2011 50th IEEE Conference on Decision and Control and
European Control Conference (CDC-ECC)
Orlando, FL, USA, December 12-15, 2011

Freshwater-Saltwater Boundary Detection Using Mobile Sensors Part I: Drifter Deployment

Yu Ru and Sonia Martinez

Abstract— **Due to a reduction in the supply of freshwater in estuaries, saltwater can intrude deeply into river channels. We focus on the freshwater-saltwater boundary detection problem, and identify a point on the boundary that reflects the degree of salinity intrusion via deploying mobile drifters with powered propellers. Our approach consists of deploying two drifters to explore the boundary in different directions and obtaining the estimate of the point based on the estimates from these two drifters. We show that the proposed algorithms can achieve an arbitrarily accurate estimate under certain assumptions on the salinity field.**

I. Introduction

Due to a reduction in the supply of freshwater in estuaries, saltwater can intrude deeply into river channels, which adversely impacts inland biota. The degree of salinity intrusion is usually identified by the relative location of the freshwater-saltwater boundary with respect to the water mouth. More precisely, a salinity threshold can be used to define the freshwater-saltwater boundary. For example, the threshold could be 2ppt (parts per thousand) since the 2ppt bottom salinity position could be used as a habitat indicator for estuarine populations as studied in [1]. Detecting the boundary is a challenging task because there are also other factors that could affect the boundary, such as tidal forcing, wind mixing, and gravitational circulation [2]. Previously the freshwater-saltwater boundary has been studied based on simplified 1-D theoretical analysis [2], numerical analysis [3], or static measurements [4]. However, none of these approaches is capable of obtaining a relatively accurate boundary (that might change slowly with time).

In this paper, we study tracking the freshwater-saltwater boundary via mobile drifters with powered propellers that can run against river flows. Instead of recovering the exact boundary, the objective is to deploy the drifters automatically to determine a point (on the boundary) which goes furthest inland. This point can reflect the degree of salinity intrusion into a river channel and can be used for the operation of control structures (e.g., the Hiram M. Chittenden Locks in Salmon Bay, Washington) to prevent salinity intrusion. We propose boundary point detection, boundary exploration, and boundary exploitation algorithms which adaptively explore the boundary, and show that the proposed algorithms can achieve an arbitrarily accurate estimate under certain assumptions on the salinity field. Simulation results are consistent with the theoretical analysis. Note that in this paper we focus on the way-point based high level motion planning of drifters, and adaptively sample the river environment based on drifters' locations to save energy consumption. In the companion paper [5], we study the low level minimum energy control to implement the way-point based navigation.

Boundary tracking/estimation problems have been studied extensively, e.g., a survey [6] and some recent work [7]–[12]. In [7], the authors proposed algorithms that allow a mobile sensor network to track and distribute along a dynamic boundary. In [8], a high-level adaptive control for rapid imaging of samples in atomic force microscopy is presented, which drastically reduces the area to be imaged. In [9], the authors developed a framework for environmental boundary tracking and estimation via elliptic approximations, and reported the implementation in [12]. In [10], the authors utilized a formation of four moving sensor platforms to explore a noisy scalar field. In [11], the authors proposed an algorithm to optimally approximate an environmental boundary with a polygon. We depart from previous work by focusing on the search of specific points on a given salinity level set, which is sufficient for determining salinity intrusion, and leads to more robust solutions with respect to variations in the boundary shape. The proposed novel adaptive sampling strategy does not require drifters to have an initial estimate of the (shape of the) boundary (as required in [7], [9], [11]) or stay on the boundary (as assumed in [7], [11]), and can be used for sequential drifter deployment.

II. Problem Formulation

For simplicity, we assume that the river banks are parallel and the deployment base is on the freshwater side. Using the Cartesian coordinate system, we define the centerline of the river as the x-axis, and choose the y-axis to be perpendicular to the x-axis and pass through the deployment base. Suppose that the distance from the deployment base to the x-axis is $\frac{L}{2}$ (note that $L > 0$ is chosen to be less than the width of the river so that drifters can be operated safely). Then the studied river environment can be described by the two dimensional region $D := \{(x\ y)^T \in \mathbb{R}^2 \mid -\frac{L}{2} \le y \le \frac{L}{2}\}$. Since we focus on the high level planning of mobile drifters, we omit the descriptions of velocity field and drifter dynamics (for details, refer to [5]), and assume that drifters can run against the river flow.

This work was supported in part by NSF Award CNS-0930946. The authors are with Department of Mechanical and Aerospace Engineering, University of California, San Diego, 9500 Gilman Dr, La Jolla CA, 92093, {yuru2,soniamd}@ucsd.edu

The salinity field is a mapping[1] $S : D \mapsto \mathbb{R}_0^+$, assigning $S(x,y)$ to $(x\ y)^T$, where $\mathbb{R}_0^+$ is the set of nonnegative real numbers and $S(x,y)$ is the salinity at location $(x\ y)^T$. Given the salinity threshold B_{th}, the freshwater-saltwater boundary is defined as a level set $\text{Boundary} = \{(x\ y)^T \in D \mid S(x,y) = B_{\text{th}}\}$. Refer to [3] for illustrations of typical boundaries.

Each drifter is equipped with an accurate position sensor and an accurate salinity sensor, and is able to communicate with the deployment base directly. The objective is to deploy a set of drifters from the base and find a point p_{min} on the boundary with the smallest x coordinate, i.e.,

$$p_{\text{min}} = \underset{(x\ y)^T \in \text{Boundary}}{\arg\min}\ x\ .$$

If there are multiple points on the boundary that minimize x, we only need to find one because we are interested in $x_{p_{\text{min}}}$ instead of p_{min}.

III. Drifter Deployment

In this section, we study how to deploy two drifters to find the point p_{min}. There are 3 stages in the drifter deployment:

I Drifter A is first deployed to explore one half of the boundary and obtains its estimate;

II Drifter B is then deployed to explore the other half of the boundary and obtains its estimate;

III The deployment base determines an estimate of p_{min} based on the estimates from drifters A and B.

Salinity sensors are actively managed to take measurements only at certain time and/or locations to save battery power.

A. Stage I

At Stage I, drifter A leaves the base and first reaches the origin p_A^{01} (the low level motion control is studied in [5]). Then it follows the flow along the centerline and samples the river with a fixed sampling interval $\Delta T > 0$. The fixed interval sampling ends if there are two consecutive salinity measurements at locations p_A^1 and p_A^2 such that $(S(p_A^1) - B_{\text{th}}) \times (S(p_A^2) - B_{\text{th}}) \le 0$. In the fixed interval sampling period, a series of locations p_A^{01}, p_A^{02}, …, p_A^{0n} and the corresponding salinity measurements $S(p_A^{01})$, $S(p_A^{02})$, …, $S(p_A^{0n})$ are obtained. We construct a new series using $k_A^{0i} = \frac{S(p_A^{0i+1}) - S(p_A^{0i})}{\|p_A^{0i+1} - p_A^{0i}\|}$ for $i = 1, 2, \ldots, n-1$, where $\|\cdot\|$ is the 2-norm. Essentially k_A^{0i} reflects how salinity changes with respect to distance. For simplicity, we suppose salinity changes either linearly or exponentially[2] with respect to distance. The salinity change trend can be determined based on the standard deviation of the above series, and will be used when drifter A adaptively determines a boundary point.

Now drifter A explores the line segment $p_A^1 p_A^2$ to determine a boundary point b_A^1 using Algorithm 1. The prescribed input variable ε is used to bound the estimation error. Steps 1-3 check if p^1 or p^2 satisfies $|S(p) - B_{\text{th}}| \le \varepsilon$; if true, the algorithm terminates. Step 4 generates a point p^3 that lies on the line segment p^1p^2 using a function $f(p^1, p^2)$ (see Eqs. (1) and (2) below). Based on the salinity measurement at p^3, we update either p^1 or p^2, and then repeat the procedure starting from Step 1. The function $f(p^1, p^2)$ could be as simple as $\frac{p^1+p^2}{2}$. To achieve faster convergence, we could choose $f(p^1, p^2)$ based on the salinity change trend. More specifically, if salinity changes linearly (or exponentially), then use Eq. (1) (or Eq. (2)) as below

$$f(p^1, p^2) = p^1 + (p^2 - p^1)\frac{B_{\text{th}} - S(p^1)}{S(p^2) - S(p^1)}\ , \tag{1}$$

$$f(p^1, p^2) = p^1 + (p^2 - p^1)\frac{\ln B_{\text{th}} - \ln S(p^1)}{\ln S(p^2) - \ln S(p^1)}\ . \tag{2}$$

Algorithm 1 Boundary Point Detection

Input: Salinity measurements at p^1, p^2 satisfying $(S(p^1) - B_{\text{th}}) \times (S(p^2) - B_{\text{th}}) \le 0$, and a positive threshold ε

Output: A boundary point p satisfying $|S(p) - B_{\text{th}}| \le \varepsilon$

```
1:  if $|S(p^1) - B_{\text{th}}| \le \varepsilon$ (or $|S(p^2) - B_{\text{th}}| \le \varepsilon$) then
2:      Output $p^1$ (or $p^2$) and exit;
3:  end if
4:  Move to $p^3 = f(p^1, p^2)$, and take measurement $S(p^3)$;
5:  if $(S(p^3) - B_{\text{th}}) \times (S(p^1) - B_{\text{th}}) > 0$ then
6:      Set $p^1 = p^3$;
7:  else
8:      Set $p^2 = p^3$;
9:  end if
10: Go to Step 1.
```

After obtaining b_A^1, drifter A initializes p_{min}^A (namely, its estimate of p_{min}) using b_A^1. Then it moves to point $p_A^3 = (x_{b_A^1}\ y_{b_A^1} + \Delta y_1)^T$, and takes the salinity measurement at p_A^3, where Δy_1 is a positive constant. Depending on the salinity measurement at p_A^3, there are two possibilities:

- $S(p_A^3) \ne B_{\text{th}}$: if $S(p_A^3) < B_{\text{th}}$ (or $S(p_A^3) > B_{\text{th}}$), drifter A moves downstream (or upstream), samples the river with fixed sampling interval ΔT, and stops if there are two consecutive salinity measurements at locations p_A and p_A^4 such that $(S(p_A) - B_{\text{th}}) \times (S(p_A^4) - B_{\text{th}}) \le 0$. Now drifter A explores $p_A p_A^4$ using Algorithm 1 to obtain the boundary point b_A^2, as shown in Fig. 1;
- $S(p_A^3) = B_{\text{th}}$: then set b_A^2 to be p_A^3.

Once the second boundary point b_A^2 is obtained, drifter A compares $x_{b_A^2}$ with $x_{p_{\text{min}}^A}$: if $x_{b_A^2} < x_{p_{\text{min}}^A}$, it updates p_{min}^A with b_A^2; otherwise, no update is necessary. Now drifter A can repeat the procedure starting from the point b_A^1 by first moving to $p_A^5 = (x_{b_A^2}\ y_{b_A^2} + \Delta y_2)^T$ where Δy_2 is a positive constant, taking the salinity measurement at p_A^5, and then taking further actions depending on $S(p_A^5)$. Δy_2 can be chosen adaptively:

- If $x_{b_A^2} > x_{b_A^1}$, drifter A is exploring a region of a local maximum[3], and Δy_2 could be larger than Δy_1;

[1]Note that we assume the salinity field is static. However, our proposed algorithms also work if the evolution of the salinity field is slow with respect to the timescale of the boundary explorations.

[2]Salinity could change exponentially as studied in [2].

[3]A point p on the freshwater-saltwater boundary is called a local minimum (or maximum) if for any other point p' on the boundary that is sufficiently close to p, $x_p < x_{p'}$ (or $x_p > x_{p'}$) holds.

- If $x_{b_A^2} = x_{b_A^1}$, Δy_2 could be the same as Δy_1;
- If $x_{b_A^2} < x_{b_A^1}$, drifter A is exploring a region of a local minimum, and Δy_2 could be smaller than Δy_1 and depends on the ratio $k_2 = \frac{y_{b_A^2} - y_{b_A^1}}{x_{b_A^1} - x_{b_A^2}}$.

Essentially, Δy_2 can be chosen to be $g(b_A^1, b_A^2)$, a function of b_A^1 and b_A^2. In general, for $i \geq 2$, Δy_i can be chosen to be $g(b_A^{i-1}, b_A^i)$. For easier implementation, we use

$$g(b_A^{i-1}, b_A^i) = \begin{cases} \Delta y_{\text{max}} & \text{if } x_{b_A^i} \geq x_{b_A^{i-1}} \ , \\ \frac{\Delta y_{\text{min}} + \Delta y_{\text{max}}}{2} & \text{if } x_{b_A^i} < x_{b_A^{i-1}} \text{ and } k_i \geq \alpha \ , \\ \Delta y_{\text{min}} & \text{if } x_{b_A^i} < x_{b_A^{i-1}} \text{ and } k_i < \alpha \ , \end{cases}$$

where Δy_{min} and Δy_{max} are positive constants which are discussed in Section IV-A, $k_i = \frac{y_{b_A^i} - y_{b_A^{i-1}}}{x_{b_A^{i-1}} - x_{b_A^i}}$, and α is a positive threshold. Δy_1 is chosen as $\frac{\Delta y_{\text{min}} + \Delta y_{\text{max}}}{2}$. Other choices of the function $g(b_A^{i-1}, b_A^i)$ can also be used given that it is consistent with the intuition discussed earlier.

Algorithm 2 Boundary Exploration

Input: Two boundary points b_A^0, b_A^1, a counter i initialized as 1, and a positive threshold ε

Output: p_{min}^A, the estimate of p_{min} by drifter A

```
1: Initialize $p_{\text{min}}^A$ to be $b_A^1$;
2: if $b_A^i$ is on the line $y = \frac{L}{2}$ then
3:     Send $p_{\text{min}}^A$ back to the deployment base, and exit;
4: end if
5: Compute $\Delta y_i = g(b_A^{i-1}, b_A^i)$;
6: if $y_{b_A^i} < \frac{L}{2}$ and $y_{b_A^i} + \Delta y_i > \frac{L}{2}$ then
7:     Set $\Delta y_i = \frac{L}{2} - y_{b_A^i}$;
8: end if
9: Move from point $b_A^i$ to point $p_A^{2i+1} = (x_{b_A^i}\ y_{b_A^i} + \Delta y_i)^T$,
   and then take the salinity measurement at $p_A^{2i+1}$;
10: if $S(p_A^{2i+1}) \neq B_{\text{th}}$ then
11:     Move downstream (or upstream) if $S(p_A^{2i+1}) < B_{\text{th}}$
        (or $S(p_A^{2i+1}) > B_{\text{th}}$), sample the river with fixed
        sampling interval $\Delta T$, and stop if there are two
        consecutive points $p_A$ and $p_A^{2i+2}$ satisfying $(S(p_A) -$
        $B_{\text{th}}) \times (S(p_A^{2i+2}) - B_{\text{th}}) \leq 0$. Explore $p_A p_A^{2i+2}$ using
        Algorithm 1 to obtain the boundary point $b_A^{i+1}$;
12: else
13:     Set $b_A^{i+1}$ to be $p_A^{2i+1}$;
14: end if
15: Compare $x_{b_A^{i+1}}$ with $x_{p_{\text{min}}^A}$: if $x_{b_A^{i+1}} < x_{p_{\text{min}}^A}$, update $p_{\text{min}}^A$
    with $b_A^{i+1}$;
16: if $i = 1$ then
17:     Send points $b_A^i, b_A^{i+1}$ to the deployment base;
18: end if
19: Set $i$ to be $i + 1$, and go to Step 2.
```

Drifter A keeps exploring the boundary and updates its estimate p_{min}^A by repeating the above procedure, stops exploration when it reaches a boundary point on the line $y = \frac{L}{2}$. At the end of the exploration, drifter A sends its estimate back to the deployment base. Part of the exploration process is shown in Fig. 1. The detailed exploration algorithm is given in Algorithm 2. For drifter A, a virtual boundary point b_A^0 (which is the same as b_A^1) is introduced purely for the sake of Step 5 when i is 1. Note that $g(b_A^0, b_A^1)$ is defined to be Δy_1. The counter i is used to label the points. Steps 6-8 guarantee that the boundary point on the line $y = \frac{L}{2}$ is explored. In the algorithm, communication with the deployment base occurs twice: i) drifter A sends back b_A^1 and b_A^2 once b_A^2 is determined, which will be used at Stage II when deploying drifter B; ii) drifter A sends back its estimate of p_{min} at the end of the exploration algorithm, which will be used to compute the estimate of p_{min}.

Note that the estimate might not be very accurate since drifter A might miss the exact local minimum due to the chosen Δy. If there exist three consecutive boundary points b_A^i, b_A^{i+1} and b_A^{i+2} such that $x_{b_A^i} > x_{b_A^{i+1}}$ and $x_{b_A^{i+2}} > x_{b_A^{i+1}}$, then there must be a local minimum on the boundary between b_A^i and b_A^{i+2}. By exploring the region of the boundary between b_A^i and b_A^{i+2}, we can obtain a refined estimate of a local minimum. Detailed procedures are given in Algorithm 3.

Algorithm 3 Boundary Exploitation

Input: Three points p^1, p^2 and p^3 satisfying $x_{p^1} > x_{p^2}$, $x_{p^3} > x_{p^2}$, and $y_{p^1} < y_{p^2} < y_{p^3}$, p_{start} initialized as p^3, and positive thresholds ε, ξ

Output: A point p satisfying $|y_p - y_{p^*}| \leq \xi$, where p^* is a local minimum (on the boundary between p^1 and p^3)

```
1: If $|y_{p^1} - y_{p^3}| \leq \xi$, output $p^2$ and exit;
2: Choose $\alpha \in (0, 1)$ randomly such that $y_p \neq y_{p^2}$, where
   $p = \alpha p^1 + (1 - \alpha) p^3$;
3: Move to point $p$ from $p_{\text{start}}$, and take salinity measure-
   ment $S(p)$;
4: if $S(p) \neq B_{\text{th}}$ then
5:     Move downstream (or upstream) if $S(p) < B_{\text{th}}$ (or
       $S(p) > B_{\text{th}}$), sample the river with fixed sampling
       interval $\Delta T$, and stop if there are two consecutive
       points $p'$ and $p''$ satisfying $(S(p') - B_{\text{th}}) \times (S(p'') -$
       $B_{\text{th}}) \leq 0$. Explore $p'p''$ using Algorithm 1 to obtain
       the boundary point $b_A^p$;
6: else
7:     Set $b_A^p$ to be $p$;
8: end if
9: Set $p_{\text{start}}$ to be $b_A^p$;
10: if $x_{b_A^p} > x_{p^2}$ then
11:     If $y_{b_A^p} > y_{p^2}$ (or $y_{b_A^p} < y_{p^2}$), update $p^3$ (or $p^1$) with
        $b_A^p$;
12: else if $x_{b_A^p} < x_{p^2}$ then
13:     If $y_{b_A^p} > y_{p^2}$ (or $y_{b_A^p} < y_{p^2}$), update $p^1$ (or $p^3$) with
        $p^2$, and $p^2$ with $b_A^p$;
14: else
15:     Go to Step 2;
16: end if
17: Go to Step 1.
```

In Algorithm 3, the input variable p_{start} tracks the location of drifter A, and the output is a point p which approximates a local minimum p^* with a prescribed precision ξ. Note that p^* must lie in the region bounded by points p^1 and p^3. Step 1 checks if p^2 is accurate enough; if true, the algorithm exits.

Step 2 generates a point p which has y_p different from y_{p^2}, and Steps 4-8 determine a boundary point b_A^p which lies on the line $y = y_p$. Step 9 updates p_{start}. Steps 10-13 update the three points p^1, p^2, p^3 based on the relative location of b_A^p; for example, if $x_{b_A^p} < x_{p^2}$ and $y_{b_A^p} > y_{p^2}$, the local minimum lies between p^2 and p^3, and therefore, we update p^1 with p^2 and p^2 with b_A^p to guarantee that the local minimum still lies between p^1 and p^3.

Remark 1 If there exist three consecutive boundary points b_A^i, b_A^{i+1} and b_A^{i+2} such that $x_{b_A^i} > x_{b_A^{i+1}}$ and $x_{b_A^{i+1}} = x_{b_A^{i+2}}$, then one can explore another boundary point b_A^{i+3}; since $x_{b_A^{i+2}} < x_{b_A^{i+3}}$ (this is guaranteed if $\Delta y_{\max}$ is chosen carefully; for detailed discussion, refer to Section IV-A), there must be a local minimum on the boundary between b_A^i and b_A^{i+3}, and the local minimum can be refined by setting p^1, p^2, p^3 in Algorithm 3 to be b_A^i, b_A^{i+1} and b_A^{i+3}. ■

B. Stages II and III

Once the deployment base receives the points b_A^1 and b_A^2 from drifter A, drifter B can be deployed to explore the other half of the boundary at Stage II. Drifter B treats b_A^2 as b_B^0 and b_A^1 as b_B^1, moves to point b_B^1 from the deployment base, and then explore the boundary points in a way similar to drifter A but with descending y coordinates. Algorithms 2 and 3 can be slightly modified for drifter B to explore the boundary and refine its estimate. At Stage III, once the deployment base receives $p_{\min}^A$ and $p_{\min}^B$, x_{estimate}, its estimate of the smallest x, is $\min(x_{p_{\min}^A}, x_{p_{\min}^B})$. Note that there is no direct interaction between drifters A and B.

C. Extension

Note that we could also deploy only one drifter to achieve the same estimate instead of using two drifters. If we want to achieve a highly accurate estimate of $p_{\min}$ with less time, multiple drifters can be deployed in the following way: i) first we deploy drifters A and B using Algorithm 2, ii) then whenever a new region of a local minimum is detected, we deploy a drifter to get a refined estimate using the exploitation Algorithm 3, and iii) when the deployment base receives all the estimates, it can determine an estimate of the point $p_{\min}$. To alleviate the dependency on the deployment base and deal with dynamic boundaries, drifters need be able to communicate with each other. Then our approach could be combined with connectivity maintenance methods and min consensus algorithms to deploy multiple drifters to get an estimate of $p_{\min}$.

IV. Analysis and Simulation

A. Convergence and Accuracy

Theorem 1 If the salinity field S is continuously differentiable and:

1) For any y satisfying $|y| \leq \frac{L}{2}$, $S(x,y)$ as a function of x is strictly increasing, which implies that for any point p with $y_p = y$, $\frac{\partial S}{\partial x}|_p \geq 0$, and $\frac{\partial S}{\partial x}|p = 0$ holds only for a finite set of points;

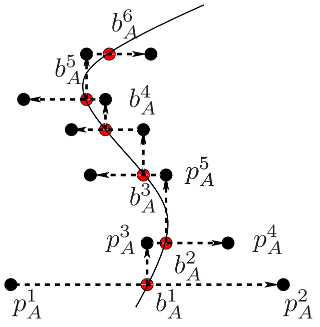

Fig. 1. Drifter A exploring the boundary.

2) For any point p on the boundary, if $\frac{\partial S}{\partial x}|_p = 0$, then $|\frac{\frac{\partial S}{\partial y}}{\frac{\partial S}{\partial x}}|_p$ is bounded;
3) For any point p on the boundary, $\frac{\partial S}{\partial y}|_p = 0$ holds only for a finite set of points, in which $\frac{\partial S}{\partial x}|_p > 0$,

then the following results hold:

a) Algorithms 1 and 2 stop after a finite number of steps;
b) If there is any local minimum lying strictly inside the region D, there exists $\Delta y_{\max} > 0$ (used in Algorithm 2) such that Algorithm 3 must be invoked to refine the local minimum, and stops after a finite number of steps;
c) $\forall \eta > 0$, there exists $\varepsilon > 0$ (used in Algorithms 1, 2 and 3), and $\xi > 0$ (used in Algorithm 3), such that $|x_{\text{estimate}} - x_{p_{\min}}| \leq \eta$, where $x_{\text{estimate}} = \min(x_{p_{\min}^A}, x_{p_{\min}^B})$, and $p_{\min}^A$ and $p_{\min}^B$ are obtained by drifters A and B using Algorithms 1-3.

Proof: Due to condition 1), given the salinity threshold B_{th}, for any y satisfying $|y| \leq \frac{L}{2}$, there is only one x that satisfies $S(x,y) = B_{\text{th}}$. Therefore, the boundary can be represented as $x = h(y)$ for some function h. Note that h is continuously differentiable because $\frac{\partial S}{\partial x}$ and $\frac{\partial S}{\partial y}$ exist and are continuous, and $\frac{dx}{dy} = -\frac{\frac{\partial S}{\partial y}}{\frac{\partial S}{\partial x}}$ is well defined because of condition 2), which guarantees that for any point p on the boundary, x_p is finite. Due to condition 3), the boundary is not a vertical line, and can be partitioned into a finite number of regions $\Omega = \{R_1, \ldots, R_{n-1}\}$ based on the sign of the function $\frac{\partial S}{\partial y}$ as follows:

- First find $P(\Omega)$, i.e., the set of points intersecting with the lines $|y| = \frac{L}{2}$ and points (on the boundary and strictly inside the region D) satisfying $\frac{\partial S}{\partial y} = 0$;
- Then sort points in $P(\Omega)$ into $p_1, p_2, \ldots, p_n$ according to ascending y coordinates, where $n \geq 2$ (note that $y_{p_1} = -\frac{L}{2}$ and $y_{p_n} = \frac{L}{2}$);
- Region R_i is the region of the boundary between points p_i and p_{i+1}, where $i = 1, 2, \ldots, n-1$.

Based on this partition, we define

$$K = \min_{i=1,2,\ldots,n-1} |y_{p_{i+1}} - y_{p_i}| \ ,$$

which is a positive constant due to condition 3). Since drifter B starts its boundary exploration from b_A^1 (namely, the first boundary point determined by drifter A) and uses the same algorithms as drifter A, the exploration process using drifters A and B is equivalent to deploying only drifter A to explore the whole boundary from $y = -\frac{L}{2}$ to $y = \frac{L}{2}$. Therefore, we may not explicitly distinguish drifter B from drifter A in the following analysis. Now we are ready to prove the results.

a) Algorithm 1 explores p^1p^2 (which is parallel to the x axis) to determine a boundary point $b = (x_b \ y_b)^T$ where $y_b = y_{p^1}$. Since $S(x_b, y_b) = B_{\text{th}}$ and the salinity field is continuous, for any $\varepsilon > 0$, there exists $\delta > 0$ such that for any $|x - x_b| \leq \delta$, $|S(x, y_b) - S(x_b, y_b)| \leq \varepsilon$. With $f(p^1, p^2)$ being $\frac{p^1+p^2}{2}$ or functions given in Eqs. (1) and (2), $|x_{p^3} - x_b|$ keeps decreasing; eventually $|x_{p^3} - x_b| \leq \delta$ and the algorithm stops. Algorithm 2 eventually stops since drifter A explores boundary points with ascending y coordinates until reaching $y = \frac{L}{2}$ and the step size is always larger than or equal to $\Delta y_{\text{min}} > 0$.

b) Suppose there is at least one local minimum lying strictly inside the region D. Choose

$$0 < \Delta y_{\text{max}} \leq \frac{K}{2} \ . \tag{3}$$

It can be verified that, via contradiction, there must be at least two points that are explored in any region R_i for $i = 1, 2, \ldots, n-1$ by either drifter A or B using Algorithm 2, which implies that there are at least four points explored in any two adjacent regions R_i and R_{i+1} for $i = 1, 2, \ldots, n-2$.

Note that for any local minimum point p^* lying strictly inside the region D, $\frac{\partial S}{\partial y}|_{p^*} = 0$, i.e., p^* must belong to $P(\Omega) \setminus \{p_1, p_n\}$; therefore, there must exist i such that point p^* lies between the regions R_i and R_{i+1}. In these two regions, there must be at least four points explored. We choose two points p_A, p_B satisfying $y_{p_A} < y_{p_B}$ from the region R_i such that $|y_{p_A} - y_{p^*}|$ and $|y_{p_B} - y_{p^*}|$ are smaller than other explored points in region R_i. Without loss of generality, we assume that for any point p' in region R_i, $\frac{\partial S}{\partial y}|_{p'} > 0$, and then we have $x_{p_B} < x_{p_A}$. Similarly, we choose two points p_C, p_D satisfying $y_{p_C} < y_{p_D}$ from the region R_{i+1} such that $|y_{p_C} - y_{p^*}|$ and $|y_{p_D} - y_{p^*}|$ are smaller than other explored points in region R_{i+1}. Since for any point p' in region R_{i+1}, $\frac{\partial S}{\partial y}|_{p'} < 0$, we have $x_{p_C} < x_{p_D}$. If $x_{p_B} < x_{p_C}$, then p_A, p_B, p_C invokes Algorithm 3 to refine the local minimum p^*; if $x_{p_B} = x_{p_C}$, then p_A, p_B, p_D invokes Algorithm 3 to refine the local minimum p^*; if $x_{p_B} > x_{p_C}$, then p_B, p_C, p_D invokes Algorithm 3 to refine the local minimum p^*. In summary, for any local minimum strictly inside the region D, Algorithm 3 must have been invoked to refine the local minimum.

It can be verified, via contradiction, that there is only one local minimum and no local maximum on the boundary between p^1 and p^3 when Algorithm 3 is invoked. The existence of only one local minimum guarantees that $|y_{p^1} - y_{p^3}|$ in Algorithm 3 keeps decreasing, and eventually $|y_{p^1} - y_{p^3}| \leq \xi$ holds, which guarantees the termination of Algorithm 3.

c) There are two cases for a point p_{min} depending on its y coordinate: i) $y_{p_{\text{min}}}$ equals to $\frac{L}{2}$ or $-\frac{L}{2}$, and ii) $|y_{p_{\text{min}}}| < \frac{L}{2}$. For both cases and any $\eta > 0$, we show that a point p satisfying $|x_p - x_{p_{\text{min}}}| \leq \eta$ must be found given properly chosen ε and ξ.

If $y_{p_{\text{min}}} = \frac{L}{2}$ in case i), p_{min} is the point p_n in the partition, and must be explored by drifter A (refer to Steps 6-8 in Algorithm 2). Suppose the two points p^1 and p^2 (that invoke Algorithm 1) satisfy $x_{p^1} < x_{p^2}$. Since $S(x, \frac{L}{2})$ for $x \in [x_{p^1}, x_{p^2}]$ is continuous and strictly increasing, $S(x, \frac{L}{2})$ is a one-to-one mapping from $[x_{p^1}, x_{p^2}]$ to $[S(p^1), S(p^2)]$. Therefore, $x = S^{-1}(z)$ for $z \in [S(p^1), S(p^2)]$, the inverse mapping of $S(x, \frac{L}{2})$, exists and is continuous. Then for any $\eta > 0$, there exists $\varepsilon_n > 0$ such that as long as $|S(p^n_{\text{estimate}}) - B_{\text{th}}| \leq \varepsilon_n$ where p^n_{estimate} is the output of Algorithm 1, $|S^{-1}(S(p^n_{\text{estimate}})) - S^{-1}(B_{\text{th}})| = |x_{p^n_{\text{estimate}}} - x_{p_n}| \leq \eta$ because $S^{-1}(B_{\text{th}}) = x_{p_n}$. Similarly, if $y_{p_{\text{min}}} = -\frac{L}{2}$ in case i), p_{min} is the point p_1, and must be explored by drifter B. For any $\eta > 0$, there exists $\varepsilon_1 > 0$ such that as long as $|S(p^1_{\text{estimate}}) - B_{\text{th}}| \leq \varepsilon_1$, $|x_{p^1_{\text{estimate}}} - x_{p_1}| \leq \eta$. In both scenarios, ξ can be chosen to be an arbitrary positive constant.

In case ii), $|y_{p_{\text{min}}}| < \frac{L}{2}$. Then we only need to consider local minima that lie strictly inside the region D. As shown in b), every local minimum must be refined via Algorithm 3 by either drifter A or B.

Suppose p_{min} is a local minimum p_i for $i \in \{2, 3, \ldots, n-1\}$. Since in condition 3) $\frac{\partial S}{\partial y}|_{p_i} = 0$ requires that $\frac{\partial S}{\partial x}|_{p_i} > 0$, we can choose δ such that for any point p on the boundary that satisfies $|y_p - y_{p_i}| \leq \delta$, $\frac{\partial S}{\partial x}|_p > 0$ because the boundary is continuous. For such p, $|\frac{dx}{dy}| = |-\frac{\frac{\partial S}{\partial y}}{\frac{\partial S}{\partial x}}| \leq C_p$ where C_p depends on the point p. Let $\Delta x = x_p - x_{p_i}$ and $\Delta y = y_p - y_{p_i}$. Then we have $\left|\frac{\Delta x}{\Delta y}\right| = \frac{|\int_{y_{p_i}}^{y_p} \frac{dx}{dy} dy|}{|\Delta y|} \leq \frac{\int_{\min(y_p, y_{p_i})}^{\max(y_p, y_{p_i})} |\frac{dx}{dy}| dy}{|\Delta y|}$ $\leq \max_{p'} C_{p'}$, where p' is on the boundary between p and p_i. The maximum of $C_{p'}$ (denoted as C_i) exists because the boundary is continuously differentiable and the part of the boundary between p and p_i is closed. Given $\eta > 0$, there exists $\xi_i = \min(\frac{\eta}{2C_i}, \delta)$ such that for any p on the boundary that satisfies $|y_p - y_{p_i}| \leq \xi_i$, we have $|x_p - x_{p_i}| \leq C_i|y_p - y_{p_i}| \leq \frac{\eta}{2}$. In Algorithm 3, p^1 and p^3 will eventually fall into the ξ_i neighborhood of the local minimum p_i because $|y_{p^1} - y_{p^3}|$ keeps decreasing and p_i always lies between p^1 and p^3. Let the output of Algorithm 3 be p^i_{estimate}. Let p be a point on the boundary such that $y_p = y_{p^i_{\text{estimate}}}$. Then given ξ_i, we have $|y_p - y_{p_i}| = |y_{p^i_{\text{estimate}}} - y_{p_i}| \leq \xi_i$, $|x_p - x_{p_i}| \leq \frac{\eta}{2}$. Since p^i_{estimate} must be obtained by applying Algorithm 1, given $\eta > 0$, there exists $\varepsilon_i > 0$ such that for any p^i_{estimate} satisfying $|S(p^i_{\text{estimate}}) - B_{\text{th}}| \leq \varepsilon_i$, $|x_{p^i_{\text{estimate}}} - x_p| \leq \frac{\eta}{2}$ as argued in case i). In summary, given $\eta > 0$, there exists ξ_i and ε_i such that $|x_{p^i_{\text{estimate}}} - x_{p_i}| = |x_{p^i_{\text{estimate}}} - x_p + x_p - x_{p_i}| \leq \eta$.

Suppose $P_{\text{lmin}}(\Omega) \subseteq P(\Omega)$ is the set of local minima (which could include p_1 and p_n in the partition). For any $p_i \in P_{\text{lmin}}(\Omega)$, there exists ξ_i and ε_i such that the output of Algorithm 3 p^i_{estimate} satisfies $|x_{p^i_{\text{estimate}}} - x_{p_i}| \leq \eta$, i.e., $x_{p_i} - \eta \leq x_{p^i_{\text{estimate}}} \leq x_{p_i} + \eta$. Let $\xi = \min_{p_i \in P_{\text{lmin}}(\Omega)} \xi_i$ and $\varepsilon = \min_{p_i \in P_{\text{lmin}}(\Omega)} \varepsilon_i$, then the estimate $x_{\text{estimate}} = \min_{p_i \in P_{\text{lmin}}(\Omega)} x_{p^i_{\text{estimate}}}$. Since $x_{p_{\text{min}}} = \min_{p_i \in P_{\text{lmin}}(\Omega)} x_{p_i}$, $x_{p_{\text{min}}} - \eta \leq x_{\text{estimate}} \leq x_{p_{\text{min}}} + \eta$. Thus, c) holds. ■

Remark 2 The assumption that the salinity field is continuously differentiable is commonly made (e.g., [3]). Assumption 1) is reasonable in practical applications if tidal forcing

TABLE I

ESTIMATION ERROR WITH VARYING PARAMETERS

PARAM	Value	$S_{\text{linear}}, h_{\text{sin}}$	$S_{\text{exp}}, h_{\text{sin}}$	$S_{\text{linear}}, h_{\text{poly}}$	$S_{\text{exp}}, h_{\text{poly}}$
Alg. 3	off	$7.07e{-}3$	$7.07e{-}3$	0.193	0.193
	on	$2.56e{-}4$	$4.71e{-}4$	$1.49e{-}2$	$4.13e{-}4$
ε	$1e{-}2$	$2.56e{-}4$	$4.71e{-}4$	$1.49e{-}2$	$4.13e{-}4$
	$1e{-}4$	$1.14e{-}7$	$1.43e{-}4$	$1.06e{-}4$	$5.19e{-}5$
	$1e{-}6$	$1.02e{-}7$	$1.54e{-}8$	$5.36e{-}7$	$1.70e{-}8$
	$1e{-}8$	$1.75e{-}7$	$3.88e{-}7$	$1.03e{-}7$	$1.34e{-}9$
ξ	$1e{-}2$	$2.56e{-}4$	$4.71e{-}4$	$1.49e{-}2$	$4.13e{-}4$
	$1e{-}4$	$2.29e{-}3$	$7.07e{-}3$	$7.78e{-}3$	$1.37e{-}3$
	$1e{-}6$	$7.07e{-}3$	$1.30e{-}4$	$2.25e{-}4$	$1.11e{-}2$
	$1e{-}8$	$9.03e{-}4$	$1.82e{-}5$	$4.13e{-}2$	$9.79e{-}4$
$\begin{bmatrix}\Delta y_{\text{min}} \\ \Delta y_{\text{max}}\end{bmatrix}^T$	$(1\ 2)$ $(1\ 1.5)$	$2.56e{-}4$	$4.71e{-}4$	$1.49e{-}2$	$4.13e{-}4$
	$(1\ 4)$ $(1\ 4)$	$1.24e{-}3$	$7.07e{-}3$	$1.96e{-}3$	$9.97e{-}3$
	$(3\ 4)$ $(2\ 4)$	$1.42e{-}14$	$8.14e{-}4$	$2.91e{-}4$	$5.61e{-}5$
α	0.1	$9.98e{-}3$	$3.96e{-}3$	$3.47e{-}2$	$7.45e{-}3$
	0.3	$1.95e{-}3$	$5.56e{-}4$	$7.63e{-}4$	$1.25e{-}3$
	0.5	$2.56e{-}4$	$4.71e{-}4$	$1.49e{-}2$	$4.13e{-}4$
	0.7	$8.09e{-}4$	$4.67e{-}3$	$7.06e{-}3$	$1.37e{-}2$
	0.9	$4.23e{-}4$	$7.50e{-}5$	$5.00e{-}5$	$6.95e{-}3$
$N_A = 1e{-}4$	0	$3.19e{-}10$	$1.52e{-}8$	$3.48e{-}8$	$7.03e{-}7$
	1	$6.03e{-}4$	$7.79e{-}4$	$1.66e{-}4$	$7.93e{-}4$
	2	$4.37e{-}4$	$1.63e{-}4$	$3.07e{-}4$	$2.40e{-}4$
	3	$2.10e{-}4$	$2.88e{-}4$	$3.79e{-}4$	$2.15e{-}5$
$N_A = 1e{-}2$	0	$2.43e{-}7$	$3.70e{-}8$	$3.70e{-}10$	$3.62e{-}6$
	1	0.0728	0.0849	0.0891	0.0601
	2	0.0603	0.0470	0.0410	0.0341
	3	0.0420	0.0290	0.0433	0.0491

is weak, and Assumption 2) holds for well mixed estuaries. Assumption 3) is imposed purely for the sake of proof, and we would like to relax it in our future work. ■

B. Simulations

We use $L = 40$ and $x \in [0, 400]$ for the region D. Then we generate four types of salinity fields by choosing the salinity field as i) $S_{\text{linear}}(x, y) = M_1(x - h(y))$, and ii) $S_{\text{exp}}(x, y) = e^{M_2(x-h(y))}$, and by choosing the function $h(y)$ as i) $h_{\text{sin}}(y) = A_s \sin w_s y$, and ii) $h_{\text{poly}}(y) = ay^5 + by^4 + cy^3 + dy^2 + ey + f$ which has a unique global minimum. The parameters $M_1, M_2, A_s, w_s, a, b, \ldots, f$ are chosen so that for $x \in [0, 400]$ and $|y| \le 20$ the salinity is between 0 and 33 (note that salinity ranges from 0 to 33ppt in estuaries). The salinity threshold is set to be 10. It can be verified that the conditions in Theorem 1 are all satisfied. For $h_{\text{sin}}(y)$, $\frac{K}{2}$ in Eq. (3) can be calculated as 2.146, while for $h_{\text{poly}}(y)$, $\frac{K}{2}$ can be calculated as 1.513. We use $\varepsilon = \xi = 1e{-}2$, $\Delta y_{\text{min}} = 1$, $\Delta y_{\text{max}} = 2$ for the salinity field with h_{sin}, $\Delta y_{\text{max}} = 1.5$ for the salinity field with h_{poly}, and $\alpha = 0.5$ as the default parameters. Here, Δy_{max} is chosen to satisfy Eq. (3).

We first examine the effect of Algorithm 3 on the estimate. As shown in Table I, the estimation error $|x_{p_{\text{min}}} - \min(x^A_{p_{\text{min}}}, x^B_{p_{\text{min}}})|$ decreases dramatically when Algorithm 3 is used. If we decrease ε in Algorithm 1, the estimation error decreases dramatically, as observed in the row "ε". In contrast, if we decrease ξ in Algorithm 3, the estimation error does not change much, as observed in the row "ξ". In the row "$(\Delta y_{\text{min}}\ \Delta y_{\text{max}})$", the first of the pair "(1 2), (1 1.5)" is used for the first two salinity fields, and the second is used for the last two salinity fields. The results show that even when Eq. (3) is violated, the estimation error could still be very small. This observation makes our approach promising for practical applications. If we increase α in the function $g(b_A^{i-1}, b_A^i)$, the algorithm will explore more boundary points, which decreases the estimation error.

To examine the accuracy of the estimate with noisy salinity measurements, we generate four types of noises: i) type 0, i.e., no noise, ii) type 1, i.e., uniform in $[-N_A, N_A]$ where $N_A > 0$, iii) type 2, i.e., Gaussian with mean 0, stand deviation $\frac{N_A}{3}$, and being limited to $[-N_A, N_A]$, and iv) type 3, i.e., Gaussian with mean 0 and stand deviation $\frac{N_A}{3}$. We change ε to be $1e{-}6$. The estimation errors for $N_A = 1e{-}4$ and $N_A = 1e{-}2$ are listed in Table I. In all cases, the algorithms stop but run longer; estimation errors increase if the noise magnitude increases, and have the same order as the noise magnitude. In general, type 1 noises tend to cause relative larger estimation errors.

V. CONCLUSION

In this paper, we studied drifter deployment to determine a point on the freshwater-saltwater boundary that minimizes its x coordinate, and proposed boundary point detection, boundary exploration, and boundary exploitation algorithms which adaptively explore the boundary and can achieve an arbitrarily accurate estimate under certain assumptions on the salinity field. In practical applications, there are several challenges: i) the measurements from position sensors could also have noise and the drifters are subject to (usually unknown) disturbances in river environments, ii) sensors, actuators, and communication devices could fail, and iii) the salinity field can change with the depth of the river and time (which implies that the boundary can be dynamically changing). Dealing with these issues will be part of our future work.

REFERENCES

[1] A. D. Jassby, W. J. Kimmerer, S. G. Monismith, C. Armor, J. E. Cloern, T. M. Powell, J. R. Schubel, and T. J. Vendlinski, "Isohaline position as a habitat indicator for estuarine populations," *Ecological Applications*, vol. 5, pp. 272–289, Feb. 1995.

[2] R. Brockway, D. Bowers, A. Hoguane, V. Dove, and V. Vassele, "A note on salt intrusion in funnel-shaped estuaries: Application to the incomati estuary, mozambique," *Estuarine, Coastal and Shelf Science*, vol. 66, pp. 1–5, Jan. 2006.

[3] M. Fossati and I. Piedra-Cueva, "Numerical modelling of residual flow and salinity in the río de La Plata," *Applied Mathematical Modelling*, vol. 32, pp. 1066–1086, 2008.

[4] R. J. Uncles and J. A. Stephens, "The freshwater-saltwater interface and its relationship to the turbidity maximum in the tamar estuary, United Kingdom," *Estuarie*, vol. 16, pp. 126–141, Mar. 1993.

[5] Y. Ru and S. Martinez, "Freshwater-saltwater boundary detection using mobile sensors — Part II: Drifter movement," to appear at *IEEE Conf. on Decision and Control*, Dec. 2011.

[6] Y. Chen, K. Moore, and Z. Song, "Diffusion boundary determination and zone control via actuator-sensor networks (MAS-net) – challenges and opportunities," in *Proc. of SPIE Conf. on Intelligent Computing: Theory and Applications II*, Apr. 2004.

[7] Y. Cao and R. Fierro, "Dynamic boundary tracking using dynamic sensor nets," in *Proc. of the 45th IEEE Conf. on Decision and Control*, Dec. 2006, pp. 703–708.

[8] S. B. Andersson, "Curve tracking for rapid imaging in afm," *Ieee Transactions On Nanobioscience*, vol. 6, pp. 354–361, Dec. 2007.

[9] Z. Jin and A. L. Bertozzi, "Environmental boundary tracking and estimation using multiple autonomous vehicles," in *Proc. of the 46th IEEE Conf. on Decision and Control*, Dec. 2007, pp. 4918–4923.

[10] F. Zhang, E. Fiorelli, and N. E. Leonard, "Exploring scalar fields using multiple sensor platforms: Tracking level curves," in *Proc. of the 46th IEEE Conf. on Decision and Control*, Dec. 2007, pp. 3579–3584.

[11] S. Susca, S. Martínez, and F. Bullo, "Monitoring environmental boundaries with a robotic sensor network," *IEEE Transactions on Control Systems Technology*, vol. 16, pp. 288–296, Mar. 2008.

[12] A. Joshi, T. Ashley, Y. R. Huang, and A. L. Bertozzi, "Experimental validation of cooperative environmental boundary tracking with on-board sensors," in *Proc. of 2009 American Control Conference*, Jun. 2009, pp. 2630–2635.