Path Planning and Navigation for Mobile Robots in a Hybrid Sensor Network Without Prior Location Information

Zheng Zhang¹, Zhenbo Li¹, Dawei Zhang¹ and Jiapin Chen¹,*

¹ National Key Laboratory of Nano/Micro Fabrication Technology, Key laboratory for Thin Film and Microfabrication of Ministry of Education, Research Institute of Micro and Nano Science and Technology, Shanghai Jiao Tong University, Shanghai, P. R. China

* Corresponding author E-mail: chenjp@sjtu.edu.cn

Received 4 Aug 2012; Accepted 10 Jan 2013

DOI: 10.5772/55790

© 2013 Zhang et al.; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract In a hybrid wireless sensor network with mobile and static nodes, which have no prior geographical knowledge, successful navigation for mobile robots is one of the main challenges. In this paper, we propose two novel navigation algorithms for outdoor environments, which permit robots to travel from one static node to another along a planned path in the sensor field, namely the RAC and the IMAP algorithms. Using this, the robot can navigate without the help of a map, GPS or extra sensor modules, only using the received signal strength indication (RSSI) and odometry. Therefore, our algorithms have the advantage of being cost-effective. In addition, a path planning algorithm to schedule mobile robots’ travelling paths is presented, which focuses on shorter distances and robust paths for robots by considering the RSSI-Distance characteristics. The simulations and experiments conducted with an autonomous mobile robot show the effectiveness of the proposed algorithms in an outdoor environment.

Keywords Mobile Robot, Wireless Sensor Network, Navigation, Received Signal Strength Indication

1. Introduction

Hybrid sensor networks with static and mobile nodes open a new frontier in the research into wireless sensor networks (WSN). Static nodes support environmental sensing and network communication. Mobile nodes have more resources for sensing and computing and can move to particular locations to perform more complicated missions such as repairing the network [1] or providing in-depth analysis [2]. Introducing mobility to WSN not only reduces the costs of the system but also improves its adaptability to dynamic environments. In this paper, we focus on the problem of navigating mobile robots to the locations where events occur in the outdoor sensing field. The mobile robot can move along the planned path composed of a series of static sensor nodes, hop-by-hop, to the destination.

First, a routing protocol must exist that can connect and pass messages in WSN from the initial location of the robot to its final destination. A wide range of routing protocols exists in the currently available literature such as DSR and AODV, used in the ad hoc network [3]. These
algorithms have been shown to produce a “near-minimal” communication route that connects the nodes in the sensor field. So the communication route can be used as the planned path for robots navigating between any two nodes in the sensor field. However, these route algorithms for communication only select the path with the fewest hops. Thus, the route algorithm is not suitable for use in communication without changes to mobile robot path planning approaches. This paper focuses particularly on navigation for mobile robots in a low-power WSN and proposes a distance and robustness aware path planning (DRAPP) algorithm to obtain a robust path with shorter geographic distance suitable for robot navigation using WSN.

Navigation is regarded as one of the most significant and fundamental problems in mobile robot research. Some researchers present the idea of using WSN for robot navigation [4, 5]. However, navigation with WSN, which is based on sensor nodes in WSN, is totally different from traditional navigation methods which provide the robot with a path containing position details. With the traditional methods, because the position information of the path is given, the robot can go from one location to another easily. Whereas with WSN-based methods, even though the robot knows which navigation node it should move to next, without location information, it still does not know how to get there. In order to localize the mobile node, the coordinates of deployed sensor nodes should be known in advance [6-8] and additional GPS devices are required in [9, 10]. Thus, it is disadvantageous as well as inefficient to localize the mobile node by obtaining the exact coordinates of static sensor nodes because sensor nodes are generally deployed randomly.

Navigation of mobile robots in WSN without prior location information has been extensively researched in recent years. Some researchers have already been aware of the difference between the traditional navigation methods and those with WSN, and proposed some further algorithms to enable the robot to navigate with a path composed of a series of sensor nodes. Jern-Ruey Jiang et al. mounted two antennas on a mobile robot [11]. The robot rotates its directional antenna from 0° to 360° and chooses the direction with the strongest RSSI for its next movement. However, its implementation efficiency is very low because the robot has to stop to rotate its directional antenna for each step. Li et al. used an artificial potential field for guiding mobile robots across a sensor network along the safest path, away from danger detected by the sensors [4]. However, the algorithm does not account for the actual robot movement from node to node along the pathway. The GNAT’s project [12, 13] enabled a LEGO robot to navigate across a sensor network composed of 156 nodes by path planning without any localization techniques. However, the network considered needs a fairly regular lattice-structure with small spacing between nodes. Teimoori and Savkin showed that a wheeled mobile robot can encircle a stationary node using only range to target measurements [14], but a laser range finder is needed to get an accurate enough range estimate. Woo-Yong Lee et al. presented a method for obtaining the motion direction of a robot to navigation node by calculating its rotation angle through geometric relations [15], but it requires extra ultrasonic sensors to acquire range information. Ettore Ferranti and Niki Trigoni proposed practical solutions that allow a mobile agent to identify and localize fixed sensors deployed in its vicinity, and accurately move towards a carefully selected fixed sensor [16]. This work also adopts the range method and makes use of camera technologies to localize sensor nodes and navigate towards them. Travis Mercker et al. introduced two navigation algorithms that permit a robotic vehicle equipped with an encoder and a compass to travel along a communication path established in an ad-hoc fashion [17]. It only needs additional information about the communication boundary of the static nodes. However, the navigation algorithms had been tested under the assumption of perfect communication and network operations (no signal interference effects).

The algorithms proposed in this paper aim to help a robot move from the initial location to the destination along navigation nodes only with the use of RSSI and odometry. Some properties of our algorithms which differ from previous work are summarized as follows:

1. To calculate the motion parameters for the robot, we only need the RSSI and odometry information between the mobile robot and the static navigation node on the planned path. No extra sensor is required.
2. Neither the mobile nor the static nodes have location information, and they are deployed randomly.
3. We consider the path planning algorithm for robot navigation in a real WSN, where the static nodes are cost and power limited, and where communication and node failures may occur.

The subsequent sections of this paper are organized as follows. In Section 2, the hardware used for the static sensor node and the mobile robot in this work are introduced. Section 3 analyzes the relationship between RSSI and distance. Then, a distance and robust aware path planning algorithm for robot navigation in WSN is described in Section 4. Next, the two navigation algorithms which only use the RSSI and odometry information are presented in Section 5. Section 6 presents the results of simulations of the proposed algorithms and Section 7 details the results of the practical experiments. Finally, Section 8 concludes the paper and mentions possible future works.
2. Sensor node and robot hardware

To verify the proposed robot navigation algorithms, a hardware platform was developed. The static node is shown in Figure 1a. It is an embedded system that possesses a microcontroller MSP430 and an IEEE 802.15.4 compliant radio transceiver, CC2420 [18], which uses an omnidirectional antenna. The static nodes used, based on CC2420, were an effective low-power hardware solution for the implementation of a low-cost sensor network. There are two built-in measurement parameters in the CC2420 chip: RSSI and LQI (link quality indicator), which are important for our navigation algorithms. LQI is a metric for received signal quality, which reflects non-ideal channel effects such as noise, multipath effects, carrier frequency offset, etc. The robot developed for this work is of the differential drive type. As is shown in Figure 1b, it uses a standard commercially available chassis with two wheels attached to motors mounted on each side, both equipped with an encoder. A set of IR sensors were equipped to detect and avoid obstacles. A CC2420 module is attached to act as the on-board radio for communication.

3. Relationship between RSSI and distance

The ranging technique based on RSSI is much more popular since most of today’s radio chips for WSN provide it at no extra hardware cost. RSSI is usually regarded as a kind of rough ranging technique. Reflection, multipath propagation, NLOS propagation, antenna gain and other issues will be significant for the propagation loss with the same distance. RSSI techniques are very sensitive to noise and interference. Many algorithms require calibration and a model of the environment. Nowadays, how to improve the accuracy of RSSI-based ranging is a more meaningful question. The work in [19] proves that averaging the RSSI samples collected at different carrier frequencies will mitigate the multipath fading effect, thus potentially improving the RSSI-based distance estimate.

In this paper, we use the same method as in [19] to coordinate data exchange between nodes, which exploits multichannel transmission in order to mitigate the multipath effect that hampers ranging estimation. The relationship between RSSI and distance are established online in the initialization phase of the navigation. Measurements were taken in an outdoor space, because we wanted to place the nodes in a real environment, in which the nodes would normally be operating when deployed. A static node was placed at the initial position of the mobile node and then the mobile node moved away from the static node along its initial heading over a straight distance of 20m in steps of 50cm measured by odometry. At each step, the receiver of the static node acquired different RSSI samples from 16 different available radio channels with the transmitting power with a value of 0dbm and then computed the RSSI mean of the RSSI samples; this routine was repeated 10 times [19]. The average value and the standard deviation of the RSSI mean was calculated and recorded for each distance, respectively. Thus, we can establish the relational database between the average value and the standard deviation of the RSSI mean and the distance. Experiment results are shown in Figure 2; the RSSI mean drops as the distance increases, but this relationship changes weakly as the distance increases. When the distance is greater than 15m, CC2420 cannot detect the changing relationship. Our evaluation results show RSSI to be a promising indicator when its value is above a certain sensitivity threshold.

![Figure 1](image1.jpg)

**Figure 1.** (a) static node, (b) mobile robot

![Figure 2](image2.jpg)

**Figure 2.** The measurements of the relationship between RSSI and distance

The model used here assumes that the values of the RSSI mean under a fixed distance approximately meet the Gaussian distribution expressed as follows:

\[ \text{RSSI}_k = \mu(d_k) + N(0, \sigma(d_k)) \]  

(1)

It provides a good account of the instability of the RSSI value which arises from outside interference. A least squares process is used to compute the \( \mu(d) \) and \( \sigma(d) \) functions that best fit the set of data. \( \mu(d) \) uses the log normal distribution channel of the wireless communication model, and \( \sigma(d) \) is considered approximately as a linear function, which increases slowly with the distance \( d \). The functions are expressed as follows:
\[ \mu(d) = p(d_0) - 10n_p \log \left( \frac{d}{d_0} \right), \quad \sigma(d) = ad + b \tag{2} \]

Once obtained, the functions can be used online in the navigation process. In the following algorithms, the static nodes use the RSSI mean as the RSSI value by averaging the RSSI samples from 16 different channels and then send it to the mobile robot for navigation.

4. Distance and robustness aware path planning algorithm

In this section, we attempt to solve the problem of selecting the shortest and most robust path by wireless communication.

![Figure 3. An example of the path planning for robot navigation in WSN](image)

In Figure 3, the mobile robot round static node B1 received the “Event” message from static node B10. According to the method in [20], selecting the path with the minimum hops as its planned path, the robot will move along the path L1 (B1-B2-B3-B4-B5) hop-by-hop to reach its destination. However, the path is not optimal, the main reasons are as follows:

1. Because sensor nodes are generally deployed in a random distribution, the path with the minimum hops is not necessarily the shortest by geographic distance. In Figure 3, the path L2 is shorter than L1.
2. The static nodes are generally simple and cheap devices with limited energy, the probability of the failure of such a device is larger due to the energy drain. Should one node fail, the navigation system has to search for a new path. It not only consumes large system resources but could also potentially lead to the “Event” task failing due to navigation time being too long;
3. According to Figure 2, the RSSI attenuation is very considerable over a short distance, but as the distance increases the attenuation curve changes slowly. When the one-hop distance is too long, the mobile robot has to consume more time and energy to make decisions owing to the weak signal strength attenuation. In addition, there may be stronger noise interference round some static nodes due to the dynamic environment. Therefore, all these factors are disadvantageous in providing quick navigation for the robot and may even cause a failure.

In order to guide a mobile robot to its destination quickly and reliably, this paper proposes a distance and robust aware path planning (DRAPP) algorithm. In the DRAPP algorithm, each static node should have its own neighbour RSSI tables (NRT) to maintain the RSSI information from itself to its neighbour nodes. When a mobile robot receives an “Event” message, it checks the location of where the event happened, and then initiates the DRAPP to find a path to the destination.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dst</td>
<td>The address of the destination node that detected the event.</td>
</tr>
<tr>
<td>Seq</td>
<td>A sequence number for a Path-Finding message that is incremented each time a mobile node sends a new Path-Finding message.</td>
</tr>
<tr>
<td>TD</td>
<td>Total distance</td>
</tr>
<tr>
<td>NEN</td>
<td>The number of nodes that the Path-Finding Message has passed.</td>
</tr>
<tr>
<td>LEN</td>
<td>The address list of nodes that the Path-Finding Message has passed.</td>
</tr>
</tbody>
</table>

Table 1. The format of the Path-Finding message

The mobile robot broadcasts Path-Finding messages (PFNDs) in order to conduct the path search, the format of which contains five fields as shown in Table 1. PFNDs are then delivered to all the one-hop neighbours. After receiving the message from its neighbour node, the current static node first checks if the RSSI, LQI and the power saved are greater than the predefined thresholds RSSIth, LQIth and PWRth. If these values meet the requirements, the current static node is eligible to be a navigation node, otherwise the message will be discarded. Moreover, if \( N_{EN} \) is bigger than the MaxNEN that is predefined according to the size of the network, the message will also be discarded. Then, if the current navigation node is not the destination node, it forwards the reserved PFND to its neighbour nodes after a certain number of timeouts. This process is repeated until the PFND is received by the destination node. When the destination node finally receives the PFND, a Path-ACK packet (PACK) containing LEN list is sent to the mobile robot. Thus, the robot follows the address list of the selected static navigation nodes to travel.

In the Path-Finding process, when a static node receives the PFND, it scans its NRT and updates TD, N_{EN} and LEN in the message. The update equations are as follows:

\[ TD = TD + D_{m-n} \quad N_{EN} = N_{EN} + 1 \tag{3} \]

\( D_{m-n} \) is the distance between itself and the neighbour node that sends the PFND, which can be calculated using the
RSSI value in NRT and equation (2) established online in the initialization phase. Each navigation node maintains a PFND Message Buffer to store the PFND Message with the maximum degree of distance and robustness (PDRD), the expression of which is defined as (4).

\[
\text{PDRD}_a = k_1 \left( \frac{1}{D_{\text{hop}a}} \right)^{1 / TD_a} + k_2 \left( \frac{1}{D_{\text{hop}a}} \right)^{1 / TD_b} \\
\text{PDRD}_b = k_1 \left( \frac{1}{D_{\text{hop}b}} \right)^{1 / TD_a} + k_2 \left( \frac{1}{D_{\text{hop}b}} \right)^{1 / TD_b}
\]

Let \(D_{\text{hop}}\) denote the average one-hop distance of the path that the PFND is passed along. Let \(k_1, k_2\) denote the weight of TD and \(\frac{1}{D_{\text{hop}}}\) respectively, and \(D_{\text{hop}} = \frac{\text{TD}}{\text{NIN}}, k_1 + k_2 = 1\). The subscript notation \(a, b\) denote the PFNDs from two different paths. As seen in Figure 4, Let node \(B\) have a PFNDs with a TD value of 19 and a NIN value of 3 in its PFND Message Buffer. If it receives a PFND with a TD of 23 and a NIN of 2 within a certain number of timeouts, thus PDRD, is bigger than PDRDs, it will ignore the received PFND; else if PFNDs has a PDRD bigger than PDRDs, it replaces the old PFND in its PFND Message Buffer with the new one. After a certain number of timeouts it forwards the reserved PFND to its neighbour nodes. As the PFND transmission process is repeated, only the PFND with the biggest PDRD is sent to the destination. As a result, the path consisting of the selected navigation nodes has the shortest geometrical one-hop distance and total distance. PFNDs from different paths reach the destination node through such a process. Thus, the PFND Message Buffer of the destination node is continually updated by the PFND with the maximum PDRD until the timeouts expire. Then, the destination node sends a PACK to the mobile node. The process of sending a PACK from the destination node is shown in Figure 4.

**Figure 4.** An example of the process of DRAPP

In the proposed algorithm, it is assumed that no obstacle exists in the sensor field. If the robot has sensors to detect and avoid obstacles, they can avoid the obstacle and then find a path from the current position to the destination using DRAPP. Moreover, the transmitting power and the average distance between neighbour nodes should meet the particular values required to ensure that there are enough eligible navigation nodes to compose a path to the destination. Once the mobile robot has an invalid initial path after it has begun to move toward a destination due to changes in the network environment, it has to search for a new path from other eligible neighbour nodes. Hence, there is a trade-off between the density of the static nodes and the reliability of the navigation.

5. Navigation for mobile robots

Since our goal is to guide the robot so it can move along the navigation nodes one by one, in the following illustration, we take one navigation node as an example and calculate the motion parameters needed for the robot to reach this node. If the robot can reach one static node, it just needs to repeat the foregoing procedures in order to arrive at other navigation nodes. This solution is not as simple as it might look. It must be considered that a robot, initially, does not have any idea about its heading. So, it must make some tentative motions to deduce the heading and its position relative to the navigation node. We assumed that the mobile robot is allowed to make a straight line motion and a rotation, and no errors occur in the robot’s movement.

5.1 RAC algorithm

We aim to enable the robot to find the right direction to the navigation node, within a bounded time, using RSSI and odometry. The principle is to follow the positive gradient of the RSSI field, exploiting the monotonicity of the RSSI-distance relationship and calculate its rotation angle based on range and cosine algorithm (RAC).

The basic idea of the algorithm is rather simple, to make a robot reach a navigation node along the shortest path, we just need to regulate the robot’s rotation angle in line with the node’s RSS field’s steepest ascent direction. Let \(R\), \(R_1\), \(\Delta R\), \(\Delta R_1\) denote the RSSI values and the differences between the \(R\) and \(R_1\) in the current and last position respectively, \(\text{Len}_{\text{pace}}\) denotes the length of pace, and \(\Delta_{\text{thr}}\) denotes a threshold value for the RSSI variation that indicates that the robot is moving towards the navigation node. The robot selects a random direction at the first step. If \(\Delta R_{\text{thr}}\), the robot’s rotation angle can be calculated for the next step. Otherwise, the robot keeps moving along the former direction until \(\Delta R > \Delta_{\text{thr}}\).

**Figure 5.** RAC algorithm
Figure 5 presents the details of this navigation algorithm, namely how the robot arrives at the vicinity of the navigation node. The robot is now located at A, and N is the navigation node's location, which the robot should reach. Without a map, compass or GPS module, the robot cannot recognize in which direction N lies. Thus, it moves forward to the initial direction and records the number of steps moved at the same time. When \( |\Delta R_c| > \Delta \theta_{thr} \), the robot has moved \( n \) steps and arrived at B. In order to move along BN, the RSSI field's steepest ascent direction, the mobile node just needs to turn \( \theta \); \( \theta \) can be calculated with \( R_c, R_1 \) and \( nLen_{pace} \) by the following (5).

\[
\theta = \arccos\left(\frac{\hat{d}_1(R_1)^2 - \hat{d}_c(R_c)^2 - (nLen_{pace})^2}{2d_c(R_c) nLen_{pace}}\right)
\]  

(5)

However, the robot does not know which way to turn (i.e. right along BN' or left along BN). To decide which way to turn, the robot turns left 0 tentatively and moves on until \( |\Delta R_c| > \Delta \theta_{thr} \); then if \( |\Delta R_c| > \Delta R_1 \), which means the robot is moving closer to the RSSI field's steepest ascent direction and so turning left is correct, the robot calculates \( \theta \) for the next step. Otherwise, it is confirmed that the robot has moved in the wrong direction, the robot should turn right 20 to correct its direction and in the next step the robot first turns right 0 tentatively, after which \( \theta \) is calculated. When \( R_c \) crosses the required RSSI threshold \( R_{thr} \), the mobile node arrives at the vicinity of the navigation node N; then it communicates with the next navigation node and repeats the foregoing procedures until arriving at the destination node. The RAC algorithm is described by Algorithm 1.

**Algorithm 1 RAC algorithm**

1: \( \alpha_{sign} \): positive and negative sign of 0;
2: \( n = 0, \Delta R_1 = \Delta \theta_{thr}, \theta = 90^\circ \);
3: \( R_c, R_1 \) GetCurrentRSSI;
4: while \( R_c < R_{thr} \) do
5: \( \Delta R_c > \Delta R_1 \) then
6: \( \Delta R = \Delta \theta_{thr} \); \( \theta = \arccos(\hat{d}_1(R_1), \hat{d}_c(R_c), nLen_{pace}) \);
7: \( \alpha_{sign} = -\alpha_{sign} \); \( \theta = 2\theta \); endwhile
8: \( \Delta R_c > \Delta R_1 \) then
9: \( \theta = \arccos(\hat{d}_1(R_1), \hat{d}_c(R_c), nLen_{pace}) \);
10: \( \alpha_{sign} = -\alpha_{sign} \); \( \theta = 2\theta \); endwhile
11: \( \Delta R_1 \) = \( \Delta \theta_{thr} \); \( n = 0 \);
12: Move(\( nLen_{pace} \)), \( n = n + 1 \);
13: \( R_c \) GetCurrentRSSI;
14: \( \Delta R_c = R_c - R_1 \);
15: endwhile

5.2 IMAP algorithm

The RAC algorithm provides a simple solution to finding the right direction for the navigation node. We now consider a slightly more sophisticated algorithm, which needs more storage capacity and stronger computing power. A common method to localize the navigation nodes is to use the RSSI ranging technique together with the least squares (LSQ) algorithm, which minimizes the residual of the navigation node position estimation [21]. However, the average localization error of this method is large due to the large ranging error of the RSSI ranging technique. To reduce the average localization error, we adopt a localization algorithm based on maximum a posteriori. This algorithm uses the Bayes’ formula to deduce the probability density of the navigation node’s distribution in the local target region from RSSI values. Then, the mobile node takes the point with the maximum probability density as the navigation node’s estimated location. Through simulations, we show that this algorithm outperforms the LSQ algorithm with the steps the robot travels.
$X=(x,y)$ denote the coordinates of the navigation node which is the state to be estimated. The information about the state will be obtained from the set of measurements $Z_{0:k}$ measured up to step $k$. The coordinates and heading angle of the mobile robot can be calculated with the odometric data. Then, we detail the algorithm that estimates the position of the navigation node based on the maximum a posteriori iteratively (IMAP). As shown in Figure 6, firstly, the robot records the measurement $Z_n$ in the initial position, and then it carries out a tentative step along its initial heading with a pace of length $\text{Len}_{\text{pace}}$ and records the measurement $Z_1$. Once the samples are collected, the robot estimates the navigation node position based on the maximum a posteriori (at this step, there may be two positions estimated, we choose one at random) and then moves along the computed direction. In Figure 6, after the robot’s second step, the coordinates of node $N$ are estimated as $N' (\hat{x}, \hat{y})$, and the heading angle is $\theta$. Thus, the robot can obtain its rotation angle $\alpha$ to $N'$ by following (6):

$$\beta = \arctan \left( \frac{\hat{y} - y_1'}{\hat{x} - x_1'} \right), \quad \alpha = \beta - \theta$$

(6)

After each step is taken, the measurement $Z_k$ collected at the new position is added to a queue until a maximum size $\text{Len}_{\text{queue}}$ is reached. This queue is used as a ring buffer, in which newer measurements replace older ones. When the queue has $k$ samples, we can obtain the posterior density from the following formula:

$$p(X | Z_{0:k}) = \frac{p(X | Z_{0:k}) p(X)}{p(Z_{0:k})}$$

(7)

$$p(X | Z_{0:k}) \propto p(Z_{0:k}) p(X)$$

The function $p(Z_{0:k})$ plays a very important role in the estimation process. We assume that the RSSI values under a fixed distance approximately meet the Gaussian distribution expressed as follows:

$$R_k = \mu(d_k) + N(0, \sigma(d_k))$$

(8)

$$d_k = ||X - X_k'|| = \sqrt{(x - x_k')^2 + (y - y_k')^2}$$

In this case, the function (9) expresses the probability of obtaining a RSSI value on the navigation node from the robot (at position $X_k'$) given the position of the navigation node $X$.

$$p(Z_k | X) = \frac{1}{\sigma(d_k) \sqrt{2\pi}} \exp\left(-\frac{(R_k - \mu(d_k))^2}{2\sigma(d_k)^2}\right)$$

(9)

Since the $k$ measurements are independent of each other the following formula holds for all the measurements:

$$p(Z_{0:k} | X) = \prod_{n=0}^{k} p(Z_n | X)$$

(10)

The prior consideration is then a uniform distribution on an annulus, in which the inner and outer radiuses depend on the estimated mean and variance of RSSI (see Figure 6). Therefore, the position of the navigation node can be estimated by the following equation:

$$\hat{X} = (\hat{x}, \hat{y}) \triangleq \arg \max_X p(X | Z_{1:k}) = \arg \max_X p(Z_{1:k} | X)$$

$$= \prod_{n=1}^{k} p(Z_n | X) = \prod_{n=1}^{k} \frac{1}{\sigma(d_n) \sqrt{2\pi}} \exp\left(-\frac{(R_n - \mu(d_n))^2}{2\sigma(d_n)^2}\right)$$

(11)

We define the function $f(x, y)$ as follows:

$$f(x, y) = \sum_{n=1}^{k} \frac{(R_n - \mu(d_n))^2}{2\sigma(d_n)^2} + \ln \sigma(d_n)$$

(12)

Then, the coordinates with the minimum value of $f(x, y)$ are taken as the estimated position of the navigation node.

$$\hat{X} = (\hat{x}, \hat{y}) = \arg \min_{(x,y)} f(x, y)$$

(13)

The coordinates can be calculated by the equation:

$$\frac{\partial f(X)}{\partial X} |_{(x,y)} = 0$$

(14)

Each time a new measurement is obtained, the new estimated position of the navigation node and the new rotation angle are calculated, and then the robot moves along the computed direction as it did in the last step. It repeats the foregoing procedures until $R_c$ crosses the required RSSI threshold $R_{thr}$. The IMAP algorithm is described by Algorithm 2.

**Algorithm 2 IMAP**

1: $k$: index of the ring buffer;
2: $n$: number of the samples in ring buffer;
3: $k = 0, n = 0$;
4: $Z_k, \theta_k$ $\triangleleft$ GetCurrentMeasurement;
5: $k = k + 1, n = n + 1$;
6: Move($\text{Len}_{\text{pace}}$);
7: $Z_k, \theta_k$ $\triangleleft$ GetCurrentMeasurement;
8: while $R_k < R_{thr}$ do
9: if $n < \text{Len}_{\text{queue}}$ then
10: $n = n + 1$;
11: end if
12: $(\hat{x}, \hat{y})$ $\triangleleft$ MAP ($Z_{0:n}$);
13: $\beta = \arctan(\hat{y} - \hat{y}_k', \hat{x} - x_k')$, $\alpha = \beta - \theta_k$;
14: Turn($\alpha$), Move($\text{Len}_{\text{pace}}$);
15: if $k < \text{Len}_{\text{queue}}$ then
In this section, several simulations are presented to show the performance of our proposed algorithms. We firstly evaluate the DRAPP Algorithm by discussing one-hop distance and the distance of the entire path performed. Then, simulations are performed to compare the performance of the RAC and IMAP algorithms and the influence of some related parameters are analyzed.

6.1 Simulation for DRAPP

The network dimensions for the simulations are [120m, 120m] and the static nodes (based on IEEE 802.15.4) are uniformly deployed at random within the region. To avoid communication holes among the nodes, the distance between any two nodes is kept to within 20m, which is the communication range of each node. For the purposes of our analysis, let us consider an open environment (no obstacles) so that the robot can move in a straight line between adjacent nodes. The simulations are carried out using C++.

For the purposes of comparison, we adopt the Ad hoc On-Demand Distance Vector (AODV) routing algorithm to decide the path for the robot, which finds the conventional shortest path with minimum-hop. Thus it does not minimize the geographic distance but the hops. In order to evaluate the efficiency of the DRAPP algorithm, we investigate two parameters: the average one-hop distance and the node-to-node distance of the entire path.

50 nodes were randomly placed. The green and blue lines show the planned path of DRAPP and AODV respectively. The figure indicates that despite having more hops, the path of DRAPP is closer to the geometrically shortest path from the source node to the destination node than that of AODV.

To estimate the distance of the path obtained by each algorithm, we compare the node-to-node distance of the entire path with the 2D Euclidean distance from its source node to destination node. So, we define the Path-Cost as:

\[
\text{Path\_Cost} = \frac{\text{node\_to\_node\ distance\ of\ the\ entire\ path}}{\text{Euclidean\ distance\ from\ source\ to\ destination}} \quad (15)
\]

20 simulations were performed with different source nodes and destination nodes for each algorithm. The results are presented in Figure 8. The solid line is the path of DRAPP. The dashed line is the path of AODV. Note that the paths of DRAPP and AODV are the same in 9 simulations. In other simulations, the distance of the path of DRAPP is shorter than that of AODV. Thus, the performance of each algorithm seems to be more closely related to the topology of the network, the positions of the source and destination nodes, but DRAPP always finds the shorter path owing to the FDRD in DRAPP. In particular, in simulation 9, the distance of the path of DRAPP was shorter than that of AODV by 19m.

Figure 7 shows the results of the planned paths for the robot from the source node to the destination node after

![Figure 7. The planned path of DRAPP and AODV](image)

![Figure 9. One-hop distance of AODV and DRAPP in 20 simulations](image)
In the simulations, we also compare the average one-hop distance and the maximum one-hop distance of each algorithm. As mentioned previously, one-hop distance is an important parameter since our navigation algorithms exploit the RSSI-distance relationship. According to Figure 2, the signal strength attenuation is considerable over short distances and as the distance increases the attenuation curve changes slowly. Owing to the weak signal strength attenuation over long distances, the mobile node has to move more steps to find the right direction to the current navigation node and thus consumes more time and energy. So, the shorter one-hop distance is good for the quick navigation of the robot. The results are shown in Figure 9. Clearly, the path of DRAPP has a shorter one-hop distance.

Figure 10(a) and 10(b) show the comparison of the Path-Cost and one-hop distance in different density of nodes respectively. Figure 10(a) shows the Path-Cost versus the number of nodes using AODV compared with DRAPP. The figure clearly shows that DRAPP finds shorter paths than AODV in all cases. Using DRAPP, Path-Cost decreases as the density of nodes increases, but this trend is not obvious in AODV. This is because AODV attempts to find a path with the fewest hops. Unlike DRAPP, which uses PDRD as a selection criterion, AODV has no selection criteria to reduce the geographic distance. As for DRAPP, we see that the larger the density of nodes in the field, the larger the number of the candidate paths obtained as a geographically shorter path. Thus, the path obtained by DRAPP becomes increasingly close to the distance of a straight line as the density of nodes increases. Therefore, navigation will become more efficient as the density of the sensor nodes increases. In Figure 10(b), it is expected that with the density of nodes increasing, the one-hop distance of the path obtained by each algorithm will decrease; but the path using DRAPP has a shorter one-hop distance.

6.2 Simulation for RAC and IMAP

The simulations involve only a mobile robot and a navigation node. A random position and heading are generated for the robot within one-hop communication distance of the navigation node, and both navigation algorithms are implemented until the $R_{thr}$ value is reached.

The first simulation is carried out to determine how the number of steps travelled by the mobile robot varies as the distance increases between the initial positions of the robot and the navigation node. A Monte Carlo analysis is performed for the initial distance from 5m to 15m with a step of 1m. For each distance, 50 simulations are run, and the average steps are plotted. For the simulations, the pace length of the mobile node is set to 1m, $\text{Len}_{queue}$ is set to 8 and $R_{thr}$ is set to -56dbm. The relationship between RSSI and distance is in accordance with the real data collected in the experiments.

The results are shown in Figure 11(a). Clearly, for a majority of the distances, the robot travels fewer steps using IMAP. The main problem with the IMAP algorithm is that at the beginning of its movement, the robot needs enough samples in the ring buffer fed to the IMAP to evaluate the right direction to the navigation node. This explains why the steps of the IMAP algorithm are more than those of the RAC for a shorter initial distance. As the initial distance increases so does the average number of steps. However, the signal strength attenuation becomes weaker and weaker. This leads to the average number of steps of RAC increasing more rapidly for initial distances longer than 10m. It also affects the performance of IMAP. When the robot is stuck in a location which is too far and where the samples are very similar, it is possible to estimate the direction wrongly. Once enough samples are filled in the buffer, a more precise estimation becomes possible at each step in the iterative way. From the results, it can be seen that the steps of the IMAP algorithm increase slowly and nearly linearly for distances smaller than 12m, so within these distances, the IMAP algorithm is more efficient.

![Figure 10](image_url)  
**Figure 10.** Path-Cost (a) and One-hop Distance (b) in different density of nodes

In short, the simulation results above show that DRAPP performs better than AODV regardless of the distance of the entire path or the one-hop distance despite the possibility of it having more hops than AODV.

![Figure 11](image_url)  
**Figure 11.** (a)Steps vary with the initial distance; (b) Steps vary with the $\text{Len}_{queue}$
Next, we test how the $\text{Len}_{\text{queue}}$ impacts on the number of steps. Our algorithms are compared with the ILSQ to evaluate their performance. The initial distance between the nodes is set to 8 m. It can be seen from Figure 11(b) that as $\text{Len}_{\text{queue}}$ increases, the average number of steps of IMAP and ILSQ decreases. When $\text{Len}_{\text{queue}}$ is larger than 8, both algorithms travel fewer steps than RAC. But the IMAP algorithm can achieve a smaller average number of steps with the same $\text{Len}_{\text{queue}}$.

Then, we test how noise conditions impact upon the number of steps. The ranging error is a primary cause of the direction error, which depends on the RSSI deviation: the larger the RSSI deviation, the larger the ranging error. In the simulation, the RSSI deviation varies from $1\sigma(d)$ to $2\sigma(d)$ according to the real data collected. Figure 12 shows the average steps of the RAC, IMAP and ILSQ algorithm. It can be seen that as the noise increases, RAC yields the worst performance; IMAP with a maximum $\text{Len}_{\text{queue}}$ yields the best performance. It is expected that more samples are needed to better estimate the right direction in noisier conditions. With the same $\text{Len}_{\text{queue}}$ and noise conditions, IMAP has a little better performance than ILSQ.

![Figure 12. Steps vary with the RSSI deviation from $1\sigma(d)$ to $2\sigma(d)$](image)

7. Experiment

This section details the experiments conducted to test the above algorithms. As can be seen in Figure 13, a WSN composed of 10 static nodes was deployed in a flat and open environment free from obstacles. Each node was approximately 6–8 m apart.

Firstly, the relationship between RSSI and the distance was built according to the method mentioned in Section 3. Then, the two navigation algorithms were tested on the robot. After an entire path to the destination node was set up within the sensor field using the DRAPP algorithm, the experiments were conducted under the same conditions and repeated several times, both with the RAC and IMAP algorithms. The odometry and RSSI measurement data were recorded for performance analysis. The steps of travel were recorded for each attempt as a performance metric.

![Figure 13. Experiment](image)

According to the results of the local navigation simulation, the more samples the ring buffer is filled with, the more accurately rotation can be calculated using IMAP. To solve this problem, when the robot detects the RSSI from the next navigation node greater than RSSI in the process of moving to the current navigation node, we make it prepare the other ring buffer for the next navigation node immediately and collect the samples from the next navigation node at the same time, so that it can collect more samples for the next stage of navigation in advance.

Figure 14 shows the results of a 4-hop path. The steps travelled for each hop are averaged over a series of experiments. These values are then plotted along with the one-hop distance of each hop in the path for the RAC and IMAP algorithms respectively. From an analysis of Figure 14, it is clear that the number of steps needed in the first hop by IMAP is much larger than in the later hops. This is easily explained. While in the first hop there is a lack of samples, in the following hops there are enough samples available to feed the algorithm, which makes a much more precise estimation possible.

![Figure 14. The steps of one-hop in a 4-hop path](image)

An example of the robot trajectory using IMAP is shown in Figure 15. From Figure 16, we also can see the relative monotone rise of the RSSI values using IMAP, which shows how effective the algorithm is. In addition, there is
an error introduced from using odometry in the process of navigation. However, this error is bounded since only the latest Len queue samples matter, and it are typically outrun by the RSSI noise. Using RAC, the robot sometimes travelled more steps and sometimes travelled fewer steps. In some cases, it took too more steps to approach the navigation node. We can see that it was affected by the one-hop distance and RSS deviation. This demonstrates a weak point in the RAC algorithm: the heavy dependence on a good relationship between the RSSI and the distance.

These experiment results suggest that IMAP is more efficient and outperforms the RAC algorithm in an outdoor environment.

Figure 15. An example of the trajectory of a 4-hop path by IMAP

![Figure 15](image)

Figure 16. RSSI values from each node vary with steps by IMAP

8. Conclusion

In this paper, several algorithms are proposed to allow the mobile node to navigate without the help of a map, GPS, or extra sensor devices, only using the RSSI and odometry information. In addition, navigation is possible without prior geographic knowledge of the robot and static nodes, which greatly expands its application area.

We firstly analyze the relationship between RSSI and distance, and propose the online method for establishing a Distance-RSSI database. Then, we propose the distance and robustness aware path planning (DRAPP) algorithm suitable for mobile robot navigation using the RSSI. As seen from the simulation results, compared with the conventional packet routing algorithm, the DRAPP provides a robust path with a shorter average one-hop distance and node-to-node geometrical distance of the entire path, which are the parameters of importance for robot navigation.

RAC and IMAP algorithms are developed to enable a robot to navigate through a wireless sensor network with no a priori location information in the sensor field. Using only the RSSI from static navigation nodes and odometry information, the mobile robot can successfully efficiently navigate through the field. Through simulations and practical experiments, and although both algorithms proved to be successful under ideal conditions (no obstacles, no signal interference effects) and the RAC proved to be simpler and requires less memory, the IMAP algorithm is shown to be more robust in its performance with more samples in noisier environments. In addition, the IMAP algorithm also shows a better performance than the iterative least squares minimization (ILSQ) algorithm. This work suggests that a mobile robot could indeed autonomously perform cost effective navigation within a low-cost, low-power, infrastructure-free outdoor sensor network.

However, there are certain limitations to our proposed methods. The DRAPP incurs large overheads as it relies on flooding the entire network. Therefore, we need to develop an efficient scheme to reduce the redundant forwarding packets. Furthermore, we will conduct real experiments to evaluate the performance of IMAP in the harsh multipath and shadowing environments, the robot will be equipped with an absorbing plate below the antenna to avoid interference with signals reflected by the ground [22]; a more accurate channel propagation model will be considered and studied in this scenario. We will also further evaluate IMAP by comparing the performance of IMAP and other novel location estimation approaches [23], in order to make some improvements. We leave these for future research.

9. Acknowledgments

This work was supported by the National Natural Science Foundation of PRC (No.61175100), National Natural Science Foundation of PRC (No. 51275285), Postdoctoral Science Foundation of China (No.2012M510087) and the Research Fund of Medicine and Engineering of Shanghai Jiao Tong University (No.YG2011ZD01).

10. References


