Robust Tracking of Vessels in Oceanographic Airborne Images

Jorge Matos
ISR/IST, Universidade de Lisboa
Av. Rovisco Pais 1, 1049-001 Lisboa
Email: jorge.p.s.matos@tecnico.ulisboa.pt

Alexandre Bernardino
ISR/IST, Universidade de Lisboa
Av. Rovisco Pais 1, 1049-001 Lisboa
Email: alex@isr.tecnico.ulisboa.pt

Jorge Salvador Marques
ISR/IST, Universidade de Lisboa
Av. Rovisco Pais 1, 1049-001 Lisboa
Email: jsm@isr.ist.utl.pt

***Abstract*—In this work we present and evaluate an algorithm for tracking vessels in oceanographic airborne image sequences. Such sequences are challenging due to sun reflections, wakes, wave crests and fast motions, which significantly degrade the performance of general purpose tracking algorithms. The proposed method is based on state-of-the-art correlation filter tracking complemented with an image segmentation and blob analysis stage. The purpose of this later stage is to re-center the target in the tracking window to compensate for drifts in the correlation filter. We evaluate our proposal using a known benchmark in the field and compare it with general purpose tracking algorithms. Results show that our method beats the general purpose state-of-the-art tracking algorithms in the airborne maritime scenario both in performance and in computation time. The dataset used to perform the evaluations was obtained during the SEAGULL project. We contribute with annotations to this dataset which is available online for further research.**

I. Introduction

The SEAGULL project [1] developed an intelligent maritime surveillance system using unmanned autonomous vehicles (UAVs) equipped with various types of optical sensors (visible, infrared, multi- and hyper- spectral). This system is particularly interesting because it is affordable, easy to deploy and with few infrastructure requirements, in contrast to other systems used today. A fleet of fixed wing UAVs are equipped with computers running vision algorithms for the automatic detection of maritime vessels, whose coordinates are communicated to a coastal ground station via a radio link. The developed algorithms work in real time on the embedded hardware and present a low rate of false detections [2]. Nevertheless, the developed methods have difficulties mainly in the presence of sun reflections, breaking waves and boat wakes. The focus of our work is the development of a tracking algorithm more robust to these effects and to perform comparisons with other state-of-the-art tracking algorithms in the maritime scenario.

The main contribution of the present work is the development of a new approach to maritime vessel tracking, combining an adaptive correlation filter [3] with a local re-detection step consisting of blob analysis for the correction of tracking offsets. The main idea consists in applying a detection step on the region of interest (ROI) to correct the target center when the conditions are favorable, i.e. when the boat is outside sun glare and boat wake regions (low background clutter). These conditions are detected at the blob analysis phase, using a few heuristics applied to the number, area and location of segmented blobs. Using this approach, it is possible to maintain the robustness of the correlation filter tracking, allowing it to keep the track during longer time spans. We compare the proposed approach with several state-of-the-art object tracking algorithms, using a dataset of video sequences acquired during the SEAGULL project [1]. This dataset is composed by thousands of annotated frames publicly available for further research.

The paper is organized as follows. In the next section, Sec. II, we present the current state-of-the-art in general purpose tracking algorithms and a few existing applications to maritime scenarios. Then, in Sec. III we present the proposed approach and its main components: the correlation filter, the features used, and the blob analysis step. In Sec. IV we describe the used datasets, the evaluation methods, and the experiments done to assess the performance of our methods in comparison to the state-of-the-art. Results are presented in Sec. V, illustrating the advantages of our methods both in tracking performance and computation time. Finally, Sec. VI summarizes the main conclusions of this work and refers to directions for future research.

II. State-of-the-art

Object tracking is one the most important and difficult tasks in computer vision. According to the Object Tracking Benchmark (OTB) [4], several aspects make visual tracking a very challenging task: illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutters, and low resolution. Annually, the state-of-the-art tracking algorithms are presented at the Visual Object Tracking (VOT) challenge [5]. In the 2015 competition, some of the top performers were based on correlation filter tracking using different kinds of approaches. Usually, correlation filters are designed to produce correlation peaks in the estimated target location while having low responses in the background. Although effective, the training of these methods was impractical for online tracking until the proposal of the Minimum Output Sum of Squared Error (MOSSE) filter [6]. Later developments extended MOSSE in a number of ways, such as the introduction of kernel methods by the Kernelized Correlation Filter (KCF) [3] tracker, and the Discriminative Scale Space

Tracker (DSST) [7]. Both of these methods used multiple channel HOG features [8] instead of the raw pixels used by MOSSE. Later, MUlti-Store Tracker (MUSTer) [9] proposed an approach using short-term and long-term tracking methods, working in a complementary manner. The correlation filter tracker (CFT) works as the short-term tracker while the long-term part is based on key-points and is used to locate the target when the CFT fails.

In the latest VOT competition, the top correlation filter trackers were the Spatially Regularized Correlation Filter (SRDCF) [10] and the DeepSRDCF [11], both from the same authors. They introduced a spatial regularization component in the optimization problem to penalize the correlation filter coefficients farther from the target. Recently, features based on convolutional neural networks (CNNs) [11] [12] have shown state-of-the-art results in various visual recognition tasks including visual tracking. The DeepSRDCF introduces the use of convolutional neural network multi-channel features to discriminate the target with the VGG-2048 [13] neural network used for image classification. More recently, [12] used the output of multiple layers of a CNN to train multiple correlation filters. The idea behind this approach is that early layers of CNNs have higher spatial resolution allowing for precise localization while the features from deeper layers capture more semantic information and are robust to significant appearance changes.

Other tracking methods that are not based in correlation filters also show state-of-the-art results, specially the ones based on deep learning. In [14] the authors propose the Multi-Domain Convolutional Neural Network (MDNet) tracker, winner of the 2015 VOT challenge. In this method, a CNN is pre-trained using a training set of tracking videos. The main aspect of this CNN is that the last layer is reinitialized at the beginning of each new video (domain-specific) while the deeper layers are updated online.

The Deep Learning Tracker (DLT) is proposed in [15]. A stacked denoising autoencoder is trained offline to learn robust generic image features. Online tracking is made using the encoder trained from the previous autoencoder as a feature extractor, and an additional classification layer. One interesting finding is that the filters in the first layer of the trained feature extractor resemble the Gabor filters for edge detection.

Other methods also present good results in many benchmarking criteria and are worth mentioning. The STRUCK [16] method uses a kernelized structured output support vector machine (SVM) with Haar features and histogram features for tracking. The MEEM tracker [17] proposes a multi-expert restoration scheme to address the problem of model drift in online tracking using a linear SVM. The ASMS tracker [18] is based on the popular mean-shift object tracking method. The authors propose various changes to address the problem of scale estimation while processing frames at a high rate.

Despite all work on visual tracking, very few methods address detection and tracking on airborne images of maritime scenarios. These scenarios present several specific challenges, the most significant ones due to sun reflections (illumination variations), waves and wakes originated by the vessel motion (background clutter), and the fast motion of the camera due to UAV maneuvers. Given the top-down perspective of the camera attached to the UAV, other problems exist, such as the in-plane and out-of-plane rotations. Usually, these types of rotations are not present in a motionless camera. The use of infrared cameras has been proposed to reduce the effects of sun reflections, waves and wakes. In [19] the author proposes a method to detect and classify maritime objects in infrared videos recorded from an autonomous platform. A fusion of three detection methods is made to generate hypotheses of possible boat locations in the image. The first is based on a track-before-detect algorithm using spatio-temporal integrated blob strength, the second exploits stable image regions and the third is based on tracking salient points of the image. Next, a two-stage-classification step with Support Vector Machines (SVMs) is performed to classify the vessels. In [20] it is proposed an image saliency method and entropy analysis to detect vessels on long wave infrared images.

When the algorithms have to run on limited computational resources on board the UAV, additional concerns must be taken in their development. A recent algorithm for boat detection [2] uses simple blob analysis, based on spatial and temporal constraints and is capable of operation in real-time on board an UAV. In [21] a binary classifier using simple features is used to classify a target as vessel or background from optical satellite images. The classifier has a cascade structure which rejects background clutter in the earlier stages to improve the computational performance. In [22] a complete system for maritime coastal surveillance is proposed. The boat detection is performed using a Haar-like classifier and a temporal filter. The tracking module uses a nearest neighbor policy with the Bhattacharyya distance between the HSV histograms, allowing for multiple target tracking.

Upon the analysis of the current state-of-the-art, the main concerns are the computational cost of the approaches that are usually the top contenders in the visual object tracking challenge. Furthermore, from our experience in the application of general purpose tracking methods to maritime scenarios, we have noticed frequent failures in tracking regions with sun reflections, waves and wakes. On longer sequences the tracking tends to drift, and approaches based on key-points fail due to the typical low resolution and lack of texture of the target. In this paper we propose a method that is able to mitigate many of these problems.

III. Methodology

The architecture of this system is shown in Fig. 1. The main components are a correlation filter module and a blob analysis module.

In this work we use a Kernelized Correlation Filter (KCF), following the work of [3]. The correlation filter is initialized by training it with an image patch cropped from the first frame around the target bounding box (BB), either using the raw image (grayscale or RGB) or using different kinds of features. In the present work we use raw image pixels, HOG features [8]

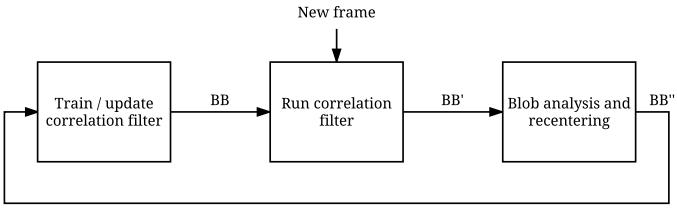

Fig. 1. System architecture. The system receives a new frame and it runs the correlation filter around the previous bounding box (BB) location returning a updated location of the target BB'. After, the error of this estimation (BB') is decreased using blob analysis to detect the vessel and recenter the bounding box (BB"). The correlation filter is updated with the information at the new target location.

and CNN features [23]. The area around this patch is then used as the region of interest (ROI) for the detection in the next frame. The computation of the correlations is performed in the frequency domain where convolutions are replaced with element-wise multiplications using the Discrete Fourier Transform (DFT) of the images. The response map, where the maximum corresponds to the estimated target location, is obtained using the inverse DFT. This information is used to update the location of the target in the new image and define a new bounding box around it (BB' in Fig. 1). However, due to clutter and noise, the new target estimated position may still be offset with respect to the true position. To improve the tracking accuracy a blob analysis step is performed. The image is segmented in its most representative blobs and if a dominant large blob is present, its centroid and area are used to update the location and scale of the target, thus defining a new bounding box (BB" in Fig. 1). Finally, the correlation filter is updated with the information from a new patch around the new detection. In the next sections we will detail the theoretical and practical aspects of the methods used in our implementation.

A. Correlation Filters

Correlations filters are specially tuned to a particular image pattern. They are reminiscent to Template Matching techniques in image processing, where a cropped patch around the target h is used to represent the filter. The correlation of this template with the new image x produces a response map y:

$$y = x * h^{-} \tag{1}$$

where h^{-} is the reflection in both coordinates of the patch and $*$ is the convolution operator. The location corresponding to the maximum value of the response map will indicate the new position of the target. To improve the computational efficiency of the filter, the convolution is computed in the Fourier domain. Because we are in a discrete and finite domain, the convolution must be replaced by the circular convolution ($\otimes$). This produces some boundary artifacts that can be mitigated by pre-weighting the patches with a windowing function. The computations become:

$$y = x \otimes h^{-} = \mathcal{F}^{-1}(\hat{x} \odot \hat{h}^{*}), \tag{2}$$

The hat symbol represents the discrete Fourier transform of a vector and the $\mathcal{F}^{-1}$ represents the inverse Discrete Fourier Transform. The $\odot$ represents the element-wise multiplication and superscript $*$ indicates the complex conjugate.

Using a simple cropped patch to represent the filter produces strong peaks to the target but also responds falsely to the background. To address this problem, methods have been proposed to learn filter coefficients h that produce a more desirable response convolution map y to a given input target x. Typically y is defined as an isotropic Gaussian function with a small standard deviation. A simple way to compute an exact filter is proposed in [24], using the Fourier domain:

$$\hat{h}^{*} = \frac{\hat{y}}{\hat{x}} \tag{3}$$

where the division is element-wise. Still, this approach is not very robust to noise and transformations other than pure translations, so [24] proposed the ASEF algorithm (Average of Synthetic Exact Filters) that, as the name suggests, averages multiple exact filters trained for different transformations of the target:

$$\hat{h}^{*} = \sum_{i} \frac{\hat{y}_i}{\hat{x}_i} \tag{4}$$

where the x_i are transformed image patches of the target, and y_i are the corresponding desired response maps (Gaussian centered in the location of the target).

One year later, the same author proposed the MOSSE filter (Minimum Output sum of Squared Error) that allows better performance than ASEF with a smaller number of training patterns [6]. The MOSSE correlation filter is the solution of:

$$\min_{\hat{h}^{*}} \sum_{i} ||\hat{x}_i \odot \hat{h}^{*} - \hat{y}_i||^2, \tag{5}$$

where i indexes each training sample (image). The solution of this optimization problem is given by:

$$\hat{h}^{*} = \frac{\sum_i \hat{y}_i \odot \hat{x}_i^{*}}{\sum_i \hat{x}_i \odot \hat{x}_i^{*}}. \tag{6}$$

This method allowed correlation filters to be trained more efficiently and be more robust to variations in lighting, scale, pose and non-rigid deformations.

B. Kernelized Correlation Filters

Despite the success of the MOSSE filter, it is only composed of linear operations on the input signals. Non-linear geometric or brightness transformations of the target can only be represented by increasing the training set of images, which make the method slower. In [3], it was proposed that the correlation filters can be extended to take advantage of the kernel trick to allow for non-linear operations. Lets assume the one-dimensional case, for a matter of clarity. This way it is possible to gain some intuition of the problem. The 2D case is more difficult to analyze but the properties used for the derivations on the 1D case are also true in the 2D case and the solution is true for both cases as stated by [3]. The correlation filter is formulated as a linear classifier such that the input x is mapped to the output label as $f(x) = \langle w, x \rangle$, where $\langle \cdot, \cdot \rangle$ represents the dot product. Using the Regularized Least

Squares (also known as Ridge Regression) the optimization problem becomes:

$$\min_{w} \sum_{j} (y(j) - f(x_j))^2 + \lambda ||w||^2, \tag{7}$$

where x_j is a transformation of the input and $y(j)$ the desired output (scalar), corresponding to the j^{th} entry of y, and λ is a regularization term. The solution is given by:

$$w = (X^H X + \lambda I)^{-1} X^H y, \tag{8}$$

where X is a matrix with one sample x_j per row, X^H means the Hermitian transpose $X^H = (X^*)^T$ and I is the identity matrix. The Hermitian transpose is used because the computations will be performed in the Fourier domain where values are usually complex.

By using the kernel trick, the tracking performance can be improved by making the classification on a high-dimensional feature space. The input data is mapped implicitly to a non-linear feature space by the function $\varphi(x)$, with the dot product defined by the kernel as $\kappa(x, x') = \langle \varphi(x), \varphi(x') \rangle$. Then, the Representer Theorem [25] states that the solution w is expressed by a linear combination of the inputs:

$$w = \sum_{i} \alpha_i \varphi(x_i) \tag{9}$$

and the solution to Eq. (7) is now:

$$\alpha = (K + \lambda I)^{-1} y, \tag{10}$$

where K is the kernel matrix with elements $K_{ij} = \kappa(x_i, x_j)$ and I is the identity matrix.

If the transformations in x_j are circular shifts, the computation of the inverse matrix in Eq. (10) can be avoided by the introduction of circulant matrices. A circulant matrix is defined by its first row, since all the other rows are all the possible shifts of the first row. It is usual to denote a circulant matrix as $U = C(u)$ where u is the first row of U.

Circulant matrices have various interesting properties that will allow the computations of the Ridge Regression solution to be computationally less expensive. For example, the sum, products and inverses of circulant matrices are circulant. Also, a circulant matrix has the following property:

$$U = F \text{diag}(\hat{u}) F^H \tag{11}$$

i.e. the circulant matrix can be made diagonal with the DFT matrix F, used to compute the DFT of a vector ($\mathcal{F}(u) = \sqrt{n} F u$) and the DFT of the base vector u [3].

By using this property on Eq. (8) the solution of w can be expressed in the following form in the Fourier domain

$$\hat{w} = \frac{\hat{x}^* \odot \hat{y}}{\hat{x}^* \odot \hat{x} + \lambda}, \tag{12}$$

where the division is element-wise. This solution replaced the matrix inversion by element-wise operations and the DFT which are computationally less expensive.

The solution above does not yet include the kernel trick. To use the circulant matrix method the kernel matrix K has to be circulant. For that to be true one condition has to be imposed – given circulant data $C(x)$, the Kernel matrix K is circulant if the kernel function satisfies $\kappa(x, x') = \kappa(Mx, Mx')$ for any permutation matrix M [3]. Some kernels that satisfy this condition are the Gaussian kernel and the linear kernel. In these cases $K = C(k^{xx})$, where k^{xx} is the kernel auto-correlation of the image x. The Gaussian kernel correlation is defined for some generic variables x_a and x_b as

$$k_g^{x_a x_b} = e^{-\frac{1}{\sigma^2}(||x_a||^2 + ||x_b||^2 - 2\mathcal{F}^{-1}(\hat{x}_a^* \odot \hat{x}_b))} \tag{13}$$

and the linear kernel correlation as

$$k_l^{x_a x_b} = \mathcal{F}^{-1}(\hat{x}_a^* \odot \hat{x}_b). \tag{14}$$

In this way, Eq. (10) can be written in the Fourier domain as

$$\hat{\alpha} = \frac{\hat{y}}{\hat{k}^{xx} + \lambda}, \tag{15}$$

where k^{xx} is obtained either using the Gaussian kernel (13) or the linear kernel (14) using the training sample x which replaces the generic variables x_a and x_b in those equations. Finally, with Eqs. 13, 14 and 15 it is possible to train the correlation filter.

Another important aspect of correlation filters is how to update the filter with the new information acquired with the new detection. Usually, the running average is the chosen method. In the Kernelized Correlation Filter the coefficients $\hat{\alpha}$ are updated as

$$\hat{\alpha}_t = \eta \frac{\hat{y}}{\hat{k}^{xx} + \lambda} + (1 - \eta) \hat{\alpha}_{t-1}, \tag{16}$$

where t defines the current step and η is the learning rate. The target model is updated as

$$\hat{x}_t = \eta \hat{x} + (1 - \eta) \hat{x}_{t-1}, \tag{17}$$

where x is the new sample extracted from the current estimated position of the target.

Given the trained parameter α, the base training samples x_t and the candidate patches for detection z, the detection can be made with the following confidence response map y':

$$y' = \mathcal{F}^{-1}(\hat{k}^{x_t z} \odot \hat{\alpha}) \tag{18}$$

by finding the location with maximum value in y', where $\hat{k}^{x_t z}$ is the DFT of the kernel correlation defined on Eq. (13) or Eq. (14) with the generic variables x_a and x_b replaced by x_t and z respectively.

There is one problem of working with images and the Fourier transform. The images have discontinuities in their borders and due to the periodic nature of the DFT this gives origin to leakage in the Fourier representation of the image. In signal processing, to alleviate this problem, it is usual to use a windowing function to weight the signal and smooth the boundary discontinuities. We choose the Hanning window to weight the image and alleviate the boundary effects.

The training part of the correlation filter has the following steps:

Fig. 2. From left to right: the region of interest x, the desired output as a 2D Gaussian y and the corresponding correlation filter coefficients α of the first frame, assuming the use of raw image pixels as features, obtained using Eq. (15) and a Gaussian kernel.

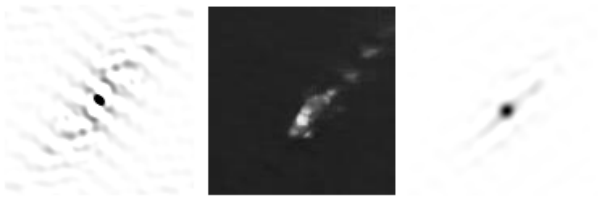

Fig. 3. From left to right: the correlation filter coefficients α trained and updated throughout 226 frames, the candidate patch for detection z after 227 frames since the initialization, and the response map y'. Using the trained correlation filter it is possible to estimate the translation of the target relative to the previous position using Eq. (18)

1: Weight x with a Hanning window
2: Compute $\hat{k}_g^{xx} = \mathcal{F}(e^{-\frac{1}{\sigma^2}(||x||^2+||x||^2-2\mathcal{F}^{-1}(\hat{x}^*\odot\hat{x}))})$
3: Update $\hat{\alpha}_t = \eta\frac{\hat{y}}{\hat{k}^{xx}+\lambda} + (1-\eta)\hat{\alpha}_{t-1}$
4: Update $\hat{x}_t = \eta\hat{x} + (1-\eta)\hat{x}_{t-1}$

and the detection part of the correlation filter has the following steps:

1: Weight z with a Hanning window
2: Compute $\hat{k}_g^{x_t z} = \mathcal{F}(e^{-\frac{1}{\sigma^2}(||x_t||^2+||z||^2-2\mathcal{F}^{-1}(\hat{x_t}^*\odot\hat{z}))})$
3: Compute $y' = \mathcal{F}^{-1}(\hat{k}^{x_t z}\odot\hat{\alpha}_{t-1})$
4: Determine arguments of the maxima of y'

In Fig. 2 and Fig. 3 it is possible to get a better understanding of these mathematical concepts during training and detection respectively, by analyzing the image patch and the color maps of the desired output and the obtained correlation filter.

C. Features

Correlation filters work with any kind of dense features that maintain the spatial information. Examples of these features that have shown good performance are raw image pixels, Histogram of Oriented Gradients (HOG), Color Names (CN) [26], and more recently Convolutional Neural Network (CNN) features. In this work raw image pixels, HOG and CNN are used. When using raw gray pixels the computations are the ones given in the previous section, but for multiple channel features such as RGB, HOG or CNN the kernel correlation has a slight difference. For both the linear and the Gaussian kernel the vectors from different channels can be simply added together:

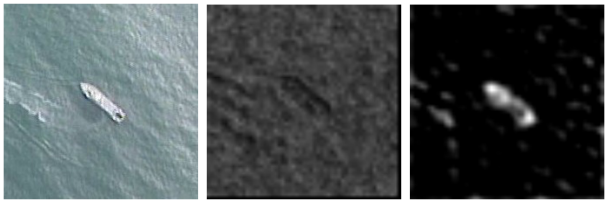

Fig. 4. From left to right: original image, one channel of the FHOG features from a total of 31 and one channel from the CNN features from a total of 256 channels.

$$k_l^{x_a x_b} = \mathcal{F}^{-1}\left(\sum_c \hat{x}_{a,c}^* \odot \hat{x}_{b,c}\right) \tag{19}$$

$$k_g^{x_a x_b} = e^{-\frac{1}{\sigma^2}(||x_a||^2+||x_b||^2-2\mathcal{F}^{-1}(\sum_c \hat{x}_{a,c}^*\odot\hat{x}_{b,c}))}, \tag{20}$$

where c identifies the channels.

The HOG features used are the ones proposed by Felzenszwalb et al. [8] and are sometimes called Felzenszwalb HOG (FHOG) features. These features are composed of 31 channels where 9 are contrast insensitive, 18 are contrast sensitive and 4 capture the overall gradient energy in different areas. Fig. 4 shows an example of one channel of the resulting 31 channels after processing a maritime vessel.

The VGG-Net [23] won the first and second place in the ImageNet ILSVRC-2014 localization and classification tasks respectively. In the VGG-Net, a preprocessed image is given as input and this image will go through 19 layers that are composed of convolutional layers with the rectification (ReLu) non-linearity, max-pooling and fully connected layers. The last layer gives the confidence of an image being one of the 1000 different classes. The appearance of the images is encoded in the neural network weights and the activations of the convolutional hidden layers can be used as multiple channel features for the correlation filter. We feed the image to the CNN and use the activations of selected convolutional hidden layers as features.

Note that different layers or even different convolutional neural networks can be used to extract features and the results can vary with the choice.

When using FHOG, [3] states that the Gaussian kernel has better results, as we confirmed experimentally. When testing the CNN features, our results show that the linear kernel achieves a better performance than the Gaussian kernel. Consequently, when using CNN features, we changed the algorithms to use the linear kernel correlations k_l^{xx} and $k_l^{x_t z}$.

D. Blob Analysis

The results in [2] show interesting properties of a blob analysis framework in tracking boats in maritime images. Because the appearance of a boat on a maritime background has a blob-like appearance, we propose the use of image segmentation and blob analysis to detect the maritime vessel in the region of interest (instead of the whole image) and to correct the correlation filter tracker.

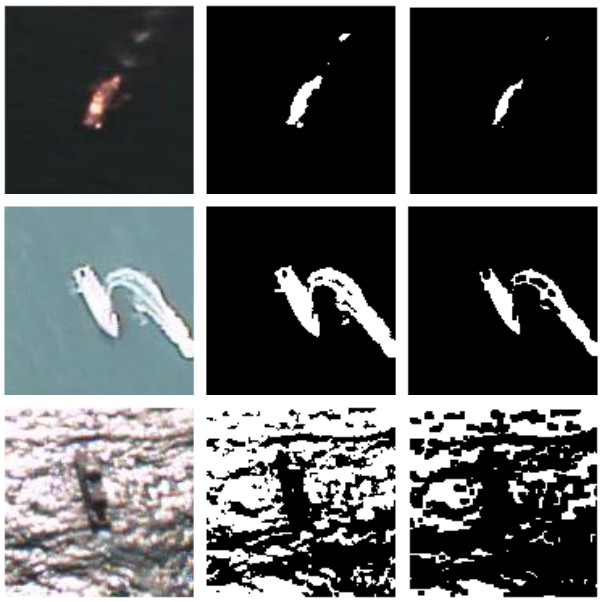

Fig. 5. Original image, segmented image and eroded image (left to right). Three examples shown: a regular case, a case with irregular wake and a case with sun reflections (top to bottom).

Image Segmentation. Before the detection of the connected components, usually referred to as blobs, the image patch of interest has to be segmented. In this setting the images are usually composed of two main components: i) vessels and ii) the ocean. Thus, we adopt a binarization approach via the Otsu's method [27] to separate bright and dark parts of the image that, in principle, will correspond to the boats and the ocean surface, respectively.

The Otsu's method finds a threshold that separates the two peaks of the image histogram from a bimodal image. In mathematical terms, the algorithm exhaustively searches for the threshold t that minimizes the intra-class variance (the variance within the class), defined as a weighted sum of the variances of the two classes:

$$\sigma^2_\omega(t) = \omega_0(t)\sigma^2_0(t) + \omega_1(t)\sigma^2_1(t), \tag{21}$$

where $\omega_{0,1}$ are the class probabilities separated by the threshold t and $\sigma^2_{0,1}(t)$ are the class variances.

The segmented image is then eroded (Fig. 5) to isolate the vessel from nearby distractors like wakes, life rafts or other vessels, and also to remove some of the noise originated from waves and sun reflections.

Blob Detection. Sometimes, the background clutter (waves and wakes) and the sun reflections can interfere with the segmentation. In these cases, the blob detection cannot be used to correct the estimation of the correlation filter because the segmentation failed to separate the vessel from the ocean. Some examples of both accurate and failed segmentations can be seen in Fig. 5.

After analyzing a set of problematic situations where the image segmentation has poor results, we propose the use of context analysis (set of heuristics) to determine if the vessel is going through a region that is challenging for the image segmentation method. The segmented images of vessels going through regions with sun reflections usually have the two following characteristics:

1) The number of blobs present in the segmented image is higher than the general case;
2) There are blobs touching the border of the segmented image.

Also, when the vessels are traveling at high speeds, the wakes are bigger than in the general case and are also highly irregular. In this case the second statement is also true. The last problematic situation is the appearance of small blobs that might belong to regions of small waves, wakes or sun reflections.

With all the previous information in mind we defined the following heuristic rules to determine if the conditions near the vessel are favorable for image segmentation:

1) Are there blobs touching the border?
2) Is the number of blobs higher than a threshold T_n?
3) Does the chosen blob have an area smaller than a threshold T_a?

The two first conditions filter the presence of sun reflections. The first condition filters the presence of highly irregular waves originated from vessels traveling at high speeds. The last one filters false detections of small waves, wakes and reflections. If none of these conditions are met, we correct the track to the center of the chosen blob (see next paragraph). Otherwise we accept the correlation filter estimated location without correcting its position.

To detect the group of pixels (blob) that corresponds to the maritime vessel, we use a method to find the contours of connected components in binary images proposed by [28]. Then, we can find the center of mass and area of the detected blobs using their contours and image moments. We propose to correct the track to the blob that is closest to the KCF estimated location of the target. This is called nearest neighbor approach and corresponds to solving a nearest neighbor search problem.

One of the main challenges of current tracking methods is how to adjust the bounding box to the target that has rotated or suffered appearance changes. Using the contour of the target we can adjust a up-right rectangular bounding box to the target. In practice, since we eroded the segmented image, we compensate this by increasing the size of the obtained bounding box by 25%. This has the added effect of adding more context around the target which increases performance of the correlation filter.

The complete set of steps of the proposed method for one iteration (one frame) is as follows:

1: **if** first frame **then**
2: Train correlation filter
3: **else**
4: Estimate translation with the correlation filter
5: Perform binary image segmentation around the location estimated by the correlation filter

if no blobs touch the border **and** the number of blobs is smaller then the threshold T_n **then**
  Determine the blob that is closest to the correlation filter estimated location
  if the blob's area is higher than a threshold T_a **then**
    Set the track to the center of that blob
    Set new width and height to 1.25 the size of the bounding box adjusted to the blob contour.
  end if
end if
Update the correlation filter
end if

IV. Experiments

A. Evaluation Metrics

To perform the evaluation of the proposed method and compare it to state-of-the-art tracking algorithms, the Object Tracking Benchmark (OTB) [4] framework was used. This methodology evaluates the tracking methods by computing Precision and Success plots of the tracking under two different initialization strategies denoted Temporal Robustness Evaluation (TRE) and Spatial Robustness Evaluation (SRE).

The Precision plot is based in center location error, defined as the Euclidean distance between the center locations of the tracked targets and the manually labeled ground truths. The most usual approach is to determine the percentage of frames whose estimated location error is below a given threshold. This percentage is computed for an interval of threshold values (0 to 50 pixels). The score chosen to rank the trackers is percentage value for a threshold of 20 pixels as suggested by [4].

The Success plot evaluates the bounding box overlap of the detection with the ground truth. Mathematically, given a detected bounding box r_d and the ground truth bounding box r_{gt} the overlap ratio is given by:

$$S = \frac{|r_d \cap r_{gt}|}{|r_d \cup r_{gt}|}, \tag{22}$$

where $\cap$ and $\cup$ represent the intersection and union, respectively, and $|.|$ represents the number of pixels in that region. The Success plot shows the percentage of frames whose bounding box overlap ratio is higher than a given threshold for threshold values from 0 to 1, with steps of 0.05, where 1 means perfect match of the detection and ground truth and 0 meaning lost target. To rank the different algorithms, the area under the curve (AUC) of each Success plot is used.

The Temporal Robustness Evaluation consists in initializing the trackers at different frames, not just the first, and running them until the end of the sequence. Each sequence is evaluated by initializing in 20 different frames. The initial frames are chosen by starting with the first frame of the sequence and stepping through them at a regular interval. The step is approximately the number of frames of the sequence divided by 20.

The Spacial Robustness Evaluation consists in introducing error in the initialization by shifting the bounding box by 10% of the target size in 8 different directions, and scaling it by 0.8, 0.9, 1.1 and 1.2 of the ground truth size. This results in 12 different initializations.

These evaluations are pertinent because in a real world scenario the trackers would be initialized with a vessel detector that is likely to introduce error in the initialization.

TABLE I
Description of all sequences of the dataset.

Video	Description	Frames	Date
TASE	A medium and a small vessel. Irregular wakes and waves. Intense sun reflections.	1685	21-04-2015
JAI 1	A medium and a small vessel. Irregular wakes and waves. Intense sun reflections.	4384	22-04-2015
JAI 2	A medium sized vessel. Irregular wakes and waves. Intense sun reflections.	297	22-04-2015
JAI 3	A medium sized vessel. Irregular wakes and waves.	101	22-04-2015
JAI 4	A medium and a small vessel. Irregular wakes and waves. Intense sun reflections.	2280	22-04-2015
GOBI 1	Medium sized vessel. Fast motion.	1718	22-04-2015
GOBI 2	Medium sized vessel. Fast motion.	704	02-06-2015

B. Dataset

The dataset used for the evaluation is composed of two different videos of the visible spectrum, one acquired with a JAI's AD-080GE camera and the other with a TASE150 camera, and two videos from the Long-Wave Infrared (LWIR) spectrum obtained with a GOBI-384 camera. These videos were obtained in three different days in the coastal area of Portimão. Given the limitations of the OTB framework on evaluating videos with out-of-view targets, these videos were cut into smaller sequences. In Table I a small description of each sequence is presented. In total, the tracking algorithms run on 8747 manually annotated frames of the visible spectrum and 2422 of the LWIR spectrum. These sequences include cases where a vessel goes through regions of sun reflections, cases of highly irregular wakes, and cases where the target deploys smaller vessels and life rafts (see Table I).

These and more annotations can be accessed online[1] for further research and validation of the results presented below. See Fig. 6 for some examples of frames from the dataset.

C. Tested Algorithms

Some of the state-of-the art methods mentioned above that were available online for testing purposes were evaluated using the OTB framework. The tested trackers were ASMS [18], DSST [7], KCF [3], CF2 [29], SRDCF [10], MUSTer [9], MEEM [17], MDNet [14] and the proposed method either using HOG features or CNN features (12th layer) in the visible spectrum and raw image features and CNN features (12th layer) in the LWIR spectrum.

[1] http://vislab.isr.ist.utl.pt/seagull-dataset/

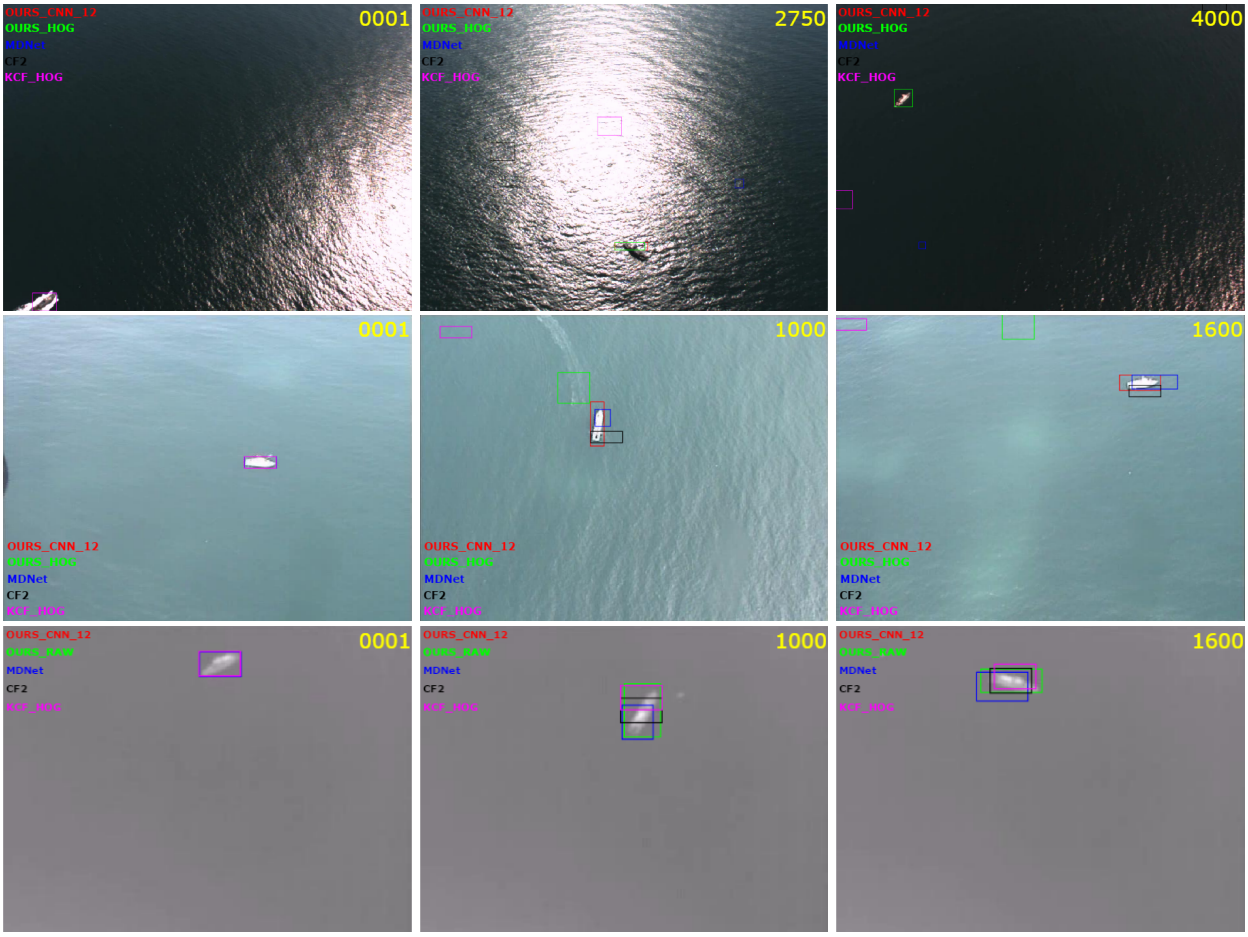

Fig. 6. Frames from the JAI 1, TASE and GOBI 1 sequence (top to bottom) with the results, as bounding boxes, of some of the best performing methods: our methods with different features (red and green), MDNet (blue), CF2 (black) and KCF with HOG (pink). In the first frame (left column) all trackers are initialized by the same BB which means only one is visible (KCF).

V. Results

The implementation of the proposed method was made in Python and can be accessed online[2]. To evaluate this method, the parameters used for the correlation filter and blob analysis were as follows. The region of interest is defined as a square 2.5 times larger than the target bounding box which defines the M and N values. The desired output y for the training step is a M by N Gaussian with its peak at the target center:

$$y(m,n) = e^{-\frac{(m-M/2)^2+(n-N/2)^2}{2\sigma^2}}, \tag{23}$$

where $m = 0, 1, 2, ..., M$, $n = 0, 1, 2, ..., N$, and $\sigma = 0.1 \cdot \sqrt{a}$ is the peak width, proportional to target's bounding box area a. The regularization weight λ used was 10^{-4} and the learning rate η was set to 0.02. When using the HOG features, the kernel used was the Gaussian kernel, with a standard deviation of 0.5, while the linear kernel was used for the CNN features.

The FHOG features were extracted using the Dlib [30] implementation while the CNN features were extracted using the Caffe [31] framework.

[2]https://github.com/Magnesiam/CFT_OTB/

For the image segmentation method [27] and the blob detector [28], we use the OpenCV Library [32] implementations. The threshold used for the number of blobs was $T_n = 2$ and the minimum size threshold was $T_s = 80$ pixels.

Visible spectrum. As can be seen in Fig. 7, our method, either using FHOG or CNN features, shows better results than the other methods in the airborne maritime scenario. The key is the blob detection and target re-centering steps, which allow greater precision, especially with the version using CNN features, either in the Spatial Robustness Evaluation (SRE) or in the Temporal Robustness Evaluation (TRE). The FHOG version also presents competitive results when compared to the MDNet (winner of the VOT Challenge 2015) but running with a frame-rate that is two orders of magnitude faster (Table II). Some trackers that have great results in general purpose tracking challenges and benchmarks seem not to generalize well enough for the particular case of airborne maritime imagery, such as the ASMS and the DSST. The DSST, even tough being a part of the correlation filter tracking family, does not use the kernel trick and the scale space search does not work well for the rotations present in this scenario.

In the Success rate evaluation there are clearly three top

ranking algorithms: ours and MDNet. This is because most trackers do not adapt the bounding box to the tracked target. Usually the bounding box width and height stay constant. In the DSST and ASMS the scale change is estimated but the aspect ratio (width/height) stays the same. In our setting, when the target changes direction, its aspect ratio changes due to the top down perspective. The only trackers that are able to adjust the bounding box to the target are both our methods and the MDNet. This explains why they show better results than the others in the Success evaluation.

LWIR spectrum. We evaluate our method using raw image pixels since they have better results in this spectrum compared with the FHOG features, as shown by our preliminary tests. This is the case because, in this spectrum, the challenges of sun reflections and irregular waves and wakes are not present. We also evaluate using the CNN features (12th layer).

In Fig. 8 we show the results of our methods and other state-of-the-art tracking methods in the LWIR spectrum. Our methods, either using raw image pixels or CNN features, outperform the other methods in every evaluation. This is the case because in this spectrum the blob analysis step is specially robust due to the Otsu's method working very well for bimodal images (which is the case when the target is in the field-of-view). The other methods rank similarly with the visible spectrum relative to one another, but in absolute terms they have better performance in the LWIR as expected due to the lack of the most challenging situations present in the visible spectrum such as sun reflections and wakes.

In this spectrum, both our methods have very similar results in all thresholds. This happens because the LWIR spectrum is a less challenging scenario compared with the visible spectrum and the more complex preprocessing of the CNN is not so useful. We conclude that the choice of image features is not as significant in the LWIR spectrum compared with the visible spectrum.

Another important factor to take into account is the computation complexity. These evaluations were made using an Intel Xeon CPU W3503 at 2.40GHz and a GeForce GTX 750 graphics card. The CNN computations were run in parallel in the graphics card. Two of the three top methods, ours using CNN features and MDNet, run at a low frame rate. Ours, with CNN features, has an average frame rate of 2.39 FPS and MDNet has an average of 0.37 FPS. The best performing approach that is suitable to be used in a real-time system is the method proposed in this work which uses FHOG features in the visible spectrum and raw image pixels in the LWIR spectrum. Ours with FHOG averages 32.77 FPS in the visible spectrum and ours with raw image pixels averages 45.02 FPS in the LWIR spectrum.

A video with some results of the top tracking methods on the video sequences of our dataset can be viewed online[3].

TABLE II
FRAME RATE OF THE EVALUATED TRACKING METHODS.

	FPS (Visible)		FPS (LWIR)		
Method	**TRE**	**SRE**	**TRE**	**SRE**	**Average**
Ours_CNN_12	2.01	2.17	2.41	2.95	2.39
Ours_HOG	34.33	31.20	NA	NA	32.77
Ours_RAW	NA	NA	51.93	38.10	45.02
MDNet	0.37	0.39	0.36	0.37	0.37
CF2	4.81	4.11	5.18	5.05	4.79
KCF_HOG	205.17	171.94	139.46	197.11	178.42
ASMS	75.00	67.39	165.40	360.24	167.01
DSST	28.60	24.39	24.34	25.24	25.64
MEEM	9.11	8.63	10.63	11.46	9.97
SRDCF	6.97	5.39	7.02	7.36	6.69
MUSTer	1.17	1.00	4.54	5.01	2.93

VI. CONCLUSIONS

In this work we have presented and benchmarked a new algorithm for tracking that performs beyond the state-of-the-art in airborne maritime scenarios. Some of the problems concerning the detection and tracking of vessels in airborne oceanographic imagery, namely sun reflections, waves, wakes and long-term tracking, are overcome thanks to a combination of correlation filters and blob analysis. We present two versions of the algorithms. The best performing is based on CNN features and significantly outperforms current state-of-the-art general purpose trackers. The second version uses HOG features (raw in the LWIR spectrum) and attains results competitive with the current state-of-the-art, but at a much faster frame rate, suitable for real-time operation on airborne systems.

For future work it is proposed the fusion of a detection module with the tracking step proposed here to allow for a completely autonomous system. Also, improvements on the bounding box or target segmentation could increase the precision and success rate of the algorithm.

[3]https://www.youtube.com/watch?v=oWQ5CuXC5DA

ACKNOWLEDGMENTS

This work was partially supported by project PTDC/EEIPRO/0426/2014 and the Portuguese Government through FCT project [UID/EEA/50009/2013].

REFERENCES

[1] M. M. Marques, P. Dias, N. P. Santos, V. Lobo, R. Batista, D. Salgueiro, A. Aguiar, M. Costa, J. E. da Silva, A. S. Ferreira, J. Sousa, M. de Fátima Nunes, E. Pereira, J. Morgado, R. Ribeiro, J. S. Marques, A. Bernardino, M. Griné, and M. Taiana, "Unmanned aircraft systems in maritime operations: Challenges addressed in the scope of the seagull project," in *OCEANS 2015 - Genova*, May 2015, pp. 1–6.

[2] J. S. Marques, A. Bernardino, G. Cruz, and M. Bento, "An algorithm for the detection of vessels in aerial images," in *Advanced Video and Signal Based Surveillance (AVSS), 2014 11th IEEE International Conference on*. IEEE, 2014, pp. 295–300.

[3] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *Pattern Analysis and Machine Intelligence, IEEE Trans.*, vol. 37, no. 3, pp. 583–596, 2015.

[4] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[5] "The visual object tracking vot2015 challenge results," Dec 2015. [Online]. Available: http://www.votchallenge.net/vot2015/

[6] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2544–2550.

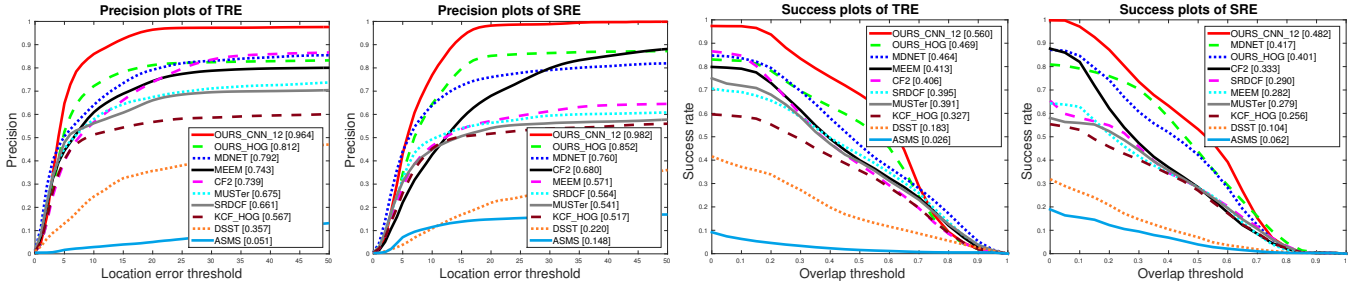

Fig. 7. Visible spectrum. Precision plots of the SRE and the TRE with our methods and state-of-the-art tracking methods. The values on the legend correspond to the precision for a location error threshold of 20. Success plots of the SRE and the TRE with our methods and state-of-the-art tracking methods. The values on the legend correspond to the area under the curve.

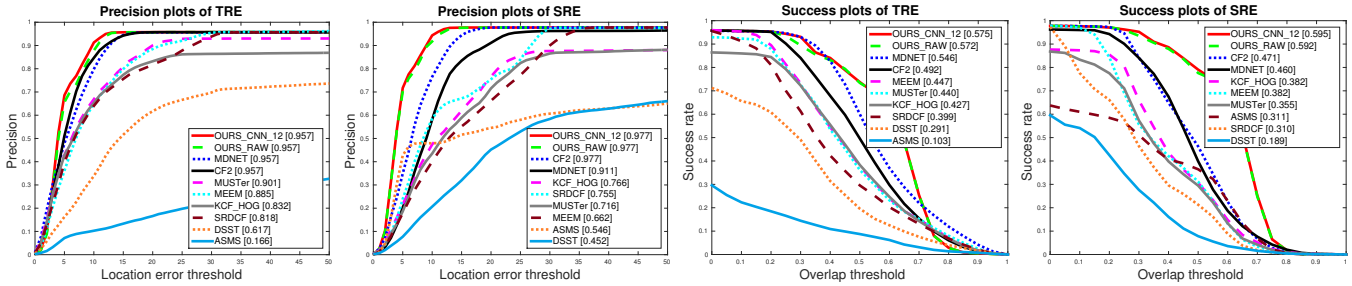

Fig. 8. LWIR spectrum. Precision plots of the SRE and the TRE of the KCF tracker using different features. The values on the legend correspond to the precision for a location error threshold of 20. Success plots of the SRE and the TRE of the KCF tracker using different features. The values on the legend correspond to the area under the curve.

[7] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[9] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 749–758.

[10] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4310–4318.

[11] M. Danelljan, G. Hager, F. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the IEEE Int. Conf. on Computer Vision Workshops*, 2015, pp. 58–66.

[12] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[13] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.

[14] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," *arXiv preprint arXiv:1510.07945*, 2015.

[15] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 809–817.

[16] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. Hicks, and P. Torr, "Struck: Structured output tracking with kernels," 2014.

[17] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: robust tracking via multiple experts using entropy minimization," in *Proc. of the European Conference on Computer Vision (ECCV)*, 2014.

[18] T. Vojir, J. Noskova, and J. Matas, "Robust scale-adaptive mean-shift for tracking," *Pattern Recognition Letters*, vol. 49, pp. 250–258, 2014.

[19] M. Teutsch and W. Krüger, "Classification of small boats in infrared images for maritime surveillance," in *2010 International WaterSide Security Conference*. IEEE, 2010, pp. 1–7.

[20] G. Cruz and A. Bernardino, "Image saliency applied to infrared images for unmanned maritime monitoring," in *International Conference on Computer Vision Systems*. Springer, 2015, pp. 511–522.

[21] G. Mattyus, "Near real-time automatic vessel detection on optical satellite images," in *ISPRS Hannover Workshop*. ISPRS Archives, 2013, pp. 233–237.

[22] D. Bloisi, L. Iocchi, M. Fiorini, and G. Graziano, "Automatic maritime surveillance with visual target detection," in *Proc. of the International Defense and Homeland Security Simulation Workshop (DHSS)*, 2011, pp. 141–145.

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[24] D. S. Bolme, , B. A. Draper, and J. R. Beveridge, "Average of synthetic exact filters," in *Computer Vision and Pattern Recognition (CVPR), 2009 IEEE Conference on*. IEEE, 2009, pp. 2105–2112.

[25] B. Schölkopf and A. J. Smola, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[26] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1090–1097.

[27] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[28] S. Suzuki *et al.*, "Topological structural analysis of digitized binary images by border following," *Computer Vision, Graphics, and Image Processing*, vol. 30, no. 1, pp. 32–46, 1985.

[29] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3074–3082.

[30] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[31] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[32] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.