Affective vs. Deliberative Agent Control

Matthias Scheutz[⋆] and Brian Logan[†]

[⋆]School of Computer Science
University of Birmingham
Birmingham, B15 2TT, UK
mxs@cs.bham.ac.uk

[†]School of Computer Science and IT
University of Nottingham
Nottingham, NG8 1BB, UK
bsl@cs.nott.ac.uk

Abstract

In this paper, we outline a research strategy for analysing the properties of different agent architectures, in particular the cognitive and affective states/processes they can support. We demonstrate this architecture-based research strategy, which effectively views cognitive and affective states as architecture-dependent, with an example of a simulated multi-agent environment, where agents with different architectures have to compete for survival. We show that agents with "affective" and "deliberative" capabilities do best in different kinds of environments and briefly discuss the implications of combining affective and deliberative capabilities in a single architecture. We argue that such explorations of the trade-offs of alternative architectures will help us understand the role of affective processes in agent control and reasoning, and may lead to important new insights in the attempt to understand natural intelligence and evolutionary trajectories.

1 Introduction

Deliberative mechanisms and processes (such as planning, searching, reasoning, etc.) have been a major focus of research activities ever since the beginning of artificial intelligence. More recently, affective states have become another area of attention, especially in the design of artificial "believable" agents (Simon, 1967; Sloman and Croucher, 1981; Damasio, 1994; Reilly, 1996; Picard, 1997; Hatano et al., 2000). However, the interaction of affective and deliberative processes in biological agents and the possibilities of integrating affective and deliberative components in control systems of artificial agents (synthetic or robotic) are not yet well understood. This is partly due to the complexity of the subject matter, but also partly to complications brought about by an overwhelming number of different definitions and concepts of affective states. We believe that the definitional morass can be separated from substantive scientific and technical questions by a strategy which involves exploring a variety of information processing architectures for various sorts of agents. The idea is to use agent architectures to (1) study families of concepts supported by each type of architecture and (2) explore the functional design tradeoffs between different architectures in various contexts.

Understanding the complex interplay of cognition and affect requires a close analysis of the properties of different information processing architectures and the states and processes they can support. We are pursuing various such analyses within the context of the "Cognition and Affect project" at the University of Birmingham (Sloman, 2000). In this paper,[1] we focus on one current track of the Cognition and Affect project, which studies the interaction of "affective" and "deliberative" behaviours in agent control.

2 Kinds of Architectures

We can view an agent as consisting of three main components (e.g., Russell and Norvig (1995)):

- the *agent program* implements a mapping from percepts to actions (this is sometimes called the action selection function or action composition).
- the *agent state* includes all the internal representations on which the agent program operates. This may include representations of the agent's environment and goals, the plans it has for achieving those goals, which parts of the plan have been executed and so on.
- the *agent architecture*, a (possibly virtual) machine that makes the percepts from the agent's sensors

[1] Presented at Symposium on Emotion, Cognition, and Affective Computing at the AISB'01 Convention, 21st - 24th March 2001

available to the agent program, runs the agent program, updates the agent state, and executes the primitive action(s) chosen by the agent program.

Our main concern is with the agent architecture. The architecture defines the atomic operations of the agent program, and implicitly defines the components of the agent. For example, *load* and *store* operations in a conventional CPU imply some sort of memory, otherwise the operations would not have the effect they are supposed to: calling *load* after *store* would not return the saved value. The architecture also determines which operations happen automatically without the agent program having to do anything, e.g., incrementing the program counter in a conventional CPU or production firing and conflict resolution in a rule-based system.

In practice, the distinction between agent program, state and architecture is often a matter of interpretation or convenience. In an implemented agent there are a whole hierarchy of virtual machines: the agent program is expressed in terms of the primitive operations provided by the architecture; the architecture is usually implemented in terms of a programming language, which in turn is implemented using the instruction set of a particular CPU (or another virtual machine such as the JVM). Likewise some "agent programs" together with their architecture can implement a new, higher-level architecture (virtual machine). In what follows, "agent architecture" used without qualification means the most abstract architecture or the highest level virtual machine.

The primitive operations supported by the architecture, together with the things that happen automatically, determine what kind of architecture it is, for example, whether an architecture is reactive or deliberative. In this paper we will focus on three kinds of agent (and hence on three kinds of architecture): reactive, affective and deliberative.

2.1 Reactive Architectures

A *reactive* architecture is one in which percepts directly trigger actions. The selection of which action(s) to perform is determined by the agent program. When more than one action is potentially appropriate in a given situation, the agent program must choose which of the possibly conflicting actions to perform. Actions which do not interfere can be executed in parallel (within the limitations of the underlying architecture, e.g., the number of degrees of freedom). However if the set of possible actions cannot be executed in parallel, either because of the limitations of the architecture or because the actions are logically inconsistent, the agent must either select the most appropriate subset of the actions to perform or synthesise a new action by combining the candidate actions into a single composite action e.g., simultaneously moving towards a goal while avoiding an obstacle—the obstacle avoidance modifies the motion towards the goal, by deflecting the path around the obstacle.

Reactive architectures may make use of simple representations of the state of the world and/or the agent, but these representations do not explicitly encode goals, hypothetical states of the world or sequences of possible actions. We may ascribe intentional states such as beliefs and desires to a reactive agent, but the agent architecture contains no explicit representation of these states. Rather such states supervene on the architecture. In such cases we can view the agent as an *intentional system*, that is, we ascribe to it the beliefs and goals it *ought* to have, given what we know of its environment, sensors and (putative) desires Dennett (1996). For example, an agent which has an 'avoid obstacles' behaviour, can be said to have a goal of 'avoiding collisions', even though this goal is not explicitly represented in the agent.[2]

Reactive architectures are often implemented in dedicated, parallel hardware using many simple components. The limited amount of processing necessary for a percept or set of percepts to trigger an action, the use of dedicated parallel hardware and the lack of complex representations means that reactive systems typically respond quickly to changes in the environment. Indeed the absence of complex internal representations often mandates the use of tight sensorimotor feedback loops with frequent sampling of the environment.

2.2 Affective Architectures

An *affective* architecture is one in which there are explicit representations of affective control states such as preferences, desires or emotions. Such states are directly encoded within the agent's state, e.g., in a connectionist unit, real-valued variable etc. rather than being supervenient on the architecture as in the case of a reactive agent. Note that this does not mean that *all* intentional states are explicitly represented in an affective architecture, for example, beliefs and goals may be supervenient. Nor does it mean that all affective states are directly represented in the architecture, only that some are. The fact that some affective states are explicitly represented within the architecture and do not merely supervene on it means that the architecture to monitor the achievement or non-achievement of such states, and allows them to take a role in learning, deliberation, the modification of reactive behaviours, etc..[3]

[2] Dennett calls this approach "adopting the intentional stance".

[3] Note that while supervening affective states can have the same behavioural potential as explicitly implemented affective states, their counterfactual potential with respect to architecture extensions is not the same: take two agents with "identical behavioural capacities", where in the first an affective states supervenes, and in the second the same state is part of the architectures (and trivially supervenes too). Then there are extensions of the latter that can make use of the state, whereas there no extensions of the former that could make use of the supervenient state unless the "add" mechanisms to monitor this supervenient state, which would effectively amount to changing the architecture to make this supervenient state "explicit".

2.3 Deliberative Architectures

A *deliberative* architecture in one in which there is some consideration of alternative courses of action before an action is taken.

A deliberative architecture is one in which at least some of the states are counterfactual in the sense of referring to hypothetical past or future states or as yet unexecuted actions (or sequences of such actions) and in which at least some of the basic operations of the architecture produce/read/write such counterfactual states. Such states include goals (descriptions of states to be achieved), plans (sequences of unexecuted actions), states describing the imagined consequences of performing an action in the current state or some hypothetical state, partial solutions generated during planning or problem solving, the hypothetical states of the agent's beliefs generated during belief revision and many others. We further require that such states should be influential in the production of actions, in the counterfactual sense that, had the (counterfactual) state not been generated, the agent would have chosen a different action to execute.

Note that this definition implies no commitments as to whether the states and operations are fine grained, e.g., dealing with partial plans or alternative solutions and their generation and comparison, or whether the states and operations are 'coarse grained', e.g., a single 'plan' operator which takes a goal and a description of the current state and returns a plan with the rest of the fine-grained states and operators buried in the implementation of the architecture and invisible to the agent program and the agent state. Both cases have at least one counterfactual state and one operator that takes a non-counterfactual state and returns a counterfactual state.

To represent counterfactual states, a deliberative agent requires representations with compositional semantics, in the sense that the meaning of the representations is a function of the meanings of their parts. It also implies a reusable working memory for the construction and comparison of hypothetical states and some means of deriving the consequences of actions performed in these states. At its simplest, this might be memories of the consequences of performing the action in similar states in the past. The use of a common working memory limits the number of alternative courses of action that can be considered in parallel, and hence the degree of parallelism possible within a deliberative architecture.

All other things being equal, a deliberative architecture must be slower and require more resources than a reactive architecture which encodes a solution to any specific goal solvable by the deliberative architecture, since the generation of alternatives must take time. However a deliberative architecture will typically be more space efficient than an equivalent reactive architecture, even though it will often require more space than a reactive solution to any given problem instance, since it can solve a *class* of problems in a fixed amount of space, whereas a reactive architecture requires space proportional to the number of problems.

We can view this as an example of the standard space-time tradeoff, though in this case there is also the time required to code or evolve all the reactive solutions. For example, to understand English sentences a reactive architecture needs to encode the meaning of every possible input sentence separately, whereas a deliberative system simply needs a grammar and a parser. The problem for the reactive approach is that there is an unbounded number of possible sentences (and choosing the potentially relevant ones might not be possible ahead of time).

Note that at a given level of abstraction, a component of an architecture cannot be both reactive and deliberative, since deliberation presupposes representational capabilities which by definition are missing from a reactive architecture. However, a given component can be both affective and deliberative, as we shall see.

3 Affective and Deliberative Agent Control

In many cases, the generative potential of deliberative capabilities opens up realms that are inaccessible to reactive agents (unless they have vast memories with pre-computed strategies for all possible eventualities), justifying their additional computational cost. However, there are cases where the same (if not better) results can be achieved using reactive systems augmented by simple affective states. Such trade-offs are not always obvious, and careful and detailed explorations in design space may be needed in order to find good designs to meet particular requirements.

In the following we compare (1) adding different types of deliberative extensions to a reactive architecture with (2) adding some simple states recording current needs, along with behaviours triggered by those states which modify the agent's reactive behaviours. Option (2) can be loosely described as adding primitive "affective" (or "emotional") states. In a number of experiments, we demonstrate that both approaches can have a powerful influence on an agent's ability to survive in dangerous multi-agent environments containing different kinds of agents, obstacles, food sources, and the like.

In the following, we focus on two main kinds of agents, the "affective agents" (A-agents) and "deliberative agents" (D-agents). A-agents have reactive mechanisms augmented by simple "affective states", whereas D-agents have representational and planning abilities in addition to the same reactive mechanisms.

3.1 The SimWorld Environment

The experiments were conducted in a simple artificial environment implemented using the SimAgent toolkit[4]. The simulation can run either in "display mode" or "batch

[4]See http://www.cs.bham.ac.uk/research/simagent/

mode". Display mode provides a graphical representation of the simulation and allows user interaction; batch mode dispenses with the display but allows the collection of statistical information during the runs. The display mode is intended to aid in the design of evolutionary experiments, which can then be run much faster in batch mode.

The simulated environment (the "world") consists of a rectangular surface of fixed size (usually around 600 by 600 units) populated with various kinds of objects:

- static obstacles (displayed as rectangles of varying size, usually around 10 by 10)
- moving obstacles (displayed as rectangles of varying size moving at particular speed in a particular direction without ever changing it)
- energy sources—"food items" (displayed as small circles that pop up at random locations within the world and stay there for a pre-determined period of time, after which they disappear unless consumed by agents)
- various kinds of agents (displayed as circles with a small square on the circumference and a text string indicating the direction the agent is heading in and its type, respectively)

The environment is continuous in the sense that the agents' positions are real-valued (rather than being confined to a grid). Agents can move in any direction (from 0 to 359 degrees, where 0 means "east"), and consume energy proportional to the speed at which they move. However, even when stationary, agents will still consume a certain amount of energy per timestep. Agents which run out of energy "die" and are removed from the simulation. They are also removed if they run into obstacles or other agents (in the latter case all agents involved in the collision will be removed). In the environments studied, agents typically die of hunger or as a result of collisions within 1000 timesteps, thereby obviating the need to limit their lifetime explicitly.

All agents are equipped with three kinds of exteroceptive sensors: sonar, smell and touch. In addition, some agents have a vision sensor, which allows them to gather information about the size and position of objects within their visual field.

Sonar is used to detect obstacles and other agents, smell to detect food, and touch to detect impending collisions. For sonar and smell, gradient vectors are computed pointing in the direction of obstacles and food within the respective sensor range. These vectors can then be combined in various ways and mapped onto the effector space, yielding a direction in which to move to avoid obstacles and/or move closer towards food.

The touch sensor is connected to a global alarm system, which triggers a reflex to move away from whatever the agent touches (unless it is food, which will be consumed). These movements will be initiated automatically and the agent cannot control them. They are also somewhat erratic and will slightly reorient the agent.

In addition to the three exteroceptive sensors, all agents also have two proprioceptive sensors, which measure their energy-level and their orientation, respectively (some have an additional orientation sensor which keeps track of their heading).

The agents also have a number of effectors: they have motors for locomotion (forward and backward), motors for turning (left and right in degrees) and a mechanism for consuming food. Agents need to sit on top of a food source in order to be able to consume it. Consuming food takes time proportional to the energy stored in the food source and the maximum amount of energy an agent can extract in a timestep.

After a certain number of simulation cycles, agents reach maturity and can reproduce asexually. The number of offspring produced depends on the energy level of the "parent", and the offspring are created in the immediate vicinity of the parent (temporarily increasing the local competition for resources and increasing the likelihood of collisions). The energy necessary to create each new agent is subtracted from the parent.

Before a run of the simulation, which can typically take anywhere from 10,000 to 1,000,000 simulation update steps, various parameters of the environment must be specified, including:

- the size of the world
- the number and sizes of stationary obstacles,
- the number, sizes, speeds and directions of moving obstacles,
- the number of energy sources together with their energy capacities, frequency of appearance, and life time

For agents at the least the following parameters need to be set:

- the respective sensor ranges for sonar, smell, and touch
- the maximum food intake per time step
- the procreation age and the energy expenditure for each offspring
- the maximum speed of movement and the energy expenditure for it
- the different concurrently active modules making up the agent's cognitive system and their speed of execution relative to a simulation update step

Usually, agents, obstacles and food are placed at random locations in the environment to be able to "average out" possible advantages due to their location over a large number of trials. However, it is also possible to fix locations in advance, e.g., to study how different kinds of agents would fare in the same situation.

3.2 The Agents ...

While different kinds of agents may have different short term goals at any given time (e.g., getting around an obstacle or avoiding a predator), common to all of them is the implicit goal of survival and procreation, i.e., to get (enough) food and avoid getting killed (i.e., run into/get run over by an obstacle/other agent) to be able to live long enough to have offspring.

In the following we will consider various different kinds of agents, which differ solely with respect to their architecture:

1. reactive agents (R-agents)
2. (simple) affective agents (A-agents)
3. pseudo-deliberative agents (PD-agents)
4. (advanced) deliberative agents (D-agents)
5. combined affective and pseudo-deliberative agents (PC-agents)
6. combined affective and (advanced) deliberative agents (C-agents)

These reflect two different kinds of extensions of a basic reactive architecture: (1) the addition of primitive affective states and (2) the addition of primitive and advanced deliberative capabilities. Each agent has the reactive mechanisms of R-agents. A-agents extend R-agents by simple affective states such as "hunger", "fear", "persistence", "caution", etc. (still located within the reactive layer). PD-agents extend R-agents by a simple planning and plan execution mechanism (i.e., by a rudimentary deliberative layer), whereas D-agents are genuine deliberative agents with complex representational and planning capacities (as explained below). The combined PC- and C- agents integrate the capabilities of PD- and A-agents and D- and A-agents, respectively.

The reactive layer of R-agents (which is common to all other agents as well), is based on augmented finite state machines, which run in parallel and can influence each other (related to the style of Brooks' subsumption architecture, e.g., see (Brooks, 1986)). The finite state machines process sensor information and produce behavioural responses using a schema-based approach (in SimAgent these finite state machines are realized as rule systems). The reactive behaviours take sensor information and compute a sensor vector field for each sensor (i.e., the simulated equivalents of a sonar and a smell sensor), which are then combined and transformed into the agent's motor space (e.g., see Arkin (1989)). The transformation function mapping sensory to motor space is given by $\delta S + \gamma F$ (where 'S' and 'F' are the sonar and food vector fields and δ and γ the respective gain values).[5].

[5]Note that this formula leaves out many details, such as the mappings for the "touch" sensor, for ease of presentation.

A-agents differ from R-agents in that they possess "inner" states which can influence the way in which sensory vector fields are combined: these states alter the gain values of the perceptual schemas in the transformation function mapping sensory to motor space (e.g., see Arkin (1998)). Thus the very same sensory data can get mapped onto different motor commands depending on the affective state. For example, a primitive "fear" state could modify the gain value of the obstacle vector and thus the degree to which the agent will be repelled by obstacles: an agent, which is less "afraid", will have a lower gain values than an agent which is very afraid, resulting in different locomotion behaviour in affective agents. In our experiments, we used A-agents with a single "hunger" state, which modifies the gain value of the "food" vector: if hunger is low, the gain value for hunger is slightly negative and the agents tend to move away from food (possibly corresponding to the feeling of being repelled by food one has if one has eaten too much).

PD-agents, on the other hand, possess an additional primitive deliberative layer, which allows them to produce a "detour plan" when their path to food is blocked (by an obstacle, predator, or any other agent). The plan is a sequence of motor commands, which override those given by the reactive mechanisms. To be more precise, a PD-agent uses explicit representations of the food and obstacle vectors to compute a trajectory to the food which avoids the obstacles. Once a decision has been reached, PD-agents start moving to points on the trajectory, suppressing the influence from the food schema on the overall combined behaviour completely until plan execution is completed. An "alarm" system interrupts plan execution if a PD-agent comes too close to an obstacle and triggers replanning, in which case the agent will attempt to make a more extensive detour. Once the execution of a plan is finished, the agent uses its reactive mechanisms to move towards food, which should now not be obstructed, unless the world has changed (e.g., the obstacle was not static).

D-agents extend PD agents in various ways. First, they have a vision sensor, which they use to spot obstacles and food (PD-agents, on the other hand, need to "extract" obstacle and food locations from the force vectors of the respective vector field, which is only possible to a very rough degree). Second, they are able to remember the location of obstacles and food they have encountered relative to their current position (i.e., in an agent-centric polar coordinate system).[6] They have mechanisms to update their internal representations of food and obstacles when they move so as to adjust the relative angles and distances according to their movements. They also possess a coherency mechanism, which deletes a memorized item if it does not agree with what is being perceived (e.g., if the agent expects a food item to be in a particular loca-

[6]In the current implementation agents never "forget" anything they have committed to memory, but it is possible to associate a "decay"-rate to items in memory to simulate "forgetting", so that after a certain time the item will be erased from memory.

tion in visual field, but no food item can be found in this area, the agent will erase the item from memory).

Third, D-agents have a simple route planning mechanism which allows them to find a route to the nearest food item, avoiding obstacles. The planner is given a list of obstacles and food items known to the agent, and returns a plan to the nearest reachable food item.[7] The plan is a list of headings and distances and is executed by the underlying motor behaviours of the agent.

Planning is triggered by the alarm mechanism in response to an imminent collision with an obstacle. A collision is considered to be "imminent" if the obstacle is within a predefined "imminent collision range" and the agent is facing the obstacle (within $+/-$ 60 degrees of the current heading). The imminent collision range is relatively large, and it possible for the agent to get well inside the collision region before actually colliding with or even noticing the obstacle. For example, if the agent enters the collision region from the "side" (not directly facing the obstacle) and then turns towards the obstacle, the alarm will be triggered. As a result, the planner has to be capable of producing plans which take the agent out of the collision region without re-triggering the alarm mechanism. This is an example of the issues that arise in integrating the continuous (i.e., real-valued), gradient-based, relatively imprecise reactive behaviours of the agent with the discrete representation used by the planner.

The planner uses a discrete model of the environment with relatively large plan steps, giving a coarse grained grid representation centred on the agent. Plans are constructed to the nearest grid point to the goal, at which point the reactive behaviours of the agent take over to guide it to the food item. There are eight operators which allow the agent to reach the eight adjacent grid cells from the current cell. Operators are disallowed if the resulting plan step would take the agent outside the environment or outside a "planning region" which constrains the distance to the farthest point on the plan to be no greater than a multiple of the distance from the start point to the goal. In practice, we have found a planning region with a radius of 2.5 times the distance to the goal to be sufficient.

The planner is based on a simplified version of the A^*_ϵ algorithm Pearl (1982). A^*_ϵ is a variant of A^* in which the cost of the solution returned is guaranteed to be no greater than $1 + \epsilon\times$ the cost of the optimum solution. A^*_ϵ is a good choice for a route planning agent as all we need are good (rather than optimal) plans. The cost of a plan is the distance the agent has to travel to reach the goal, with a penalty for routes which pass through the collision region around an obstacle. There is a very steep cost gradient in the vicinity of obstacles, which means that the first step of any plan which starts in a collision region will be away from the obstacle. This re-orients the agent, so that it is no longer facing the obstacle and prevents the alarm mechanism being triggered again on the next cycle.

PC- and C-agents combine the capabilities of A- and PD-agents and A- and D-agents respectively. PC-agents combine the simple affective state of A-agents with primitive deliberation of PD-agents. C-agents combine the simple affective state of A-agents with the more advanced representational and deliberative capabilities of D-agents.

3.3 ... and Their Resultant Behaviours

As one would expect, the differences in the architecture give rise to different behaviour of the agents: R-agents are always interested in food and go for whichever food source is nearest to them (often manoeuvring themselves into fatal situations). They can be described as "greedy". Similarly, PD-agents are also always interested in food, yet they attempt to navigate around obstacles and predators using their (limited) planning capacity though constantly driven by their "greed". Although their deliberative abilities make good use of all locally available information, this can have the consequence that the agent ends up too far from food and starves in situations where it would have been better to do nothing for a short period of time. By then the obstructing obstacles and predators might no longer be blocking the direct route to food. PD-agents (like R-agents) constantly move close to danger in their attempts to get to food, and can therefore die for food which they do not yet really need.

A-agents, on the other hand, are only interested in food when their energy levels are low (i.e., they are not constantly "greedy", and seek food only when "hungry"). When they are "hungry", they behave like R-agents in that they chase down every food source available to them. However, their route around obstacles is dependent on their "hunger level": when they are less hungry, the repulsive effect of an obstacle will have a greater effect on their route. Otherwise they tend to avoid food and thus competition for it, which reduces the likelihood of getting killed because of colliding with other competing agents or predators.

Finally, PC-agents, behave like PD-agents as far as their maneuvers are concerned, but like A-agents with respect to food in that they will not navigate towards food if they are not hungry.

Finally, D- and C-agents are similar to PD- and PC-agents respectively in their overall behaviour, except that their planning mechanism is superior and often leads them to food in a very efficient way.

4 Experiments

We have conducted various experiments to compare the performance of the different kinds of agents. Before being able to compare advantages and disadvantages of agents

[7] Some food items are too close to an obstacle to be reachable by the agent, however the reactive behaviours used by the all agents will persist in trying to reach the food. In such cases the planner can be useful both in finding a route to a reachable food item, and in moving the D-agent out of the local minimum represented by the unreachable food item, into an area where the food items are (hopefully) reachable via reactive behaviours.

in multi-agent environments with different kinds of agents, it is necessary to check whether any given agent kind can survive as a group in an environment on its own. This result can be taken to be a yard-stick against which one can measure their performance in environments where they have to compete with other kinds of agents. For the following experiments, we fix the "food rate" at 0.25, i.e., new food will appear on every forth environmental update on average. Furthermore, we fix the procreation age for all agents at 250 updates.

4.1 Preliminary Experiments

The preliminary experiments, where groups of 5 agents of one kind were placed in the environment at random locations, show that each of the tested agent kinds can survive in the long run in various kinds of environments, from environments with no obstacles to very "dangerous" environments with many obstacles.

Table 1 shows for R-, A-, PD-, and PC-agents the average (μ) number of surviving agents of that kind taken over 10 different runs of the simulation, each for 10000 environmental updates for a given environment (where "(n,k)-env" is intended to indicate that n static and k moving obstacles were placed at random in the environment). In addition, the standard deviation (σ) and the confidence interval for $\alpha = 0.05$ (Con) are given too.

Given that each agent kind can survive on its own (although with different success), we are now interested in comparing the performance of various A-, PD- and D-agents in "mixed environments" (i.e., environments that contain more than one agent kind). The first set of experiments is concerned with the performance of A-agents as compared to R-, PD-, or PC-agents. Only if A-agents prove superior in a wide-range of environments is it necessary to compare them with more complex deliberative agents (like the D- or C-agents).

4.2 Series 1: Affective vs. Primitive Deliberative Control

Surprisingly, we found that A-agents reliably outperform not only R-agents, but also PD- and PC-agents in all the environments considered above (the food rate is again 0.25) if n agents of each kind competing against each other for $n = 3, 5, 8$ (see Table 2, Table 3, and Table 4)[8].

The above experiments show that regardless of the initial distribution, the number of agents in the environment and the number of moving and static obstacles, we get the following "ranking" (from best to worst):

1. A-agents
2. PC-agents
3. R-agents
4. PD-agents

This ranking can be shown independently by experiments in which only 2 kinds of agents are placed in different environments initially. We have done these experiments for all six combinations in all seven environments and observed the same results: A-agents outperform all other agents, PC-agents outperform R- and PD-agents, and finally R-agents beat PD-agents.

The reason why PD-agents perform worse than R-agents is that they almost never win a "duel" for food with R-agents, as their deliberative mechanism does not distinguish between obstacles and agents, hence they even make detour plans if competing for a food source with an R-agent. In such a case, they R-agent will get the food while the PD-agent attempts to get "around" the R-agent. PD-agents that can discriminate between obstacles and other agents perform as well as R-agents. It is also worth pointing out that PD-agents usually die out of hunger, whereas R-agents more often die because of collisions (in particular in crowded environments).

It is not surprising that PC-agents perform better than R-agents. Particularly in less crowded environments, PC-agents can make use of their affective states in the same way as A-agents, as their deliberative mechanisms do not get activated all that often. Hence, we find that there is a good chance that some PC-agents will survive. In more crowded environments, however, this advantage disappears and the disadvantages of deliberative detour plans outweighs the advantage of avoiding competition using the affective hunger state. When the PC-agents get "hungry" enough to seek food in an environment with lots of obstacles and other agents, their primitive deliberative capabilities are frequently triggered by proximity to obstacles or other agents. The resulting "detour plans" can sometimes lead them farther and farther away from food resulting in starvation.

In addition, the experiments show that 20,20 and 30,30-environment seem to be the ones where other agent kinds stand the best chance against A-agents. This is because these environments are crowded enough to make it more difficult for A-agents to get food, while still not too crowded for the other agents to mainly die because of collisions (as happens in 40,40 and 50,50 environments; the latter already becomes challenging for A-agents as well).

If the food rate is varied, then we find that higher food rates (e.g., a food rate of 0.5) do not change the picture, rather they show even more clearly the ability of affective agents to coexist in large groups. On the other hand, lower food rates (in the range 0.125–0.25) make survival in crowded environments impossible, as there are simply too many obstacles obstructing the paths to food. With these low food rates the advantage of A-agents over R-agents slowly disappears as waiting for hunger to grow

[8] The rationale for choosing these numbers is that with more than 8 agents, the environment is too crowded and, on average, the same number of agents of each kind will die in collisions, thus reducing the overall number very quickly. Having fewer than 3 agents distorts the statistics as the results are very sensitive to the initial (random) positions of the agents.

Table 1: The average number of surviving agents in an n,k-environment when started with 5 agents of only one kind.

	R-agents			A-agents			PD-agents			PC-agents		
Env	μ	σ	Con	μ	σ	Con	μ	σ	Con	μ	σ	Con
0,0	14.60	2.80	1.73	19.20	2.74	1.70	16.70	3.09	1.92	18.00	3.62	2.24
5,5	13.20	4.78	2.96	17.20	3.05	1.89	13.60	2.07	1.28	16.30	2.58	1.60
10,10	11.90	3.81	2.36	17.20	3.77	2.33	12.80	3.85	2.39	16.10	1.79	1.11
20,20	11.60	3.47	2.15	15.40	3.95	2.45	8.00	3.89	2.41	14.80	4.64	2.87
30,30	7.50	4.43	2.75	13.00	3.56	2.21	4.30	4.37	2.71	10.50	3.44	2.13
40,40	2.90	3.57	2.21	10.40	3.57	2.21	0.60	1.90	1.18	7.70	4.88	3.02
50,50	0.20	0.63	0.39	8.00	3.56	2.21	0.00	0.00	0.00	1.00	1.94	1.20

Table 2: The average number of surviving agents in an n,k-environment when started with 3 agents of each kind.

	R-agents			A-agents			PD-agents			PC-agents		
Env	μ	σ	Con	μ	σ	Con	μ	σ	Con	μ	σ	Con
0,0	1.10	3.48	2.16	14.40	5.87	3.64	0.00	0.00	0.00	1.40	2.37	1.47
5,5	1.70	3.95	2.45	14.40	5.40	3.35	0.00	0.00	0.00	1.40	3.27	2.03
10,10	0.00	0.00	0.00	16.00	2.18	1.35	0.00	0.00	0.00	0.00	0.00	0.00
20,20	0.00	0.00	0.00	14.80	4.18	2.59	0.00	0.00	0.00	0.60	1.90	1.18
30,30	1.00	3.16	1.96	10.90	7.88	4.88	0.00	0.00	0.00	3.10	6.74	4.18
40,40	0.00	0.00	0.00	11.30	3.09	1.92	0.00	0.00	0.00	0.00	0.00	0.00
50,50	0.00	0.00	0.00	4.92	5.32	3.30	0.00	0.00	0.00	0.00	0.00	0.00

before moving towards food is not a good strategy (if missing out on one food source could be fatal).

4.3 Series 2: Affective vs. Advanced Deliberative Control

Given the superior performance of A-agents (even with different food rates), we were particularly interested in comparing them to advanced deliberative agents. D- and C-agents have much more processing power in addition to another very powerful visual sensory organ, so it is clear that these two kinds are not on a par with A-agents. However, if the goal is to discover the limits of affective control, it seems only fair to employ more powerful mechanisms to test the waters.

The setup for the following experiments is identical to the previous ones, except that in the second series only static obstacles were used. The reason for this restriction is that D-agents do not have a tracking mechanism for moving obstacles, and hence would wrongly classify moving obstacles as "static", very much to their disadvantage.[9]

We found that, depending on the environment, A-agents still do well in competition with D-agents. In environments with very few static obstacles (up to 10), D-agents rarely plan, as their alarm mechanism is only triggered by obstacles, not other agents. In these environments, therefore, D-agents behave like R-agents. And since A-agents beat R-agents in such environments, it does not come as a surprise that they beat D-agents. In environments with a large number of obstacles, the compound obstacle vectors used by the A-agents become uninformative, and A-agents must negotiate their way around obstacles by trial and error, increasing the distance they have to travel to food and the likelihood of collisions. In these situations, the D-agent's ability to plan routes around obstacles that lead them directly to food pays off (see the first two columns of Table 5).

The comparison between A- and C-agents is even more interesting, and produced some results that we did not expect. *Prima facie* it seems that C-agents should have an advantage over both A-agents and D-agents, since they inherit the capabilities of both. However, behavioural properties of parts of an agent architecture do not simply "add up": while C-agents have about the same performance as A-agents for environments with very few obstacles (up to 10), in medium-obstacle environments (over 10 and less than 40) they perform worse than both A-agents and D-agents (which beat A-agents in these environments). Only in more crowded environments (over 40), do C-agents perform better than A-agents. The reasons for this unexpected "weakness" of C-agents are quite interesting.

In low-obstacle environments the affective control of C-agents is in command most of the time and deliberative control is rarely used (hence their similar performance to A-agents in these environments). In high-obstacle envi-

[9] In a second group of experiments, we added an additional perceptual mechanism to D- and C-agents which allows them to distinguish between static and moving obstacles, so that only static obstacles are entered in the agents' map. As expected, D- and C-agents did worse than most other agents in n,n-environments, since many of their plans are based on wrong assumptions about the environment, and hence do not improve their ability to get to food. We are currently working on an extension of the D-agents that can–to a limited extent–track moving obstacles.

Table 3: The average number of surviving agents in an n,k-environment when started with 5 agents of each kind.

	R-agents			A-agents			PD-agents			PC-agents		
Env	μ	σ	Con	μ	σ	Con	μ	σ	Con	μ	σ	Con
0,0	0.00	0.00	0.00	12.30	7.32	4.54	0.00	0.00	0.00	5.30	7.63	4.73
5,5	0.00	0.00	0.00	14.70	7.06	4.37	0.00	0.00	0.00	2.50	5.76	3.57
10,10	0.00	0.00	0.00	14.40	5.66	4.42	0.00	0.00	0.00	5.66	3.51	2.74
20,20	0.00	0.00	0.00	15.40	6.19	3.83	0.00	0.00	0.00	2.10	5.13	3.18
30,30	0.20	1.10	0.68	10.63	5.59	3.47	0.27	1.46	0.91	0.77	2.43	1.51
40,40	0.00	0.00	0.00	11.20	6.03	3.74	0.00	0.00	0.00	0.00	0.00	0.00
50,50	0.00	0.00	0.00	7.60	5.23	3.24	0.00	0.00	0.00	0.00	0.00	0.00

Table 4: The average number of surviving agents in an n,k-environment when started with 8 agents of each kind.

	R-agents			A-agents			PD-agents			PC-agents		
Env	μ	σ	Con	μ	σ	Con	μ	σ	Con	μ	σ	Con
0,0	0.70	1.64	1.01	14.00	7.16	4.44	0.00	0.00	0.00	3.00	6.13	3.80
5,5	0.00	0.00	0.00	16.00	3.29	2.04	0.00	0.00	0.00	0.00	0.00	0.00
10,10	0.00	0.00	0.00	14.60	2.99	1.85	0.00	0.00	0.00	0.50	1.58	0.98
20,20	0.20	0.63	0.39	15.60	2.22	1.38	0.00	0.00	0.00	0.00	0.00	0.00
30,30	0.30	0.95	0.59	13.60	5.10	3.16	0.00	0.00	0.00	0.00	0.00	0.00
40,40	0.00	0.00	0.00	8.40	6.60	4.09	0.00	0.00	0.00	1.30	2.75	1.70
50,50	0.00	0.00	0.00	9.00	5.89	3.65	0.00	0.00	0.00	0.30	0.95	0.59

ronments the alarm is triggered very often, hence planning is active most of the time and a C-agent uses its deliberative mechanism to move towards food rather than its reactive "scent following", which is less efficient–the performance of the C-agent here is similar to that of the D-agent.

However, in medium-obstacle environments the affective and deliberative control do not complement each other, but rather "compete" with each other in such a way that the resultant behaviour is of no advantage to the C-agent. As with A-agents, when the energy level is high, the influence of the affective hunger state keeps the agent from approaching food aggressively. As the agent's energy level falls, it will eventually move towards food. However in C-agents, the "small detour" imposed by the planning system in order to move the agent safely around obstacles is less efficient than the straightforward reactive control of A-agents, which follow smell gradients (like D-agents, C-agents are required to stay further than the imminent collision range from the obstacle, rather than just not hit it as with A-agents). In medium obstacle environments, A-agents have few alarms/collisions, since the sonar vectors give relatively good information about the location of obstacles in uncluttered environments. Only if an A-agent's "desire" for food is very strong, can the vector gradients lead the agent too close to obstacles, thereby triggering the alarm. Otherwise, the A-agent will be able to manoeuvre around obstacles without bumping into them, with the result that they are faster than C-agents. In medium obstacle environments, the D-agents compensate for the excessive caution of their plans by aggressively seeking food, rather than waiting until they are hungry. This early start makes up for the "small detour" imposed by following the plan, to the extent that they outperform A-agents (and hence C-agents) in these environments.

These results show that the integration of control mechanisms which are advantageous in different environments can lead to new weaknesses which are difficult to predict from the behavioural descriptions of the individual mechanisms. And while such extensions, which can be viewed as "specializations" from an evolutionary point of view, might lead to better adapted individuals for some environments, it can "backfire" and reduce the individual's fitness for others. Hence, evolutionary trajectories that lead to such integration of different mechanisms will have to take place in special environments.

5 Discussion

Our experimental studies have shown that the success of the agents as measured by their ability to survive depends on various environmental parameters. In some environments, A-agents are more likely to survive for a given time period than D-agents, while in other environments the D-agents are more likely to survive. We can clearly see that the affective states that guide A-agents are powerful control mechanisms, which allow large groups of A-agents to coexist in certain environments (as they reduce the competition for food). The advantages of such mechanisms, however, are outweighed by the disadvantage of not being able to navigate efficiently around obstacles in crowded environments.

We are currently investigating different affective states,

Table 5: The average number of surviving agents in an n,0-environment when started with 5 A-agents and 5 D-agents, and 5 A-agents and 5-C agents, respectively, for a food rate of 0.125

	A-agents			D-agents			A-agents			C-agents		
Env	μ	σ	Con	μ	σ	Con	μ	σ	Con	μ	σ	Con
0,0	13.00	2.31	1.43	3.80	5.29	0.00	10.30	6.48	4.02	7.20	4.96	3.08
5,0	12.30	7.89	4.89	5.00	6.45	4.00	5.90	7.62	4.72	11.40	9.70	6.01
10,0	9.90	7.29	4.52	6.10	6.97	4.32	8.90	7.23	4.48	8.00	6.67	4.13
20,0	6.20	7.22	4.48	8.20	6.92	4.29	9.80	4.32	2.67	2.70	4.37	2.71
30,0	4.50	5.08	3.15	7.20	6.36	3.94	9.40	5.93	3.67	3.40	4.25	2.63
40,0	3.20	5.16	3.20	6.40	4.81	2.98	3.30	4.35	2.96	6.70	5.72	3.54
50,0	0.50	1.58	0.98	8.10	5.47	3.39	3.60	4.74	2.94	3.60	4.06	2.52
60,0	0.70	2.21	1.37	3.20	4.32	2.67	1.20	2.57	1.59	3.20	3.74	2.32
70,0	0.00	0.00	0.00	1.20	2.57	1.59	0.00	0.00	0.00	1.10	2.02	1.25
80,0	0.00	0.00	0.00	0.70	1.64	1.01	0.00	0.00	0.00	0.40	0.84	0.52

such as a "higher order" affective state that measures the frequency of alarm triggerings within a given time interval and allows agents to "retreat" from what this mechanism implicitly assumes to be a "dangerous" area. Furthermore, A-agents with the ability to distinguish what causes their alarm mechanism to be activated could have two such states, one for "competition among agents", and one for "area crowded with obstacles". Preliminary experiments, however, show that while there might be some advantage for such extended A-agents over regular A-agents, this advantage does not outweigh that of being able to produce "safe routes to food" as in the case of the D- or C-agents.

So the question remains whether planning (such as realised in D-agents), and thus deliberative control, does indeed mark a significant evolutionary improvement over mere "affective" control in more than a few special environments. Obviously, more experiments are needed to confirm such a conjecture.

We believe that the above theoretical and experimental studies are a viable strategy to reach an understanding of the role that affective processes play in deliberation. In particular, we are convinced that it will be relevant to understanding evolutionary trajectories from reactive to deliberative organisms.

Acknowledgements

This work was partly funded by a grant from the Leverhulme trust.

References

R. C. Arkin. Motor schema-based mobile robot navigation. *International Journal of Robotic Research*, 8:92–112, 1989.

R. C. Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge, MA, 1998.

R. A. Brooks. Achieving Artificial Intelligence through building robots. AI Memo 899, MIT Artificial Intelligence Laboratory, May 1986.

A. R. Damasio. *Descartes' Error, Emotion Reason and the Human Brain*. Grosset/Putnam Books, 1994.

D. C. Dennett. *Kinds of Minds: Towards an understanding of consciousness*. Basic Books, 1996.

G. Hatano, N. Okada, and H. Tanabe, editors. *Affective Minds*, Amsterdam, 2000. Elsevier.

J. Pearl. A^*_ϵ — an algorithm using search effort estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(4):392–399, 1982.

R. Picard. *Affective Computing*. MIT Press, Cambridge, Mass, London, England, 1997.

W. S. N. Reilly. *Believable Social and Emotional Agents*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA,, May 1996. Technical Report CMU-CS-96-138.

S. Russell and P. Norvig. *Artificial Intelligence: a modern approach*. Prentice Hall, 1995.

H. A. Simon. Motivational and emotional controls of cognition, 1967. Reprinted in *Models of Thought,* Yale University Press, 29–38, 1979.

A. Sloman and M. Croucher. Why robots will have emotions. In *Proceedings of the 7th International Joint Conference on AI*, pages 197–202, Vancouver, 1981.

A. Sloman. Architecture-based conceptions of mind. In *Proceedings 11th International Congress of Logic, Methodology and Philosophy of Science*, Synthese Library Series, Dordrecht, 2000. Kluwer. (to appear).