The robustness-fidelity trade-off in Grow When Required neural networks performing continuous novelty detection

Lenka Pitonakova, Seth Bullock*

Department of Computer Science, University of Bristol, Merchant Venturers' Building, Woodland Road, Bristol, BS8 1UB, United Kingdom

Abstract

Novelty detection allows robots to recognise unexpected data in their sensory field and can thus be utilised in applications such as reconnaissance, surveillance, self-monitoring, etc. We assess the suitability of Grow When Required Neural Networks (GWRNNs) for detecting novel features in a robot's visual input in the context of randomised physics-based simulation environments. We compare, for the first time, several GWRNN architectures, including new Plastic architectures in which the number of activated input connections for individual neurons is adjusted dynamically as the robot senses a varying number of salient environmental features. The networks are studied in both one-shot and continuous novelty reporting tasks and we demonstrate that there is a trade-off, not unique to this type of novelty detector, between robustness and fidelity. Robustness is achieved through generalisation over the input space which minimises the impact of network parameters on performance, whereas high fidelity results from learning detailed models of the input space and is especially important when a robot encounters multiple novelties consecutively or must detect that previously encountered objects have disappeared from the environment. We propose a number of improvements that could mitigate the robustness-fidelity trade-off and demonstrate one of them, where localisation information is added to the

*Corresponding author
Email address: seth.bullock@bristol.ac.uk (Seth Bullock)

input data stream being monitored.

Keywords: novelty detection, self-organised neural networks, unsupervised learning

1. Introduction

Novelty detection is a desired capability of autonomous robots that operate in heterogeneous environments given that their performance may be compromised if they encounter some situations that they were not designed for. This type of capability is usually delivered by a novelty detector algorithm that operates in parallel with other algorithms that deal with robot perception, image recognition and actuator control (Marsland et al., 2002; Vieira Neto & Nehmzow, 2007a; Gonzalez et al., 2018). A novelty detector has two functions. First, a model of what is *normal* is learned, representing what is usually encountered by the robot. Second, *novelties* (sometimes referred to as "anomalies") that do not fit into the learned model of normality are detected as and when they occur (Chandola et al., 2009). A novelty detector can be utilised to recognise unexpected objects in a robot's sensory field (Marsland et al., 2005; Markou & Singh, 2006; Miskon & Russell, 2009), for instance, to identify potential dangers (Sofman et al., 2011; Ross et al., 2015), or to divert a robot's attention to new environmental features that need to be examined (Lepora et al., 2010; Gatsoulis & McGinnity, 2015; Merrick et al., 2016). This can be useful in applications such as surveillance and reconnaissance. It is also possible to use a novelty detector to identify abnormalities in proprioceptive sensor signals in order to detect possible hardware (Wang et al., 2013; Gonzalez et al., 2018) or software faults. Other applications may include detection of extreme conditions in a robot's environment and monitoring of other agents to recognise when their behaviour becomes abnormal as a result of failures, cyber attacks, etc.

Novelty detection is often explored in the context of a novelty detection task. In a one-shot version of such a task, learning and novelty detection are performed consecutively, with an initial period of learning followed by the introduction

of one novelty that is either successfully detected or not (Sohn et al., 2001; Marsland et al., 2005; Markou & Singh, 2006; Miskon & Russell, 2009; Lepora et al., 2010). In a continuous version of the novelty detection task, after the same initial period of learning, novelty detection and learning are performed concurrently, with novel objects repeatedly introduced into the environment and the robot expected to both detect their presence and incorporate them into an updated learned model (Crook & Hayes, 2001; Vieira Neto & Nehmzow, 2007b; Wang et al., 2013; Gatsoulis & McGinnity, 2015).

In this paper, we consider the Grow When Required Neural Network (GWRNN) (Marsland et al., 2002), which is capable of learning an input space representation in a self-organised, unsupervised fashion in order to detect when a novel input is presented. All experiments are performed in an agent-based simulation environment with 3D physics, allowing us to precisely control the robot's sensory-motor parameters and noise, and to generate randomised environments for repeated experiments, improving the robustness of our results. Following Vieira Neto & Nehmzow (2007a) and Gatsoulis & McGinnity (2015), the robot first pre-processes its visual input to generate a vector of salient features, which are then fed into the GWRNN novelty detector.

We compare, for the first time, the performance of several different GWRNN architectures (Section 3), including Standard architectures previously reported in the literature (Marsland et al., 2002, 2005; Vieira Neto & Nehmzow, 2007a; Gatsoulis et al., 2010) and new Plastic architectures introduced here, where the number of activated input connections for individual neurons is adjusted as the robot senses a varying number of salient visual features. By studying these networks in both one-shot (Section 4) and continuous (Section 5) novelty reporting tasks, we show that there is a trade-off between the network's robustness, which is achieved by generalisation over the input space, and high fidelity, which results from learning larger, more detailed models. While robustness is important because it minimises the effect of network and environmental parameters on the network's performance, high fidelity is crucial when it comes to distinguishing inconspicuous features of individual objects, a capability that is important for

successfully reporting novelties that occur consecutively and for detecting when objects that were previously learned are no longer present. We show that providing localisation information as part of the network's input data stream improves fidelity of all tested architectures (Subsection 5.3) and propose a number of other improvements that could mitigate the robustness-fidelity trade-off in GWRNNs (Section 6).

2. Background

A number of approaches have been applied for distinguishing novel or anomalous data from normal data. Novelty detection is a difficult task and, therefore, a particular approach usually solves a specific version of the problem (Chandola et al., 2009). For example, Gaussian mixture models and kernel density estimators can approximate the shape of a data distribution, making it possible to identify data points that lie outside the normal distribution (Chandola et al., 2009; Drews Jr et al., 2010; Ross et al., 2015). However, their performance is limited when the training set is small or when data is not normally distributed (Gatsoulis et al., 2010; Pimentel et al., 2014). Information-theoretic metrics, such as changes in dataset entropy, have been used to identify data outliers (He et al., 2005), but they only work correctly when enough novel data exists and are also computationally expensive (Pimentel et al., 2014). Within the neural networks domain, reinforcement learning can be used to train a network that is then able to classify inputs as belonging to a "known" or an "unknown" class (Markou & Singh, 2006), but labelled data has to be prepared for training.

When no assumptions about the data set distribution can be made, or when a labelled data set is not available, self-organising networks that learn in an unsupervised fashion are a promising approach (Decker, 2005; Pimentel et al., 2014; Fink et al., 2015; Gonzalez et al., 2018), despite their black-box nature (Gatsoulis et al., 2010). They have previously been used for novelty detection on robots (Marsland et al., 2002; Vieira Neto & Nehmzow, 2007a; Miskon & Russell, 2009), for detection of abnormalities in data sets (Marsland et al., 2002;

Albertini & de Mello, 2007), for clustering of concepts in text documents (Hung & Wermter, 2003) and for shopper preference analysis (Decker, 2005).

The Kohonen Self-Organising Map (SOM) (Kohonen, 1982) is inspired by the human brain architecture and includes an input layer and a clustering layer (CL). CL neurons are connected to input neurons via weighted connections, and also connect to a number of "neighbour" CL neurons. When an input is presented, a CL neuron with input connection weights that are the most similar to the input values is marked as the *winning neuron*. CL neurons that are connected to the winning neuron are referred to as the *winning cluster*. The input connections of the winning neuron, and, to a lesser extent, of its neighbours in the winning cluster, are slightly adjusted towards the current input vector. Neighbourhood size and learning rate decrease over time. A SOM eventually learns a topology-preserving representation of the input space, meaning that CL neurons that are connected to each other represent similar inputs.

In the Habituating Self-Organising Map (HSOM) (Marsland et al., 2000), CL neurons are connected to a novelty-outputting neuron via habituating synapses. Habituation, i.e., reduction of neural response strength upon repeated perception of stimulus, enables the HSOM to report new inputs as novel (Marsland et al., 2000), and to adapt existing neuron connections when a new input is presented in consecutive time steps (Gatsoulis et al., 2010). The Growing Neural Gas (GNG) (Fritzke, 1995; Fink et al., 2015) is also based on the SOM. Instead of changing the neighbourhood size like the SOM does, the GNG grows its clustering layer and establishes new connections between CL neurons for every n^{th} input.

Capabilities of the HSOM and the GNG are combined in the Grow When Required Neural Network (GWRNN) (Marsland et al., 2002), that improves the GNG growth algorithm by adding new CL neurons at any time, provided that the similarity of the winning neuron to the input vector and the winning neuron's habituation are both lower than some pre-defined thresholds. The advantages of GWRNNs over other self-organising networks include a reduced tendency to report false positives (Albertini & de Mello, 2007), improved ability to deal with changes in input space distribution and the fact that its structure and growth

rate do not need to be pre-set (Marsland et al., 2002). It has been shown that a GWRNN learns and performs similarly to a Hopfield network (Crook & Hayes, 2001) and incremental principal component analysis (Vieira Neto & Nehmzow, 2007b) in visual novelty detection tasks. For discussion of the relationship between GWRNNs and the wider class of adaptive neural network architectures, including seminal examples such as Grossberg's Adaptive Resonance Theory (ART) networks (Carpenter & Grossberg, 1988), see Marsland et al. (2002).

Even though novelty detection is a popular topic, its application on robots operating in dynamic environments has received limited attention. In many cases, a robot traverses a simple environment and reports when a single novelty occurs. In such studies, the learning and novelty detection stages of an experiment are clearly separated (Sohn et al., 2001; Marsland et al., 2005; Markou & Singh, 2006; Miskon & Russell, 2009; Lepora et al., 2010). In contrast, we are interested in a more ecologically embedded version of the task, where a robot undergoes an initial learning stage, but is then asked to perform continuous novelty detection and learning in an environment where novelties are introduced consecutively (as in Crook & Hayes, 2001; Vieira Neto & Nehmzow, 2007b; Wang et al., 2013; Gatsoulis & McGinnity, 2015).

3. Methods

We used a GWRNN-equipped MarXBot (Bonani et al., 2010) in simulated environments similar to those used in previous studies (Marsland et al., 2002; Vieira Neto & Nehmzow, 2007a,b; Miskon & Russell, 2009). In the initial *learning phase*, the robot traversed a completely unknown environment and progressively learned its features in an unsupervised fashion. We allowed 20 learning passes through the environment, although the network usually stopped adapting before the final passes. In the consequent *operational phase*, that started in pass 21, novelties were introduced into the environment and the robot needed to recognise them and incorporate them into its learned model.

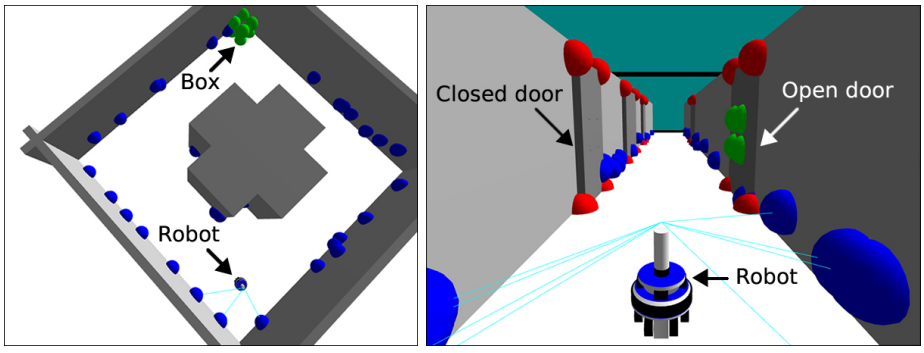
All experiments were performed in the ARGoS simulator with realistic 3D

physics and sensory-motor dynamics (Pinciroli et al., 2012). By using simulation, we could precisely control parameters such as variability of the environment and noise in robot sensors. Moreover, we could randomise positions of objects in the environment, ensuring that our results were not biased by a particular feature of the environment that the network could exploit. We conducted 100 independent runs for each experiment, which is a significant increase on the number of runs normally performed with real robots (Marsland et al., 2005; Gatsoulis et al., 2010; Lepora et al., 2010; Gonzalez et al., 2018).

3.1. Simulation environment

There were two virtual environments, *Room* (as in Vieira Neto & Nehmzow, 2007a) and *Corridor* (as in Marsland et al., 2002), both of which represented an approximately 10 m large area to traverse (Figure 1). We assumed that an input dimensionality reduction algorithm existed on the robot, that could process raw camera data and identify salient visual features, such as discontinuities in colour, intensity, object shape, etc. (as in Vieira Neto & Nehmzow, 2007a; Drews Jr et al., 2010; Gatsoulis & McGinnity, 2015). We modelled this capability by letting the robot sense coloured lights that represented the visual features.

Room (Figure 1a) consisted of four walls that were each 4 m long, surrounding a central, cross-shaped structure. The robot travelled around the room, visiting



(a) Room (b) Corridor

Figure 1: The two simulated experimental environments, Room and Corridor.

each corner. The walls were represented by blue lights and boxes with green lights attached to their corners could be added or removed from the environment. The positions of the blue lights were random in each experimental run.

Corridor (Figure 1b) consisted of two 10 m long walls with doors on both sides. The robot travelled on a straight line along the corridor and was transported back to the beginning of the corridor every time it reached the end (as in Marsland et al., 2002, 2005). As in the Room environment, blue lights were placed at random positions on the walls. Door corners were represented by red lights and the door locations were also randomised. An open door had four green lights in its centre. A closed door had no lights in the centre.

The robot used an omnidirectional light sensor with a range of 0.9 m to sense the coloured lights. Each feature reading consisted of three real numbers (Figure 2), representing a colour value, C, and the relative distance and angle of a light, each normalised between 0 and 1. Three distinct environmental colours were employed, each associated with a unique sensory C value: $C_{\text{red}} = \frac{1}{6}$, $C_{\text{green}} = \frac{1}{3}$ and $C_{\text{blue}} = \frac{1}{2}$.

The robot's sensory input vectors thus had $F \times 3$ elements, where F was the number of currently sensed features. The features in the robot's sensory field

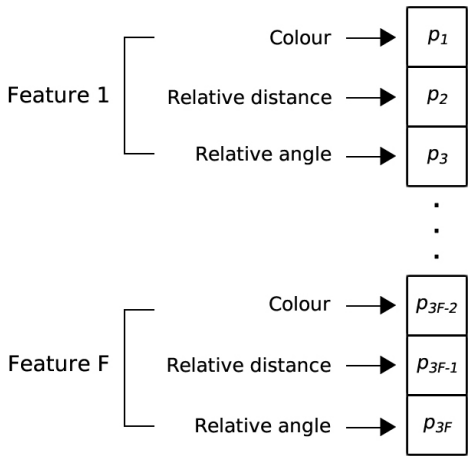


Figure 2: The sensory input vector of elements p_1 through p_{3F}, representing the three sensed attributes associated with each of F sensed environmental features.

were represented in the robot's sensory input vector in an order determined by their distance from the robot, starting with the closest. For a robot in a real-world scenario, these features could be obtained using feature extraction algorithms commonly utilised during, e.g., Visual SLAM (Davison et al., 2007). Unlike in Vieira Neto and Nehmzow's 2007a paper, a robot could sense a different number of features at different times, as would likely be the case in real-world heterogeneous environments.

For example, a robot travelling down a corridor first sensed a number of blue wall features, with angles and distances towards them changing as the robot traversed the environment. In addition, the order in which individual features were presented in the sensory input vector changed with relative distance of the robot to the features, so that closer features always appeared earlier in the list. As the robot approached a closed door, red corner features first appeared in the tail of the sensory input vector and then were pushed towards the front of the vector as the robot got closer to the door. Similarly, when the robot travelled away from the door, red corner features gradually occupied elements further from the front of the sensory input vector, until the door completely disappeared from the robot's visual field.

3.2. The Standard GWRNN

The Grow When Required neural network consists of an input layer, a clustering layer (CL) and an output neuron (Figure 3). The number of input neurons is denoted N_I, the vector of the input neuron activations is $\vec{i}$ and the number of clustering layer neurons is N_C. Each clustering layer neuron c is connected to all of the input neurons via a vector of weighted adaptive input connections $\vec{w_c}$ of size N_{W_c}. In addition, CL neurons may be connected to each other by non-weighted connections forming neighbourhoods, where connected CL neurons represent similar features. The novelty output is equal to the current habituation of the winning CL neuron, $h_c \in (0, 1)$.

In its Standard implementation, (see Marsland et al., 2002, 2005; Vieira Neto & Nehmzow, 2007a), the network has a fixed number of input neurons and

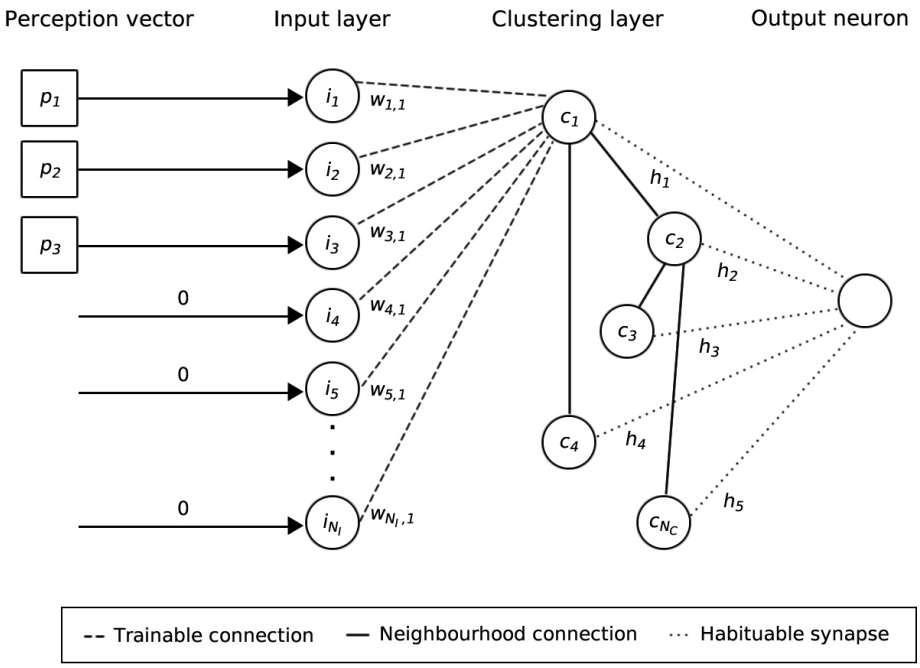


Figure 3: The Standard GWRNN. In each time step, a set number N_I of input neurons gets activated. The input layer is fully connected to the clustering layer (in the image, only connections to a single CL neuron, c_1, are shown in the interest of clarity). Connections between CL neurons define neighbourhoods, where connected neurons represent similar inputs. Each CL neuron has habituation, h_i, and habituation of the winning neuron represents the output of the network. Note that the image shows a situation in which the sensory input vector has length 3 and is therefore shorter than the number of input neurons ($3 < N_I$). Remaining input neurons are activated with the value of 0. In the opposite case, the most distant visual features in the sensory input vector would be discarded.

$N_{W_1} = N_{W_2} = ...N_{W_{N_C}} = N_I$. When the network is initialised, the clustering layer contains only two neurons ($N_C = 2$) with input connection weights set to random values between 0 and 1 and with the neuron habituations set to 1. The network learns in every time step as follows:

1. Receive a sensory input vector of feature attributes of size $N_P = 3F$. Set the activations of the input neurons to corresponding values:

   (a) If $N_P < N_I$, for each input neuron, i, if $i \leq N_P$ set the activation of i to the i^{th} value in the input vector, else if $i > N_P$ set the activation of i to zero. For example, when $N_I = 6$, but the robot only senses one feature ($N_P = 3$), set the activation of the last three input neurons to 0 (as in Figure 3).

   (b) If $N_P \geq N_I$, for each input neuron, i, set the activation of i to the i^{th} value in the input vector. Discard the remaining input vector values that relate to the visual features furthest from the robot.

2. Find the winning neuron, s, according to Eq. 1 and 2, where d_c denotes the distance of neuron c from the input vector $\vec{i}$ (which comprises only the attributes of currently sensed features ordered by their current distance to the robot), i_j is the j^{th} input value and $w_{j,c}$ is the weight of the connection between input neuron j and clustering layer neuron c (Marsland et al., 2002):

$$d_c = \sqrt{\sum_{j}^{N_I = N_{W_c}} (i_j - w_{j,c})^2} \tag{1}$$

$$s = \underset{c \in N_C}{\operatorname{argmin}}\, d_c \tag{2}$$

3. Find the second-best neuron, t, in a similar fashion.

4. If s and t are not connected, connect them in order to form a neighbourhood relationship between them.

5. Calculate the activation a_s of neuron s (Marsland et al., 2002):.

$$a_s = \exp(-d_s) \tag{3}$$

6. Add a new clustering layer neuron, r, if $a_s < \Theta_A$ and $h_s < \Theta_H$, where Θ_A and Θ_H are the activation and habituation thresholds, respectively. Then (Marsland et al., 2002):

   - Set the new neuron weights as

$$w_{j,r} = (w_{j,s} + i_j)/2 \tag{4}$$

   - Set the new neuron habituation to $h_r = 1$.
   - Connect the new neuron r with both s and t. Remove the neighbourhood connection between s and t.

7. Adapt the input connection weights of s and of all the neighbouring neurons n that it is connected to:

$$\Delta w_{j,s} = \eta \times h_s \times (i_j - w_{j,s}) \tag{5}$$

$$\Delta w_{j,n} = \min(1, \frac{\psi a_n}{a_s})\eta \times h_n \times (i_j - w_{j,n}) \tag{6}$$

   where η is the network learning rate (Marsland et al., 2002, 2005) and $0 \leq \psi \leq 1$ is the proportionality factor (Vieira Neto & Nehmzow, 2007a).

8. Decrease the habituation of s and of its neighbours (Vieira Neto & Nehmzow, 2007a):

$$\Delta h_s = (\alpha(1 - h_s) - 1)/\tau) \tag{7}$$

$$\Delta h_n = (\alpha(1 - h_n) - 1)/\min(1, \frac{a_s}{\psi a_n})\tau) \tag{8}$$

Unlike in (Marsland et al., 2002), we never deleted connections when they were too old, since setting the maximum connection age often led to worse network performance in the environments explored here. The network parameters were

set to $\Theta_H = 0.3$ (Vieira Neto & Nehmzow, 2007a), $\eta = 0.3$ (Marsland et al., 2002), $\alpha = 1.05$, $\tau = 3.33$ (Marsland et al., 2002; Vieira Neto & Nehmzow, 2007a) and $\psi = 0.1$ (Vieira Neto & Nehmzow, 2007a). The parameters Θ_A and N_{W_c} strongly affected the network behaviour and different network architectures thus resulted from different combinations of their values. Two of these architectures, Specialist and Generalist, are identified in Section 4.1.

3.3. The Plastic GWRNN

In the Plastic GWRNN, introduced here for the first time, different CL neurons may each have a different number of input connections, reflecting the fact that a robot exists in a heterogeneous environment and may thus receive feature vectors of variable length. At the beginning of an experiment, two CL neurons in a Plastic network are initialised with $N_{W_c} = 3$, allowing them to represent a single visual feature. In order to study how the way in which neurons adapt to the size of the current sensory input vector affects the network's performance and robustness, two types of Plastic GWRNN are explored here.

3.3.1. Rapid Growth Plastic GWRNN

The Rapid Growth (RG) Plastic network implements a solution in which all neurons that are currently being adapted, i.e., the winning neuron as well as its neighbours, automatically grow new input connections when they cannot represent the full sensory input vector. There are the following differences in the learning algorithms of the RG Plastic and the Standard GWRNN:

- In Step 1, make the number of input neurons match the size of the sensory input vector, i.e., in each time step, $N_I = N_P$.
- For each neuron c, use only the first M_c values when calculating its distance d_c from the current input vector:

$$M_c = \min(N_I, N_{W_c}) \tag{9}$$

$$d_c = \sqrt{\sum_{j}^{M_c} (i_j - w_{j,c})^2} \tag{10}$$

This change affects which neurons are selected as the winning neuron and the runner-up in Steps 2 - 3, as well as the value of the winning neuron activation value, a_s, calculated in Step 5.

- In Step 6, in addition to the existing conditions under which a new clustering layer neuron is added to the network, also create a new neuron when the winning neuron does not have enough input connections to fully represent the input, i.e., when $N_{W_s} < N_I$. The new neuron is assigned N_I input connections.

- In Step 7, adapt only the first M_c connections of the winning neuron and its neighbours when $N_I < N_{W_c}$.

  If the winning or a neighbouring neuron does not have enough input connections to represent the input vector ($N_I > N_{W_c}$), add new connections with weights initialised randomly between 0 and 1, so that $N_{W_c} = N_I$ and then adapt those connections using Eq. 5 or 6.

The RG Plastic networks are designed to initially grow rapidly and generalise input features to a high degree, but stop learning relatively early.

3.3.2. Balanced Growth Plastic GWRNN

The Balanced Growth (BG) Plastic network represents a solution where growth of new neurons and input connections follows more conservative rules. Neural growth is triggered using the Standard GWRNN rules, but neurons are "punished" by having a lower activation when the number of their input connections does not match the current input vector size. Unlike in the RG Plastic network, this ensures that new neurons are added both when the winning neuron has too many or too few input connections, and that growth of new neurons in situations when the network already represents the current sensory

input quite well is prevented. Moreover, when it comes to input connections of existing neurons, only the winning neuron, but not its neighbours, grow them in order to match the size of the current input vector.

Specifically, there are the following differences in the learning algorithms of the BG Plastic and the Standard GWRNN:

- In Step 1, make the number of input neurons match the size of the current sensory input vector, i.e., in each time step, $N_I = N_P$.
- In Step 2, add 1 to the distance d_c of a CL neuron for each missing or redundant input connection, penalising neurons that do not match dimensionality of the input vector:

$$O_c = \max(N_I, N_{W_c}) - M_c \tag{11}$$

$$d_c = \sqrt{\sum_{j}^{M_c}(i_j - w_{j,c})^2} + O_c \tag{12}$$

  As in the RG Plastic network, the updated equation for d_c affects the network update algorithm Steps 2, 3 and 5.

- In Step 6, a newly created neuron is assigned N_I input connections.
- In Step 7, similar to the RG Plastic network, adapt only the first M_c connections of the winning and the neighbouring neurons when $N_I < N_{W_c}$.

  Also, similar to the RG Plastic network, add input connections to the winning neuron so that $N_{W_s} = N_I$ if $N_I > N_{W_s}$. However, keep the number of neighbour neuron input connections the same.

The clustering layers of BG Plastic networks will tend to contain neurons with a more varied number of input connections, reflecting the dimensionality of the input space more closely.

4. One-shot novelty reporting

Following the experimental methodology of Marsland et al. (2005); Miskon & Russell (2009); Lepora et al. (2010), we first performed experiments in the one-shot novelty reporting task, where the robot was asked to report that a single new object appeared in the environment after the initial learning phase. In Room, learning was performed with only walls visible to the robot and one box was added in a random corner at the beginning of the operational phase. In Corridor, learning was performed with all doors closed, after which one random door was opened. We first performed parameter analysis and identified viable network architectures that performed the task well. We also tested how efficient and robust the various architectures were.

4.1. Network architectures

There are two parameters that significantly impact how quickly the GWRNN learns the initial environment and how successful it is in reporting novelties, namely, the number of input connections, N_{W_c}, and the activation threshold, Θ_A. A well-performing network should report novelty in at least 90%[1] of experimental runs (i.e., have $h_s \geq 0.9$ when the robot is near a novel object[2]). It should also learn quickly, i.e., stop outputting $h_s \geq 0.9$ during the learning phase as soon as possible. By extension, the number of times when a well-performing network that stopped learning reports false positives in one-shot novelty task is 0. There are two ways in which a Standard well-performing network can be setup - either by combining a small N_{W_c} with a high Θ_A, or vice-versa.

[1] Reported success rates over a number of experimental runs on various novelty detectors are: support-vector-machine-based – above 80% (Sofman et al., 2011), Bayes-classifier–based - above 90% (Lepora et al., 2010), Gaussian-Mixture-Model-based – around 90% (Drews Jr et al., 2010). 90% success rate is also an expected standard for detecting unusual component behaviour in unmanned aircrafts (McCall et al., 2013).

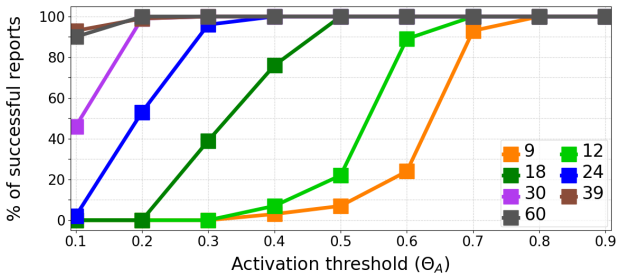
[2] The value of $h_s \geq 0.9$ was chosen experimentally to maximise reporting performance and minimise learning time of networks.

When N_{W_c} is relatively small ($N_{W_c} \leq 12$, Figure 4), the network can only consider a small subset of features found in the environment. Therefore, instead of re-training existing neurons in the clustering layer to approximate a large number of input vectors, it is more beneficial to add more neurons that can specialise in encoding different small parts of the input space. This can only be ensured when Θ_A is relatively high (see Section 3.2, Step 6). In such *Specialist* networks, the final size of the clustering layer, N_C, is relatively large and the network requires many passes through the environment, L, to learn it (Table 1).

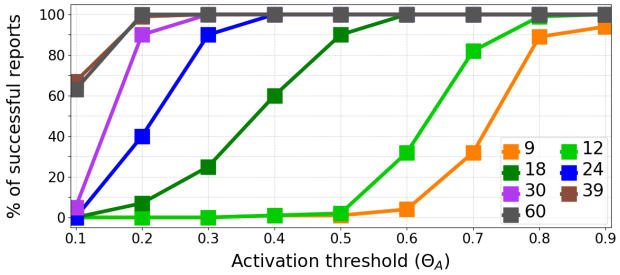
On the other hand, when N_{W_c} is relatively large ($N_{W_c} \geq 30$), each neuron in the network encodes a large input vector. Neurons can more easily be adapted to generalise over a number of input vectors, and it is thus not necessary to set Θ_A to a large value. Resulting *Generalist* networks thus have a relatively small size and learning time.

Table 1 lists key parameters and learning time for the Specialist and Generalist networks in Room and Corridor. Note that the parameters needed to be set differently in each environment in order to optimise the learning time and reporting success rate. Specifically, in Room, the input vectors that the networks received tended to be shorter and less diverse than in Corridor - a smaller set of features was usually sensed by the robot and the red lights of the door corners from Corridor were not present. As a result, the networks tended to require a smaller number of input connections and the size of the final clustering layer tended to be smaller. In addition, in the case of the Specialist network, a lower activation threshold than the one used in Corridor could be set, maintaining the network success rate (Figure 4), while reducing its learning time.

As Figure 4 and Table 1 demonstrate, suitable values for N_{W_c} and Θ_A are environment-dependent and setting them is thus time-consuming. In contrast, the Plastic networks adjust N_{W_c} automatically and the same Θ_A can be used in both environments to achieve the target performance. Neurons in the Plastic networks tend to have a smaller average number of input connections and their clustering layers tend to be smaller and learned faster than those of the non-Plastic networks.



(a) Room



(b) Corridor

Figure 4: The impact of network architecture parameters on novelty detection performance. Each data point shows the percentage of 100 simulation runs in which the presence of a novel object was successfully reported when the robot encountered it. Each curve depicts the influence of the neural activation threshold (Θ_A) on performance for a GWRNN with neurons receiving a specific number of input connections, $N_{W_c} \in \{9, 12, 18, 24, 30, 39, 60\}$.

The Plastic networks also exhibit more efficient neuron usage (averaging over 100 independent experimental runs). For example, in Corridor, around 60% of Specialist network neurons were redundant and did not fire, i.e., were not selected as winning neurons in the operational phase. A single neuron fired on average 1.6 times. Fire counts of neurons in the Balanced Growth Plastic and Generalist networks were distributed more evenly and the average fire count was 6.5 and 7.4, respectively, while the clustering layers in these networks were smaller than in Specialist. This means that individual neurons in BG Plastic and Generalist networks were reused more often, i.e., that the networks were more efficient in representing the input space. The Rapid Growth Plastic network used

Table 1: Characteristics of the different network architectures after the initial learning phase: Number of input connections (N_{W_c}), activation threshold (Θ_A), number of clustering layer neurons (N_C), and number of learning passes (L). Where these properties vary from network to network they are reported as medians from 100 experimental runs (with standard deviations in parentheses).

	Specialist	Generalist	RG Plastic	BG Plastic
N_{W_c}	9	30	25.5 (2.8)	18 (2.0)
Θ_A	0.7	0.2	0.2	0.2
N_C	48.5 (5.3)	17.5 (3.6)	13 (4.2)	23 (59.6)
L	8 (5.8)	6 (5.7)	1.5 (3.9)	3 (4.6)

(a) Room

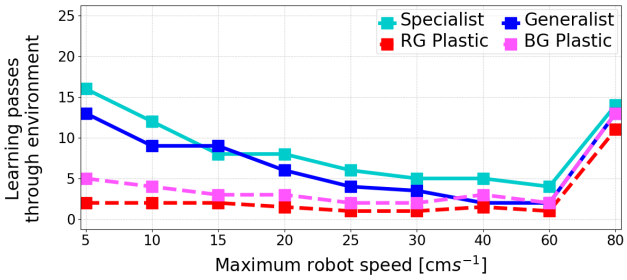
	Specialist	Generalist	RG Plastic	BG Plastic
N_{W_c}	12	39	34 (3.7)	25 (2.7)
Θ_A	0.8	0.2	0.2	0.2
N_C	304.5 (17.7)	70.5 (16.7)	46 (15.9)	77 (54.7)
L	13 (2.9)	11 (5.5)	3 (3.9)	6.5 (5.1)

(b) Corridor

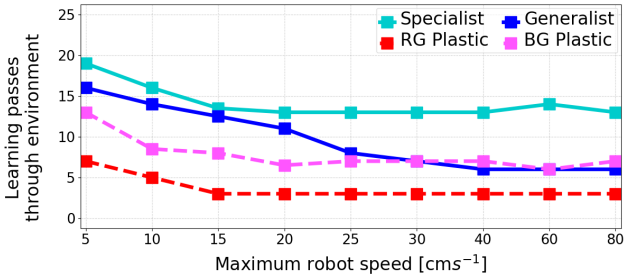
its neurons most efficiently, with an average fire count of 11.3 and the smallest clustering layer size. Similar comparisons between the networks were found in Room, although the neuron fire counts in this environment were more evenly distributed and the average fire rate of neurons was between 11 (Specialist) and 39 (RG Plastic network).

The improved efficiency of the Plastic networks relates to the way in which the networks grow. When a new neuron is added to a Standard network, the neuron can have either too many or too few input connections. In the first case, i.e., when $N_I > N_P$, the neuron initially tries to represent the 0s added to the end of the input vector (see Section 3.2, algorithm Step 1). However, if the neuron or its neighbours repeatedly fire again when the robot is somewhere else, the neuron is re-trained to an input vector with a different number of 0s, significantly diverging from the input vector that it was originally created for. In

the opposite situation, when $N_I < N_P$, the neuron does not have enough input connections to capture the full stimulus received by the robot and may miss features that are important but further away. By contrast, when a new neuron is added to the Plastic network, its input connections vector is set to the same size as that of the current input vector, meaning that it is able to represent the new input fairly well. Moreover, if a new shorter input vector is presented to the neuron later on, only the first $N_I(t)$ connections are adapted and representations of more distant features are not affected.

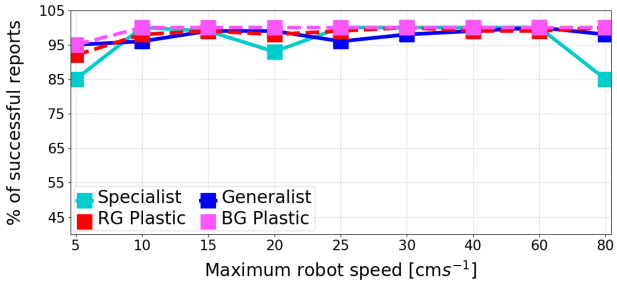


(a) Room

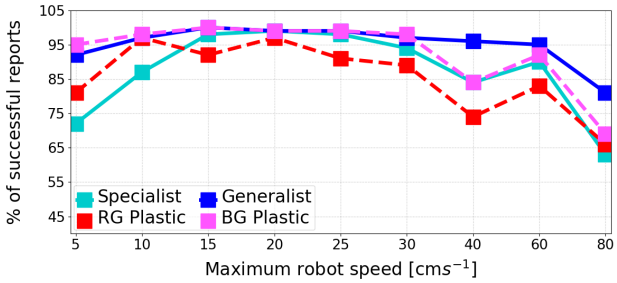


(b) Corridor

Figure 5: Each curve depicts the impact of maximum robot speed on novelty detection performance in the learning phase for one of the four classes of GWRNN. Each data point shows the median number of passes through the environment that a robot required in order to learn the Room or Corridor environment, calculated over 100 simulation runs.



(a) Room



(b) Corridor

Figure 6: Each curve depicts the impact of maximum robot speed on novelty detection performance in the detection phase for one of the four classes of GWRNN. Each data point shows the percentage of 100 simulation runs in which the presence of a novel object was successfully reported when the robot encountered it.

4.2. Robustness

Aside from being smaller, learning faster and using their neurons more efficiently, the Plastic neural networks, especially the Rapid Growth network, are also more robust to different robot speeds and to sensory noise (though see Sections 5 for consideration of performance in scenarios that are more involved than the one-shot novelty detection task explored here).

In the first set of robustness experiments, the speed with which the robot moved through the environment was varied. Previously, the robot travelled at a maximum of 20 cms^{-1}. When the robot travelled more slowly, a sequence of successive inputs tended to be more similar to one another. Neurons were thus adapted for each particular feature set to a higher degree, "forgetting" inputs

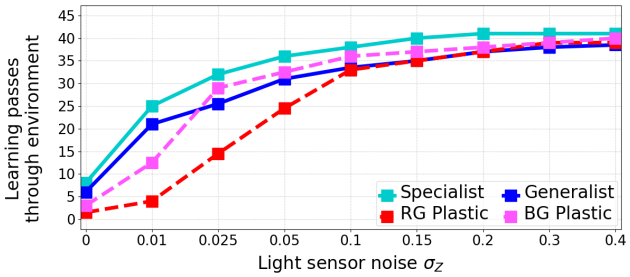
that they learned before. As a result, the networks needed a longer time to learn their environment (Figure 5). However, the effect of robot speed on learning time was significantly weaker in the Plastic networks, especially in Room. The Rapid Growth network was the most robust to robot speed.

After learning was complete, robot speed did not affect the true positive rate of the networks in Room which remained at or close to 100% (Figure 6). In Corridor, performance deteriorated slightly for most networks at speeds ≥ 40 cms^{-1}, while the Generalist network was able to maintain $\geq 90\%$ true positive rate for speeds of up to 60 cms^{-1}. When the robot travelled more quickly, its sensor was less often exposed to the green lights of the open door that were higher up than the wall and the door corner lights, causing the number of input vectors that began with novel values to be smaller. The Specialist neurons, that only considered the four closest features, were thus more likely to miss the novelty. Similarly, while neurons of Plastic networks each received an input vector of the same size as the raw sensory input vector, this input vector was sometimes matched with neurons that represented the closer features of walls (Eq 9). On the other hand, when a Generalist encountered a longer pattern of wall and open door features, the tail of its input vector that encoded the more distant open door green lights was significantly different from 0s that were found in tails of input vectors learned when only walls were sensed.

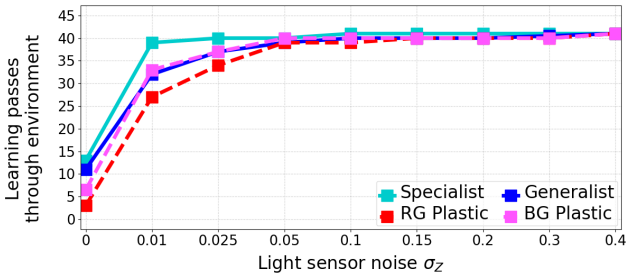
In the second experiment, Gaussian noise with a standard deviation σ_Z was added to each of the input values that the robot received from its light sensor. This type of noise modelled cases when, for instance, the pre-processing algorithm would make incorrect evaluation of the relative distance, direction or colour of a feature as a result of noisy camera input. All networks required more time to learn when noise was added, and we thus introduced novelty into the environments after 40 instead of 20 passes.

While the time required for a network to learn its environment was affected by the level of sensory noise in a way that depended on the environment and the network architecture (Figure 7), the ultimate performance achieved by the networks after learning was not affected as significantly by the level of sensory

noise (Figure 8). For Room, all networks reported novelty over 90% of the time irrespective of senory noise; for Corridor, all networks reported novelty around 70% of the time or greater for $\sigma_Z \geq 0.01$. Noise had a larger impact on learning time in Corridor, where features were more diverse and therefore generally more difficult to learn. The Specialist network was the least robust to noise, and could only learn sufficiently quickly (i.e., before a novelty was introduced) when $\sigma_Z \leq 0.1$ in Room and $\sigma_Z < 0.01$ in Corridor. When the noise was stronger, many input vectors looked very different from one another. As a result, neurons in the Specialist network, which was only able to consider small feature sets, had to be added significantly more often, causing it to grow to a size an order of magnitude larger than the other networks and preventing it from forming

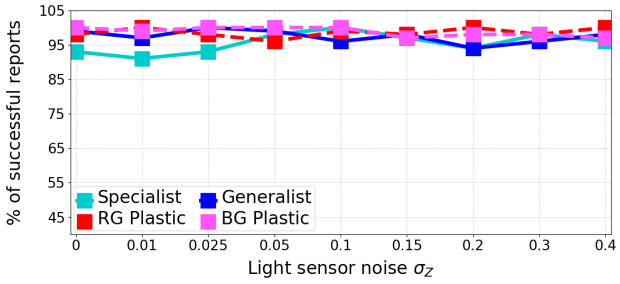


(a) Room

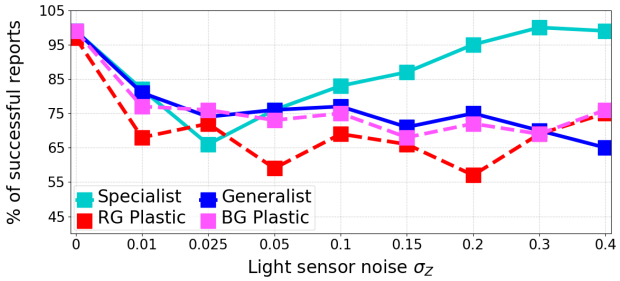


(b) Corridor

Figure 7: Each curve depicts the impact of sensor noise standard deviation, σ_z, on novelty detection performance in the learning phase for one of the four classes of GWRNN. Each data point shows the median number of passes through the environment that a robot required in order to learn the Room or Corridor environment, calculated over 100 simulation runs.



(a) Room



(b) Corridor

Figure 8: Each curve depicts the impact of sensor noise standard deviation, σ_z, on novelty detection performance in the detection phase for one of the four classes of GWRNN. Each data point shows the percentage of 100 simulation runs in which the presence of a novel object was successfully reported when the robot encountered it.

useful representations. As a consequence, specialist networks tended to report novelty very frequently, regardless of whether a novel object was present or not. By contrast, the Rapid Growth Plastic network, where an average neuron generalised over the largest number of features, was the most robust, especially in Room.

5. Continuous novelty reporting and learning

Having identified several GWRNN architectures that were in principle capable of learning an effective representation of their input space and of identifying novelties, we tested the networks in more realistic environments, where novelties occurred consecutively and thus needed to be reported and learned repeatedly.

Specifically, we were interested to identify:

1. Whether the networks were capable of detecting successive, distinct novelties, i.e., whether a series of successive novelties would each be reported with the same success rate irrespective of when they occurred

2. How quickly the successive novelties were learned

3. Whether a learned representation was stable, i.e., whether exposing the networks to successive distinct novelties would cause false positives to be reported

The networks were tested on two different tasks:

- ***Spot Presence***: A series of four novel objects appeared in the environment, with each new appearance separated from the next by a period of T_A seconds.

- ***Spot Absence***: Four objects that were initially present in the environment were removed sequentially, with each removal separated from the next by a period of T_A seconds.

Note that an "object" in Room corresponded to a box. In the Spot Presence task, each of the four corners of the room were initially empty and one box was added to a random empty corner every T_A seconds. In the Spot Absence task, each of the four corners had a box in it initially, and one random box was removed every T_A seconds. In Corridor, an "object" corresponded to an open door. In the Spot Presence task, all doors were initially closed and one randomly chosen door opened every T_A seconds. In the Spot Absence task, all doors were initially opened and one door closed every T_A seconds.

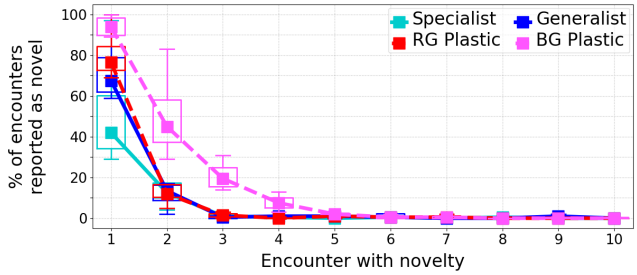
There were two variants of each task, characterised by the value of T_A. In a *Quick* task, $T_A = 50$ s, i.e., a novelty occurred each time a robot completed a pass through the environment. In a *Slow* task, $T_A = 250$ s, i.e., a novelty occurred each fifth pass through the environment.

5.1. The Spot Presence task

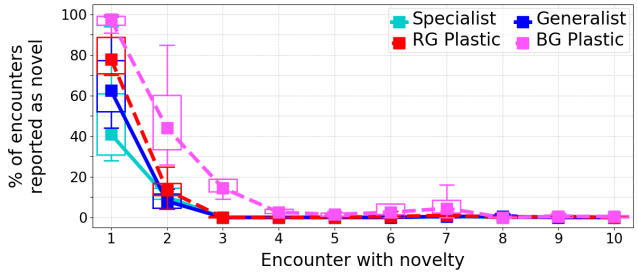
The novelty reporting rate of the networks in the Spot Presence task is shown in Figures 9 and 10. Note that there are three aspects of these figures that are important. The first is the network reporting performance, i.e., the median success rate during the first encounter with a novelty, which should be as high as possible. The second is the variance of the success rate, represented by the whisker box around each median data point, which shows how the reporting performance varied across the four novelties that a robot encountered. A low variance is desirable, because it indicates that each of the four novelties was reported with a similar success rate. The third aspect that Figures 9 and 10 show is the learning speed, characterised by how many encounters with novelties it takes for a network to stop reporting them as novel. A desired learning speed may be context dependent - for example, during surveillance, unexpected intruders may need to be reported no matter how many times a robot successively encounters them, while in environments and tasks that are expected to be dynamic, it may be desirable to detect novelties quickly and report them, but quickly learn that they are now present in the environment in order to never report them again.

While the change interval, T_A, did not significantly impact the ability of the networks to report novelties, the performance was generally lower for Room (Figure 9) compared to Corridor (Figure 10). This related to the difficulty networks experienced in attempting to distinguish between different boxes in Room, as is indicated by a high performance variance of each network, especially during the first encounter with a novelty. While the appearance of the first box was usually reported around 90% of the time, subsequent novel boxes were reported much less frequently. For instance, when $T_A = 250$s, Specialist GWRNNs reported the last box in only around 30% of experiments while Generalist GWRNNs reported it in around 45% of experiments. In general, this problem was less common in Corridor, where individual doors and wall features close to them were more varied.

There were also significant differences in how many encounters it took for the networks to learn a particular novelty. The Generalist and Rapid Growth Plastic



(a) Room, Quick Spot Presence task ($T_A = 50$ s)



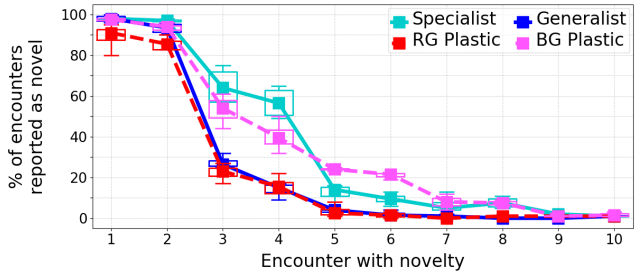
(b) Room, Slow Spot Presence task ($T_A = 250$ s)

Figure 9: Each curve depicts the impact on novelty detection performance of exposure to a sequence of four novel objects in (a) quick succession or (b) slower succession in the Room environment for one of the four classes of GWRNN. Each whisker box summarises the N^{th} encounter with any of the four distinct novelties encountered, averaged across 100 runs (boxes and whiskers show the inter-quartile range and 1.5 times the inter-quartile range, respectively).

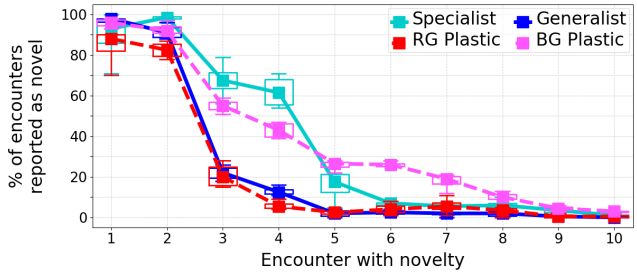
networks usually required a smaller number of encounters before they stopped reporting an opened door or a box as novel. Interestingly, despite being able to learn faster than Generalist in the initial learning phase, the Balanced Growth Plastic network required approximately as many encounters as Specialist to learn a novelty in Corridor and the largest number of encounters when learning in Room.

Finally, all networks reported false positives very rarely. We calculated the *false discovery rate* (FDR) of a network as:

$$FDR = FP/(FP + TP) \tag{13}$$



(a) Corridor, Quick Spot Presence task ($T_A = 50$ s)



(b) Corridor, Slow Spot Presence task ($T_A = 250$ s)

Figure 10: Each curve depicts the impact on novelty detection performance of exposure to a sequence of four novel objects in (a) quick succession or (b) slower succession in the Corridor environment for one of the four classes of GWRNN. Each whisker box summarises the N^{th} encounter with any of the four distinct novelties encountered, averaged across 100 runs (boxes and whiskers show the inter-quartile range and 1.5 times the inter-quartile range, respectively).

where FP was the number of false positives reported after the first encounter with a novelty and TP was the number of reported true positives. In all networks, FDR was below 0.1, i.e., for each reported false positive, there were at least 10 true positives reported.

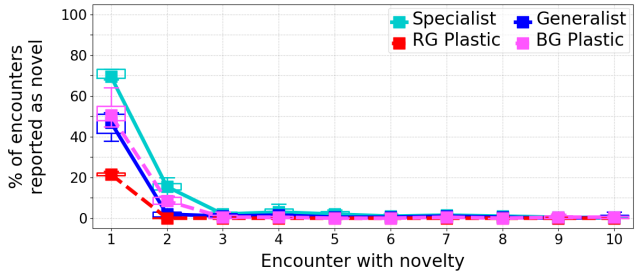
Overall, the Balanced Growth Plastic network exhibited the most robust performance across the two tested environments, with over 90% success rate in reporting novelties, while it normally needed a larger number of subsequent encounters with the same novelty in order to integrate it into its model of normality.

5.2. The Spot Absence task

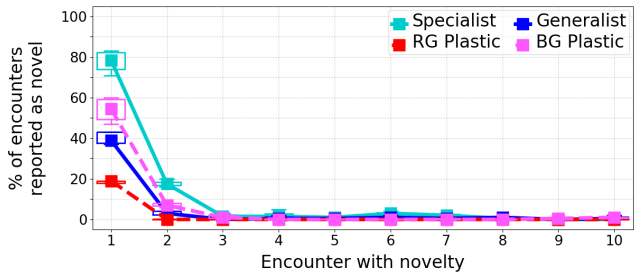
The performance of the networks on the Spot Absence task differed significantly from that on the Spot Presence task reported above. In the Spot Presence task, the networks first learned the properties of a particular environment with very similar features (mostly blue wall features and occasional red door corners in Corridor) and then operated in an environment featuring completely new sensory input vectors that included green colours for opened doors and boxes. It was therefore relatively easy to flag the presence of this type of novelty in the sensory input vector. On the other hand, in the Spot Absence task, the networks first learned a relatively feature-rich environment with all possible features present and were then asked to report the absence of these features.

As a result, the Rapid Growth plastic neural network achieved a very poor performance (around 15%-25% success rate, depending on the environment, see Figures 11 and 12) due to its strong tendency to generalise input vectors. When a previously learned object disappeared, the network simply matched the new sensory input vector with walls that it had already learned at a different location. The performance of the Generalist and the Balanced Growth Plastic network was better (usually around 40%-60% success rate), but the networks could never outperform Specialist. Since the Specialist network tended to encode small parts of the environment using separate neurons, it had a much greater ability to recognise new input vectors, even when novel scenes were composed of relatively commonly occurring environmental features.

It is also interesting to note that in Room, the Specialist network performed significantly better at the Spot Absence task (Figure 11) than at the Spot Presence task (Figure 9), by a margin of around 30%-40%. In the Spot Presence task, the robot approached each box at a similar angle, received a very similar input vector and thus was less able to recognise boxes added later on as novel. On the other hand, in the Spot Absence task, the walls uncovered after each box disappeared were more dissimilar from each other, since positions of the blue wall lights were randomised. As a result, the network detected each novel absence with a similar success rate.



(a) Room, Quick Spot Absence task ($T_A = 50$ s)



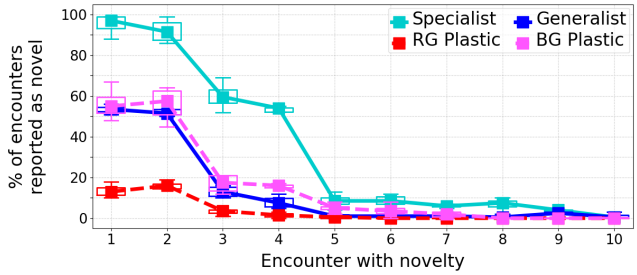
(b) Room, Slow Spot Absence task ($T_A = 250$ s)

Figure 11: Each curve depicts the impact on novelty absence detection performance of exposure to a sequence of four object removals in (a) quick succession or (b) slower succession in the Room environment for one of the four classes of GWRNN. Each whisker box summarises the N^{th} encounter with any of the four distinct removed objects, averaged across 100 runs (boxes and whiskers show the inter-quartile range and 1.5 times the inter-quartile range, respectively).

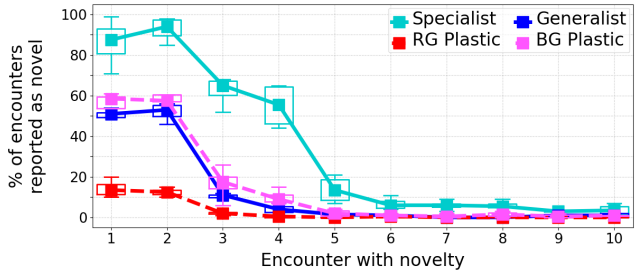
As in the Spot Presence tasks, false positives were reported very rarely by all networks. Consecutive encounters with novelties were reported less frequently than in the Spot Presence task, which was expected given the generally smaller success rates during the first encounters with a particular novel absence.

5.3. Location data

The inability of the networks to distinguish between individual successive novelties and to report the absence of learned objects can be addressed by dealing with a serious drawback with the network input data, namely the fact that there was no way for the networks to associate the given sensory input vector with the robot's location. We therefore added two input values at the beginning of the



(a) Corridor, Quick Spot Absence task ($T_A = 50$ s)



(b) Corridor, Slow Spot Absence task ($T_A = 250$ s)

Figure 12: Each curve depicts the impact on novelty absence detection performance of exposure to a sequence of four object removals in (a) quick succession or (b) slower succession in the Corridor environment for one of the four classes of GWRNN. Each whisker box summarises the N^{th} encounter with any of the four distinct removed objects, averaged across 100 runs (boxes and whiskers show the inter-quartile range and 1.5 times the inter-quartile range, respectively).

input vectors, representing the x and y coordinates of the robot, normalised to lie between -1 and 1 with respect to the designated arena size (Figure 13).Achieving this type of localisation information is not trivial for a real robot, but can be approached using techniques such as Visual SLAM (Davison et al., 2007).

Adding location data allowed all networks in the Spot Presence task to report 100% of the novelties encountered in the Room environment, which was a significant improvement over the previous performance (Figure 14a). Improvements were also achieved in Corridor (Figure 14b), although in this environment, the network performance was already relatively good prior to using localisation. Similarly, having location data significantly improved performance

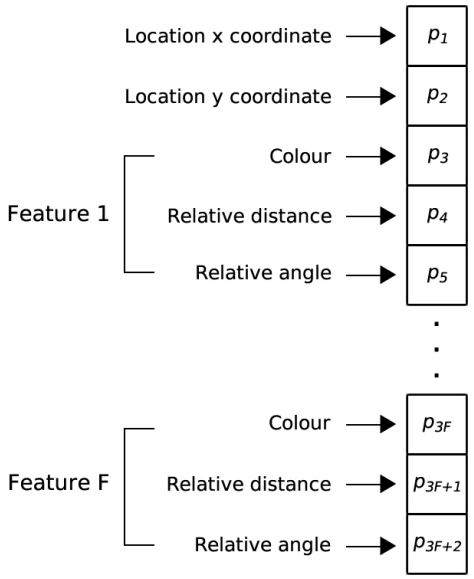


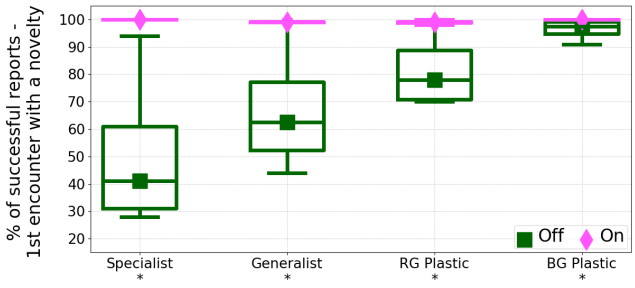
Figure 13: The sensory input vector of elements p_1 through p_{3F+2}, representing the three sensed attributes associated with each of F sensed environmental features, plus added normalised location data (p_1 and p_2).

of the networks in the Spot Absence task (Figure 15).

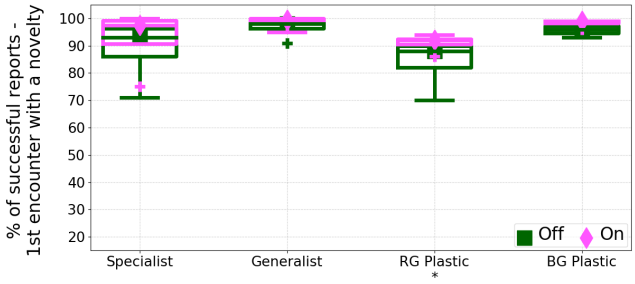
The false discovery rate of the networks and the number of encounters required to learn novelties were not significantly affected by the addition of location data.

6. Discussion

It was demonstrated in Sections 4 and 5 that performance of a GWRNN can vary significantly depending on the environment and the nature of the novelty detection task. The Balanced Growth plastic networks exhibited the most robust performance across all the experiments presented here, although the networks had, in some cases, lower than 90% success rates. A distinguishing feature of this type of network was that the number of input connections varied significantly between individual neurons, pointing towards the possibility that having neurons that fulfill different roles could be beneficial to further improve the performance and robustness of the GWRNN. The next step in our research will therefore



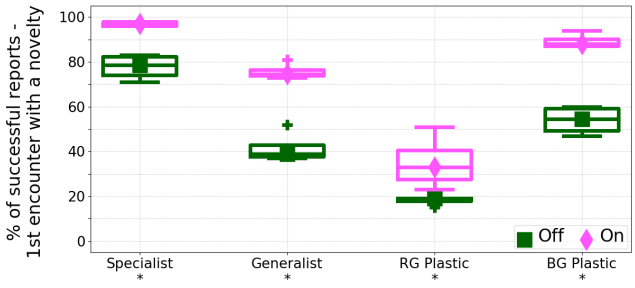
(a) Room, Slow Spot Presence task



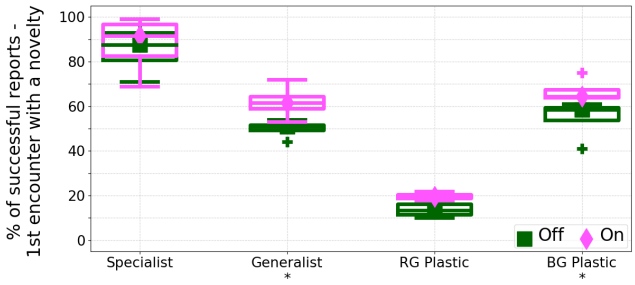
(b) Corridor, Slow Spot Presence task

Figure 14: The impact of providing location data as part of the sensory input vector on the success rate of reporting the first encounter with a novelty in the Slow Spot Presence task. Statistically significant differences are indicated by an asterisk (Wilcoxon signed-rank test, $p < 0.01$). Similar effects were found in the Quick variant of this task.

involve identifying a mechanism by which both specialist and generalist neurons can be grown in a single network, and thus minimising the extent to which the trade-off between robustness and fidelity has to be made. Candidate solutions to this problem include a) making the activation threshold of individual neurons adaptive and b) training multiple models with different parameters at the same time, and having a mechanism to decide which output is currently more relevant, possibly in a multi-layer network architecture. In attempting to achieve systems that are able to carry out robust novelty detection over multiple time scales, there may also be some value in adopting an approach informed by the distinction between "fast" and "slow" decision-making systems, described most famously by Kahneman (2013). One additional avenue that could be explored is reviewing the



(a) Room, Slow Spot Absence task



(b) Corridor, Slow Spot Absence task

Figure 15: The impact of providing location data as part of the sensory input vector on the success rate of reporting the first encounter with a novelty in the Slow Spot Absence task. Statistically significant differences are indicated by an asterisk (Wilcoxon signed-rank test, $p < 0.01$). Similar effects were found in the Quick variant of this task.

extent to which the findings reported here for GWRNNs (and our interpretation of them) generalise to other network architectures that involve network growth and adaptation.

The learning algorithm of the GWRNN could also be improved by preventing the network from learning rare outliers and thus allowing for their more robust detection. For example, in a novelty detector based on Support Vector Data Description (Wang et al., 2013), support vectors were only updated when a new data point was close to the boundary of the learned data distribution. Similarly, in AnyNovel, a semi-supervised learning algorithm that builds a fine-grained Baseline Learning Model (Abdallah et al., 2016), a distinction was made between "concept drift", i.e., gradual adaptation of a previously learned

normality model, and "anomalies" that were very different from the learned model. In the GWRNN, a metric based around the Euclidean distance of a new input vector from the winning neuron input connection weights could be used in a similar fashion to control learning.

Another issue that needs to be addressed is estimating confidence level when the GWRNN reports a novelty. Each experiment presented here was repeated in 100 independent runs. It was therefore possible to say, for instance, that the Balanced Growth Plastic network would report a new box in the Room around 95% of the time during the first encounter with it and around 45% of the time during the second encounter (Figure 9a). However, since novelty detection is effectively a two class classification problem, it is currently unclear how a confidence level of 95% could be assigned to the first and 45% to the second encounter in a particular run. It has been suggested elsewhere (Ma & Perkins, 2003) that novelty should only be reported if an algorithm classifies the current input as novel at least N times in a given T-second long time interval, especially when noisy sensors are involved. Following this reasoning, confidence level could be assigned as a function of N/T. Another solution, specific to GWRNNs, could involve evaluating the magnitude of changes in the winning neuron's habituation value during a recent time interval.

On a related subject, it would also be beneficial to devise a mechanism by which the novelty threshold, i.e., in the case of the GWRNN, the minimum value of a winning neuron's habituation that corresponds to novelty, could be adjusted in an ad-hoc fashion. Such adjustment can be performed in auto-associative neural networks using the learned shape of the data distribution (Sohn et al., 2001) and in Hopfield networks using the input vector size (Crook & Hayes, 2001). One related approach could be to keep the activation threshold fixed, but implement a variable, adaptive radius for the activation function. Vieira Neto and Nehmzow (2007a), for instance, use a fixed Gaussian radial basis function with a radius that depends on a parameter, σ, which could be adapted to cope with the changing character of a robot's environment or the varying dimensionality of its sensory input vector.

Despite its current challenges, the Grow When Require neural network shows great promise when it comes to novelty detection, since it does not require labelled data to be able learn a model of normality and since it can compress the input space to which it is exposed to into an efficient representation, similarly to Hopfield networks (Crook & Hayes, 2001) and Gaussian Mixture Models (Drews Jr et al., 2010). This is in contrast with, for example, novelty detectors based on Support Vector Machines (Ma & Perkins, 2003), where computational complexity grows with the number of samples that the algorithm has learned, and with supervised learning approaches (Markou & Singh, 2006), where a network is first trained on labelled data and then classifies new inputs.

Apart from novelty detection, GWRNN could also be used for other purposes. For example, (Vieira Neto & Nehmzow, 2007b) have shown how visual representations can be reconstructed from the learned models of normality. This could be useful, for instance, to aid explainability of robot behaviour. It may also be possible to highlight parts of the visual field (or another input modality) that are novel by finding out which parts of the input vector contribute the most to the difference from the learned model. This would enable *novelty segmentation*, previously demonstrated as a capability of a novelty detection system based on Gaussian Mixture Models of 3D point clouds (Drews Jr et al., 2010).

7. Contributions and Conclusion

The ability to detect novelties in their own performance and in their environment could make autonomous robots more robust and more safe. This paper focused on detecting novelty in a robot's visual field (similar to, e.g., Crook & Hayes, 2001; Marsland et al., 2005; Markou & Singh, 2006; Vieira Neto & Nehmzow, 2007a; Gatsoulis & McGinnity, 2015). The study presented here was conducted using the Grow When Required Neural Network (GWRNN). In Section 4, we have shown that there are two important parameters affecting the network's performance and learning speed – the number of neuron input connections, N_{W_c} (which determines how many features a network can consider

at a given point in time), and neuron activation threshold, Θ_A (which controls the network's growth). We have demonstrated that using a new Plastic variant of the GWRNN, where the value of N_{W_c} varies from one clustering neuron to another based on the size of input vectors that they represent, leads to more robust performance. In general, we conclude that it is desirable to create adaptation mechanisms that automatically adjust a novelty detectors parameters but that are parameter-less themselves.

Secondly, we have demonstrated in Section 5 that a novelty detector may perform fundamentally differently in the one-shot and in the continuous variant of the novelty detection task. In the latter, a series of successive distinct novelties take place sequentially and a novelty detector needs to be able to distinguish between different novelties that may have similar features but that occur in different places or at different times. This may be especially important in robotic applications such as surveillance and intruder detection.

Thirdly, we have also shown that performance of a novelty detector may vary significantly when it comes to recognising different types of novel events, for instance, the appearance of previously unseen objects versus disappearance of familiar objects.

Finally, we have demonstrated that adding localisation information to the sensory input vector of a novelty detector often improves its ability to distinguish between different objects and to detect the absence of previously learned objects. Some benefits of using location data in heterogeneous environments were previously outlined by Miskon & Russell (2009), although in their paper trained neurons in a self-organising network were associated with spatial regions using a look-up table. Here we have shown that it is possible for networks to benefit from simply adding normalised location data into the sensory input vector directly. It should be noted, however, that the impact of location data noise on learning speed and on the likelihood of reporting false positives needs to be studied in detail.

The experiments were performed with a number of different GWRNN architectures (Specialist, Generalist, Rapid Growth Plastic and Balanced Growth

Plastic) that facilitated generalisation over observed features to different degrees. A conclusion can be drawn that there is a trade-off between a network's ability to be robust, which is a result of generalisation, and its ability to detect small differences in input vectors with a high fidelity, which results from individual neurons being able to specialise in representing particular detailed features. None of the tested networks were able to report novelties with a sufficient success rate in all environments and tasks. The robustness-fidelity trade-off was most prominent when comparing the Specialist and the Rapid Growth Plastic networks. Specialist networks grew to the largest size, required a long learning time and had poor robustness, but were able to successfully recognise minor differences in input vectors when familiar objects disappeared, revealing a background of relatively unremarkable features. On the other hand, the Rapid Growth Plastic networks were the most efficient and robust, but performed very poorly when it came to distinguishing between distinct, consecutive novelties, or recognising that familiar objects had disappeared. A similar effect was described by Vieira Neto & Nehmzow (2007b), where neither GWRNNs nor incremental principal component analysis were able to distinguish inconspicuous features due to generalisation during learning, suggesting that the robustness-fidelity trade-off is not unique to neural networks.

Acknowledgements

This work was funded and delivered in partnership between the Thales Group and the University of Bristol and with the support of the UK Engineering and Physical Sciences Research Council Research Grant Award Reference EP/R004757/1 entitled "Thales-Bristol Partnership in Hybrid Autonomous Systems Engineering (T-B PHASE)".

We would like to thank Tom Kent for his help with the background literature and to Sam Daw, Tim Munn and Trevor Woolven from Thales for useful discussions regarding novelty detection in real-world systems.

The source code and data set are openly available at the University of Bristol

data repository, data.bris, at:
https://doi.org/10.5523/bris.249ybeqe4c73n2hmq5r9c5dr8u

References

Abdallah, Z. S., Gaber, M. M., Srinivasan, B., & Krishnaswamy, S. (2016). AnyNovel: Detection of novel concepts in evolving data streams. *Evolving Systems*, *7*, 73–93.

Albertini, M. K., & de Mello, R. F. (2007). A self-organizing neural network for detecting novelties. In *Proceedings of the 2007 ACM Symposium on Applied Computing (SAC '07)* (p. 462). New York, USA: ACM.

Bonani, M., Longchamp, V., Magnenat S., Philippe, R., Burnier, D., Roulet, G., Vaussard, F., Bleuler, H., & Mondada, F. (2010). The MarXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010)* (pp. 4187 – 4193). Piscataway, NJ: IEEE.

Carpenter, G. A., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organising neural network. *IEEE Computer*, *21*, 77–88.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection. *ACM Computing Surveys*, *41*, 1–58.

Crook, P. A., & Hayes, G. (2001). A robot implementation of a biologically inspired method for novelty detection. In *Proceedings of the Towards Intelligent Mobile Robots Conference (TIMR'01)*. doi:10.1111/j.1365-294X.2008.03906.x.

Davison, A. J., Reid, I. D., Molton, N. D., & Stasse, O. (2007). MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, *29*, 1052–1067.

Decker, R. (2005). Market basket analysis by means of a growing neural network. *The International Review of Retail, Distribution and Consumer Research*, *15*, 151–169.

Drews Jr, P., Nunez, P., Rocha, R., Campos, M., & Dias, J. (2010). Novelty detection and 3D shape retrieval using superquadrics and multi-scale sampling for autonomous mobile robots. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)* (pp. 3635–3640). Piscataway, NJ: IEEE.

Fink, O., Zio, E., & Weidmann, U. (2015). Novelty detection by multivariate kernel density estimation and growing neural gas algorithm. *Mechanical Systems and Signal Processing*, *50-51*, 427–436.

Fritzke, B. (1995). A Growing Neural Gas Network learns topologies. In G. Tesauro, D. S. Touretzky, & T. K. Leen (Eds.), *Proceedings of the 1994 Conference on Advances in Neural Information Processing Systems* (pp. 625–632). Cambridge, MA: MIT Press.

Gatsoulis, Y., Kerr, E., Condell, J. V., Siddique, N. H., & McGinnity, T. M. (2010). Novelty detection for cumulative learning. In *Proceedings of the 11th Towards Autonomous Robotic Systems Conference (TAROS 2010)* (pp. 62–67). Berlin: Springer.

Gatsoulis, Y., & McGinnity, T. M. (2015). Intrinsically motivated learning systems based on biologically-inspired novelty detection. *Robotics and Autonomous Systems*, *68*, 12–20.

Gonzalez, R., Apostolopoulos, D., & Iagnemma, K. (2018). Slippage and immobilization detection for planetary exploration rovers via machine learning and proprioceptive sensing. *Journal of Field Robotics*, *35*, 231–247.

He, Z., Xiaofe, X., & Schengchun, D. (2005). An optimization model for outlier detection in categorical data. *Advances in Intelligent Computing*, *3644*, 400–409.

Hung, C., & Wermter, S. (2003). A dynamic adaptive self-organising hybrid model for text clustering. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003)* (pp. 75–82). Piscataway, NJ: IEEE.

Kahneman, D. (2013). *Thinking, Fast and Slow*. Farrar Straus Giroux.

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*, 59–69.

Lepora, N. F., Pearson, M. J., Mitchinson, B., Evans, M., Fox, C., Pipe, T., Gurney, K., & Prescott, T. J. (2010). Naive Bayes novelty detection for a moving robot with whiskers. In *The 2010 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (pp. 131–136). Piscataway, NJ: IEEE.

Ma, J., & Perkins, S. (2003). Online novelty detection on temporal sequences. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03)* (pp. 613–618). New York, NY: ACM.

Markou, M., & Singh, S. (2006). A neural network-based novelty detector for image sequence analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*, 1664–1677.

Marsland, A., Nehmzow, U., & Shapiro, J. (2000). Detecting novel features of an environment using habituation. In J. A. Meyer, A. Berthoz, & D. Floreano (Eds.), *From Animals to Animats, Proceedings of the Sixth International Conference on Simulation of Adaptive Behaviour* (pp. 189—-198). Cambridge, MA: MIT Press.

Marsland, S., Nehmzow, U., & Shapiro, J. (2005). On-line novelty detection for autonomous mobile robots. *Robotics and Autonomous Systems*, *51*, 191–206.

Marsland, S., Shapiro, J., & Nehmzow, U. (2002). A self-organising network that grows when required. *Neural Networks*, *15*, 1041–1058.

McCall, M. S., Stephan, D. G., & Lackey, J. B. (2013). *ADS-79D-HDBK, Aeronautical Design Standard: Handbook for Condition Based Maintenance Systems for US Army Aircraft Systems*. Redstone Arsenal, AL, USA: US Army Aviation and Missile Research.

Merrick, K., Siddique, N., & Rano, I. (2016). Experience-based generation of maintenance and achievement goals on a mobile robot. *Paladyn, Journal of Behavioral Robotics*, *7*, 67–84.

Miskon, M. F., & Russell, R. A. (2009). Mapping normal sensor measurement using regions. In *Proceedings of the IEEE International Conference on Industrial Technology (ICIT 2009)* (pp. 1080–1085). Piscataway, NJ: IEEE.

Pimentel, M. A. F., Clifton, D. A., Clifton, L., & Tarassenko, L. (2014). A review of novelty detection. *Signal Processing*, *99*, 215–249.

Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Caro, G., Ducatelle, F., Birattari, M., Gambardella, L. M., & Dorigo, M. (2012). ARGoS: A modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, *6*, 271–295.

Ross, P., English, A., Ball, D., Upcroft, B., & Corke, P. (2015). Online novelty-based visual obstacle detection for field robotics. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA 2015)* (pp. 3935–3940). Piscataway, NJ: IEEE.

Sofman, B., Neuman, B., Stentz, A., & Bagnell, J. A. (2011). Anytime online novelty and change detection for mobile robots. *Journal of Field Robotics*, *28*, 589–618.

Sohn, H., Worden, K., & Farrar, C. R. (2001). Novelty detection under changing environmental conditions. In *Proceedings of SPIE's 8th Annual International Symposium on Smart Structures and Materials* (pp. 108–118). Bellingham, WA: The International Society for Optical Engineering.

Vieira Neto, H., & Nehmzow, U. (2007a). Real-time automated visual inspection using mobile robots. *Journal of Intelligent and Robotic Systems*, *49*, 293–307.

Vieira Neto, H., & Nehmzow, U. (2007b). Visual novelty detection with automatic scale selection. *Robotics and Autonomous Systems*, *55*, 693–701.

Wang, S., Yu, J., Lapira, E., & Lee, J. (2013). A modified support vector data description based novelty detection approach for machinery components. *Applied Soft Computing Journal*, *13*, 1193–1205.