Using Reinforcement Learning to Attenuate for Stochasticity in Robot Navigation Controllers

James Gillespie, Iñaki Rañó, Nazmul Siddique, Jose Luis Santos, Mehdi Khamassi

Using Reinforcement Learning to Attenuate for Stochasticity in Robot Navigation Controllers

James Gillespie
Intelligent Systems Research Centre
Ulster University
Derry, Northern Ireland
gillespie-j10@ulster.ac.uk

Iñaki Rañó
Embodied Systems for Robotics and Learning
University of Southern Denmark
Odnese, Denmark
igra@mmmi.sdu.dk

Nazmul Siddique
Intelligent Systems Research Centre
Ulster University
Derry, Northern Ireland
nh.siddique@ulster.ac.uk

José Santos
Pervasive Computing Research Group
Ulster University
Jordanstown, Northern Ireland
ja.santos@ulster.ac.uk

Mehdi Khamassi
Institute of Intelligent Systems and Robotics
Sorbonne Université, CNRS
Paris, France
mehdi.khamassi@upmc.fr

***Abstract*—Braitenberg vehicles are bio-inspired controllers for sensor-based local navigation of wheeled robots that have been used in multiple real world robotic implementations. The common approach to implement such non-linear control mechanisms is through neural networks connecting sensing to motor action, yet tuning the weights to obtain appropriate closed-loop navigation behaviours can be very challenging. Standard approaches used hand tuned spiking or recurrent neural networks, or learnt the weights of feedforward networks using evolutionary approaches. Recently, Reinforcement Learning has been used to learn neural controllers for simulated Braitenberg vehicle 3a – a bio-inspired model of target seeking for wheeled robots – under the assumption of noiseless sensors. Real sensors, however, are subject to different levels of noise, and multiple works have shown that Braitenberg vehicles work even on outdoor robots, demonstrating that these control mechanisms work in harsh and dynamic environments. This paper shows that a robust neural controller for Braitenberg vehicle 3a can be learnt using policy gradient reinforcement learning in scenarios where sensor noise plays a non negligible role. The learnt controller is robust and tries to attenuate the effects of noise in the closed-loop navigation behaviour of the simulated stochastic vehicle. We compare the neural controller learnt using Reinforcement Learning with a simple hand tuned controller and show how the neural control mechanism outperforms a naïve controller. Results are illustrated through computer simulations of the closed-loop stochastic system.**

***Index Terms*—Braitenberg Vehicle, Reinforcement Learning, Stochastic Environment**

I. Introduction

Braitenberg Vehicles are a set of qualitative models of sensor-based steering behaviour in animals navigating towards, or away from, a stimulus [1]. They have been used for decades in different navigation tasks, and empirical works on real robots have shown that robots using these controllers can work in harsh and dynamic environments [2]. Because of their qualitative biological nature, these controllers have been typically implemented using different types of neural networks. Qualitative models of the closed-loop have been derived for the case of ideal (noiseless) sensors [3], where the control mechanism is modelled as a generic function. The assumption of deterministic state and perception had the advantage of greatly simplifying the analytic treatment of the closed-loop motion models. Moreover, it has been recently shown that Policy Gradient Reinforcement Learning can be used to find neural controllers that approximate functions leading to closed-loop stable behaviour of Braitenberg Vehicle 3a [4]. However, these controllers might not perform well in presence of sensor noise. Recently, a new model of Braitenberg Vehicle 3a has been derived for noisy sensors as a non-linear Stochastic Differential Equation [5], which allows us to simulate and obtain new theoretical results on the closed-loop behaviour of this robot navigation controller. Like in the deterministic model, the navigation control function is modelled as a generic function of the sensor measurements, yet, in the stochastic case finding a controller to generate a stable closed-loop navigation behaviour is even more challenging. This paper uses Policy Search Reinforcement Learning to learn appropriate control functions, leading to stable closed-loop behaviour, modelled as feedforward neural networks. Results show that the learnt neural controllers adapt to the stochasticity of the system trying to minimise the effects of noise. We rely on the stochastic model of Braitenberg vehicle 3a to analyse the resulting control functions approximated by the networks. Furthermore, we compare the performance of these controllers with a naïve linear controller, which has been shown to generate stable navigation in presence of sensor noise [5].

Because of their flexibility, Braitenberg vehicles have been used for multiple robotic tasks like; finding sources of chemicals through mobile robots [2], navigating against a water current [6], obstacle avoidance [7], sound source localisation [8], and underwater navigation with electric sensors [9]. All these examples used empirical tuning of the sensory-motor connection in the robots. However, as already stated some

research works made use of learnt neural controllers to implement navigation behaviours. A recurrent neural network is used in [10] using the qualitative principles of Braitenberg vehicles to control a legged robot. The navigation network is able to negotiate obstacles and interfaces the pre-processed sensory input with a Central Pattern Generator module that controls locomotion. Navigation towards a sound source (phonotaxis) is another existing application of Braitenberg vehicles implemented as neural controllers. In a series of works [11] [12] [13] phonotaxis was implemented using spiking neural networks. The neural controllers replicate the motion of female crickets navigating towards the chirping males in wheeled robots. The robots were tested in both indoor and outdoor environments. Feedforward neural networks have been also used to learn, through evolutionary techniques, Braitenberg like controllers for sensor-based obstacle avoidance [14].

As we can see from the literature, neural controllers have been widely used to implement Braitenberg vehicles with desired closed loop dynamics of the robots. However, in the examples using spiking neural networks the neural controllers were hand tuned, and only in some cases were the controllers actually learnt through optimisation techniques, which are computationally costly. Reinforcement Learning (RL), meanwhile, can greatly reduce the time of learning compared to evolutionary techniques, enabling the learning agent the discovery of an optimal behaviour autonomously by trial-and-error according to a reward function [15]. RL has been widely used in robotics because it provides solutions for hard to engineer and sophisticated problems of behaviour implementation that roboticists typically face. For example, Reinforcement Learning was used in several works to learn navigation strategies for mobile robots, both in simulated [16] and on real robot [17]–[19]. All of these works used Value Function Approaches to solve navigation problems, which are not well suited to deal with continuous state and action spaces. For instance, a change in the value function for some state-action pair can alter the whole policy [15], making these approaches prone to fail in highly stochastic problems. Recent research in RL focuses instead on Policy Search techniques which are better suited for continuous domains and stochastic environments.

Ng et al. [20] use a policy search method to learn a robust helicopter controller for autonomous inverted flight; a difficult problem as helicopters have high-dimensionality, are non-linear, complex, and stochastic. The Pegasus policy search algorithm is used to learn a controller which can hover in place and execute a number of manoeuvres. In this work, a model of the helicopter is used for the learning. Although this speeds up the process of learning the controller, it can be dangerous because of modelling uncertainties, meaning the controller may not work when deployed on the real robot. State of the art work by [21] shows model-free policy search used to control a manipulator in a stochastic scenario: In a series of experiments, first a door position was varied in the x and y space in an environment and the robotic arm was able to consistently open the handle using monocular RGB images. In a second experiment, the manipulator learns to pick up a bottle regardless of position in a 30cm $\times$ 40cm grid.

As we can see, multiple experimental works have shown that Braitenberg vehicles can perform target seeking behaviours with noisy sensors, yet none can ensure optimal or robust behaviour. Meanwhile, we have seen RL used to learn navigation controllers and robust controllers for stochastic systems. This work shows that RL can be used to learn a robust controller for a stochastic Braitenberg vehicle 3a. The learnt controller will be optimal with respect to the reward function, thus by carefully engineering the reward function for target seeking behaviour, we can take the first steps towards optimal controller design for a stochastic Braitenberg Vehicle 3a. The rest of the paper is organised as follows: Section II reviews the assumptions, the mathematical model of Braitenberg vehicle 3a as a stochastic differential equation, and how to use RL to learn the sensori-motor connection function. Section III presents results of the Stochastic Braitenberg Vehicle 3a with different amounts of noise on the sensors using a non-linear connection function learnt using the Policy Gradient Reinforcement Learning algorithm. The learnt controllers are tested under noise and are compared against a hard-coded connection function. The paper concludes with a summary of the presented findings and some future work in Section IV.

II. A Stochastic Model of Braitenberg Vehicle 3a

Before applying any Reinforcement Learning techniques, we will first briefly introduce the stochastic closed-loop model of Braitenberg vehicle 3a as drift-diffusion equations, already presented in [5], which we will later use for our simulations.

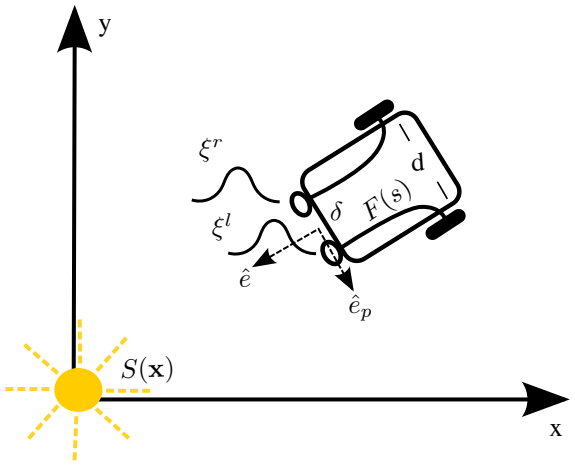


Fig. 1. A Stochastic Braitenberg Vehicle 3a.

As seen in figure 1, vehicle 3a is a unicycle type robot with a wheelbase d and distance between the sensors δ. The two symmetrically arranged sensors have an inhibitory connection to the corresponding wheel motor on the same side of the vehicle as the sensor, and have a white noise signal following a Gaussian distribution in our sensor readings (ξ_r, ξ_l), turning the standard model of vehicle 3a into a stochastic one. Since the sensori-motor connection is decreasing, illustrated using

the '–' sign in the figure, the vehicle's wheels turn at a faster speed when low stimulus readings are recorded by the sensors, and as the sensed stimulus gets stronger closer to the origin, the wheel speeds slow down until they eventually stop moving when the vehicle has reached the source (which has the highest stimulus reading). This inhibitory connection also controls the steering of the vehicle. For example, if the stimulus source is to the left side of the vehicle, the intensity recorded by the left sensor will be higher than the intensity recorded by the right sensor. This results in the left wheel turning slower than the right one, hence the vehicle steers towards the stimulus source. [22] presents the mathematical model of this bio-inspired non-linear controller for unicycle type robots, although this work assumes noiseless sensors. [5] however develops a stochastic closed-loop model of Braitenberg vehicle 3a using drift-diffusion equations which allows us to model sensor noise.

Let us assume in a domain $\mathcal{D}$ there is a non-negative smooth scalar stimulus $S(\mathbf{x})$. The connection function between the wheels and sensors is a function of the stimulus, $F(s)$, and since the connection for vehicle 3a should be decreasing, that means $F'(s) < 0$ for all perceived values s of the stimulus. To model the sensor noise we define a random variable ξ with a variance smoothly dependent on the measured stimulus value as:

$$\xi = \sigma(s)dW_t \tag{1}$$

where $\sigma(s)$ is the variance of the noise, describing how far from the true value of the measured stimulus the random perturbations can go, and W_t is a Wiener process. The sensor noise has a zero mean Gaussian distribution at every point in time. Although the noise is different for each sensor, we will assume it has the same statistical properties. Under these assumptions, for a vehicle with the left and right sensors located at $\mathbf{x}_l$ and $\mathbf{x}_r$ respectively, the speed of the left and right wheels are:

$$v_l = F(S(\mathbf{x}_l) + \sigma(S(\mathbf{x}_l))dW_t^l)$$

$$v_r = F(S(\mathbf{x}_r) + \sigma(S(\mathbf{x}_r))dW_t^r) \tag{2}$$

where dW_t^l and dW_t^r are independent noise processes on the left and right sensors. Remembering that the distance between the sensors is δ, we can approximate v_l and v_r as a Taylor series around the middle point between the sensors $\mathbf{x}$. We can use these approximations of the wheel velocities (v_l and v_r) to obtain the forward speed (v) and turning rate ($\omega = \dot{\theta}$) of the vehicle:

$$\omega = -\frac{\delta}{d}\nabla F(\mathbf{x})^T\hat{\mathbf{e}}_p - \frac{1}{d}D_1(\mathbf{x},\theta)dW_t^- - \frac{1}{d}D_2(\mathbf{x},\theta)dW_t^+$$

$$v = F(\mathbf{x}) + \frac{1}{2}D_1(\mathbf{x},\theta)dW_t^+ + \frac{1}{2}D_2(\mathbf{x},\theta)dW_t^- \tag{3}$$

where d is the wheelbase of the vehicle, $dW_t^+ = dW_t^l + dW_t^r$ and $dW_t^- = dW_t^l - dW_t^r$ are two stochastic processes combining the two sensors noise, and the diffusion terms $D1(\mathbf{x},\theta)$ and $D2(\mathbf{x},\theta)$ are:

$$D_1(\mathbf{x},\theta) = \frac{\delta^2}{4}F''(\mathbf{x})\sigma(\mathbf{x})[\nabla S(\mathbf{x})\cdot\hat{\mathbf{e}}_p]^2 + F'(\mathbf{x})\sigma(\mathbf{x})$$

$$D_2(\mathbf{x},\theta) = \frac{\delta}{2}\nabla S(\mathbf{x})\cdot\hat{\mathbf{e}}_p[F'(\mathbf{x})\sigma'(\mathbf{x}) + F''(\mathbf{x})\sigma(\mathbf{x})] \tag{4}$$

If we substitute equations (3) into the unicycle kinematic model we get the following closed-loop system of stochastic differential equations:

$$\begin{aligned}
dx_t &= F(\mathbf{x}_t)\cos\theta_t dt + \frac{1}{2}D^-(\mathbf{x}_t,\theta_t)\cos\theta_t dW_t^r \\
&\quad + \frac{1}{2}D^+(\mathbf{x}_t,\theta_t)\cos\theta_t dW_t^l \\
dy_t &= F(\mathbf{x}_t)\sin\theta_t dt + \frac{1}{2}D^-(\mathbf{x}_t,\theta_t)\sin\theta_t dW_t^r \\
&\quad + \frac{1}{2}D^+(\mathbf{x}_t,\theta_t)\sin\theta_t dW_t^l \\
d\theta_t &= -\frac{\delta}{d}\nabla F(\mathbf{x}_t).\hat{\mathbf{e}}_p(\theta_t)dt - \frac{1}{d}D^+(\mathbf{x}_t,\theta_t)dW_t^l \\
&\quad + \frac{1}{d}D^-(\mathbf{x}_t,\theta_t)dW_t^r
\end{aligned} \tag{5}$$

where $D^-(\mathbf{x}_t,\theta_t) = D_1(\mathbf{x}_t,\theta_t) - D_2(\mathbf{x}_t,\theta_t)$ and $D^+(\mathbf{x}_t,\theta_t) = D_1(\mathbf{x}_t,\theta_t) + D_2(\mathbf{x}_t,\theta_t)$. Note we have eliminated the compound functions in the equations to simplify the notation, i.e. $F(\mathbf{x}_t) = F(S(\mathbf{x}_t))$, $F'(\mathbf{x}_t) = F'(S(\mathbf{x}_t))$, and so on. The diffusion terms in equations (4) rely on the noise variance $\sigma(s)$, the connection function $F(s)$, and each of their derivatives. They model non-additive noise since the terms multiplying the increments of the Wiener Processes dW_t^l and dW_t^r have a functional dependency on the state of the system, $\mathbf{x}_t$ and θ_t.

Now that we have obtained the system of Stochastic Differential Equations needed to simulate our Stochastic Braitenberg Vehicle 3a, we can use Reinforcement Learning to learn the connection function, $F(s)$, between the wheels and the noisy sensors.

A. The Reinforcement Learning Problem

The problem of learning the target seeking behaviour for a stochastic Braitenberg vehicle 3a given a stimulus function can be restated as finding a non-increasing function of the stimulus to compute the velocities of the vehicle's wheels. Given the stimulus $S(\mathbf{x})$ defined in an environment $\mathcal{D} \subseteq \Re^2$, and with some initial vehicle pose $(x_0,\theta_0) \in \mathcal{D}\times S^1$ the trajectory followed depends on the connection function $F(s)$. Since the trajectory unfolds in time $(\mathbf{x}(t),\theta(t))$ but also depends on the initial pose and $F(s)$ we will write $(\mathbf{x}(t,\mathbf{x}_0,\theta_0,F),\theta(t,\mathbf{x}_0,\theta_0,F))$. We define a scalar reward function $r(\mathbf{x},\theta)$ for each pose of the vehicle to measure how good being at that state is. However, since the trajectory depends on the initial pose and the connecting function, we have to make the reward along a given trajectory a function

of the initial pose and the connecting function too, that is $r(\mathbf{x}, \theta) = r(t, \mathbf{x}_0, \theta_0, F)$.

Since the problem we are trying to solve is a stochastic dynamical system (a partially observable Markov decision process), we need to choose a Reinforcement Learning methodology which best leans itself towards stochastic data. While this renders Value Function Approaches unsuitable, Policy Search Methods are highly effective when the policy is stochastic – the most common of which are Policy Gradient Methods. Policy Gradient Methods are a type of Reinforcement Learning technique which works by optimising parameterised policies w.r.t. the expected return using gradient ascent. We decided to use the Simultaneous Perturbation Stochastic Approximation Algorithm [23] to estimate the gradient of the policy, and used Adam [24], a state of the art algorithm for stochastic optimisation to tune the weights of the neural network that would be used to approximate the sensori-motor connection function $F(s)$. This method would be used to learn a robust controller for our stochastic Braitenberg Vehicle 3a. We opted to use a Radial Basis Function Feed-Forward Neural Network. The final form of the RBF was as follows:

$$F(s) = \left(\sum_{i=1}^{N} \phi_i(s) w_i \right) \tag{6}$$

where N is the number of basis functions, the radial basis functions $\phi_i(s)$ were Gaussian kernel functions centred at fixed equidistant positions s_i within the range of the stimulus (from $s = 0$ to $s = S(\mathbf{0})$), and w_i is the weight of the i-th neuron. By denoting the weight vector of the RBF as $\Phi = (w_i)$ we can state the reinforcement learning problem as maximising the following total return:

$$R[\Phi] = \int_{(\mathbf{x}_0, \theta_0) \in \mathcal{D} \times S^1} p(\mathbf{x}_0, \theta_0) d\mathbf{x}_0 d\theta_0 \int_0^{\infty} r(t, \mathbf{x}_0, \theta_0, \Phi) dt \tag{7}$$

where $p(\mathbf{x}_0, \theta_0)$ is the probability of the initial conditions being $\mathbf{x}_0, \theta_0$ and we need to integrate for all the initial conditions. In other words, we need to integrate for all the initial poses of the vehicle – the starting coordinates in the workspace $\mathbf{x}_0$ and all orientations in the unit circle $\theta_0 \in S^1$ – while also needing to integrate over the whole trajectory. It is impossible, however, to evaluate these integrals due to the total return function depending on the solution of the non-linear dynamical system modelling the Braitenberg vehicle. Furthermore, in our case it is not feasible to integrate over the whole trajectory, so we changed the upper integration limit in equation (7) to a finite time t_f for our simulations. To solve the problem of evaluating the total return we used the sampling trick – where we sampled the space using random initial poses of the vehicle – to estimate, through simulations, the value of the integral. By using the Simultaneous Perturbation Stochastic Approximation Algorithm (SPSA), we can estimate the expected return:

$$\Delta \hat{R}_i \approx R(\Phi_i + \Delta\Phi_i) - R(\Phi_i - \Delta\Phi_i) \tag{8}$$

where R is a function which generates an episode, returning the total return from equation (7), Φ_i is the RBF weight vector, $\Delta\Phi_i$ is a d-dimensional vector of perturbations defined as $\Delta\Phi_i = C_k \cdot X \sim \pm 1. Bernoulli(\frac{1}{2})$ and $C_k = C_0/k^{\gamma}$, where k is the iteration and γ is a non-negative coefficient. From this, we can estimate the gradient of the total return using regression yielding:

$$\nabla_{\Phi} R[\Phi] = (\Delta\Phi^T \Delta\Phi)^{-1} \Delta\Phi^T \Delta R \tag{9}$$

With the initial estimate of the gradient $\nabla_{\Phi} R[\Phi]$ obtained, we can use the Adam algorithm for Stochastic Gradient Descent on the return to tune the weight vector Φ of the RBF:

$$\Phi_{k+1} = \Phi_k + \alpha \cdot \hat{m}_k / (\sqrt{\hat{v_k}} + \epsilon) \tag{10}$$

where $\hat{m}_k$ and $\hat{v}_k$ are bias-corrected first and second moment estimates of the gradient $\nabla_{\Phi} R[\Phi_k]$, tuned using the default settings from [24], α is the step size, and ϵ is a parameter whose value is 10^{-8}.

As we mentioned, the environment of the vehicle is the whole plane $\mathcal{D} = \Re^2$, however we can simplify things in the domain of the integral in equation (7), i.e. the domain in which the initial conditions are selected for the sampling process by defining the environment as a square region around the origin $\mathcal{D} = [X_{-3}, Y_{-3}] \times [X_3, Y_3]$ where $[X_{-3}, Y_{-3}]$ is the minimum starting coordinates and $[X_3, Y_3]$ is the maximum starting coordinates. To further simply the learning process we decided to select the initial angular directions of the vehicle to be pointing towards the source within a $\pm 90^{\circ}$ range, i.e. $\theta_0 \in [\theta_t - \pi/2, \theta_t + \pi/2]$, where $\theta_t = \arctan\left[\frac{y_0}{x_0}\right] - \pi$.

III. Simulations

This section presents the results of learning a robust sensori-motor connection for a stochastic Braitenberg Vehicle 3a using the Reinforcement Learning methodology presented in Section II. In the experiments, the sensor noise variance is increased while the system tries to effectively attenuate for the stochastic environment. The results are compared against a naïve linear sensori-motor connection, previously used for Stochastic Braitenberg vehicle 3a control in [5]. In the RL experiments, the RBF function approximator was initialised using 11 neurons, the perturbation size C_0 was initialised to 0.7, the maximum iterations was set to 500, and the number of randomly initialised episodes per iteration was 330. The parameters for the Adam optimiser followed the default settings in the documentation and were $\alpha = 0.001$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. To simulate the Braitenberg Vehicle we integrate numerically the Stochastic Differential Equations presented in equations (3) using the Euler-Maruyama method with a fixed time step of h$= 0.05$ and a simulation time of $t_f = 8$.

A. Defining the reward function

Instead of directly imposing conditions on the learning, RL uses an agent to preform trial and error interactions with its environment, learning via a reward signal which indicates whether the generated behaviour of the agent was

desirable. However, defining a reward function which will generate the exact behaviour we want the agent to learn is often seen as a difficult to engineer problem. While there are multiple ways we could allocate reward to achieve the target seeking behaviour of a stochastic Braitenberg Vehicle 3a, we have to be careful that the reward function we design cannot be "gamed" by the agent, learning some function which maximises the return but does not produce the behaviour we want – a phenomenon known as reward hacking. To ensure we avoid reward hacking by the agent, we kept the reward function simple since it is noted that the more complicated the reward function, the higher the likelihood and severity of any hacking [25]. Since our robot is biologically inspired, we wanted to base our reward function on how animals in their natural environment learn a navigation task i.e. based on the perceived strength of a stimulus using their receptor organs. The function we designed allocated reward directly corresponding to the strength of the perceived stimulus at the agents current location, and since the sensors are noisy the reward function would take the noisy observations as it's parameters. We summed these noisy readings recorded by the vehicle's two sensors and used this as the reward:

$$r = S_l \xi_l + S_r \xi_r \tag{11}$$

where $S_{l|r}$ are the true sensor readings and, as mentioned previously, $\xi_{l|r}$ are white noise signals following a Gaussian distribution with a variance smoothly dependent on the measured stimulus value. To fulfil the criteria that the motion of the vehicle is never backwards, i.e. $F : \Re^+ \cup \{0\} \rightarrow \Re^+ \cup \{0\}$, the reward function will only assign reward when the movement of the vehicle is in the forward direction.

B. Learning target seeking with a stochastic vehicle

Although our Braitenberg vehicle could be used to find the origin of any stimulus source, the most popular application is the phototaxis Braitenberg Vehicle i.e. using light sensors to find the source of a light stimulus. Given that a light source follows the inverse-square law, we can define the stimulus as a function of the position of the vehicle $S(\mathbf{x}) = \frac{g_0}{1+\eta \mathbf{x}^\mathbf{T} A \mathbf{x}}$ where $\eta = 0.25$, $g_0 = 4$ and A is a 2×2 identity matrix defining the isotropic nature of the stimulus source. Since the goal of this work is target seeking, we want our vehicle to reach as close to the origin of the stimulus source as possible. Based on our definition of the light source, we know that in theory the limit of proximity to the source and thus the ideal stopping point of the vehicle is when the light sensors on the vehicle measure the maximum stimulus value, i.e. $S(\mathbf{0}) = g_0$, which is recorded when $\mathbf{x}$ = (0, 0). However, due to the stimulus being sampled by the sensors the vehicle can never reach this point. To calculate the real limit of proximity to the source we need to account for δ, the distance between the sensors, as the vehicles cannot get closer to the origin than this – thus the maximum "S" they can reach is $S\left(\frac{1}{\sqrt{2}}\begin{bmatrix}\delta\\ \delta\end{bmatrix}\right)$. Combining δ with the stimulus function we get

$$S(\mathbf{x}) = \frac{g_0}{1 + \eta \mathbf{x}^\mathbf{T} + \left(\frac{1}{\sqrt{2}}\begin{bmatrix}\delta\\ \delta\end{bmatrix}\right)^T A\mathbf{x} + \left(\frac{1}{\sqrt{2}}\begin{bmatrix}\delta\\ \delta\end{bmatrix}\right)} \tag{12}$$

where, in our case $\delta = 0.1$ hence the largest stimulus value we can ever expect the vehicle to perceive is 3.99, instead of $g_0 = 4$.

For our first experiment, we want to check if we can effectively use Reinforcement Learning to train a neural controller when there is noise present in the learning process. We will use the Newton-Raphson method for obtaining the roots of functions to calculate the stimulus value at the equilibrium point of the learned functions. This will allow us to analyse how well our controllers preform as we increase the amount of noise on the sensors. Ten learning experiments were ran each time we increased the sensor noise variance (σ), from 0 to 0.1 in 0.005 increments. Although the noise is applied in the simulations used for the learning process, it should be emphasised that the results presented in this section are being evaluated in a noise-free environment so we can focus on the effectiveness of the algorithm at attenuating for noise.

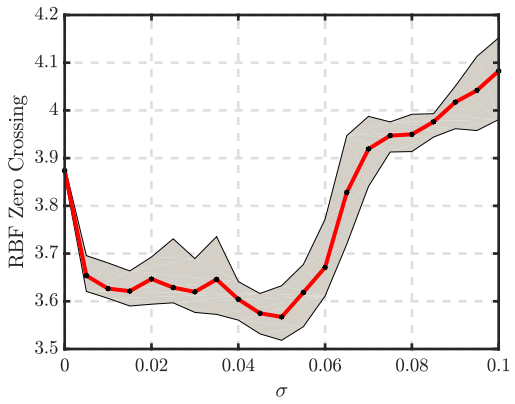


Fig. 2. Stimulus values at the RBF zero crossings as the variance of the sensor noise σ is increased from 0 - 0.1 in 0.005 increments.

Figure 2 shows a plot of stimulus values at the vehicles equilibrium point (i.e. the RBF zero crossing) averaged over 10 trajectories, as a function of the sensor noise variance σ. The grey shaded area shows the maximum and minimum equilibrium points over the 10 trials, while the red line is the average. While $S(\delta) = 3.99$ is the ideal RBF zero crossing, we can see from figure 2 that in the experiment where $\sigma = 0$ i.e. when there is no noise on the sensors, the equilibrium point is at $S(\mathbf{0}) = 3.87$. We can therefore use this value as our new baseline to compare the effect sensor noise has on the learning process. As soon as we add noise to the sensors, the robot stops at an equilibrium point further from the source. Even at the lowest level of noise, $\sigma = 0.005$, the equilibrium point is $S(\mathbf{0}) = 3.65$, a significant change from the no noise scenario. However, as the variance increases from $\sigma = 0.005$ to $\sigma = 0.06$, the equilibrium point remains largely unaffected displaying a mean of 3.6232 across the range with a standard

deviation of 0.0304, showing that Reinforcement Learning is robust with noise present in the learning process. After this point, the vehicles equilibrium point starts increasing. The last experiment which shows an average equilibrium point below $S(\mathbf{0}) = 4$ is when the sensor noise variance $\sigma = 0.085$, with variance $\sigma = 0.09$ experiments showing an average equilibrium point of $S(\mathbf{0}) = 4.0172$.

Although we may think the Reinforcement Learning algorithm is no longer able to attenuate for the stochasticity in the system effectively once the sensor noise variance increases beyond $\sigma = 0.085$, later we will see that using the neural controller learnt even when $\sigma = 0.1$ is still very effective when it is used for stochastic vehicle control.

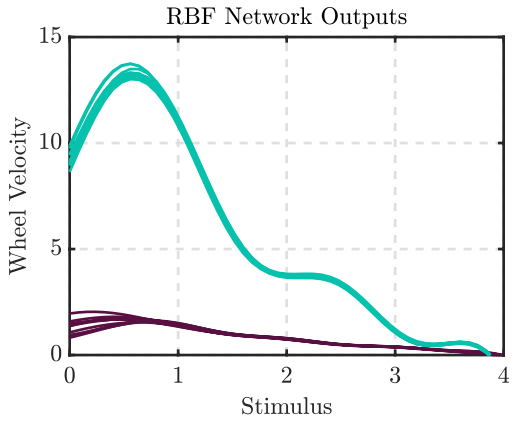


Fig. 3. RBF network outputs defining the connection function

Figure 3 shows a comparison between the trained RBF Neural Networks under no sensor noise (turquoise plot) and when the sensor noise variance σ is 0.08 (plum plot). Ten learnt networks are shown on each plot. These networks map the relationship between the wheel velocity and the strength of the stimulus. While the plots help visualise the equilibrium points presented in figure 2, what is interesting is that as the sensor noise variance increases, the standard deviation of the RBF networks remains the same i.e. the stability of the algorithm remains unaffected as noise is added to the system. All 10 trials in each case take positive values meaning the vehicles are moving forward, and have a negative slope which indicates the positive taxis behaviour of Braitenberg Vehicle 3a. The figures show that the slope of the RBF is much steeper when there is no noise in the system (turquoise plot) compared to when the RBF is trained under noise (plum plot), a behaviour expected as the neural network attenuates for noise. Notice that as the variance σ increases, as well as the steepness of the slope at the equilibrium point decreasing, the velocity of the vehicle at any of the sampled stimulus values is also smaller. This is because the weights of the network are tuned using the estimated gradient of the total return. In our simulations, if the angular direction of the vehicle was not pointing towards the source within a $\pm 90^o$ range during any stage of the simulation, the simulation stopped. Therefore, once the noise in the system is increased, the heading angle is much more likely to deviate from this range during the simulation. This is confirmed by our results: in the deterministic case with no noise, there was a 60% success of full simulation, compared to 47% success of full simulation at $\sigma = 0.005$ (the lowest amount of sensor noise variance we applied to our system) and 15% success of full simulation when $\sigma = 0.08$. This leaves us with a smaller ΔR compared to the deterministic case, thus the weight updates are not as large and therefore not as large a velocity is learnt.

C. Testing the controller on a simulated stochastic vehicle

The previous section has shown that Reinforcement Learning can effectively learn a robust sensori-motor connection function for Braitenberg Vehicle 3a control, even under varying levels of sensor noise. However, we also need to test how effective the learnt neural controllers are at attenuating for stochasticity when the vehicle experiences sensor noise during its target seeking task, i.e. we need to ensure that the robot with noisy sensors can successfully preform target acquisition using the learnt connection functions. To give some comparison, we ran simulations of the stochastic Braitenberg vehicle under the same levels of noise using the learnt sensori-motor connection and a hard coded linear connection function, defined as $F(s) = 4 - s$ where s is the strength of the stimulus, so we can see if using RL improves Stochastic Braitenberg Vehicle performance.

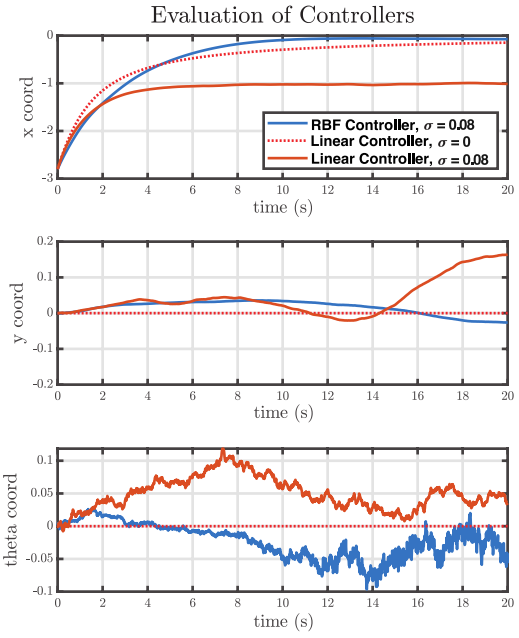


Fig. 4. The mean of 4000 trajectories evaluated using the naïve controller and RBF controller when $\sigma = 0.08$, compared against a noise-free trajectory.

The plots in Figure 4 show an example of the time evolution of the mean of 4000 simulated trajectories using the naïve hard coded connection function (orange plot) and the RBF connection function learnt using our RL algorithm (blue plot).

Each simulation is ran for 20 seconds. The figure shows the results when when the sensor noise is following a normal distribution with a variance $\sigma = 0.08$. The red dashed line is our control experiment and represents the evolution obtained from the deterministic equations of the moments, i.e. how the vehicle performs in optimal conditions when the sensor noise variance is zero. In each simulation, we start the vehicle at $\mathbf{x} = (-2.8, 0), \theta = 0$, while the target is $\mathbf{x} = (0, 0), \theta = 0$. We can see from the orange plot which uses the hard coded connection function, that by introducing sensor noise there is a large effect on the trajectory of the vehicle. Not only does the x coordinate not reach close to zero (stops at $x = -1.0129$), the y coordinate and θ coordinate drift. Comparatively, the experiment which uses the RBF connection function performs much better when tested under the same level of sensor noise. Although the x coordinate does not reach zero ($x = 0.0144$), it still performs much better than the hard coded controller counterpart. Additionally, the y and θ coordinates, although slightly noisy over the entire evolution of the trajectory, do not drift. We can further analyse the effectiveness of using the RBF controller over a hard coded linear controller by evaluating the strength of the stimulus perceived by the robot at the end of the simulation, $T = 20$. Remembering that the maximum stimulus value that can be perceived is $\mathbf{S}(\mathbf{0}) = 4$, when we use the hard coded connection function, the robot stops moving at $\mathbf{S}(\mathbf{0}) = 3.1666$, while the connection function learned using RL stops at $\mathbf{S}(\mathbf{0}) = 3.9992$.

Table I documents the complete results of the above experiment for all tested variances in the sensor noise σ, from 0 to 0.1 in 0.01 increments. As above, each controller is evaluated 4000 times and the mean is taken. Then, using the coordinates of the vehicle when $T = 20$ we calculate the mean value of the perceived stimulus at the end of the target seeking task for each realisation of sensor noise. The results indicate that the hard coded naïve controller performs well up to $\sigma = 0.05$, however after this point the controller starts to significantly deteriorate. The RBF controller on the other hand maintains a consistent performance as σ is increased, even at $\sigma = 0.1$. Interestingly, compared to the first experiment (see figure 2), instead of overshooting the target and continuously roaming the environment when $\sigma = 0.1$, the vehicle now stops close to the source due to having noisy sensors.

As we have seen above, the neural controller preforms well and locates the stimulus source under different levels of sensor noise when the controller is evaluated over 4000 trajectories and the mean taken. However, this does not necessarily tell a fair story about the success rate of the controller in a real life navigation task since a real robot only takes one trajectory to attempt to locate the source of the target. Thus, in our final experiment, we seek to find out how the target seeking robot performs over individual trajectories when there is noise in the system. We tested both controllers under each level of noise as we increased the variance from 0 to 0.1 in 0.01 increments. We ran 50 simulations for each experiment and computed how many trajectories converged at the source of the stimulus. Each vehicle stimulation started from the same

Sigma	Naïve Controller	RBF Controller
0	3.979	3.865
0.01	3.980	3.631
0.02	3.985	3.756
0.03	3.991	3.716
0.04	3.979	3.796
0.05	3.991	3.788
0.06	3.711	3.870
0.07	3.647	3.992
0.08	2.967	3.995
0.09	2.821	3.885
0.1	2.415	3.708

TABLE I
STIMULUS VALUES AT THE EQUILIBRIUM POINTS OF THE NAÏVE AND RBF CONTROLLERS FOR DIFFERENT REALISATIONS OF SENSOR NOISE. MAXIMUM STIMULUS IN ENVIRONMENT IS 4.

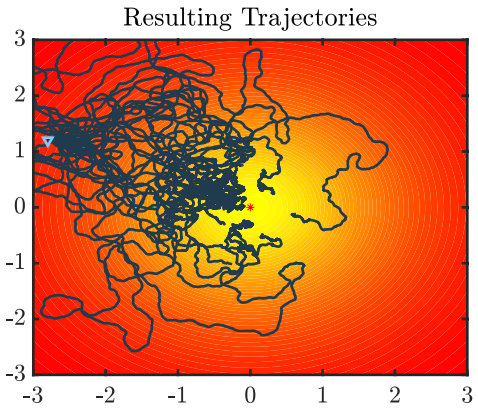
coordinates, $\mathbf{x} = (-2.8, 1.2), \theta = 0$, while the goal was at the origin, $\mathbf{x} = (0, 0)$. We classified a successful single trajectory as one which reached ± 0.9 distance units from the source (i.e. the recorded stimulus value at the vehicles equilibrium point had to be at least 3.3) Table II documents the complete results of successful individual trajectories as the sensor noise variance is increased. Both controllers remain unaffected by the sensor noise in the initial experiments, each displaying a 100% success at target seeking until the variance reaches 0.03. At this point, the RBF controller manages 98% (one trajectory did not reach the source) while the naïve controller still achieves a 100% success. However, the experiments from $\sigma = 0.04$ until the maximum noise level, $\sigma = 0.1$, show that the naïve controller's performance is consistently diminishing until, at $\sigma = 0.1$, the controller can only converge to the stimulus 46% of the time. We compare these results to our neural controller which consistently is able to attenuate for the noise, and at $\sigma = 0.1$ shows a 88% successful convergence rate.

Sigma	Naïve Controller	RBF Controller
0	100%	100%
0.01	100%	100%
0.02	100%	100%
0.03	100%	98%
0.04	92%	96%
0.05	90%	88%
0.06	80%	84%
0.07	68%	84%
0.08	62%	90%
0.09	54%	86%
0.1	46%	88%

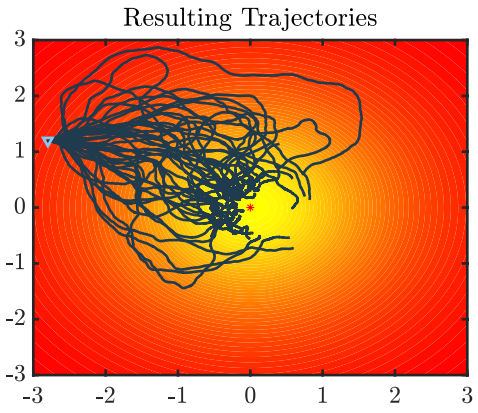
TABLE II
PERCENTAGE OF CONVERGED INDIVIDUAL TRAJECTORIES OVER 50 TRIALS FOR DIFFERENT REALISATIONS OF SENSOR NOISE.

Figure 5 illustrates the results: while Figure 5(a) shows the

successful results of 50 trajectories using the naïve controller, Figure 5(b) shows the successful trajectories using the RBF controller, both when the variance of the sensor noise was $\sigma = 0.1$. Each simulation was initialised with the same starting coordinates of the vehicle, $\mathbf{x} = (-2.8, 1.2), \theta = 0$, marked on the figures using a blue triangle. The goal was $\mathbf{x} = (0, 0)$, marked with a red asterisk. The heatmap represents the stimulus strength at each point in the environment. From Figure 5(a), which uses the hand tuned controller, we can see that out of the 50 test trajectories, only 23 (46%) successfully converge at the stimulus source. We can also see that the trajectories are jagged as the naïve controller cannot attenuate for the noise in the closed-loop dynamical system. Meanwhile, Figure 5(b), which uses the neural controller, shows 44 (88%) successfully converged trajectories at the stimulus source. As well as almost doubling the successful number of trials, we can see from figure 5(b) that the trajectories are much more smooth as the robot successfully attenuates for the noise in its sensors and easily navigates to the source of the stimulus.



(a) Naïve controller preforming target seeking in real world scenario. 23/50 trajectories reach the source with jagged trajectories as the controller fails to attenuate for the sensor noise.



(b) RBF controller preforming target seeking in real world scenario. 44/50 trajectories reach the source with very smooth trajectories as the controller successfully attenuates for the sensor noise.

Fig. 5. Simulated trajectories using each control mechanism, when sensor noise variance is $\sigma = 0.1$. Only successful trajectories are shown.

IV. Conclusions

This paper has presented the first implementation of Reinforcement Learning for a target seeking Stochastic Braitenberg Vehicle 3a. While previous empirical works use hand tuned parameters or Evolutionary strategies to optimise a fitness function on a real robot with potentially noisy sensors, this work uses Reinforcement Learning with a simulated robot to demonstrate that a Policy Gradient algorithm can be used with a RBF neural network to learn a robust connection function, even under white Gaussian noise. Our results show that the connection function can be learnt under varying amounts of noise without affecting the stability of the algorithm. Moreover, the connection function learnt using RL performs significantly better when tested under noise compared to the hard coded linear connection function, allowing the robot to reach much closer to the origin of the source before stopping. Finally, when simulated in a real life scenario, i.e. when we cannot take the average of the trajectories, the neural controller significantly outperforms the naïve controller. This is demonstrated in the results when measuring the successfully converged trajectories, where some levels of noise show a converged trajectory improvement of 42% using the learnt controller compared to the naïve hard coded controller.

Future work will focus on implementing the Reinforcement Learning algorithm onto a real robot for a target seeking problem. Eventually, this will enable real robots to be used more effectively in disaster zones where there is likely to be noise affecting the sensors, for example in nuclear disaster areas or earthquakes where there are large amounts of dust.

Acknowledgements

This work was partially supported by the Royal Society International Exchange Scheme under grant IE151293.

References

[1] V. Braitenberg, *Vehicles. Experiments in synthetic psycology*. The MIT Press, 1984.

[2] A. J. Lilienthal and T. Duckett, "Experimental analysis of smelling braitenberg vehicles," in *Proceedings of the IEEE International Conference on Advanced Robotics (ICAR 2003)*. IEEE, 2003.

[3] I. Rañó, "Biologically inspired navigation primitives," *Robotics and Autonomous Systems*, vol. 62, no. 10, pp. 1361–1370, 2014.

[4] J. Gillespie, I. Rañó, N. Siddique, J. Santos, and M. Khamassi, "Reinforcement learning for bio-inspired target seeking," in *Towards Autonomous Robotic Systems*, Y. Gao, S. Fallah, Y. Jin, and C. Lekakou, Eds. Springer International Publishing, 2017, pp. 637–650.

[5] I. Rañó, K. Wong-Lin, and M. Khamassi, "A drift diffusion model of biological source seeking for mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2017.

[6] T. Salumäe, I. Rañó, O. Akanyeti, and M. Kruusmaa, "Against the flow: A braitenberg controller for a fish robot," in *Proc. of the Intl. Conf. on Robot. and Autom.*, 2012, pp. 4210–4215.

[7] E. Bicho and G. Schöner, "The dynamic approach to autonomous robotics demonstrated on a low-level vehicle platform," *Robotics and Autonomous Systems*, vol. 21, pp. 23–35, 1997.

[8] D. Shaikh, J. Hallam, J. Christensen-Dalsgaard, and L. Zhang, "A Braitenberg lizard: Continuous phonotaxis with a lizard ear model," in *Proc. of the 3rd Intl. Work-Conf. on The Interplay Between Natural and Artificial Computation*, 2009, pp. 439–448.

[9] V. Lebastard, F. Boyer, C. Chevallereau, and N. Servagent, "Underwater electro-navigation in the dark," in *Proc. of the Intl. Conf. on Robot. and Autom.*, 2012, pp. 1155–1160.

[10] S. Steingrube, M. Timme, F. Wörgötter, and P. Manoonpong, "Self-organized adaptation of a simple neural circuit enables complex robot behaviour," *Nature physics*, vol. 6, no. 3, pp. 224–230, 2010.

[11] B. Webb, *A Spiking Neuron Controller for Robot Phonotaxis*. The MIT/AAAI Press, 2001, pp. 3–20.

[12] A. D. Horchler, R. E. Reeve, B. Webb, and R. D. Quinn, "Robot phonotaxis in the wild: a biologically inspired approach to outdoor sound localization," *Advanced Robotics*, vol. 18, pp. 801–816, 2004.

[13] R. E. Reeve, B. Webb, A. D. Horchler, G. Indiveri, and R. D. Quinn, "New technologies for testing a model of cricket phonotaxis on an outdoor robot," *Robotics and Autonomous Systems*, vol. 51, pp. 41–54, 2005.

[14] F. Mondada and D. Floreano, "Evolution of neural control structures: Some experiments on mobile robots," *Robotics and Autonomous Systems*, vol. 16, no. 2-4, pp. 183–195, 1995.

[15] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[16] C. Weber, S. Wermter, and A. Zochios, "Robot docking with neural vision and reinforcement," in *Applications and Innovations in Intelligent Systems XI*, M. Bramer, R. Ellis, and A. Macintosh, Eds. London: Springer London, 2004, pp. 213–226.

[17] T. Martinez-Marin and T. Duckett, "Fast reinforcement learning for vision-guided mobile robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 4170–4175.

[18] C. Gaskett, L. Fletcher, and E. Zelinsky, "Reinforcement learning for visual servoing of a mobile robot," in *In Proc. of the Australian Conference on Robotics and Automation*, 2000.

[19] B. Zuo, J. Chen, L. Wang, and Y. Wang, "A reinforcement learning based robotic navigation system," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2014, pp. 3452–3457.

[20] A. Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang, *Autonomous Inverted Helicopter Flight via Reinforcement Learning*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 363–372.

[21] Y. Chebotar, M. Kalakrishnan, A. Yahya, A. Li, S. Schaal, and S. Levine, "Path integral guided policy search," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Apr. 2017.

[22] I. Rañó, "A steering taxis model and the qualitative analysis of its trajectories," *Adaptive Behavior*, vol. 17, no. 3, pp. 197–211, 2009.

[23] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3-4, pp. 229–256, May 1992. [Online]. Available: https://doi.org/10.1007/BF00992696

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[25] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Man, "Concrete problems in ai safety," *arXiv preprint arXiv:1606.06565*, 2016. [Online]. Available: https://arxiv.org/pdf/1606.06565.pdf