Detection of Human Over Board in Maritime Environment

Sara-Jane Bittner, *s.bittner@student.utwente.nl, s2876574, Master Interaction Technology*
and József-Hunor Jánosi, *j.janosi@student.utwente.nl, s2832208, Master Computer Science*
and Aimé Ntagengerwa, *m.a.ntagengerwa@student.utwente.nl, s2641933, Master Computer Science*

***Abstract*—Human-over-board in maritime scenarios is a crucial situation since many people go unsaved. Out on the sea rough weather conditions as well as darkness can make salvation hard. For this it is important to develop robust and accurate algorithms that are able to detect and track objects on the sea despite them going underwater or unseen due to waves and camera shakiness. In this project we are discussing the robustness of horizon detection video stabilization, object detection through interest point detection and distance calculation. Results show acceptable accuracy, but they are sensitive to camera stabilization.**

***Keywords*—computer vision, distance estimation, maritime object detection, video stabilization**

I. Introduction

THE following report contains the development and evaluation of a computer vision algorithm for *human-over-board* scenarios in a maritime environment.

Human-over-board (HOB) scenarios are often referred to as *man-over-board* scenarios. In regards of a wider diversity and equality in maritime context, we will refer to it in context of this report as *HOB*. These scenarios describe incidents in which a human falls over board and is in need of rescue [15]. While no global statistics are widely available, research has shown that in average only around 17% to 25% of the total humans falling over board get rescued[17], [19]. Aligning with this, *HOB* incidents were ranked as the second highest influence for fatal accidents in commercial fishing in the USA [16]. In this context, 206 commercial fishermen fell into the sea in between 2000 and 2016 of which a high majority of 89.3% were not able to be found in the water after the incident [16]. Additionally, research of the *USBOAT Foundation* of boating fatalities has indicated that 79% died during-day time and even 90% of the incidents occured during calm weather conditions [18]. Based on these findings, which indicate a lack of solutions, the importance of robust and efficient algorithms for *HOB* scenarios becomes clear.

The detection and tracking of *HOB* includes a range of difficulties as the maritime environment is prone to various rapid change in weather conditions. Moreover, the moving of the attached camera system, due to waves, includes noise, which needs to be handled additionally. The aim of this project is to develop a robust and efficient algorithm for the detection and tracking of *HOB* scenarios. In order to do so the following research questions were set:

1) *Is horizon detection and adjustment a robust video stabilizer?*
2) *To what extent can an interest point detecting algorithm, and distance calculation from the object's last seen coordinates detect and track a distinctive object in the sea, when waves and camera shaking is obstructing?*

II. Methods and Materials

A. Materials

In context of this course a variety of materials were used. Firstly, software-wise, *matlab*, a programming and numeric computation platform was used. For this, functions from the *image processing* and *computer vision toolbox* were applied.

The theoretical background for our methods, lecture notes were taken into account from the course of *Image Processing and Computer Vision*, at the *University of Twente*, as well as various research papers that dealt with: *horizon, edge, or corner detection* [2], [4], [6], and *computer vision solutions for human-overboard* [15]. Extensive review and research has been done on the field of *maritime object detection*, and on the comparison of the different horizon detection, background and foreground filtering, image registration in maritime scenarios by Prasad et al. [12], [9].

B. Methods

1) Analysis: There are three main problems discussed in this paper:

- Image Stabilization
- Small Object Detection and Tracking
- Distance Estimation

The input for our algorithm is a video of the sea, in which a small object, e.g., a buoy or a man overboard, is seen, but which can disappear due to waves. The video's stability is oftentimes questionable, due to the camera's location that is usually in another boat or buoy that rocks with the sea.

The desired output is real-time detection of the small object, that can be still tracked if waves obstruct vision, and an estimated distance from the camera to the object.

The challenge of **Camera Stabilization** is to find fixed points in every frame of the video that can serve as a baseline. As a task of image registration, every frame can be stabilized so that these fixed points overlap in every frame [10].

Object tracking in maritime scenarios is difficult due to maintain since the object itself can go underwater, disappear, or get covered by waves. There are several methods to solve this problem. From one frame to another motion vectors can be obtained, and a motion model could be constructed which would estimate the object's location if it went unseen. This can be done with the *Lucas-Kanade algorithm* [11]. Another possible solution is to apply *Blob Analysis* on the possible objects of the frame and filter out the ones that do not interest us, e.g., by area or shape. We chose to track the object by means of calculating the distance between the estimated object's coordinates and between all possible objects detected in the image.

This algorithm needs to be robust to weather conditions, to heavy wind and waves, to great amount of shakiness of the camera.

The **Distance Estimation** seems to be the most complex task, since it involves the most amount of parameters that need to be taken into consideration, e.g., the earth's curvature and radius, the distance of the horizon that is still noticeable for the human eye, the camera's intrinsic and extrinsic parameters as well as its distortion, and the fact that closer objects are bigger than objects that are further away.

The following subsections cover the approach of providing an algorithm for man over board scenarios in a maritime environment as well as the performance evaluation of this algorithm. For this the section is divided in four main steps: *Camera Calibration*, *Image Stabilization*, *Detecting and Tracking of the Buoy*, and *Distance Estimation*.

*2) **Camera Calibration**:* As of the first step of the algorithm, the camera has been calibrated with 26 images of a checkered board, so to undistort the camera, and to obtain the camera's intrinsic parameters that are being used in the distance estimation. More detail can be derived from section II-B5.

*3) **Image Stabilization** :* The following section covers the stabilization of input frames. The result of this step is a stabilized frame, which has the horizon lying horizontally at the center of the output frame. The camera itself is subject to movement (translation) in three dimensions, as well as rotation in all three euclidean axes. Rotation on these orthogonal axes describe tilting, panning and pitching. The horizon is chosen as a reference point for stabilization. Under reasonably clear weather conditions, the horizon is nearly always visible at sea, in all compass directions. This makes it so that this part of the proposed program could be applied in many situations. The horizon is also very stable in its position relative to the camera. Lastly, being a straight line with significant changes in image properties on each side of it [2], the horizon gives us enough information to compensate for (significant) vertical movement, and the tilting and pitching of the camera.

Four main steps were executed to achieve horizon detection and image stabilization: **a)** The edges of the image were detected based on the *Canny Edge Detection Algorithm*. **b)** The detected edges were *morphologically dilated* to close small gaps between line segment and create longer uninterrupted lines. **c)** *Hough Transformation* was used on the dilated edge map to get a spatial intensity map. From this, we identify the longest straight line; a segment of the *horizon*. **d)** This segment is interpolated to a linear function, describing the orientation and position of the horizon in image coordinates. We apply affine transformations (translation and rotation) to place this horizon horizontally at the center of the y-axis of the output frame.

Doing this for every frame results in a stable video which preserves information about the relationship between any two points in the frame.

a) Canny Edge Detection: The *Canny Edge Detection* was applied to find the lines of the image. It returns a binary edge map which is further used for calculations in the paragraphs II-B3b and II-B3c. The algorithm consists of five steps:

1) The *Gaussian* (see equation 2) is convolved with the image to smooth it as seen in equation 1. [4], [1].

$$S = I \cdot g(x, y) \tag{1}$$

$$g(x, y) = \frac{1}{\sqrt{2\pi}\sigma} e^{-} \frac{(x-y)^2}{2\sigma^2} \tag{2}$$

2) The Derivative of the filtered image is computed as can be derived from equation 3.

$$\bigtriangledown S = \bigtriangledown(g \cdot I) \tag{3}$$

3) The gradient magnitude and the gradient direction are detected.

$$Magnitude = \sqrt{S_x^2 - S_y^2} \tag{4}$$

$$Direction = \theta = tan^{-1} 1 \frac{S_x^2}{S_y^2} \tag{5}$$

4) The *Non-maximum-suppression* is applied. In this step points with the largest magnitude are detected along the curve. For this,the curve is divided in slices. Then for every slice, the point with the maximum magnitude is chosen. Resulting in a line of points with maximum magnitude for the curve. Thus the other pixels, that not represent a maximum in $|\triangle S|$ are suppressed.

$$|\triangle S| = \sqrt{S_x^2 - S_y^2} \tag{6}$$

5) Lastly, the *Hysteresis Thresholding* is applied to detect true lines.For this, an upper threshold ($\sigma = 0.0019$) and a lower threshold ($\sigma = 0.0016$) were set. If the gradient magnitude at a pixel is higher than the upper threshold it is categorized as a true edge, while pixels with a gradient magnitudes that fall under the lower threshold are discarded. If gradients of pixel are between both thresholds it is checked if they connect to a detected true edge (with a gradient above the upper threshold). If this applies, the are counted to the true edges.

After applying the *Canny Edge Detection*, the binary edge map showcased several smaller detected lines aligning roughly with the horizon line. However, no connected horizon line was detected. Additionally, false edges were detected throughout the image.

b) Morphological Dilation: Morphological Dilation was applied on the edge map of *Canny Edge Detection* to bridge between the gaps of the various smaller lines that align with the natural horizon line. This way, the detected lines on the horizon are visually connected. Thus, enabling future calculation steps to calculate on the visual appearance of one continuous horizontal line.

Mathematical Morphology is used to extract components of an image which are useful for the representation and description of a region shape. [1]. Further, *morphological dilation* is defined with A and B as sets in Z^2, the dilation of A by B, denoted $A \oplus B$:

$$A \oplus B = \{z|(B)_z \cap A \neq \emptyset\} \quad (7)$$

The set of all displacements z, the translated B and A overlap by at least one element as can be derived from equation 8:

$$A \oplus B = \{z|[(B)_z \cap A] \subseteq A\} \quad (8)$$

For the dilation of the lines a *structuring element* of 3x3 was used. A *Structuring element* is defined as a sub set which has an origin. [1]. To enforce the visual horizon line based on the edge map of the *Canny Edge Detection*, the displacements z need to dilate the detected lines horizontally along the x axis.

c) Hough Transform: The *Hough Transform* was applied on the edge map with dilated lines to derive the longest straight line in the frames. This one corresponds with the *horizon line* [2]. For this the image is translated to the *Hough Space* which represents an intensity map. In this space bright areas (high values) correspond to straight lines [1].

The algorithm detects straight lines in three steps.

1) It detects line elements.

2), a accumulator array H is set up on basis of $N\ x\ M$. Here every column indices are mapped to a representation of θ in the range -90° to +90°. Thus, covering straight lines. In comparison, row indices are mapped to a representation of ρ. Resulting in representation of a point as $H = (\rho, \theta)$.

3) Lastly, for all detected line elements the column indices θ a parametric representation of the line are calculated. The corresponding formula can be derived from the equation 9. Based on that the accumulator array is incremented so that $H(\rho, \theta) := H(\rho, \theta + 1)$. [3], [2]

$$p = x \cdot cos(\theta) + y \cdot sin(\theta) \quad (9)$$

The horizon line is assumed to be the most prominent line, based on the natural characteristics of the horizon [2] and the applied dilation, which amplifies it. Therefore, the peaks of the intensity map of the *hough space* indicate the position of the horizon line. Based on the return straight lines which represent the same *hough transform* are connected. This results in the return of the connected horizon line.

d) Applying Geometrical Transformation: Having found a segment of the horizon, we can interpolate it to a linear function which describes a straight line in the image coordinate space. Over the interval of the image width, this function represents the horizon. The Hough transform yields two points at the far ends of the detected segment of the horizon. We first find the slope of the linear function. We can then determine the vertical translation of that function with respect to the center of the image. We determine the orientation of the horizon with respect to the horizontal image axis and, lastly, we apply affine geometric transformations to the input frame in an effort to stabilize it. These transformations are the inverse of the transformations that are applied to a centered horizontal line to obtain the horizon function. This results in aligning the translated and rotated horizon in the input image with the x-axis of the image coordinate space. The output of this step is the stabilized input frame.

4) ***Detect & Track Buoy****:* The following section covers the detection of the buoy in a region of interest of a frame, and tracking of that detected point. The algorithm is illustrated on figure 2, and on algorithm 1.

a) Cropping a large ROI: As a first step, we have cropped the original image down to a 201x31 rectangle, that covers the area where the buoy could move in the entirety of the video. This crop can be seen on the images on figure 2. For the actual tracking and detection of the buoy happened on this crop throughout the video.

On the first frame, the buoy was pinpointed to get the initial coordinates before the algorithm starts.

b) Static Dark Background Substraction: During the algorithm a mask was applied to the frame so that only light pixels are showcased. On the grayscale of the image an intensity threshold of 160 was applied, so that the pixels darker than 160 were zeroed out, e.g., water of the sea. On the second image on figure 2 it can be seen that only the buoy and the white foam of the waves are left, leaving us with a few blobs on a single frame. It is assumed that the object that needs to be detected has a higher intensity, i.e., it is light or white. This is an *image statistics* method that is similar to the method of Bhanu and Holben [12] [13].

c) Harris Corner Detection: Based on this output, the *Harris Corner Detection* has been used to detect the potential key points of the ROI, i.e., the blobs' locations.

Harris' corner detector takes the differential of the corner score into account with reference to direction directly, instead of using shifting patches for every 45 degree angles [6]. A corner is a point whose local neighborhood stands in two dominant and different edge directions. Corners are the important features in the image, i.e., interest points which are invariant to translation, rotation and illumination [8].

d) Euclidean Distance & Threshold: Lastly, the euclidean distance for each returned corner point from the *Harris Corner Detection* to the position of the buoy in the prior frame is calculated. The minimal euclidean distance represents the current location of the buoy in the current frame. The output of the corner which represents the buoy by the algorithm can be derived from figure 2. For the initial detection the buoy

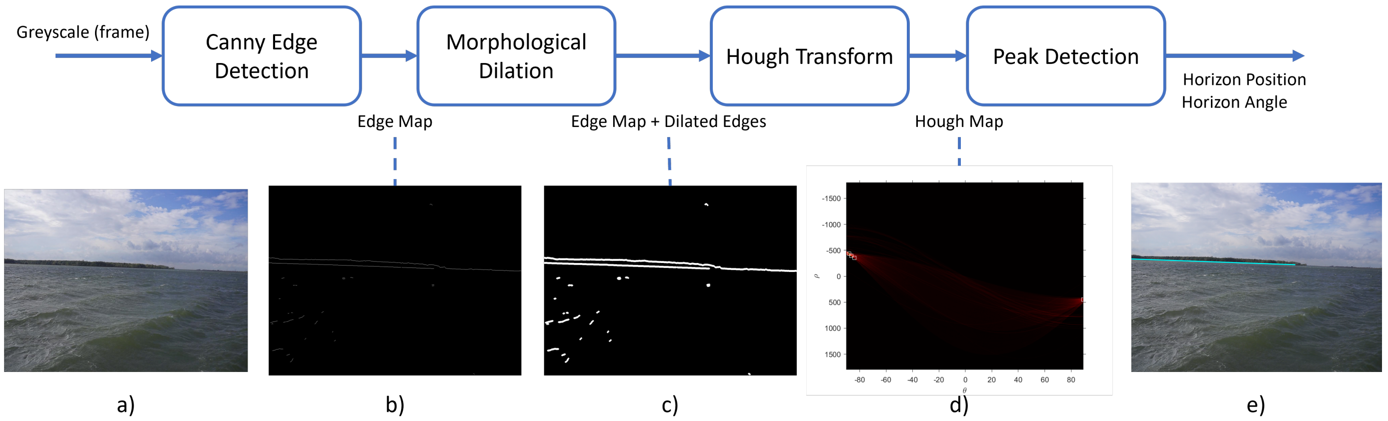

Fig. 1: Edge-based Horizon Detection

was pinpointed to the derive the starting x and y coordinates. After this firs step, the prior location is updated to the current location of the buoy in the following frame. The updated buoy locations is determined through the detected *Harris Corner Point*, with the minimal euclidean distance. To increase robustness, the system covers for the visual disappearance of the buoy, when it is covered by waves or it is under water. For this a threshold was applied for the maximum euclidean distance of *Harris Corner Points*, which are considered as potential candidates. The threshold *11* corresponds to the value that showed the best results based on a series of experiments.

For the distance calculation the *2D Euclidean Distance* was used. The *harris Corner Detection* outputs a matrix corresponding to the detected corner points

Algorithm 1 Buoy detection

```
1: procedure DETECTBUOY
2:     imCrop ← 201x31 crop ROI of grayscale frame
3:     buoyCoord ← pinpoint buoy coords on first imCrop
4:     imCrop ← threshold the intensities > 160
5:     corners ← HarrisCornerDetection on imCrop
6:     closestCorner ← closest corner to buoyCoord
7:     if EuclideanDistance(closestCorner, buoyCoord) < distanceThreshold then
8:         buoyCoord ← closestCorner
```

*5) **Distance Estimation:*** The third and last main problem this paper will discuss is that of distance estimation. With this, we attempt to estimate the distance between the camera's position on the water and the buoy in real-world units (meters). We make a distinction between slant range distance - which is the length of a line straight from the camera to the point of interest on the surface of the earth - and ground distance - which is measured as the distance to the point of interest as if walking (or rather swimming) there from the coordinate position of the camera along the surface of the earth. For the output of this step, we are interested in the ground distance between the camera and the point of interest. First see Figure 3 to get an intuition about the geometry of distance estimation.

To estimate the ground distance between the camera and the detected buoy, we use trigonometry. A given is the elevation of the camera above sea level, and the horizon serves as a point of reference.

We make a few assumptions and approximations here. Firstly, we assume the shape of the earth to be a perfect sphere. We give this spherical model the globally-average radius of the earth; we approximate this to be 6371 kilometers [14]. Second, the camera's elevation above sea level is assumed constant, and stands at 2.5 meters.

Knowing the slant range distance between the camera and the horizon allows us to calculate the angle θ between the earth surface normal line at the position of the camera, and the line of sight from the camera towards the horizon.

$$\Theta = \arctan\left(\frac{r}{\sqrt{(h+r)^2 - r^2}}\right) \tag{10}$$

Next, we approximate that the bottom-center pixel $P_{ground-position}$ in the image maps to the ground position of the camera. We then trace a line through this ground point pixel and the position of the buoy in the image P_{buoy}, and obtain the point $P_{horizon}$ where this line intersects with the horizon. A vector $V_{buoy-horizon}$ is constructed between P_{buoy} and $P_{horizon}$.

Matlab's camera calibration tool gives us access to focal lengths F_x and F_y in the unit of pixels (in the x and y direction respectively). These are the product of the hidden attributes F (focal length in world units) and S_x, S_y, which are the number of pixels per world unit of the camera sensor in the x and y direction. Because these attributes are hidden, we must infer the focal length of the camera by determining the focal length (in pixels) in the direction of our vector $V_{buoy-horizon}$.

To find the focal length of the camera in the direction of $V_{buoy-horizon}$, we construct an ellipse in an independent 2D reference frame, which intersects the x- and y-axis at F_x and F_y respectively. We then place $V_{buoy-horizon}$ in this reference frame, and trace a line from the origin in its direction.

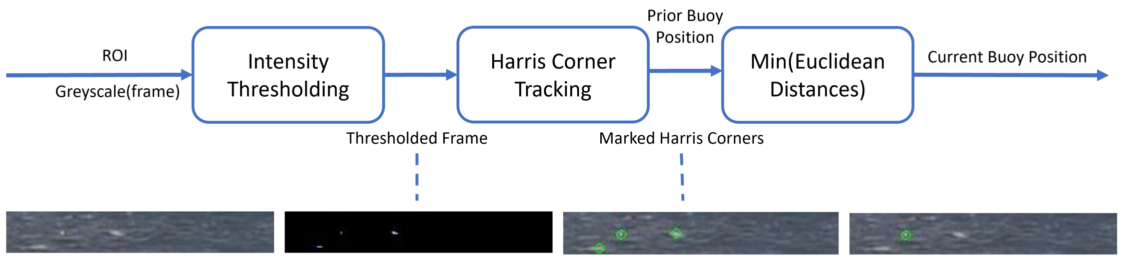

Fig. 2: Detect and Track Buoy

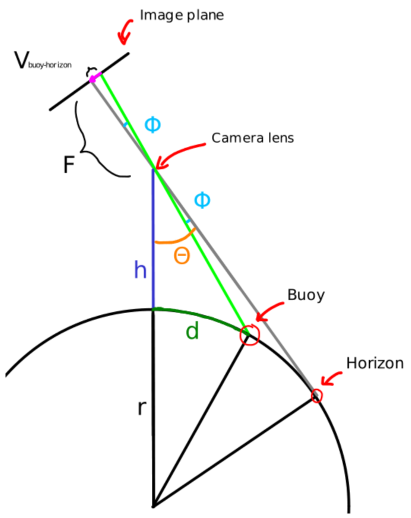

Fig. 3: The geometry of distance estimation

We create a new vector between the origin and point where this line intersects the ellipse. This new vector has the same orientation as $V_{buoy-horizon}$, but its magnitude approximates the focal length F_{pixels} (in pixels) of the camera in that direction.

Having obtained F_{pixels}, we can use it and the magnitude of $V_{buoy-horizon}$ to determine the angular size ϕ of $V_{buoy-horizon}$.

$$\phi = \arctan\left(\frac{|V_{buoy-horizon}|}{F_{pixels}}\right) \tag{11}$$

This step assumes that the camera sensor if orthogonal to the line of sight between the camera and $P_{horizon}$: that the camera is pointed straight at the horizon. This is generally not the case (see Section II-B3), and so this assumption introduces some degree of error in the resulting distance estimation.

Knowing the angular size of $V_{buoy-horizon}$ allows us to describe a line of sight between the camera and the buoy. We can then determine the intersection point (x_0, y_0) of that line with the surface of the earth.

Then we determine, at the center of the earth, the angle μ between the normal line of the earth's surface at the position of the camera and the buoy.

$$\mu = \arcsin\left(\frac{x_0}{r}\right) \tag{12}$$

Lastly, we determine the distance D_{buoy} from the camera to the buoy in meters.

$$D_{buoy} = \mu\frac{C_{earth}}{2\pi} \tag{13}$$

with C_{earth} being the circumference of the earth in kilometers.

The distance estimation between consecutive frames can vary in the order of several meters. However, in the short time interval between consecutive frames the buoy is quite stationary with respect to the camera position. The large degree of variation is likely due to imperfect image stabilization (see Section IV-A). To reduce this, we apply a rolling average over the last ten distances. This significantly reduces the erroneous variation in distance estimation output between consecutive frames.

6) Performance evaluation: The following section covers the performance evaluation of the developed algorithm. For this, the *Intersection over Union (IOU)* [7] algorithm is proposed. This evaluation measures the degree of overlap between the *ground-truth (gt)*, so the real location of the buoy, and the *detected (dt)*, in this case, the location where the buoy is estimated by the algorithm. The formula of the *IOU* can be derived from equation 14. It divides the are of union from the area of overlap.

$$IOU = \frac{area(gt \bigcap dt)}{area(gt \bigcup dt)} \tag{14}$$

The algorithm will return a value between 0 and 1 of which 0 equals *no overlap* and 1 equals *perfect overlap*. To detect the object a threshold α is chosen. If the result is higher or equal than the threshold it is characterized as a *True Positive (TP)*, while lower than α means that it classifies it as a *False Positive (FP)*. Lastly, if a *ground- truth* was missed and *IOU* is ¡α, then it represents a *False Negative (FN)*. An illustration about this classification can be seen in figure 4. The green rectangle shows the hand-labelled gold standard location of the object, while the red rectangle represents the detected location of the object. From these values the metrics *precision*, so the degree of detected buoys, that were buoys, and *recall*, so how many of the total spotting of buoys were detected, can be calculated. In context of the problem statement, *recall* is considered as

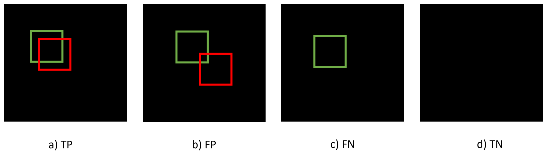

Fig. 4: IOU Illustration

the more relevant evaluation metrics, because it is crucial to keep the period of time, where the person in water can not be detected as small as possible in order to rescue them time-efficiently and prevent consequences as drowning.

The mathematical representation and the description of the classification corresponding to the buoy detection can be derived in more detail from table I.

TABLE I: Overview - Evaluation Metrics

Metrics	Formula	Description
True Positive	$IOU(gt, dt) \geq \alpha$	Buoy correctly detected
False Positive	$IOU(gt, dt) < \alpha$	Incorrect detection made by the detector.
False Negative	$missed(gt) + IOU(gt, dt) < \alpha$	A Ground-truth not detected by the object detector.
Recall	p = $\frac{TP}{TP+FN}$	How many of the total spottings of the buoy were detected
Precision	p = $\frac{TP}{TP+FP}$	Degree of detected buoys that were buoys

For this, three steps are needed: Get the *ground-truths coordinates*, get the *detected coordinates* and calculate the *IOU*.

1 First, for the *ground-truth* an algorithm was written in which the *labeller* can pinpoint the location of the buoy for each frame. For frames, in which the buoy is not visible, the *labeller* can point outside of the *ROI*. This will be marked than as an empty entry. The list of hand-labelled pixel coordinates represents the *gold standard*.The function saves a cell-array which includes the one-row vectors the pinpointed pixel coordinates. The code can be derived from the appendix B-A.
2 Second, the *detected* pixel-coordinates need to be derived. To achieve this, for each frame the pixel coordinates from the detected buoy location is saved in a cell array as one-row vectors.

3 Lastly, the *IOU* for each corresponding pixel coordinate pair needs to be calculated. The gold label will now be held against the standard through calculation of the *IOU* and classified as *TP*, *FP* and *FN*. From this the recall and precision will be calculated. For this, first, a same sized rectangular area was defined around the each frame for the two corresponding *gt* and *pd* coordinate. These two shapes were used to then calculate the the IOU. After this the metrics were derived.The code can be derived from the appendix B-B.

TABLE II: Confusion matrix of buoy detection

		Ground Truth	
		Positive	Negative
Prediction	Positive	TP = 320	FP = 76
	Negative	FN = 5	TN = 23

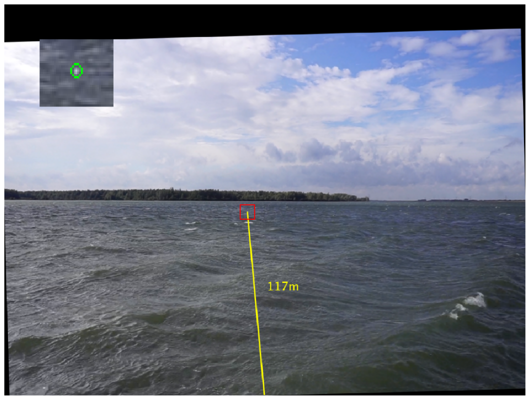

Fig. 5: Detected buoy, it’s ROI, and the distance estimated between the camera and the buoy. This being displayed on the image as well, next to a line connecting the buoy and the center of the bottom of the image.

III. Results

The following section covers the results from this project. For this the outcomes of the three main parts of the algorithm *Stabilization*, *Detection and Tracking of the Buoy* and *Distance Estimation* will be listed and described.

a) Stabilization: On figure 5 we can see the result of the camera stabilization by using the horizon detection algorithm. It can be noticed that it has been tilted, because of the black border around the sky.

b) Object detection: The object of interest, in our case a buoy, is detected, and a green circle is placed on top of it. A 31x41 red rectangle is placed on the image to show the region of interest, where the buoy lies, and also which is magnified and displayed on the left top corner. This can be seen on figure 5.

Using the metrics presented in section II-B6, the overlapping degree has been calculated for every frame for the detected location of the object and the actual location of it. The results can be seen on table II. Based on it, the precision is 0.9846, and the recall 0.8081.

From the video it can be seen that the algorithm hasn’t detected waves or other artifacts as our object of interest. This is due to the thresholded Euclidean distance. However there are frames, where the buoy is visible to the human eye, but the algorithm does not pick it up.

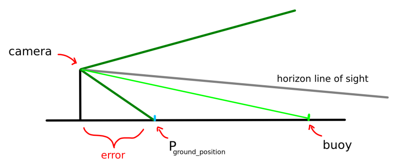

Fig. 6: Constant distance estimation offset error

c) Distance estimation: On figure 5 we can see that the displayed and detected object of interest is estimated to be 117 meters away from the viewer on the particular videoframe.

The distance is changing at almost every frame. It is ranging between 94-152 meters, and averaging at 110 meter. If the buoy is not seen and detected, the distance is not measured.

IV. Discussion

From the results, and from the resulting video we can see that the **horizon detection and stabilization** is working in an acceptable manner. The major shakes, and vibrations are stabilized, however there is room for improvement: we are still left with minor jitter. This is due to the fact that the horizon is not detected perfectly, since there is no one clear line that constitutes to the horizon, but it can have minor, couple pixel differences, which can have bigger impact on the stability of the overall frame. Because of the stabilization's impurity, the buoy detection, as well as the distance estimation is lacking efficiency.

The **detection of the interest object** is working with 80.81% recall. This is considered as an acceptable score, because there are frames where the model does not recognize the object, but it is clearly visible to the human eye. This is a crucial point, since in a real life, maritime scenario, where a possible man is overboard, seconds could determine if the person is detected and tracked, or lost. This is also caused by the improper stabilization.

However it does not detect other artifacts only our object, which is positive, since the object's coordinates are not shifted away with the supposed object that could be the foam of the wave.

There are several assumptions made for distance estimation. Figure 6 shows that there is a constant offset error between the point $P_{ground-position}$ and the true position of the camera. This is due to the fact that the program does not take the field of view (FOV) of the camera into account. Matlab's camera calibration is indeed not able to supply the needed attributes, but it could perhaps in future work be determined from a more advanced camera calibration tool, or obtained from a specific's camera hardware specifications. It is also conceivable that the FOV can be inferred from the focal length and the size of the image, but this has not been looked into.

The **estimated distance** to the buoy is making quite large jumps, ranges between 94 and 154 meters. It is not possible that the buoy - almost static with respect to the camera position - is travelling such a great distance between two video frames. This due to the imperfect stabilization; the stabilized frame can jitter, leading to the detected buoy being estimated closer to or further from the camera.

We come to an average distance of approximately 110 meters from the camera to the buoy.

A. Limitations

Limitations are that, being a conceptual ”ring” around an observer, the **horizon** moves with that observer as it translates in any compass direction. Also, because of this homogeneous shape in the horizontal axis from the observer's position, the information we can obtain from a horizon is not sufficient to compensate for panning. Further limitations include external factors, such as **adverse weather conditions**, a clear view of the horizon being obstructed by an object or landmass of significant size, or the horizon falling outside of the viewport of the camera. Mitigating these external limitations is outside the scope of this research.

With regards to the **tracking** of the object, in the current from we assume that when the buoy goes undetected, covered, or under water, it pops back up in a matter of a few frames. However if it goes unseen for a longer period, it would move further from the last point it was detected, and because of the Euclidean thresholding that we are using during the re-detection of the object, it would pop up outside of our region where we are searching for the object, thus it would not be re-detected; it would get lost.

This problem could be mitigated, in future work, by taking into account the motion model of the object, and if it disappears, the model would estimate where it is, in which direction it is moving and with what speed, based on its previous motion.

For further work, state-of-the-art methods could be tried and compared. Recent algorithms have combined multi-scale filtering and projection based approaches for providing state-of-the-art results [12].

V. Conclusion

The proposed solution successfully answers both main research questions. It addresses the stabilization of video frames based on horizon detection, and the results show that this is can be applied successfully. It has drawbacks and limitations, however, which are discussed. Improvements to this approach are proposed. The method itself is considered a suitable approach.

The paper also addresses the question to which extent a buoy can be detected at sea under realistic circumstances (with obstructed view and shaky camera footage). The results are promising, showing accurate detection and tracking of the buoy. The solution is able to re-detect the buoy after the view of it being obstructed by large waves. Section III-0b shows that the accuracy of the detection and tracking is reliable.

Maritime video processing problem poses challenges that are absent or less severe in other video processing applications. It needs unique solutions that address these challenges [9]. The algorithms and solutions that are described in this paper are able to successfully address several such problems.

TABLE III: Matlab files used for HOB algorithm

buoyX.mat
buoyY.mat
cameraParams.mat
evaluationAlgorithm.m
gtCoord.mat
handLabelling.m
main.m
predictedCoord.mat
ut_edge.m
ut_gauss.m
ut_levelx.m

APPENDIX A
FILES

APPENDIX B
MATLAB - EVALUATION

A. Matlab- Hand-Labelling Algorithm

```
%hand-label location of buoy
    [x, y] = getpts;

    % check if pinpointed point is out of ROI
    if x > 201 | x < 0 | y > 31 | y < 0
        % set 0 0 as coordinate vector
        v = [0 0];
    else
        % set buoy coordinates as vector
        v= [x y];
    end

    % add pixel coordinates vector to groundTruth cell array
    groundTruth{end+1} = v;
```

B. Matlab - Evaluation Algorithm

```
% array of hand-labelled ground-truth pixels-coordinates
load gtCoord.mat ground_truth

%array of corresponding detected pixel-coordinates
load predictedCoord.mat detected

%set threshold for overlap ratio
alpha = 0.5;

%Set parameters
TP=0; FP=0; FN=0; TN=0;
buoyVisible = true;

%% Algorithm
% calculate overlap ratio
for i = 1:length(ground_truth)
    % convert cell arrays to vector
    current_gt = cell2mat(ground_truth(i));
    current_dt = cell2mat(detected(i));

    % check if buoy is not visible
    if (current_gt(1) == 0) && (current_gt(2) == 0)
        buoyVisible = false;

        % check if nothing was detected in
        if (current_dt(1) == 0) && (current_dt(2) == 0)
            %if current_gt and current_dt == [0 0], then it is TN
            TN = TN + 1;
            continue;
        end
    else
        buoyVisible= true;
    end

    % create detection boxes around corresponding pixel coordinates
    gtBox =[current_gt(1),current_gt(2), 10,10];
    predBox =[current_dt(1),current_dt(2), 10,10];

    %calculate overlapRatio
    overlapRatio = bboxOverlapRatio(gtBox,predBox);

    % check if pixel coordinate pair = TP | FN | FP
    if overlapRatio >= alpha
        TP = TP +1;
    else
        if buoyVisible
            FN = FN + 1;
        else
            FP = FP + 1;
        end
    end
end

%% Calculate precision and recall
precision = TP/(TP+FP);
recall = TP/(TP+FN);
```

REFERENCES

[1] R. Gonzales and R. Woods, *Digital Image Processing*, . Prentice Hall, 2002.

[2] B. Zafarifar and H. Weda, *Horizon detection based on sky-color and edge features*, In Visual Communications and Image Processing (Vol. 6822, pp. 683-691). 2008.

[3] M. Sonka and V. Hlavac, *Image processing, analysis, and machine vision*, Cengage Learning. 2014.

[4] L. Ding and A. Goshtasby, *On the Canny edge detector*, Pattern recognition 34rd ed. 2001.

[5] P. Danielson, *Euclidean distance mapping*, Computer Graphics and image processing volume 4 3rd ed. 1980.

[6] Harris, C. & Stephens, M. A combined corner and edge detector. *In Proc. Of Fourth Alvey Vision Conference*. pp. 147-151 (1988)

[7] H. Rezatofighi, *Generalized intersection over union: A metric and a loss for bounding box regression.*, Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019.

[8] K. G. Derpanis, *The Harris Corner Detector*, York University. 2004.

[9] Prasad, D., Prasath, C., Rajan, D., Rachmawati, L., Rajabaly, E. & Quek, C. Challenges in video based object detection in maritime scenario using computer vision. (arXiv,2016), https://arxiv.org/abs/1608.01079

[10] Zitová, B. & Flusser, J. Image registration methods: a survey. *IMAGE AND VISION COMPUTING*. **21** pp. 977-1000 (2003)

[11] Lucas, B. & Kanade, T. An iterative image registration technique with an application to stereo vision. *In IJCAI81*. pp. 674-679 (1981)

[12] Prasad, D., Rajan, D., Rachmawati, L., Rajabally, E. & Quek, C. Video Processing From Electro-Optical Sensors for Object Detection and Tracking in a Maritime Environment: A Survey. *IEEE Transactions On Intelligent Transportation Systems*. **18**, 1993-2016 (2017)

[13] Fefilatyev, S., Smarodzinava, V., Hall, L. & Goldgof, D. Horizon Detection Using Machine Learning Techniques. *2006 5th International Conference On Machine Learning And Applications (ICMLA'06)*. pp. 17-21 (2006)

[14] Moritz, H. *Geodetic Reference System 1980*, by resolution of the XVII General Assembly of the IUGG in Canberra. (1980)

[15] S. Selmy and R. Woods, *The Need of Man Overboard (MOB) detecting and tracking system descriptive analyses*, . 2016.

[16] S. Case et al., *TFatal Falls Overboard in Commercial Fishing - United States 2000 -2016*, Mortal Wkly Rep. . 2018.

[17] S. Case et al., *Safety, Security, Health and Social Responsibility.*, Papathanassis, A. (eds) Cruise Business Development. Springer . 2016.

[18] S. Case et al., *How to PrePare For a Man oVerBoard*, Boat U.S. Magazine, Springer . 2012.

[19] R. Klein, *Safety, Security, Health and Social Responsibility.*, A. (eds) Cruise Business Development, Springer . 2016.