EURASIP Journal on Image and Video Processing
a SpringerOpen Journal

RESEARCH **Open Access**

Detection and tracking of moving objects in a maritime environment using level set with shape priors

Duncan Frost and Jules-Raymond Tapamo*

Abstract

Over the years, maritime surveillance has become increasingly important due to the recurrence of piracy. While surveillance has traditionally been a manual task using crew members in lookout positions on parts of the ship, much work is being done to automate this task using digital cameras coupled with a computer that uses image processing techniques that intelligently track object in the maritime environment. One such technique is level set segmentation which evolves a contour to objects of interest in a given image. This method works well but gives incorrect segmentation results when a target object is corrupted in the image. This paper explores the possibility of factoring in prior knowledge of a ship's shape into level set segmentation to improve results, a concept that is unaddressed in maritime surveillance problem. It is shown that the developed video tracking system outperforms level set-based systems that do not use prior shape knowledge, working well even where these systems fail.

Keywords: Tracking; Level sets; Shape priors; Maritime surveillance

1 Introduction

While the word 'pirate' brings to mind thoughts of the swashbuckling, one-eyed seafarers of childhood fantasy, the term still, unfortunately, has use in today's modern world. Costing an estimated US$13 to 16 billion a year [1], piracy remains a pertinent problem in areas such as the coast of Somalia and the Gulf of Guinea. Despite increased security, piracy in these areas is increasing over the years. While recent years have seen a slight drop in reported incidents of piracy, 439 attacks were reported in 2011 according to the International Maritime Bureau [2].

Due to the increased threat of piracy, surveillance is an absolute must on cargo ships travelling in these dangerous areas. While radar systems have been extensively used in maritime environments, these generally require large, metallic targets. Modern pirates favour small, fast rigid inflatable boats that are mainly non-metallic and thus difficult to detect [3]. While the solution to this would seem to be the use of manual detection using dedicated crew members on board, the small number present at any given time makes this unfeasible. Unlike humans that grow tired, automated video surveillance systems are able to constantly monitor camera feeds and keep track of a number of objects of interest around the ship.

Szpak and Tapamo [4] present an approach that attempts to track objects using a closed curve in the image (a method known as level set segmentation) after they had been detected using a motion-based detection system. While the tracking results are very good, object detection suffers from detection of a large number of non-ship objects due to motion from waves. To address these shortcomings, this work investigates the possibility of integrating prior-known shape information into segmentation.

The rest of the paper is structured as follows: Section 2 covers the background on level set methods and their implementation and reviews the theory associated with shape priors. Section 3 discusses applications of image processing and level sets within the maritime surveillance environment. Section 4 introduces the proposed video tracking system and details various subsystem functionalities. The various subsystems are implemented in Section 5, and the final system is established. Section 6 concludes and outlines future work.

*Correspondence: tapamoj@ukzn.ac.za
School of Engineering, University of KwaZulu-Natal, Durban 4041, South Africa

2 Background

There have been a number of attempts to address the problem of detecting and tracking objects at sea. Some of the main tasks to achieve this goal are understanding of the nature of the sea clutter and its modelisation for accurate segmentation of moving objects.

In [5], temporal characteristics of sea clutter and that of a range of small boats are analysed using a comprehensive set of recorded datasets. This is done in an attempt to understand the dynamics and associated reflexivity of small boats. An empirical sea model is then derived. It then allows the development of advanced detection and tracking algorithms, which will help improve the performance of surveillance and marine navigation radar against small boats.

Vicen-Bueno et al. [6] propose neural networks-based signal processing techniques to reduce sea clutter. In [7], several machine vision techniques that could help in easing the search tasks in maritime environment are investigated. Hidden Markov model-based tracking models are then used to design a system that improves detection performance.

In this paper we propose an approach to ship detection and tracking based on active contours. Active contour methods are segmentation techniques that use an iteratively evolving contour or interface in an image that separates different regions of interest. Active contours can be expressed using one of two principle approaches [8]:

- Explicit or Lagrangian approach resulting in an interface known as *snake*.
- Implicit or Eulerian approach resulting in an interface called a *level set*.

Kass et al. [9] initially introduced the concept of active snakes for expressing the contour in an image in which a parameterised spline is guided in the image to a desirable position by a number of forces. The major problem with active snakes is its inability to deal with changes in topology [10]. Closed contours expressed as active snakes are unable to deal with splitting or merging of regions in an image. Level set methods were originally introduced by Osher and Sethian [11] as a method to evolve a contour with a speed proportional to its curvature. The main advantage of this method is that, unlike active contours, it allows for cusps, corners and automatic topological changes [10].

Given an image $I(x, y)$, where (x, y) are image coordinates, a three-dimensional surface defined by a level set function $\Phi(x, y)$ is defined on top of it. The contour C is in the image and implicitly defined as the zeroth level set (hence the name) of the level set function ϕ:

$$C = \{(x, y)|\phi(x, y) = 0\}. \tag{1}$$

To visualise this expression, see Figure 1 where a contour in the image shown in Figure 1a is defined as part of the image in Figure 1b that is cut by the level set surface at $\Phi = 0$.

To ensure a one-to-one mapping between the level set function and its corresponding contour, the level set function Φ is constrained to a signed distance function, that is, $|\nabla\Phi| = 1$ almost everywhere with $\Phi > 0$ inside the contour and $\Phi < 0$ outside the contour [12]. Informally, this can be thought of as the distance to the zero-level contour inside the contour and its additive inverse outside.

2.1 Chan-Vese energy functional

The original implementation introduced in [11] attempts to simulate active snake behaviour using a level set function's contour. The contour is evolved according to active-snake-like forces F in the normal direction to the contour (in the image) to simulate the behaviour of active snakes. To do so,the following differential equation is used:

$$\frac{\partial\Phi}{\partial t} = F|\nabla\Phi| \tag{2}$$

Rather than using a predefined set of forces in the image, variational level set methods seek to produce a level set function that minimises a predefined cost function, more specifically known as energy functional also sometimes called a *cost functional*.

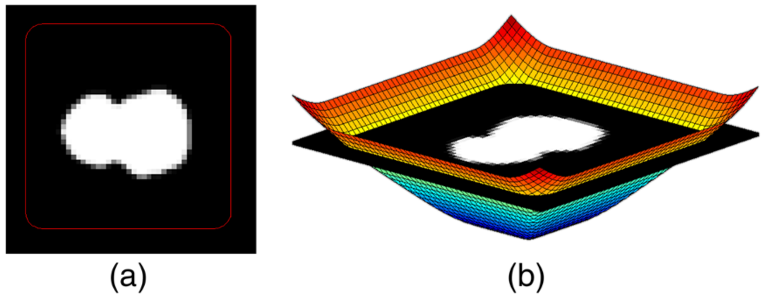


Figure 1 Example of a contour in an image (a) and the level set function that defines it (b).

Chan and Vese introduce a regional-based variational formulation in [10] that is designed to work with images without edges by *minimising the variation in pixel intensity inside and outside the contour*. The expectation is to move the level set contour around an image segment that is homogenous with respect to pixel intensity. The Chan-Vese energy consists of two internal energies, E_{length} and E_{area}, that penalise the length of the contour and the area within it respectively (therefore favouring small, short contours) and two external energies, $E_{\text{var\_inside}}$ and $E_{\text{var\_outside}}$, that penalise variation in pixel intensity inside the contour and outside the contour respectively:

$$E_{\text{cv}} == \mu E_{\text{length}} + \nu E_{\text{area}} + \lambda_1 E_{\text{var\_inside}} + \lambda_2 E_{\text{var\_outside}}, \tag{3}$$

where μ, ν, λ_1 and λ_2 are parameters weighting the importance of their respective penalty terms. The contour needs to be evolved until E_{cv} is minimised.

Given an image, I, with its domain Ω, the inner and outer pixel variation terms can be expressed as

$$E_{\text{var\_inside}} = \int_\Omega |I - c_1|^2 H(\Phi) dx. \tag{4}$$

$$E_{\text{var\_outside}} = \int_\Omega |I - c_2|^2 (1 - H(\Phi)) dx, \tag{5}$$

where H is the Heaviside function defined as

$$H(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0. \end{cases} \tag{6}$$

In a level set formulation, the Heaviside function is used to specify areas inside the contour, where $H(\Phi) = 1$, and outside the contour, $1 - H(\Phi) = 1$. The values c_1 and c_2 are the average pixel intensities inside and outside the level set contour respectively calculated as

$$c_1(\Phi) = \frac{\int_\omega I \times H(\Phi) dx}{\int_\Omega H(\Phi) dx}. \tag{7}$$

$$c_2(\Phi) = \frac{\int_\omega I \times (1 - H(\Phi)) dx}{\int_\Omega (1 - H(\Phi)) dx}. \tag{8}$$

The resulting evolution equation that minimises the functional in Equation 9 is as follows:

$$\frac{\partial \Phi}{\partial t} = \delta(\Phi) \left[\mu div \left(\frac{\nabla \Phi}{|\nabla \Phi|} \right) - \nu - \lambda_1 (I - c_1)^2 + \lambda_2 (I - c_2)^2 \right]. \tag{9}$$

2.2 Level sets with shape priors

Shape priors can be very useful in segmentation, mainly when the object to be segmented is corrupted. This is often the case in real-world applications, such as maritime surveillance. In [13], it is shown how an object partially occluded can be accurately segmented. Adding shape knowledge can be achieved by modifying a variational energy functional designed for segmentation by adding an additional term that penalises deviation from a particular shape.

The majority of techniques that incorporate shape priors use a linear combination of two functionals with one as a common segmentation functional (as discussed in Equation 9) and the other a shape difference [14]. The purpose of the shape difference term is to penalise level set contours that deviate from a predefined shape. A rudimentary example introduced by Rousson and Paragios [15] is the squared difference between the segmenting level set function Φ and a predefined level set function that incorporates the desired shape Ψ:

$$E_{\text{shape}}(\Phi, \Psi) = \int_\Omega (\Phi(x) - \Psi(x))^2 dx. \tag{10}$$

The term in Equation 10 is added to the segmentation-based term (such as Chan-Vese). It is often multiplied by a weighting factor α to control the balance between the two terms:

$$E(\Phi, \Psi) = E_{\text{cv}}(\Phi) + \alpha E_{\text{shape}}(\Phi, \Psi). \tag{11}$$

Parametric methods of incorporating shape information impose this knowledge directly into the level set by defining the level set as the output of some parametric function and minimising segmentation energy functionals with respect to these parameters. Tsai et al. [16] introduce a parametric level set function that is an affine transformed version of a prior-known shape, defined in a level set function Ψ, for the use of shape priors. Parameters controlling translation, scale and rotation of the original shape Ψ are then optimised, rather than the level set function Ψ.

The use of multiple shape priors is documented in the literature and involves the use of selective and competing shape priors. Tsai et al. [16] propose expressing a level set function as a parametric combination of the principle components from a set of training shapes. In [15,17], the authors create a model of a set of prior shapes by assuming that shape priors follow pixel-wise Gaussian distributions. Cremers et al. [18] implement a shape model using a kernel density estimator to produce a statistical shape model that can model fairly distinct training shapes. A shape distance measure is then defined based on Chan and Zhu's work [14].

3 Image processing approach to ship tracking

3.1 Traditional approach

The maritime environment is a particularly challenging one for tracking, and thus, the majority of works discussed here deal with very low level solutions to the problem. As the ocean, constantly filled with moving waves, is prone to producing erroneous detection with methods that detect moving objects, some authors choose to characterise it and label pixels that do not match this characterisation as objects of interest. Sanderson et al. [3] implement an algorithm that does this using frequency information. Smith and Teal [19] implement a similar approach using a histogram-based descriptor of the appearance of the sea. Voles and Teal [20] propose a method that is based on the use of crude descriptors of tiles in an image; the image is divided into overlapping tiles that increase in size as they get closer to the bottom of the image. This method often results in imprecise segmentation results. Voles et al. [21] improve the method presented in [20] by adding motion information obtained using frame differencing; the algorithm designed is purely pixel-based and therefore fails to segment larger maritime objects.

Gupta et al. [22] describe the development of the Maritime Activity Analysis Workbench; this project aims at overcoming limitations of the current maritime surveillance systems. In [23], Ponsford et al. present the design and implementation of an integrated maritime surveillance system based on high-frequency surface wave radars. Leung et al. [24] combine genetic algorithm and radial basis function neural network to search optimal values of a detector model. This model is then used to detect small surface targets in various sea conditions. In [25], an estimation of the ship size using an ANN-based clutter reduction system is proposed.

Socek et al. [26] present a method of combining existing object detection methods with colour information. It initially segments the foreground using background modelling with a Bayes decision framework that works best for backgrounds with complex variations and that are not periodic. In a maritime environment, the algorithm suffered from poor segmentation results having inaccurate boundaries and many scattered pixels. The authors seek to solve these issues by combining results with that of colour-based segmentation. Colour segmentation is treated as a graph-partitioning problem, and combining it with background subtracted output results in enhanced performance. There have also been many other attempts to build maritime surveillance systems; some example could be found in [27,28].

3.2 Level set approach

Szpak and Tapamo [4] introduce an approach that uses level set methods as a way to track detected maritime objects in a scene. Object detection is implemented using a modified method of single Gaussian background subtraction. Where normal background subtraction deals with pixels in isolation, spatial-smoothness constraint is enforced to deal with neighbourhoods of pixels. The constraint assumes that real-world objects are spatially consistent entities and requires that a whole group of pixels, rather than single ones, exhibit motion behaviour before marking them as such. The output of this method is further segmented using level set methods. The contour is used in the tracking phase as described by Bertalmio et al. [29], where the final contour from the previous frame is used as an initial contour in the next. The algorithm was tested in 17 test sequences and showed promising results. It was able to successfully track in all but three of the 17 given sequences. The algorithm even showed good results in overcast and rainy conditions. It failed in sequences where there was insufficient contrast between the ocean and the target and thus fails to pick up specific motion of the target, when the target moves too slowly and is thus considered part of the background and when there is a lot of glint in the scene. Due to its high success rate and possible avenues for improvement, it was decided to base further work on the model proposed in [4].

4 Proposed model of ship tracking using level sets with shape priors

The fundamental contribution of the work presented in this paper is the proposition of a method of incorporating shape knowledge into the system discussed in [4].

4.1 Model overview

There are two main types of video tracker architecture. These include architectures that apply detection and tracking separately and those that perform them jointly [30]. In the first case, possible object regions are produced by the object detector, and the object tracker makes correspondences between these regions from frame to frame. In the second case, the object regions and their correspondences are jointly estimated by keeping object and region information from previous frames and simply updating them for the current one. Level set tracking falls into the latter architecture of the above examples. Here the level set segmentation is run on each frame where the starting contour for each particular frame is the final contour of its predecessor [31,32]. As the proposed system uses a level set for tracking, it is based largely around the second architecture. There is, however, a single difference in that the level set contour is unable to detect objects by itself and must rely on a separate subsystem to initialise it. While this subsystem mainly serves as a tracker initialisation stage, it is theoretically a form of object detection and will be thus be labelled as such. It should be emphasised that unlike the object detection stage of the separate detector/tracker architecture, this subsystem does not operate on every frame. While level set segmentation forms an integral part of the object tracking system, it also forms a part of the object detector, the details of which will be discussed later. An overview of the entire proposed system is shown in Figure 2.

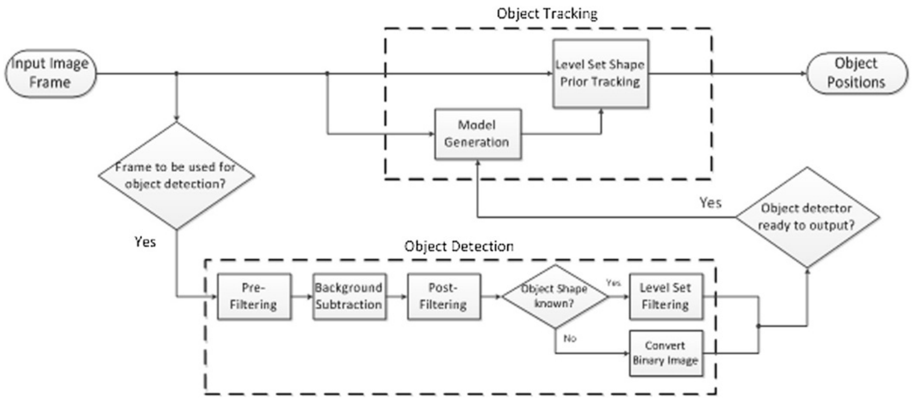


Figure 2 System overview.

The input to the system is a sequence of grey scale image frames from a video of a maritime scene. The output of the system is ideally the same set of image frames with various maritime objects of interest highlighted by a level set contour. An overview of how the system functions is as follows:

- The object tracker does not run until it has received a set of initial object positions from the object detector.
- The object detector uses a background subtraction algorithm that only uses input frames periodically at a fixed spacing. Only once it is filled a buffer of frames is it able to produce an output, and so until then the tracker, and thus the system, has no output.
- Once the buffer is full, the detector is able to output a set of objects to the tracker.
- Once it has obtained a set of initial object positions, the object tracker continues to track these objects.

The object detector is consist of a pre-filtering stage, followed by a background subtraction algorithm. The resultant binary images are then filtered again in a post-filtering algorithm to remove false positives in the image. If it is desired to find a particular shape that is known beforehand, binary level set shape prior segmentation can be further applied in a level-set-filtering algorithm before the input to the object tracker.

The object tracker extracts features of the detected objects in the first frame after detection to create a model for each object. For each subsequent frame, the level set tracking algorithm uses this Figure 3 model, combined with its shape, to track the object. This information is fed back into the tracker for use in tracking the object in the next frame.

4.2 Level set segmentation algorithm

While Szpak and Tapamo [4] used a general level set segmentation algorithm based on the work of Chan and Vese in [10], the work presented here deals with incorporating shape knowledge into the system. There is then a need for a different method to introduce shape priors.

4.2.1 Level set algorithm of Tsai et al.

The system to be designed is based on the work of Tsai et al. [16]. The method presented in this work is based on a parametric function, which offers a number of benefits:

- Parametric models allow for a faster evolution that is less prone to getting stuck in local minima as the

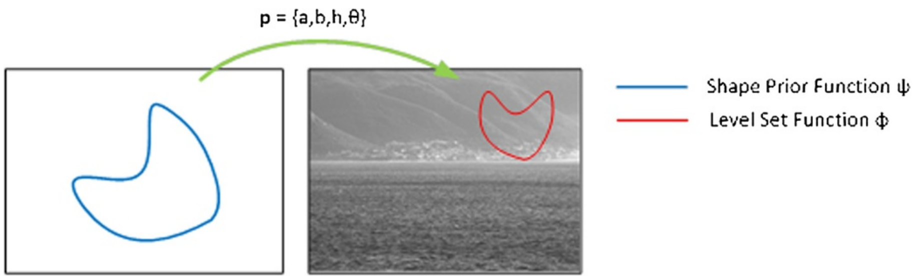


Figure 3 Example of a shape prior. Encoded in function Φ manipulated using **p** to produce a level set function Ψ that produces a contour in the image.

energy is minimised directly by manipulating a few parameters rather than the entire contour.

- Parametric models do not require function re-initialisation. The resultant segmenting level set function is always a transformed version of an original signed distance function, which itself is thus a signed distance function.
- The limited degrees of freedom allows for more 'brute force' numerical methods of energy optimisation, which allow the contour to evolve according to any arbitrary energy functional without the need for symbolic differentiation.

Tsai et al. [16] deal with the incorporation of multiple shape priors in segmentation. For simplicity, however, the system described in this paper has been designed to use a single shape prior. For each sequence that is tested, the desired shape prior is manually set to the shape of an object that appears in that sequence. Simplification to a single shape prior allows the level set function to be manipulated using a set of pose parameters only (i.e. the one shape is known in advance and is given by the shape prior).

The set of pose parameters represented by the vector $p = [a, b, h, \theta]^T$ is introduced, where a and b control horizontal and vertical translation, h controls the scaling, and θ controls the rotation. The level set function can be parameterised in terms of these pose parameters. Consider a level set function Ψ that has a contour in the form of the desired shape prior. The level set function Φ can be defined as a translated, scaled and rotated version of this original function:

$$\Phi[p](x, y) = \Psi(\widetilde{x}, \widetilde{y}), \tag{12}$$

where $\widetilde{x}$ and $\widetilde{y}$ are calculated according to

$$\begin{aligned}\begin{bmatrix}\widetilde{x}\\ \widetilde{y}\\ 1\end{bmatrix} &= T[\mathbf{p}]\begin{bmatrix}x\\ y\\ 1\end{bmatrix}\\ &= \underbrace{\begin{bmatrix}1 & 0 & a\\ 0 & 1 & b\\ 0 & 0 & 1\end{bmatrix}}_{M(a,b)}\underbrace{\begin{bmatrix}h & 0 & 0\\ 0 & h & 0\\ 0 & 0 & 1\end{bmatrix}}_{H(h)}\underbrace{\begin{bmatrix}\cos(\theta) & -\sin(\theta) & 0\\ \sin(\theta) & \cos(\theta) & 0\\ 0 & 0 & 1\end{bmatrix}}_{R(\theta)}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}.\end{aligned} \tag{13}$$

To evolve the energy functional, $\mathbf{p}$ is manipulated in a manner that decreases a predetermined energy functional. This is done using a gradient descent method:

$$\mathbf{p}^{t+1} = \mathbf{p}^t - \alpha_{\mathbf{p}}\nabla\mathbf{p}E, \tag{14}$$

where $\mathbf{p}^t$ and $\mathbf{p}^{t+1}$ are the current and next values of $\mathbf{p}$, respectively. $\nabla_{\mathbf{p}}E$ is the gradient of the energy with respect to $\mathbf{p}$, and α_p is a step-size parameter controlling the speed of evolution.

This evolution minimises the energy functional E by moving p (i.e. a, b, h, θ) in a direction of decreasing energy. This is made clearer by expanding $\mathbf{p}$ into its various parameters:

$$\begin{aligned}a^{t+1} &= a^t - \alpha_a\nabla_a E\\ b^{t+1} &= b^t - \alpha_b\nabla_b E\\ h^{t+1} &= h^t - \alpha_h\nabla_h E\\ \theta^{t+1} &= \theta^t - \alpha_\theta\nabla_\theta E.\end{aligned} \tag{15}$$

4.2.2 Modifications to the work of Tsai et al.

The gradient terms $\nabla_a E, \nabla_b E, \nabla_h E, \nabla_\theta E$ in Equation 15 are derived in [16] by symbolically differentiating the energy functional with respect to each of the parameters. This process is mathematically complex and undesirable. This section addresses how to estimate these gradient terms, thus avoiding the difficult derivation. Provided that the energy functional is differentiable with respect to its input parameters, it is possible to estimate the gradient with respect to each parameter using a numerical method known as the central *difference approximation*, thereby avoiding mathematically complex differentiation. For level set evolution, the following notation is introduced:

- $\Phi_{a,b,h,\theta}$ is the level set function defined by a, b, h, θ.
- $E(\Phi_{a,b,h,\theta})$ is the energy functional calculated from $\Phi_{a,b,h,\theta}$.

The gradient term $\nabla_a E$, for example, can then be approximated using the central difference approximation as

$$\nabla aE \approx \frac{E(\Phi_{a,b,h,\theta}) - E(\Phi_{a-\epsilon,b,h,\theta})}{2\epsilon}. \tag{16}$$

This can similarly be repeated for the remaining gradient terms $\nabla_b E, \nabla_h E$ and $\nabla_\theta E$. These are then used to evolve the pose parameters normally according to Equation 15. Gradient estimation schemes are in fact more resource intensive than calculation from symbolically derived gradients as they require recalculation of two new level set functions and associated energy functionals every time a gradient is estimated. That being said, their main benefit is that any arbitrary energy functional can be plugged into the algorithm without the need for complex symbolic derivations.

4.3 Object detector

4.3.1 Pre-filter

To remove possible noise in the image before it is sent to the background subtraction stage, a pre-filter may be used. Two pre-filters have been proposed: the 3 × 3 Gaussian filter and the 3 × 3 Median filter.

4.3.2 Background subtraction

Once the image has been filtered, the system detects objects using a background modelling and subtraction algorithm. By producing a model of the background and subtracting it from the image, it is assumed that what remains will be objects of interest.

The background model used is based on Elgammal and Duraiswami's work [33] that use kernel density estimation to model the background. Using the previous L values of a particular pixel value $\{I_{t-L}, I_{t-L+1}, \ldots, I_{t-1}\}$, the probability that the next pixel value I_t has a value x is estimated as

$$f(p) = \frac{1}{Lh} \sum_{i=t-L}^{t-1} K\left(\tfrac{p-I_i}{L}\right). \tag{17}$$

K is the kernel with bandwidth h. The Gaussian kernel was used:

$$K(u) = \frac{1}{\sqrt{2\pi}} \exp\left(\tfrac{-u^2}{2}\right). \tag{18}$$

While Silverman [34] shows that the best choice for h for a Gaussian kernel is

$$h = \left(\frac{\widehat{\sigma}^5}{3n}\right)^{\frac{1}{5}}, \tag{19}$$

where $\widehat{\sigma}$ is the standard deviation of the data, in this case it is the previous L values of the pixel. When a new pixel value I_t is observed, the probability of its value is calculated from this density estimate. A *high probability of observation* would indicate that the given pixel is *likely part of the background* whereas a low probability would indicate a foreground pixel. The background subtraction output BS_t is thus

$$\mathrm{BS}_t(x,y) = \begin{cases} 1 & \text{if} f_{t,(x,y)}(I_t(x,y)) > \mathrm{Th} \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where Th is a predefined threshold that needs to be decided upon. A number of improvements can be made to this model to better suit its application of background subtraction. It is obvious that more recent pixel values from P are more relevant to the density estimation. For kernel density estimation with time series data, Harvey and Oryshchenko [35] suggest using a weighting scheme such that

$$f(p) = \frac{1}{h} \sum_{i=t-L}^{t-1} K\left(\tfrac{p-I_i}{L}\right) \times \omega_i, \tag{21}$$

where ω_i is the weight for the i^{th} kernel. Here $\sum_{i=1}^{n} \omega_i = 1$. In order to weight more previously viewed pixel values higher, the following weighting scheme was chosen:

$$\omega_i = \frac{i}{n} \tag{22}$$

such that the weighting increases linearly with i. This will ensure that the most *recent* pixel value will have the *highest weight*. A further modification to the system is the use of frame spacing. Rather than using the entire set of previous L pixel values, a buffer of n values is created from pixels spaced s frames apart:

$$\{I_{t-L}, I_{t-L+1}, \ldots, I_{t-1}\} \rightarrow \{I_{t-ns}, I_{t-(n-1)s}, \ldots, I_{t-2s}, I_{t-s}\} \tag{23}$$

Take the example of a sequence of a fairly slowly moving object (such as a ship) captured using a high-frame-rate camera. Suppose a buffer of 50 previous frames is used to build a background model. The slow speed of the object combined with the high frame rate of the camera would probably result in very little object movement for these frames, resulting in most of the object becoming part of the background model. By using frames that are spaced apart, the resultant background model is less likely to include the object because it will have moved over these frames. The disadvantage of this method is that this places an upper limit on how fast an object may be moving so that it is not missed by the spaced frames. The spacing must be decided upon by the system designer. This is implemented in the decision block in Figure 2 where frames are only sent to the object detection stage at fixed intervals. Obviously, this also means that the background subtraction stage is not able to provide a corresponding output for every input frame. This, however, is acceptable as the object tracker stage keeps track of objects for every frame after detection.

4.3.3 Post-filtering

A major problem in using background subtraction algorithms alone for maritime surveillance is the motion of the sea. Although kernel density estimation would be able to filter out pixels which oscillate between two values, it still would classify a wave moving across the image, for example, as a legitimate motion. It is for this reason that the binary image is filtered after background subtraction. This subsection details a number of different filters that can be used to remove these unwanted white pixels while keeping the desired ones.

- *Motion persistence filtering.* Motion persistence filtering is a novel method introduced by this work that attempts to remove white pixels that only appear in a few background subtracted frames. The logic behind motion persistence filtering is that while waves will produce legitimate motion pixels in a background subtraction algorithm, unlike those of ships, this motion is short-lived and may last merely over a few frames.
  Assuming an input set of motion images $\{B_1, B_2, \ldots, B_t\}$, this filter operates in a similar

fashion to a two-dimensional kernel density estimator: For every white pixel in every image, a two-dimensional Gaussian kernel is placed (centred) over the pixel and its surrounding neighbours. The bandwidth of each Gaussian is set to be the distance to the nearest white pixel in the image. This builds a two-dimensional probability estimate, where more 'persistent' motion pixels have higher probabilities. The algorithm proceeds to find pixels connected to these high-density areas (over a fixed threshold) in the most recent motion frame B_t. It was empirically decided to use three previous background subtracted frames for motion persistence filtering. If background subtraction with frame spacing is used, there should be considerable changes in sea motion across these fames. The threshold for the method was likewise set at 0.000008.

- *Fixed-threshold connected component filtering.* Connected component filtering is a low-level image processing technique that simply removes connected components (or blobs) from a binary image depending on some criteria. The first connected component filtering algorithm that was proposed simply removes blobs below a certain threshold on blob size using 8-connectivity to determine blobs.
- *Variable threshold connected component filtering.* Voles and Teal note in [20] that because a maritime scene is an outdoor one with considerable depth of field, objects close to the camera are projected near the bottom of the image and thus appear larger than those further away from the camera. The second connected component filtering algorithm is based around this idea. Here the threshold on blob size is no longer a preset constant, but a linear function of a blob's y-coordinates:

$$\text{Threshold} = M(y). \tag{24}$$

- *Spatial-smoothness filtering.* Szpak and Tapamo [4] suggest that methods based on thresholding the area of connected components as described above are not suitable as targets may be smaller than some waves in the image and thus be erroneously removed. While a variable threshold should solve this problem, their suggested method of spatial-smoothness filtering is implemented for comparison. This technique is built into the proposed single Gaussian background subtraction before pixels are thresholded and thus requires some modification; however, the expected behaviour is the same. For a pixel at (i,j) with probability $f_{\text{KDE}}(I(i,j))$, Γ is calculated for a window of $2r \times 2c$ pixels around it as

$$\Gamma = \sum_{p=-r}^{r} \sum_{q=-c}^{c} w_{i+p,j+q} \times f_{\text{KDE}}(I(i+p,j+q)). \tag{25}$$

Γ is thus the weighted sum of the input pixel (i,j) and its neighbours' probabilities. This effectively is a smoothing operation before the probability estimates are converted into a binary image in the background subtraction algorithm. The output of the background subtraction (BS) for this pixel is then modified as follows:

$$\text{BS}(x,y) = \begin{cases} 1 & \text{if } \Gamma < Th \times 2r \times 2c \\ 0 & \text{otherwise,} \end{cases} \tag{26}$$

where Th is the background subtraction threshold normally used. In [4], a 3×3 filter is used with constant weights with values of 1; these parameters will then be used for testing.

4.3.4 *Level set filtering*

If the shape of a particular object is known beforehand, it is possible to filter the binary image further after post-filtering and detect only blobs that match its shape. Yezzi and Tsai [36] propose the binary mean model:

$$E_{\text{binary}} = -\frac{1}{2}(c_1 - c_2)^2, \tag{27}$$

where c_1 and c_2 are the mean values for the pixels inside and outside Φ as calculated by Equations 7 and 10, respectively. This energy tries to separate the image into two regions of homogeneous pixel intensity by maximising the difference between c_1 and c_2. It is known that in an ideal case, the level set contour sits tightly around a white blob. In this case, the ideal values for c_1 and c_2 are known. Specifically, ideally we have $c_1 = 1$ and $c_2 = 0$. The binary mean model in Equation 27 is thus modified accordingly:

$$E = -\frac{1}{4}[\,(c_1 - 1)^2 + c_2^2]\,. \tag{28}$$

The first term of this energy penalises inner mean values (c_1) that are not equal to 1, while the second term penalises outer mean values c_2 that are not equal to 0. This energy is minimised according to the modified version discussed above. The segmentation is applied for every blob or connected component in the image in isolation. Naturally, the blob most likely to be the object sought is that with the lowest energy.

4.4 Object tracker

Once an object has been detected, its shape and position are known for a single frame. It is necessary to track the object for every frame thereafter. To do this, the object tracker makes use of a single level set function that evolves itself to sit around the object in each frame. The level set function is initialised in the image using the object detector and can come directly from the level set shape-filtering stage of the object detector in the form of a single shape, or the binary image at the output of the filtering stage in the form of a set of shapes. Assuming that the level set contour surrounds the object correctly in the first frame after detection, the tracker makes use of pixel information that is within the contour. The initial contour and its inner pixel information in the first frame are henceforth known as the *target contour* and *target model*, respectively. For every subsequent frame, the current level set contour (which now probably will not lie around the object) and the information about its inner pixels are known as the *candidate contour* and *candidate model*, respectively. By creating an energy functional that penalises deviations of the candidate model from the target model, one is able to force the candidate contour around objects appearing similar to those surrounded by the target contour in the original frame. Different energy functionals may be created by comparing the target and candidate models in various ways. The various functionals are discussed next.

4.4.1 *Energy functionals for tracking*

The following are energy functionals used for tracking:

- *Histogram.* The simplest feature that can be drawn from the pixels is the histogram. Pixels are put into k bins where h_i^t is the number of pixels that falls into the ith bin for the target t, andh_i^c for the candidate c. The energy is the sum of squared differences of the bins:

$$E = \sum_{i=1}^{k} (h_i^t - h_i^c)^2. \tag{29}$$

- *Fast Fourier transform.* Frequency information may be utilised to make the feature more invariant to changes in lighting. Given a bounding box around the contour $M \times N$ pixels in size, a modified Fast Fourier transform is used to only extract frequency information from pixels within the contour:

$$F_\Phi(x,y) = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} \Phi H(\Phi(m,n)) I(m,n) e^{-i2\pi\left(\frac{xm}{M}+\frac{yn}{N}\right)}. \tag{30}$$

The energy function is then defined as the difference in target and candidate spectra:

$$E = \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} |F_{\Phi_c} - F_{\Phi_t}|. \tag{31}$$

- *Statistical descriptors.* Statistical descriptors of the target pixels can be calculated. This approach has been used previously in maritime tracking work by Voles and Teal in [20]. The following descriptors in Table 1 have been modified to suit a level set case, once again for a bounding box around the contour $M \times N$ pixels in size:
After normalising with respect to a maximum value, these descriptors can be thought of as vectors that form a basis for a 4D space. The target contour's pixel distribution is then represented as a point within this space $D^t = [d_1^t, d_2^t, d_3^t, d_4^t]$ and similarly so for a candidate contour $D^c = [d_1^c, d_2^c, d_3^c, d_4^c]$. The Euclidean distance between these two points can then be used as the energy functional:

$$E = \sqrt{\sum_{k=1}^{4} (d_k^t - d_k^c)^2}. \tag{32}$$

Table 1 Various statistical descriptors

Descriptor	Formula
Energy	$d_1 = \sum_{m=0}^{M}\sum_{n=0}^{N} H(\Phi(m,n)).I(m,n)^2$
Entropy	$d_2 = \sum_{m=0}^{M}\sum_{n=0}^{N} H(\Phi(m,n)).I(m,n).\log(I(m,n))$
Homogeneity	$d_3 = \sum_{m=0}^{M}\sum_{n=0}^{N} \frac{H(\Phi(m,n)).I(m,n)}{1+\lvert m-n\rvert}$
Contrast	$d_4 = \sum_{m=0}^{M}\sum_{n=0}^{N} H(\Phi(m,n)).(m-n)^2.I(m,n)$

4.4.2 *Normalisation for rotation/scale invariance*

Apart from energy functionals for tracking, the second pertinent issue in object tracking is consideration of possible rotation or scale changes of the object being tracked. Evolution using the above-mentioned functionals may become erroneous for large differences in scale and rotation between the candidate and target level set functionals. To remove this error, the candidate contour is normalised with respect to its rotation and scale with respect to the target before evaluating the tracking energy. Assuming an arbitrary scale parameter h and rotation parameter θ that produces the candidate function Φ_c, both the current image frame I and the candidate function Φ_c are transformed according to:

$$\widetilde{\Phi}_c(x,y) = \Phi_c(\widetilde{x},\widetilde{y}) \tag{33}$$

$$\widetilde{I}(x,y) = I(\widetilde{x},\widetilde{y}), \tag{34}$$

where $\widetilde{x}$ and $\widetilde{y}$ are calculated as:

$$\begin{bmatrix} \widetilde{x} \\ \widetilde{y} \\ 1 \end{bmatrix} = \begin{bmatrix} 1/h & 0 & 0 \\ 0 & 1/h & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \tag{35}$$

This transforms both the candidate function and the pixels it contains to the same scale and rotation as the target function. The transformed function $\widetilde{\Phi}_c$ and image $\widetilde{I}$ are then used in the evaluation of the energy functional. It should be emphasised that the values of h and Φ do not change and that the original candidate contour and image remain intact: their transformed values are used exclusively for evaluating energy functionals.

5 Experimental results and discussion

The system was implemented in MATLAB and tested using a set of ten maritime sequences obtained from the Council of Science and Industrial Research (South Africa). These sequences include a variety of scenes, weather conditions and maritime objects of interest. A specific target object was chosen for each of the sequences and used to test both object detection and tracking—the target objects for each of the sequences. This section introduces performance metrics regarding object detection and tracking and uses these metrics to compare the various choices for detection and tracking described previously. Thereafter, the final system is proposed.

5.1 Performance metrics

5.1.1 Object detection

The object detection algorithm is a form of classification task, where pixels are either classified as belonging to a maritime object or not. The detector's output is a binary image with pixels that are classified as objects being labelled as 1. Given the actual classification of pixels (ground truth) and the output of the system, four outcomes are possible. Outcomes where the system agrees with the actual data are labelled true positives (TP) or true negatives (TN) depending on whether the pixel belongs to an object or not. If the system incorrectly labels a pixel as an object when in actuality there is not one there, this is called a false positive (FP), while a false negative (FN) is a case where an object is present but the system fails to detect it. For the classification task, *Precision* is defined as, given the actual classifications of particular subjects, the proportion of cases where the subject was classified as positive and was actually the case [37]. In terms of the totals discussed above, Precision is calculated as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{36}$$

Recall is defined as the proportion of subjects which were actually positive and were classified as such [37]. In terms of the above totals:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{37}$$

Hripcsak and Rothschild [37] define the F measure, which is a harmonic mean of the two metrics:

$$F_\beta = \frac{(1+\beta^2) \times \text{Recall} \times \text{Precision}}{\beta^2 \times \text{Precision} + \text{Recall}}, \tag{38}$$

where β is a parameter that allows one to weight Precision or Recall more heavily. F_β is the notation used to indicate the β used in a particular F score. Szpak and Tapamo [4] note that for a surveillance system, the reduction of false negatives is top priority. For this reason, Recall was weighted twice as much as Precision by setting $\beta = 2$. This F score can be used to test both the output of the background subtraction algorithm and the post-filter. To test the accuracy of level set filtering, the output of the post-filtering stage was segmented using binary level set shape prior segmentation for each sequence. For an input binary image with n blobs and assuming that energy is positive at all times, the segmentation proficiency score is defined as:

$$P = \frac{\text{argmin}_{i \neq t \leq n} E(i)}{E(t)}, \tag{39}$$

where $E(x)$ is the final energy obtained from the level set segmentation of blob x in the image and t is the index of the blob belonging to the object that is to be found. This essentially measures the contrast between the energy associated with segmentation of the actual object and the blob with the lowest energy of the remaining blobs. If the desired object has the lowest energy, P will be greater than 1, indicating a correct classification. However, if another blob has a lower energy than the desired object, P will be less than 1.

If the object is successfully detected, the quality of its segmentation is measured. Krishnaveni and Radha [38] suggest using Dice's coefficient [39] as a method of performance evaluation for level set methods:

$$s = \frac{2(A \cap B)}{A + B}, \tag{40}$$

Table 2 Optimal background subtraction parameters for various sequences

Sequence number	Optimal *n*	Optimal *s*	Optimal Th
1	19	10	0.015
3	19	15	0.005
4	18	14	0.01
5	14	13	0.015
8	8	8	0.01
9	18	5	0.015

Table 3 Area of the smallest ground truth object in each sequence

Sequence	Object area
1	458
3	1047
4	696
5	677
8	70
9	1468

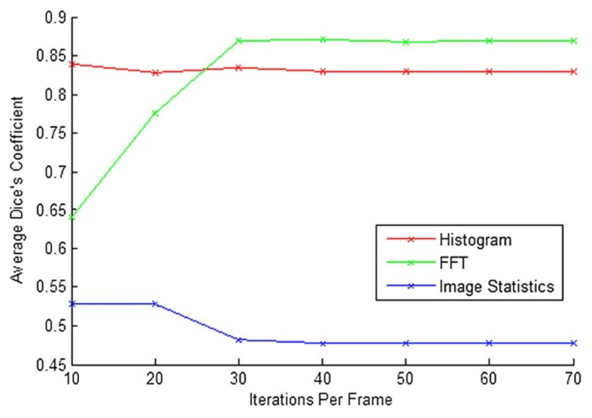


Figure 4 Average Dice's coefficient versus the number of iterations for various energy terms.

where A and B are the ground truth and resultant segmentation regions, respectively, and s varies from 0 to 1 depending on the proportion of pixels shared between the ground truth and the segmenting contour.

5.1.2 Object tracking

The tracker detection rate (TDR) is the average number of frames where an object is successfully tracked and is defined by Porikli and Bashir [13] as:

$$\mathrm{TDR} = \frac{\mathrm{TPF}}{\mathrm{TG}}, \tag{41}$$

where TPF is the number of frames where the system contour overlaps the ground truth object. TG is the number of frames in which the ground truth object is present. There are different strategies to test if object overlap occurs, the simplest of which is to test if the system contour's centroid lies within the ground truth object's bounding box. To measure the degree of success for a tracked object, the object tracking error (OTE) [13] is defined as:

$$\mathrm{OTE} = \frac{1}{\mathrm{TPF}} \sum_{i=1}^{\mathrm{TG}} \mathrm{Dist}(p_i^{\mathrm{GT}}, p_i^{\mathrm{Sys}}) \times G(\mathrm{GT}_i, \mathrm{Sys}_i), \tag{42}$$

where Dist() measures the distance between the centroids for the ground truth object p_i^{GT} and system contour p_i^{Sys} in the i^{th} frame, respectively. $G(\mathrm{GT,Sys})$ is an overlap function defined as:

$$G(\mathrm{GT,Sys}) = \begin{cases} 1 & \text{if GT and Sys overlap} \\ 0 & \text{otherwise.} \end{cases} \tag{43}$$

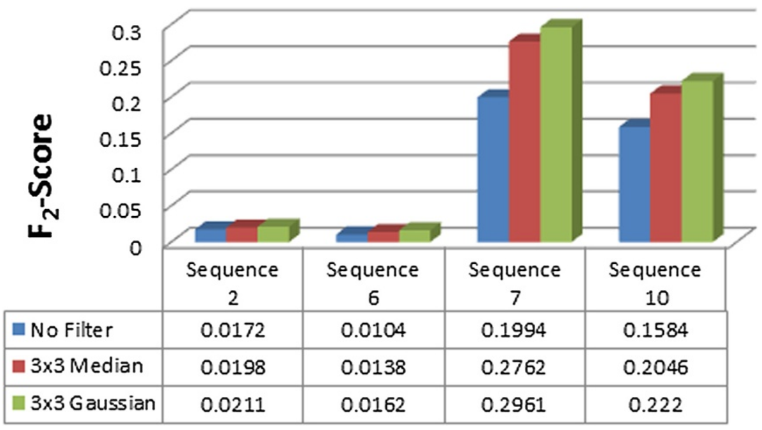
5.2 Optimisation

There are a number of parameters that need to be decided upon in the object detection and tracking subunits. Given the performance metrics defined in the previous section, the parameter values are deduced in an optimal way.

5.2.1 Object detection

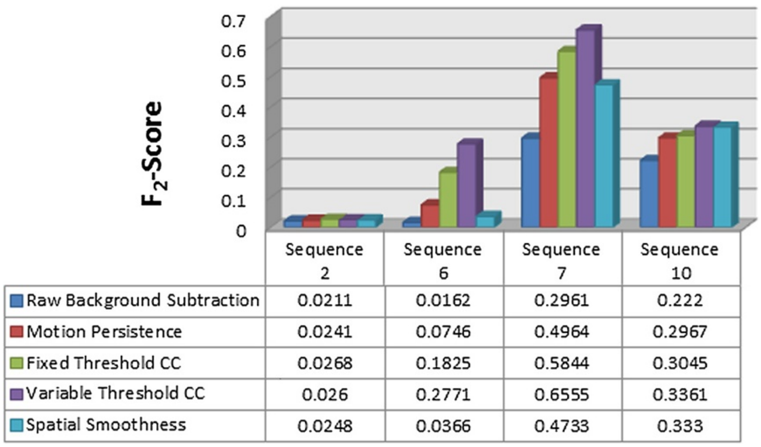
While the common strategy of splitting a dataset into two-thirds for training and one-third for testing is shown to work well for reasonably sized datasets (over 100 cases [40]), this was used for the system dataset despite its small size. The training could be repeated over a larger data set as future work, and the concepts are emphasised here with the term 'optimal' being used loosely. Sequences 1, 3, 4, 5, 8 and 9 were randomly chosen as a training set for optimisation, while the remaining sequences were used for testing.

Discussion of the two possible pre-filters, the 3×3 Gaussian filter and the 3 × 3 Median filter, is postponed until last. No optimisation can be performed at the moment, but later in this paper, performance of the two filters is compared.



	Sequence 2	Sequence 6	Sequence 7	Sequence 10
🟦 No Filter	0.0172	0.0104	0.1994	0.1584
🟥 3x3 Median	0.0198	0.0138	0.2762	0.2046
🟩 3x3 Gaussian	0.0211	0.0162	0.2961	0.222

Figure 5 F_2 **scores for various pre-filtering algorithms.**



	Sequence 2	Sequence 6	Sequence 7	Sequence 10
🟦 Raw Background Subtraction	0.0211	0.0162	0.2961	0.222
🟥 Motion Persistence	0.0241	0.0746	0.4964	0.2967
🟩 Fixed Threshold CC	0.0268	0.1825	0.5844	0.3045
🟪 Variable Threshold CC	0.026	0.2771	0.6555	0.3361
🟦 Spatial Smoothness	0.0248	0.0366	0.4733	0.333

Figure 6 Comparison of various F_2 scores for various filtering algorithms.

- *Background subtraction.* In order to achieve a more satisfactory result, the background subtraction algorithm must be optimised with respect to the following:

  - The number of frames used for kernel density estimation (n)
  - The spacing between each frame (s)
  - The probability threshold used (Th)

  These variables were adjusted to maximise the F_2 score calculated for a small window around an object of interest in an image. A brute force search was run to find optimal parameters for each of the above-mentioned training sequences. For every sequence, the values of n, s and Th were varied and the cost function was measured. Those that resulted in the lowest cost function were considered optimal for each sequence and are shown in Table 2.
  The averages of these parameters were used in the final algorithm hoping that these would give fairly decent results across most of the sequences. Therefore, the optimally chosen parameters are as follows:

  $$n = 16, s = 11 \text{ and } Th = 0.0117$$

- *Post-filtering.* Out of the possible post-filters listed, the fixed threshold connected component algorithm has an optimal threshold based on blob size, while the variable threshold algorithm has an optimal threshold function that may be used. To find an optimal fixed threshold for fixed threshold connected component filtering, the area of the smallest ground truth object in each sequence was measured in order. The algorithm has to filter out blobs which are superfluous but still keep those that belong to actual maritime objects. To do this, a lower threshold for blob size must be selected. Table 3 shows the area of the smallest ground truth object in each sequence. The smallest object, observed in sequence 8, has an area of 70 pixels. To ensure that objects above this size are kept, and allowing 10 pixels for safety, the threshold was set at 60. The area of each of the

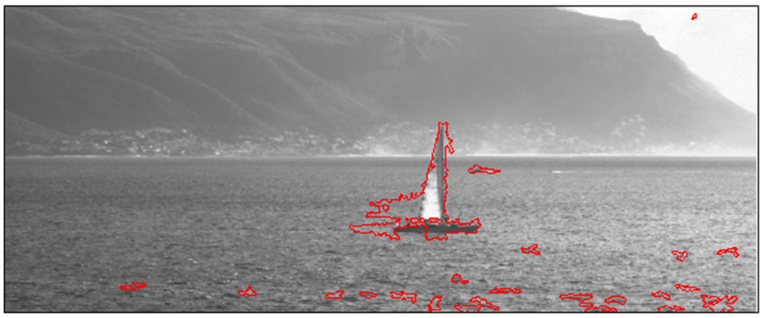


Figure 7 Output from background subtraction overlaid on the original image for sequence 7.

training objects was plotted versus its normalised vertical position. The following area threshold function that included each object was chosen:

$$M(y) = 90y + 10, \tag{44}$$

where y is the normalised vertical position. This function ensures that blobs near the top of the image need only be over 10 pixels in size to be kept in the image, while blobs at the bottom of the image need to have an area over 100 pixels to be kept.

5.2.2 Object tracking

For each possible formulation of the energy functional, the tracking algorithm was run on a hundred frames for various iterations per frame. Three randomly selected sequences were used as a source of these frames. The average Dice's coefficient versus the number of iterations per frame for each energy term is shown in Figure 4.

5.3 Object detection results

Given the deduced background subtraction and post-filtering parameters, the performance of the object detection subsystems is not measured. The choice of pre-filter is now addressed. The optimised object detection algorithm was run on the test set of sequences: sequences 2, 6, 7 and 10. The entire system has been broken into its individual subsystems which have been tested individually. Subsystems that have a number of different algorithms at their disposal have been tested using each of these techniques and compared.

5.3.1 Background subtraction and pre-filter

Figure 5 shows F_2 scores for various pre-filtering algorithms that have been tested. The comparatively low scores for sequences 2 and 6 can be explained by the large amount of glint in sequence 2 and the small object size in sequence 6. Despite these low scores, pre-filtering was able to improve scores for every sequence. It is clear that using the 3×3 Gaussian filter provides the best results, and so this was used for testing in the next stages of the algorithm.

5.3.2 Post-filtering

Figure 6 shows the F_2 scores of various post-filtering methods. These have been applied to the output of the background subtraction used with a Gaussian pre-filter, and so F_2 scores for the raw background subtracted output have been included for comparison. Every filtering method was able to improve the score for every sequence. F_2 scores for sequences with large objects are less sensitive to false positives as they do not compare much proportionally to the number of pixels in the object. The converse is true for smaller objects such as sequence 6 where the largest improvement from filtering is shown.

As they are the most aggressive filtering methods, it comes as no surprise that the connected component filters (both fixed and variable threshold) give the biggest improvement in scores. One should bear in mind the possibility that if a target were too small, these filtering algorithms would remove it from the image, and so there is an associated risk with using them. Due to its increasing threshold at the bottom of the image, the variable threshold connected component algorithm was able to remove more false positives than the fixed threshold, producing the highest F scores out of all the filters. This algorithm yielded an average increase of 78% in F_2 scores for all

Table 4 *P* scores for various image sequences using Chan-Vese energy as classification criteria

Sequence	*P* score	Classification (pass/fail)
2	6.9316	Pass
6	0.4642	Fail
7	4.2569	Pass
10	13.1468	Pass

Table 5 Segmentation results from image sequences that passed level set filtering

Sequence	Dice's coefficient
2	0.8698
7	0.6113
20	0.6667

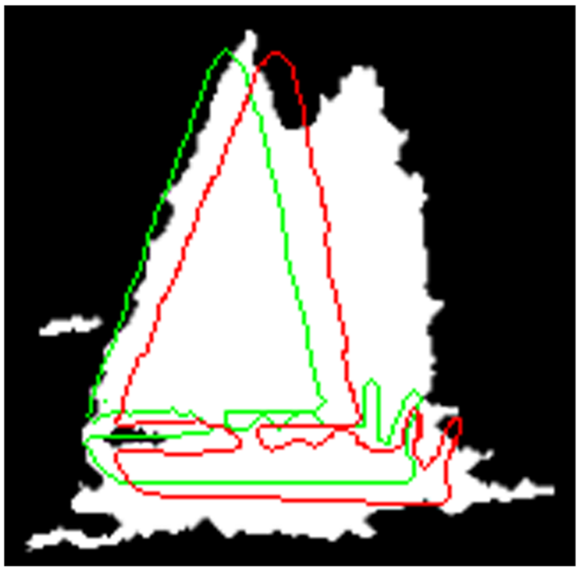


Figure 8 Segmentation result of sequence 10 using level set filtering. The green contour represents the ground truth object, while the red represents the segmenting contour. Smudging effects from background subtraction degrade segmentation results.

Table 6 Performance metrics for each sequence

Sequence	TDR	OTE
1	1	1.8795
2	1	3.7144
3	1	1.4506
4	0.85	10.799
5	0.71	4.8505
6	0.38	7.2204
7	1	20.62
8	1	2.7759
9	0.7933	1.5088
10	1	2.6944

The sequences have been calculated using sequences of 300 frames.

test sequences. The inability of post-filtering algorithms to improve test scores in sequence 2 can be attributed to the large amount of glint and ocean movement in the image. Figure 7 shows the output of post-filtering, where the majority of false positives at the bottom of the image are removed.

5.3.3 Level set filtering

To test the level set segmentation technique, the output from the variable threshold connected component algorithm was used as input data as it had the best results out of all the filtering methods. Table 4 shows P scores for each sequence and indicates which passed or failed at classification. Apart from sequence 6, all the sequences were correctly classified with very good P scores. A likely reason for sequence 6's failure is the similarity in the shape of the blob around the ship with false positive blobs in the image. Despite its poor F_2 score, the algorithm was able to locate the object in sequence 2. Table 5 shows actual segmentation results for the sequences that were correctly classified.

While sequence 2 can be considered as a good segmentation, smudging effects from the background subtraction algorithm caused poor segmentation results for sequences 7 and 10. These smudging effects trailing the objects are simply artefacts of the threshold selection for the sequences. As the ship in sequence 10 (Figure 8), for example, moves from right to left, its pixel values become part of the density estimate for pixels trailing it. This decreases the probability of seeing a sea pixel, and so when one is seen, this may be marked as the foreground if the threshold is not set low enough. The segmenting contours have tried to position themselves to include as many of these pixels as possible resulting in offsets from the ground truth.

5.4 Object tracking results

Table 6 shows TDR and OTE for each sequence. The main reason for poor performance in sequences (especially 4 and 5) can be attributed to change in frequency characteristics of the pixels that make up target objects. Figure 9 shows an example of this from sequence 5. The target contour in the first frame (above, in yellow) and the tracking contour in the frame where it starts to drift away from the object (bottom, in red) are shown. In this case, glint from the sun strongly defines the borders of the object in the target frame, while the contour starts to drift when this glint is no longer present, making the target almost indistinguishable from the background to the human eye.

Poor tracking results in sequence 6 can be attributed to small object size and similar frequency characteristics of the object with the ocean, whereas camera panning ruined the otherwise perfect tracking results in sequence 9. Although the tracking contour successfully overlapped the object in every frame of sequence 7, the sequence has a comparatively high object tracking error of 20.62. This is due to a local minimum in the tracking energy functional under the object that left the tracking contour offset from its target in most frames.

5.5 Speed of algorithm

On a 2.0-GHz dual-core processor, our system could process approximately ten frames per second with a MATLAB implementation of the algorithms. This performance can be greatly improved by doing a C++ implementation and parallelising some of the algorithms (e.g. background subtraction). The running time of the video tracker developed has purposefully been given

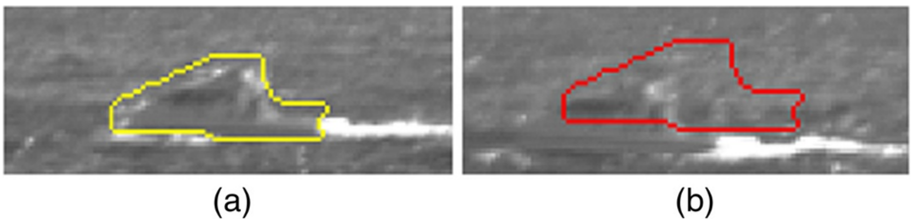


Figure 9 Change in frequency characteristics of the pixels that make up target objects. Comparison of target contour from sequence 5 **(a)** and the frame in which the contour started to drift from the object **(b)**.

lesser priority in order to focus on the development of concepts and methodology. Since this work addresses unexplored topics, the main focus is on the development of models and algorithms to address fundamental issues rather than address supplementary practical issues such as algorithm speed which may be optimised in future work for real-time computing.

6 Conclusions

This paper has investigated the use of prior knowledge of a ship shape to assist level set segmentation in video tracking for a maritime surveillance problem. It shows that integrating shape priors into level set segmentation is feasible and results in promising video tracking performance. While the system did produce an acceptable set of results, it still requires some assumptions that would not be practical in a real-life situation. Future work would allow for relaxation of these assumptions. The system only uses a single shape prior that must be manually preset for every sequence that would not be feasible in a real-life system. To remove the reliance on the user, a bank of multiple training shapes could be modelled using a method such as kernel density estimation. This model would then be used in place of a fixed shape prior for segmentation. It has to be noted that the system would be far more accurate if trained with a larger training set.

Competing interests
Both authors declare that they have no competing interests.

Acknowledgements
The authors gratefully acknowledge PRISM/CSIR for funding this research. We would also like to thank the Optronic Sensor Systems at the Center for Scientific and Industrial Research (CSIR) for making the dataset available.

Received: 14 February 2013 Accepted: 2 July 2013
Published: 26 July 2013

References
1. G Stubblefield, B Parlatore, Condition yellow. PassageMaker, Winter, 85 (1999)
2. Piracy attacks in East and West Africa dominate world report (ICC Commercial Crime Services, 2012), [http://www.icc-ccs.org/news/711-piracy-attacks-in-east-and-west-africa-dominate-world-report]. Accessed 09 Nov 2012
3. J Sanderson, M Teal, T Ellis, Characterisation of a complex maritime scene using Fourier space analysis to identify small craft, in *Seventh International Conference on Image Processing and its Applications* (Conf. Publ. No. 465), vol. 2 (IEE, Manchester, 1999), pp. 803–807
4. ZL Szapk, JR Tapamo, Maritime surveillance: tracking ships inside a dynamic background using a fast level-set. Expert Syst Appl. **38**(6), 6668–6680 (2011)
5. PL Herselman, CJ Baker, de HJ Wind, An analysis of X-band calibrated sea clutter and small boat reflectivity at medium-to-low grazing angles. Int. J. Navigation Observation (2008). doi:10.1155/2008/347518
6. R Vicen-Bueno, R Carrasco-Alvarez, M Rosa-Zurera, JC Nieto-Borge, Sea clutter reduction and target enhancement by neural networks in a marine radar system. Sensors. **2009**(9), 1913–1936 (2009)
7. P Westall, J Ford, P O'Shea, S Hrabar, Evaluation of machine vision techniques for aerial search of humans in maritime environments, in *Digital Image Computing: Techniques and Applications (DICTA) 2008* (Canberra, 1–3 Dec. 2008), pp. 176–183
8. A Maistrou, Level set methods - overview (2008), [http://campar.in.tum.de/twiki/pub/Chair/TeachingSs08EvolvingContoursHauptseminar/ActiveContours.pdf]
9. M Kass, A Witkin, D Terzopoulos, Snakes: active contour models. Int J. Comput. Vis. **1**(4), 321–331 (1988)
10. T Chan, LA Vese, Active contours without edges. IEEE Trans Image Proces. **10**(2), 266–277 (2001)
11. S Osher, J Sethian, Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations J. Comput. Phys. **79**, 12–49 (1987)
12. D Cremers, Dynamical statistical shape priors for level set based tracking. IEEE Trans. Pattern Anal. Mach. Intell. **28**(8), 1262–1273 (2006)
13. F Bashir, F Porikli, Performance evaluation of object detection and tracking systems, in *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)* (IEEE Computer Society, New York, 2006), pp. 7–14
14. T Chan, W Zhu, Level set based shape prior segmentation, in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 2 of CVPR 2005* (San Diego, 20–25 June 2005), pp. 1164–1170
15. M Rousson, N Paragios, Shape priors for level set representations, in *Computer Vision – ECCV 2002, ed. by A Heyden, G Sparr, M Nielsen, and P Johansen,* Lecture Notes in Computer Science, vol. 2351 (Springer, 1039 Heidelberg, 2002), pp. 78–92
16. AJ Tsai, A Yezzi, W Wells, D Tempany, C Tucker, A Fan, W Grimson, A Willsky, shape-based approach to the segmentation of medical imagery using level sets. IEEE Trans. Med. Imaging. **22**(2), 137–154 (2003)
17. M Rousson, N Paragios, R Deriche, Implicit active shape models for 3D segmentation in MRI imaging, in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2004*, ed. by C Barrillot, DR Haynor, and P Hellier (Springer, Heidelberg, 2004), pp. 209–216
18. D Cremers, S Osher, S Soatto, Kernel density estimation intrinsic alignment for shape priors in level set segmentation. Int. J. Comput. Vis. **60**(3), 335–351 (2006)
19. AA Smith, M Teal, Identification and tracking of maritime objects in near-infrared image sequences for collision avoidance, in *Seventh International Conference on Image Processing and its Applications* (Conf. Publ. No. 465), vol. 1 (IEE, Manchester, 1999), pp. 250–254
20. P Voles, M Teal, Maritime scene segmentation, http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/VOLES/marine.html. Accessed 12/04/2012
21. P Voles, M Teal, J Sanderson, Target identification in a complex maritime scene, in *IEE Colloquium on Motion Analysis and Tracking* (IEE, London, 1999), pp. 15/1–15/4
22. KM Gupta, DW Aha, R Hartley, PG Moore, Adaptive maritime video surveillance, in *Proceedings of SPIE-09, Volume 7346 of Visual Analytics for Homeland Defense and Security* (SPIE, Bellingham, 2009)
23. AM Ponsford, L Sevgi, H Chan, An integrated maritime surveillance system based on high-frequency surface-wave radars. 2. Operational status and system performance. IEEE Antennas Propagation Mag. **43**(5), 52–63 (2001)
24. H Leung, N Dubash, N Xie, Detection of small objects in clutter using a GA-RBF neural network. IEEE Trans. Aerosp. Electron. Syst. **38**, 98–118 (2002)
25. R Vicen-Bueno, R Carrasco-Alvarez, M Rosa-Zurera, J Nieto-Borge, M Jarabo-Amores, Artificial neural network-based clutter reduction systems for ship size estimation in maritime radars. EURASIP J. Adv. Signal Process. **2010**(9), 380473 (2010)
26. D Socek, D Culibrk, O Marques, H Kalva, B Furht, A hybrid color-based foreground object detection method for automated marine surveillance, in *Advanced Concepts for Intelligent Vision Systems – ACIVS 2005,* ed. by J Blanc-Talon, W Philips, D Popescu, and P Scheunders, Lecture Notes in Computer Science, vol. 3708 (Springer, Heidelberg, 2005), pp. 340–347
27. S Fefilatyev, D Goldgof, C Lembke, Tracking ships from fast moving camera through image registration, in *2010 International Conference on Pattern Recognition* (IEEE Computer Society, Los Alamitos, 2010), pp. 3500–3503
28. M Michael Seibert, BJ Rhodes, NA Bomberger, PO Beane, JJ Sroka, W Kogel, W Kreamer, C Stauffer, L Kirschner, E Chalom, M Bosse, R Tillson, SeeCoast port surveillance, in *Proceedings of SPIE Vol. 6204: Photonics for Port and Harbor Security II* (Orlando, 17 Apr 2006)

29. M Bertalmio, G Sapiro, G Randall, Morphing active contours. IEEE Trans. Pattern Anal. Mach. Int. **22**(7), 733–737 (2000)
30. A Yilmaz, O Javed, M Shah, Object tracking: a survey. Surv. **38**(4), 13–20 (2006)
31. C Bibby, I Reid, Real-time tracking of multiple occluding objects using level sets, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, Los Alamitos, 2010), pp. 1307–1314
32. Y Shi, W Karl, Real-time tracking using level sets, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2 (IEEE, Los Alamitos, 2005), pp. 34–41
33. A Elgammal, R Duraiswami, Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. Proc. IEEE. **90**, 1151–1163 (2002)
34. B Silverman, Density estimation for statistics and data analysis. Monogr. Stat. Appl. Probability. **1**, 1–22 (1986)
35. A Harvey, V Oryshchenko, Kernel density estimation for time series data. Int. J. Forecasting. **28**, 3–14 (2012)
36. A Yezzi, A Tsai, A Willsky, A statistical approach to snakes for bimodal and trimodal imagery, in *Proceedings of the Seventh IEEE International Conference on Computer Vision, Volume 2 of ICCV 1999* (Kerkyra, 20–27 Sept 1999), pp. 898–903
37. G Hripcsak, A Rothschild, Agreement, the F-measure, and reliability in information retrieval. J. Am. Med. Inf. Assoc. **12**(3), 296–298 (2005)
38. M Krishnaveni, V Radha, Quantitative evaluation of segmentation algorithms based on level set method for ISL datasets. Int. J. Comput. Sci. Eng. **3**(2), 2361–2369 (2011)
39. L Dice, Measures of the amount of ecologic association between species. Ecology. **26**(3), 297–302 (1945)
40. K Dobbin, R Simon, Optimally splitting cases for training and testing high dimensional classifiers. BMC Med. Genomics. **41**(7), 1–31 (2011)

doi:10.1186/1687-5281-2013-42
Cite this article as: Frost and Tapamo: **Detection and tracking of moving objects in a maritime environment using level set with shape priors.** *EURASIP Journal on Image and Video Processing* 2013 **2013**:42.