Journal of System Simulation

Volume 32 | Issue 1 Article 4

1-17-2020

Low-complexity APIT Algorithm and Its OPNET Simulation of Underwater Acoustic Sensor Networks

Jiahui Xu
Key Laboratory of Underwater Acoustic Communication and Marine Information Technology of Ministry of Education, Xiamen University, Xiamen 361005, China;

Keyu Chen
Key Laboratory of Underwater Acoustic Communication and Marine Information Technology of Ministry of Education, Xiamen University, Xiamen 361005, China;

En Chen
Key Laboratory of Underwater Acoustic Communication and Marine Information Technology of Ministry of Education, Xiamen University, Xiamen 361005, China;

Low-complexity APIT Algorithm and Its OPNET Simulation of Underwater Acoustic Sensor Networks

Abstract

Abstract: Due to the energy limitations of underwater acoustic sensor networks, low-complexity location algorithms are more suitable for underwater acoustic sensor networks. The traditional APIT algorithm can obtain better location accuracy with less control overhead, which is beneficial to the location of underwater sensor networks, but it has high complexity and large redundancy errors. This paper *proposes a low-complexity APIT algorithm* replaced the traditional grid SCAN algorithm with a point scanning method, and *builds an underwater acoustic sensor network environment on the OPNET platform, and elaborates the implementation process of the location algorithm in underwater sensor network*. Simulation results show that increasing the density of the unknown nodes and anchor nodes helps to improve the algorithm. The algorithm in this paper has higher location accuracy than the traditional APIT algorithm in the same condition.

Keywords

underwater acoustic sensor networks, APIT algorithm, OPNET(Optimized Network Engineering Tool), the SCAN algorithm

第 32 卷第 1 期 系统仿真学报© Vol. 32 No. 1
2020 年 1 月 Journal of System Simulation Jan., 2020

水下传感网络的低复杂度 APIT 算法及 OPNET 仿真实现

许佳慧，陈柯宇*，程恩
(厦门大学水声通信与海洋信息技术教育部重点实验室，福建 厦门 361005)

摘要：由于水声传感网络具有能量的局限性，所以低复杂度的定位算法更适用于水声传感网络。传统的 APIT 算法能够以较少的控制开销获得较好的定位精度，有利于水下传感网络定位的实现，但其复杂度高，冗余误差较大。以点扫描的方式取代传统网格扫描法，*提出一种低复杂度的 APIT 算法，并在 OPNET 平台上搭建水声传感网络环境，阐述该算法在水下传感网络节点定位的实现过程。*仿真结果表明，待定位节点与锚节点密度的增加有助于改善算法的性能，且在同等条件下本文算法比传统 APIT 算法定位精度更高。

关键词：水下传感网络；APIT 算法；OPNET(Optimized Network Engineering Tool)；网格扫描法
中图分类号：TN929.3 文献标识码：A 文章编号：1004-731X (2020) 01-0027-08
DOI: 10.16182/j.issn1004731x.joss.17-CACIS007

Low-complexity APIT Algorithm and Its OPNET Simulation of Underwater Acoustic Sensor Networks

Xu Jiahui, Chen Keyu, Chen En*
(Key Laboratory of Underwater Acoustic Communication and Marine Information Technology of Ministry of Education, Xiamen University, Xiamen 361005, China)

Abstract: Due to the energy limitations of underwater acoustic sensor networks, low-complexity location algorithms are more suitable for underwater acoustic sensor networks. The traditional APIT algorithm can obtain better location accuracy with less control overhead, which is beneficial to the location of underwater sensor networks, but it has high complexity and large redundancy errors. This paper *proposes a low-complexity APIT algorithm* replaced the traditional grid SCAN algorithm with a point scanning method, and *builds an underwater acoustic sensor network environment on the OPNET platform, and elaborates the implementation process of the location algorithm in underwater sensor network.* Simulation results show that increasing the density of the unknown nodes and anchor nodes helps to improve the algorithm. The algorithm in this paper has higher location accuracy than the traditional APIT algorithm in the same condition.

Keywords: underwater acoustic sensor networks; APIT algorithm; OPNET(Optimized Network Engineering Tool); the SCAN algorithm

引言

水下传感网络是由大量的传感节点组成，并利用传感节点收集处理数据，实现所需要的功能。传感节点的定位是水下传感网络实现多数应用的基础，有着举足轻重的地位。定位算法大致可以分为两类：基于测距(Range-based)定位算法和无需测距(Range-free)定位算法。其中，Range-based 定位算法定位精度高，但其对硬件设备要求高，功耗和成本大。由于部署于水下的传

收稿日期：2017-05-08 修回日期：2017-07-25;
基金项目：国家自然科学基金(61501386，61471308，61671398)，福建省自然科学基金(2017J05109)；
作者简介：许佳慧(1993-)，女，福建泉州，硕士生，研究方向为水声通信与网络；陈柯宇(通讯作者 1985-)，男，河北邯郸，博士，工程师，研究方向为水声通信与网络。

感节点具有能量有限和不易替换性，所以在设计水下传感网络节点定位算法时需要将能耗作为重要指标[1-2]。Range-free 定位算法无需距离和角度信息，仅根据网络连通性等信息实现定位，虽然定位精度不如 Range-based 定位算法，但可以依靠传感节点相互协作实现粗定位，满足大多数应用的需求，如路由和目标跟踪等，所以考虑传感节点体积、成本、能量以及水下环境的限制下，Range-free 定位算法在对定位精度要求不高，且低成本、低能耗和低复杂度的水下传感网络节点定位应用中具有优势。Range-free 定位算法包括质心算法、APIT 算法、DV-Hop 算法、Amorphous 算法等[3-6]。相比其他的非测距定位算法，APIT 算法定位精度高，且通信开销小，但传统 APIT 算法利用网格扫描法后，所得的最大多边形重叠区域形状未知，因此增加了算法寻求重心的复杂度。本文提出一种低复杂度的 APIT 算法，该算法不同于传统 APIT 的网格扫描法，而是利用构成最大重叠区域的所有点的坐标求算术平均，并用该值作为待定位节点的估计位置。

1 近似三角形内点测试法

1.1 传统 APIT 算法概述

近似三角形内点测试(APIT)法是一种典型的非测距定位算法，其理论基础是三角形内点测试(PIT)法。这 2 种方法具有共同的特点，即待定位节点从所有能够监听到的锚节点中任选 3 个构成三角形，然后判断该节点与三角形的位置关系，遍历所有三角形后得到最大重叠区域，然后计算该区域的重心，以此作为待定位节点的坐标[4]。但是 PIT 和 APIT 判断节点与三角形位置关系的方法不同。

PIT 的基本原理：若存在这样一个方向，当待定位的节点沿着该方向运动时将会同时接近或者远离三角形的 3 个顶点，那么判定待定位节点在三角形外部，反之，在三角形内部。PIT 原理如图 1 所示。

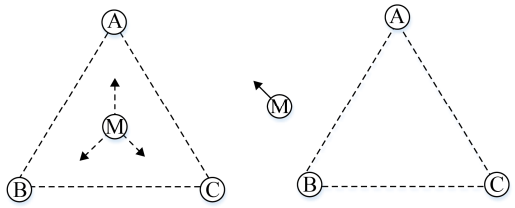

图 1 PIT 原理图
Fig. 1 Theory of PIT

该方法存在一定的缺陷，即实际上无法确认是否遍历了所有的方向，且待定位节点通常为固定的节点，不可移动，故 APIT 算法修改 PIT 判断方法，即待定位节点若存在某一邻居节点，与待定位节点相比，该邻居节点同时接近或者远离三角形的 3 个顶点，则判定该待定位节点在三角形外部，反之在三角形内部。APIT 的原理图如图 2 所示。

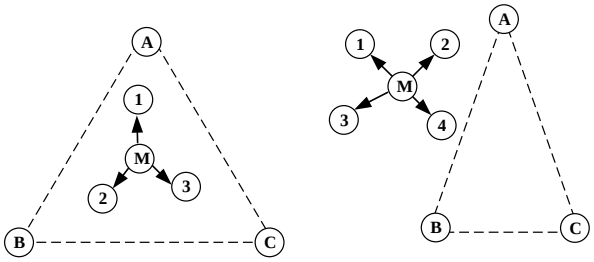

图 2 APIT 原理图
Fig. 2 Theory of APIT

因为对于任一节点，其发射的功率随着传播距离的增加而减少，故当节点 M 比较发现邻居节点从某个锚节点接收的信号强度大于自身收到的信号强度，则表明节点 M 比邻居节点更靠近该锚节点的位置，反之则远离。

在表 1 中，其中 M 表示待定位节点，1，2,…，*n* 表示节点 M 的 *n* 个邻居节点。根据上述原理，将 M 接收到的 A，B，C 三个锚节点的信号强度分别与其所有邻居节点收到这 3 个锚节点的信号强度进行对比，当节点 M 收到的 3 个信号强度都大于或者小于某个邻居节点接收的信号强度，那么就认为待定位节点 M 位于三角形的外部；反之，节点 M 位于三角形的内部。如表 1 所示，邻居节点 *n* 接收到的锚节点 A，B，C 的信号强度都大于 M 接收到的信号强度，因此，与节点 *n* 相比，待

DOI: 10.16182/j.issn1004731x.joss.17-CACIS007

定位节点 M 同时接近 A，B，C 三个锚节点，即节点 M 位于三角形的外部。

表 1 待定位节点 M 与其邻居节点的信号强度
Tab. 1 Signal strength of unknown node M and its neighbors /mV

锚节点	M	1	……	*n*
A	1	2		6
B	2	3		7
C	3	1		7

1.2 传统的网格扫描法

网格扫描法(Grid SCAN algorithm)的基本思想是用一个矩阵来表示待定位节点有可能出现的区域。具体做法：将整个网络按照一定的步长分为若干个网格，并利用 APIT 算法判断待定位节点与三角形的关系。若待定位节点在三角形内部，则将三角形内所有的网格的值加 1，反之则减 1。待遍历所有的三角形后，具有最大值的网格所构成的区域就是待定位节点有可能出现的最大区域。如图 3 中黑色的三角形是待定位节点，而阴影区域就是待定位节点可能出现的最大多边形区域。

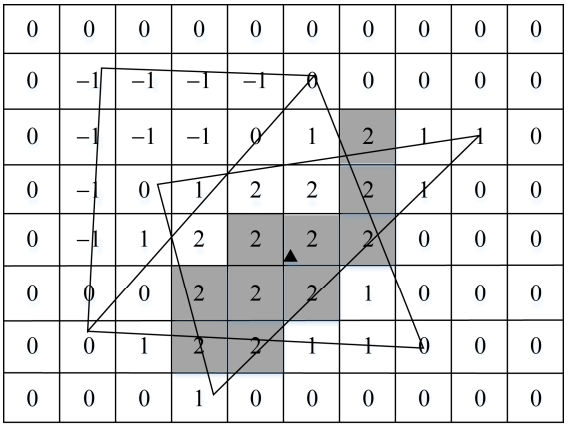

图 3 传统的网格扫描法
Fig. 3 Traditional grid SCAN algorithm

1.3 低复杂度的 APIT 算法

传统的网格扫描法是以单位网格进行扫描，扫描的结果是以网格构成的多边形区域，该多边形的形状未知，故难以求得该多边形重心的复杂度。本文基于这个问题，提出以网格顶点为单位进行扫描，即网格点扫描法，然后采用 APIT 算法判断未知节点与三角形的位置关系，若待定位节点在三角形内部，则将三角形内所有的点的值加 1，反之则减 1。待遍历所有的三角形后，将具有最大值点的坐标求算术平均，作为待定位节点的坐标。如图 4 中黑色的三角形是待定位节点，黑色圆点即为最大值的点。

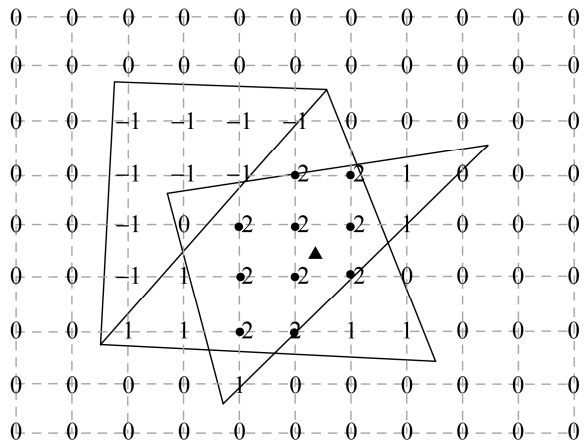

图 4 低复杂度 APIT 算法
Fig. 4 Low-complexity APIT Algorithm

传统的 APIT 算法以网格为单位进行扫描，由于判断网格是否在三角形内部没有一个固定的标准且当网格不够小时，无疑会引入一定的冗余；在算法的实现上，该算法先按点划分网络，再由每相邻的 4 个点构成一个网格，而计算重叠区域的重心时又需要将网格还原成点的形式，这使得算法的复杂度大大提高。本文提出的低复杂度 APIT 算法以点为单位，判断点是否在三角形内部时不存在疑义，缩小了交集区域，解决了传统 APIT 算法可能出现的冗余问题；此外，该算法在具体实现上省去了降维和升维的过程，相比于传统的 APIT 算法，算法复杂度大大降低。

2 APIT 算法节点建模

本文利用 OPNET 平台实现低复杂度 APIT 的定位过程[7]，网络的拓扑结构中只有 2 种相互独立的节点模型，即锚节点和待定位节点。本节将详细阐述这 2 种节点的建模过程。

2.1 创建包格式

在低复杂度 APIT 算法中，锚节点需要向所有待定位节点广播数据包(“APIT_tx_address”)，告知自身的 ID 以及坐标，并且邻居节点需要向所有待定位节点广播数据包(“APIT_neighbor_power”)，告知自身 ID 以及其能够监听到的所有锚节点的能量。数据包的包格式如图 5~图 6 所示。

n_id (4 bit)	tx_x_position (10 bit)	tx_y_position (10 bit)

图 5 锚节点包格式定义(“APIT_tx_address”)
Fig. 5 Package format definition of anchor node

n_id (8 bit)	n_anchor_power (inherited bits)

图 6 待定位节点包格式定义(“APIT_neighbor_power”)
Fig. 6 Package format definition of unknown node

在图 5 中，“n_id”用于存储锚节点的 ID，其大小为 4 bit，故锚节点的可以从 0~16 进行编号；“tx_x_position”和“tx_y_position”用于存储锚节点的位置，其大小为 10 bit，故可以表示的最大范围为 1024×1024。在图 6 中，“n_id”用于存储待定位节点的 ID，其大小为 8 bit，故待定位节点的可以从 0~256 进行编号；“n_anchor_power”内存储待定位节点所能接收的所有锚节点的能量，其属性为“packet”，即该字段是一个无格式的数据包。

2.2 锚节点

2.2.1 锚节点模型

锚节点将会向所有的待定位节点广播携带自身信息的数据包，此节点包含包生成模块“gen”、包处理模块“tx_proc”和无线发射机“tx”。图 7 为锚节点的节点模型。

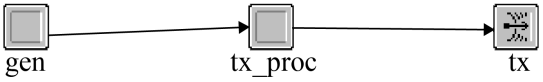

图 7 锚节点模型
Fig. 7 Model of anchor node

将包生成模块“gen”中的“process model”修改为“simple_source”，目的是使“gen”能够按照一定的时间间隔产生数据包，在“Packet Format”设置产生的数据包格式为“APIT_tx_address”；无线发射机“tx”的传输速率为 1024 bps，发送包格式“APIT_tx_address”带宽为 2 kHz，最小频率为 0.03 MHz，并将管道模型更改为水声管道模型。

2.2.2 进程模型

包处理模块“tx_proc”的进程模型如图 8 所示，其目的是将锚节点的信息写入来自包生成模块“gen”的数据包中发送出去。其中，状态“init”初始化所有的全局变量，当满足条件“PKT_ARVL”时，即收到来自“gen”模块的包，引发流中断，状态由“idle”转变为“tx_pkt”，执行状态“tx_pkt”的入口函数，该函数将锚节点的 ID 和坐标存入数据包中，并通过无线发射机发送出去。

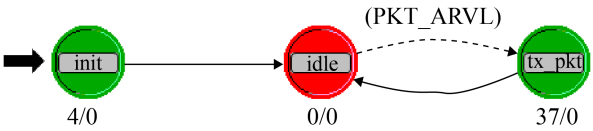

图 8 锚节点进程模型
Fig. 8 Process model of anchor node

状态“tx_pkt”的入口函数为：

```
out_pkt = op_pk_get (IN_STRM);
tx_node_addr = op_topo_parent (op_id_self());
op_ima_obj_attr_get (tx_node_addr, “x position”, &src_nx);
op_ima_obj_attr_get (tx_node_addr, “y position”, & src_ny);
op_ima_obj_attr_get (tx_node_addr, “user id”, & tx_id);
op_pk_nfd_set (out_pkt, “tx_x_position”, src_nx);
op_pk_nfd_set (out_pkt, “tx_y_position”, src_ny);
op_pk_nfd_set (out_pkt, “n_id”, tx_id);
op_pk_send (out_pkt, OUT_STRM);
```

2.3 待定位节点

2.3.1 待定位节点模型

待定位节点接收来时锚节点的包，也接收来自邻居节点的包，同时待定位节点也需要发送自身接收的所有锚节点能量数据包。此节点包含无线接收机 rx、数据处理模块 rx_proc 和无线发射机 rt。待定位节点的节点模型如图 9 所示。

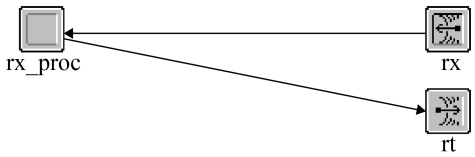

图 9 待定位节点模型
Fig. 9 Model of unknown node

这里需要注意，无线发射机与无线接收机的信道属性必须与锚节点的无线发送机一致，即传输速率为 1 024 bps，带宽为 2 kHz，最小频率为 0.03 MHz，也需要将管道模型更改为水声管道模型，模拟水下环境。但不同的是，无线接收机“rx”中“Packet Format”设置能接收的数据包为“APIT_tx_address”，“APIT_tx_address”；而无线发射机“rt”中“Packet Format”设置能接发送的数据包只有“APIT_tx_address”。

2.3.2 进程模型

数据处理模块“tx_proc”的进程模型如图 10 所示，其功能是解析接收的数据包，提取来包的 ID，坐标以及能量信息，并发送自身能量信息。其中，状态“init”初始化全局变量。

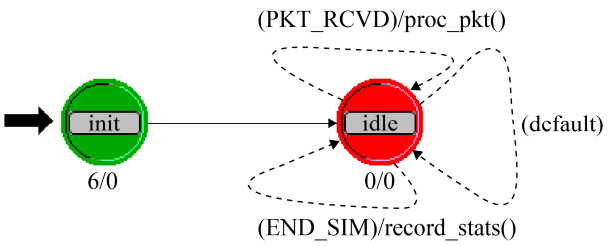

图 10 待定位节点进程模型
Fig. 10 Process model of unknown node

当满足条件“PKT_RCVD”，即待定位节点接收到数据包时，引发流中断，执行函数“proc_pkt()”，该函数根据接收包的 ID 判断发包者是锚节点还是邻居节点。

1) 若是锚节点，则存储锚节点的坐标信息，提取能量；

```
rcvd_power = op_td_get_dbl (in_pkt, OPC_TDA_RA_SNR);
op_pk_nfd_get (in_pkt, “tx_x_position”, & tx_nx);
op_pk_nfd_get (in_pkt, “tx_y_position”, & tx_ny);
```

然后将该未知节点接收到的所有锚节点能量存入自定义的无格式包中，并设置当收到 3 个(或以上)不同的锚节点数据包时，则发送自身能量数据包；

```
if (anchor_number>2) {
    send_pkptr =
    op_pk_create_fmt (“APIT_neighbor_power”);
    send = op_pk_create (7*10);
    for (w=0; w<10; w++) {
        op_pk_fd_set
        (send,w,OPC_FIELD_TYPE_DOUBLE,
        all_anchor_power[self_id][w],7);
        op_pk_nfd_set (send_pkptr, “n_id” ,
        self_id);
        op_pk_nfd_set_pkt (send_pkptr,
        “n_anchor_power”, send);
        op_pk_send (send_pkptr,OUT_STRM);}}
```

2) 若接收的包来自邻居节点时，则提取邻居节点收到的所有锚节点的能量信息。

```
else if (rcvd_id >= 100) {
    neighbor_nid = rcvd_id - 100;
    if ((in[neighbor_nid]<1) && (rcvd_id != self_id)) {
        in[neighbor_nid] = 1;
        for (s=0; s<10; s++) {
        op_pk_nfd_get (in_pkt,“n_anchor_power”,
        &neighbor_anch or_power);
        op_pk_fd_get (neighbor_anchor_power,s,
```

```
&neighbor_power[neighbor_number][s]); }
++neighbor_number;}
if (in[neighbor_nid] ==1)
in[neighbor_nid] = 1;}
```

当满足条件“END_SIM”，即仿真时间结束时，执行“record_stats()”，该函数根据 APIT 算法求出待定位节点的坐标，并计算定位误差。

1) 首先按照一定的步长划分网络，并存储所有顶点的坐标。本文构建了一个 1 000 m×1 000 m 的水下传感网络，网格的边长选取 10 m，故网络可以划分为 10 000 个面积为 100 m^2 的网格。也就是有 10 201 个顶点。

```
for (u=0; u<= 100; u++) {
    for (v=0; v<= 100; v++) {
        G_x[u][v] = 10*v;
        G_y[u][v] = 10*u;}}
```

2) 然后对这 10 201 个点进行测试，判断每个顶点与三角形的位置关系。本文采用叉积的运算方法判断点与三角形的关系[9]，其基本原理是：沿着三角形的边按顺时针的方向走，判断测试点与每条边的位置关系，若该点在每条边的右侧，那么点在三角形内，否则在三角形外。这种算法避免了除法运算，开方运算，以及三角函数，仅仅利用 3 次叉乘，故效率高，精度也高。具体过程如下：

若三角形的 3 个顶点坐标为 A(x_1,y_1)，B(x_2,y_2)，C(x_3,y_3)，待测试点的坐标为 M(x,y)。

step 1：求出 3 个向量坐标 MA，MB，MC；

step 2：计算 MA×MB，MB×MC，MC×MA(其中，×表示叉乘)

step 3：若上步中的 3 组向量叉乘的结果是同号的(同为正或同为负)，则说明点 M 位于三角形每条边的同侧，即点位于三角形内部，否则，必在外部。

```
for (p=0; p<= 100; p++){
    for (q=0; q<= 100; q++){
    G_Cross_ab[p][q] = (anchor_x[i]-G_x[p][q]) * (anchor_y[j]-G_y[p][q]) - (anchor_y[i]-G_y[p][q]) * (anchor_x[j]-G_x[p][q]);
    G_Cross_bc[p][q] = (anchor_x[j]-G_x[p][q]) * (anchor_y[k]-G_y[p][q]) - (anchor_y[j]-G_y[p][q]) * (anchor_x[k]-G_x[p][q]);
    G_Cross_ca[p][q] = (anchor_x[k]-G_x[p][q]) * (anchor_y[i]-G_y[p][q]) - (anchor_y[k]-G_y[p][q]) * (anchor_x[i]-G_x[p][q]); }}
```

其中：i，j，k 表示待定位节点接收的锚节点的 ID 号，这 3 个锚节点必须满足不在同一直线上这一前提。若 G_Cross_ab[p][q]、G_Cross_bc[p][q]及 G_Cross_ca[p][q]为同号，则该点位于三角形的内部。

3) 接着利用邻居节点的能量信息判断待定位节点与三角形的位置关系。

```
for (d=0; d < neighbor_number; d++) {
    div_e_a [d] = anchor_power[i] -
    neighbor_power[d][i];
    div_e_b [d] = anchor_power[j] -
    neighbor_power[d][j];
    div_e_c [d] = anchor_power[k] -
    neighbor_power[d][k];
    if ((div_e_a[d] > 0 && div_e_b[d] > 0
&&div_e_c[d] > 0) || (div_e_a[d] < 0 && div_e_b[d]
    < 0 && div_e_c[d] < 0))
    decision[d] =1;
    else
    decision[d] = 0;
    sum_decision = sum_decision + decision[d];}
```

其中：i，j，k 表示待定位节点接收的锚节点的 ID 号，这 3 个锚节点也必须满足不在同一直线上这一前提。若至少存在一个邻居节点从这 3 个锚节点接收到的信号能量，同时大于或小于待定位节点接收到的能量，即 sum_decision>0 时，则待定位节点位于三角形的外部，否则，在三角形外部。

4) 最后对所有最大值点的坐标求算数平均值，作为待定位节点的坐标，并计算定位误差。其中，定位误差为所有待定位节点的估计值与实际值

的差值的平均值，并定义归一化平均定位误差为平均定位误差与通信半径 R 的比值：

$$\overline{error} = error/R = \sum_i \sqrt{(\tilde{x}_i - x_i) + (\tilde{y}_i - y_y)} \Big/ R \quad (1)$$

```
for(h=0; h< max_grid_number; h++){
    sum_x = sum_x + max_l[h]*10; }
for(l=0; l< max_grid_number; l++){
    sum_y = sum_y + max_h[l]*10; }
APIT_x = sum_x/max_grid_number;
APIT_y = sum_y/max_grid_number;
rx_node_addr = op_topo_parent(op_id_self());
op_ima_obj_attr_get (rx_node_addr,"x position",
&rx_nx);
op_ima_obj_attr_get (rx_node_addr,"y position",
&rx_ny);
error_xy=sqrt(((APIT_x-rx_nx)*(APIT_x-rx_nx)+
        (APIT_y - rx_ny)*(APIT_y - rx_ny)))/R;
```

3 网络模型及结构分析

本文构造了一个 1 000 m×1 000 m 的水下传感网络，传播距离 1 000 m。锚节点分别设为 4 和 5 个，未知节点数由 5~20 变化，采用本文提出的低复杂度 APIT 算法对传感节点进行定位。

图 11 为本文所提出的低复杂度的 APIT 定位误差图，其中，纵坐标为归一化平均定位误差。观察发现，锚节点数为 4 和 5 时，随着待定位节点数的增加，网络的定位精度都提高了，这是因为邻居节点数增多时，降低了待定位节点与三角形位置关系的误判概率。并且与锚节点数目为 4 的网络相比，锚节点数为 5 的定位精度更高。这是因为锚节点数目增多时，缩小了最大重叠区域。在仿真过程中，当待定位节点以及锚节点数目进一步增多，例如，锚节点增加至 6 个时，有些待定位节点接收的锚节点数目少于 3 个，或者接收不到邻居节点的信息，使得这些节点无法定位，降低了定位精度。这是因为网络负载大时，碰撞也随之增多，致使定位效果变差。

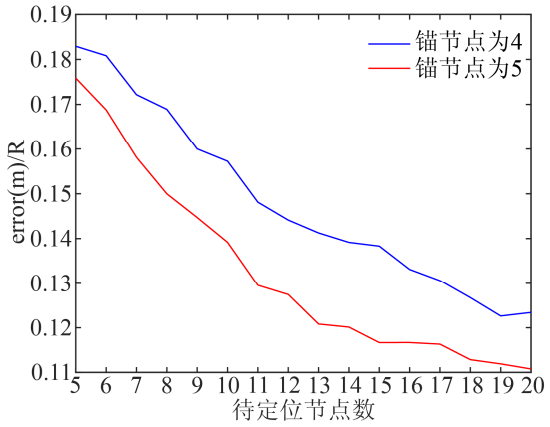

图 11 低复杂度的 APIT 定位误差

Fig. 11 Location error of the low-complexity of APIT algorithm

图 12 比较了本文所提的 APIT 算法与传统 APIT 算法的定位性能，其中，锚节点数保持 5 个不变，未知节点数从 3 个到 20 个变化，纵坐标为归一化平均定位误差。观察发现，本文所提出的算法不仅能够获得比传统 APIT 算法更好的定位性能，而且其大大降低算法的复杂度，减少能量开销，这在水下传感网络中是很重要的。

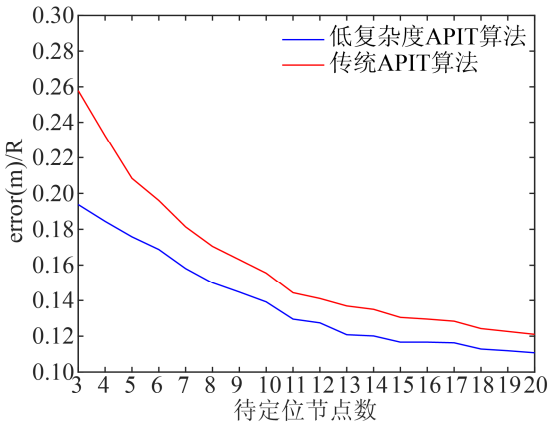

图 12 低复杂度的 APIT 算法与传统 APIT 算法定位误差

Fig. 12 Location error of the low-complexity APIT algorithm and the traditional APIT algorithm

4 结论

本文通过在 OPNET 平台上构建了接近于真实环境的水下传感网络，并在该网络中对所提出的基于水下传感网络的低复杂度 APIT 算法进行建模仿真，总结如下：

(1) 本文提出的低复杂度 APIT 算法是基于传统 APIT 算法提出的，将网格扫描法修改为网格顶点扫描法，直接利用重叠区域所有内点的算术平均值作为待定位节点的坐标，降低了算法的复杂度。

(2) 影响 APIT 算法精度的因素有：传输距离、部署方式、锚节点和待定位节点的个数等，在理想条件，即无冲突碰撞条件下，传输距离越远，锚节点数目越多，待定位节点数目越多，则定位误差越低。但是在实际仿真过程中，若增加邻居节点数目，将会降低待定位节点与三角形位置关系的误判概率，若增加锚节点的数目，将会缩小最大重叠区域，然而当网络负载很大时，碰撞也随之增加，此时增加锚节点或者未知节点的数目时，并没有预期的效果，甚至会出现恶化。

(3) 未来的工作将围绕着碰撞展开，预计将在网络中加入 MAC 协议，以分配信道资源，降低碰撞率，使定位算法更精确、高效。

参考文献:

[1] Erol-Kantarci M, Mouftah H T, Oktug S. A Survey of Architectures and Localization Techniques for Underwater Acoustic Sensor Networks[J]. IEEE Communications Surveys & Tutorials (S1553-877X), 2011, 3(13): 487-502.

[2] Akyildiz I F, Pompili D, Melodia T. Underwater Acoustic Sensor Networks: Research Challenges[J]. Ad Hoc Networks (S1570-8705), 2005, 3(3): 257-279.

[3] Bulusu N, Heidemann J, Estrim D. GPS-less cost outdoor localization for very small devices[J]. IEEE Personal Communication (S1558-0652), 2000, 7(5): 28-34.

[4] He T, Huang C, Blum B M, et al. Range-free localization and its impact on large scale sensor networks[J]. ACM Transaction on Embedded Computing Systems (S1539-9087), 2005, 4(4): 877-906.

[5] Niculescu D, Nath B. DV-base positioning in Ad-hoc networks[J]. Telecommunication Systems (S1018-4864), 2003, 22(1/4): 267-280.

[6] Nagpal R, Shrobe H, Bachrach J. Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network[C]. Information Processing in Sensor Networks, Second International Workshop, IPSN 2003, Palo Alto, CA, USA, April 22-23, 2003, Proceedings. 2003: 333-348.

[7] 陈敏. OPNET 网络仿真[M]. 北京: 清华大学出版社, 2004.
Chen Min. OPNET network simulation[M]. Beijing: Tsinghua University press, 2004.

[8] Yalewoo. 向量叉积和应用: 判断点是否在三角形内部[EB/OL]. [2017-03-03]. http://www.yalewoo.com/in_triangle_test.html.
Yalewoo. Vector Cross Product and Applications: Determine Whether the Points Are Inside the Triangle. [2017-03-03]. http://www.yalewoo.com/in_triangle_t est. html.

DOI: 10.16182/j.issn1004731x.joss.17-CACIS007