AIAA Guidance, Navigation, and Control Conference
2 - 5 August 2010, Toronto, Ontario Canada

AIAA 2010-7572

Optimization of The Motion of a Mobile Gateway to Improve Connectivity in a Network of Autonomous Underwater Vehicles

Camila C. Françolin*
Anil V. Rao†
Department of Mechanical and Aerospace Engineering
University of Florida
Gainesville, FL 32611

Christiane Duarte‡
Gerald Martel§
Naval Undersea Warfare Center
Newport, RI 02841

The concept of using a mobile gateway vehicle to improve the connectivity of a network of underwater vehicles is investigated. A communications architecture is used to develop a simulation of a fleet of autonomous underwater vehicles patrolling an area of interest. A mobile gateway surface vehicle is added to the configuration and is commanded to move in an optimal manner so its location improves the connectivity of the underwater vehicle network. The simulation framework is tested on various underwater vehicle configurations in order to show the difference in the connectivity of the network with and without the gateway surface vehicle. The results of this study show that including a mobile gateway significantly improves the connectivity of the underwater vehicle network.

Nomenclature

p_i	= position of agent A_i in three-dimensional Euclidean space
α	= radius of communications limit
$\Delta_{i/k}$	= distance between agents i and k
λ	= one time interval
θ	= vehicle heading
A_0	= denotes the gateway vehicle
A_i	= Agent i
C_{max}	= maximum number of connections in a network
J_f	= final time step

*Ph.D. Student. Email: francoli@ufl.edu.
†Assistant Professor. E-mail: anilvrao@ufl.edu. Corresponding Author.
‡Email: christiane.duarte@navy.mil
§Email: gerald.martel@navy.mil

N_C = connectivity number
N_P = propagation number
u = vehicle maximum turn rate
v = vehicle velocity
W_i = optimization weight associated with agent i
x = longitudinal position expressed in UTM coordinates
y = latitudinal position expressed in UTM coordinates

I. Introduction

In recent years the problem of autonomous control of underwater vehicles has been of great interest. Applications for underwater vehicle autonomy include ocean floor mapping, mine sweeping, and area patrolling. Such tasks are usually accomplished through coordinated control of cooperating vehicles, making it necessary to incorporate communications and connectivity constraints to the motion of each vehicle in the network. Previous research on improving underwater vehicle network connectivity has focused on restricting the motion of the underwater vehicles so that each vehicle is required to stay within communications range of one another.[1–6] Ref. 3 relied on maintaining vehicle distances and line-of-sight restrictions, while Ref. 5 devised a plan to improve connectivity in a network by a method of reactive behavior coupled with planning. Ref. 6 devised a reactive control strategy in which the controller is a weighted sum of two goals: the first term guides each robot to its goal position and the second term maintains the constraints that need to be satisfied to maintain network connectivity. These algorithms, however, often prove to be too restrictive as vehicles have to choose between performing mission requirements and maintaining connectivity.[4] As opposed to restricting the motion of the underwater vehicles, in this paper we improve the connectivity of an underwater vehicle network by introducing an Autonomous Surface Vehicle (ASV) into the network to act as a mobile gateway. The motion of the ASV is commanded in an optimal manner by solving a nonlinear optimal control problem using a real-time software implementation of the Gauss Pseudospectral Method.[7–10]

Traditionally, path planning in autonomous systems has been achieved through the use of search algorithms such as *A**.[11] More recently, Ref. 12 proposed a modified fast marching algorithm *FM** which provides a continuous path when it is implemented in a discrete representation of the environment. FM* allowed the curvature of the final path to be constrained, which enabled the turning radius of the robot in use to be taken into account. Ref. 13 presented an interpolation-based planning and re-planning algorithm called *Field D** in which the cost could be calculated for arbitrary positions within the grid and was not constrained to a small set of possible headings. Ref. 14 uses a method of sequential quadratic programming to perform path planning with obstacle avoidance where the obstacles are posed as constraints in the search space. Genetic algorithms have also been used toward motion planning of underwater vehicles, though desired fitness levels are not always attained.[15,16]

Cooperative networks of autonomous vehicles base behavioral or autonomous decisions on information gathered about the environment as well as about the rest of the network. Environmental information can typically be sensed by each vehicle, however information about the network must be passed from vehicle to vehicle. The network of Autonomous Underwater Vehicles (AUVs) considered in this paper is designed so that each vehicle patrols a particular zone of a patrol area and transmits a "status" message at a prescribed time step. The status message broadcast by each vehicle in the network contains situational information about that particular vehicle, as well as about the network in general. It is thus imperative that all vehicles stay connected through communications with other vehicles, so that the network stays cohesive. The mobile gateway is a surface ship that relays status messages between vehicles that are conceivably out of communications range.

The contributions of this paper are as follows: (1) application of an integrated communications framework and guidance to control the motion of a network of AUVs patrolling a region of interest; (2) implementation of behavioral controllers on AUVs to execute a mission requirement; and (3) a real-time implementation of a pseudospectral method to generate a guidance command for a surface vehicle, optimizing its motion to remain within communication range of the network and thereby improving the connectivity of the AUV network.

II. Problem Formulation

Consider a network of n cooperating AUVs patrolling an area of interest. We define each AUV in the network as an agent, denoting by $p_i(t_j)$ the position of agent A_i in three-dimensional Euclidean space, $\mathbb{E}^3$, at time step t_j. The network is dynamic in size and reconfigurable, where agents are continuously leaving or joining the mission according to their energy state. Each agent is independently controlled through the built-in behaviors *waypoint and loiter*, with capabilities also including *retask and refuel*, necessary to fulfill the mission requirements. All agents communicate through a *status* message which relays situational information about each particular agent as well as the network in general.

Each agent broadcasts its current estimated position $p_i(t_j)$ as latitude and longitude expressed in decimal degrees. Internally, this position is converted into a northing and easting coordinate pair of the Universal Transverse Mercator (UTM) coordinate system. The UTM coordinate system is a grid-based method of representing positions on the earth by breaking it up into a series of sixty zones, and approximating a flat earth inside each zone. Using this method, we define $\Delta_{i/k}(t_j)$ as the distance between agents i and k at time step t_j

$$\Delta_{i/k}(t_j) = \sqrt{(x_i^{utm}(t_j) - x_k^{utm}(t_j))^2 + (y_i^{utm}(t_j) - y_k^{utm}(t_j))^2} \tag{1}$$

As a starting point, we assume a purely distance-based model for the underwater modem communications signal strength. Underwater modems can have a large connectivity range, depending on water depth, quality, and communications bandwidth requirements. For our simulations we have assumed a binary connectivity model which decays to zero at a distance of $\alpha = 750$ meters. Using this model for the strength of the modem communication signal, we then define the network's *connectivity* as the number of *included* agents throughout the simulation normalized by the number of available connections in the network. An agent is said to be *included* if it is within communications range of at least one of its adjacent neighbors at time step t_j. The connectivity number N_C is then defined as

$$N_C \equiv \frac{\sum_{i=1}^{n}\sum_{k=1}^{n} bool(A_i \leftrightarrow A_k)}{n(n-1)} \tag{2}$$

where n is the number of active agents in the network, and $bool(A_i \leftrightarrow A_k)$ is the Boolean operation which defines the inclusion of agent i with respect to agent k, as defined in Table 1.

As important as the concept of connectivity in a network is the concept of information propagation. Each agent in the network carries situational information about not only itself but also about all other agents in the network. Information propagation is thus an important concept which relates how information heard by one agent will be made available to other agents through *secondary connections*. We then define a propagation number N_P which is an average of the sum of

Table 1: Condition for Boolean operation $A_i \leftrightarrow A_k$.

$\mathbf{A_i \leftrightarrow A_k}$	**Condition**
0	$i = k$ or $\Delta_{i/k}(t_j) > 750$
1	$i \neq k$ and $\Delta_{i/k}(t_j) \leq 750$

included agents and *secondary connections* in the network throughout the simulation.

$$N_P \equiv \frac{N_C + \sum_{i=1}^{n} \sum_{k=1}^{n} bool(A_i \Leftrightarrow A_{kp})}{n(n-1)} \qquad p = 1, \ldots, n, \quad p \neq i, \quad p \neq k \tag{3}$$

Where $bool(A_i \Leftrightarrow A_{kp})$ is the Boolean operation defining the existence of secondary connections between agents i and p, as seen in Table 2. The connectivity number and the propagation number give us distinct and important information about the cohesiveness of the network. The connectivity number gives an average of how connected the network is at each time step, whereas the propagation number gives an overall look at how cohesive the network is throughout the entire simulation.

Table 2: Condition for Boolean operation $A_i \Leftrightarrow A_{kp}$.

$\mathbf{A_i \Leftrightarrow A_{kp}}$	**Condition**
0	$bool(A_i \leftrightarrow A_k) = 0$
1	$bool(A_i \leftrightarrow A_k) = 1$ and $\Delta_{k/p}(t_j) \leq 750$

Fig. 1 gives a graphical representation of the mission for this simulation scenario. The area to be patrolled can be broken down into concentric rings, or tiers, encircling an area of interest. The tiers are deemed to be more important the closer they are to the center of the ring. Each tier is also subdivided into zones, where the number of zones increases in number according to the size of the tier's surface area. A tasking behavior assigns each agent in the simulation a tier and a position at initialization; this position will be the vehicle's patrol zone i and will define that agent's *task* until further reconfiguration is needed due to the addition or removal of agents for refueling purposes.

Each agent patrols their respective zone using a *waypoint* behavior, in which the waypoints are set up in a "figure eight" pattern. Time is discretized into intervals, where an interval $\lambda = t_{j+1} - t_j$ is defined as the time necessary for an individual agent to go from waypoint *j* to waypoint *j+1*. The waypoints are distributed inside each patrol zone in such a way as to make each time interval uniformly $\lambda = 150$ seconds.

All agents in the network communicate through an underwater modem on the same frequency channel using a Time Division Multiple Access (TDMA) scheme, in which they each send out their status message sequentially. This scheme prevents communications overruns ensuring all the communications are "heard" throughout the network. Each vehicle broadcasts their message once before reaching a waypoint and once after leaving a waypoint, that is at the beginning and end of each time interval. The status messages broadcast throughout the network are essential to

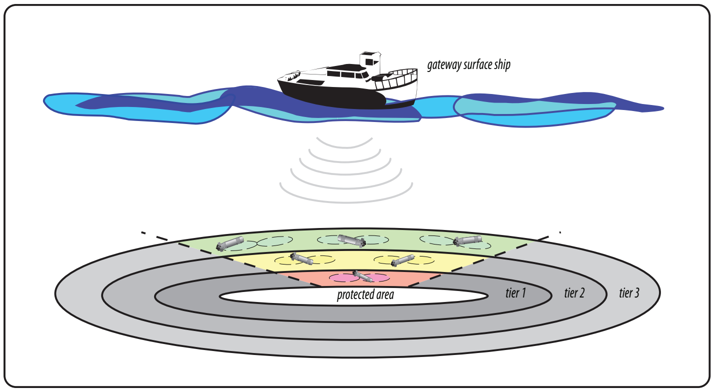

Figure 1: Graphical representation of the simulation scenario; The area of interest is divided into tiers and zones. Each vehicle is assigned a patrol zone at initialization. These zones are reconfigured as vehicles join or leave the network.

the mission completion; the behaviors built into each agent use the information acquired through these status messages to make behavioral decisions during the mission. Table 3 describes each field in the status message and gives the number of bits each field is allotted.

We focus our attention to one quarter of the patrolled circumference, assuming symmetry to the remaining areas. Assuming three surrounding tiers, the patrolled area is broken down into six distinct zones, each patrolled by an agent, giving a maximum number of six patrolling agents. As agents must leave their patrol zones to refuel, it becomes necessary to reconfigure the patrolling zones to account for the decreased number of agents. Fig. 2 shows how this reconfiguration is done as vehicles leave or re-enter the network. It can be seen from the reconfiguration diagram that the minimum number of agents to patrol the area is three. If any agents need to refuel while in this minimum configuration, it will have to wait for another agent to return before leaving. This effort can be coordinated between the agents through communications of power resources in their respective status messages along with the *refuel* and *retask* behaviors.

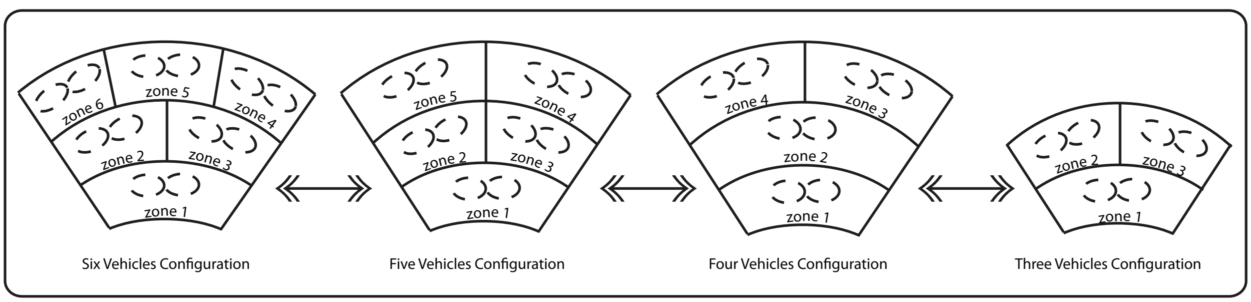

Figure 2: Patrol zones reconfiguration as vehicles must leave to refuel.

A. Mobile Gateway

We aim to improve the connectivity of the network by the introduction of a mobile gateway. The main purpose of the gateway vehicle is to relay status messages between vehicles that are conceivably out of communications range. It does so by performing a real-time optimization of the network connectivity, positioning itself to maximize transmissions. We implement the mobile

Table 3: Status message format and field sizes.

Field	No. of Bits	Notes
Message ID	4	1 for status message, 2 for other
Platform ID	5	Vehicle Identification No.
Destination ID	5	0 for network broadcast
Timestamp	32	In UTC *seconds*
Latitude	23	Current latitude in *decimal degrees*
Longitude	24	Current longitude in *decimal degrees*
Depth	10	Current depth in *meters*
Heading	11	Current heading in *degrees* (North = 0°)
Speed	8	Current speed in *m/s*
Power	10	Current vehicle energy status
Ocean Current	23	Vector indicating ocean current, if available
Current Task	4	Current vehicle patrol zone
Current Plan	4	Current network tasking plan/ No. of patrol vehicles
Next Latitude	23	Next waypoint latitude in *decimal degrees*
Next Longitude	24	Next waypoint longitude in *decimal degrees*
Vehicle 1 Status	4	*Integer* number of time steps since last communication with A_1
Vehicle 2 Status	4	*Integer* number of time steps since last communication with A_2
Vehicle 3 Status	4	*Integer* number of time steps since last communication with A_3
Vehicle 4 Status	4	*Integer* number of time steps since last communication with A_4
Vehicle 5 Status	4	*Integer* number of time steps since last communication with A_5
Vehicle 6 Status	4	*Integer* number of time steps since last communication with A_6
Bit Sum	**234**	**30 bytes**

gateway as an ASV, acting as agent A_0 , and having no restrictions on its motion in $\mathbb{R}^2$. We assume a maximum speed for the gateway of twice that of the underwater agents, such that it may span the entire patrol area and is not confined to a zone.

The mobile gateway autonomous controller is a process implemented in MATLAB. There are two major components that drive the mobile gateway. The solution to an optimization problem acts as the vehicle's guidance at each time step, while a behavior-based controller controls the vehicle. The controller makes decisions based on information received from the AUVs through each status message. As can be seen in Fig. 3, the guidance algorithm is run from a loop that is closed at each prescribed time step.

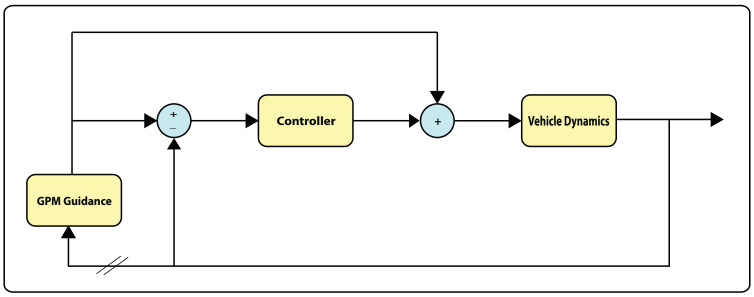

Figure 3: Block diagram representation of Mobile Gateway System.

The gateway solves a guidance optimization problem at each time step that can be described as follows. Minimize the cost functional

$$J = -\sum_{i=1}^{n} \mathbf{W_i} \arctan(\alpha - \mathbf{\Delta_{0/i}}) \tag{4}$$

Subject to the dynamic constraints of the vehicle. We assume a fully actuated kinematic model for the gateway

$$\begin{aligned} \dot{x} &= v\cos(\theta) \\ \dot{y} &= v\sin(\theta) \\ -u \leq \dot{\theta} &\leq u \end{aligned} \tag{5}$$

and subject to the constraint $t_f \leq \lambda$. W_i is an optimization weight associated with agent i, α is the communications range parameter, $\Delta_{0/i}$ is the UTM distance between the gateway and agent i, x and y are the longitudinal and latitudinal coordinates of the position of the gateway expressed in UTM coordinates, v is the velocity of the vehicle, θ is the heading of the vehicle, and u is the maximum turn rate of the vehicle. The constants α and v, together with the desired time interval λ, are set in a configuration file that is read at startup.

In order for the optimizer to maximize communications at each time step, it must choose the necessary information from the status message of each AUV. Most importantly, the gateway must know where each AUV will be at the beginning of the next time step so it has time to position itself optimally for the beginning of the next transmission. Also important in the status message are the current plan that the network is performing, which indicates how many agents are on task, and the current task each vehicle is performing, which indicates what zone that particular vehicle is patrolling.

The cost function for the optimization given in Eq. (4) was developed with the intention to allow the optimizer to pick the solution that keeps the maximum number of AUVs inside the gateways communications range. Fig. 4 is a plot of our cost function for one vehicle, J_1, as a function of the parameter $\Delta_{0/i}$ for the case when $\alpha = 750$ m. It can be seen from this plot that the cost function sharply crosses the x-axis at the point where modem communications cease. In order for the cost to be maximized the vehicles must therefore be within range.

Each vehicle in the network is also assigned a weight W_i described by Eq. 6, which is dependent on how many time steps have elapsed since that particular vehicle has last successfully communicated its status message.

$$W_i = \frac{TimeStamp(t_j) - TimeStamp(t_{j-1})}{\lambda} \tag{6}$$

This weight will tilt the optimization such that it will favor being closer to vehicles that have been out of communications range the longest.

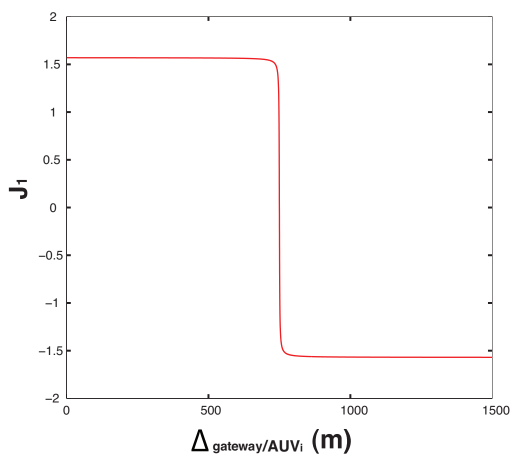

Figure 4: Cost Function J as a function of distance for a vehicle (n=1) with unity weight and a communications range parameter of $\alpha = 750m$. It can be seen that the function sharply crosses the x-axis at the point where communications cease.

Table 4: Latitudinal and longitudinal positions for all agents in test case scenarios. Positions were chosen in a non-trivial arrangement such that it is impossible for the gateway vehicle to maintain all agents within communications range.

	Agent 0	**Agent 1**	**Agent 2**	**Agent 3**	**Agent 4**	**Agent 5**
x Position (m)	750	50	1,450	1,450	750	50
y Position (m)	50	50	50	850	850	850

In order to evaluate our optimizer performance, we run a series of test cases with $n = 6$ agents, five underwater vehicles and the gateway vehicle, at a time step t_1 of length $\lambda = 150$ s and a communications range parameter of $\alpha = 750$ m. We place the underwater agents in a non-trivial arrangement such that it is impossible to maintain all agents within communications range of the gateway vehicle. Table 4 describes the x and y locations of each agent. The initial position of the gateway vehicle is denoted by a "$*$," and its overall path during the time interval is shown as the blue line, ending with a triangle marker. Each agent is denoted as a circle if it is within the gateway's communications range, and an x if it is outside the gateway's communications range at the end of the time step. We first examine the case where all five underwater vehicles carry the same weight $W_i = 1$ and notice that the gateway attempts to position itself in the middle of the five underwater agents, thus ensuring communications with a maximum possible number of underwater agents, a desirable result. We then increase the weight of agent A_1, which was out of communications in the previous case, to $W_1 = 2$. With this change, we see the gateway moves closer to this agent, giving it communications preference due to its greater weight. Finally, we change the weight of A_2 such that $W_2 = 3$ and again notice the desired performance as the gateway aims to maintain the network connectivity of the two vehicles with the greatest weight. Fig. 5 shows the test cases results.

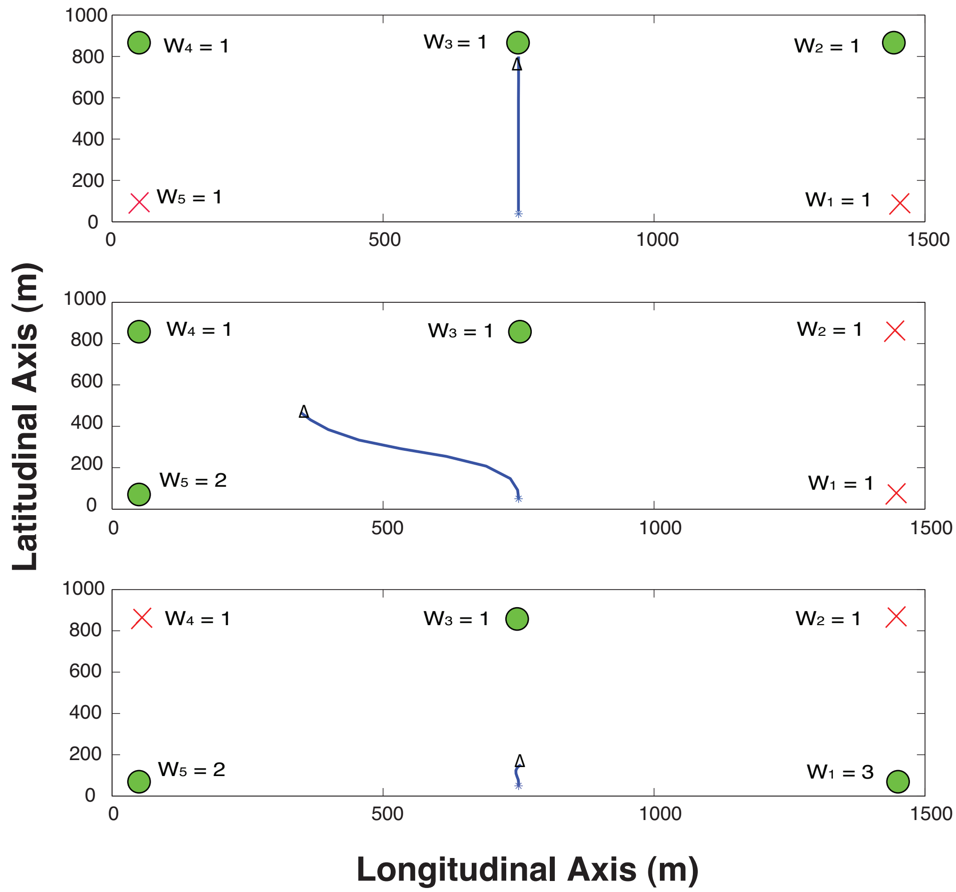

Figure 5: Test Cases showing performance of mobile gateway's optimizing function. The gateway vehicle's initial location is denoted by an asterisk, and its overall path during the time interval is shown as the blue line, ending in a triangle marker. Each agent is denoted as a circle if it is within the gateway's communications range, and an x if it is outside the gateway's communications range at the end of the time step. It can be seen that the gateway attempts to reach communication range of agents with higher weights.

III. Simulation Framework

The simulation was implemented within the Mission Oriented Operating Suite/Interval Programming (MOOS-IvP) architecture for autonomous control, developed and maintained by the Naval Undersea Warfare Center, the Massachusetts Institute of Technology (MIT), and Oxford University. MOOS-IvP is a set of open-source C++ modules for controlling the operations of autonomous marine vehicles.[17] The suite consists of a set of distinct processes communicating through a *publish-subscribe* type database called the MOOSDB.[17] Each variable *published* to the MOOSDB by a process will be available to any other processes which are *subscribed* to that particular variable. Variable publications and subscriptions can be done in real-time. IvP stands for Interval Programming, which is a mathematical programming model for multi-objective optimization.[17]

The MOOS architecture was chosen for this simulation due to its completeness and proven ability to handle communications for networks of cooperating underwater vehicles.[18,19] The MOOS processes include the necessary control functions as well communications modules, with the MOOSDB providing the unified interface standard that enables the fully autonomous integration of modeling, processing, and control. Vehicle control in the MOOS architecture is handled through the idea of separation between *vehicle control* and *vehicle autonomy*. The vehicle *dynamic control* is a MOOS process referred to as the *frontseat driver* which provides a navigation and control system steering the vehicle capable of streaming the vehicle's position and trajectory information to the vehicle's *autonomy* controller, called the *backseat driver*, and accepting back a stream of autonomy decisions that guide the vehicle such as heading, speed, and depth. Thus the vehicle is controlled through a frequent exchange of data between the *autonomy* controller, or the *backseat driver*, and the *dynamic* controller, or the *frontseat driver*; all guidance decisions being made by the *autonomy* controller, while all navigational information is handled by the *dynamic* controller.[17] For in-water tests the *frontseat driver* controls the vehicle directly.[20] For our simulation purposes, we use a vehicle dynamics model implemented in C++ which mimics the vehicle's motion in $\mathbb{E}^3$ and is controlled by input of the desired *speed, depth* and *heading*.

The MOOS communications suite includes processes which handle the parsing and distribution of each agent's status message. This stack is exclusively developed and maintained by MIT's Laboratory for Autonomous Marine Sensing Systems (LAMSS).[21] The processes used consist of *pAcommsHandler*, and *pGeneralCodec*, which are linked through the MOOSDB. The process *pAcommsHandler* queues all messages which are to be sent by the underwater modem according to a priority level, and *pGeneralCodec* is a compressor/decompressor which encodes *status message* data into binary format for transmission by the underwater modem, and decodes the data received from the underwater modem. The format of the data is defined in a human-readable XML file.

As was previously mentioned, the autonomy of all underwater agents is handled with MOOS behaviors which include *waypoint and loiter*. The real-time simulation then includes running an environment model along with the underwater modem application and each agent's dynamics simulation in one computer, running each agent's controller along with any necessary behaviors and the MOOSDB in another computer equipped with the MOOS communications suite, and running the gateway *autonomy* controller in a MATLAB equipped computer. This simulation structure can be seen with all its components in Fig. 6.

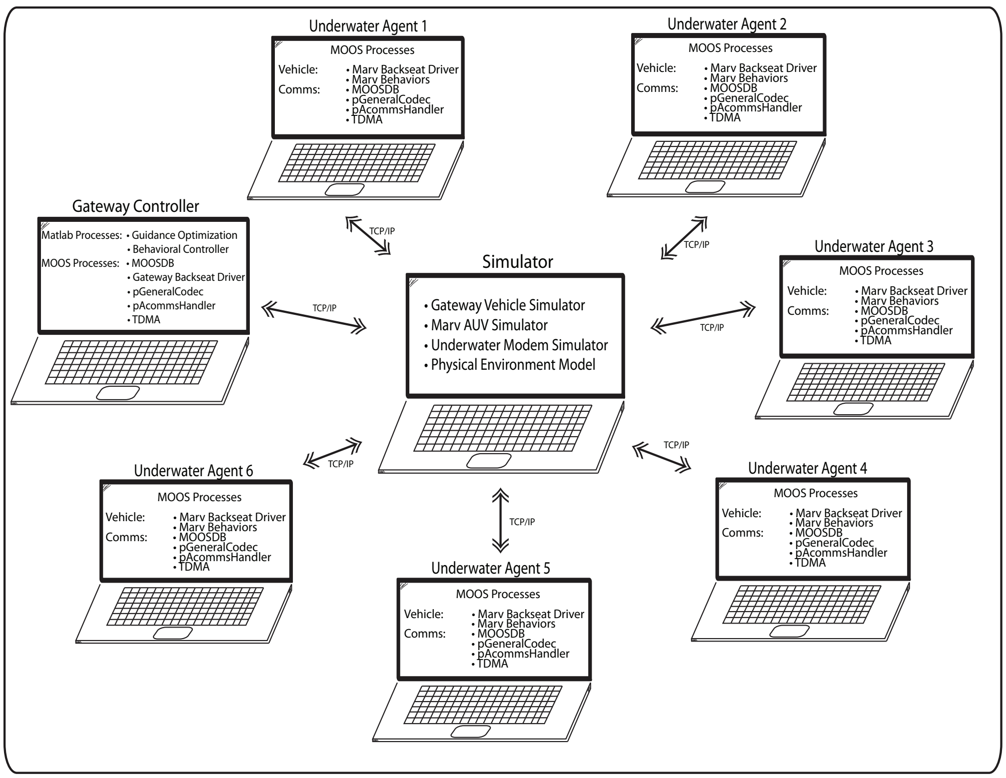

Figure 6: Schematic of simulation framework. Each computer runs a simulated vehicle along with the necessary MOOS communication processes.

A. Guidance Algorithm

The solution to the optimization problem which provides the guidance commands to the gateway vehicle is found using GPOPS, an open-source MATLAB implementation of the Gauss Pseudospectral Method (GPM).[10] The Gauss Pseudospectral Method is a direct collocation method for solving optimal control problems. In the GPM, the continuous-time optimal control problem is transcribed to a nonlinear programming problem (NLP).[7,8] The resulting NLP can be solved numerically by well-developed algorithms, which attempt to satisfy the set of Karush-Kuhn-Tucker (KKT) conditions associated with the NLP.[7,8]

The Gauss Pseudospectral Method was used as the optimization algorithm for the mobile gateway due to its proven robustness to solve such optimization problems.[22] Pseudospectral methods are a class of *direct collocation* where the optimal control problem is transcribed to a nonlinear programming problem by parameterizing the state and control using global polynomials and collocating the differential-algebraic equations using nodes obtained from a Gaussian quadrature. The three most commonly used sets of collocation points are *Legendre-Gauss* (LG), *Legendre-Gauss-Radau* (LGR), and *Legendre-Gauss-Lobatto* (LGL) points. The GPM focuses on the LG collocation scheme, where the differential-algebraic equations are collocated in the domain $[-1, 1]$, but contain neither of the endpoints. A brief description of the GPM follows.

Consider the following optimal control problem in Bolza form. Determine the state, $\mathbf{x}(\tau) \in \mathbb{R}^p$, control, $\mathbf{u}(\tau) \in \mathbb{R}^m$, initial time, t_0, and final time, t_f, that minimize the cost functional

$$J = \Phi(\mathbf{x}(-1), t_0, \mathbf{x}(1), t_f) + \frac{t_f - t_0}{2} \int_{-1}^{1} g(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) d\tau \tag{7}$$

subject to the constraints

$$\frac{d\mathbf{x}}{d\tau} = \frac{t_f - t_0}{2} \mathbf{f}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \tag{8}$$

$$\boldsymbol{\phi}(\mathbf{x}(-1), t_0, \mathbf{x}(1), t_f) = \mathbf{0} \tag{9}$$

$$\mathbf{C}(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau; t_0, t_f) \leq \mathbf{0} \tag{10}$$

It is noted that this problem formulation can be transcribed to the time interval $t \in [t_0, t_f]$ via the affine transformation

$$t = \frac{t_f - t_0}{2}\tau + \frac{t_f + t_0}{2} \tag{11}$$

Now, in the GPM the state is approximated using the initial point and N LG points as

$$\mathbf{x}(\tau) \approx \mathbf{X}(\tau) = \sum_{i=0}^{N} \mathcal{L}_i(\tau)\mathbf{x}(\tau_i) \tag{12}$$

where the Lagrange polynomials $\mathcal{L}_i(\tau)$ $(i = 0, \ldots, N)$ are defined as

$$\mathcal{L}_i(\tau) = \prod_{\substack{j=0 \\ j \neq i}}^{N} \frac{\tau - \tau_j}{\tau_i - \tau_j} \tag{13}$$

The approximation to the time derivative of the state is then applied at the N LG collocation points as

$$\dot{\mathbf{x}}(\tau_k) \approx \dot{\mathbf{X}}(\tau_k) = \sum_{i=0}^{N} \dot{\mathcal{L}}_i(\tau_k)\mathbf{X}(\tau_i) = \sum_{i=0}^{N} D_{ki}\mathbf{X}(\tau_i), \qquad (k = 1, \ldots, N) \tag{14}$$

where $D_{ki}, \ (k = 1, \ldots, N; i = 0, \ldots, N)$ is the $(N) \times (N+1)$ *GPM differentiation matrix*. The dynamics are collocated at the N LG points as

$$\sum_{i=0}^{N} D_{ki}\mathbf{X}(\tau_i) - \frac{t_f - t_0}{2}\mathbf{f}(\mathbf{X}(\tau_k), \mathbf{U}(\tau_k), \tau_k; t_0, t_f) = \mathbf{0}, \qquad (k = 1, \ldots, N) \tag{15}$$

Next, in order to account for the initial and terminal points (i.e., the *boundary points* $\tau_1 = -1$ and $\tau_N = 1$), an additional variable $\mathbf{X}(\tau_N)$ is defined via a Gauss quadrature as

$$\mathbf{X}(\tau_N) \equiv \mathbf{X}(\tau_1) + \frac{t_f - t_0}{2}\sum_{i=1}^{N} w_i \mathbf{f}(\mathbf{X}(\tau_i), \mathbf{U}(\tau_i), \tau_i; t_0, t_f) \tag{16}$$

where w_i are the Gauss weights. The continuous cost function is approximated using a Gauss quadrature as

$$J = \Phi(\mathbf{X}_0, t_0, \mathbf{X}_f, t_f) + \frac{t_f - t_0}{2}\sum_{i=2}^{N-1} w_i g(\mathbf{X}_i, \mathbf{U}_i, \tau_i; t_0, t_f) \tag{17}$$

Finally, the boundary constraints and path constraints can be expressed as

$$\begin{aligned} &\phi(\mathbf{X}_0, t_0, \mathbf{X}_f, t_f) = \mathbf{0} \\ &\mathbf{C}(\mathbf{X}_k, \mathbf{U}_k, \tau_k, t_0, t_f) \leq \mathbf{0}, \qquad (k = 1, \ldots, N) \end{aligned} \tag{18}$$

The cost function and the algebraic constraints define a Non-Linear Program (NLP) whose solution is an approximate solution to the continuous optimal control problem. The open-source MATLAB implementation of this method, GPOPS, utilizes the sparse nonlinear optimizer SNOPT.[23]

An interpolated initial guess is used to initialize the optimization in order to avoid convergence to a local minimum. The initial guess is found very simply through a linear interpolation from the initial gateway position $p_0(t_j)$ at time t_j to a final position $p_0(t_{j+1})$ at time t_{j+1} which is assumed to be at the centroid of the system composed of all agents in the network at time t_{j+1}

$$p_0(t_{j+1}) = \frac{\sum_{i=1}^{n} p_i(t_{j+1})}{n} \tag{19}$$

B. Gateway Behavioral Control

The controller is initialized when the first set of status messages is received. All status messages are received through the MOOS database, which is simulating all communications processes. Once this initialization takes place, the optimization will run twice at each prescribed time step. The first cycle of the optimizer runs with the weights of each vehicle identically equal to one, and is labeled as *solution 1*. The second cycle of the optimization is run with the calculated weights as described by eq. 6, and is labeled as *solution 2*, for all vehicles. This is done to ensure that if a solution exists which keeps all existing vehicles in range it will not be overlooked because of unequal weights. One of these two solutions is then chosen according to a set of conditions as found in table 5. The conditions are structured in an *if-then-else* construct, such that priority is given to the conditions from the top down.

As was previously mentioned, in this simulation scenario each AUV patrols a particular zone of the area of interest. The zone each vehicle patrols is indicated by its current task, which is broadcast in the status message. If a vehicle is outside communications range for more than three

Table 5: Decision hierarchy for behavioral controller to accept its next position from the guidance solutions.

Choice	Condition	Solution Used
1	All agents in solution 1 are in range	Solution 1
2	All agents in solution 2 are in range	Solution 2
3	More agents in solution 1 are in range than in 2	Solution 1
4	More agents in solution 2 are in range than in 1	Solution 2
5	Solution 1 encompasses agents with weight ≥ 2	Solution 1
6	Solution 2 encompasses agents with weight ≥ 2	Solution 2
7	SNOPT did not reach a minimum for solution 2	Solution 1
8	SNOPT did not reach a minimum for solution 1	Solution 2
9	If none of the above conditions are satisfied	Solution 2

time steps, it becomes a priority to the gateway, and a decision is then made to override the optimization result and go to the center of that vehicle's zone in an attempt to re-establish a link with that vehicle. The gateway algorithm controls the gateway vehicle by sending commands to a kinematics model of the vehicle through the MOOS database. The kinematics model sends out navigational information that includes its current latitude, longitude, and velocity once per second. The controller replies each second with a velocity, heading, and depth command. Once the controllers desired location is reached, it will send out a velocity and heading command of zero and wait for the next set of transmissions from the AUVs. This process is repeated at each time step and can be seen in Fig. 7.

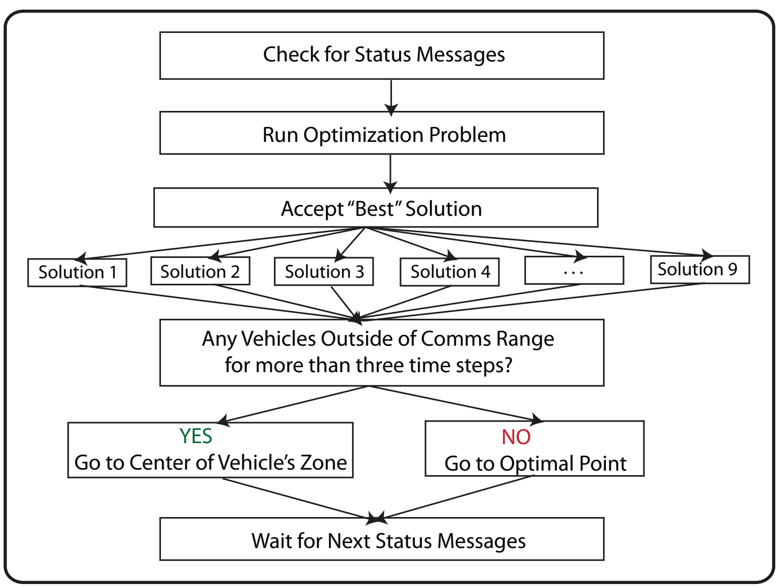

Figure 7: Graphical representation of mobile gateway controller decision-making algorithm for each time step. The controller makes a number of decisions based on the information available through other vehicles in the network.

IV. Simulation Results

Simulations were carried out such that the locations of the underwater agents were varied. One simulation scenario placed all underwater agents in a *close configuration*, in which all vehicles are always within communications range of adjacent vehicles, and therefore the mobile gateway's main role is to help propagate the information in the network more quickly. One of the vehicles ceased communications for an extended period of time after six time intervals elapsed in this simulation, allowing for the gateway vehicle to test out its "polling" behavior. Another simulation scenario included all agents in a *spread configuration* in which all underwater agents were placed such that communication was not always possible between adjacent vehicles. In this case the mobile gateway's main task is to attempt to maintain connectivity in the network, linking as many adjacent agents as possible.

For either scenario, a maximum of six underwater agents were included in the network. A maximum number of connections $C_{\max}$ for n agents is given by

$$C_{\max} = n(n-1) \tag{20}$$

and we see $C_{\max} = 15$ for $n = 6$, and $C_{\max} = 21$ for $n = 7$. We first run each simulation with six underwater agents and no gateway ($n = 6$). We then add the mobile gateway to the simulation, making the total number of agents $n = 7$. Both connectivity and propagation numbers are normalized by $C_{\max}$ for each simulation. Table 6 gives the results of each simulation. We see from these results that the mobile gateway gives improvements to the cohesiveness of the network in all test cases.

Table 6: Connectivity and propagation numbers for four simulation runs. A close and a spread out configuration was run each with and without the addition of the mobile gateway. We see an increase of connectivity with the addition of the gateway in all four test cases.

	$\%\mathbf{N_C}$			$\%\mathbf{N_P}$		
	No Gateway	**Gateway**	**Improvement**	**No Gateway**	**Gateway**	**Improvement**
Close Configuration	50.9	55.1	4.2	69.5	75.4	5.9
Spread Configuration	45.6	50.8	5.2	71.5	84.3	12.7

The connectivity and propagation numbers, N_C and N_P, are plotted in fig. 8 for the simulation in which the vehicles were kept in a *close configuration*. It can be seen that the connectivity and propagation numbers are increased with the addition of the gateway vehicle. However, the connectivity number is periodically decreased with the addition of the gateway vehicle at interval seventeen. This period is seen as the dotted part of the gateway result line in fig. 8. This reduction connectivity is due to the fact that after one of the underwater vehicles ceases communications for an extended period of time, the mobile gateway vehicle will trade-off overall network connectivity in order to attempt to regain communications with the lost vehicle.

Fig. 9 shows the connectivity and propagation numbers, N_C and N_P, for the simulation case in which the vehicles were kept in a *spread configuration*. It can be seen that the connectivity and propagation numbers are greatly improved in this case. It is interesting to note that no improvements are seen in the beginning of the simulation due to the fact that the mobile gateway was initialized in a position that required a few time intervals for it to reach its optimal point. This decreased connectivity period is shown as the dotted part of the gateway result line in fig. 9. Once the optimal point is reached the improvements become obvious.

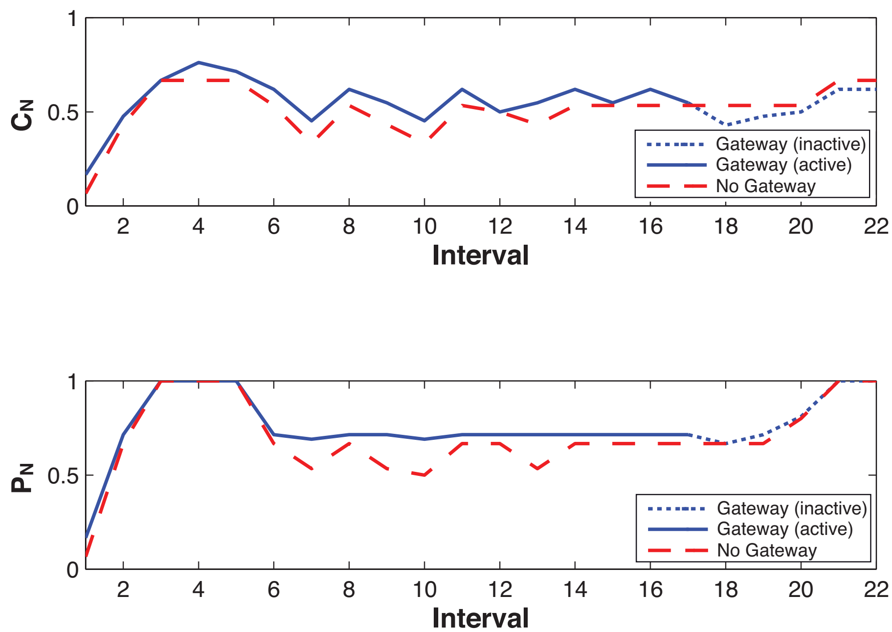

Figure 8: Connectivity and propagation numbers for a close configuration simulation. It can be seen that the addition of the mobile gateway improves both connectivity and propagation until a decision is made at interval seventeen to trade-off overall network connectivity in order to reconnect a lost vehicle. This period of decreased network connectivity can be seen as the dotted part of the gateway results line.

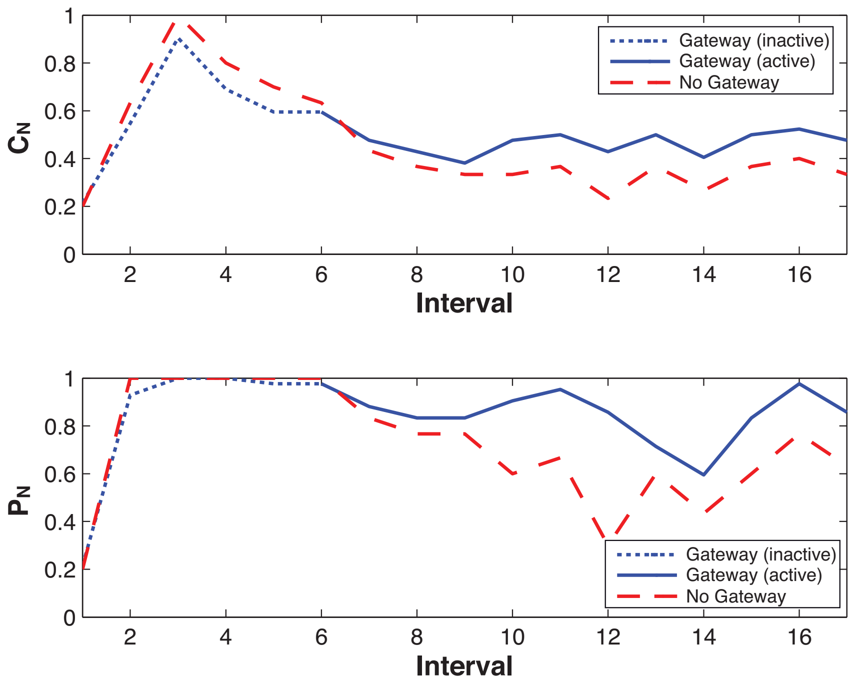

Figure 9: Connectivity and propagation numbers for a spread configuration simulation. It can be seen that the addition of the mobile gateway improves both connectivity and propagation numbers. Note that no improvements are seen in the beginning of the simulation due to the fact that the mobile gateway was initialized in a position that required a few time intervals for it to reach its optimal point. This initial period of decreased connectivity can be seen as the dotted part of the gateway results line.

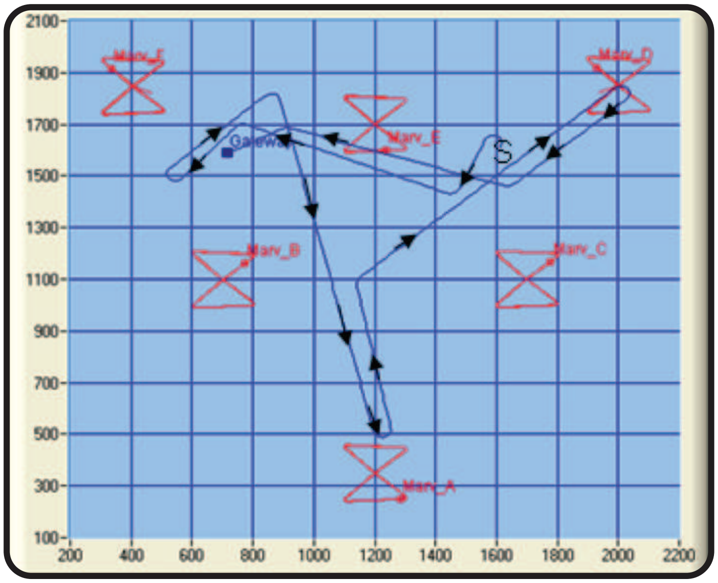

Figure 10: Snapshot of mobile gateway's path during simulation. Each underwater agent is shown performing figure-eight patrols. The mobile gateway attempts to maintain a centralized position to relay information to each agent. As any one agent is not heard from after a period of three full intervals the mobile gateway will move that agent's zone to regain communications, as can be seen with agent MarvA.

The evolution of the gateway's path through one simulation run can be seen in fig. 10. The mobile gateway attempts to maintain a centralized position throughout most of the simulation in order to propagate information more efficiently. As one of the agents (MarvA) stays out of communications range for longer than three time intervals the gateway is seen to move down to attempt to include that agent back into the network. It can be seen from these results that the mobile gateway vehicle concept is most useful when a network of underwater vehicles is *spread apart* and dynamically reconfiguring itself such that it requires connectivity assistance. Results for a network which is in *close configuration* seem promising, though the addition of the mobile gateway can be redundant since the network is already well connected from the start.

V. Conclusions

We have seen that the addition of an autonomous mobile gateway can improve the connectivity of a network of underwater vehicles while still leaving each vehicle free to perform its mission task. Results of running the simulations showed an improvement in the connectivity number of 5 percent in average and in the propagation number of 12 percent in average throughout an entire simulation. Note, however, that these average numbers can be skewed due to initial vehicle positioning and initialization at the start of the simulation. Actual improvements of up to 20 and 55 percent were seen in the connectivity and propagation numbers, respectively, for one interval in the spread apart simulation scenario. We have also shown in this paper the use of the Gauss Pseudospectral Method for optimal path planning in a real-time underwater scenario.

We can therefore conclude that using a gateway vehicle to improve network connectivity will best serve its purpose in a dynamically reconfigurable network of underwater vehicles, such that it can ensure that no vehicles will be left out of connectivity for extended periods of time. In a case where the network is already connected on a known mission path, the use of a gateway vehicle may prove to be redundant and costly, as an extra vehicle must be added to the network for its implementation. However in the case where network connectivity is unreliable or even nonexistent, the increase in connectivity provided by the gateway vehicle will far outweigh the cost of adding an extra vehicle, and allow a network to perform it mission.

Acknowledgments

The authors gratefully acknowledge support for this research from the U.S. Office of Naval Research Under Grant N00014-08-1-1173.

References

[1]Zavlanos, M. M. and Pappas, G. J., "Distributed Connectivity Control of Mobile Networks," *IEEE Transactions on Robotics*, Vol. 24, No. 6, December 2008, pp. 1416–1428.

[2]Zavlanos, M. M. and Pappas, G. J., "Potential Fields for Maintaining Connectivity of Mobile Networks," *IEEE Transactions on Robotics*, Vol. 23, No. 4, August 2007, pp. 812–816.

[3]Sweeney, J., Brunette, T., Yang, Y., and Grupen, R., "Coordinated Teams of Reactive Mobile Platforms," *Proceedings of the International Conference on Robotics and Automation*, Washington, DC, 2002.

[4]Thibodeau, B., Fagg, A., and Levine, B., "Signal Strength Coordination for Cooperative Mapping," University of Massachusetts Department of Computer Science, Amherst, MA, 2004.

[5]Wagner, A. and Arkin, R., "Multi-Robot Communication-Sensitive Reconnaissance," *Proceedings of IEEE International Conference on Robotics and Automation*, New Orleans LA, June 2004.

[6]Hsieh, M. A., Cowley, A., Kumar, V., and Taylor, C., "Maintaining Network Connectivity and Performance in Robot Teams," *Journal of Field Robotics*, Vol. 25, No. 1, January 2008, pp. 111–131.

[7]Benson, D. A., *A Gauss Pseudospectral Transcription for Optimal Control*, Ph.D. thesis, MIT, 2004.

[8]Benson, D. A., Huntington, G. T., Thorvaldsen, T. P., and Rao, A. V., "Direct Trajectory Optimization and Costate Estimation via an Orthogonal Collocation Method," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 6, November-December 2006, pp. 1435–1440.

[9]Huntington, G. T., *Advancement and Analysis of a Gauss Pseudospectral Transcription for Optimal Control*, Ph.D. thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 2007.

[10]Rao, A. V., Benson, D. A., Darby, C. L., Patterson, M. A., Francolin, C., Sanders, I., and Huntington, G. T., "Algorithm: GPOPS, A MATLAB Software for Solving Multiple-Phase Optimal Control Problems Using The Gauss Pseudospectral Method," *ACM Transactions on Mathematical Software*, Vol. 37, No. 2, April-June 2010.

[11]Carroll, K., McClaran, S., Nelson, E., Barnett, D., Friesen, D., and Williams, G., "AUV path planning: an A* approach to path planning with consideration of variable vehicle speeds and multiple overlapping time-dependent exclusion zones," *IEEE Proceedings of the 1992 Symposium on AUV technology*, Autonomous Underwater Vehicle Technology, Washington, DC, June 1992.

[12]Petres, C., Pailhas, Y., Patron, P., Petillot, Y., Evans, J., and Lane, D., "Path Planning for Autonomous Underwater Vehicles," Vol. 23, *IEEE Transactions on Robotics*, April 2007, pp. 331–341.

[13]Ferguson, D. and Stentz, A., "Using Interpolation to Improve Path Planning: The Field D* Algorithm," *Journal of Field Robotics*, Vol. 23, No. 2, February 2006, pp. 79–101.

[14]Petillot, Y., Ruiz, I., and Lane, D., "Underwater Vehicle Obstacle Avoidance and Path Planning Using a Multi-Beam Forward Looking Sonar," *IEEE Journal of Oceanic Engineering*, Vol. 26, No. 2, April 2001, pp. 240–251.

[15]Wang, H., Zhao, J., Bian, X., and Shi, X., "An Improved Path Planner based on Adaptive Genetic Algorithm for Autonomous Underwater Vehicle," *Proceedings of the IEEE International Conference on Mechatronics and Automation* , Niagara Falls, Canada, July 2005.

[16]Wang, H., Zhao, J., Bian, X., Ding, F., and Xia, G., "A GA Path Planner Based on Domain Knowledge for AUV," *Proceedings of the IEEE Techno-Ocean*, November 2004.

[17]Benjamin, M. R., Newman, P. M., Schmidt, H., and Leonard, J. J., *An Overview of MOOS-IvP and a Brief Users Guide to the IvP Helm Autonomy Software*, June 2009.

[18]Eickstead, D. P., Benjamin, M. R., Wang, D., and Schmidt, H., "Behavior Based Adaptive Control for Au-

tonomous Oceanographic Sampling," *Proceedings of the IEEE International Conference on Robotics and Automation*, April 2007, pp. 4245–4250.

[19]Eickstead, D. P., Benjamin, M. R., Schmidt, H., and Leonard, J. J., "Adaptive Control of Heterogeneous Marine Sensor Platforms in an Autonomous Sensor Network," *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, October 2006, pp. 5514–5521.

[20]Eickstead, D. P. and Sideleau, S. R., "The Backseat Control Architecture for Autonomous Robotic Vehicles: A Case Study with the Iver2 AUV," *Proceedings of IEEE/MTS Oceans'09 Conference*, Biloxi, MS, October 2009.

[21]Schmidt, H., Balasuriya, A., and Schneider, T., *MOOS-IvP Autonomous Command and Control for Undersea Network Nodes User's Guide*, December 2008.

[22]Baumgartner, K. A. C., Ferrari, S., and Rao, A. V., "Optimal Control of an Underwater Sensor Network for Cooperative Target Tracking," *IEEE Journal of Oceanic Engineering*, Vol. 34, No. 4, October 2009, pp. 678–697.

[23]Gill, P. E., Murray, W., and Saunders, M. A., *User's Guide for SNOPT Version 7: Software for Large Scale Nonlinear Programming*, February 2006.