Autonomous Tracking of a Horizontal Boundary

Nuno A. Cruz and Aníbal C. Matos
INESC Porto and FEUP–DEEC
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
Email: {nacruz, anibal}@fe.up.pt

***Abstract*— The ability to employ autonomous vehicles to find and track the boundary between two different water masses can increase the efficiency in waterborne data collection, by concentrating measurements in the most relevant regions and capturing detailed spacial and temporal variations. In this paper we provide a guidance mechanism to enable an autonomous vehicle to find and track the steepest gradient of a scalar field in the horizontal plane. The main innovation in our approach is the mechanism to adapt the orientation of the crossings to the local curvature of the boundary, so that the vehicle can keep tracking the gradient regardless of its horizontal orientation. As an example, we show how the algorithms can be used to find and track the boundary of a dredged navigation channel, using only altimeter measurements.**

I. Introduction

The use of autonomous marine vehicles has been growing exponentially in the last years, yielding tremendous increase in the efficiency of data collection at sea. In most of these applications, the unmanned vessels are programmed to follow geo-referenced trajectories, while collecting relevant data from the payload sensors, with minimum intervention from an operator. However, there are many scenarios in which the main objective is to find and track a given feature, like an oceanographic front, for example, or to map other features that can be characterized by its boundary, like a pollution plume or the area affected by a harmful algae bloom episode. In these cases, a standard *lawn-mower* pattern may not be adequate, not only because of the small percentage of useful data, but mainly because the time wasted sampling the whole scalar field may prevent the vehicle to capture the spacio-temporal variations of the phenomenon. Instead of this brute force approach, it is possible to take advantage of the on-board computational power to process the data received from the payload sensors in real time, identify points in the boundary, and guide the vehicle so that it will continue to sample the region closest to the boundary. With this strategy, most of data will be collected around the boundary region, further increasing the efficiency in the observation procedure, in a process frequently known as *adaptive sampling*.

Even though this is not a new concept, it was only during the last decade that there were the first practical implementations of real-time adaptive sampling on robotic marine vehicles. The first examples include the use of segmentation algorithms on video images from ROVs to track benthic boundaries [1], and, also, the use of AUVs for searching for the sources of chemical plumes, trying to mimic the real behavior of lobsters or bacterium in odor source localization [2]–[4]. In [5], the authors used adaptive sampling primitives on MOOS [6] to control the motion of an autonomous surface vehicle to detect the horizontal thermal gradient. More recently, there has been some successful experiments on autonomous vertical controllers for thermocline tracking, carried out on gliders [7] and other AUVs [8]–[12]. In [13], the authors used the Tethys AUV to detect a coastal upwelling front off the coast of California. Later, in [12] the authors adapted the algorithms to effectively track the front by crossing it at fixed angles, taking advantage of some *a priori* knowledge of the typical orientation.

There have also been some proposals for the use of cooperating platforms to collectively sample the environment [14], but with few implementations in the field, probably due to practical difficulties. Recently, the possibility of using adaptive sampling with multiple AUVs has regained attention, as a means to improve even further the efficiency in the sampling process, such as the algorithms proposed in [15] and [16] to track plumes and other dynamic ocean features.

II. Tracking a Horizontal Boundary

The work presented in this paper is an evolution of the work described in [8] and [9], where the MARES AUV[1] has been used to identify and track thermoclines in real time. The thermocline is a vertical transition layer in the water column, separating the warm surface layer, or *mixed layer*, from the cold deep water bellow it. It is typically a very thin layer, where the water temperature drops nearly linearly as the depth increases, with a significant gradient as compared to the rest of the profile. The MARES AUV has independent horizontal and vertical controllers, and in order to track the thermocline, only the vertical controller was affected. The navigation system had a standard reference in the horizontal plane, while the vehicle was continuously performing vertical *yo-yo*'s, for which the limits were set according to the temperature profiles being captured in real time.

In this paper, we expand this approach to two dimensions and show how it can be applied to the efficient characterization and tracking of a horizontal boundary for a specific scalar field $\Phi(x,t)$, or Φ for simplicity. We assume that the scalar field Φ has a transition *zone* with a higher gradient than in the rest of the area, which is a typical characteristic of many oceanographic phenomena, like fronts or plume boundaries. We further assume that the vehicle can take only local

[1]MARES – *Modular Autonomous Robot for Environment Sampling [17]*

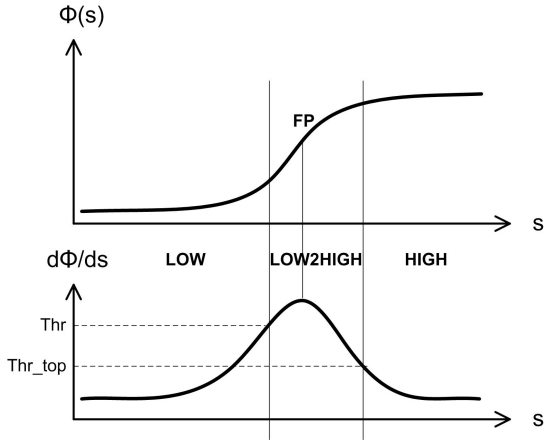

Fig. 1. A simplified view of a transect along a scalar field, passing from the region with *low* values to the region with *high* values. Below, the derivative along the same trajectory, with the frontal point located at the maximum derivative.

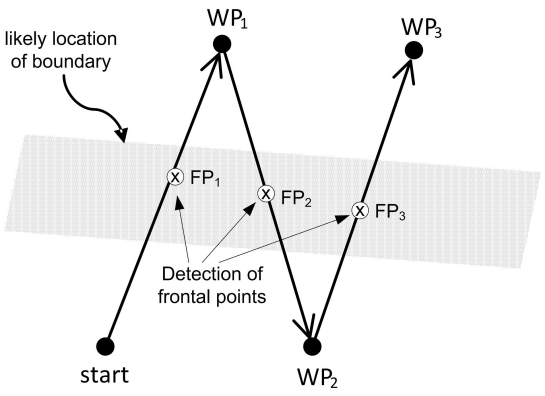

Fig. 2. Initial phase of the mission, finding the boundary. WP_1–WP_3 are programmed so that the likely location of the boundary is crossed three times. For each crossing, a frontal point, FP, should be detected.

measurements of Φ, therefore in order to find the maximum horizontal gradient, the vehicle needs to cross the transition zone, moving from the region where the scalar field has *high* values (the HIGH region, to be defined later) to the region with *low* values (the LOW region), and vice-versa. A simplified view of an expected transect along this scalar field can be seen in figure 1.

In our approach, the successful tracking of a horizontal gradient results in a combination of a main direction of travel (aligned with the horizontal location of the boundary), together with a *zig-zag* around that direction. Note that neither the evolution of the boundary line, nor the amplitude of the *zig-zag* are known at the beginning of the mission.

Our strategy for implementing this tracking in the horizontal plane develops in two phases, as follows.

A. Phase I – Finding the Boundary

Even though it may be preferable to have a full autonomous system, in most of the missions there is some *a priori* information about the phenomena being studied, at least locally. In our case, we assume that there is an area where it is likely that the boundary is located and we start the mission close to that area. For simplicity, let's assume that we start in the LOW part of the scalar field. We define three initial waypoints WP_1–WP_3, so that the first three trajectories cross the likely location of the boundary, as represented in figure 2.

For each of these trajectories we need to acquire a full profile with a shape similar to figure 1, and find the location of each frontal point. At the end of this first stage, there should be the first three points of maximum gradient (or frontal points FP_1–FP_3), so that the strategy can proceed to phase II. Depending on the application scenario, this first processing can be made online, taking advantage of the availability of the full profile, or it can be done offline by an operator, particularly if the temporal variations of the scalar field are slow. Note also that the trajectory represented in figure 2 can be replaced by a different sampling pattern, just to ensure that we find the first three frontal points FP_1–FP_3.

B. Phase II – Tracking the Boundary

As soon as the vehicle gets to WP_3, it enters a fully autonomous mode, in which all processing is done online to keep tracking of the boundary. This requires two algorithms, to be detailed in section III:

1) Automatic generation of new waypoints;
2) Real time, online detection of the maximum gradient along a trajectory.

In order to track the boundary, the vehicle uses algorithm #1 to generate WP_4, *i.e* it projects a new heading (extending to waypoint WP_4) that will ensure an additional crossing of the boundary and, at the same time, some progress along the same boundary, as shown in a simplified version in figure 3. As soon as the vehicle starts this new transect, it also starts algorithm #2, processing the samples of the scalar field as soon as they are available and trying to detect the new frontal point FP_4. If algorithm #2 is successful, FP_4 is detected before the vehicle reaches WP_4, so that WP_5 can be generated and the algorithms continue iteratively. WP_4 can then be seen as the maximum excursion that the vehicle will travel if algorithm #2 fails.

III. Algorithms for Boundary Tracking

A. Automatic Generation of New Waypoints

The simplified mechanism shown in figure 3 only works well for the generation of waypoints when the boundary follows a line. In fact, if the boundary follows a relatively straight line, there is no need for three frontal points, as only two would be enough to define such line. However, if there is some curvature in the boundary, we need to know (at least) the last three crossings in order to estimate the local curvature and generate the following waypoint.

The proposed algorithm to adapt the crossings with the local curvature is illustrated in figure 4. In this figure, WP_i represent

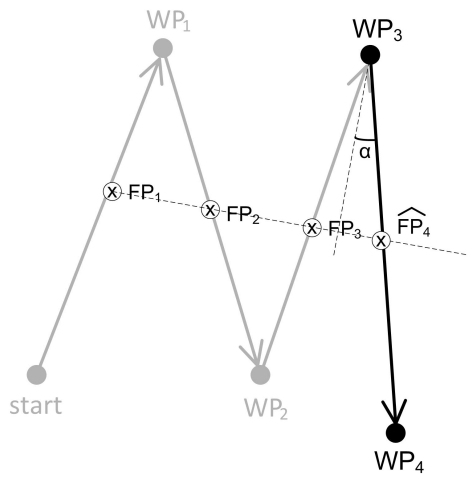

Fig. 3. Beginning of fully autonomous mode tracking the boundary. WP_4 is calculated when the vehicle is at WP_3, in such a way as to ensure some progress along the boundary, given by parameter α, so that the boundary is expected to be crossed close to $\widehat{FP_4}$.

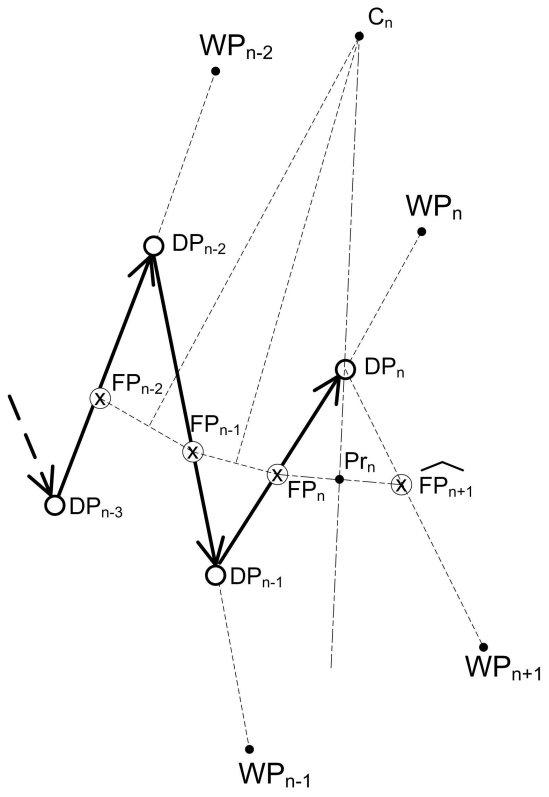

Fig. 4. Generation of a new waypoint WP_{n+1}, based on detections of frontal points $\{FP_{n-2}, FP_{n-1}, FP_n\}$ and the curvature they define. The location of the new frontal point is expected at $\widehat{FP_{n+1}}$.

the waypoints and FP_i represent the points of maximum gradient, or frontal points. Note that a certain FP_i is detected when the vehicle is moving towards WP_i, but before actually getting there. We identify the points DP_i as the places where that detection is made.

Consider that the vehicle has crossed the gradient and has just found front point FP_n (this happens when the vehicle is located at DP_n). The algorithm to generate the next waypoint, WP_{n+1}, is:

1) Compute C_n, the center of the circumference defined by the last three frontal points, $\{FP_{n-2}, FP_{n-1}, FP_n\}$, by intersecting the bisector of $\overline{FP_{n-2}FP_{n-1}}$ with the bisector of $\overline{FP_{n-1}FP_n}$. This circumference represents the local curvature of the boundary.
2) Connect C_n with DP_n to define an axis of symmetry.
3) Project the last frontal point, FP_n to the other side of this axis to get the next estimated frontal point, $\widehat{FP_{n+1}}$.
4) Determine the new waypoint WP_{n+1}, extending the line connecting DP_n with $\widehat{FP_{n+1}}$, up to a *reasonable* distance.

This algorithm ensures that each "V" shape of a *zig-zag* is oriented towards the center of the local curvature, therefore adapting the crossing angles to ensure future detections of FP's. In terms of practical implementation, however, the uncertainty in the determination of FP_n may result in an incorrect estimate of C_n. This, in turn, can yield a projection $\widehat{FP_{n+1}}$ either *backwards* or too far from FP_n. For this reason, we limit the rate of advance along the boundary within a preset interval, and we assess the direction of motion, verifying that

$$\overrightarrow{FP_{n-1}FP_n} \cdot \overrightarrow{FP_n\widehat{FP_{n+1}}} > 0$$

As an illustrative example, we've synthesized a scalar field representing a contour (for example, a plume), and used this algorithm to produce new waypoints. Figure 5 demonstrates the ability to track the curvature of the boundary. Figure 6 details some of the simulated trajectories, where the adaptation of the crossing angles is clear.

B. Online Detection of the Maximum Gradient

When looking at a full profile similar to the shape in figure 1, there are several options to find the maximum gradient and extracting the diferent regions of the data. However, in a practical implementation, the main challenge is to maintain the vehicle as close as possible to the region of the maximum gradient. This means that the vehicle has to detect as early as possible a significant decrease in the gradient and avoiding as much as possible to navigate within those *flatter* regions. A difficult problem in detecting the gradient is then to estimate de derivatives of the scalar field along the track, $\frac{d\Phi}{ds}$ from a limited number of previous data points and decide if those derivatives are sufficient to conclude that the maximum gradient has already been passed and there is no need to proceed further. More, it is quite likely that these data will be updated several times per second, possibly showing small scale variations, it may not be uniformly distributed and, surely, may have errors.

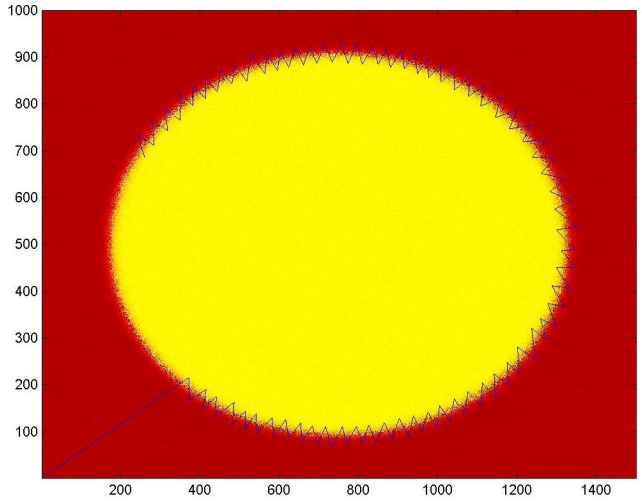

Fig. 5. Demonstration of the waypoint generation mechanism being used to track a boundary along a closed contour. Note the *zig-zag*'s oriented towards the center of the ellipse.

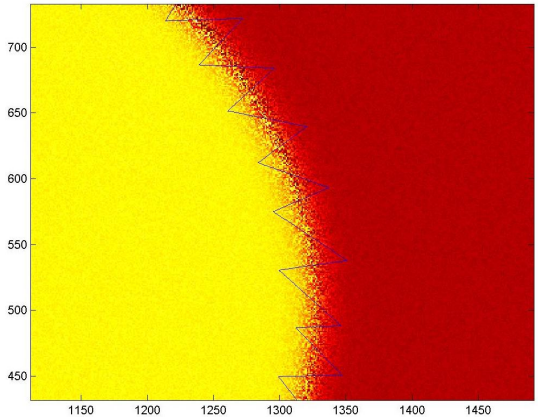

Fig. 6. Detail of the vehicle trajectory while tracking a horizontal boundary with a sharp gradient.

In order to solve this problem and estimate the gradient of the scalar field, we follow the approach described in [8] and [9], which has produced very good results in tracking the temperature gradient associated with the thermocline. We cluster data points into bins of distance traveled and take the differences of the averaged values. The size of the bins depends on the estimated slope that we intend to track and the amount of data available (which is a function of the sampling rate of the sensor and vehicle velocity). This size acts as a low pass filter which may affect the ability to detect gradients. Smaller bins result in large variations in gradient estimation, while larger bins smooth the variations but hinder the separation of gradients.

As an example, consider the bathymetry map represented in figure 7, and suppose we intend to follow the line of maximum gradient. Note that in this case, the channel wall drops from roughly 3.5m to roughly 5.5m in less than 5m.

Although the map seems to be relatively smooth, if we analyze the raw altimeter measurements taken during the

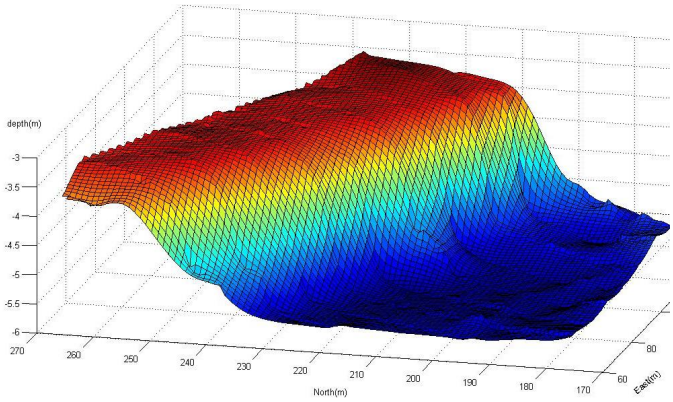

Fig. 7. Example of a bathymetry map of the slope of a navigation channel, acquired by the Zarco ASV [18] in July 2014, at Base Naval do Alfeite, Portugal.

bathymetry mission, we can verify that they are relatively noisy, as can be seen in the example of a profile shown in figure 8 (the topmost left plot). It is also quite difficult to quantify the variations of the gradient along track (in the topmost right plot). However, in the other plots of figure 8 we show the result of aggregating the measurements into bins of increasing size. The smoothness of the data points is obvious, as well as the increasing evidence of the maximum gradient (around 0.5 m/m).

The full algorithm developed for online gradient tracking can be described by the state machine represented in fig. 9, where the dark arrows represent the transitions that are expected during a successful tracking. These transitions will cycle the state machine through the most relevant states:

- HIGH - The vehicle is located at the higher part of the scalar field.
- HIGH2LOW - The vehicle is descending the gradient, towards the lower part of the scalar field.
- LOW - The vehicle is located at the lower part of the scalar field.
- LOW2HIGH - The vehicle is climbing the gradient, towards the higher part of the scalar field.

Suppose for simplicity that the vehicle is in the HIGHER part of the scalar field, moving towards a given waypoint. During the trajectory, the vehicle will evaluate the gradient of the scalar field and compare it with a given threshold, **Thr**. When this threshold is exceeded, it will assume it has started descending the gradient towards the lower part of the scalar field, entering the HIGH2LOW state. The vehicle will proceed towards the same waypoint until it detects a reduction in the gradient to a level below **Thr_bot**. In order to confirm the *lower limit* of the gradient and avoid (early) false detections, an additional test is performed, verifying that the vehicle has moved a minimum distance away from the frontal point, FP, *i.e.* if $||\mathrm{x} - \mathrm{FP}|| > \mathrm{s}_{\mathrm{span.low}}$, where x is the current position, FP is the position of the frontal point detected, and $\mathrm{s}_{\mathrm{span.low}}$ is a parameter set by the user to allow further excursion into

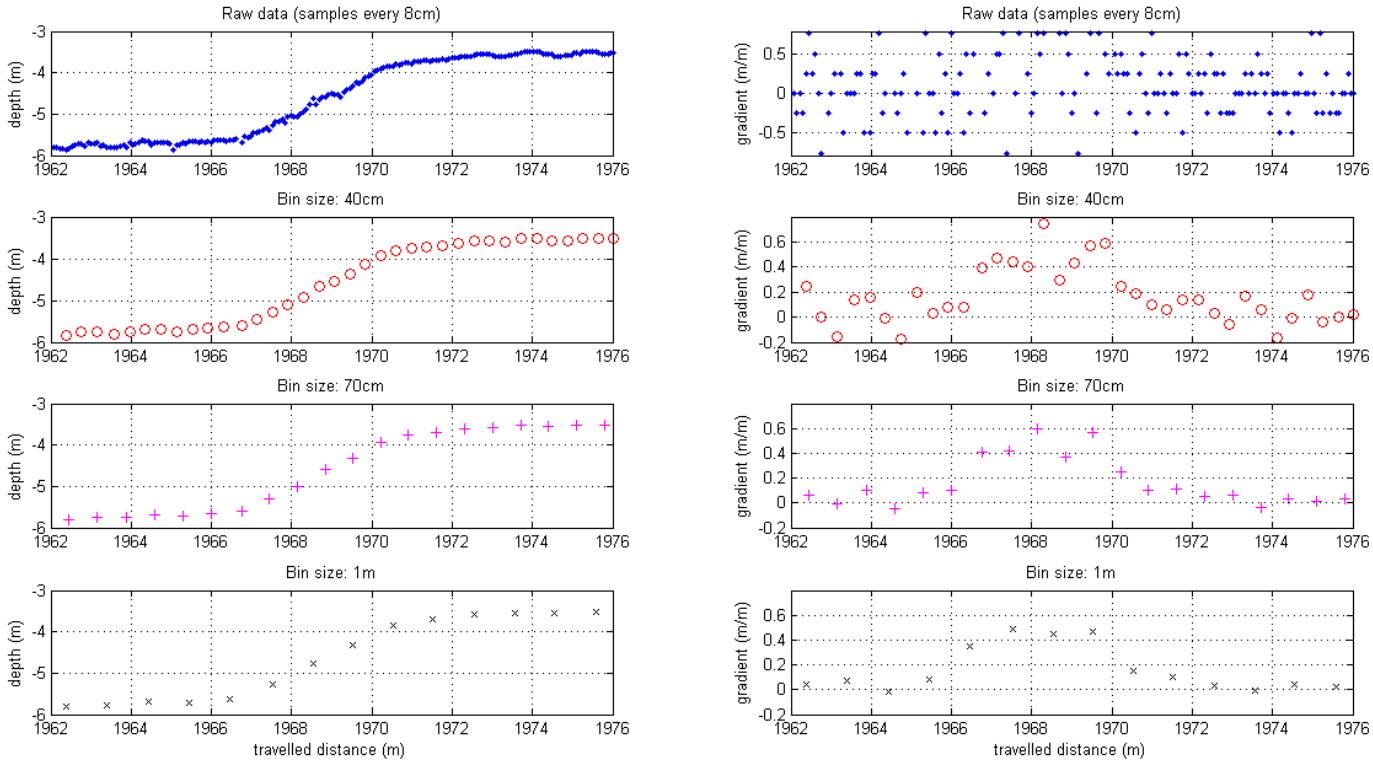

Fig. 8. Example of altimeter samples collected across the slope of the map shown in figure 7 (left-side plots) and gradient estimated from differences (right-side plots). As the samples are aggregated into larger bins (top to bottom), both the profile and the gradient gets smoother.

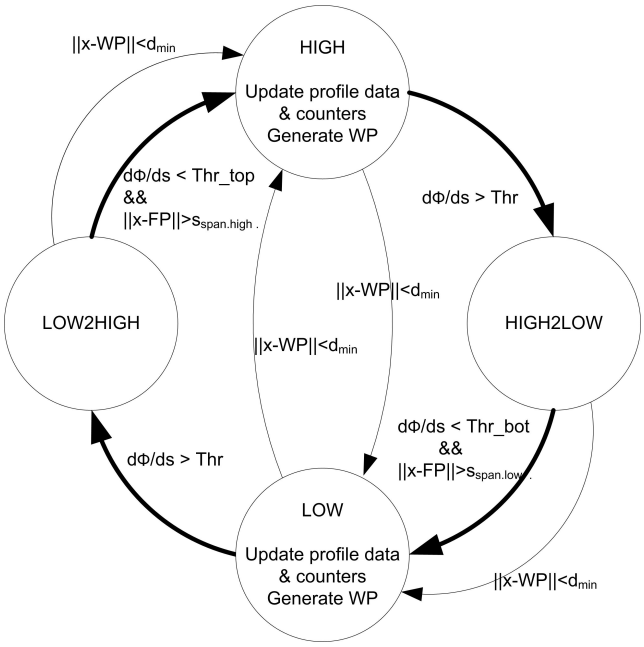

Fig. 9. State machine and transitions representing the gradient tracking maneuver. The bold arrows represent the expected cyclic transitions during normal tracking.

the LOW part of the scalar field (for example to capture complementary data). When both these conditions are met, the vehicle enters the BOTTOM state.

Note from the state machine of fig. 9 that the LOW state is also reached if $||x - WP|| < d_{min}$, which is a safety mechanism to ensure that the trajectory is limited to reaching the waypoint, even if the algorithm is not able to positively detect the *upper limit* or the *lower limit* of the gradient. When the vehicle enters the LOW state, the characteristics of the gradient are extracted from the previous profile (in particular, the maximum gradient and the limits of the regions) and this information is used to adapt the thresholds for the gradient detection during the next trajectory. The new waypoint is then generated with the algorithm described before, and passed on to the onboard navigation system. As long as this process is active, the above cycle will be maintained, resulting in a *zig-zag* pattern around the boundary.

In order to evaluate the performance of the algorithm, a simple mechanism is to maintain 3 different counters that are incremented depending on the state transitions that lead to the reversal of the profiles (HIGH and LOW states). The first counter, **full_track**, is increased when the vehicle detects both the beginning and the end of the gradient, *i.e.* reaches the HIGH or LOW states through a dark arrow; the second, **begin_only**, is increased when the vehicle enters a transition state (HIGH2LOW or LOW2HIGH) but does not detect the end of the gradient, *i.e.* it changes trajectory because it reaches

the waypoint. Finally, a third counter, **fail**, is increased if there is a direct transition from the HIGH state to LOW, or vice-versa. Naturally, the information provided by these counters can be used individually or in combination (for example, abort the mission if **fail** > 3 or if **fail/full_track** > 0.1).

IV. Conclusions and Future Work

This paper describes the algorithms required to find and track a horizontal boundary with an autonomous marine vehicle and provide evidence of their efficacy. These algorithms interpret the samples from the scalar field and produce geographic waypoints or directions that the vehicle has to follow, therefore they can can be implemented on virtually any ASV or AUV that is able to move at a constant depth. The vehicle is commanded to cross the transition zone in constant *zig-zag*'s, while advancing along the boundary. On each crossing, the maximum gradient is detected and its position is used to generate a new waypoint, taking into account the local curvature of the boundary.

During the tracking phase, a set of counters is maintained to assess the performance of the algorithms and make appropriate decisions in case of poor performance. The parameters of the transition zone (location and gradient) can be passed onto other processes, for example to switch any special sensor, to guide another vehicle with complementary sensors, or to trigger an underwater sampler to capture a relevant sample of water for lab analysis, like the work described in [19].

Even though we have tested the algorithms thoroughly with data taken from field samples, the next stage of development is to validate the integrated system in the field, tracking a challenging boundary both with an ASV and also with an AUV. At a later stage, we intend to combine this horizontal tracking with the vertical tracking demonstrated for thermocline sampling, resulting in a 3D adaptive sampling mechanism. Finally, we will also provide metrics for the evaluation of the sampling efficiency, for example comparing the total track length covered by the vehicle with the length of the projection on the boundary.

Acknowledgement

This work is financed by the ERDF European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project FCOMP-01-0124-FEDER-037281.

References

[1] C. Barat and M. J. Rendas, "Benthic boundary tracking using a profiler sonar: A mixture model approach," in *Proc. MTS/IEEE Int. Conf. Oceans'03*, San Diego, CA, USA, Sep. 2003, pp. 1409–1416.

[2] J. A. Farrell, W. Li, S. Pang, and R. Arrieta, "Chemical plume tracing experimental results with a REMUS AUV," in *Proc. MTS/IEEE Int. Conf. Oceans'03*, San Diego, CA, USA, Sep. 2003, pp. 962–968.

[3] S. Pang and J. A. Farrell, "Chemical plume source localization," *IEEE Trans. Syst., Man, Cybern. B*, vol. 36, no. 5, pp. 1068–1080, Oct. 2006.

[4] W. Naeem, R. Sutton, and J. Chudley, "Chemical plume tracing and odour source localization by autonomous vehicles," *The Journal of Navigation*, vol. 60, no. 2, pp. 173–190, May 2007.

[5] D. P. Eickstedt, M. R. Benjamin, D. Wang, J. Curcio, and H. Schmidt, "Behavior based adaptive control for autonomous oceanographic sampling," in *Proc. Int. Conf. Robot. Autom*, Rome, Italy, Apr. 2007.

[6] P. M. Newman, "MOOS – mission orientated operating suite," Massachusetts Institute of Technology, USA, Tech. Rep. 08, 2008.

[7] H. C. Woithe and U. Kremer, "A programming architecture for smart autonomous underwater vehicles," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems IROS'09*, St. Louis, MO, USA, Oct. 2009.

[8] N. Cruz and A. Matos, "Reactive AUV motion for thermocline tracking," in *Proc. IEEE Int. Conf. Oceans'10*, Sydney, Australia, May 2010.

[9] ——, "Adaptive sampling of thermoclines with autonomous underwater vehicles," in *Proc. MTS/IEEE Int. Conf. Oceans'10*, Seattle, WA, USA, Sept. 2010.

[10] Y. Zhang, J. G. Bellingham, M. Godin, J. P. Ryan, R. S. McEwen, B. Kieft, B. Hobson, and T. Hoover, "Thermocline tracking based on peak-gradient detection by an autonomous underwater vehicle," in *Proc. MTS/IEEE Int. Conf. Oceans'10*, Seattle, WA, USA, Sept. 2010.

[11] S. Petillo, A. Balasuriya, and H. Schmidt, "Autonomous adaptive environmental assessment and feature tracking via autonomous underwater vehicles," in *Proc. IEEE Int. Conf. Oceans'10*, Sydney, Australia, May 2010.

[12] Y. Zhang, J. G. Bellingham, J. P. Ryan, B. Kieft, and M. J. Stanway, "Two-dimensional mapping and tracking of a coastal upwelling front by an autonomous underwater vehicle," in *Proc. MTS/IEEE Int. Conf. Oceans'13*, San Diego, CA, USA, Sept. 2013.

[13] Y. Zhang, M. A. Godin, J. G. Bellingham, and J. P. Ryan, "Using an autonomous underwater vehicle to track a coastal upwelling front," *IEEE J. Oceanic. Eng.*, vol. 37, no. 3, pp. 338–347, July 2012.

[14] E. Fiorelli, P. Bhatta, N. E. Leonard, and I. Shulman, "Adaptive sampling using feedback control of an autonomous underwater glider fleet," in *Proc. Int. Symp. Unmanned Untethered Submersible Tech. UUST'03*, Durham, NH, USA, Aug. 2003.

[15] S. Petillo, H. Schmidt, and A. Balasuriya, "Constructing a distributed AUV network for underwater plume-tracking operations," *Int. J. Distr. Sensor Networks*, vol. 2012, 2012.

[16] B. Reed and F. Hover, "Oceanographic pursuit: Networked control of multiple vehicles tracking dynamic ocean features," *Methods in Oceanography*, 2014.

[17] N. Cruz and A. Matos, "The MARES AUV, a modular autonomous robot for environment sampling," in *Proc. MTS/IEEE Int. Conf. Oceans'08*, Quebec, Canada, Sept. 2008.

[18] N. Cruz, A. Matos, S. Cunha, and S. Silva, "Zarco – an autonomous craft for underwater surveys," in *Proc. 7th Geomatic Week*, Barcelona, Spain, Feb. 2007.

[19] Y. Zhang, R. S. McEwen, J. P. Ryan, and J. G. Bellingham, "An adaptive triggering method for capturing peak samples in a thin phytoplankton layer by an autonomous underwater vehicle," in *Proc. MTS/IEEE Int. Conf. Oceans'09*, Biloxi, MI, USA, Oct. 2009.