无人机遂行编队飞行中的无源定位模型

摘要： 随着社会的发展，无人机已经逐渐走进了人们的生活当中，而对无人机的实时精准定位则成为了近期的热门话题。无人机定位大体上可分为两类：无源定位和有源定位，而本文主要基于无源定位技术提供一些创新性的算法和模型。

针对问题一（1），基于三个已知位置和编号的无人机信号的夹角信息，构建相交圆模型，由两圆位置关系以及三个方向夹角的制约关系，通过画图得出唯一交点（即所求解），利用此交点构造两个特征三角形，利用正弦定理和三角变换等工具求解得出接收信号无人机的精确位置，从而实现定位和调整的最终目的。

针对问题一（2），由于在问题一的基础上确实了某些发射信号无人机的编号信息，故从 4 架飞机发射信号开始讨论，只需证明在此情况下成立即可。采用 4 选 3 的方式发射三轮不同的信号组合，得出接收点关于三组圆心角以及三组方向角的两个连等方程式组，再通过消元和系数向量秩为 3 的进阶分析，得出接收点坐标唯一确定的结论，从而得出需要在增添 2 架无人机才能满足题设的答案。

针对问题一（3），由于直接从全局考虑难以实现，故采用贪婪策略（即贪心算法），每一次选取偏差最小的点与其他两个位置编号准确的点相组合，并假设位置准确，进行一轮统一的调整，并不断重复相同策略进行有限轮的调整，每一轮结束后计算对应轮次的总偏距离和（表征误差大小的指标），利用 Excel 软件绘制总偏距离和随轮次的变化曲线，得出进行一轮调整总体上为最佳方案（即优化率最高）。

针对问题二，基本原理同问题一，所谓“锥形”实际上可看成若干个“隐圆”的叠加（本题中为三个），利用问题一（3）中对于圆形编队姿态的贪心算法调整方案，可以同样实现精确定位的目标。

最后，本文还对所提出模型的灵敏度进行了详细的分析，利用偏导数的数值分析其对于不同数据变化的依赖程度，得出灵敏度较好的结论。另外，还对模型的优缺点进行了较为客观的分析，并给出了相对应的改进和推广方式。

关键词：无源定位 三角函数变换 贪婪策略 偏微分灵敏度测试

一、问题重述

1.1 问题背景

无人驾驶飞机简称“无人机”（“UAV”），是利用无线电遥控设备和自备的程序控制装置操纵的不载人飞行器。无人机实际上是无人驾驶飞行器的统称，从技术角度定义可以分为：无人固定翼飞机、无人垂直起降飞机、无人飞艇、无人直升机、无人多旋翼飞行器、无人伞翼机等。与载人飞机相比，它具有体积小、造价低、使用方便、对作战环境要求低、战场生存能力较强等优点。（见图 1）

图 1 无人机示意图

无人机集群在遂行编队飞行时，为避免外界干扰，应尽可能保持电磁静默。为保持编队队形，拟采用纯方位无源定位的方法调整无人机的位置。无源定位，又称被动定位，即由编队中某几架无人机发射信号、其余无人机被动接收信号，接收端提取方向信息定位发射端位置，从而调整自身的位置。编队中每架无人机均有固定编号，且在编队中与其他无人机的相对位置关系保持不变。接收信号的无人机所接收到的唯一方向信息为：该无人机与任意两架发射信号无人机连线之间的夹角。

1.2 问题重述

问题一：（1）该问题由圆心无人机和其余两个已知编号的无人机发射信号，其他无人机被动接收信号后根据仅有的已知条件：三个方向夹角的大小以及这三个角分别从属的无人机编号信息，确定自己现在的位置，然后调整自身微小的位置偏差，达成最终均匀分布的圆形编队姿态。由于三架主动发射信号的无人机位置已知且准确，故三者所成

夹角以及圆形编队的半径均为已知量，只需将每一个被动接收信号的无人机位置用已知的夹角，方向夹角和半径表示出来，则问题得解。

（2）承接第（1）问，两问最终需要达成的目的都是精准的定位和调整，不同的是该问仅知道圆心无人机和圆上一架飞机的编号和位置，那么要完成第一问的任务，必然需要更多的无人机参与发射，本题需要解出的就是最少需要参与的无人机个数（记为 k）。

（3）该问涉及更加宏观层面的方案设计。为方便解决问题，可设无人机每次调整的时间忽略不计，则该问需要在有限的步骤里达成误差尽量小的结果，下表（图 2）中的数据为一个特例，可以从具体事例中帮助提炼相对抽象的算法方案。

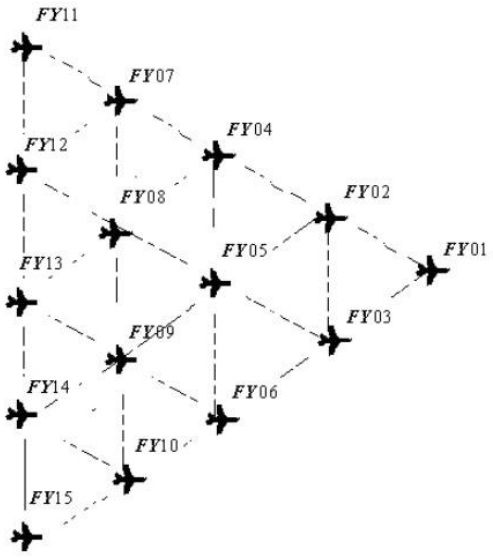

表 1　无人机的初始位置

无人机编号	极坐标 (m,°)
0	(0,0)
1	(100,0)
2	(98,40.10)
3	(112,80.21)
4	(105,119.75)
5	(98,159.86)
6	(112,199.96)
7	(105,240.07)
8	(98,280.17)
9	(112,320.28)

图 2 表 1 无人机的初始位置　　图 3 锥形无人机编队示意图

问题二：实际飞行中，无人机集群也可以是其他编队队形，例如锥形编队队形（直线上相邻两架无人机的间距相等，如 50 m，如图 3 所示）。需要解决的问题与第 1 题第 3 问类似，同样也需要设计具体的方案，但由于形状的不同，具体的实现细则会有所分野。

二、问题分析

2.1 问题一分析

无人机具自身感知的高度信息，可保持在同一个高度上飞行，系二维问题。理想几何图景为其中 9 架无人机（编号 FY01~FY09）按 40° 间隔均匀分布在某一圆周上，另 1 架无人机（编号为 FY00）位于圆心。

本问为一个纯几何问题。首先建立以 FY00 为原点 (0,0)，FY00 一 FY01 为 x 轴正方向的直角坐标。假设 FY00,FY01 发射信号，设另一发射信号的无人机为 FY0K（K=2,3,……,9）。若已知被动接收信号的无人机与 00、01 所成的夹角为α_1，与 00、0K 所成的夹角为α_2，与 01、0K 所成的夹角为α_3。

（1）2 个定点与 1 个动点形成的动点作为顶点的角，如果角度是一个定值，那么动点的轨迹是 1 个半径一定的圆弧，已知 3 个夹角为α_1、α_2、α_3，可根据上述理论做出 3 段圆弧，其共同的交点就是动点所在的准确位置。因为α_1、α_2、α_3之间存在两者之和等于第三者的关系，所以 3 段圆弧一定会交于同一点，所以我们只需做出 2 段圆弧并且求出它们的交点即可。2 个圆最多有 2 个交点，其中一个是 FY00（原点），另一个交点即为待定位无人机 A 的位置。

（2）第 2 问相当于前一题的推广，现在初始的 FY00，FY01 依然已知，但其余的发射信号的无人机（设为 FY0X、FY0Y）位置均未知。那我们只需要确定 FY0X 和 FY0Y 的位置即可将问题转化为第 1 问。此时 FY00，FY01，FY0X 与待定位无人机 A 所构成的 3 个夹角分别设为α_{11}、α_{12}、α_{13}；FY00，FY01，FY0Y 与待定位无人机 A 所构成的 3 个夹角分别设为α_{21}、α_{22}、α_{23}。由于编号未知，故此文所需要的无人机数量一定多于第 1 问，即再增添 1 架一定无法满足题设，故若能证明再增添 2 架（即上述的 FY0X、FY0Y）就能满足要求，则此问得解。

（3）本题为一个最优化方案设计问题，有两大评判“最优”的指标：步骤数和精确程度，由于直接制定一个普适的方案存在一定难度，故考虑从所给的特定数据入手，制定针对此特定问题的最优方案，从中提炼普遍规律和抽象算法，实现在有限的步骤内达到角度和误差尽可能小的精确结果。从全局谋划难以实现，故采用贪婪策略（贪心算法），即选择当前步骤下最准确的无人机参与发射，重复第 1 小问的实现方式，最终谋求误差的最小化。

2.2 问题二分析

本题不作为一个锥形模型来处理，应用问题一所研究的圆形模型，以某一架无人机为圆心，周围若干架无人机为圆周上的点进行如问题一第（3）题的拟合调整。

三、模型假设

为了适当的对模型进行合理简化，本文给出如下假设：

所有飞机都保持在同一高度（只考虑二维情况）。

主动发射信号的无人机可以同时接收信号。

不同无人机发射的信号种类不一样，因此接收信号的无人机可以对其加以区分。

所有需要调整的无人机位置均为稍有偏离，理解为没有很大的改变图形性质的偏离，例如不改变待定位无人机与发射信号无人机之间夹角（即 α_1 、 α_2 、 α_3 的锐角或钝角这一属性）

无人机每一次调整的时间可忽略不计，即设计方案时，步骤数只有有限和无限的区分，而主要矛盾，即决定性的判断标准是与理想位置的接近程度（角度和位置的偏差大小）。

四、符号说明

α_1	接收信号机与 FY00、FY01 所成的夹角
α_2	接收信号机与 FY00、FY0K 所成的夹角
α_3	接收信号机与 FY01、FY0K 所成的夹角
θ	FY00 与 FY01、FY0K 所成角，即 FY0K 在极坐标中的极角
$A(A1、A2、A3、A4)$	接收信号的无人机
γ	线段 FK00、A 与线段 FK00、FK01 的夹角，或线段 FK00、A 与线

	段 FK00、FK0K 的夹角
$S1$、$S2$、$S3$、$S4$、$S5$、$S6$	问题一第（1）问所划分的 6 个区域
l	FK00 与 A 之间线段的长度
R	标准半径（FK00 与 FK01 的距离）

五、模型的建立与求解

5.1 问题一第（1）问

5.1.1 问题一第（1）问模型的建立和求解

位于圆心的无人机（FY00）和编队中另 2 架无人机发射信号，其余位置略有偏差的无人机被动接收信号，且发射信号的无人机位置无偏差且编号已知。考虑圆心无人机和圆周上任一两架无人机的所有情况，得到如下模型信息：

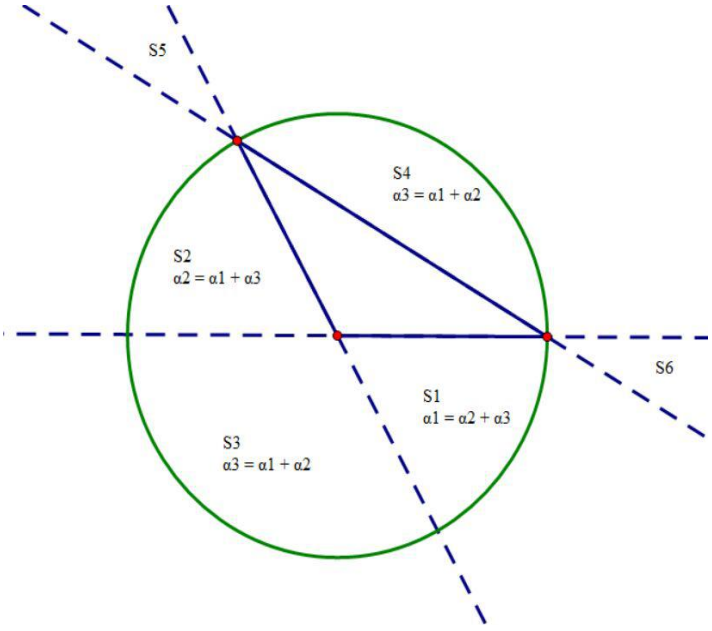

图 4 分类讨论区域的划分

三架无人机两两所在直线将平面区域分成六块。记作 S1，S2……S6。由无源定位能得知 3 个角度大小，那么根据关系式容易得到：

当待定位无人机处于 S1 区域时，其对应角度满足:

$$\alpha_1 = \alpha_2 + \alpha_3;$$

当待定位无人机处于 S2 区域时，其对应角度满足:

$$\alpha_2 = \alpha_1 + \alpha_3;$$

当待定位无人机处于 S3，S4 区域时，其对应角度满足:

$$\alpha_3 = \alpha_1 + \alpha_2;$$

而倘若找条件区分该两块区域：不难发现，当无人机位于 S3 区域圆周上时，其 $\alpha_3 < \frac{\pi}{2}$ (圆周角等于圆心角的一半)；当无人机位于 S4 区域圆周上时，其 $\alpha_3 > \frac{\pi}{2}$ (圆上内接四边形对角互补)。

待定位无人机 A 在虚线圆内时，α_3为钝角，在虚线圆外时，α_3为锐角。

当待定位无人机 A 位于 S3 区域时，由模型假设第 4 条，A 一定位于虚线圆外，否则偏离目标点过远。同理，当 A 位于 S4 区域时，A 一定位于虚线圆内，否则偏离过远不符合模型假设第 4 条。

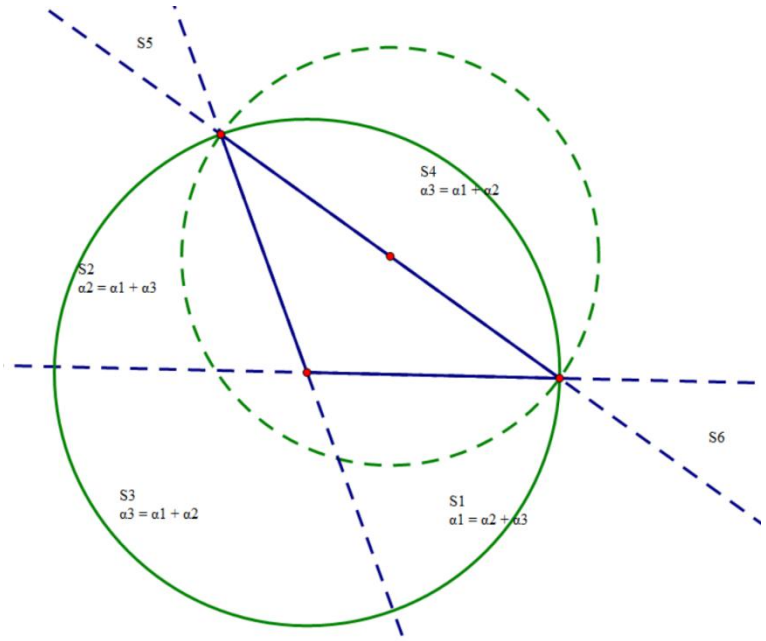

图 5 S3、S4 区域划分的判定

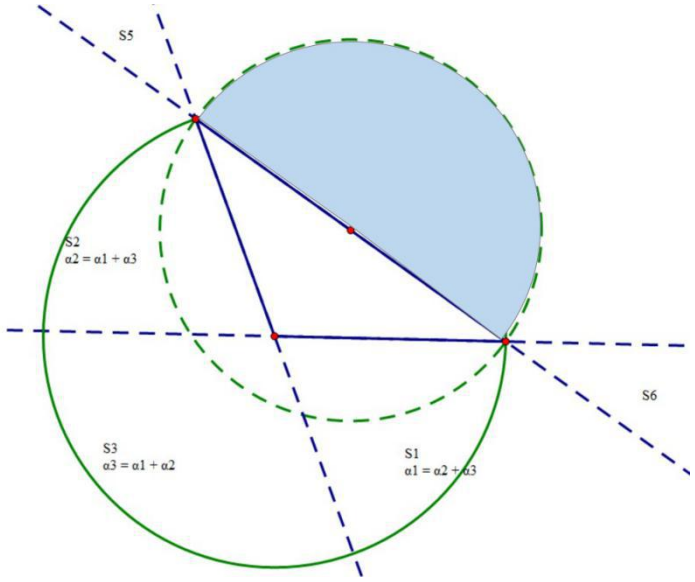

图 6 以 S4 区域为例，α 3 一定是钝角

而 S5、S6 全部在圆 FY00 外，并且该区域距离无人机理想位置最近的点分别是 FY01、FY0K，均为已知发射信号点，又由于无人机理想位置是非连续的分散的，因此距离其他待定位无人机距离过远，与模型假设第 4 条矛盾。因此 S5、S6 区域不会存在待定位无人机，故在以下计算中不予讨论。

设$k \in [\![2,9]\!]$：

case1 待定位无人机位于 S1 区域：

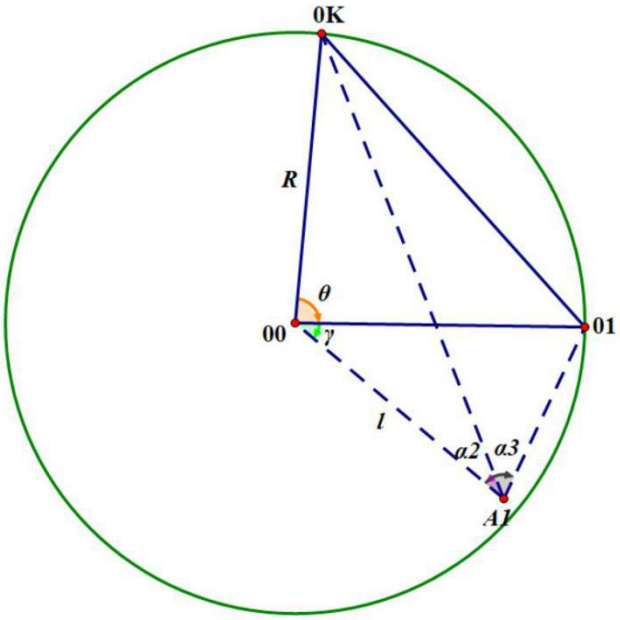

图 7 待定位无人机位于 S1 区域示意图

由正弦定理：

$$\begin{cases}\Delta 00,0K,A1:\frac{R}{sin\alpha_2}=\frac{l}{sin(\theta+\alpha_2+\gamma)}\\ \Delta 00,01,A1:\frac{R}{sin\alpha_1}=\frac{l}{sin(\alpha_1+\gamma)}\end{cases}$$

容易得到

$$\sin\alpha_2 sin(\gamma+\alpha_1)=sin\alpha_1 sin(\gamma+\theta+\alpha_2)$$

令$\beta=\theta+\alpha_2$

不难得

$$sin\alpha_2(sin\gamma cos\alpha_1+sin\alpha_1 cos\gamma)=sin\alpha_1(sin\gamma cos\beta+sin\beta cos\gamma)$$

整理得

$$sin\gamma(sin\alpha_2 cos\alpha_1-sin\alpha_1 cos\beta)=cos\gamma(sin\alpha_1 sin\beta-sin\alpha_1 sin\alpha_2)$$

设$A=sin\alpha_2 cos\alpha_1-sin\alpha_1 cos\beta, B=sin\alpha_1 sin\beta-sin\alpha_1 sin\alpha_2$

显然有

$$sin\gamma=\frac{B}{\sqrt{A^2+B^2}}, cos\gamma=\frac{A}{\sqrt{A^2+B^2}}$$

而

$$l=\frac{sin(\alpha_1+\gamma)}{sin\alpha_1}R=\frac{sin\alpha_1 cos\gamma+sin\gamma cos\alpha_1}{sin\alpha_1}R$$

得到

极坐标系（Polar Coordinates）下：

$A_1(l,-\gamma)$

$$\left(\frac{sin\beta cos\alpha_1-sin\alpha_1 cos\beta}{\sqrt{sin^2\alpha_1+sin^2\alpha_2-2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}R,\right.$$

$$\left.sin^{-1}\left(\frac{sin\alpha_1 sin\beta-sin\alpha_1 sin\alpha_2}{\sqrt{sin^2\alpha_1+sin^2\alpha_2-2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}\right)\right).$$

其中$\beta=\theta+\alpha_2$

而$\theta=min\left\{\pi,(k-1)\frac{\pi}{4},2\pi-(k-1)\frac{\pi}{4}\right\}$

平面直角坐标系（Cartesian Coordinates）下：

$A_1(lcos\gamma,-lsin\gamma)$

$$= A_1(\frac{sin\alpha_1 cos\gamma sin\alpha_2 cos\alpha_1 + sin\gamma sin\alpha_2 cos^2\alpha_1 - sin^2\alpha_1 cos\beta cos\gamma - sin\alpha_1 sin\gamma cos\alpha_1 cos\beta}{sin\alpha_1\sqrt{sin^2\alpha_1 + sin^2\alpha_2 - 2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta - 2sin^2\alpha_1 sin\alpha_2 sin\beta}}R,$$

$$\frac{sin^2\alpha_1 cos\gamma sin\alpha_2 + sin\gamma sin\alpha_1 sin\alpha_2 cos\alpha_1 - sin^2\alpha_1 sin\beta cos\gamma - sin\alpha_1 sin\gamma sin\beta cos\alpha_1}{sin\alpha_1\sqrt{sin^2\alpha_1 + sin^2\alpha_2 - 2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta - 2sin^2\alpha_1 sin\alpha_2 sin\beta}}R).$$

case 2 待定位无人机位于 S2 区域：

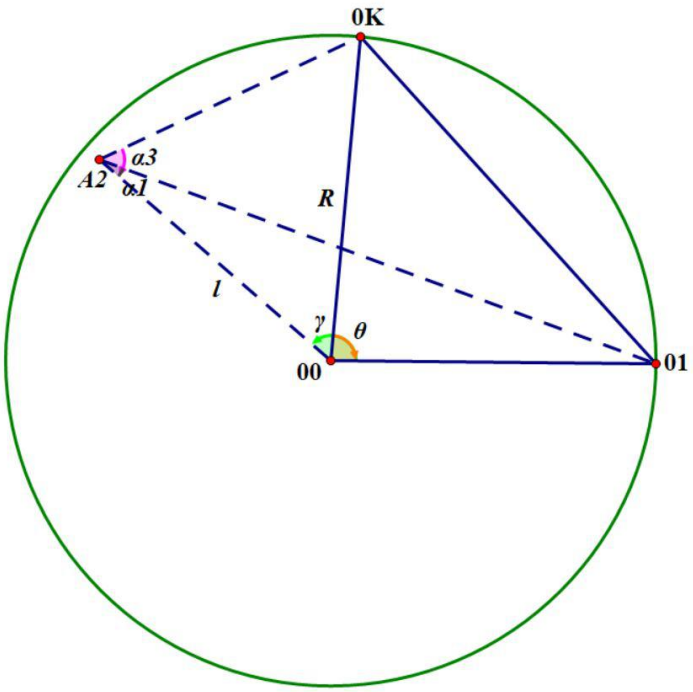

图 8 待定位无人机位于 S2 区域示意图

$$\begin{cases} \Delta 00, 0K, A2 : \frac{R}{sin\alpha_2} = \frac{l}{sin(\alpha_2+\gamma)} \\ \Delta 00, 01, A2 : \frac{R}{sin\alpha_1} = \frac{l}{sin(\theta+\alpha_1+\gamma)} \end{cases}$$

容易得到

$$\sin\alpha_2 sin(\theta + \gamma + \alpha_1) = sin\alpha_1 sin(\gamma + \alpha_2)$$

令$\beta = \theta + \alpha_1$,

不难得

$$sin\alpha_1(sin\gamma cos\alpha_2 + sin\alpha_2 cos\gamma) = sin\alpha_2(sin\gamma cos\beta + sin\beta cos\gamma)$$

整理得

$$sin\gamma(sin\alpha_2 cos\alpha_1 - sin\alpha_1 cos\beta) = cos\gamma(sin\alpha_1 sin\beta - sin\alpha_1 sin\alpha_2)$$

设$A = sin\alpha_1 cos\alpha_2 - sin\alpha_2 cos\beta, B = sin\alpha_2 sin\beta - sin\alpha_1 sin\alpha_2$

显然有

$$sin\gamma = \frac{B}{\sqrt{A^2+B^2}}, cos\gamma = \frac{A}{\sqrt{A^2+B^2}}$$

而

$$l = \frac{sin(\alpha_2+\gamma)}{sin\alpha_2}R = \frac{sin\alpha_2 cos\gamma + sin\gamma cos\alpha_2}{sin\alpha_2}R$$

得到极坐标系（Polar Coordinates）下：

$$A_2(l, \theta+\gamma)$$

$$= (\frac{sin\alpha_2 sin\beta cos\alpha_2 - sin^2\alpha_2 cos\beta}{sin\alpha_1\sqrt{sin^2\alpha_1 + sin^2\alpha_2 - 2sin\alpha_1 sin\alpha_2 cos\alpha_2 cos\beta - 2sin^2\alpha_2 sin\alpha_1 sin\beta}}R,$$

$$\theta + sin^{-1}(\frac{sin\alpha_2 sin\beta - sin\alpha_1 sin\alpha_2}{\sqrt{sin^2\alpha_1 + sin^2\alpha_2 - 2sin\alpha_1 sin\alpha_2 cos\alpha_2 cos\beta - 2sin^2\alpha_2 sin\alpha_1 sin\beta}}).$$

其中$\beta = \theta + \alpha_1$

而$\theta = min\left\{\pi, (k-1)\frac{\pi}{4}, 2\pi - (k-1)\frac{\pi}{4}\right\}$

平面直角坐标系（Cartesian Coordinates）下：

得到$A_2(lcos(\theta+\gamma), lsin(\theta+\gamma))$

$$= A_2(\frac{sin\alpha_1 cos\gamma sin\alpha_2 cos\alpha_1 + sin\gamma sin\alpha_2 cos^2\alpha_1 - sin^2\alpha_1 cos\beta cos\gamma - sin\alpha_1 sin\gamma cos\alpha_1 cos\beta}{sin\alpha_1\sqrt{sin^2\alpha_1 + sin^2\alpha_2 - 2sin\alpha_1 sin\alpha_2 cos\alpha_2 cos\beta - 2sin^2\alpha_2 sin\alpha_1 sin\beta}}R,$$

$$\frac{sin^2\alpha_1 cos\gamma sin\alpha_2 + sin\gamma sin\alpha_1 sin\alpha_2 cos\alpha_1 - sin^2\alpha_1 sin\beta cos\gamma - sin\alpha_1 sin\gamma sin\beta cos\alpha_1}{sin\alpha_1\sqrt{sin^2\alpha_1 + sin^2\alpha_2 - 2sin\alpha_1 sin\alpha_2 cos\alpha_2 cos\beta - 2sin^2\alpha_2 sin\alpha_1 sin\beta}}R).$$

case 3 待定位无人机位于 S3 区域

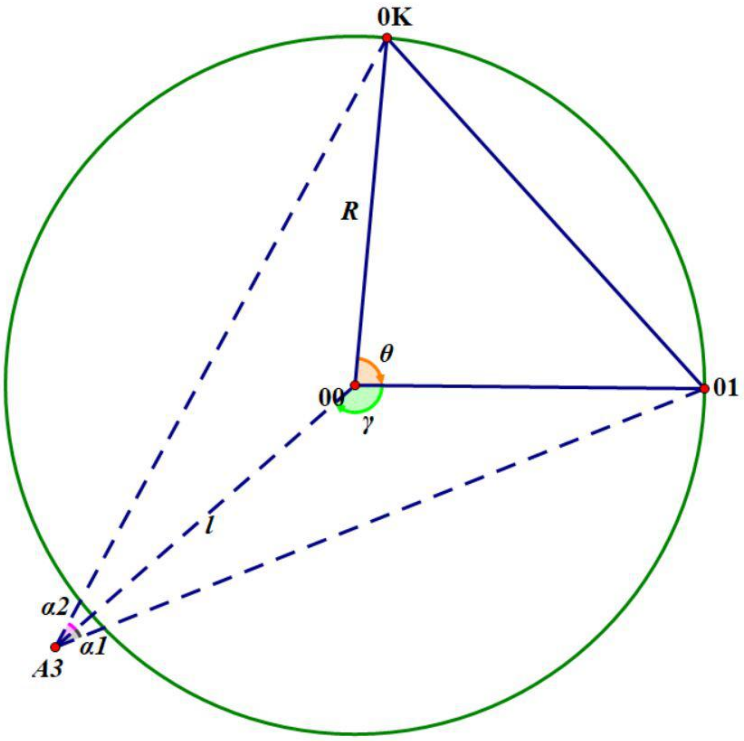

图 9 待定位无人机位于 S3 区域示意图

$$\begin{cases} \Delta 00,0K,A3: \frac{R}{sin\alpha_2} = \frac{l}{sin(-\theta+\alpha_2-\gamma)} \\ \Delta 00,01,A3: \frac{R}{sin\alpha_1} = \frac{l}{sin(\alpha_1+\gamma)} \end{cases}$$

容易得到

$$\sin\alpha_2 sin(\gamma+\alpha_1) = sin\alpha_1 sin(-\gamma-\theta+\alpha_2)$$

令$\beta = -\theta+\alpha_2$

不难得$sin\alpha_2(sin\gamma cos\alpha_1 + sin\alpha_1 cos\gamma) = sin\alpha_1(-sin\gamma cos\beta + sin\beta cos\gamma)$

整理得$sin\gamma(sin\alpha_2 cos\alpha_1 + sin\alpha_1 cos\beta) = cos\gamma(sin\alpha_1 sin\beta - sin\alpha_1 sin\alpha_2)$

设$A = sin\alpha_2 cos\alpha_1 + sin\alpha_1 cos\beta, B = sin\alpha_1 sin\beta - sin\alpha_1 sin\alpha_2$

显然有

$$sin\gamma = \frac{B}{\sqrt{A^2+B^2}}, cos\gamma = \frac{A}{\sqrt{A^2+B^2}}$$

而 $$l = \frac{sin(\alpha_1+\gamma)}{sin\alpha_1}R = \frac{sin\alpha_1 cos\gamma + sin\gamma cos\alpha_1}{sin\alpha_1}R$$

得到

极坐标系（Polar Coordinates）下：

$$A_3(l,-\gamma)$$

$$= A_3(\frac{sin\alpha_1 cos\beta + cos\alpha_1 sin\beta}{\sqrt{sin^2\alpha_1 + sin^2\alpha_2 + 2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta - 2sin^2\alpha_1 sin\beta \sin\alpha_2}}R,$$

$$sin^{-1}(\frac{sin\alpha_1 sin\beta - sin\alpha_1 sin\alpha_2}{\sqrt{sin^2\alpha_1 + sin^2\alpha_2 + 2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta - 2sin^2\alpha_1 sin\beta \sin\alpha_2}}))$$

平面直角坐标系（Cartesian Coordinates）下：

得到

$$A_3(lcos\gamma, -lsin\gamma)$$

$$= A_3(\frac{sin\alpha_1 cos\beta \cos\gamma + cos\alpha_1 sin\beta \cos\gamma}{\sqrt{sin^2\alpha_1 + sin^2\alpha_2 + 2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta - 2sin^2\alpha_1 sin\beta \sin\alpha_2}}R,$$

$$-\frac{sin\alpha_1 cos\beta \sin\gamma + cos\alpha_1 sin\beta \sin\gamma}{\sqrt{sin^2\alpha_1 + sin^2\alpha_2 + 2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta - 2sin^2\alpha_1 sin\beta \sin\alpha_2}}R)$$

case 4 待定位无人机位于 S4 区域

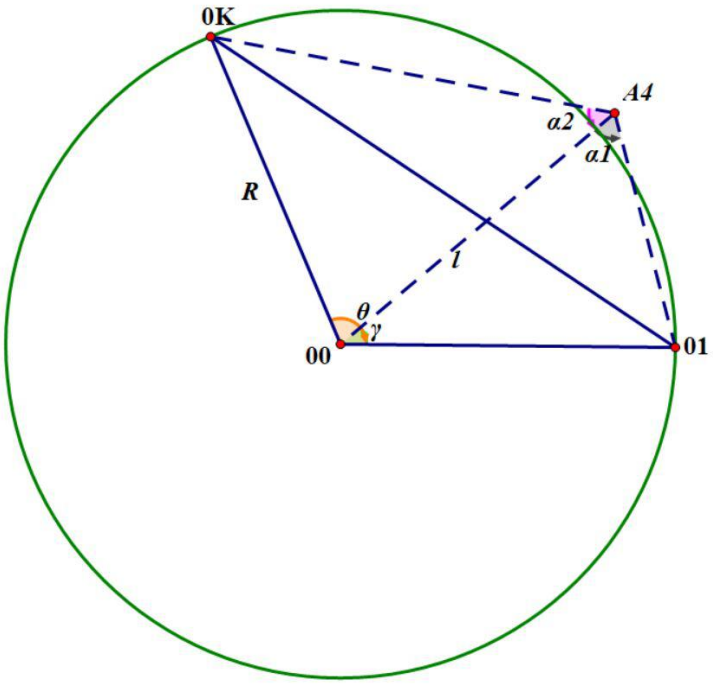

图 10 待定位无人机位于 S4 区域示意图

$$\begin{cases}\Delta 00,0K,A4:\frac{R}{sin\alpha_2}=\frac{l}{sin(\theta+\alpha_2-\gamma)}\\ \Delta 00,01,A4:\frac{R}{sin\alpha_1}=\frac{l}{sin(\alpha_1+\gamma)}\end{cases}$$

容易得到

$$\sin\alpha_2 sin(\gamma+\alpha_1)=sin\alpha_1 sin(-\gamma+\theta+\alpha_2)$$

令$\beta=\theta+\alpha_2$

不难得$sin\alpha_2(sin\gamma cos\alpha_1+sin\alpha_1 cos\gamma)=sin\alpha_1(sin\gamma cos\beta-sin\beta cos\gamma)$

整理得$sin\gamma(sin\alpha_2 cos\alpha_1-sin\alpha_1 cos\beta)=cos\gamma(-sin\alpha_1 sin\beta-sin\alpha_1 sin\alpha_2)$

设$A=sin\alpha_2 cos\alpha_1+sin\alpha_1 cos\beta, B=sin\alpha_1 sin\beta-sin\alpha_1 sin\alpha_2$

显然有

$$sin\gamma=\frac{B}{\sqrt{A^2+B^2}}, cos\gamma=\frac{A}{\sqrt{A^2+B^2}}$$

而$$l=\frac{sin(\alpha_1+\gamma)}{sin\alpha_1}R=\frac{sin\alpha_1 cos\gamma+sin\gamma cos\alpha_1}{sin\alpha_1}R$$

极坐标系（Polar Coordinates）下：

得到

$$A_4(l,\gamma)$$

$$=A_4(\frac{sin\alpha_1 cos\beta+cos\alpha_1 sin\beta}{\sqrt{sin^2\alpha_1+sin^2\alpha_2+2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}R,$$

$$sin^{-1}(\frac{sin\alpha_1 sin\beta-sin\alpha_1 sin\alpha_2}{\sqrt{sin^2\alpha_1+sin^2\alpha_2+2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}))$$

平面直角坐标系（Cartesian Coordinates）下：

得到

$$A_4(lcos\gamma,lsin\gamma)$$

$$=A_4(\frac{sin\alpha_1 cos\beta\cos\gamma+cos\alpha_1 sin\beta\cos\gamma}{\sqrt{sin^2\alpha_1+sin^2\alpha_2+2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}R,$$

$$\frac{sin\alpha_1 cos\beta\sin\gamma+cos\alpha_1 sin\beta\sin\gamma}{\sqrt{sin^2\alpha_1+sin^2\alpha_2+2sin\alpha_2 cos\alpha_1 sin\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}R)$$

5.1.2 问题一第（2）问

Step1：3 架无人机无法完成定位【圆心处 1 架（FY00），圆周上 2 架（FY01、FY0K）】

由于无人机 FY0K 编号信息缺失，无法得知其位置，即第 1 问中的 θ 角在本题中为未知变量，因此根据第 1 问的方程无法解出唯一解，因此 3 架无人机无法完成定位。

Step2：4 架无人机可以完成定位【圆心处 1 架（FY00），圆周上 3 架（FY01、FY0X、FY0Y）】，如下图

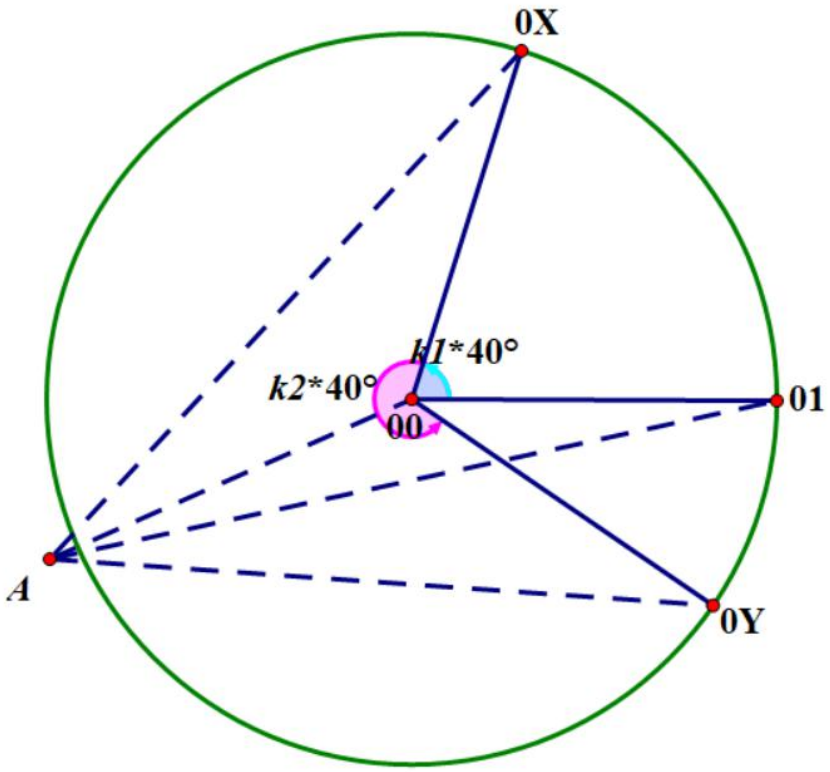

图 11 问题一第 2 问 4 架无人机示意图

设

弧$01,\widehat{00,}0X$对应的角度为$k_1\frac{2\pi}{9}$

优弧$0X,\widehat{00,}0XY$对应的角度为$k_2\frac{2\pi}{9}$

现分三轮进行发射、接收信号的操作：

第 1 轮：FY00、FY01、FY0X 进行发射，A 进行接收

假设 A 的接收角分别为α_{11}、α_{12}、α_{13}

可以根据第 1 问的结果得出，A 点的坐标为$(x_1(k_1,k_2),y_1(k_1,k_2))$

其中，函数x_1、y_1有α_{11}、α_{12}、α_{13}作为常量参数

第 2 轮：FY00、FY01、FY0Y 进行发射，A 进行接收

假设 A 的接收角分别为α_{21}、α_{22}、α_{23}

可以根据第 1 问的结果得出，A 点的坐标为$(x_2(k_1,k_2),y_2(k_1,k_2))$

其中，函数x_2、y_2有α_{21}、α_{22}、α_{23}作为常量参数

第 3 轮：FY00、FY0X、FY0Y 进行发射，A 进行接收

假设 A 的接收角分别为α_{31}、α_{32}、α_{33}

可以根据第 1 问的结果得出，A 点的坐标为$(x_3(k_1,k_2),y_3(k_1,k_2))$

其中，函数x_3、y_3有α_{31}、α_{32}、α_{33}作为常量参数

由 A 点坐标存在且唯一确定，可以得到：

$$\begin{cases} x_1(k_1,k_2)=x_2(k_1,k_2)=x_3(k_1,k_2) & ① \\ y_1(k_1,k_2)=y_2(k_1,k_2)=y_3(k_1,k_2) & ② \end{cases}$$

由图 10 可知，9 个已知量α_{**}实际上只有 3 个是有效已知量，

即向量$(\alpha_{11},\alpha_{12},\alpha_{13},\alpha_{21},\alpha_{22},\alpha_{23},\alpha_{31},\alpha_{32},\alpha_{33})$的秩为 3，

即$rank(\alpha_{11},\alpha_{12},\alpha_{13},\alpha_{21},\alpha_{22},\alpha_{23},\alpha_{31},\alpha_{32},\alpha_{33})=3$

由①②可以唯一解出k_1,k_2，即确定了 FY0X 和 FY0Y 的编号进而确定了坐标。进而应用问题一第（1）问的结论即可定位待定位无人机 A。

因此，4 架无人机可以完成待定位无人机的定位。

5.1.3 问题一第（3）问

采取贪心的调整办法，记除去 FY00、FY01 后，当前与目标点距离最小（即目前位置最精确）的无人机为 FY0K，每次选取 FY00、FY01 和 FY0K 为发射信号无人机，其余为接收信号无人机，并且在接受到信号进行计算和定位时按 FY00、FY01 和 FY0K 的位置均在理想位置上进行处理。

其余接受信号的无人机根据自己接收到的信号对自己进行定位，并与理想位置进行比较，计算出需要调整的方向与距离并进行调整。以上过程称为 1 轮调整。

程序流程大致如下：

Step1. 计算 FY02~FY09 与理想位置的距离，选出距离差最小的无人机编号记为 FY0K

Step2. 以 FY00、FY01、FY0K 为基准，发射信号，其余无人机接收信号，并计算假设 FY00、FY01、FY0K 均在准确位置上时，自己的所在位置$(\hat{x}_i, \hat{y}_i)$

Step3. 计算当前定位与准确位置的差向量，即$(\hat{x}_i, \hat{y}_i) + (\Delta x_i, \Delta y_i) = (x_i, y_i)$

Step4. 对当前实际位置$(\widetilde{x_i}, \widetilde{y_i})$进行调整得到新位置$(\widetilde{x_i}, \widetilde{y_i}) + (\Delta x_i, \Delta y_i)$，重新计算偏差距离

Step5. 对剩余 7 架圆周上的无人机均进行上述调整之后，1 轮调整结束，计算总偏差距离，与调整前进行比较

Step6. 循环进行下一轮调整（即返回 Step1）

程序用 Java 13 实现，代码见附录。运行 100 轮，取比较有代表性的数据（如图 12）

调整轮数	0	1	2	3	4
总偏距离和/m	52.0826	18.3268	17.2259	16.0574	15.5872
调整轮数	5	10	20	100	
总偏距离和/m	15.1687	13.7694	12.5149	11.0218	

图 12 调整轮数与总偏距离和数据

以调整轮数为 x 轴，总偏差为 y 轴作图：

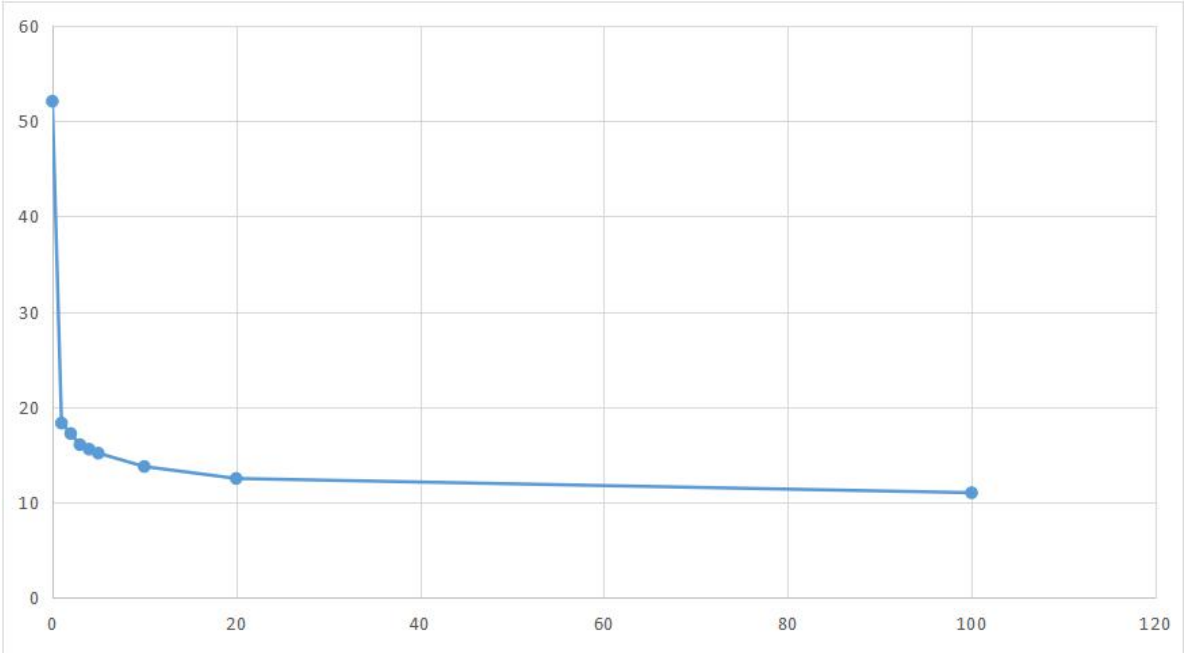

图 13 调整轮数与总偏距离和图线

从数据以及图线中可以看出，第一轮调整有比较明显的效果，但后续调整效果不佳。考虑到复杂度以及调整成本，根据以上描述只进行一轮调整为最佳方案，将总偏距离和从 52.0826m 优化到 18.3268m，优化率为 64.8%

5.2 问题二

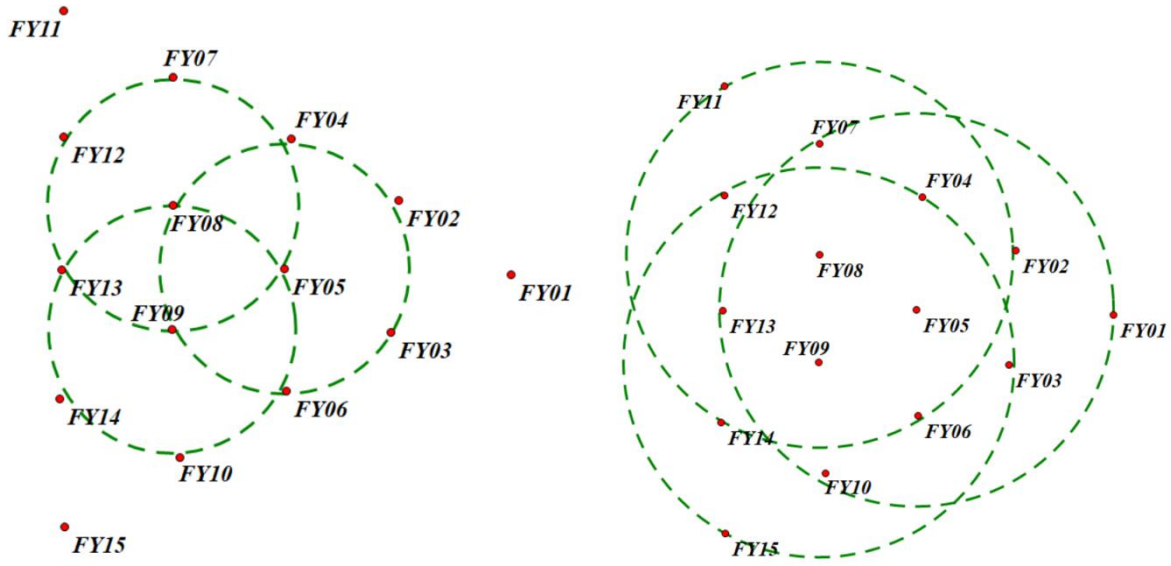

图 14 锥形编队校准示意图

以 15 架锥形编队的无人机为例，选取 FY05、FY08、FY09 为圆心，50m 为半径，对周围的 6 架无人机进行如问题一第（3）问的校准。对于角落上的无人机，同样选取取 FY05、FY08、FY09 为圆心，86.6m 为半径进行如问题一第（3）问的校准。

此模型适用范围较广，当无人机编队多达10^2、10^3甚至更大数量级的时候都具有可操作性和实用性。

六、定位模型的灵敏度分析

由于定位问题并未用到任何近似故只考虑不确定度的传递、实际操作过程中的反三角等运算对精度的影响以及区域确定即 5.1.1.a 涉及到的问题，来作该模型下的灵敏度分析：

6.1 不确定度及精度损失

设测量位置坐标泛函为：

$$\omega = f(\alpha_1, \alpha_2, \theta, R)$$

考虑$\alpha_1 = \alpha_1^* + \delta_{\alpha_1}, \alpha_2 = \alpha_2^* + \delta_{\alpha_2}, \theta = \theta^* + \delta_\theta, R = R^* + \delta_R$

其中$i = i^* + \delta_i, i \in \{\alpha_1, \alpha_2, \theta, R\}$表示真值，而$i^* \in \{\alpha_1^*, \alpha_2^*, \theta^*, R^*\}$表示计算出的数值或是无人机反馈出来的数据。

那么不确定度为：

$$\sigma_\omega = \sqrt{(\frac{\partial f}{\partial \alpha_1})^2 \delta_{\alpha_1}^2 + (\frac{\partial f}{\partial \alpha_2})^2 \delta_{\alpha_2}^2 + (\frac{\partial f}{\partial \theta})^2 \delta_\theta^2 + (\frac{\partial f}{\partial R})^2 \delta_R^2}$$

考虑题中表一数据，$\omega = \omega^* \pm \delta_\omega$,

而应当关注的是$|\frac{\delta_\omega}{\omega} \times 100\%|$,通过演算可以知道，该式值为 0.9%。

若使用反三角函数（如sin^{-1}）且取弧度制保留小数点一位时，那么根据

$$l \propto cos\gamma, ork_1 cos\gamma + k_2 sin\gamma,\ \theta \propto sin^{-1}(k_3)$$

可估计出，在此模型下，精度丢失对于极坐标结果影响在 5‰以内。

6.2 “略微偏差”

见图五，仅考虑 S4 情形，且认为此偏差的角度在$\pm 20^\circ$之内。当无人机位于虚线圆上时

（临界状态），距离换言之$\alpha_1 \to \frac{\pi}{2}$，圆心无人机位置最近时，即在偏差刚好为$20^\circ$，通过解$\Delta 00AM$得到：

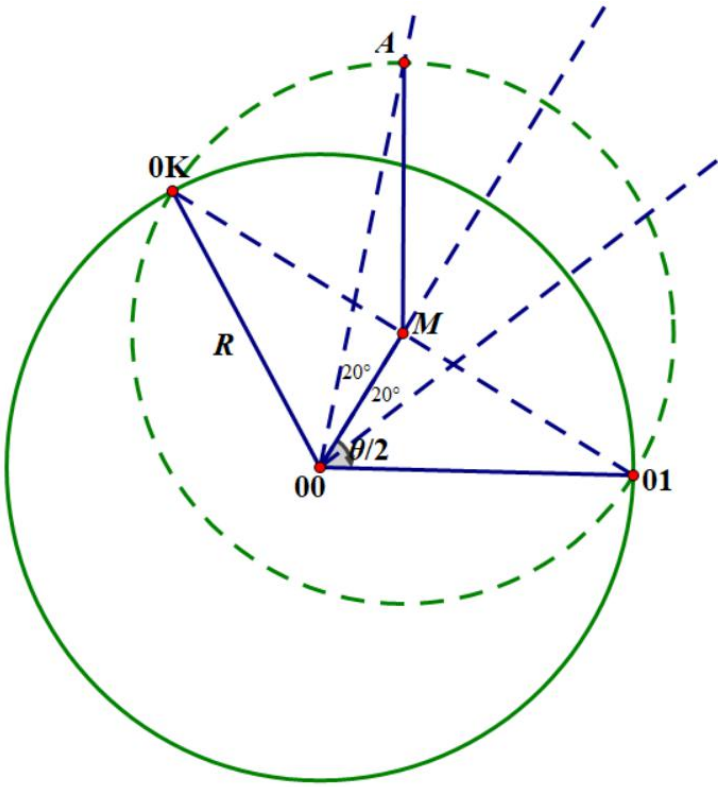

图 15

$$\frac{Rsin\frac{\theta}{2}}{sin\frac{\pi}{9}} = \frac{Rcos\frac{\theta}{2}}{sin\widehat{MA00}} = \frac{|A00|}{sin(\frac{\pi}{9} + \widehat{MA00})}$$

即 $$|A00| = R\sqrt{1 - sin\theta cos(\frac{8\pi}{9} - sin^{-1}(\frac{\frac{\pi}{9}}{tan\frac{\theta}{2}}))}$$

那么对于 $$\frac{|A00|}{R} = \sqrt{1 - sin\theta cos(\frac{8\pi}{9} - sin^{-1}(\frac{\frac{\pi}{9}}{tan\frac{\theta}{2}}))}$$，使用作图工具得到：

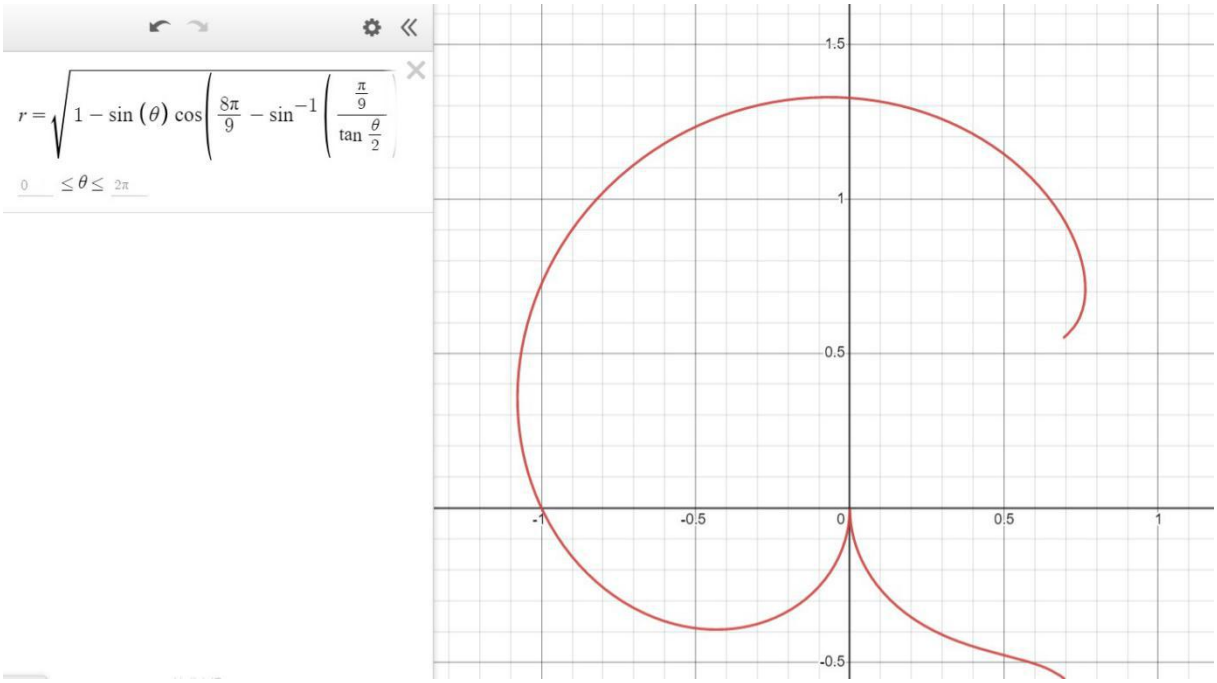

图 16

在极坐标系下理解，可知该比值在$\theta \in [\frac{2\pi}{9}, \frac{8\pi}{9}]$情况下在 1.3 附近浮动，而这明显是指该距离与目标半径可比，不属微小偏差假设。

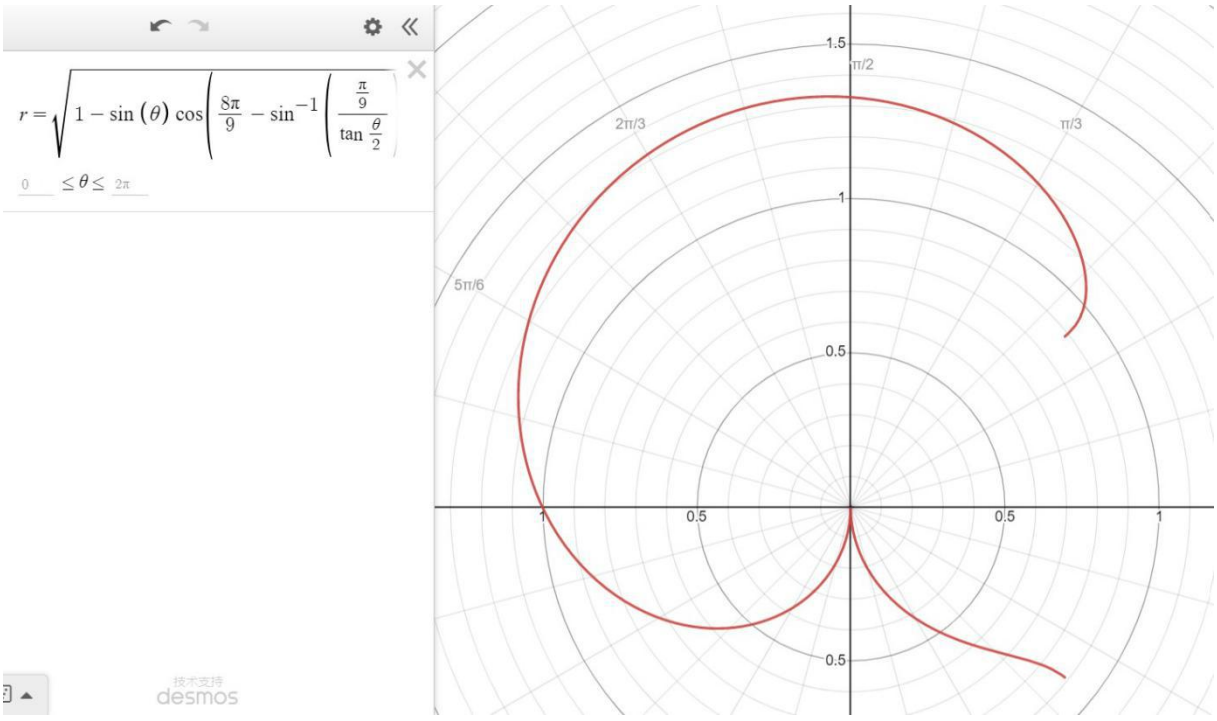

图 17

6.3 灵敏度

综合之前讨论，我们直接对该模型做灵敏度分析：（以 case1 为例）

$$\frac{\partial r}{\partial R}=\frac{sin\beta cos\alpha_1-sin\alpha_1 cos\beta}{\sqrt{sin^2\alpha_1+sin^2\alpha_2-2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}}independentofR$$

结果与 R 无关，从距离的观点来看，此模型非常可靠。但是β的不确定性更大，即

$$\frac{\partial r}{\partial \beta}=\frac{cos(\alpha_1-\beta)(sin^2\alpha_1+sin^2\alpha_2-2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta)-(sin\beta cos\alpha_1-sin\alpha_1 cos\beta)(sin\alpha_1 sin\alpha_2 cos\alpha_1 sin\beta)-sin^2\alpha_1 sin\alpha_2 cos\beta}{sin^2\alpha_1+sin^2\alpha_2-2sin\alpha_1 sin\alpha_2 cos\alpha_1 cos\beta-2sin^2\alpha_1 sin\alpha_2 sin\beta}R$$

利用 Fourier expansion 将其展开，并略去二阶及以上项，得到

$$\frac{\partial r}{\partial \beta}(\beta)=\frac{\beta(cos\alpha_1-2sin\alpha_1)-0.5\sqrt{sin^2\alpha_1+sin^2\alpha_1-4sin\alpha_1 sin\alpha_2 cos\alpha_1\cdot\beta-2sin^2\alpha_1 sin\alpha_2\cdot\beta}}{sin^2\alpha_1+sin^2\alpha_1-4sin\alpha_1 sin\alpha_2 cos\alpha_1\cdot\beta-2sin^2\alpha_1 sin\alpha_2\cdot\beta}$$

绘制出：

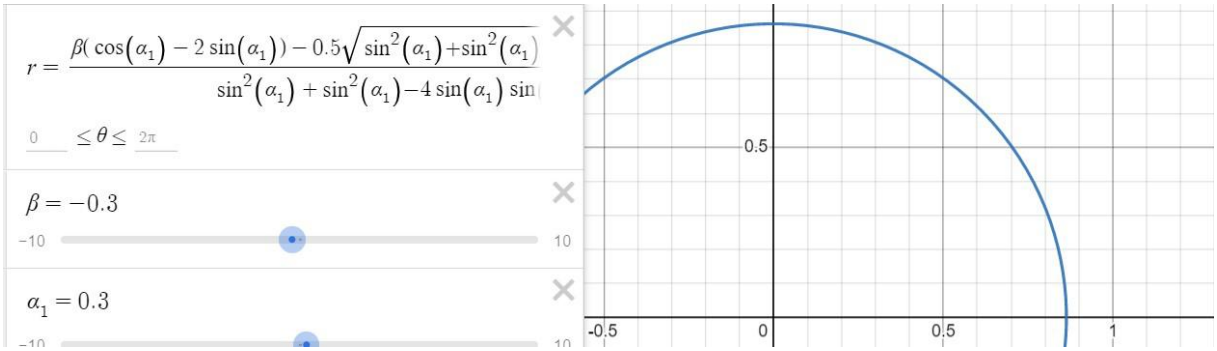

图 18

分析该图，可以认为该模型对于角度β而言，灵敏度较好。

七、模型的评价

7.1 模型的优点

主要使用三角函数知识，如正余弦定理，变量代换进行求解，模型复杂程度低，无需复杂编程算法的介入。

由于均为理论公式推导，所以在已知量准确的前提下，模型中其他相关变量的理论计算值均不存在误差，唯一误差仅会出现在使用计算机实践时精度的缺失，模型准确度高。

秉持分类讨论的基本原则，充分考虑各个方向各个位置的无人机情况，模型普适性强。

宏观方案设计中采取贪心算法，无需考虑多步复杂推理且最终结果与理想情况适配度极高，在最大程度上兼顾了算法简洁度和模型准确度。

7.2 模型的缺点

单纯采用解三角形的方式，在变量增加且形式复杂时计算难度扩增，且三角函数计算在常规的计算机算法中难以获得明显简化，故计算速度有待提高。

主要考虑二维情况，而三维的空间坐标系难以进行同等逻辑体系的推导，推广性相对较弱。

试用贪心算法，难以从整体上确保准确度最优，故模型存在优化空间。

7.3 模型的推广

任何以圆为根本图形的二维形态编队都可以以本文所述模型求解，如正多边形，对角互补四边形编队等。

贪婪策略的操作方式，即每次选取当前最准确的一架无人机进行后续定位校准的方法可以推广至所有情况。

八、参考文献

[1]屈小媚,刘韬,谈文蓉.基于多无人机协作的多目标无源定位算法[J].中国科学:信息科学,2019,49(05):570-584.

[2]余翊森. 利用多径信息的无人机无源定位与跟踪算法研究[D].电子科技大学,2017.

[3]高擎峰. 多无人机被动目标定位与跟踪技术研究[D].南京理工大学,2017.

[4]蔡伟,张晓峰.一种无人机群协同无源定位资源优化调度方法[J].舰船电子对抗,2018,41(03):54-58.DOI:10.16426/j.cnki.jcdzdk.2018.03.013.

[5]谭畅. 无人机无源测向方法与实现技术研究[D].电子科技大学,2019.

九、附录

附录 1

目录

附录 2 Main.java

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<FY> list = new ArrayList<>();
        double[] target = {0, 0, 40, 80, 120, 160, 200, 240, 280, 320};

        list.add(new FY());
        for (int i = 1; i < 10; i++) {
            double len = sc.nextDouble();
            double ang = sc.nextDouble();
            list.add(new FY(i, len, ang, Util.distance(len, ang, 100, target[i])));
            //System.out.println("id " + list.get(i).getId() + " " + list.get(i).getLength() +
" " + list.get(i).getAngle());
        }

        int cnt = 0; // 轮数
        while (cnt <= 100){

            double distAll = 0;
            for (int i = 1; i < 10; i++) {
                distAll += list.get(i).getDist();
            }
            System.out.println("cnt: " + cnt + "    dist: " + distAll);

            cnt++;
            Collections.sort(list, new Comparator<FY>(){
                @Override
                public int compare(FY o1, FY o2) {
                    if ((o1.getDist() - o2.getDist() > 0 ? 1 : o1.getDist() == o2.getDist() ?
```

```
0 : -1) != 0){
                        return (o1.getDist() - o2.getDist() > 0 ? 1 : o1.getDist() ==
o2.getDist() ? 0 : -1);
                    } else {
                        return (o1.getId() - o2.getId() > 0 ? 1 : -1);
                    }
                }
            });
//              for (int i = 0; i < 10; i++) {
//              System.out.println("id:" + list.get(i).getId() + " len:" + list.get(i).getLength()
+
//                      " ang:" + list.get(i).getAngle() + " dist:" + list.get(i).getDist());
//              }
//              System.out.println("----------------------");

            for (int i = 3; i < 10; i++) {
                double[] lenAng = Util.calculate(list, i);

                list.get(i).setLength(list.get(i).getLength() + 100 - lenAng[0]);
                list.get(i).setAngle(list.get(i).getAngle() + target[list.get(i).getId()] -
lenAng[1]);
                list.get(i).setDist(Util.distance(list.get(i).getLength(),
list.get(i).getAngle(), 100, target[list.get(i).getId()]));
            }
//              for (int i = 0; i < 10; i++) {
//              System.out.println("id:" + list.get(i).getId() + " len:" + list.get(i).getLength()
+
//                      " ang:" + list.get(i).getAngle() + " dist:" + list.get(i).getDist());
//              }

        }

    }
}
```

附录 3 Util.java

```
import java.util.ArrayList;

public class Util {

    static double distance(double l1, double a1, double l2, double a2){
        double x = Math.cos(a1 * Math.PI / 180) * l1 - Math.cos(a2 * Math.PI / 180) * l2;
        double y = Math.sin(a1 * Math.PI / 180) * l1 - Math.sin(a2 * Math.PI / 180) * l2;
```

```
        return Math.sqrt(x * x + y * y);
    }


    static double[] calculate(ArrayList<FY> list, int i){
        FY FY00 = list.get(0);
        FY FY01 = list.get(1);
        FY FY0X = list.get(2);
        FY A = list.get(i);


        return threeFY(FY00, FY01, FY0X, A);
    }


    private static double[] threeFY(FY FY00, FY FY01, FY FY0K, FY A){
        double Ax = A.getLength() * Math.cos(A.getAngle() * Math.PI / 180);
        double Ay = A.getLength() * Math.sin(A.getAngle() * Math.PI / 180);
//        System.out.println("Ax " + Ax);
//        System.out.println("Ay " + Ay);
        if (
                (FY0K.getId() == 2 || FY0K.getId() == 3) && (Ay <= 0 && Ay <= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180)) ||
                (FY0K.getId() == 4 || FY0K.getId() == 5) && (Ay <= 0 && Ay >= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180)) ||
                (FY0K.getId() == 6 || FY0K.getId() == 7) && (Ay >= 0 && Ay <= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180)) ||
                (FY0K.getId() == 8 || FY0K.getId() == 9) && (Ay >= 0 && Ay >= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180))
        ){ // S1
            double alpha1 = inAng(FY00, FY01, A);
            double alpha2 = inAng(FY00, FY0K, A);
            double a = Math.sin(alpha2 * Math.PI / 180) * Math.cos(alpha1 * Math.PI / 180) -
                    Math.sin(alpha1 * Math.PI / 180) * Math.cos((alpha2 + 40*(FY0K.getId()-1))
* Math.PI / 180);
            double b = Math.sin(alpha1 * Math.PI / 180) * Math.sin((alpha2 + 40*(FY0K.getId()-1))
* Math.PI / 180) -
                    Math.sin(alpha1 * Math.PI / 180) * Math.sin(alpha2 * Math.PI / 180);
            double[] lenAng = new double[2];
            lenAng[0] = 100 * (a / Math.sqrt(a*a + b*b) + b / (Math.sqrt(a*a + b*b) *
Math.tan(alpha1 * Math.PI / 180)));
            lenAng[1] = 360 - Math.asin(b / Math.sqrt(a*a + b*b)) * 180 / Math.PI;
            return lenAng;
        } else if (
                (FY0K.getId() == 2 || FY0K.getId() == 3) && (Ay >= 0 && Ay >= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180)) ||
                (FY0K.getId() == 4 || FY0K.getId() == 5) && (Ay >= 0 && Ay <= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180)) ||
                (FY0K.getId() == 6 || FY0K.getId() == 7) && (Ay <= 0 && Ay >= Ax *
Math.tan(FY0K.getAngle() * Math.PI / 180)) ||
```

```
                (FYOK.getId() == 8 || FYOK.getId() == 9) && (Ay <= 0 && Ay <= Ax *
Math.tan(FYOK.getAngle() * Math.PI / 180))
        ){
            double alpha1 = inAng(FY00, FY01, A);
            double alpha2 = inAng(FY00, FYOK, A);
//              System.out.println("alpha1 " + alpha1);
//              System.out.println("alpha2 " + alpha2);
//              System.out.println("θ  " + 40*(FYOK.getId()-1));
            double a = Math.sin(alpha1 * Math.PI / 180) * Math.cos(alpha2 * Math.PI / 180) -
                    Math.sin(alpha2 * Math.PI / 180) * Math.cos((alpha1 + 40*(FYOK.getId()-1))
* Math.PI / 180);
            double b = Math.sin(alpha2 * Math.PI / 180) * Math.sin((alpha1 + 40*(FYOK.getId()-1))
* Math.PI / 180) -
                    Math.sin(alpha1 * Math.PI / 180) * Math.sin(alpha2 * Math.PI / 180);
            double[] lenAng = new double[2];
            lenAng[0] = 100 * (a / Math.sqrt(a*a + b*b) + b / (Math.sqrt(a*a + b*b) *
Math.tan(alpha2 * Math.PI / 180)));
            lenAng[1] = Math.asin(b / Math.sqrt(a*a + b*b)) * 180 / Math.PI + 40*(FYOK.getId()-1);
            return lenAng;
        } else if (
                (FYOK.getId() == 2 || FYOK.getId() == 3) && (Ay <= 0 && Ay >= Ax *
Math.tan(FYOK.getAngle() * Math.PI / 180)) ||
                (FYOK.getId() == 4 || FYOK.getId() == 5) && (Ay <= 0 && Ay <= Ax *
Math.tan(FYOK.getAngle() * Math.PI / 180)) ||
                (FYOK.getId() == 6 || FYOK.getId() == 7) && (Ay >= 0 && Ay >= Ax *
Math.tan(FYOK.getAngle() * Math.PI / 180)) ||
                (FYOK.getId() == 8 || FYOK.getId() == 9) && (Ay >= 0 && Ay <= Ax *
Math.tan(FYOK.getAngle() * Math.PI / 180))
        ){
            double alpha1 = inAng(FY00, FY01, A);
            double alpha2 = inAng(FY00, FYOK, A);
            double a = Math.sin(alpha2 * Math.PI / 180) * Math.cos(alpha1 * Math.PI / 180) +
                    Math.sin(alpha1 * Math.PI / 180) * Math.cos((alpha2 - 40*(FYOK.getId()-1))
* Math.PI / 180);
            double b = Math.sin(alpha1 * Math.PI / 180) * Math.sin((alpha2 - 40*(FYOK.getId()-1))
* Math.PI / 180) -
                    Math.sin(alpha1 * Math.PI / 180) * Math.sin(alpha2 * Math.PI / 180);
            double[] lenAng = new double[2];
            lenAng[0] = 100 * (a / Math.sqrt(a*a + b*b) + b / (Math.sqrt(a*a + b*b) *
Math.tan(alpha1 * Math.PI / 180)));
            lenAng[1] = 360 - Math.asin(b / Math.sqrt(a*a + b*b)) * 180 / Math.PI;
            return lenAng;
        } else {
            double alpha1 = inAng(FY00, FY01, A);
            double alpha2 = inAng(FY00, FYOK, A);
//              System.out.println("alpha1 " + alpha1);
//              System.out.println("alpha2 " + alpha2);
//              System.out.println("θ  " + 40*(FYOK.getId()-1));
```

```
            double a = Math.sin(alpha2 * Math.PI / 180) * Math.cos(alpha1 * Math.PI / 180) +
                    Math.sin(alpha1 * Math.PI / 180) * Math.cos((alpha2 + 40*(FYOK.getId()-1))
* Math.PI / 180);
            double b = Math.sin(alpha1 * Math.PI / 180) * Math.sin((alpha2 + 40*(FYOK.getId()-1))
* Math.PI / 180) -
                    Math.sin(alpha1 * Math.PI / 180) * Math.sin(alpha2 * Math.PI / 180);
            double[] lenAng = new double[2];
            lenAng[0] = 100 * (a / Math.sqrt(a*a + b*b) + b / (Math.sqrt(a*a + b*b) *
Math.tan(alpha1 * Math.PI / 180)));
            lenAng[1] = Math.asin(b / Math.sqrt(a*a + b*b)) * 180 / Math.PI;
            return lenAng;
        }
    }

    private static double inAng(FY f1, FY f2, FY A){
        double a = distance(f1.getLength(), f1.getAngle(), f2.getLength(), f2.getAngle());
        double b = distance(f1.getLength(), f1.getAngle(), A.getLength(), A.getAngle());
        double c = distance(f2.getLength(), f2.getAngle(), A.getLength(), A.getAngle());
        return Math.acos((b*b + c*c - a*a) / (2*b*c)) * 180 / Math.PI;
    }
}
```

附件 4 FY.java

```
public class FY {
    private int id;
    private double length; // 径长
    private double angle; // 角度（0 - 359.999）
    private double dist; // 与目标点的距离

    public FY(int id, double length, double angle, double dist) {
        this.id = id;
        this.length = length;
        this.angle = angle;
        this.dist = dist;
    }

    public FY() { }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
```

```
    }

    public double getLength() {
        return length;
    }

    public void setLength(double length) {
        this.length = length;
    }

    public double getAngle() {
        return angle;
    }

    public void setAngle(double angle) {
        this.angle = angle;
    }

    public double getDist() {
        return dist;
    }

    public void setDist(double dist) {
        this.dist = dist;
    }
}
```