Mobility-Aware Ad Hoc Routing Protocols for Networking Mobile Robot Teams

Saumitra M. Das, *Member, IEEE,* Y. Charlie Hu, *Member, IEEE,*
C. S. George Lee, *Fellow, IEEE,* and Yung-Hsiang Lu, *Member, IEEE*

Abstract

Mobile multi-robot teams are useful in many critical applications such as search and rescue. Explicit communication among robots in such mobile multi-robot teams is useful for the coordination of such teams as well as exchanging data. Since many applications for mobile robots involve scenarios in which communication infrastructure may be damaged or unavailable, mobile robot teams frequently need to communicate with each other via ad hoc networking. In such scenarios, low-overhead and energy-efficient routing protocols for delivering messages among robots are a key requirement. Two important primitives for communication are essential for enabling a wide variety of mobile robot applications. First, unicast communication (between two robots) needs to be provided to enable coordination and data exchange. Second, in many applications, group communication is required for flexible control, organization, and management of the mobile robots. Multicast provides a bandwidth-efficient communication method between a source and a group of robots.

In this paper, we first propose and evaluate two *unicast* routing protocols tailored for use in ad hoc networks formed by mobile multi-robot teams: Mobile Robot Distance Vector (MRDV) and Mobile Robot Source Routing (MRSR). Both protocols exploit the unique mobility characteristics of mobile robot networks to perform efficient routing. Our simulation study show that both MRDV and MRSR incur lower overhead while operating in mobile robot networks when compared to traditional mobile ad hoc network routing protocols such as DSR and AODV.

We then propose and evaluate an efficient *multicast* protocol Mobile Robot Mesh Multicast (MRMM) for deployment in mobile robot networks. MRMM exploits the fact that mobile robots know what velocity they are instructed to move at and for what distance in building a long lifetime sparse mesh for group communication that is more efficient. Our results show that MRMM provides an efficient group communication mechanism that can potentially be used in many mobile robot application scenarios.

Index Terms

Mobile robotics, Wireless communication, Protocol architecture

S. M. Das, Y. C. Hu, C. S. G. Lee and Y-H. Lu are with School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907. Email: {smdas, ychu, csglee, yunglu}@purdue.edu.

I. Introduction

Communication between mobile robots is useful and even critical in many applications. In applications such as search and rescue, communication infrastructure may be damaged or not present requiring the mobile robots to form an ad hoc network using each other as forwarding nodes to enable communication. Message routing among the mobile robots or from the robots to a human operator thus requires routing protocols that can operate without central control and handle dynamic topology changes due to the mobility of the mobile robots. Messaging requirements for mobile robot applications can be categorized into two major primitives: (1) Unicast messages sent from a mobile robot to another are required by many applications for sending data and images, asking for assistance, etc.; (2) Multicast messages sent from a human operator or one mobile robot to a group of receivers are typically used for coordination and control.

Unicast messaging is useful for coordination-oriented communication. Coordination-oriented communication is defined to be the transmission related to the coordination and control of robot teams and has been extensively studied in the literature. In particular, the use of communication for robot control was studied in [1]. It has been found that communication provides certain performance improvement for particular tasks [2].

Group communication among the mobile robots requires protocols that can operate without central control and handle dynamic topology changes due to the mobility of the mobile robots. Multicast is the most important group communication primitive and is critical in applications where close collaboration of teams (e.g. rescue teams, search teams) are needed. It is very useful when audio, video, images, and other such data need to be shared among team members. Multicast provides an efficient means of sending the same data to multiple recipients. Compared to multiple unicasts, multicast minimizes the link bandwidth consumption, sender and router processing, and delivery delay. Since multicast involves minimizing the number of transmissions in order to reach a group of receivers, it is effective in minimizing the energy consumption of group communication in a network of mobile robot teams. Multicast protocols can also be leveraged to provide reverse multicast or many-to-one communication which is very important when mobile robots are used as mobile sensor networks [3].

The specific use of multicast as a group communication primitive in mobile robot networks can be envisioned in the following three scenarios: (1) Multicast messages in mobile robot networks can be from one mobile robot to a set of other mobile robots. For example, in a large mobile robot team, only some subset of the nodes may be equipped with GPS or other localization equipment. These nodes can be organized into a multicast group so as to periodically notify each other of their positions and act as landmarks and navigation guides for other robots that do not know their positions. Multicast groups created based on capabilities can also be used by a robot to search for other robots with a specific capability or resource. Thus, all robots with similar capabilities can organize themselves into a multicast group to enable efficient cooperation and coordination. (2) Multicast messages may also need to be sent from human operators to a group of mobile robots in order to efficiently manage them. For example, the human operator may want to communicate with and command only the group of robots that have a specific chemical sensor. As another example, the human operator may want to organize the robots whose remaining energy is above a certain threshold into a *high availability* group in order to manage them. Thus, a team of robots can be partitioned into subteams using multicast groups for ease and flexibility of management and control. These subteams can also be grouped together, split apart, or hierarchically organized depending on the scenario and application. (3) Multicast messages may also be sent from a mobile robot to multiple controllers or sinks similarly as in a sensor network. For example, a team of demining robots may need to communicate its findings to a group of platoon commanders outside the minefield. Thus multicast can also be used for data dissemination.

Although traditional unicast and multicast routing protocols developed for the Internet cannot be used in mobile robot applications due to the presence of dynamic topology, many unicast and multicast routing protocols proposed for mobile ad hoc networks (MANETs) can be readily applied in such scenarios. However, protocols proposed for MANETs do not take into account two unique characteristics of mobile robot applications: (1) Mobile robots have significantly planned movement patterns unlike random human movement which is used as a mobility model (i.e., the random waypoint model [4]) in the vast majority of MANET protocol studies. For example, mobile robots typically may know the velocity and direction in which they are traveling, how far they have been given a command to travel and how far they have traveled from a previous checkpoint. (2) Mobile robots frequently pause at various points and often the duration of the pause can be estimated or is known beforehand. For example, if a robot pauses to measure some environmental parameter, the amount of time taken for the task can typically be estimated or known in advance.

In this paper, we propose to exploit the mobility characteristics of mobile robot applications to improve communication protocols for mobile robot teams. The key idea is that while we do not know or try to predict the mobility pattern of the robot (since it depends on the application and circumstances such as sensor reading and environment), at any given point in time, we can query the robot control software to decipher what it has been told to do by the application and also use its odometry devices and sensors to find information such as the current velocity, distance traveled from the last checkpoint and the current task's average duration. Such mobility information can then be used to optimize the routing protocols. For example, as a robot performs a search and rescue mission, its path from an outside perspective may look zig-zag and entirely unpredictable since the movement will be dynamically adjusted by the robot application software. However, the movement is really composed of a series of commands given to the robot by the controlling application, and the routing layer can extract mobility information by snooping the commands given by the application, and probing the robot odometry and sensors. In other words, we propose to exploit the fact that robots move "on command" and not randomly like humans and adapt the routing protocols to take advantage of such mobility information of the robot team in order to improve routing performance.

The rest of the paper is as follows. We first give a taxonomy of typical communication patterns in a team of mobile robots in Section II. We then give a formulation of the efficient communication problem in a team of mobile robots in Section III. We then propose, design, and evaluate unicast and multicast routing protocols for efficient performance in mobile robot applications. First, in Section IV, we propose two unicast routing protocols, Mobile Robot Source Routing (*MRSR*) and Mobile Robot Distance Vector (*MRDV*), for mobile robot networks based on the DSR [5] and AODV [6] protocols developed for mobile ad hoc networks. We then propose Mobile Robot Mesh Multicast (*MRMM*), an efficient multicast protocol for mobile robot teams based on the ODMRP [7] protocol designed for MANETs in Section V. All three protocols exploit the fact that mobile robots know what velocity they are instructed to move at and for what distance in order to build a long-lifetime sparse mesh for group communication. Finally, we discuss related work in Section VI and conclude the paper in Section VII.

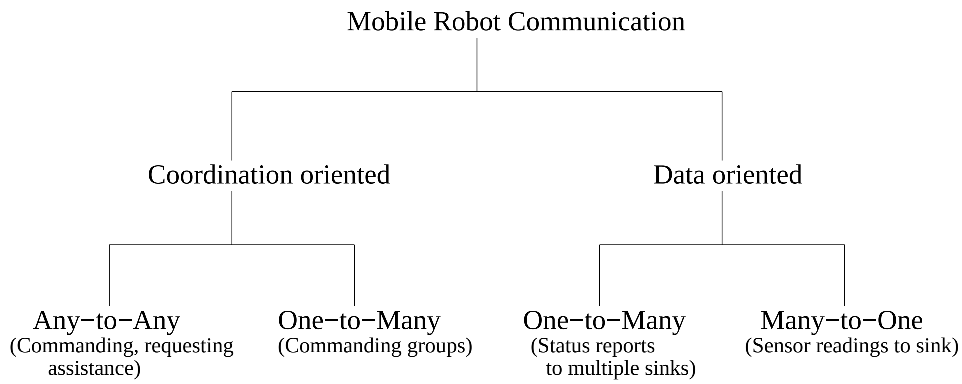


Fig. 1. A taxonomy of communication requirements in mobile robot networks and example applications.

II. Taxonomy of Communication Requirements in Mobile Robots

In this section, we provide a taxonomy of communication requirements for networks formed by mobile robot teams. A mobile robot network is formed when a team of mobile robots are deployed for accomplishing certain missions. A taxonomy of communication requirements for mobile robots is depicted in Figure 1. As shown in Figure 1, we can classify communication in mobile robot networks system to be of two types: *Coordination-oriented communication* and *Data-oriented communication*.

A. Coordination-oriented communication

This is defined as communication that takes place between robots in the team. Coordination-oriented communication can be both implicit (e.g. through the environment, based on peer robot behavior) or explicit (e.g. using directed/unicast or broadcast intentional messaging). Coordination-oriented communication is transmission related to the coordination and control of robot teams. Both multicast and unicast can be useful for coordination-oriented communication. For example one robot may need to send a message to another robot to command it to move and help in searching a particular area. Additionally, one robot can communicate with a group of robots (one-to-many) that have GPS capabilities to send it beacons so it can better localize itself using RF localization. Thus, coordination-oriented communication can require both unicast (any-to-any) and multicast (one-to-many) communication.

B. Data-oriented communication

This is defined as communication that takes place between members of the robot team and the operator or *sink* (data collection point in sensor networks). Note that there may be multiple operators or sinks in the network. Data-oriented communication is the transmission of data information collected by the robots as well as communication to query such data that the robots may store. This communication occurs based on the application parameters and can be independent of the coordination-oriented communication that may occur for coordinating the team members. In a mobile robot network, we anticipate two sources of data-oriented communication. We expect that either the robot's transducer itself senses application events (e.g. finding a certain substance) at random instants of time. These events are of timely interest and need to be reported to the operator right away. Additionally, we expect a periodic communication between the operator and each robot to determine various statistics and the current status of the robot. This periodic communication should occur even when no events are being sensed. These patterns exhibit a *many-to-one* (from all robot team members to the sink) as well as a *one-to-many* (from an operator to many robots or from one robot to many operators) pattern. Thus data-oriented communication mainly require multicast (one-to-many) and reverse multicast (many-to-one) communication.

In summary, mobile robot networks require protocols for efficient unicast as well as multicast in order to perform well in many application scenarios.

III. Problem Formulation

We focus on an application scenario where many robots are used to form a mobile robot network. Each robot has a simple sensory ability and limited computational power. This makes it practical to build a large number of such robots. The communication among the robots is based on wireless ad hoc communication (IEEE 802.11) in which robots forward each other's packets when the source and the destination nodes of a packet are not within direct reach. This reflects our mobile robot testbed in which mobile robots are equipped with laptops and PDAs and use Orinoco Wavelan wireless cards to communicate with each other.

Mobile robot networks have the following unique mobility characteristics different from traditional MANETs. Consider Figure 2. On the left side, a pausing robot is depicted. A robot may pause because it is awaiting further instructions from a human operator or is busy doing a task like detecting a gas. The pause duration (T) is thus composed of a time t to complete the current task followed by a random period of time $T - t$ after which the robot moves again. Typically, a small set of predetermined tasks (e.g. chemical analysis, taking a video or picture,

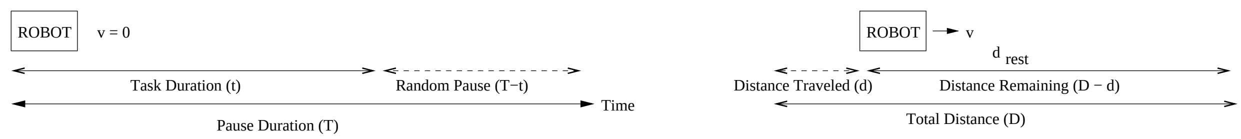


Fig. 2. Mobile robot model and assumptions.

analyzing air etc) can be performed by a robot and most such tasks take on average a known amount of time. We thus assume a set of k tasks with possibly different task durations that the robot can undertake. Given this information, a robot can calculate the time remaining for task completion t_{rem} by subtracting the amount of time elapsed since the task started from current task duration t. On the right side, a moving robot is depicted. In this case, most commands to the robot specify an amount of distance D to travel in a particular direction. Thus, even without GPS or localization support, the robot can at the very least measure with some error the distance d already traveled and thus the distance $D - d$ which we denote the *distance-to-rest* (d_{rest}).

Thus we assume that the robot can estimate its d_{rest} when mobile, and the time remaining to complete a task t_{rem} when static. In this paper, we do not try to characterize the random pause duration after task completion. Simple heuristics like the exponentially weighted moving average (EWMA) of past random pause times can potentially be used to further improve the performance of routing protocols. We also assume that the robot has some means of calculating its current velocity. This is a valid assumption and has already been implemented in our mobile robot testbed. Additionally, the velocity may in many cases be determined by an energy efficient path planning algorithm and thus known to the control software.

A. Practical Feasibility

In this section, we discuss the feasibility of enabling unicast and multicast protocols using the mobility information described above.

Our mobile robot testbed consists of five Pioneer 3DX [8] mobile robots and eight smaller Amigobots [8]. The control software on the mobile robots is enabled through a PDA or an ultra-light laptop mounted on top of the robot. The wireless communication is enabled by supplying each PDA or laptop with a Lucent Wavelan 802.11b PCMCIA wireless card. The wireless routing protocols can be implemented in the Linux operating system using either in-kernel implementation [9] or using some form of redirection and user-space processing [10]. The robot control software has a peer-to-peer communication architecture using TCP and UDP sockets to enable communication over the underlying unicast or multicast protocol. The robot control software has an API to set and obtain the current velocity of the robots as mentioned in the problem formulation. It also stores the distance moved using encoders based on the wheel movement. Thus we can get the value of d from the robot and consequently d_{rest}. Therefore, it is practically feasible to implement the protocols and the associated mobility information API in a real mobile robot testbed.

Given this application scenario, in the following sections we explore the design of protocols for efficient unicast and multicast in mobile robot networks.

IV. Efficient Unicast for Mobile Robots

In this section, we first explain the design of MRSR followed by that of MRDV. We then evaluate both protocols in a detailed wireless packet-level simulator.

A. Mobile Robot Source Routing (MRSR)

MRSR is based on Dynamic Source Routing (DSR) [5], a well known multi-hop routing protocol for MANETs which is based on the concept of source routing in contrast to hop-by-hop routing. MRSR incorporates three mechanisms, *route discovery*, *route construction*, and *route maintenance*.

1) Route Discovery: Route discovery is the process by which a source robot discovers a route to a destination robot for which it does not already have a route in its cache. The process broadcasts a ROUTE REQUEST packet that is flooded across the network in a controlled manner. In addition to the address of the original initiator of the request and the target of the request, each ROUTE REQUEST packet contains a route record, which records the sequence of hops taken by the ROUTE REQUEST packet as it propagates through the network. ROUTE REQUEST packets use sequence numbers to prevent duplication.

Robots *probabilistically rebroadcast* ROUTE REQUEST packets based on a probability value p_r. This mechanism is used to reduce routing overhead and energy consumption by limiting the propagation of ROUTE REQUEST packets. In addition, this mechanism ensures that highly mobile robots become part of a route less frequently so as to increase the route lifetime, as MRSR exploits the mobility knowledge present in the robots in calculating p_r. Once the ROUTE REQUEST is received by the robot, it calculates p_r and rebroadcasts the ROUTE REQUEST with a probability of p_r.

If the robot is currently stationary (i.e. v = 0), p_r is set to 1. Otherwise, p_r is calculated using a combination of the current velocity v and the distance weighting factor γ of the robot. Since we assume that the robot knows how much distance it has to travel and the distance it has already traveled, it can calculate its *distance to rest* d_{rest} by subtracting these quantities. Given a wireless transmission range of R, the distance weighting factor is given by:

$$\gamma = \begin{cases} \frac{d_{rest}}{R/2} & \text{if } d_{rest} \geq \ R/2 \\ 0 & \text{otherwise} \end{cases}$$

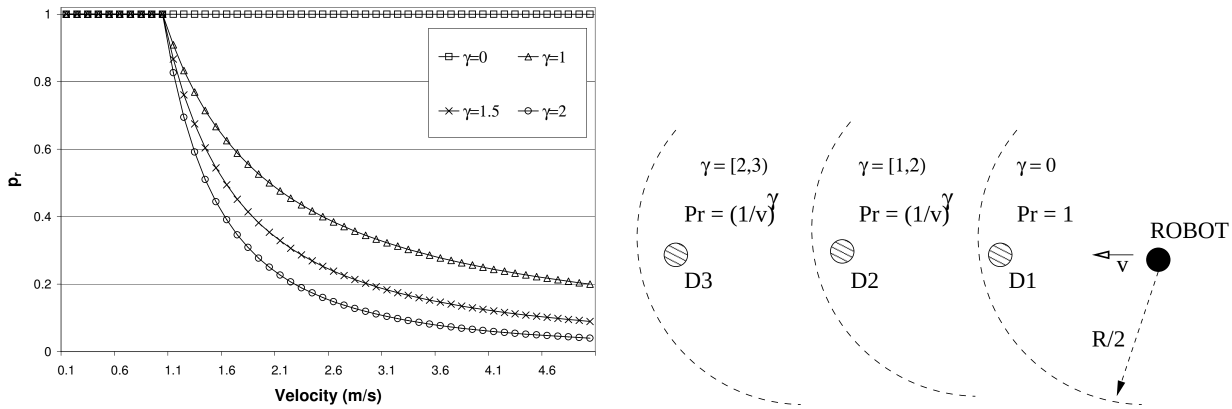
Finally, p_r is defined to be inversely proportional to the current velocity v of the robot, so that robots that are highly mobile are unlikely to rebroadcast a ROUTE REQUEST packet. In addition, γ is used to exponentially decay the value of p_r. This is done because a higher value of γ indicates that the robot is more likely to move out of range and disrupt connectivity and hence the value of p_r should decrease quickly (exponentially) as γ is increased. On the other hand, if γ is zero, p_r is set to 1, since the robot will come to a pause without breaking the path. p_r is finally calculated as:

$$p_r = min(1, (\frac{1}{v})^{\gamma})$$

where velocity v is measured in m/s[1]. Thus, when the robot is traveling at a high velocity, γ takes into account wireless range and route validity to adjust the rebroadcast probability p_r. Figure 3(a) shows the sensitivity of p_r to the robot velocity for different values of γ. Figure 3(b) shows how p_r is calculated. D1, D2 and D3 denote different possible values of d_{rest} and the corresponding values of p_r calculated by the protocol are shown next to them. For example, if d_{rest}=D2, then $2 \cdot \frac{R}{2} \geq d_{rest} \geq \frac{R}{2}$ and correspondingly γ can take on a value in the interval [1,2). This γ value is used to find p_r depending on the exact position of D2.

Once the ROUTE REQUEST is received, it is answered by a ROUTE REPLY packet either from the destination node or an intermediate node that has a cached route to the destination. The ROUTE REPLY contains the reversed route record (source route) used by the ROUTE REQUEST which is used to reach back to the source. MRSR requires each robot encountered along the ROUTE REPLY path to encode its mobility information into the ROUTE REPLY packet. This information consists of an estimated timeout value which indicates when the robot will move out of the range of the route being set up and a 1-bit flag to indicate if the robot is moving or

[1]Velocity v typically ranges from 0 - 5 m/s in most mobile robot applications.



(a) p_r sensitivity. (b) Graphical depiction

Fig. 3. Probabilistic stationarity based route discovery.

paused (M/P). If the robot is paused, it estimates its remaining task completion time (t_{rem}) and inserts that into the ROUTE REPLY as the timeout[2]. If the robot is moving, it encodes the timeout as the time taken to move out of the radio range based on the estimate of its current speed. This timeout is calculated assuming that the robot is on average $\frac{R}{2}$ away from the previous node on the route. If the robot estimates it will stop before moving out of the radio range, the timeout value is set to a high value (e.g. 25 seconds), since this means that route will be maintained by this robot for a longer period of time. This mobility information (i.e. estimated timeouts and flag assigned by each robot) is learned by each node that forwards the ROUTE REPLY back. This timeout value associated with a node is used to efficiently manage and utilize routes before they break as explained in the next section.

2) Route Construction: Similar to DSR, MRSR also uses aggressive caching to reduce the frequency and propagation of route discoveries. MRSR uses a graph cache similar in structure to that proposed in [11]. However, the algorithm used to manage the graph cache is modified for use in mobile robot networks. MRSR stores individual links of routes to build a topological graph of the network. Specifically, each mobile robot in the network maintains a *graph cache*, i.e., a topological view of the network based on discovered and forwarded link information. A single-source shortest path algorithm such as Dijkstra's algorithm can then be used to compute source routes to other mobile robots to whom paths exist in the graph.

MRSR expires links based on timeouts learned from ROUTE REPLY packets. The timeout of a link $l_{i,j}$ in the graph is calculated as the minimum of the timeout values learned for the endpoint

[2]Since we do not have the information about the random pause duration following the task completion ($T - t$), our scheme conservatively assumes that the robot will move after the task is complete in estimating this timeout.

nodes i and j of that link. If the timeout of a link $l_{i,j}$ is found to be 0, it is assumed to be broken. When a link $l_{i,j}$ breaks as indicated by the reception of a ROUTE ERROR packet, the link is immediately removed. Since there can be another link $l_{i,k}$ which may still be valid although $l_{i,j}$ is broken, instead of removing all links containing node i, MRSR marks only the link $l_{i,j}$ as broken and multiplicatively reduces the timeout associated with endpoints of the broken link, i.e., nodes i and j. This achieves the desired effect: the link in error is removed from the graph, and all other links containing either endpoint of the link in error have reduced timeouts reflecting the recent mobility of those endpoints.

When a route needs to be found to a destination node, MRSR first tries to use Dijkstra's algorithm to construct a route using only paused robot nodes in the graph. If such a route cannot be constructed, it tries to find a path with the longest lifetime based on the mobility information provided by the other robots. Specifically, the route that has the highest minimum timeout among all of its links is chosen. If there are multiple such routes, the shortest one is chosen.

3) Route Maintenance: Route maintenance is similar to DSR and consists of monitoring the operation of the route and informing the sender of any routing errors. If a route breaks due to a link failure, the detecting host sends a ROUTE ERROR packet to the source which upon receiving it, removes all routes in the host's cache that use the hop in error.

Note that in some situations, e.g., if the mobility is very high on all nodes along the path between source and destination, it is possible that a route request is not rebroadcasted and fails to reach the destination. In this case, on expiry of the route request timer (included in the original protocols), the source will send out a basic route request (indicated by a flag set in the packet) that is rebroadcasted with probability 1 by all nodes that receive the route request, to ensure connectivity. In our simulations, such route requests were not seen since all route discoveries were successful through some path.

B. Mobile Robot Distance Vector (MRDV)

MRDV is based on the well known AODV [6] routing protocol. MRDV shares on-demand behavior with MRSR/DSR and the use of hop-by-hop routing and destination-based sequence numbers with another ad hoc routing protocol DSDV [12].

1) Route Discovery: Route discovery is performed via a process similar to MRSR. However, MRDV stores routing information as one entry per destination in contrast to MRSR which caches multiple entries per destination. Like MRSR, a node satisfies the ROUTE REQUEST by sending a ROUTE REPLY back to the source or by increasing the hop count and re-broadcasting to its neighbors. As the ROUTE REQUEST propagates from the source to various nodes, a reverse path is set up from these nodes back to the source. When the ROUTE REPLY travels backward to the source using the set up reverse path, a forward path is also set up towards the destination.

MRDV also probabilistically rebroadcasts ROUTE REQUEST packets. Each node along the ROUTE REQUEST sets a timeout value to indicate how long it is likely to remain as part of the route (i.e. before breaking the route). An estimated timeout with respect to the route being established is calculated similarly as in MRSR, and the timeout is inserted into the ROUTE REQUEST packet. Note that unlike MRSR, no source routing is done so only one timeout value can be inserted into the packet. However, since the weakest link in the route (i.e. the robot with the lowest timeout) decides the time of link breakage, one worst case value is sufficient to accurately characterize the lifetime of the route. Each subsequent node on forwarding the ROUTE REPLY checks the timeout set by the previous node. If the forwarding node's timeout is lower (it is more mobile than all preceding nodes on the route), it modifies the timeout encoded in the packet to its own timeout value and forwards the packet. If its timeout value is higher, the forwarding node just rebroadcasts the packet. When the ROUTE REQUEST finally reaches the destination, the final timeout value is encoded in the resulting ROUTE REPLY back to the source. The timeout encoded in the ROUTE REPLY is used to set the timeout of the route in the routing table at the source. In addition, intermediate nodes also use the current timeout in forwarded ROUTE REQUEST packets and the final encoded timeout in ROUTE REPLY packets to set timeouts for routes to source and destination nodes respectively in their routing tables.

2) Route Maintenance: Similar to AODV, MRDV uses timers to expire routes that have not been used recently. However, in MRDV, these timeouts are set using the encoded value in the ROUTE REPLY unlike the statically assigned values used by AODV. When a ROUTE REPLY sets up a forward path to a destination node, each node along the way assigns the route a timeout based on the timeout value returned in the ROUTE REPLY packet.

We note that unlike in MRSR, only one route will be discovered in MRDV. This route may not have the longest lifetime out of all the possible routes. Instead, MRDV avoids route error by using the discovered lifetime for this route to expire it at the appropriate time.

The destination sequence numbers in control packets ensure loop freedom and freshness of routing information. MRDV ensures wider propagation of ROUTE ERRORS by using a per destination predecessor list at each node, similarly as in AODV. The MRDV version in this study uses link layer feedback for detection of broken links similar to AODV-LL [4].

Similar to MRSR, if a route discovery fails, a normal route discovery (without probabilistic rebroadcast) is sent out to ensure connectivity.

Note that we only consider modifications to reactive routing protocols (DSR and AODV). The reason we concentrate on reactive protocols is because robot networks are resource constrained and may communicate only occasionally in the duration of performing their tasks. In such scenarios, using proactive protocols like DSDV [12] and OLSR [13] which continuously exchange routing table updates to maintain routes may lead to high energy drain. Reactive protocols avoid the need to actively maintain routes until they are necessary to transfer data.

C. Evaluation Methodology

We use the Glomosim simulator [14] to evaluate the performance of MRDV and MRSR. We implemented both protocols in Glomosim and compare them to the existing implementations of DSR and AODV.

Network Model: We simulate 50 robots in an area of 1500m x 300m similar to the studies in [4], [15] which evaluated the performance of DSR and AODV. We also study a scenario of 100 robots in an area of 2200m x 600m. The two-ray path loss model is used. No obstacles are modeled currently in the terrain. A radio model with a bit-rate of 2 Mbps and a transmission range of 250m is used. The effect of multiple access interference, capture, RF propagation, signal strengths, and propagation delays are modeled. The link layer used is based on the IEEE 802.11 DCF (Distributed Coordination Function). All the results are obtained by averaging over 25 random scenarios.

Mobility pattern: The robot behavior is modeled as a series of movements and pauses in a given area: each robot moves towards a particular destination location, *pauses* at the destination location performing a task and/or waiting for the next command from the application, and then moves to the next destination[3]. This effectively models the movement of a group of robots performing tasks. Recall from Section III that a pause duration (T) is composed of a time t

[3]This is similar to the random waypoint model used in the evaluation of mobile ad hoc networks. However we also model a task duration as a subset of each pause time

to complete the current task followed by a random period of time $T - t$ after which the robot moves again to the next destination. At each destination, the robot chooses a random task from a set of k tasks that have a task duration t such that $t \leq T$ (*pause time*) seconds. It then pauses for T seconds following which it chooses the next random destination and moves towards it. For each movement the robot chooses a speed uniformly between 1 and 5 meters/second. Note that the speed chosen is lower than those considered for MANET routing protocol comparison studies (typically 10-20 m/s) which assume faster participants (e.g. cars).

Although the mobility of robots in the simulation is random, the robots know their velocity, *current* task duration t, current remaining task completion time t_{rem} and d_{rest}. We have considered *pause times* ranging from 900s to 0s and a simulation duration of 900s. Note that the random task selection is not done for T=900s and 0s. A pause time of 900s in a simulation of duration 900s signifies that no robots move and no tasks are performed. A pause time of 0 second signifies that the robots are in constant motion and do not stop to perform any tasks.

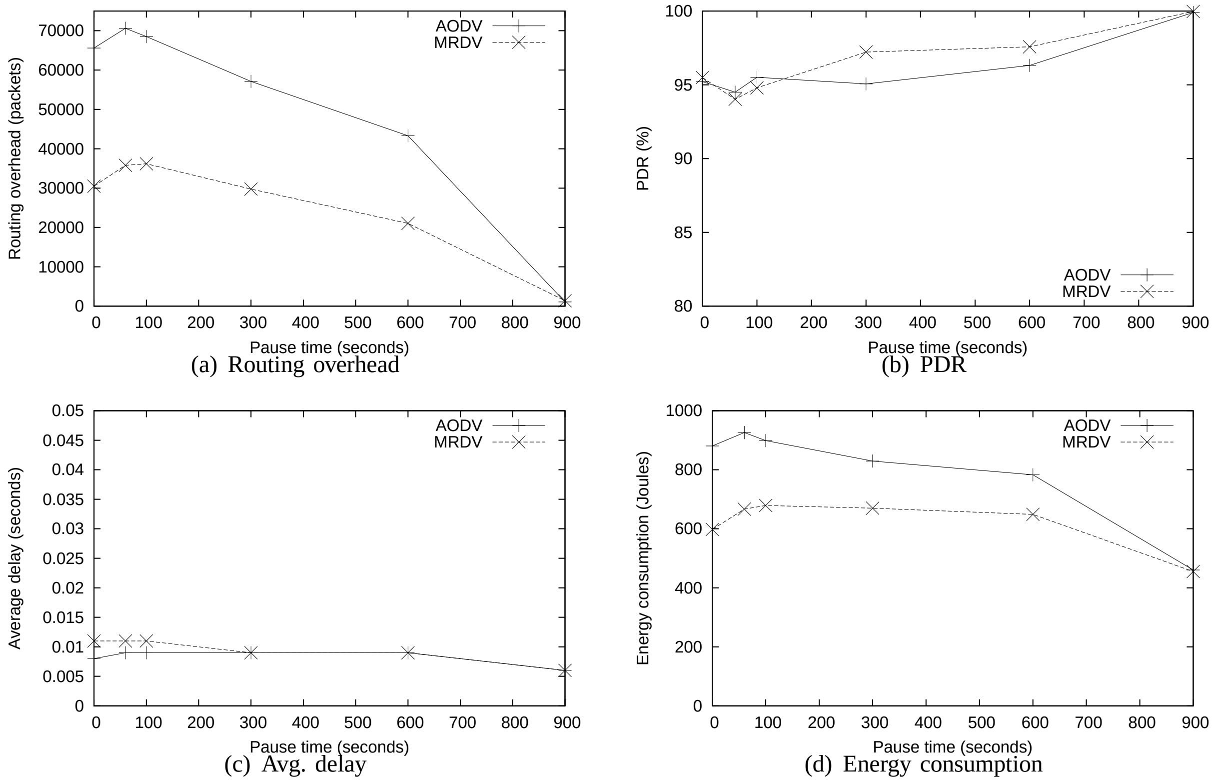
Traffic pattern: Each robot initiates data connections to one other randomly chosen member robot. Each packet sent is 64 bytes in size and the packet rate is 1 packet every second per connection. The connection is maintained throughout the simulation as the connection endpoints move.

Energy Model: Since our wireless communication is based on IEEE 802.11, we adopt the energy model and measurements for ad hoc networks proposed in [16].

Metrics: The following metrics are evaluated for the routing protocols: (1) Routing Overhead – The total number of control packets transmitted, with each hop-wise transmission counted as one transmission; (2) Packet Delivery Ratio – The ratio of the data received at the destination over the number of data packets transmitted by source robots; (3) Average Delay – The average delay incurred in sending a data packet; and (4) Energy Consumed – The amount of energy consumed by the robot team for communication. This includes energy spent during sending and receiving both data and control packets. We do not consider idle energy since it is not directly related to the operation of a particular protocol.

D. Evaluation Results

In this section, we first compare MRDV with AODV and MRSR with DSR using the four metrics discussed above for a network of 50 nodes with varying mobility.

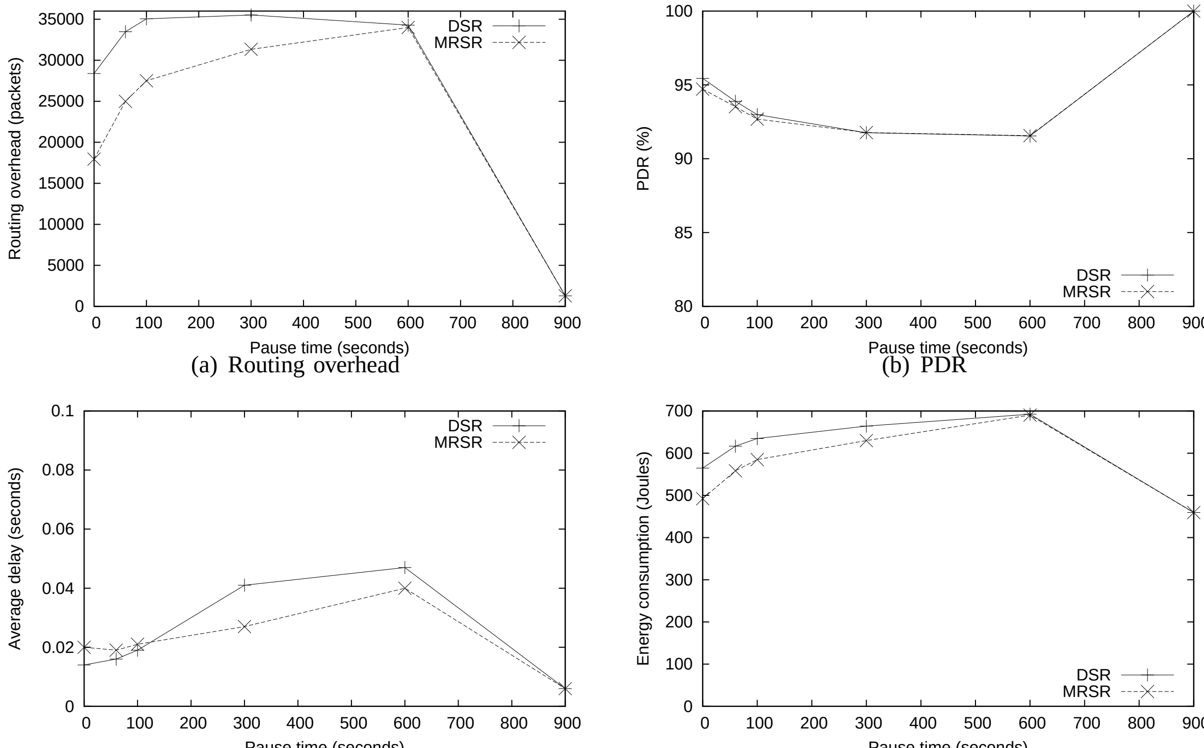


(a) Routing overhead (b) PDR (c) Avg. delay (d) Energy consumption

Fig. 4. Comparison of the distance vector routing protocols.

1) MRDV Performance: The routing protocol performance for MRDV and AODV are depicted in Figure 4. The metrics are measured by varying the pause time and thus the extent of mobility in the network. A higher pause time results in a network with lower mobility and a pause time of 900 seconds (same as the simulation duration) results in a completely static network.

The following observations can be made from the results. First, as Figure 4(a) shows, MRDV reduces the overhead due to control messages significantly when compared to AODV by exploiting the mobility characteristics of mobile robot networks. As expected, when the network is static, both MRDV and AODV perform similarly. In all cases in which the network is mobile, MRDV provides a 50% reduction in overhead due to control messages when compared to AODV. More importantly, this reduction is achieved without any loss in packet delivery performance for MRDV (Figure 4(b)). In fact, for medium mobility scenarios, MRDV has a slightly higher packet delivery ratio than AODV. A high PDR is desirable since successful coordination of mobile robots require reliable delivery of commands. In addition, the average delay depicted in Figure 4(c) is also similar for MRDV and AODV. Figure 4(d) shows the energy consumption due to communication for MRDV and AODV. Due to its lower control overhead, MRDV reduces the radio usage during the transmit mode as well as the receive mode. This reduction in radio



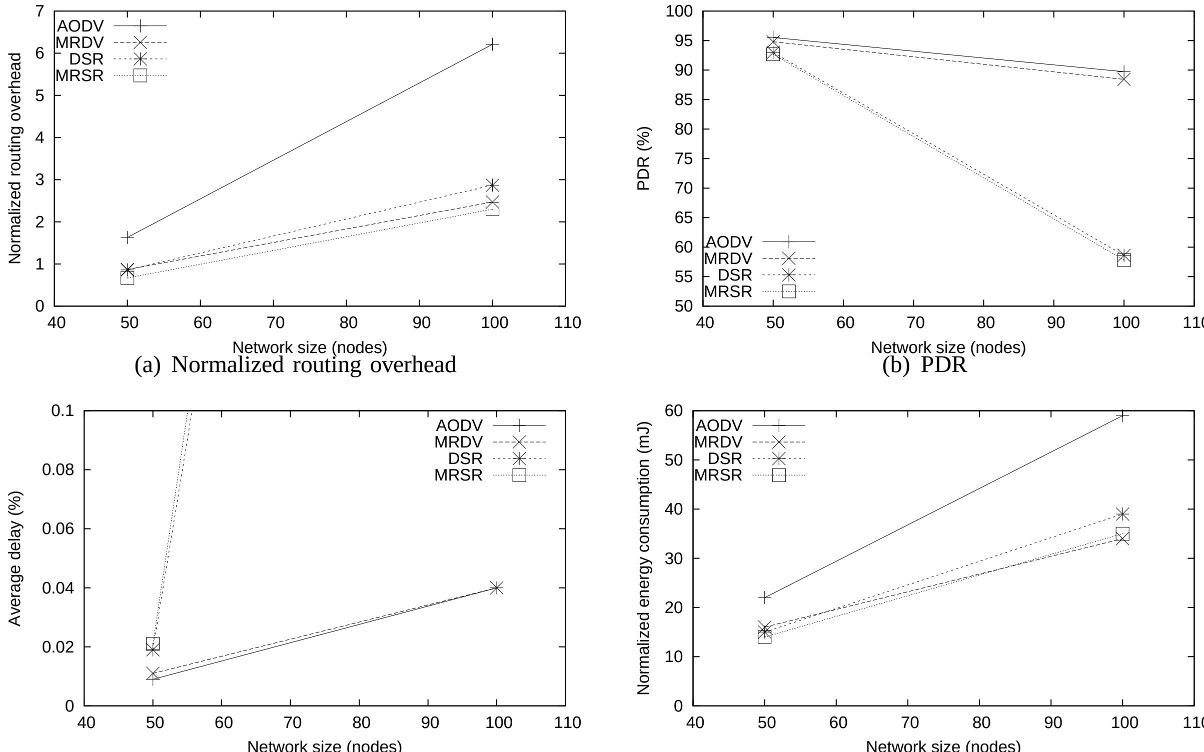
(a) Routing overhead (b) PDR (c) Avg. delay (d) Energy consumption

Fig. 5. Comparison of the source routing protocols.

usage translates to energy savings ranging from 30% for high mobility scenarios to 18% for medium to low mobility scenarios. As expected, for a static network, both protocols consume similar amount of energy.

2) MRSR Performance: The routing protocol performance for MRSR and DSR are depicted in Figure 5. Similar to the previous section, the metrics are measured by varying the pause time and thus the extent of mobility in the network.

The following observations can be made from the results. First, as Figure 5(a) shows, MRSR reduces the overhead due to control messages when compared to DSR. Note that DSR employs very aggressive caching and also uses a graph cache [11] and thus typically has low overhead. Despite that, MRSR results in a further reduction in overhead. In high mobility scenarios, MRSR provides a 30% reduction in overhead while for medium mobility scenarios the reduction is 12% when compared to DSR. As expected, when the network is static, both MRSR and DSR perform similarly. However, due to efficient caching, both protocols perform similarly for pause time 600s, unlike the comparison in the previous section. MRSR achieves this reduction while maintaining a PDR similar to that of DSR (Figure 5(b)). In addition, Figure 5(c) shows that MRSR has a lower average delay in many scenarios. Figure 5(d) shows the energy consumption due to



(a) Normalized routing overhead (b) PDR (c) Avg. delay (d) Normalized energy consumption

Fig. 6. Comparison among all protocols for varying network sizes. Pause time is 100 seconds modeling medium mobility.

communication for both MRSR and DSR. Due to its lower control overhead and increased route lifetimes, MRSR reduces the radio usage. The obtained energy savings range from 12% for high mobility scenarios to 5% for medium mobility scenarios. As expected, for a static network, both protocols consume similar amount of energy.

3) Comparison among the protocols:

a) Impact of Network Size: In this section, we compare all 4 protocols: AODV, MRDV, DSR and MRSR for two network sizes of 50 and 100 nodes. A medium mobility scenario with a pause time of 100 seconds is chosen for both network sizes. In this section, we show the normalized routing overhead and the normalized energy consumption in mJ (normalized by the number of successfully delivered data packets) in order to clearly present the results.

For smaller mobile robot networks such as 50 nodes, both MRDV and MRSR have similar performance. MRSR has a slightly lower overhead than MRDV due to the use of aggressive caching. However, MRSR also has a slightly higher delay than MRDV due to the use of larger packets for encoding source routes. Note that MRDV is as efficient in energy consumption and overhead as DSR and MRSR while AODV has a much higher overhead and energy consumption. However, for larger mobile robot networks such as 100 nodes, the distance vector protocols are superior in packet delivery performance (Figure 6(b)). This has been observed in earlier studies such as in [15] and been attributed to not using source routing which increases packet sizes and congestion as network size increases. In 50 nodes, MRDV provides a 47% reduction in overhead from AODV for this pause time. However, when the network size is increased to 100 nodes, Figure 6(a) shows that MRDV now provides a reduction of 60% in overhead while maintaining similar delays and PDR. Additionally, Figure 6(d) shows that the energy savings provided by MRDV increases from 27% to 42% when compared to AODV as the network size is increased.

In summary, for smaller mobile robot networks, both MRSR and MRDV are attractive options with MRSR providing higher energy savings at the cost of slightly increased delay. For larger mobile robot networks, MRDV provides a superior packet delivery performance with increased energy savings compared to AODV and MRSR.

We now discuss the impact of traffic load and network density on the performance of our protocols. Due to lack of space we do not depict these results but discuss our main findings.

b) Impact of traffic load: Our results have shown that both MRSR and MRDV provide comparable delivery performance to DSR and AODV respectively with significantly reduced overhead. However, as the traffic load was increased to 3 packets per second, MRSR and MRDV were found to have benefit even in packet delivery ratios. This is because at high traffic loads, our protocols performed better due to the lower routing overhead which left more capacity for routing data packets. For example, at 3 packets per second traffic load, the overhead of MRSR and MRDV still remained lower than DSR and AODV by approximately similar margins to the original simulations with 1 packet per second, while the PDR of MRSR and MRDV were now around 8-10% higher than the DSR and AODV, respectively. Overall the delivery ratios of all protocols reduced as expected due to high traffic loads.

c) Impact of network density: The network density used in our evaluation is similar to several previous work on DSR and AODV so that we have a common ground to compare results. If the network is made more sparse, there is a possibility of partitions. However, this is expected to affect both our protocols and previous ad hoc routing protocols in a similar manner, e.g., routes may break frequently and not be repaired for a long time due to disconnection. An interesting question is whether discovering routes in MRSR and MRDV is more prone to disconnection (due to probabilistic rebroadcast) if nodes on the path are highly mobile, especially in sparse networks. We performed simulations for this scenario by reducing the density to 30 nodes in an area of 1500x300m and found that the conclusions drawn are similar to for the denser network. Overall both MRSR/MRDV and DSR/AODV have larger numbers of route errors and lower packet delivery ratio. However gain from using our protocols still occurs. For example, at 0 second pause time, the routing overhead of MRSR is 25% lower than DSR, and that of MRDV is 40% lower than AODV. At the same time, the packet delivery ratios of MRSR and MRDV remained close to those DSR and AODV, respectively.

V. Efficient Multicast for Mobile Robots

In this section, we first explain the design of Mobile Robot Mesh Multicast. We then evaluate and compare the protocol with ODMRP in a detailed wireless packet-level simulator.

A. Mobile Robot Mesh Multicast

Mobile Robot Mesh Multicast (MRMM) is based on ODMRP (On Demand Multicast Routing Protocol) [7] developed for MANETs. MRMM is an extension of the ODMRP protocol with specific features for efficient operation in mobile robot applications. Like ODMRP, MRMM also uses a mesh to enable redundancy and consequently more reliable delivery as well as to avoid the drawbacks of tree maintenance in mobile networks. Similar to ODMRP, MRMM also consists of two major phases: (1) Mesh construction and maintenance: A mesh is created using a subset of the mobile robots that are a part of the network. The mesh is a structure in the network such that all group members are part of the mesh and certain number of non-members are recruited to forward packets so that no disconnections occur and some redundancy is present. This mesh has to be dynamically reconfigured in adapting to disconnections due to mobility. (2) Data delivery: Data packets are broadcast by robots that are part of the mesh so as to be received by all the group members at that point in time.

1) Mesh Construction and Pruning: Similar to ODMRP, when a multicast source in MRMM has packets to send, but has no mesh set up for the multicast group, it floods a JOIN QUERY control packet through the entire network. When an intermediate node receives the JOIN QUERY packet, it stores the source ID and the sequence number in a data structure to detect any potential duplicates. The routing table is updated with the appropriate node ID (i.e. backward learning) from which the message was received for the reverse path back to the source node. If the message is not a duplicate and its Time-to-Live field (TTL) is greater than zero, it is rebroadcast. When

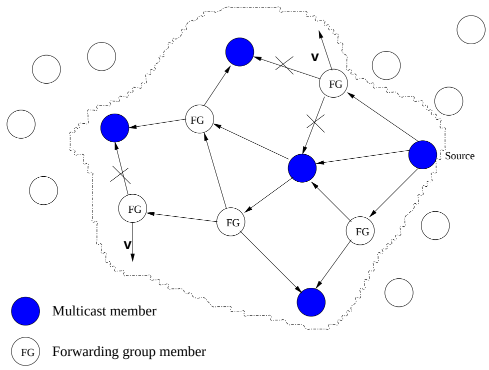


Fig. 7. MRMM mesh construction and pruning. The source first builds the mesh to reach all the group members (shaded) using the forwarding group (marked FG), the FG nodes are then pruned and outgoing edges from pruned FG nodes are removed.

the JOIN QUERY packet reaches multicast group members, it creates and broadcasts a JOIN REPLY to its neighbors. When the JOIN REPLY is received by a node, it checks if the next hop ID of one of the entries in the JOIN REPLY packet matches its own ID. If this is true, the node realizes that it is on the path to the source and becomes part of the forwarding group by setting the Forwarding Group (FG) flag. In this way, each FG member propagates the JOIN REPLY until it reaches the multicast source via the selected path (shortest). This whole process constructs (or updates) the routes from sources to receivers and builds a mesh of nodes, i.e., the *forwarding group*. Because the nodes maintain soft state, the JOIN QUERY packets have to be *periodically* flooded to refresh membership, maintain the mesh and rebuild the parts of the mesh that have disconnections due to node mobility. This is also done on-demand, i.e., when the source has data to send. Thus if we denote the set of nodes (robots) that make up the entire mobile robot network as G, then a JOIN QUERY selects from among these G nodes a set of nodes $F \subseteq G$ that are on the path from the source to all the group members. The set of group members is denoted as M. Figure 7 depicts the set of nodes F which is comprised of the nodes labeled FG and the set of group members. Note that $F \cap M \neq \phi$.

In contrast to ODMRP, MRMM exploits the mobility knowledge present in mobile robot networks, i.e., the knowledge of d_{rest}, v and t_{rem} as mentioned in Section III in order to run a *mesh pruning algorithm*. The objective of the pruning algorithm is to select a new set of nodes $P \subseteq F$ that maximizes the lifetime of the mesh without greatly affecting the redundancy and path lengths of the resulting mesh formed by the set of nodes P. This new set of nodes P is the newly selected, smaller set of forwarding group nodes. The nodes that are removed from F to get P are selected to be the nodes that are most likely to break apart from the mesh.

Since P is generally smaller than F, the number of rebroadcasts and consequently the overall control overhead will be reduced in MRMM. For example, in Figure 7, the nodes depicted with a velocity vector are moving away from the mesh[4] and the mesh pruning algorithm will select these nodes to be removed from F. Thus as shown in Figure 7, those nodes will not rebroadcast the JOIN QUERY, reducing the number of JOIN QUERY transmissions.

Another important consequence of this is that the data packets will travel over a sparser mesh resulting in lower number of data transmissions required to deliver all the data packets. Thus, MRMM will have an improved forwarding efficiency.

However, the goal is to reduce the set of forwarding group members without affecting other factors such as reliability and packet delivery. Making the mesh too sparse may affect the reliability of the mesh resulting in lower packet delivery performance. Instead, MRMM uses ω, the *mesh sparseness factor* to control how aggressively nodes are removed from the set F. In general, a sparser mesh is less reliable and has lower packet delivery performance. However, since MRMM constructs the mesh out of the least volatile nodes, even a sparser mesh provides similar packet delivery performance, as will be demonstrated by the results.

In order to enable mesh pruning in a distributed manner, each node, upon receiving a JOIN QUERY packet, forwards the packet (and thus includes itself in the forwarding group) probabilistically. A node i will uniformly choose a random number between 0 and 1. If this value is less than a threshold value p_i, it will rebroadcast the packet. Otherwise it will silently absorb the packet and not become part of the mesh. The threshold value p_i is defined as:

$$p_i = min(1, \frac{1}{v}^{min(\omega, \frac{d_{rest}}{R})})$$

where R is the radio range and d_{rest} is the distance to rest as specified in Section III. The parameter ω controls how sparse the mesh will be. If it is large, p_i will be smaller and the mesh sparser due to fewer nodes rebroadcasting the JOIN QUERY. In the simulation, we use a value of 2.5 as a heuristic for ω which we found experimentally to provide good performance. Finally, if v=0 (i.e., the node is static) or $v \leq$1, p_i is set to 1.

In some scenarios, when many nodes have high mobility, the forwarding group formed may be so sparse that all group members are not reachable. To alleviate this problem we require

[4]Note that we do not require directionality information and only use the magnitude of velocity to perform mesh pruning. Moving away from the mesh indicates that the robot is moving from its current position in the data delivery structure.

that nodes that wish to join/leave a group in MRMM flood a JOIN/LEAVE message with the group identifier so that group membership information is available in the entire network. Thus, any source that sends out a JOIN QUERY can check that all group members are reachable with the currently set up forwarding group. To enable this, the JOIN REPLY piggybacks information about group members that originate the JOIN REPLY (similar to source routes in DSR). When the source receives all the JOIN REPLY packets it can verify that the entire group membership is reachable via some forwarding group member[5]. Finally, if the source finds that all the group members are not represented, a basic JOIN QUERY is instead sent (identified by a flag bit) so that no mesh pruning occurs (i.e., p_i is 1 at all nodes) and the maximum group members are now reachable. MRMM then switches back to the original JOIN QUERY (with mesh pruning) in the JOIN QUERY after 5 successive query periods.

2) Data Delivery: Data is delivered in a similar manner as in ODMRP. When receiving a multicast data packet, a node that is a member of the FG forwards only non-duplicate packets. Nodes in the forwarding group are demoted to non-forwarding nodes if not refreshed before they timeout. Thus, over time the mesh will disappear if not used.

In contrast to ODMRP, each node checks (using odometry readings) if it has moved a threshold distance (500m) away from the time the mesh state was set up for that data source. It then stops broadcasting the data packets for that source even if they arrive from some other point in the mesh. This feature also helps to improve the forwarding efficiency of MRMM.

3) Reverse Multicast: As mentioned in Section II, reverse multicast (many-to-one) communication is useful in many mobile robot application scenarios, e.g., when many robots want to report their sensor readings to the operator. The mesh constructed in MRMM can also be used for this purpose. The operator acts like a source and constructs a mesh to all the robots it wants to receive sensor readings from. The robots can then report their readings by sending packets upstream on the mesh that is constructed by the operator.

4) Discussion: Using Global Positioning Information: We discuss the use of global positioning information to improve multicast performance. Previously, this idea has been proposed in [17] for MANETs. The same idea can be applied to mobile robots as follows. If mobile robots are GPS enabled, each robot can easily calculate how soon it will move out of the radio range

[5]Certain group members may be reachable through multiple forwarding group members. This is typical of ODMRP whose mesh based multicast has inherent redundancy.

of its neighbors. Thus, if we model the mesh as a graph, we will know the lifetimes of each edge in the graph. The source can learn the minimum of these lifetime values during the mesh construction phase, and then set the JOIN QUERY periodic interval to the minimum timeout value of the mesh.

However, in our mobile robot network we do not assume GPS availability since the robots may be used indoors or in areas where signal quality is bad and GPS equipment may not be economical to put on all robots. Thus, in our scenario, timeouts need to be estimated by assuming that each of a node's neighbors is exactly some distance away (e.g., 125 m or half the radio range).

B. Evaluation Methodology

We use the Glomosim simulator [14] to evaluate the performance of MRMM. We implemented MRMM in Glomosim and compare it to ODMRP. We simulated a network of 100 robots randomly distributed in an area of $1000m$x$1000m$. A simulation duration of 500 seconds is chosen.

Mobility pattern: The mobility model used is the same as the one used to evaluate the unicast routing protocol (Section IV-C).

Traffic pattern: Each robot that is a multicast source sends multicast data packets to a group. Each packet sent is 512 bytes in size and the packet rate is 1 packet every second per source.

Performance Metrics: The following metrics are measured in comparing the protocols: (1) Control overhead: Number of control packets transmitted by the multicast protocol. In both protocols these consist of JOIN QUERY, JOIN REPLY, and JOIN ACK packets. (2) Multicast delivery ratio (MDR): Fraction of multicast data packets originated by the source that are received by the receivers. For example, in a multicast group with K members, each originated packet needs to be received by K members in order for the MDR to be 100%. (3) Average delivery latency (Delay): Packet delivery latency averaged over all of the multicast packets delivered to all the receivers. (4) Forwarding efficiency (FE): Average number of data packet transmissions per delivered data packet. This is an indicator of the bandwidth efficiency of the protocol since the lower the FE, the lesser is the amount of bandwidth used to deliver a set of data packets.

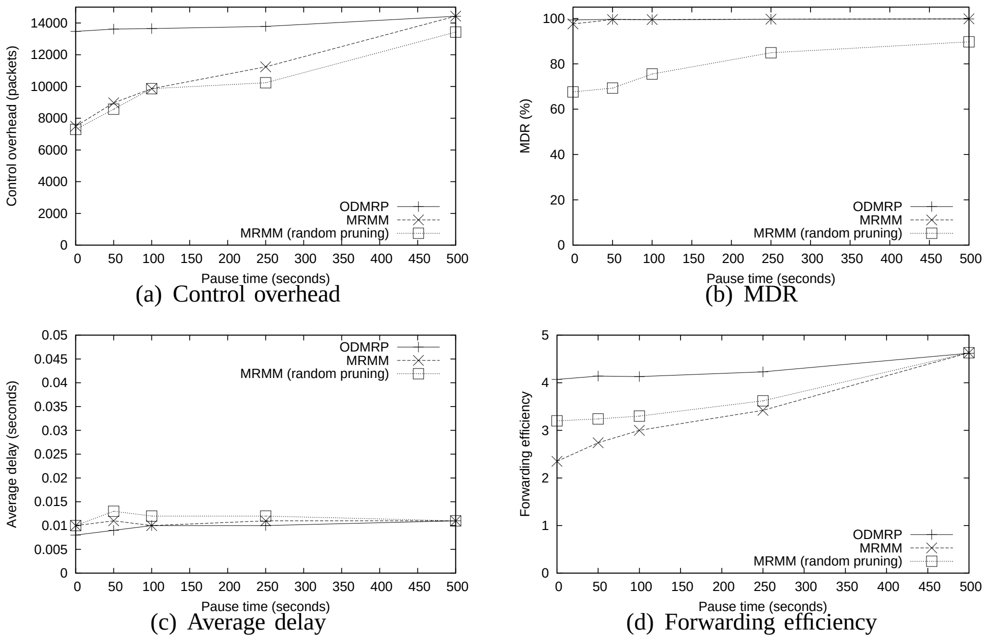


Fig. 8. Performance comparison with varying mobility.

C. Evaluation Results

In this section, we compare MRMM with ODMRP using the four metrics discussed above for a network of 100 robots. We examine the effects of the robot mobility, the multicast group size, and the number of multicast sources on the performance of the two protocols.

1) Effects of Mobility: In this section, we vary the pause time from 0 to 500 seconds to simulate the conditions of a continuously mobile to a completely static mobile robot network. We assume a single multicast group in the network with 10 members. An example of such a scenario is a human operator or a controller sending control messages to all 10 robots with video cameras which form a multicast group.

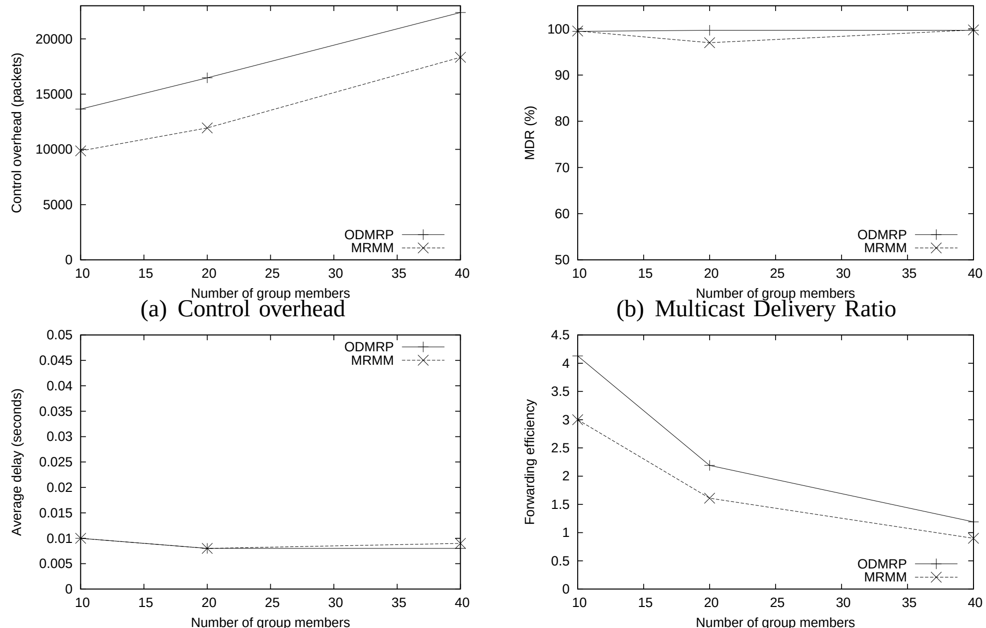
The performance results are depicted in Figure 8. The first observation is that the control overhead of MRMM is significantly lower than that of ODMRP for a wide range of pause times. As seen in Figure 8(a), for the continuous movement scenario, the overhead of MRMM is 44% lower than that of ODMRP. The control overhead remains lower for a wide range of pause times and finally is equal when the network is completely static. The control overhead reduction in MRMM is primarily due to a reduction in the propagation of JOIN QUERY packets. Since nodes that are likely to break the mesh are less likely to rebroadcast the JOIN QUERY, the number of total transmissions of JOIN QUERYs is reduced, which leads to reduced control overhead. In addition, since the mesh is composed of a smaller set of nodes, the number of JOIN

ACKS transmitted is also slightly reduced. For a static network with pause time 500 seconds, MRMM cannot distinguish which nodes are likely to break the mesh, and consequently all nodes rebroadcast JOIN QUERYs resulting in the performance being similar to ODMRP. While schemes could be designed to reduce the size of the mesh in this scenario, it is beyond the scope of MRMM.

More importantly, Figure 8(b) shows that despite having low overhead, MRMM delivers as many multicast packets as ODMRP. This is interesting since the mesh formed by MRMM is sparse and consequently more susceptible to failures due to mobility. However, by finding the set of long-lifetime nodes, the MRMM mesh is more resilient to failures from mobility which balances out the lower redundancy in the mesh. This results in the MDR of MRMM being close to that of ODMRP. Additionally, Figure 8(c) shows that the delay performance of both protocols are similar. ODMRP delivers packets with slightly lower delay since with a larger mesh, in many cases, it is able to deliver packets using shorter transmission paths. Another significant benefit of MRMM is the improved forwarding efficiency. The forwarding efficiency measures the bandwidth usage of a multicast protocol by indicating how many transmissions of a data packet are required on average to deliver it. Since MRMM builds a sparser mesh, the number of times a data packet is transmitted is lower thus improving the forwarding efficiency. For example, at a pause time of 100 seconds, MRMM requires 27% fewer transmissions per successfully delivered data packet than ODMRP while at 0 second the FE is lower by 42%.

Finally, we also compared the effectiveness of a *random pruning strategy* for MRMM where nodes randomly rebroadcast JOIN QUERY packets with probability $\frac{1}{2}$ instead of using mobility information. While this technique also reduces overhead by pruning the mesh (Figure 8(a)), it also reduces the delivery ratio in comparison to mobility based pruning in MRMM (Figure 8(b)). This is because with random pruning, the mesh is made sparse and will contain nodes that are likely to move out of range and break the data delivery structure. If such mobile nodes are close to the source in the delivery structure they severely hamper the delivery ratio by disrupting delivery to a large number of group members down the mesh until the mesh is refreshed through the next periodic JOIN QUERY. Due to its low delivery ratio, the forwarding efficiency of random pruning is also worse (Figure 8(d)) than mobility based pruning.

In summary, MRMM with mobility information based pruning reduces the control overhead and improves the forwarding efficiency for a wide range of mobility scenarios in comparison to



(a) Control overhead (b) Multicast Delivery Ratio

(c) Average delay (d) Forwarding efficiency

Fig. 9. Performance comparison with varying group size.

ODMRP.

2) Effects of Group Size: In mobile robot networks, it is likely that different multicast groups may have different sizes. Some groups may be large, e.g., the group consisting of all robots with sonars, whereas some groups with more costly equipment may be small, e.g., the robots equipped with video cameras. Also the group size may vary over time. For example, a team of robots with remaining energy above a certain threshold is likely to change in size as individual robot's energy is gradually depleted or replenished. Thus, it is important to study the performance of multicast protocols with varying group sizes.

In this section, the pause time considered is 100 seconds. Only one group exists in the mobile robot network while the group size is varied from 10 to 40.

The results are depicted in Figure 9. The results show that the overhead of both protocols grow with the group size. This is because an increase in the number of members results in an increase in transmissions of JOIN QUERY and JOIN REPLY packets. MRMM has consistently lower control overhead than ODMRP even as the group size increases. Figure 9(b) shows that the MDR of MRMM is comparable to that of ODMRP despite its lower overhead. This result is significant since as the group size increases, using a sparser mesh for multicast in general could potentially result in a reduced delivery ratio. However, since the nodes in the MRMM mesh are on average more reliable[6] than the ones in the ODMRP mesh, the packet delivery ratio remains similar.

The delay performance of both protocols are also similar as the group size is increased. The forwarding efficiency of both protocols improves (decreases in value) as the group size is increased. This is expected since as the number of members increases, more and more members are likely to be reached with the same number of transmissions. This is because MRMM, like ODMRP, is a broadcast based protocol and incorporates the use of *wireless multicast advantage*. Wireless multicast advantage allows multiple members to receive packets using a single broadcast transmission in the mesh. Thus as the number of members increases, the number of data transmissions required per successfully delivered data packet decreases. Note that the forwarding efficiency of MRMM is superior to that of ODMRP although the gap reduces as the group size is increased. MRMM actually achieves a forwarding efficiency below 1 for large group sizes which indicates that on average less than 1 data packet transmission is required per successfully delivered data packet. This is because as the group size increases, there are significant opportunities for delivering to multiple receivers with a single data broadcast and this can decrease the FE to below 1. This is more apparent in MRMM due to the use of a sparser mesh that reduces redundant broadcasts.

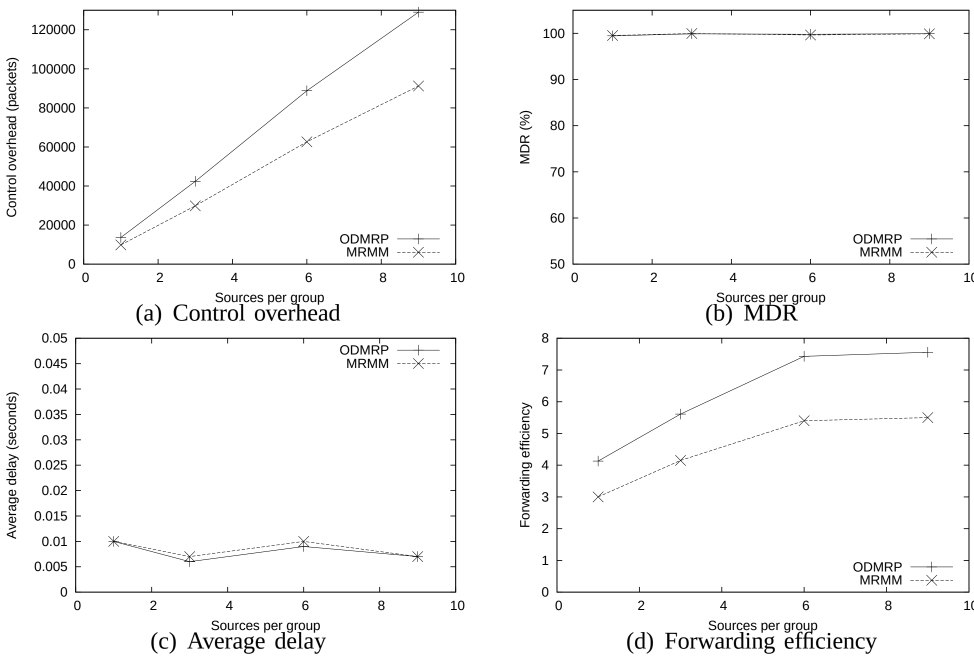
In summary, MRMM consistently has lower control overhead and higher forwarding efficiency than ODMRP for various group sizes while maintaining similar delivery ratios and delays.

3) Effects of Number of Traffic Sources: In a mobile robot network, it is likely that each robot has lot of sensor data which it wants to share with the rest of the robot team. In addition, in many applications a robot may need to multicast its current position and status to the rest of the team. In such scenarios, each multicast group is likely to have more than one source.

In this section, we consider a mobile robot network with a pause time of 100 seconds. The number of members in the group is 10, same as in the previous section. We then compare the performance of both protocols by varying the number of sources of multicast traffic within the group as 1, 3, 6 and 9 sources.

Figure 10 depicts the results of this comparison. The first observation is that as the number of sources increases, the control overhead of ODMRP increases since each source needs to build

[6] We use the term “reliable” loosely to denote the stability of a mobile node in upholding a routing path. The nodes chosen in an MRMM forwarding group are considered more “reliable” because they are less likely to break the forwarding path due to their low speeds or distances remaining to resting being small.



(a) Control overhead
(b) MDR
(c) Average delay
(d) Forwarding efficiency

Fig. 10. Performance comparison with varying number of multicast sources.

and maintain its own mesh for multicast. Thus, the high overhead mesh building and maintenance of ODMRP results in a large control overhead for ODMRP as the number of sources increases. In MRMM, each source also needs to build and maintain its own mesh. However, since each mesh is sparse, ODMRP always has 40% higher control overhead. The MDR of both protocols is close to 100% and similar across a wide range of traffic sources. However, due to its higher control overhead and consequent higher congestion, it is expected that as the traffic sources are further increased, ODMRP will be able to support fewer number of traffic sources than MRMM.

The delay performance of both protocols are similar with varying sources. Since each source builds its own mesh, the averages delay of each mesh is similar to that of the previous section since the network is not yet congested and queuing delays are insignificant. As shown in Figure 10(d), the forwarding efficiency of MRMM is superior across a wide range of traffic sources. When only one traffic source exists, ODMRP requires 1 extra packet transmission for every delivered data packet when compared to MRMM. When the traffic sources are increased, ODMRP requires 2 extra packet transmissions for every delivered data packet in comparison to MRMM.

In summary, MRMM has superior performance with respect to a varying number of traffic sources. Its lower control overhead potentially allows it to scale to a larger number of traffic sources than ODMRP.

VI. Related Work

There has been a large body of work on unicast and multicast routing protocols for wireless ad hoc networks. A survey on these work can be found in [18]. Several reactive unicast routing protocols (e.g. AODV [6] and DSR [5]) have been proposed for wireless ad hoc networks and their performance compared in random mobility scenarios [4], [15]. Reactive protocols only generate control traffic in response to a request for data transfer. Thus while they cause small delays in initiating a connection, their overhead is low. On the other hand, proactive protocols (e.g. DSDV [12] and OLSR [13]) continuously exchange control traffic to maintain routes to all destinations. While this reduces the delay in initiating a connection, the control overhead can be higher than reactive protocols. Since mobile robot teams are resource constrained, we focus on improving low overhead reactive routing protocols in this paper.

Similarly, many multicast protocols have also been proposed for multicast in wireless ad hoc networks. These include traditional tree-based or mesh-based protocols such as MAODV [19], ODMRP [7], overlay-based protocols such as AMRoute [20], PAST-DM [21], and back-bone-based protocols such as MCEDAR [22], and stateless protocols such as DDM [23], HDDM [24], and RDG [25]. Different from these works, in this paper we focus on exploiting the additional mobility information available specifically in mobile robot networks to improve the performance of unicast and multicast routing protocols in application-specific ad hoc networks such as mobile robot networks. The work in [26] is similar to our work in that it proposes to exploit non-random behavior of mobile users to predict the future state of network topology and consequently proactively reconstruct routes before connectivity is broken. In contrast, we target mobile robot networks in which unique mobility information is available and does not need to be predicted. Additionally, the contribution of our study is not to improve routing protocols for general mobile ad hoc networks, but to study the specific needs of mobile robot teams and design, adapt and improve routing protocols for networks formed by mobile robot teams.

The use of ad hoc communication has been proposed for applications in mobile robotics. For example, in [27], the authors propose the use of autonomous communication relays for tactical robots. They further discuss the use of such relays for allowing tactical ground robots to communicate with a base [28] and also operate in hazardous environments [29]. In [3], the authors propose and evaluate the use of ad hoc networking techniques and the corresponding algorithms for communication in mobile robot networks for both coordination and sensing oriented communication.

VII. Conclusions

In this paper, we first motivated the need for distributed communication protocols for mobile robot teams. We identified the application scenarios and usefulness of communication protocols for mobile robot applications. Unicast communication among robots in such mobile multi-robot teams is useful for the coordination of such teams as well as exchanging data. Multicast is useful for mobile robot team applications that involve coordination among team members, sensing, and data collection. Multicast also enables flexible control and operation of the multi robot team by supporting organization of the mobile robots into many structures such as sub-teams and hierarchies. Finally, multicast also enables efficient discovery and usage of distributed resources available in the network due to heterogeneous capabilities and resources of each robot.

We then designed and evaluated two new protocols for unicast messaging and one new protocol for multicast in mobile robot networks. Together, these protocols provide a suite of techniques for a wide range of communication needs for mobile robot networks. All protocols proposed exploit the fact that abundant mobility information exists in a mobile robot network. Unlike typical MANETs, in which mobility cannot be predicted easily, mobile robots have specific instructions and perform specific tasks which govern their mobility and pausing behavior.

Both unicast protocols, MRSR and MRDV, exploit this information in the following ways: (1) constructing longer lifetime routes through the use of robots that are likely to remain as part of the route, (2) intelligently assigning timeouts to links and routes based on the mobility information, and (3) reducing the large overhead of route discoveries by probabilistic rebroadcast of Route Request packets. These features allow MRSR and MRDV to provide unicast routing in mobile robot networks with lower overhead without incurring any reduction in packet delivery performance in comparison to the state-of-the-art MANET routing protocols.

The MRMM protocol for multicast also exploits the mobility information of mobile robots to build a more energy efficient multicast mesh. It also reduces the overhead and bandwidth usage and consequently reduces the energy for group communication in mobile robot networks. Multicast allows for efficient one-to-many and many-to-one communication that are critical in many mobile robot application scenarios. In our future work, we will study how multicast can be enhanced by localization.

ACKNOWLEDGMENTS

This work was supported in part by the National Science Foundation under Grant IIS-0329061.

REFERENCES

[1] P. E. Rybski, S. A. Stoeter, M. Gini, D. F. Hougen, and N. Papanikolopoulos, "Effects of Limited Bandwidth Communications Channels on the Control of Multiple Robots," in *Proc. of IEEE IROS*, 2001.

[2] T. Balch and R. C. Arkin, "Communication in reactive multiagent robotic systems," *Autonomous Robots*, vol. 1, no. 1, pp. 27–52, 1994.

[3] S. M. Das, Y. C. Hu, C. Lee, and Y.-H. Lu, "Supporting many-to-one communication in mobile multi-robot ad hoc sensing networks," in *Proc. of ICRA*, 2004.

[4] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols," in *Proc. of ACM MobiCom*, October 1998.

[5] D. B. Johnson and D. A. Maltz, *Dynamic Source Routing in Ad Hoc Wireless Networks*. Kluwer Academic, 1996.

[6] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," in *Proc. of IEEE WMCSA*, February 1999.

[7] S.-J. Lee, M. Gerla, and C.-C. Chiang, "On-Demand Multicast Routing Protocol," in *Proc. of IEEE WCNC*, September 1999.

[8] "Activmedia robotics," http://www.activrobots.com/.

[9] D. Maltz, J. Broch, and D. Johnson, "Quantitative lessons from a full-scale multi-hop wireless ad hoc network testbed," in *Proc. of WCNC 2000*, September 2000.

[10] H. Lundgren and E. Nordstrom, "AODV implementation code (Uppsala University)," 2004, http://core.it.uu.se/AdHoc/AodvUUImpl.

[11] Y.-C. Hu and D. B. Johnson, "Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks," in *Proc. of ACM MobiCom*, August 2000.

[12] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. of ACM SIGCOMM*, August 1994.

[13] T. Clausen, P. Jacquet, C. Adjih, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L.Viennot, "Optimized link state routing protocol (OLSR)," RFC 3626, Oct 2003.

[14] X. Zeng, R. Bagrodia, and M. Gerla, "Glomosim: A library for parallel simulation of large-scale wireless networks," in *Proc. of PADS Workshop*, May 1998.

[15] S. R. Das, C. E. Perkins, and E. M. Royer, "Performance comparison of two on-demand routing protocols for ad hoc networks," in *Proc. of IEEE INFOCOM*, March 2000.

[16] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in *Proc. of IEEE INFOCOM*, April 2001.

[17] S.-J. Lee, W. Su, and M. Gerla, "On-demand multicast routing protocol in multihop wireless mobile networks," *MONET*, vol. 7, no. 6, 2002.

[18] E. Royer and C. Toh, "A review of current routing protocols for ad-hoc mobile wireless networks," *IEEE Personal Communications*, April 1999.

[19] E. M. Royer and C. E. Perkins, "Multicast operation of the ad-hoc on-demand distance vector routing protocol," in *Proc. of MobiCom*, August 1999.

[20] J. Xie, R. R. Talpade, A. Mcauley, and M. Liu, “AMRoute: ad hoc multicast routing protocol,” *Mob. Netw. Appl.*, vol. 7, no. 6, pp. 429–439, 2002.

[21] C. Gui and P. Mohapatra, “Efficient overlay multicast for mobile ad hoc networks,” in *Proc. of IEEE WCNC*, March 2003.

[22] P. Sinha, R. Sivakumar, and V. Bharghavan, “MCEDAR: Multicast Core-Extraction Distributed Ad hoc Routing,” in *Proc. of IEEE WCNC*, September 1999.

[23] L. Ji and S. Corson, “Differential Destination Multicast–A MANET Multicast Routing Protocol for Small Groups,” in *Proc. of IEEE INFOCOM*, April 2001.

[24] C. Gui and P. Mohapatra, “Scalable multicasting for mobile ad hoc networks,” in *Proc. of IEEE INFOCOM*, March 2004.

[25] J. Luo, P. T. Eugster, and J.-P. Hubaux, “Route Driven Gossip: Probabilistic Reliable Multicast in Ad Hoc Networks,” in *Proc. of IEEE INFOCOM*, March 2003.

[26] W. Su, S.-J. Lee, and M. Gerla, “Mobility prediction and routing in ad hoc wireless networks,” *Int. J. Netw. Manag.*, vol. 11, no. 1, pp. 3–30, 2001.

[27] H. Nyugen, N. Pezeshkian, M. Raymond, A. Gupta, and J. Spector, “Autonomous communication relays for tactical robots,” in *Proc. of IEEE ICAR*, July 2003.

[28] H. Nyugen, N. Farrington, and N. Pezeshkian, “Maintaining communication link for tactical ground robots,” in *Proc. of AUVSI Unmanned Systems North America*, August 2004.

[29] H. Nyugen, N. Pezeshkian, A. Gupta, and N. Farrington, “Maintaining communication link for a robot operating in a hazardous environment,” in *Proc. of ANS 10th Int. Conf. on Robotics and Remote Systems for Hazardous Environments*, March 2004.