Experiments with navigation based on the RSS of wireless communication

Luis Oliveira, Hongbin Li, Luis Almeida

***Abstract*— This paper addresses the general field of mobile robots navigation within fixed sensor networks. We focus on indoor and anchor-less navigation methods based only on RF communications and the respective RSSI (Received Signal Strength Indicator). We implement two distance-based methods, namely a simple oblivious method – based on random directions – and a more complex and demanding iterative method – based on a Maximum Likelihood Estimator. These methods are then evaluated with practical experiments. In particular, in this work we made a small autonomous robot move back and forth between two RF beacons, using RSSI information collected from low-power radios with omni-directional antennas. The results show the superiority of the iterative method concerning the speed of reaching the target, at the cost of extra computations, confirming the viability of RSSI-based navigation.**

***Index Terms*— Robot Navigation, wireless communication, RSSI**

I. INTRODUCTION

Mobile robots with specific sensors, actuators or other instrumentation, can provide a valuable help in covering larger areas or objects, complementing the information that can be obtained with networks of fixed sensors. This is typically the case in application domains such as surveillance [1] and exploration [2]. The benefits can be further extended having several mobile robots cooperating among themselves [3]. In these cases, the robots have to navigate by using aids for that purpose – from landmarks, to GPS (Global Positioning System). However, landmarks are frequently undesired either because they cannot be deployed or they interfere with the environment, or are not resistant and fade out or can even be removed maliciously. Similarly, GPS has reception problems in in-door environments.

In such situations, relative localization and navigation becomes more attractive. For example, the team of robots can assume a formation that is more convenient to provide longer reach or wider area coverage while maintaining connectivity, just by knowing how they are positioned with respect to each other. In another case, one can take advantage of the pre-installation of wireless beacons, that can be nodes of a wireless sensor network for example, and make robots navigate through a specific sequence of beacons in order to visit certain points in space and perform an approximate desired trajectory.

Hongbin Li is with the State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou 310027, China. Email: hbli@iipc.zju.edu.cn

Luis Almeida is with IEETA / DEEC-FEUP, Univ Porto, Porto, PORTUGAL. Email: lda@fe.up.pt

Luis Oliveira is with INESC PORTO, Porto, Portugal. Email: luis.q.oliveira@inescporto.pt

In this paper, we specifically address this latter case using RF (Radio Frequency) beacons and we will focus on the navigation towards each beacon using the respective RSSI (Received Signal Strength Indicator) as an estimation of relative distance, despite the coarse relationship between both. This has been successfully achieved in previous work such as [4], [5] to provide relative localization in a team of robots. The work in this paper is a continuation of that reported in [6], [7] particularly with an extension of the practical experiments. Note that in this work we do not present a theoretical contribution, instead, we use previously published methods – sliding filter, oblivious and MLE method – and apply them to the RSSI readings, thus providing useful information to the navigation in RSS fields, which have low gradients and high noise. We will use one robot and two beacons, and we will compare two methods to make the robot go back and forth between the two beacons, without any pre-planned path information and using solely the robot relative displacements and the RSSIs of the received beacons.

II. BACKGROUND

A. RSSI field

It is known that the intensity of the electromagnetic signal of the wireless communication attenuates as it travels in open air. The exact expression that relates such attenuation with distance is rather complex and depends on several environmental parameters. Moreover, it is only valid in the absence of obstacles, either because of the extra attenuation they cause or because of the signal reflections that lead to multi-path interference. Nevertheless, as long as there is a line of sight between sender and receiver, they are away from large metallic parts and adequate filtering is used, the result is a well-behaved RSS with a relatively good gradient [7]. This allows using gradient-based navigation methods as well as other methods based on beacon localization.

For simplicity, this work uses relative, anchor-less and, thus, easy to deploy beacon localization methods. There are a variety of such methods – e.g. based on angular detection and distance estimation. The first require directional antennas to detect the heading of the beacon with respect to the robot, which are more expensive and not always available. The second use simple, common omni-directional antennas and use an estimation of the distance between the robot and the beacon. In both cases, the robot has to move between different points in space and analyse the variation in angle/distance to the beacon. The methods used in the work are based on distance estimation, only, for which the robots use the RSSI of each periodically received beacon message.

B. Navigation Using an oblivious method

This is a rather simple method that compares the current RSSI reading with the previous one to decide on the direction to head to. This method has been used in both [6] and [7] and it has the main advantage that it can be applied to very simple robots – e.g. the robot does not need encoders attached to the motors. The decisions are merely based on three premises: the robot approached the beacon; moved away from it; or the variation was not significant. In the former case, the heading is kept, in the second case, the heading is reverted 180 and in the latter case a random heading is chosen.

C. Navigation Using MLE

The MLE (Maximum Likelihood Estimation) method is a more complex approach that estimates the most likely position of the beacon based on the RSSI readings obtained in several different points. This method is iterative and, as such, it requires a larger capacity in both memory and processing. Moreover, the relative position of the points where the RSSI readings are taken is also necessary and thus a more sophisticated robot with encoders to measure movement is required. Note that the sole function of the encoders is to calculate the relative displacement between each pair of RSS measurements and that the error of their readings is negligible both due to the much larger intrinsic error associated to the RSSI readings and to the fact that the encoder readings are accumulated in a moving window of 12 robot steps, only, thus with a bounded, non-persistent and relatively small error. This method has been used in [7] both theoretically, in a Matlab simulation, and experimentally, using a moving platform. The same work also shows that, by tuning some parameters, this method is feasible in either noiseless or noisy environments with a faster or slower convergence to the beacon. The algorithm needs a certain number of initial sample points (N_{ten}) to provide an estimation of the beacon position. The higher this parameter the more time it takes for the algorithm to actually start working. After such number of samples, the algorithm continues saving samples in a circular buffer with a certain depth (N_{queue}) while producing a new estimate with the samples in the buffer every time a new sample is received. In this case, the deeper the buffer, the more accurate is the estimation.

III. Implementing the navigation strategy

This implementation has three important elements which will be described below – the beacons, the robot and the computer.

A. The beacons

To begin with, crossbow's MicaZ motes were used, Figure 1. As [7] suggests, the beacons used are actually a set of 3 nodes separated by 5cm. This is done to reduce the impact of RSSI noise. These nodes, which have one antenna each, are properly synchronized and their purpose is to emulate a single node with three antennas. The beacons are set up in the following way: each beacon has a master

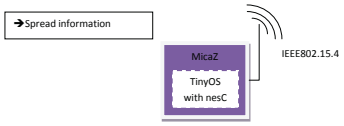

Fig. 1. The MicaZ beacons setup

node, whose functions are trigger the beacon transmission and synchronize with other master nodes to avoid collisions; additionally each beacon has two slave nodes, these nodes, upon receiving the message from their master, transmit after a small amount of time. The synchronization is made using an Adaptive-TDMA method based on [8]. The transmission period is set to 500ms and the message sent by the nodes contains the extended connectivity matrix, an ageing vector, and the requested and performed moves.

When the other motes receive this beacon, and the LQI is above a given threshold (in our case 100), they update their RSSI table accordingly and save the information the CC2420 chip provides about the transmission (Link Quality Indicator and Received Signal Strength Indicator) for future input of their own information in the table. If, on the other hand, the LQI is too low the message is drop. According to CC2420 datasheet, RSSI is made available in a register with a value between -60 and 40, corresponding to an RSSI between -100 and 0 dBm, so, in order to make it positive, for a simpler transmission, an offset of 60 is added. This is shown in Figure 2. Also periodically, the values on the RSSI table are first cleaned up (if too old), and new values (previously saved) are included in the table.

B. The robot

To begin with, since these experiments include a moving robot controlled by a MicaZ mote the above explanation is still valid, see Figure 3.

Adding to that the mote also has the task of putting the robot in motion. This task has been simplified by making the moves very simple: rotate and move forward, Figure 4. But navigation still has a problem: the existence of obstacles in the way of the robot. To cope with obstacles, we used infrared sensors in the robot sides and front. Since our objective was not the quality of the obstacle avoidance algorithm, these sensors were used in a very simple manner. When the robot reaches the desired rotation angle it checks whether there are obstacles in his heading. If the heading is clear, it stops, otherwise it continues rotating until it finds a clear front path. Similarly, in each forward movement, the robot advances in a straight line until covering the desired distance. If an obstacle is detected before, the robot stops.

C. The Computer

The computer setup is as shown in Figure 5. As it can be seen it receives data from a node, which triggers a sequence of operations. These operations are filter the RSSI data, control the progress of the robot, and generate new moves. The program runs in Matlab and calls Java methods.

```
TDMA:
1 if(source is the master of this group)
2     set_send_time;
3 else
4     if(source is another master and i am a master)
5         resync;
6     endif
4     RSSI=0;
5 endif
6 LQI=receivedLQI;

Get RSSI table:
1 if(piggyback has RSSI table)
2     get_data_from_message;
3     for i=other_nodes
4         if(received_table_age[i] is newer)
5             replace_local_table[i];
6             replace_local_age[i];
7         endif
8     endfor
9 endif

Get requested moves:
1 if(piggyback has requested moves)
2     if(received_requested_move[i].step is newer)
3         get_requested_move[i];
4         if(is for me) start_moving;
5     endif
6 endif

Get performed moves:
1 if(piggyback has performed moves)
2     if(received_performed_move[i].step is newer)
3         get_performed_move[i];
4     endif
5 endif

Check and set send moves status:
1 if(performed_moves_steps > = requested_moves_steps)
2     send_only_performed_move;
3 else
4     send_only_requested_move;
5 endif

Send data to computer:
1 if(connected to computer)
2     send_data_to_computer;
3 endif
```

Fig. 2. New message processing by MicaZ

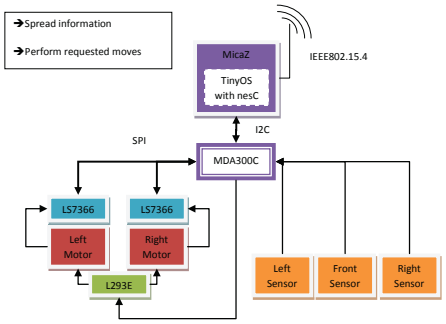

Fig. 3. The robot setup

```
Move Robot:
1 if(robot is IDLE)
2     rotate;
3 else
4     if(robot is ROTATING)
5         go_forward;
6     else
7         idle;
8     endif
9 endif
```

Fig. 4. Robot's moving algorithm

1) RSSI data Processing: Note that up to this point the received information is still from 7 different nodes. So, in order to finalize the multi-antenna emulation, the mean of the several non-zero RSSI values corresponding to the nodes in a given beacon must be calculated. Finally, to conclude the RSSI data acquisition each time this program receives the extended connectivity matrix, from the MIB600 board via TCP/IP, it writes it on a sampling table, which holds the information of three previous matrices (sampling window). Then, this information is used to calculate a mean for each non-zero element in the table, creating the RSSI sample, see [4].

2) Movement Planning: In addition to the current RSSI, the information about the performed moves also arrives to the computer. This information is saved and used to perform a set of operations.

First, since this means the requested move is done, the information on the new position and angle is computed and saved (note that this information is only important in the

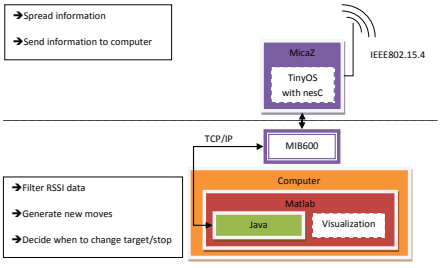

Fig. 5. The computer setup

```
1 if (abs(RSSI-RSSI_old)<RSSI_THRESHOLD)
2    Random_Move;
3 else
4    if(RSSI-RSSI_old>0.0)
5       Keep_Going;
6    else
7       Turn_Around;
8    endif
9 endif
10 RSSI_old= RSSI;
11 while(angle>180.0)
12    angle-=360.0;
13 endwhile
14 while(angle<-180.0)
15    angle+=360.0;
16 endwhile
17 send_move_request();
```

Fig. 6. generateMove Method With Oblivious

```
1 if(Nsamples<=Nten)
2    random_move();
3 else
4    positions=getPositions(Nsamples);
5    distances=getDistances(Nsamples);
6    Beacon = MLE(Nsamples,positions, distances);
7    angle = atan2(y_Beacon-y_RobotPosition,x_Beacon-
x_RobotPosition);
8    angle = angle-currentDirection;
9    while(angle>180.0)
10      angle-=360.0;
11   endwhile
12   while(angle<-180.0)
13      angle+=360.0;
14   endwhile
15 endif
16 send_move_request();
```

Fig. 7. generateMove Method With MLE

MLE method as the oblivious does not need to know the position or angle). Then, the current RSSI data is saved and associated with the current position. Once this is done, it is finally time to perform the new move. This will be done in different manners, depending on the method in use, as described further on.

3) Movement with the oblivious method: In the beginning of the run, the robot goes to a random direction. If the difference between the RSSI reading in this location and the reading in the previous location proves to be positively greater than a set threshold, then the robot proceeds in the same direction, since it means the robot is approaching the beacon. If, on the other hand, the value is negatively greater than the threshold, the robot turns around, since the beacon is further. Finally, if the threshold is not met, the robot rotates randomly and proceeds. Once the computer decides what to do, it sends the information to the robot, which will perform the move. This is shown in Figure 6.

4) Movement with the MLE method: To begin with, the MLE method, Figure 7, needs some information to start the iterations, which will, eventually, lead to the objective. This is done by making a small number of random tentative moves, in order to acquire data to feed to the algorithm. After this initial set of data is available, the iterations start, and the robot starts to effectively approach the beacon.

As mentioned before this initial set it's not enough in high noise environments and as such, after each step is taken, more data is collected filling a queue until a maximum size, N_{queue}. This queue is used as a circular buffer, in which newer information replaces the oldest one. This allows the collection of more data while already approaching the objective, and, in a low noise environment, a quick approach to the objective. The experiments in [7] suggest the use of $N_{ten} = 4$ and $N_{queue} = 12$ so that, in a low noise environment, after four steps, the robot is already targeting the objective; in a high noise environment, the robot collects data up to twelve steps and is still able to reach the objective. Note that, since the computer has all the data regarding the RSSI readings from all the nodes, once the robot changes target, the data fed into the MLE algorithm is data previously collected. So, instead of beginning with zero entries in the queue, it begins with the most recent entries already collected, up to a maximum of twelve. This data, which is composed by positions and RSSI readings, is used during the iterations to estimate the position of the beacon. The first thing to is do, is to transform the RSSI reading in signal space distances and then feed the positions and these calculated distances to the MLE algorithm. This algorithm uses the system with n equations that describe the distance between two points:

$$\begin{cases} (\bar{x}_{beacon} + x_1)^2 + (\bar{y}_{beacon} - y_1)^2 = d_1^2 \\ \vdots \\ (\bar{x}_{beacon} + x_n)^2 + (\bar{y}_{beacon} - y_n)^2 = d_n^2 \end{cases} \quad (1)$$

This, by subtracting the $n-th$ equation, allows writing: $A\bar{x} = b$

where $A = \begin{bmatrix} 2(x_n - x_1) & 2(y_n - y_1) \\ \vdots & \vdots \\ 2(x_n - x_{n-1}) & 2(y_n - y_{n-1}) \end{bmatrix}$ and $b = \begin{bmatrix} d_1^2 - x_1^2 & +x_n^2 - y_1^2 + y_n^2 \\ \vdots & \\ d_{n-1}^2 - x_{n-1}^2 & +x_n^2 - y_{n-1}^2 + y_n^2 \end{bmatrix}$.

The next step is to solve the system in order to get the estimate beacon position, $\bar{x} = \begin{bmatrix} \bar{x}_{beacon} \\ \bar{y}_{beacon} \end{bmatrix}$:

$$\bar{x} = (A^T A)^{-1} A^T b \quad (2)$$

Note that this is actually the least squares method, which minimizes the residual to obtain the beacon position estimate. Finally, once the beacon estimate position is calculated, the computer can calculate how much the robot needs to rotate to face the target. This is done by transforming the Cartesian

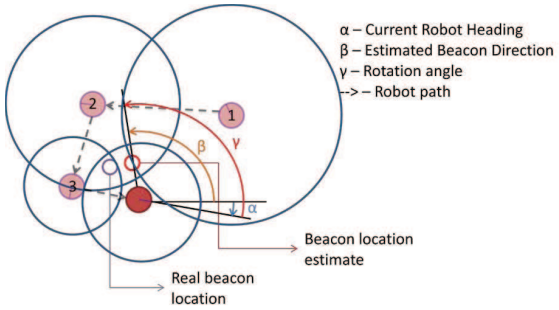

Fig. 8. Multi-lateration with MLE

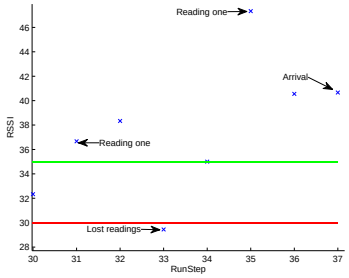

Fig. 9. Arrival Condition

coordinates (x, y) into polar coordinates (distance, angle) and, finally, by subtracting to this calculated angle the angle the robot is currently pointing to. This new generated move is then requested back to the robot. All this process is depicted in Figure 8.

5) Condition of arrival at the beacon: The final function of the computer is to detect an arrival at a beacon. The strategy followed to consider a valid arrival was based on observation of the behaviour of the RSSI with the distance. From observation we concluded that near the beacon the RSS values would easily be above thirty five. So, we made this value into a threshold so that, when the RSS rises above it, the robot is in the vicinity of the beacon. But, a problem still persists at this point – interferences allow such a high reading to be received far away from the beacon. So, three readings above that threshold were made compulsory so that the program considers a successful arrival. Adding to this, one last issue remains, the robot can move away from the beacon or even go to an area that has destructive interference. Therefore, in order to finally settle this, a second lower threshold, with the value of thirty, was created so that the count does not reset while the RSSI doesn't drop below that value. This is illustrated in Figure 9.

IV. Experimental Results

The objective of this experiment is to test the capability of navigation in a multi-beacon environment using the two methods described before, oblivious and MLE, being the only difference between them the move decision made. The initial positions of the units can be seen in Figure 10 and the task the robot has to perform is to go from the starting point (0, 0), to beacon zero (0, 250), then go to beacon

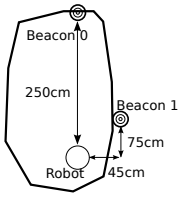

Fig. 10. Experiments Initial Setup

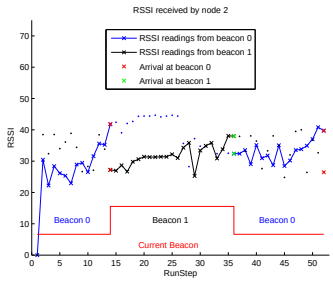

Fig. 11. Oblivious Method Experiment 1

one (45, 75) and, finally, return to beacon zero (0, 250). All this based only on the received RSSI readings, i.e. no path backtracking will be done to navigate back to beacon zero. The following experiments were all done with the transmission power set to $-19.17dBm$ and in channel 26 as advised in [9]. Additionally the experiments were also made in the same conditions and repeated several times, with both methods. Remember that each beacon is actually a set of 3 nodes separated by $5cm$ and that there is only one node in the robot.

By a quick analysis of Figures 11, 12, 13, 14, 15, and 16 it becomes clear clear that the number of steps needed to make the first approach, using the MLE method, is much larger than in the second and third approaches. This is easily explained. While in the first approach there is a lack of values on the queue of data fed to the MLE, in the following approaches there are 12 values available to feed the algorithm, which makes a much more precise estimation possible. Another point of interest is the monotonic rise of the RSSI values with this method, which shows how effective the algorithm is. On the oblivious method, on the other hand, there are rises and falls in the readings and the results are much more inconstant. Also, while with the MLE, the first

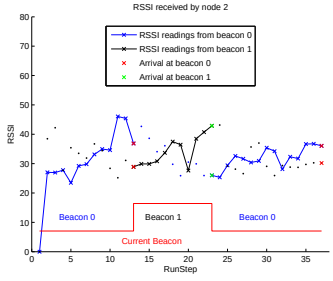

Fig. 12. Oblivious Method Experiment 2

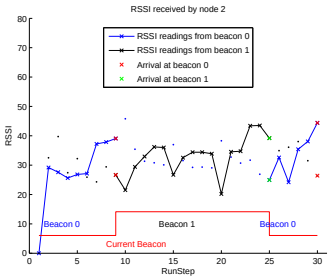

Fig. 13. Oblivious Method Experiment 3

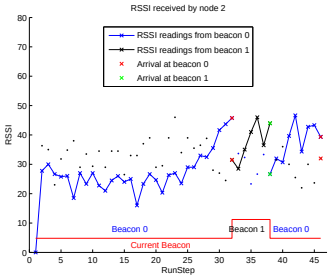

Fig. 14. MLE Method Experiment 1

approach is slow and the subsequent are faster, with the oblivious method sometimes they are faster and sometimes they are slower. This is not at all unexpected due to the random nature of the oblivious algorithm.

Finally in experiment 14, a lot of going back and forward is visible. Although this seems contradictory to the MLE algorithm, by observing the experiment, it was possible to see that that was caused by the existence of obstacles which did not allow the robot to move where it wanted to and, therefore, collect RSSI values with a larger difference. This shows a possible weak point of the MLE algorithm – the big dependence on a good relationship between the RSSI with the distance – which makes the robot, if trapped in a location where the readings are very similar, to take a while or not be able to proceed. A possible solution for the MLE problem mentioned above is to check the positions the robot was at, and where it is. Based on that, and on the beacon estimates, is possible to make the robot move somewhere he hasn't been in recent time so that it can collect more and different information. Also interesting would be to

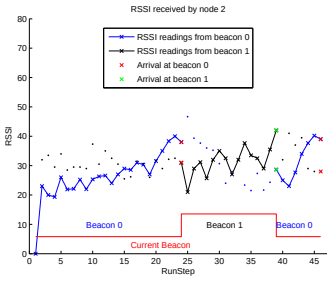

Fig. 15. MLE Method Experiment 2

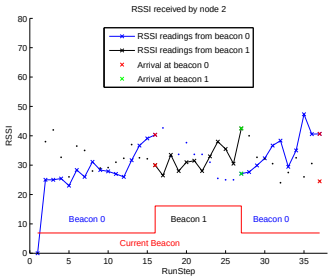

Fig. 16. MLE Method Experiment 3

perform these experiments in an obstacle free environment. This would avoid interferences caused by the obstacles in the algorithms, either helpful or detrimental. Another interesting point would be to compare the results of a robot approaching a beacon with a robot approaching a robot. These tests are appealing because the reactivity of the methods would be put to the test.

V. Conclusions

The resulting work was experimentally evaluated, with emphasis on the comparative evaluation of the oblivious and MLE methods. This evaluation proved to be possible to navigate in a multi-beacon environment using only the RSSI information. In the performed experiments, the MLE has a slow start, since it has to collect information to make a good estimate. But, after the initial steps, needed to acquire a good notion of the target direction, MLE was much faster directed to the position where the beacon was, with relatively little deviations. The oblivious method, on the other hand, shows a constant and higher tendency to deviate from the beacon due to its randomness. Finally, the MLEs big dependence on a good relationship between the RSSI and the distance was exposed as a weak point when using this method.

References

[1] Q. Limited, "Quadratec," in *http://www.quadratec-ltd.co.uk/Security_Surveillance_systems.htm*, 14-10-2009.

[2] S. Baek, S. Ahn, and S.-Y. Oh, "Fast localization algorithm for the cleaning robot by using self-organization map," *International Symposium on Computational Intelligence in Robotics and Automation*, pp. 19–24, 2007.

[3] A. S. Fukunaga, Y. Cao, and A. B. Kahng, "Cooperative mobile robotics: Antecedents and directions," *Autonomous Robots*, vol. 4, pp. 226–234, 1997.

[4] H. Li, L. Almeida, Z. Wang, and Y. Sun, "Relative positions within small teams of mobile units," *Mobile Ad-Hoc and Sensor Networks*, 2007.

[5] Y.Shang, W. Rumi, T. Zhang, and M. P. J. Fromherz, "Location from mere connectivity," 2003, pp. 201–212.

[6] H. Li, L. Almeida, F. Carramate, Z. Wang, and Y. Sun, "Connectivity-aware motion control among autonomous mobile units," *SIES 2008, International Symposium on Industrial Embedded Systems, 2008.*, pp. 155–162, 2008.

[7] ——, "Using low-power radios for mobile robots navigation," *FET 2009 - 8th IFAC Conference on Fieldbuses and Networks in industrial and embedded systems*, 2009.

[8] F. Santos, G. Currente, L. Almeida, N. Lau, and L. S. Lopes, "Self-configuration of an adaptive tdma wireless communication protocol for teams of mobile robots," 2007.

[9] Crossbow, "Avoiding rf interference between wifi and zigbee."