A Location-Based Algorithm for Multi-hopping State Estimates within a Distributed Robot Team

Brian J. Julian*†, Mac Schwager*, Michael Angermann‡, and Daniela Rus*

Abstract Mutual knowledge of state information among robots is a crucial requirement for solving distributed control problems, such as coverage control of mobile sensing networks. This paper presents a strategy for exchanging state estimates within a robot team. We introduce a deterministic algorithm that broadcasts estimates of nearby robots more frequently than distant ones. We argue that this frequency should be exponentially proportional to an importance function that monotonically decreases with distance between robots. The resulting location-based algorithm increases propagation rates of state estimates in local neighborhoods when compared to simple flooding schemes.

1 Introduction

Robots in a team need to communicate state estimates to self-organize. Since many applications desire the team to spread over large-scale domains, resulting distances between robots can become larger than their capable peer-to-peer transmission ranges. These configurations require multi-hop networking to distribute state information over the entire system. To facilitate the transportation of data packets in a multi-hop fashion, many mobile ad hoc networks implement sophisticated routing schemes. Due to the mobile nature of such networks, these schemes consume a significant amount of communication capacity for maintaining knowledge about network topology. While some routing strategies take spatial configurations into account, the robots

*Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Cambridge, MA 02139, USA
†Currently working at MIT Lincoln Laboratory, 244 Wood Street, Lexington, MA 02420, USA
‡Institute of Communications and Navigation, German Aerospace Center (DLR), P.O. Box 1116, D-82234 Wessling, Germany
e-mail: bjulian@mit.edu, schwager@mit.edu, michael.angermann@dlr.de, rus@csail.mit.edu

are agnostic to the relevance of the actual data being transferred. There is no concept of data importance from the robots' point of view, often resulting in the suboptimal allocation of communication resources (e.g. time, bandwidth, power) to transfer packets.

The strategy in this paper allows robots to better manage communication resources for relaying state estimates. Since the collaboration of robots takes place in the physical world, spatial relationships between robot states can give insight into the importance of transferring each estimate. This location-based approach gives a quantitative answer to the question: how important is it for one robot to broadcast state information about another robot? We represent the importance of transmitting a state estimate as a function that is inversely proportional to the distance between robots.

From this importance function we develop a deterministic algorithm that ensures state estimates propagate throughout a robot network. The proposed location-based algorithm is efficient in terms of bandwidth and computational complexity; it does not require network topology information to be transmitted or computed. We used Monte Carlo simulations to show increased propagation rates of state estimates in local neighborhoods. Then with real control and wireless hardware, we simulated a nine robot team running a Voronoi coverage controller to show the algorithm's effectiveness in solving distributed control problems. Experimental results for the propagation of state estimates are also presented with five AscTec Hummingbird quad-rotor flying robots and four stationary robots.

A substantial body of work exists on location-based routing for mobile ad hoc networks. Haas proposed a zone-based routing protocol using a radius parameter to reduce the number of control messages [4]. Ni et al. developed a distance-based scheme to decide when a node should drop a rebroadcast [5], while Sun et al. adapted a similar scheme for setting defer times [7]. Ying et al. discussed how these ad hoc schemes influence flooding costs [1].

Our proposed algorithm is related to this body of work in that location is used to broadcast information through a mobile ad hoc network. However, instead of routing actual data packets to a predetermined receiver, we are deterministically transmitting state information to be used by the entire team of robots. This allows all transmissions to be treated as simple broadcasts, for which the sender uses the algorithm to select state estimates. This strategy is applicable for many distributed control problems, such as coverage control algorithms for mobile sensing networks.

2 Importance of Broadcasting State Estimates

A common assumption for distributed control algorithms is that robots have access to state estimates of other nearby robots. This assumption is often translated into unrealistic requirements on communication range. The most common requirement is that estimates need to be directly shared between robots that are within a specified distance. Another common requirement is

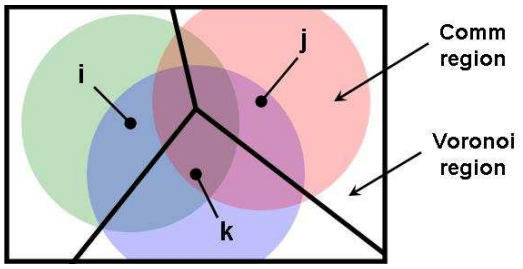

Fig. 1 A simple example where robots i and j share a Voronoi boundary but cannot communicate their state estimates directly. This problem is easily resolved using a mobile ad-hoc network topology to route information through robot k.

for information to be shared between robots of a defined spatial relationship (e.g. adjacent Voronoi regions [2] or overlapping fields of view [6]).

These communication requirements are too simplistic to be realized in practice. Actual network topologies depend on more than simple distance criteria, such as environment geometry, channel interference, or atmospheric conditions. Even if transmission ranges are ideal in the physical sense (e.g. the ideal disk model), spatial relationships for certain distributed controllers cannot guarantee peer-to-peer connectivity. Figure 1 shows a configuration where a direct communication link cannot be created between the Voronoi neighbors i and j. Moreover, robots that are spatially disconnected may decide not to route state estimates to one another. If they move to become spatially connected, the lack of shared data will prevent the robots from learning about their new neighbors. Thus, no new communication links will be established. We are motivated by these serious and unavoidable complications to develop an algorithm that ensures state estimates flow throughout a team of robots.

2.1 Broadcast Scheme

Consider n robots moving in a space[1], $\mathcal{P}$. Each robot, $i \in \{1, \ldots, n\}$, knows its current state, $p_i(t) \in \mathcal{P}$, by some means of measurement (e.g. GPS or visual localization). We propose that each robot maintains a list of state estimates, $[p_1(t_{i1}), \ldots, p_n(t_{in})]$, where t_{ij} denotes a time stamp at which robot i's estimate of robot j's state was valid. We have that $t_{ij} \leq t$ and $t_{ii} = t$. Each robot's state estimate is initialized to infinity to indicate that a valid estimate is lacking, except for its own state which is always current.

We desire to communicate state estimates throughout the robot network. For simplicity, we use Time Division Multiple Access (TDMA)[2] to divide the data stream into time slots of equal length, m. During a time slot, one

[1] Although it is easiest to think of the space being $\mathbb{R}^2$ or $\mathbb{R}^3$, the strategy we describe is equally useful with more detailed state estimates (e.g. velocity, acceleration, joint positions, state machine information, etc.)

[2] In this paper we primarily discuss implementing the proposed strategy using TDMA; however, many other channel access methods are appropriate (e.g. FDMA or CDMA).

assigned robot is allowed to broadcast over the shared frequency channel. Other robots broadcast one after the other in a predetermined order. One complete broadcast cycle is referred to as a frame.

To broadcast its own state estimate once per frame, the robot's time slot must be long enough to transmit the estimate and an associated time stamp. Such a time slot is considered to have length of $m = 1$. Clearly time slots of unit length are not sufficient to transmit information throughout the network; each robot would only be updated with the state estimate of its neighbors on the network. For multi-hop networking, the robots need longer time slots to broadcast the estimates of other robots.

One naive strategy is to assign a time slot length equal to the number of robots, $m = n$, so that each robot can broadcast its entire list of state estimates, thus creating a simple flooding scheme. Robots that are adjacent on the network use this information to update their own list, retaining only the most current state estimates. The process is repeated for each time slot, naturally propagating state estimates throughout the network without the need of a complicated routing protocol.

Although simple to implement, this strategy is not scalable for a large number of robots. Consider the rate a system can cycle through all time slots to complete one frame. This frame rate, r_f, gives insight into how quickly state estimates are being forwarded, and therefore how confident distributed controllers can be in using the estimates. For a network of fixed baud rate, r_b, the maximum frame rate[3] is given by $\max(r_f) = r_b/mnb$, where b is the data size of a state estimate and its associated time stamp. For $m = n$, increasing the number of robots in the system will decrease the frame rate *quadratically*. This inherent trade-off provides motivation to reduce the length of the time slot; however, if a robot cannot broadcast *all* state estimates within one time slot, which estimates are considered more important to broadcast?

2.2 Importance Function

Many distributed controllers are dependent on spatial relationships between robots. When selecting which state estimate to broadcast, the selection process should also depend on these relationships. This makes sense because a robot's state is more likely to be useful to controllers in proximity. However, it cannot be considered useless to controllers that are distant due to the mobile nature of the system. We propose that the importance of robot i broadcasting robot j's state estimate is inversely proportional to the distance between robot states.

Since the robots only have access to the state estimates they receive, a distance estimate is used to give the following importance function

$$f_{ij}(t) = d\left(p_i(t), p_j(t_{ij})\right)^{-\alpha} \tag{1}$$

[3] We are ignoring overhead associated with TDMA (e.g. guard periods, checksums, etc.)

where $d(\cdot,\cdot) \geq 0$ is a distance function and $\alpha \in (0,\infty)$ is a free parameter, both of which are selected for the given distributed controller. For example, a Voronoi coverage controller dependent on linear spatial separation may use a Euclidean distance function with $\alpha = 1$. This same distance function is appropriate for a sensor-based controller dependent on light intensity, although $\alpha = 2$ may be used since light intensity decays quadratically with distance from the source. Conversely, the distance function does not need to be Euclidean or even of continuous topology, such as for truss climbing robots with a finite configuration space. In any case, a robot should consider its own state estimate to be the most important to broadcast. This is reflected in the model since f_{ii} is infinite for any valid $d(\cdot,\cdot)$ and α.

3 Location-Based Algorithm for Broadcasting States

We use the importance function in Equation (1) to develop a deterministic algorithm. For a given time slot, this algorithm selects which state estimates a robot will broadcast. We first describe a probabilistic approach to help formulate the final algorithm.

3.1 Probabilistic Approach

Consider a robot that needs to select m state estimates to broadcast during its time slot. We provided motivation in Section 2.2 that some selections are more important than others. However, the robot should *not* systematically select the state estimates associated with the highest importance; doing so can prevent estimates from fully dispersing throughout the system. Instead, we propose that the probability of robot i selecting the state estimate of robot j is

$$P^{ij}_{\mathcal{M}_i}(t) = \frac{f_{ij}(t)}{\sum_{k\in\mathcal{M}_i} f_{ik}(t)}, \quad j \in \mathcal{M}_i \tag{2}$$

where $\mathcal{M}_i$ is the set of robot indices associated with selectable estimates.

Prior to the first selection for a given time slot, $\mathcal{M}_i$ is the set of all robot indices. From the full set the robot always selects its own state since it has infinite importance. The robot then removes its index from $\mathcal{M}_i$ to prevent wasting bandwidth. Since Equation (2) is a valid probability mass function, the robot can simply choose the next state estimate at random from the corresponding probability distribution, then remove the corresponding index from $\mathcal{M}_i$. This means estimates of closer robots are more likely to be chosen than ones that are farther away. By repeating this process, the entire time slot of length m can be filled in a straightforward, probabilistic manner.

Algorithm 1 Deterministic Method for Selecting State Estimates

n is the number of robots in the system and m is the time slot length.
Require: Robot i knows its state $p_i(t)$ and the state estimate of other robots $p_j(t_{ij})$.
Require: Robot i knows its running counter $[c_{i1}, \ldots, c_{in}]$.
$\mathcal{M}_i \leftarrow \{1, \ldots, n\}$; $\mathcal{N}_i \leftarrow \emptyset$;
for 1 to m **do**
$P^{ij}_{\mathcal{M}_i}(t) \leftarrow \frac{f_{ij}(t)}{\sum_{k \in \mathcal{M}_i} f_{ik}(t)}$, $\forall j \in \mathcal{M}_i$; $c_{ij} \leftarrow c_{ij}[1 - P^{ij}_{\mathcal{M}_i}(t)]$, $\forall j \in \mathcal{M}_i$;
$k \leftarrow \arg\max_{k \in \mathcal{M}_i}(c_{ik})$; $\mathcal{M}_i \leftarrow \mathcal{M}_i \backslash \{k\}$; $\mathcal{N}_i \leftarrow \mathcal{N}_i \cup \{k\}$; $c_{ik} \leftarrow 1$;
end for
return $\mathcal{N}_i$

3.2 Deterministically Selecting Estimates

It is not ideal in practice to probabilistically select which state estimates to broadcast. Consecutive selections of a particular robot index can be separated by an undesirably long period of time, especially concerning distant robots. By developing a location-based deterministic algorithm, we can increase the average rate at which all state estimates of a given time stamp will propagate throughout a team. In the deterministic case, propagation time is bounded above by the longest path taken among the estimates. No such bound exists in the probabilistic case, resulting in a positively skewed distribution of propagation times and a larger mean.

We propose that each robot maintains a list of counters, $[c_{i1}, \ldots, c_{in}]$, which are initially set to a value of one. Using the probability mass function in Equation (2), each counter represents the probability that the corresponding index has *not* been selected. Consider a robot's first selection, which will always be its own index. The probability, $P^{ii}_{\mathcal{M}_i}(t)$, of selecting index i is equal to one, while all other probabilities, $P^{ij}_{\mathcal{M}_i}(t)$ subject to $j \neq i$, are equal to zero. This implies that the counter c_{ii} is multiplied by $[1 - P^{ii}_{\mathcal{M}_i}(t)] = 0$, or a zero probability of not being selected, while all other counters, c_{ij}, are multiplied $[1 - P^{ij}_{\mathcal{M}_i}(t)] = 1$, or a probability of one. By selecting the index with the lowest counter value, we are deterministically guiding our method to behave according to the probability distribution described by Equation (2). The selected index (in this case i) is removed from the set $\mathcal{M}_i$, and its corresponding counter (c_{ii}) is reset to a value of one. This process is iteratively applied to completely fill a time slot with m state estimates, with counters maintaining their values between frames. The complete deterministic strategy of $\mathcal{O}(mn)$ time is given in Algorithm 1.

4 Simulations and Experiments

We provide insight into the performance of the location-based algorithm in three ways: we conducted Monte Carlo simulations for 100 stationary robots, we used real control and wireless hardware to simulate nine robots running a

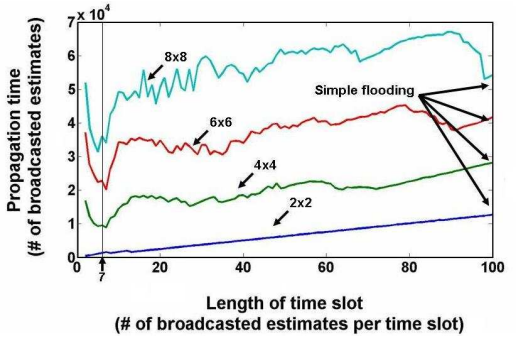

Fig. 2 This figure shows the average propagation time for the location-based algorithm running on a 10×10 stationary robot grid. Averages were taken over 1000 Monte Carlo simulations. For small subgraphs (i.e. 2×2), update rates of state estimates increased with decreasing time slot lengths. For larger subgraphs, the optimal length was around $m = 7$.

distributed coverage algorithm, and we implemented this hardware on five flying and four stationary robots. We first describe the Monte Carlo simulations used to measure information propagation throughout the robot team. Propagation time is the main performance metric for the algorithm. This metric depends on the length of the time slot, or in other words, the number of state estimates communicated during one robot broadcast. We compare these results to the case when the time slot length equals the number of robots, since allowing robots to broadcast every state estimate is the simplest multi-hop scheme. This scheme is referred to as simple flooding.

In a MATLAB environment, we simulated a team of 100 stationary robots arranged in a 10×10 square grid. Each robot, initialized knowing only its own state estimate, was able to receive broadcasts from its adjacent neighbors along the vertical and horizontal directions. Each robot ran Algorithm 1 in distributed fashion. Over 1000 Monte Carlo simulations were executed for time slots of varying lengths, with each run having a random order for the time slot assignments. For the 2×2, 4×4, 6×6, and 8×8 subgraphs centered on the 10×10 graph, we measured the time it took for all subgraph members to exchange state estimates.

Figure 2 plots average propagation time for the Monte Carlo simulations. For the smallest subgraph (i.e. 2×2), state estimates propagated faster with smaller time slot lengths. This relationship makes sense since we are maximizing the frame rate, thus increasing update rates for the local state estimates of highest importance. As the subgraph size increases, very small time slot lengths become less effective at propagating estimates, especially between robots at opposite sides of the subgraph. By using a slightly larger time slot length, a significant improvement in performance over simple flooding is obtained; propagation times for all subgraphs decreased by more than 47% using a time slot length of $m = 7$. Analyzing such Monte Carlo plots provides a heuristic technique for selecting an acceptable time slot length for a given control problem.

We then tested the algorithm in a simulated robot scenario using real control and wireless hardware. We implemented a Voronoi coverage controller [2] on nine custom ARM microcontroller modules, each using a 900 MHz xBee module to wirelessly broadcast state estimates during its assigned time slot.

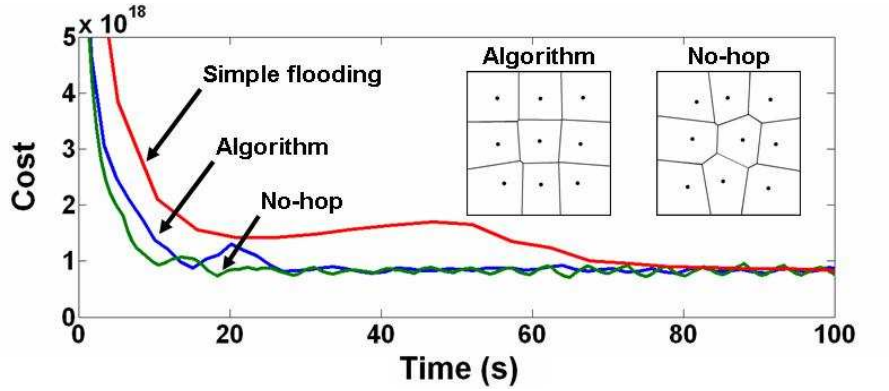

Fig. 3 Coverage costs are shown for a nine robot system simulated on real hardware running a Voronoi coverage controller. The system has a frame rate of 1.7 Hz when using a no-hop scheme ($m = 1$). The system initially performs well, but its inability to multi-hop state estimates resulted in a suboptimal final configuration. A simple flooding scheme ($m = 9$) improved steady state performance, however, the slow frame rate of 0.2 Hz caused the system to initially oscillate in a high cost configuration. The location-based algorithm with a time slot of length $m = 3$ performed the best overall by combining fast update rates with multi-hop capabilities. The final Voronoi configurations for the algorithm and no-hop simulations are also shown.

Each control module simulated the dynamics of a flying robot, creating a virtual distributed robot team. In addition, a communication range was implemented such that packets from "out-of-range" robots were automatically dropped. We investigate the performance of the location-based algorithm in a simple scenario where nine virtual robots were tasked to cover a square area. For this scenario the optimal configuration is for the robots to be arranged in a 3×3 square grid.

For the location-based algorithm, a time slot length of $m = 3$ was selected using the Monte Carlo technique previously discussed. We also selected the Euclidean distance function with $\alpha = 1$ given that the Voronoi coverage controller is linearly dependent on such distance. Each state estimate for the virtual flying robot is constructed of six 32-bit integers (robot identification, time stamp, latitude, longitude, altitude, and yaw), resulting in a data size of 192 bits. Given that the wireless hardware could reliably operate at 3000 baud, the resulting frame rate was about 0.6 Hz. For comparison, the simple flooding ($m = 9$) and no-hop ($m = 1$) schemes ran at about 0.2 Hz and 1.7 Hz, respectively. Figure 3 shows the resulting coverage cost profiles from these simulations. The location-based algorithm had better initial performance than the simple flooding scheme and better steady state performance than the no-hop scheme. The final Voronoi configurations for the algorithm and no-hop simulations are also shown.

Finally, we implemented the location-based algorithm on five AscTec Hummingbird quad-rotor flying robots [3] and four stationary robots, thus creating a nine robot team. Each flying robot was equipped with an AscTec AutoPilot

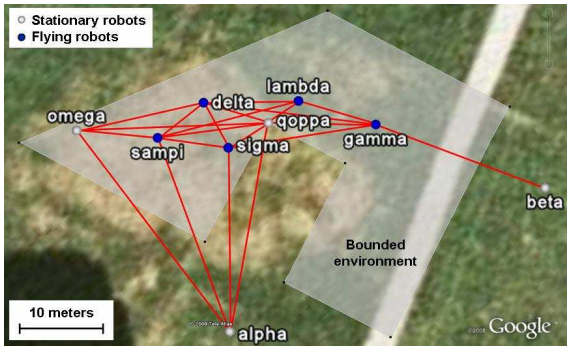

Fig. 4 An example mobile ad hoc network graph from the quad-rotor flying robot experiment is plotted in Google Earth. For this nine robot system, the location-based algorithm routes state estimates through the entire team. The bounded environment from the downward facing camera coverage problem is also shown.

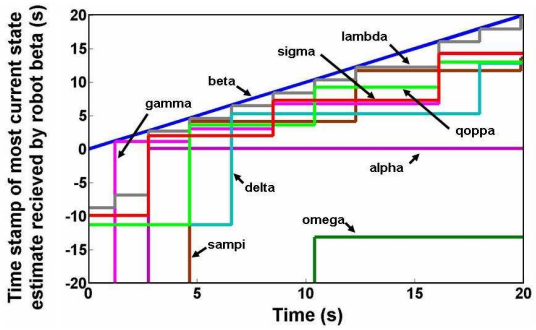

Fig. 5 This plot shows the time stamp of the most current state estimates received by the stationary robot beta. Estimates of closer, more important robots are updated more frequently and tend to be more current, which validates the location-based algorithm.

board capable of capturing GPS, altitude, and yaw positions. The previously described control and wireless modules were installed on these AutoPilot boards. In addition, four separate modules were deployed at fixed locations to represent the stationary robots.

This experimental setup was designed to run a downward facing camera coverage controller for hovering robots [6]. Since this controller has a spatial dependence similar to the Voronoi coverage controller, the same time slot length, distance function, and α were used. Figure 4 shows the network topology of a random deployment configuration prior to starting the coverage controller. Here we limited the communication range to 30 meters; in previous experiments we were able to produce links in excess of 100 meters. Figure 5 plots the time stamp of the most current state estimates as received by the stationary robot beta, which can be considered the "worst case" receiver since it is the most remote robot in the team. As previously discussed, beta's own state estimate is always considered to be current. Estimates of other robots are updated as they are received by team broadcasts, whether directly from the originating robot or indirectly in a multi-hop fashion. Since closer robots are considered more important in the algorithm formulation, this results in their state estimates being more current with more frequent updates.

5 Conclusion

In this paper we presented a location-based strategy for exchanging state estimates in a distributed robot team. We developed a deterministic algorithm that, based on an importance function, broadcasts estimates of nearby robots more frequently than distant ones. Simulations using real control and wireless hardware show that the algorithm outperforms simple flooding schemes for large robot networks.

Our experiments consisting of five Asctec Hummingbird quad-rotor flying robots among four stationary robots showed the successful exchange of state estimates in a multi-hop fashion. Using the location-based algorithm, we successfully ran the coverage controller from [6] on three flying robots with downward facing cameras. Coverage results for 5+ flying robots will be presented in future publications.

We desire to further develop this work to exploit the spatial reuse of time slots for robots separated by multiple hops. This direction allows for virtually infinite team sizes and spatial coverage.

Acknowledgements This work was done in the Distributed Robotics Laboratory at MIT and is supported in part by the MURI SWARMS project grant number W911NF-05-1-0219, NSF grant numbers IIS-0513755, IIS-0426838, CNS-0520305, CNS-0707601, EFRI-0735953, the MAST project, MIT Lincoln Laboratory, and the Boeing Corporation. This work is sponsored by the Department of the Air Force under Air Force contract number FA8721-05-C-0002. The opinions, interpretations, recommendations, and conclusions are those of the authors and are not necessarily endorsed by the United States Government.

References

1. Cai, Y., Hua, K., Phillips, A.: Leveraging 1-hop neighborhood knowledge for efficient flooding in wireless ad hoc networks. Performance, Computing, and Communications Conference, 2005. IPCCC 2005. 24th IEEE International pp. 347–354 (2005)
2. Cortes, J., Martinez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. Robotics and Automation, IEEE Transactions on **20**(2), 243–255 (2004)
3. Gurdan, D., Stumpf, J., Achtelik, M., Doth, K.M., Hirzinger, G., Rus, D.: Energy-efficient autonomous four-rotor flying robot controlled at 1 khz. Robotics and Automation, 2007 IEEE International Conference on pp. 361–366 (2007)
4. Haas, Z.: A new routing protocol for the reconfigurable wireless networks. Universal Personal Communications Record, 1997. Conference Record., 1997 IEEE 6th International Conference on **2**, 562–566 vol.2 (1997)
5. Ni, S.Y., Tseng, Y.C., Chen, Y.S., Sheu, J.P.: The broadcast storm problem in a mobile ad hoc network. In: MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking, pp. 151–162. ACM, New York, NY, USA (1999)
6. Schwager, M., Julian, B., Rus, D.: Optimal coverage for multiple hovering robots with downward facing cameras. In: Proc. of International Conference on Robotics and Automation (ICRA09). Kobe, Japan (2009)
7. Sun, M.T., Lai, T.H.: Location aided broadcast in wireless ad hoc network systems. Wireless Communications and Networking Conference, 2002. WCNC2002. 2002 IEEE **2**, 597–602 vol.2 (2002)