Active QoS Flow Maintenance in Robotic, Mobile, Ad Hoc Networks

John D. Sweeney, Roderic Grupen and Prashant Shenoy
Laboratory for Perceptual Robotics
Department of Computer Science
University of Massachusetts Amherst
{sweeney, grupen, shenoy}@cs.umass.edu

Abstract

This paper considers a system of mobile robots that is able to form an ad hoc network. Robots within the system need to form connections to other members with certain quality of service (QoS) requirements. We present a distributed control architecture that allows robots participating in routing a QoS flow to maintain the required level of service while addressing secondary objectives. This is based on a distributed control system that allows global properties to be maintained using error-suppressing controllers that guarantee best-effort adherence to global system constraints.

Robots involved in routing a QoS flow may cause a routing fault either by moving out of range of transmission with neighboring nodes, or by becoming incapable of routing at the specified QoS. If a robot is able to predict when such a fault may occur, it can then recruit some other robot capable of providing the appropriate QoS to take over its routing duty, without QoS interruptions. We present the QoS Hand Off Protocol (QHOP) to perform this task.

We evaluate the control architecture and QHOP protocol in simulation, using the ns-2 network simulator and Player/Stage robot simulator platforms.

1 Introduction

In this paper, we examine teams of mobile robots that can form ad hoc wireless networks. One possible application scenario is a rescue situation where a team of robots must search a building for trapped humans. Each member of the team has similar motor and sensing capabilities, and they communicate via 802.11b wireless Ethernet. This task requires that the robots explore the environment while maintaining network connectivity among the team. If one robot senses a human nearby, it may wish to relay a real-time video stream of the scene back to human operators, through some communications hub. This means that the robot must initiate a connection through the network back to the hub that has certain quality of service (QoS) requirements necessary for real-time video. These requirements could include: minimum bandwidth, length of connection, and maximum allowable jitter, among others. Once a route for the QoS flow has been established, we can guarantee connectivity along the route by controlling the motion of the robots that are routing. At the same time, the robots should be able to address other tasks concurrently. We present a distributed control architecture that allows the robots to address secondary objectives while maintaining routing connectivity.

Unlike most common mobility models used in ad hoc network research [4], the mobility model in this setting assume that the nodes are able to actively move in service to network-related goals. The nodes can reconfigure themselves to achieve a configuration that is better-suited for the current network task, which, in this case, is QoS routing. From the networking perspective, this controllable mobility model makes the ad hoc networking problem simpler, since the system is able to actively reduce the chance of a routing fault, as opposed to other mobility models, where faults are a common occurrence. In fact, it is the presence of network routing faults caused by mobility that makes ad hoc routing difficult. However, from the robotics perspective, the distributed control of a large system, which must concurrently address multiple goals (in this case QoS routes) is a very interesting problem.

However, even with a controllable mobility model, routing faults may still occur. For example, a robot in a flow may not be able to continue routing if it needs to move outside the transmission range of its neighbors in order to address a higher priority task. Or its battery level may be too low to continue transmitting at the specified bit rate. The benefit of being able to control mobility of the nodes and be aware of their internal state is that a routing fault can be predicted before any connection is broken. Using this fact, the team can actively plan to reroute traffic so that no loss of service is detected. The QoS Hand Off Protocol (QHOP) for that process is developed and presented in Section 4.

2 Related Work

2.1 Multi-Robot, Multi-Objective Controllers

In previous work, we have developed an architecture for multi-robot control that allows behaviors to be constructed by combining closed-loop controllers via null space projections [21, 22]. Like the subsumption architecture of Brooks [2], behavior is specified by combining lower-level controllers. In the case of subsumption this combination is performed via inhibition and excitation networks. However, this approach requires the system programmer to take into account all possible control interactions at the level of their interconnection.

Burridge et al. [3] describe a scheme for robot control based on the sequential composition of Lyapunov stable component controllers. Each controller works to bring the robot to a state that is within the domain of the next controller. Because of the Lyapunov stability of the constituent controllers, the robot is guaranteed to reach the goal state given that it starts with the domain of a controller. Although our controllers are not required to be Lyapunov stable, they are "best-effort" in the sense that the closed-loop response is to reject perturbations to the system to the best of their ability.

The filter cascade method [23] for multi-objective control handles multiple objectives by creating sets of filters that act to progressively reduce the input space, which is then ranked by an objective function to determine the next control action. Pirjanian [16] presents a method for multi-objective control where the control actions are determined by optimizing a weighted, linear objective function. Dias [8] describes a market-based method for multi-robot control, where robots assign values to tasks and "trade" them with other robots based on their ability to complete the tasks.

2.2 QoS in Mobile Ad Hoc Networks

Routing in ad hoc networks that takes quality of service into account has received considerable attention in the literature [5, 12, 6]. Most of the research has examined how QoS-aware routes can be formed in ad hoc networks. Zhu [24] presents a TDMA QoS routing protocol that is based on the Ad Hoc On-demand Distance-Vector routing (AODV) [13] protocol. In this paper we assume that such a capability is available to the robots, and in our simulation we use a form of AODV that has been modified to support bandwidth-aware QoS routes [15].

Location-based routing in ad hoc networks has been presented in the DREAM protocol [1], which uses location information to inform routing decisions. Shah [20] also describes a location-based method for proactive routing that takes QoS into account. Their protocol is based on an update protocol that informs nodes of their locations at a certain frequency. Nodes then try and predict the future locations of nodes and use this prediction to form new routes if necessary.

Goff et al. [10] propose a scheme for preemptive routing; nodes keep track of the received signal strength of their routing neighbors, and if it drops below a threshold, then path discovery is initiated. This allows the node to find a new path before the current path fails. If the signal strength is low, then the assumption is that it will continue to decrease and a routing failure will occur. Additionally, with most routing schemes, it takes several timeouts to detect a path failure before a new route discovery is even initiated. Thus, proactively finding a new path will reduce data latency. Our approach is similar in that we wish to take advantage of local state information in order to make routing decisions before failures occur. We have the advantage of knowing the movements of the nodes, so that if signal strength were to temporarily decrease, due to small scale fading effects, we can examine the desired motion of the robot to know whether to expect the signal strength to continue to decrease or whether to initiate route discovery.

3 Distributed Multi-Robot Control

Let R be a team of n robots with heterogeneous sensing capabilities, but similar wireless network equipment. In this system, robots may address secondary tasks while participating in routing a QoS flow. For example, some robots may be involved in routing more than one QoS flow simultaneously. Both tasks generate motor commands for the robot to move so that it stays connected to each routing neighbor, but there must be some way to coordinate those actions so that they do not destructively interfere with each other.

Quality of service is a guarantee that a specified level of service will be maintained on a given connection. There are a number of different aspects of a connection that can be specified: minimum bandwidth, minimum connectivity time, and maximum jitter, among others. In this paper, we are only examining the minimum bandwidth and connectivity quality of service parameters, meaning that our controllers act to ensure connectivity and the appropriate bandwidth is available between routing hosts. Hereafter, a "QoS flow" refers to a connection between nodes that has a specified minimum level of bandwidth and routing connectivity from source to destination.

3.1 Control Basis Approach

Controllers used in this work are constructed using the control basis approach [7]: a controller $\phi_{\mathcal{E}}^{\mathcal{S}}$ is constructed by associating a state estimator, $\mathcal{S}$, and effectors, $\mathcal{E}$, with an objective function, or artificial potential, ϕ. For example, a control task that a robot within the team may perform is a search task. This requires the robot to use its local sensors to map obstacles and goals into its local map. The search controller is

$$\phi_i^S, \tag{1}$$

where robot i achieves search goal states S by greedy action on ϕ. The robot running the search controller is also assumed to be the robot to initiate a QoS flow, which it uses to relay sensor data relating to its search.

The other control task examined in this paper is QoS preservation. Given a pair of robots, $i, j \in R$, and an existing QoS flow f, which specifies a certain minimum bit rate b be available between i and j, then there is a defined area of the environment where i and j can communicate at that bandwidth. We need to calculate that area as a goal set for a "QoS controller."

3.2 RF Channel Model

To motivate the calculation of the QoS goal area, we give a brief description of the model of the transmitters and receivers in our ad hoc network. The RF signal captured at a receiver is made up of a superposition of transmitted signal and its reflections off other objects. Depending on the relative phase of these *multipath* waves, the interference can be constructive or destructive. The *path loss* is a measure of the signal attenuation from the transmitter to the receiver. Large-scale path loss is used to describe the loss in signal due to propagation over large distances and can be modeled as

$$P_r = \frac{kP_t}{d^n}, \tag{2}$$

where P_r and P_t are the signal powers at the receiver and transmitter, respectively, k is a constant related to the antenna gain, d is the distance between the transmitter and receiver, and n is typically between 2 and 4 [18]. Small-scale path loss, or *fading*, is a rapid fluctuation in signal quality over a short period of time, caused by reflections off objects, Doppler effects, and the relative movement of the receiver, transmitter, and objects in the environment. The total path loss is calculated as the sum of the large and small scale path loss. The effect of fading on the signal is typically modeled using a Ricean distribution,

$$p(r) = \frac{r}{\sigma^2} e^{-(\frac{r^2}{2\sigma^2}+K)} I_0(2Kr), \tag{3}$$

where r is the received signal power, K represents the peak amplitude of the dominant signal and $I_0(\cdot)$ is the modified Bessel function of the first kind and zero-order [18]. The K parameter denotes the effect of the line of sight portion of the signal.

Thus we see that both line of sight and distance between transmitter and receiver affect the received signal strength. We construct our controllers such that the large scale path loss is minimized by the QoS controller, and the LOS controller acts to minimize the fading of the signal.

We assume that robots in R are equipped with 802.11b transceivers. The 802.11b specification allows for multirate transmission, at 11, 5.5, 2, or 1 Mbps based on the received signal strength. The standard does not specify an algorithm for selecting the current transmission rate [11]. Although

11 Mbps	5.5 Mbps	2 Mbps	1 Mbps
25 m	35 m	40 m	50 m

Table 1: Nominal transmission rates as a function of distance, for the Orinoco transceiver, in an enclosed environment.

the details of how the transmission rate is calculated is beyond the scope of this paper, given the above model, the signal strength is related to the distance and line of sight properties between transmitter and receiver. Furthermore, we assume that the greater the received signal strength, the higher the allowable bit rate. There is research into multi-rate adaption algorithms [19], but for our purposes we adopt the specifications given by the manufacturer of our transceiver, the Orinoco brand PC Card adapter [17]. Table 1 shows the data rates as a function of receiver distance in an enclosed area, taken from [17].

3.3 QoS Controller

Given the above channel model, the possible bandwidth between a pair of robots is a function of the distance between the pair and whether they are within LOS. Let the transmission range on robot i for bandwidth level b be denoted r_b^i. The QoS goal set $\mathcal{Q}_{ij}^f$ is computed as

$$\mathcal{Q}_{ij}^f \quad = \quad LOS_{ij} \wedge Q_{ij}^f, \tag{4}$$

where LOS_{ij} is the LOS region between i and j, and Q_{ij}^f is a circle with radius $\min\{r_b^i, r_b^j\}$, which is the region where i and j can communicate with bandwidth b. Both LOS_{ij} and Q_{ij}^f are centered about robot i. For j to compute $\mathcal{Q}_{ij}^f$, i must send the pair $\langle LOS_{ij}, r_b^i \rangle$.

The shape of LOS_{ij} depends on the sensing abilities of robot i. For example, if i has only local proximity detection sensors, then i may only be able to sense freespace within 1 m of the robot, which may be a small subset of the QoS area. If robot i uses a high-resolution camera for determining the LOS region, then it may be able to specify a large LOS region that subsumes multiple levels of QoS bandwidth. If a map of the environment is known, then a robot can calculate the true LOS region between it and its peer, which may be greater than the range of its sensors. This allows a longer range, with less bandwidth used between robots to communicate the QoS area.

Given the QoS goal set $\mathcal{Q}_{ij}^f$, the QoS controller is

$$\phi_j^{\mathcal{Q}_{ij}^f}, \tag{5}$$

where robot j must achieve QoS-maintaining states $\mathcal{Q}_{ij}^f$ by greedy action on ϕ.

3.4 Coordinated, Distributed Controllers

So far we have described two controllers for accomplishing two distinct tasks, searching and QoS maintenance. Robots involved in routing a flow may need to perform multiple tasks concurrently. A method is needed for eliminating destructive control interactions, in other words, allowing both controllers to run concurrently while ensuring they both continue to make progress towards their goals.

A team of mobile robots can be viewed in a control framework as a redundant system with many degrees of freedom. We use the control framework presented in [21, 22]. The system Jacobian for the team may be redundant, which allows secondary tasks to be addressed by projecting secondary control actions onto the null space of the Jacobian. This requires that all addressable tasks be arranged by priority, in increasing order, so that subordinate tasks may be projected onto the null space of superordinate tasks. Each robot locally determines its prioritization of tasks, which may be based on local state information. For example, a robot may prioritize tasks based on the expected probability of success, which it calculates from local state such as its battery level.

Since control actions are derived by descending artificial potentials, a secondary control action will not interfere if it moves the robot along an equipotential line of the primary control potential. In other words, if we project secondary control actions onto the null space of the primary controller, then the new action does not interfere with the primary control action. The subject-to operator "$\triangleleft$" performs the null space projection. For example, a robot i that is maintaining a QoS flow with robot j may also wish to perform a search concurrently. In this case, QoS-maintenance is the primary task, so all search actions are projected onto the null space of the QoS controller. The combined, pairwise controller is written as

$$\phi_i^S \quad \triangleleft \quad \phi_j^{\mathcal{Q}_{ij}^f}. \tag{6}$$

This controller will allow i to search while j maintains QoS by moving to $\mathcal{Q}_{ij}^f$. The robots move as

a pair where robot i is the leader, since it is specifying $\mathcal{Q}_{ij}^f$ to j. While "following" i, robot j may address secondary objectives using the null space projection. By cascading null space projections, subordinate control actions can be addressed.

In addition to two robots, multiple robots can form a serial, kinematically-related chain by combining QoS controllers. Let robots i, j, and k be a chain which routes a flow from i to k. Robot i is the source of the flow and executes the controller in equation (6), while the middle robot j is involved in a pairwise pull controller with both neighbors:

$$\phi_j^{\mathcal{Q}_{ij}^f} \quad \triangleleft \quad \phi_k^{\mathcal{Q}_{jk}^f}. \tag{7}$$

This combination of controllers (6) and (7) allows the three-robot chain to move about the environment while maintaining QoS connectivity. For an arbitrary QoS flow f through R, each robot along the route of f must execute a QoS controller with its up- and down-stream neighbors. The chain constrains the movements of its members to within those areas that maintain QoS requirements for f. The maximum number of robots that can form such a QoS chain depends on the real-time process requirements of the robots. This bound is discussed in more detail in [22].

3.5 QoS Threshold

Assuming robots i and j are executing the controller in (6), if j is not within the QoS region $\mathcal{Q}_{ij}^f$, then a routing fault will occur if the specified bandwidth for f cannot be achieved between the robots. Consequently, j may predict when a routing fault will occur as it approaches the boundary of $\mathcal{Q}_{ij}^f$. If j cannot continue routing f, we would like to able to recruit some other member of R to take its place in routing the flow. For this to occur, j must act to recruit a replacement before the fault occurs. We define the QoS threshold region $RF_{ij}^f \subseteq \mathcal{Q}_{ij}^f$ that will serve as an enlarged routing-fault boundary. Upon entering RF_{ij}^f, robot j should initiate the QoS routing hand off protocol described in Section 4. The size of RF_{ij}^f should vary depending on the speed of j; RF_{ij}^f should increase as j moves faster, since more time is required to perform a routing hand off.

It may be unavoidable for a robot to move into the RF area; for example, if it must avoid a dynamic obstacle. The robot can decide, based on its desired movement vector, whether it will remain in the RF area, or it returns to the QoS goal area. By having this local state, unnecessary rerouting can be avoided.

4 QoS Flow Hand Off

The robot team has a certain network topology based on connectivity and QoS capabilities. We are interested in determining how the team may reconfigure itself to address new tasks. For example, given a QoS flow f, some participant j may not be able to continue servicing f, in which case another robot could be recruited to take over for j. Situations that could cause such a failure include a node's battery level falling below the level at which it needs in order to transmit at the given QoS, or it may have another, more important, objective to address that keeps it from moving into the desired region, $\mathcal{Q}_{ij}^f$, necessary to continue routing.

It is desirable if another robot with available resources can join in the existing flow to take the place of the failing member. A protocol is needed to determine which team member will take over, and how the hand off should occur so as to make sure that QoS of the flow is maintained. We have developed the QoS Hand Off Protocol (QHOP) to implement this functionality.

This protocol may be used on a network with a fixed topology as well. For example, a node may hand off routing to another node that is on the same LAN segment, or if using wireless, then it can reroute to a neighboring node.

4.1 QoS Hand Off Protocol (QHOP)

In this paper, we assume flows have only a minimum bandwidth and connectivity time QoS specification, written as

$$\vec{Q}_f \quad = \quad (b, t_f)^T, \tag{8}$$

where f is the flow, b is the minimum bandwidth, and t_f is the estimated time the flow will end. We assume that the time parameter can be adjusted during the flow by sending an update to each routing host. This way each host knows how long it may have to maintain QoS connectivity for f. The basic mechanism for flow hand off works for any sort of QoS property. The protocol works for non-QoS flows as well, since they can be specified as QoS flows with a requirement of zero minimum bandwidth. Table 2 shows all the messages used in QHOP.

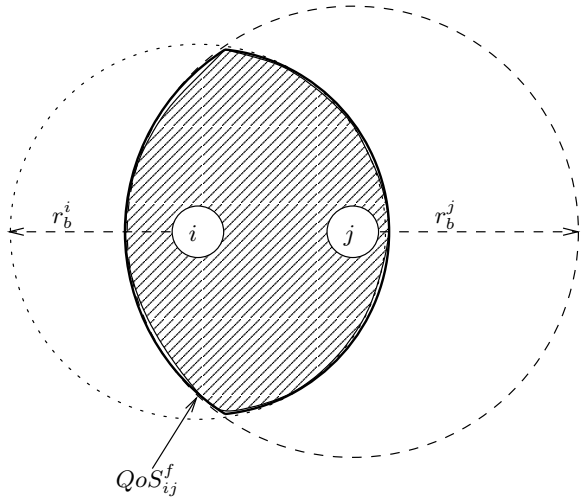

Figure 1: The QoS region for two robots.

We assume that all the robots in R form a connected tree and have some network-level ad hoc routing protocol in place that will allow the team to find acceptable routes for QoS flows. There are a number of ways to do QoS routing in an ad hoc network [5, 14, 13]. The following protocol is described at the network layer, using dynamic methods similar to AODV that do not require global state information. In our simulation we assume that the robots are using the AODV routing protocol [13] with QoS extensions [15]. It is also necessary for the team to share a global coordinate frame and map, so that they can navigate to other robots when a hand off is required.

Let f be an existing QoS flow, with minimum bandwidth b, where $R_f \subseteq R$ is the set of robots participating in the routing of f. Let $s, d \in R_f$ be the source and destination of f, respectively. Assume that at some point while f is active, robot $j \in R_f$ determines that it will not be able to continue providing the QoS level required by f.

Each instance of a hand off procedure has its own unique identifier, $HID = \langle FID, SEQ_H \rangle$, where FID is a flow ID and SEQ_H is a hand off sequence number generated by j each time it initiates the hand off procedure. The flow ID is the pair $FID = (s, SEQ_s)$, where SEQ_s is an increasing sequence number generated by s for every QoS flow it initiates. It is assumed that every member of R_f is told $\langle s, SEQ_s \rangle$ when the routing of f is set up, and this is stored along with other necessary routing state for f at each host. The hand off ID, SEQ_H, allows j to try the hand off procedure multiple times for a given flow.

To start the process, j generates a `REQUEST` (`HREQ`) message which is flooded throughout the network. Besides the hand off ID the `HREQ` message also contains the QoS parameters of the flow, $\vec{Q}_f$, the addresses of the up- and down-stream neighbors of j, as well as the position of j in world coordinates, $\vec{x}_j$. It also contains a timestamp of when it was sent, t_{sent}. The `HREQ` message is flooded to every robot R, with each robot recording the robot from which it is first received, so that replies can follow the reverse path back to the sender. Robots should keep this information for a time of `REQ-TIMEOUT`, which is on the order of seconds.

When a robot $h \in R$ receives a `HREQ` message, it should be able to determine, based on its local state and the information in the message, whether it is capable of taking over the routing of f for robot j. The factors that h must consider include:

- Can h provide the QoS required in f? Does h have enough power and hardware capability, or do h's other tasks allow it to move within range of f?
- The Cartesian distance to j. Can it find a path to j?
- Can h move close enough to j to take over in the flow?
- Will f still exist by the time h is able to per-

Message Name	Body	Abbreviation	Sender	Receiver
REQUEST	$\langle HID, \vec{Q}(f), i, k, \vec{x}_j, t_{sent}, t_f \rangle$	HREQ	j	R
REPLY	$\langle HID, h, \alpha_h, \vec{x}_h \rangle$	HREP	h	j
ACCEPT	$\langle HID, h, \vec{x}_j \rangle$	HACC	j	R
ACCEPT-ACK	$\langle HID \rangle$	HACCA	h	j
GOALSET	$\langle HID, QoS^f_{ij} \wedge QoS^f_{jk} \rangle$	HGLS	j	h
NOTIFY	$\langle HID, j \rangle$	HNOT	h	i, k
NOTIFY-ACK	$\langle HID \rangle$	HNOTA	i, k	h
COMPLETE	$\langle HID \rangle$	HCOMP	h	j

Table 2: The messages used in the hand off protocol.

form the hand off?

The last point illustrates one constraint upon this type of problem, in that the timescales between events in this system differ by a few orders of magnitude. For example, it is reasonable to assume that any sort of network-level routing changes may take on the order of milliseconds, to hundreds of milliseconds to occur. For example, in a typical ad hoc routing scheme, when a routing fault occurs, the delay before a new route is set up is determined by how long it takes the appropriate routing messages to be transmitted through the network. However, for a robot to move from a start position to a position that allows the robot to takeover that flow will take on the order of seconds to minutes, depending on the mobility of the robot. Thus, for QHOP to be a useful protocol, a necessary assumption is that the QoS flows that must be rerouted have a relatively long duration.

Using some algorithm, which is not specified by the protocol, h computes an 8-bit integer ranking, α_h, where 0 means it cannot perform the take-over, and 255 means that it is certain that it can. The ranking is a sort of probability of success in performing the take-over. If $\alpha_h > 0$, then h sends a REPLY (HREP) message which is forwarded on the reverse path back to j. The HREP message contains the flow and hand off identifiers, the hand off ranking, α_h, as well as the position of h in world coordinates, $\vec{x}_h$.

The initiator of the hand off, j, must wait for HREP messages. It is up to j to choose the node that will take over its position, the "successor" node, based on the available nodes which have sent HREP messages. The process by which j chooses a successor is not specified by the protocol, but j may wait until the REP-WAIT-TIMEOUT expires for replies. A suitable heuristic for choosing a successor may be to pick whichever nodes replies first, or wait for a given number of replies, or until the timeout, and choose the node with the largest ranking α_h.

Once a successor h has been chosen, j sends the ACCEPT (HACC) message, which is flooded to all nodes. HACC contains the name of the successor node, as well as j's current position, which may be different than the position given in the initial HREQ message. HACC is flooded throughout the network so that all possible successors know that h has been chosen, and so that nodes other than h know to erase any forwarding state for the hand off. On receiving the HACC message, successor h must reply with an ACCEPT-ACK (HACCA) message back to j. After sending HACC, j sets the timer ACCEPT-TIMEOUT. If this timer goes off before HACCA reaches j, then j can resend the HACC once more. If that times out as well, then j may restart the hand off process.

After sending HACCA, h begins moving toward the position $\vec{x}_j$ given in the HACC message. At this point, h may lose connectivity with the team as it moves toward j. Due to the dynamic nature of the environment, it is very difficult for a robot to guarantee that it will be able to move into the correct area required for a routing hand off. In real-world applications, this may be the most common cause of failure. A keep-alive querying mechanism can be used by j to check on the status of h as it moves to the hand off position.

If h fails to reach j before GOALSET-WAIT-TIMEOUT expires, then the hand off process ends, and j can choose to reinitiate the protocol. Robot h can keep its own timer to determine if it has failed to reach j in time. Or, once it is reconnected to the network, it may receive a newer HREQ message from j, at

which point h may be in a better position to retry the hand off.

Assuming that h is able to reach j in time, then once h is within transmission range of j, j sends a `GOALSET` message. (We assume a lower level capability that allows the robots to determine the absence or presence of neighbor nodes.) This message contains the information needed for h to calculate the QoS goal set area that j is using, in world coordinates. Robot h computes the QoS goal region using $\min\{r_b^j, r_b^h\}$. Figure 2 shows a configuration of robots involved in the hand off, and the goal set that would be transmitted from j to h. This is the area where j can maintain QoS with both its neighbors. Robot h also knows the up- and down-stream neighbors in the flow, and so it knows it can successfully perform a hand off if its bandwidth to these neighbors is greater than the bit rate specified in the `HREQ`.

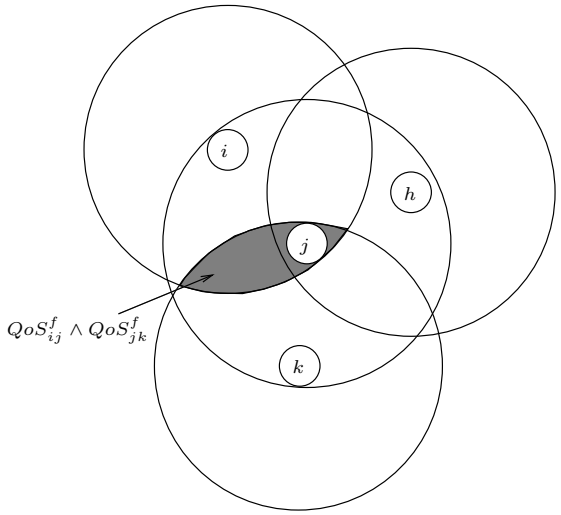

Figure 2: The goal set region transmitted in `GOALSET` from j to h.

Using the new goal set, h moves into a region within the goal-set, which ensures it has connectivity with j's up- and down-stream neighbors, i and k, respectively. Once connectivity has been established between h and the down-stream neighbor, k, node h sends the notification message to k, `NOTIFY` (`HNOT`). This message contains the name of the node that is leaving the flow, j, and notifies k that h will take its place at some point. Node k should prepare to begin accepting packets for the current flow that had been coming from j to come from h. So all that is changing is k's routing table for packets from j for flow f.

After receiving `HNOT`, node k sends an acknowledgment message back to h, `NOTIFY-ACK`. If h does not receive the acknowledgment before the timer `NOTIFY-TIMEOUT` expires, then h will resend `HNOT`. Node k will continue to receive packets from j, and will continue to forward them until it receives the hand off `COMPLETE` (`HCOMP`) message from h, signifying that j is no longer part of f. Also, any packets that are moving upstream, from the destination d to s, will reach k, at which point they will be routed to h instead of j. After sending `HNOT` to k, h should ensure that it has buffer space allocated for the packets from k, which it will forward when the hand off is complete.

After notifying the downstream neighbor, h notifies the upstream neighbor by sending the same `HNOT` message to i. Upon reception of `HNOT`, i must change its routing table to send packets for f to h instead of j. Robot i acknowledges the receipt of the `HNOT` message by sending a `HNOTA` back to h. Note that until i receives `HNOT`, j is still routing packets from i to k for f. It is assumed that compared to the routing task itself, handling `HNOT` messages will not cause i or k to disrupt f so as to lose the QoS level.

Robot h completes the protocol by sending `HCOMP` messages to i,j, and k. At this point, the routing hand off has completed, and packets for f are being routed from i through h to k. Node h can forward along to i any upstream packets it had received from k and buffered earlier. When robot j receives the `HCOMP` message it can remove any routing state associated with f.

5 Evaluation

5.1 Simulator Platform

In order to simulate a robotic, mobile, ad hoc network, we must simulate the appropriate network conditions as well as the control of the robots. To achieve this, we merged the simulation capabilities of a network simulator, ns-2, and a mobile robot simulator, Player/Stage [9]. The Player/Stage simulator simulates the mobility and sensing aspects of the robot team, while ns-2 simulates the ad hoc network traffic and QoS capabilities of the robots.

We have implemented the QoS controllers in Player/Stage, and the QHOP protocol for 802.11b

wireless nodes in ns-2. The simulators are coordinated via shared memory so that at each simulated time step, ns-2 nodes receive position information from their counterpart robots in Player/Stage. Using this position information, ns-2 simulates the routing performance and current bandwidth level, which is passed back to the Player/Stage nodes. The Player/Stage simulated robots adjust their position based on the network information from ns-2.

This work assumes that the robots have some sort of QoS ad hoc routing capabilities in place. For our evaluation we developed a QoS ad hoc routing protocol based on AODV, described in [13].

5.2 QHOP Performance

To examine the performance of QHOP, we applied it to a situation with a fixed ad hoc networking topology. In this case a node can immediately determine whether it can perform a hand off by checking whether it has the required bandwidth to the up- and down-stream neighbors. In this case, the time to complete a QHOP process depends on propagation and processing delays, which are on the order of hundreds of milliseconds. However, with the mobile robot case, the time it takes to complete the rerouting will depend on how long it takes for the robot to reach the goal region and takeover routing duties. This can range from seconds to minutes, depending on the environment and mobility of the robot.

For a given random network topology, the success of a routing hand off will depend on the locations of the nodes. Figure 3 shows a graph of the success at performing a hand off versus the number of nodes in the environment. In this case, the size of the environment is fixed at 300 m $\times$ 300 m, and each node remains in its initial position for the entire simulation. Each data point represents the percentage of flows that were successfully rerouted by QHOP. For each run, 20 routes were created with the specified bandwidth of 5.5 Mbps, and then QHOP was started at some point during the flow. The probability that a route can be successfully handed off depends on the likelihood of a neighbor residing close enough to fulfill the bandwidth requirements. As the density of the nodes within the environment increases, the chance of success in handing off will also increase. Likewise, as the bandwidth requirement decreases, the maximum allowable distance between hops will increase, thus increasing the chances of successfully handing off the connection.

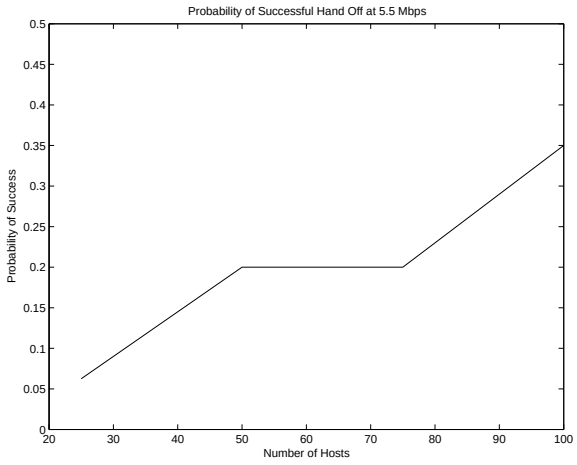

Figure 3: The chance of successfully completing a QHOP hand off as a function of the number of nodes in a 300 m $\times$ 300 m environment at a rate of 5.5 Mbps.

The routing overhead of QHOP is measured as the number of packets that are sent by all nodes in response to a QoS hand off request. Since QHOP relies on flooding for distributing the request and hand off acknowledgment messages, the total number of packets sent is a linear function of the number of nodes in the connected network, and is not related to the length of the flow being rerouted. Figure 4 shows the relative overhead of the QHOP protocol as a function of the number of nodes in the network. Each point is the ratio of QHOP-related packets to the total packets sent during one flow with a constant bit rate of 70 kbps for 20 seconds. The same route is used for each point. The number of data packets sent remains constant, but as the size of the network grows, each flooding operation results in more QHOP-related packets. Figure 5 shows the same overhead ratio given a fixed route in a 50 node network. At each point a different data rate is chosen, but the network size remains fixed, so more data packets are sent relative to the number of QHOP-related packets sent.

5.3 QoS Control with Hand Off

To demonstrate the QoS controllers and QHOP protocol, we performed a simulation with four robots in a 10 m $\times$ 10 m environment. Each robot is a differential driver mobile robot with a ring of IR obstacle detection sensors. The leader initiates an 11 Mbps route between three robots, and the robots must try to maintain QoS and LOS constraints while moving in the environment. At a

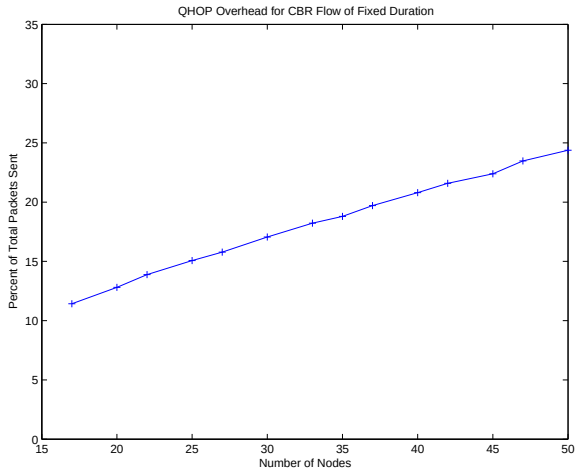

Figure 4: This shows the ratio of QHOP-related packets sent to other packets sent in a flow with a 70 kbps bit rate, as a function of the number of nodes in the connected network.

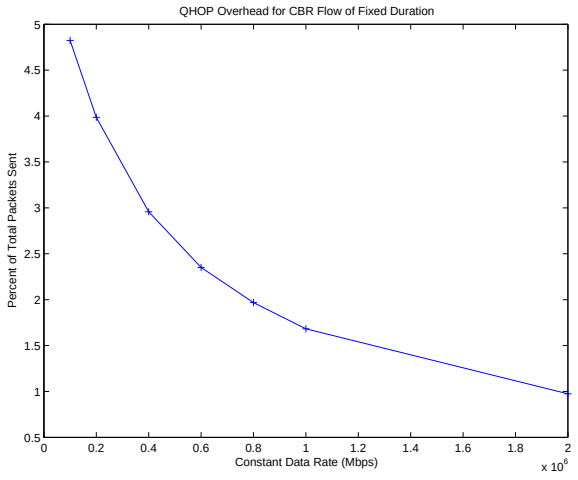

Figure 5: This shows the ratio of QHOP-related packets sent to other packets sent in a flow with a fixed route in a 50 node network, as a function of the bit rate of the flow.

certain time the middle robot becomes unable to continue QoS routing and initiates a QHOP hand off. Once the routing hand off occurs, the robots resume their formation. The controllers can be written as

$$\phi_0^S \quad \triangleleft \quad \phi_1^{\mathcal{Q}_{01}^f} \tag{9}$$

$$\phi_1^{\mathcal{Q}_{01}^f} \quad \triangleleft \quad \phi_2^{\mathcal{Q}_{12}^f}, \tag{10}$$

where equation (9) is the coordinated search-QoS controller for the leader, Robot 0, and its follower, Robot 1. Equation (10) describes the controller used to maintain QoS between Robot 1 and the tail of the chain, Robot 2.

For the purposes of simulation, the QoS region was set at 0.5 m, meaning that the robots must be within 0.5 m of each other in order to maintain an 11 Mbps connection. The LOS region was also set at 0.5 m, which corresponds to the maximum sensing distance of the IR obstacle detectors used on the robot. Figure 6 shows the paths of the four robots in the environment. The arrow indicates the starting position of Robot 3, the robot that performs the QHOP hand off. In this example, the start of the hand off process was chosen arbitrarily. Figure 7 shows the differences in distance between the four robots as they execute the task. The maximum QoS and LOS distance is marked with a dashed line. The discontinuity represents the position where the robot that performs the routing hand off, Robot 3, joins the LOS chain and starts to follow Robot 0. In this example, there were no boundary regions for the LOS and QoS goals, which results in the two follower robots periodically exceeding the QoS and LOS boundary. This also illustrates the best-effort nature of the controller; as soon as the goal region is lost, the robot acts to return to the goal region.

6 Conclusion and Future Work

We have presented a set of distributed controllers that maintain QoS commitment between pairs of mobile robots involved in an ad hoc network. The controllers handle multiple objectives by combining control actions through a generalized null space projection. We have demonstrated the controllers in simulation.

We have also presented the QHOP link layer protocol for proactive routing path in response to possible routing faults in mobile ad hoc networks. This protocol acts to find new nodes to continue routing when a member of a QoS-constrained flow can not maintain its QoS commitments. The protocol scales linearly with the number of nodes in the connected network. We have demonstrated the use of the protocol in the context of QoS routing with mobile robots.

For future work, we want to extend the QoS estimation model used to determine the QoS bound-

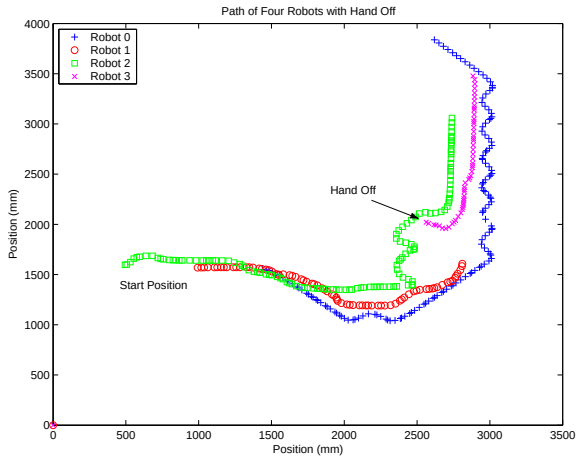

Figure 6: This graph shows the paths of the four robots. They start in a linear configuration near the middle of the workspace. The arrow indicates the location of the robot that performs the routing hand off.

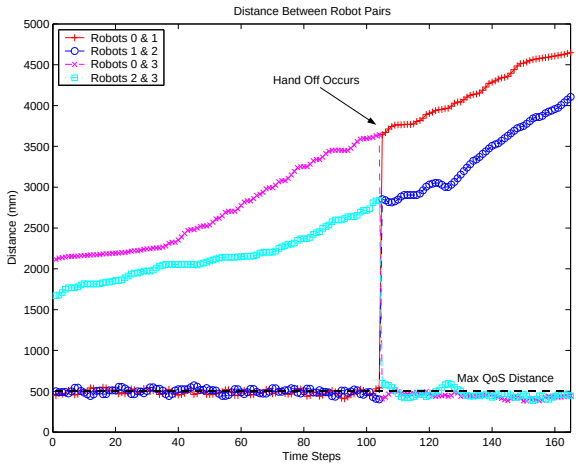

Figure 7: This graph shows the differences in distance between the four robots involved in the task over time. The dashed line shows the maximum QoS distance. The discontinuity shows the time at which the hand off occurs, and the new robot joins the LOS chain.

aries between hosts. Currently a simple distance-based estimate is used, but this could be extended by building an estimation model based on sampled signal strengths from local and neighboring nodes. We also plan to implement this system on a team of mobile robots.

References

[1] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward. A distance routing effect algorithm for mobility (DREAM). In *Proceedings of MOBICOM '98*, pages 76–84. ACM, 1998.

[2] R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, March 1986.

[3] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Sequential composition of dynamically dexterous robot behaviors. *The International Journal of Robotics Research*, 18(6):534–555, June 1999.

[4] T. Camp, V. Davies, and J. Boleng. A survey of mobility models for ad hoc network research. *Wireless Communication & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.

[5] S. Chen and K. Nahrstedt. Distributed quality-of-service routing in ad-hoc networks. *IEEE Journal on Selected Areas in Communications*, 17(8), August 1999. Special Issue on Ad-Hoc Networks.

[6] T.-W. Chen, J. Tsai, and M. Gerla. QoS routing performance in multihop, multimedia, wireless networks. In *Proceedings of ICUPC '97*, 1997.

[7] J. Coelho and R. Grupen. A control basis for learning multifingered grasps. *Journal of Robotic Systems*, 14(7):545–557, 1997.

[8] M. B. Dias and A. Stentz. Opportunistic optimization for market-based multirobot control. In *Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2002.

[9] B. Gerkey, R. T. Vaughn, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th International Conference on Advanced Robotics*, June 2003.

[10] T. Goff, N. B. Abu-Ghazaleh, D. S. Phatak, and R. Kahvecioglu. Preemptive routing in ad hoc networks. In *Proceedings of SIGMOBILE '01*. ACM, July 2001.

[11] IEEE. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: Higher-Speed Physical Layer Extension in the 2.4 Ghz Band*, 1999. IEEE Std 802.11b-1999.

[12] S.-B. Lee, G.-S. Ahn, X. Zhang, and A. T. Campbell. INSIGNIA: An IP-based quality of service framework for mobile ad hoc networks. *Journal of Parallel and Distributed Computing*, 60:374–406, 2000.

[13] C. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *MILCOM '97 panel on Ad Hoc Networks*, November 1997.

[14] C. E. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[15] C. E. Perkins, E. M. Royer, and S. R. Das. *Quality of Service for Ad Hoc On-Demand Distance Vector Routing*. IETF Mobile Ad Hoc Networking Group, July 2000. draft-ietf-manet-aodvqos-00.txt.

[16] P. Pirjanian and H. I. Christensen. Behavior coordination using multiple-objective decision making. *Sensor Fusion and Decentralized Control in Autonomous Robotic Systems*, 3209:78–89, October 1997.

[17] Proxim Corporation. *ORiNOCO 11b Client PC Card*, 2003.

[18] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice-Hall PTR, second edition edition, 2002.

[19] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly. Opportunistic media access for multirate ad hoc networks. In *Proceedings of MOBICOM '02*. ACM, September 2002.

[20] S. H. Shah and K. Nahrstedt. Predictive location-based QoS routing in mobile ad hoc networks. In *Proceedings of IEEE International Conference on Communications (ICC 2002)*. IEEE, 2002.

[21] J. Sweeney, T. Brunette, Y. Yang, and R. A. Grupen. Coordinated teams of reactive mobile platforms. In *Proceedings of IEEE International Conference on Robotics and Automation*. IEEE, 2002.

[22] J. Sweeney, H. Li, R. Grupen, and K. Ramamritham. Scalability and schedulability in large, coordinated, distributed robot systems. Submitted to the International Conference on Robotics and Automation (ICRA) 2003, 2003.

[23] B. J. Thibodeau, S. W. Hart, D. R. Karuppiah, J. D. Sweeney, and O. Brock. Cascaded filter approach to multi-object control. Submitted to 2004 IEEE International Conference on Robotics and Automation (ICRA).

[24] C. Zhu and M. S. Corson. QoS routing for mobile ad hoc networks. In *Proceedings of the Joint Conference of the IEEE Computer and Communications Societies*. IEEE, 2002.