Exploration Strategies for Homeostatic Agents

Patrick Andersson(✉), Anton Strandman, and Claes Strannegård

Department of Computer Science and Engineering,
Chalmers University of Technology, Gothenburg, Sweden
andersson.patrick95@gmail.com

Abstract. This paper evaluates two new strategies for investigating artificial animals called *animats*. Animats are homeostatic agents with the objective of keeping their internal variables as close to optimal as possible. Steps towards the optimal are rewarded and steps away punished. Using reinforcement learning for exploration and decision making, the animats can consider predetermined optimal/acceptable levels in light of current levels, giving them greater flexibility for exploration and better survival chances. This paper considers the resulting strategies as evaluated in a range of environments, showing them to outperform common reinforcement learning, where internal variables are not taken into consideration.

Keywords: Artificial general intelligence · Multi-objective reinforcement learning · Exploration strategies · Homeostatic regulation · Animat

1 Introduction

One way to approach *artificial general intelligence* (AGI) is to construct artificial animals, or *animats*, endowed with sensors and motors: i.e., sensorimotor capabilities. These agents act to satisfy predefined needs [11]. By simulation of more and more complex agents and environments, one can hope to obtain progressively better insight into the general nature of intelligence [12].

The notion of a homeostatic agent comes from physiology [2]. Through homeostasis, the agent attempts to minimize deviation of some number of key internal variables from their optimal levels [1], thus satisfying basic "needs"; for many animals, these are things like food and water. All living organisms are homeostatic agents. That said, the concept of homeostasis can be extended in careful, limited fashion to non-living systems that require balancing of multiple parameters: e.g., a hydroelectric power plant that "needs" to balance energy generation with water levels.

Research supported by the Torsten Söderberg Foundation Ö110/17.

P. Hammer et al. (Eds.): AGI 2019, LNAI 11654, pp. 178–187, 2019.
https://doi.org/10.1007/978-3-030-27005-6_18

While the notion of a homeostatic animat addresses the "what" and "why" of intelligent behaviour, it provides no clear answers concerning the "how". Reinforcement learning (RL) fills this gap by providing a framework for sequential decision making. RL maps states (sensory perceptions) to actions (motor actions) via the feedback of a reward signal.

A crucial task for agents operating in any environment is to balance exploration with exploitation. This paper evaluates exploration strategies that use internal variables to help agents survive longer by responding more robustly to parametric changes: i.e., adapting dynamically to the environment.

Section 2 describes how the homeostatic agent is modeled and how its exploration strategies take advantage of changes in its internal variables. Section 3 explains how those strategies are best evaluated. Finally, Sect. 5 discusses the results to conclude that RL as traditionally carried out can, indeed, be improved upon.

2 Homeostatic Agents

While the techniques used for RL share many similarities with those used for designing animats – notably those for observation and control – there is one key difference. The homeostatic animat must satisfy its needs and balance its internal variables by considering multiple objectives simultaneously. RL as traditionally conceived is only concerned with one objective: optimizing a single scalar return [10]. The problem of balancing internal variables must be imported into RL.

A homeostatic agent can be modeled in many ways [3,5–7,9]. For present purposes, two things are key: determining a scalar reward from the internal variables and, consequently, giving the agent access to those variables. This can be accomplished with a simple computational model known as *homeostatic drive reduction* [3].

First, one must convert the homeostatic problem into a single objective maximization problem appropriate for RL. Then, one must consider how the information gained from the internal variables can be applied to an exploration strategy.

2.1 Homeostatic RL

In our homeostatic agent, internal variables $\mathbf{h}$ are scaled $-1 \leq h_i \leq 1$ with optimal value $h_i^* = 0$. They are applied in three ways. First, they are fed into the drive-reduction framework [3] to calculate drive using Eq. 1. The "reward" is based on the change in drive at each time step, as seen in Eq. 2. This reward is what the reinforcement learning algorithm tries to maximize.

$$d(\mathbf{h}_t) = \sqrt[m]{\sum_{i=1}^{N} |h_i^* - h_{i,t}|^n} \tag{1}$$

$$r_t = d(\mathbf{h}_t) - d(\mathbf{h}_{t+1}) = d(\mathbf{h}_t) - d(\mathbf{h}_t + \mathbf{k}_t) \tag{2}$$

Vector **k** represents the outcome at a given time step: i.e., the change that has occurred in the internal variables. Hyper-parameters $n > m > 1$ push drive away from the optimum. By analogy, eating when hungry is more rewarding than eating when full. This creates an incentive to fulfill needs that lie further from the optimum, as a change in these needs will have a larger effect on reward.

Second, the internal variables are used as input to the RL algorithm along with observations from the environment. This increases the state space by enabling the algorithm to be "aware" of the agent's internal state. We chose to use a deep Q network (DQN) architecture [8]. Within the drive reduction framework, the Q-values for each action depend on internal state and external observations. An action indicating "eat" should have higher Q-value if the agent is hungry.

Third, the internal variables are fed into the agent's exploration strategy, becoming a part of the action decision process: the focus of this paper.

The resulting transformation of the traditional RL agent can be seen in Fig. 1. Note that the animat receives no explicit reward from its environment; instead, reward is calculated using drive reduction on the internal variables, which are perturbed but not determined by the environment.

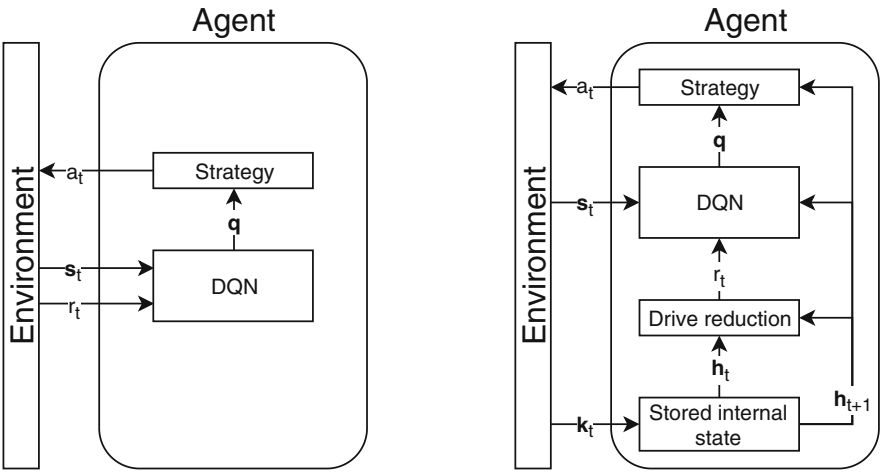

Fig. 1. Diagram visualizing the difference between a traditional RL agent, left, and a multi-objective homeostatic RL agent, right.

2.2 Homeostatic Exploration

The typical exploration strategy used in RL is ϵ-greedy [10]. It takes a random action with probability ϵ or, otherwise, the action with highest Q-value. It is typically implemented as annealed ϵ-greedy, where ϵ decays over time toward some lower bound. While this strategy has proven successful in numerous settings, it has drawbacks – the most obvious being that it is often a function of t.

That requires the hyper-parameters to be manually tuned for every environment to find the right amount of exploration. If the environment changes, the agent requires retraining with human intervention. Ideally, the agent should be capable of adapting – retraining itself – without such intervention.

By contrast, we apply a simple heuristic to the agent's internal variables to create two exploration strategies: *explore when good* (EWG) and *explore when bad* (EWB). These are dynamic strategies capable of balancing exploration with exploitation continuously, based on how the agent performs over the course of its lifetime. They operate similarly to ϵ-greedy but calculate ϵ's value at each time step based on Eqs. 3 and 4, where θ is the hyper-parameter *threshold*. The values of ϵ are represented for the agent by two internal variables, as shown in Fig. 2. It should be clear that, for EWG, the value of ϵ is highest when both variables are near optimal, whilst for EWB it is highest when they are furthest away.

$$\text{EWG:} \quad \epsilon = \max\left(0, 1 + \frac{\max_i |h_i|}{\theta - 1}\right) \tag{3}$$

$$\text{EWB:} \quad \epsilon = \max\left(0, \frac{\theta - \max_i |h_i|}{\theta - 1}\right) \tag{4}$$

Threshold parameter $0 \leq \theta < 1$ defines an interval for which the agent should be maximally greedy, selecting only the best possible actions for itself. For EWG, this threshold is set near the limit of the internal variables' acceptable levels, while for EWB it is set near the optimum. In consequence, EWG will stop exploring when any internal variable approaches the acceptable limits, while EWB will only start exploring when one or more variables are no longer close to optimum. These strategies are not dependent on time but instead change the rate of exploration dynamically, moment by moment, based on how the agent is performing in relation to optimal levels of its internal variables.

3 Method

The EWG and EWB strategies are compared to annealed ϵ and constant ϵ in a number of environments designed to evaluate various properties of the strategies. The agent balances two internal variables both of which decay when an action is performed, simulating energy usage. The variables increase or decrease according to actions in certain states. In all, thirteen environments are used, at three levels of complexity.

3.1 Environments

First are the simple worlds having no observable external state, though the agent still observes its internal state: "Greed", "Random" and "Dual". "Greed" is a static world where action outcomes are predetermined: i.e., all actions modify

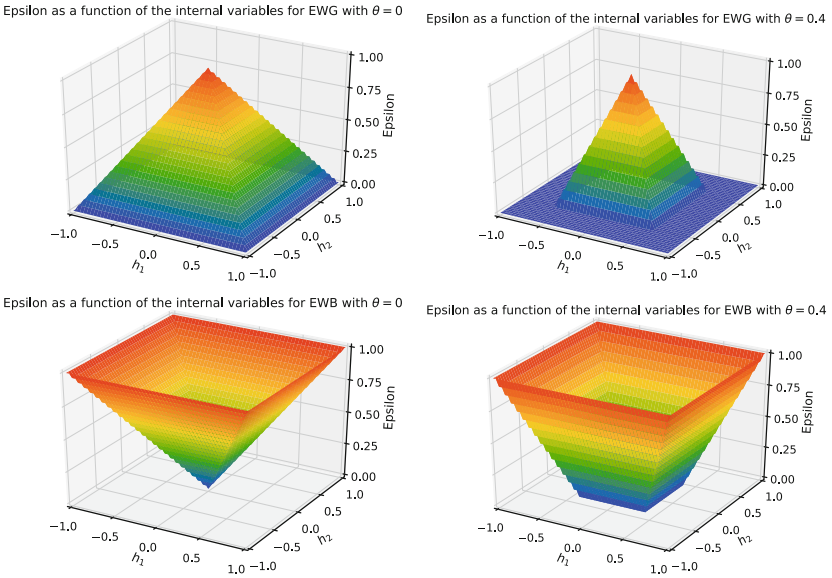

Fig. 2. A visual representation of ϵ as a function of internal variables under the EWG and EWB strategies. Threshold θ creates a buffer zone in which the agent only exploits and does not explore.

internal variables in a strictly predictable manner. "Greed" tests the strategies' ability to adopt a greedy policy quickly. "Random" randomly determines the outcome of an action once the action has been selected. This tests the strategies' ability to adapt to a highly stochastic world. "Dual" is a combination of "Greed" and "Random". Every fifty time steps, the world switches its behaviour. This tests the strategies' ability to adapt to temporal instability.

Next are grid-world variants that give the agent access to its coordinate position. The action set allows movement in the cardinal directions as well as the option of not moving, with the benefit of reduced resource decay. Four variants of a 3×3 grid world and two corridor worlds – "Corridor A" (10×1) and "Corridor B" (20×1) – are implemented.

The 3×3 grid worlds are designed to test the same properties as the simple environments, using both static and dynamic environments. Food sources change position when consumed. The agent's ability to balance its internal variables is evaluated by means of a "hostile" tile (coordinate position) that reduces one of the variables, whereas the safe environment lacks this tile. The corridor environments are designed to favour the annealed ϵ strategies. A successful agent initially explores to find both ends of the corridor, then uses this knowledge to be "greedy". The hyper-parameters are chosen so that too many exploratory actions will cause the agent's premature death.

The final set of worlds are grid-world variants adding observations of the RGB color triplets for the 3×3 set of tiles (coordinate positions) centered on

the agent. The same properties as in the smaller grid worlds are evaluated in four environmental variants. As these worlds are more complex, their size is limited to a 5×5 grid initially, gradually expanded every 10,000 time steps with one tile in each cardinal direction.

The environments are non-trivial and survival is not guaranteed. The dynamic variants require exploration throughout the simulations, while the static variants promote greedy policies. The hostile environments increase environmental complexity and reduce internal variable symmetry. For subsequent discussion, the environment names are shortened to **G** for "grid world" and **ECG** for "expanding color grid world" followed by a suffix: **SS** for "static safe", **SH** for "static hostile", **DS** for "dynamic safe" and **DH** for "dynamic hostile".

3.2 Agent

The agent uses the DQN architecture [8] with one of two configurations, the difference being the size of the underlying network. A larger network with three dense layers of 64 units is used for the color grid worlds and corridor worlds, a smaller dense network with two layers of four units for the remaining environments. The agents have the hyper-parameters drive $m = 3$ and $n = 4$; discount factor $\gamma = 0.9$; and optimizer Adam [4], with learning rate 0.001, batch size 32, and experience-replay size 10^6. Online and target network switch after 200 steps; training starts after 500 steps. To speed up learning, a random, uniform initializer in the range $(-10^{-6}, 10^{-6})$ is used for the output layer.

3.3 Exploration Strategies

Multiple hyper-parameter settings of both annealed ϵ and constant ϵ are used as baselines to ensure that the best possible baseline is found for each environment. $\epsilon \in \{0, 0.01, 0.1, 0.2, 0.3, 0.4, 1\}$ is tested for constant ϵ. For annealed ϵ, all combinations of $\epsilon_{\text{min}} \in \{0.1, 0.01\}$ and $\epsilon_{\Delta} \in \{0.001, 0.0001, 0.00001\}$ are tested. In similar fashion, the EWG and EWB strategies are evaluated with levels of $\theta \in \{0, 0.2, 0.4, 0.6, 0.8\}$.

The agent structure is held constant for all evaluations in each environment, the only differences being the exploration strategy and random seed. Consequently, all agents have the same potential to learn about their environment. How well and how quickly they do so depends on how they explore that environment.

3.4 Evaluation

Each simulation lasts 1,000,000 steps. The primary metric for the strategies' success is the number of deaths they accumulate in each environment. Ten trials are made for each possible combination of strategy and environment. A death (i.e., terminal state) occurs when an internal variable exceeds 1 or falls below -1. Upon death, environment and agent-internal variables are reset, and the simulation continues.

The number of non-optimal actions serves as a secondary metric for gauging the number of exploratory actions [10]. This highlights any differences in strategic behaviour.

4 Results

The results indicate that EWG and EWB outperform the baseline strategies, the agents dying less often over the course of 1,000,000 steps. The agents find equivalent or better policies, faster, which they maintain throughout the simulation despite often having a higher rate of exploratory actions.

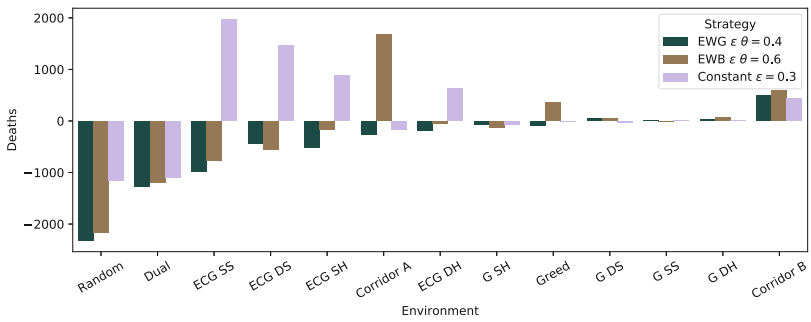

Fig. 3. The average number of deaths under the best-performing strategies of each type, relative to the best annealed baseline strategy ($\epsilon_{\min} = 0.1, \epsilon_{\Delta} = 0.0001$). Lower values are better; a value below zero indicates better than annealed baseline performance.

Figure 3 shows the best hyper-parameter settings of the EWG, EWB, and constant ϵ strategies compared to the best annealed ϵ baseline, with $\epsilon_{\min} = 0.1, \epsilon_{\Delta} = 0.0001$. One can see that, without human interaction, the EWG strategy adapts to environments better than any of the other strategies. This holds even when each strategy is tuned to the environment, as shown in Fig. 4.

Taking a closer look at the effects of the hyper-parameter θ on the EWG strategy, Fig. 5 shows that adding a small threshold makes the strategy more stable across environments. Making the threshold value too high may cause the agent not to explore enough: the internal variables quickly drop below the threshold and the agent becomes greedy, as the corridor environments reveal. However, the strategy appears fairly robust to changes in the threshold, with $\theta \in \{0.2, 0.4, 0.6\}$ performing well in almost all environments.

The EWG and EWB strategies prove capable of quite different levels of exploration depending on environment, without any need for human intervention – as can be seen in Fig. 6. The different environments afford a varying range of possible actions, which changes the number of non-optimal actions slightly. EWG and EWB change rate even within environments of the same type even as the constant ϵ and annealed ϵ do not. This reduces the need for human intervention when the agent is introduced to new or dynamic environments.

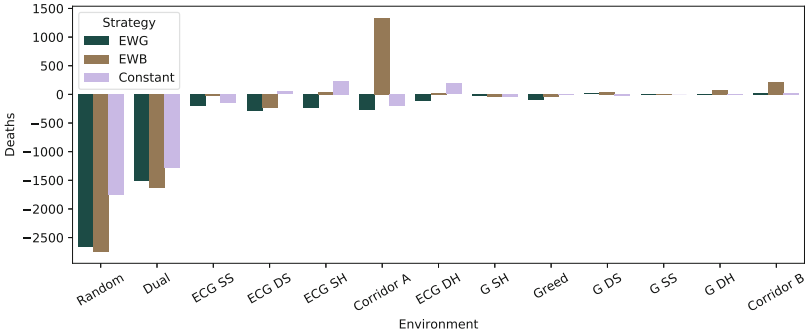

Fig. 4. Even when all strategies are tuned to find the optimal settings, EWG still performs best overall. EWB performs slightly better than EWG in "Random" and "Dual"; however it performs worse than EWG in all others.

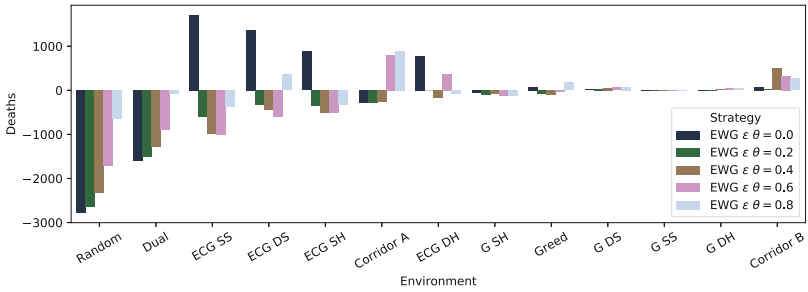

Fig. 5. Comparison of all EWG strategies to the best annealed baseline strategy ($\epsilon_{\min} = 0.1, \epsilon_{\Delta} = 0.0001$). A small value of θ makes the EWG strategy more robust across environments, while a too-high value increases the risk of the agent not exploring enough.

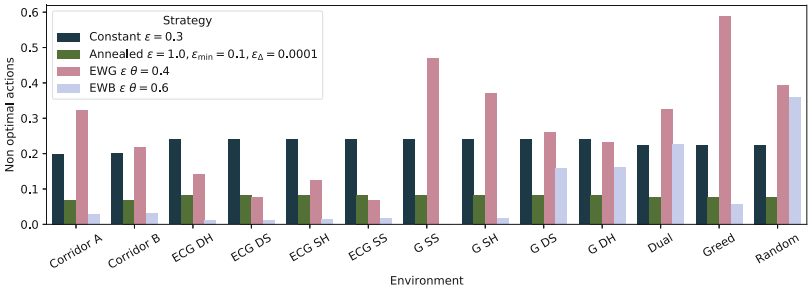

Fig. 6. The percentage of exploratory actions taken in each environment for the best performing hyper-parameter setting of each strategy type. The number changes slightly based on the number of available actions. Both EWG and EWB change their rates of exploration dynamically within the same environment type, based on how well they perform.

5 Discussion

The EWG and EWB exploration strategies take advantage of the information provided by the agent's internal variables. These provide an approximate metric for the agent's well-being and current performance. In contrast to traditional RL strategies, which typically do not have access to information regarding performance, these strategies prove better able to choose when to take exploratory actions – despite being very simple. Indeed, more sophisticated strategies might perform even better.

One of the proposed strategies, EWG, outperforms the others under a range of hyper-parameter configurations, apparently fueled by the additional information it is able to exploit. Simply adding a small threshold to EWG improves performance across most environments. The EWB strategy also performs well, but falls short in some of the environments.

We can extrapolate from the results to a more general insight: the value of internal variables need not be constrained to animats. As mentioned in the introduction, these variables might as well be the energy production and water level of a hydroelectric power plant. In any case – living system, artificial agent modeling a living system, or virtually any other, sufficiently complex non-living system – deviation from optimal levels may successfully be used to facilitate exploration.

6 Conclusion

We introduce two simple strategies for exploration of the homeostatic reinforcement learning problem. These dynamic strategies outperform traditional RL exploration strategies.

References

1. Bersini, H.: Reinforcement learning for homeostatic endogenous variables. In: From Animals to Animats 3: Proceedings of the Third International Conference on the Simulation of Adaptive Behavior, pp. 325–333 (1994)
2. Davies, K.J.: Adaptive homeostasis. Mol. Asp. Med. **49**, 1–7 (2016). Hormetic and regulatory effects of lipid oxidation products
3. Keramati, M., Gutkin, B.S.: A reinforcement learning theory for homeostatic regulation. In: Advances in Neural Information Processing Systems, pp. 82–90 (2011)
4. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization (2014)
5. Kompella, V.R., Kazerounian, S., Schmidhuber, J.: An anti-hebbian learning rule to represent drive motivations for reinforcement learning. In: del Pobil, A.P., Chinellato, E., Martinez-Martin, E., Hallam, J., Cervera, E., Morales, A. (eds.) SAB 2014. LNCS (LNAI), vol. 8575, pp. 176–187. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08864-8_17
6. Konidaris, G., Barto, A.: An adaptive robot motivational system. In: Nolfi, S., Baldassarre, G., Calabretta, R., Hallam, J.C.T., Marocco, D., Meyer, J.-A., Miglino, O., Parisi, D. (eds.) SAB 2006. LNCS (LNAI), vol. 4095, pp. 346–356. Springer, Heidelberg (2006). https://doi.org/10.1007/11840541_29

7. Konidaris, G.D., Hayes, G.M.: An architecture for behavior-based reinforcement learning. Adapt. Behav. **13**(1), 5–32 (2005)
8. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
9. Oubbati, M., Fischer, C., Palm, G.: Intrinsically motivated decision making for situated, goal-driven agents. In: del Pobil, A.P., Chinellato, E., Martinez-Martin, E., Hallam, J., Cervera, E., Morales, A. (eds.) SAB 2014. LNCS (LNAI), vol. 8575, pp. 166–175. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08864-8_16
10. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (2018)
11. Wilson, S.W.: Knowledge growth in an artificial animal. In: Narendra, K.S. (ed.) Adaptive and Learning Systems, pp. 255–264. Springer, Boston (1986). https://doi.org/10.1007/978-1-4757-1895-9_18
12. Wilson, S.W.: The animat path to AI. In: Meyer, J.A., Wilson, S.W. (eds.) From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior, pp. 15–21. MIT Press (1991)