing	.9132	.9404
Water	-0.0364	39.7792
Nothing	-2.3906	1.1563
Nest	-2.7926	48.5432

Water	Directions	Distance
Obstacle	.5187	18.8376
Nest	.9132	39.7792
Food		

Food	Directions	Distance
Obstacle	2.6548	18.1714
Nest	2.1217	33.9678
Water		

Nest	Directions	Distance
Obstacle	.5187	48.5432
Food	-1.0182	33.9678
Water	-2.2267	39.7792

Figure 6.2 Memory entries made by Locality mechanism

Design of Experimental Runs

Experiments for examining the survival of autobiographic agents in static environments can be separated into two categories, namely *Trace-back* memory agents versus a Purely Reactive agent, and *Locality* memory agents versus a Purely Reactive agent. Figure 7.1 shows the design for experimental runs for *Trace-back* memory agents. Each condition has 50 runs and takes 2 to 3 hours for running on a desktop computer (Pentium 4 2GHz, 512MB main memory); there are 25 conditions for different types of agents which are uniquely parameterized. For experiments with noise interference, Gaussian noise with a standard deviation of 5 degrees was applied to turning angles.

Conditon	PR	EB mem20	EB mem50	EB mem100	TB 50 mem20	TB 50 mem50	TB 50 mem100	TB 100 mem20	TB 100 mem50	TB 100 mem100	TB 200 mem20	TB 200 mem50	TB 200 mem100
Mem_Size	0	20	50	100	20	50	100	20	50	100	20	50	100
Time_Step					50	50	50	100	100	100	200	200	200
Noise		N	N	N	N	N	N	N	N	N	N	N	N

Condition	EB mem20	EB mem50	EB mem100	TB 50 mem20	TB 50 mem50	TB 50 mem100	TB 100 mem20	TB 100 mem50	TB 100 mem100	TB 200 mem20	TB 200 mem50	TB 200 mem100
Mem_Size	20	50	100	20	50	100	20	50	100	20	50	100
Time_Step				50	50	50	100	100	100	200	200	200
Noise	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

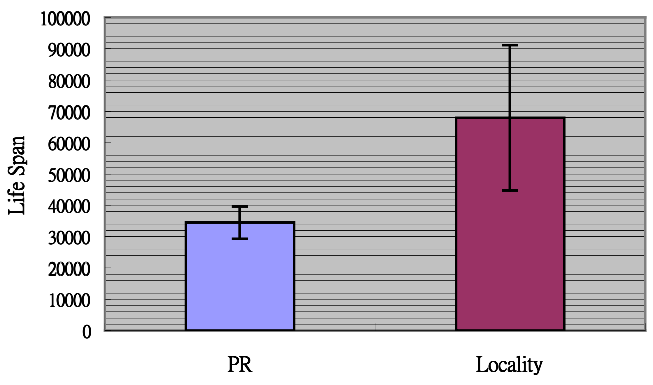
Figure 7.1 Experimental runs for Trace-back memory agents, compared with Purely Reactive agents (PR). Different parameters for Even-based (EB) and Time-based (TB) are shown in both tables, experiments in the upper table are without noise interference, experiments in the lower table are with noise interference. MemX is the length of the circularly-ordered memory, Time_step is the period between making entries for the TB architectures.

For a comparison of the *Locality* memory agent and the Purely Reactive agent, another two sets of experiments of 50 runs for each type have been carried out in the second environment.

Results

Figure 8.1 and 8.2 illustrate the results of the average lifetime for each set (50 runs) of the experiments. Confidence values are shown as error bars.

Figure 8.1 Experimental results showing life span of Purely Reactive agents vs. Locality memory



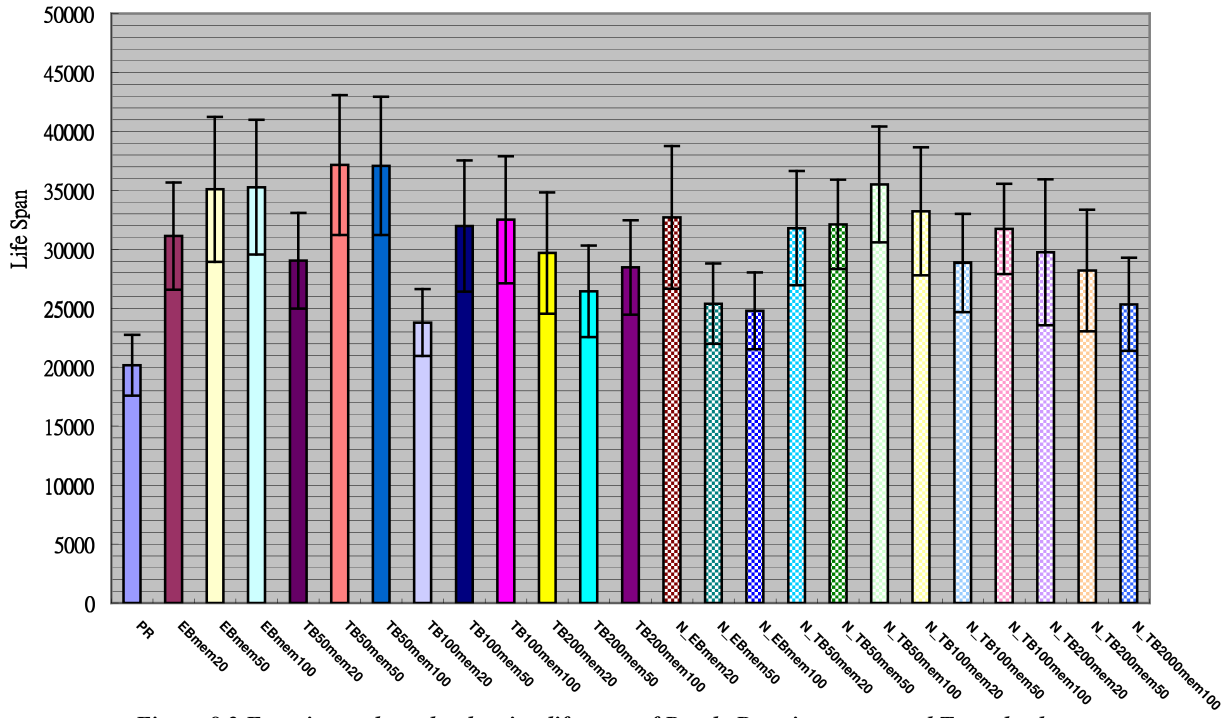


Figure 8.2 Experimental results showing life span of Purely Reactive agents and Trace-back memory agents. Experimental conditions starting with N indicate Gaussian noise.

Discussion of Experiments

The experiments confirm our research hypothesis, as the results showed that both approaches of autobiographic memory effectively extended the agents' lifetime in most cases. The *Locality* memory agent performs far better than a Purely Reactive agent in the specific static environment that we used. The general results without noise interference, both for Time-based and Event-based mechanisms of *Trace-back* memory, indicate that agents with a larger number of memory entries have longer lifetimes (except in the Time-based mode when the number of time steps of making memory entry equals 200; this exception might be explained by the fact that a very large memory may be cluttered with unimportant events.) With noise interference, experimental results are basically reversed. Since a larger number of memory entries allows the accumulation of more noise disturbances, incremental deviations of rotation values can cause the agent to get "lost" in the environment when memory rotation values are retrieved from memory and applied to the actuators.

Furthermore, experiments also reveal a disadvantage of the *Trace-back* mechanism, especially when the agent has a large number of memory entries. In case a useful resource was encountered a long time ago, and the memory entry for this resource occurred much earlier in comparison to the current entry, then the agent could die easily halfway through the trace back process. This might be remedied by cost-benefit analysis by the agent to determine whether it would pay off to use an old memory.

The *Locality* mechanism seems to be an approach for using long-term memory for the agent to remember only information which is significant to its survival in the environment. Nevertheless, in the current stage of experiments, object distribution and environmental exploration are limited by applying this approach, and only a static environment has been studied.

Conclusion and Future Work

The comparison of the agents' lifetimes show significant differences between Purely Reactive agents and Autobiographic agents. Our results indicate that autobiographic memories increase the adaptability (in terms of survival) of the agent in static environments. At present we are designing a more sophisticated memory architecture, which has been inspired by autobiographic human memory [11]. Dynamic environments for investigating existing autobiographic agent architectures will be studied with dynamic resource allocation, i.e. the existence of resources will be made less predictable by introducing time varying features to the environment, such as 'seasons'. In addition to further investigating the potential of the *Locality* memory approach, different types of memory approaches for autonomous agents, such as geometric maps, cognitive maps, and internal world models, are going to be evaluated and studied for a comparison to current approaches. Multi-agent issues including agents competing for resources or cooperating with others to achieve certain goals in the environment, can also be considered as studies of agent communication. For example, agents could selectively trust the 'hints' from one of their companions who has provided helpful information in the past.

Reference

[1] Nelson, K. (1993) The Psychological and social origins of autobiographical memory. *Psychological Science*, 4, 7-14.
[2] Dautenhahn, K. (1996) Embodiment in Animals and Artifacts, *Proc. AAAI FS Embodied Cognition and Action*, AAAI Press, Technical report FS-96-02, 27-32.
[3] Nehaniv, C. & Dautenhahn, K (1998), Embodiment and Memories - Algebras of Time and History for Autobiographic Agents, Embodied Cognition and AI symposium. *Proc. of 14th European Meeting on Cybernetics and Systems Research*, 651-656.
[4] Dautenhahn, K. (1999) Embodiment and Interaction in Socially Intelligent Life-Like Agents. In: C. L. Nehaniv (ed): *Computation for Metaphors, Analogy and Agent*, Springer Lecture Notes in Artificial Intelligence, Volume 1562, Springer, 102-142.
[5] Dautenhahn, K. (1998) The Art of Designing Socially Intelligent Agents - Science, Fiction, and the Human in the Loop. *Applied Artificial Intelligence*, Vol 12 (7-8), 573-617.
[6] Aylett, R. & Luck, M. (2000) Applying Artificial Intelligence to Virtual Reality: Intelligent Virtual Environments. *Applied Artificial Intelligence* 14(1): 3-32.
[7] Brooks, R. (1985) A Robust Layered Control System for a Mobile Robot. *MIT AI Lab Memo,* 864.
[8] Steels, L. (1992) The PDL reference manual. *VUB AI Lab memo*. 92-5.
[9] Dautenhahn, K. & Coles, S. (2000) Narrative Intelligence from the Bottom Up: A Computational Framework for the Study of Story-Telling in Autonomous Agents, *The Journal of Artificial Societies and Social Simulation*, *31st January 2001.*
[10] Saraswat, V, A., Jagadeesan, R., Gupta, Vineet. (1994) Foundations of Timed Concurrent Constraint Programming. *Proceedings of the 1994 IEEE Symposium on Logic in Computer Science, July 1994.*
[11] Schank, R, C. & Abelson, R, P. (1995) Knowledge and memory: the real story. In Robert S. Wyer, editor, *Knowledge and memory: the real story*, pages 1-85, Hillsdale, New Jersey. Lawrence Erlbaum Associates.