2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments.

Dov Kruger, Rustam Stolkin, Aaron Blum, Joseph Briganti

Abstract—**This paper addresses the problems of automatically planning Autonomous Underwater Vehicle (AUV) paths which best exploit complex current data, from computational estuarine model forecasts, while also avoiding obstacles. In particular we examine the possibilities for a novel type of AUV mission deployment in fast flowing tidal river regions which experience bi-directional current flow. These environments are interesting in that, by choosing an appropriate path in space and time, an AUV may both bypass adverse currents which are too fast to be overcome by the vehicle's motors and also exploit favorable currents to achieve far greater speeds than the motors could otherwise provide, while substantially saving energy. The AUV can "ride" currents both up and down the river, enabling extended monitoring of otherwise energy-exhausting, fast flow environments. The paper discusses suitable path parameterizations, cost functions and optimization techniques which enable optimal AUV paths to be efficiently generated. These paths take maximum advantage of the river currents in order to minimize energy expenditure, journey time and other cost parameters. The resulting path planner can automatically suggest useful alternative mission start and end times and locations to those specified by the user. Examples are presented for navigation in a simple simulation of the fast flowing Hudson River waters around Manhattan.**

I. Introduction

CONVENTIONAL deployment of Autonomous Underwater Vehicles (AUVs) typically involves a human user specifying a mission path in terms of a series of waypoints, [1]. The AUV then uses onboard control strategies to keep to a path which passes through these waypoints. Various control techniques respond intelligently to sensor stimuli in order to correct position errors and also to autonomously plan local path deviations for obstacle avoidance, e.g. [2], [3]. These techniques have proved adequate for long range missions in open sea, where forecast currents are relatively homogeneous over large regions and where deviations from the intended path can be relatively easily and safely corrected.

In contrast we wish to deploy an AUV in a swift flowing tidal river estuary which contains currents that are faster than the maximum speed of the robot; vary rapidly with position in the river (often reversing direction with a few meters of position change); and which change significantly with time (often reversing direction over a tidal cycle). It is a fundamental requirement, for deploying a robot in these environments, that the path be optimized with respect to a-priori knowledge of these currents which can be reliably derived from computational estuarine model now-casts and forecasts, [4], [5]. Unfortunately, the optimal choice of path in these complex and varying conditions will rarely be obvious to a human user. It is also desirable that the robot be able to regularly re-plan the remainder of its mission in order to compensate for previous motion errors, avoid newly detected obstacles and exploit updated and corrected current information which can be communicated to the robot periodically throughout its mission. We present a technique which enables an AUV to autonomously plan optimal paths which exploit useful currents, avoid adverse currents, maximize speed, minimize energy expenditure and avoid obstacles. The user must specify approximate locations and times for the beginning and end of the mission. Additional midpoints or waypoints may be specified if desired. The path planner is capable of suggesting useful modified or alternative positions and times for any user specified points.

Much of the AUV path planning literature focuses on short term local path planning for the avoidance of detected obstacles, e.g. [2], [3]. A smaller body of literature addresses the problems of long range global path planning and some of this work explores ways of optimizing paths with respect to current velocity data. Previous work predominantly approaches this problem by reducing the space to a small set of nodes, assigning costs to straight line edges connecting nodes, and then searching the resulting graph for the best route. In contrast, our technique requires no such discretization or simplification of the motion space.

Sequeira and Ribeiro, [6], [7], reduce the motion space to a graph of connected nodes which include the vertices of known obstacle regions, modeled as cuboidal blocks. Currents are handled in a very simplistic fashion, with large regions of significant current represented as cuboidal blocks of constant current speed and direction. The centroids of these current regions are also added as nodes to the graph of motion space. Energy costs are now added to graph edges in proportion to any adverse current velocities which they traverse and a minimum cost route through the graph is

Manuscript received September 15, 2006. This work was supported by the U.S. Office of Naval Research, Swampworks, grant #N00014-05-1-00632.

D. Kruger and R. Stolkin are researchers at the Center for Maritime Systems, Stevens Institute of Technology, Hoboken, NJ 07030 USA. Phone: 201-216-5300; fax: 201-216-8214; e-mail: DKruger@stevens.edu, RStolkin@stevens.edu.

A. Blum and J. Briganti, Jr., are undergraduate students in the Department of Computer Science at Stevens Institute of Technology. E-mail ABlum@stevens.edu, JBriganti@stevens.edu.

found using the Dijkstra algorithm. More recent work (Garau et al., [8]) is restricted to considerations of motion on a 2D plane. The plane is divided into a coarse grid of equi-spaced nodes, which are connected by edges to form a graph. Each edge is assigned a cost according to estimated journey time, assuming constant motor speed, and the minimum cost route is found with an A* search. This approach has a number of drawbacks for estuarine navigation. Firstly, since only 2D motion is addressed, the technique is unable to plan paths which dive or climb to avoid adverse current layers or to exploit useful current layers. This ability is crucial to successful navigation in a tidal estuary where two stratified layers of water are often traveling in opposite directions at high speeds. Secondly, the technique is unable to plan for variable AUV power output. Thus an AUV is unable to fully exploit the currents for certain applications where drifting with useful currents at minimal motor speeds is essential for prolonging mission lifetimes. Thirdly, although the authors consider spatially varying current fields, the problem of current velocities which vary with time is not addressed. In contrast, in our estuarine application, currents vary substantially over relatively short timescales, often reversing direction over tidal cycles. The ability to handle temporal current variation is important. Fourthly, Garau et al. note a substantial computation time of around 35 seconds to plan paths of order 1km length, presumably using a dedicated offline CPU. If the method were to be extended to the necessary 3D motion space, this time would increase significantly, even more so if only part of the CPU time of an onboard AUV controller could be assigned to the task. This makes the method inappropriate for enabling an AUV to regularly re-plan the remainder of its path in response to updated current or obstacle data during the mission. Carroll et al., [9], also use A* search to find safe paths through a map of known obstacles and unsafe regions, derived from a detailed database of depth information, obstacles and exclusion zones. This work does not address the problem of path optimization with respect to currents. The authors also note significant computation times for A* path planning (order 10-100s CPU time with currents sampled only once every nautical mile). Alvarez et al., [10], do address variable current speeds in a 3D environment, using genetic algorithms to find an optimal path while avoiding convergence on local cost minima. The approach also involves partitioning the search space into a relatively coarse grid of cells and then searching a space of possible paths which link these cells. 3D currents are simplified to representations as a small number of discreet 2D layers. The current speeds encountered are relatively small (maximum 0.65ms^{-1}) compared to those addressed in this paper and the currents are approximately locally constant (varying over tens or hundreds of km) in contrast to those addressed in this paper which vary significantly over distances of a few meters. Other uses of genetic algorithms include [11], which describes a method for optimal path planning for Unmanned Aerial Vehicles (UAVs) with respect to forecast wind velocities. The genetic algorithm enables the evolution of multiple candidate paths composed of component primitives (e.g. arcs and straight line sections). The authors also propose that the robot should regularly re-plan the remainder of its mission to take advantage of updated wind speed forecasts. An advantage of genetic algorithms is that they can be particularly robust against convergence on local minima. A disadvantage is that they require a large number of different path candidates to be considered and evaluated simultaneously which can be computationally expensive. Other approaches to handling complex, varying currents include Fuzzy Logic, e.g. [3] which is primarily concerned with rapid local control of the AUV to avoid detected obstacles in the presence of ocean currents. Long range global path planning is not addressed.

In contrast to previous work, our method offers a highly flexible path parameterization which avoids the need for any discretization or coarse partitioning of the motion space. The method allows for current velocities in any direction, which vary continuously in space and time, with arbitrary resolution to whatever level of accuracy can be supplied by the current model being used. The method enables the continuous adjustment of waypoints in both space and time with an arbitrarily fine resolution, i.e. no discretization, forming paths of any shape through 3D space and time. As well as finding an optimal route between user specified start and end points, the path planner is able to suggest modified locations and times for start and end points which may be advantageous to the mission. Path planning parameters can be easily modified by the user in order to emphasize different mission priorities, e.g. minimum energy expenditure versus minimum journey time. Unlike much of the literature, we consider energy costs due to acceleration in addition to drag forces. The efficient path planning routine runs rapidly (less than 1s with un-optimized Java code on an ordinary 2.6GHz PC) and so is suitable for enabling the robot to regularly re-plan the remainder of its mission in order to avoid newly detected obstacles and exploit any updated current velocity data which might become available during the mission.

II. Estuarine Circulation

An estuary has large currents that can easily exceed the top speed of an AUV. There are two repeatable primary driving forces; tides which oscillate the water with a predictable superposition of sinusoidal constituents, and gravitational flow, where a lens of lighter fresh water at the surface pushes out to sea, while heavier salt water flows in at the bottom. Superimposed on this are also wind effects which are far more variable. For proof of principle of our AUV path planner, we have used a simplified analytical solution of two-layer estuarine flow due to Officer, [12], where the river width and depth are constant and river flow is balanced by gravitational currents:

$$v_y = \frac{1}{48}\frac{g\lambda h^3}{\rho K_m}\left(1 - 9n^2 + 8n^3\right) + \frac{3}{2}v_0\left(1 - n^2\right) \qquad \textbf{(1)}$$

where $\quad n = \frac{z}{h}, \quad v_0 = \frac{r}{h}, \quad r = \int_h^0 v_y dz$

where v_y is the current speed in the along river (*y*) direction, *h* is the (uniform) depth of the water, *z* denotes vertical position relative to the surface, and *z=0* is mean sea level. Velocity is maximum at the surface, zero approximately 40% down in the water column, maximum inflow (up-river current) is approximately 80% of the way down, and the current is zero at the bottom (see figures for profile shape).

In a real estuary, the situation is more complicated. The full solution to the Navier-Stokes governing equation is not analytically solvable, so computer models are used. For the New York area, we use ECOMSED, a derivative of the Princeton Ocean Model (POM), [4], a full 3D ocean circulation model using a curvilinear grid in the horizontal, sigma coordinates in the vertical, and optimized for configuring the details of an estuarine environment. Our operational New York Harbor Observing and Prediction System (NYHOPS) model runs daily, producing a 24 hour hind-cast and 48 hour forecast, which can be viewed on our website at http://www.stevens.edu/maritimeforecast. A new high resolution model is currently in experimental development.

We have created a continuous interface to this model so that we can unplug the simplified estuarine flow, used here for proof of principle, and plug in the full nonlinear NYHOPS daily run output. Since the functional interface requests the environmental values as a function of *x, y, z,* and time, the AUV path planner is encapsulated from the details of the particular grid or model being used, and as new models improve, the path planner can therefore take advantage of the latest available information.

III. Path Planning

A. *Path parameterization and cost evaluation*

Many underwater robot vehicles are non-holonomic, however we assume simple holonomic kinematics for long range path planning since the path length is very large compared with the size of the robot and sharp turns are usually unnecessary. Also, for our application, we envision the use of AUV vehicles such as the NPS Phoenix, [3], which incorporate additional thrusters to enable kinematic decoupling of various different motion directions, enabling almost fully holonomic motion. At any point in time, the robot position and absolute velocity (relative to land) can be expressed as a state vector:

$$\overline{\mathbf{S}}_t = \left(x_t, y_t, z_t, \dot{x}_t, \dot{y}_t, \dot{z}_t\right)^T = \left(\overline{\mathbf{x}}, \dot{\overline{\mathbf{x}}}\right)^T \qquad \textbf{(2)}$$

The environment is characterized by 3D current velocities, $\overline{\mathbf{C}}$ derived from our computational estuarine model, which vary in space and time:

$$\overline{\mathbf{C}} = \begin{pmatrix} C_x \\ C_y \\ C_z \end{pmatrix} = f\left(x, y, z, t\right) = f\left(\overline{\mathbf{x}}, t\right) \qquad \textbf{(3)}$$

A candidate path is represented as a series of *n* straight line segments which connect *n+1* nodes. Since node times and inter-nodal speeds can be varied, the nodes occupy a 4D space consisting of 3D position and time. Each path segment is parameterized by the start and end node positions, $\left(\overline{\mathbf{N}}_i, \overline{\mathbf{N}}_{i+1}\right)$ and a duration for the segment Δt_i. Hence the entire path can be described with a single vector of *n+1* nodal positions and desired inter-nodal times:

$$\text{Path} = \left(\overline{\mathbf{N}}_0, \Delta t_0, \overline{\mathbf{N}}_1, \Delta t_1, ..., \overline{\mathbf{N}}_i, \Delta t_i, ..., \overline{\mathbf{N}}_n, \Delta t_n\right)^T$$
$$= \left(X_0, Y_0, Z_0, \Delta t_0, ..., X_i, Y_i, Z_i, \Delta t_i, ..., X_n, Y_n, Z_n, \Delta t_n\right)^T \qquad \textbf{(4)}$$

where the next segment duration at the end of the journey, Δt_n can be constrained to be zero. Note that the combination of node positions and segment durations determines unique values for the robot's net velocity components during the *i*th path segment:

$$\dot{\overline{\mathbf{x}}}_i = \frac{\left(\overline{\mathbf{N}}_{i+1} - \overline{\mathbf{N}}_i\right)}{\left|\overline{\mathbf{N}}_{i+1} - \overline{\mathbf{N}}_i\right|} \cdot \frac{\left|\overline{\mathbf{N}}_{i+1} - \overline{\mathbf{N}}_i\right|}{\Delta t_i} = \frac{1}{\Delta t_i}\left(\overline{\mathbf{N}}_{i+1} - \overline{\mathbf{N}}_i\right) \qquad \textbf{(5)}$$

Any candidate path can be evaluated according to a cost function. The entire path is evaluated progressively from beginning to end. For each successive pair of nodes, $\left(\overline{\mathbf{N}}_i, \overline{\mathbf{N}}_{i+1}\right)$, the straight line inter-nodal path segment is broken down iteratively into small, straight line sub-steps until the difference in average current speeds between any two consecutive steps falls below a specified value.

Since a constant net speed is demanded for the segment it is possible to determine the start and end times for each of these sub-steps so that current velocities can be found, even when currents are time varying. Each sub-step is assigned a current velocity, $\overline{\mathbf{C}}$ found by averaging the current velocity values at the beginning and end of the sub-step. Now, the components of the overall velocity which must be contributed by the robot's motors can be found, for any sub-step, as:

$$\overline{\mathbf{M}}_t = \begin{pmatrix} M_x \\ M_y \\ M_z \end{pmatrix} = \begin{pmatrix} \dot{x}_t \\ \dot{y}_t \\ \dot{z}_t \end{pmatrix} - \begin{pmatrix} C_x \\ C_y \\ C_z \end{pmatrix} \qquad \textbf{(6)}$$

giving the required motor speed (speed relative to the water due to the robot's thrusters) for the sub-step as:

$$V_{motor} = \sqrt{{M_x}^2 + {M_y}^2 + {M_z}^2} \qquad \textbf{(7)}$$

This speed can be used with an energy cost function to determine the cost of energy expenditure for the sub-step. For proof of principle we use a simple energy cost function comprising two components, a drag term and acceleration term:

$$\Delta\text{cost}_{energy} = \left(F_{drag} + F_{accel}\right) \times \overline{\mathbf{M}}_t \times \Delta t_{sub\_step}$$
$$= \left\{k_1 \left|\overline{\mathbf{M}}_t\right|^2 + k_2 \left|\dot{\overline{\mathbf{M}}}_t\right|\right\} \times \left|\overline{\mathbf{M}}_t\right| \times \Delta t_{sub\_step} \quad \textbf{(8)}$$

where F_{drag} and F_{accel} are the forces required to overcome drag and provide acceleration, $\overline{\mathbf{M}}_t$ and $\dot{\overline{\mathbf{M}}}_t$ are the velocity and acceleration of the AUV relative to the water and k_1 and k_2 are constants relating to the water density, effective AUV cross section area and the AUV mass. Note that any other energy function can be easily substituted here as required.

Each sub-step is checked to see if the required motor speed V_{motor} exceeds the greatest speed of the vehicle $V_{\max}$. If so, an additional cost term is added:

$$\Delta\text{cost}_{excess\_speed} = L + \left(V_{motor} - V_{\max}\right)L^2 \quad \textbf{(9)}$$

where *L* is a large number. The first part of this additional cost term prevents optimization algorithms from choosing any path which requires an impossible motor speed, by providing a large step jump in cost for such a path. The second part provides a steep gradient which rapidly forces the optimization algorithm to change the path to one with allowable motor speeds, if the path optimization process happens to be initialized with a path which already requires impossible motor speeds.

The total energy cost for the entire path can now be evaluated as a series of numerical integrations over all sub-steps of each inter-nodal path segment:

$$\text{Cost}_{energy} = \sum_{N_0}^{N_1} \Delta\text{cost}_{energy}.\Delta t_{0_{sub}} + \sum_{N_1}^{N_2} \Delta\text{cost}_{energy}.\Delta t_{1_{sub}}$$
$$+ \ldots + \sum_{N_i}^{N_{i+1}} \Delta\text{cost}_{energy}.\Delta t_{i_{sub}} + \ldots + \sum_{N_{n-1}}^{N_n} \Delta\text{cost}_{energy}.\Delta t_{n-1_{sub}} \quad \textbf{(10)}$$

where $\Delta t_{i_{sub}}$ denotes the duration of sub-steps on the path segment connecting nodes *i* and *i+1*.

This technique can easily be extended to incorporate obstacle avoidance by the addition of potential field-like cost terms. Obstacles and forbidden regions are modeled by boundaries which enclose the physical space of the obstacle plus an additional safety margin. To exclude the AUV from known obstacle regions, a large step cost is added to any path sub-step which lies inside the obstacle region plus an additional gradient term, based on distance of the AUV from the object boundary, to force the path out of the region. Because of the speed of the path planning algorithm, this method can be used for avoiding newly detected obstacles as well as known existing obstacles, provided that the time until collision is large compared to order 1s. For more imminent collisions, emergency path detouring algorithms can be adopted from the literature, e.g. [3]. Note that new obstacle information might not only be derived from the robot's onboard sensors but also might be communicated to the AUV from external sources, for example the vision based surveillance system that we are developing for boat tracking on the Hudson River or radar-based tracking of large ships.

Finally, additional cost terms are added to reflect the priorities of the user. If speed is important, a cost term proportional to the total journey time is added. It may also be desirable to relax the user-specified mission start and end positions and times so that the path planner can suggest close alternatives that may be advantageous. In this case a cost term should be added which increases with the difference between the user-specified start and end points and those suggested by the path planner. This cost term is necessary to prevent the path planner from collapsing the end point onto the start point.

Now all cost terms are summed to give a total cost for the candidate path:

$$\text{Cost}_{total} = \text{Cost}_{energy} + \text{Cost}_{obst} + \text{Cost}_{time} + \text{Cost}_{ends} \quad \textbf{(11)}$$

Additional weights can be added to each cost term to reflect the user's priorities. For example, with suitable cost weights, the path planner will find a trajectory which sacrifices energy efficiency for reduced journey time or vice versa.

B. *Path optimization*

We note that successful optimization depends largely on a careful choice of path parameterization and consideration for the fact that certain parameters are intrinsically coupled with respect to energy cost. It might seem convenient to express a path as a series of nodes $\left(x_i, y_i, z_i, t_i\right)$ in space and time. Thus *n* nodes give a 4*n* dimensional vector which could be solved with a standard gradient based non-linear optimization technique. Unfortunately, these parameters are not independent with respect to cost, which leads to two important optimization problems. Firstly, if the optimizer changes the time at one node, the time intervals for the path segments on both sides of that node are affected, often causing unexpected changes in speed and thus energy cost, such that a useful nodal time change can be erroneously rejected. Hence we instead use the parameters $\left(x_i, y_i, z_i, \Delta t_i\right)$ at each node so that an optimizer can modify the duration of one path segment independently of the others. Secondly, if the optimizer attempts to change the position of a node without changing its time, useful nodal position changes can be rejected because the position change has imposed unexpected speed (and hence energy) requirements on the path segments on both sides of the node. Therefore, whenever our optimization algorithm seeks to modify a nodal position, we simultaneously modify the inter-nodal duration time in order to ensure that the position change does not demand corresponding speed changes. For example, if the optimizer modifies the position of node *i* by $\Delta\overline{\mathbf{x}}$, the parameters are re-set with the following three steps:

$$\overline{\mathbf{N}}_{i_{new}} = \overline{\mathbf{N}}_{i_{old}} + \Delta\overline{\mathbf{x}} \quad (12)$$

$$\Delta t_{i-1_{new}} = \frac{\left|\overline{\mathbf{N}}_{i_{new}} - \overline{\mathbf{N}}_{i-1}\right|}{\left|\overline{\mathbf{N}}_{i_{old}} - \overline{\mathbf{N}}_{i-1}\right|} \cdot \Delta t_{i-1_{old}} \quad (13)$$

$$\Delta t_{i_{new}} = \frac{\left|\overline{\mathbf{N}}_{i+1} - \overline{\mathbf{N}}_{i_{new}}\right|}{\left|\overline{\mathbf{N}}_{i+1} - \overline{\mathbf{N}}_{i_{old}}\right|} \cdot \Delta t_{i_{old}} \quad (14)$$

Note that the time intervals, Δt_i, are still held as independent members of the path parameterization vector and so can still be modified individually by the optimizer, independently of the position parameters (x_i, y_i, z_i). Using a time interval rather than an explicit nodal time enables these times to be modified independently of the rest of the path. The path vector could now be optimized using a variety of well established nonlinear optimization techniques, [13], [14], providing they are suitably modified to include the steps above (equations 12, 13, 14). For proof of principle, we have used a simple, gradient based approach which iteratively modifies each path parameter in succession, starting with coarse step sizes and progressively refining with smaller step sizes. The initial coarse step sizes aid in avoiding local minima. In our experiments, this technique has consistently outperformed Powell's method (dog-leg non-linear least squares technique), [15], which appears to be particularly prone to local minima convergence in this application. Future work may examine alternative optimization techniques including genetic algorithms.

C. *Determining the number of nodes*

For a given number of nodes, we have so far described how to evaluate the cost of any path and how to optimize the nodal positions and times with respect to that cost. We now describe how to determine the best number of nodes to describe the path.

The path is initialized as a straight line with only three nodes, i.e. user specified start and end positions and times and a middle node which interpolates the positions and times of the start and end nodes. These three nodes are then optimized as described above. During the final cost evaluation of the three node path, the cost values of every path segment sub-step are recorded. For each node-node path segment, if the difference between the maximum and minimum sub-step costs exceeds a specified threshold, then an additional node is introduced at the segment midpoint and the four node path is now optimized again. This procedure continues until no new nodes are added.

If the path is two way, or the AUV is required to enter specific areas, additional mid-points or way-points may be specified by the user. If so, these way-points are initially connected by straight line path segments with additional interpolated nodes placed at the segment midpoints. The total path is then optimized using the same procedure described above.

Note that this procedure significantly enhances optimization speed, as compared with performing a single optimization on a pre-specified number of nodes. The majority of the optimization is performed on a small number of nodes, i.e. in a low dimensional optimization space. As new nodes are added, and the dimension of the optimization space increases, only a small number of additional iterations are usually needed at each stage. This results in an algorithm which is highly efficient whilst maintaining great flexibility by avoiding any need to discretize or partition the motion space.

IV. Results

We demonstrate this technique with example missions intended for the Hudson River, using our 3D AUV mission simulation environment, based on the Graphite visualization toolkit, [16]. The mission is planned with respect to a bi-directional, stratified current profile which closely resembles profiles encountered in the Hudson River and other estuaries. At different depths, stratified currents flow both up-river and down-river at speeds in excess of the maximum motor speed achievable by the AUV. The path planning system overcomes and exploits these currents by finding an optimal trajectory which enables an AUV to "ride" the currents in both directions. Obstacle avoidance capabilities are also demonstrated.

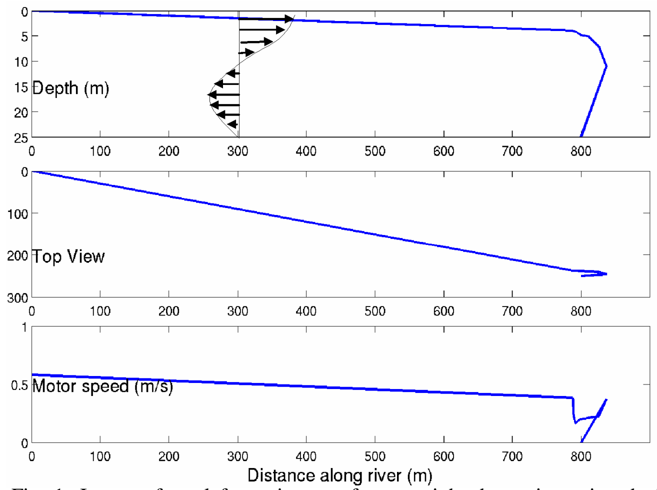

Fig. 1. Journey from left up-river, surface, to right down-river, river bed, minimizing energy expenditure. CPU time for path generation 617ms. Arrows denote current velocity profile with depth.

In figure 1 the AUV is tasked with minimizing energy expenditure. Hence it stays near the surface (maximum down-river current velocity) for most of the journey, only using its motors in the cross-river direction while drifting with the down-river surface current. To minimize the need for motor thrust, the AUV must descend very slowly. To enable this, it deliberately overshoots the target, picking up the reverse direction (up-river) current on a deeper strata as it descends, with which it drifts back onto the target destination without the need for energy expenditure.

In figure 2, the AUV is tasked with conserving energy

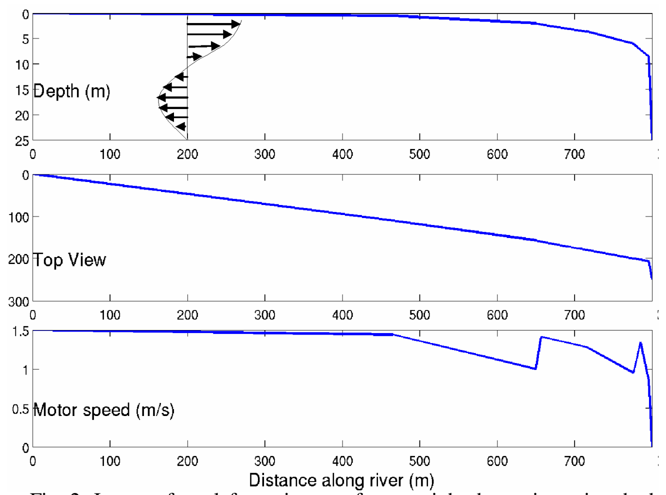

Fig. 2. Journey from left up-river, surface, to right down-river, river bed, modest emphasis on speed. CPU time for path generation 711ms. Arrows denote current velocity profile with depth.

while placing a modest emphasis on minimizing journey time. Now the AUV stays closer to the surface to exploit the maximum down-river current speed and uses greater downwards thrust to avoid the delay of overshooting.

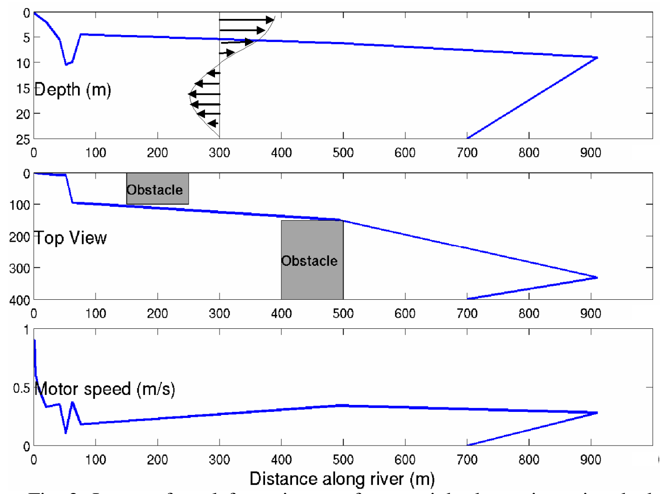

Fig. 3. Journey from left up-river, surface, to right down-river, river bed, avoid the obstacles, minimizing energy expenditure. CPU time for path generation 865ms. Arrows denote current velocity profile with depth.

In figure 3, the AUV must avoid two pier-like obstacles while minimizing energy expenditure. Note the initial dive to the exact depth of zero current speed, to minimize energy cost while traversing the first obstacle in the cross-river direction. Again the AUV overshoots the target, enabling a very slow descent and traverse, with minimum motor usage.

V. Conclusion

This paper has proposed a novel kind of AUV mission in which an AUV "rides" the currents in both directions up and down a river estuary, enabling continuous patrolling of otherwise energy intensive, complex, changeable, high speed current regions, for extended mission lifetimes.

A novel path planning strategy has been proposed which enables very general path shapes to be rapidly optimized with respect to complex, time varying current fields and obstacles. The technique enables an AUV to exploit prior knowledge of current velocities derived from a computational estuarine model. The technique is computationally efficient compared to other methods and runs sufficiently rapidly (<1s) to enable frequent re-planning of the remainder of the path in response to updated current information and newly detected obstacles. The technique offers more flexible and general treatment of arbitrary current velocities, path shapes and variable motor speeds than we have encountered elsewhere in the literature.

We are currently extending this work to make use of real current data in the Hudson River, derived from our high resolution computational estuarine model. We are also incorporating real bathymetry and other true map data for the Hudson and Manhattan Island area. We hope to test this technique with a sophisticated AUV platform in the Hudson River estuary in the near future.

References

[1] E. Desa, R. Madhan and P. Maurya. "Potential of autonomous underwater vehicles as new generation ocean data platforms", *Current Science,* vol. 90, no. 9, May, 2006.

[2] C. W. Warren, "A technique for autonomous underwater vehicle route planning", *IEEE J. Oceanic Engineering*, vol. 15, no. 3, July 1990.

[3] V. Kanakakis, K. P. Valavanis and N. C. Tsourveloudis, "Fuzzy-Logic Based Navigation of Underwater Vehicles", *Journal of Intelligent and Robotic Systems*, vol. 40, pp. 45-88, 2004.

[4] A.F. Blumberg and G.L. Mellor. "A description of a three-dimensional coastal ocean circulation model", Coastal and Estuarine Sciences, 4, 1987.

[5] A. Blumberg, M. Bruno. The New York Harbor Observing and Prediction System (NYHOPS). http://onr.dl.stevens-tech.edu/webnyhos3/, 2003.

[6] J. Sequeira, M. I. Ribeiro, "A Two Level Approach to AUV Path Planning", *Proc. IEEE Conf. OCEANS'94*, pp. II-87,II-91, Brest, France, 1994.

[7] J. Sequeira, M. I. Ribeiro, "AUV Path Planning under Current Constraints", in *Autonomous Underwater Vehicles*, Kluwer Publishers, 1996.

[8] B. Garau, A. Alvarez, G. Oliver, "Path Planning of Autonomous Underwater Vehicles in Current Fields with Complex Spatial Variability: an A* Approach", *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 194-198, Barcelona, Spain, April 2005.

[9] K. P. Carroll, S. R. McClaran, E. L. Nelson, D. M. Barnett, D.K. Friesen and G. N. Williams, "AUV path planning: An A* approach", *Proceedings of the Symposium on AUV Technology (AUV92)*, pp. 3-8, 1992.

[10] A. Alvarez, A, Caiti and R. Onken, "Evolutionary Path Planning for Autonomous Underwater Vehicles in a Variable Ocean", *IEEE J. Oceanic Engineering*, vol. 29, no. 2, April, 2004.

[11] J. C. Rubio, S. Kragelund, "The trans-Pacific crossing: long range adaptive path planning for UAVs through variable wind fields", *22nd Digital Avionics Systems Conference*, Indianapolis, 2003.

[12] C. B. Officer. Physical Oceanography of Estuaries (and Associated Coastal Waters). John Wiley & Sons ISBN 0-471-65278-4, pp. 116-121.

[13] W. Press, S. Teukolsky, W. Vetterling, B. Flannery. Numerical Recipes in C. 2nd Edition. Cambridge Univer-sity Press. ISBN 0 521 43108 5, 1992.

[14] J. Nocedal, S. Wright, Numerical Optimization. Springer, 1999.

[15] M. Powell. A Hybrid Method for Nonlinear Equations. In P. Rabinowitz, editor, Numerical Methods for Nonlinear Algebraic Equations, pages 87–144. Gordon and Breach Science, London, 1970.

[16] D. Kruger, The Graphite Visualization Toolkit, *Masters Thesis*, Stevens Institute of Technology, 2006.