# Adaptation of Learning Agents Through Artificial Perception

Journal Title
XX(X):1–9

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

**Mirza Ramicic[1] and Andrea Bonarini[1]**

**Abstract**
The process of online reinforcement learning also creates a stream of experiences that an agent can store to re-learn from them. In this work, we introduce a concept of artificial perception affecting the dynamics of experience memory replay, which induces a secondary goal-directed drive that complements the main goal defined by the reinforcement function. The different perception dynamics are capable of inducing different "personality" types able to govern the agent behavior, possibly enabling it to exhibit an improved performance over an environment with specific characteristics. Experimental results show that different personalities show different performance levels when facing environment variations, therefore showcasing the influence of artificial perception in agent's adaptation.

**Keywords**
Reinforcement learning, artificial agents, affective computing, cognitive architectures, perception

## Introduction

Reinforcement learning is a process aimed at reducing uncertainty about which actions an agent is required to take at each time step in order to better adapt to its environment. This process of adaptation is driven by a specific feedback the agent receives, usually provided by a reinforcement function ranging on some signals from the environment. The main goal of the agent is to create a mapping from its sensed state to the probability of selecting an action that would represent the best choice to maximize the received reinforcement in the long run. This probability distribution is called an optimal policy $\pi^*$ and it is a result of an iterative update of the previous policies $\pi$ over the learning process.

Given that we may have multiple optimal policies $\pi^*$ all of them achieving optimal behavior in different ways, by selecting different actions, it might be possible to influence the agent behavior by further narrowing action selection, while still keeping it focused on the main goal of maximizing the reinforcement. This work exploits this possibility by modifying the agent's behavioral characteristics by changing the way the agent perceives the feedback it receives from its immediate environment; this creates an additional secondary drive that augments the main one given by the reinforcement function. Perception is not represented by the data collected by the agent, but as the way the feedback from its immediate environment is encoded and managed by the learning mechanism. Influencing the way data are encoded, we can modify the dynamics of the agent's perception and thus influence its behavioral characteristics.

Due to the increase in state space dimension, modern approaches to reinforcement learning rely on a function approximator, such as an artificial neural network, to approximate the state-action values. The proposed approach takes an advantage of the dynamics of the *stochastic gradient descent* algorithm which is used to perform the learning update on the weights of the artificial neural network at each time step in order to better approximate the values. An agent learns from a sequential stream of experiences and after each perception it updates its belief about the optimal state-action value $Q^*(s, a)$ by performing a *gradient descent* on the parameters or weights of neural network $\Theta$ in order to minimize the square error of the previous estimate and the expected value of $Q$ for the perceived state $s$ and the taken action $a$. For the accurate approximation a *gradient descent* should be performed on each dataset point and, in our case, this means computing an algorithm not just on the current experience, but on the whole history of experiences the agent perceived so far, which is computationally impractical for most implementations. To alleviate this approximation problem a *minibatch stochastic gradient descent* first proposed by Lin (1993) is used, which, instead of re-learning from all of the experiences selects a batch of few random experiences from a sliding window buffer of experiences called *replay memory* and re-learns from this batch at each time step. The dynamics of determining which experiences to store and replay using minibatch approach form the basis for the implementation of the *artificial perception* mechanism that is the focal point of this work.

A reinforcement learning mechanism is concerned not only with the information that its environment provides, but also with the way this information is perceived. This perception factor becomes the additional secondary drive that could further reduce the agent's uncertainty during the

[1]Politecnico di Milano,Dipartimento di Elettronica, Informazione e Bioingegneria

**Corresponding author:**
Mirza Ramicic, Politecnico di Milano,Dipartimento di Elettronica, Informazione e Bioingegneria, Piazza Leonardo da Vinci, 32, 20133 Milano
Email: mirza.ramicic@polimi.it

learning process. The process of a secondary drive controlled by the perception mechanism allows for a further and more subtle modification of an agent's behavior without interfering with the primary reward maximization behavior. Similarly in humans we see different dispositions to a range of behaviors that alter they way they achieve their primary goals. These dispositions are known as personality traits and they are a product of the different physical characteristics of human brains altering the chemical compositions that governs the behavior. For example, extroverted and introverted individuals can achieve the same primary goal of enjoying, but in a different way; the first will most probably engage in social activity, while the latter will prefer to stay home, away from people. This paper proposes an algorithm to manage perception able to model agents with a specific set of *personality types*, similar to human ones, that will enable them to better adapt and thrive in environments with different or changing characteristics.

## Related Work

### Prioritized sampling and replay

To reduce the time needed to collect enough experience to converge to a good estimation of Q, it has been proposed to keep in memory some of the already collected experiences and to replay them.

Dyna approach by Sutton (1990) incorporates learning with planning as two radically similar processes. Planning is taking advantage of the world model to generate imaginary experiences that can be used by a model-free learning algorithms such as q-learning. In doing so it can incorporate previous experiences in the process of learning similar to replay memory approach. It has also been extended to support learning with linear function approximation by Sutton et al. (2012), which is consistent with the approach presented in this paper.

Similar to the ideas of Dyna algorithm the *internal simulation* by Jirenhed et al. (2001); Ziemke et al. (2005) relied on the fact that human perception can allow for self-generated sensory experience in the absence of external stimuli. The agent's *inner world* is able to build an internal representation of the external world and by generating experience partly alleviate the need for costly trial and error actions. The model used a recurrent neural network in which the previous inputs are able to influence future ones providing an *internal state* or context.
The *internal simulation* approach was also used by Shanahan (2006) to model cognitive functions such as anticipation and planning.

In a more recent work Ramicic and Bonarini (2019) used a genetic algorithm to evolve an artificial neural network crisp selector that was able to determine if the selected experience would be sampled into the replay memory. The approach evolved a specific perception filter similar to the one presented in this paper that was able to further adapt the agents to the environment characteristics.

Since the introduction of the replay memory mechanism in Deep Q-Learning (DQL) Mnih et al. (2013) many works have been aimed at further improving the efficiency of learning, by focusing, or giving priority, to certain types of experiences over others, both in sampling and replay. One of the first successful approaches Schaul et al. (2015) used stochastic prioritizing on the experiences stored in replay memory, with high *Temporal Difference* (TD) error, under the assumption that high TD error of experience transition would make the training faster because of its higher deviation from the current approximated Q-value for the state-action pair.

Another approach Zhai et al. (2016) further argued that prioritized sampling performed by Schaul et al. (2015) may suffer from loss of potentially valuable transitions with higher TD error especially in the beginning of the learning process when the transitions with rewards are mainly the ones that account for high TD error levels. Instead of uniform sampling, their approach was to sample all the transitions into two separate replay memory buffers: one containing the transitions with the immediate reward, and the other the rest of the transitions. Stochastically sampling from the two memory buffers with different priorities allowed them to reach a learning speed higher than in Schaul et al. (2015).

Another approach Ramicic and Bonarini (2017b) introduced a different criterion for prioritizing on the experiences, based on the Shannon's entropy level of the starting state of a transition, $s_0$. This criterion directly influences the learning performance given that during each approximation, state $s_0$ is backpropagated through the ANN on input together with the TD error of the taken action $a_0$ on the output in order to update our prediction of $Q^*(s, a)$. Experimental results from Ramicic and Bonarini (2017b) show that the transitions with higher entropy levels of $s0$ are more likely to train the approximator faster, and even more when they are coupled with a higher level of TD error on the output. Therefore, combining the two criteria leads to an improvement of speed.

In Ramicic and Bonarini (2017a), we wanted to further explore the social aspect of agent behavior. In a multi-agent setting, agents had the ability to communicate by exchanging a positive reinforcement upon contact and thus motivating it. Agents had the possibility of learning to choose whether to engage more with other agents, which provided an indirect communication reinforcement, or focus more on the exploration of the environment in search for more direct reinforcement coming from abundant food sources. Experimental results have shown that agents prioritizing on the transitions that led to communication between the agents can modify their behaviour, making them more inclined to engage with the interaction, while agents that focused on the transitions that are related to exploration showed tendency to stay away from other agents. These two types of agents are used to model the two extremes in the main five factor John et al. (1999) personality dimensions of Extroversion; on the higher end of the scale there is the agent type that gravitates more towards interaction with other agents and, similarly, on the low scale we have the more selfish, exploratory agent type.

In the work we are presenting here, we built upon this approach using the same prioritization principle, but instead of focusing on the social aspect we have developed a general framework to fully support the concept.

### Agents with personality traits

In the work by Recchia et al. (2014) agents with varying personality traits were developed based on Myers-Briggs

personality type indicator by assigning reward adjustments specific to the trait. The agents were tested in a multi-agent cooperative environment and results were measured by the team performance on all of the agents under the *commander* agent. The experiments showed that the best team performance was under the *commander* agent that implemented team oriented personality types instead of introverted selfish ones.

Sun and Wilson (2014b,a) explore the role of computationally modelled instinct as a product of basic human motivation in forming of various personality traits. The personality types are represented by three axis; social/shy, confident/anxious and responsible/lazy. The experiments performed interpret the influence that a specific personality has on behavior preferences in agents.

A more recent work by Metcalf et al. (2018) proposes a method for learning agents to adopt turn-taking behaviors such as passive, aggressive and stochastic while interacting with rule-based agents.

### *Agents with emotional states*

In literature starting from Goleman (2006) emotion is often defined as a complex and dynamic process that is mediated by personality, time and experience. A work by Rumbell et al. (2012) provides a good introduction and review of the work that dealt with the emergence of emotional states in artificial learning agents.

One of the first approaches in modeling emotion in reinforcement learning is presented in a work by Gadanho and Hallam (1998) implementing a bottom-up model with four basic emotions of sadness, happiness, fear, and anger that are induced by a combination of feelings and a simple artificial hormone system. Instead of providing a secondary drive goal as outlined in the approach presented here, the main role of the emotions in this work is the decomposition of the main drive into more basic parts which can be activated by emotions and thus become more manageable separately.

Expanding on Gadanho and Hallam (1998) another biologically inspired work was presented by Tsankova (2009). This work proposed a model of Amygdala; an emotion controlling brain cluster which is found in complex vertebrates. The emotion model which is mediating the action selection is based on the computational model of information flow in Amygdala and its being used primarily to model the emotion of fear. The performance was measured in agents effectiveness during tasks of obstacle avoidance with varying degrees of complexity.

Salichs and Malfaz (2006) proposed models of three emotions; happiness, sadness and fear which affected the algorithm in different ways. The emotion of happiness was used as an intrinsic motivation as its maximization represented the final goal of an agent. Happiness/sadness dichotomy also provided positive/negative reinforcement values while the emotion of fear was used by an agent to avoid potential dangers.

In the work by Ahn and Picard (2005) the emotions are induced by their components of valence and arousal, which, similarly to our approach, takes into account the cognition flow of the agent's experience stream. The valence component takes into consideration the history of previous experience to calculate the anticipation of the reward, and it is positive when a choice is expected to give a reward higher than expected, and negative if that is not the case. The arousal component increases with the cognitive uncertainty associated with the decision.

Sato et al. (2004) similarly to Ramicic and Bonarini (2017a) focused on idea of emotions as emerging properties of social multi-agent interaction while each agent implemented a set of behavioral rules that affected its action selection. The emotions were represented by two dimensions; valence and activity which in turn affected the behavioral rules. The agents were able to learn how to show positive emotion towards benevolent agents and negative towards the hostile ones and to decide whether to flock towards them or avoid them. This behaviour was made possible as they were able to detect the valence of the other agents and consequently their dispositions.

## Theoretical Background

### *Reinforcement Learning*

Reinforcement learning is a process of creating a policy $\pi$ that maps the agent's perceived state to the probabilities of selecting the possible actions available to an agent. In this process the policy $\pi$ is constantly updated by an agent's interaction with the environment in discrete time steps to maximize the total cumulative reward over the long run and converge towards optimal policy $\pi^*$ Sutton and Barto (1998). The steps are defined as transitions over a Markov Decision Process and they are represented by a tuple $(s_t, a_t, r_t, s_{t+1})$. An optimal policy mapping requires an optimal action-value function $Q^*(s, a)$ defined as the maximum expected return while following the policy $\pi$ shown in Equation 1.

$$Q^*(s, a) = \max_{\pi} \mathbb{E}[R_t | s_t = s, a_t = a, \pi] \tag{1}$$

Bellman Optimality Equation 2 guarantees the convergence when $i \to \propto$ and it indicates that the expected Q value is equal to the immediate reward plus the discounted value of the expected next state. Using this equation in iteration over all of the state-action pairs is impractical for most cases where we are faced with a high dimensional state space so in this case we use a function approximator, such as an artificial neural network, to approximate $Q^*(s, a)$.

$$Q_{i+1}(s, a) = \mathbb{E}\left[r + \gamma \max_{a'} Q_i(s', a') | s, a\right] \tag{2}$$

A function approximation gives us an estimates of the Q values for each of the possible next actions given the input of the state-action pair. After each iteration one computes the expected Q value using Equation 2 and compare it to our previous estimate $Q(s, a; \Theta) \approx Q^*(s, a)$ that we receive by forwarding the state-action pair at the input of the ANN. The difference between our previous estimate and our expectation is then backpropagated through the ANN in order to update our current approximation of $Q^*(s, a)$.

Backpropagation is performed by updating the weights of the neural network $\Theta$ by performing a gradient descent on the loss $L_i(\Theta_i)$ according to Equation 3:

$$\nabla_{\Theta_i} L_i(\Theta_i) = (y_i - Q(s, a; \Theta_i)) \nabla_{\Theta_i} Q(s, a; \Theta_i), \tag{3}$$

where $y_i = r + \gamma \max_{a'} Q(s', a'; \Theta_{i-1})$ represents the Bellman equation as our target value.

## Model Architecture and Learning Algorithm

The agent performs a transition in the environment by moving from state $s_t$ to next state $s_{t+1}$ using action $a_t$ and experiencing a scalar reward $r_t$ in the process. These observed values contain enough information to perform an iteration of a learning process by updating our previous belief about the value $Q(s, a)$ of the state-action pair. This update is performed at $(d)$ block of Figure 1 as a part of the main learning loop $(b)$ and it consists of updating the weights vector $\Theta$ of artificial neural network detailed in Figure 2 making our approximation of $Q$ a step closer to the target value as given by Equation 2. The process creates an experience stream of sequential transitions the agent has experienced from the beginning of the learning process. Each of these experiences carries a potential for re-learning that would be lost if they are discarded after the initial update; therefore, instead, they are recycled through a perceptive buffer called *replay memory* as shown in $(a)$ section of Figure 1.

Due to the limited capacity of the memory buffer, this form of perception, just like in humans, should be able to be selective over types of experiences that are being sampled. The perception itself becomes a kind of "reality filter", able to influence how does the agent gather information from its environment and further interacts with it to reach its goal. The premise of this approach is that the contents of the *replay memory* that represents a subset of the *experience stream* form the perceptive layer able to provide a secondary drive that, in turn, influences the agent's behaviour. The idea of perception being a filter of experiences that protects the agent from oversaturation is implemented through stochastic sampling shown in the *artificial perception* box included in block $(f)$ of Figure 1. As we can see from Algorithm 1 the *artificial perception* block is implemented as a probability of sampling the transition $P(i)$ into *replay memory* structure $D$.

## Experimental Setup

Experiments were performed to compare the performance of six agent types implementing different perception dynamics or sampling strategies in two different environments: Waterworld and Lunar Lander. The first one, Waterworld Karpathy (2013) represents a continuous learning process implementing a high-dimensional state space and a crisp reinforcement function, while a more applicable and realistic setup of Lunar Lander Brockman et al. (2016) environment is characterized by an episodic task, with a continuous and complex reinforcement function, and a low dimension of state space. Waterworld was implemented in three variations depending of the good to bad food ratio, which together with Lunar Lander evaluation accounted to a total of four separate batches of experiments. Each of the agent types were evaluated separately in a single-agent setting and the agents were initialized at a random position with a random approximation ANN. The effectiveness of each agent was measured by a total cumulative reinforcement received, or how well it performed in the given environment.

**Algorithm 1** Q-learning with artificial perception block

Initialize replay memory D with capacity N
Initialize action-value function Q with random weights
**for** episode = 1, M **do**
  **for** t = 1, T **do**
    With probability $\epsilon$ select a random action $a_t$
    otherwise select $a_t = \arg\max_a Q^*(s_t, a; \Theta)$
    Execute action $a_t$, observe reward $r_t$ and state $s_{t+1}$
    Store transition $(s_t, a_t, r_t, s_{t+1})$ in $D$ according to the probability $P(i)$
    Sample random batch of transitions $(s_t, a_t, r_t, s_{t+1})$ from $D$

$$\text{set } y_i = \begin{cases} r_i, & \text{terminal } s_{i+1} \\ r_i + \gamma \max_{a'} Q(s_{i+1}, a'; \Theta), & \text{non terminal} \end{cases}$$

    Perform a gradient descent step on $(y_i - Q(s_i, a_i; \Theta))^2$ according to Equation 3
  **end for**
**end for**

### *Waterworld environment*

The Waterworld environment as a part of ReinforceJS framework Karpathy (2013) consists of food pieces that are generated as they are consumed, at a random position with a random velocity vector. They move freely in the environment. There are two types of food: good and bad. The dynamics of this specific environment allowed us to perform different sets of experiments, exploring the effect of different amounts of food. In the first one, the number of bad and good food pieces in the environment are the same, namely 2 of each type. In the second one, the proportion of good and bad food is brought to 2:1, making it *benevolent*. In the third, *hostile* one, the proportion of good and bad food is inverted so that the environment contains an amount of bad food twice than the good one.

The agents aim at discovering an optimal policy that would allow them to eat (touch) the most good food pieces, while avoiding as much as possible the bad ones. The reinforcement is, respectively, +1 for the good, and -1 for the bad food sources that the agent can reach. The environment contains equal amounts of good and bad food pieces and the ratio is kept constant by regenerating the consumed piece of food.

The agent perception includes 30 directional sensors, each of which can detect 5 continuous variables: distance and speed in $x$ and $y$ direction of the different perceived objects that include good food, bad food, and distance to wall or the boundaries. This, with an agent's own speed components in $x$ and $y$ direction accounts for a quite high dimensional state space of 152 continuous variables.

Approximation of $Q(s, a; \Theta) \approx Q^*(s, a)$ is obtained using an ANN with one hidden fully connected layer of 100 neurons, which are producing as output the Q values of all five actions available to the agent: up, down, left, right, stay.

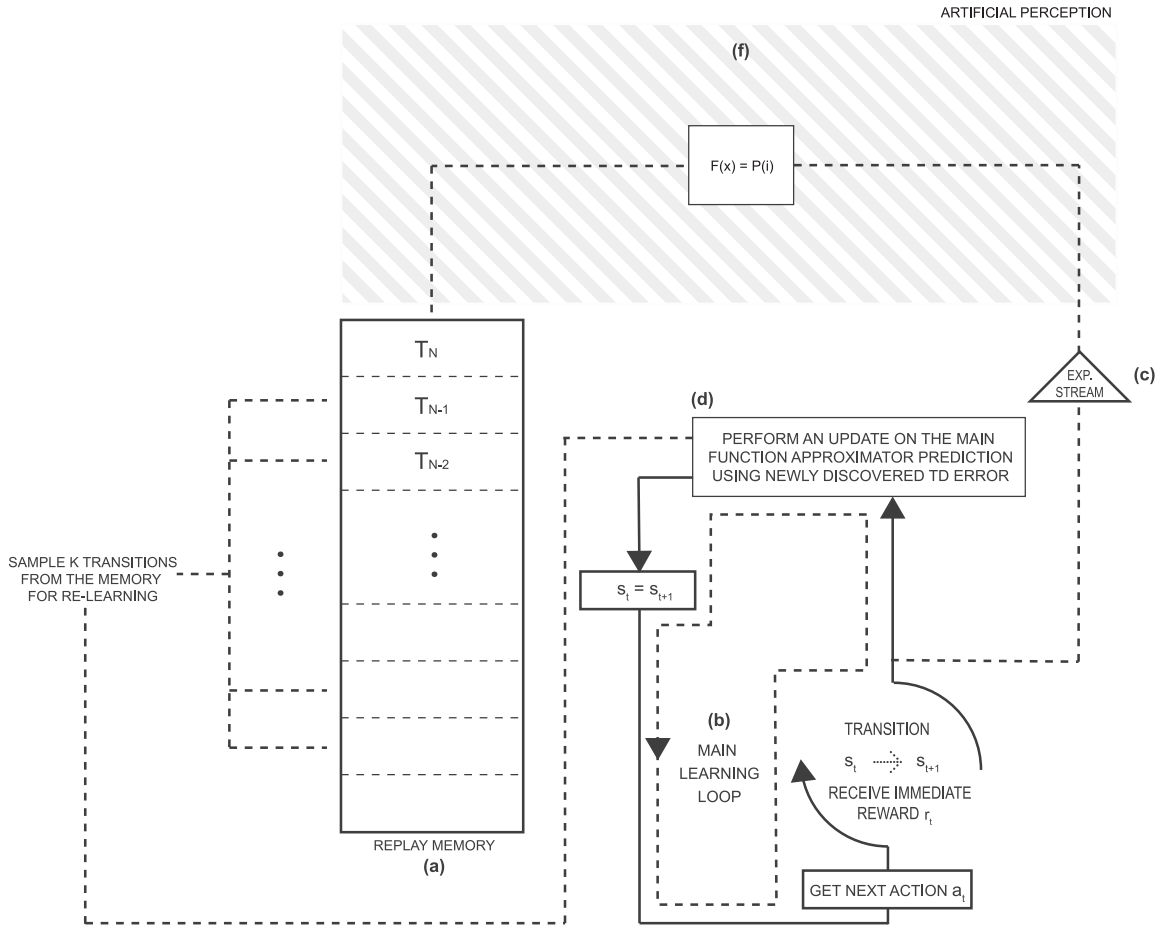


**Figure 1.** General learning model architecture including *attention focus block*: $(a)$ Replay memory buffer; $(b)$ Main learning loop; $(d)$ Q-value function approximator detailed in Figure 2; $(c)$ Raw stream of the experiences; $(f)$ Artificial perception block

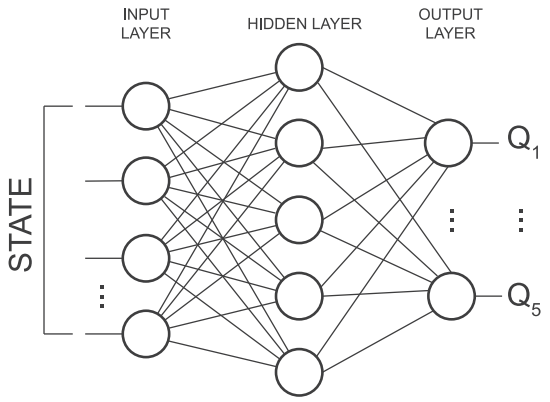


**Figure 2.** Main function approximator ANN implemented in the (d) block of Figure 1: it receives a 152-dimensional state as its input and approximates it to $Q$ values for each of the five possible actions available to an agent at its output, therefore providing an approximation for $Q(s, a)$ pairs.

The learning rate of an approximator $\alpha$ is set to a low 0.05 and the capacity of the replay memory buffer is 5000. The value of $\epsilon$ is set to 0.2 at the beginning and adjusted to 0.1 at the 30.000-th step of the learning to exploit more the learned behavior. The discount factor $\gamma$ is set to 0.9.

## *Lunar Lander environment*

A part of OpenAI Gym framework by Brockman et al. (2016), Lunar Lander environment consists of a craft equipped with thrusters attempting to land to a designated platform on a surface under the influence of a weak lunar gravity.

The craft has a total of four discrete actions at its disposal: do nothing, fire main thruster, fire left thruster, fire right thruster. The world state as perceived by the agent consists of 8 variables; crafts angle, angular velocity, $x$ and $y$ components of velocity, $x$ and $y$ position relative to the landing platform and a contact of each of the two legs with the surface.

The reinforcement function is proportional to the distance of the craft to the landing platform while each leg contact is awarded by +10 and usage of a main engine by -0.3. An episode finishes if the lander crashes or comes to rest, receiving additional -100 or +100 of reinforcement.

The experiment consisted of 20 batches each running 1600 episodes, which proved to be more than enough to consider the environment as solved. The starting $\epsilon$ of 1.0 was decayed after each episode multiplying it by 0.998. $\gamma$ was set to 0.99. Artificial neural network was initialized with two hidden layers which consisted of 128 and 64 nodes respectively with the learning rate parameter set at a low 0.0001.

### Perception Dynamics

In order to see how does the *artificial perception* component contribute to behavioral characteristics and performance, a total of six behavioral traits or personalities are outlined in Table 1 each of them corresponding to a different perception dynamics given by their experience sampling strategy. The sampling strategies induce dynamics by modifying two dimensions of the experience stream: the type of the transition given by the reward obtained and the amount of transitions sampled, reported in Table 2. The rows of the table are showing the transition type which is *good* if the reinforcement is positive, *bad* if it is negative, and *random* if the agent doesn't care about the reinforcement. A low or high amount of each type of experience can be sampled according to what reported on the columns of Table 2. This gives us a total of six combinations of perception dynamics; each one corresponding to an agent's personality disposition. The amount of experience columns divide the agent types in two meta-groups, which can be associated with the extroversion/introversion main axis found in major personality indicators such as John et al. (1999); Briggs (1976). Extroverted agents have a tendency to sample more experiences while the introverted ones are conservative and tend to limit the amount of sampling.

The actual probability values used in varying sampling dynamics are reported in Table 3.

**Table 1.** Mapping of agent's personality traits and sampling strategies.

| Emotion state | Strategy |
|---|---|
| happy | There is something good in any experience. It selects a lot of experiences, randomly. |
| sad | There is always something bad in new things. It selects randomly few experiences. |
| fearful | It is afraid to get bad things. It selects a good number of negative experiences, so to learn to avoid them. |
| greedy | Always get as much as possible. It selects a lot of good experiences, discarding the bad ones. |
| cautious | It aims for good, but it tends to be conservative. It selects few good experiences. |
| provident | It is ready to face problems, and it conservative. It selects few negative experiences to learn from them. |

**Table 2.** Agent personality trait types represented as combinations of two sampling dimensions with columns representing the quantity of sampled transitions and rows represent the type.

| amount/type | few | many |
|---|---|---|
| good | cautious | greedy |
| bad | provident | fearful |
| random | sad | happy |

**Table 3.** Agent sampling strategies probability values used in experiments.

| type/amount | few | many |
|---|---|---|
| good | p(.05) or (if(r=1) and p(.5)) | p(.05) or (if(r=1) and p(1)) |
| bad | p(.05) or (if(r=-1) and p(.5)) | p(.05) or (if(r=-1) and p(1)) |
| random | p(.15) | p(.5) |

## Experimental Results

### Waterworld environment

Normal environment which contained equal amounts of positive and negative reinforcement sources showed a divergence in agents score implementing different perception dynamics as we can see from Figure 3. We can also notice that in this type of balanced environment the conservative sampling strategy of the *provident* agent type proved to be most effective, while the optimistic *happy* agent was the worst. Agents with *cautious* and *greedy* attitude performed slightly better than the conservative one, and the random sampling strategy of the *sad* one still underperformed when compared to the second best *fearful* strategy.

Hostile environment showcased in Figure 4, which was mostly populated with negative reinforcement (ratio good/bad food is 1:2), shows even more divergence among the agent types than the normal one. This harsh environment have proved to be best faced by the agents that adapted a *fearful* strategy allowing them to thrive and perform far more than the other types. The conservative strategies of *provident* agents also showed adaptability scoring second best followed by an ultra-optimistic *happy* one. The strategies that haven't adapted to this kind of environment were the ones that focused on positive experiences, such as *cautious* and *greedy* and the conservative random approach of *sad* agent.

Figure 5 compares the performance in a benevolent environment containing more positive than negative reinforcement (ratio 2:1). This environment variation was faced in a similar way by all the perception dynamics, except the optimistic *happy* agent that underperformed despite the high availability of the positive food sources.

### Lunar Lander environment

Contrasting the results on default or *Normal* configuration of Waterworld reported in Figure 3, from the Figure 6 it is evident that the Lunar Lander environment favored the random approaches of optimistic *happy* agent and the conservative *sad* agent. The third best performance belonged to a selective, but conservative, *provident* type followed by an ultra-conservative *fearful* agent. The *greedy* and *cautious* agent types performed the worst in the Lunar Lander

environment dynamics. which proved to be consistent with the *hostile* variation of Waterworld shown in Figure 4.

## Discussion

Looking at the all-round performance across the environments and their variations it seems that conservative, cautious strategies like *provident* and *fearful* deliver the best performances. The fearful approach to perception seems to give a good consistent performance in all of the experiments in the setup, suggesting that negative experiences play a more important role than the positive ones in the learning process. Even in environments that are scarce in positive reinforcement like the hostile Waterworld, focusing on the negative experience still gives a performance that is equal or better than the other sampling strategies. Not surprisingly, the *fearful* agent performs the best in an environment that is hostile by focusing on avoidance rather than exploration, but, interestingly, it provides an effective strategy also for the environments that are by nature more supportive.

The sampling strategy implemented through *provident* agent's narrower selection of negative experiences is insightful of the nature of perception itself. Its superior performance in the baseline Waterworld and a good overall performance in Lunar Lander gives us a clue about the fact that perception is inherently highly selective of experiences and that it can be best represented as a filter with respect to the environment. Acting as a mediator of agent's cognition it can protect its cognitive gateway of limited capacity, the replay memory, from being oversaturated with experiences. Oversaturation posed a problem for a Waterworld environment, but it had a positive effect on Lunar Lander, where a larger replay memory buffer was implemented. All of the variations of Waterworld showed a low performance of the *happy* strategy, while the same strategy proved to be the best in Lunar Lander environment. In the Waterworld and its variations the "happy" selection of a lot of random experiences oversaturated the replay memory quickly and this had a great effect on the agent's performance.

In Waterworld variations the *cautious* and *greedy* strategies of focusing on the good experiences proved to be a good approach in the baseline, where they outperformed the random ones of *happy* and *sad*, but due to the lack of positive reinforcement in hostile world they could not achieve high performance there.

On the contrary, *cautious* and *greedy* preference for positive experience proved to be the least favorable approach considering the dynamics of *Lunar Lander* environment. Interestingly enough, the lack of performance of *cautious* and *greedy* agent types became evident only in the long run while their performance over the first 6000 episodes was comparable or even better than the one of the other types. It seems that in these early stages of learning the focus on positive experiences enabled the agent to learn the more rewarding landing tactics and to progress more quickly, while the lack of negative experiences prevented it to further perfect them. The hostile nature of *Lunar Lander* environment favored the agents that were inclined towards negative experiences as we can notice from the performance of *provident* and *fearful*, but the best trend throughout for this environment belongs to the neutral dispositions such as *happy* and *sad*.

Overall the agents that focused their perception on the positive experiences showed a more exploratory behavior, while the ones that focused on the negative were more conservative and static.

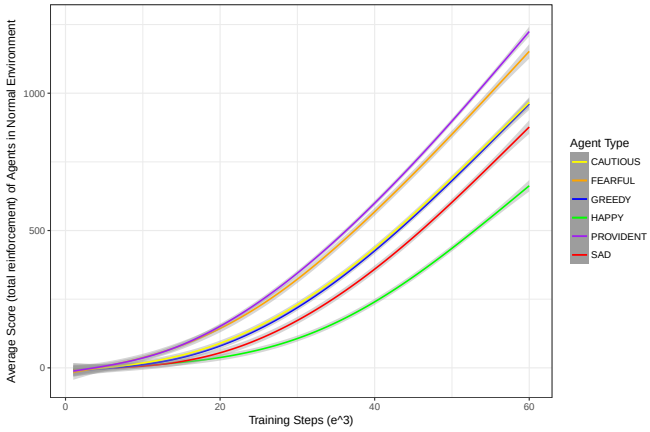


**Figure 3.** Average score or total reinforcement received over 50 trials with agents with different perception dynamics in Normal environment type during first 60.000 learning steps.

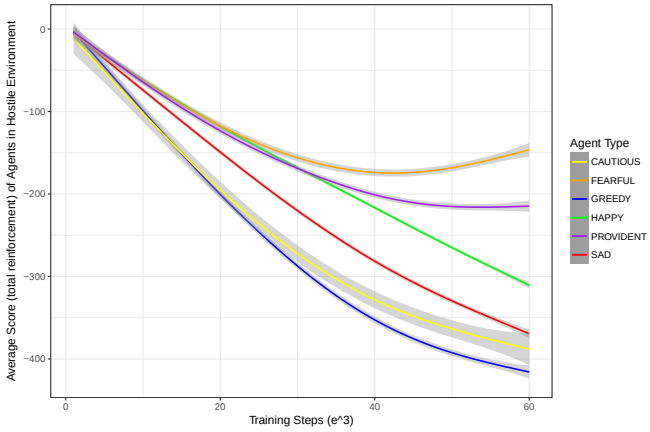


**Figure 4.** Average score or total reinforcement received over 50 trials with agents with different perception dynamics in Hostile environment type during first 60.000 learning steps.

## Conclusion

We presented a model of *artificial perception* capable of modifying the way an agent selects information from its environment during learning. The perception layer creates additional possibility for influencing the agent's behavior which can be used as a secondary goal-oriented drive in addition to the reinforcement function. Using this technique, an agent is able to better adapt its learning to a specific environment only by changing the use of its perceived experiences without modifying its reinforcement function. The main point is that the algorithms simplicity implemented only by a cognitive filter can elicit a complex behavioral pattern without defining motivational goals or drives as in Sun and Wilson (2014b,a) or modifying the experience as in Recchia et al. (2014). Compared to Dyna

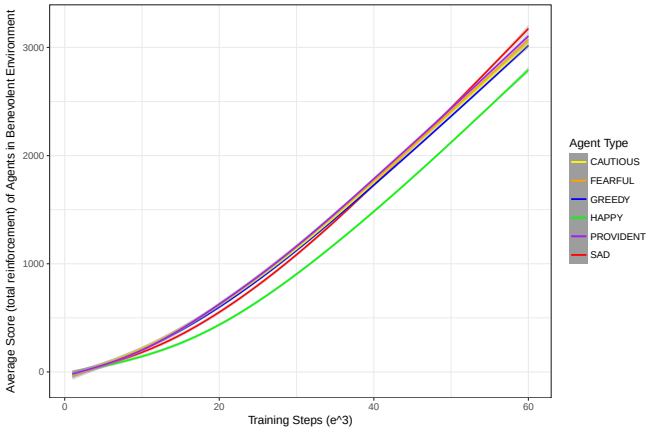


**Figure 5.** Average score or total reinforcement received over 50 trials with agents with different perception dynamics in Benevolent environment type during first 60.000 learning steps.

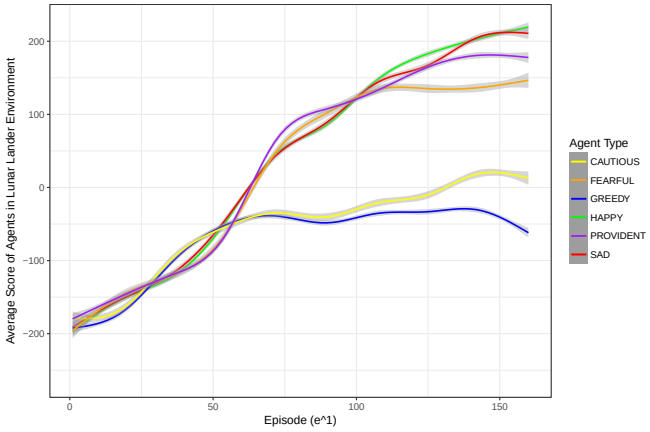


**Figure 6.** Average score or total reinforcement received over 20 trials with agents with different perception dynamics in Lunar Lander environment during first 1600 episodes.

approaches Sutton (1990); Sutton et al. (2012) and *internal simulation* ones Jirenhed et al. (2001); Ziemke et al. (2005) our algorithm relies only on filtering in different ways the actual data perceived from the environment.

Looking beyond the effectiveness, *artificial perception* can create a contextual filtering buffer between the agent's cognition and the learning algorithm, possibly producing more efficient data exploitation, and preventing the mechanism of replay memory to be oversaturated, thus becoming ineffective.

## References


Ahn H and Picard RW (2005) Affective-cognitive learning and decision making: A motivational reward framework for affective agents. In: *International Conference on Affective Computing and Intelligent Interaction*. Springer, pp. 866–873.

Briggs KC (1976) *Myers-Briggs type indicator*. Consulting Psychologists Press Palo Alto, CA.

Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J and Zaremba W (2016) Openai gym.

Gadanho SC and Hallam J (1998) Exploring the role of emotions in autonomous robot learning. *DAI RESEARCH PAPER* .

Goleman D (2006) *Emotional intelligence*. Bantam.

Jirenhed DA, Hesslow G and Ziemke T (2001) Exploring internal simulation of perception in mobile robots. In: *Proceedings of the Fourth European Workshop on Advanced Mobile Robots*. pp. 107–113.

John OP, Srivastava S et al. (1999) The big five trait taxonomy: History, measurement, and theoretical perspectives. *Handbook of personality: Theory and research* 2(1999): 102–138.

Karpathy A (2013) Reinforcejs framework. `https://github.com/karpathy/reinforcejs`. Accessed: 2018-09-06.

Lin LJ (1993) Reinforcement learning for robots using neural networks. Technical report, DTIC Document.

Metcalf K, Theobald BJ and Apostoloff N (2018) Learning sharing behaviors with arbitrary numbers of agents. In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 1232–1240.

Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D and Riedmiller M (2013) Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* .

Ramicic M and Bonarini A (2017a) Attention-based experience replay in deep q-learning. In: *Proceedings of the 9th International Conference on Machine Learning and Computing*. ACM, pp. 476–481.

Ramicic M and Bonarini A (2017b) Entropy-based prioritized sampling in deep q-learning. In: *Image, Vision and Computing (ICIVC), 2017 2nd International Conference on*. IEEE, pp. 1068–1072.

Ramicic M and Bonarini A (2019) Selective perception as a mechanism to adapt agents to the environment: An evolutionary approach. *IEEE Transactions on Cognitive and Developmental Systems* : 1–1DOI:10.1109/TCDS.2019.2896306.

Recchia T, Chung J and Pochiraju K (2014) Performance of heterogeneous robot teams with personality adjusted learning. *Biologically Inspired Cognitive Architectures* 7: 87–97.

Rumbell T, Barnden J, Denham S and Wennekers T (2012) Emotions in autonomous agents: comparative analysis of mechanisms and functions. *Autonomous Agents and Multi-Agent Systems* 25(1): 1–45.

Salichs MA and Malfaz M (2006) Using emotions on autonomous agents. the role of happiness, sadness and fear. *Integrative approaches to machine consciousness, part of AISB* 6: 157–164.

Sato S, Nozawa A and Ide H (2004) Characteristics of behavior of robots with emotion model. *IEEJ Transactions on Electronics, Information and Systems* 124: 1390–1395.

Schaul T, Quan J, Antonoglou I and Silver D (2015) Prioritized experience replay. *arXiv preprint arXiv:1511.05952* .

Shanahan M (2006) A cognitive architecture that combines internal simulation with a global workspace. *Consciousness and cognition* 15(2): 433–449.

Sun R and Wilson N (2014a) A model of personality should be a cognitive architecture itself. *Cognitive Systems Research* 29: 1–30.

Sun R and Wilson N (2014b) Roles of implicit processes: instinct, intuition, and personality. *Mind & Society* 13(1): 109–134.

Sutton RS (1990) Integrated architectures for learning, planning, and reacting based on approximating dynamic programming.

In: *Machine Learning Proceedings 1990*. Elsevier, pp. 216–224.

Sutton RS and Barto AG (1998) *Reinforcement learning: An introduction*. Cambridge, MA: MIT press.

Sutton RS, Szepesvári C, Geramifard A and Bowling MP (2012) Dyna-style planning with linear function approximation and prioritized sweeping. *arXiv preprint arXiv:1206.3285* .

Tsankova DD (2009) Emotional intervention on an action selection mechanism based on artificial immune networks for navigation of autonomous agents. *Adaptive Behavior* 17(2): 135–152.

Zhai J, Liu Q, Zhang Z, Zhong S, Zhu H, Zhang P and Sun C (2016) Deep q-learning with prioritized sampling. In: *International Conference on Neural Information Processing*. Springer, pp. 13–22.

Ziemke T, Jirenhed DA and Hesslow G (2005) Internal simulation of perception: a minimal neuro-robotic model. *Neurocomputing* 68: 85–104.