Unsupervised Self-Development in a Multi-Reward Environment

Benjamin Dittes, Christian Goerick

2010

Unsupervised Self-Development in a Multi-Reward Environment

Benjamin Dittes **Christian Goerick**
Honda Research Institute Europe GmbH,
Carl-Legien-Straße 30,
63073 Offenbach / Main, Germany
benjamin.dittes@honda-ri.de

Abstract

Self-development is an important quality for artificial agents, allowing skill development or improvement. In this contribution we analyze this problem for a scenario with multiple rewards, some easier to reach than others. There is no provided sequence of tasks to enforce self-development; rather, the agent must have an intrinsic motivation to discover more difficult reward sources even if a trivial one is always at hand. Then, by removing simple reward sources, the development performance can be measured. We describe the scenario and discuss as well as measure the applicability of standard learning methods. Based on this analysis we present two techniques to allow the desired self-development: a learning rule for quick trajectory learning and a multi-model learning for multiple reward sources. Simulations show the validity of the presented methods.

1. Introduction

Self-development is a popular line of research, especially in the area of developmental robotics (Lungarella et al., 2003). The ability of an artificial agent to develop new skills, optimize existing ones or discover new solutions to tasks is very desirable. Previous work deals with motivation systems (Konidaris and Barto, 2006), exploration strategies (Oudeyer et al., 2007, Mikhailova et al., 2006, Schmidhuber, 2006), action-selection (Brock et al., 2005, Beaudry et al., 2005, Chernova and Arkin, 2007) and self-development frameworks (Bonarini et al., 2006). However, most contributions, especially (Bonarini et al., 2006), see self-development as a technique to ultimately solve one task, or a series of increasingly difficult tasks. This provides guidelines to the development process and thus facilitates and directs this process.

In this work, we would like to address self-development without these guides: in a world with multiple reward sources, some sources of reward will be easier to reach than others. Which intrinsic motivation, exploration and learning methods are necessary to still discover all reward sources and exploit this knowledge if some of them disappear?

To analyze these questions we first introduce a scenario with the following properties: First, it provides a set of reward sources which require increasingly complex action sequences to reach. Second, it provides a sufficiently rich state, sensor and actor space so that finding the more difficult reward sources is not trivial. Third, during the simulation there is a phase where both trivial and difficult reward sources are present, followed by a phase where the trivial reward sources are removed.

During the first phase the agent has the chance to explore the environment in addition to receiving reward from the trivial source. During the second phase the agent has to prove whether he developed the skills to reach more difficult reward sources. This is not the same as presenting the agent with a sequence of increasingly difficult tasks: the second phase serves only to verify whether the self-development in the first phase was successful.

Thus, agents without the ability for 'unsupervised self-development' will either fail to explore the environment in the presence of trivial reward (first phase) or fail to apply the knowledge of other reward sources when the trivial one is removed (second phase). In both cases, this will lead to significantly less reward during the second phase, which can be easily observed.

We will present a concrete simulation environment in Sec. 2. Sec. 3. will then attempt a straightforward SARSA(Rummery and Niranjan, 1994) learning and a simple exploration strategy to solve the problem, followed by an analysis of the shortcomings of these methods. We will then present two extensions which together allow to achieve the desired level of self-development: a transitive extension of SARSA in Sec. 4. and an automatic generation of multiple reward models in Sec. 5.

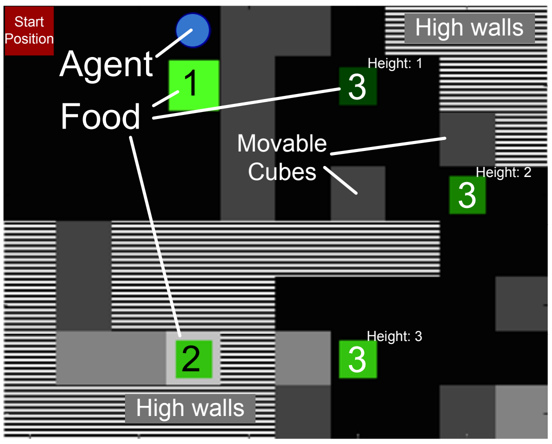

Figure 1: **Simulated Maze** – The figure shows the three-dimensional maze in which all experiments are conducted. The world is composed of unit cubes which can be picked up and moved by the agent, black areas are of height 0, dark gray and light gray areas are of heights 1 and 2, respectively. At several positions there are food sources which are either attached to the ground ('1' and '2') or floating above the ground in various heights ('3').

2. The scenario

The environment is a simulated three-dimensional world of unit cubes which an agent can climb and move, see Fig. 1. The agent has 11 possible actions: it can stay at the current position, move north, south, east or west to an adjacent cube at the same height, climb north, south, east or west to an adjacent cube with height +/- 1, grab the cube it is standing on (thus carrying up to one cube) and put down the carried cube at the current position. Staying, walking and climbing are possible while carrying a cube. At every timestep, the agent executes one possible action (a fitness function inherent to each action prevents execution of impossible actions, like picking up a second cube or walking off the maze).

This ability for climbing and carrying cubes opens the possibility of reward sources with different difficulties: the agent starts each episode in the top left corner and is thus closest to food source 1 ('F1'). While F1 can be reached without climbing or moving cubes, food source 2 ('F2') requires climbing actions and is further from the start position. Finally, food source 3 ('F3') is positioned above ground and can be reached only by moving cubes to the right positions; there are three such 'floating' food sources, at heights 1, 2 and 3 (see Fig. 1).

Sensor information is given to the agent in form of it's current three-dimensional position at every timestep, a flag indicating whether a box is carried or not is not given.

The agent receives a reward of 1.0 for every timestep if it is at a position with food and 0.0 otherwise. We monitor the agent's performance in the world by keeping track of a 'hunger' level, which decreases by 0.1 per timestep when receiving reward and grows by 0.005 per timestep otherwise. Although food sources of different difficulty are in different distances to the starting position, the ability of the agent to 'stay' at each of these food positions allows it to gather enough food to reduce hunger to 0 within one episode for all food sources.

All experiments are conducted with the same conditions: one episode lasts 100 timesteps, resulting in a complete reset of the world (only the agent's hunger level is kept). One complete run consists of 150 episodes, split into three phases of 50 episodes each: In phase 1, all food sources are available. In phase 2, after 50 episodes, F1 is removed. Finally, in phase 3, after 100 episodes, F2 is removed, after which the simulation continues until the end of episode 150. The last phase, from episode 101 to 150, shows whether the agent developed the necessary skills to reach the more difficult source F3, which can be observed by looking at the hunger level of episodes 101 to 150.

We believe this scenario meets the criteria introduced in Sec. 1. for allowing (episodes 1 to 100) and measuring (episodes 101 to 150) the agent's ability for unsupervised self-development. The goal, in other words, is to find an agent which does not starve in the last phase. To this end we will start by applying straightforward learning and exploration techniques in the following section.

3. Standard approaches

3.1 Straightforward SARSA

The first and simple food model we will use is the State-Action-Reward-State-Action learning, short SARSA, as presented in (Rummery and Niranjan, 1994). It is based on the fact that by knowing the five quantities $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$, where s represents a state of the agent, a represents an action of the agent and r represents a reward, the agent can learn the predicted value V_{s_t,a_t}, which is an approximate over all future rewards expected from executing action a_t in state s_t. Then, in a new state s^*, an agent can look for the action $a^* = \text{argmax}_a V_{s^*,a}$. For a more complete introduction to reinforcement learning, please see (Kaelbling et al., 1996).

We employ a SARSA(0) with $\alpha = 0.5$, $\gamma = 0.99$ combined with epsilon greedy exploration with $\epsilon = 0.1$, i.e. in 10% of cases, choose a random action instead of a^*. The update rule for the model thus reads:

$$V_{s_t,a_t} \leftarrow V_{s_t,a_t} + \alpha(r_t + \gamma V_{s_{t+1},a_{t+1}} - V_{s_t,a_t}) \quad (1)$$

The parameter α is the learning rate, it mediates

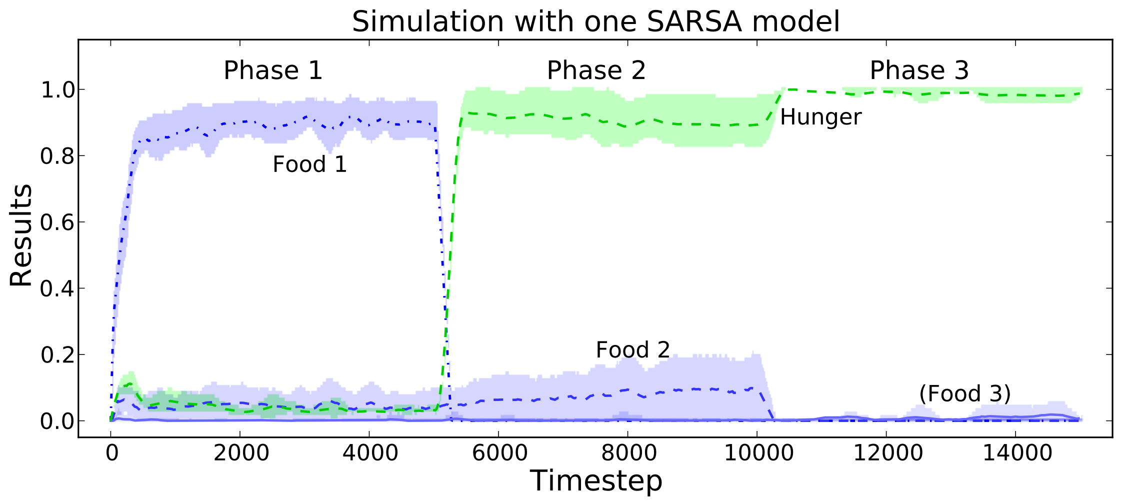

Figure 2: **Simulation with epsilon greedy exploration** – The figure shows the hunger level of the agent and the amount of time spent at food sources F1, F2 and F3, averaged over 100 runs (see Appendix for details). During phase 1, the agents are content with visiting F1 (about 10% of agents visit F2), but do not perform any exploration towards F3. In phase 3, most agents are unable to find F3 and the hunger level stays at almost 1.

between stored and new information, the parameter γ is the discount factor, it mediates between short-term and long-term expected reward.

The results of this approach are shown in Fig. 2, as an average over 100 runs. Agents reliably find F1, 10% even find F2, but there is no exploration towards F3 and when F1 and F2 are removed in phase 3 the agents starve.

The obvious shortcoming of this approach is the missing exploration towards alternative food sources in phase 1. There is no motivation beyond reaching (any) one food source, thus no effort towards self-development can be observed. To compensate this, an exploration strategy is necessary and although many more elaborate have been presented, e.g. (Oudeyer et al., 2007, Schmidhuber, 2006), we will start with a very simple one in the next subsection and then discuss necessary extensions.

3.2 Novelty-based Exploration

To provide the agent with a second motivation in addition to finding food, we extend the previous approach by a novelty-based exploration: A second SARSA model (same parameters) learns in parallel to the one responsible for food and is rewarded internally by the agent whenever it reaches a position it has not visited before (up to 10 times for each position with a reward of 0.1 each time). This second model will be called 'exploration model' as it learns actions which lead to the agent being in novel states.

The agent can now switch between the two models for action selection based on the hunger level: if the hunger is above 0.5, the food model is queried, otherwise the exploration model is queried. Both models learn about all timesteps (with different reward sources) and the epsilon greedy exploration is applied to both as well.

The results of this approach can be seen in Fig. 3, again averaged over 100 runs. Now, the agents spend a little time at F2 and F3 in phases 1 and 2. However, when F2 is removed, the agents are unable to exploit this knowledge: although the exploration model led to sporadic visits to alternative food sources, the food model remained focused on F1.

We draw two conclusions: First, the very simple novelty-based exploration method applied here was sufficient to find alternative food sources – thus, a more elaborate exploration strategy would probably lead to quicker and more reliable results but would still suffer from the model's inability to store other food sources and would therefore not benefit the analysis of self-development. Second, the main effort to enable unsupervised self-development should deal with the food model. We will analyze why the simple food model used so far was unable to solve the problem in the next subsection and present two necessary extensions in Secs. 4. and 5.

3.3 Shortcomings of the Simple Approaches

Three factors contribute to the bad performance of the simple food model despite the exploration of alternative food sources.

The first problem is the dominance of actions towards F1 in all states close to F1, including the starting position in the top left of the maze (see Fig. 1). Epsilon greedy exploration leads to small variations of the trajectory around F1, thus creating a model which is able to reach F1 from practically every po-

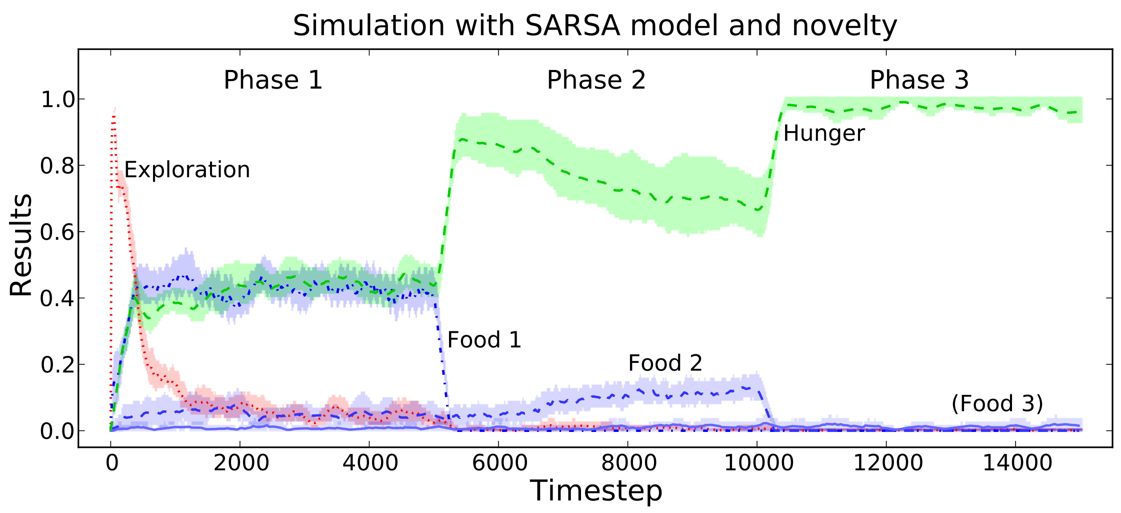

Figure 3: **Simulation with novelty-based exploration** – The figure shows the hunger level of the agent, the received novelty reward and the amount of time spent at food sources F1, F2 and F3, averaged over 100 runs (see Appendix for details). Behavior of the agents is governed by a food-oriented model for hunger > 0.5 and by a novelty-oriented model for hunger < 0.5. Although hard to observe, visitation rate to F2 in phase 2 is above the level seen in Fig. 2, which accounts for the lower hunger level. Agents now explore the entire environment during phases 1 and 2, which can be seen by the sporadic visits to F2 and F3 (both 'Food 2' and 'Food 3' curves are repeatedly above zero). However, this does not lead to good performance in phase 3, see Sec. 3.3 for details.

sition in the top left quarter of the maze. The probability of 'escaping' this area to reach other reward sources (using the food model) is therefore very low during all phases.

The second problem is the fact that reaching F2 and F3 requires more complex combinations of actions, leading to fewer visits in general (which can be seen in Fig. 3). This results in a slow propagation of discounted reward from the food sources towards the starting position, thus maintaining the dominance of actions towards F1 in phase 1 of the simulation and leading to slower stabilization of trajectories to F2 and F3 in phases 2 and 3, respectively.

The third problem is the value of expected discounted reward of actions towards F1. When F1 is removed in phase 2 and no reward is received at the position of F1 any more, many updates of the food model are required to slowly reduce this expected reward. This leads to a high number of repetitive visits to F1 in phases 2 and 3 until the information about this lack of reward propagates through the model (this effect can be seen in Fig. 3 only by the increase of hunger level, visits to removed food sources are not shown).

In the following sections, we will present two techniques to address these three problems.

4. Transitive SARSA for Rapid Learning

From the analysis of straightforward SARSA learning and simple exploration we now move to two techniques which together allow the agent to show the requested self-developmental qualities. We first address problems two and three from the analysis in Sec. 3.3: the SARSA learning of Eqn. 1 requires too many learning episodes to propagate the knowledge of other reward sources back to the starting position (thus requiring too many visits to other reward sources) and the knowledge of the missing F1 is not applied to the surrounding states quickly enough.

We therefore propose a transitive extension of SARSA learning ('TransSARSA'): instead of updating the value function based on the given reward only in the state preceding the reward – and waiting for the value function to propagate further into the past – we apply the properly discounted reward back to the state/action history immediately.

Given a sequence $\{(s_{t-H+1}, a_{t-H+1}, r_{t-H+1}), (s_{t-H+2}, a_{t-H+2}, r_{t-H+2}), \ldots, (s_t, a_t, r_t)\}$ with the history of H timesteps leading up to a reward (thus $r_t > 0, r_i = 0, \forall_{t-H<i<t}$), we apply for all steps $k = 1..H-1$

$$\begin{aligned} V_{s_{t-k},a_{t-k}} \leftarrow \; & V_{s_{t-k},a_{t-k}} + \alpha\beta\gamma V_{s_{t-k+1},a_{t-k+1}} \\ & + \alpha(1-\beta)\gamma^k r_t - \alpha V_{s_{t-k},a_{t-k}} \quad (2) \end{aligned}$$

Here, α and γ are standard SARSA parameters, $V_{s,a}$ is the learned discounted reward and β is a new parameter which mediates between local updates ($\beta = 1$ means standard SARSA learning) and transitive updates ($\beta = 0$ means updates learn only the discounted final reward).

The method is inherently an online method as it performs updates every time a reward is received

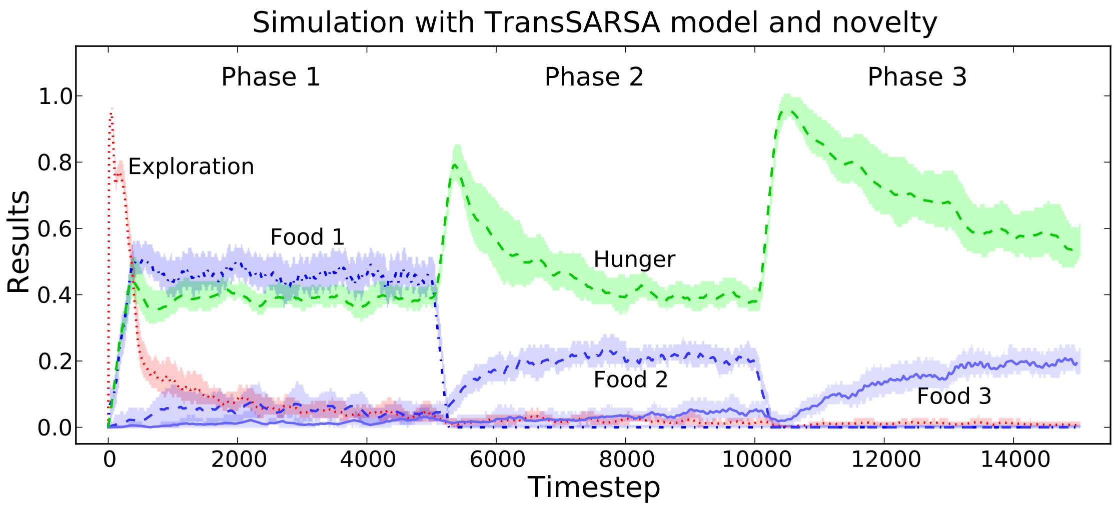

Figure 4: **Simulation with novelty-based exploration and TransSARSA** – The figure shows hunger level, novelty reward and food performance averaged over 100 runs (see Appendix for details). Behavior of the agents is governed by a food-oriented TransSARSA model for hunger > 0.5 and by a novelty-oriented TransSARSA model for hunger < 0.5. Exploration behavior is similar to experiments without TransSARSA (see Fig. 3), but in phases 2 and 3 the learning is much faster, leading to a quick adaptation to the new food sources, once they are found. However, at the beginning of phase 2 (around timestep 6000) and at the beginning of phase 3 (around timestep 11000), the current food model proves useless since it focused strongly on one food source. Therefore the hunger level spikes at these times until the agents are able to find and utilize new food sources.

for the H timesteps preceding the reward. In case no reward is received for $H_{\max}$ timesteps, learning is performed according to Eqn. 2 with $H = H_{\max}$, $r_t = 0$. This is relevant to the rapid unlearning required when one reward source disappears: the zero-reward is immediately propagated to all $H_{\max}$ actions of the trajectory towards the missing food source. The method can be applied as an offline method, but then requires splitting of the offline data into sequences leading to reward or integrating successive rewards, but this was not implemented for this contribution.

The results of a simulation with the simple exploration strategy from Sec. 3., combined with one TransSARSA model ($\alpha = 0.5$, $\beta = 0.4$, $\gamma = 0.99$, $H_{\max} = 20$) can be seen in Fig. 4. There is no significant difference in phase 1. In phase 2 and 3 the learning progress is much faster, leading to a quick adaptation of F2 and F3, respectively. However, this is not a result of self-development in phase 1, as can be seen by the spikes of hunger levels at the beginning of phase 2 and 3. Rather, the food model used during phase 1 proves useless at the beginning of phase 2, thus the agent performs a random walk and, having found F2, is able to quickly store this information in the TransSARSA model and utilize it. We therefore conclude that TransSARSA increases the learning and unlearning speed, but it does not solve the problem of exploiting the discovery of F2 and F3 in phases 2 and 3.

An additional note on the reinforcement properties of TransSARSA: The method is inherently on-policy[1], for the reasons that it is based on SARSA and, in addition, that it updates the value function based on one specific state-action-trajectory. This is intended, but it leads to a non-convergent value function which is much better suited for quickly adapting to changing rewards than for learning an optimal policy in static scenarios. The same 'transitive-update' technique can also be applied to Q-learning, but as this is an off-policy method it is not surprising that the performance for rapid relearning is much worse and we will not go into detail here. Finally, although the experiments done in this paper use a simple grid-world, the update function of TransSARSA can also be applied to continuous sensor spaces, given a continuous function approximation model, e.g. LWPR(Vijayakumar et al., 2005).

5. Multi-model Learning for Rapid Switching

After improving the learning behavior itself with the TransSARSA learning method, the question still remains how to utilize the discovery of F2 and F3 during phase 1 in later phases. So far, all learning models focused too much on F1 during phase 1 and thus

[1]Reinforcement learning strategies are often separated into on-policy, i.e. the learning depends on the policy implicit to V_{s_t,a_t}, and off-policy, i.e. the learning does not depend on V_{s_t,a_t}. Off-policy methods are said to be more appropriate for static problems where convergence is essential.

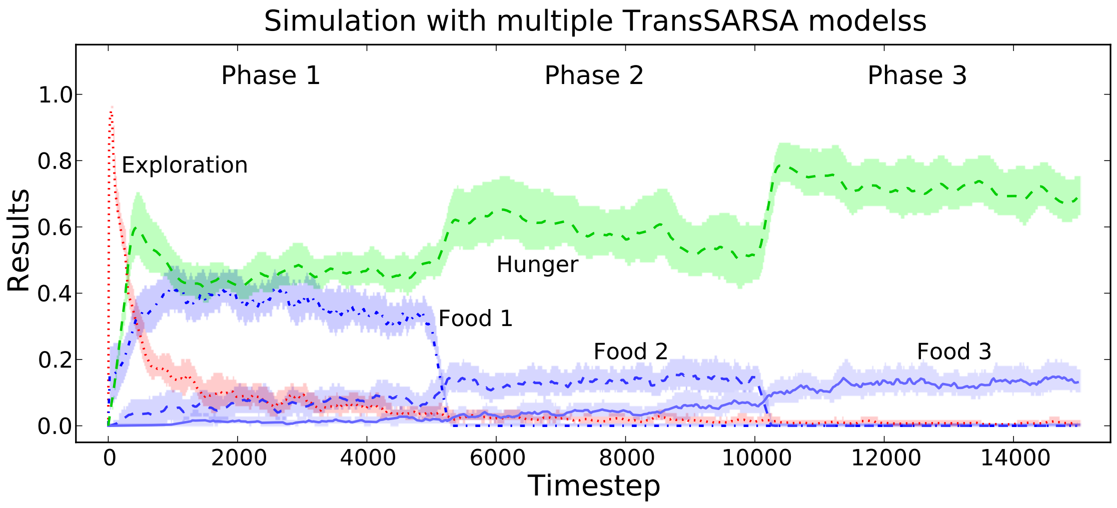

Figure 5: **Simulation with novelty-based exploration and multi-model TransSARSA learning** – The figure shows hunger level, novelty reward and food performance averaged over 100 runs (see Appendix for details). The agent performs TransSARSA learning, with separate models for each discovered food source, and chooses between exploration for hunger < 0.5 and food search for hunger > 0.5. Exploration behavior is similar to previous experiments, but at the beginnings of phases 2 and 3 agents are now able to quickly switch to pre-existing models to reach food sources F2 and F3, respectively, without a hunger spike as observed in Fig. 4. This demonstrates that they were able to construct models for non-trivial food sources in phase 1 (showing the demanded unsupervised self-development) and utilize those models when more simple food sources disappeared.

did not store the information of other food sources sufficiently.

We solve this problem by dynamically generating TransSARSA models every time a position with reward is discovered. The agent starts with the exploration model alone and moves through the environment. Once it receives food reward in a state s^*, a TransSARSA model V^{s^*} is created and associated to this specific world state. This model is now updated online as described in Sec. 4. and receives food reward from the environment, but only if the agent is in the associated state s^*. Thus, a model is learned which is designed to allow the agent to reach s^*.

The same is done for every other environment state where the agent receives food reward: for every new state where reward is received, a new food model is created and then updated. During phase 1, this leads to up to five different food models for the different reward states (F3 has three states), each updated online in parallel to the others.

To select an action, the agent now choses the exploration behavior for hunger < 0.5 and a random food model for hunger > 0.5. In the latter case the agent does not switch to another food model until the hunger level drops below 0.5. This last constraint leads to a slight reduction of quality in phases 2 and 3: although the agent may know very well how to reach one food source, the selection mechanism may force it to look for a previously seen more difficult food source, thus reducing the average time spent at a food source (please observe the slightly higher hunger level in Fig. 5 as compared to Fig. 4) – on the other hand, this leads to an improvement of alternative models, which is in the interest of unsupervised self-development.

In phase 2 and 3 it may now happen that the agent visits a state s^* which has an associated food model V^{s^*} but no reward is received from the environment. In this case, the model V^{s^*} is deleted from the pool of food models. In other words, once the agent realizes a food source has disappeared it will never try to visit it again. This is a very crisp heuristic which may be extended to include food source confidences or disabling instead of deleting food models, but it serves the purpose of reacting to environment changes for our problem.

The results of this approach, using the same TransSARSA parameters as in Sec. 4., can be seen in Fig. 5.

During the first 50 episodes, exploration goes along the same lines as in Fig. 3, with the addition that F2 and F3 are visited slightly more often due to the random fixation on one specific food model when hunger goes above 0.5. At the beginning of phase 2, there is no spike in hunger levels anymore as could be observed using only one TransSARSA model. This is because the agents could remove the food model for F1 quickly and start using the food model for F2, which contains all information necessary to reach F2. The same can be observed for phase 3: the agents 'switch' to a food model for F3 fluently and without a hunger spike and then slightly improve this model.

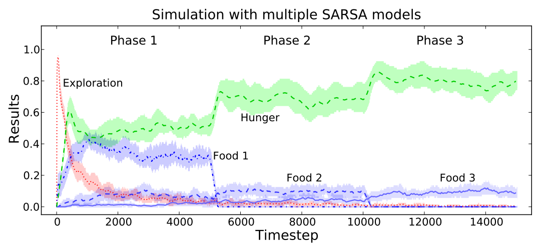

Figure 6: **Simulation with novelty-based exploration and multi-model SARSA learning** – Same setup as Fig. 5 but with traditional SARSA learning. Due to slower convergence of the individual models the performance is worse than with TransSARSA learning.

It is this ability to switch between models that shows two things: First, the fact that the agents where able to switch proves that they already constructed a model to reach F2 during phase 1. Second, the fact that the agents hunger level does not spike shows that these models were stored in a way that allows application after an environment change. Together this means that the innate properties of the agent are now sufficient to reach unsupervised self-development in this simulated environment.

The presented approach of multi-model learning is related but not similar to options-based reinforcement learning(Sutton et al., 1999) in that it does not split models into different skills but into different reward sources. Learning of options to achieve reward sources, even sharing of such skills among different reward sources, is possible but in our opinion not crucial to the central point of self-development.

6. Discussion & Conclusion

This paper presents a scenario and a set of algorithms which allow and measure unsupervised self-development. We presented a simulated maze with one important quality: it allows placing reward sources with a variety of difficulties, from very trivial to reach to more complex to reach. This enabled us to define 'unsupervised self-development' as the ability of an agent to discover more complex food sources although simple ones are present and apply this knowledge when the simpler ones are removed.

We analyzed the performance of standard learning and exploration techniques to achieve this property and formulated two aspects which required better solutions: the speed with which models adapt to environment changes (i.e. the removal of a food source) and the decoupling of stored trajectories towards different food sources.

We then presented two techniques able to overcome these two problems. TransSARSA is a learning method which extends SARSA by propagating reward directly along the followed trajectory and thus adapts very quickly to environment changes. Multi-Model learning disentangles different reward sources by dynamically creating and destroying reinforcement models, each responsible for reaching exactly one environment state where reward was received.

Both techniques, combined with the simple novelty-based exploration strategy employed in the initial experiment, are able to produce the desired agent behavior: Despite the presence of a very simple food source, the exploration strategy drives the agent towards other food sources which are quickly stored in dynamically created TransSARSA models. Once the agent realizes that food sources have disappeared, the associated models are removed and the remaining models allow the agent to find the remaining, more complex food sources.

It is not the aim of this work to present the best possible solution for each algorithm and combine them in a highly complex environment; the presented scenario is limited and discrete, the exploration method is not optimized to the task, the TransSARSA learning is quick but non-convergent and the rules for creating and destroying food models in the multi-model learning are only just sophisticated enough for this scenario. However, our focus is the analysis of the interplay of standard algorithms and extensions thereof to produce the desired property of 'unsupervised self-development'. The presented scenario and algorithms allow this analysis and thus make a step towards understanding the necessary innate properties needed to reach the desired property in the simplest non-trivial system.

Future work will focus on the application of the developed methods to interactive robotics. Especially during phases where the robot is left to explore its environment without tutor interaction, unsupervised self-development is very important. When the tutor returns, he can enforce or weaken the robot's developed skills to shape them towards solving one specific task – which we expect will require much less tutor input than would be required when incrementally developing all skills.

Appendix

Quantitative Details of Visual Results

The results shown in Figs. 2, 3, 4, 5 and 6 are computed using 100 simulation runs each with the specified simulation parameters. The visualization is tuned to allow an intuitive visual perception of the relation between food reward, hunger and exploration.

Food levels for the three food sources F1, F2 and F3 are shown separately as dark, medium and light blue lines. The value for each agent is 1 if he was at the position(s) associated with the food source in the last 20 timesteps and 0 otherwise, this reflects the 1-to-20 relation in hunger level increase and decrease (see Sec. 2.). Hunger level is shown as a green dashed line. Novelty level is shown as a red dotted line, with a value at each timestep equal to the sum of exploration reward given over the 20 previous timesteps, this reflects $H_{\max} = 20$ in the learning of the exploration model (see Sec. 4.).

All figures show means of all values, which are computed over the values of all 100 runs of each experiment. Since food and exploration values may oscillate rapidly within each agent, the depicted standard deviations are not computed between the global mean and individual agents but between the global mean and the means of 30 random 10-out-of-100 subsets, thus giving an impression of how the results of small subsets vary around the result of all agents.

References

Beaudry, E., Brosseau, Y., Côté, C., Raïevsky, C., Létourneau, D., Kabanza, F., Michaud, F., et al. (2005). Reactive planning in a motivated behavioral architecture. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 1242. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Bonarini, A., Lazaric, A., Restelli, M., and Vitali, P. (2006). Self-development framework for reinforcement learning agents. In *Proceedings of the Fifth International Conference on Development and Learning, Bloomington, IN, USA*. Citeseer.

Brock, O., Fagg, A., Grupen, R., Platt, R., Rosenstein, M., and Sweeney, J. (2005). A framework for learning and control in intelligent humanoid robots. *International Journal of Humanoid Robotics*, 2(3):301–336.

Chernova, S. and Arkin, R. (2007). From deliberative to routine behaviors: a cognitively inspired action-selection mechanism for routine behavior capture. *Adaptive Behavior*, 15(2):199.

Kaelbling, L., Littman, M., and Moore, A. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.

Konidaris, G. and Barto, A. G. (2006). An adaptive robot motivational system. *Lecture Notes in Computer Science*, 4095:346–356.

Lungarella, M., Metta, G., Pfeifer, R., and Sandini, G. (2003). Developmental robotics: a survey. *Connection Science*, 15(4):151–190.

Mikhailova, I., von Seelen, W., and Goerick, C. (2006). Usage of general developmental principles for adaptation of reactive behavior. In *Proceedings of the 6th International Workshop on Epigenetic Robotics, Paris, France.*

Oudeyer, P., Kaplan, F., and Hafner, V. (2007). Intrinsic motivation systems for autonomous mental development. *IEEE Transactions on Evolutionary Computation*, 11(2):265–286.

Rummery, G. and Niranjan, M. (1994). On-line q-learning using connectionist systems. *Technical report, Cambridge University Engineering Department.*

Schmidhuber, J. (2006). Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, 18(2):173–187.

Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112:181–211.

Vijayakumar, S., D'souza, A., and Schaal, S. (2005). Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634.