U.P.B. Sci. Bull., Series C, Vol. 85, Iss. 3, 2023 ISSN 2286-3540

PATTERN-BASED TRILATERATION POSITIONING ALGORITHM WITH LOW COMPUTING COST

Yong SHI[1,*], Zhaoling CHU[1], Yan QU[1], Guiyang BIAN[1]

Trilateral positioning with low computing cost is suitable for the large-scale promotion of location-based service. The paper proposed a novel trilateral positioning method for practical engineering applications to reduce the computational workload. Firstly, proposes six-zone patterns, and defines the vertex coordinates and anchor node deployment identification. On this basis, the positioning process is divided into the zone confirmation stage, which is actually to find three anchor nodes; and divided into locate calculation stage, which is optimized according to the pattern. A simulation experiment shows the effectiveness of the proposed algorithm. When the anchor scale is 2000, the zone finding time is reduced by 50%, the trilateral positioning time is reduced by 6.4%, and the overall process is reduced by 6.8%. When the anchor scale is 5000, the zone finding time is reduced by 68%, the trilateral location time is reduced by 10.9%, and the overall process is reduced by 15.9%. When the anchor scale is 10000, the zone finding time is reduced by 58%, the trilateral positioning time is reduced by 8.9%, and the overall process is reduced by 16.3%. So it can reduce energy consumption and prolong the working time of mobile devices.

Keywords: Trilateral Positioning, Computing cost, Zone Pattern, Pattern Confirm, Simplify

1. Introduction

With the Internet of Things and intelligence terminal industry developed, the Received Signal Strength Indicator (RSSI) based indoor positioning system has got vast concern as an essential technological infrastructure [1,2]. The positioning implementation methods can be divided into two categories: the trilateration-based positioning method and the fingerprint-based positioning method [3,4]. The fingerprint positioning has higher accuracy [5-7], but from the process perspective of engineering, it needs to collect and maintain huge amounts of fingerprint data, and it needs to match these fingerprints [8] which workload is relatively complex especially when in a large scenario. On the other side, the trilateration-based positioning is vastly applied for it can achieve acceptable accuracy [9], it does not need a lot of data acquisition and maintenance work, and it is relatively simple and direct in engineering with advantages such as wide deployment, low cost, and high flexibility.

[1] School of Computer Information and Engineering, Changzhou Institute of Technology, Changzhou 21300, China
*Correspondence: Yong SHI, shiy@czu.cn

So, this paper will contribute to further reducing the workload of trilateration-based positioning method, to promote and adapt to large-scale applications, considering that practical accuracy would unpredictably diminish due to energy consumption [10,11]. In this paper, an algorithm is proposed to simplify the positioning process based on zone pattern, and the contributions as follows:

(1). Propose six zone patterns, and sensors deployment, and improve the target area search and location calculation process.

(2). Propose improved trilateration positioning algorithm, including pattern Template confirmation, and algorithm simplification.

(3). Take a simulation experiment, furtherly discuss the advantage compared to the traditional positioning, and discuss the improvement of the proposed algorithm.

The remainder is arranged as follows. Section II gives a review of RSSI-based trilateration. Section III details the proposed zone patterns, and the trilateration algorithm based on it Section IV introduces a test to analyze positioning performance and compares performance with traditional positioning. Finally, Section V summarizes and concludes the paper.

2. Literature Review

Generally, trilateration-based positioning method includes an offline stage and an online stage [12]. The offline stage includes steps such as anchor sensor deployment and path-loss exponent fitting [13]. The online stage includes steps such as RSSI collecting real-time, distance estimation, and coordinates calculating [14].

2.1. The Offline Stage

The primary goal in the stage is to map RSSI into the distance. In practical applications, researchers usually use piecewise fitting to obtain the propagation coefficient, for interferences from the environment and multipath propagation cannot be avoided, as shown in formula (1), and map RSSI to distance (formula 2) [11]:

$$c = \alpha - 10\beta \log(d_{fix}) \quad (1)$$

$$d_n = 10^{((c-\alpha)/(10*\beta))} \quad (2)$$

Where c is the path-loss exponent, α is the RSSI at some referenced location (usually 1 m), the β is the environment exponent generally with a value in the range of 1.6–1.8 in an indoor environment, d_{fix} is the difference fixed distance for β collected; and d_n is the distance from the current tag to source beacon.

2.2. The Online Stage

The primary goal in the stage is positioning. Fig 1 illustrates the positioning of fixed three anchors and the current tag(x,y). As a simplified engineering positioning process, it usually involves three steps. Firstly, the appropriate three anchor nodes are selected, and then the distance from the tag to these anchor nodes is calculated. Finally, the trilateral positioning algorithm is constructed [12,15].

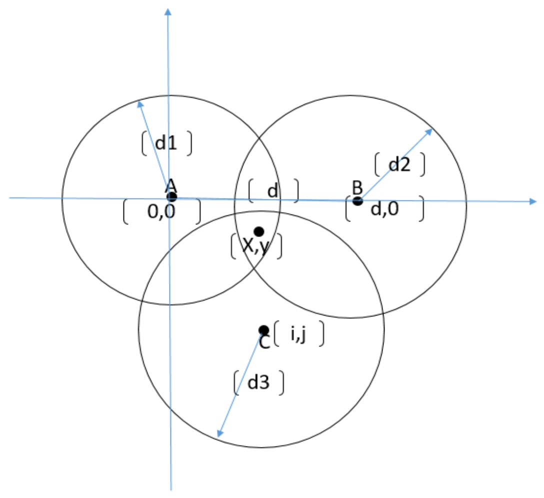

Fig. 1. Simplified Trilateration Diagram

The trilateration algorithm is as follows according to Fig1.

(1) Set selected beacons original coordinate as A0(x1,y1), B0(x2,y2) ,C0(x3,y3),and set A(0,0), B(d,0) ,C(i, j).

(2) Calculate distance from A to B as d, distance from A to C as dac, distance from Cto B as dbc using Euclidean distance function.

(3) Where C(i, j) is calculated as follows, $\begin{cases} i^2 + j^2 = {d_{ac}}^2 \\ (i-d)^2 + j^2 = {d_{bc}}^2 \end{cases}$ (3)

(4) Using the formula (2) to calculate d1, d2, d3.

(5) Using The Least square and Maximum likelihood [15,16] to calculate point(x, y).

$$\begin{cases} x = \left({d_1}^2 - {d_2}^2 + d^2\right)\big/_{2d} \\ y = \left({d_1}^2 - {d_3}^2 - x^2 + (x-i)^2 + j^2\right)\big/_{2j} \end{cases} \quad (4)$$

The whole positioning algorithm can be divided as the three anchors finding and the coordinate calculating process. The proposed algorithm is expected to achieve breakthroughs from these two aspects.

3. Positioning method based proposed pattern

3.1. Patterns

As shown in the left of Fig. 2, where six zone patterns are proposed named Pattern A-Pattern F. Patterns A and B are equilateral triangles with a side length is L, Patterns C, D, E and F are right triangles with a side length is L and another L/2, and the other side length is$L\sqrt{3}/2$.In practice, the pattern is used to segment the location area, as shown in the right of Figure 2, and deploy anchors at the vertices of each pattern.

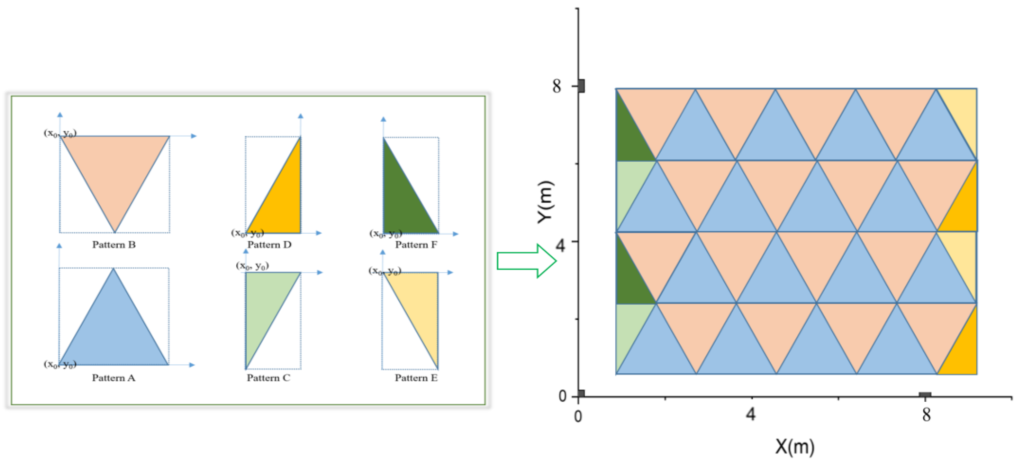

Fig. 2. Proposed Zone Pattern

In Fig.2, The origin of the coordinates is the lower left corner, where the vertices of each template, expressed in rows and columns, are defined as follows.

$$\text{Vmn}=(x_m, y_n) \in \left\{\left(\frac{L}{2}i, \frac{L\sqrt{3}}{2}j\right), i, j = 0,1,2,\ldots\right\} \tag{5}$$

Where V is the vertex and mn is row m and column n, And the deployed anchor at this vertex is,

$$\text{Amn}=\left((x_m, y_n), M_i\right) \tag{6}$$

Where A is the anchor, and(x_m, y_n) is the location of Vmn, M_i is the MAC values of the Amn.

3.2. Pattern conformed

The pattern-conformed process is designed the same as the three-anchor finding process, as shown follows.

```
Algorithm 1. Pattern Confirm
Input: Anchers = {Amn, m,n=0,1,2,···}
```

```
        // the deployed anchors, according to formula(6)
        SamplingRSSI = {RSSIi, i=0,1,2,⋯}
        // where i is the id of some Anchor according to formula(6)
Output: PatternVertexes={Vx1y1,Vx2y2,Vx3y3}
Define:
         getDistence(RSSIi);
         //gets the distance from current location rssi to sampling collection
         MaxR(SamplingRSSI)=i
         // Minimum distance function, get from the sampling to Anchor
         MaxA(i)=m,n
         // change m,n to i according to formula(6)
         PatternVertexes = ϕ  // temporal collections of pattern vertex
         AnchersTemp= ϕ   // temporal collections of anchors
First:
     For each (RSSIi∈ SamplingRSSI)
        { // get the first anchors from samplingRSSI
                 i=MaxR(getDistence(RSSIi));
                 MaxA(i)=m,n
                 // add RSSIi to temp
                 AnchersTemp ={ Akp,k=m±1,p=n±1}
                 //
                 PatternVertexes={ Amn }
         }
Second:
          For each (RSSIi∈ AnchersTemp)
         { // get the second anchors from AnchersTemp
              i=MaxR(getDistence(RSSIi));
                 MaxA(i)=m1,n1
                 AnchersTemp ={ Akp,k=m-1,p=n or k=m1,p=n-1}
                 PatternVertexes={ Amn , Am1n1 }
         }
Third:
          For each (RSSIi∈ AnchersTemp)
         { // get the third anchors from AnchersTemp

             i=MaxR(getDistence(RSSIi));
                MaxA(i)=m2,n2
                PatternVertexes={ Amn , Am1n1,Am2n2 }
```

```
                  }
        EXIT(-)
```

3.3. Positioning Simplify

After the pattern is selected, the trilateral positioning can be carried out using formula (4). Under the support of the template, where,

$$\boldsymbol{d = L\ (Pattern\ A\ or\ B) or\ L/2 (Pattern\ C, D, E\ or\ F), i = \frac{l}{2}, j = \sqrt{3}L/2} \quad (7)$$

So the formula (4) can be simplified as follows.

When Pattern A or B,

$$\begin{cases} x = \left(d_1{}^2 - d_2{}^2 + d^2\right)\Big/_{2d} = \dfrac{\left(d_1{}^2 - d_2{}^2\right)}{2L} + \dfrac{L}{2} \\ y = \left(d_1{}^2 - d_3{}^2 - x^2 + (x-i)^2 + j^2\right)\Big/_{2j} = \dfrac{\left(d_1{}^2 - d_3{}^2\right)}{\sqrt{3}L} - \dfrac{2\left(d_1{}^2 - d_2{}^2\right)}{\sqrt{3}} + \dfrac{\sqrt{3}L}{4} \end{cases} \quad (8)$$

When Pattern C, D, E or F,

$$\begin{cases} x = \left(d_1{}^2 - d_2{}^2 + d^2\right)\Big/_{2d} = d_1{}^2 - d_2{}^2 + \dfrac{L}{4} \\ y = \left(d_1{}^2 - d_3{}^2 - x^2 + (x-i)^2 + j^2\right)\Big/_{2j} = \dfrac{\left(d_1{}^2 - d_3{}^2\right)}{\sqrt{3}L} - \dfrac{\left(d_1{}^2 - d_2{}^2\right)}{\sqrt{3}} + \dfrac{\sqrt{3}L}{6} \end{cases} \quad (9)$$

4. Performance Evaluation

4.1. Experimental Design

This paper designs a simulation program to simulate the scale of anchor nodes, verify the effectiveness under different scales of the calculation solution, and judge and confirm the computational burden of the algorithm through the execution time.

The experimental design is as follows:

1. Simulation platform, a desktop workstation with Windows X64, CPU 1.9G, memory 8G, and JDK 20.

2. Simulate the positioning with 2000, 5000, and 10000 anchor nodes respectively, where the numbers reflect the scale.

3. Randomly select the target point T and set the starting point S of the algorithm.

4. For each simulation scale:

(1) Anchors finding, with the recommended coarse positioning method and traditional method.

(2) Location computing, with the proposed coordinate solving method and traditional method.

5. Repeat the process 5 times.

6. Positioning results and comparative analysis.

4.2. Positioning results

Tests show that the proposed algorithm can achieve the same positioning result. Table 1, Table 2, and Table 3 give the calculation time results when the anchor point scales are 2000, 5000, and 10000, in milliseconds. And as a comparison, we selected the time for selecting three anchor nodes and the time for calculating coordinate points.

Table 1

Comparison when anchor scale is 2000

Test	Anchors finding time (ms)		Location computing time (ms)	
	Proposed method	Traditional method	Proposed method	Traditional method
1st	0	2	20	22
2nd	1	1	21	21
3rd	1	2	20	21
4th	1	2	20	22
5th	1	1	21	23
average	0.8	1.6	20.4	21.8

Table 2

Comparison when anchor scale is5000

Test	Anchors finding time(ms)		Location computing time(ms)	
	Proposed method	Traditional method	Proposed method	Traditional method
1st	1	5	21	25
2nd	1	2	20	23
3rd	1	4	22	24
4th	2	3	21	24
5th	1	5	22	23
average	1.2	3.8	21.2	23.8

Table 3

Comparison when anchor scale is10000

Test	Anchors finding time(ms)		Location computing time(ms)	
	Proposed method	Proposed method	Proposed method	Traditional method
1st	3	4	24	24
2nd	1	3	21	23
3rd	2	5	23	25
4th	2	6	24	26
5th	1	6	21	26
average	2	4.8	22.6	24.8

The results in the above tables demonstrate that the proposed algorithm may save a lot of time:

(1) When the anchor scale is 2000, the anchor finding time is 0.8ms, while the traditional time is 1.6ms, and the location time is 20.4ms, while the traditional time is 21.8ms.

(2) When the anchor scale is 5000, the anchor finding time is 1.2ms, while the traditional time is 3.8ms, and the location time is 21.2ms, while the traditional time is 23.8ms.

(3) When the anchor scale is 10000, the anchor finding time is 2ms, while the traditional time is 4.8ms, and the location time is 22.6ms, while the traditional time is 24.8ms.

4.3. Performance analysis

Further analysis reveals that when the anchor scale is 2000, the anchors finding stage saving time is 0.8ms, the location stage saving time is 1.4ms, and the whole process saving time is 2.2ms; when the anchor scale is 5000, the anchors finding stage saving time is 2.6ms, the location stage saving time is 2.6ms,and the whole process saving time is 5.2ms; when the anchor scale is 10000, the anchors finding stage saving time is 2.8ms, the location stage saving time is 2.2ms, and the whole process saving time is 5ms, as shown in Table 4.

Table 4

Time-saving of the traditional and proposed algorithms

Anchors scale	Anchors finding	Location computing	Whole process
2000	0.8 ms	1.4 ms	2.2 ms
5000	2.6 ms	2.6 ms	5.2 ms
10000	2.8 ms	2.2 ms	5ms

The analysis also reveals that when the anchor scale is 2000, the anchor finding time is reduced by 50%, the location time is reduced by 6.4%, and the overall process is reduced by 6.8%. When the anchor scale is 5000, the anchor finding time is reduced by 68%, the location time is reduced by 10.9%, and the overall process is reduced by 15.9%. When the anchor scale is 10000, the anchor finding time is reduced by 58%, the location time is reduced by 8.9%, and the overall process is reduced by 16.2%, as shown in Table 5.

Table 5

Time-saving rate of the traditional and proposed algorithms

Anchors scale	Anchors finding	Location computing	Whole process
2000	50%	6.4%	6.8%
5000	68.4%	10.9%	15.9%
10000	58.3%	8.9%	16.2%

5. Conclusions

This paper proposed and evaluated a simplified trilateration-based positioning method, which ensures the efficiency and practicability of the indoor positioning system. Experiments have confirmed that our proposed algorithm can effectively reduce the computation time. Especially when the positioning system needs zone-level positioning accuracy, the proposed three-anchor finding can save more than 50% of the computation time. And when needs specific coordinates, the proposed location calculation in this paper can save more than 6% of the computation time. So it can reduce energy consumption and prolong the working time of mobile devices and is suitable for the large-scale promotion of location-based service.

Acknowledgments

This work was supported in part by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 15KJB170006), in part by Jiangsu Collaborative Innovation Center for Cultural Creativity (Grant No. XYN1702, No. XYN1703. in part by Changzhou Science and technology support program of China (Grant No. CE20172023), in part by National College Students' innovation and entrepreneurship training program (NO. 202211055038E), in part by Jiangsu College Students' innovation and entrepreneurship training program (NO. 202211055015Z, NO. 202211055042Y, NO. 202211055064Y).

REFERENCES

[1] *Ferreira, A.G.; Fernandes, D.; Catarino, A.P.; Monteiro, J.L*. Performance analysis of ToA-based positioning algorithms for static and dynamic targets with low ranging measurements. Sensors 2017, 17, 1915.

[2] *Ruan, L.; Zhang, L.; Zhou, T.*; Long, Y. An improved Bluetooth indoor positioning method using dynamic fingerprint window. Sensors 2020, 20, 7269.

[3] *B. Wang et al*, A Novel Weighted KNN Algorithm Based on RSS Similarity and Position Distance for Wi-Fi Fingerprint Positioning. in IEEE Access, vol. 8, pp. 30591-30602, 2020.

[4] *S. Subedi, H. Gang, N. Y. Ko, S,* Hwang and J. Pyun. Improving Indoor Fingerprinting Positioning with Affinity Propagation Clustering and Weighted Centroid Fingerprint. IEEE Access, vol. 7, pp. 31738-31750, 2019.

[5] *Chen Yunfei, Du Taihang, Jiang Chundong, et al.* Optimal processing of singular values of *K*-means clustering algorithm in indoor location. Science Technology and Engineering, 2018,18(10): 95-99.

[6] C*handrawanshi, Veervrat Singh, Tripathi, Rajiv Kumar, and Pachauri, Rahul*. An Intelligent Energy Efficient Clustering Technique for Multiple Base Stations Positioning in a Wireless Sensor Network. An Intelligent Energy Efficient Clustering Technique for Multiple Base Stations Positioning in a Wireless Sensor Network, 1 Jan. 2019: 2409-2418.

[7] *Zhang, W., Hua, X., Yu, K., Qiu, W., Zhang, S. and He, X. (2019)*. A novel WiFi indoor positioning strategy based on weighted squared Euclidean distance and local principal gradient direction. *Sensor Review*, Vol. 39 No. 1, pp. 99-106.

[8] *C. Zhang, F. Zhang and L. Zhao*, Research and Optimization of BLE Fingerprint Indoor Positioning Algorithm Based on Fusion Clustering. *2019 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, Dalian, China, 2019, pp. 95-100.

[9] Ilçi, V.; Gülal, E.; Alkan, R.M. Performance comparison of 2.4 and 5 GHz WiFi signals and proposing a new method for mobile indoor positioning. Wirel. Pers. Commun. 2020, 110, 1493–1511.

[10] *Suining He, S.-H. Gary Chan.* Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons. IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 18, NO. 1, FIRST QUARTER 2016:466-491.

[11] *A. Neishaboori and K. Harras.* Energy saving strategies in WiFi indoor localization. In *Proc. ACM MSWiM*, 2013, pp. 399–404.

[12] *SHI YONG, SHI WENZHONG, LIU XINTIAO, etc.* An RSSI Classification and Tracing Algorithm to Improve Trilateration-Based Positioning. Sensors, 2020, 20, 4244:1-17.

[13] *Xiuyan Zhu, Yuan Feng.* RSSI-based Algorithm for Indoor Localization. *Communications and Network*, 2013, 5, 37-42.

[14] *Fen Liu, Jing Liu, Yuqing Yin, etc.* Survey on WiFi-based indoor positioning techniques. *IET Commun.*, 2020, Vol. 14 Iss. 9, pp. 1372-1383.

[15] *Yong SHI, Long YI, Zezhong Xu, etc.* Indoor RSSI Trilateral Algorithm Considering Piecewise and Space-Scene. 2017 IEEE International Conference on Smart Cloud, 278-282.