Temporal perspectives: Exploring robots' perception of time

Inês de Miranda de Matos Lourenço
ineslourenco@tecnico.ulisboa.pt
Instituto Superior Técnico

***Abstract*—Time perception is a concept used to represent the phenomenological experience of time by an individual, present in every activity of our daily lives. Sensory information has been proven to have an impact in the way we perceive the passage of time. Artificial agents, however, perform their actions based on functions that assume a linear metric of time given by a clock, and lack a variable sense of time.**

The first part of the work of this thesis consists on studying whether an artificial agent is able to estimate time through the second-order statistics of the natural environment, assuming these behave like Gaussian processes with an Ornstein-Uhlenbeck covariance function. It was concluded that other models should more correctly represent the statistics of the sensory streams. The resulting estimate could act as a time basis for the robotic tasks.

The second part focuses on the implementation of a temporal task in a Reinforcement Learning problem. The firing rate of dopaminergic neurons resembles the Temporal-Difference (TD) error in TD learning algorithms. However, TD errors produced by models possessing precise temporal representations, such as those in computers, fail to capture observed patterns of dopaminergic activity. Alternative temporal representations that reflect increasing uncertainty about elapsed time with duration may more accurately capture observed neural and behavioural data from animals, as is the case of Microstimuli. In the implemented problem traditional algorithms are compared to biologically inspired ones, proving the latter not only to correctly represent the dopamine, but also being the most computationally efficient.

***Index Terms*—temporal perception, reinforcement learning, gaussian processes, robotics, microstimuli.**

I. Introduction

Even though nowadays a big amount of information about the human brain has been discovered, a lot remains unknown. The work presented in this paper focuses on the reproduction of specific brain mechanisms, using biologically inspired approaches to validate neuro-scientific theories in artificial agents. One of the neural mechanisms is commonly called time perception, and is a concept used to describe the phenomenological experience of time by an individual. Multiple brain areas, such as the frontal cortex, basal ganglia, parietal cortex, cerebellum, and hippocampus, are known to be responsible for the way we perceive time.

Successfully reproducing and implementing these processes in an artificial agent could eventually bring advantages that have surprisingly not yet been explored. Studying some properties of this phenomenon should therefore be a good way to bring new ideas about this still so unknown area and make the reader think more deeply about the (dis)advantages, implications and consequences of having a perception of time, that can be used to explore the real concept behind sentences such as "Time flies when we are having fun", [1].

Even though the multiple areas of the brain have different ways to represent time according to the tasks they perform, and there is not any evidence that one specific area of the brain might be responsible for temporal cognition, this is not what happens in robotics. Current algorithms consider that time comes from a specific central mechanism, a clock, ordinarily treated like a variable that is represented by an index and has its own dimension. The idea is that instead, the time basis of robots should come naturally and adequately for each algorithm and be dynamically applied in its functions.

The first step of this thesis is the estimation of the elapsed time through sensory information, which can contribute to:

- Validate the idea that external stimuli influence the perception of time
- Study the possibility of estimating time through sensory information
- Find a suitable model for the second order statistics of the environment
- Since neither humans or animals have an internal clock like robots do, study whether applying brain principles to artificial agents can be advantageous for their performance in certain tasks.

In the second step, a reinforcement learning problem is conducted using time representations that are supposed to be similar to the ones used by the brain. The contributions are:

- Checking how accurate each of these representations is
- Even though different time representations have previously been studied as explained in section II-B, to the best of the authors knowledge this has never been applied in a reinforcement learning problem with action selection before.
- Compare the efficiency of biologically inspired *vs.* traditional reinforcement learning algorithms.

Combined, the two steps can give rise to a biologically inspired model in which an agent navigates through an environment, collects information from the sensors and creates an estimate of the time that has elapsed, which can be converted into a reinforcement learning state that can be used to teach the robot to succeed in a temporal task.

Section II provides an overview of the most important concepts needed to fully understand the subject, section III focus on the framework for the first step, this is, a Bayesian

Model of sensory change to estimate time, and section IV focus on a specific function that can be performed using the estimated time, in particular, a reinforcement learning time-dependent task, with the representation of time called Microstimuli. Finally, in section V, some conclusions regarding the developed work are taken and future guidelines are defined.

II. Background

As an interdisciplinary work, this thesis uses knowledge from different areas such as Machine learning, Neurosciences and Psychology.

A. Reinforcement learning

Reinforcement learning (RL) is used as a way for an agent to learn which actions to do when acting in an environment, in order to maximize the number of expected accumulated discounted rewards to be obtained.

The agent interacts with the environment in the following discrete way: at a certain moment it is in a certain state $s \in$ S, and, according to the chosen action $a \in$ A that it chooses to apply in the environment it moves to a state $s \in$ S in the next timestep. With these state transitions there is a reward associated, that evaluates how good the action seems to have been. The goal of the reinforcement learning agent is therefore to maximize the numerical reward signal, by learning how to predict the expected value of a sequence of states.

Its learning method can vary. Here we focus on Temporal-Difference (TD) learning methods, explained in the next section.

1) Temporal-Difference learning : TD learning algorithms are often used in RL to predict the future expected amount of reward. Here, a value function for the future actions is learnt based on the current state and previous actions and rewards, but the agent has to learn how to adjust his predictions to match the future rewards before the end of the sequence.

One TD learning algorithm, that focus on learning the action-value function $Q(s,a)$ instead of the value function $V(s)$ is described in [2] and follows

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1},a_t) - Q(s_t,a_t)] \quad (1)$$

$Q(s_t,a_t)$ is the maximum expected future reward starting from state s_t, and r_{t+1} is the total discounted cumulative reward, that represents the expected return from all future actions. The error function, called TD error, δ_t, is given by the second term of the right side of the equation and represents the error made when estimating the value of a state. It is given by the difference between the estimated reward at a given state and the actual reward received. Here, $\gamma \in [0,1]$ is the discount rate and is used to compare the importance of future rewards relatively to immediate ones. The TD error is multiplied by a parameter $\alpha \in [0,1]$, called learning rate, that represents how much importance is given to the new information compared to the previous one and can be used to optimize the speed of learning.

This formulation is used for tabular representations of the problem, where the number of states and actions is small enough to allow a representation in a table. On the other hand, when that is not the case a function approximation formulation can be used instead, in which each state is represented by a set of features ϕ. The equations that represent the Q-learning algorithm in this case are the following:

$$w_{t+1}(d) = w_t(d) + \alpha \delta_t e_t(d) \quad (2)$$

$$\delta_t = r_t + \gamma Q(s_{t+1}) - Q(s_t) \quad (3)$$

$$e_{t+1}(d) = \gamma \lambda e_t(d) + \phi_t(d) \quad (4)$$

The TD error δ comes again from the comparison of the received reward r in each timestep with the expected one. Here the TD error is multiplied by e, called eligibility traces, that extend the influence of the presence of a state through time, influencing the weight w given to each one regarding its responsibility for a certain reward received. The parameter λ is introduced, as the decay parameter that determines the plasticity window of recent stimuli.

The Q-values are calculated by:

$$\begin{aligned} Q(s,a) &= w_1\phi_1(s,a) + w_2\phi_2(s,a) + ... + w_N\phi_N(s,a) \\ &= \sum_{i=1}^{d} w_i\phi_i(s,a), \end{aligned} \quad (5)$$

concluding that the goal is adjusting the weights according to the importance of each feature in each decision, without having to adjust all the individual state-action values as in the tabular case. The eligibility traces, e_t, combine the information of how frequent and how recent a state is, acting as a memory trace of the recent happening of an event. It acts as a scaling factor for the TD error, triggering value updates proportional to the recently visited states whose trace is different from zero. The representation used in the algorithms was given by:

$$e_t(s,a) = \begin{cases} 1 + \gamma\lambda e_{t-1}(s,a), & \text{if } Q_{t-1}(s_t,a_t) = \max\limits_a Q_{t-1}(s_t,a) \\ 0, & \text{if } Q_{t-1}(s_t,a_t) \neq \max\limits_a Q_{t-1}(s_t,a) \\ \gamma\lambda e_{t-1}(s,a), & \text{otherwise.} \end{cases} \quad (6)$$

B. Time Perception in biology

An important parameter of RL algorithms with feature representation is the way how time is represented, and for a proper representation understanding how it is done in the brain conveys is an important task.

One hypothesis that is believed to provide an accurate explanation about how time is encoded and represented in the brain is how the role of the dopaminergic system can be seen as an internal clock in the brain. It is believed by some that time is intrinsically encoded in the neural populations of the different regions of the brain due to their time-dependent changing behaviour, being an inherent property of the underlying neural dynamics [3]. This is in line with the research in [4], that shows that the firing pattern of a group of neurons evolves in a decodable way with time. More specifically, they showed that the dynamics of striatal neural populations could predict the judgement of durations made by the agent, through the speed with which their state changes. So these changes can be seen as the basis of animals to judge the passage of time.

C. Gaussian Processes

A Gaussian process [5] is a stochastic process where any finite set of random variables follows a multivariate Gaussian distribution, $X \sim \mathcal{N}(\mu, \sigma^2)$. Gaussian processes for modelling selection and prediction of observed data became very useful, since, as the extension of multivariate Gaussians to infinite-sized collections of real-valued variables, the same properties are maintained and the problems are not only consistent but also computationally tractable, making learning and inference easy.

One of the common uses of gaussian processes is in *Bayesian inference*. Bayesian inference begins with a prior distribution that is updated as data points are observed, from which the posterior distribution over functions is obtained. The Gaussian Process assumes that $p(f(x_1), , f(x_N))$ follows a jointly distributed Gaussian distribution, with mean $\mu(x)$ and covariance $cov(x)$. This covariance is given by a kernel function k, that dictates the similarities of the relation of x_i and x_j with y_i and y_j, $k(x_i, x_j)$. The fact that they are completely described by the mean and covariance (second order statistics) is what makes them so useful.

Covariance functions are the crucial part in the prediction of Gaussian processes since these reflect the assumptions about the model. The shape and parameters of the covariance function are what defines the function to be learnt, therefore reflecting the differences in the processes behaviour.

A special case of the Matérn Class of covariance functions gives the stationary Ornstein-Uhlenbeck covariance function, a Brownian motion process with friction [6]. Its covariance function is given by:

$$K(\tau) = \exp(-\lambda|\tau|) + \sigma^2\delta(\tau) \tag{7}$$

In the first term, τ is the difference between two time intervals and λ is the inverse of the length-scale parameter, l, that represents how "how close" two points x and x' have to be to influence each other significantly. The second term represents the presence of noise in the model, in which δ is the Kronecker δ and σ is the standard deviation of the noise fluctuations, that models instantaneous noise in the observation process for the sensory stream. Together, λ and σ are the hyperparameters of the model, this is, the variables that define the covariance function, and consequently the process.

Model selection is the method of estimating the values of the hyperparameters of the covariance function that better represent the statistical properties of a certain process. Using Bayesian inference, the hyperparameters are selected based on the Maximum *a posteriori* (MAP) of the chosen prior. If the prior is uniform, the MAP corresponds to the Maximum Likelihood Estimate (ML). The processes y thus become completely defined by the hyperparameters of K, that maximize the expression

$$\log p(y|X,\theta) = -\frac{1}{2}y^T {K^{-1}}_y - \frac{1}{2}\log|K_y| - \frac{n}{2}\log 2\pi \tag{8}$$

III. From Sensing to Time

This chapter describes the first main algorithm implemented in this work, whose goal is to estimate the passage of time from information received and gathered from the environment through the sensors. This problem can be divided in multiple steps:

The first consists on how time is coded in the brain and was explained in II-B.

The second concerns how this is extended into computational models that can apply the same principles and give temporal cognition to artificial agents. Multiple papers have done this. For example, in [7] the authors implemented a time perception model that uses a coevolutionary neural network to encode the chain of temporal events. In [8] an artificial agent learns from a Genetic Algorithm in which each of the artificial chromosomes is used to encode a different configuration of a self-organizing Continuous Time Recurrent Neural Network (CTRNN), with the goal of teaching the agent how to perceive the duration of sounds and act accordingly. In [9] they used a CTRNN model with a circuit inspired by neurocortical connectivity that was able to discriminate different temporal patterns, in a way that, without being programmed to do so, naturally transformed the temporal information into a spatial code.

In third place, besides knowing how time perception is encoded in the brain, it is important to know, when it comes to the use of external information from the sensors, how it affects our internal neural mechanisms. This starts by how sensory information has been previously modelled. In [10] it was concluded that seeing a moving stimulus acted as a bias for the perceived length of the elapsed time. The degree of this bias was concluded to be proportional to the speed of the movement, that is, the faster the objects moved, the longer the duration was perceived.

The final question to be answered regards how, from this sensory information, estimation of time has been addressed. As for this matter, to the best of the authors' knowledge very few information can be found. One exception is the paper [11], that followed an interesting direction and used a mathematical approach to show that through a process of stochastic sensory stimuli, an estimate of time can be obtained and combined with the internal estimation of time in the brain. This framework is going to be a basis for this first part of the paper, and is explained in more detail in the next section.

A. Theoretical Framework

In the paper [11] published in 2011 by Misha B. Ahrens and Maneesh Sahani, a Bayesian model is used to combine our internal perception of time with an innate sensory-based estimation of duration based on the probabilistic expectations of stimulus change. The sensory information is based on the study of the statistical properties of external dynamic stimuli, more precisely, the second-order statistics of the natural environment. The idea is to use these statistics to model sensory streams as Gaussian Processes, from which the Bayesian observer takes measurements with the goal of estimating the elapsed time between them. The suggested Bayesian model is given by

$$P(\tau|\text{obs.}) \propto P(\text{obs.}|\tau) \times P(\tau) \tag{9}$$

in which τ is the elapsed time and obs are the observations taken from the environment. $P(\text{obs.}|\tau)$ represents the likelihood, through the knowledge of the environment and $P(\tau)$ is the prior, internal stimulus-independent belief about the elapsed time. $P(\tau|\text{obs.})$ is the posterior distribution whose peak corresponds to the maximum *a posteriori* (MAP). This MAP will be considered the estimate of elapsed time by the agent, combining information from both internal functions of the brain as well as sensorial.

The prior distribution representing our internal estimate of time without the use of the sensors is still not clear and well studied, reason for which an uniform distribution is used. This means considering that our brain does not have *a priori* information about the elapsed time, giving therefore all the importance to the information received from the sensors. In this case, the MAP will be equivalent to the maximum of the likelihood, reason why the estimate is, in this case, called ML (Maximum-Likelihood) estimate.

As for the likelihood function, it represents how likely it is that, for a certain time interval, the data has been observed. This is, what is the probability of observing the data for each time interval. Maximizing this corresponds to finding which time interval is the most likely to have really passed given that those observations were collected.

An important feature to take into consideration is that both images and sound do not change randomly, rather, they are structured and regular at many levels of complexity, showing patterns of high correlation in both space and time [12]. This is usually simplified by focusing only on the second-order statistics of the environment, in order to avoid handling an excessive amount of information. These statistics are given by a correlation matrix between two points in a natural time-varying image. Furthermore, assuming that these statistics are stationary in time, the mean becomes constant. It is then assumed that the sensory processes can be seen as Gaussian and stationary.

The computed estimates are therefore equivalent to the ones given by an ideal observer of stationary Gaussian Processes, and, as so, for each of the possible elapsed times, the probability of having observed the data is given by a joint Gaussian distribution over observations, completely specified by the covariance function of each sensory process, at N specified times

$$\begin{aligned} &P(y_i(t_1), y_i(t_2), ..., y_i(t_N)|t_1, ..., t_N) = \mathcal{N}(0, K_i) \\ &= \det[2\pi K]^{-\frac{1}{2}} \exp -\frac{1}{2}[y(t_1), ..., y(t_N)]K^{-1}[y(t_1), ..., y(t_N)]^T \end{aligned} \quad (10)$$

The covariance function in these equations is then one of the most important properties of the model. In the same paper, [12], is explained why the second-order statistics of natural scenes have a spatial power spectrum that can be approximated by $\frac{1}{f^2}$, created by objects moving with a relative velocity, at a wide range of depths, where f is the spatial frequency. Since this can be approximated by the power spectrum of the Ornstein-Uhlenbeck covariance function in (7), by choosing the appropriate value for λ, the process should approximate a natural sensory process.

The first goal of the work carried was extending the simulations conducted in the paper [11] with the model in (10) to the estimation of time from real sensory data. The complexity of the problem is highly increased, since, unlike in the paper's simulations, the model of the data obtained from real sensors is not known. Therefore, to be able to estimate the passage of time from sensors, these have to first be modelled as Gaussian processes. This corresponds to finding the models hyperparameters, through techniques of model selection. As explained in section II-C, one of the ways this can be done is through a process of Bayesian model selection, by maximizing the marginal likelihood with respect to these hyperparameters, in equation (8). Given that the covariance function is the Ornstein-Uhlenbeck in (7), the derivatives of the previous equation have to be computed with respect to the hyperparameters, θ.

This step represents therefore being able to estimate a time interval based on the hyperparameters of the Gaussian process that more suitably represent the sensory inputs.

B. Implementation

1) Collecting the processes: The first experience done in this work was the reproduction of the results presented in [11], which includes using simulated Gaussian processes with known statistical properties, in this case a Ornstein-Uhlenbeck covariance function as in (7), to apply in the model and check whether the obtained results behave as expected. However, if the processes are obtained from sensory information, some pre-processing may have to be done. This can include, for example, applying a whitening transformation to the input signals to decorrelate them spatially or passing them through a filter to avoid undesirable behaviour.

2) Estimating the hyperparameters: In the case of real sensory information, the following step is studying its statistical properties as to obtain the corresponding Gaussian process, that can later be applied in the stochastic model. This study of the second-order statistics represents the characteristics of the robots sensors. In this case, since these processes are assumed to have the characteristic of natural scenes, their covariance function is assumed to be of the Ornstein-Uhlenbeck type. In this scenario, now it is the time to apply model selection and estimate its parameters, λ and σ. Notice that this is out of the scope of the original paper already, since, there, the exact statistical properties of the processes are known and do not need to be estimated. So what we are doing here instead is learning the statistics of the timeseries from training data. Then, we need to evaluate how well, knowing these statistics, we can estimate time intervals.

For this purpose the maximization of the likelihood described in II-C was used, receiving the vector of time instants and the corresponding observations for those times, and estimating the more likely values for λ and σ. The parameters here calculated are the ones that more correctly represent a process that is likely to have produced the observations, that is, the samples of the timeseries.

3) Applying the model: The average value obtained through the application of the algorithm described in the paper over

all trials is then our estimate of the chosen duration, and is developed according to algorithm 1. The main idea is that, by maximizing the likelihood equation (8), the τ that best explains the observations is found and corresponds to the estimated elapsed time.

Algorithm 1 Bayesian model to estimate the elapsed time

1: **for** each process construction (row) **do**
2: Get the hyperparameters values, λ and σ
3: Choose the duration of the interval to be estimated, and how many of those intervals will be taken from the processes
4: **for** each one of them (each trial) **do**
5: **for** each of the possible values of τ **do**
6: Compute the covariance function
7: Compute the likelihood of each process
8: Compute the likelihood of the processes together
9: **end for**
10: **end for**
11: Calculate the posterior from the likelihood and the prior
12: Compute the value of τ for which the posterior is maximum
13: **end for**
14: Collect the maximum for each of the trials, and compute their mean and standard deviation

C. Results

Firstly, the algorithm was tested to replicate the results of the paper, which means using simulated processes with known covariance. Then, it was tested using the sensory streams obtained from a robot.

1) Simulated data: For the case where known processes were simulated, the chosen hyperparameters were, as in the original paper, $\lambda = 0.01$ and $\sigma = 0.1$, as a way to match the $\frac{1}{f^2}$ power-law statistics of natural scenes. So for 20 seconds, Gaussian processes with these properties are represented in the top of figure 1. Applying the code between lines 4 and 14 of the algorithm 1 to these processes should therefore allow us to get an estimate of the elapsed time between the beginning and the end of the process. For each of the 12 processes created, a likelihood distribution is computed, representing, for that process, which time interval is more likely to have passed given that that process was obtained. In figure 1, four of the twelve likelihood distributions are represented, as well as the normalization of the likelihood function of the combination of twelve processes (black curve). This shows how, even though individually each sensory stream is not informative enough about the elapsed time, the combination of multiple of them provides a strong source of information. The estimated duration in this trial is the peak of the combined likelihood in black, therefore estimated to be around 16 seconds. Variability comes from the stochasticity of the Gaussian processes, and, by repeating this process over multiple trials, the maximum of each of the combined likelihoods gives a sample of a time estimate for that trial. The results for 20 trials are represented in figure III-C1. As shown in the paper, in average, these estimates will accurately represent the real elapsed time.

In this case, the real time interval of the created processes was 20 seconds, and the mean estimated interval was 21.265 seconds, with a standard deviation of 7.138 over the 20 trials. Table I shows the obtained values for different intervals.

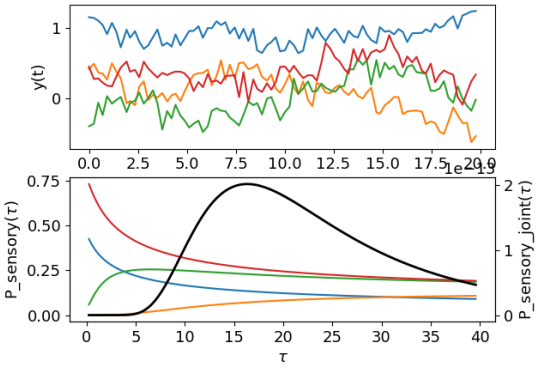

Fig. 1. Individual likelihood distribution of each sensory stream and combined one, in black.

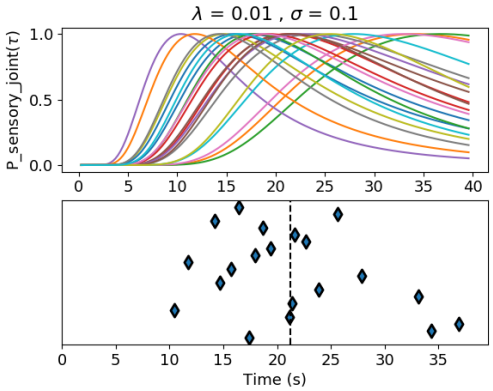

Fig. 2. On the upper plot is represented the combined likelihood of all the processes for the 20 trials, and on the bottom plot the maximum value of each of them. The average of the maximum over each trial gives the duration estimation, represented in a dashed line.

2) Sensory data: Hyperparameters estimation. In this case the statistical properties of the sensory streams obtained from the robot are unknown, so assuming they follow a Ornstein-Uhlenbeck covariance function its parameters have to be estimated.

A visual comparison between the sensory streams and the processes with statistical characteristics estimated to be similar to the ones from the original processes, through the method previously defined is presented in figure 3. Keep in mind that the original process and the estimated one are not supposed to be exactly equal, instead, it is only their statistical properties that should be similar. The blue and red lines correspond to two of the timeseries collected from 12 angles of the laser, during 20 seconds, after passing through a whitening process to remove spatial correlations. The black line shows the estimated function to most properly represent them as a OU process, that was calculated to have $\lambda \approx 1.03$ and $\sigma \approx 0.37$. When used without any time of pre-processing, the collected timeseries are very different from each other, due to being taken from different angles of the laser. This should make it expectable that they can all be accurately represented by a Gaussian process with a single covariance

TABLE I

AVERAGE AND STANDARD DEVIATION OF THE TIME ESTIMATES AS TIME INCREASES. THE FIRST ROW SHOWS THE REAL VALUE OF TIME, THE SECOND THE AVERAGE ESTIMATED AND THE THIRD THE STANDARD DEVIATION OF THE MEASURES, ALL IN [S]. THE LAST ROW SHOWS THE ERROR PERCENTAGE, IN [%].

Real	2	3	5	7	9	10	12	14	15	17	20	22	25	30
Av.	2.15	3.28	4.45	6.45	8.50	12.4	13.9	15.6	16.5	19.3	21.9	25.6	31.3	31.3
S.d.	1.13	1.69	2.17	2.41	3.68	4.1	5.37	5.9	6.6	8.4	9.3	10.6	10.2	12.1
Error	7.5	9.4	11.0	7.8	5.5	4.0	2.9	1.1	3.8	2.7	3.5	0.6	2.2	4.2

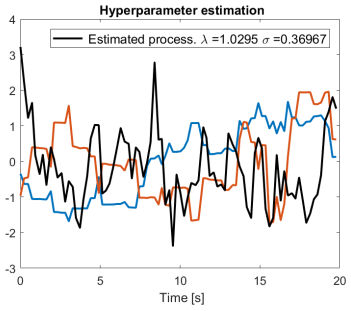

Fig. 3. $\lambda = 1.0295, \sigma = 0.36967$, estimated for the whitened processes with the robot moving forward with a constant speed.

function, namely of the OU type. The consequences are that using this approximation for the estimation of time will likely lead to wrong results, meaning that other covariance functions should be tried instead.

This same process was applied for multiple movements of the robot, with different speeds. *Time estimation*. Once the values of the hyperparameters are known, it is possible to apply the processes in the time estimation model. Four of the 12 sensory streams are represented on the top side of figure 4, and, by applying them to the model with the previously calculated values for the hyperparameters of the covariance for this specific case, the results are shown in the bottom of this figure as well as in figure 5.

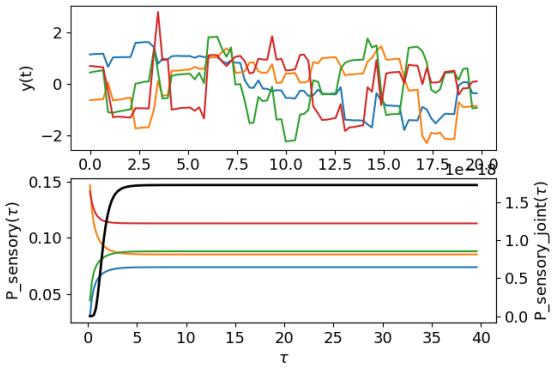

Fig. 4. Individual and combined likelihood distributions of the sensory processes

The real value of the interval was 20 seconds, and the estimated interval using only sensory information was 21.029 with a 15.805 standard deviation. As expected, there is a big inconsistency in the obtained estimates for each trial, but in this case the average estimated value is somewhat accurate and does not seem to contain any bias in the estimation.

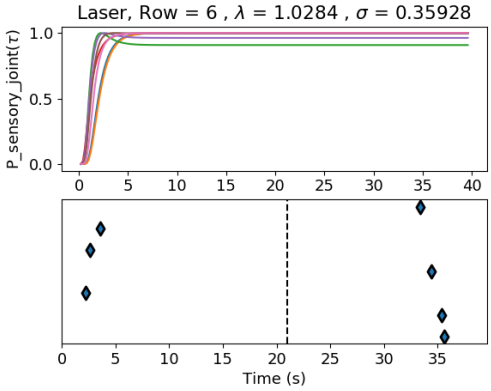

Fig. 5. ML estimate and duration estimation for the sensory processes

IV. FROM TIME TO ACTION

This chapter focuses on the second part of the problem suggested, that assumes that an artificial agent has some source of temporal information, and has to use that information to perform its tasks. This source can be a clock, as is the case in most of the current algorithms, or can be something more biologically inspired, such as the framework explained in chapter III, in which the temporal source is a sensory-based clock. One of the ways in which it is possible to test the role of time representation is in reinforcement learning problems that aim at solving time-dependent tasks.

A relevant question at this point is how time is represented in a reinforcement learning task, in a way that correctly represents what is already known about the timing mechanisms in the brain. This is the main focus of this chapter, alongside with a comparison between these and traditional algorithms that do not consider a specific time representation, in order to test whether these can also have a good performance in timing tasks. From [13] we can say that in RL models of the Basal Ganglia, cortical inputs to the Striatum are what makes this structure encode the estimated value of time, being these inputs the basis of the feature representation. The set of features is what represents a state, such as in (5), and is believed to be where some of the aspects of an animal's experience are encoded. In this scenario, the strengths of the corticostriatal synapses are the ones represented by the set of weights $w_t(1), , w_t(D)$ from the same equation.

The two main theories used nowadays to represent stimulus, given the need of consistency with what happens in the Basal Ganglia, are:

Complete Serial Compound, or CSC, [14] [15], is the current standard representation in TD models. It considers each feature as a timestep of the stimulus, since its onset. This means that, in order to know how many timesteps have passed since the stimulus happened, the only thing needed is counting the number of features that was activated, which corresponds to a perfect clock notion. Even though this representation is useful for the examination of TD learning rules, it presents inconsistencies in representing characteristics of the dopamine system [16].

Microstimuli came up as an alternative time representation in [17]. A number of microstimuli are deploited by a stimulus, and, as time goes by, different sets of microstimuli become more or less active since later ones are wider, shorter and have a later peak. So knowing how much a microstimulus has decayed due to its slowly decaying memory trace can be seen as a basis for the elapsed time, providing a coarse code of the trace height.

A. Theoretical Framework

An example of a time-dependent task in which the time estimate can be used is teaching an artificial agent how to act in an environment in order to reach a certain goal, using a Temporal-Difference learning algorithm presented in section II-A, and described by (1). The goal is to specifically estimate the sequence of actions to perform in a certain task. For this, four variations of Q-learning are implemented for the same task, with the goal of being compared, and, particularly, understanding whether variations that include a realistic time representation, such as Microstimuli, may prove to be more useful and efficient than others that have been the baseline until now and rely on a perfect clock instead.

A task similar to the one explained in [1] with mice was implemented in an experiment with an artificial agent, in order to compare the results with the ones verified in the original one. In this experiment, the agent is, at each timestep, in any of three states: S = 0) Init, 1) Sound, 2) Wait, and in each of them can choose any of four actions: A = Start, Wait, Short, Long. The setup of the experiment consists on three buttons that the agent can choose to press: A "Start" button, that, if pressed in the beginning of the experiment, initializes it and gives rise to the appearance of two equal sounds separated by a certain time interval, i.e, a certain number of timesteps between them. The choice of the interval duration is done based on 50% of probability of being short, and 50% of being long. Inside the chosen interval, there is an uniform probability of choosing any value within that interval. The action "Wait" corresponds to the agent doing an action that does not interfere with the state of the experiment, which means that no button is pressed at that timestep. If the agent correctly presses the "Start" button to start the experiment and does the action "Wait until hearing the first and second sounds, then it has to, according to the perceived duration of the interval between the two, press the "Short" or "Long" button. If the button corresponding to the correct interval duration is pressed, then a reward signal is given to the agent. Otherwise, it gets a negative reward corresponding to a punishment. In the real experiment this corresponds to giving either water/food or an unpleasant sound, respectively. The schematic representation of the experiment can be seen if figure 6.

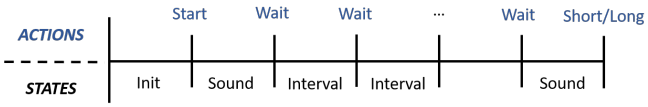

Fig. 6. RL task. The horizontal line represents the passage of time, from left to right, in which the timesteps are constant and from one to the next there is always a state transition. The sequence of states shown is the optimal one in case the agent chooses the corresponding actions in blue. The last action to choose can be either Short or Long according to the number of states "Interval" that exist, and the reward is given after that last action is taken.

In each episode, the algorithm's performance is evaluated in 5 steps, that show how well the agent behaved in that episode:

0) Did not press the "Start" button.
1) Presses the button "Start.
2) Waits after hearing the first sound.
3) Waits as many times as needed, until hearing the second sound.
4) Makes the correct choice according to the length of the interval.

The four variations of Q-learning implemented for this same task with the goal of being compared are described ahead.

*1) **Tabular Markovian Q-learning**:* With a tabular representation, the action-values are stored in variables and accessed when the value of performing a certain action in a certain state is needed, as well as updated when a reward is received.

In this case, after getting the reward and going to the next state, the agent learns the utility of performing this action in this state by updating the value of that pair state-action, through equation (1). Here, each row of the Q-matrix corresponds to a state s and each column to an action a, and the crossing between both corresponds $Q(s, a)$, which is the Q-value for performing action a in state s.

The complete algorithm is represented by pseudo-code in algorithm 2.

Algorithm 2 Tabular Markovian Q-learning

```
Initialize Q-values table = 0
Initialize the Q-learning agent, with α and γ.
Initialize the explorer, ϵ-greedy
Initialize the environment
for each episode do
    Initialize the agent
    for each step do
        a = max_a(Q(s, a))
        Get the new state s' resulting from performing action a in state s,
        and the corresponding reward, r, for that transition.
        Q(s, a) = Q(s, a) + α(r + γ max_a(Q(s', a)) − Q(s, a))
    end for
end for
```

The results are shown in figure 7. It can be seen that the performance of the agent converges to step 3, which means that the agent does all the actions correctly until hearing the second sound, but then does not know which button to press. Again, this is the expected behaviour for a Markovian implementation, and the results can be confirmed in the Q-values table IV, where values in bold indicate the less negative action to do in each state, therefore the one that the agent will prefer to choose. In the state "Init", the action with the highest value

TABLE II
Q-VALUES FOR TABULAR MARKOVIAN Q-LEARNING.

Q(s, a)		Actions			
		Start	Wait	Short	Long
States	Init	**2.1716e-05**	**1.947e-05**	-0.99	-0.99
	Sound	-0.99	**5.210e-56**	-0.85	-0.49
	Interval	-0.99	**5.452e-57**	-0.99	-0.99

is the "Start", in the "Interval" between two sounds, the agent correctly does the action "Wait", but then when it is in the "Sound" state, it does not have enough information to decide if is hearing the first or second sound, therefore always does the action corresponding to the first one, that is "Wait".

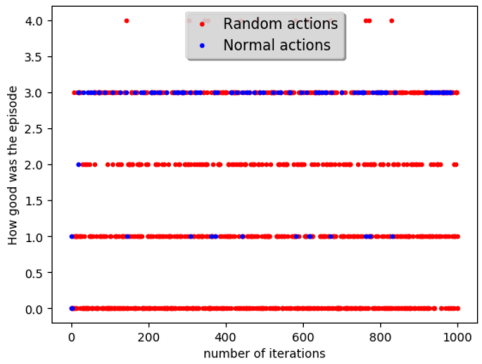

Fig. 7. Tabular Markovian algorithm. The x-axis of the figures represent each of the 1000 episodes, and the y-axis the final step reached during each of the episodes, which is equivalent to the performance of the agent during that episode.

*2) **Tabular non-Markovian Q-learning**:* Unlike previously, in this case a list of the previous states is given to the agent with their history. This transforms the problem in a trivial one, but in order to properly estimate the duration, more information needed to be provided to the agent than just the current state. It needs to know at least the number of states between the first and second tones to be able to make a proper decision.

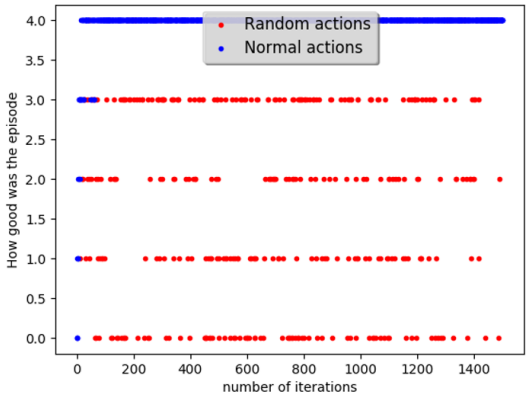

Fig. 8. Performance of the tabular non-Markovian Q-learning algorithm, over 1400 episodes.

TABLE III
COMPUTATIONAL RESOURCES REQUIRED BY THE TABULAR NON-MARKOVIAN Q-LEARNING APPROACH.

Interval	Processing Time	Dict. size	Table size	Converges in
2	1.37	30	120	20
4	2.07	363	1452	60
6	1.96	3,279	13116	63
8	2.40	29,523	118,128	115
10	6.35	265,719	1,062,876	90
12	35.28	2,391,483	9,565,932	160
14	395.06	21,523,359	86,093,436	135

TABLE IV
Q-VALUES FOR TABULAR NON-MARKOVIAN Q-LEARNING

Q(s, a)		Actions			
		Start	Wait	Short	Long
States	Init	**-0.28**	-1.13	-1.14	-1.14
	Sound	-1.15	**-0.59**	-1.16	-1.13
	Interval	-1.12	**-0.07**	-1.11	-1.10

As seen in figure 8, around episode 80 the agent already knows the correct sequence of actions to perform as to maximize the amount of reward.

Even though it solves the timing problem, this algorithm comes with the disadvantage of demanding a big amount of computational resources such as memory and processing time, which will increase exponentially with the increase of the maximum interval between the two sounds. The values for the usage of these resources are represented in table III. The first column is the maximum interval between the two sounds, in timesteps, the second is the processing time it takes for the algorithm to do 1400 iterations, the two next are the number of rows of the dictionary and table, respectively, and the last column is the episode in which the agent learns the sequence of actions to perform.

*3) **Function Approximation with CSC representation**:* In the case of function approximation, each singular state is not directly saved in memory, rather, is represented as a set of features. This is something that seems natural to better represent the brain mechanisms, and can be applied for larger problems as well as speed up learning. However, a proper choice of features is absolutely necessary for the success of the algorithm.

Algorithm 3 Q-learning with Function Approximation

1: Initialize $Q(s,a) = 0$
2: Initialize $w = [w_1, w_2, ..., w_n]$ randomly (e.g. $w_i \in [0,1]$)
3: **for** each repetition **do**
4: **for** each episode **do**
5: Initialize s
6: **for** each step **do**
7: Choose a from s using policy derived from Q (e.g: ϵ-greedy)
8: Take action a, observe r, s
9: $\delta = [r + \gamma \max a' Q(s,a)]Q(s,a)$
10: $e_{tracesn} \leftarrow \gamma\lambda e_{tracesn} + x_n$
11: $w_n(a) \leftarrow w_n(a) + \alpha\delta e_{tracesn}$
12: $s \leftarrow s'$
13: **end for**
14: Until s is terminal
15: **end for**
16: **end for**

In this case, the choice of features x_n in line 9 of algorithm

TABLE V
PROPERTIES OF THE FUNCTION APPROXIMATION APPROACH WITH THE CSC TIME REPRESENTATION. THE FIRST AND LAST COLUMNS ARE IN TIMESTEPS, AND THE SECOND ONE IS THE COMPUTATIONAL TIME IT TAKES TO COMPUTE.

Maximum Interval	Time (s)	Features' size	Weights' size	Converges in
2	19.6	6	24	200
4	24.2	8	32	150
8	41.5	12	48	260
10	54.0	14	56	400
12	66.5	16	64	410
16	70.9	20	80	500
20	104.9	24	96	600
30	176.9	34	136	800

3 is considered to be given by a Complete Serial Compound (CSC) time representation, described in (11).

$$x_{i,j}(t) = \begin{cases} 1, & \text{if element } j \text{ of stimuli } i \text{ is present in } t. \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

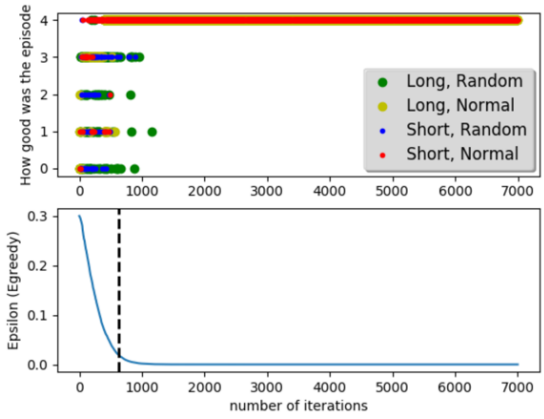

Fig. 9. Convergence graph of Q-learning with CSC representation for a temporal discrimination task.

The problem is limited by the number of features that can be used, which can be biologically seen as memory constraints, making it impossible to save all past events. Here there is no generalization between time instants, since only one feature is active for each stimulus.

*4) **Function Approximation with Microstimuli representation**:* On the other hand, with the Microstimuli representation a set of features is triggered every time there is a stimulus or a reward. The same Q-learning with function approximation algorithm 3 is used, but now the features are encoded according to figure 10, in which a set of temporal basis functions, represented in the middle by Gaussians, are uniformly distributed along the trace height, in the left. The features are then a function of the basis functions represented by

$$x_t(i) = y_t \times f(y_t, \frac{i}{m}, \sigma), \tag{12}$$

where m is the number of microstimuli per stimulus, i is the total number of microstimuli, $x_t(i)$ is the level of each existing

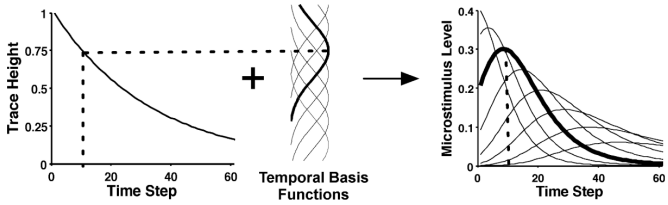

Fig. 10. Microstimuli creation. Extracted from [17]

microstimuli at time t, and y_t the trace height. $f(y, \mu, \sigma)$ are the basis functions, that, if Gaussians, are given by

$$f(y, \mu, \sigma) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{(y-\mu)^2}{2\sigma^2}). \tag{13}$$

As Gaussians, μ is the centre and σ the width of each basis function. y_t decays exponentially according to

$$y_t = \exp(-(1 - decay) \times t). \tag{14}$$

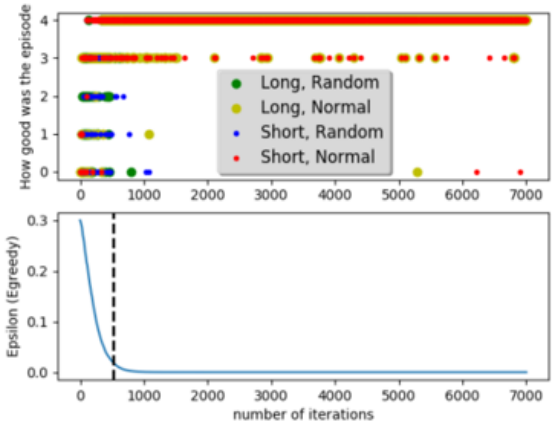

Fig. 11. Convergence graph of Q-learning with Microstimuli for a temporal discrimination task. Maximum interval = 8 timesteps.

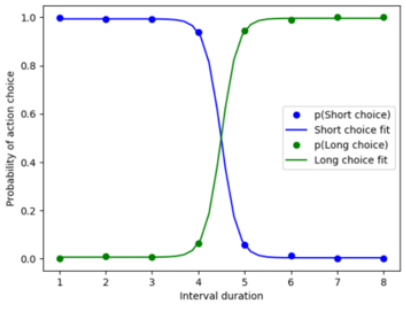

Fig. 12. Number of intervals wrongly classified according to their corresponding duration. The maximum interval duration was 8 timesteps.

The previously described behaviour of the algorithm can be seen in figure 11, where the algorithm does not converge due to the incorrectly classified intervals in the boundary, as shown in figure 12. Here it can be compared with the one of the mice performing the experiment in the paper [1], showing that the agent behaves similarly to the way as mice did in the original experiment.

TABLE VI
MICROSTIMULI CHARACTERISTICS

Maximum interval	Time (s)	Size of features and weights	Converges in (last outside border)
2	34.3	31, 124	1000
4	47.4	31, 124	750
6	54.02	31, 124	627
8	54.1	31, 124	969
10	56.4	31, 124	1300
12	66.8	31, 124	2500 (1100)
14	78.8	31, 124	1295 (789)
16	91.1	31, 124	1124 (984)
18	82.5	31, 124	1745 (969)
20	90.7	31, 124	2274(2274)
22	99.2	31, 124	1621 (1388)
30	126.2	31, 124	6800 (3400)

In table VI are shown the computational characteristics of this algorithm, implemented for 7000 trials, while the other algorithms were for 1000.

The comparison with the previous algorithms is shown in figure 13, and is the collection of values from tables III and VI. The problem of the previous algorithms is that, if we want

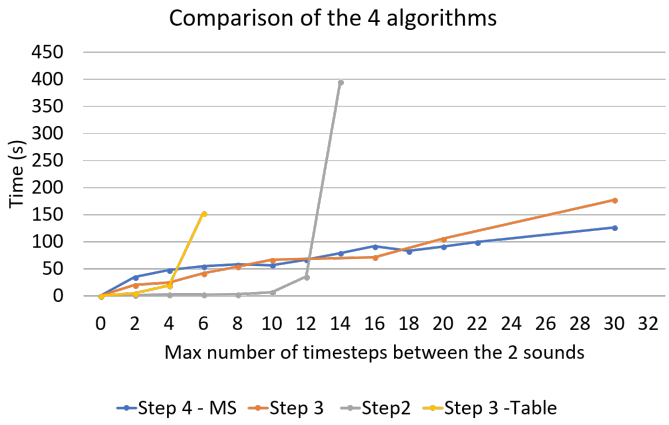

Fig. 13. Comparison of the computational time of the tested algorithms.

to increase the duration of the interval to be timed or, even keeping this fixed, increasing the temporal discretization, then the computational requirements become too demanding. On the other hand, with Microstimuli representation even if the interval increases the number of features remains the same, being more scalable.

V. CONCLUSIONS

When it comes to the estimation of time through sensory information, we conclude that, firstly, it is very important to choose a proper way to take care of external information collected by sensors. It was seen that different ways to collect the sensory streams lead to very different results, as well as when it comes to pre-processing techniques such as whitening transformations, being that the whitening transformation to spatially decorrelate the sensory streams provides better results than using them directly as they come, since the statistical properties resemble better those of Gaussian processes with Ornstein-Uhlenbeck covariance functions. Furthermore, even though these have before been used to represent the natural statistics of the environment, in this work we compute the numerical likelihood of the sensory streams and conclude that other covariance functions should be tried instead.

In the application of biologically inspired algorithms on reinforcement learning problems, The performance of these algorithms implemented as to represent brain functions was compared to that of traditional Q-learning algorithms in a timing task. It can be concluded that the former are not only more realistic and accurate, but also more efficient that the others. The choice of features plays an important role, being that the Microstimuli representation was shown to be the not only more efficient but also most similar time representation with the dopamine behaviour:

- There is uncertainty in the interval boundary between short and long sounds
- Uncertainty increases with the length of the interval.
- The TD error of TD learning behaves similarly to the dopamine firing rate in the brain, decreasing as a reward starts being expected.

REFERENCES

[1] S. Soares, B. V. Atallah, and J. J. Paton, "Midbrain dopamine neurons control judgment of time," *Science*, vol. 354, no. 6317, pp. 1273–1277, 2016.
[2] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 2011.
[3] M. D. Mauk and D. V. Buonomano, "The neural basis of temporal processing," *Annu. Rev. Neurosci.*, vol. 27, pp. 307–340, 2004.
[4] T. S. Gouvêa, T. Monteiro, A. Motiwala, S. Soares, C. Machens, and J. J. Paton, "Striatal dynamics explain duration judgments," *Elife*, vol. 4, 2015.
[5] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*. Springer, 2004, pp. 63–71.
[6] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of brownian motion," vol. 36, pp. 823–841, 01 1930.
[7] M. Maniadakis and P. Trahanias, "When and how-long: a unified approach for time perception," *Frontiers in psychology*, vol. 7, p. 466, 2016.
[8] M. Maniadakis, E. Hourdakis, and P. Trahanias, "Robotic interval timing based on active oscillations," *Procedia-Social and Behavioral Sciences*, vol. 126, pp. 72–81, 2014.
[9] D. V. Buonomano and M. M. Merzenich, "Temporal information transformed into a spatial code by a neural network with realistic properties," *Science*, vol. 267, no. 5200, pp. 1028–1030, 1995.
[10] S. W. Brown, "Time, change, and motion: The effects of stimulus movement on temporal perception," *Perception & Psychophysics*, vol. 57, pp. 105–116, 1995.
[11] M. B. Ahrens and M. Sahani, "Observers exploit stochastic models of sensory change to help judge the passage of time," *Current Biology*, vol. 21, no. 3, pp. 200–206, 2011.
[12] D. W. Dong and J. J. Atick, "Statistics of natural time-varying images," *Network: Computation in Neural Systems*, vol. 6, no. 3, pp. 345–358, 1995.
[13] T. V. Maia, "Reinforcement learning, conditioning, and the brain: Successes and challenges," *Cognitive, Affective, & Behavioral Neuroscience*, vol. 9, no. 4, pp. 343–364, 2009.
[14] P. R. Montague, P. Dayan, and T. J. Sejnowski, "A framework for mesencephalic dopamine systems based on predictive hebbian learning," *Journal of neuroscience*, vol. 16, no. 5, pp. 1936–1947, 1996.
[15] R. S. Sutton and A. G. Barto, "Time-derivative models of pavlovian reinforcement." 1990.
[16] N. D. Daw, J. P. O'doherty, P. Dayan, B. Seymour, and R. J. Dolan, "Cortical substrates for exploratory decisions in humans," *Nature*, vol. 441, no. 7095, p. 876, 2006.
[17] E. A. Ludvig, R. S. Sutton, and E. J. Kehoe, "Stimulus representation and the timing of reward-prediction errors in models of the dopamine system," *Neural computation*, vol. 20, no. 12, pp. 3034–3054, 2008.