Autonomous Navigation System for Unmanned Surface Vehicles

Yuanchang Liu, University College London, London/UK, yuanchang.liu.10@ucl.ac.uk
Rui Song, University College London, London/UK, r.song.11@ucl.ac.uk
Wenwen Liu, University College London, London/UK, w.liu.11@ucl.ac.uk
Richard Bucknall, University College London, London/UK, r.bucknall@ucl.ac.uk

Abstract

This paper presents work on the development of a real-time autonomous navigation system for Unmanned Surface Vehicles (USVs). The navigation system being developed is using an embedded hosting platform consisting of navigational data fusion processes together with algorithms used for path planning and collision avoidance when the USV is operating alone or in cooperation. An improved A path planning algorithm based on rasterized map is developed for single USV operation; whereas the fast marching square algorithm is implemented for multiple USVs. Both algorithms have been tested using a practical simulation environment. The resulting trajectories are guaranteed to be the shortest collision-free path.*

1. Introduction

In recent years, due to the benefit of reducing human causalities as well as increasing mission efficiencies, there has been an increasing deployment of USVs in both military and civilian applications. However, currently available USV platforms have low payload capacities and short endurance times. In order to overcome these constraints, the future trend of USV operation would be to deploy them as a formation fleet to allow cooperative operations. The benefits of USV formation operation include wide mission area, improved system robustness and increased fault-tolerant resilience

Efficient and intelligent path planning algorithms are core of the autonomous system to ensure the safety for both single USV and multi-USV formations. Currently, the two approaches evolving for path planning are stochastic and deterministic. For the stochastic approach, *Smierzchalski (1999)* and *Cheekuang (2010)* used a genetic algorithm to search for an optimized collision free path for the USV. *Ming-Cheng (2010)* implemented the 'ant colony' algorithm to design a decision-making system, which can assist vessels to navigate. For the deterministic algorithm, *Wasif (2013)* designed a COLREGs[1] compliant path planner using an A* algorithm. *Yanzhuo (2013)* improved the Artificial Potential Field method to provide a search safe collision free path in congested environments. *CheeKuang (2013)* proposed a cooperative path planning algorithm based on navigator manoeuvre experiences as well as collision avoidance regulations.

It is notable that deterministic algorithm appears to becoming the dominant solution for maritime navigation because of its better algorithmic consistency and completeness. The same resulting navigation path can be assured when running the deterministic algorithm as long as the planning environment does not change. However it is important to improve the confidence level of the navigation path achieved before its wide adoption.

Despite the advantages of deterministic algorithm, it also suffers some shortcomings. One of them is the lack of practicability of the path generated. The trajectory usually contains a number of turns, which are not practical in real-time maritime navigation where the path with least turns is nearly always preferred. Also, a path planned by a deterministic algorithm is unlikely to be smooth, especially when using A* algorithm, which makes it hard to be tracked by the vessel's autopilot.

Therefore, to overcome these issues, the deterministic algorithm with improved practicability property

[1] COLREGs: International Regulations for Preventing Collisions at Sea

is proposed in this work. For the single USV, the A* algorithm is chosen because of its efficiency. The 4-geometry cell connection style is used to improve the performance of the traditional A* algorithm. To make it more adaptable for USV navigation, two novel path smoothers are developed. The first smoother is able to reduce unwanted 'jags' produced by the conventional A* algorithm and the second one uses cubic spline interpolation methodology to make the path continuous. For the USV formation path planning, the fast marching square (FMS) algorithm is applied. Also, to better address the dynamic environment where moving vessels are involved, the constrained FMS algorithm is proposed to model the characteristics of moving obstacles.

The navigational devices used by current USVs are mostly cameras (*Weihong (2009), Terzakis (2013)* and *Tall (2010)*) and Global Positioning System (GPS), *Alberto Romano et al. (2012), R Sutton et al. (2011).* All of these sensors can only provide navigational information of own ship (OS[2]) as well as waypoints but are not able to generate reliable information for target ships (TS[3]). TS are normally located by using marine radar or Automatic Identification System (AIS). Hence, in this paper a data acquisition and fusion system based on Kalman filter technique is developed to merge data from sensors such as GPS, Inertial Measurement Unit (IMU) and AIS to provide real-time information with increased accuracy.

This paper outlines the proposed USV navigation system and describes each subsystem in detail in Section 2. Section 3 illustrates the methods employed to develop the navigation system. It includes the hardware development of data acquisition systems combined with single path planning algorithm. The concepts and structures of the formation path planning algorithms are also explained. The simulation results for different environments are demonstrated and evaluated in Section 4. Section 5 concludes the paper with discussions.

2. Navigation system structure

Presently, the navigation of a USV is achieved either through remote control approach or by onboard PC based navigator. The remote control approach generates navigation information on land and then transmits heading and speed to the USV via wireless communication. Such a strategy makes USVs highly reliant on robust communication channels which are thus prone to malfunction if the channels suffer severe data delay or there is loss of data. In contrast, the onboard PC approach is able to generate guidance information in real-time, which enormously increases the robustness of USV. However, most PCs are powered by a battery, which usually have low endurance times when the PC is engaged in high computing processing work. Thus, it is more reliable to equip USVs with high performance embedded systems, which can capably deal with complex tasks but with low energy consumption.

The proposed embedded USV navigation system is shown in Fig. 1, which consists of three sub-systems in terms of the modularity: 1) embedded decision-making system, 2) single USV path planning system and 3) USV formation path planning system

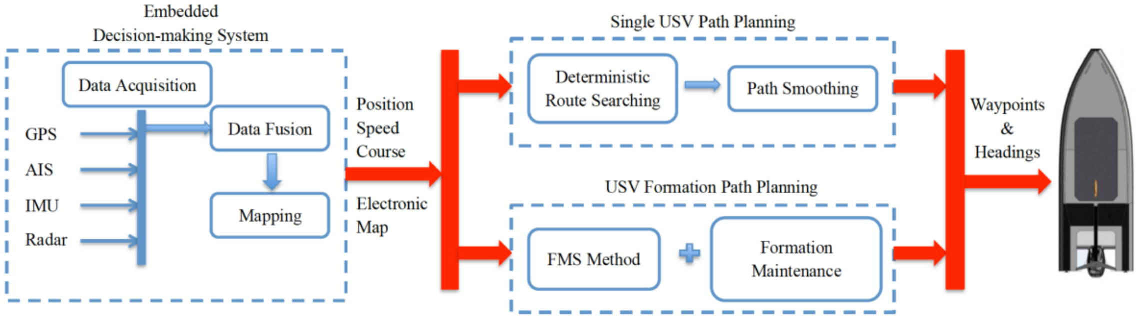


Fig.1: USV system diagram

[2] OS: In this paper, ship in direct control is referred as own ship.
[3] TS: Any other ship besides OS is referred to as target ship

The embedded decision-making system module is designed to fuse data that is acquired from on board sensors, such as GPS, AIS, IMU and Radar. It has three processes: data acquisition, data fusion and mapping. Global and local data from sensors is collected first, and then the Kalman Filter is applied for data fusion. The electronic environment map is generated according to the fused data. It contains navigational information such as position, speed and course of the OS and TS overlaid on a maritime chart. The synthetic electronic map is the benchmark map for the path planning system.

The path planning module is an embedded system that has two sub-modules, i.e. single USV path planning and USV formation path planning. The system will select the algorithm based on the task requirement. The path planning algorithm generates waypoints and accordingly headings, which are subsequently tracked by the USV's autopilot system. The autopilot system can be refereed to several works, such as *SK Sharma (2012)*.

3. Methods

3.1. Data acquisition and fusion

3.1.1. Extended Kalman Filter

Kalman filter is a popular technique applied to navigation algorithms as an optimal estimator for linear systems. However, most practical system processes are non-linear and therefore need to be linearised before they are estimated and this may be done by using of an extended Kalman filter (EKF). Its simplicity and robustness for practical implementation make the EKF suitable for this work. The theory behind it is briefly described below.
Assume the system state model is a non-linear differential equation:

$$\mathrm{x(k)} = \mathrm{f(x(k\text{-}1), u(k), w(k\text{-}1))} \tag{3.1}$$

with a measurement:

$$z(k) = h(x(k), v(k)) \tag{3.2}$$

where $f(x, w)$ is the system dynamic model and $h(x, v)$ is the measurement model. $w(k)$ is the process noise and $v(k)$ is the measurement noise. They are both assumed to be independent white noise with normal probability distribution with zero mean and variances (Q, R): $p(w) \sim N\ (0, Q)$ and $p(v) \sim N\ (0, R)$.

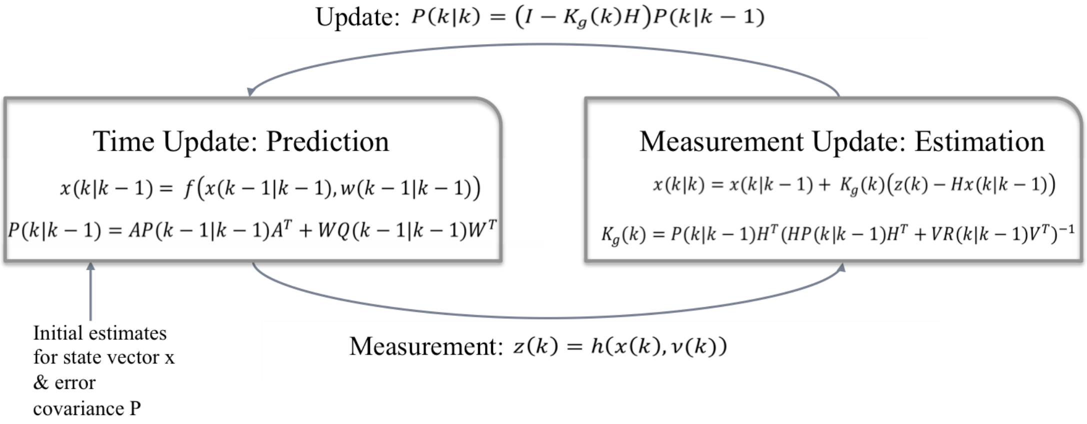

Update: $P(k|k) = (I - K_g(k)H)P(k|k-1)$
Time Update: Prediction
$x(k|k-1) = f\big(x(k-1|k-1), w(k-1|k-1)\big)$
$P(k|k-1) = AP(k-1|k-1)A^T + WQ(k-1|k-1)W^T$
Measurement Update: Estimation
$x(k|k) = x(k|k-1) + K_g(k)\big(z(k) - Hx(k|k-1)\big)$
$K_g(k) = P(k|k-1)H^T(HP(k|k-1)H^T + VR(k|k-1)V^T)^{-1}$
Initial estimates for state vector x & error covariance P
Measurement: $z(k) = h(x(k), v(k))$

Fig.2: Extended Kalman Filter (EKF) Process

The EKF involves two updates: measurement update and time update (Fig. 2). With the initial estimates for state vector x and its error covariance p, the first predicted state of system can be calculated by the system state model, where $p(k)$ is the system covariance matrix. The system will then estimate the optimal state by applying the measurement. This is the measurement update. After the estimation, the system will update its covariance matrix and enter the next state to make a new prediction, which is called the time update. This prediction-estimation process iterates the system and reduces the system error covariance to obtain the optimal state.

3.1.3 Kalman Filter Implementation

- State vector

USV's position, speed and course required by path planning algorithm are defined in the state vector as follows:

$$\mathrm{x} = [\mathrm{p}\ \mathrm{v}\ \theta\ \mathrm{b_a}\ \ \mathrm{b_g}]^{\mathrm{T}} \tag{3.3}$$

where p is the two axis coordinates of the USV position, v is the velocity and θ is the course of USV, b_a and b_g are the unpredictable biases of accelerometer and gyroscope integrated in the IMU.

- System Dynamic Model

In this work, the acceleration and angular velocity measured by the IMU is used to predict the next state. Due to instrument and environment limitation, the accelerometer and gyroscope suffer from various errors and the bias drift and scale factor errors are affected most. Therefore the following models are used to define the acceleration and angular velocity measured by the IMU.

$$\text{The acceleration model: } a_o = a_i + b_a + w_a \tag{3.4}$$
$$\text{The angular velocity model: } \omega_o = \omega_i + b_g + w_g \tag{3.5}$$

where a_i and ω_i are the inputs of the sensors (actual motion of the USV), a_o and ω_o are the outputs (motion measured by sensors), b_a and b_g are the biases of the accelerometer and gyroscope, w_a and w_g are the scale factor noise, which are assumed to be white with normal probability distribution $p(w) \sim N\ (0, Q)$, *Nebot et al. (1999).*

Then after applying Newton's Law, we can define the system dynamic model of USV navigation as follows:

$$\begin{pmatrix} \dot{p} \\ \dot{v} \\ \dot{\theta} \\ \dot{b_a} \\ \dot{b_g} \end{pmatrix} = \begin{pmatrix} v \\ C_b^n a_o - b_a - w_a \\ C_b^n \omega_o - b_g - w_g \\ W_a \\ W_g \end{pmatrix} \tag{3.6}$$

where C_b^n is the direction cosine matrix (DCM) from the body frame to navigation frame. It is expressed in terms of Euler angle yaw (y) as follows:

$$C_b^n = \begin{bmatrix} cosy & -siny \\ siny & cosy \end{bmatrix}. \tag{3.7}$$

- System measurement model

In the meantime, the GPS receiver provides position measurement of the system while magnetometer provides heading measurement of the system.

$$z(k) = \ p(k) + \theta(k) + b_m + v_m + v_{gps} \tag{3.8}$$

where b_m is the bias of magnetometer, ν_m and ν_{gps} are magnetometer and GPS measurement noises, which are assumed to be white noise and has normal probability distribution $p(\nu_{gps}{\sim}N\ (0,R))$.

3.2. Initial single path planning algorithms for hardware testing

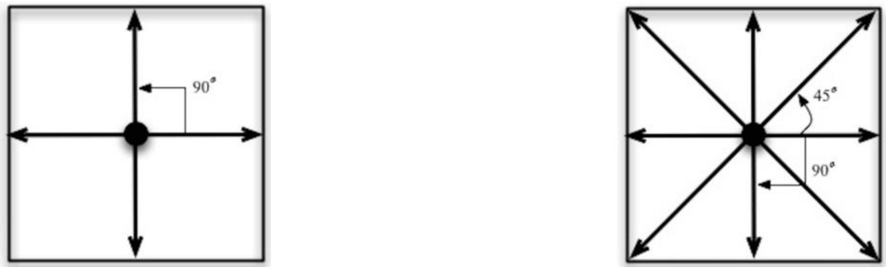
3.2.1. Assumptions

To simplify the algorithms implementation and make them more intuitive, some adoptable assumptions are predefined:

- USV is demonstrated as a mass point.
- The velocity at start and end point of the path are assumed to be zero.
- The algorithms use 4-geometry cell connection to represents 8 movement orientations of the USV on grid map.
- Euclidean distance calculation is applied because it is physically represents the smallest possible distance. Thus, the generated route can be assured as optimal and admissible.

3.2.2. 4-geometry A* algorithms

The A* algorithm is applied over a grid map to heuristically search for the shortest path with minimum distance cost. In the grid map, 2-geometry cell connection is mostly used to represent path turnings. In Fig. 3(a) turning degree of 2-geometry (4 neighbours) is $90°$. However, it is impractical to retain the same turning angle $(90°)$ in heading change manoeuvre. Hence, in this work, a 4-geometry connection is used to improve path property. As shown in Fig. 3(b), besides 90 degree turning, 45 degree turning is added as a new option.



(a) 2-geometry (b) 4-geometry

Fig.3: Grid cell connection styles

3.2.3. Path smoothing and interpolation

Route generated by the A* algorithm consists of a number of short line segments, which connect waypoints. There are two principle shortcomings associated with the route. First, since the main focus of A* algorithm is to minimise distance cost of the path, the jags generated are not usually optimised. Second, the path obtained is not continuous, which makes it hard to be tracked by the USV's autopilot in real applications. Therefore, to overcome these disadvantages, an advanced path smoothing algorithm has been developed.

Within the path smoothing algorithm, two path smoothers are developed. One is called path smooth method and another is the interpolation method. Fig. 4 shows the flowchart of the whole path smoothing algorithm and each smoother will be explained in later sections.

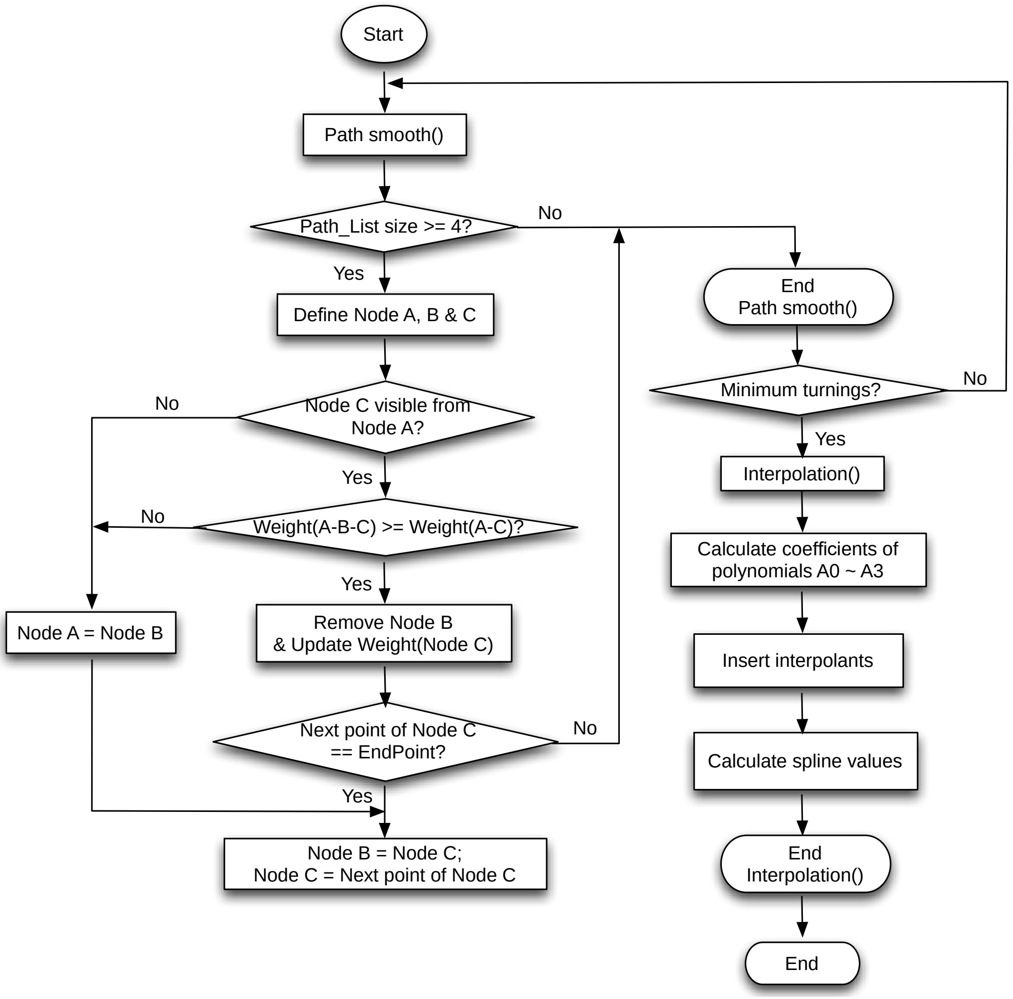
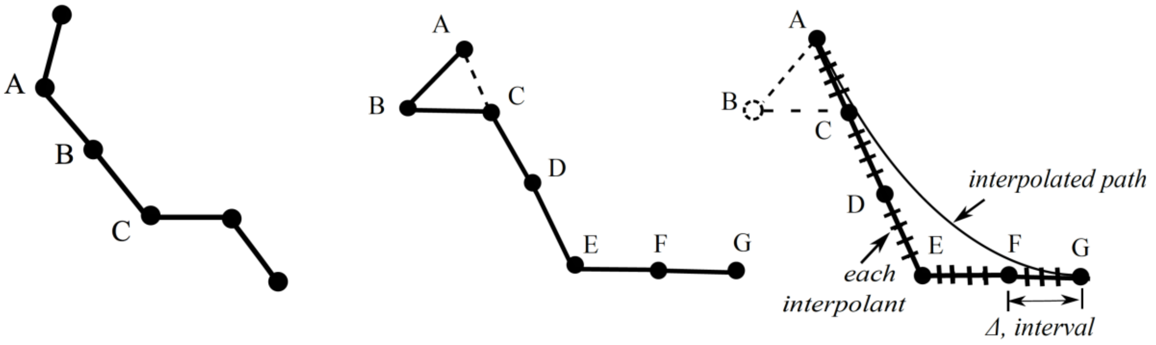


Fig.4: Path smoothing algorithm flow chart

- Path smooth method

The path smooth method is developed based on line-of-sight theory. It is applied to reduce the number of turning points after A* route searching procedure. The precondition to use the path smooth method is that there must be at least four path points in the path list. The method starts by checking the number of path points generated from A* algorithm. Then it iterates from the first path point to the last. For each iteration, the path smooth method checks each node in a group of three: 1) Current checking node, denoted as node A; 2) its next via point in the path, denoted as node B; and 3) next via point of node B, denoted as node C.

Fig. 5 illustrates how the method works. In Fig. 5(a), the nodes A, B and C are on the same line, therefore the path will be kept the same. However, if the original path is A-B-C-D and node C is directly visible to node A as shown in Fig. 5(b), then the node B will be removed from the path. The new path becomes A-C-D as shown in Fig. 5(c). The number of path points in the path list is reduced after each loop. The path smooth method keeps running until the number of turnings cannot be reduced and the obtained path is optimised in terms of least turnings. However, the path is still connected with rigid lines. The second path smoother interpolation method is required to make it continuous.



(a) (b) (c)

Fig.5: Path smoothing and interpolation

- Interpolation method

With the purposes to further improve the continuity of the path and achieve the final optimal output, a cubic spline interpolation is used. Among a number of interpolation methods, cubic spline is employed in this research due to its feature of least deviation from the original path. It is a process of approximation to depict parametric curve from the discontinuous path, shown in Fig. 5(c).

The main idea of the cubic spline is to solve the polynomials of the equation (3.9) (*George (1988)*) according to the control points. The control points are defined as the waypoints in the path list obtained after path smooth method.

$$y_k = f_k(x) \; = \; A_3(X - X_k)^3 + A_2(X - X_k)^2 + A_1(X - X_k) + A_0 \,, \qquad (3.9)$$

where k is the index of the control points. By given the x and y values of each control point, the polynomials can be calculated using the following constraints:
1) Cubic polynomials match the values of the equation at both ends of the interval (between two control points) $[X_k,\; X_{k+1}]$.
2) The first (velocity) and second (acceleration) derivatives of the interval are also continuous.
3) The second derivatives at start and end points of the path are zero as has been assumed.

The final optimised path can be achieved by inserting numbers of interpolants of *X* values between the start and end points in the path. The respected spline values can be calculated by solving the cubic polynomial equation.

3.3. USV formation path planning

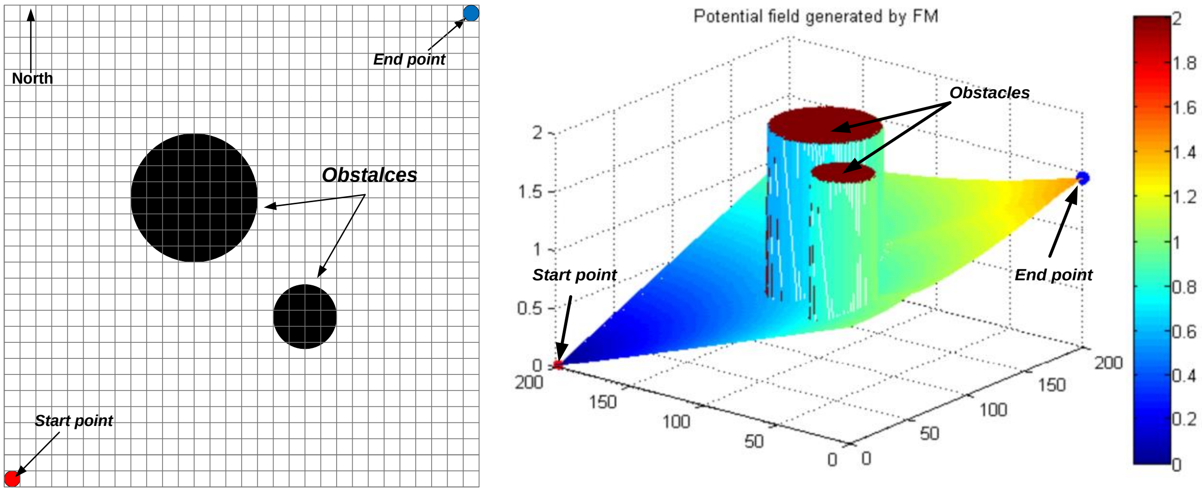
3.3.1. Fast marching algorithm

Fast marching (FM) method is first proposed by J. Sethian in 1996 to track the evolution process of advancing interface. It numerically solves the viscosity solution of eikonal equation:

$$|\nabla(T(\boldsymbol{x}))|V(\boldsymbol{x}) = 1 \qquad (3.10)$$

where $\boldsymbol{x}$ represents the point in metric space, i.e. $\boldsymbol{x}$ = (x, y) in 2D space and $\boldsymbol{x}$ = (x, y, z) in 3D space. $T(\boldsymbol{x})$ is time matrix representing the arriving time of interface front at point $\boldsymbol{x}$**,** and $V(\boldsymbol{x})$ is speed matrix and describes local propagating speed at point $\boldsymbol{x}$.

When applying FM method to the path planning problem, a more intuitive way to interpret it is from potential field perspective. This is based on path planning on a grid map, shown in Fig. 6(a), where two round obstacles are located near the centre of map, and the start and end points are at southwest and northeast corners respectively. The map is represented by binary grid map, where each grid in collision free space (C_{free}) has value 1 and girds in obstacle areas (C_{obs}) have value 0.



(a) (b)

Fig.6: (a) Grid map example (b) Potential field generated by FM

Running FM algorithm by taking grid value as speed matrix $V(\boldsymbol{x})$ creates a potential field, which is shown in Fig. 6(b). The Potential value at each point represents local arriving time of the interface, which subsequently indicates local distance to the start point if a constant speed matrix is used. The Interface begins propagating from the start point, the potential of start point is therefore the lowest and is equal to zero. Potential values at other points increase as the interface advances and reaches highest value at the end point. Because the interface is not allowed to transmit inside an obstacle area, obstacles' potentials are infinite.

Based on the potential field obtained, the gradient descent method is then applied to find the shortest collision free path by following the gradient of the potential field. It is notable that shortest path is defined in geodesic terms, which means that path has shortest Euclidean distance if the environment has constant $V(\boldsymbol{x})$ and is a weighted Riemannian manifold with varying $V(\boldsymbol{x})$

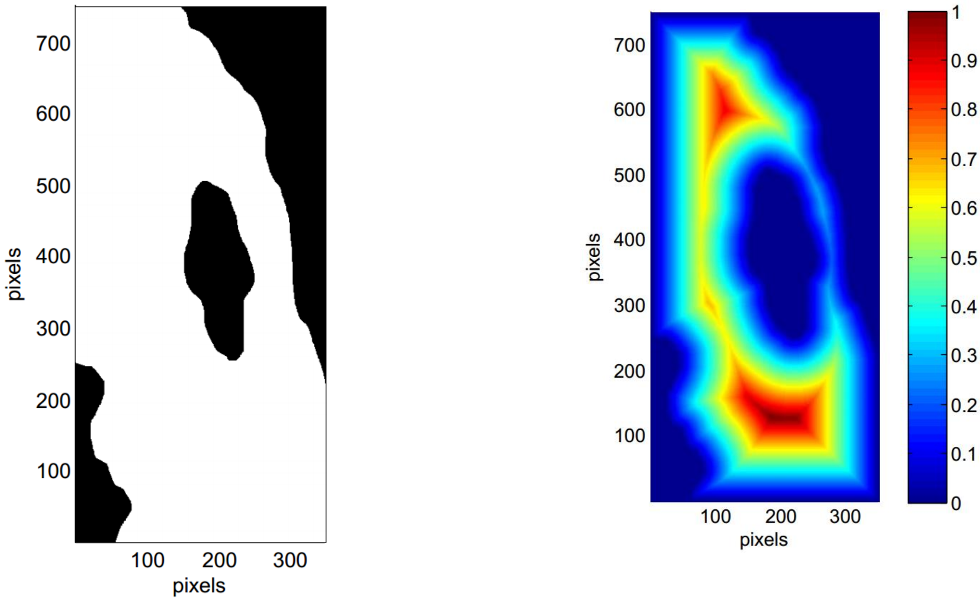
3.3.2. Fast marching square (FMS) algorithm

One of the disadvantages by using FM algorithm for path planning is that the generated path could be too close to obstacles if the environment has uniform speed matrix $V(\boldsymbol{x})$. Such a drawback is especially unpractical for USVs, because near distance areas around obstacles (usually islands and coastlines) are usually shallow water, which is not suitable for marine vehicles to navigate. Hence, it is important to keep the planned path a certain distance away from obstacles. In this research FMS algorithm is used to achieve this.

As an improved version of FM method, FMS algorithm is proposed in *Javier (2013)* to increase the practicability of path. Basic concept behind FMS is to apply conventional FM algorithm twice but with different purposes:

1) Step 1: FM is applied on original binary environment (***M_o***) to create safety map (***M_s***). Constant speed in C_{free} is used here. However, instead of calculating a single interface's propagation by using a USV's mission start point; in this process, multiple interfaces are emitted from all points that represent obstacles (points with value 0 in binary map) and continue to advance until it reaches the map boundary. Generated map (***M_s***) is shown in Fig. 7 (b), where each point is assigned with a value, ranging from 0 to 1, representing the shortest local arriving time. Since constant propagating speed is used, the local shortest arriving time further determines the shortest distance to obstacles. The further the distance to obstacles is, the higher the value will be. Such values can be viewed as indices to indicate the safety of local points. Low values represent current locations may be too close to obstacles and conse-

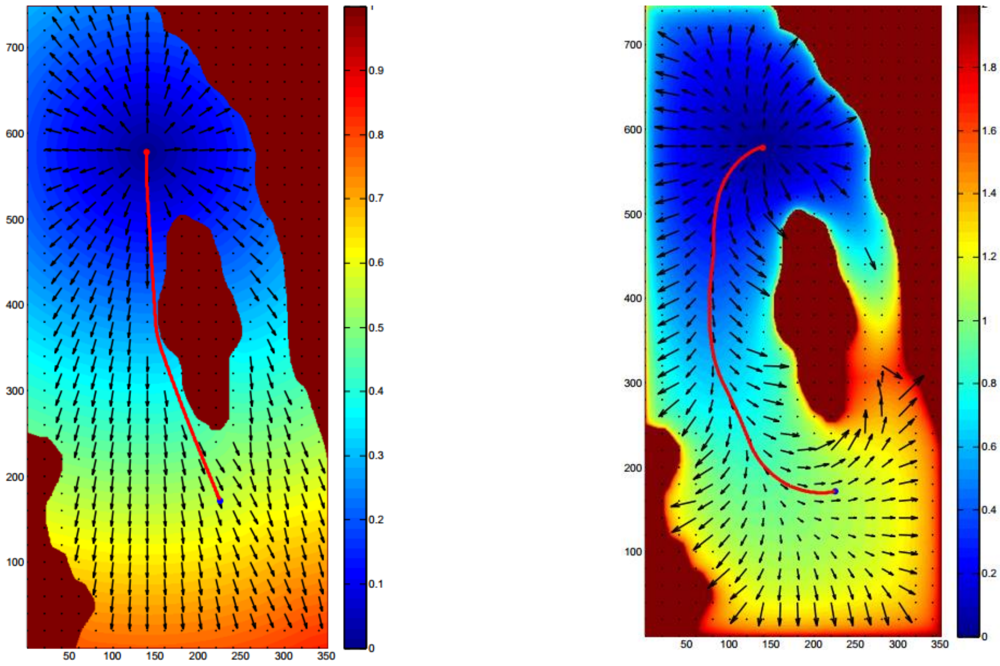
quently may not be safe to proceed; hence USVs should be encouraged to keep travelling in the areas with high index value.



(a) Binary map (b) Safety map

Fig.7: Comparison between binary map and potential index map

2) Step 2: FM is now used again over the safety map (***M_s***) to generate the potential field. USV's mission start point is the algorithm's start point. Since ***M_s*** is used as speed matrix in this step with inconstant speed over the space, the interface now tends to remain in places with high propagating speed. The generated potential field should follow the trace of the interface, which is shown in Fig. 8. Note that field's shape is obviously different to Fig. 7(b), which is generated by using constant propagating speed matrix. Potential near obstacles is always higher than other places', which is acting as a protecting layer to prevent the path to pass too close to obstacles. This can be proved by result paths shown as red lines in Fig. 8(a) and Fig. 8(b).



(a) FM (b) FMS

Fig.8: Path generated by FM and FMS

3.3.3. Target ship modelling by using constrained FMS

To prevent collision by moving obstacles, most studies in path planning research adopted the concept of ‘safety area’ to model the area for which all other vehicles are prohibited. The shape of such an area is usually circular and the centre of the area is located on the obstacle’s instantaneous position. However, in USV path planning, circular shape safety areas are not always practical, especially when a ship is travelling at high speed, which holds more risks at fore areas than aft and sides. It is more ideal to assign the shape of safety area of ship according to its velocity and in this work, is done by using constrained FMS algorithm.

Similar to FMS algorithm, constrained FMS method also consists of two steps of applying FM algorithm, but each time it is done in a constrained area. The area is constructed according to the relative velocity of OS and TS such that the area enlarges its domain when velocity increases (*Cheekuang (2010)*). When velocity is low, a circular shape is generated, meaning that collision risk is equally distributed around TS. In contrast, high velocity creates a half-elliptical area, where longer area exists at fore section. Fig. 9 shows the safety area under the relative speed of 12, 24, 42 and 54 knots respectively.

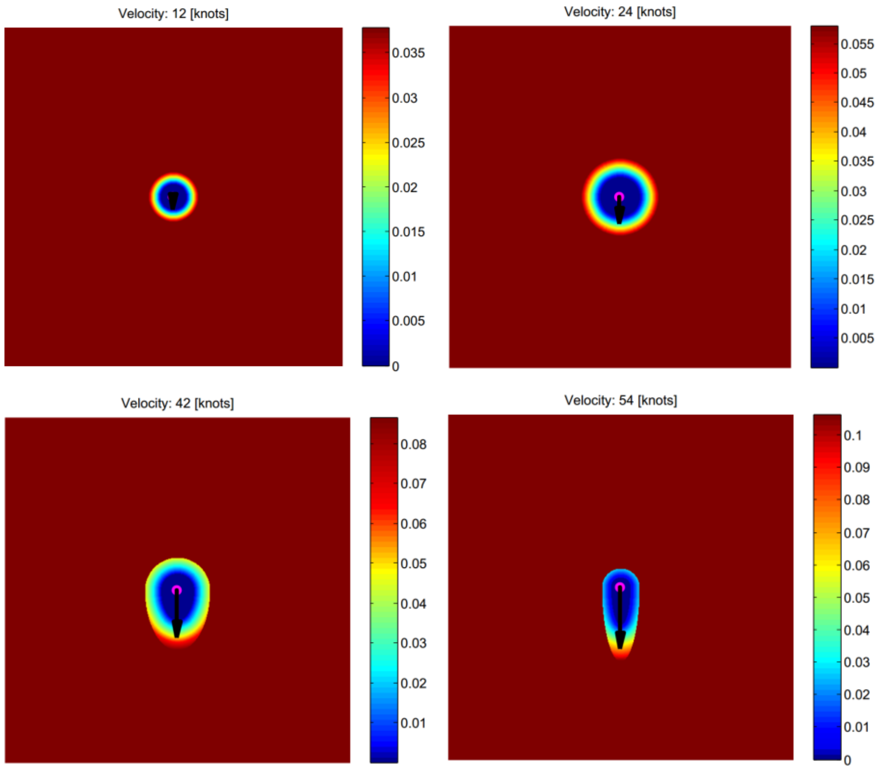


Fig.9: Ship domain generated by constrained FMS

3.3.4. Formation shape maintenance

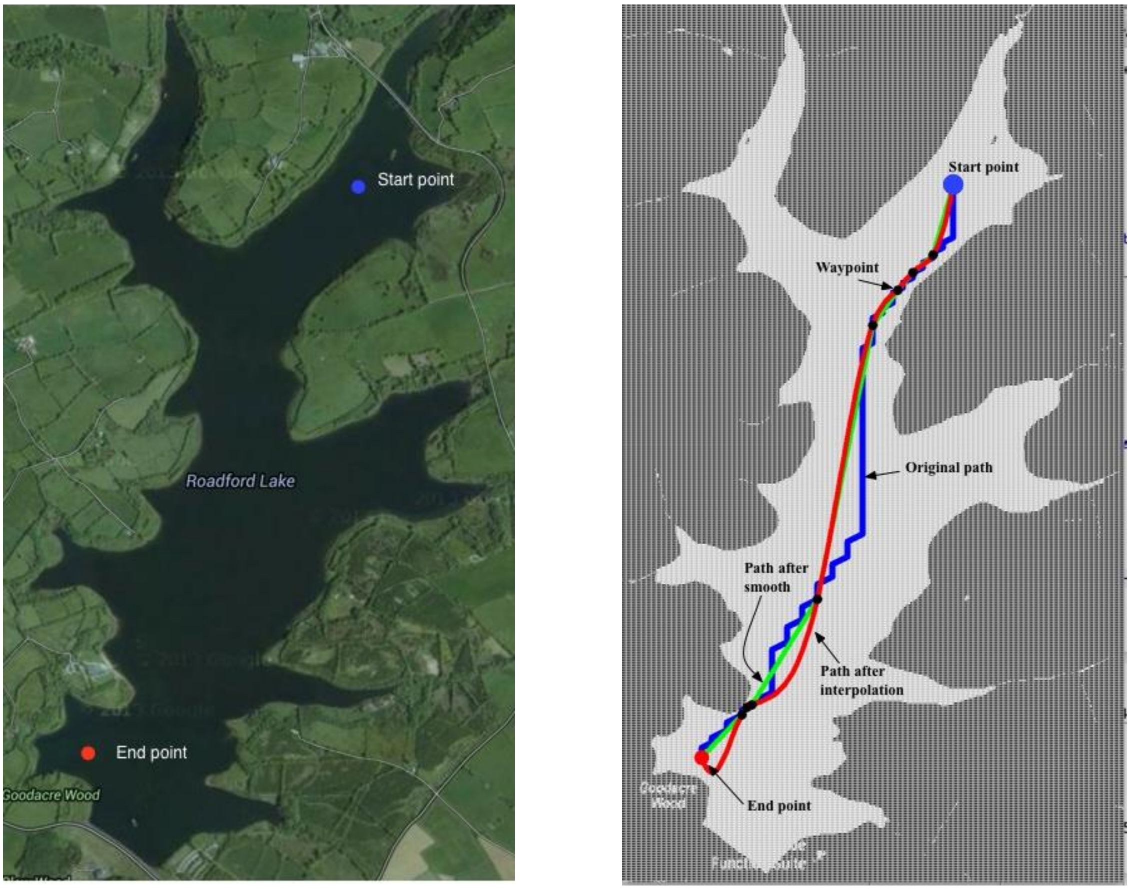
To maintain the formation shape, leader-follower formation control strategy by a centralised control connection is used in this paper. One of the vehicles is designed as leader USV whereas others are followers. All vehicles are equipped with a path planner, which is able to search trajectory based on FMS algorithm. As the guider of formation, the leader USV not only searches its own collision free trajectory, but also allocates followers’ waypoints. Waypoints are calculated based on leader’s current location as well as desired formation shape and transmitted down to each follower via wireless communication.

4. Simulation Result

Simulations have been run to validate the proposed algorithms as well as embedded system. Real maps located in Plymouth area are selected as test field to prove the practicability of algorithms.

4.1. Single USV simulation

The single USV path planning algorithms is being tested at the Roadford Lake using Springer USV. The map is first transferred into binary grid map, which has 100*350 pixels dimension. Simulation results in Fig. 10 show how the algorithm calculates the optimized path. The line in blue is the path generated by the unmodified A* algorithm. It can be seen that a number of unwanted jags are generated. The path in green is the trajectory obtained by running the first path smoother. It can be observed that the jags have been reduced dramatically, and the path reaches the minimum number of turnings. The red path represents the final continuous path obtained by interpolation smoother. The results show that the single USV path planning algorithm achieves a smooth and continuous path successfully without any collision with the coastlines.



(a) Roadford Lake map (b) Grid map with generated paths

Fig.10: Single path planning result

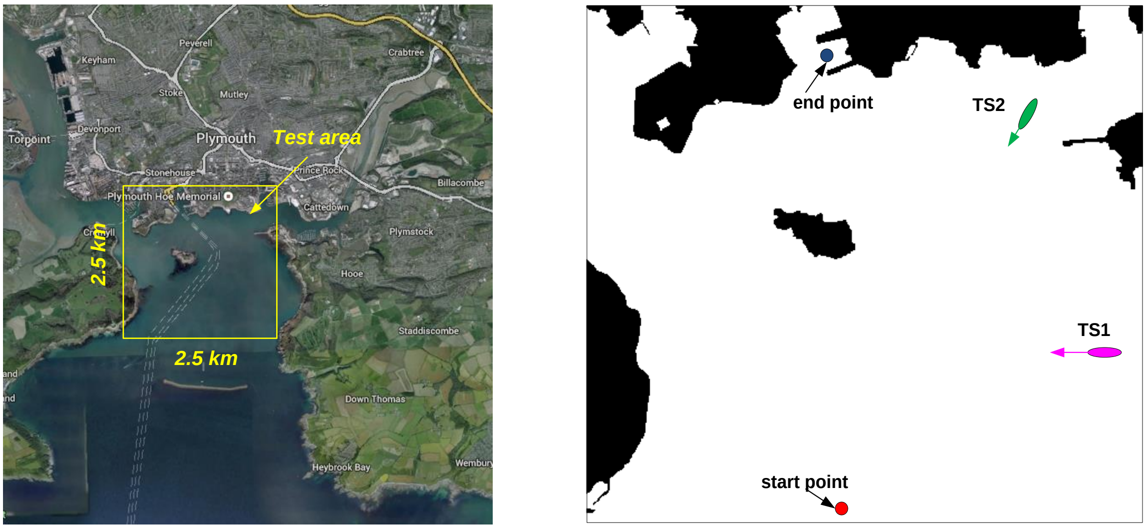
Further algorithm tests have been undertaken by selecting different start and end points. Table I below shows the results of these tests. The total movement distance cost and algorithms computation time are recorded. The results of running the A* algorithm and the whole single USV path planning algorithm with path smoothing are listed separately. Although the intrinsic property of A* algorithm is to find the shortest path on the grid map, it can be observed from the results that the costs are reduced further by using the single path planning algorithm. Additionally, it can be seen that it takes a longer time to complete the entire algorithms. This is because the path smoothing algorithm consumes more time to generate the optimised path. However, it can be indicated from these test results that the computation of single path planning algorithms is still fast enough (less than 1 second). Therefore, the algorithms can be applied practically for real-time applications.

Table I: Cost and computation time

Test	Start point (m, m)	End point (m, m)	Total cost (m)		Computation time (ms)	
			A*	Smoothed A*	A*	Smoothed A*
1	(639, 896)	(1491, 2016)	257.00	254.48	56	236
2	(319, 2464)	(1491, 2016)	155.40	152.31	26	137
3	(1278, 1120)	(1278, 2464)	184.50	182.99	19	150
4	(639, 2800)	(1278, 1120)	349.95	345.95	77	392

4.2. USV formation path planning simulation results

USV formation path planning has also been tested in a practical environment. A 2.5 km*2.5 km area near Plymouth harbour has been selected (Fig. 11(a)). The binary map of this area is shown in Fig. 11(b), where the start point of formation is located at the bottom and end point is located at the top of area. Besides static obstacles, two moving target ships have also been added into simulation to show the capability of the algorithm in dealing with a practical problem. The velocities of two target ships are 0.45 km/min and 0.15 km/min respectively.

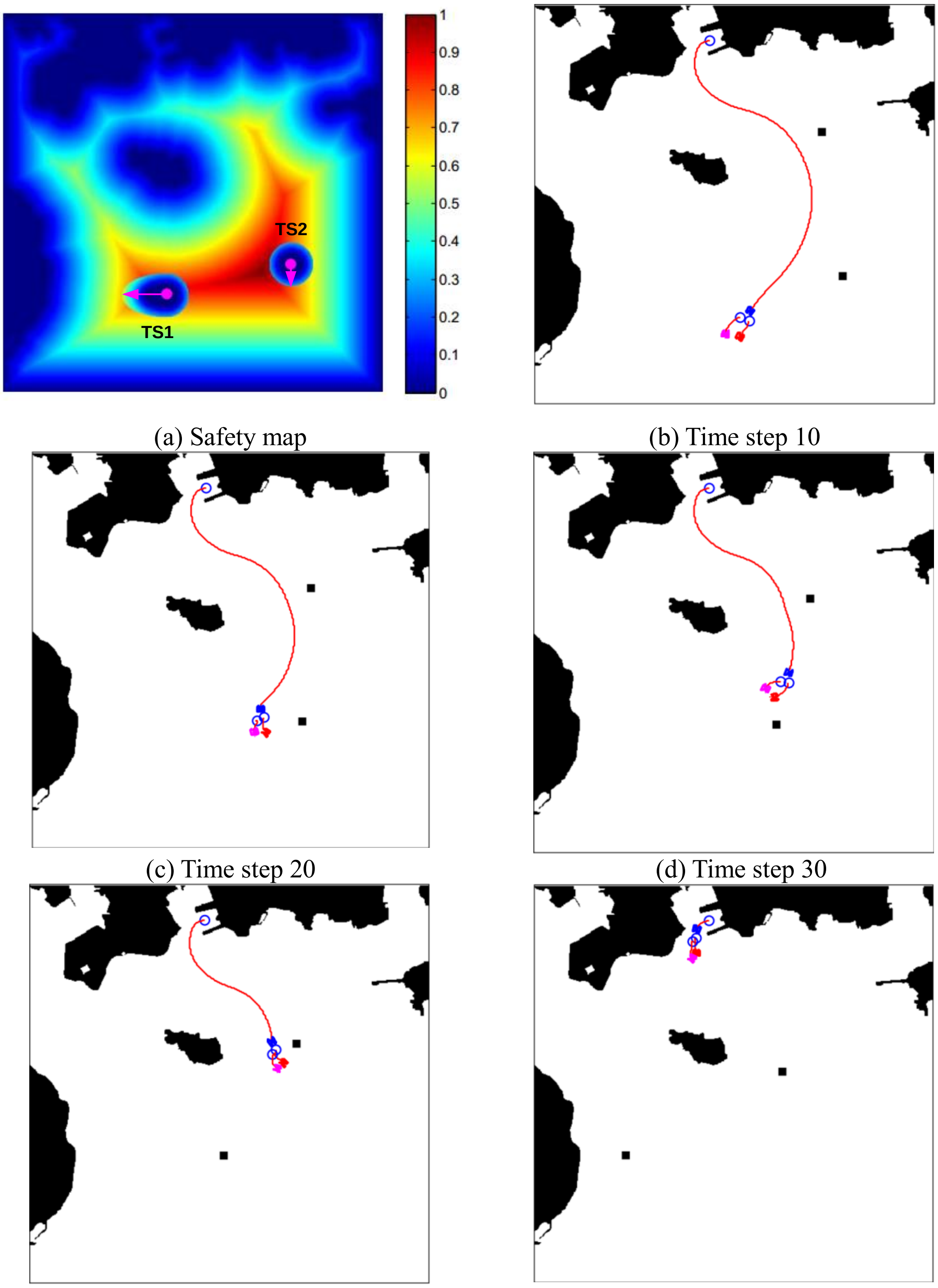


(a) Test area near Plymouth (b) Test area near Plymouth

Fig.11: Formation path planning environment map

Simulation results are presented in Fig. 12. Fig. 12(a) shows the safety map generated by algorithm at a certain time step. It clearly indicates that red area with highest safety index is the safest area for USV formation to navigate and hence the algorithm plans the path in it. Also, two moving obstacles are modelled with different safety areas. TS1 has faster speed and hence is given a half-elliptical area whereas TS2 travelling with low speed is modelled with a more circular shape.

Fig. 12(b)-(f) illustrate how the USV formation navigates in the environment without colliding with obstacles. The trajectory is re-planned at each time step to better avoid the moving target ships. Also, the formation shape can be adjusted according to environment. The shape is assigned as a triangular shape in the beginning. However, in time step 60 (Fig. 12(f)), when the formation enters into harbour area, triangular shape is no longer suitable, therefore the formation adopts a linear shape to pass through the harbour entrance.



(a) Safety map

(b) Time step 10

(c) Time step 20

(d) Time step 30

(e) Time step 45

(f) Time step 60

Fig.12: Formation path planning results

6. Conclusions

The main contribution of this paper is that it introduces the embedded system as the main USV platform to verify the practicability of path planning algorithms for both single path planning and formation path planning. The path planning algorithms have been improved in terms of consistency and efficiency. The approach of integrating the work from sensing to multi-level USV cooperation has been established building on early work.

In terms of near future work, path planning in dynamic environments will be improved by first aiming to further enhance the practicability of the navigation system. Secondly, environmental influences such as wind and current will be considered as additional optimisation criteria within the path planning process. Last, but not the least, algorithms are going to be tested in real-time in different environments. It is expected that the WAM V USV Florida Atlantic University (FAU) and Springer USV from Plymouth University will be two main testing platforms for our path-planning and collision avoidance algorithms within the next year. For multi-USV operations it is planned that these will be tested in 2015 at FAU.

Acknowledgements

The authors would like to acknowledge the supports of the Engineering and Physical Sciences Research Council (EPSRC) and Plymouth University especially Professor R. Sutton and Dr S. Shama for the opportunity to test algorithms using Springer. Also, gratitude is given for financial support from the Atlantic Center for the Innovative Design and Control of Small Ships (ACCeSS) group. The ACCeSS group is a National Naval Responsibility for Naval Engineering (NNRNE) programme funded by the US Navy's Office of Naval Research (ONR) with grant number N0014-10-1-0652. The authors would also like to thank Konrad Yearwood for his valuable critique of this paper.

Reference:

A. Calce, P. Mojiri Forooshani, A. Speers, K. Watters, T. Young and M. Jenkin, *Roboboat - building unmanned surfaced vessels from RC motorboats,* Computer Science and Engineering, York University

Alberto Romano, Pierluigi Duranti (2012), *Autonomous unmanned surface vessels for hydrographic measurement and environmental Monitoring*, TS04D-Hydrpgraphic Technologies, 6118

CheeKuang Tam, Richard Bucknall (2010), *Path-planning algorithm for ships in close-range encounters*, J. Marine Science and Technology 15/4, pp.395-407

CheeKuang Tam, Richard Bucknall (2013), *Cooperative path planning algorithm for marine surface vessels*, J. Ocean Engineering 57, pp 25-23

George Wolberg (1988), *Cubic spline interpolation: A review*, Technical Report CUCS-389-88

Javier V. Gomez, Alejandro Lumbier, Santiago Garrido, Luis Moreno (2013), Planning robot formations with fast marching square including uncertainty conditions, J. Robotics and Autonomous Systems, 61/2, pp.137-152

Ming-Cheng Tsou, Sheng-Long Kao, Chien-Min Su (2010), Decision support from genetic algorithms for ship collision avoidance route planning and alerts, J. Navigation, 63/01, pp.167-182

Nebot E., Durrant-Whyte H. (1999), *Initial calibration and clignment of low cost inertial navigation units for land vehicle applications*, J. Robotics Systems, 16/2, pp. 81-92

Noureldin, Aboelmagd, Karamat, Tashfeen B., Georgy, Jacques (2013), *Fundamentals of inertial navigation, satellite-based positioning and their integration*, Springer-Verlag Berlin Heidelberg, chapter 2, pp.21-63

R Sutton, S Sharma, T Xao (2011), *Adaptive navigation systems for an unmanned surface vehicle*, J. Marine Engineering and Technology 10/3, pp. 3-20(18)

SK Sharma, W Naeem, R Sutton (2012), *An autopilot based on a local control network design for an unmanned surface vehicle*, J. Navigation, 65/02, pp.281-301

Smierzchalski, Roman (1999), *Evolutionary trajectory planning of ships in navigation traffic areas*, J.

Marine Science and Technology 4, pp.1–6

Tall, M. H. Rynne, P. F. Lorio J. M. and von Ellenrieder K. D. (2010), *Visual-Based navigation of an autonomous tugboat*, J. MTS, 44/2, pp.37-45

Terzakis,G., Culverhouse,P., Bugmann,G., Sharma,S., Sutton,R, (2013). *Epipolar geometry in practice: 3D runtime odometry with a web-camera.* MIDAS Technical Report, MIDAS SMSE, TR.007

Wasif Naeem, George W, Irwin, Aolei Yang (2012), COLREGs-based collision avoidance strategies for unmanned surface vehicles, J. Mechatronics, 22/6, pp.669-678

Weihong Zhang ; Ping Zhuang ; Les Elkins ; Rick Simon ; David Gore, et al. (2009), *A stereo camera system for autonomous maritime navigation (AMN) vehicles*, Proc. SPIE 7332, Unmanned Systems Technology XI, 733216

Yanzhuo Xue, D.Clelland, B.S.Lee, DuanfengHan (2011), Automatic simulation of ship navigation. J. Ocean Engineering, 38/17-18, pp.2290-2305