Novelty detection with self-organizing maps for autonomous extraction of salient tracking features

Yann Bernard, Nicolas Hueber, Bernard Girau

Novelty detection with self-organizing maps for autonomous extraction of salient tracking features

Yann Bernard[1,2], Nicolas Hueber[2] and Bernard Girau[1]

[1] Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France
[2] French-German Research Institute of Saint Louis, F-68300 Saint-Louis, France
Yann.Bernard@loria.fr - Nicolas.Hueber@isl.eu - Bernard.Girau@loria.fr

Abstract. In the image processing field, many tracking algorithms rely on prior knowledge like color, shape or even need a database of the objects to be tracked. This may be a problem for some real world applications that cannot fill those prerequisite. Based on image compression techniques, we propose to use Self-Organizing Maps to robustly detect novelty in the input video stream and to produce a saliency map which will outline unusual objects in the visual environment. This saliency map is then processed by a Dynamic Neural Field to extract a robust and continuous tracking of the position of the object. Our approach is solely based on unsupervised neural networks and does not need any prior knowledge, therefore it has a high adaptability to different inputs and a strong robustness to noisy environments.

Keywords: self-organizing maps, saliency map, dynamic neural fields

Introduction

Visual tracking is currently an important research topic in computer vision. It is a complex problem that requires a strong robustness and adaptation to environmental variability when used in a real world context that current methods do not offer convincingly [6]. The field of computer vision has been historically dominated by models without or with limited learning capabilities, so that the algorithm performances were dependent on prior knowledge of the object to track and a fixed architecture that only took into account a selected number of arbitrarily chosen features. Recent works highlighted the efficiency of deep neural networks to detect and classify objects in a video stream [9] in a supervised way. But this approach relies on a considerable amount of labelled data and considerable computation. Our idea is to use unsupervised neural network properties to efficiently and robustly detect and track objects in a video input stream. Contrary to supervised learning, unsupervised methods need much less data and no labels to learn features, which in turn results in much less computation required and opens the way to embedded tracking in video surveillance for instance.

Self-Organizing Maps (SOM) have already been used as a novelty detection tool or rather as a fault or anomaly detection tool as in [15], [7] or [8]. SOMs are well known for their vector quantization and clustering properties, and for preserving neighborhood relations of the input space when projecting data onto the neural map. Novelty detection relies on these properties by detecting elements that are too far from the neural clusters and that do not fit the topology learned. These properties can be interestingly applied to the image processing field, as in [2] or [16]. Our aim is to use these models to perform novelty detection within images without any prior knowledge, so as to be able to extract unexpected targets from image sequences and track them. Current change detection algorithms struggle with problems like a moving camera, intermittent motions and turbulence [14]. With our method, the change detection will be robust to camera movements and turbulences, as it does not rely on precise previous pixel values in the image. It also has the advantage of not relying on local motion information (optical flow) to detect novelty and therefore it is able to track static objects or objects that stopped moving.

Following the seminal work of [3], we choose to couple our autonomous novelty detection tool to a robust bio-inspired tracking technique based on Dynamic Neural Fields (DNF). DNF are populations of partial differential equations first mathematically analyzed by [1] in a continuous framework. We use a discrete DNF built from populations of excitatory and inhibitory neurons that interact continuously, with a on-center off-surround approach modeled as a synaptic kernel computed as a difference of gaussians applied to the distance between neurons in the neural map. These DNF have been successfully applied to sequential visual exploration of an environment [3] or in [13], with great robustness properties that can even improve with some adaptation like the use of simple spiking neurons [12].

The paper is organized as follows. After a short description of the standard SOM model and of the notations used throughout the paper, section 1 explains how SOM can be applied to image compression by means of a quantization of the thumbnails extracted from the image. Section 2 briefly describes the DNF model and its main properties. The proposed coupling between SOM and DNF for tracking novelty in video sequences is detailed in section 3 and preliminary results obtained with real-world images are given in section 4.

1 Image Representation with SOM

1.1 Self-Organizing Maps

In this paper, we use a standard Self-organizing map (SOM) with a 2D grid neural structure as can be found in [5]. Self-organizing maps (SOMs), initially proposed by Kohonen [4], consist of neighbouring neurons commonly organized on one- or two- dimensional arrays that project patterns of arbitrary dimensionality onto a lower dimensional array of neurons. More precisely, each neuron in a SOM is represented by a d-dimensional weight vector, $\mathbf{m} \in \mathbb{R}^d$, also known as prototype vector, where d is equal to the dimension of the input vectors $\mathbf{x}$. The neurons are connected to adjacent neurons by a neighbourhood relationship, which defines the structure of the map. The mechanism for selecting the winning neuron requires a centralized entity, so that the usual Kohonen SOM is not a fully distributed model as in the cortex organization [10]. After learning, or self-organization, two vectors that are close in the input space will be represented by prototypes of the same or of neighbouring neurons on the neural map. Thus the learned prototypes become ordered by the structure of the map, since neighbouring neurons have similar weight vectors.

It starts with an appropriate (usually random) initialization of the weight vectors, $\mathbf{m}_i$. The input vectors are presented to the neural map in multiple iterations. For each iteration, i.e., for each input vector $\mathbf{x}$, the distance from $\mathbf{x}$ to all the weight vectors is calculated using some distance measure. The neuron whose weight vector gives the smallest distance to the input vector $\mathbf{x}$ is usually called the best matching unit (BMU), denoted by c, and determined according to:

$$\|\mathbf{x} - \mathbf{m}_c\| = \min_i \|\mathbf{x} - \mathbf{m}_i\| \tag{1}$$

where $\|\cdot\|$ is the distance measure, typically the Euclidean distance, $\mathbf{x}$ is the input vector and $\mathbf{m}_i$ is the weight vector of neuron i. The winner c and its neighbouring neurons $i \in N_w$ update their weights according to the SOM rule:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) + \alpha(t)h_{ci}(t)[\mathbf{x}(t) - \mathbf{m}_i(t)] \tag{2}$$

where t denotes the time, $\mathbf{x}(t)$ is an input vector randomly drawn from the input data set at time t, $\alpha(t)$ the learning rate at time t, and $h_{ci}(t)$ is the neighbourhood kernel around c. The learning rate $\alpha(t)$ defines the strength of the adaptation, which is application-dependent. Commonly $\alpha(t) < 1$ is a monotonically (e.g. linearly) decreasing scalar function of t.

The neighbouring kernel $h_{ci}(t)$, which is a function of the distance between the winner neuron c and neuron i, can be computed using a Gaussian function:

$$h_{ci}(t) = \exp\Big[-\frac{\|\mathbf{r}_c - \mathbf{r}_i\|^2}{2\sigma^2(t)}\Big] \tag{3}$$

The term $\|\mathbf{r}_c - \mathbf{r}_i\|$ is the distance between neuron i and the winner neuron c. The precise value of $\sigma(t)$ does not really matter, as long as it is fairly large in the beginning of the process, for instance in the order of 20% of the longer side of the SOM array, after which it is gradually reduced to a small fraction of it, e.g. 5% of the shorter side of the array [5].

We chose to compute the distance between neurons weights and the input vectors as an euclidean distance. We parameterized it with a linearly decreasing σ starting from 0.5 down to 0.001. The learning parameter α starts at 0.6 and linearly decreases to a final value of 0.05. We ran the SOM for 40 epochs for the training and with 10×10 neurons. Increasing the number of epochs or neurons improves the quality of the image representation at the cost of more computation. But as our experimental result could be assimilated to a binary result (the new object is tracked or is not tracked) it is not sensitive to a small performance change. So we chose to limit ourselves to a small but sufficient number of neurons and epochs.

1.2 Image Representation

In order to train the SOM to learn an image, we inspired ourselves from the common application of lossy image compression [2]. A picture or series of pictures to be compressed is split into smaller $k \times k$ pixels wide thumbnails. When the image height or width is not divisible by k, we crop it on the right and bottom. We then use these thumbnails as training samples of a Vector Quantization model. Once the training is finished, the compressed image is composed of the whole codebook, and the index of the Best Matching Unit for each thumbnail extracted from the image or sequence of images. The result is similar to the original image, but with every thumbnail replaced by the codeword learned by its Best Matching Unit. Figure 1 illustrates this compression process.

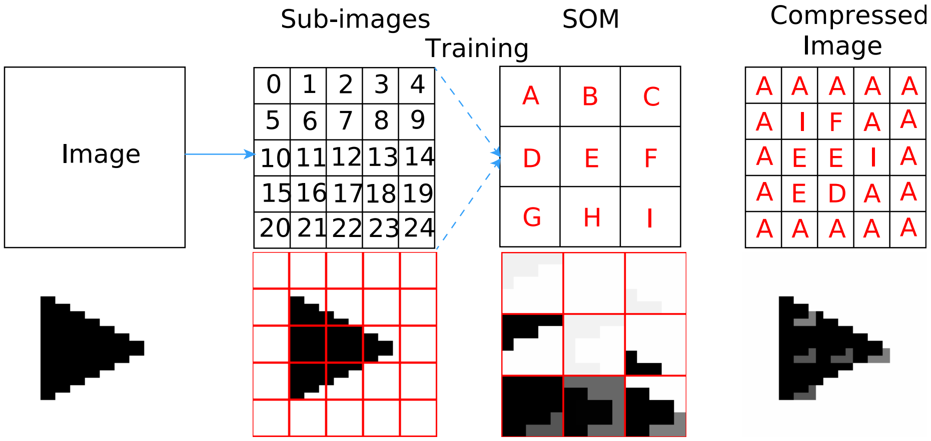

Fig. 1. Simplified scheme of the image compression process (with only 25 sub-images and 9 neurons) with a simple test example underneath.

2 Dynamic Neural Fields

Continuous Neural Fields Theory (CNFT) has lead to the development of two dimensional Dynamic Neural Fields (DNF) [11]. Neural fields are models that represent the evolution of a population of neurons. In our case, we use a two dimensional DNF. The number of neurons is dependent and equal to the size of the input map, because neurons are connected in a retinotopic way to afferent inputs, and are connected in an all-to-all connection scheme between them. All neurons also have a real value attached to them that we call potential. This potential $u(x,t)$, with x being the neuron position in the field and t the time of the simulation, is ruled by the following differential equation :

$$\tau \frac{\partial u(x,t)}{\partial t} = -u(x,t) + \int u(x',t)\omega(||x-x'||)\delta y + \text{Input}(x,t)$$

With :

- τ is the time constant.
- $-u(x,t)$ is the decay term. It is meant to suppress already activated neurons when there is no input or lateral excitation.
- $\omega(||x-x'||)$ is the lateral interaction. It represents the effect of the other neurons onto this neuron's potential. We are using a difference of gaussian with the excitatory gaussian part being narrow with high intensity and the inhibitory one being wide with low intensity. This leads to close neurons having an excitatory effect onto each other and far away neurons inhibiting themselves.
- $Input(x,t)$ is the current value of the afferent input extracted from the input map for this neuron.

For the sake of simplicity and computability, we implement a spatially and temporaly discretized version of the previous formula. It is obtained by handling potentials of a discrete set of neurons (neural map instead of neural manifold) and by using a simple Euler method to estimate the state of $u(x,t+\Delta t)$ knowing $u(x,t)$:

$$u(x,t+\Delta t)=u(x,t)+\frac{\Delta t\left(-u(x,t)+\sum u(x',t)\omega(||x-x'||)+\text{Input}(x,t+\Delta t)\right)}{\tau}$$

Δt is the time step between two estimations, it can be the same for all neurons (synchronous) or different each time (asynchronous). It should be noted that in the original DNF formula, there are more parameters such as resting potential but since we do not use them here, we did not mention them.

It is often difficult to understand how a DNF will behave just from the formula. We have set it up with optimized parameters in order to have a winner-takes-all behaviour where the most prominent and spatially coherent features in the input map create a local bubble of activation in the neural map and suppress the ability of other such bubbles to appear elsewhere in the map.

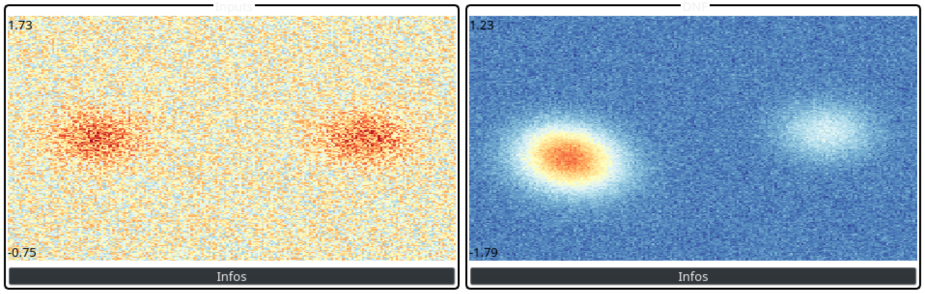

Fig. 2. Example of a DNF. The noisy input with two attractors is on the left, and on the right there is the DNF potentials with a winner takes all behavior.

3 Our Tracking Application

DNF have already been used for tracking applications [3]. It has shown strong resistance to noise and distractions but it needs an a priori knowledge of the features that are to be tracked. SOM on the other hand does not need any a priori knowledge of the input, it can learn and organize itself to represent the input as a concept, meaning that the neurons representing a certain feature will activate when the general pattern of this feature appears. But when there is a completely new input, the distance between it and the BMU will be high. We use this property in order to create a saliency map that robustly outlines novelty in the inputs.

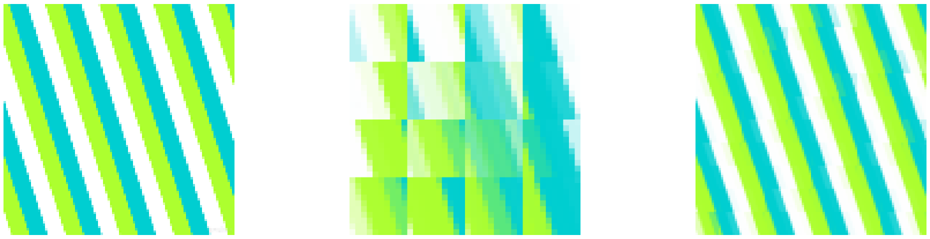

Fig. 3. The learning part of the process. On the left, our "background" image from which we will compare the following received images, composed of white, green and blue stripes. In the center, the codebook learned by the SOM with 16 neurons (4×4) and displaying their learned weights or codewords as 10×10 pixel thumbnails. On the right, the reconstructed image from the learned SOM weights and the BMU indexes (see Figure 1 for more details).

The first part of the algorithm relies on the SOM learning the features of the background. This learning will make this SOM able to construct its perception of the main features of the "usual" visual environment. The background can be composed of a single image or a series of similar images in order to have a learning that is more tolerant to small changes in the input. The learning is the slowest part in SOMs, and one advantage of our method is that we only learn once so it is not so penalizing. An example of learning can be found on figure 3.

The second part of the algorithm is the tracking. For each new frame captured by our camera, we reconstruct it with the SOM as if we were to compress and decompress it. If no new object has appeared in the image, then the compression will be pretty satisfactory and the result will look similar to the whole image captured. If a new object is present, at the compression will be much less satisfactory in the precise location of this "unexpected" object. We thus compute the salient map as the difference between the current captured image and its reconstruction by the SOM. New objects will stand out on this salient map because of this locally unsatisfactory reconstruction, along with noise on the whole map due to inherently lossy compression (particularly around the edges). Finally to extract and track the "interesting" new object and remove the noise from the saliency map, we use a DNF that will focus on the most prominent and spatially coherent activation on the salient map, and that will be able to track it despite significant variations of saliency between consecutive frames, taking advantage of the natural ability of DNFs to self-maintain bubbles of activity. This is illustrated in figure 4.

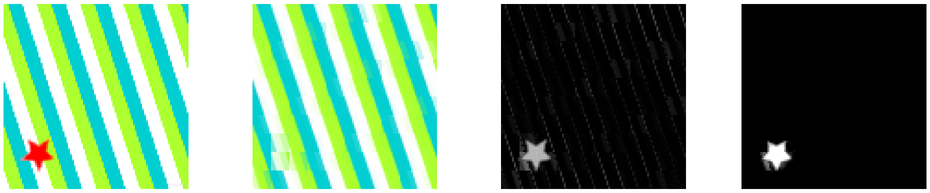

Fig. 4. Example of the tracking process. From left to right; the perceived image with a new object (a red star); the reconstructed image from the learned SOM, the star not having been learned, the SOM cannot reconstruct it; the saliency map obtained by making the difference between the first and second images; lastly the DNF output focusing on the new object and eliminating noise.

Our inputs are colored images. Two completely different ways can be considered to handle color by the SOM. The first way is to make each pixel represented by 3 color values (so 10×10 thumbnails will become input vectors of size 300 for instance instead of 100 for grey-scale images). The other way is to use one SOM by color channel. We are going to explore these two possibilities in the following section.

4 Results

In this section we present and discuss results that have been obtained on real camera footage. We have selected a few video clips where the camera is static, that have moving elements in them (like water ripples, wind or snow) and where a new object appears during the clip. This is for now only a proof of concept, so the experiments presented here are only showing the potential of this approach. But nonetheless these results are already interesting as they confirm some of our hypotheses and give us hints at what future research on this topic should focus on in order to improve the performance and robustness of this kind of coupled unsupervised novelty detection and neural tracking.

Figures 5 illustrate how the proposed approach performs on some real-world examples. Several observations result from considering the saliency map. The first one is that our assumption that new objects unknown to the SOM appear in the saliency map seems correct. We can also note that the edges of the treeline is badly learned by the SOM because it usually struggles with the sharp edges of the stem of the trees and the chaotic nature of the foliage. We have observed this phenomenon in multiple images where there is no smooth separation of colors.

A more expected source of noise is the ripple of the water but we can see in the saliency map that it is nearly invisible. This seems to indicate that there is some sort of generalization of the concept of "water flow" learned in the SOM that makes it robust to small changes there.

The DNF part manages to focus correctly on the object to be tracked when it is there, but in some cases the background noise is too strong and the signal too weak so that the bubble of activity locks itself on badly compressed parts of the image instead of new elements. Let us also note that when there is no new object the DNF focuses on some part of the background. Thus the DNF is not useful to directly detect if something new has appeared, it can only follow the stimulus after it has been detected that something significantly new has appeared. Another example of tracking can be found in figure 6.

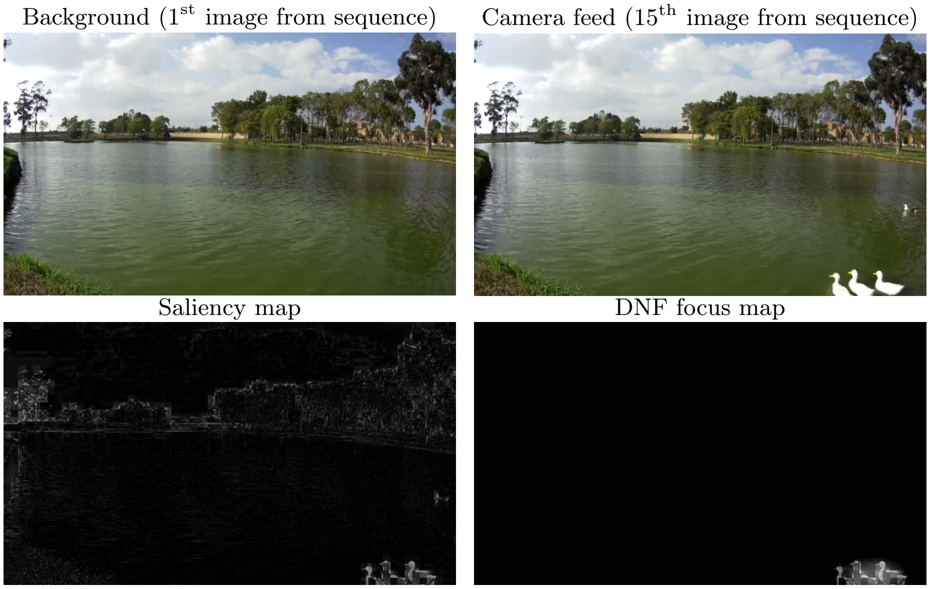
Background (1st image from sequence)
Camera feed (15th image from sequence)
Saliency map
DNF focus map

Fig. 5. We can observe on the saliency map the noise from the lossy image reconstitution by the SOM combined with the new input (the ducks). The DNF manages here to correctly focus on the new target.

Another interesting result is that separating colors into channels and learning all of them separately seems to slightly degrade the performance. The visual artefacts observed on figure 7 are due to the lack of consistency between the BMUs of different color channels. It strongly suggests that learning colors together should be preferred, even if tracking results are not significantly different in our first experiments. There is also a theoretical argument in favor of not separating colors, considering that diminishing the dimensionality of the codewords also reduces the outlierness of new elements of the image because the

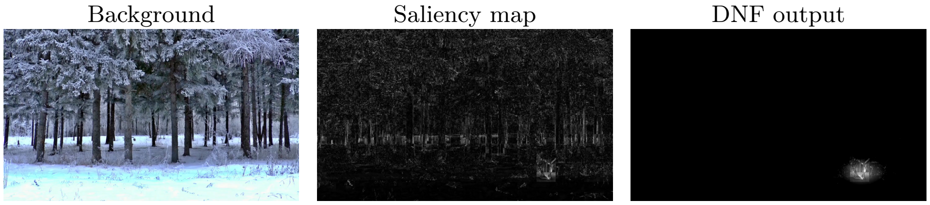

Fig. 6. A second example with a snowy landscape and a dog appearing from the bottom right corner.

distance between them and the closest neuron would be smaller, thus degrading the performance.

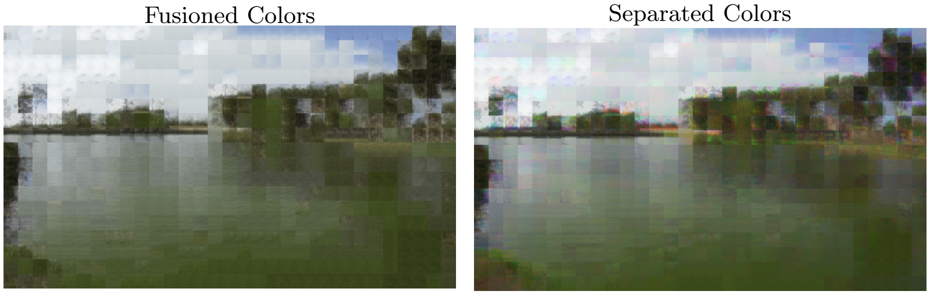

Fig. 7. Learning color channels separately on different SOMs degrades the result with visual artefacts.

Conclusion

In this paper, we have presented a new approach for autonomous tracking using Self-Organizing Maps and Dynamic Neural Fields without any pre-requisite information about the target that we want to track. We have shown that novelty detection can be used for tracking, and that some inherent robustness features of SOM and DNF are a good fit for this application. Furthermore, the unsupervised learning base makes us hope that a low computational cost, real time implementation is possible. The current obstacle to a direct application is the unequal compression of the SOM when it comes to edges and chaotic landscapes that deteriorates the quality of the saliency map. For future works, we aim to improve in this area in order to be able to compare our method with current state of the art tracking models.

Acknowledgements

The authors thank the French AID agency (Agence de l'Innovation pour la Défense) for funding the DGA-2018 60 0017 contract.

References

1. Amari S.: Dynamics of pattern formation in lateral-inhibition type neural fields, Biol. Cybern. 27: 77 (1977)
2. Amerijckx C. , Legat J.-D, Verleysen M: Image Compression using self-Organizing Maps. Systems Analysis Modelling Simulation, 43:11, 1529-1543 (2003)
3. Fix J., Rougier N. P., Alexandre F.: A dynamic neural field approach to the covert and overt deployment of spatial attention. Cognitive Computation, Springer, 3 (1), pp.279-293 (2011)
4. Kohonen T.: The self-organizing map, in Proceedings of the IEEE, vol. 78, no. 9, pp. 1464-1480 (1990)
5. Kohonen T.: Essentials of the self-organizing map, Neural Networks, Volume 37, Pages 52-65 (2013)
6. Kulchandani J. S. and Dangarwala K. J.: Moving object detection: Review of recent research trends, International Conference on Pervasive Computing (ICPC), Pune, 2015, pp. 1-5 (2015)
7. Lee H., Cho S.: SOM-Based Novelty Detection Using Novel Data. In: Gallagher M., Hogan J.P., Maire F. (eds) Intelligent Data Engineering and Automated Learning. Lecture Notes in Computer Science, vol 3578. Springer, Berlin, Heidelberg (2005)
8. Lotfi Shahreza M., Moazzami D., Moshiri B., Delavar M.R.: Anomaly detection using a self-organizing map and particle swarm optimization, Scientia Iranica, Volume 18, Issue 6, Pages 1460-1468 (2011)
9. Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi,A.: You Only Look Once: Unified, Real-Time Object Detection. CoRR, abs/1506.02640 (2015)
10. Rougier N.P., Noelle D.C., Braver T.S., Cohen J.D., and O'Reilly R.C.: Prefrontal cortex and flexible cognitive control: Rules without symbols, PNAS May 17, 102 (20) 7338-7343 (2005)
11. Taylor J.: Neural bubble dynamics in two dimensions: foundations. Biological Cybernetics, Volume 80 (1999)
12. Vazquez R., Girau B., Quinton J.-C.: Visual attention using spiking neural maps. International Joint Conference on Neural Networks IJCNN, San José, United States (2011)
13. Vitay J., Rougier N.P., Alexandre F.: A Distributed Model of Spatial Visual Attention. In: Wermter S., Palm G., Elshaw M. (eds) Biomimetic Neural Learning for Intelligent Robots. Lecture Notes in Computer Science, vol 3575. Springer, Berlin, Heidelberg (2005)
14. Wang Y., Jodoin P., Porikli F., Konrad J., Benezeth Y. and Ishwar P.: CDnet 2014: An Expanded Change Detection Benchmark Dataset, IEEE Conference on Computer Vision and Pattern Recognition Workshops, Columbus, OH, pp. 393-400 (2014)
15. Wong M.L.D., Jack L.B., Nandi A.K.: Modified self-organising map for automated novelty detection applied to vibration signal monitoring. Mechanical Systems and Signal Processing, Volume 20, Issue 3, Pages 593-610 (2006)
16. Xiao Y., Leung CS., Lam PM. et al: Self-organizing map-based color palette for high-dynamic range texture compression, Neural Computing and Applications 21: 639 (2012)