IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 13 October 2022, accepted 8 November 2022, date of publication 17 November 2022, date of current version 22 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3222810

RESEARCH ARTICLE

Fusion of Novelty Detectors Using Deep and Local Invariant Visual Features for Inspection Task

EMRE ÖZBİLGE[1] **AND EBRU OZBILGE**[2]

[1]Department of Computer Engineering, Faculty of Engineering, Cyprus International University, Nicosia, 99258 North Cyprus, Turkey
[2]Department of Mathematics and Statistics, American University of the Middle East, Egaila 54200, Kuwait

Corresponding author: Emre Özbilge (eozbilge@ciu.edu.tr)

ABSTRACT In this study, a novel framework using multiple novelty detection filters is developed to learn a model of the normality of a robot's visual perception, which is called multichannel novelty detection. Subsequently, the acquired model was used to highlight dissimilar perceptions when the robot explored an environment. The main purpose of fusing multiple novelty filters is that each novelty filter performs well in detecting specific types of novelties; therefore, a new framework is proposed that demonstrates a new way to combine multiple different purposed novelty detection filters together in order to yield an overall more robust novelty status on the visual features. To develop a multichannel novelty detection system, expectation- and appearance-based novelty detection models were used in this study. To become experts in detecting different types of novelty using these models, different features from the input image were extracted as inputs for the models. The expectation-based novelty detection model uses the MobileNetV2 deep network to extract the deep features of the input image, which is subsequently used to learn a sequential and temporal model of normality to detect novelty. By contrast, the appearance-based novelty detection model uses speeded up robust features (SURF), which provide more region-focused features within the input image, to identify whether a specific region of the image is novel. The proposed multichannel novelty detection system is a completely online and real-time approach that is very important for mobile robotics applications. The proposed framework was tested in three novel environments, and it was reported that the proposed multichannel novelty detection system performs better than expectation-based and appearance-based novelty filters separately. Statistically, in the three novel environments, the Matthews correlation coefficients are reported to be 0.94, 0.97, and 0.93, and F_1 scores are reported to be 0.95, 0.97, and 0.93, respectively, which proves that and can be concluded as almost perfect statistically.

INDEX TERMS Neural networks, incremental learning, novelty detection, robot learning, local invariant features, transfer learning, deep learning.

I. INTRODUCTION

Novelty detection was employed to identify whether the input data differed from the previously observed data. It can be a useful tool for many engineering applications, such as identifying abnormal jet engine behaviours [1], identifying breast cancer [2], and detecting abnormal sensory readings for mobile robotics [3], [4]. In order to develop a novelty detection system, an appropriate machine learning system must be used in order to ascertain the model of frequently seen data (also called 'normal data'). Subsequently, the obtained model of normality can be used to filter out any new data that are not similar to the frequently repeated data (also called 'novel data'). The main reason for developing a model of normality instead of a model of novel data is that it is not possible to design a system to directly detect novel data; in particular, the characteristic properties of novel data are unknown because these data rarely occur [2], [5].

Several novelty detection approaches have been developed for a variety of applications. A review of these approaches is provided in [6], [7], and [8]. However, most of the approaches described in the literature have a model of normality

The associate editor coordinating the review of this manuscript and approving it for publication was Huiyu Zhou.

developed offline. This means that the data required to train the model of normality must be available beforehand. Thus, it is more appropriate to use online learning systems in mobile robotics. This is because the quantity of data received from the environment is unknown; therefore, the memory capacity of the robot may be insufficient to save the received data to train the model of normality, making this type of training impossible. Therefore, the main focus is on the online novelty detection used in mobile robots.

Marsland et al. [9] developed a Grow When Required (GWR) network for online novelty detection. This network is based on a self-organising map (SOM) that clusters the input data in a topological manner. The advantage of this approach is that the GWR network is dynamic, and its structure (or feature map) grows by the addition of new nodes when the input data are novel, and shrinks structurally when the existing node has not been activated because of a lack of learned, normal input data (a node represents learned knowledge). The authors of this study integrated a habituation method [10] at the top of the network to track the novelty degree of each node in the network. The proposed novelty detection network was demonstrated in various mobile robotic applications, where the robot learned to navigate a corridor on a different floor of the building and managed to identify novel sonar-range sensory data received while patrolling [11].

Alternatively, Gatsoulis and McGinnity [12] used a GWR network to detect whether the complete object was novel, instead of detecting novelty in each extracted feature vector at every time step. This is because some features of the objects can carry similar properties; therefore, in order to decide whether the object is novel, the objects are viewed at all angles and the winner of the voting determines whether the object is novel or known. To learn the properties of the object, local invariant features from the captured image were extracted and learned by the GWR network.

Later, Neto and Nehmzow [13], [14] proposed an attention mechanism to select interesting sensory features from an input image before presenting this information to the novelty filter. The authors demonstrated this approach using a GWR network and novelty filter based on incremental principal component analysis (PCA) [15] in the visual novelty detection system of a mobile robot. The attention selection mechanism was found to improve the generalisation performance of novelty filters by focusing on interesting visual regions in the input images instead of using the entire input image to determine the model of normality. In fact, the captured image did not always carry relevant information. Sometimes, the camera captured a uniformly patterned floor or wall. Thus, these types of visual features added to the novelty filter caused the loss of small interesting regions in the image when the entire image was used without an attention-selection mechanism in the learning system.

In contrast, Contreras-Cruz et al. [16] used a pretrained deep convolutional neural network (DCNN) for feature extraction to present the novelty filter. The authors claimed that deep features are more robust and reliable than traditional visual feature extraction methods such as colour histograms, colour angular indexing, and the GIST descriptor. They conducted experiments and compared deep features and other feature extraction methods separately using simple evolving connectionist systems (SECoS) and GWR network-based novelty filters. The results showed that deep-feature-based novelty detection outperformed traditional feature extraction methods.

These two approaches, GWR network and incremental PCA, examine the current input data to determine whether they are similar to the models learned by the corresponding novelty detection approach. The main disadvantage of these approaches is that they do not consider the sequential relationships between the input data obtained from the environment, such as the connection between two or more consecutive input data points at time t, $t+1$, ..., $t+n$. Consequently, it is impossible to detect any missing input data or changes in the location of the normal input data received from the environment. To overcome the temporal model of normality issues of these robot-based novelty detection approaches, [17], [18] recently proposed online expectation-based novelty detection for mobile robots, which models the temporal relationship between a series of input data received from the robot's environment while travelling. This system also processed the input data online; therefore, its structure also changed dynamically during the learning phase, according to the amount of novel data presented to the system. In this approach, novelty can be detected whenever the model of normality is unsuitable for predicting the forthcoming expected sensory data.

Importantly, each novelty detection approach is specialised to detect certain novelties in the environment. For example, although a GWR network and an attention selection mechanism together yield robust novelty detection, this combination considers only the selected area of interest in the current input image with the learned models of normality. By contrast, the expectation-based novelty detector learns the temporal relationship between the received sensory data; however, it ignores and fails to detect small novelties in the current input image, because its design organisation learns the entire image without feature engineering.

Consequently, in this paper, a multichannel novelty detector is proposed that intelligently combines multiple different purposed novelty detectors, which work in parallel to detect novelties. This approach was inspired by the behaviour-based architecture of the robot controller developed by Brooks [19], where this subsumption architecture combines all the motor actions obtained from various task-specific controllers to produce the robot's final action. Each novelty detector in the proposed system learns the model of normality using different types of features from the input image such that each detector becomes an expert in specific types of novelties. It is difficult to identify all types of novelty by using only one model of normality. Therefore, to avoid missing abnormalities in the corresponding task, the outputs of multiple novelty detectors were combined.

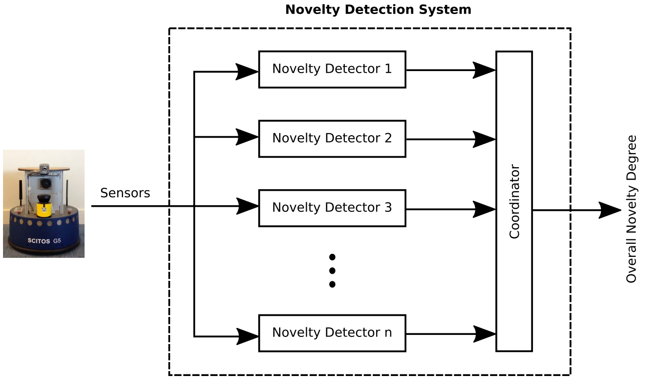


FIGURE 1. Overview of Brooks' subsumption architecture adapted to novelty detection tasks.

Two novelty detectors, expectation-based and appearance-based, were used in this study. Separate deep and local invariant features were extracted to feed these novelty detectors. Therefore, one detector focuses on local regions in the input image, whereas the other detector learns the normality model of more complex features within the input image extracted from the deep neural network. The main contributions of this study are as follows:

1) A novel framework was proposed that combined multiple novelty detectors for robust and reliable novelty detection.
2) The performances of the deep and local invariant image features as inputs to the novelty detection system were analysed and compared.
3) The weaknesses of the sublevel novelty detectors were identified, and a superior novelty detection performance was achieved using a multichannel novelty detector, which was statistically proven.

The remainder of this paper is organised as follows. In Section II, the proposed online learning-based multichannel novelty detection system for mobile robotics is described. In Section III, the experimental procedure, environment, and performance assessment metrics are described. The experimental results are discussed in Section IV. Finally, in Section V, conclusions are presented.

II. THE MULTICHANNEL NOVELTY FILTER

The proposed system comprises three main blocks, namely observation, feature extraction, and a model of normality, as shown in Figure 2. The system receives a raw colour image from the camera of the robot while it is moving through the environment. The captured image was then sent to the feature extraction block. In this block, two feature extraction techniques which yield distinct information from the environment are performed. First, a raw colour image is presented to a pretrained DCNN model to extract deep visual features. In parallel, a raw image is also presented to the visual attention selection mechanism to obtain visual features for local interest regions in the input image. Subsequently, the features extracted from the sensors passed through the associated novelty detectors. Finally, the degrees of novelty from both detectors are combined intelligently to make a final decision regarding the corresponding input image received from the environment.

The following sections break down the proposed novelty detection system into separate blocks and provide a detailed explanation of each block.

A. DEEP CONVOLUTIONAL NEURAL NETWORK

To extract the visual features from the raw image, a pretrained DCNN was used. This type of network consists of many hidden layers which are convolutional layers and fully connected dense layers, and contains normalisation and pooling operations between these layers. Because it comprises a large network, it is very expensive to train such a large network on a robot's computer which requires high memory and a fast graphics processing unit (GPU). In addition, the DCNN was trained offline (batch learning) which required many images. Therefore, a pretrained DCNN model is generally used to extract features that have already been trained in more than 14 million images of the ImageNet database [20], which is called transfer learning. In transfer learning, the pretrained network layer connection weights are frozen (*i.e.* learning-disabled), and the final classification (output) layer is removed from the network architecture. The activation values from the last layer of the DCNN model were used as the extracted deep features for the corresponding input image, which was presented to the network. Many different types of DCNN architectures are available; however, the MobileNetV2 architecture [21] was used for the feature extraction process. This network has the lowest number of parameters (≈3.5 million) compared with the other available ImageNet models in [22] which is very important for mobile robotics applications when there is limited processing power available onboard the robot.

The first version of MobileNet [23] introduced depthwise separable convolution which is computationally cheaper and faster than regular convolution and yields similar results. Regular convolution combines all input channels into one output channel by performing a simple weighted sum of the input pixels of all channels of the input image covered by a single kernel (*i.e.* filter). In practice, there is more than one kernel. Therefore, the resultant convoluted matrix has the same channel size as the number of kernels used in the convolutional layer. In contrast to regular convolution, depthwise separable convolution has two different convolution operations. First, a depthwise convolution with a 3×3 kernel is applied, which performs a convolution operation in each input channel separately; therefore, it produces the same number of channels as the input channels. Then, to combine channels obtained from depthwise convolution into a one-channel output, similar to the regular convolution, a pointwise convolution is used. This convolution is a regular convolution, but it uses a 1×1 kernel window to combine all the input channels into a one-channel output.

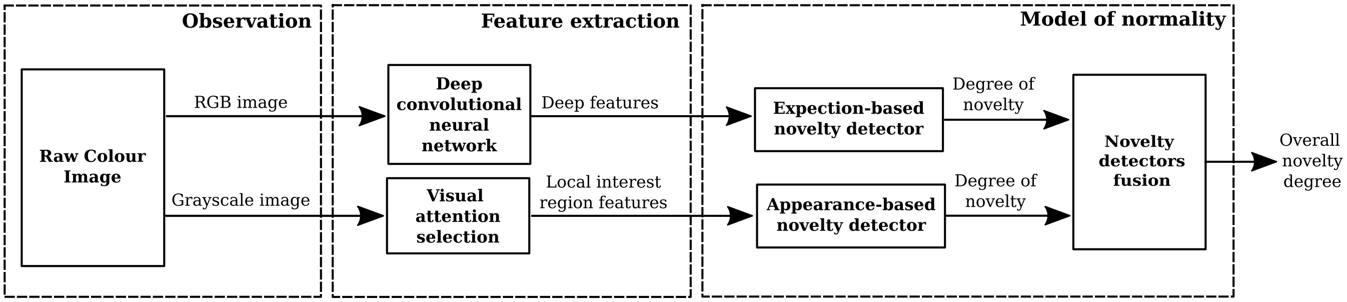


FIGURE 2. Proposed online multichannel novelty detection system.

To improve the performance of the first version of MobileNet, a bottleneck residual block was introduced with the MobileNetV2 model. In this block, three convolution operations are performed, where the first convolution is used 1×1 kernel to widen the number of input channels, which is also known as the expansion layer, and a depthwise convolution with a 3×3 kernel is performed to filter the channels. At the final convolution, 1×1 kernel is used once again to reduce the number of channels which are obtained from the depthwise convolution, which is called the projection layer. This type of layer is also known as a bottleneck layer, when the amount of data flowing through the network is reduced. By contrast, the predecessor model maintains either the number of output channels of the same or doubled channel size; hence, it has many more parameters than the newer model. It is also important to note that in the first two convolutional layers, instead of using the standard rectified linear unit (ReLU) activation function, the ReLU6 activation function is used to clamp the sum of the weighted input values to a maximum value of six, that is, $\min(\max(x, 0), 6)$. However, the projection layer does not have an activation function (*i.e.* linear activation) that prevents the loss of relevant information from the image by using a nonlinear activation function [21].

Furthermore, the residual connection for the bottleneck block of the MobileNetV2 model was implemented, and a copy of the bottleneck block's input was connected to the output of the same block. Thus, the network is less prone to vanishing gradient issues [24]. A residual connection is added whenever the input and output channel sizes of the bottleneck block are the same. Figure 3 presents an overview of the MobileNetV2 architecture used for deep feature extraction. The final convolutional layer produces $8\times10\times1280$-dimensional matrix, and the 2D global average pooling operation is then applied to the resultant matrix to obtain a feature vector that contains 1,280 features. Global average pooling averages the outputs of the convolution layer according to the channel. After obtaining the deep features, they were presented to the expectation-based novelty filter.

B. VISUAL ATTENTION SELECTION

One of the issues in using the entire colour image in any learning system is that the input image carries a large amount of data, and most of them come from a uniform colour pattern, such as the floor or wall of the environment, which can dominate the entire feature vector presented to the system. These small regions may be that are interesting and novel. However, using all data from the input image can not capture the features of these small novel regions. Hence, the visual attention selection mechanism plays an important role in determining the regions of interest within the captured image for the learning system to pay attention to the selected region on the corresponding image. A local feature extraction technique known as speeded up robust features (SURF) [25], [26] was used to obtain the feature vectors of the regions of the interest on the received image from the robot's environment. The extracted features were scale-, rotation-, brightness-, and contrast-invariant [27]. Therefore, similar features are obtained for the same objects which are observed at different distances, angles, and light angles by the robot while moving in the environment. This algorithm was inspired by the scale-invariant feature transform (SIFT) [28], and SURF is much faster than SIFT [29], [30] which is very important when working with large image data on the robot in real-time. The SURF algorithm computes an integral image [31] for the input grayscale image, which increases the performance of computing convolution with box filters. The integral image I_Σ is the sum of the intensity values above and the corresponding pixel coordinates (x, y), and is formulated as in (1). Essentially, the integral image is computed rapidly using both the calculated neighbourhood values of the required pixel coordinates (x, y) on the integral image and the associative intensity value of the original image $I(x, y)$ as given in (2).

$$I_\Sigma(x, y) = \sum_{i=0}^{i\leq x} \sum_{j=0}^{j\leq y} I(x, y) \tag{1}$$

$$\begin{aligned} I_\Sigma(x, y) &= I(x, y) + I_\Sigma(x, y-1) \\ &\quad + I_\Sigma(x-1, y) - I_\Sigma(x-1, y-1) \end{aligned} \tag{2}$$

SURF uses a Hessian matrix to detect the point of interest within the input image, where the Hessian matrix detects the local curvature by calculating second-order partial derivatives. The Hessian matrix can be calculated for a given point $\mathbf{p} = (x, y)$ in an image with scale σ:

$$\mathbf{H}(\mathbf{p}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{p}, \sigma) & L_{xy}(\mathbf{p}, \sigma) \\ L_{xy}(\mathbf{p}, \sigma) & L_{yy}(\mathbf{p}, \sigma) \end{bmatrix} \tag{3}$$

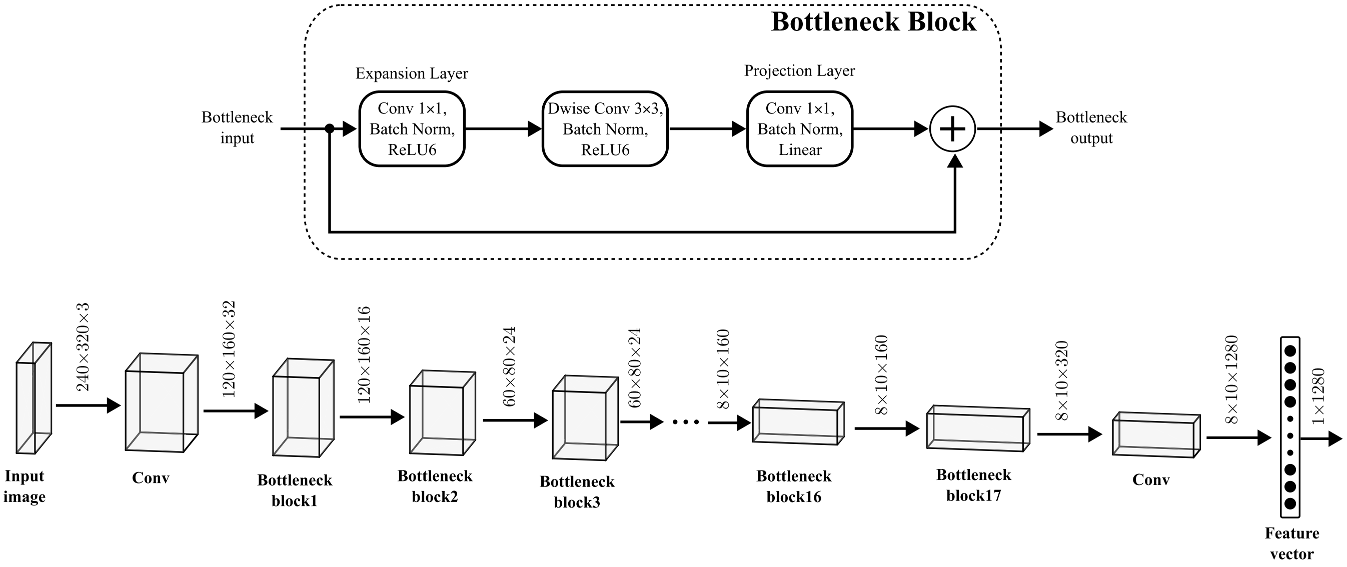


FIGURE 3. An overview of the MobileNetV2 model used for feature extraction is shown at the bottom of the figure. The top image illustrates the sequence of convolutional layers in the bottleneck block.

where $L_{xx}(\mathbf{p}, \sigma)$ denotes the convolution of the second-order Gaussian derivative $\frac{\partial^2}{\partial x^2} g(\mathbf{p}, \sigma)$ at point $\mathbf{p}$, σ denotes the width of the Gaussian, and the function $g(\cdot)$ is defined as follows:

$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{4}$$

The points of interest in the image are extracted by finding the local maxima in both the space and scale images by computing the determinant of the Hessian matrix. To calculate the four entries of the Hessian matrix, image $I(x, y)$ is first convolved with a Gaussian kernel, and then the image is convolved with the second-order derivative of the Gaussian. This operation was computationally expensive. Instead, Hessian values can be approximated using box filters and can be computed at a very low cost using an integral image. Therefore, 9×9 box filter with $\sigma = 1.2$ provides a very good approximation of Hessian matrix values. The determinant of the Hessian is approximated [32] as given in (5):

$$det(\mathbf{H}_{\text{approx}}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \tag{5}$$

where D_{xx}, D_{yy} and D_{xy} are Hessian value approximations in x, y and xy directions, respectively.

To detect scale-invariant features, the determinant of the Hessian matrix was computed for different scales of the input image. This is typically performed by implementing image pyramids, where the image size is repeatedly reduced, followed by the application of a Gaussian filter to each subsampled image. However, in SURF, instead of reducing the size of the image, the dimensions of the box filter are increased, and the width of the Gaussian (σ) is increased based on the changing ratio of the filter size. From the algorithm point of view, these operations can also be implemented in parallel with the use of a GPU, thus yielding faster computation. Once all determinant values of the up-scaled region in the image are computed, a non-maximum suppression algorithm is applied to localise the point of interest in the image over the scaled image. To detect the rotation-invariant feature of the detected interest point, SURF uses Haar-wavelet responses in the horizontal and vertical directions with a circular neighbourhood of radius 6σ where σ is the value at which the interest point is detected when the box filter is upscaled. The orientation of the detected interest point is then computed by determining finding the largest sum of the horizontal and vertical wavelet responses which is computed for each 60° angle around the interest point, thereby yielding the orientation vector of the corresponding point. Finally, the feature vector was computed using the detected interest points and their corresponding orientations. For each detected point, a square window of size 20σ was constructed around the point, and the orientation of this window was adjusted based on the acquired orientation vector for the associated point. Then, this window is divided into 4×4 subregions. For each subregion, the Haar wavelet responses in the horizontal and vertical directions for the 25 sample points were calculated. The sum of the horizontal, vertical, and absolute values in both directions constitutes the 4-element feature vector $\mathbf{v}_i$ for the subregion i as given in (6). Eventually, from 4×4 subregions, $4\times4\times4 = 64$ elements were obtained as a feature vector $\mathbf{s}_j = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{16}]$ for the jth interest point.

$$\mathbf{v}_i = \left[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right] \tag{6}$$

where d_x, d_y indicate the wavelet responses in horizontal and vertical direction.

C. MODEL OF NORMALITY

The model of normality consists of simultaneous training expectation-based and appearance-based novelty detectors. This mechanism is described as follows.

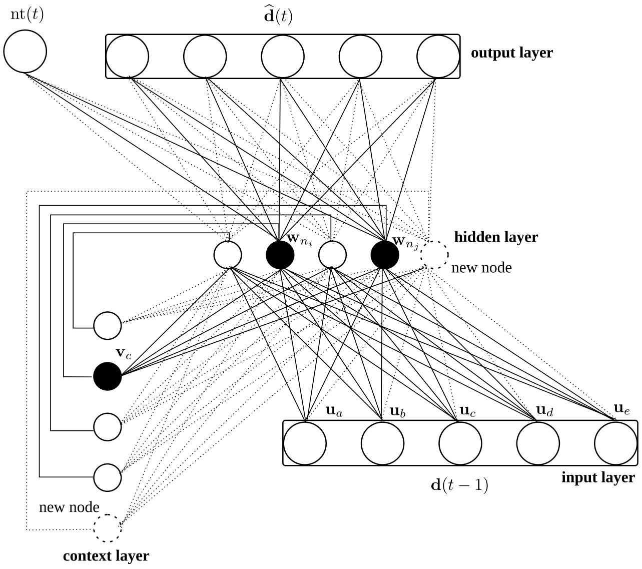


FIGURE 4. Graphical representation of EFuNN architecture. The filled circles at the hidden layer indicates the nodes which are propagated to compute the outputs of the network. The filled circle at the context layer is the node which has the highest activation value from the previous time-step. Solid lines indicate the active connections.

1) EXPECTATION-BASED NOVELTY DETECTOR

The expectation-based novelty detection algorithm was described in [17]. This novelty detector uses a modified version of the evolving fuzzy neural network (EFuNN) inspired by Kasabov [33], [34]. Figure 4 shows the architecture of the EFuNN, which consists of input, hidden, context and output layers. This network can acquire a temporal model between past observed and current input values through online and incremental learning, where the model predicts the expected input values for the current time step. The network dynamically changes its structure by adding or removing the nodes (*i.e.* learned knowledge) from the hidden layer and its corresponding node from the context layer. Initially, the network does not contain any node (*i.e.* no knowledge has been learned). Whenever the network receives a novel input, a new node is added to the hidden layer and a node at the context layer is also created to store the previous activation of the node which is created for the hidden layer. The newly created node weights are initialised with the received previous time-step inputs, and the weights from the hidden layer to the output layer are set as the current actual input data. The network prediction is computed using only the highly activated nodes from the hidden layer, as given in (7).

$$\widehat{d_i}(t) = \sum_{j=1}^{|\mathbf{n}|} \frac{a_{n_j}}{\sum_{k=1}^{|\mathbf{n}|} a_{n_k}} \cdot w_{i,n_j} \tag{7}$$

where $\mathbf{n} = \{n_1, n_2, \ldots\}$ is the index set of the highly activated nodes and a_{n_j} indicates the activation value of the hidden node n_j. The activation of a hidden node $a_i(t)$ can be computed as given in (8). After obtaining the activations of the hidden layer, the maximum activated node $b = \text{argmax}_i(a_i(t))$ can be determined.

$$a_i(t) = \min(\max(1 - S_r \cdot \gamma_i + T_r \cdot v_{c,i}, 0), 1) \tag{8}$$

where S_r and T_r are the spatial and temporal ratios, respectively; $v_{c,i}$ is the connection weight from the maximum activated node from the context layer, that is, $c = \text{argmax}_i(a_i(t-1))$, to the hidden node i; and γ_i is the normalised Manhattan distance between the input deep feature vector $\mathbf{d}(t-1)$ and weight vector $\mathbf{u}_i$ of node i as given in (9).

$$\gamma_i = \frac{|\mathbf{d}(t-1) - \mathbf{u}_i|}{|\mathbf{d}(t-1) + \mathbf{u}_i|} \tag{9}$$

The network determines the previous or current input values as novelty when:

1) The maximum activation of the hidden layer a_b is low, indicating that the novelty is based on past observations. Each hidden node has an associated node in the context layer where the copy of the previous activation value of the hidden nodes is stored, and each context node has connection weights ($\mathbf{v}_i$) to all the hidden nodes (see Figure 4). The activation of each hidden node involves spatial and temporal information as given in (8). Thus, although the previous input features ($\mathbf{d}(t-1)$) do match well with the maximum activated hidden node (indicating with a low distance value in equation (9)), if there is no strong context layer connection weight from the previously activated node, this can be occurred when the previous and current maximum hidden node are not fired consecutively before that indicates the sequence of activation values are novelty.
2) The prediction error between the predicted and actual input values was significantly high, indicating that novelty was based on the current input values. The prediction error was computed using the Euclidean distance $\epsilon(t) = ||\mathbf{d}(t) - \widehat{\mathbf{d}}(t)||$ where $\widehat{\mathbf{d}}(t)$ and $\mathbf{d}(t)$ indicate the predicted and actual extracted deep feature vectors, respectively, from the MobileNetV2 model.

The actual extracted deep feature vector $\mathbf{d}(t)$ can be highlighted as novelty whenever the error $\epsilon(t)$ exceeds the learned dynamic novelty threshold nt(t) which is estimated for the current input vector, *i.e.* $\texttt{IF}[\epsilon(t) > \texttt{nt}(t)\ \texttt{or}\ a_b < S_{\texttt{thr}}]$, where S_{thr} is the sensitivity threshold. Dynamic novelty thresholds [35] are the confidence levels of the network predictions which are learned in each distinct perception space in the environment independently by the network in an online manner, along with the network connection weights during normal environment training. As a result, for different perception spaces in the environment, there is a different level of network confidence which has been learned during the normal environment training; therefore, the network produces a local novelty threshold for the corresponding perception

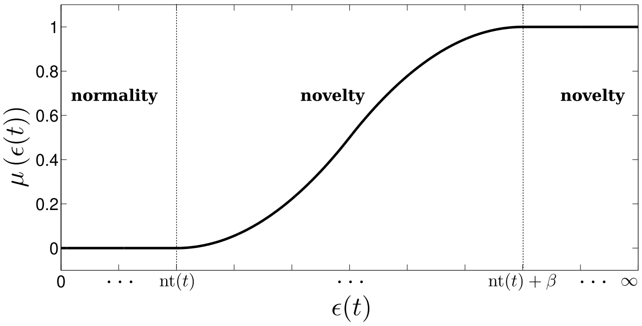


FIGURE 5. An S-shaped membership function (SMF) to calculate the degree of novelty using the prediction error $\epsilon(t)$ and novelty threshold nt(t).

space instead of using one global novelty threshold (this is one of the differences from the regular EFuNN). In fact, the network cannot make good predictions in some regions of the normal environment in comparison with other regions because the input features from these poorly predicted regions may not appear frequently in the normal environment. Consequently, if one global threshold is selected, it is possible to lose novelty detection in regions where the network makes better predictions.

To compute a local novelty threshold for validating the current received input feature vector, the estimated variances σ_i of the hidden nodes highly activated by the previous inputs to the network (note that every created hidden node is also associated with a variance parameter to keep track of the current prediction error whenever these nodes are activated) are used as follows:

$$\mathrm{nt}(t) = k \cdot \sqrt{\sum_{i=1}^{|\mathbf{n}|} \frac{a_{n_i}}{\sum_{j=1}^{|\mathbf{n}|} a_{n_j}} \cdot \sigma_{n_i}^2} \tag{10}$$

where k is the scaling constant which was set to 2 in the following experiments.

The network also removes nodes which are not activated by inputs received from the environment. This functionality works as a forgetting property such that when noisy inputs are added to the network or any inputs are learned previously but are displaced their locations in the environment (*i.e.* becoming a new normal situation), the network waits for a certain time to ensure that the inserted inputs are no longer a frequently seen feature in the environment before the network removes them from its knowledge. A detailed description of the EFuNN-based novelty filtering algorithm can be found in [17].

The computed prediction errors of the modified EFuNN are unnormalised and unbounded values, which makes it impossible to combine these values with other types of novelty detectors. This is because the network does not yield a degree of novelty between 0 and 1 as the output of the network, where the value is close to one indicate strong novelty; otherwise, it is weak. To overcome this, an S-shaped membership function (S-MF), as shown in Figure 5 is added at the top of the network; therefore, the computed unbounded prediction error is constrained to lie within the desired range to indicate the strength of the novelty. Both the prediction error $\epsilon(t)$ and estimated novelty threshold nt(t) of the current input feature vector are used to compute the novelty degree $\mu(\epsilon(t))$ of the network, as given in (11).

$$\begin{cases} 0, & \epsilon(t) < \mathrm{nt}(t) \\ 2\left(\dfrac{\epsilon(t)-\mathrm{nt}(t)}{\tilde{\mathrm{nt}}(t)-\mathrm{nt}(t)}\right)^2, & \mathrm{nt}(t) \le \epsilon(t) \le \dfrac{\mathrm{nt}(t)+\tilde{\mathrm{nt}}(t)}{2} \\ 1-2\left(\dfrac{\epsilon(t)-\tilde{\mathrm{nt}}(t)}{\tilde{\mathrm{nt}}(t)-\mathrm{nt}(t)}\right)^2, & \dfrac{\mathrm{nt}(t)+\tilde{\mathrm{nt}}(t)}{2} \le \epsilon(t) \le \tilde{\mathrm{nt}}(t) \\ 1, & \epsilon(t) > \tilde{\mathrm{nt}}(t) \end{cases} \tag{11}$$

where $\tilde{\mathrm{nt}}(t) = \mathrm{nt}(t)+\beta$, β is a small constant fraction, [nt, $\tilde{\mathrm{nt}}$] indicates the interval of the prediction errors which contain the novelty degree, and the parameter β is set to 0.5 for the following experiments.

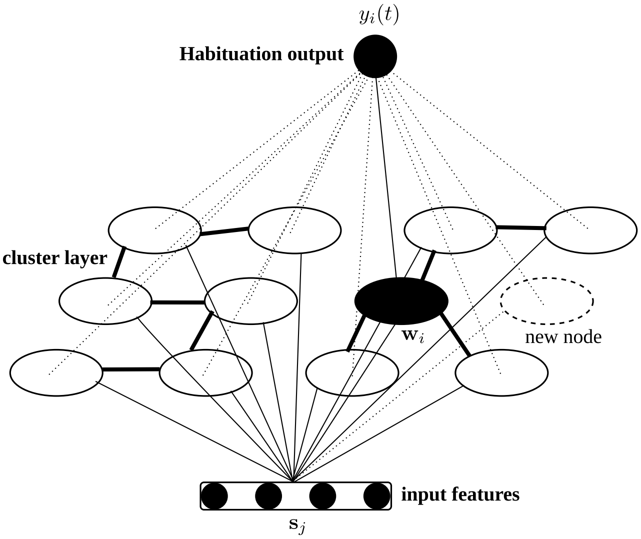


FIGURE 6. Graphical representation of GWR network architecture. The filled circle at the cluster layer indicates the winning node. The activated connections are represented with solid lines. The ticker line between the nodes represent the neighbouring connection.

2) APPEARANCE-BASED NOVELTY DETECTOR

The appearance-based novelty detector learns only the characteristic properties of the input data, and is called unsupervised learning. This type of novelty detection system does not consider the context in which inputs occur. The consideration of novelty depends only on whether the novel input data come from different distributions that the learning system has previously learned. An appearance-based novelty detector, along with an expectation-based novelty detector, is used to learn the local visual input data. Thus, regions of interest in the input image can be examined and learned individually, similar to the work of [14]. A GWR network [3], [5], [9] was used to implement an appearance-based novelty detection approach, as shown in Figure 6. This network also performs an online incremental learning approach like EFuNN, but unlike expectation-based network, it clusters the input data instead of predicting the expected input values. The local features extracted from the SURF detector are learned using a GWR network. As the number of interesting regions extracted from an image varies from image to image, the GWR network is well suited for learning this type of local feature.

The SURF features of an input image are first extracted; hence, a set of feature vectors is acquired as $\mathbf{S}(t) = \{\mathbf{s}_1(t), \mathbf{s}_2(t), \ldots, \mathbf{s}_n(t)\}$ where n denotes the number of feature vectors and is different for each input image. The SURF features for an image are then sequentially presented to the GWR network. Initially, there is no knowledge available about the environment in the GWR network; that is, no nodes are created on the network. The first node is created when the first extracted SURF feature of the input image is presented to the network. When the network receives a new SURF feature, first, the best matching node (*i.e.* the winner node) on the network is found as given in (12):

$$b = \underset{i \in C}{\operatorname{argmin}} ||\mathbf{s}_j(t) - \mathbf{w}_i|| \tag{12}$$

where C indicates the current number of nodes available in the network and $\mathbf{w}_i$ is the weight vector of node i. Subsequently, the activation of the winner node is computed as:

$$a_b = \exp(-||\mathbf{s}_j(t) - \mathbf{w}_i||^2) \tag{13}$$

Each node contains a variable $y_i(t)$ indicating the current strength of habituation which decreases exponentially over time when an input feature matches the corresponding node, as given in the first-order differential equation (14):

$$\tau_i \frac{\partial y_i(t)}{\partial t} = \alpha[y_0 - y_i(t)] - \lambda \tag{14}$$

where τ and α are the time constants that control the habituation and recovery rates, respectively; λ is set to one which indicates that the stimulus is presented; y_0 is the initial value of the habituation, which is set to one; and $y_i(t) = y_0$ initially. The habituation value is in the range of [0,1] and is used to indicate the degree of novelty of the presented input features [10]. The highest novelty degree can be indicated when the habituation value is close to one; otherwise, a value close to zero implies the lowest novelty degree, that is, normality.

Novelty can be detected when the novelty degree (*i.e.* habituation value) of the winner node $y_b(t)$ which is found in (12), is low, and the presented input SURF feature $\mathbf{s}_j$ does not match the winner node sufficiently well, as given by the condition $\texttt{IF}[a_b < a_{\texttt{thr}} \texttt{ and } y_b(t) < h_{\texttt{thr}}]$, where a_{thr} and h_{thr} are the activation and habituation thresholds, respectively. When novelty is detected, the network requires the addition of a new node to represent the novel input feature [9]. By contrast, the weight vector of the winner node and its neighbouring nodes is further adjusted to better fit the input SURF features. Furthermore, counters (*i.e.* age) are created for each neighbouring node which is increased every time for each neighbourhood node of the winning node. The neighbourhood connection is removed whenever the age of the connection exceeds the predefined threshold, and thus any nodes which have no neighbourhood connections left are simply removed from the network. More details regarding the GWR network-learning algorithm can be found in [36].

The local extracted feature vectors from the input image when presented to the GWR network sequentially can lead to localised novelty in the corresponding image. This can be a useful tool to direct the robot in the direction of novel stimuli located on the image to examine the area in more detail during patrolling, which can be developed more effectively by continually seeking to discover new, unseen (novelty) features. This can be observed in Figure 7, where the extracted SURF feature vectors are from the region in which the yellow ball is highlighted as novel by the GWR network. The habituation values from the network for these feature vectors are reported as $y_{b_i} = 1.0$ where b_i indicates the winning node on the GWR network for the corresponding SURF feature vector $\mathbf{s}_{\{i=2,3,6,7,8\}}$, indicating a strong novelty.

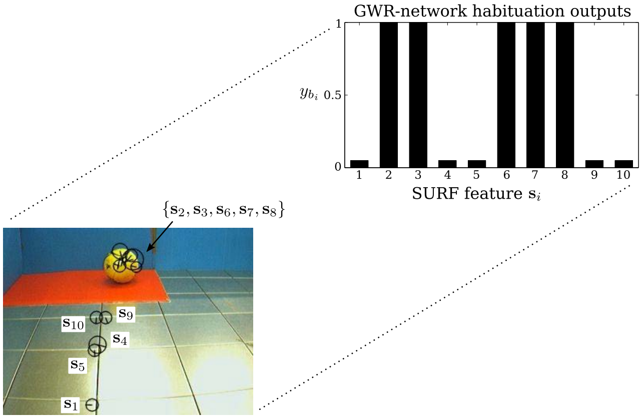


FIGURE 7. The GWR network novelty outputs for 10-SURF feature vectors extracted from the input image.

The overall degree of novelty $\bar{y}(t)$ for the input image can then be computed by averaging all the habituation values obtained for the current set of SURF feature vectors $\mathbf{S}(t)$, as given in (15):

$$\bar{y}(t) = \frac{1}{n} \sum_{i=1}^{n} y_{b_i}(t) \tag{15}$$

where b_i is the index of the winning node in the network for i^{th} SURF feature vector $\mathbf{s}_i$.

3) NOVELTY DETECTORS FUSION

Both novelty filters represent different types of novelty; the expectation-based novelty filter detects context-based novelties, whereas the appearance-based novelty filter highlights novelties depending on the current input features. It is also possible that one filter can detect the novelty situation, but the other cannot; therefore, it is important to combine both novelty filter outputs to obtain the final degree of novelty. The sum of the weighted novelty filter outputs is computed by defining the contribution of each filter, as given in (16). Thus, the weight of each filter is θ_i, where $\sum_i^2 \theta_i = 1$.

Equal weights were set to $\theta_1 = 0.5$ and $\theta_2 = 0.5$ for both filters used in the following experiments.

$$z(t) = \begin{cases} \theta_1 \cdot \mu\left(\epsilon(t)\right) + \theta_2 \cdot \bar{y}(t), & \text{if } \mathbf{S}(t) \neq \{\} \\ \theta_1 \cdot \mu\left(\epsilon(t)\right), & \text{else} \end{cases} \tag{16}$$

where $z(t)$ is a value in the range [0,1]; a zero value indicates that no novelty data is observed; a value close to zero implies there is a small deviation from the normal model; otherwise, a value close to one implies a very high deviation from the normality.

A summary of the multiple-novelty filter fusion and online training steps is provided in Algorithm 1. It is important to note that the modified EFuNN and GWR network algorithms are represented with *trainEfunNet* function at line 4, and *trainGwrNet* at line 8, respectively.

Algorithm 1 Feature Extraction and Multiple Network Normality Model Fusion for Multichannel Novelty Detection

```
Input : Input colour image $\mathbf{I}(t)$, modified EFuNN
        net$_e(t)$ and GWR network net$_g(t)$.
Output: Updated modified EFuNN net$_e(t)$, updated
        GWR network net$_g(t)$, resultant overall novelty
        degree $z(t)$.
1  Obtain the deep features: $\mathbf{d}(t)$=MobileNetV2 ($\mathbf{I}(t)$)
2  Convert colour image to grayscale image:
   $\mathbf{G}(t)$=RGB2GRAY ($\mathbf{I}(t)$)
3  Obtain SURF features: $\mathbf{S}(t)$=detectSurfFeatures
   ($\mathbf{G}(t)$)
4  Train modified EFuNN by presenting current deep
   features $\mathbf{d}(t)$: [net$_e(t)$, $\epsilon(t)$, nt$(t)$]= trainEfunNet
   (net$_e(t)$, $\mathbf{d}(t-1)$, $\mathbf{d}(t)$)
5  Perform S-shape membership function to obtain the
   novelty degree of the modified EFuNN prediction error
   $\epsilon(t)$ using equation (11)
6  Train GWR network by presenting current SURF
   features $\mathbf{S}(t)$:
7  for $\mathbf{s}_k(t)$ in $\mathbf{S}(t)$ do
8      [net$_g(t)$, $y_k(t)$]=trainGwrNet (net$_g(t)$, $\mathbf{s}_k(t)$)
9  end
10 Compute mean habituation value $\bar{y}(t)$ using
   equation (15)
11 Fusing the novelty degrees obtained from the networks
   using equation (16)
```

III. EXPERIMENTAL METHODS

A. THE ROBOT AND SOFTWARE SPECIFICATION

To perform the following experiments, a *Scitos-G5* mobile robot equipped with 24 ultrasonic range finders, an SICK S300 laser rangefinder, and a camera was used. The camera was fixed at the front of the robot, which was connected externally to an Intel Core i7 laptop placed on the robot with an NVIDIA GeForce RTX 2060 6GB computing processor featuring 1920 CUDA cores. During the experiments, all the input images captured by the robot, their corresponding EFuNN, GWR network, multichannel novelty detection outputs, and the robot's global coordinates from the Vicon tracking system were logged to the external laptop for further performance evaluation of the proposed system. All the software for the multichannel novelty filter was implemented by using Python programming language, the Keras framework [22] which is built on top of the TensorFlow, was used to implement MobileNetV2 model, and SURF was implemented using OpenCV library [37] for the following experiments.

B. EXPERIMENTAL ENVIRONMENT

The robot environment, which is used for learning the models of the normal camera images of the robot and then using these models as the novelty detector, is shown in Figure 8. During the acquisition of the network models, it is important that the robot follows a stable route every time it learns the model of the captured images from the current environment, particularly when the purpose of the experiment is to detect novelties in the same environment. Otherwise, the sensory data becomes significantly different from what it has already learned; therefore, the acquired normality network models can easily fail to highlight the novelties in the corresponding environment. In fact, any machine learning applications, when the training and test data come from the different distribution, the generalisation performance of the learned model becomes poor [38]. To overcome this, a left-hand side wall follower controller was implemented using laser rangefinder readings for the robot to move autonomously and stably in the experimental environment. Figure 8 shows the observed landmarks in the environment, such as pillars, posters, red panels, and the sticked green panel on the blue wall which were all perceived during robot exploration. It is also important to note that the reflective floor generates noise in the camera of the robot, especially when the lighting conditions of the environment are changed.

C. EXPERIMENTAL PROCEDURE

In the following experiments, four environments are used to verify the proposed multichannel novelty detection system. These are environments A-1, A-2, A-3, and A-4. Environment A-1 is shown in Figure 8 which is a normal environment in which the networks are trained. The original environment was slightly modified for each novel environment to test the trained network model as a novelty detector. First, a new red panel was introduced into environment A-2, then in environment A-3, a yellow coloured ball was placed into the environment, and in the final environment A-4, the posters were removed from the environment. During the learning of both (EFuNN and GWR network) models of normality, the robot travelled in the environment A-1 in five training laps, where both models on the robot were enabled to learn the input images of the robot from the unchanged environment online by training both models every time an input image was

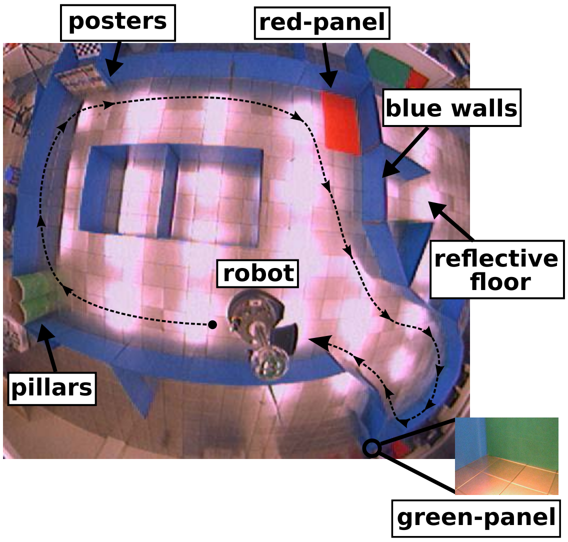


FIGURE 8. Robot environment where models of normality are trained.

captured from the robot's camera. Subsequently, the learning of both network models was disabled after the trained laps, which were then used as a multichannel novelty detector to highlight any abnormal input images from the environments A-1, A-2, A-3, and A-4 in the other five laps.

D. PERFORMANCE EVALUATION METRICS

To evaluate the performance of novelty detection systems, the ground truth of the perceived images must first be identified. To do this, approximately 6900 test images from four environments logged during the test laps were individually analysed, and all the introduced novel objects within the images were manually marked as novel. Here, only controlled novelties were identified, which were modified in the initial environment A-1 by the experimenter. It is possible that while a robot travels in the environment, it might perceive different regions of the environment because of deviations in its training route. These novelties cannot be marked manually and were not considered to confirm the validity of the novelty detection system proposed in this study. To evaluate the novelty detection system, a binary 2×2 confusion matrix was constructed for each evaluated system, as listed in Table 1. Then, the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) were computed. Once the corresponding matrix is filled, the accuracy (ACC), Matthews correlation coefficient (MCC), and F_1 score are computed for each novelty detection system. The most important statistical metric considered for the performance of the systems is MCC according to [40] and [41]. It is important to emphasise that the value of the evaluation metrics becomes one, implying a perfect outcome. The evaluation metrics were defined in Equations (17) to (19).

$$\mathrm{ACC} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}} \tag{17}$$

$$\mathrm{MCC} = \frac{\mathrm{TP}\cdot\mathrm{TN}-\mathrm{FP}\cdot\mathrm{FN}}{\sqrt{(\mathrm{TP}{+}\mathrm{FP})\cdot(\mathrm{TP}{+}\mathrm{FN})\cdot(\mathrm{TN}{+}\mathrm{FP})\cdot(\mathrm{TN}{+}\mathrm{FN})}} \tag{18}$$

$$F_1 = \frac{2\cdot\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}} \tag{19}$$

TABLE 1. Sample 2×2 confusion matrix for the evaluation of novelty detection systems.

	Novelty detected	Novelty not detected
Novelty present	True positive (TP)	False negative (FN)
Novelty not present	False positive (FP)	True negative (TN)

The training parameters of the modified EFuNN and GWR network used in subsequent experiments are listed in Table 2. Detailed descriptions of these parameters are provided in [17] and [36].

IV. EXPERIMENTAL RESULTS

A. ENVIRONMENT A-1

Initially, the robot had no knowledge of its environment. The learning of both the modified EFuNN and GWR network is enabled to learn the models of the normal images received by the robot from the environment. After the robot completes the first training lap, the learning is disabled, and the acquired network models are used in the same environment as a multichannel novelty filter to detect any difference in the deep features and SURF features from what it has been learned by the models in the first training lap. As can be seen clearly in Figure 9a with red-coloured indicators on the plot, the robot highlights many abnormalities in the area where the novel landmarks are perceived, such as when receiving images from the red panel, green panel, pillars, and posters, these can be seen with the raised novelty indicators. The high-magnitude novelty values decrease in the second training lap especially for location $(x, y)\approx(0, 2)$ m. This location is the starting location of the robot, and the network models have only been trained with only one lap in the environment. Therefore, the input features for continuous transition when the robot passes through the starting location have not been sufficiently learned, which is why this location is continuously highlighted with high novelty values in Figure 9a. However, after the second training lap is completed, the novelty status when approaching the end of one lap is completely learned, as shown clearly in Figure 9b where the combined novelty degrees of the models are approximately zero (*i.e.* no novelty). Eventually, the networks are trained more in the unchanged environment; they become more expert and reliable to predict (with modified EFuNN) or cluster (with GWR network) normal sensory data which is later used as a novelty detection system. It is also important to note that there are very low novelty values even after the 5th training lap (see Figure 9e), because the robot does not always follow

TABLE 2. Training parameters for modified EFuNN and GWR network.

Modified EFuNN		**GWR-Network**	
Parameter	*Value*	*Parameter*	*Value*
Activation threshold (A_{thr})	0.85	Activation threshold (a_{thr})	0.5
Proportionality factor (ρ)	0.1	Proportionality Factor (η)	0.1
Habituation rate (τ)	4	Habituation rate (τ)	3.33
Recovery rate (α)	1	Recovery rate (α)	1.05
Node's age threshold (old)	1000	Node's age threshold (age_{max})	20
Spatial ratio (S_r)	0.8	Habituation threshold (h_{thr})	0.1
Temporal ratio (T_r)	0.2	Learning rate (ϵ)	0.3
Pruning ratio (P_r)	0.001		
Sensitivity threshold (S_{thr})	0.9		
Learning rate for parameter $\mathbf{u}$ (η_1)	0.25		
Learning rate for parameter $\mathbf{w}$ (η_2)	0.25		
Learning rate for parameter $\mathbf{v}$ (η_3)	0.01		
Learning rate for parameter $\boldsymbol{\sigma}$ (η_4)	0.8		

TABLE 3. Novelty detection performance of the GWR network, modified EFuNN, and multichannel novelty filter in each experimental environment. Each novelty filter was tested in each environment using the colour images perceived by the robot in five test laps. Each environment contained approximately 1723 images received during the five test laps.

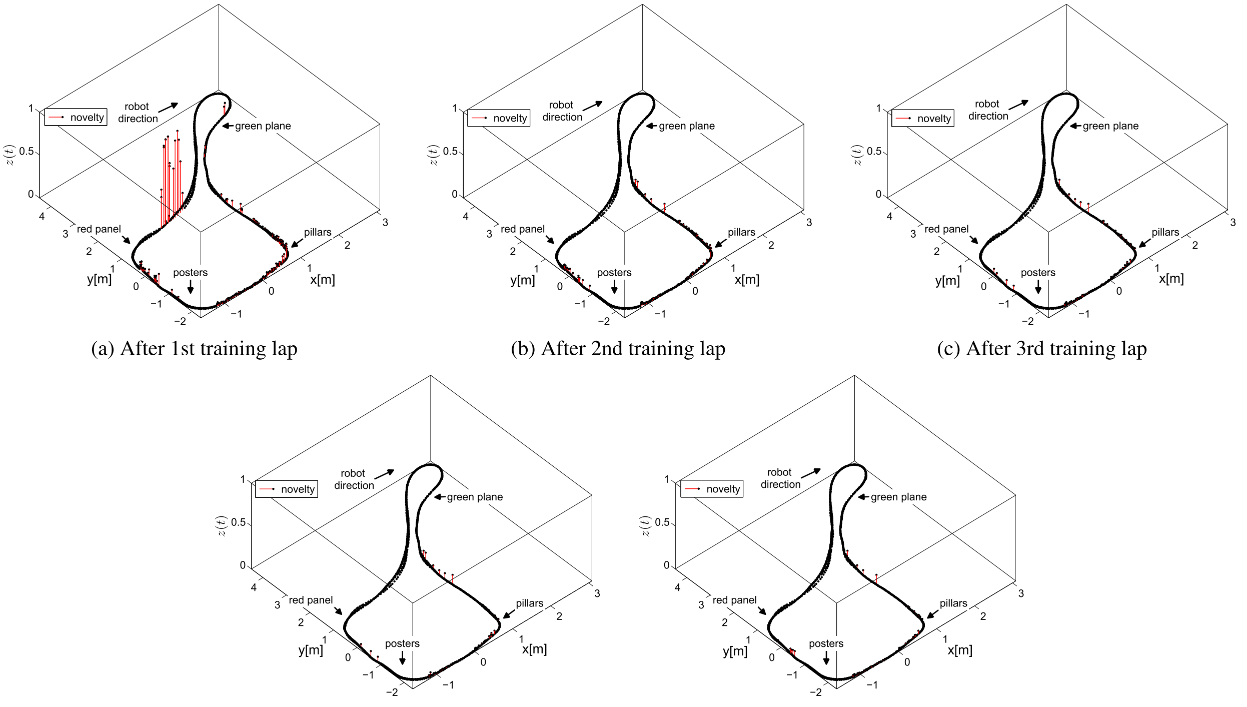
	Environment A-2			**Environment A-3**			**Environment A-4**		
	ACC	MCC	F_1	ACC	MCC	F_1	ACC	MCC	F_1
GWR Network	0.97	0.52	0.48	0.99	0.95	0.96	0.91	-0.03	n/a
Modified EFuNN	0.99	0.91	0.91	0.90	0.42	0.35	0.99	0.93	0.93
Multichannel System	0.99	0.94	0.95	0.99	0.97	0.97	0.99	0.93	0.93

its route perfectly, and there are some deviations in the orientation of the robot's heading, which causes the robot to receive slightly different sensory data from the environment. To overcome this problem, it is necessary to run the robot further in an environment to reduce false detections.

B. ENVIRONMENT A-2

Another red panel was introduced into the environment which was placed on the floor immediately before the learned pillars were perceived. The red panel is not a completely new object because another red panel has already been learned. In other words, the newly introduced red panel is not a novelty based on its appearance, but it is novelty based on the context in which the panel is perceived within the incorrect region. The modified EFuNN learns the sequence of deep features, where the acquired model maps the inputs of the past extracted deep features to the current expected deep features. This is done by having the context layer on the network architecture that behaves as short-term memory. Therefore, the network easily highlights sequence-based abnormalities even if the appearance of the object is not novel. However, the SURF detector also produces some relevant features to the GWR network, even though the red panel has a uniform colour surface which is not very interesting to the SURF detector. This is also shown in Figure 10e where the relevant features of the red panel are extracted after the novel panel is observed over 33 time steps. Nevertheless, both the novelty filters highlight the novel red panel separately, as shown in Figures 11a and 11b. After the novelty degrees of both filters are fused, the overall novelty degree for the final novelty decision is as shown in Figure 11c.

Similar results are also reported statistically in Table 3 where a smaller number of relevant SURF features causes the GWR network novelty filter to not strongly highlight the novel red panel. Therefore, the contribution of this filter was less than that of the multichannel novelty detector. It can be seen that the MCC and F_1 score measurements for the GWR network were much lower than those calculated for the modified EFuNN. However, the fused novelty degrees from both novelty filters improve the novelty detection performance in environment A-2 which is also clearly shown in the multichannel novelty detector's MCC and F_1 score measurements which are higher than those of both novelty filters separately.



(a) After 1st training lap (b) After 2nd training lap (c) After 3rd training lap

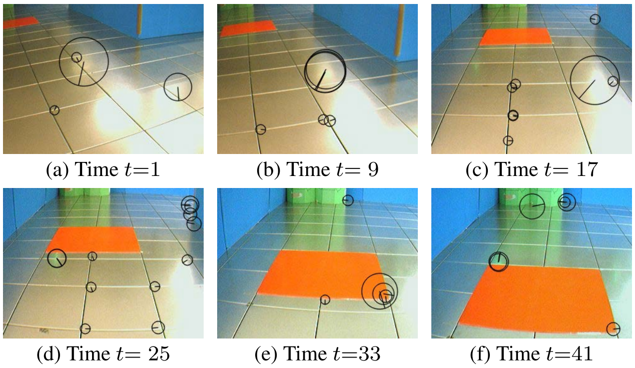
(d) After 4th training lap (e) After 5th training lap

FIGURE 9. Learning progress in the environment A-1. The acquired models were tested five laps after each training lap.

C. ENVIRONMENT A-3

A novel yellow ball was introduced into the environment in which it was placed on the learned red panel. The colour distribution of the ball has never been observed before while learning the initial environment A-1; therefore, it is expected to be easily highlighted as novel by both the novelty filters. Figure 12 shows the extracted SURF features when the robot perceived the ball. It can also be observed that the SURF algorithm produces intense and strong features when there is a colour riot on the objects. Even in the first instance, the ball is seen in the input image (see Figure 12a) and the SURF detector yields relevant and object-focused features to the corresponding novelty filter.

In contrast, the lower convolutional layers of the MobileNetV2 network detect low-level features, such as edges in the input image, such that it retains almost all the information of the raw input image. Each filter of the lower convolutional layers presents different edge-detection filters which were previously learned from the ImageNet database. However, while going to the deeper layers of the network, the activations of these layers become abstract, carrying less information but extracting more relevant features about the recognised objects within the input image. Therefore, the sparsity of the activations in the final convolutional layer is higher than that in the initial convolutional layers. Figure 13 shows the activation of various filters from selected layers on MobileNetV2 when the ball was observed for the first time (*i.e.* the raw version of Figure 12a, which has no SURF points). As can be seen clearly, the first convolutional layer (see Figure 13a) yields dense activation outputs which are indicated in yellow (*i.e.* higher activations in viridis colour maps), Figure 13b shows the activations of the filter in bottleneck block 5, and Figure 13c shows the activations of the final deepest convolutional layer where some filters are not activated at all by showing with blue colour because those filters present different object features which are not found in the input image. The relevant ball features are also obtained from the deep network along with the features of other recognised objects, which were extracted from the filters of the deepest layer. For example, some convolutional filters yield strong activation values for the detected ball, which are shown with yellow-coloured activation values at the top and centre of each filter (this is the location of the ball in the input image) in Figure 13c. These filter outputs are combined without



(a) Time t=1 (b) Time t= 9 (c) Time t= 17

(d) Time t= 25 (e) Time t=33 (f) Time t=41

FIGURE 10. SURF features are extracted in environment A-2 while perceiving a novel red panel. For (a) 4-SURF, (b) 6-SURF, (c) 8-SURF, (d) 12-SURF, (e) 5-SURF, and (f) 6-SURF are extracted, respectively.

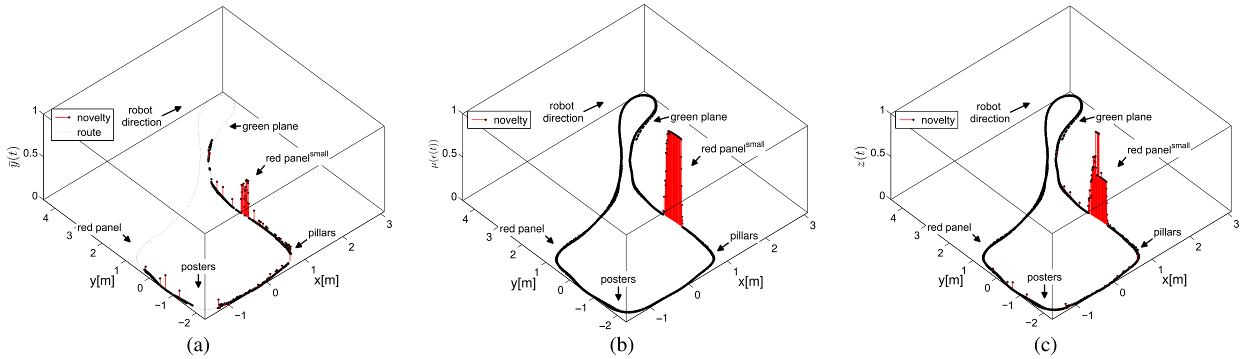
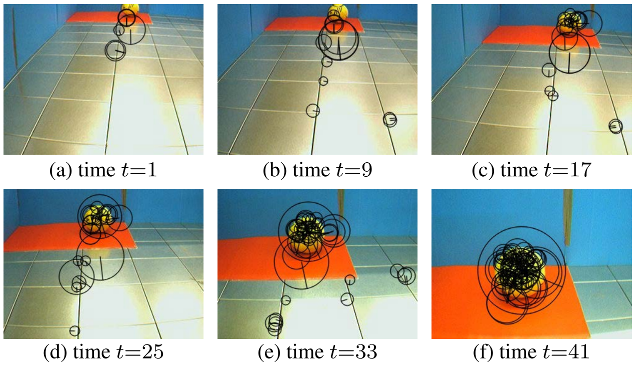


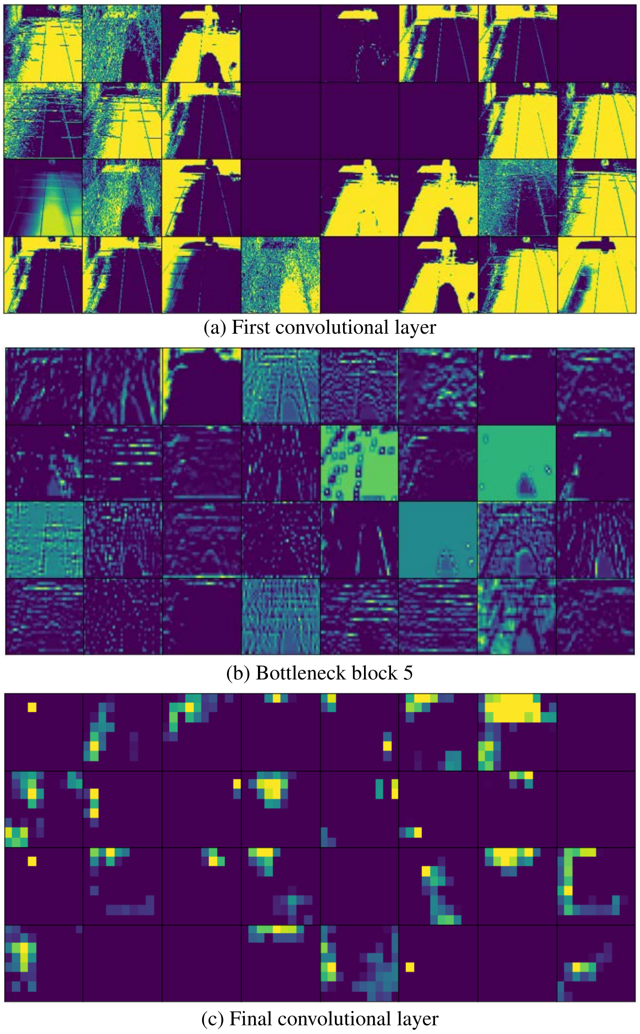
FIGURE 11. Novelty detection in environment A-2. (a) GWR network novelty detection, (b) modified EFuNN novelty detection, and (c) multichannel novelty detection.


(a) time t=1 (b) time t=9 (c) time t=17

(d) time t=25 (e) time t=33 (f) time t=41

FIGURE 12. SURF features are extracted in environment A-3 while perceiving a novel yellow ball. For (a) 6-SURF, (b) 12-SURF, (c) 19-SURF, (d) 22-SURF, (e) 44-SURF, and (f) 61-SURF are extracted, respectively.

extracting only the features of the novel object, and are then presented to the expectation-based novelty detection model. Therefore, this action spreads the impact of all the recognised object features when presented to the model. This is the desired behaviour because all objects must be identified, and the novelty is unknown. Consequently, the expectation-based novelty filter cannot highlight the novel ball when observed for the first time (see Figure 14b). In fact, the prediction errors from the modified EFuNN slightly increase at the first time ball is seen, but from the far detecting small novel ball among other recognised objects does not dominate the feature vector to increase the prediction errors dramatically. Conversely, the ball was detected as novel for the first time, as shown by the GWR network novelty filter in Figures 14a. This issue is also reported in the statistical assessments with MCC = 0.42 and $F_1 = 0.35$ in Table 3 where expectation-based novelty detection missed many novelty indications at the beginning when the ball was being perceived. However, when the robot approaches the near of the ball, the field of view becomes narrow, so more ball-oriented features are obtained, which is why the deviation from the expectation model becomes higher to identify the novelties. Eventually, both models demonstrated strong and confident novelty indications. The final fused novelty degrees are illustrated in Figure 14c.

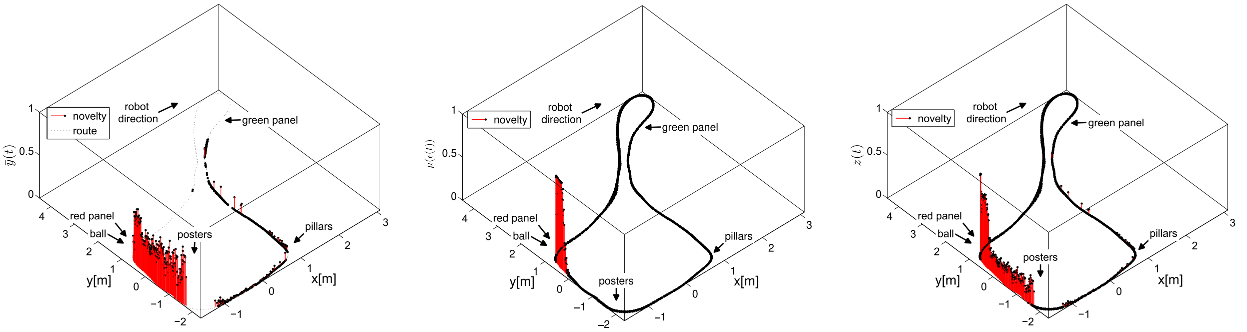

(a) First convolutional layer

(b) Bottleneck block 5

(c) Final convolutional layer

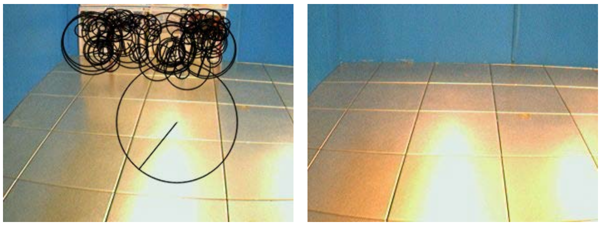
FIGURE 13. Activations of selected convolutional layers on MobileNetV2 feature extraction network. Each square image represents the activation of a corresponding filter. Note that bottleneck block 5 and the final convolutional layer have 192 and 1280 filters, respectively; however, for visualisation purposes, every sixth and fortieth filter is illustrated.

After both novelty filters are fused, the multichannel novelty detection system performs almost perfectly in environment A-3, as reported with MCC = 0.97 and $F_1 = 0.97$.



(a) (b) (c)

FIGURE 14. Novelty detection in environment A-3. (a) GWR network novelty detection, (b) modified EFuNN novelty detection, and (c) multichannel novelty detection.

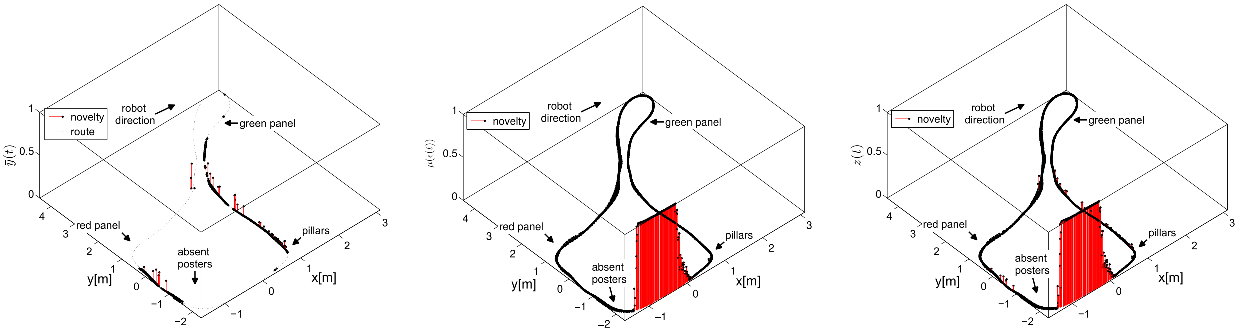


(a) (b)

FIGURE 15. Extracted SURF features for (a) posters and (b) absence of posters in environment A-4.

D. ENVIRONMENT A-4

Posters were removed from the final experimental environment A-4. If only an appearance-based novelty detector is used, the removed posters will not be detected as novel in the environment. This is a sequence-based novelty; it requires a model that represents the temporal relationship between the inputs and outputs of the novelty detection system. To predict the posters on the wall, the system must first perceive the sequence of known objects on its route before predicting the forthcoming object based on observed evidence. Another issue is that after removing the posters, the uniformly coloured blue box is not found to be interesting; therefore, the SURF detector does not yield any features to feed the GWR network novelty filter. This problem is illustrated in Figure 15. Consequently, the corresponding novelty filter becomes inactive when no features are provided by the environment, as shown in Figure 16a. Statistical analysis also proves the poor performance of the GWR network novelty filter when using SURF features, as reported in Table 3 as MCC $= -0.03$; that is, it is not better than random prediction. In contrast, novelty indications arise from the modified EFuNN when missing posters are perceived, as shown in Figure 16b. The expectation-based model predicts the expected posters on the wall, but the actual (*i.e.* missing posters) does not match the predicted one. MobileNetV2 yields one large feature vector for the entire input image, and some features of this vector have higher activation values that indicate more relevant information on the specific part of the input image, and are all eventually presented to the modified EFuNN. By using the features computed from the entire image, there is no chance of losing any information, unlike SURF feature detection. After both novelty filters are fused, the overall degree of novelty in the environment is shown in Figure 16c and all contributions to the multichannel novelty detection system are provided by the modified EFuNN novelty filter, as statistically revealed in Table 3 with the values of MCC $= 0.93$ and $F_1 = 0.93$.



(a) (b) (c)

FIGURE 16. Novelty detection in environment A-4. (a) GWR network novelty detection, (b) modified EFuNN novelty detection, and (c) multichannel novelty detection.

V. CONCLUSION

In this study, a multichannel novelty detection system is proposed that combines multiple different purposed models of normality. This system consists of two novelty filters: the modified EFuNN and GWR network, which are used to learn the colour images received from the robot. Each novelty filter has different characteristics for learning the data: the modified EFuNN learns the temporal relationship between the inputs and future values of the forthcoming inputs, and then predicts the expected input values to highlight any novel situation whenever the prediction error exceeds the learned local novelty threshold; however, the GWR network clusters the data topological, and then any input data matches with low activation output with one of the cluster nodes on the network is detected as novel. To provide robust novelty detection, each filter focuses on different extracted features within the input image, such that the networks learn better to distinguish those features from the novel class. Therefore, they have a lower generalisation error for the corresponding features, which is called a feature-specific model of normality, such as the subsumption architecture developed by [19] for behaviour-based controllers. Eventually, the modified EFuNN is trained using deep features extracted from the pretrained MobileNetV2 deep network model, which provides high-level features of the input image instead of using all the information in the image, and an abstract representation of the image is learned. By contrast, the GWR network focuses only on the regions of interest in the input image for attention selection and learns the features extracted from these regions. Thus, the network focuses on small details instead of losing them when extracting the global features from an input image. This is because global feature extraction can eliminate these small features in an image by considering them noise. To extract fast and reliable invariant features from the regions of interest within the input image, the SURF local feature extraction technique was used to generate the input features. To verify the proposed novelty detection system, three novel objects are introduced into the training environment. The outputs of both novelty filters (modified EFuNN and GWR network), as well as the merged outputs in the multichannel novelty filter, were all visualised side by side to identify their weaknesses and robustness. Consequently, when normal (learned) data are removed from the trained environment or uniformly coloured objects are introduced into the environment, the SURF detector fails to yield features relevant to the GWR network novelty filter. However, this weakness is addressed by the modified EFuNN novelty filter because it learns the temporal relationship of the inputs and uses all the activation outputs from the filters of the final convolutional layer in the MobileNetV2 model. However, the use of all activation outputs carries all recognised object information, and the response of the novelty filter can be slower than when only local SURF features are presented to the novelty filter. Although there were very small deviations from the typical input data, the GWR network with the attention selection mechanism highlighted these abnormalities, whereas the modified EFuNN detected more evident and novel features. A multichannel novelty filter can be used to obtain more robust and reliable novelty decisions.

REFERENCES

[1] P. Hayton, S. Utete, D. King, S. King, P. Anuzis, and L. Tarassenko, "Static and dynamic novelty detection methods for jet engine health monitoring," *Phil. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 365, no. 1851, pp. 493–514, Feb. 2007.

[2] L. Tarassenko, "Novelty detection for the identification of masses in mammograms," in *Proc. 4th Int. Conf. Artif. Neural Netw.*, 1995, pp. 442–447.

[3] S. Marsland, U. Nehmzow, and J. Shapiro, "On-line novelty detection for autonomous mobile robots," *Robot. Auto. Syst.*, vol. 51, nos. 2–3, pp. 191–206, May 2005.

[4] H. V. Neto and U. Nehmzow, "Automated exploration and inspection: Comparing two visual novelty detectors," *Int. J. Adv. Robot. Syst.*, vol. 2, no. 4, pp. 355–362, 2005.

[5] S. Marsland, "Novelty detection in learning systems," *Neural Comput. Surv.*, vol. 3, pp. 157–195, Jan. 2003.

[6] M. A. F. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," *Signal Process.*, vol. 99, pp. 215–249, Jun. 2014.

[7] M. Markou and S. Singh, "Novelty detection: A review—Part 1: Statistical approaches," *Signal Process.*, vol. 83, no. 12, pp. 2481–2497, 2003.

[8] M. Markou and S. Singh, "Novelty detection: A review—Part 2: Neural network based approaches," *Signal Process.*, vol. 83, no. 12, pp. 2499–2521, 2003.

[9] S. Marsland, J. Shapiro, and U. Nehmzow, "A self-organising network that grows when required," *Neural Netw.*, vol. 15, nos. 8–9, pp. 1041–1058, 2002.

[10] S. Marsland, "Using habituation in machine learning," *Neurobiol. Learn. Memory*, vol. 92, no. 2, pp. 260–266, Sep. 2009.

[11] S. Marsland, U. Nehmzow, and J. Shapiro, "Detecting novel features of an environment using habituation," in *Proc. Simulation Adapt. Behav.*, 2000, pp. 1–10.

[12] Y. Gatsoulis and T. M. McGinnity, "Intrinsically motivated learning systems based on biologically-inspired novelty detection," *Robot. Auto. Syst.*, vol. 68, pp. 12–20, Jun. 2015.

[13] H. Vieira Neto and U. Nehmzow, "Visual novelty detection with automatic scale selection," *Robot. Auto. Syst.*, vol. 55, no. 9, pp. 693–701, Sep. 2007.

[14] H. V. Neto and U. Nehmzow, "Real-time automated visual inspection using mobile robots," *J. Intell. Robot. Syst.*, vol. 49, no. 3, pp. 293–307, 2007.

[15] M. Artac, M. Jogan, and A. Leonardis, "Incremental PCA for on-line visual learning and recognition," in *Proc. Int. Conf. Pattern Recognit.*, vol. 3, Aug. 2002, pp. 781–784.

[16] M. A. Contreras-Cruz, J. P. Ramirez-Paredes, U. H. Hernandez-Belmonte, and V. Ayala-Ramirez, "Vision-based novelty detection using deep features and evolved novelty filters for specific robotic exploration and inspection tasks," *Sensors*, vol. 19, no. 13, p. 2965, Jul. 2019.

[17] E. Özbilge, "On-line expectation-based novelty detection for mobile robots," *Robot. Auto. Syst.*, vol. 81, pp. 33–47, Jul. 2016.

[18] E. Özbilge, "Experiments in online expectation-based novelty-detection using 3D shape and colour perceptions for mobile robot inspection," *Robot. Auto. Syst.*, vol. 117, pp. 68–79, Jul. 2019.

[19] R. Brooks, "A robust layered control system for a mobile robot," *IEEE J. Robot. Autom.*, vol. RA-2, no. 1, pp. 14–23, Mar. 1986.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[21] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[22] (2022). *Keras Applications*. Accessed: Sep. 16, 2022. [Online]. Available: https://keras.io/api/applications/

[23] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[25] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, ''Speeded-up robust features (SURF),'' *Comput. Vis. Image Understand.*, vol. 110, no. 3, pp. 346–359, 2008.

[26] H. Bay, ''From wide-baseline point and line correspondences to 3D,'' Ph.D. dissertation, Dept. Inf. Technol. Elect. Eng., Swiss Federal Inst. Technol., ETH Zürich, Zürich, Switzerland, 2009.

[27] C. Evans, ''Notes on the opensurf library,'' Univ. Bristol, Bristol, U.K., Tech. Rep., CSTR-09-001, Jan. 2009.

[28] D. G. Lowe, ''Distinctive image features from scale-invariant keypoints,'' *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2003.

[29] K. Mikolajczyk and C. Schmid, ''A performance evaluation of local descriptors,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[30] P. Drews, R. de Bem, and A. de Melo, ''Analyzing and exploring feature detectors in images,'' in *Proc. 9th IEEE Int. Conf. Ind. Informat.*, Jul. 2011, pp. 305–310.

[31] P. Viola and M. Jones, ''Rapid object detection using a boosted cascade of simple features,'' in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2001, pp. 1–11.

[32] H. Bay, T. Tuytelaars, and L. V. Gool, ''SURF: Speeded up robust features,'' in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany: Springer, 2006, pp. 404–417.

[33] N. K. Kasabov, *Evolving Connectionist Systems: The Knowledge Engineering Approach*, 2nd ed. London, U.K.: Springer-Verlag, 2007.

[34] N. Kasabov, ''Evolving fuzzy neural networks for supervised/unsupervised online knowledge-based learning,'' *IEEE Trans. Syst., Man, Cybern., B, Cybern.*, vol. 31, no. 6, pp. 902–918, Dec. 2001.

[35] E. Özbilge, ''Detecting static and dynamic novelties using dynamic neural network,'' *Proc. Comput. Sci.*, vol. 120, pp. 877–886, Jan. 2017.

[36] H. V. Neto, ''On-line visual novelty detection in autonomous mobile robots,'' *Introduction Mod. Robot.*, vol. 2, pp. 241–265, Jan. 2011.

[37] (2022). *OpenCV Library*. Accessed: Sep. 23, 2022. [Online]. Available: https://opencv.org/

[38] A. Ng. (2019). *Machine Learning Yearning: Technical Strategy for AI Engineers, in the Era of Deep Learning*. [Online]. Available: https://www.mlyearning.org

[39] D. Chicco and G. Jurman, ''The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation,'' *BMC Genomics*, vol. 21, no. 1, pp. 1–13, 2020.

[40] D. Chicco and G. Jurman, ''An invitation to greater use of Matthews correlation coefficient in robotics and artificial intelligence,'' *Frontiers Robot. AI*, vol. 9, p. 78, Mar. 2022.

EMRE ÖZBİLGE received the B.Sc. degree in computer engineering from Eastern Mediterranean University, Cyprus, in 2006, the double M.Sc. degrees in intelligent systems from the University of Sussex, U.K., and in computer science from the University of Essex, U.K., in 2007 and 2008, respectively, and the Ph.D. degree in cognitive robotics from the University of Ulster, U.K., in 2013. He is currently a Lecturer in the Computer Engineering Department, Cyprus International University. His research interests include deep learning, time-series modeling, disease identification, mobile robotics, novelty detection, machine learning, and image processing.

EBRU OZBILGE received the B.S. and M.S. degrees from the Department of Mathematics, Eastern Mediterranean University, Northern Cyprus, in 2000 and 2002, respectively, and the Ph.D. degree from the Department of Mathematics, Kocaeli University, Turkey, in 2006. She was worked with the Department of Mathematics, İzmir University of Economics, from 2006 to 2016. Since August 2016, she has been working with the Department of Mathematics and Statistics, American University of the Middle East, Kuwait. She is currently a Full-Time Professor with the department and serves as the Department Chair. Her current research interests include inverse problems, fractional partial differential equations, and numerical methods.