Automatic Detection, Classification and Tracking of Objects in the Ocean Surface from UAVs Using a Thermal Camera

Frederik S. Leira
frederik.leira@itk.ntnu.no

Tor Arne Johansen
tor.arne.johansen@itk.ntnu.no

Thor I. Fossen
thor.fossen@itk.ntnu.no

Centre for Autonomous Marine Operations and Systems
Department of Engineering Cybernetics
Norwegian University of Science and Technology

***Abstract*—The use of unmanned aerial vehicles (UAVs) that can operate autonomously in dynamic and dangerous operational environments are becoming increasingly common. In such operations, object detection, classification and tracking can often be one of the main goals. In recent years there has been an increased focus on embedded hardware that is both small and powerful, making UAV on-board data processing more viable. Being able to process the video feed on-board the UAV calls for fast and robust real-time algorithms for object identification and tracking. This paper discusses the development and implementation of a machine vision system for a low-cost fixed-wing UAV with a total flying weight of under 4kg. The machine vision system incorporates the use of a thermal imaging camera and on-board processing power to perform real-time object detection, classification and tracking of objects at the ocean surface. The system is tested on thermal video data from a test flight, and is found to be able to detect $99,6\%$ of objects of interest located in the ocean surface. Of the detected objects, only 5% were false positives. Furthermore, it classifies $93,3\%$ of the object types it is trained to classify correctly. The classifier is highly agile, allowing the user to quickly define which object characteristics that should be considered during classification, and what types of objects to classify. Finally, the system is found to successfully track 85% of the object types it is actively searching for in a real-time simulation test.**

TABLE OF CONTENTS

1. INTRODUCTION

The recent increase of commercial availability of small unmanned aerial vehicles (UAVs) has lead to the use of UAVs in many different applications involving inspections of structures, surveillance and tracking of objects. Research has been carried out to extend the UAV's area of applications, especially in areas where UAVs can increase safety and efficiency. This includes search and rescue (SAR) applications, where UAVs equipped with cameras are used to map large areas and locate missing objects.

[1] describes a SAR system using a fixed-wing UAV equipped with a camera sensor. By searching a predefined area, likelihood functions are constructed to determine the likelihood of localizing the object at a given location. The likelihood functions are merged into a larger mapping, a probability density function (PDF), which is maintained to express the most likely location of the object. The UAV's and the camera sensor's paths are generated from the PDF to maximize the likelihood of detecting the object of interest. The main downside of this approach is that it does not extend to an arbitrary number of objects, and that localizing a moving object would greatly complicate the process.

[2] describes another SAR system for rotary wing UAVs, also equipped with camera sensors. As described in [3], their system utilizes a trained boosted cascade classifier based on Haar-like features to automatically detect objects. Although the results are promising, the object detection module is dependent on having a stable and non-moving video feed of the object to verify the detection. This is often impossible to achieve, e.g if a fixed-wing UAV is utilized. Furthermore, the system does not take into account moving objects.

[4][5] both use state-of-the-art methods to detect, classify and track humans in aerial infrared images. This is achieved by combining methods such as background subtraction, edge detection and mean shift segmentation for detection with machine learning (support vector machine (SVM) and cascaded classifiers) for classification. For tracking, popular methods includes particle filters and Kalman filters. [6] also use a thermal camera for surveillance in a maritime environment, classifying objects using a SVM trained classifier based on a set of predefined features. However, there is little literature combining the two with an UAV capable of both detecting, classifying and tracking humans (or missing objects in general) in a marine environment.

The focus of this paper is the combination of novel methods for detection, classification and tracking specifically in a maritime setting. We describe the development and implementation of a novel payload system for search and tracking of objects for a fixed-wing UAV. Using real-time on-board analysis of thermal images, it seeks to automatically detect, classify and track objects of interest. The benefit of processing the video feed on-board is twofold. First, it greatly

reduces the need for a fast and stable communication link to the ground station, effectively extending the range of an operation at a low cost. Second, it may greatly reduce the delay in the decision-making process, i.e where the UAV should fly to maximize the visual information. In particular we develop a system, which is capable of tracking multiple objects simultaneously, where the user has direct control with the types of object to track. It would, for instance, be possible to adapt the proposed system to detect and track the motion of ice floes instead, effectively implementing one of the outlined ice monitoring systems described in [7].

The paper is organized as follows. First, the overall UAV payload and system are described. This includes a short description of each independent module and their respective tasks. Following this is an in-depth description of the algorithms and methods used to develop and implement a SAR system on-board an UAV. This includes the object detection, classification and tracking modules. Furthermore, the implementation of software for use in a real-time environment is described. The system's performance is evaluated on gathered video data from an UAV flight test. Finally the paper is summarized in a brief conclusion.

2. UAV Payload and System

The object detection, classification and tracking payload system we consider consists of several different components and subsystems. Figure 1 illustrates the system components and the information flow between them. The payload includes an autopilot (ArduPilot [8]) for flight control, a single board computer (PandaBoard [9]) for on-board image processing and an analog thermal camera (FLIR Tau2 336 [10]) as a visual sensor. The thermal camera has a sampling frequency of 9 frames per second (upsampled to 30 frames per second for analog output) and is sensitive to the long-wave infrared spectral band ($7.5 - 13.5\mu m$) with a sensitivity of $< 50mK$. It has a resolution of 336×256 pixels, which is interpolated up to 720×480 pixels. Furthermore, there is an analog to digital converter (Axis M7001 [11]) that converts the analog thermal video feed to a digital video stream which can also be broadcast to the ground station by the on-board network. Finally there is a communication link (Ubiquiti Rocket M5 [12]) to the ground station. The radio link installed in the payload enables remote control of the object detection, classification and tracking system, as well as making it possible to visually see the thermal images from the ground station. The interconnection of these components is illustrated in Figure 1.

As seen in Figure 2, the object detection, classification and tracking module can be divided into several smaller modules. An object detection module is responsible for segmenting the parts of an image that is likely to contain objects of interest. This is done by applying machine vision techniques which filter out everything but objects of interest from the original image. After having segmented out objects of interest from the original image, the object detection module passes on a list of the location of the detected objects to an object classification module. The location of the objects are given in both image frame and world frame coordinates. The image frame coordinates are found by calculating the centroid of the object of interest, while the world frame coordinates are found by georeferencing the object's image frame position using on-board GPS and AHRS (Attitude-Heading Reference System) data.

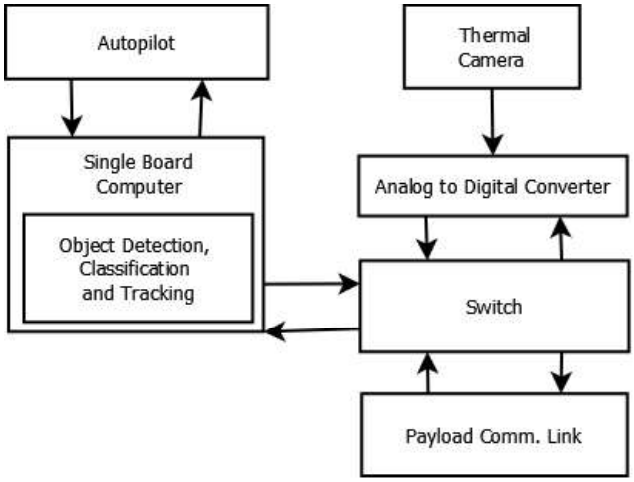

Figure 1. Overall object detection, classification and tracking payload system description.

The object classification module seeks to categorize the objects of interest into specific categories. The categories used by the system presented in this paper are the following: non-interesting or unidentified object (disregard observation), human, small boat and big boat. The classification module is given a set of reference object features, which decides what characteristics of the detected object that should be evaluated in the classification process. Hence, this module

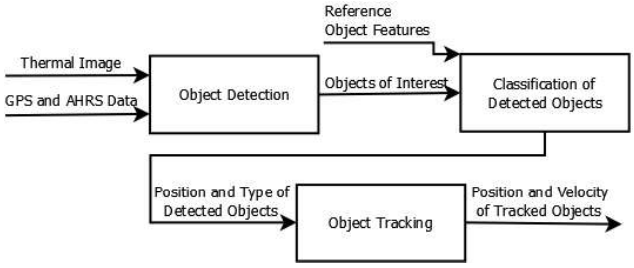

Figure 2. Data flow in the object detection, classification and tracking module.

can be changed to suit the needs of the user (e.g remove or add categories) simply by changing the set of external object features fed to the classification module. In this regard, it should be noted that the classification module works independently from the object detection module. The results from the classification process is sent to the object tracking module, in order to make the tracking module able to know what it is tracking.

The object tracking module is responsible for both estimating and predicting the position and velocity of the detected objects. The position is here given as a 2 dimensional vector, and the velocity as the speed along each of the 2 dimensions. This means that the tracking module is assuming that the object is moving in a 2 dimensional reference frame (e.g the image plane or the ocean surface). Furthermore, it associates new observations done by the object detection module with already detected objects, making the system able to track objects from one frame to the next, as well as keeping track of objects over a prolonged period of time. The output from the object tracking module is the position history (where the object has been), as well as the predicted future position

and velocity of the tracked objects. This information can in turn be sent to the path planner/controller of the UAV. This will enable the system to close the loop, i.e interconnect the detection and tracking module with a path planning module, effectively making the UAV able to follow targets. However, this is not the topic of this paper.

The following section will describe the three modules in the object detection and tracking system in some more detail.

3. Object Detection

To automatically detect objects of interest, the thermal image, $\mathbf{I}$, is first smoothed. The motivation for this is to reduce the thermal noise present in the image. This was found to make the edge detector more robust when performed on the sharp thermal images. The smoothing is done by convolving (denoted by $*$) the image with a Gaussian kernel $\mathbf{g}$. $\mathbf{g}$ is a $n \times n$ kernel approximating a Gaussian distribution with a standard deviation of σ_g, i.e

$$\begin{aligned} \boldsymbol{I_s}[x,y] =& (\boldsymbol{I} * \boldsymbol{g})[x,y] \\ =& \sum_{k=-\frac{n-1}{2}}^{k=\frac{n-1}{2}} \sum_{m=-\frac{n-1}{2}}^{m=\frac{n-1}{2}} \boldsymbol{I}[x-m, y-k]\boldsymbol{g}[m,k] \end{aligned} \tag{1}$$

$\boldsymbol{I}$ and $\boldsymbol{I_s}$ are $w \times h$ matrices, where w and h is the width

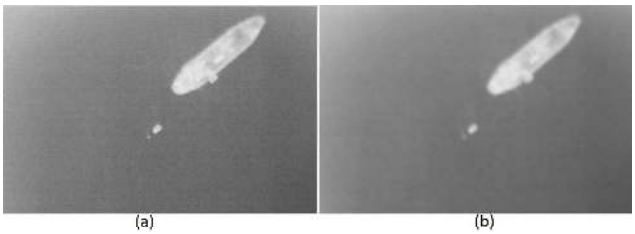

Figure 3. Before (a) and after (b) smoothing of the original image. The image is showing a large boat (length of 55m), a rigid-inflatable boat and a small buoy.

and height of the original thermal image. $[x, y]$ are integers representing a pixel coordinate in images $\boldsymbol{I}$ and $\boldsymbol{I_s}$. $[m, k]$ are integers representing a coordinate in the Gaussian kernel approximation $\boldsymbol{g}$. The result of smoothing an image showing a big boat (length of 56m), a rigid-inflatable boat (RIB) and a small buoy can be seen in Figure 3. Notice that the upper left corner has slightly brighter pixels than the rest of the image, due to inaccurate camera calibration. Now, to detect the edges in the resulting smoothed image $\boldsymbol{I}_s$, the gradient image, $\boldsymbol{G}$, of $\mathbf{I_s}$ is calculated. The gradient image of $\boldsymbol{I}_s$ is found by the following calculation

$$\begin{aligned} &\boldsymbol{G}_x[x,y] = (\boldsymbol{I}_s * \boldsymbol{P})[x,y] = \\ &\sum_{k=-1}^{k=1} \sum_{m=-1}^{m=1} \boldsymbol{I}_s[x-m, y-k]\boldsymbol{P}[m,k], \\ &\boldsymbol{G}_y[x,y] = (\boldsymbol{I}_s * \boldsymbol{P}^T)[x,y] = \\ &\sum_{k=-1}^{k=1} \sum_{m=-1}^{m=1} \boldsymbol{I}_s[x-m, y-k]\boldsymbol{P}^T[m,k], \\ &\boldsymbol{G}[x,y] = \sqrt{\boldsymbol{G}_x^2[x,y] + \boldsymbol{G}_y^2[x,y]} \end{aligned} \tag{2}$$

$\mathbf{P}$, also referred to as the Prewitt operator [13], is defined as the 3×3 matrix

$$\mathbf{P} := \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{3}$$

The resulting gradient image $\mathbf{G}$ can be seen in Figure 4a. It is seen that the big boat, the RIB and the small buoy is clearly visible after these image processing operations. However, it is apparent that the waves and ripples in the ocean in addition to some of the noise in the image are still visible, albeit smaller in magnitude than the objects of interest in the image. Because of this, removing them can be done by using

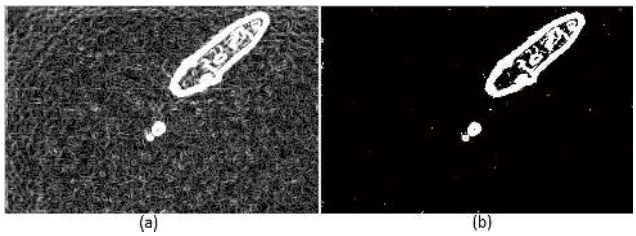

Figure 4. Before (a) and after (b) gradient magnitude thresholding.

a threshold value for the magnitude of the gradients which should be visible. That is, all pixels in the gradient image that have a magnitude less than a certain threshold T_g can be removed. This is achieved by the following operation

$$\boldsymbol{G}(x,y) = \begin{cases} maxValue & \text{if } \boldsymbol{G}(x,y) \geq T_g \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Where $maxValue$ is the maximum brightness value a pixel in image $\mathbf{G}$ can have. From Figure 4b it is readily seen that, post processing, it is mostly objects with a distinct heat signature that is left in the image.

Looking at Figure 5a, 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 in Figure 5a), and therefore has to be filtered out. To filter out the unwanted blobs from the image, a connected component algorithm [14] 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 a blob originating from an object of interest are then removed from the image. The result of this process is seen in Figure 5. The resulting image (Figure 5b) is hereby referred to as the binary image, $\mathbf{B}$, of the original image $\mathbf{I}$.

Figure 5. Before (a) and after (b) removing blobs which are either too small or too large to be an object of interest.

After applying the image analysis methods just described and finding the bounding boxes for each detected object, the

detected objects can be seen in Figure 6a. However, it is seen that big objects which has some texture in them can trigger detections within the interior of the actual object. This is because the texture of the object shows up in the edge detector. In order to make every detection of an object only show up as one unique detection, bounding boxes completely contained in a larger bounding box are removed. The result of this process is seen in Figure 6b.

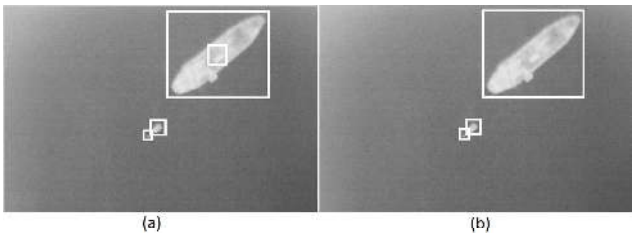

Figure 6. Before (a) and after (b) removing detections completely contained in the interior of other detections.

Looking at Figure 6b, it is apparent that the three detected objects are of further interest. The detected objects 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 measurements.

4. Object Classification

The detection step has provided the classification module with the objects of interest. 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 classification based on characteristics such as size, average temperature 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 [13] shown in Figure 6b is applied. The result of applying this algorithm can be seen in Figure 7.

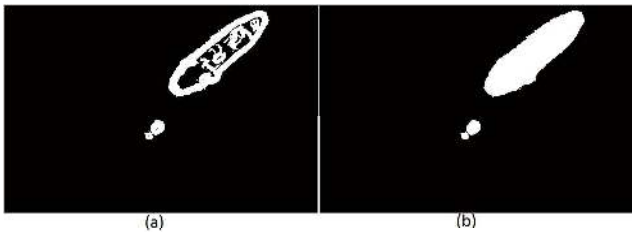

Figure 7. Before (a) and after (b) filling the interior holes in the detected objects.

Using the location of the bright pixels in the binary image seen in Figure 7b, the pixels that make out the object in the original image (Figure 3a) can be analysed. The image with filled contours is hereby denoted $\mathbf{B_{filled}}$.

In this paper, the object characteristics used to classify objects are the observed object area, perceived average object temperature and one of the scale, rotation and translation invariant moments proposed by Hu[15]. Using scale, rotation and translation invariant features when describing objects observed from the air is very important, as the altitude, angle and orientation that the object is viewed from is constantly changing. Note that both the observed object area and the perceived average object temperature will be, to some extent, invariant to the scale, rotation and translation of the object. Furthermore, keep in mind that the classification method presented can be used for a large variety of other object characteristics, as well as easily be modified to include other object features.

The invariant moments presented in [15] is based on the following moment function

$$m_{pq} = \sum_{x,y \in \boldsymbol{O}} x^p y^q \boldsymbol{B}(x,y), \quad p,q = 0,1,... \tag{5}$$

Where m_{pq} is referred to the $(p+q)$th order moment of the image region $\boldsymbol{O}$. $\boldsymbol{O}$ is defined as the set of pixels inside the object's bounding box. $\boldsymbol{B}$ is here a binary image. Note that since $\boldsymbol{B}$ 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 to the metric area of the object through the use of on-board altimeter and AHRS measurements. The metric area of an object can give good estimates of the object's type and inertia. Assuming that the UAV is flying approximately straight forward (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 $a(h)$. $a(h)$ is defined as square meters per pixel when the camera is at altitude h. This factor is given by the thermal camera lens and characteristics. Hence, the metric area of an object can be approximated by

$$A \approx a(h) m_{00} \tag{6}$$

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

$$\begin{gathered} \mu_{pq} = \sum_{x,y \in \boldsymbol{O}} (x-\bar{x})^p (y-\bar{y})^q \boldsymbol{B}(x,y), \quad p,q = 0,1,... \\ \bar{x} = \frac{m_{10}}{m_{00}}, \quad \bar{y} = \frac{m_{01}}{m_{00}} \end{gathered} \tag{7}$$

Where $(\bar{x}, \bar{y})$ is the centroid of the image region. $\boldsymbol{O}$ is still defined as in (5). 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

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}, \quad \gamma = (p+q+2)/2, \quad p+q = 2,3,... \tag{8}$$

Using the normalized central moments, Hu introduced seven moments invariant to rotation, translation and scale. However, research has shown that for images with low pixel resolution (less than 100×100 pixels), these moments may vary when the image is scaled and/or rotated. Furthermore, [16] show that the higher order moments (ϕ_{2-7}) vary much more than the lower order moment (ϕ_1) for images with small resolution. Since the resolution of most thermal imaging cameras are still quite poor, 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.

$$\phi_1 = \eta_{20} + \eta_{02} \tag{9}$$

To calculate the perceived average object temperature, the settings of the thermal imaging camera is utilized. That is, the camera can be set to capture temperatures in a range from a minimum and a maximum temperature (T_{min} and T_{max}). Furthermore, the output from the camera is a standard NTSC analog video signal, where each pixel can take on a value between 0 and $2^8 = 256$. This means that the perceived temperature of an area covering only 1 pixel will be

$$T = \frac{I(x,y)}{256}(T_{max} - T_{min}) + T_{min} \tag{10}$$

Expanding this to calculate the average perceived temperature across a detected object, we get

$$T_{avg} = \frac{\frac{\sum_{x,y \in \boldsymbol{O_p}} I(x,y)}{m_{00}}}{256}(T_{max} - T_{min}) + T_{min} \tag{11}$$

Where

$$\boldsymbol{O_p} = \{x, y \in \boldsymbol{O} | \boldsymbol{B_{filled}}(x,y) = 1\}$$

$\boldsymbol{O}$ 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 $\mathbf{B_{filled}}$. Note that the perceived temperature is found from pixel intensities in the original image $\mathbf{I}$.

Combining the perceived average object temperature with the invariant moments, we can represent any group of pixels in an image using the feature vector $\mathbf{X}$

$$\mathbf{X} = \begin{bmatrix} A \\ T_{avg} \\ \phi_1 \end{bmatrix} \tag{12}$$

However, to ensure that no single feature will dominate the orientation of the feature vector, feature rescaling is included. This results in the following scaled feature vector

$$\mathbf{X_{rs}} = \begin{bmatrix} \frac{A - A_{min}}{A_{max} - A_{min}} \\ \frac{T_{avg} - T_{min}}{T_{max} - T_{min}} \\ \frac{\phi_1 - \phi_{1_{min}}}{\phi_{1_{max}} - \phi_{1_{min}}} \end{bmatrix} \tag{13}$$

This effectively scales each feature into the range $[0, 1]$. The minimum and maximum values of each feature can be the minimum and maximum value observed during the training process, which is explained below. The scaled feature vector can also be tuned to give particular weight on certain features, if desired.

In order to use the feature vector $\boldsymbol{X_{rs}}$ to classify objects, we need some classified references in the feature space. The process of finding these reference points is referred to as classification training, and there exists a wide variety of methods for this in the literature. However, as a proof of concept, we will use a simple and straightforward method in this paper. That is, using 5 example images of each different type of object (human, small boat and big boat), the minimum and maximum value for each feature is estimated. Furthermore, an average feature vector for each class using the same 5 example images.

The average feature vector of each class will be used as a reference point, classifying detected objects based on the distance of an object's feature vector to the class specific average feature vectors. That is, using the class specific reference points in the feature space, classifying an object simply becomes a matter of calculating the object's feature vector $\mathbf{X_{obj}}$ and finding the distance from this vector to all the reference points in the feature space. The class reference point which has the shortest distance to $\mathbf{X_{obj}}$ is most likely the class describing the object. However, if none of the distances are smaller than a certain threshold, the system will disregard the observation or call for user intervention. Choosing the 2-norm to measure distance, the condition for disregarding an observation is given by

$$\|\boldsymbol{X_{obj}} - \boldsymbol{X_{rsi}}\|_2 \geq r \quad \forall i \tag{14}$$

Where $\boldsymbol{X_{rsi}}$ is the reference point in the feature space for class i. r is a tuning parameter describing the threshold for disregarding new observations. This is referred to as a nearest neighbour classifier.

In order for the classification module to handle cases where only parts of an object of interest are visible within the field of view of the camera, objects which are touching the border of the image frame are not classified until the whole object is visible. Furthermore, if an already detected but not yet fully classified object is touching the border of the image frame, the classification process is put on hold until the whole object is visible again. In other words, an object's appearance while touching the border of the image frame is irrelevant to the class type that the object is finally assigned. This is required in order to avoid frames where the object's appearance is heavily occluded to be decisive in the classification process.

5. 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. That is, for each uniquely detected object, a Kalman filter instance is created. Further detections of the tracked object is then used as measurements in the Kalman filter in order to estimate the object's position and velocity. This means that the tracking module also has to be able to link new detections together 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, i.e the history of its positions and velocities. If an object detection is not likely to originate from any of the objects currently being tracked, the tracking module creates a new Kalman filter instance for the newly detected object.

Discrete-Time Kalman Filter

The Kalman filter implemented in the detection and tracking module is based on [17], and utilizes the following linear

equations of motion

$$\begin{aligned} x_{k+1}^{obj} &= x_k^{obj} + \Delta t V_{x,k}^{obj} \\ y_{k+1}^{obj} &= y_k^{obj} + \Delta t V_{y,k}^{obj} \\ V_{x,k+1}^{obj} &= V_{x,k}^{obj} + w_k \\ V_{y,k+1}^{obj} &= V_{y,k}^{obj} + z_k \end{aligned} \tag{15}$$

Where x_k^{obj}, y_k^{obj}, $V_{x,k}^{obj}$ and $V_{y,k}^{obj}$ is the position and linear velocity of an object in the image frame coordinates at time step k and Δt is the time passed from time step k to $k+1$. w_k and z_k is Gaussian white noise representing change in velocity of an object. This yields the following observer model in state space form

$$\begin{aligned} \mathbf{x_{k+1}^{obj}} &= \boldsymbol{A}\mathbf{x_k^{obj}} + \boldsymbol{B}\mathbf{w_k} \\ \mathbf{y_k^{obj}} &= \boldsymbol{C}\mathbf{x_k^{obj}} + \mathbf{v_k} \end{aligned} \tag{16}$$

The matrices A, B and C are equal to

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \boldsymbol{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{17}$$

and we have that

$$\mathbf{w_k} = \begin{bmatrix} w_k \\ z_k \end{bmatrix} \quad \mathbf{v_k} = \begin{bmatrix} q_k \\ r_k \end{bmatrix} \tag{18}$$

w_k and z_k are here as previously defined, while q_k and r_k are Gaussian white noise terms which represent noise and errors in the measurement of an object's position.

For the two dimensional motion described in (15), the state vector $\mathbf{x_k}$ and the measurement $\mathbf{y_k}$ are equal to

$$\mathbf{x_k^{obj}} = \begin{bmatrix} x_k^{obj} \\ y_k^{obj} \\ v_x^{obj} \\ v_y^{obj} \end{bmatrix} \quad \mathbf{y_k^{obj}} = \begin{bmatrix} y_{x,k}^{obj} \\ y_{y,k}^{obj} \end{bmatrix} \tag{19}$$

Where $y_{x,k}^{obj}, y_{y,k}^{obj}$ is the detected object's position at time step k.

It should be noted that for the application in this paper, the Kalman filter is set to track objects only in the image plane. That is, the objects position and velocity is estimated and predicted in image pixel coordinates. However, assuming without loss of generality that an object is moving at a flat surface (the ocean surface), a Kalman filter based on (15)-(19) can be used to estimate an object's position and velocity also in North-East-Down (NED) frame coordinates. This is because both scenarios only have motion in 2 dimensions, in addition to that the measurements are given only as object position in its relative coordinate frame. Tracking an object in the NED frame makes it easy to keep track of an objects GPS coordinates, as converting from NED coordinates to world coordinates is arbitrary.

Data Association

When tracking objects, one of the most crucial parts is to be able to match new object detections to objects which the system is already tracking. This problem is often referred to as data association. In the present system, a global nearest neighbour (GNN) approach similar to the one found in [18] is utilized to perform data association. This involves using the following distance metric for the distance between measurement i and tracking gait j

$$D_{i,j} = \tilde{\boldsymbol{y}}_{i,j}^T \boldsymbol{S}_j(k)^{-1} \tilde{\boldsymbol{y}}_{i,j}. \tag{20}$$

where

$$\tilde{\boldsymbol{y}}_{i,j} = \left[\boldsymbol{y^{obj}}_i - \hat{\boldsymbol{y}}^{\boldsymbol{obj}}{}_j \right]$$

Here, $\boldsymbol{y^{obj}}_i$ is measurement i (given as a position in a 2 dimensional plane), $\hat{\boldsymbol{y}}^{\boldsymbol{obj}}{}_j$ 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 is 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.

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

$$\mathbf{C} = \begin{bmatrix} D_{1,1} & D_{1,2} & \cdots & D_{1,m} \\ D_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ D_{n,1} & \cdots & \cdots & D_{n,m} \end{bmatrix}. \tag{21}$$

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

$$D_{i,j} \geq d \tag{22}$$

the measurement i is not very likely to be a measurement originating from the object in tracking gait j. Hence, if all values along a column j are equal to infinity, it is assumed that a measurement for tracking gait j is not present. To cope with this, column j should be removed from $\mathbf{C}$, 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 $\mathbf{C}$, and a new tracking gait should be instantiated with measurement i as the initial position.

After calculating $\mathbf{C}$ and removing superfluous rows and columns, the data association problem is a matter of finding the combination of distances $D_{i,j}$ that yields 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 [19] to the modified matrix $\mathbf{C}$.

6. RESULTS

Two test flight using a similar payload to the one described in Section 2 were conducted with a fixed-wing UAV in order to collect experimental data. However, instead of using an

analog to digital converter to capture the thermal video feed on-board, the video feed was sent to the ground station using an analog transmitter (in the UAV) and receiver (in the ground station). Sending the video signal to the ground station before recording it introduced some noise into the thermal video feed. Because this noise will not be present when processing the video feed directly on-board, all of the following results are listed with both noise included and filtered out. The noise was filtered out by removing the frames where noise was visible from the video data.

In this paper it is assumed that the UAV was flying at a constant altitude when objects are visible in the image frame. Since this assumption is quite realistic for the flight data, m_{00} from (5) can be used instead of A in the feature vector. Hence, the results presented in this paper is based on using m_{00} instead of A for classification.

Detection Results

To perform the detection step, a couple of design parameters has to be chosen. Specifically, the size and standard deviation of $\boldsymbol{g}$ and the gradient magnitude threshold value T_g have to be set. It was found that a 9×9 kernel approximating a Gaussian distribution with a standard deviation of 5 was an appropriate choice for $\boldsymbol{g}$. The threshold value T_g in (4) was set to 80. This value was found by manually comparing the gradient magnitudes for gradients originating from objects with the gradient magnitude for gradients originating from noise, waves and ripples in the ocean. An average for the gradient magnitude of each of these two cases (object/non-object) were found, and the threshold was set at the middle value of the two.

The result of performing the detection algorithm described in Section 3 on thermal videos gathered during the two test flights is shown in Table 1. The thermal videos consists of almost 3000 thermal image frames. The images contain humans in the water, a RIB (small boat), a 55m long ship (big boat) and also some buoys and some underwater vehicles floating in the ocean surface (the last two categories are referred to as "Other" in the results). Specifically, the test data consists of 203 images in the "Human" category, 659 images in the "Small Boat" category, 1217 images in the "Big Boat" category and finally 380 images in the "Other" category. The "False Positive" category consists of detections which are neither caused by noise in the image nor contain any object of interest.

Table 1. Object Detection Algorithm Result on Dataset

Category	Detections	Percentage Detected
Human	200	98,5%
Small Boat	659	100%
Big Boat	1215	100%
Other	374	98,4%
False Positive	177	-
Noise	510	-
Total	3135	99,6%

A lot of the detections listed as a false positive was because the horizon was visible in some of the images. This is detected as a strong edge in the image, as the contrast between the skies and the ocean is big. This could have been avoided by using AHRS data to detect when the horizon is in the field of view of the camera. The remaining detections in this category was due to irregularity in the pixel intensities caused by the thermal camera. However, it should be able to fix this problem by calibrating the camera better, avoiding that some regions show up darker or lighter than what they should.

Looking at Table 1, it is seen that apart from a fairly high count of detections caused by the noise introduced by the analog transmitter and receiver, the detection step has a good performance. Around 5% false positives (average of $0,059$ false positives per image frame) is quite low, especially considering that only 9 instances $(0,37\%)$ of objects of interest remained undetected. In addition, the number of false positives can be decreased further by incorporating horizon detection in the process.

Classification Results

In order to perform classification on the output from the object detection algorithm, the class specific feature vectors is calculated according to Section 4 using 5 example images of each class. Furthermore, the threshold r in (14) has to be chosen. In Figure 8, the correct classification rate (CCR) is shown for values of r between 0.1 and 2. The CCR is defined as the number of correct classifications divided by the total number of classifications made. It is readily seen from Figure 8 that the choice for r affects the classifier's ability to correctly classify the detected objects. The difference between the best and worst r value is not more than 0.047, however, for this dataset this implies a doubling of the number of wrong classifications. This indicates that the choice for r is just as important as finding representative training examples. It should be noted that optimal value for r for the specific

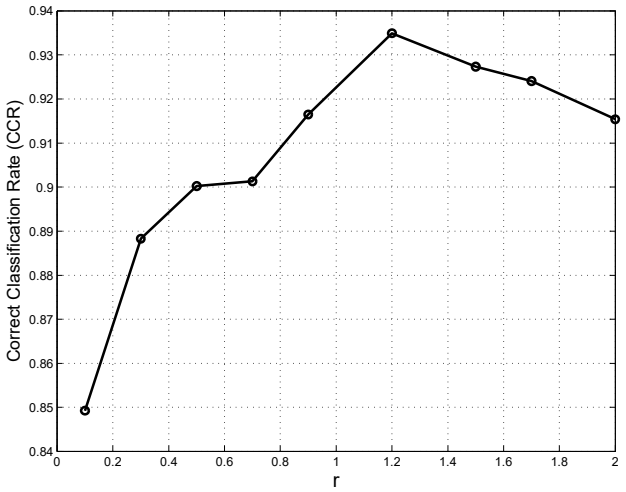

Figure 8. Classification performance for different values of the threshold r.

dataset collected during the test flights is $r \approx 1.2$. Hence, the rest of the results are based on this choice for r.

The result of using the trained classifier to classify the output from the detection algorithm is shown in Table 2. The "Other" category was omitted from this, as the classifier was not trained to classify these objects. Comparing Table 1 with Table 2 it is seen that 11 human images, 199 small boat images, 1202 big boat images, 79 false positives and 348 noise detections were touching the border of the image frame. Hence, the number of classified objects in each category is fewer than the number of detected instances. This is because

Table 2. Object Classification Result

Category	Correct	Wrong	CCR
Human	174	15	92%
Small Boat	452	8	98%
Big Boat	13	0	100%
False Positive	87	11	89%
Noise	136	26	84%
Total	862	60	93,3%

CCR = Correct Classification Rate

Table 3. Object Tracking Result

Category	Instantiated	Success	Failed	Missed
Human	6	6	0	0
Small Boat	18	17	1	0
Big Boat	23	22	1	0
Other	12	9	1	2
False	5	-	-	-
Total	64	54	3	2

the whole object is required to be within the image frame to perform classification.

Looking at Table 2 it is apparent that the classifier is successful at classifying the categories that have a reference point in the feature space. Furthermore, the classifier has a slight performance loss on unknown data such as false positives and noise. However, since the detection of false positives and especially noise will usually not persist over several subsequent frames, this should not affect the tracking algorithm to a large extent.

Considering that the training process only consisted of 5 instances of each type of object, it will actually be possible to train the classifier on-the-go, i.e while the UAV is in the air. Using images sent from the thermal camera to the ground station, reference points for certain types of object can be constructed in a matter of minutes. These reference points can then be sent back to the UAV for direct application in the classification process.

Tracking Results

For the object tracking module it is necessary to both tune the Kalman filter (i.e, deciding the covariance matrices for $\boldsymbol{w_k}$ and $\boldsymbol{v_k}$ from (15)) and to choose a value for the threshold d in (22). It should be noted that these choices are interconnected, as the distance calculated in (20) is dependent on the tuning of the Kalman filter. This is because (20) includes the Kalman filter's prediction covariance matrix $\boldsymbol{S}$.

By trial and error, the standard deviation for each element in the process noise vector $\boldsymbol{w_k}$ was set to 1. To indicate that the detected object locations (the measurements) typically will be more precise than the motion model for the tracked objects, the standard deviation of each element in the measurement noise vector $\boldsymbol{v_k}$ was set to 0.1. 50 was found to be a good value for the threshold d.

For the tracking module to be less affected by false positives in the detection step, a newly initialized tracking gait is required to be matched to a detected object in at least 3 out of 5 subsequent frames. Once this happens, the tracking gait is considered to be tracking an object of interest, and is then only required to be matched at least once ever 5 subsequent frames after that.

The performance of the tracking module can be seen in Table 3. It is seen that over the course of analysing almost 3000 thermal images, the tracking module instantiated 64 tracking gaits. Out of the 64 tracking gaits, 5 of them were not tracking an actual object. 3 of these tracking gaits were instantiated on the basis of false positive detections caused by the horizon, and hence could have been avoided using horizon detection. The other 2 were instantiated because of false detections caused by object detections caused by irregularities in the pixel intensity in the camera, and could possible be avoided by calibrating the camera better. It should also be noted that out of the 5 falsely instantiated tracking gaits, only 1 of them were classified as an actual object (small boat).

Furthermore, from Table 3 it is seen that 3 of the instantiated tracking gaits at some point failed to track the object of interest successfully. When a tracking gait fails to track the object, it means that the tracking gait at some point drifts away from the object's actual position, up to the point where the tracking gait is no longer associated with new detections of the tracked object. There are mainly two things that can make this happen. One is that the detection algorithm stops detection the object properly, making the tracking gait use the object's predicted position as a measurement, instead of the object's actual position in the image frame. The other possibility for this to happen is that the object simply is moving in a non-linear pattern and/or too fast in the image frame for the tracking gait to converge to the object's actual position. In our case, it is the latter scenario that caused the tracking to fail. More specifically, it was the sharp turns and high speed of the fixed-wing UAV that resulted in quick and non-linear object displacements in the image frame.

The 2 objects that were not tracked were also moving too fast for the tracking algorithm to instantiate a tracking gait. This was because the tracking module did not link the detections of the those objects together, and instead seeing each detection as a completely new object. This indicates that the tracking module would benefit from incorporating the GPS and AHRS measurements in the tracking process, as well as possibly using a more complex motion model for the Kalman filter. Tracking the object's in the world frame would possibly also enhance the performance, as the object's movement is more linear in this coordinate frame.

Real-Time Performance

To evaluate the real-time performance of the system, the detection, classification and tracking module was implemented and tested on the on-board single board computer. The machine vision algorithms were implemented using C++ and OpenCV [20]. OpenCV (Open Source Computer Vision) is a library of functions mainly aimed at real-time computer vision. The data association problem was implemented with a modified version of hungarian-cpp [21], an open source C++ solution to the minimum assignment problem based on the Kuhn-Munkres algorithm.

Since the on-board processing was done post flight, a real-time simulation test is used to evaluate the system's real-

time performance. That is, the time required for detection, classification and tracking for each frame is recorded for each time step. After processing an image, the system skips forward the number of frames that the system would have missed if the algorithms were running on-board during a flight. The number of frames to skip is calculated as

$$f_{skip} = ceil(t_f \times 30) \tag{23}$$

where f_{skip} is the number of image frames to skip, t_f is the time in seconds used processing image frame f, and 30 is the (upsampled) number of frames per seconds output from the thermal camera. $ceil$ is a function rounding up to the nearest integer.

The real-time test was first executed on the original dataset ($\sim$ 3000 images with 720×480 resolution) using only one of the PandaBoard's two 1.2GHz ARM processors. This resulted in a processing speed of ~ 5.5 frames per second. However, this yielded a poor tracking performance as the detected objects are moving quickly and in a highly non-linear way in the image frame. Hence, to better the tracking performance, every image was resized (in run-time) to a 510×340 resolution image before the detection, classification and tracking was performed. This resulted in a processing speed of $\sim$ 10 frames per second, and the tracking performance listed in Table 4. Note that increasing the processing speed beyond 9 frames per second would not necessarily yield better tracking performance. This is because higher processing speeds will yield measurements which are correlated in time due to the upsampling from 9 to 30 frames per second. This means that the Kalman filter should be adapted to take correlated in time measurements into account when the processing speed is greater than 9 frames per second.

Table 4. Object Tracking Real-Time Result

Category	Instantiated	Success	Failed	Missed
Human	5	5	0	1
Small Boat	18	16	1	1
Big Boat	22	20	1	1
Other	9	5	4	3
False	5	-	-	-
Total	59	46	6	6

There is a noticeable, and expected, performance loss when the detection, classification and tracking is run in the on-board hardware. However, it should be noted that of the object types that we are actively searching for (humans, small boat and big boat), 85% of them were tracked successfully. As before, the tracking gaits that failed, and the objects that were not tracked was due to the fast and non-linear movement of objects in the image frame. Hence, actively accounting for the UAVs movements by incorporating the on-board GPS and AHRS measurements in the tracking process is necessary in order to achieve better real-time tracking performance.

7. SUMMARY

This paper discusses the development and implementation of a machine vision system for a low-cost fixed-wing UAV with a total flying weight of under 4kg. The machine vision system incorporates the use of a thermal imaging camera and on-board processing power to perform real-time object detection and tracking of objects at the ocean surface. Using a simple edge detector and some filtering, the system is able to detect $99,6\%$ of objects of interest located in the ocean surface. Of the detected objects, 5% were found to be false positive. A simple nearest neighbour classifier based on object size, temperature and an invariant moment function is then constructed. Using only the average of 5 examples from each class as reference points, the classifier is able to classify $93,3\%$ of the detected objects correctly. The classifier is highly agile, allowing the user to quickly define which object characteristics that should be considered during classification, and what types of objects to classify. Furthermore, a tracking algorithm combining the Kalman filter with a simple linear motion model and a global nearest neighbour algorithm for data association is implemented in the on-board single board computer. The system is able to successfully track 85% of the object types it is actively searching for during real-time simulation, using only a 1.2GHz ARM processor. However, it is apparent that including on-board GPS and AHRS data in the tracking process is necessary to further improve the tracking performance.

ACKNOWLEDGMENTS

This work has been carried out at the Centre for Autonomous Marine Operations and Systems (AMOS). The Norwegian Research Council is acknowledged as the main sponsor of AMOS. This work was supported by the Research Council of Norway through the Centres of Excellence funding scheme, Project number 223254. A special thanks to Universidade do Porto and the Laboratrio de Sistemas e Tecnologias Subaquticas (LSTS) for support in conducting the flight experiment. The authors would also like to thank ONR Global for travel support to Portugal.

REFERENCES

[1] R. Sengupta, J. Connors, B. Kehoe, Z. Kim, T. Kuhn, and J. Wood, "Final report - autonomous search and rescue with scaneagle," September 2010.

[2] P. Doherty and P. Rudol, "A UAV search and rescue scenario with human body detection and geolocalization," in *Proceedings of the 20th Australian Joint Conference on Artificial Intelligence (AI)*. Springer Berlin/Heidelberg, 2007.

[3] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, 2001, pp. I–511–I–518 vol.1.

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

[5] A. Gaszczak, T. Breckon, and J. Han, "Real-time people and vehicle detection from UAV imagery," in *Proc. SPIE Conference Intelligent Robots and Computer Vision XXVIII: Algorithms and Techniques*, vol. 7878, no. 78780B, 2011.

[6] M. Teutsch and W. Krüger, "Classification of small boats in infrared images for maritime surveillance," in *Proceedings of 2nd NURC International WaterSide Security Conference*, Marina di Carrara, Italy, Nov. 2010.

[7] J. Haugen, L. Imsland, S. Løset, and R. Skjetne, "Ice observer system for ice management operations," in *Int. Offshore (Ocean) and Polar Eng. Conf. (ISOPE)*, vol. 21, 2011, pp. 1120–1127.

[8] (2014) Apm autopilot suite. [Online]. Available: http://www.ardupilot.com/

[9] (2011) Pandaboard es. [Online]. Available: http://www.pandaboard.org/

[10] (2011) Flir tau2 336. [Online]. Available: http://www.flir.com/cvs/cores/view/id=54717

[11] (2011) Axis m7001. [Online]. Available: http://www.axis.com/en/products/camm7001/index.htm

[12] (2011) Ubiquiti rocket m5. [Online]. Available: http://www.ubnt.com/airmax/rocketm/

[13] P. Soille, *Morphological Image Analysis: Principles and Applications*, 2nd ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003.

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

[15] M.-K. Hu, "Visual pattern recognition by moment invariants," *Information Theory, IRE Transactions on*, vol. 8, no. 2, pp. 179–187, February 1962.

[16] J. Flusser, "Moment invariants in image analysis," *World Academy of Science, Engineering and Technology*, vol. 11, pp. 376–381, 2005.

[17] A. Ali and K. Terada, "A general framework for multi-human tracking using Kalman filter and fast mean shift algorithms," *Journal of Universal Computer Science*, vol. 16, no. 6, pp. 921–937, mar 2010.

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

[19] F. Bourgeois and J.-C. Lassalle, "An extension of the munkres algorithm for the assignment problem to rectangular matrices," *Commun. ACM*, vol. 14, no. 12, pp. 802–804, Dec. 1971.

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

[21] D. Schwarz. (2012) Minimum assignment problem solver. [Online]. Available: https://code.google.com/p/hungarian-cpp/

BIOGRAPHY

Frederik Stendahl Leira *MSc Frederik Stendahl Leira received his MSc degree in Engineering Cybernetics in 2013, specializing in robotics and computer vision. He is currently in his second year of his PhD fellowship within the Center of Excellence on Autonomous Marine Operations and Systems (AMOS). His PhD topic is infrared object detection and tracking in UAVs.*

Tor Arne Johansen *Professor Tor A. Johansen (MSc, PhD) worked at SINTEF as a researcher before he was appointed Associated Professor at the Norwegian University of Science and Technology in Trondheim in 1997 and Professor in 2001. He has published several hundred articles in the areas of control, estimation and optimization with applications in the marine, automotive, biomedical and process industries. In 2002 Johansen co-founded the company Marine Cybernetics AS where he was Vice President until 2008. Prof. Johansen received the 2006 Arch T. Colwell Merit Award of the SAE, and is currently a principal researcher within the Center of Excellence on Autonomous Marine Operations and Systems (AMOS) and director of the Unmanned Aerial Vehicle Laboratory at NTNU.*

Thor Inge Fossen *Professor Thor I. Fossen received the MSc degree in Naval Architecture and the PhD in Engineering Cybernetics in 1987 and 1991 both from the Norwegian University of Science and Technology. He was elected into the Norwegian Academy of Technological Sciences in 1998. Fossen has graduated more than 150 master students and he is the main supervisor of 27 PhD candidates. He is teaching, mathematical modelling of aircraft, marine craft, unmanned vehicles and nonlinear control theory. Fossen has authored approximately 300 scientific papers and 5 textbooks including the Wiley textbooks "Guidance and Control of Ocean Vehicles" and "Handbook of Marine Craft Hydrodynamics and Motion Control". Fossen is one of the co-founders of Marine Cybernetics where he was Vice President R&D in the period 2002-2008. His paper on weather optimal positioning control of marine vessels received the Automatica Prize Paper Award in 2002. In 2008 he received the Arch T. Colwell Merit Award.*