文章编号:1671-4598(2016)02-0202-03 DOI:10.16526/j.cnki.11—4762/tp.2016.02.056 中图分类号:TP393 文献标识码:A

基于垂直平分线的非测距定位算法

邹东尧,李 晨,刘碧微
(郑州轻工业学院 计算机与通信工程学院,郑州 450001)

摘要:在无线传感器网络定位机制中,近似三角形内点测试(APIT)定位算法因思想简单、硬件成本低、定位精度高而被广泛应用,但算法性能会因锚节点分布不均匀受到严重影响;为提高节点定位精度和解决APIT定位算法边缘效应的问题,提出一种基于垂直平分线的非测距定位算法;该算法利用接收信号强度指示值(RSSI)值筛选锚节点并与等腰三角形底边中垂线过顶点的几何原理结合进行定位,可以得到比较合理的定理精度,性能相对稳定;仿真结果表明,该算法可有效提高无线传感器网络定位的精度和覆盖率。

关键词:无线传感器网络;近似三角形内点测试;接收信号强度指示;定位精度;定位覆盖率

A Non-distance Localization Algorithm Based on Perpendicular Bisector

Zou Dongyao, Li Chen, Liu Biwei
(College of Computer and Communication Engineering, Zhengzhou University of Light Industry, Zhengzhou 450001, China)

Abstract: In wireless sensor network positioning schemes, APIT is widely used because it is easy, entails low hardware cost and provides high positioning accuracy. But its algorithm performance is severely affected by uneven distribution of anchors. A non-distance localization algorithm based on perpendicular bisector is proposed to improve node positioning accuracy and solve the edge effect problem of APIT. The proposed algorithm selects anchors using RSSI values and fixes the position by combining with the geometrical principle that the mid perpendicular of the base line in the isosceles triangle passes the vertex. The positioning accuracy is enhanced in this way and the stable performance is achieved. Simulation results show that the proposed algorithm can effectively improve positioning accuracy and coverage of the wireless sensor network.

Keywords: wireless sensor networks (WSNs); APIT; RSSI; positioning accuracy; localization coverage

0 引言

无线传感器网络(wireless sensor networks, WSNs)中节点的位置信息至关重要。对于无线传感器网络中所得到的感知数据,没有相应的位置信息,通常是无意义的[1-2]。实时地确定事件发生的位置也是提供安全反恐、国防军事、环境监测与预报位置信息的前提,所以定位技术对有效应用无线传感器网络起着关键作用[3-5]。

根据定位过程中是否依靠测量实际节点间的距离,定位算法可分为基于测距和基于非测距两种。基于测距的定位算法主要有AOA、TOA、TDOA、RSSI[6-7];基于非测距的定位算法主要有质心算法、DV-hop算法、Amorphous算法、APIT算法[8-13]。基于测距的定位技术比基于非测距的定位技术精确度更优,但测距过程在一定程度上加大了通信量和计算量,也会使硬件成本增加。因此,基于测距的定位技术在大规模传感器网络中受到限制,而基于非测距的定位技术硬件成本合理,能耗低,适用范围比较广泛,但定位精度较低。APIT定位算法是经典的无需测距的定位算法之一,该算法在密集网络中,定位精度较为合理,性能相对稳定,网络成本低,实现也相对容易。但在随机分布网络中会造成定位误差较大、覆盖率较低的现象。

收稿日期:2015-08-21; **修回日期**:2015-08-29。

基金项目:河南省科技厅科技攻关项目(112102210321);河南省产学研合作项目(122107000022);2014年郑州轻工业学院研究生创新基金项目。

作者简介:邹东尧(1973-),男,河南许昌人,博士,副教授,主要从事物联网信息感知、无线传感器节点定位技术方向的研究。

为了提高定位的精度和覆盖率,本文提出了一种基于垂直平分线的非测距定位算法(perpendicular bisector algorithm, PB算法)。该算法只需提取接收到的RSSI值,且不需要将其转化为具体的距离值,减小了计算复杂度,并且对锚节点分布的方向没有要求,因此提高了定位精度、解决了边缘节点无法定位的问题。

1 APIT算法存在的问题

APIT算法是一种适用于大规模无线传感器网络的分布式非测距定位算法。该算法原理是首先从未知节点周围选取3个锚节点组成一个三角形,若未知节点在三角形内部,则将其标记,按照相同原理对未知节点周围的锚节点进行不同组合组成不同三角形,并分别标记[14]。最终计算出所标记的所有三角形的重叠区域,求其质心位置即为定位结果。APIT定位算法存在的问题如下:

1) APIT定位算法只有在锚节点数量足够多时,才可将未知节点完全覆盖到锚节点构成的三角形区域中。但由于WSNs的节点是随机部署的,可能出现因某些区域分布不均匀而导致部分节点无法定位的现象,并且影响定位精度。

2) 在WSNs中某些小区域内因未知节点的邻居锚节点个数量少于3个无法构成三角形进行定位,从而形成未确定节点。当未知节点不在邻居锚节点所构成的三角形内也会成为未确定节点。

3) APIT定位算法由质心计算公式得出重叠部分的质心位置,但是由公式所得到的质心位置不是节点的准确位置,所以会造成一定的误差。

4) 通常会采用增加锚节点数量的方法解决APIT定位过

程出现的 OutToIn 或 InToOut 问题。然而锚节点数量的增加会使整个网络的能量消耗及成本增多。

2　PB 算法

2.1　算法描述

锚节点向周围节点广播自己的 ID 编号、位置信息、发射信号强度，未知节点接收通信半径内锚节点的广播信息并计算接收到的 *RSSI* 值。未知节点根据设定的阀值 $RSSI_f$，选取其接收的 *RSSI* 值中所有大于 $RSSI_f$ 的锚节点。

如图 1 所示，选择 *RSSI* 数值相差小的两组锚节点，*RSSI* 值相差小即锚节点与未知节点距离接近。例如 P_1 和 P_2，P_3 和 P_4，分别连接 P_1 和 P_2 得线段 c_1，连接 P_3 和 P_4 得线段 c_2，线段 c_1 和 c_2 即表示为两个等腰三角形的底边。分别作两个三角形底边 c_1 和 c_2 的垂直平分线 l_1 和 l_2，设两条垂直平分线相交于点 P，根据垂直平分线定理可知，点 P 即为未知节点的定位坐标。

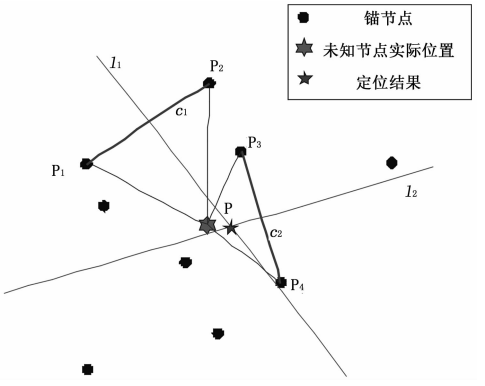

图 1　PB 定位算法

由于最终的定位结果是通过计算垂直平分线 l_1 和 l_2 的交点所得，所以当底边不合适时，对定位的结果影响较大，甚至会导致无法定位。例如若直线 l_1 和 l_2 平行则无法计算出定位坐标。如图 2 所示，此时 l_1 和 l_2 相交于无穷远处，即使两直线不平行，斜率相近也会造成误差的较大。

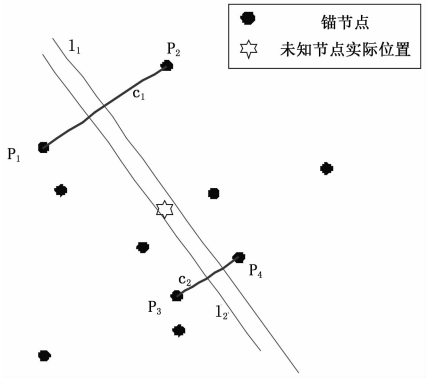

图 2　垂直平分线平行

此外若选取的锚节点过近，如图 3 所示，此时的线段 P_1P 和 P_2P 长度差距不大，但由于锚节点 P_1 和 P_2 相距过近使得偏差严重扩大，导致定位结果误差过大。因此，选取锚节点时对应的垂直平分线不能平行，还要限制选取锚节点间的距离，避免过近而造成较大的定位误差。

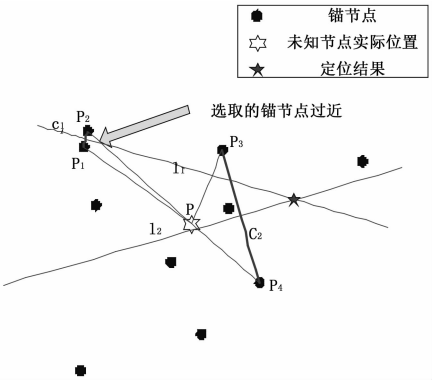

图 3　锚节点距离过近

设选定的锚节点坐标 P_1（x_1，y_1），P_2（x_2，y_2），P_3（x_3，y_3），P_4（x_4，y_4），为防止底边平行而无法定位，建立限制条件：

$$\begin{cases} |x_2-x_1|+|x_4-x_5|\neq 0 \\ \dfrac{y_4-y_3}{x_4-x_3}\neq\dfrac{y_2-y_1}{x_2-x_1} \end{cases} \tag{1}$$

为避免图 3 所示的底边过短情况的出现，建立限制条件：

$$d_i=\sqrt{(x_i-x_{i+1})^2+(y_i-y_{i+1})^2}>d_F \tag{2}$$

2.2　算法步骤

2.2.1　提取信息

各锚节点向周围节点广播自己的位置信息，未知节点接收其通信半径内或 *RSSI* 值大于阀值的锚节点信息，将锚节点的 *RSSI* 值和坐标放入矩阵 $\boldsymbol{S}$，如公式（3），其中锚节点 P_i（x_i，y_i）的 *RSSI* 值为 S_i，矩阵 $\boldsymbol{S}$ 第一行为锚节点的 *RSSI* 值，第二行第三行为对应锚节点的横纵坐标。设接收到附近 N 个锚节点的 *RSSI* 数据，那么矩阵 $\boldsymbol{S}$ 为三行 N 列矩阵：

$$\boldsymbol{S}=\begin{bmatrix} S_0 & S_1 & \cdots & S_{N-1} \\ x_0 & x_1 & \cdots & x_{N-1} \\ y_0 & y_1 & \cdots & y_{N-1} \end{bmatrix}_{3\times N} \tag{3}$$

其中：$S_i\geqslant RSSI_f(i=0,1,2,\ldots,N-1)$。

2.2.2　整理锚节点信息矩阵

对矩阵 $\boldsymbol{S}$ 按照首行 S_i 由大到小进行列排序，排序结果为 $\boldsymbol{S'}$，此时 $\boldsymbol{S'}$ 表示将锚节点按照其 *RSSI* 值由大到小排序的矩阵。将矩阵 $\boldsymbol{S'}$ 中相邻锚节点的 *RSSI* 值作差，并求绝对值 $|S_i-S_{i-1}|$。最后 $|S_i-S_{i-1}|$ 按照由小到大顺序进行列排序得到矩阵 $\overline{\boldsymbol{S}'}$。$\overline{\boldsymbol{S}'}$ 表示为锚节点间的 *RSSI* 差值从小到大排序的矩阵：

$$\overline{\boldsymbol{S}'}=\begin{bmatrix}\overline{\Delta S_a} & \overline{\Delta S_b} & \cdots & \overline{\Delta S_m}\end{bmatrix}_{5\times N-1} \tag{4}$$

2.2.3　除去不符合条件的锚节点

当选取的两个锚节点距离过近时，会增大定位的误差。根据公式（1）、（2）删除矩阵 $\overline{\boldsymbol{S}'}$ 中不符合条件的列，得到矩阵 $\boldsymbol{SS}$。

2.2.4　连接符合条件的锚节点和未知节点组成三角形，并作以两个锚节点为底边的垂直平分线，垂线的交点为定位结果。

首先选取矩阵 $\boldsymbol{SS}$ 的前两列，既相对差距最小且符合条件的两组锚节点，会得到 3 个或 4 个锚节点。设有 4 个锚节点 P_1（x_1，y_1），P_2（x_2，y_2），P_3（x_3，y_3），P_4（x_4，y_4）。将 P_1 和 P_2 相连得到直线 l_1，P_3 和 P_4 相连得到直线 l_2，l_1 和 l_2 垂直平分线的焦点为未知节点定位结果。

3 算法仿真与分析

3.1 锚节点最小距离

在 1.1 节算法描述部分，提到为了控制算法误差，设定了锚节点的最小距离参数 dF。表示在选定到未知节点距离相同或相近的锚节点时，两个锚节点的最小距离不能小于 *dF* 。为了分析参数 *dF* 的变化对定位精度的影响，仿真环境设置为 100 m×100 m 的方形区域，节点通信半径为 10 m，节点总数为 400 个，锚节点数量占节点总数的 20%、22.5%、25%……50%递增变化，每种情况测试 10 次，最终结果取其均值以减少误差，则 *dF* 在（0，12］内取值时，平均误差如图 4 所示。

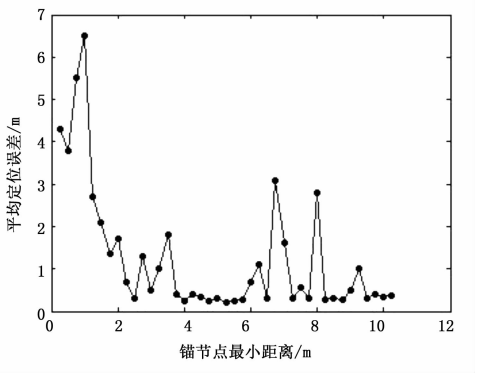

图 4 dF 与平均定位误差关系图

由图 4 可以看出，当 *dF* 小于 2 m 时定位误差非常大，此时会出现图 3 中的锚节点过近而造成的定位偏离。此现象可以通过直线的斜率公式解释，设直线 AB 上两点 A（x_1，y_1），B（x_2，y_2），若直线的斜率存在则可以表示为 $k=(y_1-y_2)/(x_1-x_2)$。点 A 和 B 比较接近时可以表示为：$y_1\approx y_2$，$x_1\approx x_2$。那么 y_1-y_2 或 x_1-x_2 有很小的变动都会导致斜率 k 的较大变化，此时误差会得到进一步扩大。因此限制所选取的锚节点距离有其必要性。

当 *dF* 大于 10.25 m 时，由于 *dF* 过大，满足条件的锚节点不足以满足定位需求而无法定位，出现图 4 中所示的后段无数据情况。当 *dF* 取值过小时，因垂直平分线可能偏离严重而造成误差较大，*dF* 取值过大时，因符合条件的锚节点过少而无法定位。而在 4 ～6 m 的取值范围内取值时，平均误差较小且变化幅度不大。因此，取 4 ～6 m 之间平均误差最小的点为最优 *dF* 值，即 dF=5.25 m。

3.2 定位精度和覆盖率

节点定位精度和覆盖率是评价一个定位算法性能好坏的关键指标[15]。在实验过程中，我们利用随机生成的场景进行仿真实验，比较两个算法在相同实验条件下的定位精度和覆盖率。在 100 m×100 m 的方形区域进行测试，节点通信半径为 10 m，节点总数为 400 个，每种情况测试 10 次得到相应数据，取其均值为最终结果。APIT 算法和 PB 算法定位误差如图 5 所示。

由图 5 可以看出，当锚节点数量达到 32.5%时，PB 算法的定位误差趋于一个定值，其定位误差低于 APIT 算法。在锚节点密度较低时，很多未知节点不能收到足够多的锚节点信息，因此 APIT 算法不能对此类节点进行定位，而 PB 算法可以对此类节点进行定位，但由于此类节点收到的锚节点信息较少，所以定对位误差较大，使 PB 算法整体定位误差提高。APIT 算法和 PB 算法定位覆盖率如图 6 所示。

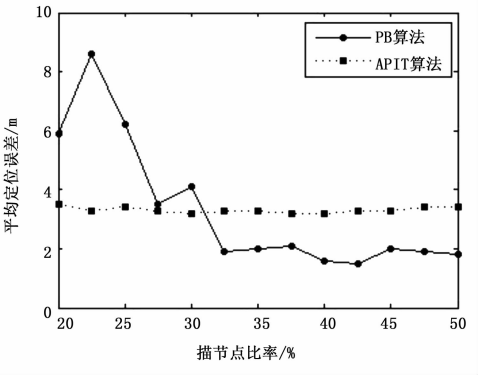

图 5 PB 算法与 APIT 算法定位误差比较

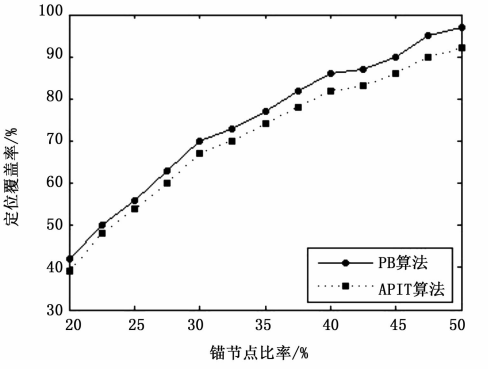

图 6 定位覆盖率比较图

由于无线传感器网络中节点的随机分布特点，导致了节点分布不均匀和边缘效应情况的出现。使用 APIT 算法定位会存在一些无法定位的节点，定位覆盖率也会随之下降。从图 6 中可以看出，两种算法随着锚节点比例的增加定位覆盖率都有所增加。由于 PB 算法只需根据锚节点建立两条直线，解出两条直线的垂直平分线交点即可，所以定位覆盖率要高于 APIT 算法。

采用 PB 算法进行定位，定位精度和覆盖率均优于 APIT 算法。此外在组网阶段，PB 算法不需要进行网格划分，形成网格阵列，从而降低了组网过程的复杂度。定位过程中 APIT 算法要进行多次区域划分，寻找重叠区域，该过程计算量大对硬件要求较高，而 PB 算法只需根据锚节点建立两条直线求其垂直平分线的交点，减少了计算难度。

4 结束语

本文提出了一种基于垂直平分线的非测距定位算法，利用 *RSSI* 值筛选锚节点并根据等腰三角形底边中垂线过顶点的几何原理进行定位，解决了 APIT 算法中锚节点分布不均匀定位误差大和边缘效应的问题。实验结果表明：在组网复杂性、计算难度、定位精度和覆盖率方面，PB 算法相比 APIT 算法均有所改善。

（下转第 208 页）