Anisotropic GPMP2: a fast continuous-time Gaussian processes based motion planner for unmanned surface vehicles in environments with ocean currents

Jiawei Meng[1], *Student Member, IEEE,* Yuanchang Liu[1,*], *Member, IEEE,* Richard Bucknall[1], *Member, IEEE*
Weihong Guo[3], *Member, IEEE* and Ze Ji[2], *Member, IEEE*

***Abstract*—In the past decade, there is an increasing interest in the deployment of unmanned surface vehicles (USVs) for undertaking ocean missions in dynamic, complex maritime environments. The success of these missions largely relies on motion planning algorithms that can generate optimal navigational trajectories to guide a USV. Apart from minimising the distance of a path, when deployed a USVs' motion planning algorithms also need to consider other constraints such as energy consumption, the affected of ocean currents as well as the fast collision avoidance capability. In this paper, we propose a new algorithm named anisotropic GPMP2 to revolutionise motion planning for USVs based upon the fundamentals of GP (Gaussian process) motion planning (GPMP, or its updated version GPMP2). Firstly, we integrated the anisotropy into GPMP2 to make the generated trajectories follow ocean currents where necessary to reduce energy consumption on resisting ocean currents. Secondly, to further improve the computational speed and trajectory quality, a dynamic fast GP interpolation is integrated in the algorithm. Finally, the new algorithm has been validated on a WAM-V 20 USV in a ROS environment to show the practicability of anisotropic GPMP2.**

Note to Practitioners: ***Abstract*—The work reported in this article will be significant for USVs to conduct missions in complex, dynamic maritime environments where various obstacles and time-varying ocean currents exit. We develop this novel motion planning algorithm based on Gaussian process and optimise the trajectory using probabilistic inferences. The new algorithm can generate collision free trajectories that also minimise the influences caused by adverse ocean currents in a highly efficient way. In addition, the planning has been undertaken in a continuous-time domain making the generated trajectory have a guaranteed smoothness and readily feasible for autopilots to track. We use a coastal area with time-varying vortexes to present a challenging practical maritime environment. The presented algorithm integrates the available information about a fluid field regarding energy consumption and hazard level, along with the density of obstacles to plan a navigational route efficiently. To increase the practical performance of the proposed method, diverse models for generating ocean currents need to be developed in the future to tackle unpredictable situations.**

***Index Terms*—Unmanned surface vehicles, ocean currents, anisotropy, continuous-time motion planning, Gaussian process.**

J. Meng[1], Y. Liu[1,*] and R. Bucknall[1] are with the Department of Mechanical Engineering, University College London, Torrington Place, London WC1E 7JE, UK (* Corresponding author: Yuanchang Liu, email: yuanchang.liu@ucl.ac.uk, tel: +44 (0)20 7679 7062).

Z. Ji[2] is with the School of Engineering, Cardiff University, Cardiff CF24 3AA, UK.

W. Guo[3] is with the Department of Industrial and Systems Engineering, Rutgers University, New Jersey 08854, USA

Fig. 1. Demonstration of the Gazebo simulation environment in ROS: WAM-V platform at the left bottom side is carrying a parcel to perform a transportation mission.

I. Introduction

Path planning is a computational problem to find a sequence of valid configurations that moves a robot from the source to destination [1]. Such a capability is essential for USVs as they are normally deployed in ocean environments, where obstacles avoidance and energy consumption minimisation are important [2], [3]. Available planning algorithms such as A*, rapidly-exploring random tree (RRT), fast marching method and evolutionary algorithms can, to some extent, be used for USVs. However, limited capacity in solving multi-objective planning problems in an efficient way has largely made these algorithms unsuitable for a practical deployment [4].

In recent years, GP (Gaussian process) motion planning (GPMP, or its updated version GPMP2) has been proposed to deal with the motion planning problem in continuous-time space with a very high computational speed and can generate a path that is both short and smooth in large-scale or high-dimensional spaces [5], [6]. Such a feature undoubtedly makes the GP based motion planning a promising solution for USVs motion planning. However, improvements are still needed to enhance these algorithms with the required capability to deal with ocean currents.

Another consideration is previous research related to motion planning for USVs usually utilises tests in simplified simulation environment under platforms such as Matlab, treating the vehicle as an infinitely small point, operating in the configuration space. A lack of data from operational USVs can give rise to unintended assumptions built into the algorithms that in turn result in simulations giving erroneous or oversimplified interpretations. This might not be ideal when transferring the simulation results as a prediction tool to help determine the general performance of a USV when applied to an actual USV in a practical environment.

In this paper, we will propose a new motion planning method based on continuous-time GP, named anisotropic GPMP2, which can deal with time-varying ocean currents and obstacles such as islands, reefs, buoys in marine environments simultaneously and highly efficiently. The proposed method is evolved initially from the maximum a posterior (MAP) estimation of the Bayes theorem with the main contributions can be summarised as:

- A new energy consumption likelihood model subject to ocean currents is introduced. Ocean currents are expressed as local anisotropy to mathematically indicate the preferred travelling direction at each location in a planning space. Then, an energy consumption likelihood is calculated using the anisotropic fast marching (AFM) based upon the anisotropy and further integrated into the GPMP2, named as anisotropic GPMP2. Generated trajectories are proven to be able to adaptively follow ocean currents.
- A new dynamic fast GP interpolation method is proposed to better deal with obstacle avoidance and further improve the computational speed. Unlike the uniform interpolation process in GPMP2, a dynamic interpolation process according to obstacles' locations has been developed to better optimise the trajectory.
- The developed algorithms are thoroughly tested in both self-constructed simulation environments, and a Gazebo-based simulation environments under ROS. A demonstration of the Gazebo simulation environment in ROS is shown in Fig. 1. Augmented practicability has been achieved by fully considering the actual dynamics of a USV and practical obstacle configurations. Such a validation can effectively reduce the testing cost, in comparison with the high cost of building a full-size physical USV.

In simulations, we compared anisotropic GPMP2 with motion planning algorithm benchmarks in a number of situations. The results demonstrate the following:

- The proposed method can minimise the effect of ocean currents in marine environments compared with state-of-the-art GP motion planning planning algorithm, namely the Gaussian process motion planner 2 (GPMP2).
- The proposed method requires a shorter execution time to generate a feasible path compared with another motion planning algorithm designed for marine environments, namely the anisotropic fast marching method (AFM).
- The proposed method has the ability to efficiently tackle large-scale motion planning problems with or without ocean currents.

The rest of the paper is organised as follows. Section II details other related works and further articulates the motivation for this research. Section III describes the mathematical model of GP motion planning in detail. Section IV presents the proposed method in detail. Section V presents the optimisation tool used in the research while at the same time analysing its structure. Section VI presents the proposed path planner's simulation results and then compares it with several existing motion planning methods. Section VII demonstrates the implementation of the proposed path planning method in ROS, followed by the conclusion in Section VIII.

II. Related works

The aim of path planning is to find an optimal path between the start point and goal point while avoiding obstacles. Previous researchers have different perspectives on the definition of optimality. Some indicate that the optimal trajectory should be the one with the shortest length; whereas, others emphasise that the optimality is subject to multiple criteria including the path's smoothness and dynamic constraints such as velocity or acceleration [7], [8], [9]. Because of the inclusion of dynamics, path planning can also be referred to as motion planning and herein we do not explicitly distinguish these two terminologies. In this paper, the main focus is to develop a motion planning algorithm that can not only optimise its path length, execution time and path smoothness, but also consider other important factors or constraints related to the dynamic environment, such as wind and ocean currents, simultaneously.

Previously, researchers have attempted to develop a variety of methods to solve the path planning problem including geometry-based methods such as Dijkstra [10], A* [11], D* [12], FM [13] and LSM [14], intelligent methods such as PSO [15], ACO [16], GA [17] and WPA [18], and probabilistic sampling-based methods such as PRM [19] and RRT* [20] according to the classifications in [21]. Geometry-based methods [10], [11], [12], [13], [14] require a relatively strict geometry-based or graph-based model of the map. Therefore, this type of method always needs a relatively long execution time to determine the path, especially for high dimensional planning spaces. Due to the long execution time, performing re-planning problems using such methods is difficult. Similar to geometry-based methods, intelligent methods [16], [17], [15], [18] also need a long execution time to determine a feasible path making them unsuitable for re-planning. Furthermore, almost all intelligent methods are prone to the local minima problem, where a locally optimal solution can generated leading to a failure in finding the global optimum upon which the final trajectory should be calculated. The probabilistic sampling-based method [19], [20], [22], [23], [24], [25] successfully solves the problem of long execution time by abandoning the concept of explicitly characterising the configuration space, simultaneously using a sampling connection rule to replace it. The randomness of this approach is conducive to provide fast solutions for path planning problems in high-dimensional configuration spaces and is suitable for re-planning problems. However, it might also be vulnerable to the local minima problem, although the optimal path can be found when there is no limitation on execution time. Additionally, paths generated by probabilistic methods are not smooth and could be sinuous, which require additional smoothing process for practical applications. Hence the challenges in the previous stage of path planning research can be summarised as: 1) improving path quality, 2) shortening execution time, and 3) enabling an efficient re-planning capability in large-scale or high-dimensional configuration spaces.

In recent years, another category of path planning methods, denoted path optimisation, have attracted an increasing attention in path planning research as they are appropriate for path planning in high-dimensional configuration spaces with paths encoded as a sequence of states and controls. Specifically, path optimisation is the process of designing a path that can minimise/maximise some measure of performance while satisfying a series of constraints [26]. Compared with the other methods mentioned above, path optimisation methods provide several benefits including: 1) the capability of smoothing and shortening the path in a coupled way during the planning process and 2) superiority in computational speed making it suitable for online planning in environments with rapidly changing factors [27].

Because of these two promising benefits, several popular optimisation based path planning methods have been proposed and developed. For example, the CHOMP [28], [9] is a seminal work of this type, which showcased its effectiveness on a variety of robotic platforms. To be more specific, the CHOMP uses a pre-computed signed distance field to detect collision and a covariant gradient descent to reduce the probability of collision and subsequently improve the smoothness of the path. Then, the STOMP [7] is proposed based on CHOMP, which samples several paths with noises to explore the configuration space around an initial path. By combining all the paths that have explored the space, an updated and better optimised path with lower cost can be generated. The critical improvement of the STOMP is that it is capable of dealing with non-differentiable constraints. However, both CHOMP and STOMP are not well suited to dealing with path planning problems with multiple constraints. To address this problem, TrajOpt [29], [8] is proposed to solve complex motion planning problems with few states designed as swept volumes to ensure trajectory safety in continuous time space. However, TrajOpt always needs post-processing on the path's smoothness, which could significantly extend the execution time.

Although the above-mentioned shortcomings present concerns on the development potential of path optimization methods, Gaussian Process (GP) based motion planning (GPMP) algorithms [5], [6], [30], [31] have been developed in recent years to successfully address these issues. First, instead of representing trajectories as a set of planned states in configuration space that has been unanimously used by conventional motion planning algorithms, GPMP algorithms regard trajectories as functions having a direct mapping from time to states in the continuous-time space, and functions can be sampled using a Gaussian process. Such a strategy can guarantee the smoothness of the generated trajectory. Second, based upon the feature of efficient structure-exploiting GP regression, GPMP can implement a fast GP interpolation, which ensures the feasibility of adjusting path smoothness during the planning process. Last but not the least, trajectory optimisation based upon the GP can be regarded as a probabilistic interference, where an initial knowledge of a trajectory can be used as *a prior* and, by considering optimisation constraints, such as collision avoidance, as *likelihood functions*, the trajectory will be optimised following *maximum a posterior*.

The probabilistic inference based GPMP, also named as GPMP2, has been successfully implemented on several practical platforms including robotic arms and mobile robots [32], [33], [34], [35]. However, no current studies report any implementation of GPMP2 on USV platforms due to following challenges. First, the current version of GPMP2 can only optimise a trajectory with regards to short distance and collision avoidance. Further constraints, especially ocean currents, should be fully taken into account for USVs when GPMP2 is employed. The trajectory should be encouraged to largely follow the direction of ocean currents to minimise the energy consumption. Second, planning a trajectory for a robotic arm is different from planning for a USV. The former focuses on planning in a high-dimensional space but with in a small work space; whereas the latter normally performs on a large-scale space (e.g. a USV can be required to travel for several kilometers) mostly in a 2D or 3D domain. Therefore, proper modifications are required to address these two differences to make GPMP2 more suitable for USVs.

In the following sections, we will first introduce the fundamental preliminaries of GPMP2 and our proposed new algorithm, anisotropic GPMP2. The results, including the comparative studies, are then provided to demonstrate the capability of our newly proposed algorithm.

III. GPMP2

In this section, we explain GPMP2 algorithm in detail. Overall, GPMP2 considers a motion planning problem as a MAP estimation, which views the information relevant to the start and goal points as *a prior* and the information relevant to various conditional constraints such as collision and energy consumption as *likelihoods*. Then GPMP2 would maximise *the posterior* of the MAP estimation to obtain an optimal solution.

A. Problem formulation and GP prior

GPMP2 algorithm can be formulated as a trajectory optimisation problem, and further, it applies Gaussian Processes to optimise trajectories in an efficient manner. Formally, the trajectory optimisation aims to determine the optimal trajectory from all feasible trajectories while satisfying any user defined constraints and minimising any user prioritised costs [23], [36]. By considering a trajectory as a function of continuous time t, such an optimisation process can be written as the standard form of an optimisation problem with continuous variables as:

$$\begin{aligned} \text{minimise} \quad & F[\theta(t)] \\ \text{subject to} \quad & G_i[\theta(t)] \leq 0,\ i = 1, \ldots, m_{ieq} \\ & H_i[\theta(t)] = 0,\ i = 1, \ldots, m_{eq}. \end{aligned} \tag{1}$$

where $\theta(t)$ is a continuous-time trajectory function mapping a specific moment t to a specific robot state θ. $F[\theta(t)]$ is an objective function to find the optimal trajectory by

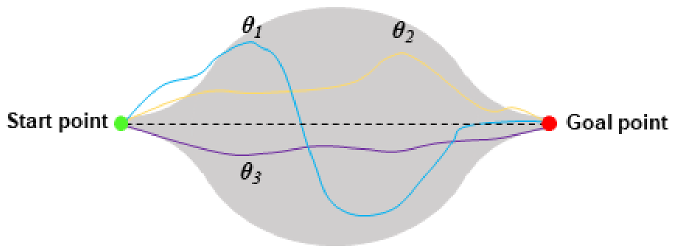


Fig. 2. Continues-time trajectories sampled by GP prior: Green point represents the start point; Red point represents the goal point; θ_1, θ_2 and θ_3 are three continues-time trajectory samples; Dashed line represents the mean trajectory $\mu(t)$ and shaded area represents the covariance $K(t,t')$.

minimising the higher-order derivatives of robot states, such as velocity and acceleration, and collision costs. $G_i[\theta(t)]$ is a task-dependent inequality constraint function and $H_i[\theta(t)]$ is a task-dependent equality constraint function that contains the desired start and goal states with specified configurations.

As stated in [5], by properly allocating the parameters of low-resolution states (defined as support states) and interpolating high-resolution states (defined as interpolated states), the computational cost of Gaussian Processes can be efficiently reduced and, as illustrated in Fig. 2, a continuous-time trajectory function represented by a Gaussian Process can be formulated as:

$$\theta(t) \sim \mathcal{GP}(\mu(t), K(t,t')), \tag{2}$$

where $\mu(t)$ is a vector-valued mean function (the dashed line in Fig. 2) and $K(t,t')$ is a matrix-valued covariance function (the shaded area in Fig. 2) that indicates the approximate area of the trajectory samples from the Gaussian Process:

$$\mu = [\mu_1 ... \mu_N]^T \tag{3}$$

$$K = [K(t_i, t_j)]|_{(0 \le i,j \le N)} \tag{4}$$

Then we consider this vector-valued GP as generated by a linear time-varying stochastic differential equation (LTV-SDE). The LTV-SDE is used to describe a system dynamics model of a robot, which, in our case, is a USV. The LTV-SDE is stated in the following equation:

$$\dot{\theta}(t) = A(t)\theta(t) + u(t) + F(t)w(t), \tag{5}$$

where $u(t)$ is a known system control input, $A(t)$ and $F(t)$ are the time-varying matrices of the selected system dynamics model, and $w(t)$ represents the white process noises. $w(t)$ is stated in the following equation:

$$w(t) \sim GP(0, Q_c\delta(t-t')), \tag{6}$$

where Q_c is the power-spectral density matrix and $\delta(t-t')$ is the Dirac delta function.

Based on the LTV-SDE stated in Eq. 5, the solution of the GP can be calculated based on the vector-valued mean function and matrix-valued covariance function by the process described in [37] as:

$$\mu(t) = \Phi(t,t_0)\mu_0 + \int_{t_0}^{t} \Phi(t,s)u(s)ds \tag{7}$$

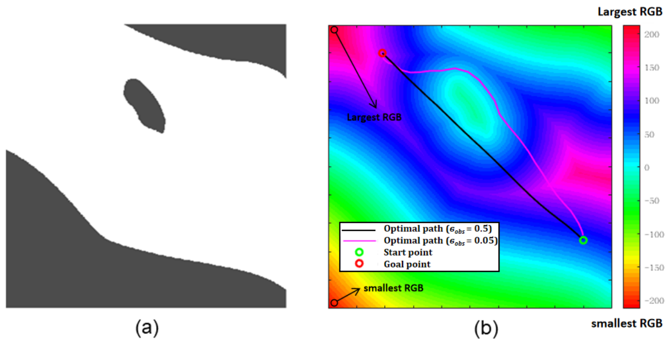


Fig. 3. Signed distance field of a coastal environment: (a) is the pixel map and (b) is the signed distance field of the map. In the signed distance field, the point with a larger RGB value is more secure than the point with a smaller RGB value. The value of σ_{obs} is inversely proportional to the weight of the behaviour (staying inside safe region). The point with a larger value on the color bar is relatively safe; on the other hand, the point with a smaller value on the color bar is relatively dangerous.

$$K(t,t') = \Phi(t,t_0)K_0\Phi(t',t_0)^T + \int_{t_0}^{min(t,t')} \Phi(t,s)F(s)Q_cF(s)^T\Phi(t,s)^T ds, \tag{8}$$

where u_0 and K_0 are initial mean and covariance of the first state, respectively. $\Phi(t,s)$ is the state transition matrix from time t to time s.

Thus, GP prior distribution is given in terms of the mean u and covariance K:

$$p(\theta) \propto exp\{-\frac{1}{2}||\theta - u||_K^2\} \tag{9}$$

Based on the proof in [5], we know the inverse matrix-valued covariance function K^{-1} is exactly sparse (or block-tridiagonal). Such a property is extremely important as it significantly reduces the computational complexity of solving the inverse of a matrix and enables a fast GP interpolation. The detailed information about fast GP interpolation will be presented in Section III-D.

B. *Collision likelihood function*

First, a binary event c_i is defined as a distribution: $c_i = 0$ (if there is no collision risk in a trajectory) or $c_i = 1$ (if there is any collision risk in a trajectory). In this solution, we are only interested in the collision-free event ($c_i = 0$).

Then a likelihood function is defined as a distribution in the exponential family that can indicate the likelihood of a collision-free trajectory:

$$l(\theta; c_i = 0) = exp\{-\frac{1}{2}||h(\theta)||^2_{\Sigma_{obs}}\}, \tag{10}$$

where the hyperparameter matrix Σ_{obs} is defined as:

$$\Sigma_{obs} = \begin{bmatrix} \sigma_{obs} & & \\ & ... & \\ & & \sigma_{obs} \end{bmatrix}, \tag{11}$$

where σ_{obs} is the "obstacle cost weight" corresponds to a specific covariance matrix and $h(\theta)$ is a vector-valued obstacle cost function:

$$h(\theta_i) = [c(d(\theta_i))], \tag{12}$$

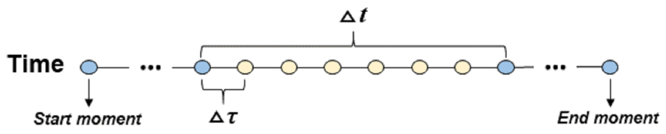


Fig. 4. The relationship between Δt and $\Delta\tau$ in fast GP interpolation: Δt is the low-resolution sample interval and $\Delta\tau$ is the high-resolution sample interval. A low-resolution sample interval is more suitable in free space whereas a high-resolution is better used in areas with cluttered obstacles.

where c is the hinge loss function defined by the following equation:

$$c(d)=\begin{cases}-d+\epsilon, & if\ d\leq\epsilon\\ 0, & if\ d>\epsilon\end{cases}, \tag{13}$$

where d is the signed distance of a point and ϵ is the safety distance. To be more specific, ϵ indicates the obstacle boundaries in the signed distance field. c is defined in Eq. 13 where $c_i = 1$ indicates a collision event. Fig. 3 demonstrates the signed distance field of a coastal environment and effects of different signed distance field parameter σ_{obs} in the same motion planning problem. Specifically, the value of σ_{obs} adjusts the weight of the behaviour (staying inside the safe region) in the motion planning problem.

C. MAP estimation

Based on the information in Section III-A and Section III-B, now we are able to conduct a Bayesian inference to obtain a locally optimised trajectory. To be specific, we have a GP prior $p(\theta)$ that contains sample trajectories from a start point to a goal point and a collision likelihood function $l(\theta; c_i = 0)$ that helps in selecting collision-free sample trajectories. Further, the collision likelihood function can also be written as $p(c_i = 0|\theta)$. Thus, the Bayesian inference of this problem can be defined as:

$$\theta^* = \arg\max_{\theta} p(\theta|c_i=0) = \arg\max_{\theta}\ p(\theta)p(c_i=0|\theta) \tag{14}$$

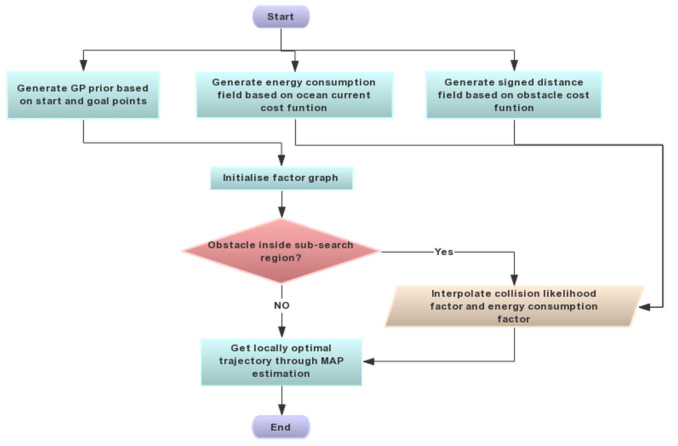
After the optimisation process of Eq. 14 is completed, a locally optimised trajectory θ^* can be found. Generally, the MAP estimation of a GP can be converted into a least squares problem, which has been thoroughly studied for many years. In this paper, we use Levenberg–Marquardt algorithm to solve the least squares problem [38].

D. Fast GP Interpolation

Based on the inference in [37], [5], [36], the posterior mean of the trajectory at any moment τ can be approximated by the Laplace method. It can then be expressed in terms of the current trajectory at moment t:

$$\theta(\tau) = \widetilde{\mu}(\tau) + \widetilde{K}(\tau,t)\widetilde{K}^{-1}(\theta-\widetilde{\mu}), \tag{15}$$

where $\widetilde{\mu}$ and $\widetilde{K}$ are the mean function and covariance matrix corresponds to the Gaussian process of the generated trajectory. In general, the computational complexity of the above equation is *O(N)*, which increases the difficulty considerably. Nevertheless, the computational complexity of Eq. 15 is *O(1)*



Start
Generate GP prior based on start and goal points
Generate energy consumption field based on ocean current cost funtion
Generate signed distance field based on obstacle cost funtion
Initialise factor graph
Obstacle inside sub-search region?
Yes
NO
Interpolate collision likelihood factor and energy consumption factor
Get locally optimal trajectory through MAP estimation
End

Fig. 5. Flow diagram of the overall algorithm.

in this case because the system dynamics model is a LTV-SDE and K^{-1} is exactly sparse (or block-tridiagonal). The proof for the computational complexity in this case is stated in [5]. Thus, we can derive the following equation as:

$$\theta(\tau) = \widetilde{\mu}(\tau) + \Lambda(\tau)(\theta_i - \widetilde{\mu}_i) + \Psi(\tau)(\theta_{i+1} - \widetilde{\mu}_{i+1}), \tag{16}$$

where

$$\begin{aligned}\Lambda(\tau) &= \Phi(\tau,t_i) - \Psi(\tau)\Phi(t_{i+1},t_i),\\ \Psi(\tau) &= Q_{i,\tau}\Phi(t_{i+1},\tau)^T Q_{i,i+1}^{-1}\end{aligned} \tag{17}$$

is derived by substituting

$$\widetilde{K}(\tau)\widetilde{K}^{-1} = [0...0\ \ \Lambda(\tau)\ \Psi(\tau)\ \ 0...0] \tag{18}$$

The expression of $Q_{a,b}$ is defined by the following equation:

$$Q_{a,b} = \int_{t_a}^{t_b} \Phi(b,s)F(s)Q_cF(s)^T\Phi(b,s)^T ds \tag{19}$$

and through this we prove the feasibility of using fast GP interpolation with changeable intervals under this circumstance. In order to provide a more intuitive understanding of fast GP interpolation, an example detailing the relationship between Δt and $\Delta\tau$ is shown in Fig. 4. In GPMP2, $\Delta\tau$ specifies the intermediate step sizes that constitute the trajectory segment associated to the time interval Δt. However, such a sampling strategy leads to unnecessary redundancy on the generated path, especially in open regions such as an ocean surface with no obstacle.

IV. Anisotropic GPMP2

In this section, we present the anisotropic GPMP2 algorithm, in detail, which includes two new parts: 1) energy consumption likelihood function and 2) dynamic fast GP interpolation. Also, the anisotropic fast marching (AFM), which is used to construct the energy consumption likelihood function is first introduced. The flow diagram of the overall algorithm is demonstrated in Fig. 5.

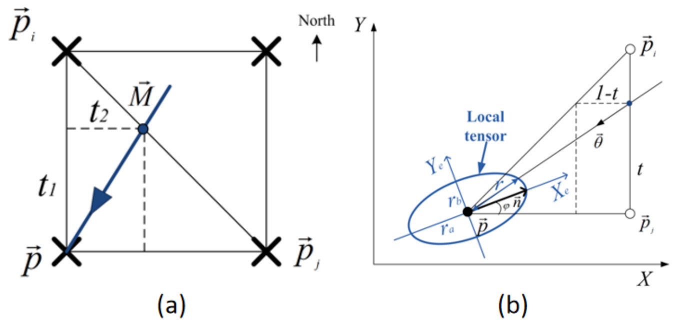


Fig. 6. Important definitions in AFM: (a) the FM method updating scheme and (b) the schematic of calculating arrival time using AFM at neighbour point [41].

A. Anisotropic fast marching method

In order to plan a trajectory to largely exploit ocean currents and subsequently save energy, an appropriate metric needs to be established to evaluate the impact of currents in an environment. In this paper, we adopt the concept of anisotropy, which is the existence of preferred directions in a domain, to construct this new energy metric. Local ocean current direction will be mathematically expressed as an anisotropy in each place, and the energy metric will be calculated such that in any area the direction that follows the ocean current will be given the highest priority. This will consequently provide an energy consumption likelihood that enables GPMP2 to fast search for a trajectory that can largely exploit the ocean current, and because of the adoption of anisotropy, this new motion planning algorithm will be denoted as the anisotropic GPMP2. The energy metric is calculated in this paper using anisotropic fast marching method (AFM), which is an updated version of the fast marching method (FM). To give a holistic introduction of the construction of energy metric, the mathematical fundamentals of FM and AFM will be introduced in following subsections. Detailed algorithm development including pseudocode can be referred to [39], [40].

1) Fast marching method (FM): The FM method is a level-set method introduced in image processing [39]. It aims to generate an arrival time map U in order to satisfy the Eikonal equation, which describes a wave front propagation scenario. The Eikonal equation can be expressed in the form of:

$$||\bigtriangledown U(\overrightarrow{p})||\tau(\overrightarrow{p}) = 1, \tag{20}$$

where $\tau(\overrightarrow{p})$ is the wave propagation speed that is related to position $\overrightarrow{p} = (x, y)$. The solution $U(\overrightarrow{p})$ is the wave arrival time at $\overrightarrow{p}$. Note that $U(\overrightarrow{p})$ can be interpreted as the distance cost from the start point to $\overrightarrow{p}$, if the wave propagation speed is constant.

As shown in Fig. 6 (a), if the optimal path to $\overrightarrow{p}$ arrives from northeast and intersects $\overrightarrow{p_i}\overrightarrow{p_j}$ at $\overrightarrow{M}$, the arrival time at $\overrightarrow{p}$ can be computed from $\overrightarrow{p_i}$ and $\overrightarrow{p_j}$, denoted as $u_{\overrightarrow{p_i}\overrightarrow{p_j}}(\overrightarrow{p})$, and expressed as:

$$u_{\overrightarrow{p_i}\overrightarrow{p_j}}(\overrightarrow{p}) = \min_{t_1 t_2} \left(t_1 u_{\overrightarrow{p_i}} + t_2 u_{\overrightarrow{p_j}} + \frac{\sqrt{t_1^2+t_2^2}}{\tau(\overrightarrow{p})} \right), \tag{21}$$

where $u_{\overrightarrow{p_i}}$ and $u_{\overrightarrow{p_j}}$ are the arrival time at $\overrightarrow{p_i}$ and $\overrightarrow{p_j}$ respectively. t_1 and t_2 satisfy the following conditions: $t_1 + t_2 = 1$ and t_1, $t_2 > 0$.

2) Anisotropic fast marching method (AFM): From Eq. 21, it can be observed that the conventional FM method only takes the distance cost into account. To integrate the orientation information, the conventional FM method was improved in [42], to a new algorithm denoted as the 'anisotropic fast marching algorithm' (AFM) and the Eikonal equation can be rewritten as:

$$u_{\overrightarrow{p_i}\overrightarrow{p_j}}(\overrightarrow{p}) = \min_{t\in[0,1]} \left(t u_{\overrightarrow{p_i}} + (1-t) u_{\overrightarrow{p_j}} + \frac{||\overrightarrow{\theta}(t)||}{\tau(\overrightarrow{\theta}(t))} \right). \tag{22}$$

Here $\overrightarrow{\theta}(t) = \overrightarrow{p} - (t\overrightarrow{p_i} + (1-t)\overrightarrow{p_j})$ is a vector that indicates the direction of the cost/speed profile. $||\overrightarrow{\theta}(t)||$ is the distance between $\overrightarrow{p}$ and the intersection point between $\overrightarrow{p_i}$ and $\overrightarrow{p_j}$. The wave propagation speed is now dependent on orientation as denoted as $\tau(\overrightarrow{\theta}(t))$. To simplify the notation, $\overrightarrow{\theta}(t)$ is replaced by $\overrightarrow{\theta}$ in the following sections. In contrast to the conventional FM method, the local cost/speed characteristic of the AFM method is no longer circular.

In [42], an elliptical shape was used to represent the local cost/speed model as conversion to a circle is a simple process (conventional FM case). The direction of each vector (direction of ocean current in our case) is defined along the major axis of the ellipse; while its minor axis is perpendicular to the vector's direction. In this case, the wave front travels along the major axis as the propagation's preferred direction. Generally, the ellipse speed profile is described as:

$$\frac{x^2}{r_a^2} + \frac{y^2}{r_b^2} = 1, \tag{23}$$

where r_a and r_b are the major and minor radii along the X and Y axes of the local ellipse frame. If the ellipse is along a direction of $\overrightarrow{n}$ (as shown in Fig. 6 (b)), then the radius r along $\overrightarrow{\theta}$, satisfies:

$$r^2 \left(\frac{(cos\angle(\overrightarrow{\theta},\overrightarrow{n}))^2}{r_a^2} + \frac{(sin\angle(\overrightarrow{\theta},\overrightarrow{n}))^2}{r_b^2} \right) = 1, \tag{24}$$

where $\angle(\overrightarrow{\theta},\overrightarrow{n})$ is the angle between $\overrightarrow{\theta}$ and $\overrightarrow{n}$, r is used as the wave propagation speed along the $\overrightarrow{\theta}$ direction; hence, $\tau(\overrightarrow{\theta})$ in Eq. 22 can now be written as:

$$\begin{aligned}\tau(\overrightarrow{\theta}) &= \frac{1}{\sqrt{\frac{(cos\angle(\overrightarrow{\theta},\overrightarrow{n}))^2}{r_a^2} + \frac{(sin\angle(\overrightarrow{\theta},\overrightarrow{n}))^2}{r_b^2}}} \\ &= \frac{||\overrightarrow{\theta}||}{\sqrt{\frac{(\theta_x cos\varphi)^2}{r_a^2} + \frac{(\theta_y sin\varphi)^2}{r_b^2}}},\end{aligned} \tag{25}$$

where φ is the angle between $\overrightarrow{n}$ and X axis. θ_x and θ_y are the components of $\overrightarrow{\theta}$ along the X and Y axes. Therefore, $u_{\overrightarrow{p_i}\overrightarrow{p_j}}(\overrightarrow{p})$ in Eq. 22 can be rewritten as,

$$\min_{t\in[0,1]} \left(t u_{\overrightarrow{p_i}} + (1-t) u_{\overrightarrow{p_j}} + \sqrt{\frac{(\theta_x cos\varphi)^2}{r_a^2} + \frac{(\theta_y sin\varphi)^2}{r_b^2}} \right). \tag{26}$$

Fig. 7 represents the energy consumption map created by the AFM. In four different scenarios each with different number of

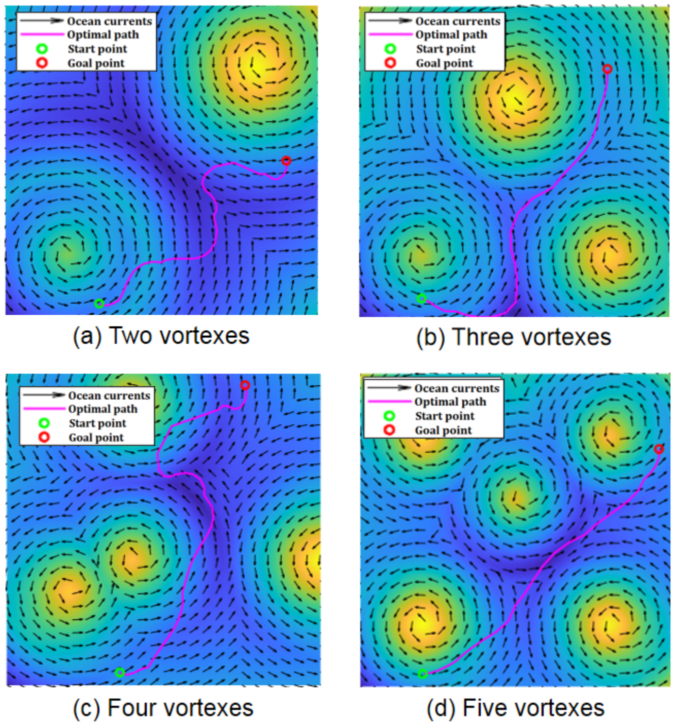


Fig. 7. Anisotropic GPMP2 with added energy consumption function likelihood works in marine environments with a variety of vortexes. The energy consumption rate at the point in the bright yellow region close to the vortex is relatively high; on the other hand, the energy consumption rate at the point in the dark blue region away from the vortex is relatively low. To demonstrate the optimal performance of anisotropic GPMP2 on tracking ocean currents, we empirically minimised σ_e in (a), (b), (c) and (d).

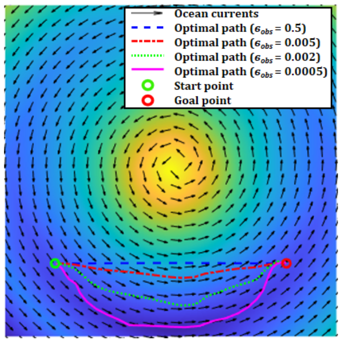


Fig. 8. The effect of σ_e in anisotropic GPMP2 motion planner. The energy consumption rate at the point in the bright yellow region close to the vortex is relatively high; on the other hand, the energy consumption rate at the point in the dark blue region away from the vortex is relatively low.

vortexes and taking the start and goal points into consideration, different areas have been assigned with different likelihood levels to indicate the difficulties of following currents locally. In Fig. 7, lower valley regions in the map, indicated with darker colours, correspond to regions where the direction of water currents is more uniform and consistent and where a USV would tend to follow.

B. Anisotropic energy consumption likelihood function

The offset angle between the ocean current and the ideal direction of a path can lead to additional energy consumption to

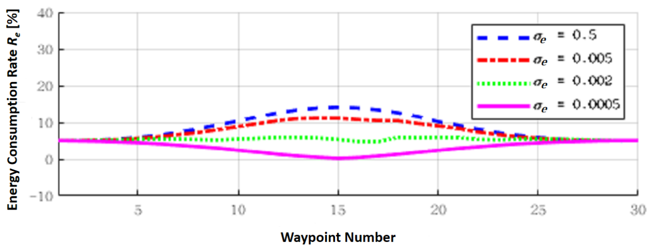


Fig. 9. The energy consumption rate at each waypoint of trajectory when using different parameters σ_e on energy consumption field. The energy consumption rate is defined by the RGB value at the point in energy consumption field generated by AFM. In Fig. 8, the energy consumption rate at the point in the bright yellow region close to the vortex is relatively high; on the other hand, the energy consumption rate at the point in the dark blue region away from the vortex is relatively low.

resist the ocean current disturbances. Hence we add an energy consumption likelihood function to minimise this consumption as well as avoid hazardous regions such as vortexes, which is defined as:

$$l(\theta; e) = exp\{-\frac{1}{2}||g(\theta)||^2_{\Sigma_e}\}, \tag{27}$$

where the definition of matrix Σ_e is the same as Σ_{obs} and $g(\theta)$ is a vector-valued ocean current cost function:

$$g(\theta_i) = [e(\theta_i)], \tag{28}$$

where e is an energy consumption function that can obtain the energy consumption rate at a point based on the energy matrix defined by AFM in Section IV-A.

Based on this energy consumption likelihood function, the Bayesian inference of anisotropic GPMP2 is defined as:

$$\theta^* = \arg\max_{\theta}\ p(\theta)[p(c_i = 0|\theta)\ p(e|\theta)] \tag{29}$$

To gain a more intuitive understanding of the feasibility of the proposed energy consumption likelihood function, the example in Fig. 7 demonstrates the influence of energy consumption likelihood function in marine environments with various vortexes. Based on the observation and measurement of ocean currents in [43], it is clear that ocean currents within a specific area comprise of a constant magnitude and variant directions. We thereby decide to use vortexes to generate ocean currents in the following simulations due to vortexes also comprise of a constant magnitude and variant directions. As shown in Fig. 7, the high energy consumption regions are represented in yellow with the low energy consumption regions in blue. The path generated by anisotropic GPMP2 with the energy consumption likelihood function follows ocean currents to minimise the energy consumption in blue regions and avoids vortexes in yellow regions to maintain safety. To demonstrate the adjustable capability of the proposed likelihood function, Fig. 8 and Fig. 9 demonstrate the results with different values of parameter σ_e, the value of which is inversely proportional to the proposed likelihood function's weight in the entire path planning process.

Algorithm 1: Dynamic Fast GP Interpolation

```
Input: Start state $\theta_0$, goal state $\theta_N$, maximum
  sampling time $t_{max}$ and search radius $r$
Precompute Continuous-time trajectory samples with
  mean $\mu$ and covariance $K$
Compute the low-resolution sample interval $\Delta t$ by
  using Eq. 30
for $i = 1, 2, \ldots, N$ do
    Compute total sample number $n_j$ in the
      corresponding sub-search region by using by Eq.
      32
    Compute high-resolution sample interval $\Delta \tau_j$ by
      using Eq. 31
    for $j = 1, 2, \ldots, n_j$ do
        Perform fast GP interpolation with
          high-resolution sample interval $\Delta \tau_j$ by using
          Eq. 33
    end
end
Output: optimal path $\theta^*$
```

C. Dynamic fast GP interpolation

GPMP2 uses uniformly distributed fast GP interpolation to generate paths. Nevertheless, such an approach leads to a particular drawback, where the execution time would be relatively long when the interpolation interval is relatively small. On the contrary, increasing the interpolation interval would impair the obstacle avoidance performances as well as compromise path smoothness. To solve such a problem, we propose dynamic fast GP interpolation as shown in Algorithm 1. This algorithm determines the distribution of obstacles when initialising the GP prior, and an adaptive $\Delta\tau$ can be defined according to the locations of obstacles for fast GP interpolation. By using the proposed dynamic fast GP interpolation algorithm, dense points can be sampled in areas with obstacles; while redundant sampling points are removed from intervals in free spaces. Such a strategy can further optimise the execution time as well as the path length by removing redundant sampling points.

At the beginning of Algorithm 1, start state θ_0, goal state θ_N, maximum sampling time t_{max} and search radius r are required as inputs. Based on this information, the GP prior can be computed using Eq. 9. Then the low-resolution sampling interval Δt in the global search region is computed by the following equation:

$$\Delta t = \frac{t_{max}}{N}, \tag{30}$$

where N is the total number of sub-searching regions. In each local sub-searching region, the high-resolution sampling interval $\Delta\tau$ is computed using the following equation:

$$\Delta \tau_j = \frac{\Delta t}{n_j}, \tag{31}$$

where n_j is the total number of sample points inside each corresponding sub-searching region and is defined as:

$$n_j = \lambda \cdot f(\frac{s_j}{s_{sub}}), \tag{32}$$

where λ is the self-defined proportionality coefficient, f is a function used to round-up to the nearest integer, s_j is the total area of obstacles inside the corresponding sub-searching region and s_{sub} is the area of the corresponding sub-searching region with its dimension equal to πr^2. Hence Eq. 16 can be written as:

$$\theta(\tau_j) = \widetilde{\mu}(\tau_j) + \Lambda(\tau_j)(\theta_i - \widetilde{\mu}_i) + \Psi(\tau)(\theta_{i+1} - \widetilde{\mu}_{i+1}), \tag{33}$$

where

$$\begin{aligned} \Lambda(\tau_j) &= \Phi(\tau_j, t_i) - \Psi(\tau_j)\Phi(t_{i+1}, t_i), \\ \Psi(\tau_j) &= Q_{i,\tau_j}\Phi(t_{i+1}, \tau_j)^T Q_{i,i+1}^{-1} \end{aligned} \tag{34}$$

is derived by substituting

$$\widetilde{K}(\tau_j)\widetilde{K}^{-1} = [0...0 \;\; \Lambda(\tau_j) \;\; \Psi(\tau_j) \;\; 0...0] \tag{35}$$

In the following simulations, we demonstrate that a trajectory with higher smoothness can be generated in obstacle-free areas by using the dynamic GP interpolation without compromising any collision avoidance performances.

V. Factor graph

In the anisotropic GPMP2, a factor graph is used to deal with MAP estimation as it offers the following advantages [44]:

- It can simplify the modelling problem and provide better clarity;
- It can improve computational performance.

To be more specific, a factor graph G is defined as:

$$G = \{\Theta, \mathcal{F}, \mathcal{E}\}, \tag{36}$$

where $\Theta = \{\theta_0, ..., \theta_N\}$ is a set of variable nodes (in our case is a set of USV's states), $\mathcal{F} = \{f_0, ..., f_M\}$ is a set of factor nodes and $\mathcal{E}$ are edges that connect the variable nodes and factor nodes. The factorisation of the posterior in our problem can be formulated as:

$$p(\theta|c, e) \propto \prod_{m=1}^{M} f_m(\Theta_m), \tag{37}$$

where f_m are factors on variable subset Θ_m. A comprehensive structure illustrating how these different factors are integrated for anisotropic GPMP2 is shown in Fig. 10 with each factor explained in following subsections.

A. Prior factor

The GP prior in our problem can be factored as:

$$p(\theta) \propto f_0^p(\theta_0) f_N^p(\theta_N) \prod_{i=0}^{N-1} f_i^{gp}(\theta_i, \theta_{i+1}), \tag{38}$$

where $f_0^p(\theta_0)$ as shown in Fig. 10 (a) defines the prior distribution on the start point and $f_N^p(\theta_N)$ defines the prior distribution on the goal point. Based on Eq. 9, we can further derive the expression of $f_i^p(\theta_i)$:

$$f_i^p(\theta_i) = exp\{-\frac{1}{2}||\theta_i - u_i||^2_{K_i}\}, i = 0 \; or \; N. \tag{39}$$

Here K_i is the covariance matrix and μ_i is the mean vector.

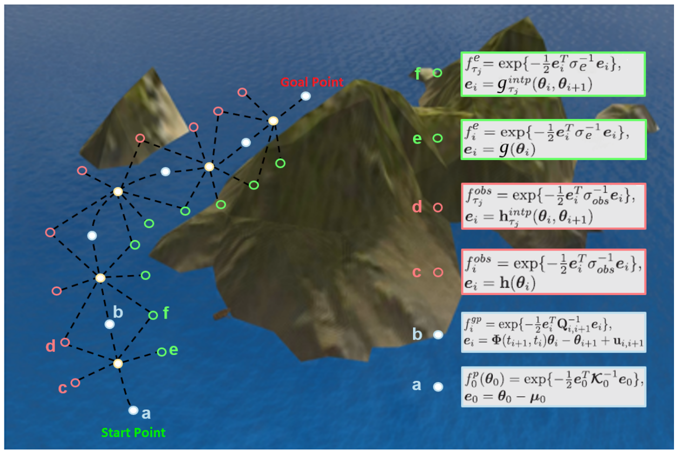

Goal Point
Start Point
a
b
c
d
e
f
$f^e_{\tau_j} = \exp\{-\frac{1}{2} e_i^T \sigma_e^{-1} e_i\}$, $e_i = g^{intp}_{\tau_j}(\theta_i, \theta_{i+1})$
$f^e_i = \exp\{-\frac{1}{2} e_i^T \sigma_e^{-1} e_i\}$, $e_i = g(\theta_i)$
$f^{obs}_{\tau_j} = \exp\{-\frac{1}{2} e_i^T \sigma_{obs}^{-1} e_i\}$, $e_i = h^{intp}_{\tau_j}(\theta_i, \theta_{i+1})$
$f^{obs}_i = \exp\{-\frac{1}{2} e_i^T \sigma_{obs}^{-1} e_i\}$, $e_i = h(\theta_i)$
$f^{gp}_i = \exp\{-\frac{1}{2} e_i^T Q_{i,i+1}^{-1} e_i\}$, $e_i = \Phi(t_{i+1}, t_i)\theta_i - \theta_{i+1} + u_{i,i+1}$
$f^p_0(\theta_0) = \exp\{-\frac{1}{2} e_0^T \mathcal{K}_0^{-1} e_0\}$, $e_0 = \theta_0 - \mu_0$

Fig. 10. The structure of the factor graph of anisotropic GPMP2: (a) demonstrates the prior factor, (b) demonstrates the GP prior factor, (c) demonstrates the obstacle factor, (d) demonstrates the interpolated obstacle factor (e) demonstrates the ocean current factor and (f) demonstrates the interpolated ocean current factor.

To connect the start node and goal node by fast GP interpolation, GP prior factor $f_i^{gp}(\theta_i, \theta_{i+1})$ as shown in Fig. 10 (b) is formulated as:

$$f_i^{gp}(\theta_i, \theta_{i+1}) = exp\{-\frac{1}{2}||\Phi(t_{i+1}, t_i)\theta_i - \theta_{i+1} + u_{i,i+1}||^2_{Q_{i,i+1}}\}, \tag{40}$$

where $u_{a,b} = \int_{t_b}^{t_a} \Phi(b,s)\mu(s)ds$, $\Phi(t_{i+1}, t_i)$ is the state transition matrix and the definition of $Q_{i,i+1}$ can be found in Eq. 19.

B. Collision likelihood factor

The collision likelihood $l(\theta; c)$ can be factored by two categories of obstacle cost factors which include: (1) a regular obstacle factor f_i^{obs} and (2) an interpolated obstacle factor $f_{\tau_i}^{obs}$:

$$l(\theta; c) = \prod_{i=0}^{N}\{f_i^{obs}(\theta_i)\prod_{j=1}^{n_j} f_{\tau_j}^{obs}(\theta_i, \theta_{i+1})\}, \tag{41}$$

where n_j represents the number of interpolated obstacle factor within each low-resolution sampling interval Δt.

The regular obstacle factor f_i^{obs} as shown in Fig. 10 (c) is a unary factor connected with each variable node (or each USV's state) and is defined as:

$$f_i^{obs}(\theta_i) = exp\{-\frac{1}{2}||h(\theta_i)||^2_{\sigma_{obs}}\}, \tag{42}$$

where $h(\theta_i)$ is an M-dimensional vector-valued obstacle cost function for each state defined in Eq. 12 and σ_{obs} is a M×M hyperparameter matrix.

The interpolated obstacle factor $f_{\tau_j}^{obs}$ as shown in Fig. 10 (d) is a binary factor connected with each of the two variable nodes representing the obstacle cost at each interpolated variable node θ_{τ_i} within each low-resolution interval Δt. And it is defined as:

$$f_{\tau_j}^{obs}(\theta_i, \theta_{i+1}) = exp\{-\frac{1}{2}||h(\theta(\tau_j))||^2_{\sigma_{obs}}\}, \tag{43}$$

where $h(\theta(\tau_j))$ can also be viewed as $h_{\tau_j}(\theta_i, \theta_{i+1})$ and this is achieved by dynamic fast GP interpolation introduced in Section IV-C.

C. Energy consumption likelihood factor

Similar to the previous factor, energy consumption likelihood can also be factored by two categories of energy cost factors which include: (1) a regular energy consumption factor f_i^e and (2) an interpolated energy consumption factor $f_{\tau_i}^e$:

$$l(\theta; e) = \prod_{i=0}^{N}\{f_i^e(\theta_i)\prod_{j=1}^{n_j} f_{\tau_j}^e(\theta_i, \theta_{i+1})\}, \tag{44}$$

where n_j represents the number of interpolated energy consumption factor within each low-resolution sampling interval Δt.

The regular energy consumption factor f_i^e as shown in Fig. 10 (e) is a unary factor connected with each variable node (or each USV's state). It is defined as:

$$f_i^e(\theta_i) = exp\{-\frac{1}{2}||g(\theta_i)||^2_{\sigma_e}\}, \tag{45}$$

where $g(\theta_i)$ is an M-dimensional vector-valued energy cost function for each state defined in Eq. 28 and σ_e is a M×M hyperparameter matrix.

The interpolated energy consumption factor $f_{\tau_j}^e$ as shown in Fig. 10 (f) is a binary factor connected with each two variable nodes. It represents the energy cost at each interpolated variable node θ_{τ_i} within each low-resolution interval Δt and is defined as:

$$f_{\tau_j}^e(\theta_i, \theta_{i+1}) = exp\{-\frac{1}{2}||g(\theta(\tau_j))||^2_{\sigma_e}\}, \tag{46}$$

where $g(\theta(\tau_j))$ can also be viewed as $g_{\tau_j}(\theta_i, \theta_{i+1})$ and this is also achieved by dynamic fast GP interpolation in Section IV-C.

VI. Simulations and Discussions

In this section, we demonstrate the performance of anisotropic GPMP2 in detail.

A. Simulation details

Three categories of simulations have been conducted to evaluate the proposed method, namely the anisotropic GPMP2. Specifically, the proposed method was quantitatively tested against two simulation benchmarks and compared with the state-of-the-art motion planning algorithms including GPMP2 [6], AFM [13], A* [11] and RRT* [20]. We also demonstrate the capabilities of the proposed method in qualitative tests. In all the simulations, GP-based methods such as GPMP2, GPMP2-dyn-intep and anisotropic GPMP2 were always initialised with a constant-velocity straight-line trajectory. Table I describes the realisations of the comparison motion planning algorithms. Table II details the specifications of the parameters used in all the motion planning algorithms. The specific parameters of GP-based motion planning, A* and RRT* in the following simulations in various resolutions are clarified. In Table II, ϵ indicates the safety distance [pixel], σ_{obs} indicates

TABLE I
DESCRIPTION OF THE COMPARISON MOTION PLANNING ALGORITHMS.

Name	Description of the motion planning algorithm
A*	Based on a precomputed grid-map, it aims to find the path with smallest cost $f(n) = g(n) + h(n)$ from the start node to the end node, where $g(n)$ is the cost of the path from the start node to n and $h(n)$ is a heuristic function that estimates the cost of the cheapest path from n to the goal.
RRT*	Within the configuration space, it grows a tree rooted at the start node until reaching the end node. A series of tree branc- -hes connecting the start and end nodes constructs a planned path. Also, it keeps re-wiring the tree branches to shorten the length of the planned path.
AFM	FM produces the potential field by simulating the propagation of an electromagnetic wave. Then it performs gradient descent to find an optimal path. AFM is an improved version of the FM method which can consider tidal currents and has higher computational efficiency than the level set method.
PSO	A series of particles are moved around in the configuration space according to a pre-defined function. The movements of the particles are guided by their own best-known position in the configuration space and the entire swarm's best-known position. Improved positions will further guide the movements of the swarm once they are discovered. By repeating this step several times, a global / local optimal path can be found.

TABLE II
SPECIFICATION OF THE PARAMETERS USED IN THE MOTION PLANNING ALGORITHMS.

Map [pixel]	GP-based Motion Planning					A*	RRT*
	ϵ	σ_{obs}	σ_e	T_{max}	N	l	l
500x500	20	0.05	0.005	2.0	5	10.0	10.0
1000x1000	20	0.05	0.005	4.0	10	10.0	10.0
2000x2000	20	0.05	0.005	8.0	20	10.0	10.0
5000x5000	20	0.05	0.005	20.0	50	10.0	10.0

TABLE III
SPECIFICATION OF THE USED HARDWARE PLATFORM.

Name of the Device	Description	Quantity
Processor	2.6-GHz Intel Core i7-6700HQ	8
RAM	8 GB	1

the obstacle cost weight, σ_e indicates the energy cost weight, T_{max} indicates the total sampling time [s], N indicates the low-resolution region number in Algorithm 1 and l indicates the step size [pixel]. In the following simulations, one pixel in the map equals one meter in the corresponding motion planning problem. Table III is a specification of the hardware platform used. In this section, the optimal path indicates the path generated by a motion planning algorithm based on a series of certain parameters and constraints.

B. System dynamics model

A constant-velocity motion model is selected in this work to represent the system dynamics model of the USV. On an actual voyage, a USV usually adjust its angular velocity to change orientation, while in the meantime, maintaining its linear velocity to maintain stability. The constant-velocity motion model can generate an initial trajectory with constant linear velocity on each point. Then offset would occur on each point on initial trajectory based on the effect of factor graph. During the deviating process, the linear velocity attribute on each point would remain constant. After this process, a new trajectory would be constructed by connecting the new points. Each point on the new trajectory has the same linear velocity compared with the corresponding point on the initial trajectory and the position of each point has already been changed. To guarantee the consistency of the new trajectory when constructing it, the angular velocity of each point on the new trajectory would then be inconsistent. Hence we finally obtain a new trajectory with constant linear velocity and time-varying angular velocity, which is consistent with an actual voyage dynamics of a USV. This prior will minimise acceleration in motion planning, thus reducing energy consumption and giving the physical generated path an increased degree of smoothness. To be more specific, the dynamics of USVs are represented with the double integrator linear system with additional white noise on acceleration. Hence the trajectory is generated by the LTV-SDE in Eq. 5 with parameters:

C. Benchmark without ocean currents

$$A = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}, x(t) = \begin{bmatrix} r(t) \\ v(t) \end{bmatrix}, F(t) = \begin{bmatrix} 0 \\ I \end{bmatrix}, u(t) = 0, \quad (47)$$

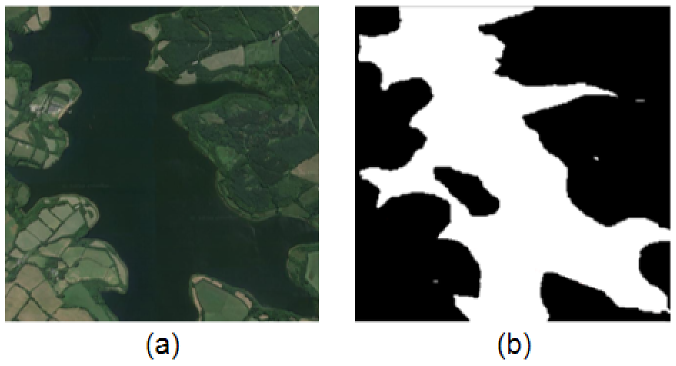


Fig. 11. Coastal Environment used in simulations: (a) is the map of Roadford Lake, Devon, UK and (b) is the binary image of it. Furthermore, (a) is a 500x500 resolution map in the simulation and a 2500x2500 [m^2] area in the real-word. This means 1 [pixel] is approximately equal to 5 [m] in the simulation.

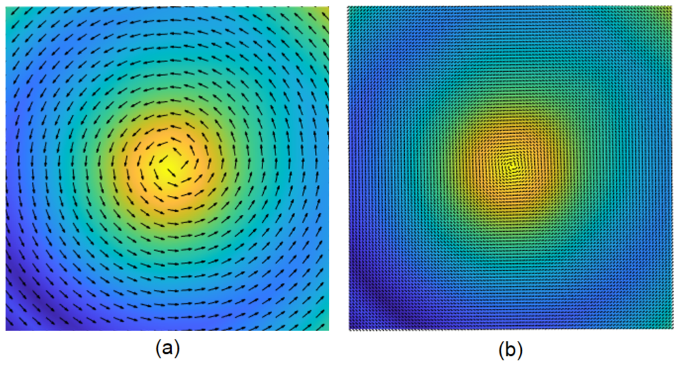


Fig. 12. A comparison of different-resolution no-obstacle environments with time-invariant ocean currents: (a) demonstrates the 500x500 resolution map (an area about 2500x2500 [m^2] in the real-world) with time-invariant ocean currents and (b) demonstrates the 2000x2000 resolution map (an area about 10000x10000 [m^2] in the real-world) with time-invariant ocean currents. A high resolution map would potential require an increased computation time.

TABLE IV
A COMPARISON OF GPMP2-DYN-INTEP (GPMP2 WITH DYNAMIC FAST GP INTERPOLATION), A* AND RRT* ON AVERAGE EXECUTION TIME (T), PATH LENGTH (L) AND SAMPLE NUMBER K IN 20 PATH PLANNING PROBLEMS WITHOUT OCEAN CURRENTS. EACH EXPERIMENT WAS TESTED 5 TIME TO CALCULATE THE AVERAGE VALUE.

Map [pixel]	Problem	GPMP2-dyn-intp			A*			RRT*		
		T [ms]	L [pixel]	K	T [ms]	L [pixel]	K	T [ms]	L [pixel]	K
500x500	1	**122.6**	**560.8**	20	2410.9	565.7	41	2939.2	624.8	61
	2	**137.7**	**688.5**	37	29786.6	688.7	62	6358.2	735.6	73
	3	**125.3**	729.2	25	16217.5	688.7	49	5095.6	**683.3**	67
	4	**102.2**	783.8	26	14628.1	**618.4**	50	9913.1	684.2	64
	5	**123.2**	**419.5**	23	8137.3	441.4	41	2251.5	451.9	44
1000x1000	6	**272.4**	1177	40	4310.5	**1131.4**	81	7995.4	1265.1	125
	7	**286.7**	1405	79	110974.6	**1359.8**	120	9189.1	1419.3	140
	8	**276.2**	1231.6	51	300677.9	**1213.3**	95	8410.1	1290.1	127
	9	**233.2**	1246.9	49	42185.4	**1207.5**	94	76804.7	1341.5	132
	10	**242.6**	1014.5	49	37621.2	**882.8**	81	5545.2	970.7	96
2000x2000	11	**386.5**	2275.9	80	5721.8	**2262.3**	161	15732.7	2548.7	246
	12	**478.5**	**2935.1**	163	-	-	-	52749.1	2947.7	289
	13	**492.2**	**2403.3**	103	-	-	-	49727.5	2762.9	270
	14	**471.9**	**2709**	95	-	-	-	72163.3	2849.2	282
	15	**420.7**	**1908.7**	98	-	-	-	8137.8	1966.2	208
5000x5000	16	**1730.8**	6655.8	200	28613.4	**5656.9**	401	80052.2	6296.2	609
	17	**1715.6**	7636.2	411	-	-	-	157234.2	**7193.5**	705
	18	**1681.7**	7144.4	265	-	-	-	71707.1	**6137.1**	601
	19	**1638.2**	**6547.7**	240	-	-	-	159036.7	6692.9	652
	20	**1700.1**	5920.5	244	-	-	-	74032.8	**4550.8**	442

Notes: GPMP2-dyn-intp only uses the proposed interpolation method (fast GP interpolation) without using anisotropy to deal with ocean currents due to the limited working performances of RRT* and A* in ocean environments.

where $r = (x, y)^T$ is the position vector, $v = (v_x, v_y)$ is the velocity vector and given $\Delta t_i = t_{i+1} - t_i$,

$$\Phi(t,s) = \begin{bmatrix} I & (t-s)I \\ 0 & I \end{bmatrix}, Q_{i,i+1} = \begin{bmatrix} \frac{1}{3}\Delta t_i^3 Q_C & \frac{1}{2}\Delta t_i^2 Q_C \\ \frac{1}{2}\Delta t_i^2 Q_C & \Delta t_i Q_C \end{bmatrix}, \tag{48}$$

Analogously, this prior is centered around a zero-acceleration trajectory.

In this subsection, we demonstrate the improvement of GPMP2 with only dynamic fast GP interpolation (GPMP2-dyn-intp) over A*, RRT* in various 2D environments including no-obstacle environment, single-obstacle environment, multi-obstacle environment, narrow-passage environment and coastal environment without the effect of ocean currents. All the aforementioned motion planning methods would stop once a feasible path has been found. The detailed information regarding the comparison between GPMP2-dyn-intp, A* and RRT* is shown in Table. IV. The aim of this simulation is to show the benefits of dynamic fast GP interpolation in various 2D environments, and that it has the shortest execution time, highest smoothness, near-optimal path length and good performance on avoiding obstacles compared with A* and RRT*. The smoothness is measured by the number and degree of the jags on path. To be more specific, a path with a small number of jags would be considered as being smooth. In addition, a jag with an obtuse angle would be considered as being smooth compared with an acute angle.

Against this benchmark, GPMP2-dyn-intp has a significant advantage on the average execution time compared with A* and RRT* in all simulations, especially in large-scale motion planning problems as shown in Table. IV. Based on the results, with the increase of problem complexity, i.e. resolution difference, GPMP2-dyn-intp has the slowest growth in its average execution time, compared with A* and RRT*. For instance, the average execution time of GPMP2-dyn-intp in

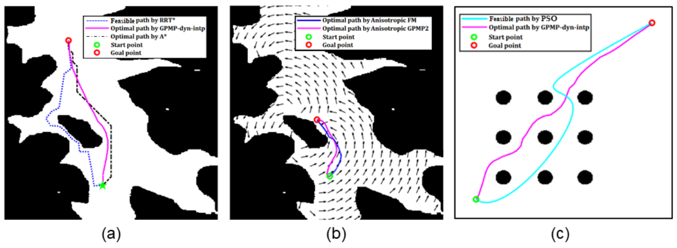


Fig. 13. Comparisons about the paths generated by various motion planning algorithms: (a) compares the paths generated by GPMP2-dyn-intp, A* and RRT* in coastal environment without the effect of ocean currents, (b) compares the paths generated by anisotropic GPMP2 and AFM in coastal environment with the effect of ocean currents and (c) compares the paths generated by GPMP2-dyn-intp and PSO in multi-obstacle environment without the effect of ocean currents.

the 500×500 resolution map is 122 ms and in the 5000×5000 resolution map is just 1693.1 ms. This is because GPMP2-dyn-intp has the prior distributions of obstacles and ocean currents, which can save lots of time by reducing the randomness of the motion planning problem. A* was not able to deliver a feasible solution in most of the large-scale motion planning problems only being able to deliver a feasible solution in different-resolution no-obstacle environments with the start and goal points initialised on the diagonal. This is because A* always needs to search possibles path in configuration space to obtain the optimal path which has high dependence on the heuristic function defined with a preferred search direction. However, this path searching strategy led to a significant growth of the complexity of A* when dealing with large-scale motion planning problems. RRT* could find a feasible solution in all 2D environments with various resolutions. However, the average execution time of RRT* increase exponentially in large-scale motion planning problems due to the randomness of

TABLE V
A COMPARISON OF GPMP2-DYN-INTEP AND PSO ON AVERAGE EXECUTION TIME (T), PATH LENGTH (L) AND SAMPLE NUMBER K IN VARIOUS PATH PLANNING SCENARIOS WITHOUT OCEAN CURRENTS. IN EACH SCENARIO, EXPERIMENT WAS TESTED 5 TIME TO CALCULATE THE AVERAGE VALUE.

Map [pixel]	GPMP2-dyn-intp			PSO		
	T [ms]	L [pixel]	K	T [ms]	L [pixel]	K
500x500	**111.3**	**590.8**	30	669.3	727.8	100

Notes: Similarly, GPMP2-dyn-intp only uses the proposed interpolation method (fast GP interpolation) without using anisotropy to deal with ocean currents due to the limited working performance of PSO in ocean environments.

RRT* increases exponentially in large-scale motion planning problems. To be more specific, the step length of RRT* is constant in maps with various resolutions and the probability of the sample point locating on a random position becomes smaller in large-scale maps. As a result, the performance of RRT* would be highly unsuitable for large-scale motion planning problems.

Overall, A* could only find a valid path with a reasonable search space size; while GPMP2-dyn-intp and RRT* were only able to find locally optimal paths without conducting re-planning. Although the paths generated by GPMP2-dyn-intp and RRT* have a similar length at the different resolutions as shown in Table. IV, GPMP2-dyn-intp has an obvious advantage on path smoothness compared with RRT*. The randomness and uniform sampling strategy of RRT* create a number of sparse and loose branches, which leads to redundancy on the path. In Table. IV, the relatively large sample number of RRT* can reflect the redundancy on the path generated by RRT*. So the paths generated by RRT* were comparatively sinuous and not consistent with robot motion constraints. In narrow-passage simulation, RRT* took a very long time to locate the position of the narrow passage due to the randomness and uniform sampling strategy. However, GPMP2-dyn-intp can quickly locate the position of the narrow passage because the GP prior has determined its search direction so it only needs to check if the narrow passage is located inside the shaded area of GP prior.

In addition, a comparative study between the GPMP2-dyn-intp and an evolutionary algorithm, i.e. particle swarm optimisation (PSO), was conducted in a 2D environment. PSO has been successfully applied for intelligent marine vehicles for task allocation and motion planning [45], [46]. Herein, based upon the work in [47], PSO is initialised with a swarm size of 150, an inertia weight of 1, a personal learning coefficient of 1.5, a global learning coefficient of 1.5 and a maximum velocity of 200 as the predefined parameters.

Detailed information regarding the comparison between GPMP2-dyn-intp and PSO is shown in Table. V. We notice that GPMP2-dyn-intp could generate a shorter path with a much faster speed compared with PSO. Moreover, the smoothness of the path generated by PSO is not always satisfactory in the conducted simulations. This might be caused by a relatively small value of the maximum number of iteration. Nevertheless, increasing the maximum number of iteration will increase the time cost. An example that compares the paths generated by GPMP2-dyn-intp and PSO in a multi-obstacle environment is shown in Fig. 13 (c).

D. Benchmark with ocean currents

In this subsection, we demonstrate the improvement of the proposed method, anisotropic GPMP2, over GPMP2 and AFM in various 2D environments including no-obstacle environment, single-obstacle environment, multi-obstacle environment, narrow-passage environment and coastal environment with the presence of ocean currents. The detailed information about the comparison between anisotropic GPMP2, GPMP2 and AFM is shown in Table. VI. This simulation attests to the benefits of anisotropic GPMP2 by metrics such as energy consumption rate, execution time, solution time, path length and number of sample points. More specifically, solution time represents the computational time used for determining the path; on the other hand, execution time represents the total computational time of the method including the solution time and the computational time used for constructing various fields such as a signed distance field and an energy consumption field. We only compared them in maps with resolution ranges from 500×500 to 2000×2000. Because the time cost for generating energy consumption field is relatively long (> 60 s) when the map resolution changes into 5000×5000.

First of all, we qualitatively compared GPMP2, anisotropic GPMP2 without dynamic fast GP interpolation and anisotropic GPMP2 with dynamic fast GP interpolation in the same motion planning problem in coastal environment based on different perspectives as shown in Fig. 14. We notice that the path generated by GPMP2 only avoids obstacles and barely follows ocean currents in Fig. 14 (a) and (b). However, the paths generated by anisotropic GPMP2 without dynamic fast GP interpolation and anisotropic GPMP2 follow ocean currents and attempt to stay in low-consumption region (dark blue region) in Fig. 14 (d), (e), (g) and (h). This demonstrates that the proposed anisotropic energy consumption likelihood function is effective. On the other hand, anisotropic GPMP2 with dynamic fast GP interpolation generated path with fewer sample points, which further optimises the path. This demonstrates that the proposed dynamic fast GP interpolation works effectively. Moreover, the paths generated these three methods are all inside the safe regions (purple regions) of the signed distance field in Fig. 14 (c), (f) and (i), which demonstrate the effectiveness of the collision likelihood function.

We then quantitatively compared anisotropic GPMP2, GPMP2 and AFM on the average energy consumption rate, average execution time, average solution time, average path length and average sample point number in Table. VI and Fig. 15. As shown in Table. VI, anisotropic GPMP2 has the shortest execution time compared with GPMP2 and AFM. This is a result of the following reasons:

- GP-based algorithms (anisotropic GPMP2 and GPMP2) have faster speeds compared with AFM due to solution cost of them are smaller compared with AFM as shown in Fig. 15. To be specific, it would seem the MAP estimation process of GP-based algorithm has a faster convergence speed compared with the gradient descent process of AFM;

TABLE VI
A COMPARISON OF ANISOTROPIC GPMP2, GPMP2 AND AFM ON AVERAGE ENERGY CONSUMPTION RATE (P_e), EXECUTION TIME (T), PATH LENGTH (L) AND SAMPLE NUMBER K IN 20 PATH PLANNING PROBLEMS WITH OCEAN CURRENTS. EACH EXPERIMENT WAS TESTED 5 TIME TO CALCULATE THE AVERAGE VALUE.

Map [pixel]	Problem	Anisotropic GPMP2				GPMP2				AFM			
		P_e [%]	T [ms]	L [pixel]	K	P_e [%]	T [ms]	L [pixel]	K	P_e [%]	T [ms]	L [pixel]	K
500x500	1	14.2	**506.4**	354.4	20	20.6	633.4	**306.3**	30	**10.5**	641.8	364.6	3093
	2	12.4	**542**	503.1	21	19.3	605.2	**393.8**	30	**11.6**	653.9	541.7	4875
	3	**8.2**	**538.7**	630.7	26	17.9	615.2	**486.5**	30	9.3	657.5	682.4	5629
	4	**12.7**	**546.2**	**711.2**	37	17.2	558.1	724.8	45	-	-	-	-
	5	7.5	**572.7**	**362.7**	27	11.2	604.5	414.1	30	**5.9**	643.7	394.3	3558
1000x1000	6	16.4	**2037.8**	651.2	40	21.3	2150.4	**604.7**	60	**10.1**	2221.2	684.5	6158
	7	13.7	**2059.5**	829.5	41	17.8	2143	**781.6**	60	**12.8**	2393.5	853.8	9574
	8	12.5	**2077.7**	1522.4	42	18.5	2167.6	**1108.6**	60	**9.2**	2422.8	1567.4	11221
	9	**15.2**	**2112.2**	929.7	40	16.2	2174.4	**880.2**	70	-	-	-	-
	10	5.1	**1961**	656.2	43	8.9	2024.2	**521.7**	60	**4.3**	2274.8	697.8	6337
2000x2000	11	7.7	**9938.3**	1810.9	80	10.6	9964.5	**1701.9**	120	**6.5**	10019.6	1886.4	12263
	12	12.8	**8823.9**	2060.3	116	21.5	9031.7	**1812.3**	120	**11.7**	9673.9	2075.1	19452
	13	14.5	10364.5	**2459.4**	110	23.5	**10211**	2726.1	120	**9.1**	10213.2	2511.5	22411
	14	**12.1**	**8041.1**	2534.7	100	16.2	8253.6	**2359.5**	130	-	-	-	-
	15	4.2	**8175.1**	1667.2	82	9.6	8260.3	**1315.4**	120	**3.7**	9203.2	1705.3	11483

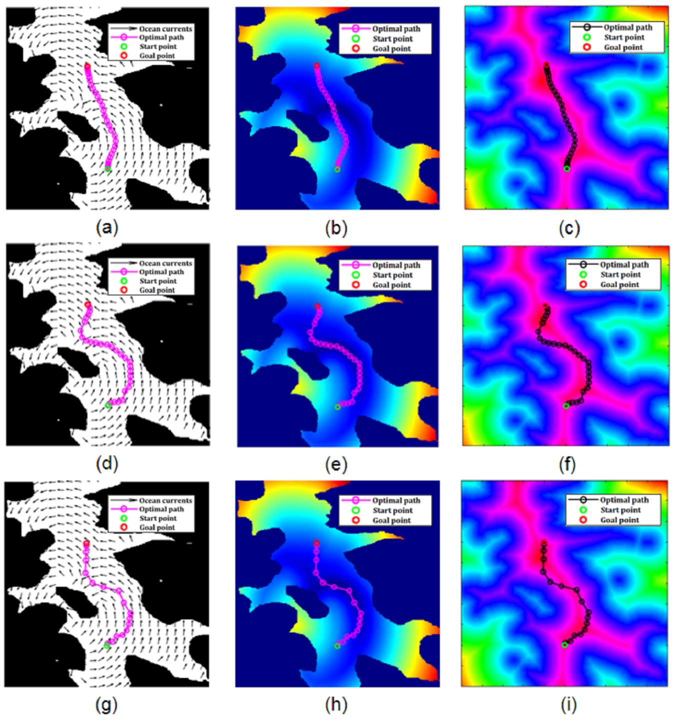


Fig. 14. A comparison between GPMP2 and anisotropic GPMP2 in the same motion planning problem in coastal environment based on different perspectives: (a) demonstrates GPMP2 in binary map; (b) demonstrates GPMP2 in energy consumption field; (c) demonstrates GPMP2 in signed distance field; (d) demonstrates anisotropic GPMP2 without dynamic fast GP interpolation in binary map; (e) demonstrates anisotropic GPMP2 without dynamic fast GP interpolation in energy consumption field; (f) demonstrates anisotropic GPMP2 without dynamic fast GP interpolation in signed distance field; (g) demonstrates anisotropic GPMP2 with dynamic fast GP interpolation in binary map; (h) demonstrates anisotropic GPMP2 with dynamic fast GP interpolation in energy consumption field and (i) demonstrates anisotropic GPMP2 with dynamic fast GP interpolation in signed distance field.

- Dynamic fast GP interpolation decreased the number of sample points, requiring the factor graph of anisotropic GPMP2 to have fewer nodes and calculation steps compared with the factor graph of GPMP2.

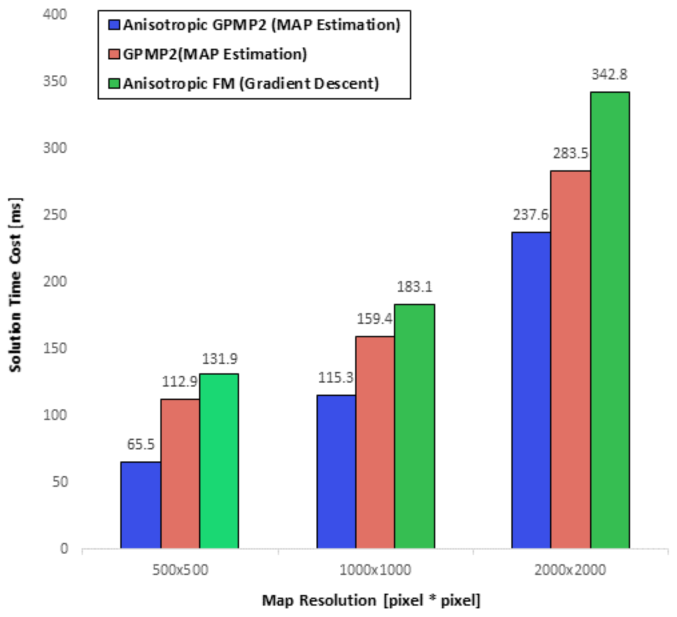


Fig. 15. A comparison of the average solution time cost of anisotropic GPMP2 (MAP estimation), GPMP2 (MAP estimation) and AFM (Gradient descent) in 2D environments with different resolutions. The solution time cost specific refers to the time cost to find an optimal solution after constructing all the necessary fields such as signed distance field and energy consumption field. The solution time of AFM does not exist in problem 4, 9 and 14, because it cannot find a feasible path in narrow-passage environment with ocean currents.

As we can see in Fig. 15, with the scale of the motion planning problem increases, the advantage of GP-based algorithms on solution cost would become more noticeable compared with AFM. This is because the computational complexity of gradient descent dramatically increases in large-scale or high-dimensional motion planning problems.

In Table. VI, the path length of GPMP2 is shorter compared with anisotropic GPMP2 and AFM. This is because the path generated by GPMP2 does not follow ocean currents. Usually, the path would be more sinuous and the average energy consumption rate would be lower when the path attempts to track ocean currents. Compared with anisotropic GPMP2, the

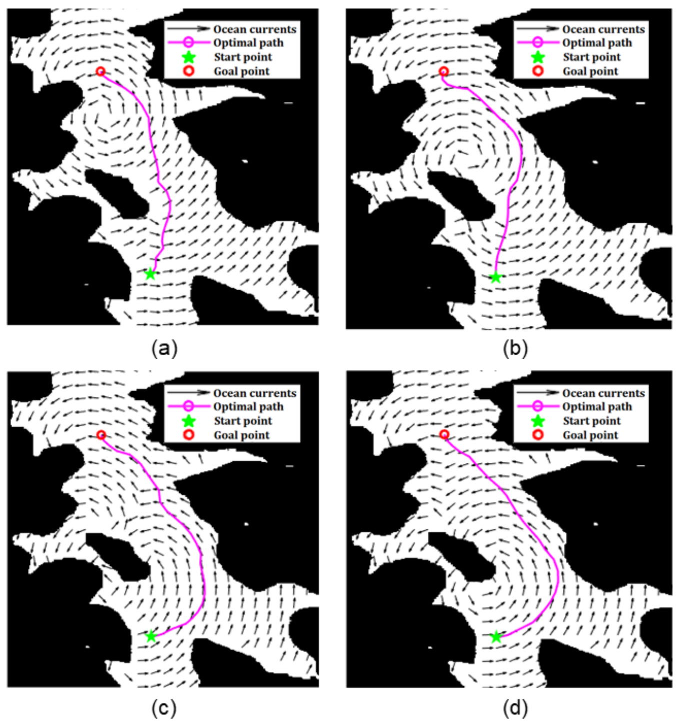


Fig. 16. Re-planning for static platform with time-varying ocean currents: From (a) to (d), the vortex moves from a position close to the platform start point to a position close to its platform goal point. In 10 repeated tests, the execution time of re-planning processes from (a) to (d) were 681.9 ms, 565.1 ms, 532.6 ms and 505 ms respectively.

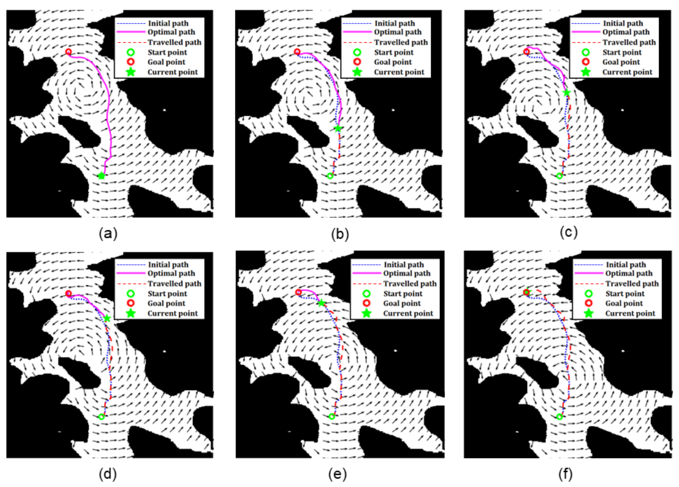


Fig. 17. Re-planning for dynamic platform with time-varying ocean currents: From (a) to (f), the vortex moves from a position close to the platform start point to a position close to the platform goal point, at the mean time, the platform continuously moves towards the goal point. In 10 repeated tests, the execution time of re-planning processes from (a) to (f) were 717.6 ms, 626.1 ms, 636.5 ms, 523 ms, 554.2 ms and 491.1 ms respectively. In this simulation, the generated path varies according to the slight variation of ocean currents at each step; thereby the adaptivity of the proposed method is proved.

path generated by AFM has the lower energy consumption rate as shown in Table. VI, which means the path generated by AFM was better at tracking ocean currents. So the path of AFM would be more sinuous and the path length would be longer as shown in Table. VI. In Table. VI, the number of sample points of AFM is greater than the number of sample points of GP-based algorithms. This is because the paths generated by GP-based algorithms are continuous-time functions. We can simply query a reduced number of sample points on the path at specific moments and then use straight lines to connect these sample points to reconstruct the path. In Fig. 13, the path generated by AFM advantageously tracks ocean currents at each waypoint; however, the path generated by anisotropic GPMP2 does not track ocean currents at the waypoints around the start and goal points. This is because anisotropic GPMP2 has an initialised straight-line GP prior, which locks the direction of the path at start and goal points as shown in Fig. 2. Hence the path generated by anisotropic GPMP2 would always be narrow on both ends and wide in the central portion. Compared with AFM, the most significant advantage of anisotropic GPMP2 is the influence of ocean currents on the path that can be adjusted by adjusting parameter σ_e. Specifically, we can encourage the path to keep tracking ocean currents in such a way as to reduce energy consumption when necessary; on the other hand, we can also decrease the influence of ocean currents so that the path can diverge from the trend of ocean currents if the path has entered into a local minimum. In narrow-passage simulations as detailed in Table. VI (problem 4, 9 and 14), the path generated by AFM cannot find a feasible path as the trend of ocean currents did not align with the narrow passage. However, the path generated by anisotropic GPMP2 can pass the narrow passage based on the guidance of the straight-line GP prior, and, in the meantime, track ocean currents as far as possible with a suitable value for parameter σ_e.

E. Re-planning with time-varying ocean currents

This subsection qualitatively demonstrates the capability of the proposed method in re-planning problems in the coastal environment with time-varying ocean currents. This simulation aims to show how quickly the proposed method would react to changes of ocean currents. Furthermore, the reaction time of each re-planning was recorded to demonstrate the efficiency. We used anisotropic GPMP2 to implement two types of re-planning problems including (i) re-planning for static USV with time-varying ocean currents and (ii) re-planning for dynamic USV with time-varying ocean currents.

1) Re-planning for a static platform: In this re-planning problem, the start and goal points as well as the current position of the platform are static. The time-varying ocean currents generate a vortex that has influence from a position close to the platform start point to a position close to the platform goal point as shown in Fig. 16. Once the variation of the ocean currents is detected, anisotropic GPMP2 would perform re-planning for the static platform so that a feasible path is generated to adapt to the updated ocean currents. The average execution time of this re-planning problem across 10 repeated tests was 571.2 ms, which means anisotropic GPMP2 demonstrates a relatively good performance in re-planning for static platforms.

2) Re-planning for dynamic platform: In the re-planning problem for dynamic platform, the start and goal points of

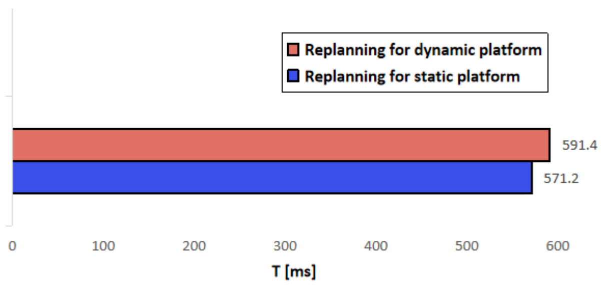


Fig. 18. A comparison of the average execution time of anisotropic GPMP2 in different re-planning problems.

the platform are static, but the current position of the platform would keep updating. The time-varying ocean currents generated a vortex that has influence from a position close to the platform start point to a position close to the platform goal point while the platform continuously moves towards the goal point as shown in Fig. 17. Once the variation of the ocean currents is detected, anisotropic GPMP2 would perform re-planning for the dynamic platform so that a feasible path is generated to adapt to the updated ocean currents. The average execution time of this re-planning problem across 10 repeated tests was 591.4 ms as shown in Fig. 18, which means anisotropic GPMP2 has a relatively good performance in re-planning for dynamic platforms.

As summarised in our previous work on testing a USV in practical environments [48], to ensure a real-time performance, the minimum control signal updating (trajectory replanning) frequency for a high-speed ($\geq$ 40 knots) USVs is 2 Hz. Based on this, we can also infer that the minimum frequency requirements for medium-speed (10 - 40 knots) and low-speed ($\leq$ 10 knots) USVs should be more than 2 Hz. In both re-plannings for static and dynamic platforms, the average execution times of re-planning processes of AGPMP2, with the PC configuration listed in Table III, were less than 600 ms, indicating that the updating frequency of the proposed method is about 1.7 Hz. This updating frequency can evidently meet the real-time requirement of the selected USV (WAM-V 20).

VII. Implementation in ROS

This section demonstrates the proposed method in the ROS environment to simulate autonomous transportation mission and autonomous inspection mission in the real world. We use these two different practical scenarios to demonstrate the versatility of the proposed method.

The system structure of the proposed method with the Gazebo simulation is shown in Fig. 19. The proposed method was run on MATLAB and connected with Gazebo through ROS nodes.

A. *Autonomous transportation mission*

In Gazebo, an almost real simulation world with sunlight, wind, ocean currents, gravity and buoyancy was created and a series of islands with different sizes and shapes as well as WAM-V 20 USV were placed inside the simulation world. The locations of the islands are shown in Fig. 20 (a); on the

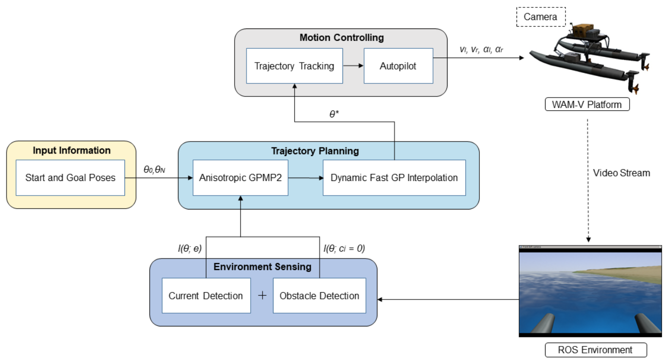


Fig. 19. Structure of the proposed motion planning system that combines anisotropic GPMP2 and WAM-V platform in marine environment. θ_0 is the start pose, θ_N is the goal pose, $l(\theta, e)$ is energy consumption likelihood function, $l(\theta, c_i = 0)$ is collision likelihood function, θ^* is the optimal path, v_l is the rotation speed of the left thruster, v_r is the rotation speed of the right thruster, α_l is the rotation angle of the left thruster and α_r is the rotation angle of the right thruster.

other hand, the start point and goal point of WAM-V 20 USV and the dangerous region on the terrain are shown in Fig. 20 (b). Furthermore, a camera was mounted at the front of the WAM-V 20 USV to detect obstacles and record videos. The video stream from the front camera was transmitted to and displayed on the Rviz interface through the WAM-V Camera node (/wam-v/sensors/cameras/front-camera/image-raw) as shown in the bottom right corner of Fig. 19.

In the transportation mission in Gazebo, the WAM-V 20 USV transported a parcel from the start point (50, 50) marked by a green buoy to the goal point (250, 450) marked by a red buoy. During this process, the USV moved along the big island boundary and made a detour at the dangerous region (blue region) in Fig. 20 (b) to avoid collision with the big island. Fig. 21 demonstrates the storyboards of the transportation mission from both the first-person and third-person perspectives.

B. *Autonomous inspection mission*

Similarly, a high-fidelity offshore wind farm inspection scenario was selected. Practical aspects including sunlight, wind, ocean currents, gravity and buoyancy were incorporated together with several wind turbines. Fig. 22 (a) details the locations of the turbines, while Fig. 22 (b) details the start point and goal point of WAM-V 20 USV. Again, a camera was mounted at the front of the WAM-V 20 USV to detect obstacles and record videos.

During the inspection mission in Gazebo, by following the trajectory generated by AGPMP2, the WAM-V 20 USV moved from the start point (50, 50) marked by a green buoy to the goal point (450, 450) marked by a red buoy. As the USV navigates, the vessel can undertake an inspection of the wind turbine areas and monitor any damages to the turbines. Fig. 23 demonstrates the storyboards of the inspection mission from both the first-person and third-person perspectives.

VIII. Conclusion

The potential impact of the work in this paper is successfully extending GP-based motion planning into fluid environments

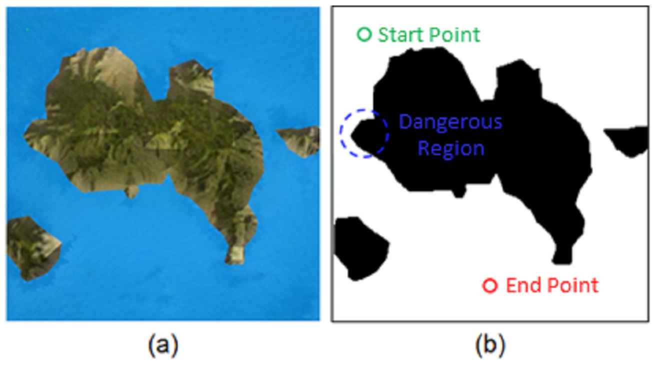


Fig. 20. Transportation mission map in Gazebo: (a) demonstrates the top view of the map and (b) demonstrates the binary image of this map, which indicates the start point (green), goal point (red) and dangerous region (blue).

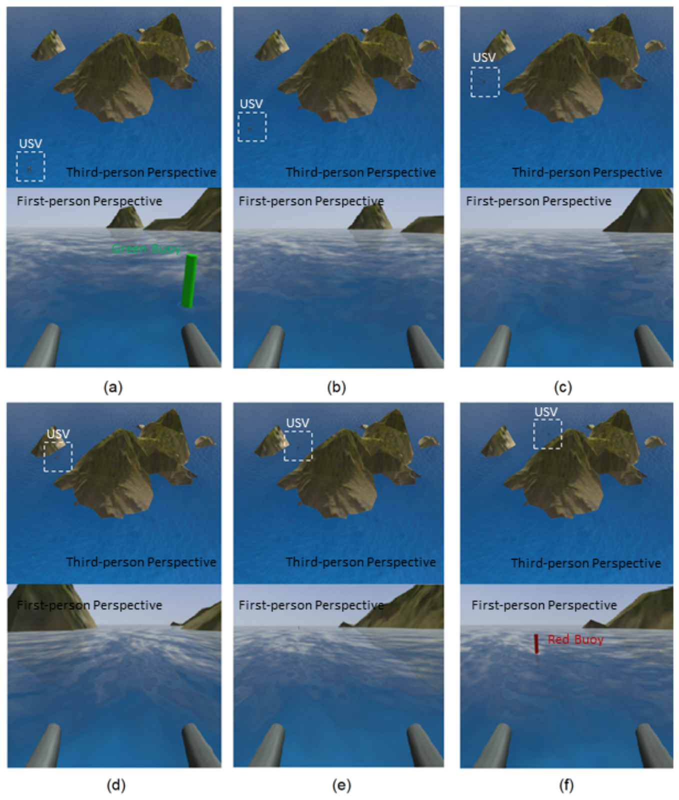


Fig. 21. The storyboards of the transportation mission based on the first-person perspective (at the lower part of each figure) and third-person (at the upper part of each figure) synchronously: From (a) to (f), the images demonstrate the locations of the USV at time equals to 1 s, 6 s, 11 s, 16 s, 21 s and 26 s, respectively. To better indicate the start and goal positions, a green buoy was placed at the start point of the USV and a red buoy was placed at the goal point of the USV.

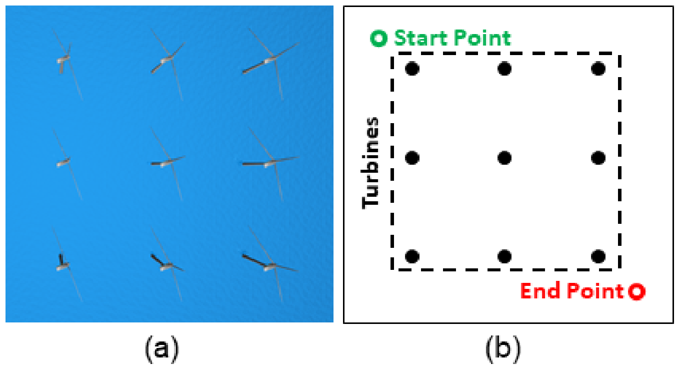


Fig. 22. Inspection mission map in Gazebo: (a) demonstrates the top view of the map and (b) demonstrates the binary image of this map, which indicates the start point (green), goal point (red) and turbines (black).

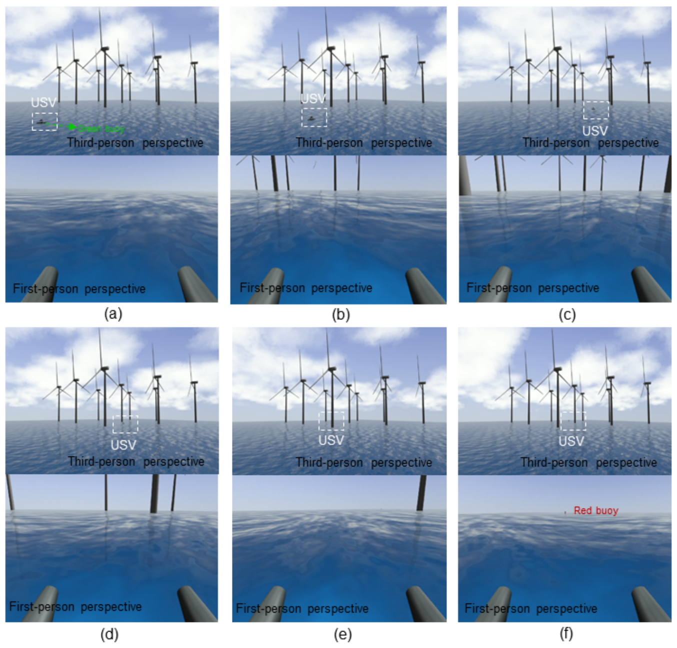


Fig. 23. The storyboards of the inspection mission based on the first-person perspective (at the lower part of each figure) and third-person (at the upper part of each figure) synchronously: From (a) to (f), the images demonstrate the locations of the USV at time equals to 1 s, 5 s, 10 s, 15 s, 20 s and 25 s, respectively. To better indicate the start and goal positions, a green buoy was placed at the start point of the USV and a red buoy was placed at the goal point of the USV.

such as ocean surfaces. Specifically, this paper presents a motion planning algorithm based on continuous-time GP, which can deal with time-varying ocean currents and obstacles simultaneously. By introducing (i) energy consumption likelihood function and (ii) dynamic fast GP interpolation, we were able to generate trajectories to avoid vortexes and follow ocean currents while at the same time removing redundant sample points to improve execution time and more effectively avoid obstacles. We derived the energy consumption field by measuring ocean current dynamics, so it can reduce energy consumption caused by ocean currents and assist USVs in avoiding hazardous areas, such as vortexes. We employed the dynamic fast GP interpolation method by taking benefits of GPs that can be parameterised by only a sparse set of support states, while the generated trajectory can be still queried at any moment of interest. By adjusting the sampling interval in each sub-search region, the proposed method can avoid obstacles by generating smoother paths for course correction, while reducing its execution time and path length.

The subjects of future research are: (i) further optimising and enriching our ROS environment to enable USVs to perform different tasks in the environment; (ii) designing a controller to simulate the path tracking process of real USVs.

To be specific, we aim to combine the motion planning and control of USVs in the next step. Hence, the experimental results of the proposed motion planning method would be more practical and can be used in the real-world environment.

Acknowledgment

This work is partially supported by Royal Society (IEC-NSFC-191633). This work was also made possible through a collaboration between UCL and Rutgers University, which was partially supported by the International Collaborative Research Grant from Rutgers Global at Rutgers, The State University of New Jersey. The authors are indebted to Mr. Konrad Yearwood for his valuable critique of this paper.

References

[1] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[2] Y. Ma, Z. Nie, S. Hu, Z. Li, R. Malekian, and M. Sotelo, "Fault detection filter and controller co-design for unmanned surface vehicles under dos attacks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1422–1434, 2020.

[3] Y. Ma, Z. Nie, Y. Yu, S. Hu, and Z. Peng, "Event-triggered fuzzy control of networked nonlinear underactuated unmanned surface vehicle," *Ocean Engineering*, vol. 213, p. 107540, 2020.

[4] R. Song, W. Liu, Y. Liu, and R. Bucknall, "A two-layered fast marching path planning algorithm for an unmanned surface vehicle operating in a dynamic environment," in *OCEANS 2015-Genova*. IEEE, 2015, pp. 1–8.

[5] M. Mukadam, X. Yan, and B. Boots, "Gaussian process motion planning," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 9–15.

[6] J. Dong, M. Mukadam, F. Dellaert, and B. Boots, "Motion planning as probabilistic inference using gaussian processes and factor graphs." in *Robotics: Science and Systems*, vol. 12, 2016, p. 4.

[7] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 4569–4574.

[8] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[9] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[10] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[11] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[12] A. Stentz, "Optimal and efficient path planning for partially known environments," in *Intelligent unmanned ground vehicles*. Springer, 1997, pp. 203–220.

[13] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, 1996.

[14] T. Lolla, M. P. Ueckermann, K. Yiğit, P. J. Haley, and P. F. Lermusiaux, "Path planning in time dependent flow fields using level set methods," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 166–173.

[15] S. MahmoudZadeh, D. M. Powers, and A. M. Yazdani, "A novel efficient task-assign route planning method for auv guidance in a dynamic cluttered environment," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016, pp. 678–684.

[16] M. Dorigo, A. Colorni, and V. Maniezzo, "Distributed optimization by ant colonies," 1991.

[17] D. Whitley, "A genetic algorithm tutorial," *Statistics and computing*, vol. 4, no. 2, pp. 65–85, 1994.

[18] L. Zhang, L. Zhang, S. Liu, J. Zhou, and C. Papavassiliou, "Three-dimensional underwater path planning based on modified wolf pack algorithm," *IEEE Access*, vol. 5, pp. 22 783–22 795, 2017.

[19] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[20] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," 1998.

[21] D. Li, P. Wang, and L. Du, "Path planning technologies for autonomous underwater vehicles-a review," *IEEE Access*, vol. 7, pp. 9745–9768, 2018.

[22] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Informed rrt*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 2997–3004.

[23] ——, "Batch informed trees (bit*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 3067–3074.

[24] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2. IEEE, 2000, pp. 995–1001.

[25] J. Meng, V. M. Pawar, S. Kay, and A. Li, "Uav path planning system based on 3d informed rrt* for dynamic obstacle avoidance," in *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2018, pp. 1653–1658.

[26] Trajectory Optimisation, 2020. [Online]. Available: https://en.wikipedia.org/wiki/Trajectory_optimization

[27] L. Petrović, J. Peršić, M. Seder, and I. Marković, "Cross-entropy based stochastic optimization of robot trajectories using heteroscedastic continuous-time gaussian processes," *Robotics and Autonomous Systems*, vol. 133, p. 103618, 2020.

[28] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 489–494.

[29] J. Schulman, J. Ho, A. X. Lee, I. Awwal, H. Bradlow, and P. Abbeel, "Finding locally optimal, collision-free trajectories with sequential convex optimization." in *Robotics: science and systems*, vol. 9, no. 1. Citeseer, 2013, pp. 1–10.

[30] E. Huang, M. Mukadam, Z. Liu, and B. Boots, "Motion planning with graph-based trajectories and gaussian process inference," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5591–5598.

[31] M. Mukadam, J. Dong, X. Yan, F. Dellaert, and B. Boots, "Continuous-time gaussian process motion planning via probabilistic inference," *The International Journal of Robotics Research*, vol. 37, no. 11, pp. 1319–1340, 2018.

[32] M. Mukadam, J. Dong, F. Dellaert, and B. Boots, "Simultaneous trajectory estimation and planning via probabilistic inference," in *Robotics: Science and Systems*, 2017.

[33] J. Dong, M. Mukadam, B. Boots, and F. Dellaert, "Sparse gaussian processes on matrix lie groups: A unified framework for optimizing continuous-time trajectories," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6497–6504.

[34] M. Rana, M. Mukadam, S. R. Ahmadzadeh, S. Chernova, and B. Boots, "Towards robust skill generalization: Unifying learning from demonstration and motion planning," in *Intelligent robots and systems*, 2018.

[35] F. Marić, O. Limoyo, L. Petrović, T. Ablett, I. Petrović, and J. Kelly, "Fast manipulability maximization using continuous-time trajectory optimization," *arXiv preprint arXiv:1908.02963*, 2019.

[36] X. Yan, V. Indelman, and B. Boots, "Incremental sparse GP regression for continuous-time trajectory estimation and mapping," *Robotics and Autonomous Systems*, vol. 87, pp. 120–132, 2017.

[37] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch continuous-time trajectory estimation as exactly sparse gaussian process regression." in *Robotics: Science and Systems*, vol. 10. Citeseer, 2014.

[38] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of applied mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[39] J. A. Sethian and A. M. Popovici, "3-d traveltime computation using the fast marching method," *Geophysics*, vol. 64, no. 2, pp. 516–523, 1999.

[40] S. Jbabdi, P. Bellec, R. Toro, J. Daunizeau, M. Pélégrini-Issac, and H. Benali, "Accurate anisotropic fast marching for diffusion-based geodesic tractography," *International journal of biomedical imaging*, vol. 2008, 2008.

[41] R. Song, Y. Liu, and R. Bucknall, “A multi-layered fast marching method for unmanned surface vehicle path planning in a time-variant maritime environment,” *Ocean Engineering*, vol. 129, pp. 301–317, 2017.

[42] Q. Lin, “Enhancement, extraction, and visualization of 3d volume data,” Ph.D. dissertation, Linköping University Electronic Press, 2003.

[43] K. Dohan and N. Maximenko, “Monitoring ocean currents with satellite sensors,” *Oceanography*, vol. 23, no. 4, pp. 94–103, 2010.

[44] F. Dellaert, “Factor graphs and gtsam: A hands-on introduction,” Georgia Institute of Technology, Tech. Rep., 2012.

[45] Z. Zeng, K. Sammut, A. Lammas, F. He, and Y. Tang, “Efficient path re-planning for auvs operating in spatiotemporal currents,” *Journal of Intelligent & Robotic Systems*, vol. 79, no. 1, pp. 135–153, 2015.

[46] A. Abbasi, S. MahmoudZadeh, and A. Yazdani, “A cooperative dynamic task assignment framework for cotsbot auvs,” *IEEE Transactions on Automation Science and Engineering*, 2020.

[47] S. MahmoudZadeh, A. M. Yazdani, K. Sammut, and D. M. Powers, “Online path planning for auv rendezvous in dynamic cluttered undersea environment using evolutionary algorithms,” *Applied Soft Computing*, vol. 70, pp. 929–945, 2018.

[48] J. Zhuang, L. Zhang, B. Wang, Y. Su, H. Sun, Y. Liu, and R. Bucknall, “Navigating high-speed unmanned surface vehicles: System approach and validations,” *Journal of Field Robotics*, vol. 38, no. 4, pp. 619–652, 2021.