

© 2019 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Optimizing autonomous underwater vehicle routes with the aid of high resolution ocean models

Miguel Aguiar*, João Borges de Sousa*, João Miguel Dias[†], Jorge Estrela da Silva[†], Renato Mendes^{‡§} and Américo S. Ribeiro[‡]

Abstract—In underwater vehicle operations in areas such as estuaries, vehicles may face currents with magnitudes equal to or exceeding the vehicle’s maximum forward speed. We propose a method which generates vehicle routes taking into account ocean current forecasts from high resolution ocean models, in order to both take advantage of the ocean current velocity and avoid its negative effects. We formulate the problem in an optimal control setting and derive the associated Hamilton-Jacobi-Bellman partial differential equation (PDE). We solve this PDE using a parallelized C++ implementation of a numerical method which allows us to obtain the solution in a few minutes on a mainstream computer. After obtaining the solution of the PDE, optimal trajectories with any initial condition can be computed efficiently. The method is illustrated using data from high-resolution ocean models of the Sado river estuary in Portugal. Two mission scenarios are analyzed, which highlight the influence of ocean currents on optimal trajectories and the benefits of considering ocean current forecasts in mission planning.

Index Terms—underwater robotics, trajectory generation, control, ocean modeling

I. INTRODUCTION

Ocean currents play an important role in many underwater vehicle operation scenarios, as they can have beneficial effects such as an increase in speed and reduced energy expenditure, but also hinder or make impossible the attainment of the operational goal. This is particularly true in long duration operations in regions with tidal-driven currents, such as estuaries, where ocean current magnitudes can equal or exceed the maximum forward speed of a typical underwater vehicle. Mission planning and trajectory generation for autonomous underwater vehicles (AUVs) should take ocean currents into account, in order to obtain vehicle routes which take the most advantage of the flow velocity while minimizing its negative effects.

We consider the problem of planning an optimal planar route for an underwater vehicle traveling from an initial position to a target region. Independently of the metric used to evaluate the quality of trajectories, the best direction of motion at a given point depends not only on the value of the ocean current at the given location and time, but on all future values of the

ocean current along the vehicle’s trajectory. Note also that due to the time-dependent nature of the ocean currents, the optimal trajectory from any particular initial position will depend on the deployment time.

A number of distinct approaches to this and related problems have been considered in the literature, such as graph search methods, level set methods, and optimal control methods. Our paper fits into the latter set of approaches, as we employ a simple kinematic model of the vehicle to formulate the problem as an optimal control problem. By applying the principle of optimality of dynamic programming, the optimal control problem is converted to the problem of solving a first order nonlinear partial differential equation known as the Hamilton-Jacobi-Bellman Equation (HJBE).

It is well-known that most HJBEs arising from nontrivial optimal control problems have no analytical solution, so one must resort to numerical methods. Numerical methods for dynamic programming, however, typically suffer from the so-called ‘curse of dimensionality’, meaning that the complexity increases significantly with the increase in the number of dimensions. In a previous paper [11] we reported large computation times (approx. 5 hours) for a realistic benchmark problem, using a MATLAB software package. Here we show that our parallelized C++ implementation of a fast sweeping method [2], [9] drastically reduces the computation time.

The rest of the paper is organized as follows. Section II introduces some notation and describes the problem. In Section III, we review existing approaches to this and related problems. Section IV discusses the approach and the numerical method implementation. Two numerical examples using data from ocean models of the Sado estuarine region in Portugal are examined in Section V. Finally, Section VI presents the conclusions and future research directions.

II. PROBLEM DESCRIPTION

In this paper we consider only planar trajectories, assuming that the AUV travels at a known and constant depth. However, the ideas apply just as well to planning in three-dimensional environments.

Let $\mathbf{x}(t) \in \mathbf{R}^2$ denote the horizontal position of the vehicle at time $t \in \mathbf{R}$. The vehicle departs from some $\mathbf{x}(t_0) = \mathbf{x}_0$ and must reach an area of interest represented by the set $\Omega \in \mathbf{R}^2$. Since there will usually be more than one trajectory which the vehicle can take from its starting position to the target area,

*LSTS, Faculty of Engineering, University of Porto, Porto, Portugal. Email: {m.ag, jtasso}@fe.up.pt

§CIIMAR, University of Porto, Porto, Portugal.

†NMEC, CESAM, DFis, University of Aveiro, Aveiro, Portugal. Email: {joao.dias, americosribeiro, rpsm}@ua.pt

‡Instituto Superior de Engenharia do Porto, Polytechnic Institute of Porto, Porto, Portugal. Email: jes@isep.ipp.pt

we consider the following cost function which allows us to compare the trajectories:

$$\mathcal{J}(\mathbf{x}(\cdot)) = q(\mathbf{x}(T)) + \int_{\tau_0}^T g(\mathbf{x}(\tau))d\tau$$

where T is the arrival time at the target region Ω , $g : \mathbf{R}^2 \rightarrow \mathbf{R}_{>0}$ and $q : \Omega \rightarrow \mathbf{R}_{\geq 0}$. If g is constant and q is zero everywhere, then the cost function is the time taken to arrive at the target.

The problem is then to find a trajectory from the vehicle's starting position to the target set which minimizes \mathcal{J} , given a forecast of the ocean currents over the operational area and a given mission time window.

In most environments there will be geographic constraints and other static obstacles which must be taken into account so that the generated trajectories make sense. In that case the trajectory should be chosen as the minimizer of \mathcal{J} among those trajectories which take into account those constraints.

III. RELATED WORK

In this section our focus is on algorithms for transit routes, so we do not mention the large body of work on path planning and trajectory generation for adaptive sampling, mapping or obstacle avoidance, for instance. Additionally, we concentrate on algorithms which take advantage of information about the value of the ocean currents on the operational area.

Inanc et al. [3] consider an optimal control formulation of the problem of generating a planar trajectory between two fixed waypoints. The optimal control problem is converted to a nonlinear programming problem via a spline parametrization. Numerical examples are given using current data from HF radar measurements. An improvement of this approach is presented by Zhang et al. [5] to account for time-varying currents.

Kruger et al. [4] use a gradient descent algorithm to obtain an optimal three-dimensional path between two waypoints which minimizes a linear combination of the expended energy and the traveling time, assuming a holonomic model. A numerical example using a simulation of an estuarine flow is presented.

As the optimization problem which has to be solved in both these approaches is nonconvex, there is no guarantee on the global optimality of solutions, which can be a problem when there are multiple paths around an obstacle, for instance. In comparison, the approach presented in this paper finds the global optimum even in the presence of obstacles.

Lolla et al. [8] present a level set method-based approach to minimum-time trajectory generation, considering fixed endpoints and starting time. The authors show that the optimal trajectory can be recovered from the solution of a partial differential equation.

A similar approach to ours is that of Rhoads et al. [7], who solve the HJBE corresponding to a minimum time problem using an extremal field method. The method has the drawback that for targets with more than one connected component the computation must be done separately for each component. The authors report a computation time of two to three hours for a realistic example using data from an ocean model.

IV. APPROACH

A. Dynamic Programming

In order to formulate the problem as an optimal control problem, a motion model must be chosen, which determines the set of admissible trajectories over which the minimization of the cost function will be performed. We use a kinematic model

$$\dot{\mathbf{x}}(t) = \mathbf{u}(t) + \mathbf{v}(t, \mathbf{x}(t)) \quad (1)$$

where $\dot{\mathbf{x}}(t) \in \mathbf{R}^2$ denotes the vehicle's velocity, $\mathbf{u}(t) \in \mathbf{R}^2$ is the velocity resulting from the vehicle's propulsion system, which satisfies $|\mathbf{u}(t)| < r$, where $r > 0$ is the vehicle's maximum forward speed, and \mathbf{v} is a time-varying vector field representing the ocean current velocity.

Given an initial condition $\mathbf{x}(\tau) = \boldsymbol{\xi}$ and a control function \mathbf{u} , the resulting trajectory $\mathbf{x}_{\tau, \boldsymbol{\xi}, \mathbf{u}}$ is determined by (1). The cost \mathcal{J} can thus be rewritten as

$$\mathcal{J}(\tau, \boldsymbol{\xi}, \mathbf{u}) = \mathcal{J}(\mathbf{x}_{\tau, \boldsymbol{\xi}, \mathbf{u}}).$$

The value function V is then defined as the function which assigns to each initial condition the cost of the optimal trajectory with that initial condition, i.e.

$$V(\tau, \boldsymbol{\xi}) = \inf \{ \mathcal{J}(\tau, \boldsymbol{\xi}, \mathbf{u}) \mid \mathbf{u} \in \mathcal{U}_{\tau} \}$$

where \mathcal{U}_{τ} is the set of measurable functions $\mathbf{u} : [\tau, +\infty) \rightarrow \mathbf{R}^2$ which satisfy $|\mathbf{u}(t)| < r$ almost everywhere.

Applying the principle of optimality, it can be shown [1] that V satisfies the Hamilton-Jacobi-Bellman equation

$$r |V_{\boldsymbol{\xi}}| - (V_{\tau} + V_{\boldsymbol{\xi}} \cdot \mathbf{v}) = g \quad (2)$$

with the boundary condition

$$V(\tau, \boldsymbol{\xi}) = q(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in \Omega.$$

The principle of optimality also shows that the optimal control $\mathbf{u}(t)$ if the vehicle is at $\mathbf{x}(t)$ is

$$\mathbf{u}(t) = -r \frac{V_{\boldsymbol{\xi}}(t, \mathbf{x}(t))}{|V_{\boldsymbol{\xi}}(t, \mathbf{x}(t))|}.$$

Thus, if V is known, an optimal trajectory with initial condition $\mathbf{x}(\tau) = \boldsymbol{\xi}$ can be obtained by integrating (1) with this choice of \mathbf{u} , starting from $\mathbf{x}(\tau) = \boldsymbol{\xi}$ and stopping when the target set Ω is reached. Note that this implies that after a single computation of the value function we are able to compute optimal trajectories starting from any position and at any deployment time.

When there are obstacles in the operational area, they can be included in the formulation in the following straightforward way. If the obstacles are represented by a set \mathcal{K} , we add \mathcal{K} to the target set, so that the new target set is $\mathcal{K} \cup \Omega$, and set $q(\boldsymbol{\xi}) = M, \boldsymbol{\xi} \in \mathcal{K}$ where M is a large value, so that if the trajectory satisfying $\mathbf{x}(\tau) = \boldsymbol{\xi}$ enters the obstacle set we will have $V(\tau, \boldsymbol{\xi}) \geq M$.

B. Numerical method

Equation 2 is a first order nonlinear partial differential equation (PDE), so it has no analytical solution except in very particular cases. In our case, since the values of the ocean current v are given as numerical data, we have no choice but to rely on a numerical method to approximate V .

Given the nature of (2), the iterative fast sweeping method described in Kao et al. [2] is the most adequate. The computation is limited to some hypercube $(\tau, \xi) \in R = [t_{\min}, t_{\max}] \times D$, where D is the operational area and $[t_{\min}, t_{\max}]$ is the mission time window. The computational domain R is discretized into a regular grid by selecting appropriate resolutions for each dimension. The PDE is discretized using first-order finite differences and a Lax-Friedrichs artificial viscosity term. Using this discretization one can derive a formula which updates the value of a grid cell using the values of the neighboring grid cells.

The numerical value function is initialized with an upper bound for V . In our case, we have the upper bound

$$V(\tau, \xi) \leq Q + (t_{\min} - t_{\max}) \cdot G \quad (3)$$

where Q and G are upper bounds on q and g , respectively. The value at grid cells which are inside the target set Ω is initialized with the corresponding value of q , as that is the true value of the solution.

In each iteration, the algorithm iterates through the gridpoints in different orders, updating them using the aforementioned formula. One can iterate through all the gridpoints in 2^3 different orders, according to whether one ascends or descends in each dimension. The motivation for updating in all possible orders is to capture all possible directions of motion of the optimal trajectories.

The algorithm stops when the difference in the value between two iterations on all the gridpoints is less than a given threshold.

C. Implementation

We implemented a numerical solver based on this numerical method in C++, using shared memory parallelism to speed up the computation.

The fast sweeping algorithm is parallelized using the hyperplane stepping method described in Detrixhe et al. [9]. This method is based on the observation that for first-order finite differences there is a straightforward way to split the gridpoints into a finite number of disjoint sets such that the points in each set can be updated in parallel. As such, near optimal speedup is achieved for large grids, i.e., the execution time is approximately equal to the execution time of the serial algorithm divided by the number of processors.

These sets are precalculated during program startup, and the points in each set are divided between the available processors in each iteration. A thread pool implementation based on the multithreading facilities available in the C++ standard library¹ is used to execute the parallel tasks.

¹<https://github.com/progschj/ThreadPool>

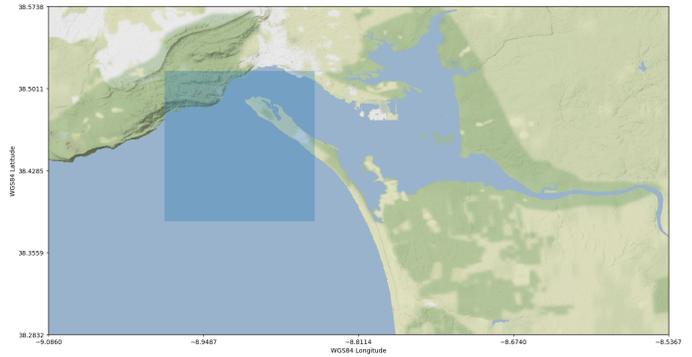


Fig. 1. The Sado estuary³

The implementation is not coupled to our particular problem, and can be used to solve any static first order Hamilton-Jacobi equation. The source code is also independent of the number of dimensions. However, the number of dimensions is set at compilation time, which enables compiler optimizations such as loop unrolling, and allows most arrays to be allocated on the stack.

The data input and output is done using the HDF5 file format, through a C++ API to the HDF5 C library².

The solver source code is available at <https://github.com/mcpc/marlin>, together with usage examples.

V. NUMERICAL EXAMPLES

In this section we use data from the Sado estuary in Portugal (Fig. 1) to plan trajectories for two mission scenarios:

- Mission 1 - the vehicle starts at sea and should enter the estuary and stop at a safe zone inside the river;
- Mission 2 - the vehicle starts inside the river and should exit towards the sea and reach a survey area in the sea.

The operational area for the two missions is highlighted in blue in Fig. 1, and it is roughly 12 km by 15 km in size.

The data used for these examples was produced by a high resolution model of the area of the Sado river estuary in September 2018. For a detailed description of the model used to generate the data, see our previous paper [11] or Ribeiro et al. [10]. The values of the ocean current velocity are defined on a spatially curvilinear grid with a mean resolution of 100 meters, with a temporal sampling rate of 10 minutes. The data is given for various depth layers, but we use only the topmost layer, as we assume the vehicle will travel at a constant depth of around 1 meter. Since the solver expects a regular grid, the data must be interpolated in space to a regular grid before we can use the data, which we do using linear interpolation. The value function was computed on a grid with a 50 meter resolution in space and a temporal resolution of 10 minutes. Since the flow is tidal-driven, a time window of 12

²<https://github.com/ess-dmcs/h5cpp>

³Map tiles by Stamen Design, under CC BY 3.0. Data by OpenStreetMap, under ODbL.

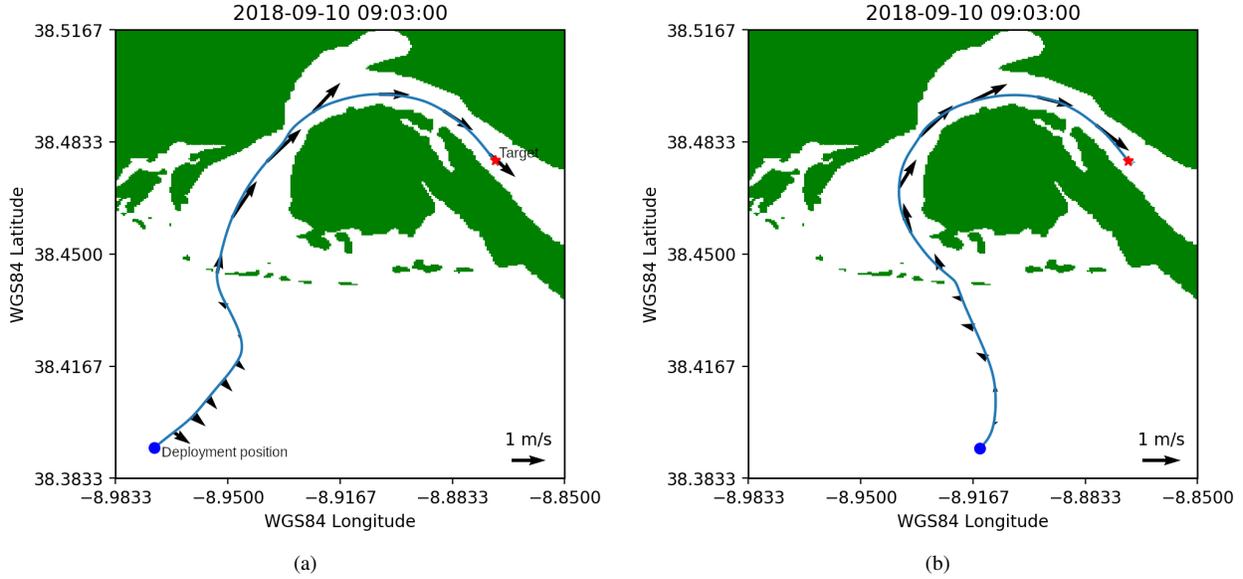


Fig. 2. Two optimal trajectories with different deployment positions - Mission 1

hours is considered for each mission. Hence the resulting grid on which V is defined has approximately 5 million points.

The operational area contains land and shallow regions which must be included in the formulation in order to exclude trajectories entering these regions. Using the GSHHG shoreline dataset⁴, an obstacle map can be created. There are also some shallow regions which are not covered by the shoreline data. To include these in the obstacle map, we use a naval chart and manually approximate regions where the seafloor depth is lower than 2 meters by polygons. Using the coordinates of the vertices of the polygons one can check which grid cells are inside the polygon.

For both Mission 1 and Mission 2, the integral cost component g is constant and equal to 1, and q is identically zero on the target set, so that the cost is equal to the time taken to reach the target. The value of r (the maximum forward speed of the vehicle) is taken to be 1 m/s.

The value function is computed on a laptop running Linux with an Intel^(R) CoreTM i7-8550U @ 1.80 GHz processor with eight hardware threads. The solver was compiled using the GCC C++ compiler version 9.1.0 with level 3 optimizations. The computation of the value function takes approximately two to four minutes of wall clock time in both cases.

A. Mission 1

The target set for the first mission is a sphere with a 100 meter radius around the point with coordinates (38°N 28' 37.73", 8°W 52' 12.57").

Fig. 2 shows two optimal trajectories with different deployment positions and the same deployment time. These are calculated by integrating (1) using Euler's method. The gradient of the value function is calculated on the gridpoints using a

finite difference formula, and interpolated to points off the grid. The initial position is shown in blue, the arrival position at the target is marked by the red star, arrows represent the values of the ocean current velocity along the trajectory and points in the obstacle map are colored green. Note that the two trajectories are calculated from a single computation of the value function.

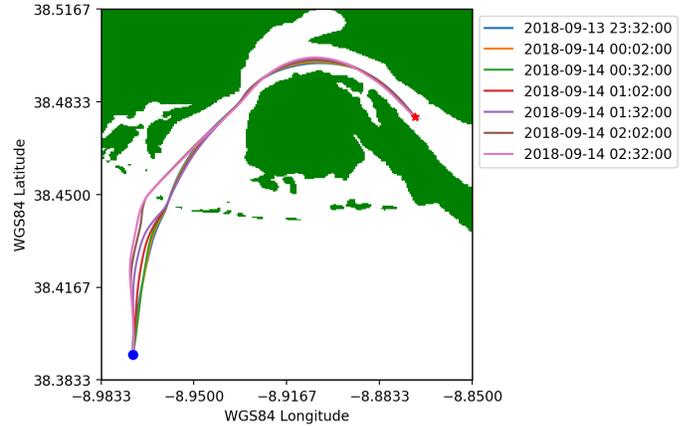


Fig. 3. Optimal trajectories for multiple deployment times - Mission 1

In Fig. 3, multiple trajectories departing from the same position at different initial times are plotted. Note that the trajectories departing at 02:02:00 and 02:32:00 are qualitatively different from the other five trajectories, as the two groups of trajectories approach the obstacle at 38.448870° N, 8.961995° W from opposite sides. This shows the influence of the ocean current on the optimal trajectories, and also suggests that the trajectories are indeed globally optimal.

Most underwater vehicles cannot track trajectories with arbitrarily low radius of curvature. Although the motion

⁴<https://www.soest.hawaii.edu/pwessel/gshhg/>

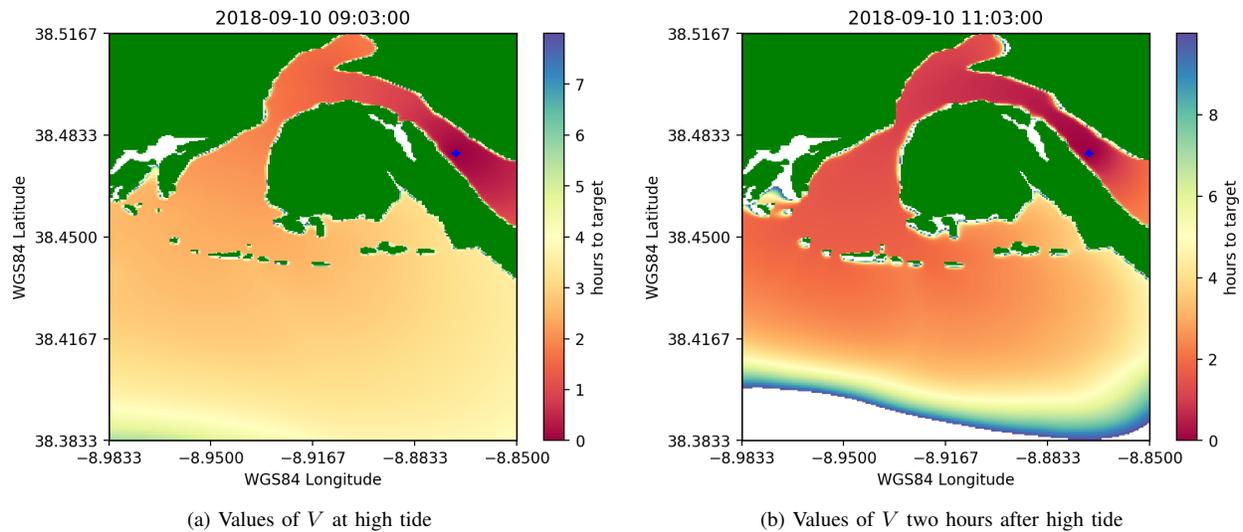


Fig. 4. Visualization of the value function - Mission 1

model (1) does not take into account any motion constraints, the generated trajectories do not have any sharp turns. For a LAUV class vehicle, the minimum turning radius is approximately 5 m at 1 m/s [6], while the minimum radius of curvature of the trajectories shown in Figures 2 and 3 is well above 100 m, which implies that the generated trajectories will be feasible for most underwater vehicles.

Besides its role in optimal trajectory generation, the value function can be useful for mission planning in multiple ways, some of which we now explore.

Fig. 4a shows the values of $V(\tau, \cdot)$, where τ corresponds to high tide. The color at each point indicates the value of the optimal cost of a trajectory which departs from that point at high tide. The target set is represented in blue. Fig. 4b is similar, but with τ equal to two hours after high tide. Note

that no value is shown for some points in the southernmost part of the operational area, which indicates that the target is not reachable from those positions when the vehicle departs two hours after high tide. We can see that the cost is not just a function of the distance to the target, as the points on the eastern side of the operational area are farther away from the target than those on the western side, but the cost values are similar.

Considering an operational scenario where an AUV is deployed from a ship or from an unmanned aerial system (UAS), this cost map allows operators to compare possible deployment positions, taking into account any constraints on the ship or the UAS's reachable positions, when the deployment time is fixed.

By minimizing $V(\cdot, \xi)$ for each ξ we can obtain a map of

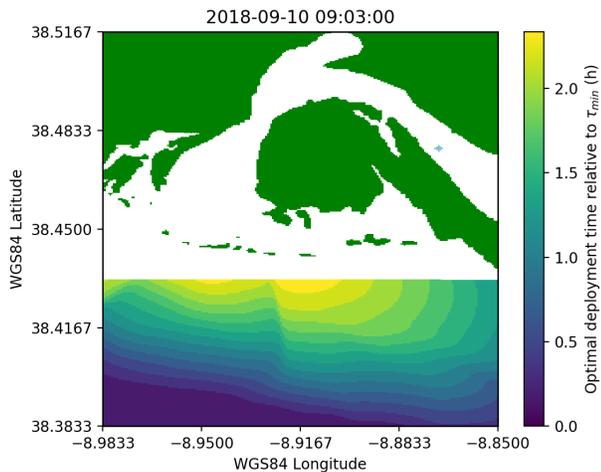


Fig. 5. Optimal deployment time as a function of the deployment position - Mission 1

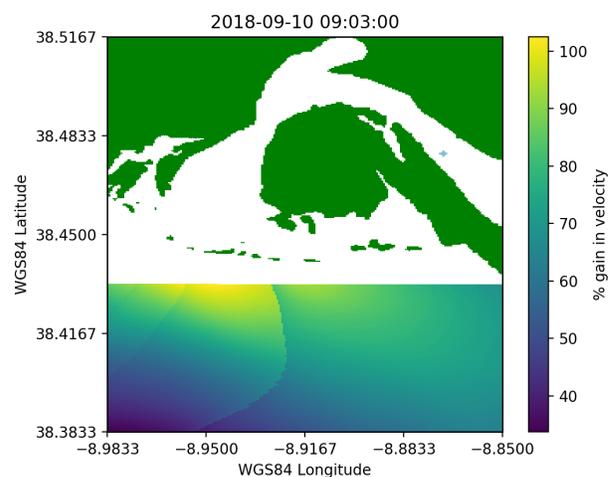


Fig. 6. Velocity gain as a function of the deployment position - Mission 1

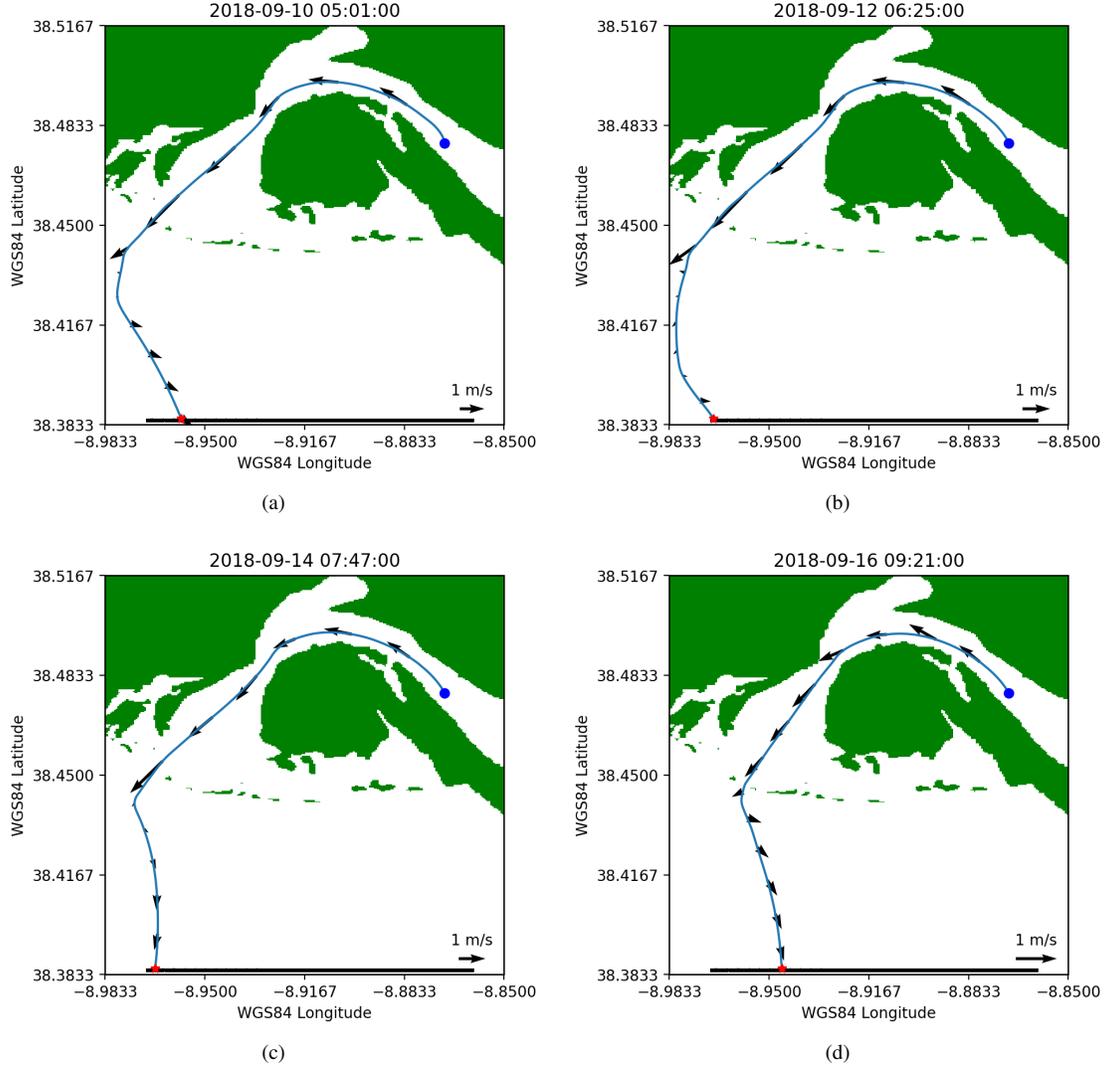


Fig. 7. Optimal trajectories with different deployment times - Mission 2

the optimal deployment time as a function of the deployment position. In Fig. 5, this map is shown for a subset of the operational area, the set of considered deployment positions. The deployment time is shown relative to the earliest possible deployment time, indicated in the figure title.

The influence of the currents on the trajectory can be verified by comparing the average velocity along the trajectory with the nominal velocity r . The gain in velocity of a trajectory can be calculated using the formula

$$g = \frac{s}{rT} - 1 \quad (4)$$

where s is the total arc length of the trajectory and T is the time taken to arrive at the target.

By calculating an optimal trajectory from each considered initial position, starting at the optimal deployment time for that position, we can get a map of the velocity gain over the operational area, as in Fig. 6. The gain exceeds 60% for most of the operational area. The discontinuity is the boundary

between groups of trajectories which avoid the same obstacle from different sides.

B. Mission 2

In this mission scenario, the vehicle starts inside the river (near the position of the target set in Mission 1) and should reach a survey area at sea. The survey area is a rectangle whose northernmost side is at latitude 38.3845° N, so we use this side of the square as the target set, assuming that the vehicle can enter the survey area at any point. If there is any preference about where the vehicle enters the survey area, q can vary along the target set to reflect that preference.

Fig. 7 shows four optimal trajectories with the same deployment position and different deployment times. The target set represented in black, the initial condition is indicated by the blue circle and the arrival position is indicated by the red star. There is a significant variation in the arrival position at the target which is due to differences in the ocean current values.

As in the first mission, the radius of curvature of the trajectories shown in Fig. 7 is adequate for tracking by most underwater vehicles. The velocity gains (as defined by (4)) for these trajectories are between 50% and 80%.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

We presented a trajectory generation method for underwater vehicles using concepts from robotics, control and trajectory generation. The method generates trajectories which are feasible for most ocean vehicles, and can take into account any known static obstacle in the operational area. The computation times achieved using the method allow us to calculate the value function in a few minutes.

After the value function has been calculated, optimal trajectories can be obtained by integrating a differential equation, a fairly simple computation which can be performed by the vehicle's onboard computer, allowing the vehicle to plan trajectories on the fly.

The numerical results show that ocean currents can have a significant influence on the optimal trajectories, and that using current forecasts for planning can bring significant benefits. We also showed how the value function, which is in some sense a byproduct of our method, can be useful for planning certain mission parameters such as the deployment time and position.

In the future we hope to test the method in the field and study its applicability to multi-stage missions and long distance, long duration missions.

ACKNOWLEDGMENTS

R.M. benefits from a Post-Doctoral grant (SFRH/BPD/115093/2016) given by the Portuguese Science Foundation (FCT).

This article was partially supported by project ENDURANCE – Sistema baseado em veículos autónomos para observação oceanográfica de longa duração, funded by Norte Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, through the European Regional Development Fund (ERDF), and by the EUMarineRobots project, funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 731103.

REFERENCES

- [1] M. Bardi and I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman Equations*. Birkhäuser Boston, 1997. DOI: 10.1007/978-0-8176-4755-1.
- [2] C. Y. Kao, S. Osher, and J. Qian, "Lax-Friedrichs sweeping scheme for static Hamilton-Jacobi equations," *Journal of Computational Physics*, vol. 196, no. 1, pp. 367–391, May 2004. DOI: 10.1016/j.jcp.2003.11.007.
- [3] T. Inanc, S. Shadden, and J. Marsden, "Optimal trajectory generation in ocean flows," in *Proceedings of the 2005, American Control Conference, 2005.*, IEEE, 2005. DOI: 10.1109/acc.2005.1470035.
- [4] D. Kruger, R. Stolkin, A. Blum, and J. Briganti, "Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, IEEE, Apr. 2007. DOI: 10.1109/robot.2007.364135.
- [5] W. Zhang, T. Inanc, S. Ober-Blobaum, and J. E. Marsden, "Optimal trajectory generation for a glider in time-varying 2d ocean flows B-spline model," in *2008 IEEE International Conference on Robotics and Automation*, IEEE, May 2008. DOI: 10.1109/robot.2008.4543348.
- [6] A. Tinka, S. Diemer, L. Madureira, E. B. Marques, J. B. de Sousa, R. Martins, J. Pinto, J. E. da Silva, A. Sousa, P. Saint-Pierre, and A. M. Bayen, "Viability-based computation of spatially constrained minimum time trajectories for an autonomous underwater vehicle: Implementation and experiments," in *2009 American Control Conference*, IEEE, 2009. DOI: 10.1109/acc.2009.5160166.
- [7] B. Rhoads, I. Mezic, and A. Poje, "Minimum time feedback control of autonomous underwater vehicles," in *49th IEEE Conference on Decision and Control (CDC)*, IEEE, Dec. 2010. DOI: 10.1109/cdc.2010.5717533.
- [8] T. Lolla, M. P. Ueckermann, K. Yigit, P. J. Haley, and P. F. J. Lermusiaux, "Path planning in time dependent flow fields using level set methods," in *2012 IEEE International Conference on Robotics and Automation*, IEEE, May 2012. DOI: 10.1109/icra.2012.6225364.
- [9] M. Detrixhe, F. Gibou, and C. Min, "A parallel fast sweeping method for the Eikonal equation," *Journal of Computational Physics*, vol. 237, pp. 46–55, Mar. 2013. DOI: 10.1016/j.jcp.2012.11.042.
- [10] A. S. Ribeiro, M. C. Sousa, J. D. L. e Silva, and J. M. Dias, "David and Goliath revisited: Joint modelling of the Tagus and Sado estuaries," *Journal of Coastal Research*, vol. 75, no. sp1, pp. 123–127, Mar. 2016. DOI: 10.2112/si75-025.1.
- [11] M. Aguiar, J. B. de Sousa, J. M. Dias, J. E. da Silva, R. Mendes, and A. S. Ribeiro, "Trajectory optimization for underwater vehicles in time-varying ocean flows," in *2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*, IEEE, Nov. 2018. DOI: 10.1109/auv.2018.8729777.