

Autonomous Enhancement of Disruption Tolerant Networks

Brendan Burns Oliver Brock Brian Neil Levine
 University of Massachusetts
 Amherst, MA

Abstract—Mobile robots have successfully solved many real world problems. In the following we present the use of mobile robots to address the novel and challenging problem of providing disruption tolerant network service. In disruption tolerant networks, all messages are transported by the physical motion of participants in the network. When these movements do not meet the service demands of the network, network performance can only be improved by adding robots that provide additional network service. The task of controlling such robots is a problem that is NP-Hard. To develop an approximate solution, we propose a nullspace-based algorithm for controlling the motion of the added robots. This controller simultaneously optimizes multiple network performance metrics. Experiments that simulate the addition of robots to a real-world disruption tolerant network show that the introduction of mobile robots running our control scheme can significantly improve the performance and service guarantees of a disruption tolerant network.

I. INTRODUCTION

Mobile robots have been applied to numerous real-world tasks including tour guiding, exploring unknown environments and assisting medical patients. Recent research in mobile ad-hoc networks known as *disruption tolerant networks* (DTNs) [1], [2], [3], [4], [5], presents a novel application for mobile robots as network service providers. In disruption tolerant networks, the movement of network participants transport messages rather than fixed connections between nodes. Because of this, the performance of disruption tolerant networks is dependent on the movement of its participants. In general the motions of these participants cannot be controlled. Consequently, adding mobile robots to ferry messages in the network is the only way to control and improve network performance.

In traditional ad-hoc networks, communication between two nodes in the network requires complete connectivity along the path connecting those nodes at the instant of communication. Unfortunately, in many situations, such as disaster relief, the combination of damaged infrastructure and wireless interference from debris limit communication. Alternately, applications such as sensor networks often feature devices with extremely limited wireless range. In both situations connectivity between nodes is intermittent and end-to-end paths between arbitrary pairs of nodes rarely exist. Communication in these scenarios requires disruption tolerant networking.

A simple example of such a scenario is shown in Figure 1. The nodes in group A are all within wireless range (shown in gray) of each other and can communicate. The same is true of the nodes in group B. However, no node in group A is within

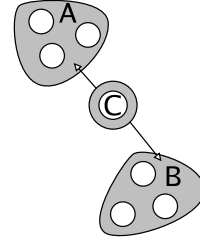
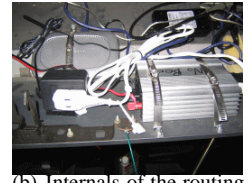


Fig. 1. A simple scenario requiring disruption tolerant networking



(a) Each bus is a node in the DTN



(b) Internals of the routing system

Fig. 2. The hardware implementation of a campus wide DTN

range of any node in group B. Therefore, using traditional ad-hoc routing protocols, communication is impossible. However, the motions of node C periodically take it within range of both group A and group B. Disruption tolerant networking allows messages from group A to be sent to node C which physically carries them within range of group B and then passes them on. Despite never having instantaneous connectivity, using DTN protocols, groups A and B can communicate.

In general, a DTN is a collection of mobile peers, such as the buses shown in Figure 2, that collaboratively construct an ad-hoc network. Each peer carries and delivers messages for themselves and on behalf of others in the network. Whenever two peers pass they exchange messages, both for themselves and on behalf of others. By holding messages for a period of time while moving physically through the space of the network, the instantaneous end-to-end link between peers required by traditional ad-hoc networks is no longer necessary.

Since communication in a DTN is provided by the physical movement of the network's peers, the performance of the DTN also depends on this movement. When peers do not move in such a way as to satisfy the requirements of network traffic, performance declines and communication may become impossible. We propose the introduction of mobile robots as DTN service providers to address this issue. The problem of optimal control for agents to maximize packet delivery has

been shown to be NP-Hard [2]. We propose an approximate control algorithm based upon nullspace composition that directs the actions of mobile robots providing disruption tolerant network service. We present experiments in a simulation of a real-world disruption tolerant network deployed on buses servicing a university campus. These experiments (Figure 2) show that the introduction of mobile robots significantly improves network performance and robustness.

II. RELATED WORK

Control of autonomous robots servicing a disruption tolerant network represents a synthesis of research in networking and robotics. The body of related work is consequently drawn from both fields.

A. Networking

Disruption tolerant networks have been studied by a number of researchers in the network community. However, to our knowledge, ours is the only work that considers the introduction of controllable peers to the network.

Davis et al. [1] proposed what we believe to be the first approach to disruption tolerant networking. This work considered routing algorithms for a DTN made up of wearable computer users with limited buffers for storing network packets.

Zhao et al. [5] have proposed DTN networks based upon the motion of ferries that have completely known and predictable movements. Peers in the network use these motions to insure their packets are delivered to a destination, much as a person riding the ferry would. Work ongoing within the IRTF Delay-Tolerant Network Research Group [3] also focuses on routing given predictable peer movement.

Other routing strategies attempt to learn patterns from the observed movements of peers. Epidemic routing [6], Drop-Least-Encountered [1] and MV routing [2] all attempt to learn routing strategies by examining the connection patterns of peers in the network. Such strategies adapt to changes in peer movement and do not require peers with fixed motion.

Recently, researchers [7] have proposed methods for designing ferry routes that provide disruption tolerant network service to static nodes. In contrast to our work, these techniques develop a static route for a number of ferries servicing nodes that do not move. The techniques assume perfect information about the location of the members of the network and the movements of the ferries. They also assume that the members of the network never move. Because the routes provided by the algorithm are also static, they can not adapt to changing network requirements.

In the context of sensor networks, rumor routing [8] is a related approach for networking sensors without the costs associated with flooding. In rumor routing each peer passes on a message with a certain probability. Rumor routing is designed for static networks, but the spread of messages through the network is very similar to DTN routing.

More recently, Das et al. [9], [10] proposed adaptations of several existing ad-hoc routing protocols in order to optimize their operation for mobile robotic domains. Like the ad-hoc protocols they extend, these protocols assume end-to-end

connectivity between communicating nodes and thus are not applicable to disruption tolerant networking scenarios.

B. Multi-Objective Control

Numerous algorithms for multi-objective control have been proposed. In the following we discuss those most relevant to our work in control for servicing DTNs.

The subsumption architecture [11] makes each individual controller a finite state machine with inputs and outputs that are connected to other controllers or to real-world sensors and actuators. These controllers are organized in a hierarchy that achieves multi-objective control by having higher-level controllers manipulate or inhibit lower-level controllers.

The algebraic nullspace has been used to construct numerous multi-objective controllers [12], [13], [14]. The nullspace of a controller ϕ can be seen as the collection of control commands that can be performed in addition to ϕ such that the objective of ϕ is not compromised. Again, by ordering the controllers in a hierarchy and projecting, one-by-one, the outputs of a lower-level controller into the nullspace of the higher-level controllers, multi-objective control is achieved.

Numerous robotic tasks have been solved with nullspace composition. Of particular relevance is work by Sweeney et al. [15] who used nullspace control to maintain network connectivity for a collection of distributed agents. In this work, agents maintain line of sight (required for communication) while pursuing the exploration of an unknown environment.

Thibodeau et al. [16] present a very similar approach to multi-objective control in the context of discrete controller output. Their approach is applied to multi-objective control for a single mobile robot.

III. ENHANCING DTNS WITH MOBILE ROBOTS

When the motion of participants in a DTN does not meet the level of service required by users of the network, the network's quality-of-service suffers. Since general peer motion can not be changed, the only way to ensure network quality-of-service is to introduce mobile robots that provide additional network service by ferrying packets. This section addresses the problem of controlling such robotic network participants. We present metrics for network performance along with the primitive controllers that optimize each individual metric. Next, we discuss how a distributed approximation of these network metrics may be maintained in the context of a DTN. Finally we propose a distributed multi-objective controller that performs a modified nullspace composition of the primitive controllers. Nullspace-based multi-objective composition is an important contribution of this work. In addition to its role in controlling the motion of the mobile robots, ordering the primitive controller hierarchy provides a network administrator with an easy way to tune the behavior of the network to suit their needs. An example of such tuning might be prioritizing message latency over total bandwidth.

A. Network metrics and primitive controllers

Controlling the movement of robots providing DTN service requires metrics for assessing the performance of the network.

These metrics are the abstract sensors that serve as input to our control algorithm. In our implementation we use the following metrics: bandwidth, unique bandwidth, message latency and peer latency.

Each of these metrics quantifies a different aspect of network performance. Bandwidth ensures that all capacity present in the network is used. Unique bandwidth ensures that this capacity is used most effectively to transmit as many different messages as possible. Message latency ensures the timeliness of message delivery and favors nodes that are sending or receiving many messages. Peer latency guards against node starvation which otherwise could occur if an unvisited node is attempting to send, but never receiving any messages.

Each primitive controller in our system has a single network metric as sensory input and proposes motions for the robot that optimize that metric. The output of each controller is a particular peer, the destination for the robot that maximally improves the performance metric that the controller is attached to. The details of each of these primitive controllers are as follows:

Bandwidth controller ϕ_B

Traveling to any peer A will increase the bandwidth of the network by at least the number of messages the agent can obtain from A. This controller selects the peer with the largest number of unseen messages, amortized by the travel time it will take the agent to reach A's location

Unique bandwidth controller ϕ_U

The unique bandwidth controller is similar to the bandwidth controller, but it travels to the node with the greatest number of messages that have not been acquired by *any* node.

Deliver latency controller ϕ_D

The delivery latency controller chooses the peer whose messages have been in the agent's buffer the longest

Peer latency controller ϕ_P

The peer latency controller chooses the peer least recently visited by an agent, unless the time that it takes to visit this peer would actually cause an increase in the peer latency statistic. Such situations only occur when one peer is extremely far away from all of the others.

Unique service controller ϕ_S

This controller is not in service of a network metric. Instead it ensures that multiple robots do not simultaneously move toward the same peer in the network. This distributes service when multiple robots are present in the network

B. Distributed maintenance of network statistics

In any real network, the maintenance of the metrics used by these primitive controllers must be done in a distributed manner. While each peer has perfect information about their own state, they must guess the state of all other peers in the network.

To achieve distributed maintenance of these network statistics, an approximate snapshot of entire network's state is

maintained by each peer. Each peer estimates every other peer's state. In addition to network metrics (messages to send, message latency, etc), this state also includes the last known location and destination of each peer in the network. This information is used as input to the unique service controller and to determine where a robot should move to find a particular peer. The state estimate for each peer is timestamped with the moment when it was last known to be correct.

Whenever two peers meet, in addition to exchanging messages, they exchange their own perfect internal state and synchronize their state estimates for the remaining peers. The peers receive the union of their shared state estimates with newer information replacing older information. From these exchanges, a distributed view of the network's state is diffused across the network to all peers. The distributed model is maintained by both robotic and non-robotic peers in the network, insuring maximal diffusion of information. For the mobile robots, the estimate of network state serves, in place of perfect information, as input to their control algorithm.

C. Managing multiple network performance metrics

In any real network, there are inter-dependencies that prevent a controller from optimizing each of the individual network performance metrics independently. Instead a multi-objective controller is needed that can combine the objectives of multiple primitive controllers and simultaneously optimize multiple network performance metrics. To construct this multi-objective controller we choose a nullspace-based approach.

1) *Nullspace control*: Any controller ϕ has a nullspace that contains the set of all actions that a robot might take without hindering the operation of ϕ . Within this nullspace, a subordinate controller is free to choose the available action that best satisfies the subordinate controller's performance metric. Because it lies in the controller's nullspace, the action chosen by the subordinate controller will never affect the behavior of the higher-level controller. This *subject-to* composition can be repeated an arbitrary number of times with each subordinate controller operating in the combined nullspace of the controllers that are above it in the hierarchy. In this way, multi-objective control is achieved in a principled manner without the need for hand-tuning by the system designer.

2) *Threshold Nullspace*: Controlling robots in service of a DTN has two specific features that differentiate it from generic nullspace control. First, the output of the controller is a discrete rather than continuous value. Each primitive controller selects the peer in the network that maximizes its particular performance metric. Second, the objective of the robot is to maintain the performance metrics of the network above a specified threshold rather than at a specific value. The *threshold nullspace control* algorithm, a variant of traditional nullspace control, takes advantage of these properties and is well suited for controlling robots in a DTN.

Given a controller whose output is finite and discrete, the nullspace of that controller is a set of discrete values that is a subset of all possible actions. In the case of DTNs, this set is a subset of the peers in the network. For any particular controller, the threshold nullspace is defined with

respect to the performance metric and some threshold T . Given some performance metric $P(x)$ and X , the set of all possible controller outputs, the nullspace of a discrete controller ϕ is:

$$N(\phi) = \{x \in X : P(x) \geq T\}$$

Given this formulation of the threshold nullspace, composition can be achieved by using the nullspace set of a higher controller as the set of possible actions the subordinate controller chooses from. Thus, any output chosen by the subordinate controller must satisfy the threshold requirements of the higher-level controller.

3) *Controller Ordering*: A threshold nullspace controller requires a hierarchy of its constituent controllers. In the context of DTNs, this ordering describes the relative importance of the different network metrics. The specification of the ordering also provides a network administrator with a simple method for adjusting the network performance characteristics. If message latency is more important for a particular network than maximal bandwidth, the message latency controller may be placed highest in the controller hierarchy. In our experiments we used the following ordering:

$$\phi_P \triangleleft \phi_D \triangleleft \phi_U \triangleleft \phi_B \triangleleft \phi_S$$

where $\phi_i \triangleleft \phi_j$ indicates that ϕ_i is subordinate to ϕ_j .

This specific ordering places the unique service controller highest in the hierarchy. The nullspace of this controller is the set of all peers that no robot is currently moving toward. The remaining controllers are ordered in terms of the relative sizes of their nullspaces. The nullspace of the bandwidth controller is necessarily larger than that of the unique bandwidth controller and so forth through the ordering. This ordering maximizes each subordinate controller's flexibility. It is worth noting that this ordering was designed by hand and is not the only valid option. We expect that future work may offer different orderings, or in fact different primitive controllers. Additionally, the thresholds used to define the nullspace of the primitive controllers were also specified by hand. We anticipate that such parameters would be manipulated by a network administrator to refine the performance of a network to suit their specific needs.

IV. EXPERIMENTAL EVALUATION

To evaluate the ability of autonomous robots to improve the capacity of a disruption tolerant network, we conducted a series of experiments in simulation. The simulation was based on data from an actual DTN that is being constructed on the campus of the University of Massachusetts Amherst. The UMass DTN is deployed on the campus buses that service the school. The DTN is intended to allow students to communicate via e-mail and other non time-critical protocols in areas that lack network service.

Each bus (pictured in Figure 2a) that services the campus has been equipped with a small computer running the Linux operating system and a wireless router (Figure 2b). The computer has a 40GB hard drive, enabling essentially unlimited buffering of messages on the network. There are a total of nine bus routes that service campus. In our experiments we

place a single bus on each route. All experiments represent the average of ten trials. For each trial, each bus' movement was generated using a single set of observed GPS locations for each route but varying the length of time it took to move from waypoint to waypoint.

Whenever two buses come within range of each other they exchange messages. Whenever two simulated buses came within 150 meters of each other they were considered connected. In addition to the buses we add simulated robots to the DTN. For our experiments we assumed that all robots were capable of moving to any location in their environment. Control of robots given physical limitations on their motion is a subject for future examination. All network traffic originated on buses and were sent to other buses (simulating communication between riders on those buses). A Gaussian distribution was used to generate the time interval between messages sent by a particular bus for each trial. Each experiment was run for twenty hours, or approximately a complete day of bus service. For message routing between peers we used the MV [2] routing algorithm.

A. Improving network capacity

The primary motivation for introducing autonomous robots into disruption tolerant networks is to improve network performance. To empirically evaluate the ability of robots to improve a DTN we introduced various numbers of robots using our controller into the network and observed the changes in the fraction of messages delivered and the latency of those messages, while varying network load. These experiments were conducted for one, three, five, seven and nine additional robots. Figure 3. The values reported are the average over all messages sent in ten different trials.

From these results it can be seen that the addition of even one robot to a DTN significantly improves the fraction of messages delivered successfully. It is worth noting that as the delivery fraction increases, message latency does as well. This is understandable since messages that fail to be delivered do not count towards delivery latency. A system that delivers more messages suffers a greater latency "penalty" for the extra messages it delivers that were not delivered by the other system.

The introduction of three robots improves performance over one robot. However, further robots do not significantly improve the fraction of messages delivered. The introduction of more agents does result in improvements in message latency, indicating that as resources are added, the threshold-nullspace controller manages to optimize multiple metrics. This observation provides important information to system designers. Deploying and supporting a large robotic team is an expensive proposition. These experiments indicate that the price of adding more than a few robots is not warranted.

B. Comparing with uncontrolled peers

Adding any additional peer to the network adds network capacity and thus improves network performance. To evaluate the effect of the control algorithm on network performance, we conducted experiments where we added additional buses

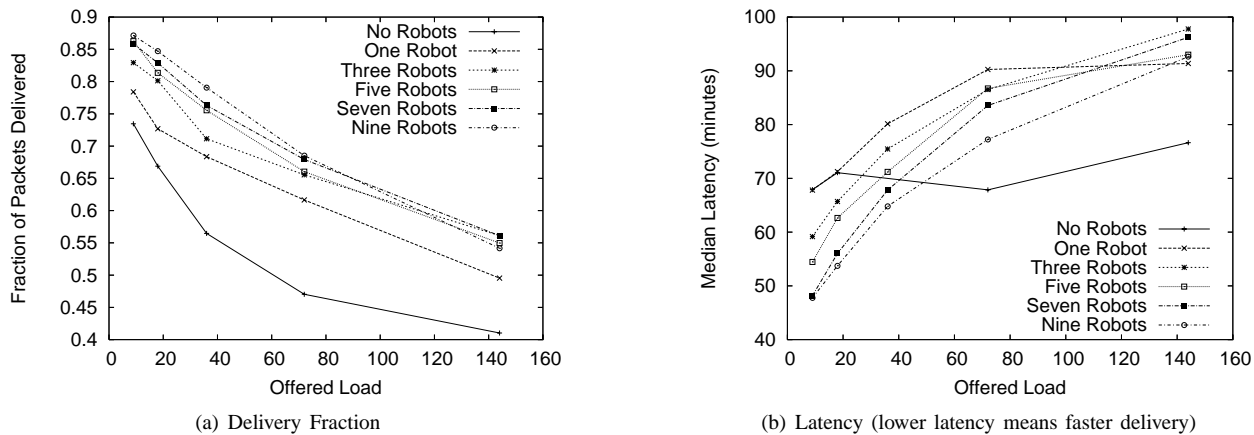


Fig. 3. Simulation experiments comparing the effect of adding different numbers of robots on delivery fraction and latency as a function of offered load.

to the network instead of robots. Like the robots, these buses transported messages but did not broadcast messages of their own. A representative graph comparing the addition of five robots with the addition of five buses is shown in Figure 4. The baseline performance of the network with no additional peers is also shown. We conducted similar experiments for different numbers of buses and robots. These results are omitted for reasons of space but show similar results. The numbers reported for each experiment represent the average of all messages sent in ten different trials.

From the graphs, it can be seen that the addition of agents increases the fraction of packets delivered, while additional buses have little effect. The addition of buses *does* significantly reduce the latency of messages. Adding a bus to a route effectively halves the average distance that a message has to travel to reach the larger network. Taken together with the results in Figure 3, these experiments indicate that good network improvement may be obtained by adding a few agents to improve the delivery fraction *and* adding greater numbers of buses to the network to decrease latency.

C. Comparisons to subsumption

In order to compare the performance of threshold nullspace control to other methods of multi-objective control we implemented a subsumption [11] controller for the robots servicing the network. For the subsumption algorithm, if the performance metric for any particular controller was below its threshold, that controller subsumed the other primitive controllers and its output was used as the control signal. If multiple controllers are below threshold, the higher controller is given priority. Figure 5 compares the performance of subsumption to threshold-nullspace for varying numbers of robots with a fixed network load of 18 packets per bus per hour.

These experiments show that threshold nullspace control is better at merging the various control objectives of the primitive controllers. Threshold nullspace is capable of merging the bandwidth and unique bandwidth objectives to achieve a higher fraction of messages delivered (Figure 5a) and as more

robots become available, achieve the secondary objective of reducing message latency (Figure 5b).

V. CONCLUSIONS

Disruption tolerant networks are an important method of networking in situations with sparse and disconnected network participants. The performance of disruption tolerant networks is dependent upon the motion of the peers in that network. Consequently, the only way to ensure network quality-of-service metrics is to introduce autonomous mobile robots as network service providers.

Providing such disruption tolerant network service is a novel challenge for mobile robotics. The performance of any network is defined by multiple metrics such as bandwidth and latency. Robotic service providers must adequately balance all of these metrics to provide quality service. We have presented a multi-objective approach to control based upon the principles of nullspace composition that approximates this simultaneous optimization. This controller is also responsive to the particular needs of a specific network. The hierarchy of primitive controllers and their threshold parameters are easily adjusted by a network administrator to develop a network that meets their particular needs. Experiments that add mobile robots to a simulation of a real-world DTN show that the incorporation of mobile robots using our proposed controller significantly improve the performance of the disruption tolerant network and outperforms both adding uncontrolled agents and an alternative approach to multi-objective control. These experiments also show that there is diminishing reward for adding additional agents. This result, especially given the cost of supporting large robotic teams, indicates that there is an optimal price point with a small number of agents supporting the network.

ACKNOWLEDGMENT

This research was supported in part by National Science Foundation awards CNS-0519881 and EIA-0080199.

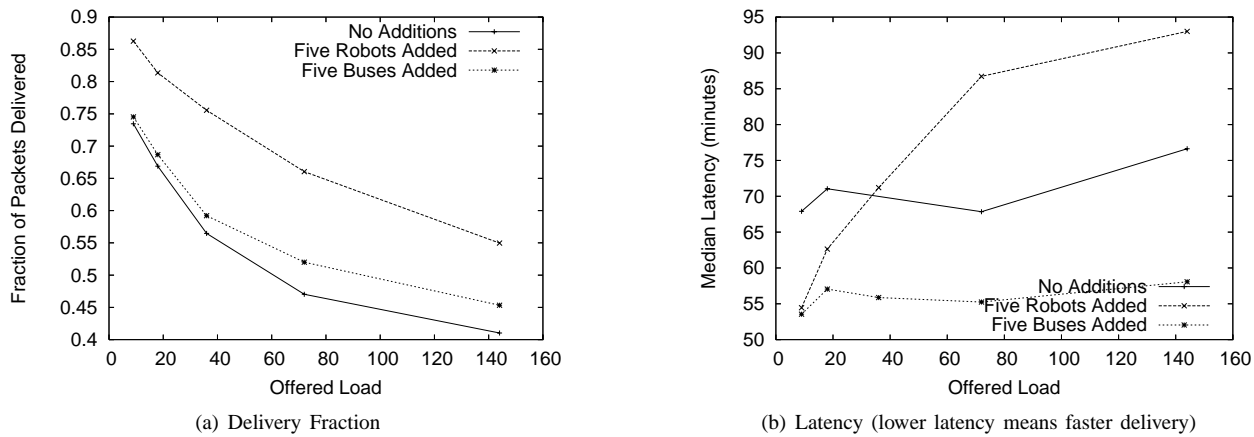


Fig. 4. Simulation experiments comparing the effect of five additional buses versus five additional robots on delivery fraction and latency as a function of offered load.

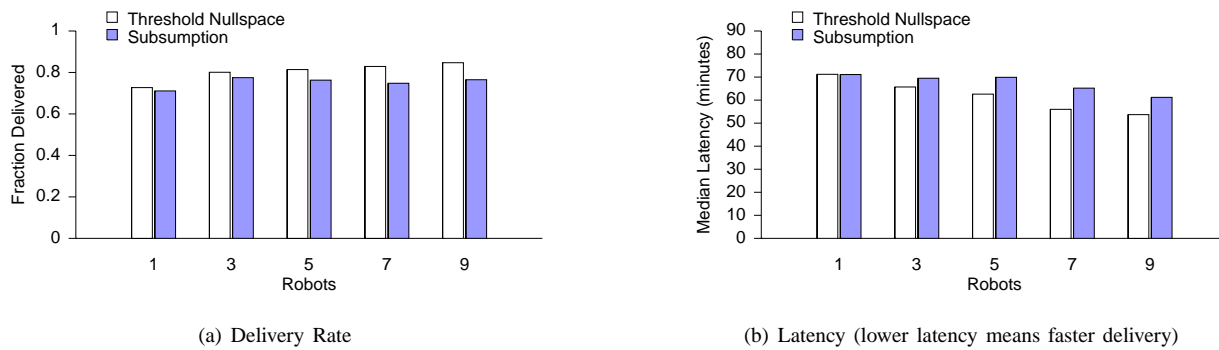


Fig. 5. Simulation experiments comparing the effect of subsumption vs threshold nullspace control on delivery fraction and latency as a function of offered load.

REFERENCES

- [1] J. A. Davis, A. Fagg, and B. Levine, "Wearable computers as packet transfer mechanisms in ad-hoc networks," in *Proceedings of the International Symposium on Wearable Computing*, 2001.
- [2] B. Burns, O. Brock, and B. Levine, "MV routing and capacity building in disruption tolerant networks," in *IEEE InfoCom*, March 2005.
- [3] "Delay-tolerant networking research group," <http://dtnrg.org>.
- [4] S. Jain, K. Fall, and R. Patra, "Routing in a delay-tolerant network," in *ACM SigComm*, August 2004.
- [5] W. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad-hoc networks," in *ACM Mobihoc*, May 2004.
- [6] A. Vahdat and D. Becker, "Epidemic routing for partially-connected ad-hoc networks," Duke University, Tech. Rep., 2000.
- [7] E. Z. Wenrui Zhao, Mostafa Ammar, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," in *IEEE InfoComm*, Miami, Florida, 2005.
- [8] D. Braginsky and D. Estrin, "Rumor routing algorithms for sensor networks," in *First International Workshop on Sensor Networks and Applications*, 2002, pp. 22–31.
- [9] S. Das, C. Hu, C. Lee, and Y. Lu, "Efficient unicast messaging for mobile robots," in *IEEE Conference on Robotics and Automation*, Barcelona, April 2005.
- [10] —, "An efficient group communications protocol for mobile robots," in *IEEE Conference on Robotics and Automation*, Barcelona, April 2005.
- [11] R. A. Brooks, "A robust layered control system for a mobile robot," *International Journal of Robotics and Automation*, vol. RA-2, no. 1, pp. 14–23, 1986.
- [12] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.
- [13] R. A. Grupen, M. Huber, J. A. Coelho, Jr., and K. Souccar, "Distributed control representation for manipulation tasks," *IEEE Expert*, vol. 10, no. 2, pp. 9–14, 1995.
- [14] J. Coelho and R. Grupen, "A control basis for learning multifingered grasps," *Journal of Robotic Systems*, vol. 14, no. 7, pp. 545–577, 1997.
- [15] J. Sweeney, H. Li, R. A. Grupen, and K. Ramamritham, "Scalability and schedulability in large, coordinated, distributed robot systems," in *Proceedings of the International Conference on Robotic Applications (ICRA) 2003*, 2003.
- [16] B. Thibodeau, S. Hart, D. Karuppiah, J. Sweeney, and O. Brock, "A cascaded filter approach to multi-objective control," in *Proceedings of the International Conference on Robotics and Automation*, New Orleans, April 2004.