A Path Planner for Unmanned Surface Vehicles: A Survey

Riccardo Polvara*

Developing a robust obstacle avoidance module is a foundamental step towards fully autonomous Unmanned Surface Vehicles. Until now, most of vehicles move in the sea following waypoints paths, usually GPS-based, totally unconcerned about possible collisions against rocks, other vessels and also human life. In this paper, the state of the art regarding obstacle avoidance for USVs and planning a safe path between a starting point and a goal is analyzed, synthetized and interpreted.

Key words: Path planner, Unmanned Surface Vehicle, control, obstacle avoidance.

1 Introduction

Marine robots represent one of the three categories in which mobile robotics can be divided, together with ground and aerial robots. This kind of vehicle can be also further distinguished in *Unmanned Surface Vehicles (USVs)* and *Unmanned Underwater vehicles (UUVs)*.

An increasing interest about USV has been expressed by the military community, expecially for those situations such as force protection, surveillance, mine warfare and so on. Multiple

*Riccardo Polvara is a PhD student with the Marine Science and Engineering School, University of Plymouth, Plymouth, England. E-mail: riccardo.polvara@plymouth.ac.uk

platforms were developed and deployed in the late 1990s (Bertram 2008); between these, two examples are given by the *Owl MK II* (Fig.1d), characterized by a low-profile hull for increased stealth and payload capability, and equipped with a sonar and a video camera, and the *Spartan USV* in (Fig.1b) developed by the US Space and Naval Warfare System Center in San Diego since 2003. Multiple unmanned marine vehicles have been built also outside the USA: in Japan, for example, Yamaha developed the Unmanned Marine Vehicle High-Speed UMV-H and the Unmanned marine Vehicle Ocean type UMV-O, involved in bio-geo-chemical monitoring. Other examples are the canadian *Barracuda* (Fig.1c), the Dolphin MK II, the Seal USV and the SARPAL AMV, all developed by the International Submarine Engineering Ltd (ISE),the Stingray used by the Israeli navy, the Delfim and Caravela developed by the Portuguese Dynamical Systems and Ocean Robotics laboratory (Alves et al. 2006), and finally the *Springer* (Fig.1a) developed by the University of Plymouth (Naeem and Sutton 2009) .

Most of the vessel cited before are dual-purpose vehicles, meaning that they can be driven by humans, on-board or remotely, but also in an unmanned way. In this way their capabilities are augmented and extended in an affordable and low-risk manner.

To navigate in a fully autonomous way the presence of an *obstacle avoidance module* is required to move the unmanned vessel from the actual track to another one if an immediate collision is expected, and then take it back on the previous one towards the goal pose (position and orientation). As it usually happens with ground robots, a path planner should be implemented: often it is distinguinshed in a *global path planner* (GPP) and *local path planner* (LPP). The goal of GPP is to find a safe path connecting the starting position, the actual pose of the robot, and the final one, called *goal pose*. Sometimes in the literature it is also called a *deliberative path planner*. Otherwise, the LPP has to react to immediate collision against unexpected obstacles, maybe not considered by the GPP, moving the autonomous vehicles far from the preplanned path in order to avoid the moving obstacle; for this reason it is also called *reactive path planner*.

(a) The Springer

(b) The Spartan

(c) The Barracuda

(d) The Owl

Figure 1: Example of different USVs. In Fig.1a is represented the Springer, developed by Plymouth University for marine observation purposes. On the right, in Fig.1b the Spartan, a multimission reconfigurable USV built by SSC-San Diego. On the second row, on the left the canadian Barracuda (Fig.1c) while on the right, in Fig.1d the Owl MKII used for port security.

The structure of the paper is divided as follows: in Section 2 it will be illustrated how to perceive the environment surronding the autonomous vessel and detect static and moving obstacles with the most used computer vision techniques; in Section 3 it will be discussed more in details the necessity of having a robust path planner, therefore in Subsections 3.1 and 3.2 it will be illustrated how a global and a local path planners, respectively, could be implemented. Finally, in Section 4 a new combined path planner based on A* and Obstacle Velocity will be presented.

2 Obstacle Detection

To perfectly avoid obstacles moving across the path of autonomous vessels, an highly accurated world model is required. In order to obtain it, different sensors can be combined and data coming from them are usually fused in a 2D or 3D representation.

In (Almeida et al. 2009) the authors suggest to use an ARP radar sensor to identify moving obstacles and shores, and classify targets in terms of collision threat. They identify a set of perimeters, shown in Fig.2, around the USV in order to decide appropriate measures: *irrilevant* perimeter(3km), *safe* perimeter(500m), *warning* perimeter(250m) and *prohibition* one(50m). Based on the *closest point of approach* (CPA), defined as the estimated minimum distance between the detected object and the USV along its path, they classify targets as *no threat* (CPA outside irrilevant perimeter), *low threat* (CPA crosses the irrilevenat perimeter but not the safe one), *potential threat* (CPA crosses the safe perimeter but not the prohibited one) and *dangerous* (CPA inside the prohibition perimeter).

Low-cost radars are also used in the work of Schuster, Blaich, and Reuter (2014), in which an on-board collision avoidance module is required to compensate the lack of an automatic identification system (AIS).

In order to detect objects, data coming from radars need to be image preprocessing in the

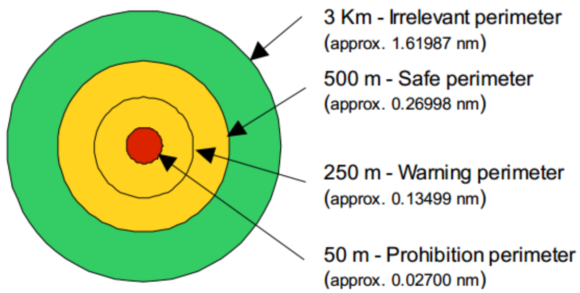


Figure 2: Multiple safety perimeters are drawn in the proximity of the USV. If the CPA of an obstacle is outside the irrelevant one, the obstacle is considered as *no threat*; if the CPA is between the irrelevant and the safe perimeter, the obstacle is considered *low threat*; if the CPA is between the safe and the warning perimeter, the obstacle is considered a *potential threat*, while is considered *dangerous* if it is inside the prohibition perimeter.

following way:

- Ego Motion Compensation: the vessel's yaw rate is compensated calculating the azimuth of each scan in respect to the vessel's heading;
- Occupancy Likelihood Determination: assuming that the probability p of a cell being reported as occupied is independent and constant, the occupancy likelihood is binomially distributed;
- Connected Component Labeling: adjacent occupied cells (whose occupacy probability is greater than 0.5) are grouped using connected component labeling (Gonzalez and Woods 2001) to create an elliptical target.

Because the extracted target positions are very noisy, a low pass filter is adopted to get the object's true position, heading and velocity. To this scope, an *Interactive Multiple Model* (IMM) filter is chosen: it runs several models in parallel and, based on the estimate of each model and the current measurement, a likelihood for each model to reflect the true motion state is

determined. The output of the filter is a weighted sum of all model estimates and it is used by the collisions avoidance algorithm to predict the movement of the other vessels.
The experiments conducted by the authors demonstrated a fast and robust technique for tracking obstacles on the sea surface, even if the accuracy in the position and velocity can be improved.

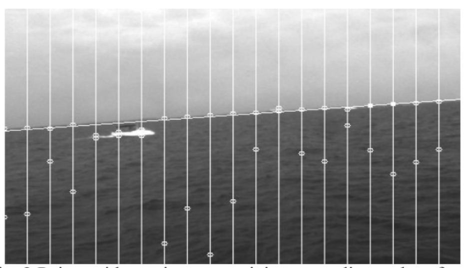
Other approaches use monocular and stereo vision methods for recording the presence of obstacles in proximity (30 to 100 m.) of the vessel. An example is offered by the work of Wang, Wei, et al. (2011) and Wang and Wei (2012), that uses two cameras mounted parallel on a metal bar: the image from the camera on the left is initially used to perform monocular obstacle detection, and then the stero approach is applied to the image from both cameras to compute the 3D detection results.
The monocular technique is composed as follows:

- Horizon Detection Module: it allows to distinguish the sea surface; it is realized using pixel profile analysis and Random sample consensus (RANSAC) (Fischler and Bolles 1981) method to perform line fitting (Fig. 3a) and extract the horizon (Fig. 3b);

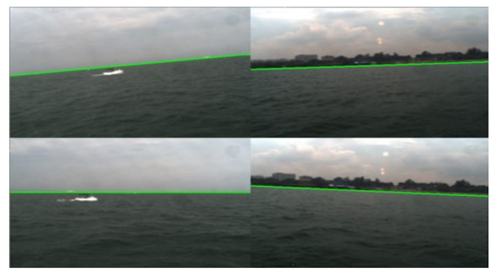
- Saliency Detection Module: as expressed in (Achanta et al. 2009), a binary mask is built and the detected salients will be given in the form of bounding boxes and considered of potential interest (Fig. 3c);

- Harris Corner Extraction and Tracking Module: using the work proposed in (Harris and Stephens 1988) and (Bouguet 1999), surface obstacles are distinguished among the potential identified previously (Fig. 3d);

- Obstacle Detection Module: the final results (Fig.4) are obtained combining data coming from previous steps; to verify the validity of a potential object as an obstacle, a tracked feature with long lifespan in the salient bounding box is labelled as of high priority and this link the obstacles in consecutive frames.

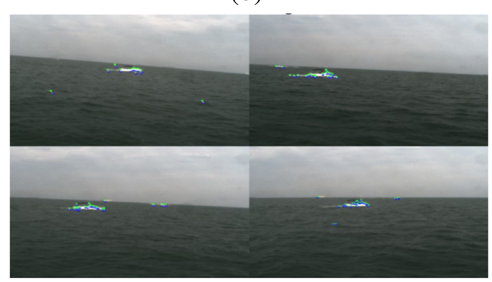


(a)



(b)

(c)



(d)

Figure 3: Monocular vision algorithm developed by Wang, Wei, et al. (2011): in Fig.3a the line separating sea and sky is fitted with the RANSAC method and then, in Fig.3b, the horizon is extracted. Obstacles on the sea surface are identified with saliency detection, Fig.3c, and their corner are extracted and tracked, as shown in Fig.3d

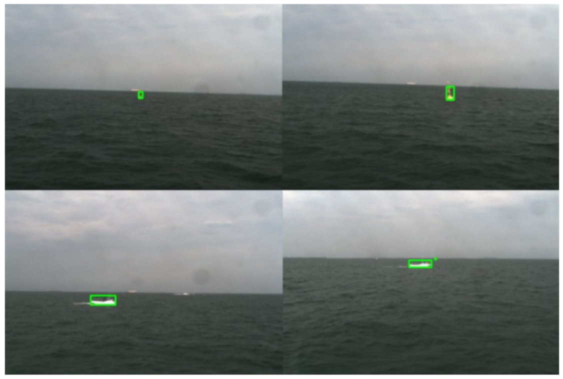


Figure 4: Results of the monocular obstacle detection approach in two different scenarios: in both cases, the obstacle is tracked among successive frames.

The stereo correspondence phase could be divided in three: an initial phase in which both cameras are calibrated and the results are used for 3D reconstruction, an intermediate phase in which *epipolar constraint* reduces the 2D search for obstacle correspondence, and then a *stereo matching* phase in which the normalized cross correlation template matching method is adopted. Here, the bounding box of obstacles detetcted by the monocular technique is searched for along its epipolar line in the right image. In the end, a Kalman filter is applied on the horizontal disparity in order to eliminate the stereo matching error and improve the range estimation accuracy. This approach has shown good results until 100 m. far from the USV, despite low-resolution (640 x 480) images were used during the experiments. Therefore it is rational to believe that with more advanced cameras is it possible to improve the performance also for longer distance, even if increasing the image resolution means increasing the computation cost too.

A work similar to the previous one but that use only monular vision is the one proposed by Azzaby *et al.* (Azzabi 2014). As before, the authors detect the horizon first and then the obstacles in the scene; last, they estimate the distance between the USV and objects, knowing the relative angle of each object and the horizion line. The algorithm developed for horizon detection can be summarized as follows:

(a) Original image (b) Greyscale image (c) Sobel transformation

Figure 5: In (Azzabi 2014) the obstacles edges are identied firstly grey-transforming the monocular image (Fig.5b) and then applying the Sobel operator (Fig.5c)

- Grey Level Transformation: the acquired image is transformed in grey scale to reduce image time processing (despite of a little loss of information) (Fig.5b);
- Sobel Operator: applied to the preprocessed image to identify those regions with high spatial frequency, mostly corresponding to edges (Maini and Aggarwal 2009) (Fig.5c);
- Hough Transform: the edge detections is performed (Hough 1962) knowing that two pixels in the image domain must lie on the intersection of two lines in Hough domain;
- Line Election: the horizon line is considered as the line that, given a couple (*a,b*), passes through the maximum number of points.

After this, objects in motion are detected using *optical flow estimation*: given an input stream, the velocity of each pixel is firstly calculated; if a pixel has a velocity higher than a threshold value and is under the horizon, it is marked as moving and tracked and in the end the optical flow vector of each pixel is plotted in the frame.

The output of the previous module is therefore used to estimate the distance from a USV to obstacles using a geometrical interpretation of the image (Fig.6).
The contribution of this work is given by the use of greyscale image, and techniques that could be applied on it, to reduce the computational effort required by other algorithm such as (Ettinger

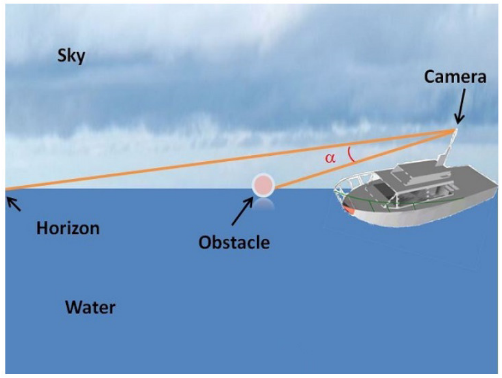


Figure 6: A trigonometric estimation of the distance from the USV to the obstacles is given by the distance to the horizon coupled the relative angle between this and the obstacle.

et al. 2003) adopted in the work of Larson, Bruch, Halterman, et al. (2007).

3 Path Planner

After having perceived the surrounding environment and the obstacles present in it with the techiques previously described,a model of the world should be realised. This can be done in a two or three dimensional space but often the first option is preferred because it can guarantee less computational time to operate with it. Having a virtual representation of the world is a key point to plan a path for moving the autonomous vessel.
As described in Section 1, the Path Planner module is usually divided in two sub-components: the global path planner aims to find a path from the actual pose of the robot to a goal one, while the local path planner tries to avoid moving obstacles close to the robot.

In the following subsections it will be described the most recent path planners used in marine robotics, to guide autonomous vessels on the sea surface among other marine crafts and moving hazards.

3.1 Global Path Planner

The goal of the GPP is to continuosly modify the existing waypoint route to plan around obstacles detected with the long-range sensors. In the work of Larson, Bruch, Ebken, et al. (2007) and Larson, Bruch, Halterman, et al. (2007), the path planners use a two-dimensional (2D) obstacle map created by dividing the environment into a discrete grid and assigning each cell location a value representing the probability of being occupied or not by an obstacle. This is filled with stationary obstacles from the *chart server* and moving obstacles provided by the radar. The underlying search technique is the A* search algorithm (Hart, Nilsson, and Raphael 1968), chosen because it can find an optimal solution in a short amount of time. Since A* uses a cost analysis at each step, Larson *et al.* inserted an added cost for proximity to obstacles for allowing a USV to set a safety barrier around obstacles.

To avoid moving obstacles, the path planner determines safe velocity ranges using the *Velocity Obstacles* (VO) method (Fig.8): a velocity space v-θ grid (where v denotes the USV speed and θ is the heading angle) is constructed as decision space to find the best velocity vector and the moving obstacle is expanded by the robot size. The reason to do this it to treat the robot as a point. As long as the robot's velocity lies outside the VO, it will not collide with obstacle, assuming that the velocity vectors are constant over time; if the velocity obstacle change over time, the VO-based approach reacts by replanning using the latest sesnsor information.

In the case that changing the velocity does not avoid collision, the path planner changes path by creating a *projected obstacle area* (POA) (Fig.7), for each obstacles and determining a safe alternative route using a A* search. A POA is the area a moving obstacle could occupy in the future. This area is identified calculating the CPA: since a moving obstacle can pose a threat to the USV along multiple stretches of the path, it is necessary to calculate the CPA of every obstacle along each path segment.

Among the previous concepts, the authors decided also to consider the International regula-

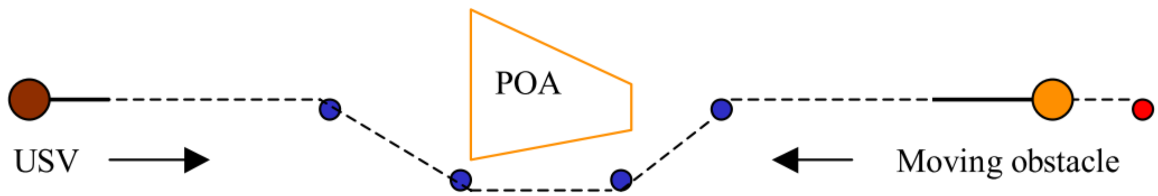


Figure 7: The USV has to avoid the obstacle coming ahead passing port-to-port its projected obstacle area.

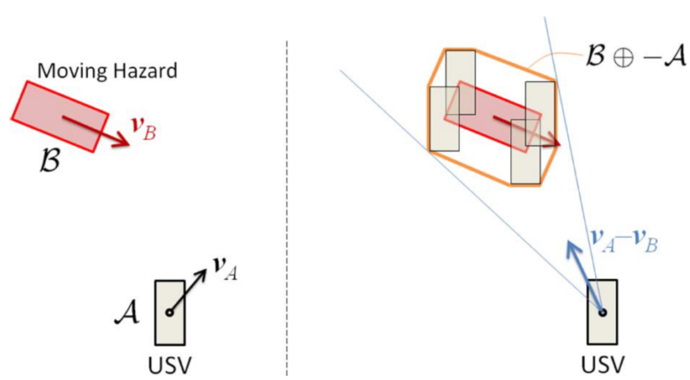


Figure 8: To avoid the collision, the relative velocity $\vec{v_A}$ - $\vec{v}_B$ has not to be inside the cone formed by the robot center and the expanded obstacle $A \oplus B$.

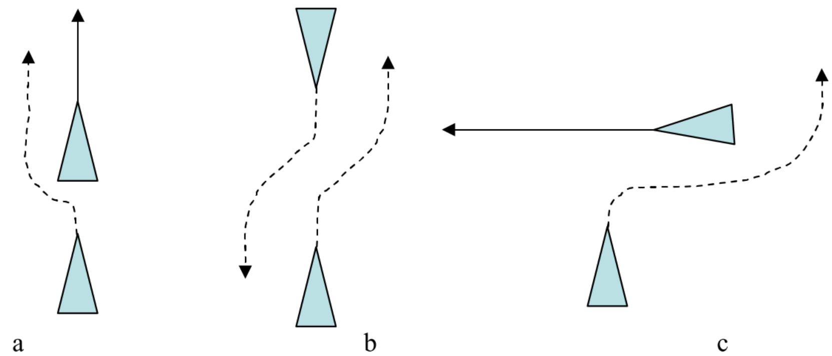


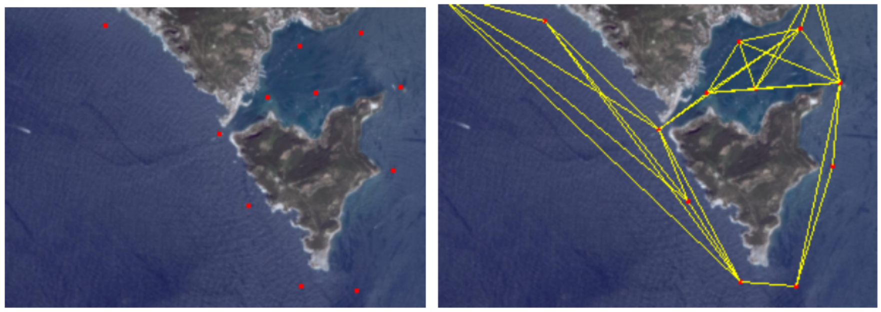
Figure 9: Rulsed defined to avoid collisions for overtaking (a), meeting (b), and crossing (c) obstacle.

tions for preventing collisions at sea (known as COLREGS, for COLlision REGulationS). In this work the authors consider three primary COLREGs, shown in Fig.9: crossing, overtaking and head-on situations. In the situation in which a traffic boat is crossing from the right, the vessel with the other on its starboard (right) side must give away. In the case of a USV overtaking a slow traffic boat, it must ensure enough clearance so that it keeps out of the way of the traffic boat being overtaken. Otherwise, if the USV and the traffic boat are moving straight toward each other, both vessels must alter their course toward the starboard, so that they pass with the other vessel to its port(left side).

The POA of a moving obstacle is calculated from the current path of the USV and the time taken to traverse that path. As the path changes, there is a need to update the POA and recalculate.

Casalino, Turetta, and Simetti (2009) suggest an approach based on the *Visibility Graph* (Fig.10) concept and on a world-model in which obstacles are polygons and the robot is a point. A visibility graph is a graph of intervisible locations, typically for a set of points and obstacles in the Euclidean plane. Each node in the graph represents a point location, and each edge represents a visible connection between them. That is, if the line segment connecting two locations does not pass through any obstacle, an edge is drawn between them in the graph.

In order to use the Visibility Graph the obstacles inside the working area had to be transformed



(a) (b)

Figure 10: A visibility graph is built identifying first a list of vertices (a) and then connect those nodes visible one from each other.

into polygons. At this point the Dijkstra's Algorithm (Dijkstra 1959) is applied between the starting point and the goal one and the resulting trajectory will not intersect any of the obstacles.

A totally different approach has been developed by Xie et al. (2014). The authors take inspiration from the concept of *Artificial Potential Field* (APF) (Khatib 1985) and improved it to be more robust such that it can avoid local minima, destinations unreachable and poor accuracy. APF consists in combining the repulsion potential field of obstacles and gravitational potential field of targets in the operational space.

In the APF model,shown in Fig.11, T represents the target which produces the attraction to the USV, O represents obstacles which produce repulsion to the USV. If X_d represents the target position, the control of the USV with respect to the obstacle O can be carried out in the artificial potential:

$$U_{art}(x) = U_{att}(x) + U_{rep}(x) \tag{1}$$

where U_{art}, U_{att}, U_{rep} represent artificial potential field, attraction potential field and repulsion potential energy respectively.

The improvement introduced consists in a regulatory factor that, in the presence of an obstacle,

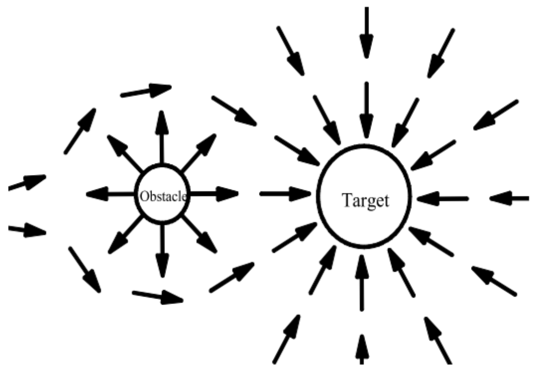


Figure 11: The model of APF

controls attraction for decreasing as a linear factor and repulsion as a higher-order function. In this way, the USV will not encounter the case of local minum or destination unreachable, but it demonstrated in simulation to be able to avoid obstacles smoothly and reach the goal.

Chen et al. (2013) adopt in their work a Micron DST sonar as the obstacle avoidance sensor. They propose a method in which the scanning angle can adapt to the distance from the sonar to the obstacle. The fuzzy logic algorithms are used to make the strategy timely and effective. Their approach is described as follows:

- Data Collection and Pretreatment: interesting byte are extracted from the sonar packages to reduce computational complexity;
- Noise Reduction: noise caused by ambient affection is reduced by a threshold filter;
- Obstacle Avoidance Algorithm Design: the scan range is set to four levels; if no obstacle is detected in a cycle, the scan range will jump to the upper level, otherwise the scan range will down to the nearest level which is bigger than the distance from obstacle to the sonar head. At this point, the type of the obstacle line is determined and the slope is calculated and sent to the next module;
- Decision-making Module: in order to make the route timely and effective, fuzzy technology

is used in motor control;

This work shows good results and the model used during experiments always avoided obstacles and shores. The only limitation encountered by the authors was given by the low precision of inertial navigation system used which compromise the accuracy of the reaction.

3.2 Local Path Planner

A first example of LPP is given by the work of Kuwata *et al.* (Kuwata et al. 2014), in which the authors suggest an algorithm to not only address hazard avoidance for stationary and moving hazards, but that also applies the COLREGS. In this work, as in (Larson, Bruch, Ebken, et al. 2007), the authors consider three primary COLREGS: crossing, overtaking and head-on situations.

Moreover, the proposed approach use the Velocity Obstacle (VO) concept: in this way a moving obstacle is transformed into a stationary one by considering the relative velocity and trajectory of the USV wih respect to the obstacle, and in the end it returns a set of USV velocity vectors guaranteeing collision avoidance.

The developed algorithm works in this way:

- Precollision Check: the closest point of approach (CPA) is computed with the current position and velocity of the USV and traffic vessels, and it is evaluated if any COLREGS rules need to be applied;
- Rule Selection: if the CPA meets temporal and distance conditions, the best COLREGS rule is applied analyzing a set of geometric constraints;
- Hysteresis: introduced to lower the rate at which the USV can change what rules to apply; this means that once a COLREGS maneuver is initiated, it continues to direct the boat for at least a minimun duration of time;

- Cost: once the constraints set of VO and COLREGS are generated, a defined cost for each v_i and θ_j ammissible is generated and the (v_i, θ_j) pair with the minimum cost is selected and the velocity command is sent to the vehicle controller.

A similar approach has been implemented by Leng *et al.*. In (Leng, Liu, and Xu 2013) the Velocity Obstacle approach is integrated with Mixed Linear Integer Programming (MILP). In solving the path planning problem, the dynamics and kinematics of the USV, sensors and uncertainty constraints of the environment all can be taken into consideration and linearized. In the end, the objective function will be

$$\text{min: } |\ L_{UG} - (v_{UG} + \Delta\ v_{UG})\Delta \text{t}\ |$$

where L_{UG} and v_{UG} denote the relative distance and velocity between the USV and the target point.

As in (Kuwata et al. 2014), the collision is checked calculating the the CPA and its distance from the vessel. After this, six types of encounter situations are identified and the USV reacts depending on this decision.

Also the authors of (Larson, Bruch, Ebken, et al. 2007; Larson, Bruch, Halterman, et al. 2007) describe a local or *reactive* path planner module; in their work, in addition to the global path planner described in Subsection 3.1, a local one is required because the long-range sensors are not capable of detecting small low-profile obstacles such as very small personal boats, or because the USV may inadvertently deviate from the planned path if GPS is jammed or the Inertial Navigation Unit (INU) drifts.

In the adopted approach, all the near-field sensors are fused into a common level world-model, and individual behaviors vote on specific navigation solutions within that model. A number of arcs are projected in front of the vehicle over the local world-model obstacle map. The number of arcs considered is a function of the map size and grid spacing, with the arcs spaced such that

one arc pases through each of the outer cells.
This approach guarantees that each cell in the grid is covered by at least one arc so that all the navigable path are considered. Each arc is given a weight or vote based on the distance the robot could travel along the arc before it encountered an obstacle. The votes are scalded from 0 to -1 so that they can be combined with votes from oher navigation behaviors.

In (Casalino, Turetta, and Simetti 2009) a new reactive path planner based on the *bounding box* concept is described. A bounding box of a track is easily defined as the rectangle that defines the area the USV should avoid in order to guarantee a safety distance from the moving object. The algorithm proposed suggests to integrate the USV actual position S and the local goal G to a graph made of the four edges of the box, by adding the edges starting from either S or G and ending on one of the vertexes of the box that are not intersecting any edge. Then the solution would be any path from S to G, which can be obtained by any graph taversing algorithm, such as A* or Dijkstra's algorithm. The problem with this approach is its sub-optimality because it does not take into account any kinematic property of the tracks nor the USV.
The algorithm implemented to reach the locally optimal avoidance is composed as follows, using an A* search:

- For each visible vertex of the bounding box, compute the angle θ such that the vessel can intercept it;
- Calculate time(cost) $\bar{t}$ and position $\bar{P}$ of intercept;
- Calculate estimated time to goal $h \equiv G - \bar{P} / v_1$, where v_1 is vehicle's speed;
- Compute global estimate cost $f = \bar{t} + h$.

At each iteration the node with the lowest f is selected and popped from the *openset*, the set of all the nodes that have been created but not yet explored in the search. It's therefore checked whether the goal can be reached without collision: if so end the search. Otherwise, for each obstacle, calculate the vertexes intercept positions and check if this path is collision free with *ray tracing* techinque. If so, add this node to the openset and proceed back to the first step.

In the successive work (Simetti et al. 2014) the authors present a refinement of this algorithm. To avoid crossing the bounding box and thus failing the obstacle avoidance, they introduce a *safety bounding box* around the original collision bounding box. All the computations are now performed against it, and its vertexes are used to performed the avoidance. If for same reasons the USV enters the safety box, it must exit it without crossing the main diagonals, ensuring that the vehicle moves away from the collision bounding box.

To increase the likelihood that the path is still collision free with the safety bounding box even under estimation uncertainty, a *supporting bounding box* is adopted, whose dimensiond depends on the position of the USV:

- if the USV is inside the safety bounding box, the supporting one coincides with the safety one;
- if the USV is far away with the safety bounding box, the supporting one coincided with a maximum bounding box;
- if the USV is in-between the maximum bounding box and the safety one, the supporting one varies with the distance, shrinking as the USV is closer to the safety one.

In this way, when the USV if far away from the incoming vessel, the computed path will be very robust to changes in speed and heading of the obstacle and its safety bounding box.

A different approach based on lane-constrained trajectory generation is proposed in (A. Tan, Wee, and T. J. Tan 2010). Here, while avoiding obstacles, the vessel has to meet some objectives. Firstly, it is required to maintain a minimum distance from each obstacles at all times (*safety distance* objective). Secondly, the USV must observe the COLREGs rules. The last two

objectives are called *cross track* and *shortest time* objectives: they means the USV should keep as close as possible to the intended path and complete it the the shortest path possible.
The idea presented is divided in two step:

- Maneuver Generation: the platform's motion is forward simulated for a fixed number of time steps using simple models of the maneuvers tracker and the boat;

- Maneuver Selection: is a multi-stage process in which objectives, divided in *rules* and *criteria*, are ordered based on fixed priorities. At each stage, a single objective is considered and candidate are eliminated based on that particular objective; as long as more than one candidate emrges from the elimination, the remaining candidates are subjected to the next stage of elimination.

A limitation of this approach is given by the generation step: because only a sample of possible maneuvers is taken, the algorithm may sometimes be unable to find a solution.

In (Blaich et al. 2015) it's proposed a specialized A* algorithm that allows velocity variations and considers different turning circles for different velocities. First of all, a three-dimensional occupancy grid is built by inserting the measurementd of the laser finder which are transformed to a local reference frame. Then the map is processed and contours of obstacles are extracted.
In parallel to the mapping procedure for static obstacles, a *Multi Object Tracket* (MOT) is implemented for moving objects.
For the evasive path generation, an A* algorithm is specialised such that it considers static obstacles, tracked agents and the mission path. A penalty depending on the total lenght of the path that had to be skipped is added to the cost function of the A* algorithm. This results in evasive path that lead back to the mission path after avoiding an obstacle. However, to guarantee the feasibility of an evasive path, the kinematic constraints of the USV have to be considered during the path search. To use the full dynamic capability of the vessel, it is necessary to allow

changing velocities in the evasive path along with the respective changes of the minimum turning circle. These alternations in velocity are enabled by adding both the velocity and the time to the search space, that consist now of four dimensions - two for position and one dimension for each time and velocity.

Tang *et al.* in (Tang et al. 2012) shows a new method called *Obstalce Avoidance Algorithm Based on Heading Window* (OAABHW) for the near-field static obstacle avoidance of USVs. The algorithm transforms dynamic windows into heading window and translational velocity window by using D̈ivide and ConquerS̈trategy, and grasp an optimized avoidance angle from solving the constraint optimization problem of heading window at second step. Then the rotational velocity could be determined by the navigation angle and heading of USV. The translational velocity of avoidance is produced by the translational velocity model which is formed by the rotational velocity and the distribution of obstacles in the surroundings.

Based on the concept that the translational velocity is less important than the heading angle, the authors build the relative coordinate of USV and upload all the obstacles to it. Then the constraint set of heading angle of USV can be calculated by dealing the obstacle in near-field on tangent method. Obstacles are replaced by their circumcircle, and the geometry size of the USV is regarded as the expanded factor of obstacle expanding, meanwhile USV can be treated as a particle.

In the near-field of USV, a number of virtual rays projected around the USV with a certain angular resolution, any angle obstructed by obstacles is defined as infeasible heading angle of USV. Upon the analysi of obstalces in the relative coordinate of USV by tangent method, the maximum angle and minum angle could be grasped and recored as θ_{obs_max} and θ_{obs_min}. Then these two angles are transferred from relative coordinate to absolute ones.

In the process of excellent heading angle selection, the degree of heading yaw is treated as optimization objective. The heading window and the infeasible set of heading angle are treated as constraints. By solving the single optimization problem, the best navigation angle can be grasped. The constrain set optimzation problem can be mathematically formulated as:

$$max : F_{Head}(\theta) = 1 - \frac{\theta_{goal} - \theta}{2\pi}$$

where θ_{goal} is the basic value of optimal angle selection which is formed by global target point and the current position of USV.

The excellent avoidance navigation angle θ_{out} is obtained and the avoidance rotational velocity ω_{out} could be calculated by it and the current heading angle θ_{USV}:

$$\omega_{out} = \omega_c + \frac{2(\theta_{out} - \theta_{USV} - \omega_c * \Delta t)}{\Delta t}$$

where ω_c is the current rotational velocity of USV, and Δt is timewindow.

The translational velocity strongly depends on the rotational one and on the obstacles distributions. If obstacles are close to USV or the avoidance rotational velocity ω_{out} is comparatively large, then it has to slow the translational velocity to pass the hazard areas. If obstacles are far away from USV and rotational velocity is small, then USV should navigate at high speed.

Because in real marine environments the trajectory of the vessels is influenced by many factors such as sea wind, currents and waves, Zhang *et al.* in (Zhang et al. 2014) illustrate a new adaprive obstalce avoidance algorithm based on Sarsa on-policy reinforcement learning (AABSRL).

It is composed of two modules: the local obstacle avoidance module (LOAM) focuses on obstacle avoidance in the environmente and ignores external disturbance factors, while the disturbances from sea wind and currents are dealt with in the adaptive learning module (ALM). The responsability of ALM is to search for a course compensation angle to offset the course deviation angle that is generated by external disturbance factors. Its core is composed by the Sarsa on-policy reinforcement learning-based adaptive learning algorithm, which is used to explore the course compensation angle of USV. The input data of ALM include the state data of

USV, the state data of sea wind and currents, and the guidance angle from LOAM. This one is realised using the OAABHW algorithm presented in (Tang et al. 2012) and previously described in this paper.

Under the disturbances of sea wind and currents, the course angle deviates from the guidance angle. To keep the safe navigation of USV, the ALM of AABSRL needs to search for a certain course compensation angle $\Delta\theta_H$ to offset the course deviation. The Boltzamann distribution-based selection strategy is used to select the action of course compensation angle in the learning process, because it is typically greedy in the limit and infinite exploration (GLIE). Moreover, under thiese assumptions it can converge to the optimal action strategy with probability 1.

4 Conclusion

References

Achanta, R. et al. (2009). “Frequency-tuned salient region detection”. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ic, pp. 1597–1604.

Almeida, C. et al. (2009). “Radar Based Collision detection developments on USV ROAZ II”. In: *Oceans09 Bremen*, pp. 1–6.

Alves, J. et al. (2006). “Vehicle and Mission Control of the DELFIM Autonomous Surface Craft”. In: *Control and Automation, 2006. MED ’06. 14th Mediterranean Conference on*, pp. 1–6.

Azzabi, T. (2014). “Obstacle detection for Unmanned Surface Vehicle”. In: 5, pp. 1–7.

Bertram, V. (2008). “Unmanned Surface Vehicles : A Survey”. In: *Skibsteknisk Selskab, Copenhagen, Denmark*, pp. 1–14.

Blaich, M. et al. (2015). “Mission Integrated Collision Avoidance for USVs using Laser Ranger”. In: *Oceans 2015 Mts/Ieee*, pp. –5.

Bouguet, J.-y. (1999). “Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm”. In: *In Practice* 1.2, pp. 1–9. arXiv: 3629719.

Casalino, G., A. Turetta, and E. Simetti (2009). “A three-layered architecture for real time path planning and obstacle avoidance for surveillance USVs operating in harbour fields”. In: *Oceans 2009-Europe*, pp. 1–8.

Chen, J. et al. (2013). “An Obstacle Avoidance Algorithm Designed for USV Based on Single Beam Sonar and Fuzzy Control”. In: December, pp. 2446–2451.

Dijkstra, E. W. (1959). “A note on two problems in connexion with graphs.” In: *Numerische Mathematik* 1, pp. 269–271.

Ettinger, S. et al. (2003). “Towards flight autonomy: Vision-based horizon detection for micro air vehicles”. In: *Florida Conference on* 7.17, pp. 617–640.

Fischler, M. a. and R. C. Bolles (1981). “Random Sample Consensus: A Paradigm for Model Fitting with Applicatlons to Image Analysis and Automated Cartography”. In: *Communications of the ACM* 24.6, pp. 381–395.

Gonzalez, R. C. and R. E. Woods (2001). *Digital Image Processing*. 2nd. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Harris, C. and M. Stephens (1988). “A Combined Corner and Edge Detector”. In: *Procedings of the Alvey Vision Conference 1988*, pp. 147–151.

Hart, P., N. Nilsson, and B. Raphael (1968). “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”. In: *Systems Science and Cybernetics, IEEE Transactions on* 4.2, pp. 100–107.

Hough, P. (1962). *METHOD AND MEANS FOR RECOGNIZING COMPLEX PATTERNS*.

Khatib, O. (1985). “Real-time obstacle avoidance for manipulators and mobile robots”. In: *Proceedings. 1985 IEEE International Conference on Robotics and Automation* 2, pp. 500–505.

Kuwata, Y. et al. (2014). “Safe Maritime Autonomous Navigation With COLREGS, Using Velocity Obstacles”. In: *IEEE Journal of Oceanic Engineering* 39.1, pp. 110–119.

Larson, J., M. Bruch, J. Ebken, et al. (2007). “Autonomous Navigation and Obstacle avoidance for unmanned surface vehicles”. In: pp. 17–20.

Larson, J., M. Bruch, R. Halterman, et al. (2007). “Advances in Autonomous Obstacle Avoidance for Unmanned Surface Vehicles”. In: *Techniques*, pp. 1–15.

Leng, J., J. Liu, and H. Xu (2013). “Online path planning based on MILP for unmanned surface vehicles”. In: *Oceans - San Diego, 2013*, pp. 1–7.

Maini, R. and H. Aggarwal (2009). “Study and comparison of various image edge detection techniques”. In: *International Journal of Image Processing . . .* 147002.3, pp. 1–12.

Naeem, W. and R. Sutton (2009). “An intelligent integrated navigation and control solution for an unmanned surface craft”. In: *IET Irish Signals and Systems Conference (ISSC 2009)*, pp. 9–9.

Schuster, M., M. Blaich, and J. Reuter (2014). “Collision Avoidance for Vessels using a Low-Cost Radar Sensor”. In: 2009, pp. 9673–9678.

Simetti, E. et al. (2014). “Experimental Results on Obstacle Avoidance for High Speed Unmanned Surface Vehicles”. In: pp. –5.

Tan, A., W. C. Wee, and T. J. Tan (2010). “Criteria and rule based obstacle avoidance for USVs”. In: *2010 International WaterSide Security Conference*, pp. 1–6.

Tang, P. et al. (2012). “Research on near-field obstacle avoidance for unmanned surface vehicle based on heading window”. In: *Proceedings of the 2012 24th Chinese Control and Decision Conference, CCDC 2012*, pp. 1262–1267.

Wang, H. and Z. Wei (2012). “Improvement in Real-time Obstacle Detection System for USV”. In: 2012.December, pp. 5–7.

Wang, H., Z. Wei, et al. (2011). “A vision-based obstacle detection system for unmanned surface vehicle”. In: *IEEE Conference on Robotics, Automation and Mechatronics, RAM - Proceedings*, pp. 364–369.
Xie, S. et al. (2014). “The Obstacle Avoidance Planning of USV Based on Improved Artificial Potential Field”. In: 12140500400, pp. 746–751.
Zhang, R. et al. (2014). “An Adaptive Obstacle Avoidance Algorithm for Unmanned Surface Vehicle in Complicated Marine Environments”. In: *IEEE/CAA Journal of Automatica Sinica* 1.4, pp. 385–396.