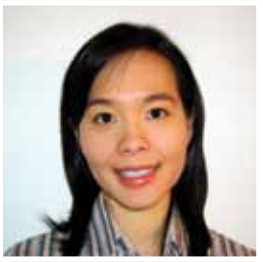
Look Ma, we're flying!



Beverley Chow

Christopher Michael Clark

Jan Paul Huissoon

Chow, Clark and Huissoon address the inability of an AUV to turn at an arbitrary rate when attempting to sample target points that are close together in the presence of a constant ocean current.

Who should read this paper?

Marine biologists, oceanographers, and other scientists who need to sample the ocean environment will be particularly interested in this paper. The ability to use multiple AUVs to sample or visit a large number of underwater locations within a short period of time so that the sample data are concurrent is important for properly capturing the state of the environment without undue temporal effects.

Why is it important?

Autonomous underwater vehicles are typically neutrally buoyant and completely rigid, propelled by a thruster at the tail and steered by two independent pairs of fins for pitch and yaw control. While they technically have six degrees of freedom (surge, sway, heave, pitch, roll and yaw) but only three independent actuators, AUVs have limited mobility.

This paper describes how algorithms for constructing an optimal sampling route for an AUV must consider the limited mobility of the vehicle, particularly when sample points are close together and the vehicle is subject to ocean currents. Determining the optimum route is not trivial and must incorporate the vehicle state (orientation, velocity) and environmental conditions (speed and direction of current) as the vehicle navigates the route. The research is innovative in that it considers how the combination of closely spaced task points, ocean currents, and the AUVs kinematic constraints affect the optimal solution to the task sequencing problem. Minimizing the time needed to sample a prescribed set of points saves energy, increases the accuracy of sampling, and reduces the risk of AUV loss.

The results of this work are completely disclosed and, as such, the methodology is immediately available for use in commercial applications.

About the authors

Beverley Chow received an M.A.Sc degree in Mechanical Engineering from the University of Waterloo in 2009. Her research interests include multi-robot systems, autonomous underwater vehicles, and artificial intelligence.

Christopher Clark is an Associate Professor in the Department of Computer Science at Cal Poly, San Luis Obispo, California, USA. His research interests include mobile robots, multi-robot systems, social networks in robotics, autonomous underwater vehicles, and intelligent vehicles.

Jan Huissoon is the Deputy Chair in the Mechanical and Mechatronics Engineering Department at the University of Waterloo, ON, Canada. His research interests include automatic calibration and intelligent control of robotic GMA welding, as well as autonomous robotic and intelligent vehicle systems.

ASSIGNING CLOSELY SPACED TARGETS TO MULTIPLE AUTONOMOUS UNDERWATER VEHICLES

Beverley Chow[1], Christopher Michael Clark[2] and Jan Paul Huissoon[1]
[1] *University of Waterloo, Ontario, Canada*
[2] *California Polytechnic State University, USA*

ABSTRACT

This paper addresses the problem of allocating closely spaced targets to multiple autonomous underwater vehicles in the presence of constant ocean currents. Targets are considered to be geographical locations that the AUVs must visit, ideally in an order that minimizes the path cost. The main difficulty of this problem is that the non-holonomic vehicles are constrained to move along forward paths with bounded curvatures. To accommodate such constraints, a new method for calculating path costs is proposed that considers vehicle kinematics, dynamics, and ocean currents. This path cost can be easily evaluated and queried from any general target sequence planner. Simulations show that the proposed method is able to create feasible paths with a lower cost when compared to solutions whose cost functions are calculated based solely on Euclidean distances. Field tests conducted on an Iver2 AUV validate the performance of the proposed algorithm in real world environments. Results show that the proposed algorithm generates paths that are feasible for an AUV to track closely, even in the presence of ocean currents.

KEYWORDS

Auction-based planner; AUV; Dubins model; Mission planning; Multi-Robot; Ocean currents; Path planning; Robot; Routing; Target sequence planner; Task allocation

NOMENCLATURE

B_i	=	bid of vehicle i	X_{prop}	=	surge force
C =	=	cost	$\mathbb{X}_{i,t}$	=	state of vehicle i and time t
D	=	targets	$\dot{\mathbb{X}}$	=	vector state derivative
K_{prop}	=	propeller torque	α	=	vehicle orientation
S_i	=	sequence of targets	δ_r	=	rudder fin angle
t	=	time	δ_s	=	stern fin angle
u_0	=	nominal vehicle speed	ψ_c	=	heading of the ocean current
u_c	=	ocean current speed			
$\mathbb{U}_{i,t}$	=	input vector	$\psi_{i,t}$	=	heading of the vehicle
V	=	vehicles	ω	=	bound on the yaw rate of the vehicle

INTRODUCTION

Autonomous Underwater Vehicles (AUVs) have been used successfully in the past to solve geological, biological, chemical, and physical oceanographic problems. This has resulted in a variety of scientific and commercial AUVs being designed, built, and deployed. With the increasing feasibility and decreasing expense of AUVs, interest in using them for ocean sampling, mapping, surveillance, and communication is growing and multi-AUV operations are beginning to be realized in the water. As with any multi-robot system, a challenge is to determine which robot should perform which task in order to cooperatively achieve the global goal in an optimal manner.

This paper investigates the task allocation problem where *n* vehicles are required to visit *m* task points. The motion of the AUV satisfies a non-holonomic constraint (i.e. the yaw rate of the vehicle is bounded) which makes the costs of going from one point to another non-Euclidean and asymmetric. Each task point is to be visited by one and only one vehicle and the problem has been simplified to limit the robots to operate in a horizontal plane. Given a set of task points and the yaw rate constraints on the vehicles, the problem is to assign each vehicle a sequence of task points to visit and to find a feasible path for each vehicle to follow so that the vehicle passes through the assigned task points. Each task point is a subgoal that is necessary for achieving the overall goal of the system that can be achieved independently of other subgoals. Task independence is assumed, where individual task points can be considered and assigned independently of each other without ordering constraints. The objective function to be minimized includes the total time to visit all of the task points.

The features that differentiate this research from similar problems previously studied are the kinematic constraints on the vehicle and the presence of a constant ocean current. This paper addresses the inability of an AUV to turn at any arbitrary yaw rate which becomes a problem when target points are close together. The Dubins model [Dubins, 1957] is a simple but efficient way to handle the kinematic characteristics of non-holonomic vehicles. It gives complete characterization of the optimal paths between two configurations for a vehicle with limited turning radius moving in a plane at constant speed.

In this paper, Dubins paths are modified to include ocean currents, resulting in paths defined by curves whose radius of curvature is not constant. To determine the time required to follow such paths, an approximate dynamic model of the AUV is queried. Specifically, a lower order model of the REMUS AUV model from Prestero [1994] is used so that the computational complexity is reduced.

The remainder of this paper is organized as follows: BACKGROUND gives an overview of the task allocation problem and describes various other techniques that have been used to solve related problems. PROBLEM STATEMENT begins with the formal problem definition. In PATH COST CALCULATION, the proposed path cost calculation method is introduced. ALGORITHM IMPLEMENTATIONS describes an implementation of the proposed method in Matlab and SIMULATION RESULTS discusses the results from

simulations to verify that the desired results are achieved. Following a satisfactory simulation, the proposed method was tested in the field at the Avila Pier in California as described in EXPERIMENTAL RESULTS. The paper concludes with a summary of results and future work in CONCLUSION AND FUTURE WORK.

BACKGROUND

The goal of the task allocation problem is to have robots visit all targets while minimizing the total travel time or distance travelled by the robots. When targets are known before the mission, it is possible to build a schedule of targets for each robot. Unfortunately, this problem is not straightforward because the cost for a robot to visit target *C* depends on whether that robot first visits target *A* or target *B*. This problem is an instance of the multiple travelling salesperson problem (MTSP), which has been studied extensively in combinatorial optimization. Even in the restricted case of one salesperson, the MTSP is strongly *NP*-hard [Korte and Vygen, 2006].

Several approaches have been applied to the general problem of allocating tasks between multiple robots in a team; refer to Gerkey and Matarić [2004] for a survey of these. Heuristic methods are typically used since optimizing the performance is often computationally intractable. Parker's ALLIANCE [Parker, 1998] is one of the earliest demonstrations of behaviour-based architectures for task allocation. Another frequently used method is based on market mechanisms, such as auctions, which have been demonstrated in Dias et al. [2006] to be fast and robust on real robots. Specific work for AUVs, often called mission planning, includes the work by Sariel et al. [2008] and vehicles with bounded curvature are considered by Jeyaraman et al. [2004]. Similar to the mission planning problem is the routing problem as investigated by Davis et al. [2008] for underwater gliders, and the path planning problem as described in Kruger et al. [2007] and Alvarez et al. [2004] for AUVs operating in an environment with complex currents. However, the vehicle dynamics are not accounted for in their strategies which this paper aims to address.

PROBLEM STATEMENT

This paper considers the allocation of *m* targets to *n* vehicles. Given a set of vehicles $\{V_1, V_2, \ldots, V_n\}$ and targets $D=\{d_1, d_2, \ldots, d_m\}$, the problem is to assign a sequence of targets S_i to each vehicle to visit and a path through the sequence S_i. The objective is to:

Minimize

$$C_{\text{total}} = \max_i C(S_i) \tag{1}$$

subject to

$$D = \bigsqcup_i S_i \qquad \text{(disjoint union)} \tag{2}$$

$$\left.\begin{aligned} \frac{dx_{i,t}}{dt} &= u_0 \cos(\psi_{i,t}) + u_c \cos(\psi_c) \\ \frac{dy_{i,t}}{dt} &= u_0 \sin(\psi_{i,t}) + u_c \sin(\psi_c) \\ \frac{d\psi_{i,t}}{dt} &= r_{i,t}, \qquad r \in [-\omega, +\omega] \end{aligned}\right\} \tag{3}$$

where u_0 denotes the nominal vehicle speed, $\psi_{i,t}$ the heading of the vehicle, u_C the ocean current speed, ψ_c the heading of the ocean current, and ω represents the bound on the yaw rate of the vehicle. In Equation (1), $C(S_i)$ is the time required for V_i to complete its tour S_i.

Note that Equation (2) dictates all tasks to be visited and restricts each task to be assigned to only one vehicle and Equation (3) considers the non-holonomic constraints of the vehicle.

Regardless of the planner used for solving this problem, an efficient and accurate method of calculating path costs which considers kinematics, dynamics, and ocean currents is required. Specifically, determining the cost to travel between any two task points is not trivial and must incorporate the vehicle state as it passes through the task points (e.g. vehicle orientation, velocities, etc.). Presented below is a method for calculating path costs that uses a modified Dubins path to incorporate vehicle kinematics, and a lower order model to approximate and incorporate vehicle dynamics.

PATH COST CALCULATION

To calculate the path and time of travel between task points, one must consider the dynamics and kinematics of the vehicle. The path cost can be calculated in two steps: 1) calculating the Dubins path that considers ocean currents and 2) calculating the time to travel along the paths using a lower order dynamic model. Described below is the dynamic model used, followed by the two main steps used for calculating path cost.

Vehicle Model

Calculating the feasible states of the vehicle in the presence of ocean current requires knowledge of the vehicle dynamics. This paper uses the REMUS AUV model created by Prestero [1994], which readers are referred to for the full derivation. The REMUS AUV has a torpedo shape with an ellipsoidal nose, a cylindrical constant radius mid-section, and a cubic spline tail section as illustrated in Figure 1.

Figure 1: REMUS AUV.

The vehicle has six degrees of freedom (DOF), namely *surge*, *sway*, *heave*, *pitch*, *roll*, and *yaw*. The AUV is assumed to be neutrally buoyant, completely rigid, and interacting with an ideal fluid. The vehicle is propelled by a thruster at its tail and steered by two independent pairs of fins for pitch and yaw control. With 6-DOF and only three independent actuators, the system is considered to be an underactuated system.

The 6-DOF non-linear model described in Prestero [1994] can be used to simulate how different control and hydrodynamic forces affect the body-fixed velocities and the overall change in position and orientation of the vehicle. The simulation requires the ability to represent the vehicle motion with respect to both body-fixed and inertial coordinates. Therefore, the twelve states of a vehicle V_i consisting of body-fixed velocities and inertial coordinates at time t are given by:

$$\mathbb{X}_{i,t} = [u_i \;\; v_i \;\; w_i \;\; p_i \;\; q_i \;\; r_i \;\; x_i \;\; y_i \;\; z_i \;\; \phi_i \;\; \theta_i \;\; \psi_i]^T . \quad (4)$$

Given the complex and highly non-linear nature of the problem, numerical integration is used to solve for the vehicle position and orientation in time. At each time step, the

vehicle state is updated by the general equation:

$$\mathbb{X}_{i,t+1} = f(\mathbb{X}_{i,t}, \mathbb{U}_{i,t}, u_c, \psi_c) \quad (5)$$

where $\mathbb{X}_{i,t}$ is the vehicle state vector, $\mathbb{U}_{i,t} = [\delta_s \; \delta_r \; X_{prop} \; K_{prop}]^T$ is the input vector, u_c and ψ_c are the magnitude and direction of the ocean current respectively. For the input vector, δ_s is the stern fin angle, δ_r is the rudder fin angle, X_{prop} is the surge force, and K_{prop} is the propeller torque.

The function *f* in Equation (5) uses the Euler method of numerical integration to yield the new vehicle state at each time step as:

$$\mathbb{X}'_{i,t+1} = \mathbb{X}_{i,t} + \left(\dot{\mathbb{X}}_{i,t} \cdot \Delta t\right) \quad (6)$$

where the state vector derivative $\dot{\mathbb{X}}_{i,t}$ is updated using the model $\dot{\mathbb{X}}_{i,t} = f(\mathbb{X}_{i,t}, \mathbb{U}_{i,t})$ from Prestero [1994]. With the presence of a fixed current, the position of the vehicle relative to the inertial-fixed frame is updated as follows:

$$\begin{aligned} x_i &= x'_i + (u_c \cos(\psi_c) \cdot \Delta t) \\ y_i &= y'_i + (u_c \sin(\psi_c) \cdot \Delta t) \end{aligned} \quad (7)$$

Where x'_i and y'_i denote the position of the i^{th} vehicle after the integration step given in Equation (6). By combining Equation (6) and Equation (7), the function *f* in Equation (5) is realized and can be used to update the general state of each vehicle. This full 6-DOF non-linear model is used to evaluate the final tour times of the sequences generated by the proposed method.

Dubins Path Calculation

In order to calculate the time required to travel between two points, the Dubins shortest path problem must first be solved. Dubins' original work [Dubins, 1957] derived conditions that characterize the optimal path between two points when both the initial and terminal orientations were specified and his work has been widely studied in path planning [Shkel and Lumelsky, 2001]. Dubins' result shows that, given any two points, the shortest path that considers the constraints expressed in Equation (3) consists of exactly three path segments consisting of a combination of a straight line segment and maximum curvature arcs.

Graphically, the algorithm starts by drawing two maximum curvature circles that are tangential to the initial state vector and two maximum curvature circles that are tangential to the terminal state vector. Dubins' result indicates that the optimal trajectory selects an arc on one of the two initial circles, and connects tangentially to an arc on one of the two terminal circles. If the separation between the initial and end points is sufficient, this can only be accomplished by a line segment. There are at most four such line segments, and computation of the travel distances is straightforward, as shown in Figure 2 for two waypoints with initial and terminal orientations, denoted α_k and α_{k+1} respectively. Note that α is measured counter-clockwise with respect to the positive x-axis.

Finding the shortest path between two points requires repetitively solving the shortest time algorithm for various entry and exit AUV orientations (i.e. α_k, α_{k+1}). The added challenge here is that there may be a family of paths that connects s_k to s_{k+1} with only one being the shortest. The multiplicity of paths connecting the two points complicates the search for

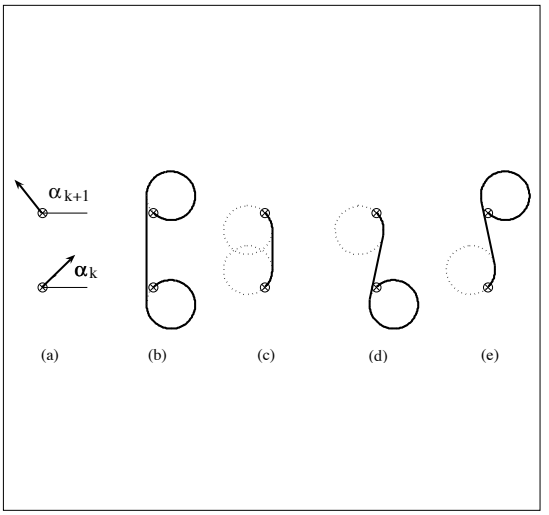


Figure 2: (a) Two waypoints (x_k, y_k) and (x_{k+1}, y_{k+1}) with $\alpha_k = \pi/4$ and $\alpha_{k+1} = 3\pi/4$. (b)-(e) Four ways of connecting two waypoints using Dubins curves.

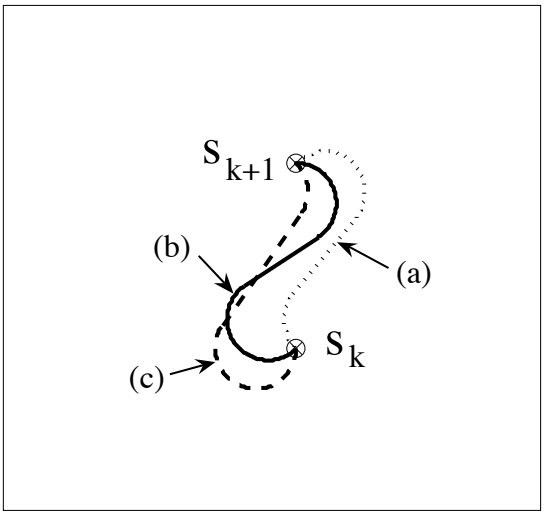


Figure 3: Multiple paths for different initial and final orientations. (a) $\alpha_k = 3\pi/4$, $\alpha_{k+1} = -3\pi/4$ (b) $\alpha_k = -3\pi/4$, $\alpha_{k+1} = \pi$, (c) $\alpha_k = -\pi/2$, $\alpha_{k+1} = 3\pi/4$.

initial and final headings so an exhaustive coarse-resolution search is implemented. In this paper, a discrete approximation is made so that α_k and α_{k+1} are constrained to $\Lambda = \{\lambda\pi/4 \mid \lambda = 0, \ldots, 7\}$. With eight possibilities for α_k and α_{k+1} and four ways of connecting them, a total of 8 x 8 x 4 = 256 paths is possible for every pair of waypoints, with one of them being the shortest path. Increasing the resolution of the discretization would result in better approximations but at a cost of increasing the computational complexity. Figure 3 shows three paths connecting s_k to s_{k+1}. Note that there are additional paths connecting the same points which are not shown and that different values for α_k and α_{k+1} yield different costs.

In the presence of ocean currents, the shortest path between two points given α_k and α_{k+1} consists of arcs that are no longer circular but elliptic. These ellipses will have different curvatures depending on the magnitude and direction of the current (Figure 4). The shape of the ellipse depends on the vehicle's orientation at the start of the turn and is calculated using the difference between the vehicle's heading ψ and the direction of the current ψ_c.

To determine the shape of the ellipse, Equation (5) was used to determine the state of the vehicle at each time step with $\mathbb{X}_0 = [1.15\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^T$, and $\mathbb{U}_0 = [0\ 1.75\ 5.16\ 0]^T$, where 1.15 m/s is the nominal speed of the AUV, 1.75 rad is the rudder fin angle, and 5.16 N is the propeller surge force. Note that finding the maximum curvature for the ellipse requires running the simulation using a maximum rudder fin angle of 1.75 radians. AUV powering was not taken into account and the vehicle was running at the nominal speed for the duration of the simulation. Data was obtained for discrete cases of $u_c = \{0.1, 0.2, 0.3, 0.4, 0.5\}$, and $\psi_c = \{\kappa\pi/8$ for $\kappa = 1, ..., 16\}$ and the vehicle's position was recorded at $\Delta\psi = \{\kappa\pi/8$ for $\kappa = 1, \ldots, 16\}$, where $\Delta\psi$ is the fraction of a complete circumnavigation of the ellipse the vehicle travels (Figure 5). The minimum turning radius of the REMUS AUV from running the simulation was calculated to be 3.3 metres.

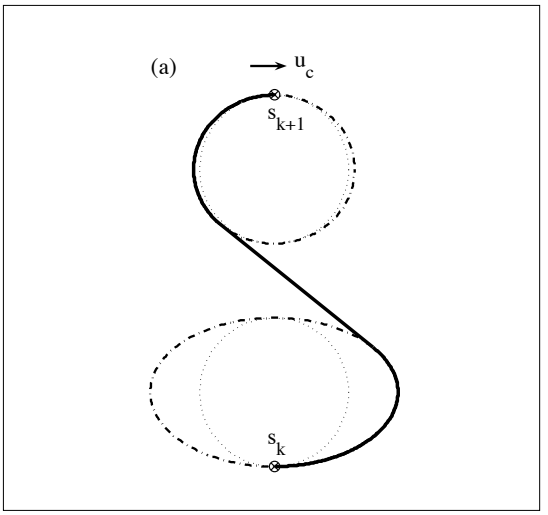


Figure 4a: Dubins Curves between two waypoints with ocean currents u_c = 0.25 m/s (a) ψ_C = 0.

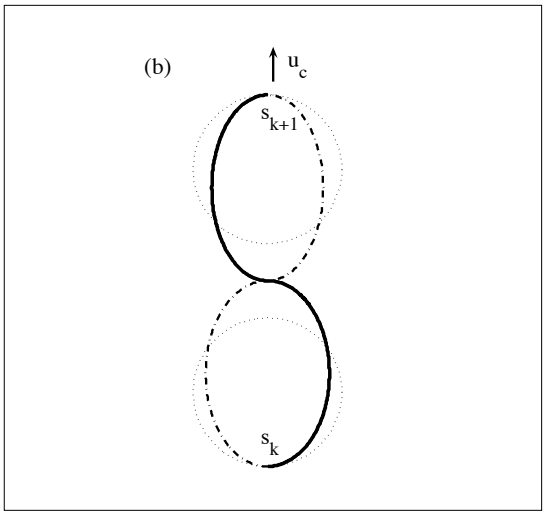


Figure 4b: $\psi_C = \pi/2$.

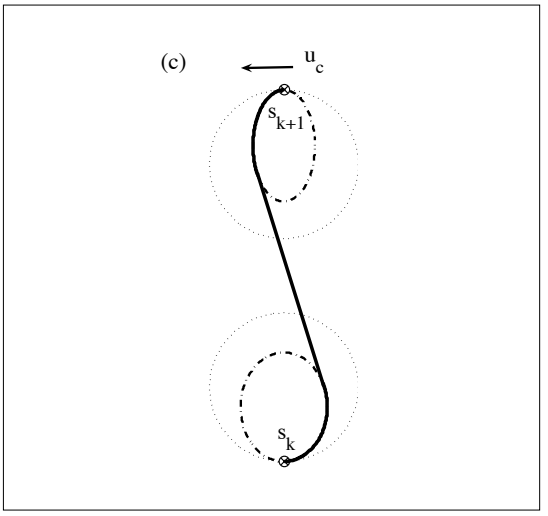


Figure 4c: $\psi_C = \pi$.

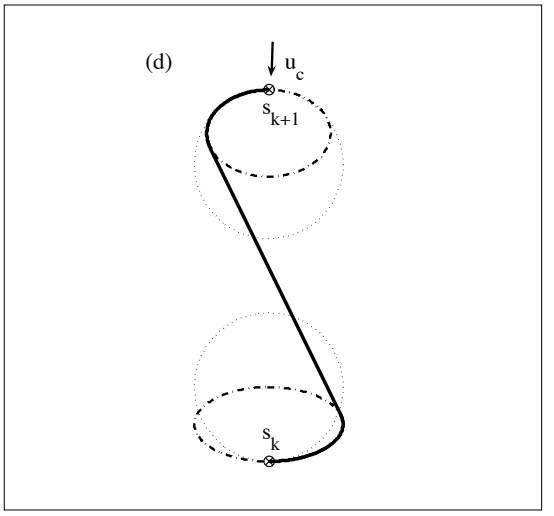


Figure 4d: $\psi_C = -\pi/2$.

Using this data, a lower order model $\tilde{f}$ was created to determine the position of the vehicle given the change in the vehicle's heading, and the magnitude and velocity of the current.

$$(\Delta x, \Delta y) = \tilde{f}(\Delta\psi, u_c, \psi_c) \quad (8)$$

The next step is to find the fraction of a complete circumnavigation of the ellipse to travel before and after the straight line segment. Finding a line segment tangent to two curves is solved by using an iterative process. As a starting point, the slope of the tangent line to two circular arcs of minimum radius (when $u_c = 0$) is calculated (Figure 6a). Using that value, P_a and P_b are found on the respective ellipses whose slope is equal to the slope of the tangent (Figure 6b). The positions of P_a and P_b are determined by using $\tilde{f}$ from Equation (8). The slope of the line segment

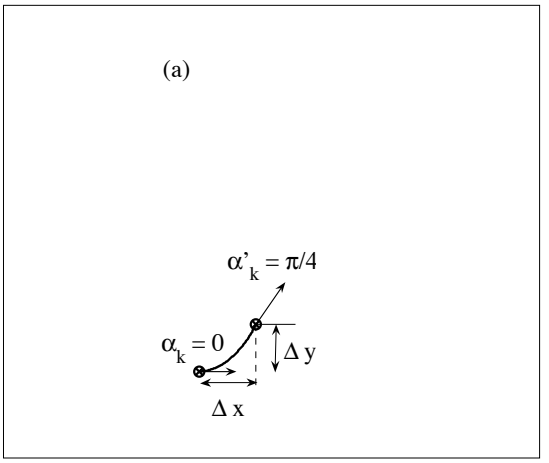


Figure 5a: Vehicle position for various values of $\Delta\psi$ as the vehicle moves along the maximum curvature ellipse. (a) $\Delta\psi = \pi/4$

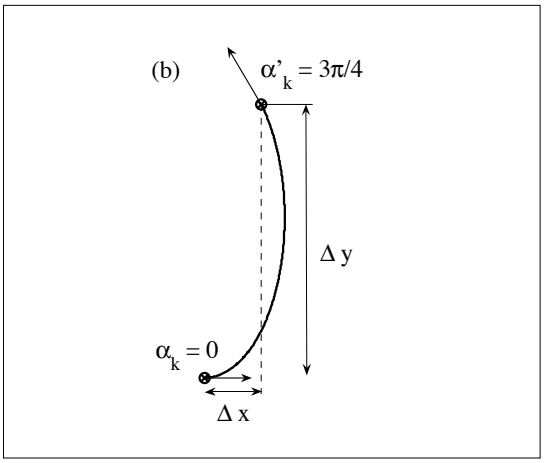


Figure 5b: $\Delta\psi = 3\pi/4$.

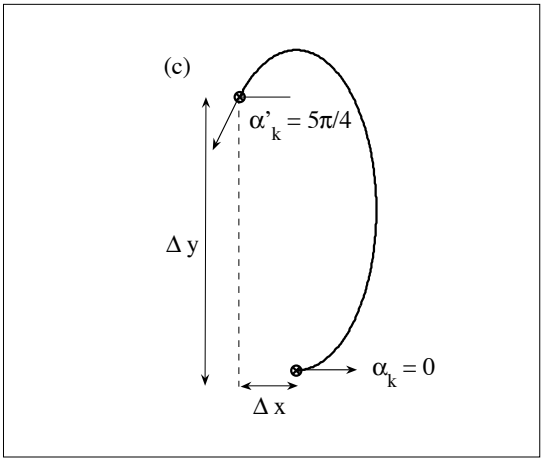


Figure 5c: $\Delta\psi = 5\pi/4$.

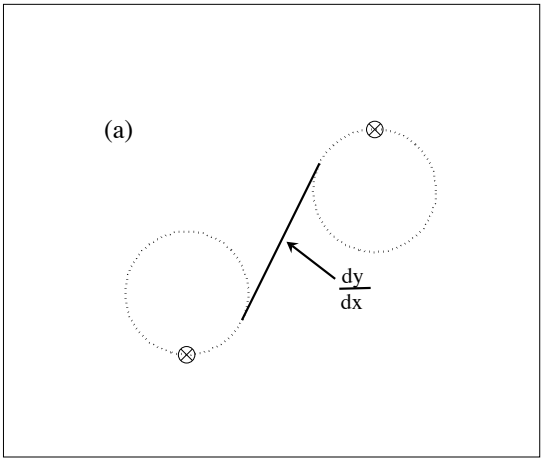


Figure 6a: Illustration of the iterative process used to find a tangent to two curves.

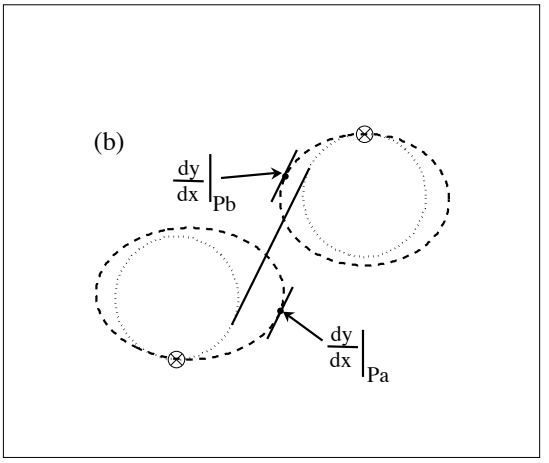


Figure 6b

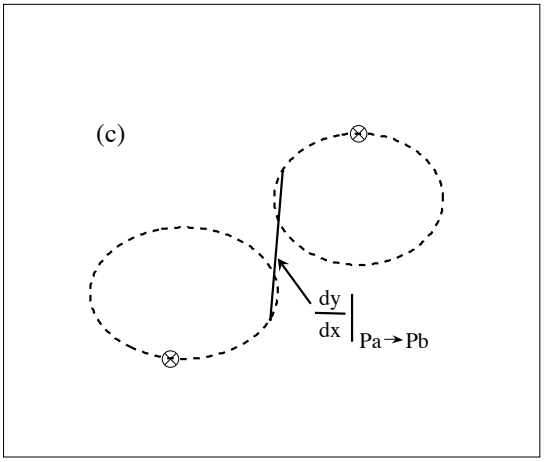


Figure 6c

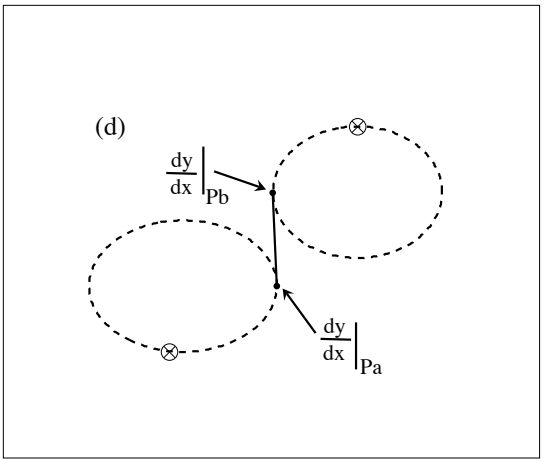


Figure 6d

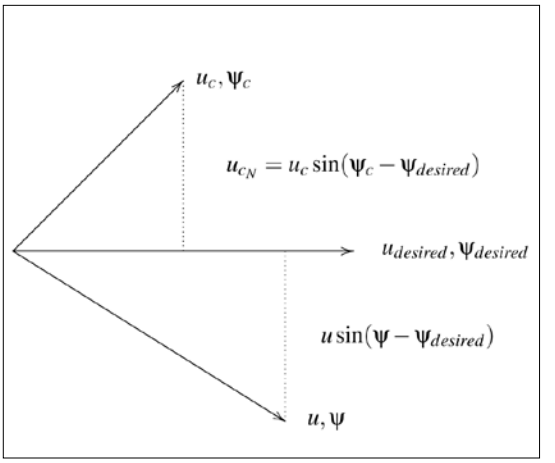


Figure 7: Illustration of the relative velocities.

from P_a and P_b is calculated and becomes the new slope for the next iteration (Figure 6c). The process continues until convergence (Figure 6d).

Note that Figure 6d depicts the desired path for the AUV to follow in the presence of ocean currents. In order for the vehicle to stay on the desired path, the vehicle heading ψ must be calculated to compensate for the effect of ocean currents.

Time Calculation

The cost of traversing the sequence S is calculated as:

$$C(S) = \min_{(\alpha_1,\ldots,\alpha_l)\in\Lambda^l} \sum_{k=1}^{l} \Delta t_{(s_k,\alpha_k)\to(s_{k+1},\alpha_{k+1})} \qquad (9)$$

To calculate the time $\Delta t(s_k,\alpha_k)\to(s_{k+1},\alpha_{k+1})$ in Equation (9), a lower order model $\hat{f}$ is created based on the full model f from (5) as:

$$\Delta t_{(s_k,\alpha_k)\to(s_{k+1},\alpha_{k+1})} = \hat{f}\,[s_k, s_{k+1}, \alpha_k, \alpha_{k+1}, u_0, u_c, \psi_c] \qquad (10)$$

For arcs, the lower order model is a piecewise linear function built from sampling the full model. Using the full model, the vehicle orientation can be determined at a certain time t. In order to find the time required to obtain a specific heading, linear interpolation is used on the data obtained from the full model at various fractions of a complete circumnavigation of the ellipse ($\kappa\pi/8$ for $\kappa = 1, \ldots, 16$).

For straight line segments, consider a vehicle moving at speed u and heading ψ through the water with current velocity u_c and direction ψ_c. The vehicle's velocity along the desired path has magnitude $u_{desired}$ and direction $\psi_{desired}$. These velocities are illustrated in Figure 7. Let $u_{c\psi desired} = u_c\cos(\psi_c - \psi_{desired})$ be the current component assisting motion along the desired direction and $u_{c_N} = u_c\sin(\psi_c - \psi_{desired})$ be the current component $\pi/2$ radians to the left of the desired direction. Staying on the desired path requires the perpendicular component of the vehicle velocity $u\sin(\psi - \psi_{desired})$ to cancel the perpendicular component of the current u_{c_N}. The heading ψ and speed $u_{desired}$ along the desired vehicle motion

direction are:

$$\psi = -\arcsin(u_{c_N}/u) + \psi_{desired}$$
$$u_{desired} = u_{c_{\psi_{desired}}} + u\sqrt{1-(u_{c_N}/u)^2}. \quad (11)$$

As long as $|u_{c_N}| < u$, the vehicle can stay on the desired path, but the velocity decreases as $|u_{c_N}| \rightarrow u$. Keeping the vehicle on the desired path is critical because making measurements at the right location requires the vehicle to stay on track in the presence of ocean currents.

The time required to travel from P_a to P_b can then be calculated as follows:

$$\Delta t_{straight} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}/u_{desired} \quad (12)$$

Combining these results in

$$\Delta t_{(s_k,\alpha_k)\rightarrow(s_{k+1},\alpha_{k+1})} = \Delta t_{arc(s_k \rightarrow P_a)} + \Delta t_{straight(P_a \rightarrow P_b)} + \Delta t_{arc(P_b \rightarrow s_{k+1})}. \quad (13)$$

ALGORITHM IMPLEMENTATIONS

This paper addresses the task allocation problem which is not possible to solve in polynomial time. The problem combines the exponential complexity of integer assignment decisions with non-linear, non-convex differential equation constraints, making it a Mixed Integer Non-linear Program with exponential growth in computational time. The focus of this paper is to illustrate the effectiveness of calculating $C(S)$ in reducing path costs, as described in detail in PATH COST CALCULATION, which can be used with any planner. To demonstrate the performance of the proposed path cost calculation method, this paper implements an exhaustive search which yields the optimal solution but can only be used for a small number of tasks, and a market-based planner that returns only an approximately optimal allocation but is tractable for a large number of tasks.

Breadth First Search Planner

One method of task allocation is to do an exhaustive search through all possible combinations. One implementation of an exhaustive search is to use a Breadth First Search (BFS) which finds the shortest-path between two nodes by exploring all vertices and edges of the graph systematically. The algorithm starts at one node and explores all the neighbouring nodes that have not been visited. From there, it explores their unexplored neighbouring nodes until all of the nodes have been visited.

The BFS planner is implemented using the proposed path cost calculation as described in PATH COST CALCULATION, which is referred to as the "BFS-Dubins Planner." As a baseline for comparison, the BFS planner is also implemented using Euclidean distances for path cost calculation. The tours are then post processed such that the order of task points is preserved but the paths through the task points incorporated Dubins' paths. This is referred to as the "BFS-Euclidean Planner."

Auction-Based Planner

Another method of task allocation is using a market-based approach as follows. Consider one particular trial with $n = 3$ and $m = 20$. Using *k-means* [Hartigan and Wong, 1979], the 20 task points are partitioned into three clusters as shown in Figure 8. The *k-means* algorithm partitions the m points into n clusters

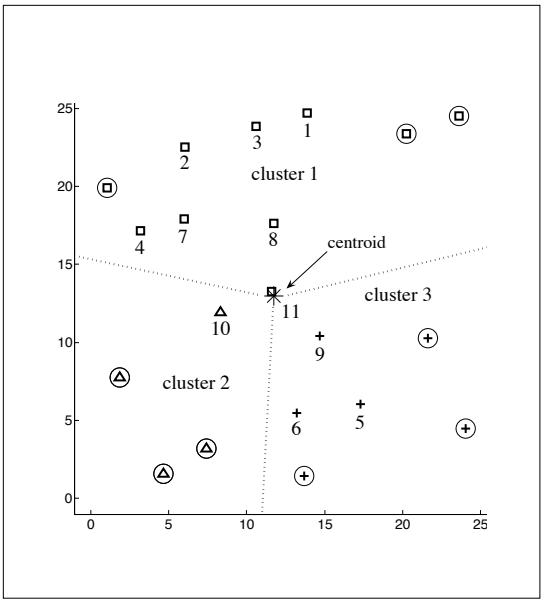


Figure 8: Results from clustering using *k*-means. Tasks with a circle around it are assigned to the vehicle responsible for that cluster.

by minimizing the total intra-cluster variance, or the squared error function. For this implementation, *k-means* is minimized with respect to the squared Euclidean distance with the initial cluster centroid positions selected uniformly at random from the range of x. Depending on the clusters created, the planner will produce local minimum solutions and, therefore, the *k-means* algorithm is repeated three times, each with a new set of initial centroids. If a cluster loses all of its member observations during the iterative process, a new cluster consisting of the one observation furthest from its centroid is created. It should be noted that Matlab uses a two-phase iterative algorithm for *k-means* clustering that only converges to a local minimum. The problem of finding the global minimum can only be solved in general by an exhaustive choice of starting points. Therefore, Matlab produces different clusters using the same dataset depending on the starting points chosen and Figure 8 is one of many solutions.

After partitioning all task points into clusters, the centroid of all task points is calculated. For all $i = 1, 2, ..., n$ clusters, the three task points in each cluster i that are farthest from the centroid are assigned to the i^{th} vehicle (tasks with circle around them in Figure 8). The number of initial task assignments is chosen to be three because for three tasks forming a loop, the ordering of the tasks does not matter and always produces the same loop.

Once each vehicle has three task points assigned to it, the remaining $m - 3n$ tasks are auctioned off using a first-price one-round mechanism similar to the work by Lagoudakis et al. [2004]. The unassigned tasks are first ordered according to their distance from the centroid, with higher priority given to tasks that are farther from the centroid (auctioning order is indicated in Figure 8). Following this order, each task is auctioned off.

Each vehicle i can bid on the task j, where the bid B_i is equal to the cost of travelling a path that consists of all previously won tasks and the current task being auctioned. Each vehicle considers the insertion of the new task at every point in the current sequence $S_i = (s_1, s_2, s_3, ..., s_l)$ where l is the number of previously won tasks by vehicle i. When a vehicle bids for task d_j, the vehicle must try every value of $\Lambda = \{\lambda\pi/4 \mid \lambda = 0, \ldots, 7\}$ for the orientation at task d_j. With the insertion of task d_j in between s_k and s_{k+1}, the optimal orientations α_k and α_{k+1} also have to be recalculated with all values of Λ. The orientations at all other task points in the sequence S is kept from the previous round of bidding since the addition of task d_j has minimal effect on the rest of the tour. Because the sequence S and the values for the

optimal orientation at each task point with the exception of α_k and α_{k+1} are kept from the previous round of bidding, Equation (10) only needs to be calculated between tasks (s_{k-1}, s_k), (s_k, d_j), (d_j, s_{k+1}), and (s_{k+1}, s_{k+2}). This simplifies the computational complexity and significantly decreases the processing time.

Each vehicle submits a bid as the lowest cost (i.e. time) to complete the new tour as:

$$B_i(d_j, S_i) = \min_{0 \le k \le l} C(s_1, ..., s_k, d_j, s_{k+1}, ..., s_l) \quad (14)$$

The i^{th} vehicle with the lowest B_i wins target d_j and updates its sequence of targets with $S_i' = (s_1, ..., s_k, d_j, s_{k+1}, ..., s_l)$. The auctioning process continues with the next round of bidding until all tasks are allocated. A benefit of an auctioning task allocation system is that it has the potential to be decentralized, online, handle asynchronous bidding, and allows for auctioning by any agent. However, the implementation described here is offline and centralized. Several issues (e.g. two AUVs requesting simultaneous auctions) must be addressed before such potential capabilities can be realized.

The auction-based planner is implemented using the proposed path cost calculation and is referred to as the "Auction-based Proposed Planner." As a baseline for comparison, the auction-based planner is also implemented using Euclidean distances for path cost calculation. The tours are then post processed using the "alternating algorithm" and is referred to as the "Alternating Algorithm Planner."

The "alternating algorithm" as described by Savla et al. [2008] solves the MTSP by creating Dubins TSP tours (i.e. sequences from

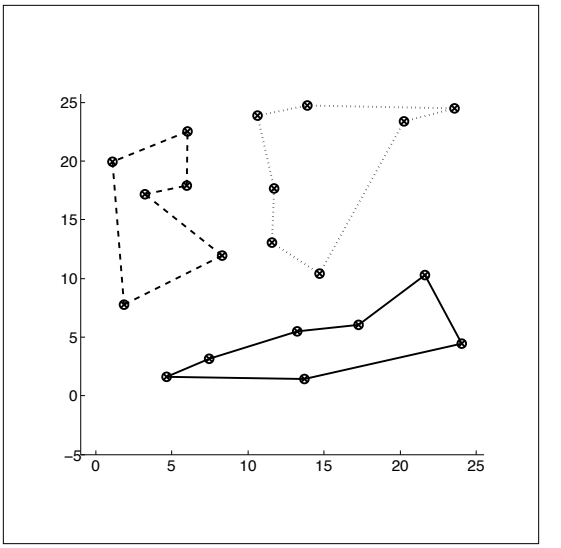


Figure 9: Auction-based Euclidean MTSP solution for allocating 20 tasks to three robots.

applying Dubins' model) with an asymptotic bound on the worst-case length. It is used to determine Dubins' tours for datasets that are too large to run an exhaustive search on. It works as follows: given a set of n points, the optimal Euclidean MTSP tours (i.e. that do not consider path curvature) are computed using auctions (Figure 9). Then, it is necessary to obtain a feasible path through these ordered points using the method in Savla et al. [2008] which includes the curvature constraints of the vehicle.

SIMULATION RESULTS

To demonstrate the performance of the proposed method, computer simulations were carried out with a model of the REMUS AUV using Matlab on an Intel 1.66 GHz Core 2 Duo processor T5500 with 2GB RAM and running Windows XP SP3.

Comparison using BFS Planners

The first set of simulations compared the BFS-Dubins Planner with the BFS-Euclidean Planner using a dataset with 16 task points that were arranged into a grid with equal spacing

between the task points. The number of task points was limited to 16 because the complexity of the search space for the problem grows rapidly with the number of task points. The results of the task allocation for three robots using the BFS-Euclidean Planner are shown in Figure 10. Using the optimal task ordering from Figure 10, Dubins TSP tours were created to find feasible paths through the task points. Simulations were conducted on this dataset with various distance between task points. Task points were spaced from 1 metre to 10 metres apart, in 1 metre increments.

The total mission time and average mission time for the two planners are plotted in Figure 11. From the graphs, it can be seen that the BFS-Dubins Planner creates shorter paths when the task points are closer than 5 metres apart. This is because extra loops are required when the vehicle changes orientation as shown in Figure 12. Recall that the turning radius of the REMUS AUV is 3.3 metres.

For a sufficiently dense set of points, it becomes clear that the ordering of the Euclidean tours is not optimal in the case of the Dubins MTSP. This is due to the fact that there is little relationship between the Euclidean and Dubins metrics, especially when the Euclidean distances are small with respect to the turning radius. An algorithm for the Euclidean problem will tend to schedule very close points in a successive order, which can imply long manoeuvres for the AUV. This was clearly demonstrated by the numerous loops that become problematic with dense sets of points. The path calculation method proposed in this paper does not rely on the Euclidean solution and, therefore, can create paths that are feasible for curvature bound vehicles.

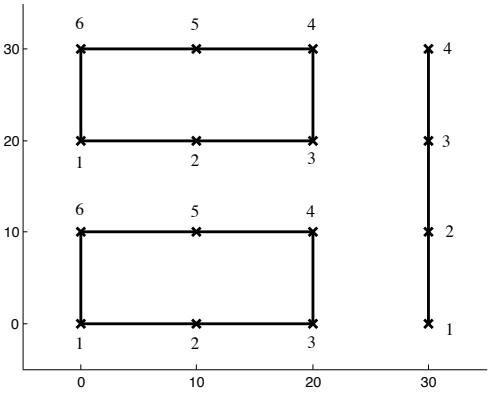


Figure 10: Optimal task allocation for 16 task points assigned to three robots using BFS-Euclidean.

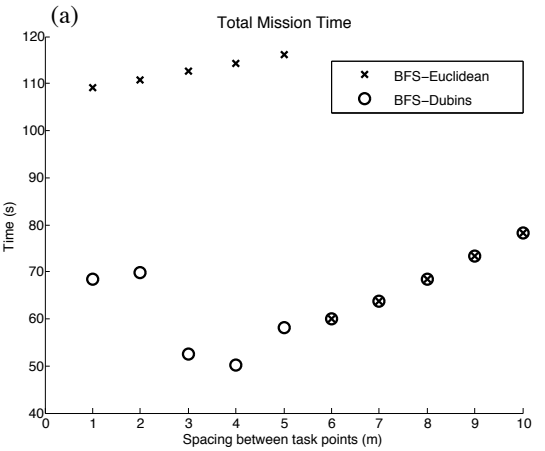


Figure 11a: Graphs comparing the performance of the BFS-Dubins Planner to the BFS-Euclidean Planner: (a) total mission time vs. spacing between task points.

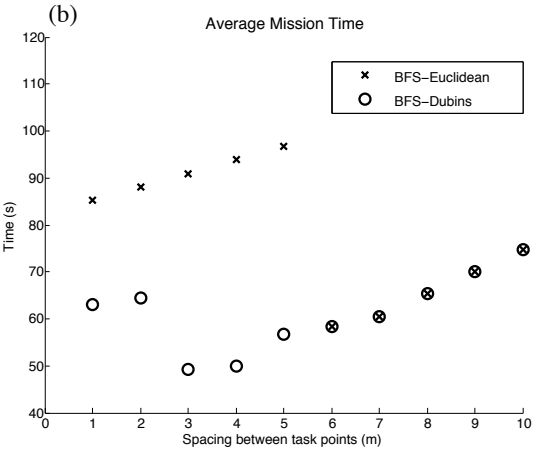


Figure 11b: Average mission time of three robots vs. spacing between task points.

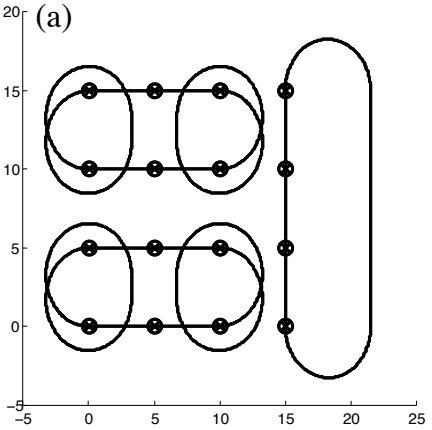


Figure 12: Dubins paths through task points: (a) order determined using the BFS-Euclidean Planner.

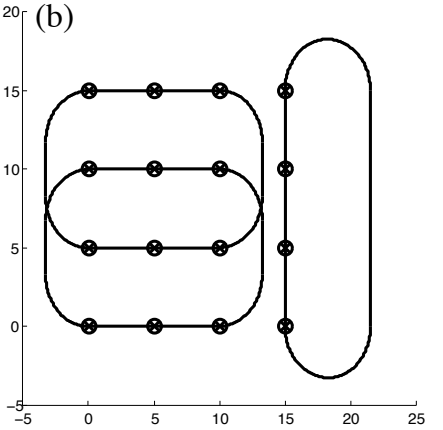


Figure 12b: Order determined using the BFS-Dubins Planner.

Comparison using Auction-Based Planners

The next set of simulations was conducted on 50 datasets, each set containing between six and 20 task points. The task points were distributed randomly with a uniform probability distribution inside a square with side lengths of 25 metres. The task points were generated close together to highlight the necessity for considering the curvature constraints.

The results from running the simulation on 50 different datasets are summarized in Table 1 using the following criteria:

$$T_{\max} = \max C_{sim}(S_i) \quad \text{and} \quad T_{\text{avg}} = \frac{\sum_{i=1}^{n} C_{sim}(S_i)}{n}.$$

C_{sim} is the cost calculated by running the planned tours S_i through the *full* dynamic model in Equation (5). On average, the Auction-Based Proposed Planner reduced T_{max} by 43% over the Auction-Based Alternating Algorithm Planner in the absence of currents and 45% in the presence of currents.

Consider one particular trial illustrated in Figure 13 and Figure 14 whose results are presented in Table 2. For the case with no ocean currents, the Auction-Based Alternating Algorithm Planner creates paths with numerous loops when two successive points are close together and the vehicle orientation does not allow for the second point to be reached without long manoeuvres (Figure 13a). This is avoided when using the Auction-Based Proposed Planner by generating sequences that are feasible but limit the number of additional loops (Figure 13b). Similar results are obtained with the presence of ocean currents as shown in Figure 14a and Figure 14b.

Note that the Auction-Based Proposed Planner produced different sequences for

	No current		With current	
	T_{max} % Improvement	T_{avg} % Improvement	T_{max} % Improvement	T_{avg} % Improvement
$n = 1$	36.1	36.1	42.8	42.8
$n = 2$	37.8	34.3	43.4	38.9
$n = 3$	47.2	37.6	48.6	46.7
$n = 4$	52.2	41.1	45.8	37.0
$n = 5$	41.7	31.2	44.1	40.5

Table 1: Percentage improvement of T_{max} and T_{avg} using $n = \{1, 2, 3, 4, 5\}$ and $m = \{6, 7, 8, \ldots, 20\}$.

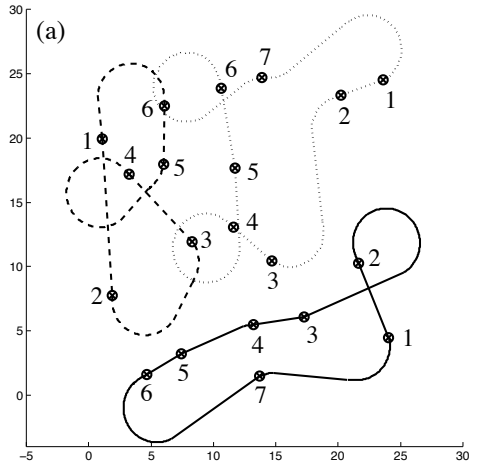


Figure 13a: Sequences generated by the Auction-Based Alternating Algorithm Planner using the dataset in Figure 8 with $n = 3$, $m = 20$, and $u_c = 0$.

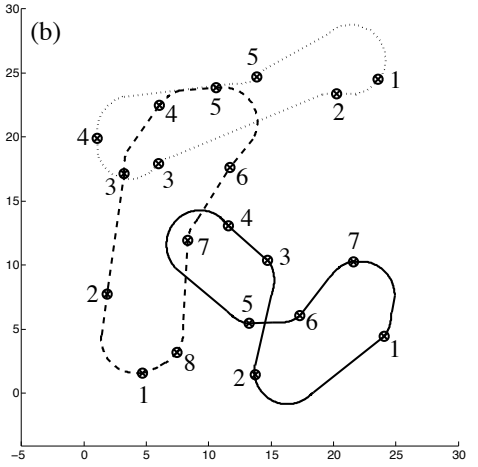


Figure 13b: Sequences generated by the Auction-Based Proposed Planner using the dataset in Figure 8 with $n = 3$, $m = 20$, and $u_c = 0$.

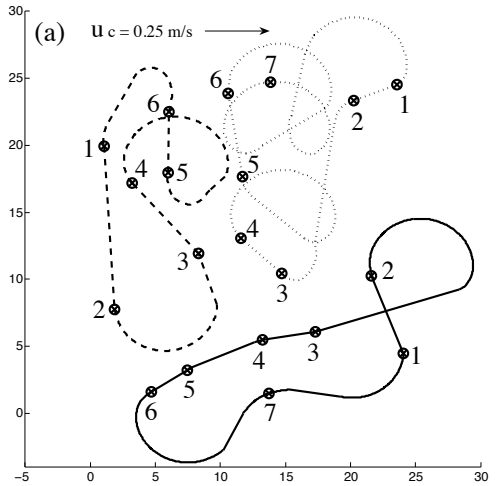


Figure 14a: Sequences generated by the Auction-Based Alternating Algorithm Planner using the dataset in Figure 8 with $n = 3$, $m = 20$, $u_c = 25$ m/s, and $\psi_C = 0$.

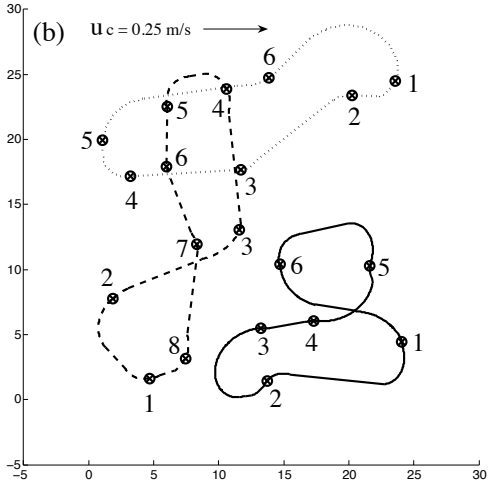


Figure 14b: Sequences generated by the Auction-Based Proposed Planner using the dataset in Figure 8 with $n = 3$, $m = 20$, $u_c = 25$ m/s, and $\psi_C = 0$.

the case with no ocean currents and the case with ocean currents. This is because the proposed path cost calculation method considers the possibility that two successive points that were reachable in the absence of ocean currents may no longer be reachable without extra loops due to an increase in turning radius from the ocean currents. Also the Auction-Based Proposed Planner attempts to avoid paths that force the vehicles to drive against the ocean current. Instead paths that allow the ocean current to aid the vehicle in the direction of travel are favoured.

EXPERIMENTAL RESULTS

Experiments were conducted at the Avila Pier in California using the Iver2 AUV as shown in Figure 15. The Iver2 is a small, low cost AUV developed by Ocean Server Technology Inc. It

	No current		With current	
	Alternating Algorithm	Proposed Planner	Alternating Algorithm	Proposed Planner
T_{max} (s)	89.9	58.4	101	59.8
T_{avg} (s)	75.1	54.5	88.1	57.1

Table 2: Summary of path costs for the dataset in Figure 8 with $n = 3$ and $m = 20$.

is 4 feet long, 6 inches in diameter, and weighs less than 50 pounds. It has independent control of all four control surfaces, a wireless network interface, a simple user interface, and a robust mechanical design. The Iver2 AUV is similar to the REMUS AUV in many aspects and, therefore, the governing equations of motion described above for the REMUS AUV also apply to the Iver2 AUV.

Missions were created based on the sequences generated by the simulations using Matlab and were tested on the Iver2 AUV. The results from running the experiments were analyzed based on the following criteria:

$$T_{\max} = \max C_{\exp}(S_i)$$
$$T_{\text{avg}} = \frac{\sum_{i=1}^{n} C_{\exp}(S_i)}{n}$$
$$D_{\text{avg}} = \frac{\sum_{j=1}^{m} D_{\min}(d_j)}{m}$$

where C_{exp} is the time taken by the Iver2 AUV to traverse the sequence S_i during a mission

Figure 15: Iver2 AUV.

and D_{min} is the minimum distance between a task point and the line indicating the actual position of the AUV during the mission.

Control Architecture

Before describing the results from field tests, the limitations on the control architecture of the Iver2 AUV must be addressed. The Iver2 AUV control architecture is based on the Underwater Vehicle Console (UVC) developed by Ocean Server Technology Inc. The UVC provides an interface to the Iver2 AUV's sensors, motors, and control processes through a remote desktop connection. However, the UVC declares victory on the approaching waypoint and will move to the next waypoint when it has reached the "waypoint success radius" which was set to 4 metres (minimum allowed value on the UVC).

Task Allocation for Multiple AUVs

To analyze the performance of the proposed path cost calculation method, experiments were conducted on three datasets, each containing 20 task points generated randomly with a uniform probability distribution inside a square with side lengths of 35 metres. These task points were allocated to three vehicles, similar to the multiple travelling salesmen problem.

The first method solves the multiple travelling salesman problem without considering the curvature constraints of the vehicle (Figure 16) and is referred to as the "MTSP Euclidean Planner." The second method tries to find feasible paths for each vehicle using the

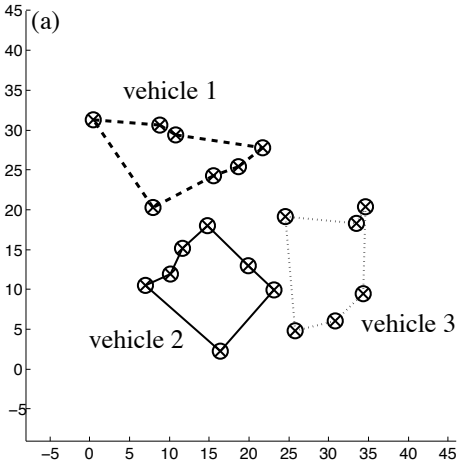


Figure 16a: Paths generated using the MTSP Euclidean Planner from MatLab.

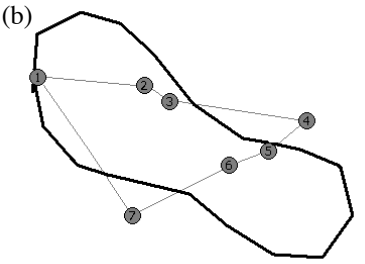


Figure 16b: Field test results for vehicle 1

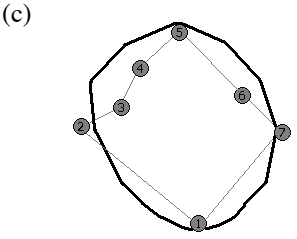


Figure 16c: Field test results for vehicle 2.

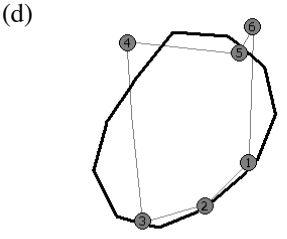


Figure 16d: Field test results for vehicle 3.

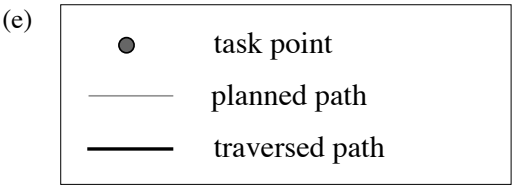


Figure 16e: Legend for (b), (c), and (d).

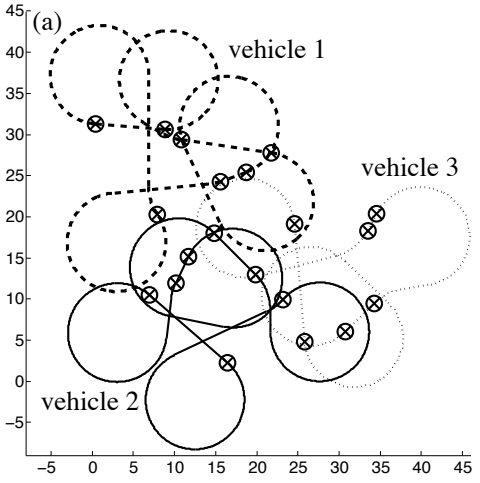


Figure 17a: Paths generated using the Auction-Based Alternating Algorithm Planner from MatLab.

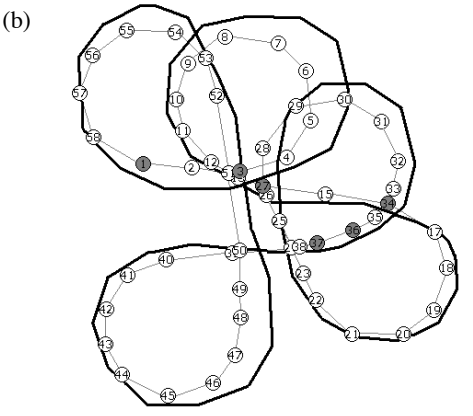


Figure 17b: Field test results for vehicle 1.

(c)

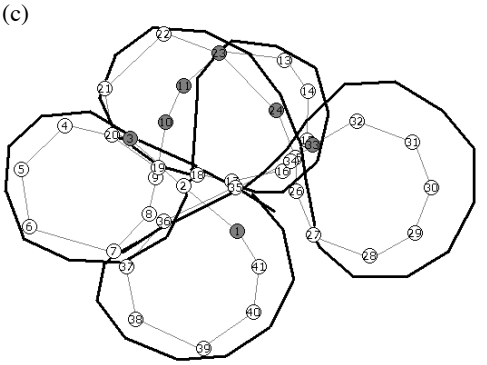


Figure 17c: Field test results for vehicle 2.

(d)

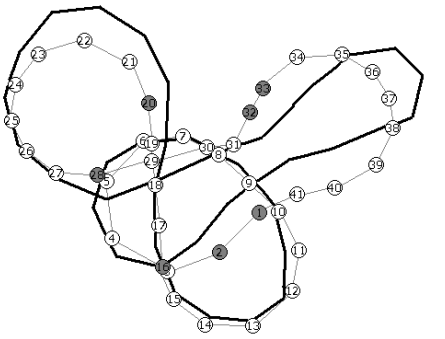


Figure 17d: Field test results for vehicle 3.

(e)

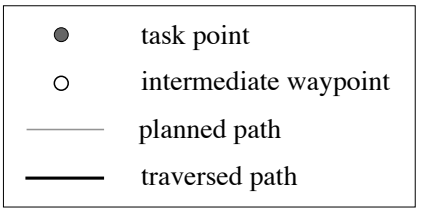


Figure 17e: Legend for (b), (c), and (d).

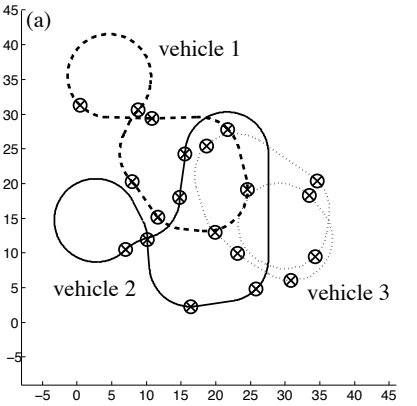


Figure 18a: Paths generated using the Auction-Based Proposed Planner from MatLab.

(b)

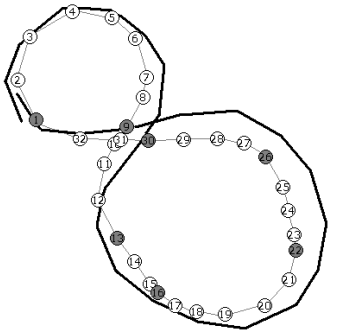


Figure 18b: Field test results for vehicle 1.

(c)

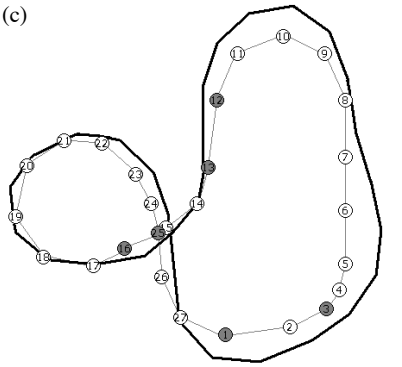


Figure 18c: Field test results for vehicle 2.

(d)

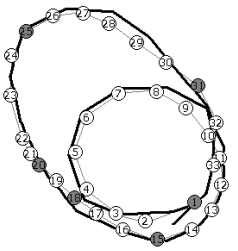


Figure 18d: Field test results for vehicle 3.

(e)

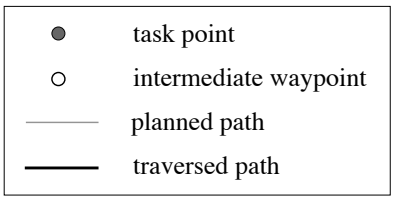


Figure 18e: Legend for (b), (c), and (d).

	T_{max} (s)	T_{avg} (s)	D_{avg} (m)
MTSP Euclidean	89.3	78.0	1.65
Alternating Algorithm	279	252	1.35
Proposed Method	145	134	0.84

Table 3: Field test results from three randomly generated datasets of 20 task points for three vehicles.

Auction-Based Alternating Algorithm Planner (Figure 17). The third method uses the Auction-Based Proposed Planner which considers the curvature constraints of the vehicle in generating the sequence for the vehicle (Figure 18). Results from field tests are summarized in Table 3.

On average, the Auction-Based Proposed Planner reduced T_{max} by 47% and reduced D_{avg} by 34% over the Auction-Based Alternating Algorithm Planner. Although the MTSP Euclidean Planner solution was 39% faster than the Auction-Based Proposed Planner solution, the average distance to task point was 49% larger since the paths generated were not feasible for the Iver2 AUV which had a turning radius of 6 metres. This resulted in the Iver2 only getting within 1.65 metres of the desired task point on average. At worst, the AUV only travelled to within 7.11 metres of one task point using the MTSP Euclidean Planner's sequence of points. The largest distance the AUV got to a task point was 3.53 metres using the Auction-Based Alternating Algorithm Planner and 2.71 metres using the Auction-Based Proposed Planner.

The Auction-Based Proposed Planner also performed better with respect to overall mission time when compared to the Auction-Based Alternating Algorithm Planner because paths were in general simpler with less loops. The Auction-Based Alternating Algorithm Planner is based on the sequence of points generated by the solving the Euclidean TSP which tends to schedule closely spaced points in a successive order. Similar to simulation results from Matlab, the Iver2 AUV was not able to drive from one point to another point nearby without long manoeuvres when the orientation of the vehicle was not "ideal." This resulted in additional loops which are harder to execute on the Iver2 AUV than straight paths, leading to longer mission times.

CONCLUSION AND FUTURE WORK

This paper addresses the task allocation of closely spaced targets for vehicles that follow paths of bounded curvature in the presence of constant ocean currents. The proposed path cost calculation method uses a modified Dubins set that considers the kinematics, dynamics, and ocean currents. Path costs are calculated using a lower order model created from the 6-DOF non-linear model to reduce the complexity of the computations.

The proposed method for path cost calculation was developed in Matlab and tested in simulations. Simulations using the full non-linear model of the REMUS AUV indicate that the BFS-Dubins Planner yielded better performance for a dense set of points when compared to the optimal Euclidean MTSP tours generated by the BFS-Euclidean Planner. For greater than 16 task points, the Auction-Based Proposed Planner was compared to the Auction-Based Alternating Algorithm Planner. It was shown that solutions based on computing Euclidean tours that do not have curvature constraints have extra loops when task points are close together relative to the turning radius of the vehicle.

To validate the proposed method for path cost calculation in a real world application, the Iver2 AUV was used for testing at the Avila Pier in California. Analysis of the log files indicated that the Auction-Based Proposed Planner outperformed the Auction-Based Alternating Algorithm Planner with respect to the overall mission time as well as the average distance to task points. The Auction-Based Proposed Planner produced paths through a set of task points that were feasible for the Iver2 AUV to track closely, even in the presence of ocean currents. From the results, it has been demonstrated that considering the kinematic constraints and ocean currents is essential for minimizing path costs when targets are closely spaced.

Future work will consist of an algorithm capable of generating paths for AUVs to track in a dynamically evolving ocean utilizing ocean model predictions. It will also incorporate underwater obstacles or instantaneous events for dynamic task allocation. Another extension is to perform trajectory planning (i.e. path parameterized by time). With a trajectory planner, the algorithm should be capable of performing collision checks with other vehicles as well as handle time-indexed waypoints. This will make the optimization problem more complex as AUV powering would have to be considered. However, it will allow multiple AUVs to operate safely in an environment with highly varying currents.

REFERENCES

Alvarez, A., Caiti, A., and Onken, R. [2004]. Evolutionary path planning for autonomous underwater vehicles in a variable ocean. *IEEE Journal of Oceanic Engineering*, Vol. 29, No. 2, pp. 418-429.

Davis, R.E., Leonard, N.E., and Fratantoni, D.M. [2008]. Routing strategies for underwater gliders. *Deep-Sea Research II*.

Dias, M.B., Zlot, R., Kalra, N., and Stentz, A. [2006, July]. *Market-based multirobot coordination: a survey and analysis*. Conference proceedings, IEEE Special Issue on Multirobot Systems, Vol. 94, No. 7, pp. 1257-1270.

Dubins, L.E. [1957, July]. On curves of minimum length with a constraint on average curvature and with prescribed initial and terminal position and tangents. *American Journal of Mathematics*, Vol. 79, No. 3, pp. 497-516.

Gerkey, B.P. and Matarić, M.J. [2004]. A formal analysis and taxonomy of task allocation in multi-robot systems. *International Journal of Robotics Research*, Vol. 23, No. 9, pp. 939-954.

Hartigan, J.A. and Wong, M.A. [1979]. A k-means clustering algorithm. *Journal of Applied Statistics*, Vol. 28, No. 1, pp. 100-108.

Jeyaraman, S., Tsourdoas, A., Zbikowski, R., White, B., Bruyere, L., Rabbath, C.-A., and Gagnon, E. [2004]. *Formalised hybrid control scheme for an UAV group using Dubins Set and Model Checking*. Conference Proceedings, IEEE Conference on Decision and Control, Paradise Island, Bahamas, Vol. 4, pp. 4299-4304.

Korte, B. and Vygen, J. [2006]. *Combinatorial optimization: theory and algorithms*, 3rd ed. Germany: Springer.

Kruger, D., Stolkin, R., Blum, A., and Briganti, J. [2007]. *Optimal AUV path planning for extended missions in complex, fast-flowing estuarine environments*. Conference Proceedings, IEEE Conference on Robotics and Automation, Roma, Italy, pp. 4265-4270.

Lagoudakis, M., Berhault, M., Koenig, S., Keskinocak, P., and Kleywegt, A. [2004]. Simple auctions with performance guarantees for multi-robot task allocation. Conference Proceedings, IEEE International Conference on Intelligent Robots and Systems, Vol. 1, pp. 698-705.

Parker, L.E. [1998]. ALLIANCE: an architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 2, pp. 220-240.

Prestero, T. [1994]. *Verification of a six-degree of freedom simulation model for the REMUS autonomous underwater vehicle* (master's thesis). Massachusetts Institute of Technology, Cambridge.

Sariel, S., Balch, T., and Erdogan, N. [2008]. Naval mine countermeasure missions: a distributed, incremental multirobot task selection scheme. *IEEE Robotics and Automation Magazine*, Vol. 15, No. 1, pp. 45-52.

Savla, K., Frazzoli, E., and Bullo, F. [2008]. Traveling salesperson problems for the Dubins' vehicle. *IEEE Transactions on Automatic Control*, Vol. 53, No. 6, pp. 1378-1391.

Shkel, A.M. and Lumelsky, V. [2001, March]. Classification of the Dubins set. *Robotics and Autonomous Systems*, Vol. 34, No. 4, pp. 179-274.