Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2020) 28: 302 – 317
© TÜBİTAK
doi:10.3906/elk-1807-287

Filter design for small target detection on infrared imagery using normalized-cross-correlation layer

H. Seçkin DEMİR[1], **Erdem AKAGÜNDÜZ**[2,*]
[1]Department of Electro-Optics Systems Design, MGEO, ASELSAN Inc., Turkey
[2]Department of Electrical and Electronics Engineering, Çankaya University, Turkey

Received: 27.07.2018 • **Accepted/Published Online:** 23.09.2019 • **Final Version:** 27.01.2020

Abstract: In this paper, we introduce a machine learning approach to the problem of infrared small target detection filter design. For this purpose, similar to a convolutional layer of a neural network, the normalized-cross-correlational (NCC) layer, which we utilize for designing a target detection/recognition filter bank, is proposed. By employing the NCC layer in a neural network structure, we introduce a framework, in which supervised training is used to calculate the optimal filter shape and the optimum number of filters required for a specific target detection/recognition task on infrared images. We also propose the mean-absolute-deviation NCC (MAD-NCC) layer, an efficient implementation of the proposed NCC layer, designed especially for FPGA systems, in which square root operations are avoided for real-time computation. As a case study we work on dim-target detection on midwave infrared imagery and obtain the filters that can discriminate a dim target from various types of background clutter, specific to our operational concept.

Key words: Small target detection, filter design, normalized-cross-correlation, convolutional neural networks

1. Introduction

Small target detection on infrared (IR) imagery is one of the basic, yet challenging problems of infrared vision. Depending on the detection range, target size, clutter level, operational success requirements, computation power availability, and power constraints, several solutions [1–3] have been proposed for various systems, such as infrared search-and-track (IRST), forward-looking infrared (FLIR), just to name a few. Although there are countless approaches to the problem, an efficient, widely accepted, and off-the-shelf solution still does not exist for small target detection or recognition problem on infrared imagery.

Conventional solutions on small target detection on IR imagery [4–19] aim at reducing the background clutter by proposing different filter types in a heuristic manner, such as mean, median, top-hat, and Gaussian. Some of these methods are specifically designed for infrared small target detection [11, 13, 15–17, 20]. Although these filters show some success to reduce the clutter, they are not inherently designed to detect a specific type of target. Or they do not have the ability to differentiate a target from a false target, which is usually not clutter, but a different object in the scene, like a bird or a bad pixel. Multiscale approaches [21–25] to the problem provide scale-invariance; thus, they are robust to target size change. However, neither the multiscale approaches nor some recent entropy [26] or saliency-based [27] methods promise sufficient performance against false targets or target-like clutter.

[*]Correspondence: akagunduz@cankaya.edu.tr

Using correlation filters to detect small or extended targets in a signal is a well-studied approach [28]. Especially for different infrared imaging subbands such as midwave (MW) or long-wave (LW) infrared, normalized-cross-correlation (NCC) is proven to be an effective template matching method [29]. However, the problem with the NCC-based matching is the ambiguity in filter selection or design. To solve this problem, the idea of supervised filter training is previously introduced, in which the required filter is designed using a dataset [30]. Especially to solve the tracking problem where the target-clutter relation constantly varies, learning-based approaches are highly effective [31, 32].

In this paper, we introduce a learning-based approach to small target detection filter design problem on infrared imagery. To this end, we propose the *normalized-cross-correlational* layer, a variation of convolutional layers of a neural network. Utilizing the NCC layer, we introduce a framework, in which supervised training is used to compute a filter bank, i.e. the optimal filters and the optimum number of filters required for a specific detection/recognition task. By feeding the proposed normalized-cross-correlational neural network (NCC-NN) structure with positive samples, such as different snapshots of the target, and negative samples, such as different types of clutter that create false alarms, a filter bank is obtained as the weights of a complete layer of the trained neural network. This way, not only the detection success is maximized but also the filters that create the minimum false alarm rates are obtained simultaneously.

Convolutional neural networks (CNNs) have recently become the state-of-the-art de facto standard of any signal-based machine learning approach. There are many recent studies that focus on using deep CNNs to detect and recognize various types of objects or targets. The main reason we choose to use the proposed NCC layer, instead of convolutional layers, is that NCC layer needs relatively less data to converge. For many operational concepts, such as detection in search-and-track systems, the amount of available training data is not sufficient to prevent a deep CNN from overfitting. We discuss the benefits of using the NCC layer instead of the conventional convolutional layer in the following sections.

We also propose the mean-absolute-deviation NCC (MAD-NCC) layer, an efficient implementation of NCC layer, designed especially for the FPGA systems. In this optimized design, square root operations are avoided for real-time computation and minimal resource usage. As a case study, we work on dim-target detection on midwave infrared imagery and benchmark the performance of different filter designs. The results of the trained NCC-NN show that instead of choosing heuristic filter designs, it is possible to converge to a filter set that would come up with improved receiver operating characteristics.

The paper is organized as follows: the next section introduces the proposed neural network layers, the NCC layer and its optimized form and the MAD-NCC layer. The dataset used in our experiment is introduced in Section 3. Results are provided in Section 4, for a case study on dim-target detection on midwave infrared imagery, while the final section outlines the conclusions of this study.

2. Normalized-cross-correlation layer for NNs

In this section we introduce the normalized-cross-correlational layer, which is an alternative to convolutional layers in neural networks. In addition, explicit formulation of forward and backward propagation functions for the proposed structure are provided.

2.1. The NCC Layer

The NCC layer is a variation of a convolutional layer of a neural network, with the exception that the input is normalized prior to being convolved with the filters (or kernels) of that layer. A simple structure of the NCC layer is given in Figure 1. As can be seen from the figure, if the normalization blocks are removed, the NCC layer becomes identical to a convolutional layer.

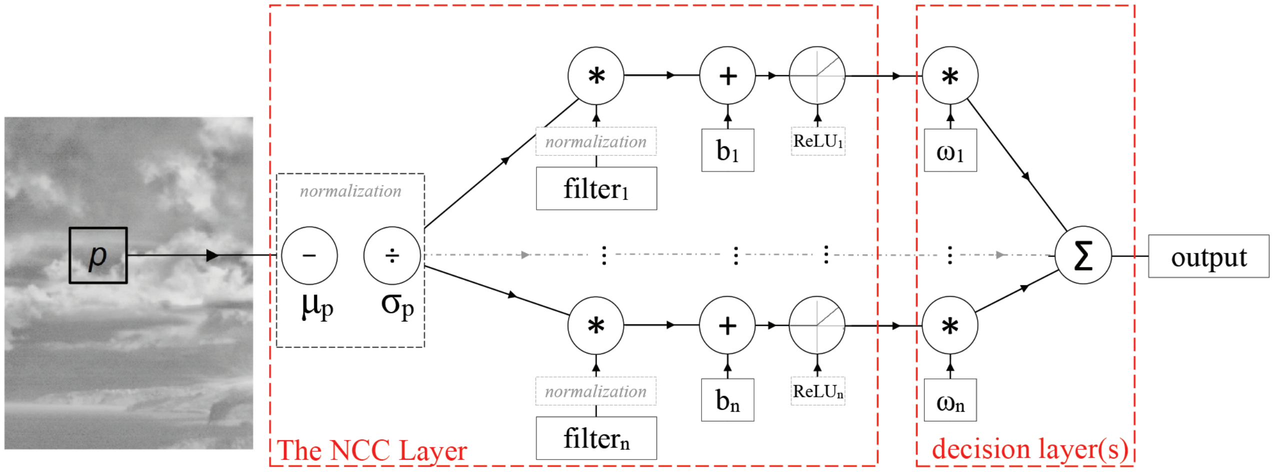


Figure 1. A sample two-layer NCC-NN structure with an NCC layer and a simple fully connected rectified linear unit (ReLU) decision layer is depicted. NCC layer is similar to a convolutional layer, with the exception that the patch is normalized prior to being convolved with the filters.

The outcome of normalizing the input before convolving it with a filter is simply limiting the output values. When both the filter and the input is normalized, the convolution operation becomes identical to normalized-cross-correlation[1] and the output is bounded in the range [–1,1]. The output of NCC between two 2D discrete signals *A(i,j)* and *B(i,j)*, defined as $\rho^{2D}_{A,B}$ in Equation 1, is a measure of linear dependence between A and B, with +1 showing complete positive dependence, –1 showing complete negative dependence and 0 showing linear independence.

$$\mathbf{p}\overline{*}\mathbf{f} = \frac{1}{(n-1)} \cdot \sum_i \frac{(\mathbf{p}(i)-\mu_p)\cdot(\mathbf{f}(i)-\mu_f)}{\sigma_p \cdot \sigma_f} \tag{1}$$

In Equation 1, the $\overline{*}$ symbol represents the normalized-cross-correlation operation, n is the total number of pixels[2] in **p** or **f**, μ_p and μ_f are the average pixel values, and σ_p and σ_f are the standard deviations of the pixel values of p and f, respectively.

2.2. Why normalize?

One of the main reasons why a normalization is conventionally not preferred for a CNN layer is the fact that bounding the output range may limit, or even diminish propagation. What a Re-Lu layer does in a CNN is [33] to destroy negative, and proportionally admit positive, forward propagation. Limiting both the negative and positive output of the convolutional node, like it happens for the proposed NCC layer, is similar to using a poor

[1]Either the filter or the signal must be real-valued and symmetric for this generalization to hold, which is a general case when CNNs are considered. The pixel values are always real valued and so our filter values. How to constrain the filter to symmetric shape or its (un)necessity for our application of the NCC layer is discussed in the following sections.

[2]Although p is a 2D real-valued signal, an image patch, the notation *p(i)* is preferred instead of *p(i,j)* for the sake of simplicity. The variable i represents a total of n pixels in both horizontal and vertical dimensions of the patch p.

activation function and will result in a poor performance for CNNs. Despite this seemingly undesired fact, we have two main motivations in utilizing the NCC layer. Firstly, the main reason NCC is a good template matching method for infrared imagery is the fact that the low signal frequencies (like the average image intensity) are destroyed in normalization. In infrared imagery this is a desired fact because depending on the dynamic range of an infrared detector, the low frequencies of an infrared image is expected to differ among different systems that use the same infrared subband. Theoretically, a CNN is capable of discriminating these low-band signal properties. However, the performance of CNNs, especially deep CNNs, depend on the availability of data. In practical situations, such as military related infrared detection problems, such data are not usually available. In those cases, CNN-based methods may easily overfit, for example, to mean intensity values of a scene (which may depend on the air temperature), when the collected samples are limited. In addition, a detection or recognition capable neural network, trained with a specific dataset, may be (and is most usually) applied to perform for another infrared subband or another detector, in which low frequency elements like the mean intensity are expected to differ, as well. Thus, the NCC layer, when infrared detection and recognition tasks are considered, has more generalization power compared to a convolutional layer and is more prone to converge to optimum weights (i.e. filter shape) with relatively limited data.

Secondly, utilizing the NCC layer structure, compatible with a general neural networks architecture, is quite beneficial considering the fact that it can easily be trained using back-propagation. This way we can extract the NCC layer as a filter bank and directly utilize it for an operational purpose (detection, recognition, etc.) In order to better experiment the proposed concept, in the following section we provide a custom NCC layer structure, which can be easily combined with any of the multilayered deep learning software libraries.

2.3. NCC layer implementation

An NCC node, as given in Figure 1, is a serial combination of a normalization node and a convolutional node. Equation 1 is an explicit formulation of the forward operation of this layer. However, in this chapter we will obtain the forward and backward propagation formulas for this layer by considering the normalization and convolution as two separate sequential operations. The reason we chose to separate these two formulas is practical. Extremely fast GPU-based solutions exist for forward and backward convolution operations in CNNs. Thus, instead of constructing the function for this new layer from scratch, it is practically much more convenient to detach two operations, derive functions for normalization only, append these functions to a convolutional layer of an existing CNN library (such as MatConvNet [34]) and experiment on a desired set of data. Below in Equation 2, an equivalent formulation of the NCC forward function is provided.

$$xcorr(p) = \overline{\mathbf{p}} \star \overline{\mathbf{f}} \tag{2}$$

The forward function of the NCC is simply the convolution of the normalized 2D discrete signal $\overline{\mathbf{p}}$ with the filter $\overline{\mathbf{f}}$. The normalized $\overline{\mathbf{p}}$ can be calculated as:

$$\overline{\mathbf{p}}=\frac{1}{\sqrt{n-1}} \cdot \frac{\mathbf{p}-\mu_p}{\sigma_p} \tag{3}$$

In Equation 3, μ_p represents the mean pixel value and σ_p represents the standard deviation of the patch $\mathbf{p}$. Below we also provide the well-known formulas of mean and standard deviation calculation because we will need to derive their derivatives for backward function calculation in what follows. Using 2 and 3, we obtain the same forward operation that was given in 1, but in an alternative form where normalization is nested within the convolution/correlation operation.

$$\mu_p = \sum_{i=1}^{N} \mathbf{p}(i) \ , \ \sigma_p = \sqrt{\frac{1}{(n-1)} \sum_{i=1}^{N} (\mathbf{p}(i) - \mu_p)^2} \tag{4}$$

In order to obtain the backward propagation functions of a node in a neural network, we find the partial derivatives of the output with respect to input patch and the filter values (weights) of that node. In our case, the forward function is a normalization function nested within convolution. In order to obtain the back propagation function, we simply apply the chain rule:

$$\frac{\partial O}{\partial \mathbf{p}} = \frac{\partial O}{\partial \overline{\mathbf{p}}} \cdot \frac{\partial \overline{\mathbf{p}}}{\partial \mathbf{p}} \tag{5}$$

$$\frac{\partial O}{\partial \mathbf{f}} = \frac{\partial O}{\partial \overline{\mathbf{f_k}}} \cdot \frac{\partial \overline{\mathbf{f_k}}}{\partial \mathbf{f_k}} \tag{6}$$

In Equations 5 and 6, O represents the scalar output of an NCC node, $\mathbf{p}$ represents the input patch, and $\mathbf{f}_k$ represents the kth filter in that layer. Equations are identical since both the input and the filter are normalized and the convolution operation is commutative. $\partial O/\partial \overline{\mathbf{p}}$ stands for the partial derivation of convolution operation, whereas $\partial \overline{\mathbf{p}}/\partial \mathbf{p}$ stands for the derivative of normalization operation (3), which is given below in Equation 7.

$$\frac{\partial \overline{\mathbf{p}}}{\partial \mathbf{p}} = \frac{1}{\sqrt{n-1} \cdot \sigma_p} \cdot (\mathbf{I} - \frac{\mathbf{1}}{n}) \cdot (\mathbf{I} - \frac{(\mathbf{p}-\mu_p)(\mathbf{p}-\mu_p)^T}{(\mathbf{p}-\mu_p)^T(\mathbf{p}-\mu_p)}) \tag{7}$$

As mentioned above, by using Equations 3 and 7, we implemented the NCC layer within MatConvNet library [34] by appending the forward and backward functions for normalization to a custom layer, which also uses the hardware-based (both forward and backward) convolution functions of this library. By using any other open source deep net library (such as Caffe [35], etc.), NCC layer can be identically implemented.

2.4. MAD-NCC layer implementation

The MAD-NCC layer is an efficient implementation of NCC layer, in which mean-absolute-deviation (MAD) operation is used instead of standard deviation operation. Thus, for the MAD-NCC layer implementation, the input patch $\mathbf{p}$ is normalized by using the equation below:

$$\tilde{\mathbf{p}} = \frac{1}{\sqrt{n}} \cdot \frac{\mathbf{p} - \mu_p}{mad_p} \tag{8}$$

In 8, mad_p stands for the mean-absolute-deviation of image patch and is calculated as:

$$mad_p = \frac{1}{n} \sum_{i=1}^{N} |\mathbf{p}(i) - \mu_p| \tag{9}$$

Consequently, the backward function $\partial O/\partial\tilde{\mathbf{p}}$ for MAD-normalization can be derived as:

$$\frac{\partial\tilde{\mathbf{p}}}{\partial\mathbf{p}}=\frac{1}{\sqrt{n}\cdot mad_p}\cdot(\mathbf{I}-\frac{\mathbf{1}}{n})\cdot(\mathbf{I}-\frac{(\mathbf{p}-\mu_p)\cdot sign(\mathbf{p}-\mu_p)^T}{n\cdot mad_p} \tag{10}$$

In Equation 10, *sign(·)* is the signum function. Similar to the NCC layer, by using 8 and 10, we implemented the MAD-NCC layer within MatConvNet library [34] by appending the mad-normalization (forward and backward) functions into a custom layer and again by using the built-in hardware-based convolution functions of this library.

Our motivation behind designing the MAD-NCC layer is basically to avoid vector-based square-root operations that exist in standard deviation calculation (Equation 4). Although there are novel approaches to square root calculation on FPGA [36], this operation is relatively slow compared to many other operations, such as division by 2^n (i.e. bit shift). This way, we expect to have a faster forward operation, with a slightly degraded performance. In the following sections we compare the performance of our NCC and MAD-NCC hardware implementations, and discuss the impact of the proposed trade-off.

3. The dataset

The problem of IR target detection is generally specific to target type, target range and camera properties. For instance, in order to detect air targets from long ranges up to 10 km, a proper systems engineering study would indicate the need for a camera with a narrow field-of-view (FOV) (smaller than 5°) and a detector of midwave infrared band. Thus, searching for or trying to create a general detection dataset for the entire IR target detection problem would be undesirable and incorrect by definition. In addition, even small target detection is a broad definition which must be narrowed down for proper academic and engineering analysis.

In order to study the proposed NCC Layer, we have constructed a "small target database" (not a small database but a database of small targets) specific to our problem. We used a midwave infrared detector and an optics with a narrow FOV[3]. The dataset includes scenes of actual targets that are captured from different ranges so that the targets in pixel coordinates are small (i.e. around a few pixels or subpixels). The targets that have been captured are various air platforms, such as UAVs, helicopters, planes. The scenes included high levels of natural clutter such as terrain, sea surface (with glints), clouds and some human constructions (Please check Figure 2).

In addition to real-world scenes, images of point targets captured by using the actual optical system and a collimator [37] are also included in the dataset. This system provides perfect scenes with virtually no clutter. However, these environments provide simple but undisturbed target signals with the actual detector characteristics, such as the detector noise and bad-pixels. IR detectors provide pixels with nonuniform transfer functions and for this purpose require a nonuniformity correction (NUC). Depending on the NUC method, some pixels may be selected as "bad" (i.e. having bad transfer functions) or dead (i.e. unresponsive) and can be replaced with the adjacent pixels [38]. However, even the best NUC and bad pixel replacement (BPR) methods leave undetected bad pixels and in most cases these bad pixels are very similar to real small targets. Thus, including collimator scenes in our dataset helps us to study these detector characteristics as well.

[3]The details of the optical system and the dataset itself are unfortunately not publicly available, due to the classified nature of the project that belongs to ASELSAN Inc.

3.1. Dataset preparation

The dataset includes 4843 frames from 7 different scenes, 2 of which belong to collimator scenes and the rest captured as real-world scenes (Please see Figure 2a). However, the dataset that we utilize, or in other words, that we feed into our proposed learning structure is not the collection of these image frames. Instead, we extract nonoverlapping 15×15 image patches (Figure 2b) from these approximately 4843 images and use them as the actual input of the proposed learning framework. We manually annotate the patches with actual targets as positive samples. Thus, we end up with 5047 15×15 patches including various small targets (some collimator scenes included multiple targets; thus, it is more than 4843). These target patches are selected such that the small target is located at the center of the 15×15 patch (Please see Figure 2c).

The rest of the patches, which do not include any small targets, are recorded as the background patches. More than two million background patches were obtained. Having around 5000 positive samples and 2 million negative samples is an extreme case of an unbalanced dataset. Thus, we have applied a very basic correlation-based distance analysis to intelligently select a subset of the background patches, which included all different types of background clutter we come across in our images, such as sky, clouds, terrain, sea surface, collimator noise, bad-pixels. This way we have shrunk our negative samples set to 123,385 number of 15×15 patches.

3.2. Data augmentation

The data that is fed into a convolutional neural network is usually raw. In other words, no feature extraction is carried out so that the network trains itself to find and utilize the necessary features to accomplish the classification task. Very similarly, in our small target detection problem, the proposed NCC-based network accepts raw image patches (i.e. pixel values) to detect targets. In these kinds of networks, the framework do not inherently have any kind of invariance property, such as transformation invariance, illumination invariance. In these cases, data augmentation is utilized to train the system in a desired invariant manner.

For this purpose, we have created the shifted and rotated versions of positive and negative samples of the 15×15 patch dataset. For each positive sample (i.e. patches with a small target), we have added 3 rotated versions (by rotating 90°, 180° and 270°). For each rotated version (and the original patch), we created 1 pixel and 2 pixels shifted versions of the patches. A shifted positive sample can be seen in Figure 2d. Thus, for each patch, using the rotated (4 orientations) and shifted (16 shifts obtained by a combination of $\pm 1, \pm 2$ pixel shifts in both x and y axes) combinations of the patches, the positive sample set is enlarged by 64 times, creating approximately more than 326.000 patches with a small target. Augmenting shifted and rotated versions of positive samples is crucial for our method, because the correlation filter is inherently not invariant to shifts and rotations.

For the negative samples, the only rotation is implemented for the purpose of augmentation. Thus, approximately half million negative samples were obtained. Examples of an original and rotated background patches can be seen in Figures 2c and 2d.

4. Experimental work

In this section, we provide the details of the experimental work and we commence by introducing the NCC structure used in our experiments.

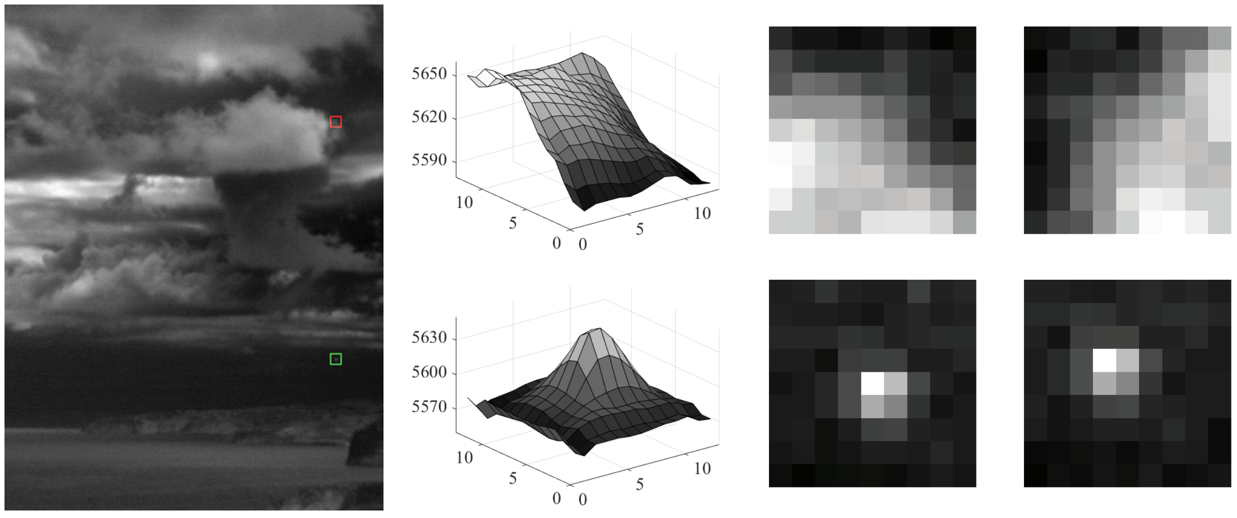


Figure 2. In (a), a sample dataset image with a sample 15×15 background region (the red square) and a 15×15 dim-target region (the green square) are depicted. The surface plots for the background (up) and the target (down) are shown in (b). The background (up) and target patches (down) are shown in (c). In (d), augmented versions (up: a rotated sample for the background patch, down: a shifted version for the target patch) are given.

4.1. Two-layered NCC network

The main motivation of this study actually comes out of a practical need to answer the question: "What are the minimum number and size of correlation filters that are required to discriminate a small target from any type of clutter we come across?". The number of filters that can be used in our FPGA-based system is limited; thus, finding the shape(s) of the filter(s) to accomplish this task is our main interest for this case study.

For this purpose, we have designed a simple two-layered structure using the proposed NCC layer. The first layer, as depicted in Figure 1, is the NCC layer with at most four filters. The maximum number of filters is chosen relatively small when compared to the state-of-the-art CNN structures. The reason behind this limitation is mainly the fact that for the FPGA-based small target detection system we want to develop, we simply cannot afford to have more than a total of four 15×15 correlation filters that can work real-time and together in our system. We wish to observe if there is a feasible solution to the problem with these requirements and hopefully obtain satisfactory results with a less number of filters. Following the first layer, the results of each correlation is fed to a separate rectified linear unit (ReLU) so that no negative correlation results are propagated in forward direction and a nonlinear discriminative function can be obtained. The second layer is a single decision layer which is just a weighted sum of the ReLU outputs, again as depicted in Figure 1. A simple representation of the proposed NCC Network is given in Table 1.

Table 1. The Two-Layered NCC Network designed for small target detection.

Input Layer	*IR patch*	15×15
NCC Layer	(N⩽4) *filters*	15×15×N
Decision Layer	*weights*	1×1×N
Output Layer	*targetness measure*	scalar

As it can been seen in Table 1, both the input patches and the NCC filters have 15×15 pixels size. Moreover, since there is no padding implemented for the NCC-layer, the decision layer has the size of $1 \times 1 \times \mathrm{N}$, where $\mathrm{N} \leqslant 4$ is the number of filter outputs. These layers are implemented as custom layers in MatConvNet library [34]. Apart from the convolution function (and its derivate), which is both CPU and GPU compatible in MatConvNet, all other forward and backward operations are implemented as explained in Section 2.3.

4.2. Training

The output of the decision layer is a single real value representing the confidence value for having a target within the input patch. Training this structure corresponds to a binary classification problem. To solve this problem, we assign +1 value for any training patch (positive sample) with small target and –1 value to any training patch (negative sample) with no small targets but clutter or noise. Then using l_1-norm as the loss function, the network is trained by back-propagation. Batch normalization is used with a batch size of 40 patches. The training[4] is performed, using MatConvNet library [34], on a desktop system with dual 2.6 GHz processors and GPU support of 2880 cores.

In order to train the proposed network, 80% of the patches are used for each experiment. Thus, for each experiment approximately 260.000 positive samples and 400.000 negative samples are used. Each experiment is executed five times, using a different image subset containing 80% of the whole dataset.

4.3. Resulting filters

For different values of N (from 1 to 4), the proposed structure is repeatedly trained. Regardless of the number of filters, in all experiments the system converged to (a) stable filter(s) before 5 epochs. In Figure 3a the resulting 15×15 filter for an experiment with N=1 and in Figures 3b–3e, the resulting filters for an experiment with N=4 are shown.

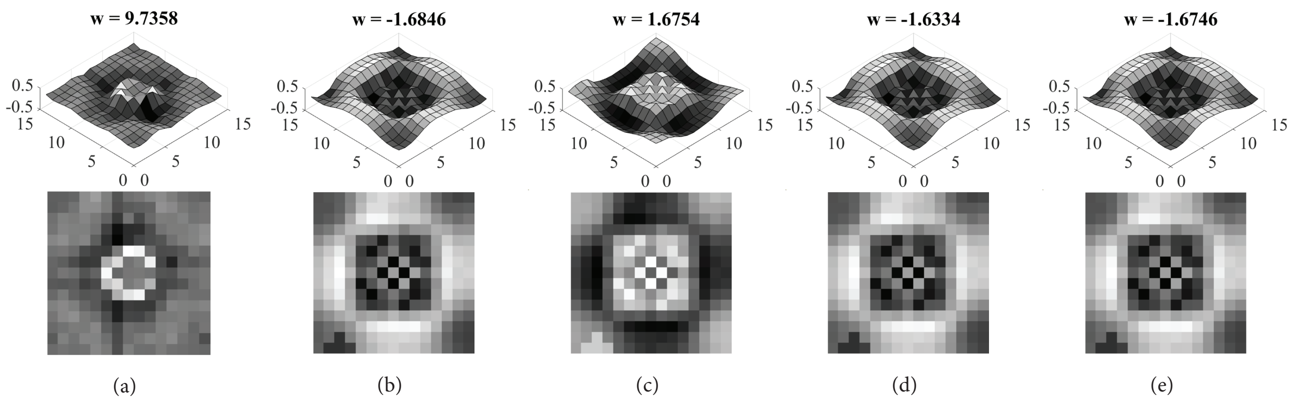


Figure 3. In (a) the resulting 15×15 filter for an experiment with N=1 and in (b,c,d,e) the resulting filters for an experiment with N=4 are shown, as 3D surface plots (above) and gray value 2D images (below).

The experiments with a single filter always resulted in a shape very similar to what we see in Figure 3a. This shape looks very much like a Mexican hat filter, with one exception that at the center of the filter there is a small pit.

[4]Stochastic Gradient Descent (SGD) algorithm with momentum is employed, considering Momentum: 0.95, Initial Learning rate: 0.001 , Weight Decay: 0.0005.

The results of an experiment with four filters can be seen in Figures 3b–3e. Similar to the case when N=1, the filters look very much like the Mexican hat filter with a pit at the center (or the negated version of this shape). In addition, for almost all runs when N=4, some of the resulting filters were almost identical to each other, such as the filters in Figures 3b, 3c, and 3e, which leads us to think that using 4 filters is unnecessary and redundant.

When we considered all the experiments, regardless of the number of filters, we always obtained a version of this Mexican hat-looking shape, with a pit at its center. Heuristically, this shape makes a lot of sense. The center is similar to a Gaussian with a pit. The Gaussian shape is to detect small-targets, whereas the center pit is to eliminate single pixel deformations, such as bad pixels. On the other had, the perimeter of the shape, a zero-sum wave is to eliminate clutter, which is usually not uniform.

These results show that this Mexican-hat like shape is the filter shape that we are looking for. Moreover, the results show that a single filter is sufficient in discriminating the clutter from the small targets in our dataset, because even with larger N, we always had a version of this filter in the results. In this study we will call this shape ”the hat” filter.

Instead of using the output of the any of the experiments (such as provided in Figure 4a), we generated the hat filter automatically by modifying a Mexican hat filter in way that it is most similar to the results of the N=1 experiments we obtained (i.e. with a simple optimization routine which would determine the interval of the Mexican hat wavelet in MATLAB). The so-called hat filter can be seen in Figure 4b. The reason we have decided to generate a filter, instead of directly using one of the experiment results, is to avoid over-fitting and to guarantee rotation invariance. The generated shape has perfect radial symmetry (imposed by the optimization procedure) and thus invariant to rotations.

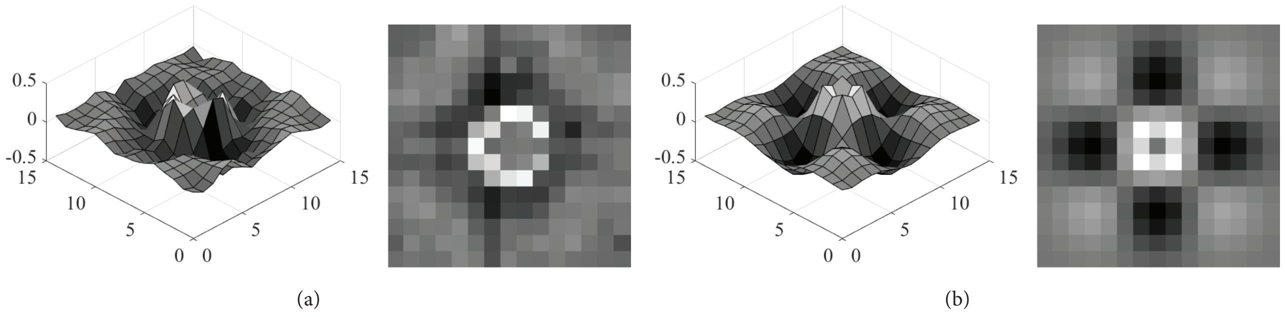


Figure 4. In (a), the result of one of the N=1 experiments can be seen both in 3D (left) and 2D (right) plots. In (b), the so-called ”hat” filter is depicted.

4.4. Benchmarking

The experimental results show that the so-called hat filter is sufficient to discriminate a small target from various types of clutter and noise. In order to further analyze the strength of the hat filter compared to other filters and to optimize its implementation on an FPGA platform, in this section we provide the benchmarking of different single detection filters. The listed filters below are tested for the entire dataset and receiver-operating characteristics are presented in the following section. Please refer to Table 2 for algorithmic complexities of the benchmarked methods.

1. MAD-ratio: This method simply checks the deviation, or mathematically speaking the absolute difference, of each pixel from the mean of the patch that the pixels belong to. For each pixel, the ratio of its individual

Table 2. Numbers of arithmetic operations for each bench-marked method, for an N-by-N image and a f-by-f filter size, are provided. IPI model [16] includes an convergence loop, and number of operations depend on the context of the image. The number of operations are indicated with a modifier k_i, where $k_i > 1$.

algorithm:	MAD-Ratio	NCC with std.	NCC with mad.	IPI model	Unnorm. corr.
# of mul.:	N^2	N^2	N^2	$k_1 \times N^2$	N^2
# of add.:	N^2/f^2	$2 \times N^2/f^2$	$2 \times N^2/f^2$	$\mathrm{k}_3 \times N^2$	N^2/f^2
# of div.:	N^2/f^2	N^2/f^2	N^2/f^2	$\mathrm{k}_3 \times N^2$	-
# of sqr. root:	-	N^2/f^2	-	$\mathrm{k}_4 \times N^2$	-

difference from the patch mean value to patch mean-absolute-deviation is calculated. If the ratio is larger than a threshold (2.5 in our experiments) the pixel is chosen to be a (part of a) target. This method is chosen because it is more FGPA-friendly than any correlation filter-based method.

2. Gaussian filter: Checking the correlation of a Gaussian filter is the most well-known method used [4, 5, 7–9] in small target detection. Although the Gaussian shape assumption for a small target is highly heuristic, it is realistic considering that the target signal is a convolution of many independent transfer functions such as atmosphere, optics, detector response. For this reason, we have included a group of Gaussians with different standard deviations (0.5, 1.2, and 2.0 pixels in our experiments) in our benchmarking.

3. The hat filter: This is the main result that we obtained from the experiments with the NCC network in Section 4.3. For repeated experiments trained with different subsets of the dataset, the filters always converged to the so-called hat filter. This result is mostly reasonable since the mid part of the hat is a Gaussian and it detects the small target, whereas the side-waves are for eliminating negative patches with background clutter. The pit at the center, we presume is the result of the bad-pixels in our dataset, which are negative samples.

   In order to further optimize this filter for the FPGA platform we have created different versions by considering the following:

   (a) Filter size N × N: This is the cropped filter size. For example 9 × 9 filter refers to the cropped version of the original 15 × 15 the hat filter, where 3 pixels are trimmed from four sides. In other words this is not down-sampling, but trimming. The reader should note that the size of the filter is very crucial for FPGA implementation.

   (b) Precision: We also simulated fixed-point calculations in MATLAB in order to observe the real results of the FPGA platform. Thus, in this section, if the filter is referred to as ”fixed”, it simply shows that fixed point calculations are used in the simulation. Otherwise, it is referred to as ”ideal”, which indicates MATLAB’s double precision.

   (c) STD vs MAD: To calculate the normalized cross correlation between two patches, we need to find the standard deviations of each patch (Equation 1). However, as mentioned in previous sections, standard deviation calculation requires expensive square-root operations. To make the correlation calculation simpler on the FPGA platform, for some experiments we use mean-absolute-deviation (MAD) instead of standard-deviation.

4. Unnormalized correlation: As discussed in Section 2.2, the reason we employ normalized-cross-correlation, instead of convolution, which is the standard operation of a CNN, is simply to avoid overfitting certain characteristics of infrared imagery. In order to experimentally portray this condition, using the same dataset, we train a two-layer plain CNN with 4 filters. This CNN is the unnormalized version of the proposed NCC-NN structure. The results of this detection method is also presented in our results, namely as unnormalized correlation.

5. Infrared patch-image (IPI) model: We have also chosen to add a method that is designed for infrared small-target detection. We believe that infrared patch-image model [16] is a representative technique for this purpose. It is based on the nonlocal self-correlation property of IR images and small target detection task is transformed into an optimization problem of recovering low-rank and sparse matrices.

4.5. Detection performance

In this subsection we present the detection performance of the benchmarked methods. As mentioned in the previous subsection, for the Gaussian filter and the hat filter, different versions are also implemented.

We have included a group of Gaussian filters with different standard deviations, 0.5, 1.2, and 2.0 pixels, because these three versions showed the best performance on the given dataset.

For the hat filter, we have implemented several versions, aiming at correctly evaluating the possible optimizations that can be applied on an FPGA platform. In addition to the ideal 15×15 filter (where ideal refers to the unlimited precision of the filter values, i.e. MATLAB double precision[5]), 9×9 and 7×7 cropped versions of the ideal filter are also included. In addition, 7×7 and 5×5 cropped versions are also implemented with fixed-point precision of 8 bits. Moreover, as explained in the previous subsection, MAD-based correlation methods are also implemented for 9×9 ideal, 9×9 fixed, 7×7 fixed-point, and 5×5 fixed-point versions.

In Figure 5, the ROC curves for each implemented methods are depicted. As it can be seen from the figure, while MAD-ratio method performs the worst, ”the hat” filters with ideal precision perform the best. The 15×15 and 9×9 ideal precision ”hat” filters give the best results, whereas the performance considerably falls when the filter size is 7×7 or lower. The Gaussian filter with 1.2 pixels standard deviation is the third best, with slightly worse false alarm rates. This is an expected result because the advantage of ”the hat” filter against the Gaussian filter is its ability to eliminate clutter.

Although Figure 5 shows that ideal precision is crucial for the best performance, it is usually not feasible for an FPGA platform. Therefore, we also search for an optimum filter with proper FPGA optimizations such as fixed-point precision and MAD-based correlation method. In Figure 5, we can see that 9×9, fixed-point and MAD-based implementation of ”the hat” filter (solid blue ROC curve) provides an average performance within the benchmarked methods. This implementation is the most feasible one considering filter size, FPGA size, and run-time speed, for our FGPA-based system. However, this is not a generalization and may change for different systems and designs.

Another important remark of the results is the high false alarm rates for the unnormalized correlation and IPI methods. As expected, the varying characteristics regarding the infrared dynamic range of different scenes increases the false alarm rates. We believe that the performances of these methods, even those of simple filters like Gaussian, or MAD-ratio, fall behind mainly because of this phenomenon.

In Figure 6, comparative results for five different scenes with and without targets for two methods, namely 9×9, fixed-point, and MAD-based hat filter (solid blue ROC curve in Figure 5) and MAD-ratio method (solid

[5]Gaussian and the unnormalized correlation filters are also ideal in precision.

dark-blue ROC curve in Figure 5) are depicted. In each image the targets are designated by red circles, while the fourth image from left has no target, but clutter. Each declaration is shown with a cyan rectangle. Thus, cyan rectangles not covered by a red circle are false alarms. In Figure 6a, the results of ”The Hat+9x9+MAD” filter is given, while in Figure 6b results of the MAD-ratio method is provided. We can see that, although optimized for an FPGA platform, in terms of precision, size and mathematical operations, “the hat” filter is highly resilient against different types of clutter, such as terrain, cloud, and bad-pixels.

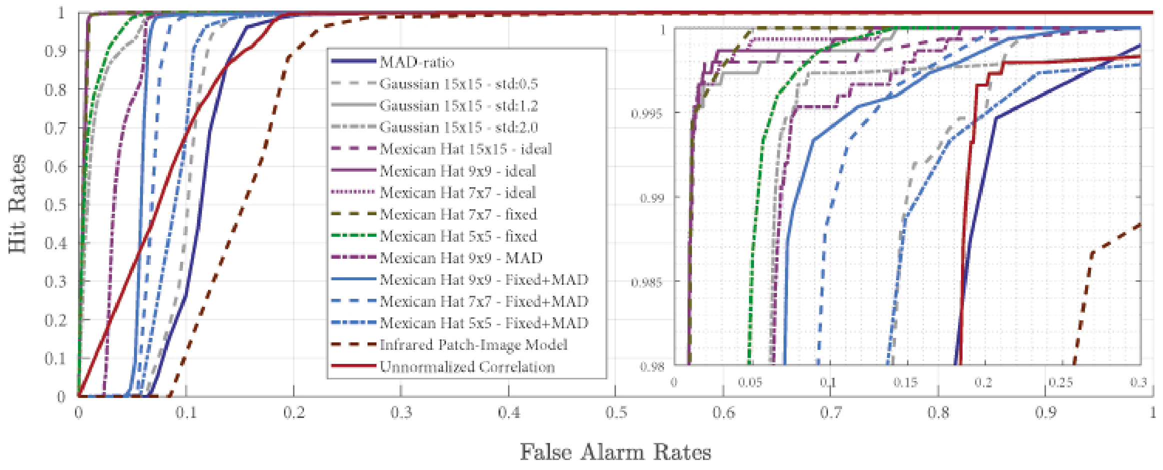
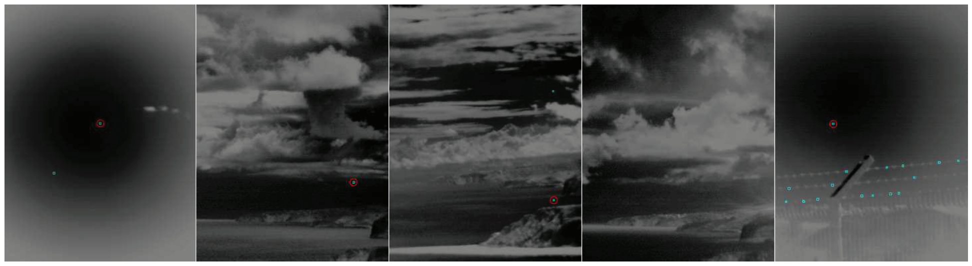
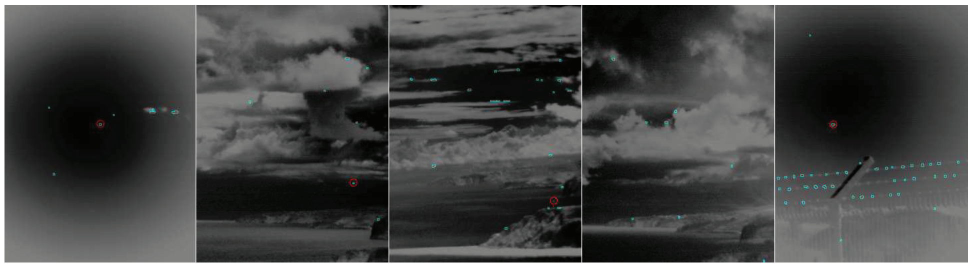


Figure 5. In (a), the result of one of the N=1 experiments can be seen both in 3D (left) and 2D (right) plots. In (b), the so-called ”hat” filter is depicted.



(a) Results of MexHat 9x9 fixed-MAD thr=20



(b) Results of MAD-ratio method

Figure 6. Comparative results for 5 different scenes with and without targets for two methods are depicted. In each image the target are designated by red circles, while the fourth image from left has no target. Each declaration is shown with a cyan rectangle. Thus, cyan rectangles not covered by a red circle are false alarms. In (a), the results of the the hat 9x9 MAD filter is given, while in (b) the results of the MAD-ratio method is provided.

5. Conclusions

We present a machine learning approach to the problem of infrared small target detection filter design. For this purpose, similar to a convolutional layer of a neural network, the normalized-cross-correlational (NCC) layer is proposed. This is a modified version of the convollutional layer of a neural network, in which both the input patch and the filter are normalized so that the convolution operation becomes identical to a correlation operation, if the filter is real and symmetric.

We work on a midwave band infrared small target dataset and train filters for detecting small targets and eliminating various types of clutter. Our benchmarking with different filters and a plain CNN shows that for our dataset, a single filter with a special shape (which we call ”the hat”) is sufficient to detect small targets and eliminate various types of clutter.

It is probable that this shape is only useful considering our dataset; thus, we can only utilize it for this IR band and a similar optics. However, we present a general framework to create this filter. That is why, for another dataset, the method can be applied and required filters can be obtained. As a future direction, we plan to obtain an extended target detection method and are working on creating a suitable dataset for this purpose.

References

[1] Zhang W, Cong M, Wang L. Algorithms for optical weak small targets detection and tracking: review. In: International Conference on Neural Networks and Signal Processing; Nanjing, China; 2003. pp. 643-647.

[2] Bai X, Zhang S, Du B, Liu Z, Jin T et al. Survey on dim small target detection in clutter background: wavelet, inter-frame and filter based algorithms. Procedia Engineering 2011; 15 (C): 479-483.

[3] Sanna A, Lamberti F. Advances in target detection and tracking in forward-looking infrared (FLIR) imagery. Sensors 2014; 14(11): 20297-20303.

[4] Warren RC. Detection of distant airborne targets in cluttered backgrounds in infrared image sequences. PhD, Engineering University of South Australia, Australia, 2002.

[5] Barnett J. Statistical analysis of median subtraction filtering with application to point target detection in infrared backgrounds. In: Infrared Systems and Components III. (SPIE); Los Angeles, CA, USA; 1989. pp. 10-18.

[6] Tom VT, Peli T, Leung M, Bondaryk JE. Morphology-based algorithm for point target detection in infrared backgrounds. In: Signal and Data Processing of Small Targets. (SPIE); Orlando, Florida, USA; 1993. pp. 2-11.

[7] Nitzberg R, Takken EH, Friedman D, Milton AF. Spatial filtering techniques for infrared (IR) sensors. In: Smart Sensors. (SPIE); Washington, D.C, USA; 1979. pp. 40-58.

[8] Schmidt WAC. Modified matched filter for cloud clutter suppression. IEEE Transactions on Pattern Analysis and Machine Intelligence 1990; 12 (6): 594-600.

[9] Wen P, Shi Z, Yu H, Wu X. A method for automatic infrared point target detection in a sea background based on morphology and wavelet transform. In: 3rd International Symposium on Multispectral Image Processing and Pattern Recognition. (SPIE); Beijing, China; 2003. pp. 248-253.

[10] Yang L, Yang J, Yang K. Adaptive detection for infrared small target under sea-sky complex background. Electronics Letters. 2004; 40 (17): 1083-1085.

[11] Bai X, Zhou F. Analysis of new top-hat transformation and the application for infrared dim small target detection. Pattern Recognition. 2010; 43 (6): 2145-2156.

[12] Bae TW, Zhang F, Kweon IS. Edge directional 2D LMS filter for infrared small target detection. Infrared Physics & Technology. 2012; 55 (1): 137-145.

[13] Deng H, Sun X, Liu M, Ye C, Zhou X. Small infrared target detection based on weighted local difference measure. IEEE Transactions on Geoscience and Remote Sensing. 2016; 54 (7): 4204-4214.

[14] Wang X, Lv G, Xu L. Infrared dim target detection based on visual attention. Infrared Physics & Technology 2012; 55 (6): 513-521.

[15] Kim S, Lee J. Scale invariant small target detection by optimizing signal-to-clutter ratio in heterogeneous background for infrared search and track. Pattern Recognition 2012; 45 (1): 393-406.

[16] Gao C, Meng D, Yang Y, Wang Y, Zhou X et al. Infrared patch-image model for small target detection in a single image. IEEE Transactions on Image Processing 2013; 22 (12): 4996-5009.

[17] Chen CP, Li H, Wei Y, Xia T, Tang YY. A local contrast method for small infrared target detection. IEEE Transactions on Geoscience and Remote Sensing 2014; 52 (1): 574-581.

[18] Han J, Ma Y, Huang J, Mei X, Ma J. An infrared small target detecting algorithm based on human visual system. IEEE Geoscience and Remote Sensing Letters 2016; 13 (3): 452-456.

[19] Wei Y, You X, Li H. Multiscale patch-based contrast measure for small infrared target detection. Pattern Recognition 2016; 58 (C): 216-226.

[20] Han J, Ma Y, Zhou B, Fan F, Liang K et al. A robust infrared small target detection algorithm based on human visual system. IEEE Geoscience and Remote Sensing Letters 2014; 11 (12): 2168-2172.

[21] Gregoris DJ, Yu SKW, Tritchew S, Sevigny L. Detection of dim targets in FLIR imagery using multiscale transforms. In: Infrared Technology XX. (SPIE); San Diego, CA, USA; 1994. pp. 62-71.

[22] Wang G, Zhang T, Wei L, Sang N. Efficient method for multiscale small target detection from a natural scene. Optical Engineering 1997; 35 (3): 761-768.

[23] Wang Z, Tian J, Liu J, Zheng S. Small infrared target fusion detection based on support vector machines in the wavelet domain. Optical Engineering 2006; 45 (7): 1-9.

[24] Kim S, Lee J. Scale invariant small target detection by optimizing signal-to-clutter ratio in heterogeneous background for infrared search and track. Pattern Recognition 2012; 45 (1): 393-406.

[25] Wei Y, You X, Li H. Multiscale patch-based contrast measure for small infrared target detection. Pattern Recognition 2016; 58 (C): 216-226.

[26] Deng H, Sun X, Liu M, Ye C, Zhou X. Entropy-based window selection for detecting dim and small infrared targets. Pattern Recognition 2017; 61 (C): 66-77.

[27] Qi S, Xu G, Mou Z, Huang D, Zheng X. A fast-saliency method for real-time infrared small target detection. Infrared Physics & Technology 2016; 77 (C): 440-450.

[28] Alam MS, Bhuiyan SMA. Trends in correlation-based pattern recognition and tracking in forward-looking infrared imagery. Sensors 2014; 14 (8): 13437–13475.

[29] Gundogdu E, Ozkan H, Demir HS, Ergezer H, Akagunduz E et al. Comparison of infrared and visible imagery for object tracking: Toward trackers with superior IR performance. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops; Boston, Massachusetts, USA; 2015. pp. 1-9.

[30] Shao L, Zhang H, de Haan G. An overview and performance evaluation of classification-based least squares trained filters. IEEE Transactions on Image Processing 2008; 17 (10): 1772-1782.

[31] Gundogdu E, Alatan AA. Good features to correlate for visual tracking. IEEE Transactions on Image Processing 2018; 27 (5): 2526-2540.

[32] Valmadre J, Bertinetto L, Henriques J, Vedaldi A, Torr PHS. End-to-end representation learning for correlation filter based tracking. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR); Honolulu, Hawaii, USA; 2017. pp. 5000-5008.

[33] Arora R, Basu A, Mianjy P, Mukherjee A. Understanding deep neural networks with rectified linear units. In: 6th International Conference on Learning Representations, ICLR, Vancouver, BC, Canada; 2018. pp. 1-6.

[34] Vedaldi A, Lenc K. MatConvNet: convolutional neural networks for MATLAB. In: 23rd ACM International Conference on Multimedia; Brisbane, Australia; 2015. pp. 689-692.

[35] Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick R et al. Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia; Orlando, Florida, USA; 2014. pp. 675-678.

[36] Kachhwal P, Rout BC. Novel square root algorithm and its FPGA implementation. In: 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT); Ajmer, Rajasthan, India; 2014. pp. 158-162.

[37] Shafer D. Infrared collimator system. Applied Optics 1993; 32(34): 7117-7118.

[38] Mudau AE, Willers CJ, Griffith D, le Roux FPJ. Non-uniformity correction and bad pixel replacement on LWIR and MWIR images. In: 2011 Saudi International Electronics, Communications and Photonics Conference (SIECPC); Riyadh, Saudi Arabia; 2011. pp. 1-5.