Omni-Directional Camera and Fuzzy Logic Path Planner for Autonomous Sailboat Navigation

Miguel Romero, Yan Guo, Sio-Hoi Ieng, Frédéric Plumet,
Ryad Benosman, and Bruno Gas

Institut des Systèmes Intelligents et Robotique, Université Pierre et Marie Curie (CNRS - UMR 7222)
4, place Jussieu 75252 Paris cedex 05, France
firstname.lastname@upmc.fr

Abstract. Autonomous surface vehicles are used to explore, measure and observe many kinds of environments which are often complex and arise many challenges. The ASAROME project [1] (Autonomous SAiling Robot for Oceanographic MEasurements) is focused on an autonomous sailboat to make measurements and observations in marine environments for extended periods. This paper describes a fuzzy inference engine which integrates a routing strategy for obstacle avoidance using an omnidirectional camera for obstacle detection system.

1 Introduction

Operating unmanned autonomous vehicle is a challenging task which is far more difficult than classic mobile autonomous robots because of the outdoor context of application. Most of the autonomous outdoor navigation works have been done for ground and air vehicles but unmanned surface vehicles (USV) have also gathered attention of robotics researchers like [1],[2],[3],[4],[5],[6] for autonomous sailboats. It is usual to equip ground or aerials vehicle with inertial and vision sensors but in the case of surface ones it is more marginal to our knowledge [7]. Active sensors like sonar, radar or even laser are more likely used in surface vehicle for obstacle detection [8].

The ASAROME project fits into this context of autonomous USV, aimed to the design and construction of an autonomous vessel able to carry out long term oceanographic measurement campaign. An accurate and efficient strategy for obstacle detection and navigation is fundamental to preserve the integrity of the ship. Multisensory approaches are usually the solution to the problem as it is shown in several past works [9],[10] [11]. In this project, several on-board sensors will be used for obstacle detection. We will focus on the omnidirectional camera system to build a vision based obstacle detector. Future work will encompass other embedded sensors. Our route determination fuzzy inference system is conceived to optimize the trajectory looking for a favorable wind propulsion through the entire route avoiding unnecessary tack maneuvers and obstacles detected by the panoramic camera.

The paper is organized in several sections. In section 2, we present the calibration method for a catadioptric camera and its use for obstacle detection. In section 3, we

[1] This work is funded under the project ANR ASAROME (Num. ANR-07-ROBO-0009)

© C. Delgado, C. Gutiérrez, R. Velazquez, H. Sossa (Eds.)
Advances in Computer Science and Electronic Systems
Research in Computing Science 52, 2011, pp. 335-346

Received 23/01/11
Accepted 20/02/11
Final version 07/03/11

introduce a fuzzy logic method for local routing and obstacle avoidance. Experimental and simulation results are presented in the last section of this paper.

2 Omnidirectional vision sensor

We are focusing on the vision sensor of the ASAROME embedded system to address the problems of obstacle detection and avoidance. Sensors data are used to feed the navigation command loops but they need firstly to be calibrated correctly, i.e., the mappings between measurements and the real world metric must be estimated.

2.1 Catadioptric sensor calibration

The omnidirectional sensor we use is a non-central catadioptric system, combining a perspective camera with a reflective surface. We have to estimate the relative poses of each component of the sensor.The perspective camera must be calibrated intrinsically using standard techniques described in literature; typically, we used [12] to achieve this task. With the knowledge of the intrinsic the matrix K, we estimate the relative pose of the camera and the mirror with a variation of the method described in [13], based on an homography between two judiciously chosen planes according to [14].

2.2 Boat pose estimation

A global coordinate frame has to be set at first to localize all sensors in relationship with the sailboat, which is defined by the axes of the ship, taken as an ellipse from the images. The boat is first segmented from the background and the covariance matrix of the pixels distribution is computed. Principals axes directions of the ellipse are then given by the matrix eigenvectors (see Fig. 1). Any structure or scene object will now be referenced within this frame for the rest of the paper.

With the sensor calibrated as described and placed on the the top of the mast, it is possible to compute each pixel's projection on the plane defined by the water under the assumption of a calm sea state, if the relative pose of this plane and the vision sensor (thus the boat) can be estimated. This operation is required if we want to extract metric information from the images without the need of stereovision. In the similar way, as we did for the camera pose with respect to the mirror estimation, we can estimate the omnidirectional sensor pose by detecting projections of the sea plane on the catadioptric image. To achieve this task we place into the water several easy-to-detect buoys that are acting as seamarks to underline the sea plane.

If the structure defined by the buoys is known, i.e., their relative distances of the buoys are known, then enough geometric constraints can be provided in addition to the coplanarity one for the plane estimation. A standard parametrization of a plane we can use is:

$$ax + by + cz + d = 0 \tag{1}$$

Given n buoys used to built the seamark, the distance between the i^{th} and the j^{th} buoys, D_{ij} is manually measured. There are $m = \binom{n}{2} = \frac{n!}{2!(n-2)!}$ of such distances and

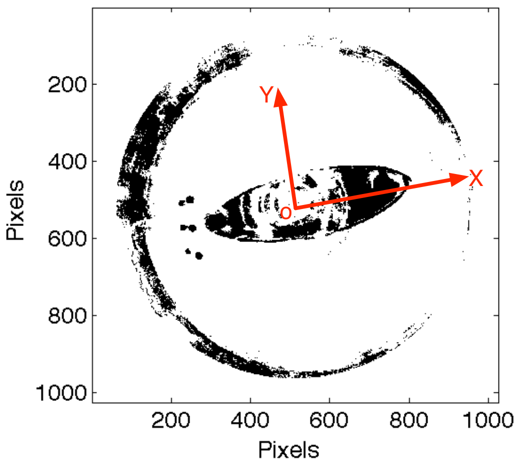


Fig. 1: The ship's axes are used to define a global coordinate frame assuming the camera is placed perpendicular to it.

m is higher than the number of parameters for $n \geq 4$ (in our case $n = 6$). The buoys projected on the camera are also reduced to their centroids $\mathbf{p}_i$ in the image plane and, with the sensor calibration, n 3D lines $(\mathbf{m}_i, \mathbf{u}_i)$ are computed for the n centroids. There is likely only one plane under the distances constraints that intersects a given bundle of rays which in our case is the set of all rays associated to the buoys. We use a Hough-based method for the plane decision. Where a, b, c are precomputed from -1 to 1 with a constant step of 0.002 and d is calculated from 1 to 500 with a constant step of 1. We define π_k, the k^{th} plane that intersects the ray bundle. Each ray intersects the plane and the distances between every two intersections are computed:

$$D_{ij}^{k} = |\mathbf{P}_i^k - \mathbf{P}_j^k| \tag{2}$$

Where $\mathbf{P}_i^k$ and $\mathbf{P}_j^k$ are the positions of intersections for the point i and j. If π_k is the plane that we are looking for, it will satisfy the condition $D_{ij} \approx D_{ij}^k$. Given the possibility to determine the camera pose with respect to the sea surface, we can map any pixel $\mathbf{p}_i$ to a 3D point $\mathbf{P}_i$ on the plane (Fig. 2). The distance of each pixel to the boat in the image can be computed as: $d = |\mathbf{p}_i|$ in the global coordinate frame introduced earlier. This latter result allows to built a resolution map of the sensor on the sea surface since we can project all the pixels on it.The plane pose estimation method is tested with synthetic data by randomly generating 3 planes defined by their four parameters. For each plane, 6 points are placed randomly on it, assuming that their relative distances with respect to each other point are known. For each of this setting, we execute the plane detection algorithm. This operation is repeated 100 times to provide enough results to produce meaningful statistics. The table 1 summarizes the results obtained. According to the result, the plane estimation algorithm accuracy is reasonable with estimates errors below 7% and a mean value of 3.33%.

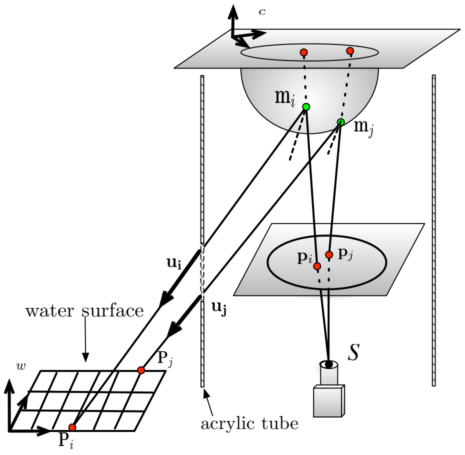


Fig. 2: The mapping of any image pixels for the catadioptric sensor and its intersection with the surface defined by the water.

Table 1: Result of estimated parameters

	Parameters estimation error rates(%)			
Parameters	a	b	c	d
Case 1	4	1	1	3.3
Case 2	1	0	0	0
Case 3	1	2	0	6.7

The resolution map is built with the mast normal to the water plane (calm sea state), however an oscillating sea is more likely expected, implying to recomputed the resolution map for different orientations of the ship with respect to the sea surface. To avoid to run constantly the costly construction, a set of maps is precomputed for several values of orientations one time. Each time the ship detects a significant change of orientation, the correct map is loaded into the computer. The maps are precomputed for angle α spanning from -90 to $90°$ with a constant step of $1.80°$.

2.3 Obstacle detection

A colorimetric criterion is used to segment objects in the image. To be able to detect an obstacle, a color signature of the sea is first computed by selecting samples of regions representing the water. Each pixel of these regions are reprojected to the three planes of the RGB coordinate frame according to their color components. This operation produces a cluster in each plane and is representative of the color distribution of the background (i.e sea).

To segment an object pixel from the background, we project it into the RGB components planes and it is considered being an obstacle if at least one of its RGB component

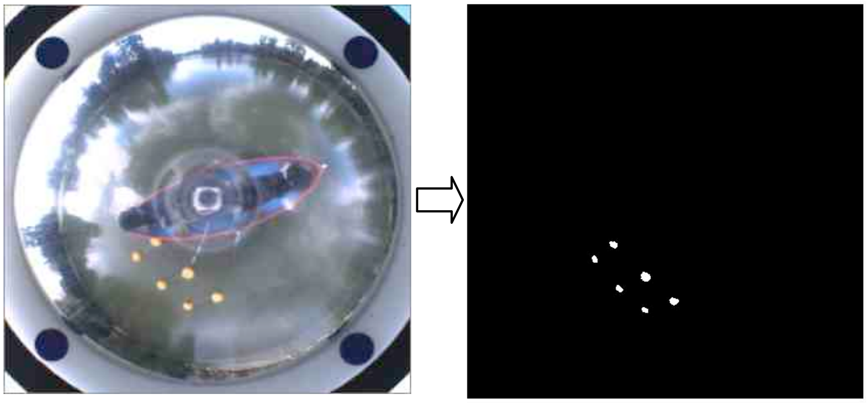


Fig. 3: Seamark detections using colorimetric signature.

does not belong to the clusters defined earlier. The Fig. 3 show an example of segmentation of a seamark formed by six yellow buoys using the combination of image difference and colorimetric signature of the water.

Detected obstacles are usually sets of pixels $\mathbf{P}^i_{obs}(x_i, y_i)$ from which the centroids $\mathbf{P}_c(x_c, y_c)$ are computed. We can also estimate roughly the size of a detected osbstacle as being the maximal distance between two pixels of the set:

$$l_{obs} = max(|\mathbf{P}^i_{obs} - \mathbf{P}^j_{obs}|) \quad \text{for } i \neq j. \tag{3}$$

With the obstacles properly detected and segmented, their positions and their respective size are parsed to the heading calculation.

3 Fuzzy Inference System for Local Routing Strategy and Obstacle Avoidance

In sailing, there are no precisely defined criterions to navigate. This situation gives to the *skeeper* the hard task to take decisions based in his own experience. Then, as Zadeh describe in [15], the problem of sailing is able to be resolved using fuzzy logic.

There are multiple approaches in literature to solve the routing problem for a sailboat using fuzzy logic as [16], [17], [18]. Also there are several papers that describes obstacle avoidance using fuzzy inference engines as [19], [20]. In this section we describe a mixed approach that uses a fuzzy inference engines to solve our sailboat navigation and obstacle avoidance problem.

For a sailboat, the propulsive force comes from the aerodynamics effects of the wind on the sails. This leads to the conventional points of sail diagram which describes a sailing boat's course with respect to the wind direction (Fig. 4). The white sectors

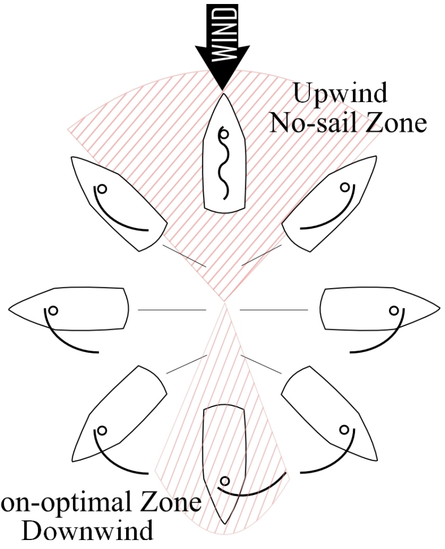


Fig. 4: Standard points of sail diagram

correspond to normal sailing zones and the two shaded zones are not sailable zones or no-go zones (in fact, the downwind zone could be sailed in theory but is not very efficient and rather unstable).

So, unlike conventional motorized robotic vehicles, where a straight line to the goal leads to a shortest path (in time and distance), in this case there is no such easy solution for a sailboat if the target is located directly upwind or downwind. In these cases, the sailboat has to beat (i.e., take a zig-zag course) to reach the goal.

The speed vector of a sailboat depends on many factors like the wind angle, wind speed, sail trimming, currents and waves. This behavior is usually represented for a given boat by a specific polar diagram (Fig. 5). This polar diagram shows the maximum boat speed along a given heading with respect to the wind. Each curve on the polar diagram corresponds to a given wind speed. As we can see on Fig. 5, the wind speed mainly modifies the amplitude without modifying the global shape of the polar diagram.

In the following, we suppose that this boat-specific polar diagram is known for a set of wind speeds.

The proposed method for local routing and obstacle avoidance is based on the calculation of an optimal heading that maximize the crisp output of the Mamdani type fuzzy inference system (FIS) . This FIS have as entries the speed to the objective and the obstacle influence value to tend to minimize the time to reach the goal and maximize the distance between the sailboat and the obstacles. This method will react to the changing of the environmental conditions (wind speed, wind angles,...) by periodically re-computing the optimal heading.

As a measure of the boat efficiency to reach the goal, we take the value of the boat speed vector $\mathbf{V}(h)$ for a given heading angle h, projected in the direction of the waypoint

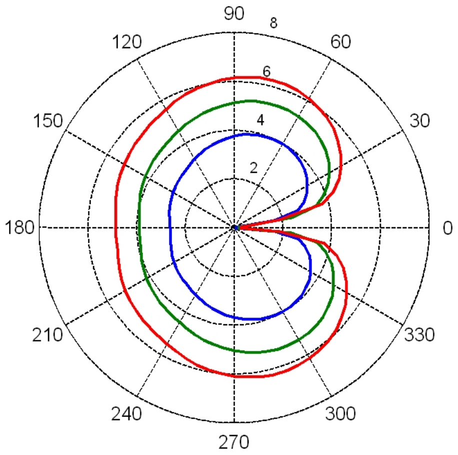


Fig. 5: Boat-specific polar diagram.

WP, that is:

$$V_G = \mathbf{V}(h)^T \cdot \mathbf{T_g} \tag{4}$$

where $\mathbf{T_g}$ is an unit vector pointing to the way point. In order to conveniently use this measure in our routing algorithm, the boat velocity is supposed to be normalized, i.e, the maximum value of the boat speed on a given polar diagram is supposed to be equal to 1, regardless of the wind speed value.

Steering through the eye of the wind, that is, into and across the flow of the wind, is usually a maneuver that must be avoided due to the speed decreasing and, in certain situations, the risks of rollover due to a wind shift. To take these two facts into consideration, we use a penalty factor η_w for the computation of the cost function to minimize the time to reach the goal:

$$C_W = \eta_w \left(V_G\right) \tag{5}$$

with η_w = 1 if the actual heading and the new computed heading are in the same side of the wind and η_w = 0.8 otherwise. Such a penalty factor leads to prioritize a new heading that keeps the course in the same side of the wind, rather than crossing the eye of the wind.

To take into account the obstacles detected by the perception system, we use a cost function based on the measured distance between the boat and an obstacle's centroid:

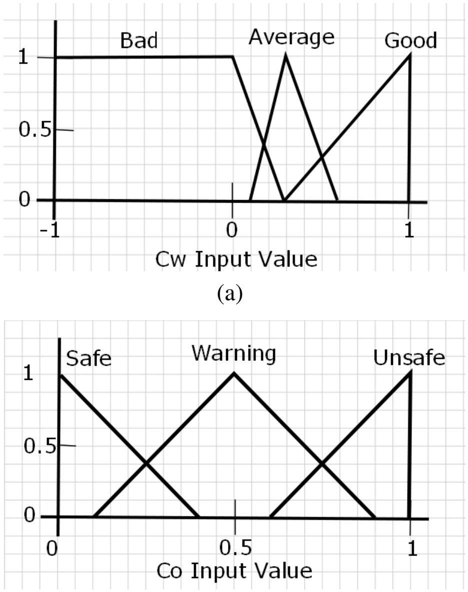
$$C_O = \begin{cases} \eta_o \left(\frac{1}{d_{obs}} - \frac{1}{d_0}\right) & if \;\; d_{obs} \leq d_0 \\ 0 & if \;\; d_{obs} > d_0 \end{cases} \tag{6}$$

with η_o as positive scaling factor, d_{obs} as the Euclidean distance from the center of the sailboat to the center of the obstacle and d_0 as the obstacle influence distance (50 m in our case, which is a trade-off between the range of our sensors and the maneuverability of the sailboat). The final cost function is:

$$C = C_W + \sum_{obs} C_0 \tag{7}$$

Input data for the FIS are the weighted velocity to goal C_w and the obstacle influence value, C_o, given by the equations 6 and 6. The fuzzy sets representing the linguistic variables are as in Fig6, where negative values of C_w represents the heading angles that stray from the goal .

In course control and obstacle avoidance the C_w and the C_o parameters are net to fuzzy controller, then, the acceptance value of the h heading angle is inferred through the fuzzy logic.



(a)

(b)

Fig. 6: Fuzzy system for heading selection

The output variable is the value of pertinence for the tested h heading angle. The fuzzy variable contains tree fuzzy sets that defines this pertinence as bad, average and good, as in Fig7.

The rule base of the heading selection FIS contains 9 rules of the form:

```
If desired direction IS x AND obstacle is y THEN
               pertinence is z
```

In order to avoid undesirable trajectory angles(Fig. 4) and to keep the course aiming the goal, we introduce three simple fuzzy rules:

1. *If(C_w is Good) then (Output is Ok);*
2. *If(C_w is Average) then (Output is Average);*
3. *If(C_w is Bad) then (Output is Bad);*

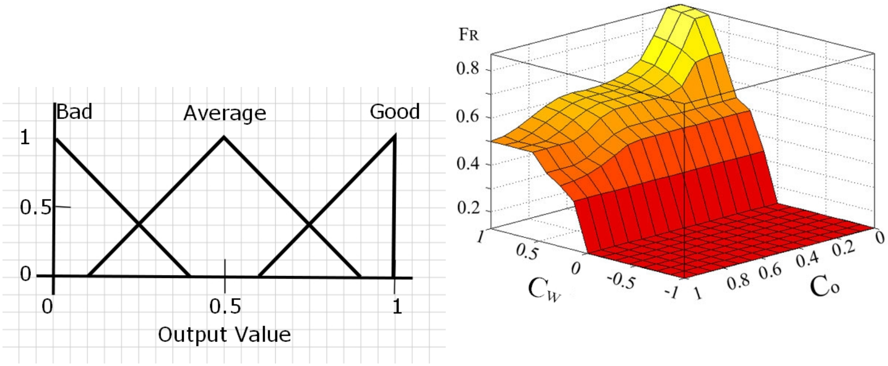
As we can observe in Fig.6a, the heading angle h that gives C_w values ≥ 0.2, which are classified with the linguistic variables *"Average"* or *"Good"*, are the only heading angles that will be considered, since they are the only ones that assure wind propulsion and a path towards the goal.

To keep the sailboat away from the obstacles we must to select a set of fuzzy rules based on the C_o value (Eq.7), then, as is shown in Fig.6b, lower is the value of C_o, safest is the proposed heading angle. From this statement we can infer the next rules:

4. *If(C_w is Good) and (C_o is Safe) then (Output is Ok);*
5. *If(C_w is Good) and (C_o is Warning) then (Output is Average);*
6. *If(C_w is Good) and (C_o is Unsafe) then (Output is Bad);*
7. *If(C_w is Average) and (C_o is Safe) then (Output is Average);*
8. *If(C_w is Average) and (C_o is Warning) then (Output is Average);*
9. *If(C_w is Average) and (C_o is Unsafe) then (Output is Bad).*

This set of nine fuzzy rules assures that the selected heading angle h will minimize Eq.7 taking the sailboat to the goal trough an obstacle free path (Figs. 7, 8).

This FIS(h) is computed periodically (0.5 sec) and the optimal heading angle maximizing this inference system is sent to the low level layer control. Choosing the angle of heading, h, which maximize the output of FIS(h) allows the sailboat to be able to navigate while keeping its course outside of the polar's forbidden area (Fig.4), taking the shortest navigable path and avoiding obstacles. Using this method, there is in fact no guarantee that the system will not be caught in a local minimum. However, since our autonomous sailboat is intended to move in open sea, the probability to have more than one obstacle at a time in the sensors range is actually very low.



(a) Output

(b) Fuzzy Surface

Fig. 7: Output fuzzy sets and surface

4 Experimentation

We demonstrate the performance of our algorithm with two different tests: obstacle detection and simulation for the trajectory planning. The first experimentation have been done on a lake, with all the data being recorded during the test. The second one is a simulation, which shows the resulting motion of the sailboat's with and without obstacles.

4.1 Real data analysis for obstacle's detection

We conducted this experimentation on a calm lake with its surface defining a plane. A special seamark with 6 buoys (1m long x 2m wide) is used for this test. Our camera tube is fixed on a 1.9m high tripod.

The seamark position is changed between each record and the distance between the camera and the center of the seamark was manually measured. The distance tested ranges from 2 meter to 14 meter. From the images which are taken by the camera panoramic, we can estimate the length and the width of seamark, also the distance between seamark and our boat.

From table 2, our algorithm seemed to have excellent performance with error rates within 8%. The distances estimated are smaller than the measured distances. The maximum value of estimated error is 0.63m for a distance of 14m, and the minimum is 0.19m for a distance of 8m. The estimated lengths are arround of 1 meter, and the estimated widths are arround of 2 meters. Like the estimated distance error, the maximum size estimated error is 7.40% for a distance of 14m, and the minimum is 2.32% for a distance of 4 m.

Table 2: Real data error estimation

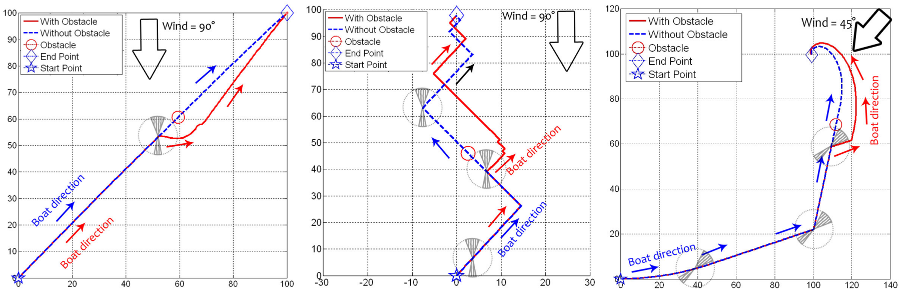
	Distance mesured(m)	Esti. dis.(m)	Esti. angle(°)	Esti. Dis. err. rate(%)	Esti. length(m)	Esti. width(m)	Esti. l+w err. rate(%)
Case 1	2	1.70	259	3.75	0.95	1.94	3.71
Case 2	4	3.72	201	3.5	0.83	2.10	2.32
Case 3	6	5.60	207	5	0.83	2.03	4.67
Case 4	8	7.81	199	2.38	0.91	1.95	5.12
Case 5	10	9.61	186	4.87	0.79	1.99	7.02
Case 6	12	11.44	208	7	0.8	2.09	3.47
Case 7	14	13.37	187	7.87	0.66	2.12	7.40

4.2 Simulation Results

Extensive simulations were conducted of sailboat traveling in different wind directions, with different goal positions and with or without obstacles.

Some representative simulations results are presented in this paper. Fig. 8(a) shows the sailboat's motion for a lateral wind. Figs. 8(b) and 8(c) shows two upwind navigation cases. Each test was conducted two times with the same conditions, except for the obstacle. For all simulation scenarios the wind speed was$(15\,Kt)$ and remains constant during the simulation time, the wind angle is (90°) for the first two cases and (45°) for the last one. The initial heading is 0°, the start position is at $(0,0)$ and the goal positions are $(100,100)$ for the cases depicted on figs. 8(a) and 8(c), and $(0,100)$ for the fig. 8(b) . Those conditions were chosen to show the behavior in a normal navigation situation and also, in cases (b) and (c), to force the sailboat to take an upwind trajectory.

We can see on Fig. 8 that, in all cases, the sailboat reaches the waypoint. In the first case (without obstacle), the sailboat will go straight to goal, in the second try (with obstacle) it turns with out crossing the eye of the wind, to avoid the obstacle. In the second case it have to tack several times to avoid the obstacle and reach the goal. Finally, in the third case, the sailboat tack one time, and then it keeps the wind on its right side.



(a) TWA = 90°　(b) TWA = 90°　(c) TWA = 45°

Fig. 8: Simulation results with different TWA and different goal position

5 Conclusion

The experimental results demonstrate the ability of panoramic vision system to obtain and process the data to give a precise position and size of the detected obstacles. The conducted simulations validate the routing algorithm because, with the data from omnidirectional camera system, it is possible to modify the motion of the sailboat to avoid the collisions, keep it on the correct wind's angle and reach the fixed waypoint. The current limitation of the presented routing method is the risk to fall in a local minimum. But, as outlined before, in open sea: the probability to have more than one obstacle at a time in the sensors range is actually very low.

The future work in a short term is aimed to perform several tests implying all the sensor for data fusion to validate the whole system in open sea.

References

1. R. Stelzer, K. Jafarmadar, H. Hassler, and R. Charwot, “A reactive approach to obstacle avoidance in autonomous sailing.” in *3rd International Robotic Sailing Conference (IRSC)*, 2010.
2. Y. Briere, “Between the drifting buoy and the autonomous sailing boat : the microtransat concept,” in *ATMA - ASSS 2008 international autonomous surface ship symposium*, France, 2008.
3. C. Sauze and M. Neal, “A raycast approach to collision avoidance in sailing robots,” in *Proceedings of the 3rd International Robotic Sailing Conference*, 2010.
4. G. H. Elkaim, “The atlantis project: A gps-guided wing-sailed autonomous catamaran,” *Journal of the Institute of Navigation*, vol. 53, No. 4, pp. 237–247, 2006.
5. J. Sliwka and L. Jaulin, “Autonomous robotic boat of ensieta.” International robotic sailing conference, july 2009, pp. 1–7.
6. H. Erckebs, G.-A. Büsser, C. Pradalier, and R. Siegwart, “Avalon, navigation strategy and trajectory following controller for an autonomous sailing vessel,” *IEEE Robotics and Automation Magazine*, vol. 17, No 1, pp. 45–54, 2010.
7. A. Subramanian, X. Gong, J. Riggins, D. Stilwell, and C. Wyatt, “Shoreline mapping using an omni-directional camera for autonomous surface vehicle applications,” in *OCEANS 2006*, 2006, pp. 1–6.
8. T. Bandyophadyay, L. Sarcione, and F. Hover, “A simple reactive obstacle avoidance algorithm and its application in singapore harbor.” in *International Conference on Field and Service Robotics*, 2009.
9. A. Bonci, G. Ippoliti, L. Jetto, T. Leo, and S. Longhi, “Methods and algorithms for sensor data fusion aimed at improving the autonomy of a mobile robot,” *in Advances in Control of Articulated and Mobile Robots*, vol. 10, pp. 192–222, 2004.
10. G. Ippoliti, L. Jetto, A. la Manna, and S. Longhi, “Improving the robustness properties of robot localization procedures with respect to environment features uncertainties,” *IEEE International Conference on Robotics and Automation*, pp. 1451 – 1458, 2005.
11. S. Giompapa, F. Gini, A. Farina, A. Graziano, R. Croci, and R. Distefano, “Maritime border control multisensor system,” in *Aerospace and Electronic Systems Magazine*, 2009, pp. 9–15.
12. J.-Y. Bouguet, *Camera Calibration Toolbox for Matlab*, http://www.vision.caltech.edu/bouguetj/calib_doc/index.html.
13. J. Fabrizio, P. Tarel, and R. Benosman, “Calibration of panoramic catadioptric sensors made easier,” in *Proceedings of the IEEE Workshop on Omnidirectionnal Vision*, june 2002.
14. P. Sturm, “Algorithms for plane based pose estimation,” in *Proceedings of the IEEE International conference on Computer Vision and Pattern Recognition*, 2000, pp. 706–711.
15. L. Zadeh, “Fuzzy sets,” *Department of Electrical Engineering and Electronics Research Laboratory, University of California*, vol. 8, pp. 338 – 353, 1965.
16. R. Stelzer and T. Pröll, “Fuzzy logic control system for autonomous sailboats,” *Robotics and Autonomous Systems*, 2007.
17. O. CALVO, “Fuzzy control of a sailboat,” *International Journal of Approximative Reasoning*, vol. 16, pp. 359–375, 1997.
18. E. Yeh and J.-C. Bin, “Fuzzy control for self-steering of a sailboat,” *Intelligent Control and Instrumentation*, vol. 2, pp. 1339 – 1344, 1992.
19. Y. Jincong, Z. Xiuping, and et.al, “Intelligent robot obstacle avoidance system based on fuzzy control,” *The 1st International Conference on Information Science and Engineering*, pp. 3812–3815, 2009.
20. R. Malhotra and A. Sarkar, “Development of a fuzzy logic based mobile robot for dynamic obstacle avoidance and goal acquisition in an unstructured environment,” *International Conference on Advanced Intelligent Mechatronics*, pp. 1198–1203, 2005.