Received: 12 December 2019 | Revised: 29 August 2020 | Accepted: 31 August 2020

DOI: 10.1002/rob.21985

REGULAR ARTICLE

WILEY

Object detection, recognition, and tracking from UAVs using a thermal camera

Frederik S. Leira | Håkon Hagen Helgesen | Tor Arne Johansen | Thor I. Fossen

Department of Engineering Cybernetics, Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, Trondheim, Norway

Correspondence
Frederik S. Leira, Department of Engineering Cybernetics, Autonomous Marine Operations and Systems, Norwegian University of Science and Technology, O. S. Bragstads Plass 2D, Trondheim, Sør-Trøndelag 7034, Norway.
Email: frederik.s.leira@ntnu.no

Funding information
Norges Forskningsråd,
Grant/Award Numbers: 223254, 269480

Abstract
In this paper a multiple object detection, recognition, and tracking system for unmanned aerial vehicles (UAVs) has been studied. The system can be implemented on any UAVs platform, with the main requirement being that the UAV has a suitable onboard computational unit and a camera. It is intended to be used in a maritime object tracking system framework for UAVs, which enables a UAV to perform multiobject tracking and situational awareness of the sea surface, in real time, during a UAV operation. Using machine vision to automatically detect objects in the camera's image stream combined with the UAV's navigation data, the onboard computer is able to georeference each object detection to measure the location of the detected objects in a local North-East (NE) coordinate frame. A tracking algorithm which uses a Kalman filter and a constant velocity motion model utilizes an object's position measurements, automatically found using the object detection algorithm, to track and estimate an object's position and velocity. Furthermore, a global-nearest-neighbor algorithm is applied for data association. This is achieved using a measure of distance that is based not only on the physical distance between an object's estimated position and the measured position, but also how similar the objects appear in the camera image. Four field tests were conducted at sea to verify the object detection and tracking system. One of the flight tests was a two-object tracking scenario, which is also used in three scenarios with an additional two simulated objects. The tracking results demonstrate the effectiveness of using visual recognition for data association to avoid interchanging the two estimated object trajectories. Furthermore, real-time computations performed on the gathered data show that the system is able to automatically detect and track the position and velocity of a boat. Given that the system had at least 100 georeferenced measurements of the boat's position, the position was estimated and tracked with an accuracy of 5–15 m from 400 m altitude while the boat was in the camera's field of view (FOV). The estimated speed and course would also converge to the object's true trajectories (measured by Global Positioning System, GPS) for the tested scenarios. This enables the system to track boats while they are outside the FOV of

the camera for extended periods of time, with tracking results showing a drift in the boat's position estimate down to 1–5 m/min outside of the FOV of the camera.

KEYWORDS
aerial robotics, computer vision, position estimation

1 | INTRODUCTION

The recent increase of commercial availability of small unmanned aerial vehicles (UAVs) has led to the use of UAVs in many different applications, such as inspections of structures, surveillance, and search and rescue.

UAV related research often emphasizes the UAV's capabilities as a remote sensing platform (Nonami, 2007). In this regard, cameras have become a common and useful remote sensing instrument for UAV platforms. Moreover, thermal cameras have recently become smaller, lighter, and cheaper, making them readily available for use in UAVs. Equipping a low-cost UAV with a thermal camera has therefore become viable, and is useful since it will be able to see and observe parts of the environment differently (and sometimes more clear) compared with visual spectrum cameras. In addition, several processing platforms have become both small and power efficient enough to be placed onboard UAVs for online processing of data. This calls for novel algorithms, not only for processing the collected video data after a UAV flight, but also for analyzing the (thermal) image data onboard and in real time. Performing the image processing onboard and in real time can extend the range of UAV operations, as the UAV will not be required to continuously maintain a stable communication link with the ground station to transfer the thermal images for processing.

In a scenario where there are several (moving) objects of interest that one wants to keep track of in an Earth-fixed coordinate frame using a single UAV equipped with a camera, it quickly becomes challenging to perform this manually with an increasing number of tracked objects. Objects need to be revisited from time to time to verify where they have moved from their expected positions, while at the same time giving all of the tracked objects enough camera time to have a sufficiently good estimate of the movement of the objects. This is a task that is well suited for automation, however, that would require the UAV payload to be able to automatically detect and track multiple objects in real time during flight. It would also require the UAV payload to be able to perform object recognition to distinguish between the UAV revisiting an already detected object and the UAV detecting a novel object.

The process of monitoring multiple moving objects is often referred to as multitarget tracking, and many different approaches to the problem are found in the literature. This problem is not restricted to UAVs as there exist many different remote sensing platforms. Nevertheless, studying solutions that can be applied for thermal cameras in UAVs is still useful and necessary since, for example, radar-based multitarget tracking is not directly applicable. Section 1.1 covers the literature in the field of (multi)target detection and tracking in UAVs.

1.1 | Related work

The problem of automated multiobject detection and tracking using a UAV with an onboard computer and camera can be separated into three different subproblems. First, in order for the process to be automated, robust, and reliable, machine vision techniques for automatic object detection need to be developed and implemented. Second, filters or estimators are needed to estimate the position (and possibly the velocity) of the detected objects in a given coordinate frame. Third, there is the problem of data association, that is, how to associate new measurements of objects' position with objects already being tracked. This section will cover some of the recent works on these three subproblems.

The problem of object detection is the problem of having a computer automatically segment an image into objects of interest and nonobject regions. That is, using an algorithm, the computer should be able to say which regions of an image that contains objects of interest. This problem is well studied, and several different approaches are described in the literature. In Qadir, Neubert, and Semke (2011) a template matching with zero mean normalized cross correlation is used to identify objects of interest in an image. This approach requires that the machine vision algorithm is supplied with an image (template) of the object of interest beforesearching novel images for said object. This is a fast and reliable method if the appearance of the object or type of objects of interest is not changing over time and/or changing with the angle and distance the object is viewed from. However, if the object can take on different appearances, and is nonsymmetric, the number of templates needed to perform the object detection increases quickly. The computation time needed to detect objects using this approach is directly proportional to the number of different templates used for an object, and also grows with the resolution of the template and image. Furthermore, in some scenarios this approach is not viable due to the fact that the appearance of the object is not exactly known before detecting the object. Teuliere, Eck, and Marchand (2011) use a combination of color segmentation (filtering an image based on color) with a Continuously Adaptive Meanshift (CAMshift) algorithm. The CAMshift algorithm was first introduced in Bradski (1998), and is an algorithm that adaptively finds the position and orientation of an object of interest in an image. However, the algorithm requires that the image already is segmented into object and nonobject regions, hence the object detection step itself in Teuliere et al. (2011) is completely dependent on the accuracy of the color segmentation step. Segmenting an image based on color and/or intensity is suitable for many applications, but often requires manual and individual tuning for specific scenarios and scenery. Furthermore, this technique will easily generate false positives (segmenting parts of the image as an object although there is no object to be found) because it treats everything in a scene with a specific color or intensity as an object of interest. Some variation of the color segmentation technique for object detection in UAVs can also be found in Shah, Hakeem, and Basharat (2006), Sengupta et al. (2010), and Štěpán, Krajník, Petrlík, and Saska (2019). These approaches handle false positives by having robust algorithms that will filter out the false positives. This could, for instance, be done by observing that the detections of actual objects are persistent, whereas detections of false positives often are not, or by filtering out detected objects with a shape or contour different from what the algorithm is expected to find.

Another approach to object detection is based on optical flow in a sequence of images, and examples where this is utilized can be found in Rodríguez-Canosa, Thomas, delCerro, Barrientos, and MacDonald (2012) and Helgesen, Stendahl Leira, Johansen, and Fossen (2016). This is an effective but often computationally heavy approach, and is mainly used for detection of moving (i.e., not stationary) objects. There is also a group of algorithms that utilize machine learning to detect objects. There exist a large variety of object detection approaches within this group, but they all have in common that they are taught which objects to detect with a training data set. Chapelle, Haffner, and Vapnik (1999) use example images of humans to train a support vector machine (SVM). The SVM projects an object's feature descriptor into a subdimensional feature space, and the training process adjusts the projection parameters in such a way that the discrimination between object and nonobject is maximized. In recent years convolutional neural network (CNN) has also gained a lot of attention, and is now one of the most popular machine learning methods for object detection. While the performance of SVM approaches will greatly depend on the human design variables, for example, which features and how many should be used to classify an image, the CNN training process trains the network to find and inherently use the most discriminating features. The CNN works as a network of neurons, and an input image propagates through the network by emulating the response of individual neurons to visual stimuli. The output of the CNN is the classification result (object or nonobject) of the input image. This approach is demonstrated in Rodin et al. (2018) for detection of boats and humans at sea in thermal images taken from a UAV. The major issue with the machine learning approaches is that their accuracy is greatly affected by the size and quality of the training set (Doherty & Rudol, 2007; Gaszczak, Breckon, & Han, 2011; Portmann, Lynen, Chli, & Siegwart, 2014; Viola & Jones, 2001). This also implies that it is difficult to make this group of object detectors able to detect objects whose appearance is not consistent or already well documented.

The detection and tracking framework presented in Kalal, Mikolajczyk, and Matas (2012), Tracking–Learning–Detection (TLD), is an open-source framework which, given a region containing an object in an initial image frame, detects the location of said object throughout the duration of the image sequence (for as long as the object is within the image frame). This is done by decomposing the long-term tracking process into three subproblems: tracking, learning, and detection. That is, the tracker follows the object from frame to frame. The detector localizes all appearances that have been observed so far and corrects the tracker if necessary, and the learning module estimates the detector's errors and updates it to avoid these errors in the future. The major weakness of the TLD framework is that the tracking results are greatly affected by the initially chosen region for the object to track, hence care has to be taken as to how this region is (automatically) chosen. Pestana, Sanchez-Lopez, Saripalli, and Campoy (2014) present a successful implementation of the TLD framework on a small multirotor UAV, but their system requires the operator to manually select an object to track and follow.

For the object tracking problem, several solutions using UAVs have been proposed. Barber, Redding, McLain, Beard, and Taylor (2006) use a ground station computer for image processing and object detection in combination with a recursive least square filter to estimate an object's position. The filter converges in approximately 40 measurements, and the object's position estimate is reported to be within 5 m of the object's actual position, albeit flying at a relatively low altitude (100 m). In addition to estimating an object's position, they simultaneously estimate the bias offset angles of the mounting of the camera gimbal. This is done by flying in circles around the object while solving a minimization problem. However, the system is assuming that there is only one single object to track, and that said object is stationary in the duration of the flight. Similar approaches to tracking, but extended to tracking multiple stationary objects, are described in Štěpán et al. (2019) and Goodrich et al. (2008).

Dobrokhodov, Kaminer, Jones, and Ghabcheloo (2006) propose a solution for single-object tracking using UAVs and offline image processing. A nonlinear parametrically varying (NLPV) filter is applied to estimate both the object's position and velocity. The NLPV filter combines the advantages of a parametrically varying model structure and the nonlinear autoregressive exogenous moving average (NARMAX) algorithm. With a UAV altitude of 500 m, an accuracy of 10–40 m in the positional estimate and 0.1–0.4 m/s in the velocity estimate is reported. Although the proposed solution is able to track a moving object, a method to expand this system to the case of multiple object tracking is not presented.

Prevost, Desbiens, and Gagnon (2007) perform object tracking by using an extended Kalman filter (EKF) to estimate an object's position, velocity, and course based on measurements of the object's position. These estimates are then used to calculate an optimal predicted object movement for a short-time horizon into the future. This is an effective approach to enable the UAV to plan its future movement; however, the work presented in Prevost et al. (2007) is only tested in simulation. Furthermore, it is only tested in the case where there exist continuous measurements of the tracked object's position throughout the whole tracking process.

Helgesen et al. (2016) use a Kalman filter and a linear motion model to estimate an object's position and velocity based on measurements of both the object's position and velocity (acquired using optical flow). The results are based on a small data set containing thermal images of a boat gathered from a UAV flying at 100 m altitude. Initial results show that even with limited visual data, the estimated position of an object is within 15 m of its actual position, and the speed estimate is within 1 m/s of the object's actual speed. However, the object's velocity estimate is prone to noise in the UAV's attitude measurements.

When multiple objects are tracked simultaneously, the problem of data association arises. This is a problem where object detections have to be associated with objects already being tracked, or as a novel object entering the image frame. Niedfeldt and Beard (2013) propose a multiple object tracking framework which performs tracking and data association simultaneously based on the Random Sample Consensus (RANSAC) algorithm. RANSAC is a powerful algorithm used in many machine vision problems (Niedfeldt & Beard, 2013), and is based on the concept that a data set contains so-called “inliers” and “outliers”. Inliers are data points which can be explained by some set of model parameters. The outliers are data points that do not fit with the set of model parameters, hence should ideally not be used when trying to estimate the model parameters. Typically algorithms would estimate the model parameters using the total data set; RANSAC, however, assumes that given a small set of inlier data points, there exists a procedure which can estimate the parameters of a model that optimally explains or fits the data. Niedfeldt and Beard (2013) and Niedfeldt and Beard (2014) present how this can be a powerful tool when performing multiple object tracking of an unknown number of dynamic objects. The strength of the approach is in its ability to handle a huge amount of false positives and false negatives from the object detection module. However, the weakness of the system is that knowing exactly how many objects currently are being tracked is difficult, as one object may have several plausible tracks, and as a consequence, several possible model parameters.

In some tracking applications, sensors which generate several distinct measurements across each object's spatial expression can be used. There is a lot of information to be extracted from such detection clusters, but when doing so it is crucial to interpret this information in a way that is consistent with the tracked object's physical state and shape. Granström, Baum, & Reuter (2016) give an excellent overview of extended target tracking approaches. Extended target tracking uses a model of objects' physical shape and incorporates it with estimating the objects' motion. Since each object can cause a cluster of detections, the process of data association is even harder, as there are more possible solutions to the data association problem. However, knowledge of the objects' physical shape and orientation can be useful information and is in fact often utilized when performing the data association step. While not directly within the extended target tracking group of tracking algorithms, it is also not unusual to use sensors which will generate several nonspatial (e.g., object color) measurements per object. The measurements not directly linked to an object's position are often combined into the object's appearance model. Kuo, Huang, and Nevatia (2010; color and histogram), Ullah, Mohammed, Cheikh, and Wang (2017; a CNN classifier), and Yinghui and Jianjun (2009; color histogram and image region covariance) all present different ways of incorporating an object's appearance model in the tracking and data association process. However, a general framework in which different appearance models can be included seems to be lacking.

In Wu, Thangali, Sclaroff, and Betke (2012), the traditional “detection-tracking” approach is replaced by stating the object detection, tracking, and data association problem as a single-objective function. The main concept of the approach is to have a robust object detector mark separate blobs in each image frame in a video sequence, and then stating the data association as a network flow problem where the goal is to optimize the flow going through the network. The strength of this approach is that the whole data set is considered simultaneously when solving the data association and tracking process. However, this is also its weakness as the computational burden becomes large for longer video sequences. Furthermore, the approach would have to be extended to deal with false positives in the object detection algorithm.

The nearest-neighbor (NN) method is regarded as one of the most straightforward approaches to data association (Konstantinova, Udvarev, & Semerdjiev, 2003). Given several possible detections for the position of an object, the associated detection is assumed to be the detection “closest” to the estimated or predicted position of the tracked object. The term “closest” is based on a predefined measurement of the distance to the object, for example, this could be the distance in image pixels between the object measurement and the estimated object position in the image frame. However, the NN method is prone to end up in nonoptimal solutions for the data association problem when multiple objects are being tracked simultaneously. This is because the order of associating measurements with tracked objects can affect the overall result (Konstantinova et al., 2003). To address the shortcomings of the NN method a global-nearest-neighbor (GNN) algorithm can be applied (Konstantinova et al., 2003). The GNN algorithm seeks to find the global minimum solution in the case of multiple object tracking, effectively avoiding the nonoptimal solution which the NN method can possibly generate. This solution is optimal in the way that the total distance between the measured object positions and the estimated position of the tracked objects is minimized. The GNN algorithm is a more computationally heavy approach, but it is otherwise very similar to the NN algorithm. Helgesen, Leira, Johansen, and Fossen (2017) extend upon the work of Helgesen et al. (2016) by presenting three different observation models and tracking methods, and evaluating their performance in a multiobject tracking scenario using the GNN approach to data association. While the performance of all of the presented observation models is similar (96% correct data association in the presented case study), the performance is majorly dependent on the objects' distance to each other. This dependency can in many cases be mitigated by incorporating objects' visual appearance into the data association process.

1.2 | Contributions

This paper extends the automatic detection, recognition, and tracking framework for objects in the sea surface presented in Leira, Trnka, Fossen, and Johansen (2015) and Leira, Fossen, and Johansen (2015) by improving the tracking module's ability to recognize objects on the UAV's revisitation of the objects it is tracking. Furthermore, the tracking process is extended from tracking objects only in the image frame to the case where the UAV is able to track the position and velocity of multiple objects of interest in Earth-fixed coordinates. This means that the tracked objects can be outside the field of view (FOV) for prolonged periods of time while their positions are still being estimated by the tracking system. The UAV payload used for experiments is also modified to meet the requirement of onboard and real-time image processing. Although parts of the object detection and tracking system are similar to already published methods, the combination of algorithms and UAV payload presented in this paper constitutes a state-of-the-art system in the robustness in which the position, speed, and course of the objects are estimated based on thermal images from a fixed-wing UAV. That is, the system is robust in the sense that the system will keep tracking objects for extended periods of time even when they are outside the FOV of the UAV camera, and also in the sense that it uses thermal image features for object recognition. The latter enables the system to better distinguish between objects when tracking multiple objects close to each other, and to recognize objects re-entering the FOV of the camera. Finally, this study also includes numerous experimental results from field tests containing (multi)object tracking scenarios with a fixed-wing UAV.

The remainder of this paper is organized as follows. First, a machine vision algorithm for automatically detecting objects in real time in a processing unit placed onboard the UAV is presented. Second, a tracking algorithm is developed to track the position and velocity of the automatically detected objects. This approach covers both the process of estimating an object's position and velocity based on image-based detections, as well as data association, that is, associating new detections with objects already being tracked. Section 5 covers the details of the field tests conducted to test the object detection, recognition, and tracking algorithms presented in this paper, while Section 6 covers the results of the conducted flights. Finally, conclusions are presented and ideas for future development of the detection, recognition, and tracking algorithms are discussed.

2 | SYSTEM OVERVIEW

This section describes a framework for object tracking with UAVs equipped with a thermal camera, with example implementations of each module designed to detect and track objects at the ocean surface in a maritime environment. However, note that adjusting these modules (or reimplementing them in a different manner) using the same framework structure would allow the system to be used in different applications in different environments.

Object tracking is here defined as the action of using the UAV to keep track of multiple objects' position and velocity in an Earth-fixed coordinate frame during the UAV flight. The tracking process does not only consider already known object positions, but also focus on detecting, recognizing, and tracking potential objects of interest with unknown positions. This can be achieved by using an onboard sensor, typically a camera, mounted in a pan-tilt gimbal, and a path controller. The overall proposed object tracking system is illustrated in Figure 1. The UAV object detection, tracking, and recognition module, also illustrated in Figure 1, is a machine vision algorithm running onboard the UAV supplying the path planner with estimates of object's position and velocity. These estimates can be found using different methods. An autonomous method is to have an onboard computer analyze the images coming from the onboard camera, automatically detecting, recognizing, and tracking objects of interest over a series of images. An example of such a system is described in Section 3. This approach is applied without any prior knowledge of the location and number of objects of interest, but the user will typically input to a machine vision module some features of the objects that are of interest (e.g., size or shape). Another option is the scenario where you know the location of the objects that you want to track, but still want to do some verification and/or surveillance of the objects. In this case the object detection, tracking, and recognition module is initialized with a list of the positions and velocities of all objects, and the path planner should then make sure that these objects will be observed by the UAV. This approach can be combined with the previously mentioned autonomous approach. That is, the object detection, tracking, and recognition module can update and/or

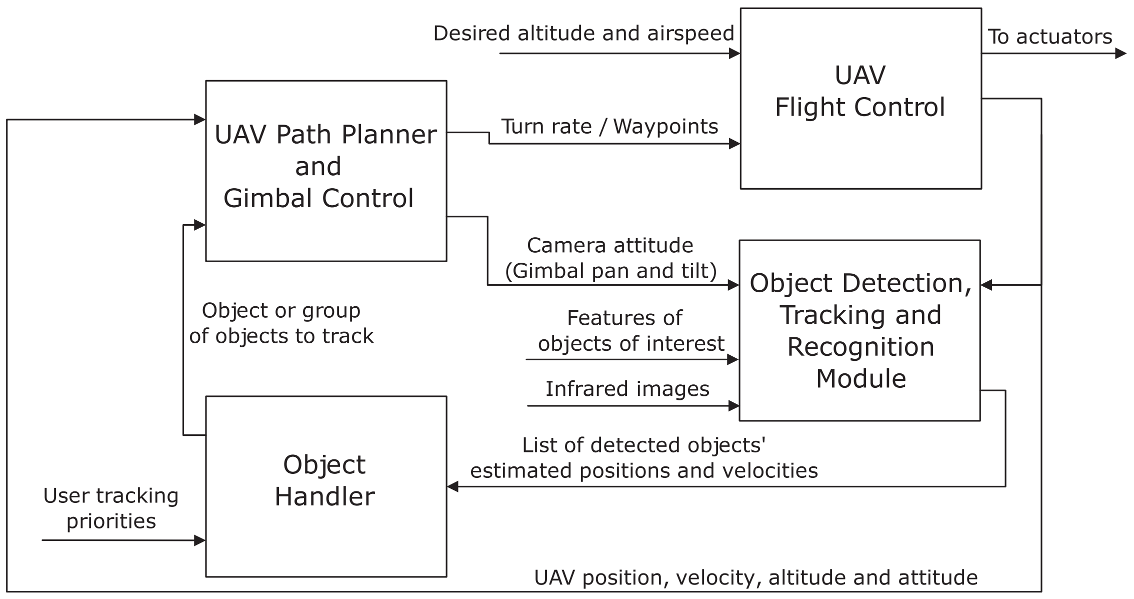

Desired altitude and airspeed
UAV Flight Control
To actuators
UAV Path Planner and Gimbal Control
Turn rate / Waypoints
Camera attitude (Gimbal pan and tilt)
Object Detection, Tracking and Recognition Module
Features of objects of interest
Infrared images
Object or group of objects to track
List of detected objects' estimated positions and velocities
Object Handler
User tracking priorities
UAV position, velocity, altitude and attitude

FIGURE 1 Overall object tracking system description. A path planner module is combined with an onboard object detection, tracking, and recognition module to detect and track objects using a UAV platform. UAV, unmanned aerial vehicle

verify an object's position and velocity when the machine vision module is automatically detecting the object, effectively estimating its current position and velocity.

The path planner is a module which seeks to determine a turning rate or a series of way-points for the UAV, as well as the optimal gimbal orientation along the pan and tilt axes, that will allow the UAV to successfully track objects of interest. The control outputs calculated by the path planner module are given to the UAV's flight controller (autopilot), which in turn has its own algorithms to control the dynamics of the UAV. The path planner module can (and should) be adjusted for different tracking scenarios. A scenario where the UAV is tracking several slow-moving objects, such as vessels at the ocean's surface, requires a different path-planning approach as opposed to tracking a fast-moving vehicle constantly changing its velocity. An example of a path-planning module that is intuitive and easily configurable can be found in Skjong, Nundal, Leira, and Johansen (2015). When performing the tracking of multiple objects simultaneously, the Object Handler is the module responsible for choosing which object to observe at any given time. This module will be highly application dependent, while the other modules are more generic in nature. An example Object Handler module used with this system, which is based on an occupancy grid map and Bayesian probabilistic reasoning, can be found in Leira, Johansen, and Fossen (2017). This reference shows how the Object Handler and UAV path planner modules can be implemented such that they enable the UAV to do a fully autonomous flight, only affected by the user's predefined tracking priorities. These two modules could also be implemented to include a search component in the path planning, meaning that the path planner could choose to search for new undetected objects instead of only focusing on the objects currently being tracked. The user could also change the UAV's tracking priorities online and in real time to accommodate needs that arise due to, for example, a change in the environment.

3 | OBJECT DETECTION AND RECOGNITION

The Object Detection, Tracking, and Recognition module uses images from an onboard camera to automatically do segmentation of the images. That is, using machine vision, the module's task is to segment pixels into foreground (object) or background (nonobject). To automatically detect objects of interest the machine vision algorithm developed in Leira, Fossen et al. (2015) is applied, with the extension of a feature vector which aids the system in the process of classification and recognition for data association.

The method is conceptually similar to the Canny edge detector (Canny, 1986), in the sense that it is fundamentally based on edge detection. The image is first smoothed using a kernel approximating a Gaussian distribution. The motivation for this is to reduce the thermal noise present in the image, which in turn will make edge detection easier as it results in a low-pass filtered gradient image (Canny, 1986). This was found to make the edge detector more robust than when performed on the raw thermal images. The result of smoothing an image showing a big boat (length of 56 m), a rigid-hulled inflatable boat (RHIB), and a small buoy can be seen in Figure 2b.

To detect the edges in the resulting smoothed image, its gradient image is calculated using the Prewitt operator (Soille, 2003). The resulting gradient image is shown in Figure 2c. It is seen that the big boat, the RHIB, and the small buoy are clearly visible after these image processing operations. Further it is apparent that the waves and ripples in the ocean in addition to some of the noise in the image are also still visible, albeit smaller in magnitude (intensity) than the objects of interest. By simply thresholding the gradient image with

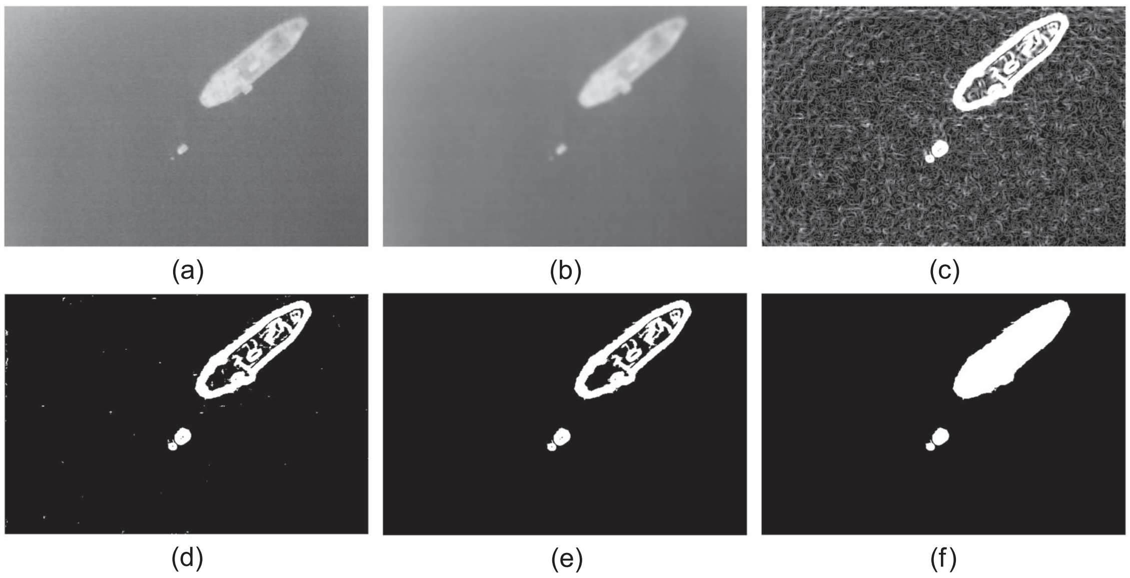


FIGURE 2 An example of a thermal image going through each step in the machine vision algorithm for automatic object detection

an appropriate thresholding value T_g, the visible ripples and waves can be removed. The result of thresholding the gradient image is seen in Figure 2d.

Looking at Figure 2d it is obvious that some of the blobs clearly do not originate from any object of interest (i.e., the small dots scattered across the image), and therefore have to be filtered out. To filter out the unwanted blobs from the image a connected component algorithm (Suzuki & Abe, 1985) is used to group and label components together in blobs. Furthermore, the area of each blob is then calculated, and blobs with a smaller or larger area than what is expected from an object of interest are then removed from the image. The result of this process is seen in Figure 2e and the resulting image is hereby referred to as the binary image, **B**, of the original image **I**.

The resulting image (Figure 2e) contains three remaining blobs which are of further interest. The three blobs are marked as detected objects, and are now ready for classification. The center positions of the remaining blobs are calculated in both the image frame and in the world frame, and then passed on to the tracking module as position measurements for the objects.

The detection step provides the position of objects of interest in the image. However, since the detector is using edge detection, the areas of an image that is highlighted as interesting will often only contain the exterior edges of an object. When performing, for instance, recognition based on characteristics, such as size, average infrared radiation, and overall form, it is crucial that the whole object is evaluated. To expand the detections to also include the interior of the objects of interest, an algorithm that seeks to fill holes in the binary image (Soille, 2003) shown in Figure 2e is applied. The result of applying this algorithm can be seen in Figure 2f, and the image with filled contours is hereby denoted $\mathbf{B}_{\text{filled}}$.

Using the location of the bright pixels in the binary image seen in Figure 2f, the pixels that make out the object in the original image (Figure 2a) can be analyzed. In this paper, the object characteristics used for recognition are the observed object area, detected average object infrared radiation and one of the scale, rotation, and translation invariant moments proposed by Hu (1962). Using scale, rotation, and translation invariant features when describing objects observed from the air are very important, as the altitude, angle, and orientation that the object is viewed from are constantly changing. Note that both the observed object area and the detected average object infrared radiation will be, to some extent, invariant to the scale, rotation, and translation of the object. However, the detected average object infrared radiation will be somewhat dependent on the UAV's distance and attitude relative to the object, hence if the UAV altitude or attitude was to change drastically, the reference for the object's detected average infrared radiation should be adjusted. It should be kept in mind that the recognition method presented can be used for a large variety of other object characteristics and be modified to include other object features.

The invariant moments presented in Hu (1962) are based on the following moment function:

$$m_{pq} = \sum_{x,y \in \boldsymbol{O}} x^p y^q \boldsymbol{B}_{\text{filled}}(x, y),\ p, q = 0, 1, \ldots, \tag{1}$$

where x and y indicate a pixel location in the horizontal and vertical directions in the image. m_{pq} is referred to as a $(p + q)$th order moment of the image region $\boldsymbol{O}$, where $\boldsymbol{O}$ is defined as the set of pixels inside the object's bounding box. Note that since $\boldsymbol{B}_{\text{filled}}$ is a binary image, the 0th moment (m_{00}) simply becomes the number of positive pixels in the image region. This in turn can be interpreted as the pixel area of the object. This is an effective parameter to use when classifying objects, especially when pixel area is converted into the metric area of the object through the use of onboard altimeter and attitude and heading reference system (AHRS) measurements. The metric area of an object can provide a good indication of the object's type and inertia. Assuming that the UAV is flying approximately straightforward (low roll and pitch values), the metric area of an object can be found by multiplying the pixel area of the object (m_{00}) with a factor $a(h)$, where $a(h)$ is defined as square meters per pixel when the camera is at altitude h. Note that it would be more accurate to also account for the UAV's attitude, but for simplicity a scaling dependent only on UAV altitude is used. This factor is given by the thermal camera lens and characteristics. Hence, the metric area of an object can be approximated by

$$A \approx a(h) m_{00}, \tag{2}$$

Central moments are also used in the calculation of Hu's invariant moments, and are given as

$$\begin{aligned} \mu_{pq} &= \textstyle\sum_{x,y \in \boldsymbol{O}} (x - \bar{x})^p (y - \bar{y})^q \boldsymbol{B}_{\underset{x,y \in \boldsymbol{O}}{\text{filled}}}(x, y),\ p, q = 0, 1, \ldots, \\ \bar{x} &= \tfrac{m_{10}}{m_{00}},\ \bar{y} = \tfrac{m_{01}}{m_{00}}, \end{aligned} \tag{3}$$

where $(\bar{x}, \bar{y})$ is the centroid of the image region. Note that the central moments are invariant to translation. This is readily seen by observing that the central moment is just m_{pq} shifted to the centroid of the image region. Now, to get scale-invariant moments, the normalized central moments are introduced. The normalized central moments are given as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}},\ \gamma = (p + q + 2)/2,\ p + q = 2, 3, \ldots. \tag{4}$$

Using the normalized central moments, Hu introduced seven moments invariant to rotation, translation and scale. However, research has shown that for objects represented with a small amount of pixels (less than 100 × 100 pixels), these moments may vary when the image is scaled and/or rotated. Furthermore, Flusser (2005) show that the higher order moments (ϕ_{2-7}) vary much more than the lower order moment (ϕ_1) for images with low resolution. Since the resolution of most thermal imaging cameras is quite low, objects of interest will not be represented by a lot of pixels. Because of this, only the first invariant Hu moment is included in the feature vector.

$$\phi_1 = \eta_{20} + \eta_{02} \tag{5}$$

It should be noted that ϕ_1 is analogous to the moment of inertia around the object's centroid, where the object's pixel intensities would be the physical density.

To calculate the detected average object infrared radiation, the settings of the thermal-imaging camera are utilized. That is, the camera can be set to capture infrared radiation intensities in a range from a minimum and a maximum intensity ($IR_{\min}$ and $IR_{\max}$). Furthermore, assuming that the output from the camera is an image where each pixel is represented by n bits, each pixel can take on a value between 0 and 2^n. This means that the detected infrared radiation of an area covering only 1 pixel will be

$$IR = \frac{I(x, y)}{2^n}(IR_{\max} - IR_{\min}) + IR_{\min}. \tag{6}$$

Expanding this to calculate the detected average infrared radiation over a detected object, we get

$$IR_{\text{avg}} = \frac{\frac{\sum_{x,y \in \boldsymbol{O_p}} I(x, y)}{m_{00}}}{2^n}(IR_{\max} - IR_{\min}) + IR_{\min}, \tag{7}$$

where

$$\boldsymbol{O_p} = \{x, y \in \boldsymbol{O} \,|\, \mathbf{B}_{\text{filled}}(x, y) = 1\}$$

$\boldsymbol{O}$ is the set of pixels in the object's bounding box. Hence, the image region $\boldsymbol{O_p}$ is given by the set of pixels in the detected object blob in the binary image $\mathbf{B}_{\text{filled}}$. Note that the detected infrared radiation is found from pixel intensities in the original image $\mathbf{I}$.

Combining the detected average object infrared radiation with the invariant moments, we can represent any group of pixels in an image using the feature vector

$$\mathbf{X} = \begin{bmatrix} A \\ IR_{\text{avg}} \\ \phi_1 \end{bmatrix}. \tag{8}$$

In the case that a new object appears in the image frame and is not matched to any of the objects currently being tracked, the object's feature vector $\mathbf{X}$ is calculated for the first m images where the whole object is visible. The average of these m feature vectors, $\bar{\mathbf{X}}$, is then stored onboard the UAV for future reference. However, since an object's detected average infrared radiation might drift over time (the thermal camera heating up) or change due to the weather or the UAV's variable distance and attitude relative to the object, this component of the feature vector is continuously averaged over the last m measurements associated with an object. The object's measured feature vector is then used when detected objects are associated with objects already being tracked. This enables the system to better recognize already tracked objects. The details of the data association process are described in Section 4.2.

4 | OBJECT TRACKING

The object tracking module is responsible for estimating and keeping track of the position and velocity of the detected objects. This is done by using Kalman filters to estimate and predict the position and velocity for each object. If an object detection is not likely to originate from any of the objects currently being tracked, the tracking module assumes that a novel object has been detected, and creates a new Kalman filter instance associated with this object. Further detections of the tracked object are then used as measurements in the Kalman filter to estimate the object's position and velocity. This means that the tracking module also has to be able to associate new detections with already existing tracked objects. This is done by associating object detections to the most likely among the tracking gaits. A tracking gait is defined as the complete state history of an object, that is, the history of its positions and velocities. An overview of the object tracking approach presented in this section is illustrated in Figure 3.

4.1 | Kalman filter

The Kalman filter implemented in the detection, recognition, and tracking module is based on Ali and Terada (2010), and utilizes a motion model based on the assumption that the UAV is tracking objects located on a flat surface (e.g., the sea surface) and moving with a constant velocity. This yields the following linear equations of motion:

$$\begin{aligned} x^{\text{obj}}_{k+1} &= x^{\text{obj}}_k + \Delta t V^{\text{obj}}_{x,k} + \tfrac{1}{2}\Delta t^2 w_k, \\ y^{\text{obj}}_{k+1} &= y^{\text{obj}}_k + \Delta t V^{\text{obj}}_{y,k} + \tfrac{1}{2}\Delta t^2 z_k, \\ V^{\text{obj}}_{x,k+1} &= V^{\text{obj}}_{x,k} + \Delta t w_k, \\ V^{\text{obj}}_{y,k+1} &= V^{\text{obj}}_{y,k} + \Delta t z_k, \end{aligned} \tag{9}$$

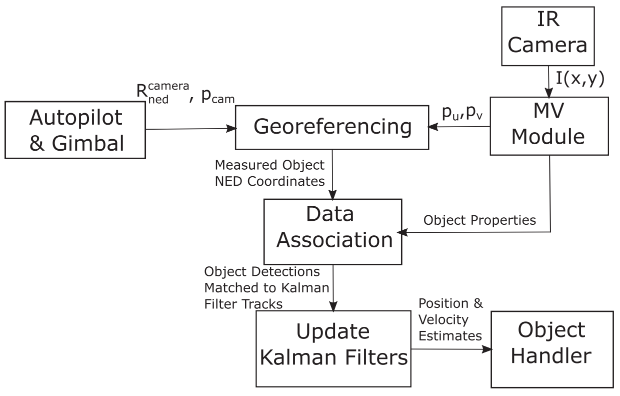

IR Camera
I(x,y)
MV Module
$R^{\text{camera}}_{\text{ned}}$, p_{cam}
Autopilot & Gimbal
Georeferencing
p_u, p_v
Measured Object NED Coordinates
Data Association
Object Properties
Object Detections Matched to Kalman Filter Tracks
Update Kalman Filters
Position & Velocity Estimates
Object Handler

FIGURE 3 An overview of the tracking process. The machine vision (MV) module receives an infrared image $\boldsymbol{I}(x, y)$ from the infrared camera, and detects the centroid (p_u, p_v) of objects of interest within the image. These data are combined with navigation data from the autopilot and gimbal (camera attitude, $\boldsymbol{R}^{\text{cam}}_{\text{ned}}$ and position $\boldsymbol{p}_{\text{cam}}$) to georeference the object's pixel location. Having found the object's measured location in the NED frame, the object is associated with a specific Kalman filter track based on the measured position and the object's features (size, infrared radiation, etc.). The Kalman filter track associated with the measurement is then used to update the Kalman filter estimates. IR, infrared; NED, North-East-Down

where x_k^{obj}, y_k^{obj}, $V_{x,k}^{obj}$, and $V_{y,k}^{obj}$ are the position and linear velocitis of an object in North-East (NE) coordinates (x is north and y is east) at time step k and Δt is the time passed from time steps k to $k+1$. w_k and z_k are the Gaussian white noise terms representing a change in velocity of an object. This yields the following model in state-space form:

$$\begin{aligned} \mathbf{x}_{k+1}^{obj} &= \boldsymbol{A}\mathbf{x}_k^{obj} + \boldsymbol{E}\mathbf{w}_k, \\ \mathbf{y}_k^{obj} &= \boldsymbol{C}\mathbf{x}_k^{obj} + \mathbf{v}_k. \end{aligned} \tag{10}$$

The matrices **A**, **E**, and **C** are equal to

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{E} = \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix}, \quad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \tag{11}$$

and we have that

$$\mathbf{w}_k = \begin{bmatrix} w_k \\ z_k \end{bmatrix}, \mathbf{v}_k = \begin{bmatrix} q_k \\ r_k \end{bmatrix}, \tag{12}$$

w_k and z_k are here as previously defined, while q_k and r_k are the Gaussian white noise terms which represent noise and errors in the measurement of an object's position. For the two-dimensional motion described in (9), the state vector $\mathbf{x}_k^{obj}$ and the measurement $\mathbf{y}_k^{obj}$ are equal to

$$\mathbf{x}_k^{obj} = \begin{bmatrix} x_k^{obj} \\ y_k^{obj} \\ v_{x,k}^{obj} \\ v_{y,k}^{obj} \end{bmatrix}, \quad \mathbf{y}_k^{obj} = \begin{bmatrix} y_{x,k}^{obj} \\ y_{y,k}^{obj} \end{bmatrix}, \tag{13}$$

where $y_{x,k}^{obj}, y_{y,k}^{obj}$ are the detected object's measured positions at time step k.

Note that $y_{x,k}^{obj}, y_{y,k}^{obj}$ are the measured positions of the object in NE coordinates, while the detection step described in Section 3 only yields the centroid of an object in image frame coordinates (pixel location of the centroid). Hence, to find the NE coordinates that this pixel location corresponds to, we need to project the pixel location onto the NE plane. This can be done by assuming the pinhole camera model (Ma, Soatto, Kosecka, & Sastry, 2012) and using the equations found in Leira, Fossen et al. (2015). That is, given the pixel coordinates of the centroid of an object ($p_{u_k}^{obj}, p_{v_k}^{obj}$) and by assuming that the object is located at the sea surface, the object's position (north and east coordinate) in the NE frame can be found by the following equation:

$$\frac{1}{\lambda_k}\begin{bmatrix} x_k^{obj} \\ y_k^{obj} \\ 1 \end{bmatrix} = \boldsymbol{G}_{xy_k}^{-1}\boldsymbol{K}^{-1}\begin{bmatrix} p_{u_k}^{obj} \\ p_{v_k}^{obj} \\ 1 \end{bmatrix}, \tag{14}$$

where $\boldsymbol{G}_{xy_k}$ is the extrinsic camera parameter matrix (camera attitude and position) given by

$$\boldsymbol{G}_{xy_k} = \begin{bmatrix} \boldsymbol{r}_{1_k} & \boldsymbol{r}_{2_k} & -\boldsymbol{R}_{\text{ned}_k}^{\text{cam}}\boldsymbol{p}_{\text{cam}_k} \end{bmatrix}, \tag{15}$$

$\boldsymbol{r}_{1_k}$where and $\boldsymbol{r}_{2_k}$ are the first- and second-column vectors of $\boldsymbol{R}_{\text{ned}_k}^{\text{cam}}$, which is the rotation matrix for the rotation from the North-East-Down (NED) frame to the UAV's camera frame. $\boldsymbol{p}_{\text{cam}_k}$ is the position of the camera center given in NED frame coordinates. λ_k is a scaling factor required to isolate the normalized homogeneous coordinates of the NE position of the object $\left[x_k^{obj}, y_k^{obj}, 1\right]^T$. $\boldsymbol{K}^{-1}$ is the inverse of the intrinsic camera parameters (Ma et al., 2012), that is, the matrix transforming the normalized homogeneous pixel coordinates $[p_{u_k}^{obj}, p_{v_k}^{obj}, 1]$ from image coordinates (pixels) to camera coordinates (world units).

At each time step k the matrices $\boldsymbol{R}_{\text{ned}_k}^{\text{cam}}$ and $\boldsymbol{p}_{\text{cam}_k}$ can be calculated using the navigation data received from the onboard autopilot. Note that since (14) will simultaneously depend on data from both the autopilot's navigation data and pixel location of a detected object from the camera's image data, the two data sources should be time synchronized as accurately as possible.

A Kalman filter can be used to calculate an estimate, $\hat{\mathbf{x}}_k^{obj}$, of an object's state at time step k using measurements of the object's position in the form of $\mathbf{y}_k^{obj}$ and predictions based on the object's current estimated state. For the Kalman filter to be efficient, the magnitude of the measurement noise, $\mathbf{v}_k$, and the process noise, $\mathbf{w}_k$, should be tuned to fit the sensor accuracy and the motion model uncertainty, respectively. That is, we define the following stochastic terms:

$$\mathbf{R}_k = \mathbf{E}\left[\mathbf{v}_k\mathbf{v}_j^T\right] = \begin{bmatrix} \sigma_{q_k}^2 & 0 \\ 0 & \sigma_{r_k}^2 \end{bmatrix}, \quad \mathbf{Q}_k = \mathbf{E}\left[\mathbf{w}_k\mathbf{w}_j^T\right] = \begin{bmatrix} \sigma_{w_k}^2 & 0 \\ 0 & \sigma_{z_k}^2 \end{bmatrix}. \tag{16}$$

Further, when a new object is detected a Kalman filter is initialized with the object's measured position as the filter's position states $\hat{x}_k^{obj}, \hat{y}_k^{obj}$. Since the object's velocity cannot be known (or estimated) from only one measurement, the object's estimated velocity is initialized as 0. Moreover, since the measured object position is uncertain due to being based on only one noisy measurement, and the estimated velocity being unknown, the filter's initial covariance matrix should be tuned to reflect this fact. That is, the Kalman filter must be tuned such that it is aware that an object's initial position and velocity are uncertain. The Kalman filter's initial a posteriori estimate of the covariance matrix is given by the following equation:

$$\mathbf{P}_{0|0} = E\left[\left(\mathbf{x}_0^{obj} - \hat{\mathbf{x}}_0^{obj}\right)\left(\mathbf{x}_0^{obj} - \hat{\mathbf{x}}_0^{obj}\right)^T\right] = \begin{bmatrix} \sigma_{x_0^{obj}}^2 & 0 & 0 & 0 \\ 0 & \sigma_{y_0^{obj}}^2 & 0 & 0 \\ 0 & 0 & \sigma_{v_{x,0}^{obj}}^2 & 0 \\ 0 & 0 & 0 & \sigma_{v_{y,0}^{obj}}^2 \end{bmatrix}. \tag{17}$$

Choosing appropriate values for $\mathbf{R}_k$, $\mathbf{Q}_k$, and $\mathbf{P}_{0|0}$ is detrimental to the accuracy and the consistency of the filter's position and velocity estimates. To find appropriate values for $\mathbf{R}_k$, the result presented in Section 6 is a good basis. More specifically, the results show the error in the georeferenced position for 4903 object detections and can be used to find mean values for the measurement error obtained in georeferencing. Note that if the UAV is in perfectly leveled flight, and the camera is pointing directly downwards, the altitude of the UAV will not affect the accuracy of a measured object position. This is not the case if the UAV is turning or changing the altitude, and the accuracy of the measured object position would scale proportionally with the UAV altitude. Experimental results from Helgesen, Leira, Bryne, Albrektsen, and Johansen (2019) also show that the error in georeferencing increases with the roll and pitch angles of the UAV. Hence, $\mathbf{R}_k$ would ideally be a function of both the UAV camera's attitude ($\boldsymbol{R}^{\text{cam}}_{\text{ned}_k}$) and altitude ($h_k$), that is, $\mathbf{R}_k\left(\boldsymbol{R}^{\text{cam}}_{\text{ned}_k}, h_k\right)$. However, the relationship between the attitude and the georeferencing error is nonlinear and difficult to generalize. Therefore, R_k is designed to be proportional with the altitude of the UAV because this is true in most situations (perfectly leveled flight is not common due to wind and other environmental effects and a pitch angle of a few degrees often occur). A value of $R_k = 0.05h_k$ was found to be an approximate model for the error distribution experienced in Section 6. Note that both $\mathbf{Q}_k$ and $\mathbf{P}_{0|0}$ can be difficult to identify, as objects in the real world usually do not move exactly, like described in (9). First, since a new track is initialized using only one object detection, a reasonable choice for the upper 2×2 matrix of $\mathbf{P}_{0|0}$ could be $\sigma_{q_k} = \sigma_{r_k} = \sigma_{x_0^{\text{obj}}} = \sigma_{y_0^{\text{obj}}}$. The remainder of the filter's parameters could potentially be tuned in accordance with prior knowledge of the objects dynamics. In this study the remaining tuning parameters were set to fixed values which were used in all tracking scenarios. This was done to investigate the filter's robustness and consistency for different object dynamics.

4.2 | Data association

When tracking objects, one of the most crucial parts is to be able correctly perform data association and recognition. In the present system, a GNN approach similar to the one found in Konstantinova et al. (2003) is utilized to perform data association. This involves using the following distance metric for the distance between measurement i and tracking gait j:

$$D_{i,j} = (1-\gamma)\tilde{\boldsymbol{y}}_{i,j}^T \boldsymbol{S}_j^{-1}\tilde{\boldsymbol{y}}_{i,j} + \gamma \tilde{\mathbf{X}}_{i,j}^T \boldsymbol{\Gamma} \tilde{\mathbf{X}}_{i,j}, \tag{18}$$

where

$$\tilde{\boldsymbol{y}}_{i,j} = [\boldsymbol{y}^{\text{obj},i} - \hat{\boldsymbol{y}}^{\text{obj},j}] \text{ and } \tilde{\mathbf{X}}_{i,j} = [\mathbf{X}_i - \bar{\mathbf{X}}_j].$$

Note that the first term is the sum of the squares of two independent Gaussian random variables, hence it will have a chi-squared probability distribution with two degrees of freedom (Konstantinova et al., 2003). This fact will be used later on to discard unlikely measurements. Here, $\boldsymbol{y}^{\text{obj},i}$ is the measurement i (given as a measurement of an object's position in NE coordinates), $\hat{\boldsymbol{y}}^{\text{obj},j}$ is the a priori predicted position of object j, and $\mathbf{S}_j$ is the prediction's associated covariance matrix. Both the predicted position and the covariance matrix are given by the Kalman filter estimating the position of tracking gait j. In this paragraph the time index k is dropped for simplicity of notation. The tuneable parameter γ is in the interval (0, 1) when the whole object (all edges) is contained within the image frame and 0 otherwise. This is because the detected object's feature vector should only be calculated when the whole object is visible. $\mathbf{X}_i$ is the feature vector corresponding to measurement i, and $\bar{\mathbf{X}}_j$ is the previously stored feature vector associated with object j. $\boldsymbol{\Gamma}$ is a tunable weighting matrix, allowing the difference between each measured and stored object feature to affect the total distance between measurement i and object j differently.

Conceptually there are two reasons to include the object features in the distance measurement. The first reason is to avoid associating measurements of a novel object with an object that is already being tracked. If object features were not included in the distance measurement, this could happen if the novel object is located on the same location as the predicted position (not the actual position) of an object that is already detected. However, by using object features it is (sometimes) possible to detect that the object is not the object that was expected to be at said location, hence registering it as a new object in the object database. The second reason is that if there are two or more objects in the same image, noise in the measured object position, and uncertainty in the position estimate of objects could easily confuse the association process. However, if the objects differ in appearance, using the object features to aid the data association is effective, as the appearance of an object is not as prone to noise as the measured position. Moreover, since a so-called hard measurement association is conducted through the GNN method, robustness is key to avoid mixing tracks and visual features are needed.

To associate a measured object position with the most likely among the tracking gaits, a matrix $\mathbf{D}$ expressing the distances from all n measurements to all m tracking gaits is calculated. Hence, the matrix $\mathbf{D}$ takes on the following form:

$$\mathbf{D} = \begin{bmatrix} D_{1,1} & D_{1,2} & \cdots & D_{1,m} \\ D_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ D_{n,1} & \cdots & \cdots & D_{n,m} \end{bmatrix}. \tag{19}$$

If the distance $D_{i,j}$ exceeds some threshold $d > 0$, $D_{i,j}$ is set to infinity. This is because in the case of

$$D_{i,j} \geq d, \tag{20}$$

the measurement i is not very likely to be a measurement originating from the object in tracking gait j. The value for the threshold d should be based on the chi-squared probabilities for two degrees of freedom since the first term of $D_{i,j}$ as previously mentioned will have a chi-squared distribution. This also means that Γ should be tuned to scale the contribution from the object's feature vector according to the same chi-squared probability. An approximation for this can be achieved by tuning Γ to make, for example, 95% of an example set of an object's features contribute less than the 5% critical value (5.991) for chi-squared distributions with two degrees of freedom.

Note that if all values along column j are equal to infinity, it is concluded that a measurement for tracking gait j is not present. To cope with this, column j should be removed from $\mathbf{D}$, and the predicted object position should be used as the best available estimate for tracking gait j. Furthermore, if all values along a row i is equal to infinity, it is assumed that there exists no probable tracking gait for measurement i. In other words, this is most likely a measurement originating from a new, not yet tracked object. Hence, row i should be removed from the matrix $\mathbf{D}$, and a new tracking gait should be instantiated with measurement i as the initial position.

After calculating $\mathbf{D}$ and removing superfluous rows and columns, the data association problem is a matter of finding the combination of distances $D_{i,j}$ that yield the global minimum distance. This implies that the combination which is selected assigns exactly one measurement to exactly one tracking gait, in a way such that the total distance between all measurements and their assigned tracking gaits is the shortest achievable distance. This is a well-studied problem and can be solved by applying the Kuhn–Munkres algorithm (Bourgeois & Lassalle, 1971) to the modified matrix $\mathbf{D}$.

5 | UAV AND FIELD TEST

To test the performance of the framework described in the previous sections, several field tests were conducted. The system was implemented on the CruiserMini platform shown in Figure 4a. It is a small fixed-wing UAV with the focus of supplying a low-cost aircraft for short-endurance ($\lesssim$2 h) UAV flights. The CruiserMini was equipped with a payload based on the work in Leira, Trnka et al. (2015), with some improvements. More specifically, the avionics and payload consist of the open-source Pixhawk 2 autopilot (Pixhawk, 2018), a pan-tilt gimbal, a thermal camera, and an onboard single-board computer capable of performing real-time image processing. The single-board computer used in the flight tests conducted in this study is an Odroid XU4 (HardKernel, 2016), which has an ARM-based 2.0 GHz octa-core CPU and 2 GB RAM. The thermal camera used is an FLIR Tau2 640 (FLIR-Systems, 2018), and by using a USB device (TEAX-Technology, 2018) it supplies the onboard computer with digital 640 × 512 pixels of radiometric data images 7.5 times/s. The camera is sensitive to the long-wave infrared spectral band (7.5–13.5 μm) with a sensitivity of <50 m K, and has a resolution of 14 bits/pixel. An example image captured with the FLIR Tau2 camera is shown in Figure 6a. The camera is placed inside a retractable R-BTC88 (MicroUAV, 2011) gimbal, which can be controlled either automatically by the onboard single-board computer or from a manual controller located in the ground station. The onboard computer is connected to both the autopilot and the digital interface of the Tau2 camera, and stores the data from these two devices (telemetry and thermal images) locally during flight testing.

For the tests conducted in this study it was the performance of the detection, recognition, and tracking modules of the system that was emphasized, hence the UAV was either remotely controlled by a UAV pilot or the path planning was done manually preflight. That is, the path planner was not connected to the real-time feedback loop illustrated in Figure 1. Hence, during the field tests the path-planning module was not doing any real-time onboard calculations, and the autopilot was simply following a predefined flight path or manual instructions from the UAV operator.

All of the tests were carried out operating from the shoreside at Agdenes outside Trondheim in Norway. Further, the gimbal was set to a fixed position, pointing directly downwards, during all of the flight tests. The gathered data set consists of four flight tests where some of the captured thermal images contain the boat Telemetron (an RHIB) and/or a small fishing boat. Both objects can be seen in the thermal image shown in Figure 6a. Telemetron is approximately 8 m long and is shown in Figure 4b, while the fishing boat is of a similar

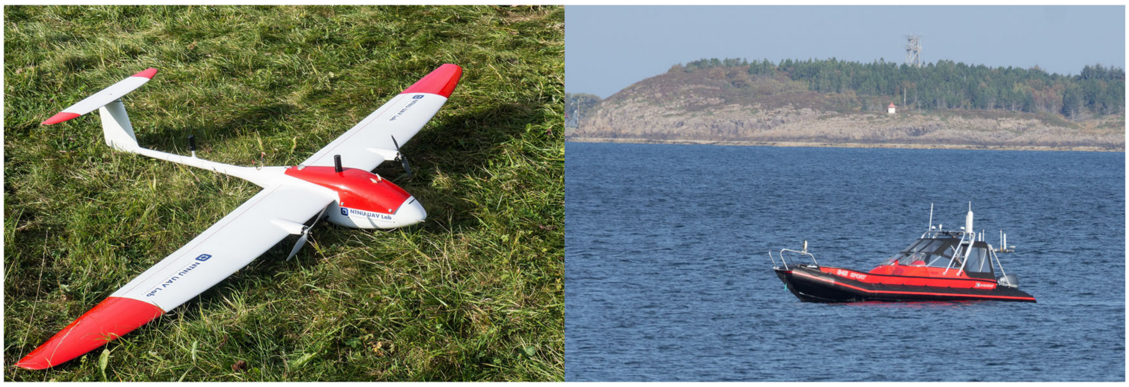

(a) The CruiserMini (b) Telemetron

FIGURE 4 The fixed-wing UAV (CruiserMini) used to implement and test the system described in this paper together with one of the boats (Telemetron) being tracked during the flight tests. UAV, unmanned aerial vehicle [Color figure can be viewed at wileyonlinelibrary.com]

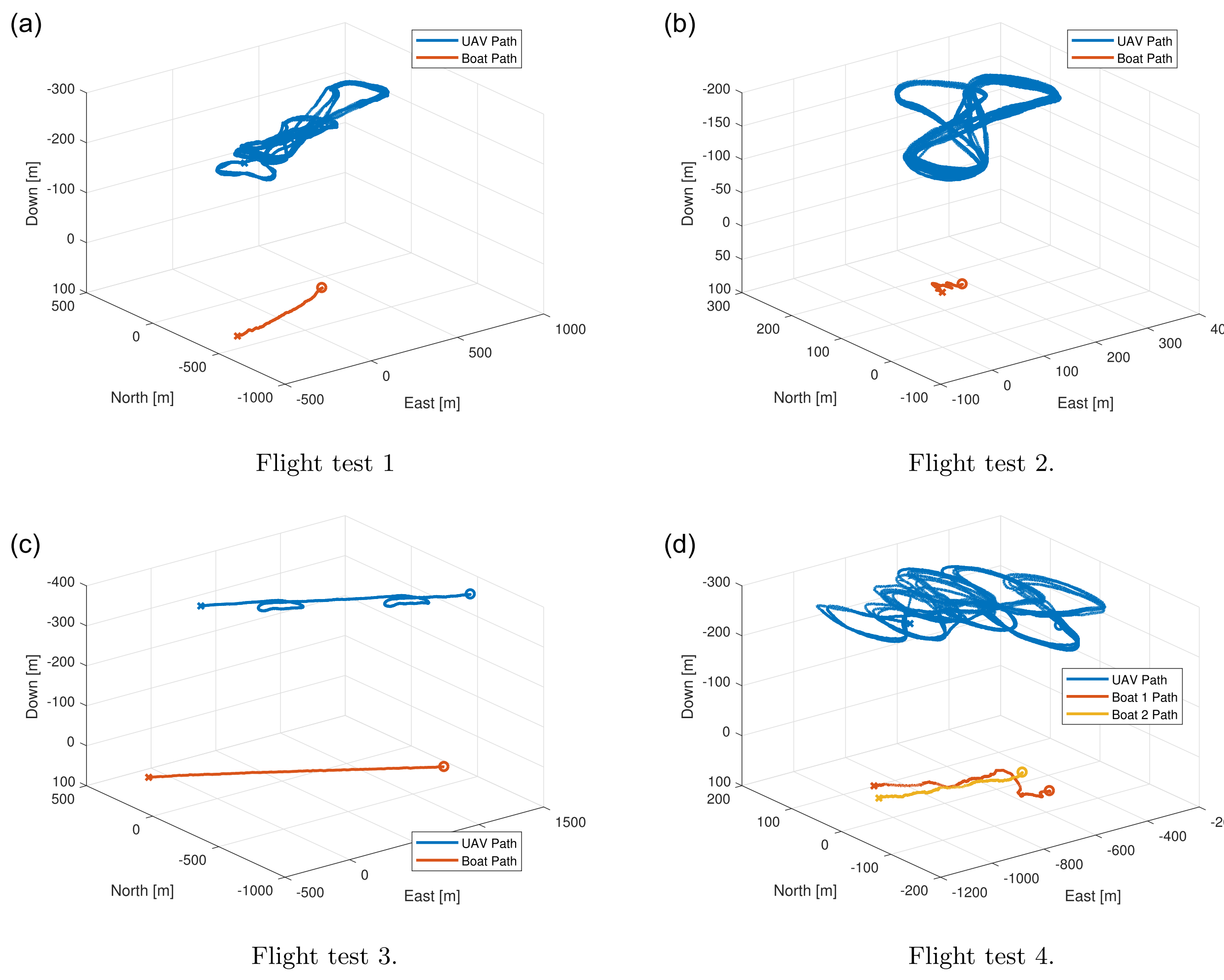


FIGURE 5 The UAV flight path (blue) together with the path of the tracked boat(s) (orange and yellow)) during the object tracking periods for flight tests 1–4. Each flight test has its own reference point for their NED frames, chosen to be in the proximity of the first object detection. The beginning (circle) and the end (cross) of each path are also shown. NED, North-East-Down; UAV, unmanned aerial vehicle [Color figure can be viewed at wileyonlinelibrary.com]

shape but smaller in size. The position and velocity of the Telemetron and the fishing boat were measured by Global Positioning System (GPS) devices located in the boats, effectively supplying a relatively accurate measurement of their true position, speed, and course (referred to as the "ground truth"). Since the ground truth is measured with a single-frequency GPS device, the accuracy of these measurements can be expected to be within 3 m of the object's actual position. In Section 6, the GPS measurements from the boats are simply referred to as measured position, speed, and course. Also note that the position estimates (originating from georeferencing the position of the boats in the thermal images) are referred to as the georeferenced position of the boats. Finally, in the results, Telemetron is denoted as 'Boat' in flights 1–3, and 'Boat 1' in flight 4. The fishing boat is denoted as 'Boat 2' in flight test 4.

The following results are the results of the detection and tracking algorithm tested in an offline real-time computation using

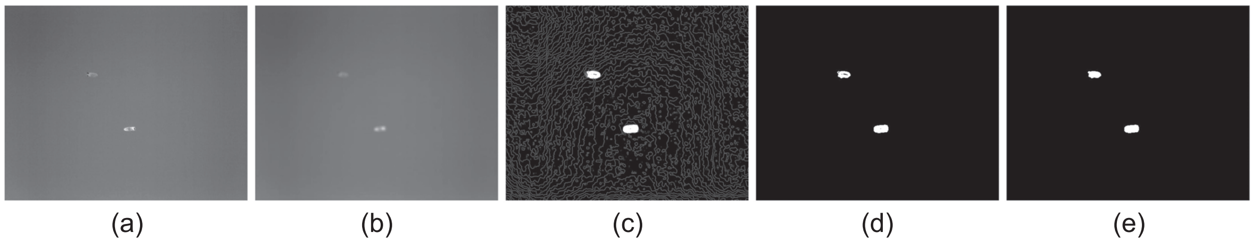


FIGURE 6 The result of applying the image processing algorithm described in Section 3 to one of the images captured during flight

data gathered during the four test flights. Computational tests for the tracking scenarios presented in this paper show that the onboard computer use on average 0.07 s to perform the processing needed for each time step (image analysis, data association, and updating Kalman filters). Since the thermal camera's frame rate is 7.5 frames/s (0.13 s between each image frame), running the object detection, recognition, and tracking module onboard/online is expected to yield identical results as the findings presented below. For demonstrations of real-time and online use of variations of the presented framework the reader is referred to Leira et al. (2017) and Mathisen, Leira, Helgesen, Gryte, and Johansen (2020).

6 | RESULTS

The tracking results presented in this paper are based on data from four test flights. During the four tracking scenarios described in this section the UAV was airborne for a total of 1 h, 1 min, and 50 s, and gathered a total of 27,866 thermal images. A subset of 4903 of the images contained complete thermal signatures of the Telemetron and/or the fishing boat, meaning the whole object was within the camera's FOV. All of the thermal signatures originating from Telemetron and the fishing boat were automatically detected by the system.

By setting the threshold, T_g, described in Section 3 to 230 and removing every detection consisting of less than 100 pixels, the object detection algorithm successfully detected the objects of interest in all of the images where they were fully contained. The threshold T_g could potentially be tuned adaptively, but 230 was found to be a value that consistently removed thermal signatures with a smaller gradient magnitude than the ones caused by the objects of interest.

Due to the ocean being so uniform in the emission of thermal radiation, the system reported only 15 false positives, which by manual inspection were found to be due to distinct thermal signatures whose origin was not identifiable by only looking at the thermal images. These detections were however easily removed through filtering by size and form. Figure 6 illustrates each step in the detection algorithm (tuned as described above) on an image containing two boats.

For the tracking process, the measurement noise variables q and r (measurement noise in the north and east directions) were set to have a standard deviation of $0.05h_k$ m ($\sigma_q = \sigma_r = 0.05h_k$ m). As discussed at the end of Section 4.1 the measurement noise is proportional to the altitude of the UAV. The four flight tests were conducted at altitudes between 200 and 400 m, where experimental data are mostly consistent and show that while the UAV is flying at an altitude h_k, $0.05h_k$ m is a reasonable choice for the standard deviation for the measurement noise. The process noise variables w and z will affect how quickly the Kalman filter adapts to perceived changes in the velocity of the object being tracked. Hence, this parameter should ideally be chosen based on the size and mode (speeding and drifting) of the object being tracked. For Telemetron and the fishing boat, the standard deviation of the process noise was set to 0.2 m/s ($\sigma_w = \sigma_z = 0.2$ m/s). That is, it was assumed that both of the boats had a more or less constant velocity with some smooth minor changes over time. The discretization time for the Kalman filter was set to $\Delta t = 1/7.5$, which is the expected rate at which the thermal camera supplies the onboard computer with a new image (measurement). The Kalman filter was initialized with a covariance matrix set to $(0.05h_k)^2$ m^2 ($\sigma_{x_0^{\text{obj}}} = \sigma_{y_0^{\text{obj}}} = 0.05h_k$ m) for the positional estimates, and 5^2 m^2/s^2 ($\sigma_{v_{x,0}^{\text{obj}}} = \sigma_{v_{y,0}^{\text{obj}}} = 5$ m/s) for the velocities. The filters were also initialized with the first measured object position (the first time the object enters the camera's FOV) as the initial position, and 0 m/s velocity in both the North and the East directions.

Below is a more detailed view on the performance of the object detection and tracking module for each flight.

6.1 | Flight 1

During the first flight test, Telemetron was manually operated to keep a constant low speed (≈ 0.5 m/s) and a fixed heading. The UAV was controlled to loiter at an altitude of 300 m in a so-called eight-pattern over the boat's location, as illustrated in Figure 5a. During the tracking period (24 min), the boat was automatically detected a total of 989 times, which approximately corresponds to only 2 min and 10 s of camera time. For each detection, an NE position of the boat's location was calculated using (14). The georeferenced position was then compared with the boat's measured ("ground truth") position at the instant that the image containing the detection was taken. The Euclidean distance from the boat's measured position to its georeferenced position was then calculated for each boat detection. The error in distance for each measurement can be seen as blue crosses in Figure 7. The pentagonal marker in the NE plot illustrates the tracked boat's measured position and heading at fixed time intervals throughout the tracking process.

Using a Kalman filter as described in Section 4 with these georeferenced position measurements, yielded the tracking trajectory illustrated in Figure 7. It is observed that the initial position estimate is at 20 m from the center of the boat's measured position. However, as more georeferenced measurements of the boat's position are processed by the Kalman filter, the estimated boat velocity and course quickly converge to the boat's actual speed and course. This is clearly illustrated in Figure 7. Comparing the speed plot and the error plot in Figure 7, it is seen that the estimated object speed converges to the measured object speed by the third batch of measurements (the third cluster of georeferenced object detections). The three clusters consist of only 109 georeferenced image boat positions spread over a time period of 2 min and 50 s. In total, 109 images correspond to 14.5 s of camera time, and illustrate the short time needed for the estimates to converge.

After the speed and course estimates have converged to the measured object speed and course, the accuracy of the boat's estimated position does not increase by adding further detections. The boat's position estimate is within 5–15 m of the boat's measured position for most of the 24-min tracking processes, only drifting less than 5 m away from the boat's measured position. This is also the case even though the boat is outside the FOV of the UAV camera for

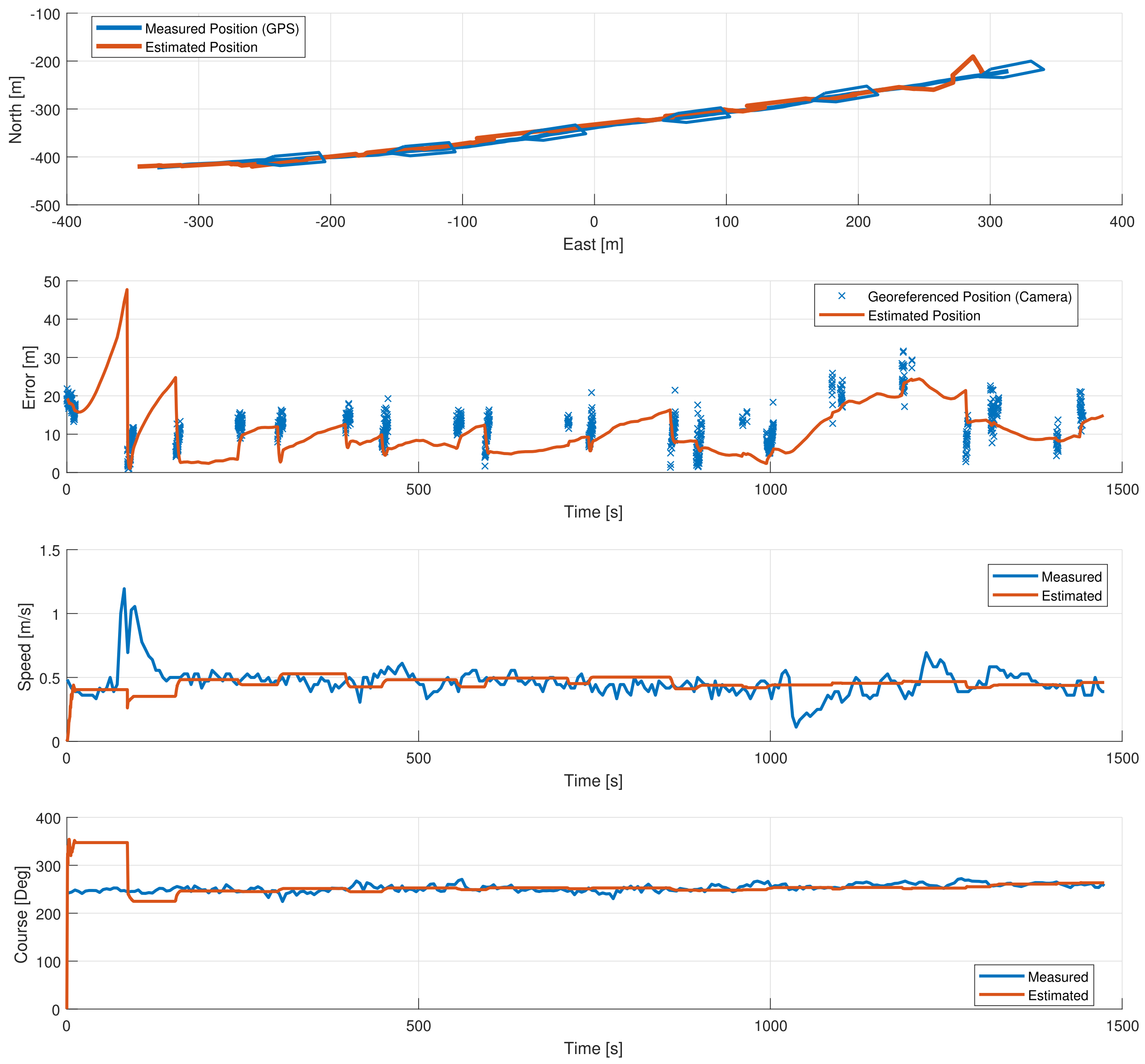


FIGURE 7 Tracking results for test flight 1. GPS, Global Positioning System [Color figure can be viewed at wileyonlinelibrary.com]

up to 70 s at a time. Some of the object detection clusters towards the end of the tracking process have degraded accuracy, which is due to the object being detected while the UAV has a large bank angle (≈40°), causing inaccuracies in the UAV's pose estimate to impact the boat's georeferenced position estimate more than in earlier segments. This illustrates the dependency on accurate navigation estimates in georeferencing as described in Helgesen et al. (2019).

6.2 | Flight 2

In the second flight test Telemetron was drifting with the current, with no input from the boat's operator. As in the first flight test, the UAV was controlled to loiter in a figure of eight above Telemetron's position, as shown in Figure 5b, but now at an altitude of 200 m. During the tracking period for this flight (16 min), Telemetron was automatically detected in a total of 1250 images, with each image giving a measurement of the boat's position in the NED frame by georeferencing the boat's image position. The accuracy of each georeferenced measurement is shown in Figure 8 as blue crosses.

The estimated position, speed, and course of Telemetron using a Kalman filter with the georeferenced boat positions as measurements are illustrated in Figure 8. The initial position estimate is ≈5 m from the boat's measured position, and the position estimate keeps well within one boat length 10 m for the majority of the tracking periods.

As seen from Figure 8, even though the speed estimate of Telemetron converges to the measured speed quickly (again around the 100th Telemetron detection), the estimate of the boat's course never converges. The mean error in the course estimate for the whole tracking period of flight 2 is 48°. The reason for this difference is probably two-fold. First, the boat is moving too slow compared with the magnitude of the measurement variance in each cluster of

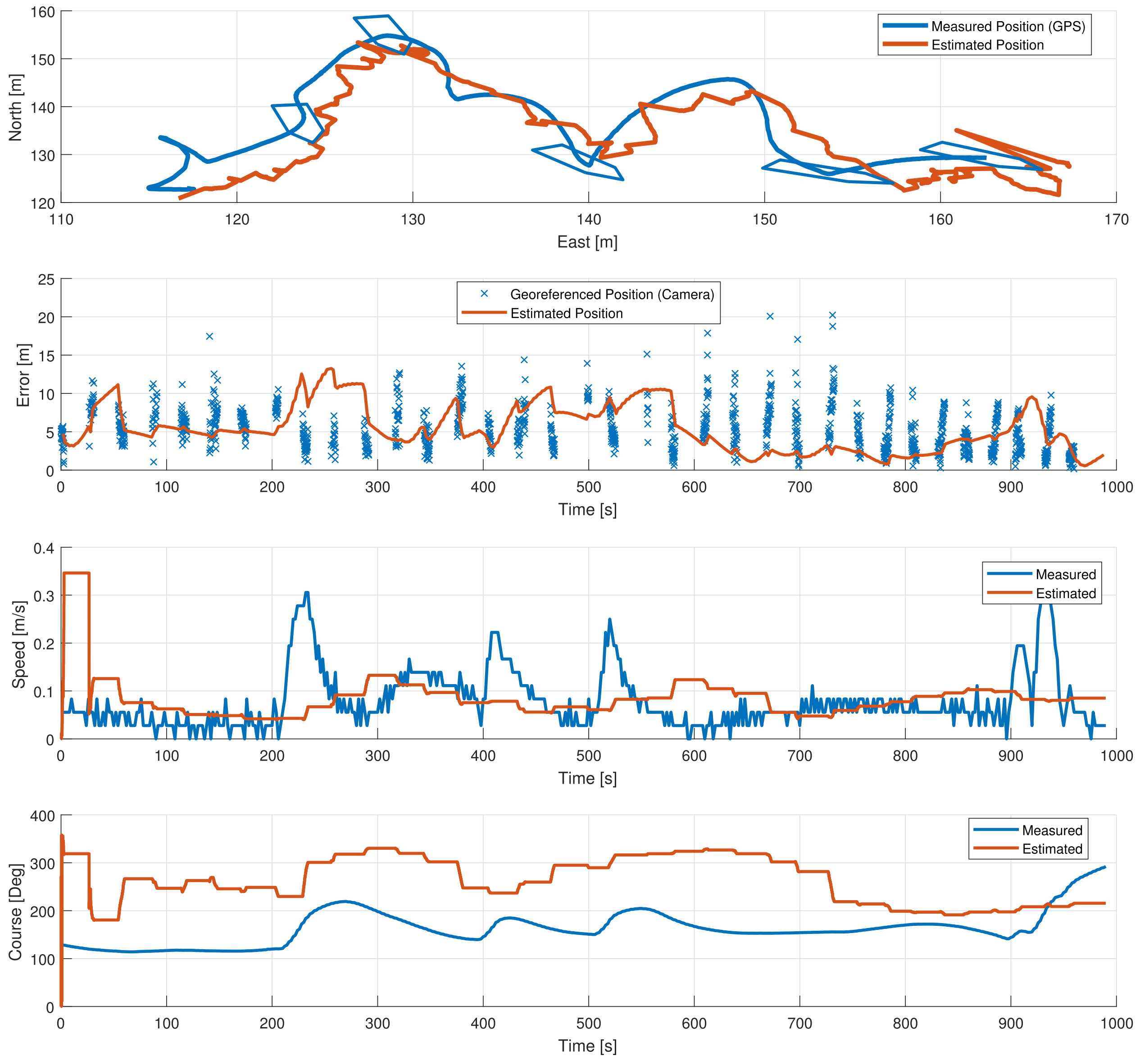


FIGURE 8 Tracking results for test flight 2. GPS, Global Positioning System [Color figure can be viewed at wileyonlinelibrary.com]

georeferenced positions, and second, since the GPS measured position also is noisy the GPS could struggle to identify the boat's actual course. However, with an average speed of 0.078 m/s (both estimated and measured by GPS), the estimate of Telemetron's position would only drift ≈55 m from the boat's actual position if the detection and tracking would rely solely on prediction after the first 100 georeferenced position measurements. Although not ideal, it is safe to say that the UAV would be able to relocate the boat after 15 min of Telemetron being outside the FOV of the UAV's camera, if the Kalman filter was used to predict the boat's future movements.

6.3 | Flight 3

In the third flight test Telemetron was manually controlled to have a higher speed (≈10 m/s) with a constant heading. The UAV was controlled to fly in a straight line in the same direction that the boat was moving, and was then controlled to do a loiter whenever the UAV ended up in front of the boat. This movement pattern is illustrated in Figure 5c. The UAV had an altitude of 400 m during the whole tracking period. The altitude for this test was chosen higher than the other two flight tests to make it easier for the UAV operator to control the UAV such that Telemetron would be visible in the UAV camera's FOV.

During the tracking period for this flight (2 min and 30 s), Telemetron was automatically detected in a total of 264 images (corresponds to 35 s of camera time), spread over three different segments in time. The accuracy of each georeferenced measurement is shown in Figure 9 as blue crosses. The first cluster consists of 88, the second of 104, and the third of 72 image detections of Telemetron.

The estimated position, speed, and course of Telemetron from the Kalman filter with the georeferenced boat positions as input are illustrated in Figure 9. It is observed that the first cluster of georeferenced measurements does not contain enough information for the speed and course estimate to fully converge to the boat's actual speed and course. Although the course and speed estimates are only 6° and 2 m/s from the GPS measured speed and course after the first cluster of measurements, the boat is moving at such a high speed that the position estimate starts to drift from the boat's measured position immediately after Telemetron leaves the FOV. While the course and speed estimates are adjusted close to the measured values once the UAV's camera has the Telemetron within the camera's FOV again (the second cluster of georeferenced measurements), the position estimate has drifted 150 m in 50 s. Note that although it might be difficult to relocate the boat in this scenario relying solely on the information gained in the first cluster of measurements, the search to relocate the boat could be restricted significantly. Given that the boat would keep a constant speed and course, the UAV would simply need to expand the search along the line in the same direction as the course estimate, while using the speed estimate and the filter's certainty to adjust the search area until the boat was relocated.

6.4 | Flight 4

The fourth flight test is a two-object tracking scenario. Both Telemetron and the fishing boat were manually operated to have a low speed (≈1 m/s) and a similar heading. Telemetron crosses the path of the fishing boat once (approximately at the 250-s mark), in addition to moving closer and then further away from the fishing boat. The path of Telemetron (Boat 1) and the fishing boat (Boat 2) for the tracking period of flight test four is seen in Figure 5d. The UAV was controlled to fly in a figure of eight above the two boats, getting

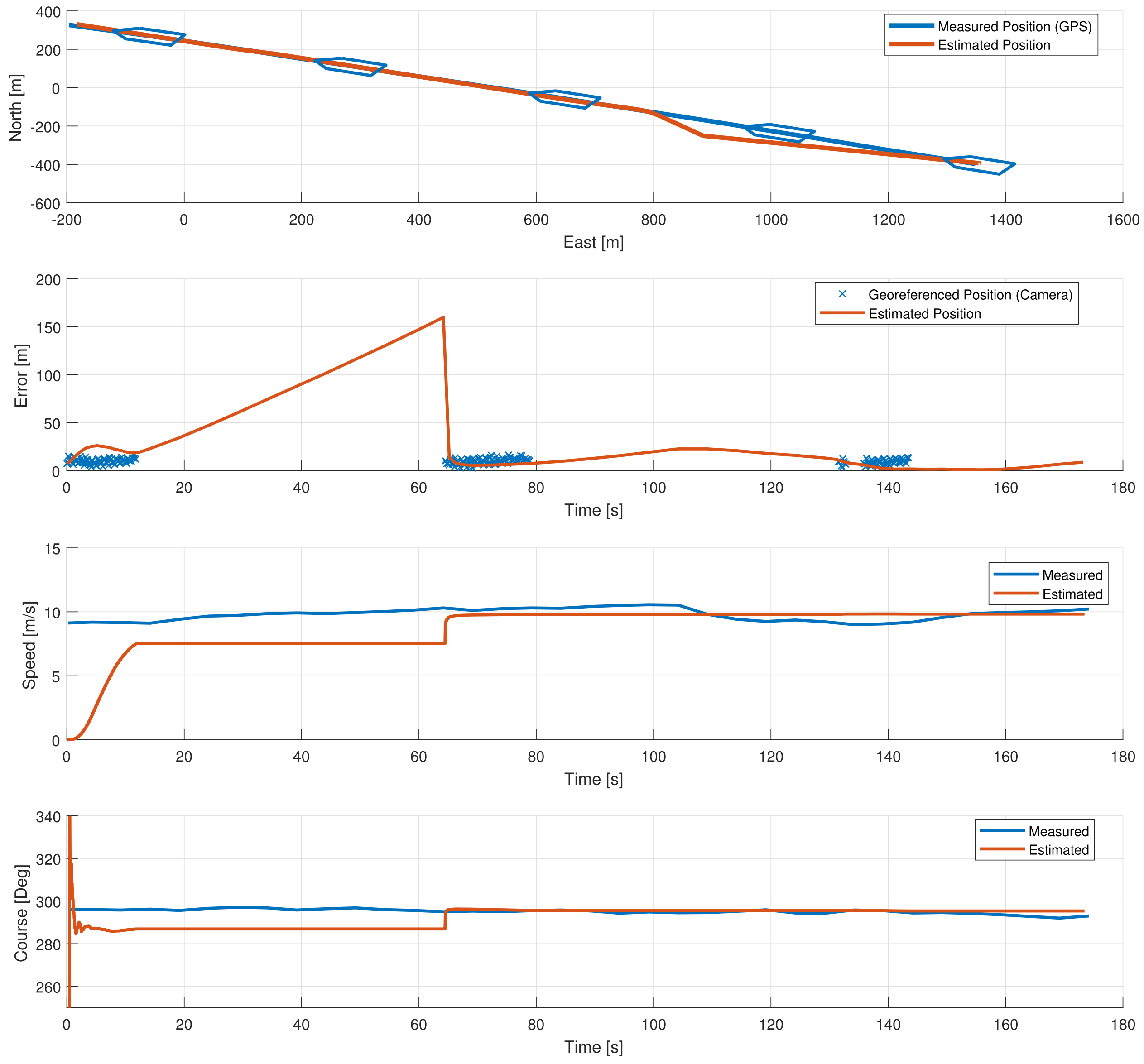


FIGURE 9 Tracking results for test flight 3. GPS, Global Positioning System [Color figure can be viewed at wileyonlinelibrary.com]

intermittently position measurements of both boats. Note that the UAV had an altitude of 300 m during the whole tracking period.

During the tracking period of the two boats (18 min and 50 s) the UAV automatically detected 2400 objects of interest. It was found, by manually classifying all detections, that 1535 of the detections were of Telemetron, and 865 were of the fishing boat. Finally, it was found that 349 of the captured thermal images had both Telemetron and the fishing boat within the FOV of the camera. The feature vector described in Section 3 was calculated for each detection and put into two groups (Boats 1 and 2) based on manual classification. The result can be seen in Figure 10. It is readily seen that, in the case of distinguishing the two boats by appearance, it is the area feature that has the biggest between-group difference, then intensity and finally the invariant Hu moment. This is expected, as the fishing boat is in fact a little bit smaller than Telemetron, and also appears to radiate and reflect less infrared energy. Note that, as expected, the detected average infrared radiation is the feature which has the biggest in-group variance. This fact justifies the choice to use the average of the 10 most recent feature vectors associated with an object as a reference for this feature.

However, since the Hu moments are both size and rotation invariant, and the geometric form of the boats is similar, it is reasonable that the difference in Hu moments between the groups is negligible. Although the area feature would be the most efficient to distinguish the two objects, all of the three features were considered when doing the data association for the tracking process. Each feature's element in the weighting matrix $\boldsymbol{\Gamma}$ was chosen according to their average in-group variance. That is, each feature's weighting element was set such that the expected cost addition from that feature (given that the measurement was compared with the measured object's average feature vector) would be in the same range as the cost addition caused by the distance element. This yielded $\boldsymbol{\Gamma} = \mathrm{diag}(10^{-5}, 10^{-4}, 10^{3})$.

To test the effectiveness of including an image recognition element in the tracking process, the trajectories of the two boats in the scenario shown in Figure 5d were tracked both with γ set to 0 and 0.5. With γ set to 0, an object's appearance is not considered when doing data association, and with γ set to 0.5, an object's visual resemblance to the object being tracked is weighted equally with the object's measured physical distance to the tracked object. One of the critical points in the tracking process of this flight test is when Telemetron crosses the path of the fishing boat, switching which side of the fishing boat Telemetron is closest to. As seen from the tracking results in Figure 11 ($\gamma = 0$), the pure physical distance data association fails at this point, switching tracks between the objects. That is, the Kalman filter that was initially tracking Telemetron ends up tracking the fishing boat and vice versa. It should also be noted that after confusing the two objects and the two boats moving apart from each other again, the Kalman filters continue to track each of their (wrongly associated) respective objects' position correctly.

Setting $\gamma = 0.5$ solves this problem, and even though the cost caused by the associating correctly based on measured physical distances to each of the boats and their estimated position is big, the cost caused by the object's thermal signature and image features when associating incorrectly is bigger. This can be seen from Figure 12, as the object being initially tracked by each of the two Kalman filters keeps constant during the whole tracking process, only associating a total of four measurements incorrectly. This was four measurements of the fishing boat's position associated with Telemetron.

6.4.1 | Flight 4 with real and simulated objects

To further investigate the usefulness of taking an object's visual features into account in the data association process we can complicate the tracking scenario by adding simulated objects. This was done by utilizing the telemetry data from flight 4, and designing the equations of motion for simulated objects such that the objects would have appeared in the camera's FOV if they were to be real

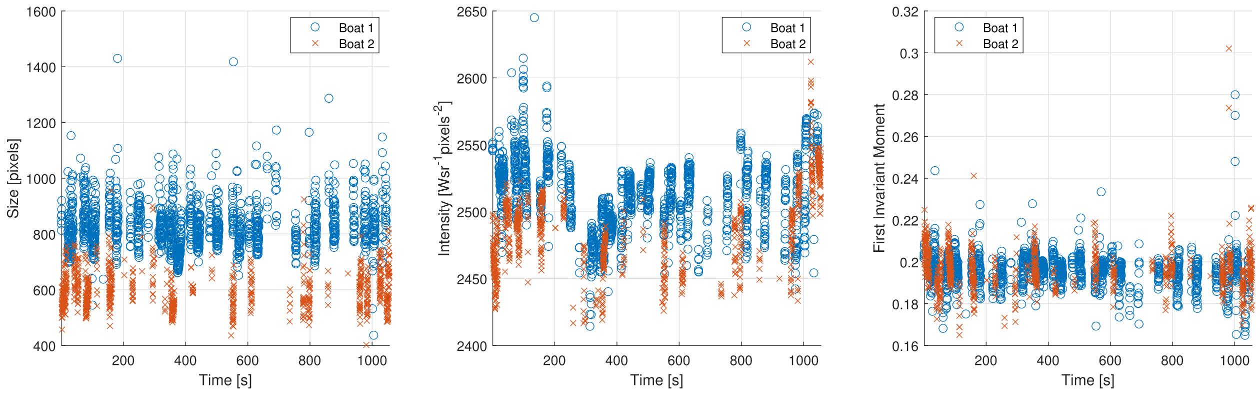


FIGURE 10 Distribution of the object features (size, intensity, and the first invariant Hu moment) for Boat 1 (blue) and Boat 2 (orange) for all detections across flight test 4 [Color figure can be viewed at wileyonlinelibrary.com]

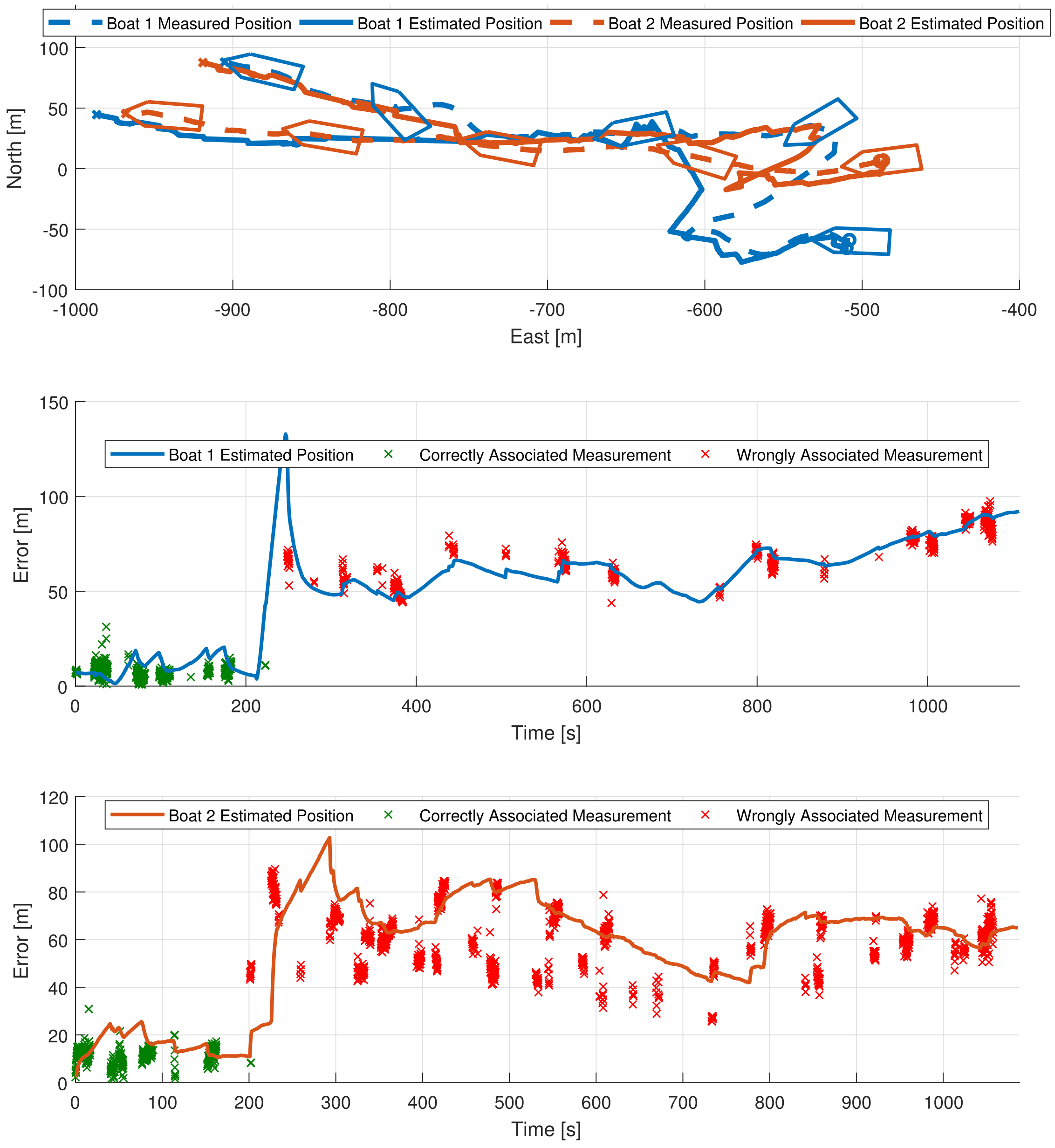


FIGURE 11 Tracking results for test flight 4, using only the distance between the expected and measured position of an object for recognition (data association) [Color figure can be viewed at wileyonlinelibrary.com]

objects. The pixel coordinates of an object position given in the NED frame can be found by the pinhole camera model, that is, in our case the inverse of Equation (14). The object is considered to be within the camera's FOV if the resulting pixel values are within the camera's image frame.

Two simulated objects were designed, and their visual features were simulated based on the between- and in-group variations seen in Figure 10. The first simulated object was given a visual feature vector with a mean value of 400 pixels in size, 2200 in intensity, and 0.185 for the first invariant moment. The second simulated object was given a mean value of 1000 pixels in size, 2100 intensity, and 0.21 for the first invariant moment. Each simulated object detection (every time one of the simulated objects was found to be located within the camera's FOV) resulted in an object detection with their mean feature values, in addition to white noise according to the observed average intraclass variation in Figure 10. That is, Gaussian white noise with a standard deviation of 160 pixels, 60 intensity, and 0.012 for the size, intensity, and first invariant moment. Gaussian white noise with a standard deviation of $0.05h_k$ (same tuning as R_k in the filter) was also added to the position measurement of the simulated objects.

Figures 13a, 14a, and 15a show flight 4 with two simulated objects in addition to the real two boats in three different scenarios, tracking with $\gamma = 1.0$ (using only the physical distance). It is observed that using this metric, a Kalman filter's identity (the object the filter is initially tracking) is prone to change up to

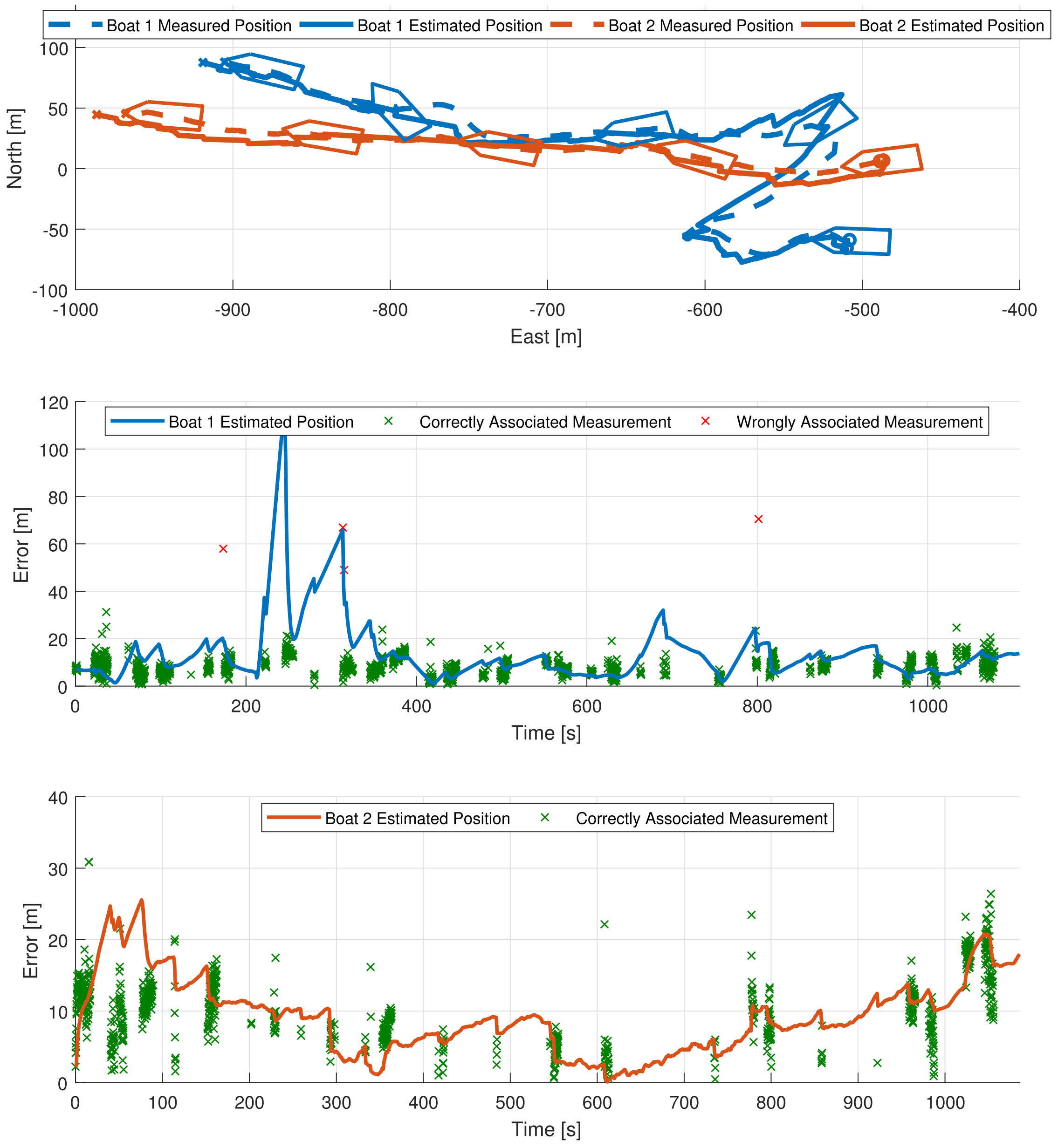
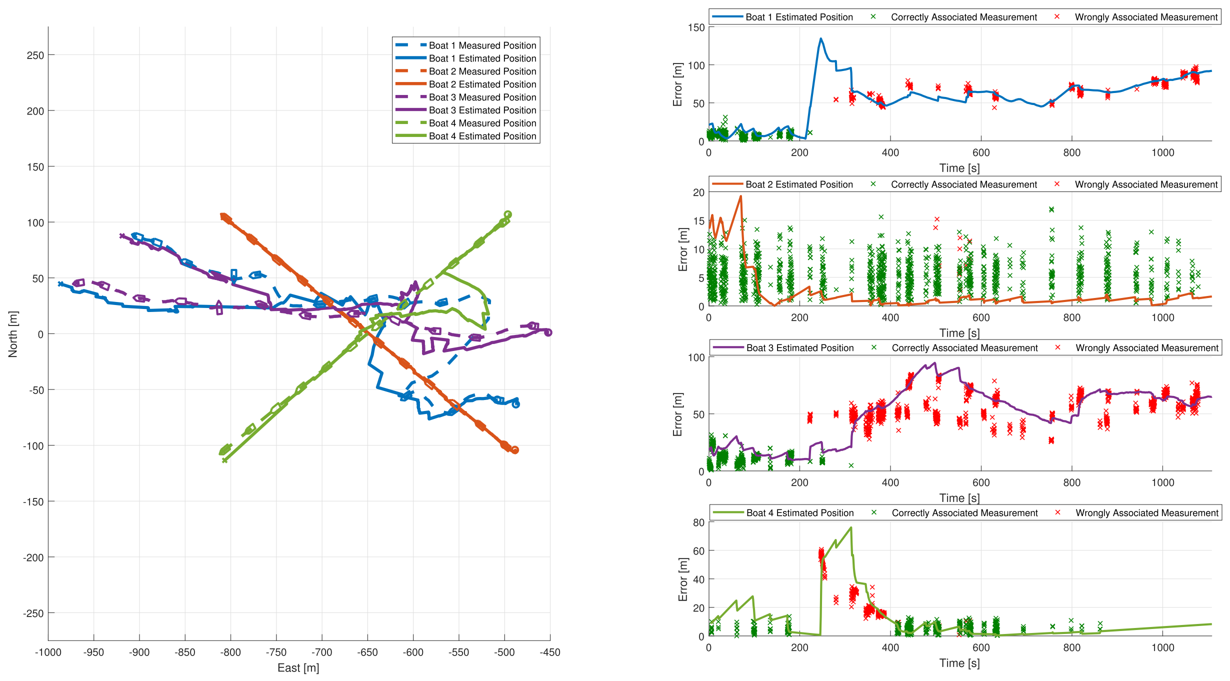


FIGURE 12 Tracking results for test flight 4, using both the physical distance and the object features described in Section 3 for recognition (data association) [Color figure can be viewed at wileyonlinelibrary.com]

several times when tracking multiple objects in the vicinity of each other. In fact, all of the presented scenarios have at least two Kalman filters changing identity completely (hereby noted an identity swap) when only the physical distance metric is used. All scenarios also include some intermittent periods of interrupted tracking, where a filter's track diverges from the object's actual path for short periods of time due to several consecutive incorrect associations. Figures 13b, 14b, and 15b show the same three scenarios, but now tracking with $\gamma = 0.5$ (using both physical distance and object features). This effectively prevents all identity swaps, and the intermittent periods of interrupted tracking are mitigated to only a couple of incorrect associations at a time, or avoided completely. Having as few incorrectly associated measurements as possible is crucial for the certainty and accuracy of the filters' estimates, as the estimates become increasingly more inaccurate and uncertain with the number of incorrectly associated measurements. Hence, including object features in the data association metric increased the performance of the overall tracking system.

The simulated objects are not always moving in accordance with the assumed constant velocity model. This can be seen from the abrupt change in velocity for Boat 2 in Figure 15, and Boat 4 (Figures 14 and 15) which is moving with a constant acceleration. Although the tracking system has a decreased accuracy in its estimates at the points where the velocity is not constant (continuously for Boat 4 and at the 550-s mark in Figure 15 for Boat 2), it is observed that the filter adapts to these changes when new object position measurements are acquired. This indicates that it

(a) Tracking scenario 1 using only the physical distance between measured and estimated position for data association.



(b) Tracking scenario 1 using both physical distance and object features for data association.

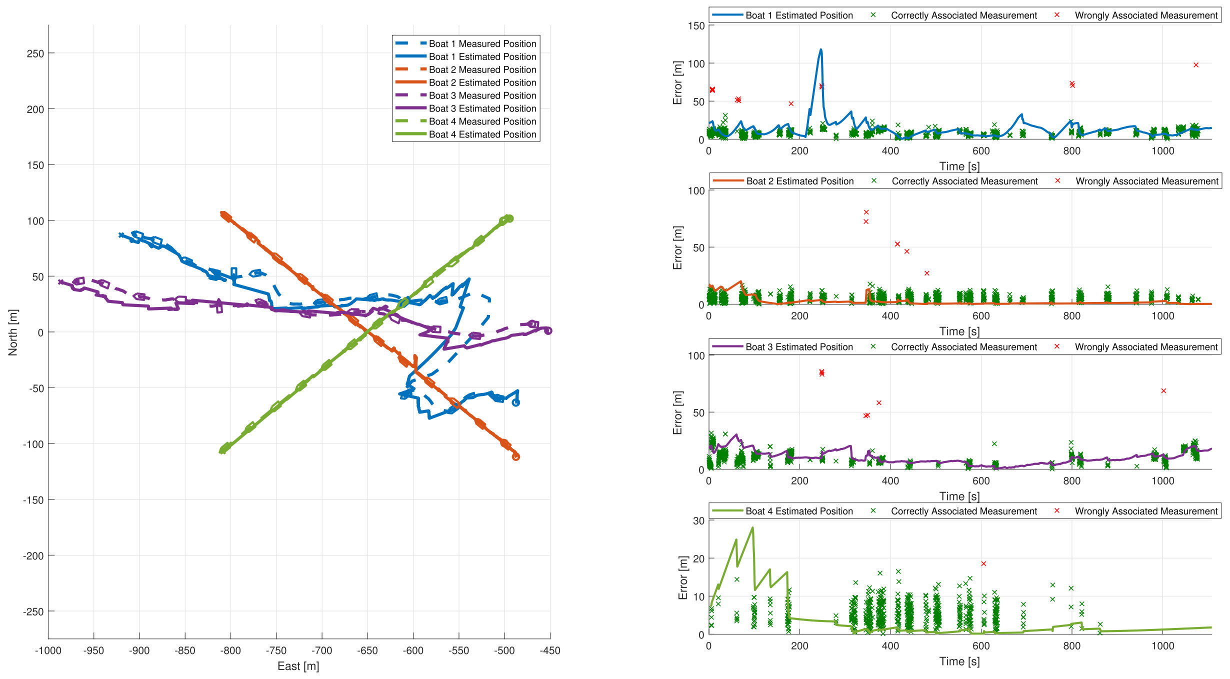


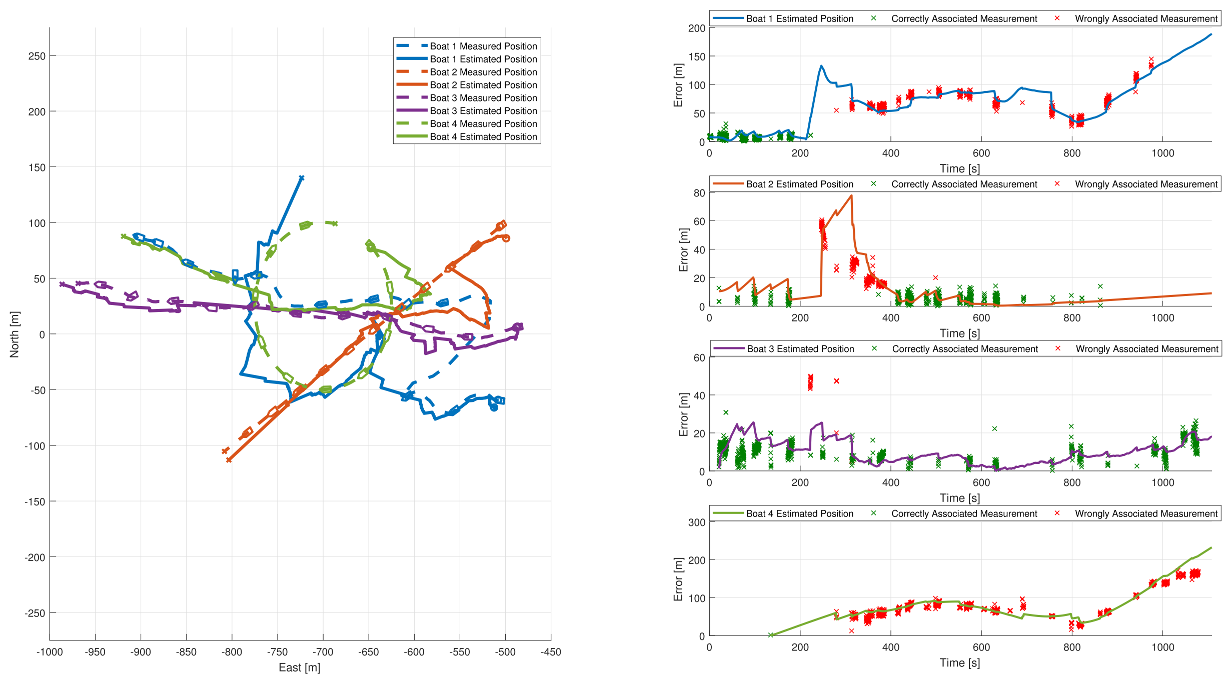
FIGURE 13 Tracking scenario 1 with two simulated objects (Boats 2 and 4) and two real objects (Boats 1 and 3). (a) Tracking scenario 1 using only the physical distance between measured and estimated positions for data association. (b) Tracking scenario 1 using both physical distance and object features for data association [Color figure can be viewed at wileyonlinelibrary.com]

would be beneficial for the path planner module in the tracking system to prioritize revisiting objects that are observed to not move in accordance with the assumed motion model. Further, it could be useful to assume a different motion model if the objects are expected or observed to move in a nonstraight pattern.

6.5 | Filter consistency analysis

Figure 16 shows the autocorrelation for the Kalman filter innovation in the north and east positions for the tracking scenarios in flight 1 (a), flight 2 (b), and flight 3 (c). The innovation is here

(a) Tracking scenario 2 using only the physical distance between measured and estimated position for data association.



(b) Tracking scenario 2 using both physical distance and object features for data association.

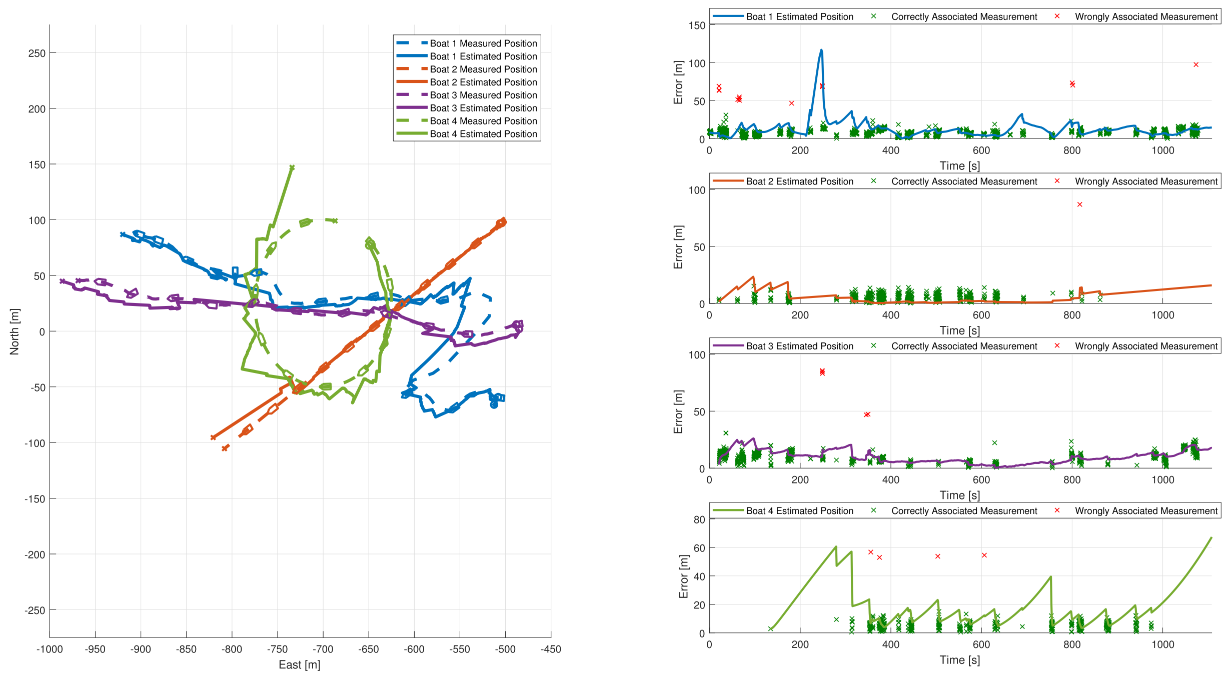
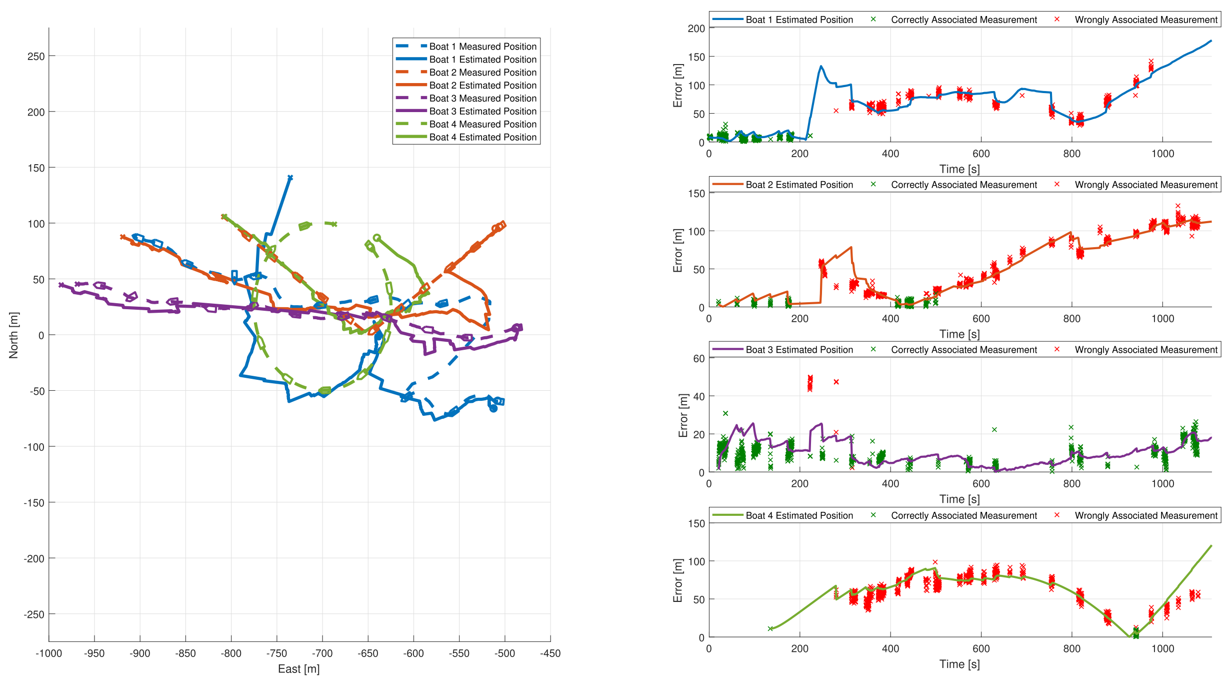


FIGURE 14 Tracking scenario 2 with two simulated objects (Boats 2 and 4) and two real objects (Boats 1 and 3). (a) Tracking scenario 2 using only the physical distance between measured and estimated positions for data association. (b) Tracking scenario 2 using both physical distance and object features for data association [Color figure can be viewed at wileyonlinelibrary.com]

defined as the difference between the current measurement and the predicted measurements by the Kalman filter. The autocorrelations in Figure 16 show that the innovations are (decreasingly) correlated in time, which means that the innovation in the previous measurements is correlated with the current measurement. This result also shows that the innovations are consistently correlated in time throughout all of the tested tracking scenarios.

Theoretically, correlation in the innovation of the filter is a violation of the optimality conditions for the Kalman filter. However, these results are expected, as an error or bias in the estimate of the UAV's position and attitude for one image is likely

(a) Tracking scenario 3 using only the physical distance between measured and estimated position for data association.



(b) Tracking scenario 3 using both physical distance and object features for data association.

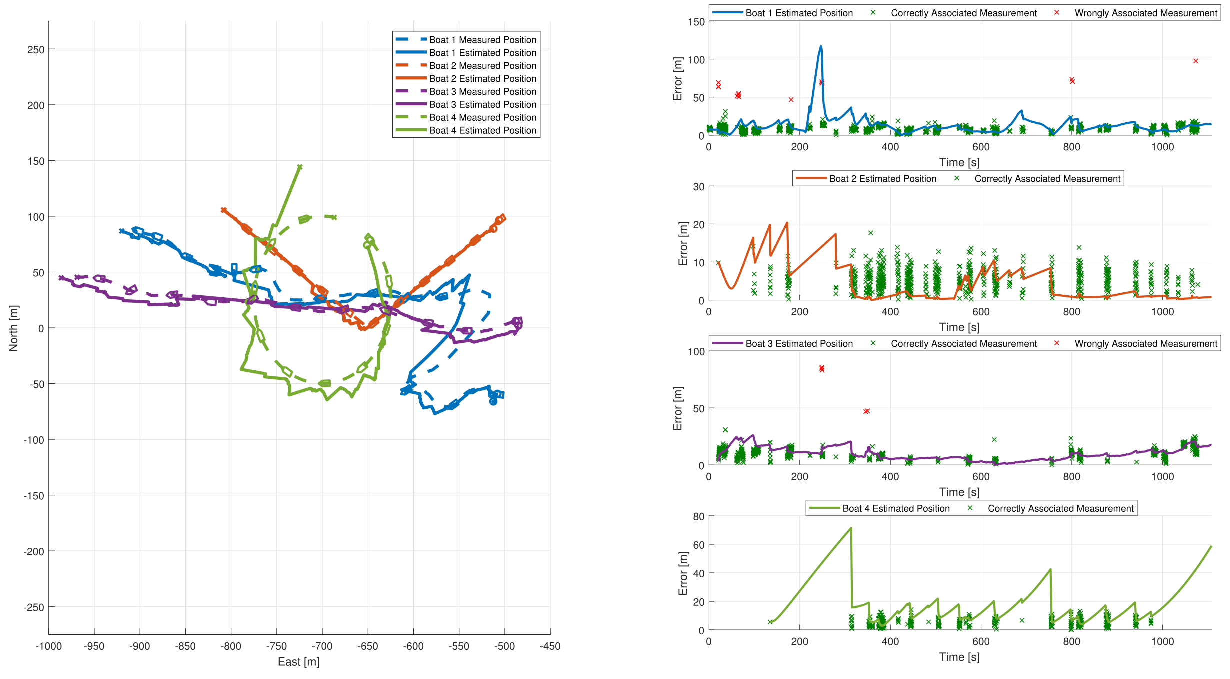


FIGURE 15 Tracking scenario 3 with two simulated objects (Boats 2 and 4) and two real objects (Boats 1 and 3). (a) Tracking scenario 3 using only the physical distance between measured and estimated positions for data association. (b) Tracking scenario 3 using both physical distance and object features for data association [Color figure can be viewed at wileyonlinelibrary.com]

to persist while the next image is captured. Furthermore, perfect white measurement noise is required for the Kalman filter to be optimal theoretically, but not to perform satisfactory. That is, high accuracy in the filter's position and velocity estimates can still be achieved as shown in all of the results in this study. Note that the results in Figure 16 show that it will be beneficial to observe the object from different UAV positions and attitudes (included level flight) so that bias or errors in the UAV's estimated navigation states will be averaged out as the object is observed from different positions and poses.

Another, more important characteristic of the Kalman filters' tracking performance on the application level is the consistency of the position estimates. That is, how confident is the filter in its estimates, and if this confidence reflects the actual situation. A plot of the 95% confidence ellipse of the filters' position estimates at fixed intervals in time can be seen for flight tests 1–3 as green circles in Figure 17. These results show that the filters' confidence in its estimates seem to be justified, as the tracked boats measured position (blue line) is within the 95% confidence interval at all times. This indicates that the tuning parameters chosen for the tracking filter are reasonable, albeit causing the filter to be overly pessimistic in its confidence at times (e.g., as seen in Figure 17b). Considering the application for the tracking system in this paper, it can be argued that it is better to tune the tracking filter this way because it yields less chance of the filter track to diverge from the boat's actual track. On the other hand, it will expand the region that needs to be searched by the UAV to relocate an object. Although this is a trade-off, the UAV's cruising speed is so high that avoiding tracking filter divergence was considered more important than restricting the search region to relocate objects.

7 | CONCLUSION AND FUTURE WORK

In this paper, an automatic object detection, recognition, and tracking algorithm has been developed. The algorithm is intended to be used in an ocean surface object tracking system for UAVs, to enable UAVs to perform multiobject tracking and situational awareness in real time. The detection algorithm uses a combination of edge detection and thresholding, together with dilatation/eroding and finding connected components to perform real-time automatic object detection from a thermal camera's video stream. Using onboard navigation data to get the UAV's and camera's attitude and altitude, the onboard computer is able to georeference each object detection to measure the location of detected objects in a local NED coordinate frame. Furthermore, the tracking algorithm uses a Kalman filter and a constant velocity motion model to perform object tracking based on the position measurements found using the object detection algorithm. To decide whether an object detection is a detection of an object already being tracked, or if the measurement originated from a novel object which has not been observed previously, the Kuhn–Munkres algorithm is applied for data association. This is achieved using a measure of distance that is based not only on the physical distance between an object's estimated position and the measured position, but also how similar the objects appear in the thermal image.

The object detection algorithm developed in this paper was found to consistently detect the objects of interest in a given thermal image. However, these detection results would of course not be expected if the UAVs were to fly over land (or a small island at sea), as the algorithm largely depends on the thermal signatures of objects to be located on a surface with a uniform emission of thermal radiation. Hence, the detection algorithm in the presented tracking system would have to be more sophisticated to be useful in less uniform environments. In general, the algorithm should be able to better filter out detections which are not of interest, either by using a digital map to filter out detections of land, or by recognition to filter out noninteresting objects.

The presented object detection, recognition, and tracking algorithm was tested in four different UAV flight tests conducted over sea from the shoreside just outside Trondheim. Given that the system had at least 100 georeferenced measurements of the boat's position, the position was estimated and tracked with an accuracy of 5–20 m while the boat was in the camera's FOV. The estimated speed and course would also converge to the object's true trajectories (measured by GPS), given that the object was moving in accordance with the model and tuning in the Kalman filter. The results show in most of the cases a drift in the boat's position estimate of only 1–5 m/min outside of the FOV of the camera. Analyzing the innovation in the Kalman filter during tracking, it was found that there was a correlation in the Kalman filter innovation given by measurements related closely in time. Although this is a violation of the theoretical optimality of the Kalman filter, the performance of the filter was still satisfactory. However, correlation in the filter innovations show that it is advantageous to observe the tracked objects from several different positions and attitudes to average out biases and get an accurate position estimate.

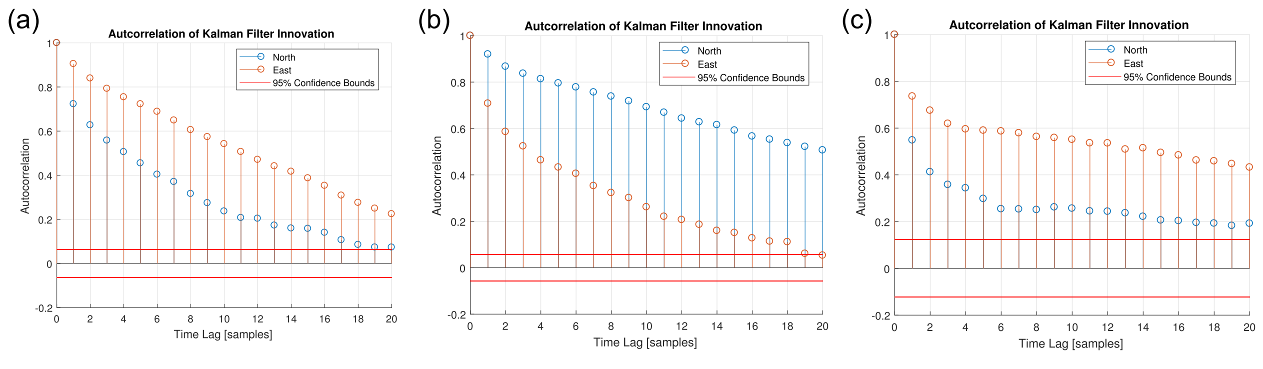


FIGURE 16 Autocorrelation for the Kalman filter innovation for the georeferenced position measurements decomposed in the north and east directions for flights 1 (a), 2 (b), and 3 (c) [Color figure can be viewed at wileyonlinelibrary.com]

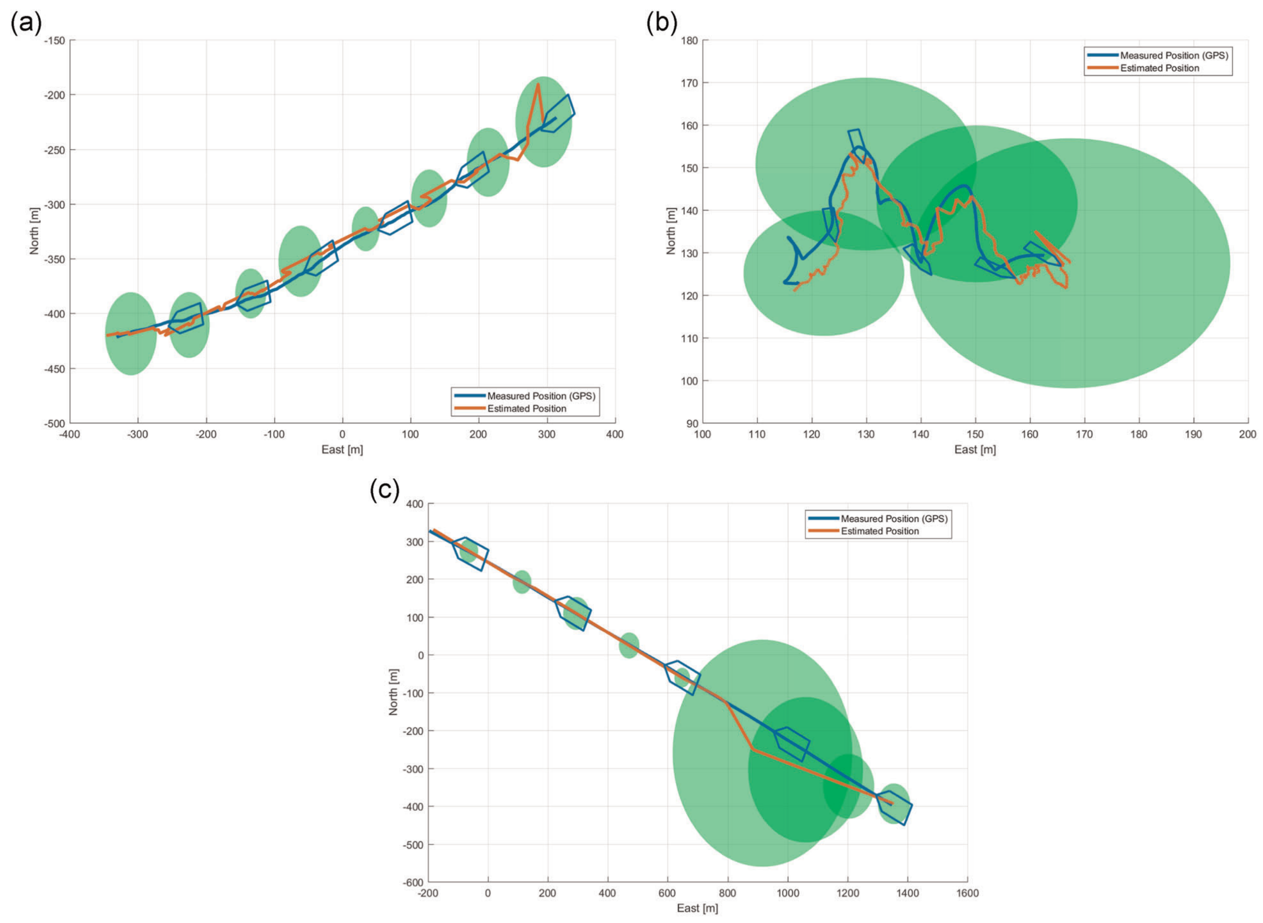


FIGURE 17 Ninety-five percent confidence ellipses for the position estimate (green circle) at equally spaced time instants for flights 1 (a), 2 (b), and 3 (c). GPS, Global Positioning System [Color figure can be viewed at wileyonlinelibrary.com]

The results indicate that the object's speed and course estimates quickly converge after ≈100 georeferenced measurements, or equivalently about 14 s within the FOV of the camera, to the measured (by GPS) speed and course. This seems to be given that the boat is moving at a constant course and a speed consistent with the choice of standard deviation for the Kalman filter's motion model noise. For extremely slow-moving objects the speed estimate still converges, but the system does not necessarily estimate the correct course. This could potentially be solved by using machine vision to calculate a boat's orientation in the water to aid the course estimate. Results also show that if the boat's speed is high, it is crucial to have the object within the FOV of the camera for an extended period of time (ideally around 100 consecutive georeferenced measurements), since both the speed estimate and the course estimate need some time to converge to the boat's actual speed and heading. This is given that the filter was initialized with a speed and course of 0 m/s and 0°, respectively. Hence, in cases with fast-moving objects, the Kalman filter either has to be initialized with a nonzero speed estimate in the vicinity of the object's actual speed (gathered either from assumption or, for instance, from an Automatic Identification System), or the boat has to be observed for a longer period of time without the object leaving the camera's FOV. In general, a controller which would move the gimbal in such a manner that a detected object will remain within the FOV for as long as possible might be useful in many cases. The fourth and final flight test demonstrates the effectiveness and usefulness of including visual appearance in the data association part of the tracking process. The results indicate that the object feature vector presented in this paper can be used for the tracking system to distinguish between objects of the same type (e.g., two boats), which can look similar but still differ in the chosen features (area, average infrared radiation, and Hu's first invariant moment). This is further demonstrated by the mixed experimental/simulated tracking scenarios presented, where considering object features in the association process outperforms using only the physical distance. Taking the object features into account, most incorrect associations are rectified and all identity swaps are prevented. Although an identity swap is not observed to cause the overall system to lose track of objects, it is detrimental in applications where a complete history of an object's movements is important. Significantly reducing the number of incorrect associations increases the accuracy and certainty of the Kalman filters' estimates, and improves the performance of the overall tracking system.

For future work, a high priority is to make the object detection algorithm able to self-tune. That is, instead of setting the threshold T_g

and the minimum object size manually, this should be set automatically or dynamically. The minimum object size could be based on the human operator's choice for what types of objects are of interest. Furthermore, this could be further improved by adjusting the minimum object size based on the UAV's altitude, as a boat of a given size will appear smaller the higher the UAV is flying. The threshold T_g could either be set online based on an initial calibration phase, or potentially chosen to be dynamically set based on analysis of the scene background.

From the results presented it is readily seen that, although successful at tracking an object in general, a priority will be to include online and real-time adjustments of the Gaussian white noise variables found in the motion model for the Kalman filter. This means adjusting the measurement noise based on both the UAV's altitude and attitude (i.e., not just altitude), as well as choosing the motion model noise based on what type of object the UAV is tracking. The features used for recognition could be useful also for such classification. This would make the tracking process more efficient and better at estimating the true position and velocity of the objects being tracked. Another useful addition to the system would be to include the feature vector used for data association in the Kalman filter, effectively allowing the filter to adapt to small changes in all of the object features over time, as well as inherently including the statistics for the variation of each feature in the data association step.

ACKNOWLEDGMENTS

The authors would like to thank Lars Semb and Pål Kvaløy for their technical support and flawless execution of the practical aspects of the field experiments. They would also like to thank Maritime Robotics for allowing us to use Telemetron during the conducted flight tests. This study has been carried out at the NTNU Centre for Autonomous Marine Operations and Systems (NTNU AMOS). The Norwegian Research Council is acknowledged as the main sponsor of NTNU AMOS. This study was supported by the Research Council of Norway through the Centres of Excellence funding scheme, Project number 223254 and NFR/MAROFF Project Number 269480 Hybrid Operations in Maritime Environments.

ORCID

Frederik S. Leira https://orcid.org/0000-0001-9726-558X

REFERENCES

Ali, A., & Terada, K. (2010). A general framework for multi-human tracking using Kalman filter and fast mean shift algorithms. *Journal of Universal Computer Science*, *16*(6), 921-937.

Barber, D. B., Redding, J. D., McLain, T. W., Beard, R. W., & Taylor, C. N. (2006). Vision-based target geo-location using a fixed-wing miniature air vehicle. *Journal of Intelligent and Robotic Systems*, *47*(4), 361-382.

Bourgeois, F., & Lassalle, J.-C. (1971). An extension of the Munkres algorithm for the assignment problem to rectangular matrices. *Communications of the ACM*, *14*(12), 802-804.

Bradski, G. R. (1998). *Computer vision face tracking as a component of a perceptual user interface* (pp. 214–219). Workshop on Applications of Computer Vision, Princeton, NJ (1998–2010).

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *8*(6), 679-698.

Chapelle, O., Haffner, P., & Vapnik, V. N. (1999). Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, *10*(5), 1055-1064.

Dobrokhodov, V. N., Kaminer, I. I., Jones, K. D., & Ghabcheloo, R. (2006). Vision-based tracking and motion estimation for moving targets using small UAVs. In *2006 American Control Conference* (p. 6). Minneapolis, MN. https://doi.org/10.1109/ACC.2006.1656418

Doherty, P., & Rudol, P. (2007). A UAV search and rescue scenario with human body detection and geolocalization. In *Proceedings of the 20th Australian Joint Conference on Artificial Intelligence (AI)*. Berlin, Heidelberg: Springer Berlin Heidelberg.

FLIR-Systems. (2018). *Tau 2*. http://www.flir.com/cores/display/?id=54717

Flusser, J. (2005). Moment invariants in image analysis. *World Academy of Science, Engineering and Technology*, *11*, 376-381.

Gaszczak, A., Breckon, T., & Han, J. (2011). Real-time people and vehicle detection from UAV imagery. J. Röning, David P. Casasent & Ernest L. Hall (eds) In *Proceedings of the SPIE Conference Intelligent Robots and Computer Vision XXVIII: Algorithms and Techniques* (Vol. 7878). International Society for Optics and Photonics: SPIE.

Goodrich, M. A., Morse, B. S., Gerhardt, D., Cooper, J. L., Quigley, M., Adams, J. A., & Humphrey, C. (2008). Supporting wilderness search and rescue using a camera-equipped mini UAV. *Journal of Field Robotics*, *25*(1-2), 89-110.

Granström, K., Baum, M., & Reuter, S. (2017). Extended Object Tracking: Introduction, Overview, and Applications. *Journal of Advances in Information Fusion*, 12(2).

HardKernel. (2016). *ODROID XU4*. http://www.hardkernel.com/main/products/prdt_info.php?g_code=G143452239825

Helgesen, H. H., Leira, F. S., Bryne, T. H., Albrektsen, S. M., & Johansen, T. A. (2019). Real-time georeferencing of thermal images using small fixed-wing UAVs in maritime environments. *ISPRS Journal of Photogrammetry and Remote Sensing*, *154*, 84-97.

Helgesen, H. H., Leira, F. S., Johansen, T. A., & Fossen, T. I. (2017). *Detection and tracking of floating objects using a UAV with thermal camera* (pp. 289-316). Cham: Springer International Publishing.

Helgesen, H. H., Stendahl Leira, F., Johansen, T. A., & Fossen, T. I. (2016). Tracking of marine surface objects from unmanned aerial vehicles with a pan/tilt unit using a thermal camera and optical flow. In *International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE.

Hu, M.-K. (1962). Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, *8*(2), 179-187.

Kalal, Z., Mikolajczyk, K., & Matas, J. (2012). Tracking–learning–detection. *Pattern Analysis and Machine Intelligence*, *34*, 1409-1422.

Konstantinova, P., Udvarev, A., & Semerdjiev, T. (2003). A study of a target tracking algorithm using global nearest neighbor approach. In *Proceedings of the 4th International Conference on Computer Systems and Technologies: E-Learning*, CompSysTech '03 (pp. 290–295). New York, NY: ACM.

Kuo, C. H., Huang, C., & Nevatia, R. (2010). Multi-target tracking by on-line learned discriminative appearance models. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 685–692).

Leira, F., Fossen, T., & Johansen, T. (2015). Automatic detection, classification and tracking of objects in the ocean surface from UAVs using a thermal camera. In *IEEE Aerospace Conference Proceedings*. IEEE Computer Society.

Leira, F. S., Johansen, T. A., & Fossen, T. I. (2017). A UAV ice tracking framework for autonomous sea ice management. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 581–590). Miami, Florida.

Leira, F. S., Trnka, K., Fossen, T. I., & Johansen, T. A. (2015). A light-weight thermal camera payload with georeferencing capabilities for small fixed-wing UAVs. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)* (pp. 485–494). Denver, Colorado: IEEE.

Ma, Y., Soatto, S., Kosecka, J., & Sastry, S. S. (2012). *An invitation to 3-d vision: From images to geometric models* (Vol. 26). Berlin, Germany: Springer Science & Business Media.

Mathisen, S. G., Leira, F. S., Helgesen, H. H., Gryte, K., & Johansen, T. A. (2020). Autonomous ballistic airdrop of objects from a small fixed-wing unmanned aerial vehicle. *Autonomous Robots*, *44*, 859-875.

MicroUAV. (2011). *Retractable BTC-88*. http://www.microuav.com/btc88

Niedfeldt, P. C., & Beard, R. W. (2013). Recursive RANSAC: Multiple signal estimation with outliers. *IFAC Proceedings Volumes*, *46*(23), 430-435.

Niedfeldt, P. C., & Beard, R. W. (2014). Multiple target tracking using recursive RANSAC. In *2014 American Control Conference* (pp. 3393–3398). IEEE.

Nonami, K. (2007). Prospect and recent research & development for civil use autonomous unmanned aircraft as UAV and MAV. *Journal of System Design and Dynamics*, *1*(2), 120-128.

Pestana, J., Sanchez-Lopez, J. L., Saripalli, S., & Campoy, P. (2014). Computer vision based general object following for GPS-denied multirotor unmanned vehicles. In *2014 American Control Conference* (pp. 1886–1891). IEEE.

Pixhawk. (2018). *Pixhawk 2*. https://pixhawk.org/modules/pixhawk2

Portmann, J., Lynen, S., Chli, M., & Siegwart, R. (2014). People detection and tracking from aerial thermal views. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Prevost, C. G., Desbiens, A., & Gagnon, E. (2007). Extended Kalman filter for state estimation and trajectory prediction of a moving object detected by an unmanned aerial vehicle. In *2007 American Control Conference* (pp. 1805–1810). IEEE.

Qadir, A., Neubert, J., & Semke, W. (2011). On-board visual tracking with unmanned aircraft system (UAS). In *Proceedings of the AIAA Infotech@ Aerospace*. St. Louis, MO.

Rodin, C. D., de Lima, L. N., Andrade, F. A. A., Haddad, D. B., Johansen, T. A., & Storvold, R. (2018). Object classification in thermal images using convolutional neural networks for search and rescue missions with unmanned aerial systems. In *International Joint Conference on Neural Networks (IJCNN)*, Rio de Janeiro.

Rodríguez-Canosa, G. R., Thomas, S., del Cerro, J., Barrientos, A., & MacDonald, B. (2012). A real-time method to detect and track moving objects (DATMO) from unmanned aerial vehicles (UAVs) using a single camera. *Remote Sensing*, *4*(4), 1090.

Sengupta, R., Connors, J., Kehoe, B., Kim, Z., Kuhn, T., & Wood, J. (2010). *Final report—Autonomous search and rescue with scaneagle*. Center for Collaborative Control of Unmanned Vehicles University of California, Berkeley PI. http://www.semanticscholar.org/paper/Prepared-for-Evergreen-Unmanned-Systems-and-Shell-.-Kehoe-Kim/607f75730024fedf72ab9ce0ac99dfed1de5a99a?p2df

Shah, M., Hakeem, A., & Basharat, A. (2006). Detection and tracking of objects from multiple airborne cameras. *SPIE Newsroom*, *1*, 1-3.

Skjong, E., Nundal, S. A., Leira, F. S., & Johansen, T. A. (2015). Autonomous search and tracking of objects using model predictive control of unmanned aerial vehicle and gimbal: Hardware-in-the-loop simulation of payload and avionics. In *International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE.

Soille, P. (2003). *Morphological image analysis: Principles and applications* (2nd ed.). Secaucus, NJ: Springer-Verlag New York Inc.

Štěpán, P., Krajník, T., Petrlík, M., & Saska, M. (2019). Vision techniques for on-board detection, following, and mapping of moving targets. *Journal of Field Robotics*, *36*(1), 252-269.

Suzuki, S., & Abe, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, *30*(1), 32-46.

TEAX-Technology. (2018). *ThermalCapture Grabber USB 640*. http://thermalcapture.com/thermalcapture-grabber-usb-640/

Teuliere, C., Eck, L., & Marchand, E. (2011). Chasing a moving target from a flying UAV. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems* (pp. 4929–4934). IEEE.

Ullah, M., Mohammed, A. K., Cheikh, F. A., & Wang, Z. (2017). A hierarchical feature model for multi-target tracking. In *2017 IEEE International Conference on Image Processing (ICIP)* (pp. 2612–2616).

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (Vol. 1, pp. 511–518).

Wu, Z., Thangali, A., Sclaroff, S., & Betke, M. (2012). Coupling detection and data association for multiple object tracking. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1948–1955). IEEE.

Yinghui, G., & Jianjun, Y. (2009). Multi-target tracking using mixed spatio-temporal features learning model. In *2009 IEEE International Conference on Automation and Logistics* (pp. 799–803).

How to cite this article: Leira FS, Helgesen HH, Johansen TA, Fossen TI. Object detection, recognition, and tracking from UAVs using a thermal camera. *J Field Robotics*. 2021;38: 242–267. https://doi.org/10.1002/rob.21985