Convergence-Guaranteed Time-Varying RRT Path Planning for Profiling Floats in 4-Dimensional Flow

Van T. Huynh[a], Matthew Dunbabin[a,b], and Ryan N. Smith[c]

[a] School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, QLD, Australia
[b]Institute for Future Environments, Queensland University of Technology, Brisbane, QLD, Australia
[c]Department of Physics and Engineering, Fort Lewis College, Durango, CO, USA
email: tv.huynh@qut.edu.au; m.dunbabin@qut.edu.au; rnsmith@fortlewis.edu

Abstract

This paper presents an extension to the Rapidly-exploring Random Tree (RRT) algorithm applied to drifting autonomous underwater vehicles. The proposed algorithm is able to plan paths that guarantee convergence in the presence of time-varying ocean dynamics. The method utilizes 4-Dimensional ocean model prediction data as an evolving basis for expanding the tree from the start location to the goal. The performance of the proposed method is validated through Monte-Carlo simulations. Results illustrate the importance of the temporal variance in path execution and demonstrate the convergence performance of the proposed methods.

1 Introduction

Collection of long-term data is necessary to enhance our understanding of large-scale, dynamic ocean processes, as well as their complex spatio-temporal variability. Such long-term data gathering requires a persistent presence in the environment. One method to capture this time-series data is through the use of Autonomous Underwater Vehicles (AUVs) that are capable of long-period deployments (on the order of weeks to months), e.g., gliders [1, 2] and profiling floats [3]. These vehicles decrease their actuation capability to save power for persistent navigation. However, with the compromise of actuation, it is challenging to reliably achieve set-point trajectories due to significant external disturbances, e.g., ocean currents. Therefore, a path planning strategy is required for such actuation-reduced aquatic vehicles that is able to predict and utilize these extreme disturbances.

Current work on path planning with uncertainties for persistent platforms is primarily focused on underwater gliders [4–8]. There are limited works considering path planning for profiling floats [9–11]. Recent research has focused on applications of profiling floats for physical oceanography and underwater imaging [12]. Nevertheless, this prior research has been the motivation to consider operation of profiling floats in highly-dynamic, coastal environments, or restrained areas like embayments. When operating in such environments, the vehicles are likely to be diverted away from the goal or the predetermined sensing location due to constant perturbation by external forces generated by spatio-temporally varying currents.

This paper presents a convergence-guaranteed path planning method for persistent ocean monitoring that takes into account both disturbances of ocean currents and their time-dependency characteristic. The approach is inspired by the ideas from Rapidly-exploring Random Tree (RRT) and the work of Hsu et al. [13] so that time-dependency and efficiency are considered. Due to the vehicle's operation in highly dynamic environments, a controllability verification step is included in the algorithm to ensure the convergence of the algorithm. Analysis for convergence is also provided. To the best of the authors' knowledge, this is the first paper dealing with the convergence-ensured, time-varying sampling-based algorithm under dynamic disturbances.

2 Background

Path planning for standard actuated AUVs (with propellers and control surfaces) in the presence of water currents has been considerably researched over the last decade. A number of approaches have been presented, such as Linear Optimization and Mixed Integer Linear Programming [14, 15], and A* search [16]. However, these works mostly assume vehicles operating in a horizontal 2-Dimensional (2D) plane.

Alvarez and Caiti [17] utilized numerical integration to predict rough paths for AUVs subject to ocean currents. Extending the work in [17], Alvarez et al. [18] developed a Genetic Algorithm (GA)-based path planning that takes into account the space-time structure of the ambient environment. In Alvarez et al. [18] the authors assume that the AUV travels at a constant speed with respect to the sea floor, i.e., the vehicle needs to vary its speed according to the current field so as to maintain its constant total velocity along the whole journey. In a similar scenario, Garau et al. [19] used the A* algorithm to search for a minimum-time path that optimizes energy consumption. Garau et al. [20] continued to apply the algorithm in the previous work in the more realistic situation of current fields. However, these works assume quasi steady-state ocean currents and the AUVs propulsive thrust is constant without actuation limits. These assumptions breakdown in real-ocean environments, particularly in the presence of strong ocean currents where many trajectories can become infeasible.

Addressing the problem of vehicle actuation lim-

its, Kruger et al. [21] use a multi-dimensional cost function to generate minimum energy, time optimum, continuous paths in estuarine environments. Inspired by this work, Witt and Dunbabin [22] dealt with the planning problem of generating time and energy optimum trajectories for actuated AUVs in dynamic ocean environments. This research showed the potential of utilizing tidal currents in such a way that vehicle range, endurance, and waypoint tracking are significantly improved. Considering all these works, there is still little research on path planning algorithms that examine the improvement of navigation for profiling drifters based on the controllability of these vehicles.

Sampling-based path planning algorithms such as the probabilistic roadmap (PRM) [23] and Rapidly-exploring Random Trees (RRTs) [24] are becoming increasingly implemented. These algorithms are well-known for their probabilistic completeness characteristic, i.e., rapidly exploring the space of interest and quickly returning a feasible solution if one exists. PRM* and RRT* have been proposed to generate paths that are provably asymptotically optimal as the computation time increases [25]. Extensions of these sampling-based algorithms have also been developed to tackle uncertainty whilst maintaining asymptotic optimality [26]. These algorithms only consider robots working without disturbances and do not deal with extreme disturbances[1] whose magnitude might be at the same order as that of the robot's controls. As such, there will be no guarantee for convergence of the algorithms in such scenarios and cannot be applied to directly to solve the motion planning problem.

Attempting to apply sampling-based algorithms in extreme conditions, Rao and Williams [27] used RRTs for determining energy-optimal paths that consider ocean current fields. A comparison of paths generated by the RRT and grid-based methods is presented demonstrating that RRTs offer an improvement in avoiding high-energy, shallow regions. However, [27] assumes that the ocean currents are time-invariant. Thus, this technique, to some extent, is unrealistic since ocean current magnitude and direction significantly vary temporally [28].

Motivated by the ideas from [13], a novel algorithm is proposed here that addresses extreme disturbances on underwater vehicles, and guarantees convergence to the end goal. The proposed methodology is applied to the general case of path planning for profiling floats. This class of vehicle is considered the general case since profilers are minimally-actuated, as compared to standard AUVs with thrusters and control surfaces. Therefore, the algorithm can be extended to other AUVs having more control authority.

3 Equations of Motion

A profiling float is an underwater vehicle that freely drifts with the ambient currents, and can actively change its depth during a mission by adjusting its buoyancy.

[1]Extreme disturbance is defined as external forces from ambient environment having the same order of magnitude as that of the internal available control force.

Intelligently controlling depth allows the profiler to choose appropriate ocean currents to carry the vehicle to a desired destination. An example profiling float is shown in Figure 1, with a detailed description found in [11].

Figure 1: A profiling float prototype manufactured by Teledyne Webb Research, Inc.

Movement of the profiling float is determined primarily by ocean currents. The equations of motion of this vehicle involve both the ocean dynamics and the vehicle kinematics. Here, we present both the oceanic propulsion, referred to as Ocean Model Predictions (OMPs), and the kinematics of the float.

3.1 Ocean Model Prediction

The path planning method proposed in this paper is built upon the predictions of a regional ocean model. The ocean model utilized is the regional ocean model system (ROMS). Details on ROMS can be referenced in Shchepetkin and McWilliams [28, 29].

The ROMS data to be used in this paper is generated by the Jet Propulsion Laboratory (JPL), California Institute of Technology. ROMS ultimately provides hindcasts, nowcasts and hourly forecasts (up to 72 hours) for the Southern Californian Bight (SBC) at various depths. Particularly, ROMS data divides non-uniformly the vertical direction into 24 discrete depths, ranging from 0 to 2000 m. Consequently, ROMS data is presented in four dimensions as it gives predictions at different points of time and at various horizontal and vertical locations.

3.2 Kinematics of Profiling Floats

Provided that the profiling float (underwater robot) is able to have neutral buoyancy at a predetermined depth, and that the centre of gravity (CG) and the centre of buoyancy (CB) are on the same vertical axis in the body-fixed coordinate frame $\mathcal{B}$, we define the vehicle's state by the vector $\boldsymbol{\eta} = (b_1, b_2, b_3)^\top$. The vector $\boldsymbol{\eta}$ depicts the position of the vehicle relative to the inertial coordinate along longitude b_1, latitude b_2, and depths b_3.

Given that the ocean prediction data is presented in discrete-time format and that the profiling float normally has unchanged control input over a single time epoch, the motion can be considered in the discrete-time domain. The discrete-time equations of motions for profiling floats in Linear Time Varying state space form are:

$$\bar{\boldsymbol{\eta}}(k+1) = \mathbf{I}_2(k)\bar{\boldsymbol{\eta}}(k) + \mathbf{B}_{\bar{\boldsymbol{\eta}}}(k)\boldsymbol{u}(k) \quad (1)$$

$$\bar{\boldsymbol{\nu}}(k) = \mathbf{C}(k)\bar{\boldsymbol{\eta}}(k). \quad (2)$$

Here, $\bar{\boldsymbol{\eta}}(k) = (b_1(k), b_2(k))^T \in \mathbb{R}^2$ is the state of the system representing its horizontal location at time epoch k. $\bar{\boldsymbol{\nu}}$ stands for the system output. $\mathbf{I}_2$ is the 2×2 identity matrix. The output matrix $\mathbf{C} = \mathbf{I}_2$.

The control input is denoted by $\boldsymbol{u}(k)$ and $\boldsymbol{u}$ represents the depth selections of the vehicle to

harness the appropriate ocean current. Since the ROMS data contains 24 depth bins, we can denote $\mathcal{U} = \{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_j, \ldots, \mathbf{u}_{24}\}$ the set of control input vectors for the discrete-time system, where $\mathbf{u}_j = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ with unique nonzero entry j. By specifying $\boldsymbol{u}$ in this form, the control input is bang-bang. Moreover, a control input of $\mathbf{u}_1$ is interpreted as commanding the vehicle to the ocean surface, whereas a control input of $\mathbf{u}_j, \forall j \in \{2, ..., 24\}$, is understood as commanding the profiling float to a specific depth corresponding to the ROMS output discretisation. In this study it was assumed the diving time of the profiler is much smaller compared to the ROMS discretization and therefore could be neglected.

The input matrix $\mathbf{B}_{\bar{\boldsymbol{\eta}}}(k)$ is a 2×24 matrix giving the eastward and northward ocean velocities at each depth at location $\bar{\boldsymbol{\eta}}$ from the ROMS model. To obtain $\mathbf{B}_{\bar{\boldsymbol{\eta}}}(k)$, we interpolate a set $\mathbb{B}$ consisting of 4 similar matrices output from the ocean prediction model. Each matrix is the discrete grid point of the ocean model prediction that defines the bounding box around the current position $\bar{\boldsymbol{\eta}}$ of the vehicle. We can consider $\mathbb{B}$ a lookup table for the predicted ocean currents at a given spatiotemporal location $\bar{\boldsymbol{\eta}}(k)$. In this paper, the prediction model contains a total of 7826 grid points. Multiplication of $\mathbf{u}$ and interpolated sets of currents $\mathbf{B}_{\bar{\boldsymbol{\eta}}}(k)$ becomes ocean currents propulsion at time epoch k corresponding to location $\bar{\boldsymbol{\eta}}$.

4 Problem statement

The path planning problem for the profiler is to seek control inputs $\mathbf{u} \in \mathcal{U}$ such that the vehicle moves from an initial configuration $\boldsymbol{\eta}_0$ to a final configuration $\boldsymbol{\eta}_f$ along locally absolutely continuous curve $\gamma : I \longmapsto Q$ when we apply such control input $\mathbf{u}$. The end conditions for γ is $\gamma(0) = \boldsymbol{\eta}_0$ and $\gamma(t_f) = \boldsymbol{\eta}_f$. To steer the vehicle, we use only depths of the vehicle as direct control inputs. Therefore, the path planning problem is to find the set of depths $\mathbf{u}(k)$ to achieve the final state $\gamma(t_f)$ as close as possible to $\boldsymbol{\eta}_f$ at time t_f.

The choice of depth inputs is defined to be accomplished only when the vehicle successfully reaches the goal destination. Section 4.1 describes a prerequisite for this condition to be satisfied, specifically a controllability check of control actions (ocean currents) in a specific region of ocean needs to be undertaken. If the controllability check for a region indicates that the region is able to direct the profiler in sufficient directions spanning the space, it is a fully-controllable region.

In brief, a solution is desired that finds a path for a profiler subject to (1) and (2) travelling from an initial starting waypoint to a predetermined goal waypoint, and satisfying the following conditions:

- the planned path is placed inside a fully-controllable region.
- the planned path needs to be time-varying and takes into account dynamical disturbances from ocean currents of the ambient environment.
- convergence to the end goal is ensured even when extreme disturbances exist.

4.1 Controllability Conditions

Given a profiling float and OMPs, is it possible to steer the vehicle to any destination within finite time? This can be considered by examining an accessibility set, the set of all admissible motion from each system state resulting from the depth dependent currents at a point in space, as presented by the authors in [11]. The primary result from [11] is stated in Proposition 1 and defines the conditions for controllability.

Proposition 1. *(Conditions for controllability) The system given in Eqs.* (1)-(2) *is controllable if and only if the convex hull of the accessibility set $\mathcal{A}$ is compact and contains the origin.*

5 Time-invariant RRT Algorithm

Algorithm 1: The time-invariant RRT algorithm with incorporation of 4D OMP for AUV

Input: An initial waypoint $\bar{\boldsymbol{\eta}}_0 = \{b_{10}, b_{20}\}$, a target waypoint $\bar{\boldsymbol{\eta}}_f = \{b_{1f}, b_{2f}\}$, predicted ROMS data, a world space (the extracted region in ROMS) $\mathcal{Q} = \{NE_{corner}, SW_{corner}\}$, and the termination ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_f, r_f}$.

Output: The switching sequence of depth control inputs.

```
world ← CreateWorld(NE_corner,SW_corner)
z_start ← GenerateNode(η̄_0,0,0,0,1)
z_end ← GenerateNode(η̄_f,0,1)
V ← {z_start} ; E ← ∅ ; G ← (V, E)
if ‖ η̄_0 − η̄_f ‖≤ r_f then
    path ← {z_start; z_end}
else
    NumPaths← 0
    while NumPaths < 1 ∧ ¬MaxIteration do
        [G, flag] ← ExtendTree(G, z_end, world)
        NumPaths ← NumPaths + flag
path ← FindMinimumPath (G,z_start,z_end)
return Depths, path
```

A time-invariant RRT that incorporates OMPs to plan a path for a profiling float through a quasi-steady ocean will firstly be presented. The construction of time-invariant RRT incorporating 4D ROMS and depth control guidance is summarised in Algorithm 1.

The first step requires the creation of a *world* space that is large enough to contain the initial and target waypoints. This world space specified by the South Western and North Eastern corners as shown in Figure 2 is situated in the region which is verified by the controllability-check process to have varied directions of ocean currents.

The *GenerateNode* function in Algorithm 1 is used to create a tree node. Each RRT node is a structure consisting of six fields. The first two contain the latitudinal and longitudinal coordinates with the third field being a flag signalling the node in the tree that connects directly to the *EndNode*. The fourth stores a cost associated with the node (the length of path from *StartNode*). The fifth field is a back-tracking pointer

pointing to the front node, with the sixth being the depth relating the ROMS depth index corresponding to the ocean current that propels the vehicle towards the *EndNode*. Therefore, the constructed RRT *tree* is an array whose elements are six-field structures.

Before expanding the tree, the algorithm checks to see if the start node and end node are connectible by an ocean current propulsion. Particularly, if *StartNode* and *EndNode* falls within the ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_f, r_f}$ centred at the *EndNode* with radius r_f, the start node can connect directly to the end node without attempting to expand the tree.

If the start node is not connectible directly to the end node immediately, then the function *ExtendTree* will be activated. Figure 2 demonstrates the development of a time-invariant RRT tree when *ExtendTree* is called. Details of the *ExtendTree* function are presented in Algorithm 2. As this section is creating a time-invariant path planner, it assumes the ocean currents do not vary temporally over the planning horizon.

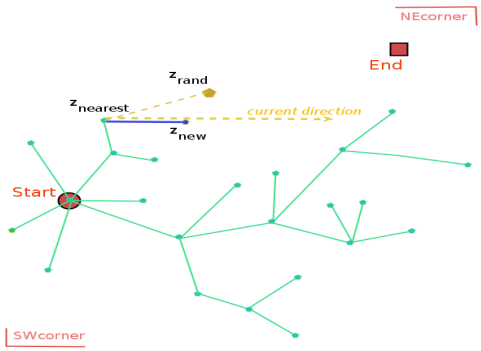


Figure 2: Tree expansion for the time-invariant algorithm illustrates how to extend $\boldsymbol{z}_{nearest}$ to $\boldsymbol{z}_{rand}$.

In Algorithm 2, at each loop epoch the function *ExtendTree* randomly generates a point in the horizontal plane $\boldsymbol{z}_{rand}$. It searches all nodes in the tree for a node $\boldsymbol{z}_{nearest}$ that is nearest to the random node. At each nearest node there are discrete layers of ocean currents which flows along different directions. Such oceanic layers are also time-dependent. If the assumption that ocean currents do not vary with time is upheld, Algorithm 2 will then choose a depth that yields a maximal ocean current from the nearest node $\boldsymbol{z}_{nearest}$ towards the random node $\boldsymbol{z}_{rand}$. The process of depth selection is carried out by the function *FindSegmentLength*. The *FindSegmentLength* function also returns the magnitude of the maximal ocean current and assigns it to *SegmentLength*. Based on direction of the selected ocean current (v_{cx}, v_{cy}) at the chosen depth (*NodeDepth*) a new node $\boldsymbol{z}_{new}$ is added to the tree. Cost of the new node will be equal to summation of the cost of $\boldsymbol{z}_{nearest}$ and the magnitude of the selected ocean current (*SegmentLength*). Due to dynamics of the minimally-actuated underwater vehicles, their movement is bounded by the magnitude of propulsion of ocean currents. Consequently, instead of expanding the nearest node directly to the random node, the new node is just a *SegmentLength* away from the nearest node. After connecting the new node to the tree, a backtracking pointer of the new node points to $\boldsymbol{z}_{nearest}$ and the last field of the new node is assigned the ROMS index of the selected ocean depth (*NodeDepth*).

When there is a node, named *next–to–end*, that is connectible to the end (goal) node, the algorithm will activate *FindMinimumPath* which returns a path originating from the start node to the goal node.

5.1 Comparison with prior work

The combination of Algorithms 1 and 2 differs from Rao's work [27] in three important areas; Firstly, the algorithm is used to guide the minimally-actuated vehicles whose control input is only depth-switching signals, whereas the algorithm by Rao is applied for gliders. Secondly, the assumption that distances between degrees of latitude and longitude are fixed is relaxed. Thirdly, it is assumed a larger column of water for the vehicle to utilize.

Furthermore, the time-invariant RRT developed here differs from conventional RRT in the functions *ExtendTree*, *FindSegmentLength*, and *FindMinPath* that exploits OMP data in the development of the tree. More importantly, the algorithm returns a 3D RRT tree, accommodating both horizontal and vertical motions of the underwater vehicle. A visual representation is shown in Figure 3. The extra dimension in the tree represents the depth switchings for the vehicle.

It is worth expanding several aspects of the *ExtendTree* algorithm. Firstly, when extending the nearest tree leaf towards the random node, it chooses a dive depth in ROMS with the largest current among the ocean currents at the nearest tree leaf location. Secondly, the tree will expand at each epoch irrespective of direction. That is, when the algorithm is unable to select a depth with a current vector pointing in the Line of Sight (LoS) direction, it will still augment to the tree a node in the opposite of LoS under the propulsion of minimum current at the tree leaf as shown in Figure 4. This characteristic differs the algorithm from conventional RRT that operates without effects from disturbances. Furthermore, it should be noted that each node contains a cost which is the accumulation of magnitude of ocean current velocities. Finally, contrary to the conventional RRT, the path planning algorithm proposed here guarantees convergence. Details of the convergence proof are presented in Section 6.1.

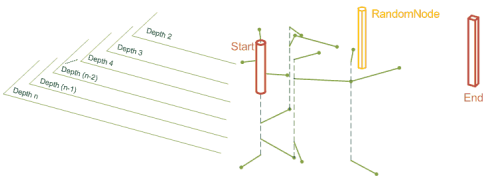


Figure 3: Illustration of 3D structure of the tree.

Unlike the $A*$ approach in [27], this approach does not grid the whole search space, but interpolates ROMS inside the horizontal space for the ocean currents at each node. The continuity of the space is preserved by taking this approach. Interpolating for the ocean current velocities also guarantees that our

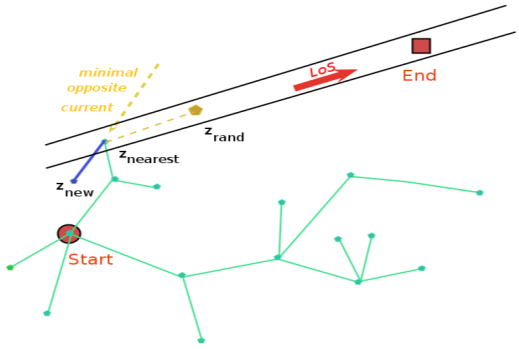


Figure 4: Expansion of the tree in the opposite LoS.

path planning algorithm aligns with real operation of underwater vehicles whose motion is not discrete but continuous.

Algorithm 2: A tree extension algorithm for the time-invariant RRT

Input: The RRT tree $G = (V, E)$, the desired end node $\boldsymbol{z}_{end}$, the world space ($world$), and the ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_f, r_f}$.
Output: An extended tree $G' = (V', E')$, a flag ($flag$) signalling connectability between the extended tree with the end node

$V' \leftarrow V; E' \leftarrow E$
$LoopFlag \leftarrow 0$
while $LoopFlag == 0$ **do**
  $\boldsymbol{z}_{rand} \leftarrow$ `RandomPointSeeding`($world$)
  $\boldsymbol{z}_{nearest} \leftarrow$ `Nearest`$(G, \boldsymbol{z}_{rand})$
  $[SegmentLength, NodeDepth, v_{cx}, v_{cy}] \leftarrow$ `FindSegmentLenth` $(\boldsymbol{z}_{nearest}, \boldsymbol{z}_{rand})$
  $cost \leftarrow$ `InferNodeCost`$(\boldsymbol{z}_{nearest}, SegmentLength)$
  $\boldsymbol{z}_{new} \leftarrow$ `Steer`$(\boldsymbol{z}_{nearest}, v_{cx}, v_{cy}, cost, NodeDepth)$
  $V' \leftarrow V' \cup \boldsymbol{z}_{new}; E' \leftarrow E' \cup \{(\boldsymbol{z}_{nearest}, \boldsymbol{z}_{new})\}; G' \leftarrow (V', E')$
  $LoopFlag \leftarrow 1$
if $\|\boldsymbol{z}_{new} - \boldsymbol{z}_{end}\| \leq r_f$ **then**
  $flag \leftarrow 1$
  $G' \leftarrow$ `MarkConvergence`(G')
else
  $flag \leftarrow 0$
return G', $flag$

6 Time-varying RRT Algorithm

In order to exploit the time-varying characteristics of ocean currents for controlling the path of profiling drifters, a sampling-based path planning algorithm is proposed.

Algorithm 3 describes a novel time-varying RRT utilising OMP. Creation of the world space is equivalent to the time-invariant RRT. However, the *GenerateTimeNode* function is now used to generate the nodes for the time-dependent tree and consists of eight fields; the six fields generated by *GenerateNode* in Algorithm 1, and two additional fields containing a branch index and time. This eight field vector is represented as $\boldsymbol{x}$. The starting node (tree root) is assigned

Algorithm 3: The time-varying RRT algorithm with incorporation of 4D OMP for AUV

Input: An initial waypoint $\bar{\boldsymbol{\eta}}_0 = \{b_{10}, b_{20}\}$, a target waypoint $\bar{\boldsymbol{\eta}}_f = \{b_{1f}, b_{2f}\}$, predicted ROMS data, a world space (the extracted region in ROMS) $\mathcal{Q} = \{NE_{corner}, SW_{corner}\}$, and the termination ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_f, r_f}$.
Output: The switching sequence of depth control inputs.

$world \leftarrow$ `CreateWorld`$(NE_{corner}, SW_{corner})$
$\boldsymbol{x}_{start} \leftarrow$ `GenerateTimeNode`$(\bar{\boldsymbol{\eta}}_0, 0, 0, 1, 1, 1)$
$\boldsymbol{x}_{end} \leftarrow$ `GenerateTimeNode`$(\bar{\boldsymbol{\eta}}_f, 0, 0, 1, branch_{end}, t_{end})$
$V \leftarrow \{\boldsymbol{x}_{start}\}$; $E \leftarrow \emptyset$; $G \leftarrow (V, E)$
if `ConnectionCheck`$(t \leftarrow 1, \boldsymbol{x}_{start}, \boldsymbol{x}_{end}, r_f)$ **then**
  $path \leftarrow \{\boldsymbol{x}_{start}; \boldsymbol{x}_{end}\}$
else
  forall the $NumberOfTrees$ **do**
    $NumPaths \leftarrow 0$
    while $NumPaths < 1$ **do**
      $[G, flag] \leftarrow$ `ExtendTVTree`$(G, \boldsymbol{x}_{end}, world)$
      $NumPaths \leftarrow NumPaths + flag$
$path \leftarrow$ `FindMinimumPath` $(G, \boldsymbol{x}_{start}, \boldsymbol{x}_{end})$
return $Depths$, $path$

$t = 1$ and a branch index of 1. However, only the goal location can be set for the end node since the final time and branch can only be determined following successful expansion of the tree.

Algorithm 4: The tree expansion algorithm for time-varying RRT.

Input: The RRT tree $G = (V, E)$, the desired end node $\boldsymbol{x}_{end}$, the world space ($world$), and the ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_f, r_f}$.
Output: An extended tree $G' = (V', E')$, a flag ($flag$) signalling connectability between the extended tree with the end node

while *Tree is not extended* **do**
  $\boldsymbol{z}_{rand} \leftarrow$ `RandomPointSeeding`($world$)
  $[\boldsymbol{z}_{nearest}, t_{nearest}] \leftarrow$ `Nearest`$(G, \boldsymbol{z}_{rand})$
  $[SegmentLength, NodeDepth, v_{cx}, v_{cy}] \leftarrow$ `FindTVSegmentLenth` $(\boldsymbol{z}_{nearest}, t_{nearest}, \boldsymbol{z}_{rand})$
  $cost \leftarrow$ `InferNodeCost`$(\boldsymbol{z}_{nearest}, SegmentLength)$
  $\boldsymbol{z}_{new} \leftarrow$ `Steer`$(\boldsymbol{z}_{nearest}, v_{cx}, v_{cy}, cost, NodeDepth)$
  $t_{new} = t_{nearest} + 1$
  $\boldsymbol{x}_{new} \leftarrow \{\boldsymbol{z}_{new}, t_{new}\}$
  $V' \leftarrow V' \cup \boldsymbol{x}_{new}; E' \leftarrow E' \cup \{(\boldsymbol{x}_{nearest}, \boldsymbol{x}_{new})\}; G' \leftarrow (V', E')$
if $\|\boldsymbol{x}_{new} - \boldsymbol{x}_{end}\| \leq r_f$ **then**
  $flag \leftarrow 1$
  $G' \leftarrow$ `MarkConvergence`(G')
else
  $flag \leftarrow 0$
return G', $flag$

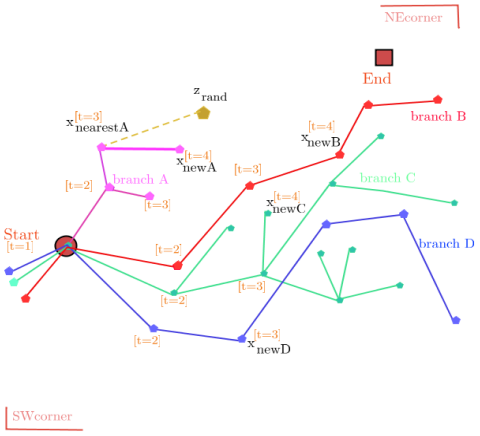


Figure 5: Illustration of time-varying expansion of the tree. Each node assigned with a time, and different branches correspond to different colors.

For every evolution cycle of the RRT, *ExtendTVTree* generates a random spatial point $\boldsymbol{z}_{rand}$. The RRT then searches against all nodes and returns the node ($\boldsymbol{x}_{nearest}$) that is spatially closest to that random point. The $\boldsymbol{x}_{nearest}$ has a node position $\boldsymbol{z}_{nearest}$ at its epoch $t_{nearest}$.

The function *FindTVSegmentLength* interpolates the ROMS data at the location $\boldsymbol{z}_{nearest}$ along both time and depth dimensions and returns the maximum ocean current originating from $\boldsymbol{z}_{nearest}$ pointing towards $\boldsymbol{z}_{rand}$. The actual vehicle motion between the epoch $t_{nearest}$ to $t_{nearest}+1$ is then the determined by the kinodynamics of the profiling float with the selected ocean current. Should the LoS-directed ocean current not exist, the minimal ocean current against the LoS will be used. The RRT adds a new node $\boldsymbol{x}_{new}$ to the tree at epoch $(t_{nearest}+1)$ with position $\boldsymbol{z}_{new}$.

Using this approach, many branches can be built with varying time indices. This approach is illustrated in Figure 5. Initially, at time 1, corresponding to the origin, the first random node creates a new branch and initiates a node $\boldsymbol{x}_{newA1[2]}$. Here, $\boldsymbol{x}_{newAn[t]}$ represents an n^{th} node of branch A at time t and at location $\boldsymbol{z}_{newAn}$.

On subsequent evolution cycles, if the random point is closest to the origin the algorithm will create a new branch, e.g., $\boldsymbol{x}_{newB1[2]}$, next to $\boldsymbol{x}_{start[1]}$. Consequently, the branches evolve independently, one in the direction of $\boldsymbol{x}_{newA1[2]}$ and the other one in the direction of $\boldsymbol{x}_{newB1[2]}$. Then $\boldsymbol{x}_{newA1[2]}$ and $\boldsymbol{x}_{newB1[2]}$ will represent the predicted positions of the vehicle at time epoch 2. A visual illustration of time-varying expansion of the tree is depicted in Figure 5 where the different branches are represented by colors.

Note that this approach is subtly different to the construction of multiple RRTs. The branches emanating from the root node grow until one branch reaches the termination condition at the goal location. To synthesize additional trees, $NumberofTrees$ in Algorithm 3 is set to a value different from unity. The *FindMinimumPath* function will extract the path with a minimum cost from amongst these branches.

6.1 Analysis

The prove of convergence of the RRT algorithm for the profiler with incorporation of OMP is given below. This proof is inspired by the works of Jouffroy et al. [10] and Smith and Huynh [11].

Convergence of the time-varying algorithms

Lemma 1. *Given the controllability check showing that desired waypoints are in fully-controllable free space χ_{free}, we can surely obtain planned paths to the goal region around $\boldsymbol{\eta}_f$.*

Proof. In the deep-sea environments, there are several layers of currents. According to Jouffroy et al. [10], when there are more than or equal to three ocean currents whose directions spanning the horizontal space, we can guide the vehicle to the destination - actually around the destination.

Specifically, we often calculate

$$\theta = [\theta_1, \theta_2, \theta_3]^\top, \tag{3}$$

where $\theta_i, \forall i = 1, 2, 3$, are the difference angle between the currents velocities $(v_{c1}, v_{c2}, v_{c3})^\top$ at each layer and the line of sight.

Consequently, if $(v_{c1}, v_{c2}, v_{c3})^\top$ spans the space, we can express any point in the horizontal space as a linear combination $\sum_{i=1}^{3} \alpha_i v_{ci}$. Therefore, by applying control $u = \text{argmin}\,(\theta)$, the profiler can always be directed towards the region around the end node. □

Probabilistic completeness

Lemma 2. *The time-varying RRT are probabilistic complete, i.e., the expected number of iteration required to connect $\boldsymbol{x}_{start}$ to χ_{end} is limited.*

The proof for probabilistic completeness is based on the work of Karaman and Frazzoli [25]. This lemma extends to the case of disturbing environments. We do this based on the controllability check of the region of interest. This controllability check assures that vertices of the RRT tree are attraction sequences of the free space χ_{free}, and are attracted to the potential well around the goal. If the vehicle lies in the partial controllability areas, the attraction of sequences in χ_{free} into $\mathcal{B}_{\bar{\boldsymbol{\eta}}_f, r_f}$ cannot be guaranteed.

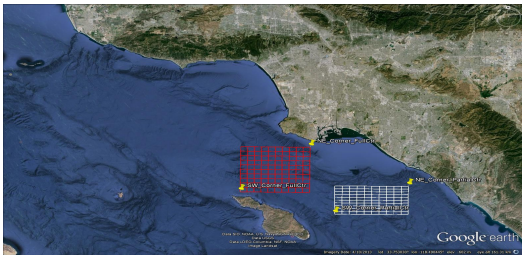
7 Results

The algorithms presented in the previous sections are evaluated using OMP from the Southern Californian Bight. Figure 6(a) shows the regions where paths are to be planned for the profiling floats. Figure 6(b) shows the controllability map using OMP data for the regions in Figure 6(a). The controllability map is generated using the process depicted in Section 4.1 and detailed [11].

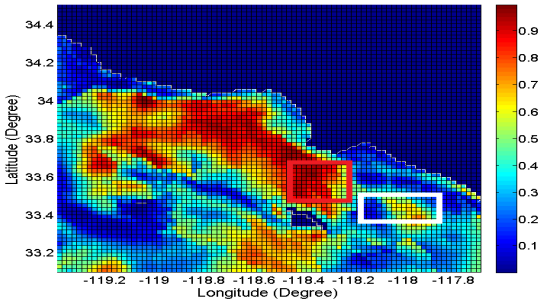
In this work, the proposed algorithms are verified in two types of regions; (1) near fully-controllable, and (2) partially-controllable regions. In Figure 6(b), the red box indicates a controllable region, whilst the white box shows a less (or partial) controllable region. Landmass is represented in dark blue and corresponds to completely uncontrollable areas.

Table 1: Summary of 100 Monte Carlo simulations results for both Time-Varying (TV) and Time-Invariant (TI) algorithms with full (FullCtr) and partial (PartialCtr) controllability regions.

	Convergence (%)	Avr Switchings	Avr Path Length (km)	Real Convergence (%)	Avr Difference (km)
FullCtr TI	97	44	21.983	4	8.593
FullCtr TV	100	40	22.489	100	0.405
PartialCtr TI	58	94	25.668	3	17.337
PartialCtr TV	61	48	22.012	100	0.368



(a) Google Earth image of the geographic region of interest (Southern Californian Bight) illustrating full (red) and partial (white) controllability regions.



(b) The controllability heat map generated from ocean model predictions. Blue refers to areas of low controllability, and red are areas of high controllability.

Figure 6: (a) The geographic area of interest highlighted by grid rectangles. (b) The controllability heat map computed for this geographic region. Note that the two images are not the same scale.

The planning regions (gridded rectangles) are chosen to have edge lengths of greater than 20 km so as to present the algorithms with opportunities to plan long-distance missions than can span several days to weeks.

Figure 9 presents a RRT tree generated from the time-varying algorithm. This tree shows the evolution of the profiling float locations at different points of time. Particularly, in this figure, red and blue subtrees represents the time-dependent evolution of the tree from the root node. Since the tree evolution accounts for longitude, latitude, depth, and time, this tree is actually of 4 dimensions.

Figure 7 shows examples of paths returned from multiple executions of the Time-Varying algorithms with the same start and end goal locations. The green line is a connecting line between the starting waypoint and the destination goal. Distance between waypoints in Figure 7 is more than 22 km. Lines in red, blue, and brown are planned trajectories returned from the different executions of the algorithms. Due to randomness of the RRT different paths are obtained after each run, however, all paths converge to the area around the goal destination.

Figure 8 illustrates the expected trajectory and depth profile when executing one of the planned paths of Figure 7. As can be seen, the profiling float changes depths along the path (orange points) to select currents that steers the vehicle towards the goal.

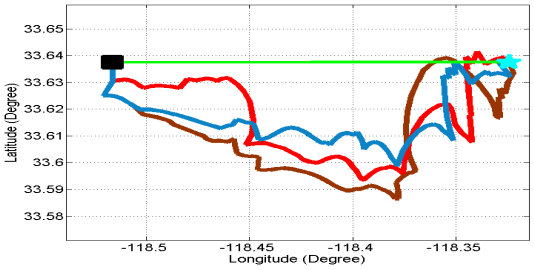


Figure 7: Three planned paths for the profiler when using the time-varying algorithm with the same start and goal locations.

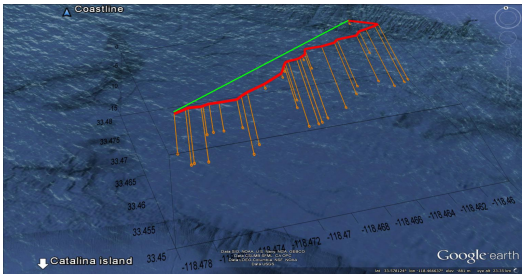


Figure 8: 3D representation of a planned path when using time-varying algorithm. Orange lines show selected depths along the path.

7.1 Monte-Carlo simulations of the path planner

Monte-Carlo simulations of the algorithms in both the fully-controllable and partially-controllable regions shown in Figure 6(a) were performed. For each region, 100 pairs of starting and goal waypoints are randomly generated for Monte-Carlo simulation with the following conditions: 1) fully-controllable time-invariant RRT, 2) fully-controllable time-varying RRT, 3) partially-controllable time-invariant RRT, and 4) partially-controllable time-varying RRT.

Table 1 summarises the results of the simulations for each case. Efficiency of the algorithms are assessed via five quality factors including: 1) percentage of convergence amongst 100 runs of the algorithms, 2)

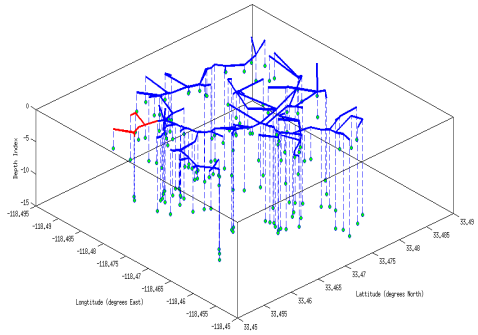


Figure 9: 3D Tree when using time-varying algorithm. Dark blues and red shows the tree structure in horizontal plane, and dotted lines with green ends represents depth switchings at each node.

an average number of depth switchings, 3) an average length of planned path, 4) percentage of real convergence, and 5) average difference (distance) between the destination goal and the planned final position. The column of real convergence shows the number of planned paths amongst 100 runs that actually terminate inside the ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_{\boldsymbol{f}},r_f}$ when implemented in the ocean model. Here an $r_f = 0.55$ km is chosen. This assessment criterion is necessary particularly for the time-invariant cases where the planned paths divert away from the destination goals once they are applied with time-varying data. Finally, the average distance criterion is the distance between the desired goal position and the actual end location of the profiler.

Table 1 shows the average number of depth switchings is greater than or equal to 40, which is equivalent to an actual deployment time of 40 hours. The average travel distance and time suggest the algorithms are suitable to motion planning for persistent navigation by profiling floats. The average number of depth switchings of the full controllability cases are less than those of the partial controllability ones. This phenomenon can be explained by the fact that the profilers operating in the partial-controllable region do not have a variety of choices of control actions, and thus tend to take longer paths which are sometimes away from the goal.

If the controllability check indicates the region is only partially controllable, both time-invariant and time-varying algorithms return less than 100% convergence. This is due to the fact that ocean currents in the region are biased towards some directions. As discussed in Section 6.1, vectors of the ocean currents in a region that do not span the space leads to uncontrollable states of the system. Additionally, the time-varying RRT is capable of searching more convergence paths than those of time-invariant even though both of the path planners operate in partially-controllable areas. In fact, the condition of time-independent currents in the time-invariant RRT restricts the number of variations of ocean currents' directions. Consequently, it is less likely that the time-invariant RRT can find a converged path as compared to the time-varying RRT. Nevertheless, whenever the time-varying path planner returned a path, it can guarantee that the path's real end position falls within the terminating ball $\mathcal{B}_{\bar{\boldsymbol{\eta}}_{\boldsymbol{f}},r_f}$.

A notable number emerging from the summary is that the profiler can be planned to converge to the goal almost surely (100%) when the time-varying RRT is applied inside the fully-controllable region. Even though the time-invariant algorithm shows 97 out of 100 simulations converge, only 4% of the planned paths terminating within $\mathcal{B}_{\bar{\boldsymbol{\eta}}_{\boldsymbol{f}},r_f}$ when executed in the ocean model. This fact is construed by observing that the average distance between the real end locations and the goal destinations for the fully-controllable time-invariant algorithm is slightly more than 8.5 km and much greater than r_f.

In summary, the time-varying RRT with prior controllability check formulates a combined algorithm that can guarantee convergence of the profiling floats. Also, the time-varying RRT found to be superior than the time-invariant one when they are applied to plan paths in 4D space.

8 Conclusion

This paper has presented a set of time-invariant and time-varying RRT algorithms that plan paths for underwater profiling floats based on predictions from an OMP that guarantee arrival to the goal. The algorithms are capable of incorporating time-varying dynamics of the ocean while still maintaining convergence properties along the path. A Monte-Carlo simulation demonstrates the effectiveness of the method, and shows in dynamic ocean current regions exhibiting near-full controllability that 100% of the time-varying RRT algorithm generated paths converged. This is compared to approximately 4% that converged when considering time-invariant conditions.

Acknowledgments

V. Huynh was supported by QUTPRA scholarship from the Queensland University of Technology. R.N. Smith was supported through a Fort Lewis College Deans award for innovative research.

References

[1] O. Schofield, J. Kohut, D. Aragon, E.L. Creed, J.G. Graver, C. Haldman, J. Kerfoot, H. Roarty, C. Jones, D.C. Webb, and S. Glenn. Slocum gliders: Robust and ready. *Journal of Field Robotics*, 24(6):473–485, 2007.

[2] G. Griffiths, C. Jones, J. Ferguson, and N. Bose. Undersea gliders. *Journal of Ocean Technology*, 2(2):64–75, 2007.

[3] D. Roemmich, S. Riser, R. Davis, and Y. Desaubies. Autonomous profiling floats: Workhorse for broad-scale ocean observations. *Marine Technology Society Journal*, 38:21–29, 2004.

[4] J. G. Graver. *Underwater Gliders: Dynamics, Control and Design*. PhD thesis, Princeton University, 2005.

[5] D.A. Paley, F. Zhang, and N.E. Leonard. Cooperative control for ocean sampling: The glider coordinated control system. *IEEE Transactions*

on Control Systems Technology, 16(4):735–744, July 2008.

[6] N. E. Leonard, D. A. Paley, R. E. Davis, D. M. Fratantoni, F. Lekien, and F. Zhang. Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in monterey bay. *Journal of Field Robotics*, 27(6):718–740, 2010.

[7] R. N. Smith, Y. Chao, P. P. Li, D. A. Caron, B. H. Jones, and G. S. Sukhatme. Planning and implementing trajectories for autonomous underwater vehicles to track evolving ocean processes based on predictions from a regional ocean model. *International Journal of Robotics Research*, 29 (12):1475–1497, October 2010.

[8] R. N. Smith, M. Schwager, S. L. Smith, B. H. Jones, D. Rus, and G. S. Sukhatme. Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *Journal of Field Robotics*, 28(5):714 – 741, Sept/Oct 2011.

[9] R. N. Smith and M. Dunbabin. Controlled drift: An investigation into the controllability of underwater vehicles with minimal actuation. In *Proceedings of the Australasian Conference on Robotics and Automation*, July 2011.

[10] J. Jouffroy, Q. Zhou, and O. Zielinski. On active current selection for lagrangian profilers. *Modeling, Identification and Control*, 2013.

[11] R. N. Smith and V. T. Huynh. Controlling buoyancy-driven profiling floats for applications in ocean observation. *IEEE Journal of Oceanic Engineering*, 2014.

[12] C. Roman, G. Inglis, and B. McGilvray. Lagrangian floats as sea floor imaging platforms. *Continental Shelf Research*, 31(15):1592 – 1598, 2011.

[13] D. Hsu, R. Kindel, J. C. Latombe, and S. Rock. Randomized kinodynamic motion planning with moving obstacles. *The International Journal of Robotics Research*, 21(3):233–255, 2002.

[14] 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 *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1083–1088, May 2008.

[15] N.K. Yilmaz, C. Evangelinos, P. Lermusiaux, and N.M. Patrikalakis. Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming. *Oceanic Engineering, IEEE Journal of*, 33(4):522 –537, oct. 2008. ISSN 0364-9059.

[16] K.P. Carroll, S.R. McClaran, E.L. Nelson, D.M. Barnett, D.K. Friesen, and G.N. William. AUV path planning: an A* approach to path planning with consideration of variable vehicle speeds and multiple, overlapping, time-dependent exclusion zones. In *Proceedings of the Symposium on Autonomous Underwater Vehicle Technology*, pages 79–84, Jun 1992.

[17] A. Alvarez and A. Caiti. Interaction of autonomous underwater vehicles with variable scale ocean currents. In *Proceedings of the IFAC World Conference Systems*, 2002.

[18] A. Alvarez, A. Caiti, and R. Onken. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *IEEE Journal of Oceanic Engineering*, 29(2):418–429, 2004.

[19] B. Garau, A. Alvarez, and G. Oliver. Path planning of autonomous underwater vehicles in current fields with complex spatial variability: an A* approach. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 194 – 198, april 2005.

[20] B. Garau, M. Bonet, A. Alvarez, S. Ruiz, and A. Pascual. Path planning for austonomous underwater vehicles in realisitic oceanic current fields: Application to glider in the western mediterranean sea. *Maritime Research*, 6:5–22, 2009.

[21] D. Kruger, R. Stolkin, A. Blum, and J. Briganti. Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments. In *Proc. IEEE International Conference on Robotics and Automation*, Rome, Italy, 2007.

[22] J. Witt and M. Dunbabin. Go with the flow: Optimal path planning in coastal environments. In J. Kim and R. Mahony, editors, *Proceedings of the 2008 Australasian Conference on Robotics and Automation*, Canberra, ACT, 2008.

[23] L.E. Kavraki, P. Svestka, J.-C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, Aug 1996.

[24] S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.

[25] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7): 846–894, 2011.

[26] A. Bry and N. Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 723–730, May 2011.

[27] D. Rao and S. B. Williams. Large-scale path planning for underwater gliders in ocean currents. In *Proceedings of the Australian Conference on Robotics and Automation*, 2009.

[28] A.F. Shchepetkin and J.C. McWilliams. Quasi-monotone advection schemes based on explicit locally adaptive dissipation. *Monthly Weather Review*, 126:1541–1580, 1998.

[29] A. F. Shchepetkin and J. C. McWilliams. The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean Modelling*, 9: 347–404, 2005.