文章编号：2096-1618(2018)01-0008-05

一种基于 WKNN 定位的改进算法

康静怡， 韩中豪， 何玉美， 付 琳
(成都信息工程大学通信工程学院，四川 成都 610225)

摘要：指纹定位算法是一种基于 RSSI 的定位算法。常见的指纹定位算法包括 NN、KNN、WKNN。其中，WKNN 是带有权重的 K 最近邻法，依据每个样本点对未知节点的贡献程度给每个指纹赋予一个权值。由现有文献可知，WKNN 的定位精度优于 NN、KNN，但是仍然存在定位精度有限的问题。为进一步提高 WKNN 算法的定位精度，减少定位误差，提出一种基于 WKNN 定位的改进算法。改进算法的思路是在 WKNN 算法的基础上结合极大似然估计算法。在 MATLAB 平台下进行仿真，仿真结果表明：在相同的仿真环境下，改进算法的定位精度明显高于现有的 WKNN 算法，定位误差明显小于 WKNN 算法。

关 键 词：指纹定位；WKNN；极大似然估计；定位精度；MATLAB 仿真
中图分类号：TP301 **文献标志码**：A
doi：10.16836/j.cnki.jcuit.2018.01.002

0 引言

指纹定位是一种利用 RSSI (received signal strength indication)的测距技术[1]，配合一定匹配算法的定位方法。在定位区域取多个样点，将样点的位置信息抽象成每个样点都具备多个信标节点的 RSSI 信号强度图。信号强度大小不一，不同的信号强度代表不同的位置，将这些信息形成指纹库。当某个未知节点接收到一组 RSSI 信号强度时，与指纹库中存入的 RSSI 信号强度对比，然后用合适的指纹匹配算法计算未知节点的位置坐标[2]。典型的位置指纹匹配算法主要包括 3 种，分别是最近邻法(nearest neighbor, NN)、K 近邻法(KNN)和 K 加权近邻法(WKNN)[2-5]。

NN[5]是最基础的指纹匹配算法。它的基本原理是未知节点采集 n 个信标的信号强度，将采集的 RSSI 值与离线采集的指纹库中的数据进行匹配，匹配度最高、最相似的指纹记录的位置坐标即为未知节点的坐标。具体匹配的过程是将采集的 RSSI 值分别与指纹库中的每一条指纹对应的 RSSI 求欧式距离，求得欧式距离最小的即是未知节点匹配度最高的指纹。

KNN[3]是任何一个未知节点都可以用指纹库中最接近的 K 个邻居表示，是在 NN 的基础上为减少定位误差改进的一种算法。在现实环境中，测量 RSSI 值的偶然性较大，单独用一条指纹记录来代表未知节点坐标的误差较大。因此，取出匹配度最高的 K 个指纹，并将 K 个指纹的坐标分别相加求平均，平均值即为未知节点的估计值。

收稿日期：2017-07-27
基金项目：国家自然科学基金资助项目(41404102)

WKNN[4]是 KNN 算法的改进算法。KNN 与 WKNN 最大的不同是 KNN 算法直接对 K 个指纹的坐标求平均得到未知节点的坐标，而 WKNN 是依据每个样本点对未知节点的贡献程度给每个指纹赋予一个权值。其中贡献程度与未知节点和指纹记录之间的欧式距离紧密相关，欧式距离越小，贡献程度越大，权值就越大。

查阅文献可知，现有的指纹算法中，WKNN 定位精度优于 KNN，KNN 优于 NN。为进一步提高指纹定位算法的定位精度，减少定位误差，选择 WKNN 算法进行进一步的优化和改进，提出了 WKNN 结合极大似然估计的改进算法。

1 定位算法分析

1.1 WKNN 匹配过程

步骤 1 计算欧式距离[6]：假设指纹数据为 $(rssi_{i1}, rssi_{i2}, \ldots, rssi_{in})$，未知节点采集各个信标节点的信号强度为 $RSSI = (rssi_1, rssi_2, \cdots, rssi_n)$，计算欧式距离。

$$D_i = \sum_{j=1}^{n} (rssi_j - rssi_{ij})^2 \quad i = 1, 2, \cdots, m \qquad (1)$$

其中 D_i 表示未知节点与第 i 条指纹的欧式距离。根据式(1)分别计算未知节点 RSSI 值与 m 个指纹 RSSI 值的欧式距离，将 m 个欧式距离按从小到大进行排序，取欧式距离最小的前 K 条指纹的坐标。

步骤 2 确定权值：欧式距离越小，代表贡献程度越大，权值就越大；欧式距离越大，贡献程度越小，权值就越小[7-8]。

$$w_i=\frac{\frac{1}{D_i}}{\sum_{i=1}^{k}\frac{1}{D_i}} \tag{2}$$

其中,w_i 表示第 i 个样本点的权值。

步骤 3　估计未知节点坐标:采用 WKNN 匹配计算公式估计未知节点的位置坐标(x,y)。

$$\begin{cases} x=\sum_{i=1}^{k} w_i x_i & i=1,2,\cdots,k \\ y=\sum_{i=1}^{k} w_i y_i & i=1,2,\cdots,k \end{cases} \tag{3}$$

其中 x_i 是第 i 个样本点的横坐标,y_i 是第 i 个样本点的纵坐标,k 为选取的样本点的个数,x,y 是未知节点的横纵坐标,w_i 是第 i 个样本点的权值。

1.2　极大似然估计算法

极大似然估计定位法是无线测距测量得到未知节点到 n 个信标的距离后,利用最小二乘法来计算目标节点的位置坐标[9]。原理如图 1 所示。

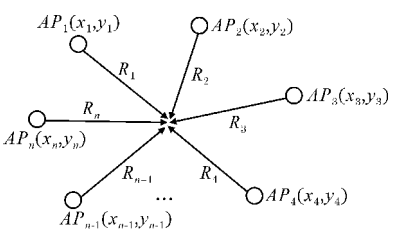

图 1　极大似然估计定位原理图

已知 n 个信标 $AP_1,AP_2,AP_3,AP_4,\cdots,AP_n$ 的位置坐标分别是(x_1,y_1)、(x_2,y_2)、(x_3,y_3),…,(x_n,y_n),未知节点为(x,y)。采用适当的测距模型根据未知节点接收到的 n 个信标节点的 RSSI 值得到未知节点到对应信标节点的距离依次是 R_1、R_2、R_3、……、R_n。求距离公式可得到以下方程组:

$$\begin{cases} (x-x_1)^2+(y-y_1)^2=R_1^2 \\ (x-x_2)^2+(y-y_2)^2=R_2^2 \\ (x-x_3)^2+(y-y_3)^2=R_3^2 \\ (x-x_4)^2+(y-y_4)^2=R_4^2 \\ \cdots \\ (x-x_n)^2+(y-y_n)^2=R_n^2 \end{cases} \tag{4}$$

将式(4)中的每个方程都用最后一个方程做减法,可得:

$$\begin{cases} x_1^2-x_n^2-2(x_1-x_n)x+y_1^2-y_n^2-2(y_1-y_n)y=R_1^2-R_n^2 \\ x_2^2-x_n^2-2(x_2-x_n)x+y_2^2-y_n^2-2(y_2-y_n)y=R_2^2-R_n^2 \\ x_3^2-x_n^2-2(x_3-x_n)x+y_3^2-y_n^2-2(y_3-y_n)y=R_3^2-R_n^2 \\ x_4^2-x_n^2-2(x_4-x_n)x+y_4^2-y_n^2-2(y_4-y_n)y=R_4^2-R_n^2 \\ \cdots \\ x_{n-1}^2-x_n^2-2(x_{n-1}-x_n)x+y_{n-1}^2-y_n^2-2(y_{n-1}-y_n)y=R_{n-1}^2-R_n^2 \end{cases} \tag{5}$$

将式(5)用线性方程表示转换成 $\boldsymbol{AX}=\boldsymbol{b}$ 的形式,其中,

$$\boldsymbol{A}=\begin{bmatrix} 2(x_1-x_n) & 2(y_1-y_n) \\ 2(x_2-x_n) & 2(y_2-y_n) \\ 2(x_3-x_n) & 2(y_3-y_n) \\ 2(x_4-x_n) & 2(y_4-y_n) \\ \cdots & \cdots \\ 2(x_{n-1}-x_n) & 2(y_{n-1}-y_n) \end{bmatrix} \tag{6}$$

$$\boldsymbol{b}=\begin{bmatrix} x_1^2-x_n^2+y_1^2-y_n^2+R_n^2-R_1^2 \\ x_2^2-x_n^2+y_2^2-y_n^2+R_n^2-R_2^2 \\ x_3^2-x_n^2+y_3^2-y_n^2+R_n^2-R_3^2 \\ x_4^2-x_n^2+y_4^2-y_n^2+R_n^2-R_4^2 \\ \cdots \\ x_{n-1}^2-x_n^2+y_{n-1}^2-y_n^2+R_n^2-R_{n-1}^2 \end{bmatrix} \tag{7}$$

采用最小二乘法求解 $\boldsymbol{AX}=\boldsymbol{b}$,求出使$||\boldsymbol{AX}-\boldsymbol{b}||^2$具有最小值的坐标$(x,y)$,$(x,y)$即为未知节点 P 的位置坐标。根据最小二乘估计法可以计算出目标节点的位置坐标为[5]:

$$\hat{X}=(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A})^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b} \tag{8}$$

2　改进算法流程

改进算法是将指纹匹配算法 WKNN 和基于测距的极大似然估计算法结合,其设计思想是在继 WKNN 取出欧式距离最小的前 K 条指纹后,继续对前 K 条指纹进行处理。整个算法分为离线采集阶段和在线定位阶段[10]。

离线采集阶段的目的是形成指纹库。

在定位区域部署 m 个信标节点并按照一定的间隔均匀选取 n 个指纹点,指纹点采集周围信标的 RSSI 数据。考虑现实环境的复杂性,同一个信标节点与指纹点之间的信号强度随时间不同而发生变化,并且有可能出现大的波动。为提高定位的准确度,对 RSSI 数据进行预处理得到更可靠的 RSSI 值。将指纹点的坐标和 RSSI 值存入指纹库,如式(9)所示。

$$\begin{bmatrix} x_1 & y_1 & rssi_{11} & rssi_{12} & rssi_{13} & \cdots & rssi_{1m} \\ x_2 & y_2 & rssi_{21} & rssi_{22} & rssi_{23} & \cdots & rssi_{2m} \\ x_3 & y_3 & rssi_{31} & rssi_{32} & rssi_{33} & \cdots & rssi_{3m} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n & y_n & rssi_{n1} & rssi_{n2} & rssi_{n3} & \cdots & rssi_{nm} \end{bmatrix} \tag{9}$$

其中 x_n、y_n 代表指纹点的位置坐标,n 代表指纹点编号,m 代表信标节点编号,$rssi_{nm}$ 表示第 n 个指纹点接收到第 m 个信标节点的信号强度值。

在线定位阶段利用离线采集建立的指纹库和未知节点接收到的 RSSI 数据对未知节点进行在线定位,主要分以下的 7 个步骤,具体流程如图 2 所示。

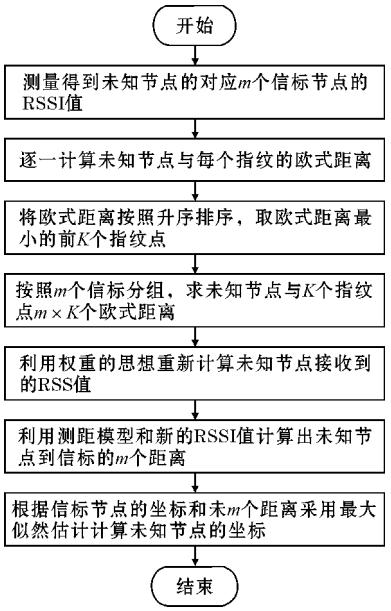

图 2 流程图

步骤 1 未知节点接收到 m 个信标节点的信号强度值,如式(10)。

$$[x_1 \quad y_1 \quad rssi_{11} \quad rssi_{12} \quad rssi_{13} \quad \cdots \quad rssi_{1m}] \tag{10}$$

其中 x_1、y_1 分别代表未知节点的横、纵坐标,$rssi_{11}$、$\cdots$、$rssi_{1m}$ 依次表示未知节点与第 1、$\cdots$、m 个信标节点之间的信号强度。

步骤 2 未知节点的 *rssi* 与指纹库中 n 个指纹的 *rssi* 形成矩阵如式(11)所示。

$$\begin{bmatrix} RSSI_{11} & RSSI_{12} & RSSI_{13} & \cdots & RSSI_{1M} \\ RSSI_{21} & RSSI_{22} & RSSI_{23} & \cdots & RSSI_{2M} \\ RSSI_{31} & RSSI_{32} & RSSI_{33} & \cdots & RSSI_{3M} \\ \vdots & & & & \\ RSSI_{N1} & RSSI_{N2} & RSSI_{N3} & \cdots & RSSI_{NM} \end{bmatrix} \tag{11}$$

把指纹库中 n 个指纹的 m 个 *rssi* 数据逐一取出来与未知节点的 m 个 *rssi* 数据利用公式(12)求 m 个信标节点产生的欧式距离之和,即求整体的欧式距离。

$$D_j = \sqrt{\sum_{i=1}^{m} (rssi_{1i} - rssi_{ji})^2} \quad j = 1,2,\cdots,n \tag{12}$$

其中,D_j 表示未知节点与第 j 个指纹的欧式距离,$rssi_{1i}$ 指的是未知节点接收到的第 i 个信标的 *rssi* 值,$rssi_{ji}$ 指的是第 j 个指纹的第 i 个信标的 *rssi* 值。求得 n 个指纹点与未知节点的整体欧式距离,形成矩阵如式(13)所示:

$$\begin{bmatrix} X_1 & Y_1 & D_1 & RSSI_{11} & RSSI_{12} & RSSI_{13} & \cdots & RSSI_{1M} \\ X_2 & Y_2 & D_2 & RSSI_{21} & RSSI_{22} & RSSI_{23} & \cdots & RSSI_{2M} \\ X_3 & Y_3 & D_3 & RSSI_{31} & RSSI_{32} & RSSI_{33} & \cdots & RSSI_{3M} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ X_N & Y_N & D_N & RSSI_{N1} & RSSI_{N2} & RSSI_{N3} & \cdots & RSSI_{NM} \end{bmatrix} \tag{13}$$

步骤 3 对式(13)的矩阵按照欧式距离从小到大进行排序,取前 K 条指纹记录,可以用式(14)所示的矩阵表示。

$$[D_1 \quad D_2 \quad D_3 \quad \cdots \quad D_K] \tag{14}$$

步骤 4 将前 K 条指纹的 *rssi* 按照信标节点的个数分成 m 组,每一个信标节点的 K 个 *rssi* 都与未知节点接收到对应信标的 *rssi* 求欧式距离,可以利用式(15)计算。

$$d_{ij} = \sqrt{(rssi_j - rssi_{ij})^2} \quad i = 1,2,\cdots k, j = 1,2,\cdots,m \tag{15}$$

其中,d_{ij} 表示的是第 i 个指纹点接收的第 j 个信标节点的 *rssi* 与未知节点接收到的第 j 个信标节点的 *rssi* 的欧式距离。求得 K 个指纹点与未知节点对应信标 *rssi* 的欧式距离,形成式(16)所示的矩阵。

$$\begin{bmatrix} D_{11} & D_{12} & D_{13} & \cdots & D_{1M} \\ D_{21} & D_{22} & D_{23} & \cdots & D_{2M} \\ D_{31} & D_{32} & D_{33} & \cdots & D_{3M} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ D_{K1} & D_{K2} & D_{K3} & \cdots & D_{KM} \end{bmatrix} \tag{16}$$

步骤 5 将矩阵(16)中的每一列分成一组,每一组利用式(2)给每列的每个元素计算一个权重,按照 WKNN 的思想对每个信标 K 个 *rssi* 值计算得到一个新的值,并将这个新的 *rssi* 值作为未知节点接收的信标节点的 *rssi* 值。计算公式为

$$rssi_j = \sum_{i=1}^{k} w_i rssi_{kj} \quad j = 1,2,\cdots,m \tag{17}$$

其中 $rssi_j$ 表示未知节点接收的第 j 个信标节点的 *rssi* 值,依次计算得到未知节点 m 个新的 *rssi* 值,可以用式(18)所示的矩阵表示。

$$[RSSI_1 \quad RSSI_2 \quad RSSI_3 \quad \cdots \quad RSSI_M] \tag{18}$$

步骤 6 将未知节点计算的 m 个新的 *rssi* 值,根据测距模型的计算公式 $d = 10^{\frac{A-RSSI}{10n}}$ [10] 计算得到未知节点到 m 个信标的距离值,可以用式(19)所示的矩阵表示。

$$[D_1 \quad D_2 \quad D_3 \quad \cdots \quad D_M] \tag{19}$$

步骤 7 已知 m 个信标节点的坐标和未知节点到 m 个信标节点的距离,然后利用极大似然估计的公式(8)计算得到未知节点的位置坐标。

3 仿真与分析

采用 MATLAB 对 WKNN 及改进的算法进行仿真,仿真场景设置如下:

(1)设定矿井仿真区域的大小为200 m×200 m的面积。

(2)整个仿真区域设置 36 个信标节点,400 指纹点,横坐标和纵坐标每隔40 m部署一个信标节点,指纹点的部署则是每隔10 m选取一个采样点,未知节点随机分布在仿真区域内。

(3)设置无线传感器网络节点的通信半径为 $20\sqrt{2}$ m,传感器节点的通信半径是根据信标节点部署的间隔进行设定的。

如图 3 所示,“+”表示信标节点,“○”表示指纹

点,“ * ”表示随机产生的未知节点的坐标。

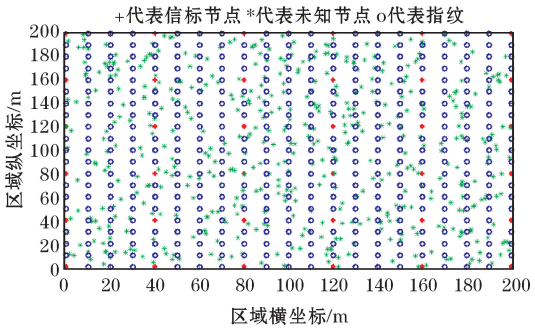

图 3　仿真场景

经过仿真测试:在此仿真环境下 $K=3$ 时的性能更好。所以,在后续的仿真中 K 值为 3。

仿真 1　在仿真场景中任意选择 8 个未知节点进行坐标估计,然后与实际坐标值比较。

表 1　未知节点的实际坐标与估计值

序号	实际坐标	WKNN	改进算法
1	(159,69)	(157,66)	(159,69)
2	(175,142)	(183,136)	(181,136)
3	(156,125)	(153,126)	(154,127)
4	(140,166)	(141,166)	(140,165)
5	(84,44)	(86,45)	(86,43)
6	(128,63)	(143,49)	(137,54)
7	(15,43)	(17,45)	(15,44)
8	(76,5)	(73,6)	(74,7)

仿真表明,对随机选择的 8 个未知节点进行坐标估计,改进后的算法坐标估计值比 WKNN 的估计值更准确。

仿真 2　在仿真环境中任意选取 500 个未知节点,分别采用 WKNN 和改进算法估计坐标值,并与实际坐标值计算估计定位误差。

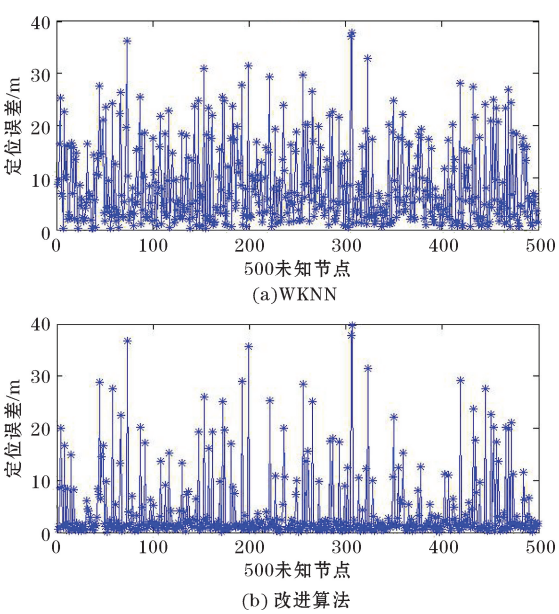

图 4　$K=3$ 时 WKNN、改进算法的定位误差

由图 4 可知:改进算法的定位误差相较于 WKNN 更小并且集中。仿真得到,在此仿真环境下,WKNN 的平均定位误差为8.27 m,改进算法的定位误差为 4.03 m。仿真结果表明,改进算法有效减少了定位误差。

仿真 3　随机选取 500 个未知节点,分别采用 WKNN 和改进算法估计坐标值并得到定位精度。如图 5 所示。横坐标是 500 个未知节点,纵坐标是未知节点定位得到的定位精度。这里纵坐标得到的定位精度是未知节点的定位误差与 $20\sqrt{2}$ 的比值。

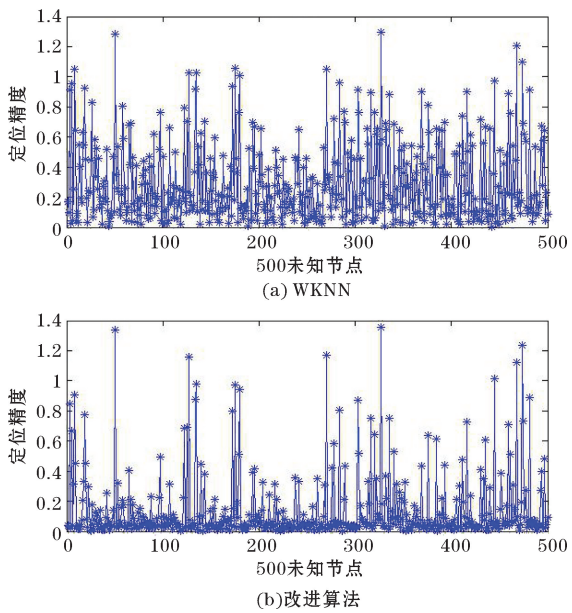

图 5　$K=3$ 时 WKNN、改进算法的定位精度

仿真结果表明,改进算法得到的未知节点的定位精度相比于 WKNN 更加集中且较小。

仿真 4　如图 6 所示是 $K=3$ 时得到的 WKNN 及改进算法误差范围概率分布图。其中横坐标是误差范围,纵坐标是误差范围内的概率分布。

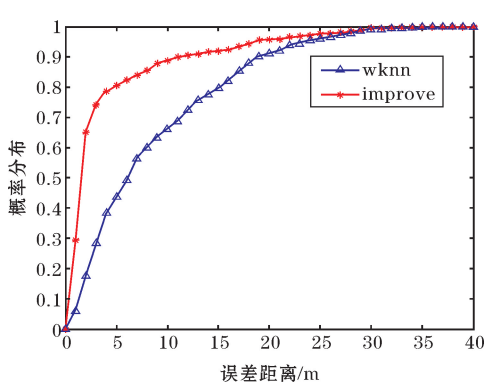

图 6　WKNN、改进算法的定位误差概率分布

仿真得到,WKNN 算法的定位误差是8.555 m,改进算法的定位误差降低到4.577 m。由图 6 可得,当定位误差为10 m时,WKNN 的概率分布大概是0.65,而改进算法的概率分布大概是0.87。显然,在误差距离相同时,改进算法比 WKNN 具有更好的概率分布。

4 结束语

主要对指纹匹配算法 WKNN 及基于 WKNN 的改进算法进行了 MATLAB 仿真和分析。仿真结果表明,在相同的仿真环境下,改进后的算法相比于 WKNN 有效的减少了定位误差。但是在现实环境中,定位区域可能远远大于200 m×200 m的仿真范围,并且文中信标节点是按照一定的规律设置的,具有一定的局限性,如何做到信标节点随机分布并且未知节点的定位误差能够有效减少成为下一步的研究内容。

参考文献:

[1] 李论. 基 RSSI 的煤矿巷道高精度定位算法研究[D]. 徐州:中国矿业大学,2015.
[2] 李艳华. 基于 RSSI 与地磁场的室内混合指纹定位算法研究[D]. 长沙:湖南师范大学,2015.
[3] Stoks, Klomp, Terheggen. Construction of high-quality NN potential models[J]. Physical review Nuclear physics,1994,49(6):2950.
[4] Bell. KNN Model-Based Approach in Classification[J]. Lecture Notes in Computer Science,2003,28(8):986-996.
[5] Gholoobi,Stavrou. RSSI Based Localization Using a New WKNN Approach[J]. IEEE Robotics & Automation Magazine,1998,15(2):14-18.
[6] 戴欢. 无线传感器网络定位算法及其应用研究[D]. 无锡:江南大学,2012.
[7] 朱登科. 基于 RSSI 的无线传感器网络测距和定位技术研究[D]. 长沙:国防科学技术大学,2010.
[8] 许建波. 基于 WLAN 位置指纹的室内定位技术研究与实现[D]. 北京:北京工业大学,2014.
[9] 王思雪. WiFi 位置指纹定位技术应用算法研究[D]. 北京:中国地质大学,2016.
[10] 肖婷婷. 基于 RSSI 测距的室内定位算法研究及改进[D]. 南昌:江西师范大学,2012.

An Improved Algorithm based on WKNN Positioning

KANG Jing-yi, HAN Zhong-hao, HE Yu-mei, FU Lin
(College of Communication Engineering Chengdu University of Information Technology, Chengdu 610225,China)

Abstract:Fingerprint localization algorithm is a localization algorithm based on RSSI. Common fingerprint localization algorithms include NN, KNN and WKNN. Among them,WKNN is K-nearest neighbor method with weight, which gives each fingerprint a weight according to the contribution of each sample point to the unknown node. According to the existing literature, WKNN has better positioning accuracy than NN and KNN,but there are still some problems with limited positioning accuracy. To further improve the positioning accuracy of WKNN algorithm and reduce positioning error, an improved algorithm based on WKNN positioning is proposed. The idea of improved algorithm is to use the maximum likelihood estimation algorithm after WKNN algorithm. Simulated in MATLAB platform, the results show that in the same environment, the localization accuracy of the improved algorithm is significantly higher than that of the existing WKNN algorithm, and the positioning error is significantly smaller than the WKNN algorithm.

Keywords:fingerprint localization; WKNN; maximum likelihood estimation; localization accuracy; MATLAB simulation