基于分组步进式优化策略的二次反射能量汇聚系统设计

摘要

二次反射光线汇聚系统作为当前太阳能利用中重要的优化技术，在双碳目标推行的当下无疑具有重要的研究价值。本文从优化角度出发，建立了光线比例优化模型，并通过分组变步长搜索方式确定参数，得到该系统下镜场排布的最优化方案，为后续相关研究推进提供了可行的参考方案

针对问题一，首先对镜场中央反射镜EF的形状与尺寸进行设计，通过文献查找与模拟比对确定反射镜形状为双曲线，并根据镜场外边界处定日镜反射光线模拟确定反射镜参数。考虑到定日镜反射光线存在宽度，无法做到反射光线严格指向双曲线上焦点，通过系统线放大率、反射镜张口半径及焦距比综合评估，确定反射镜理想参数方程。在此基础上建立坐标系，采取光线追迹方法模拟镜场光路情况，去除镜间空隙、邻镜遮挡等光线，计算得到镜场汇聚光线比例。考虑到不同位置下直线段反射镜最优长度、仰角等参数存在差异，采取分组规划策略将镜场划分为不同固定参数的区域，确定定日镜坐标与状态参数，在此基础上以镜场汇聚光线比例为目标，采取分组变步长搜索参数方法对模型进行优化，得到最优的平面镜布置参数，其大致分布规律为由内圈向边界呈“疏布长镜，密布短镜，疏布长镜”的变化趋势，在该策略基础上计算得该系统的最大汇聚比例能达到62.85%.

针对问题二，考虑到偏移的入射光线角度，首先需对镜面倾角进行修正，并取步长评估光线入射角度对阴影遮挡与光线照射比例的影响程度，得出方案调整的必要性。在问题一的基础上建立不同入射角度下光线比例优化模型在入射角变化范围内针对五种入射角度进行考虑，使用问题一建立的光线比例计算方式与优化模型，用分组变步长搜索参数方法得到每种角度下最佳布置策略。对结果可视化分析，得出当光线入射角度减小时，优化后的镜面总数下降，光线照射比例有所提升。镜场整体排布虽仍随组呈现遵循交替分布规律，但镜面数量较大组逐渐向镜场外边界转移，且由于反射镜投影的偏移，镜面数量与镜面长度的谷值所在组野以同方向移动。

针对问题三，结合曲线光学特性与提升的汇聚需求考虑，将平面镜优化为抛物面反射镜。考虑到模拟场景中入射光线很难做到与抛物面对称轴平行，反射光线存在一定散焦，定义镜面两端点反射光线在y轴上投影段与镜长比值为汇聚比，以此为标准评估抛物面汇聚作用。在问题二不同角度最优布置策略基础上增添汇聚比作为决策变量，采取分组变步长参数搜索方法遍历优化排布策略，经与问题二结果的可视化比对可以得出，使用抛物形反射镜后，光线照射最大比例均有所提高，且入射角度越大，比例提升越明显。

最后，以反射镜曲线参数、平面镜镜面长度及反射镜高度进行模型的灵敏度分析，结果反映本布置策略的鲁棒性较高，对条件变化具有良好的灵敏度。同时对模型优缺点进行评价，并提出了可行的推广方案。

关键词：二次反射；能量汇聚系统；光线追迹；分组步进式优化

一、 问题重述

1.1 问题背景

随着“双碳”目标的提出和绿色中国战略的推行,搭建新能源发电系统已经成为全休社会成员关注的焦点。太阳能的利用是国家推行新能源政策的战略重点之一。塔式光热作为目前大型商业化光热项目采取的主要形式，逐步成为光热发电研发和投资开发的重点。目前塔式光热发电系统对于太阳光的聚集和反射多采用一次反射，即通过镜场将太阳辐射聚集到距离地面一定高度的吸热器上。采用该种方式布置的吸热器，由于距离地面的高度往往在$100m$以上，吸热器外表面对流热损较大，热量损失较大，吸热器施工安装以及后期维护难度较大，这些都成为制约塔式太阳能热发电大规模发展的因素。

基于二次反射原理的塔式太阳能光热发电技术是在传统的塔式太阳能热发电技术之上拓展出来的一种聚光太阳能热发电模式。这种发电方式最早于1976年由以色列人 Rabl A 提出，通过安装在塔上的一个二次反射面将光能量会聚于离地面很近的吸热器，从而使吸热器的效率得到提高，同时解决了熔盐等导热材料上塔的难题,提升了系统的安全性能。通过对镜场的各项参数进行优化设计使得该技术的光利用效率最大，是提高定日镜场发电功率的重要手段。

1.2 问题要求

将汇聚系统简化至二维几何平面。平行光线先经过若干个长度不超过 2.5 的定日镜（直线段）反射到塔上的反射镜（曲线EF）上，再经反射后汇聚到吸热区域（直线段CD）上。已知镜场总长度为 400，吸热区域长度为 10，塔高度为 100。使用自行设计的反射镜形状，依次解决如下三个问题：

1. 在光线垂直入射的条件下，给出一种以长度不超过 2.5 的定日镜的排列方案，使得入射光线能以最大比例汇聚至吸热区域。
2. 在问题一的基础上,考虑入射光线角度在一定区间内变动，并给出相应的调整方案
3. 在问题二的基础上，如果镜面可以适当弯曲，通过调整每一段长度不超过 2.5 的光滑曲线形状和位置的设计以及角度变化。使得在入射光线角度变化过程中，光线经过两次反射后进入吸热的比例有所提高。

二、 问题分析

本题研究对象为二维定日镜场，研究内容为定日镜场的优化设计。其实际上为一道优化问题，要求我们设定定日镜场的相关参数，在不同条件下达到使得光线经过两次反射后进入吸热区域 CD 内的比例最大的目的。

2.1 曲线 EF 设计的分析

考虑到系统汇聚光特征以及简化计算需要，选择使用规则几何曲线模拟反射镜形状。将反射镜曲线中点正下方地面处作为原点建立镜场的平面直角坐标系，针对镜场边界点处的定日镜，考虑太阳—单个定日镜—反射镜—吸热器的简化系统，分析其内部光路情况，确定反射镜的的边界，从而可确定反射镜的具体尺寸。

2.2 问题一的分析

问题要求设计定日镜场参数并实现汇聚光线至区域内比例最大的目的。首先针对系统中光路采用光线追迹方法进行模拟，在确定参数的条件下得到光线比例的初始数据。为使得反射光线效率尽可能大，提出分组分区域规划策略，每一区域内分别采取不同排序方式，并以最大化汇聚比例作为优化目标建立光线比例优化模型，采用分组变步长搜索法寻找最佳的平面定日镜参数。对结果进行评估

2.3 问题二的分析

问题要求在已有模型基础上考虑入射光线角度变化的情况，并做出调整方案。首先对新条件下的光线比例计算方式进行更新，修正镜面倾角。考虑到镜面倾角对阴影遮挡存在很大影响，评估后得出需要对新数据进行重新优化。基于经过修正的光线比例计算方法和问题一建立的光线比例优化模型，使用分组变步长搜索参数策略得到每种角度下的最佳布置策略，并对结果进行分析。

2.4 问题三的分析

问题要求对直线段反射镜进行设计，修改为合适的光滑曲线，以提高光线照射比例。首先基于反射特点选择合适的反射镜形状，并定义汇聚比以描述反射镜汇平行光线的能力，并重新计算第一次反射光线的相关参数。基于前两问建立的光线比例计算方法，建立包含镜面汇聚比在内的多变量光线比例优化模型。采用分组变步长搜索法寻找最佳的反射镜参数，对结果进行评估。

三、 模型假设

1. 假设所有入射光线均为平行光，可以进行离散化处理；
2. 假设光线在反射时不存在能量损失等影响因素；
3. 假设地面上的直线段反射镜镜面平坦，对光线的反射能力相同；
4. 假设中央反射镜EF为理想的双曲线形状。

四、 符号说明

表 1 符号说明

符号	说明	单位
M_k	位于地面的第k面反射镜	/
L_k	第k面反射镜镜面长度	/
θ_k	第k面反射镜镜面与水平面夹角	rad
x_k	第k面反射镜底端坐标	/
f_1	双曲线上焦点到顶点的距离	/
f_2	双曲线下焦点到顶点的距离	/
k	f_2和f_1的比值	/
M	线放大率	/
R	中央反射镜半径	/
α	入射光线与地面夹角	rad

(x_{0i}, y_{0i})	第i根入射光线（延长线）与水平面的交点	Coordinate
(x_{1i}, y_{1i})	第i根入射光线与镜面的交点	Coordinate
(x_{2i}, y_{2i})	第一次反射光线与中央反射镜的交点	Coordinate
(x_{3i}, y_{3i})	第二次反射光线与水平面的交点	Coordinate
$\boldsymbol{i}$	入射光线方向单位向量	/
$\boldsymbol{r_1}$	第一次反射光线方向单位向量	/
$\boldsymbol{r}_2$	第二次反射光线方向单位向量	/
N	光线总数	/
N_{in}	进入区间CD的光线总数	/
P	光线照射比例	%
S_j	第j组区域长度	/
Δx_j	第j组区域内镜面间隔	/
K_j	第j组区域内镜面数量	/
L_j	第j组区域内镜面长度	/
K	镜面总数	/
P_{max}	最高光线照射比例	/
$\theta_k{}'$	修正后第k面反射镜镜面与水平面夹角	rad
ξ_k	第k面反射镜汇聚比	/

五、 模型建立与求解

5.1 曲线 EF（中央反射镜）的设计

5.1.1 坐标系建立与平面镜参数

根据题意，以反射镜曲线中点G在地面投影O为原点，水平方向为x轴，竖直方向为y轴建立竖直平面内的直角坐标系如图一所示。其中AB的长度为400，CD的长度为10，OG的高度为100.

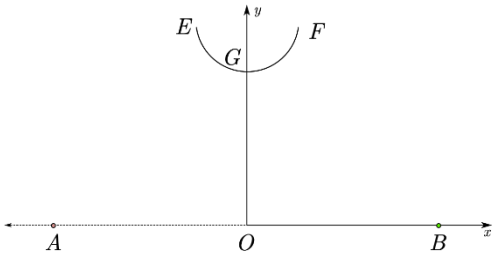

图 1 坐标系建立

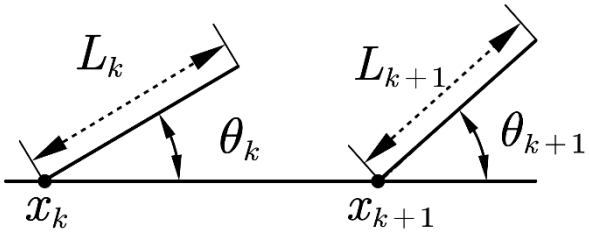

图 2 平面定日镜参数

在该坐标系内对第k面定日镜进行研究，记镜长L_k，镜旋转轴x_k，镜倾角θ_k，则定日镜状态可用数组$M_k(x_k,\theta_k,L_k)$进行表示。

5.1.2 中央反射镜 EF 形状设计

本系统的主要作用即为反转来自定日镜场的太阳光线，以便接收器可以放置在地面上。从光学角度来看，只有具有两个焦点的反射镜面才能完成这项任务，即指向其焦点之一的每条光线将被反射到位于地平面的第二个焦点。双曲线镜和椭圆镜均可完成此目标[1][2]。然而椭圆镜存在缺点：它的位置总是在汇聚点上方（即它总是需要更高的塔式反射器），而双曲面镜低于这个点。此外通过性能对比，双曲线形的实际应用效果明显优于椭圆形[3]。故本文只对双曲线反射镜进行考虑。

双曲线的光学特性为过其上焦点的光线必过其下焦点[4][5]，利用双曲线该特性，系统将定日镜反射的太阳光线对准双曲线上焦点，经双曲线二次反射至位于下焦点位置的吸热器入口。

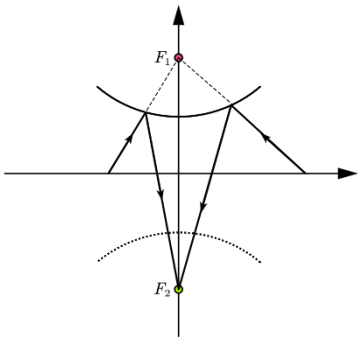

图 3 双曲线的光学性质

焦点在y轴上的双曲线标准方程为$\frac{y^2}{a^2}-\frac{x^2}{b^2}=1$，考虑到实际模型，只保留位于$y$轴正半轴的部分。记$F_1$到双曲线顶点距离为$f_1$，$F_2$到双曲线顶点距离为$f_2$.

5.1.3 中央反射镜 EF 参数设计

结合题意构建以下模型：针对镜场边界点B处定日镜，考虑太阳—单个定日镜—反射镜—吸热器的简化系统。假设太阳光垂直下射进入系统，将下焦点F_2放置于坐标系原点，平面反射镜将反射光线对准上焦点P_1，则双曲线方程可设为

$$\frac{(y-c)^2}{a^2}-\frac{x^2}{b^2}=1 \tag{1}$$

P_1、G、O三点间距离满足关系式：

$$\begin{cases} f_1=c-a=P_1G \\ f_2=c+a=OG \end{cases} \tag{2}$$

为保证所有第一次反射光线都能被双曲线反射镜接收，考虑位于边界点B的定日镜，该点的第一次反射光线沿P_1B方向，记连线P_1B与双曲线的交点为F点，从而可以得到曲线EF长度，如图 4 所示。

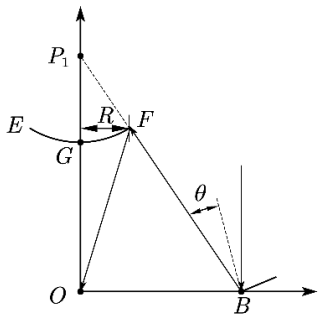

图 4 双曲线反射镜半径示意图

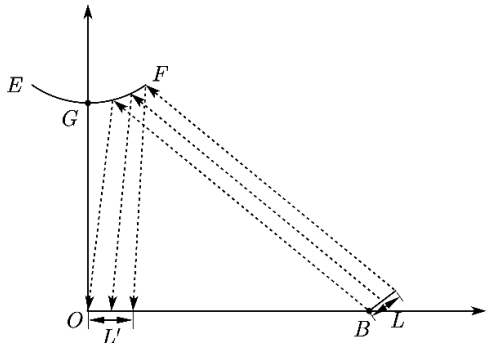

图 5 双曲线反射镜放大倍率示意图

如图 5 所示，由于定日镜反射光线为区域内平行光线，存在宽度，部分光线并非完全严指向汇聚点，故同一面定日镜对于不同面形的双曲线，投射在吸热器上的长度不同。根据几何光学物像放大关系可知，光学系统中线放大率为

$$M=\frac{L'}{L} \tag{3}$$

双曲面的面形可由 $k=f_2/f_1$ 决定，故k值变化时，双曲线张口半径R会一同变化，定日镜投射到吸热器入口面的线放大率M也随之变化。通过数据点模拟，绘制出三者的关系曲线。

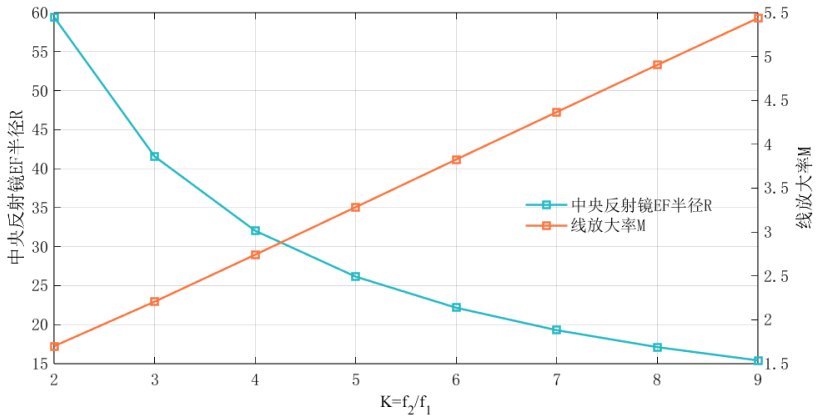

图 6 线放大率M、半径R与k值的关系示意图

图 6 表明，线放大率M和双曲面张口半径R随k的变化趋势相反，即M随k值增大而增大，双曲面半径随k的增大而减小。然而在电站设计中，不仅需要考虑吸热器表面光线的会聚状况，也要兼顾反射塔上双曲线的尺寸大小。当塔高不变时，上焦点越高，即f_1越大，反射镜EF的半径会显著增加，不仅会遮挡更多入射光线，也会给承重带来负担。如果上焦点位置较低，则会导致反射镜的线放大率M较大，光线不能很好地汇聚在区间CD内。综合上述考虑，当$k=4$时(即$M=2.74, R=32$时)，为系统于一般情形下较理想的面形设计参数。由此可以推得E、F的坐标以及反射镜的方程：

$$E(-32,107),\ F(32,107)$$

$$\frac{(y-62.5)^2}{37.5^2}-\frac{x^2}{50^2}=1$$

5.2 问题一模型建立与求解

5.2.1 光线比例计算

在问题一中，光线与AB垂直，即入射角度$\alpha=90°$。得到入射光线单位向量$\boldsymbol{i}=(0,-1)$. 为保证计算的准确性同时节省时间和复杂度，本文采用光线追迹方法进行计算。假设光线总数为N，均匀分布在线段OB上。考虑第i根入射光线，得到方程为

$$\frac{y}{i_x}=\frac{x-x_{0i}}{i_y} \tag{4}$$

其中x_{0i}表示第i根入射光线（或延长线）与x轴的夹角.

STEP1:计算入射光线是否照射在平面镜M_k上

对于平面镜M_k，由于直线段最低点的反射光线始终指向双曲线的上焦点，故可根据几何关系知，若确定平面镜参数x_k，则唯一确定平面镜与水平方向夹角

$$\theta_k=\frac{1}{2}\arctan\left(\frac{x_k}{f_1+f_2}\right) \tag{5}$$

得到平面镜的直线方程为

$$y=\tan\theta_k(x-x_k),\ x\in[x_k,x_k+L\cos\theta] \tag{6}$$

联立(4) (6)式，可解得在在平面镜的有限长度内，光线是否会与平面镜相交，即光线是否会照射在平面镜上，记交点为(x_{1i},y_{1i})。从而可知，在AB范围内存在两种情况入射光线不会被反射：

- 入射光线被中央反射镜 EF 曲线遮挡（如图 7(a)所示）
- 入射光线照射在两镜面之间（如图 7(b)所示）

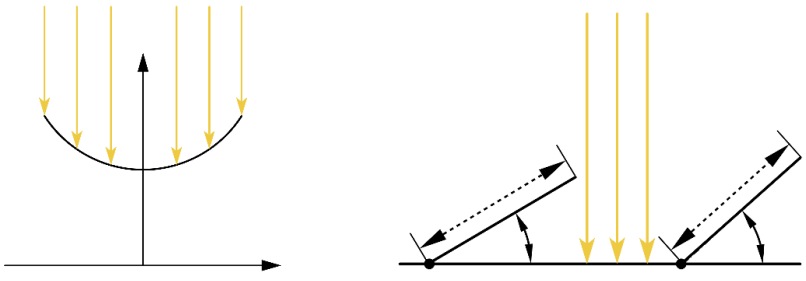

图 7 入射光线不被反射的两种情况（a）（b）

STEP2:计算第一次反射光线方程

若入射光线被平面镜M_k反射，则可知反射光线的直线方程为

$$y-y_{1i}=-\frac{1}{\tan 2\theta}(x-x_{1i}) \tag{7}$$

记$k_1=-\frac{1}{\tan 2\theta}$，则反射光线的单位向量为$\boldsymbol{r_1}=\left(\frac{1}{\sqrt{1+k_1^2}},\frac{k_1}{\sqrt{1+k_1^2}}\right)$.

对于反射光线，需判断其是否会被相邻镜面M_{k-1}遮挡。

将(6)(7)式联立

$$\begin{cases} y=\tan\theta_{k-1}(x-x_{k-1}),x\in[x_{k-1},x_{k-1}+L_{k-1}\cos\theta_{k-1}] \\ y-y_{1i}=-\dfrac{1}{\tan 2\theta_k}(x-x_{1i}) \end{cases} \tag{8}$$

可得到潜在交点(x_d,y_d)，如果满足条件

$$x_{k-1}\leqslant x_d\leqslant x_{k-1}+L_{k-1}\cos\theta_{k-1}$$

则该反射光线会被遮挡，如图 8 所示

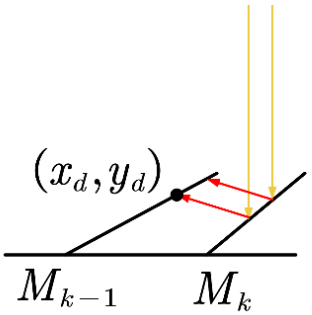

图 8 反射光线被相邻镜面遮挡

STEP3:计算第一次反射光线与中央反射镜 EF 的交点

若反射光线不被遮挡，则其一定会被中央反射镜EF双曲线接收。将(1) (7)式联立可得反射光线与中央反射镜交点(x_{2i},y_{2i})

STEP4:计算第二次反射的法线和反射光线的单位向量

对双曲线方程求导得，第二次反射点的切线斜率为

$$f'=\frac{x_{2i}}{b^2}\cdot\frac{a^2}{y_{2i}-c} \tag{9}$$

故该点法线斜率为$k=-\frac{1}{f'}$，法线单位向量为$\boldsymbol{n_2}=\left(\frac{1}{\sqrt{1+k^2}},\frac{k}{\sqrt{1+k^2}}\right)$.

根据反射定律，可得第二次反射光线的单位向量为

$$\boldsymbol{r_2}=\boldsymbol{r_1}-2(\boldsymbol{r_1}\cdot\boldsymbol{n})\cdot\boldsymbol{n} \tag{10}$$

STEP5:计算第二次反射光线在x轴上的照射点

第二次反射光线的直线方程为

$$y-y_{2i}=\frac{r_{2y}}{r_{2x}}(x-x_{2i}) \tag{11}$$

故可得其与x轴的交点为$(x_{3i},0)$. 若$x_{3i}\in[-5,5]$，则该光线可被线段CD接收。

按照以上步骤，使用光线追迹方法进行遍历。在问题一中，光线的反射情况关于y轴对称，故仅考虑位于x轴正半轴的情况。假设经过两次反射后照射到线

段CD上的光线数量为N_{in}，则比例为

$$P=\frac{N_{in}}{N}\times 100\%$$

根据上述光线比例计算方法，本文中取光线总数$N=20000$，在平面镜均匀分布、平面镜各参数相同的情况下，考虑镜面数量和镜面长度对光线汇聚比例的影响，如图 9 所示。结果表明，当镜面数量较少时，汇聚比例会随镜面长度呈“先增后减”趋势；镜面数量较多时，镜面长度较短的情况效果更好。

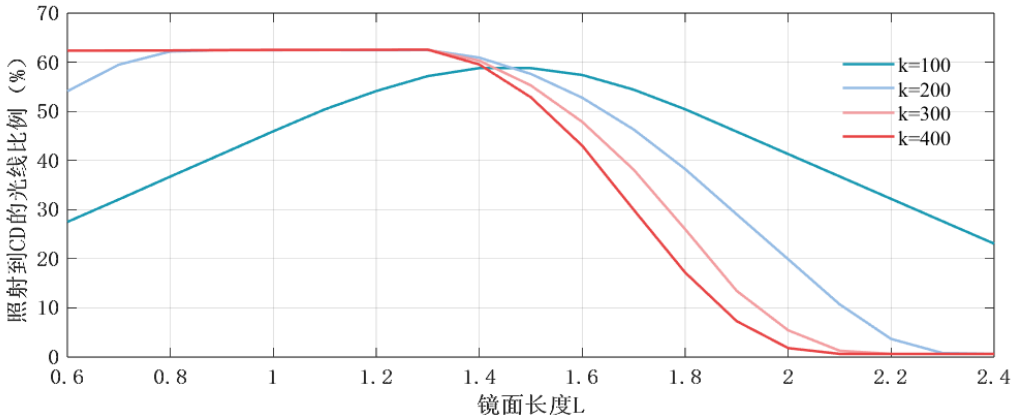

图 9 镜面数量、镜面长度对光线汇聚比例影响

5.2.2 平面镜的分区域规划策略

考虑到不同位置下定日镜最优长度、仰角等参数不尽相同，制定分组规划策略确定定日镜位置，从而达到最大转化镜场接收光线的目的。

如图 10 所示，以受太阳光照射地面区域边界点（反射镜边界点F在地面投影点）为起始点，在其向镜场边界B点延伸路径上将区域划分为n组。

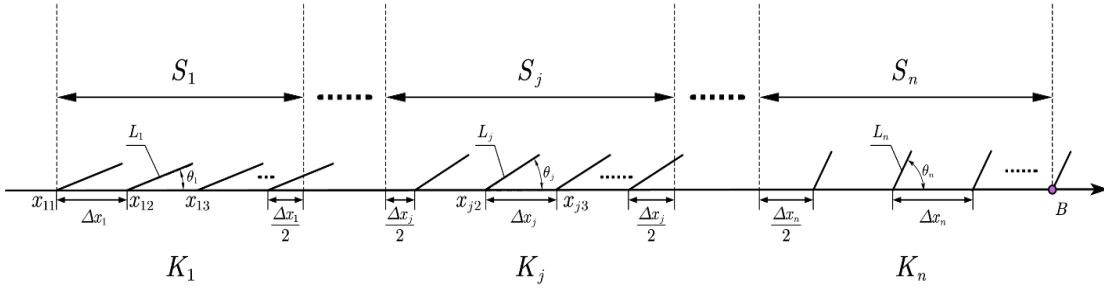

图 10 定日镜分组示意图

考虑其中第j组区域，记其范围内镜面数量为K_j，区域长度为S_j，该组内所有定日镜长度均为L_j，以Δx_j等间距排列，并按各自坐标为固定轴与地面以θ_j为倾角倾斜。与邻组交界处预留出$\frac{1}{2}\Delta x_j$作为衔接。在给定n、K_j、S_j的情况下，可以唯一确定镜场中定日镜总数、各定日镜具体坐标与状态参数。

5.2.3 光线比例优化模型的建立

本文建立了光线比例优化模型，在平面镜长度、平面镜距底座中心距离等约束条件下，确定平面镜场参数，使得入射光线以最大比例汇聚至吸热区域。

- **决策变量**

单组内平面镜镜长L_j，平面镜数量K_j的改变会影响该组的反射光线的遮挡情况和反射效率，从而影响到镜场的反射光线比例。因此决策变量设置为L_j，K_j.

- **目标函数**

问题的优化目标为光线进入CD区间内的比例最大。因此，目标函数为：

$$\max\ P=f(K_1,K_2,...,K_n,L_1,L_2,...L_n) \tag{12}$$

其中K_j、L_j分别表示第j组的定日镜镜长与镜数量

- **约束条件**

1. **平面镜尺寸约束：**平面镜镜长必须大于 0 且不超过 2.5，即

$$0<L_k\leqslant 2.5 \tag{13}$$

2. **平面镜位置约束：**平面镜必须位于镜场范围内，且在吸热区域以外，即

$$5\leqslant x_k\leqslant 200 \tag{14}$$

3. **平面镜排布顺序约束：**平面镜的排布必然是从左往右依次进行的，即

$$x_{k-1}<x_k \tag{15}$$

4. **分组内平面镜间隔约束：**平面镜间隔与组内镜面数量和分组长度关系为

$$\Delta x_j=\begin{cases}\dfrac{S_j}{K_j},j\neq 1,n\\ \dfrac{S_j}{K_j-0.5},j=1,n\end{cases} \tag{16}$$

5. **平面镜角度约束：**由几何关系和双曲线光学性质知镜面角度为

$$\theta_k=\frac{1}{2}\arctan\left(\frac{x_k}{f_1+f_2}\right) \tag{17}$$

综上，本文建立了光线比例优化模型

$$\max\ P=f(K_1,K_2,...,K_n,L_1,L_2,...,L_n)$$

$$s.t.\begin{cases}0<L_k\leqslant 2.5\\ 5\leqslant x_k\leqslant 200\\ x_{k-1}\leqslant x_k\\ \theta_k=\dfrac{1}{2}\arctan\left(\dfrac{x_k}{f_1+f_2}\right)\\ \Delta x_j=\begin{cases}\dfrac{S_j}{K_j},j\neq 1,n\\ \dfrac{S_j}{K_j-0.5},j=1,n\end{cases}\end{cases} \tag{18}$$ t

5.2.4 光线比例优化模型的求解

本文采用分组变步长搜索法寻找最佳的平面定日镜参数，在满足平面镜长度、角度、位置、排布顺序及间隔的约束条件下使得照射在区间CD内的光线比例最大化。分组变步长搜索法寻找最佳参数$K_1,K_2,...,K_n,L_1,L_2,...,L_n$的流程图如图 11 所示。

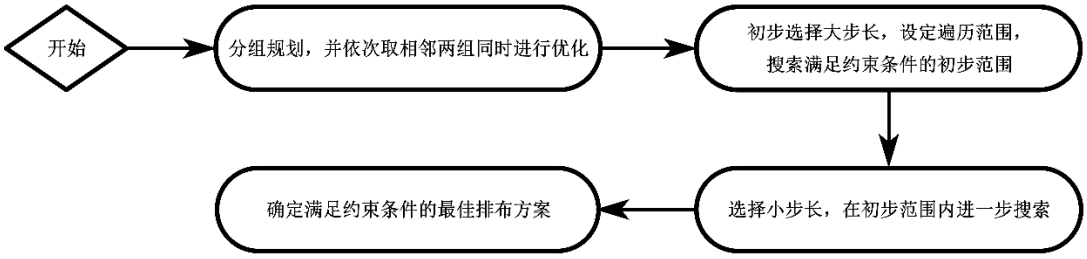

图 11 分组变步长搜索法寻找最佳平面镜参数流程图

STEP1:分区域规划，并选择第j和$j+1$组作为优化区域

STEP2:初步选取大步长和遍历范围，进行初次遍历。根据光线比例优化模型的约束条件，得到分组的参数初始搜索步长和遍历范围，如表 2 所示。

表 2 平面镜参数初始搜索步长和遍历范围

参数	镜面数量K_j	镜面长度L_j
初始步长	5	0.3
遍历范围	[10, 50]	[0.6, 2.4]

各参数在对应遍历范围内，以对应步长进行遍历，计算照射在区间CD内的光线比例，搜索得到满足约束条件的定日镜场各参数的初步范围K_j'和L_j'。

STEP3:缩短搜索步长，在初步范围内进一步搜索最优的平面镜参数。对于各平面镜参数，将搜索步长缩短，在初步范围内以小步长重新遍历镜面数量K_j和镜面长度L_j，如表 3 所示。

表 3 平面镜参数缩短步长和遍历范围

参数	镜面数量K_j	镜面长度L_j
步长	1	0.08
遍历范围	$[K_j'-3,\ K_j'+3]$	$[L_j'-0.2,\ L_j'+0.2]$

STEP4:遍历搜索结果和优化区域。经过分组变步长遍历搜索后，得到第j和$j+1$组区域内平面定日镜的最优参数。保留第j组的优化结果，针对区域$j+1$和$j+2$组重复上述步骤，直到得到整体的最优布置策略。此时，经过两次反射后照射到CD的光线比例达到最大。

5.2.5 问题一求解结果

基于光线比例优化模型和分组变步长搜索法，得到最优的平面镜布置参数如表 4 所示。分布情况关于y轴对称，镜面数量$K=542$.

表 4 最优布置策略（x负半轴区域关于此对称）

分组	1	2	3	4	5	6	7	8
镜面数量K_j	34	35	35	40	33	37	28	29
镜面长度L_j	1.24	1.24	0.64	1.40	1.54	0.94	1.54	1.70
区域左端点	32	53	74	95	126	137	158	179

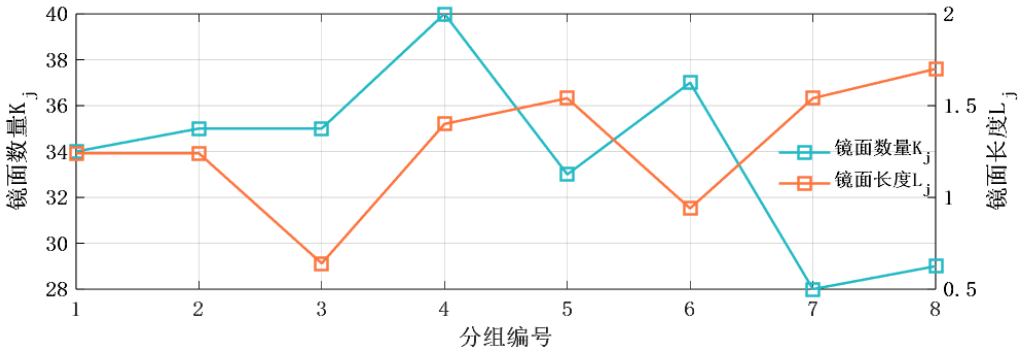

图 12 镜面数量、镜面长度的分组最优布置策略

图 12 表明，在最优排布策略下，每组的镜分布从近到远依次按“疏、密、疏”方式排布。镜分布稀疏的区域往往要求镜长更长，镜分布稠密的区域则倾向使用镜长更短的定日镜。考虑到阴影遮挡及区域长度相对固定的前提，这种排布策略无疑是具有极高的合理性的：近端定日镜倾角较小，故可采取更长镜面直铺覆盖全区域，中段相邻镜之间已存在遮挡关系考量，可通过短镜的密集叠放来消除遮挡影响，远端则由于镜倾角较大，相邻镜面间遮挡关系较为严重，故采取长镜拉远间距排布，达到最优化反射光线的目的。

在上述得出的最优排布策略的基础上，使用光线比例计算方法，得到该系统的最大汇聚比例能达到 **62.85%**.

5.3 问题二模型建立与求解

5.3.1 光线不垂直入射时光线比例计算

如图 13(a)所示，假设入射光线以与水平面夹角α照射，考虑对称性并结合题意，可得$\alpha \in \left[\frac{\pi}{4}, \frac{\pi}{2}\right]$. 设镜面$M_k$的反射光线与$y$轴夹角为$\beta_k$，结合反射定律可知，镜面与水平面的夹角为

$$\theta_k = \frac{1}{2}(\alpha + \beta_k) - \frac{\pi}{4} \tag{19}$$

当$\beta_k < \frac{\pi}{2} - \alpha$时$\theta_k < 0$，表明镜面倾斜情况与问题一中相反，如图 13(b)所示。此时需对θ_k进行修正：

$$\theta_k' = \theta_k + \pi \tag{20}$$

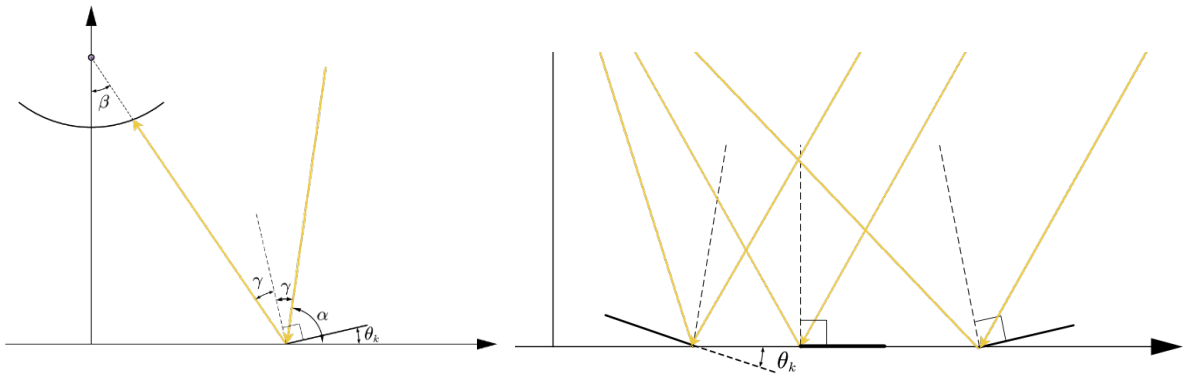

图 13 光线不垂直入射时情况分析

考虑到镜面倾角与阴影遮挡的强相关性，以9°为步长在45－90°范围内取不同光线入射角度，计算镜场各处对应镜面倾角，二者变化的关系曲线图如下所示：

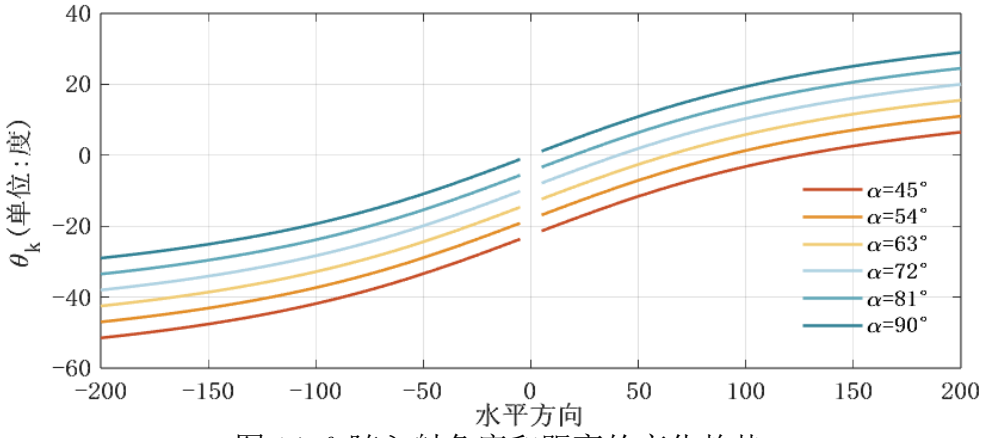

图 14 θ_k随入射角度和距离的变化趋势

从图中可以看到，入射角α对θ_k有较大影响。位于原点左侧部分的θ_k普遍小于0°，且当α较小时，θ_k会达到－50°。考虑到镜面倾角增大会导致相邻镜子间存在更多遮挡，削减汇聚光线比例。因此，利用合适的优化策略对这一模型的优化是有必要的。

在平面镜均匀分布、平面镜各参数已知且相同的情况下，评估入射角度变化对于对光线汇聚比例的影响。出于简化模型考虑和对称性，在此只考虑系统半边。对于入射角度α，所求的照射比例即为

$$P=\frac{1}{2}(P_{\alpha}+P_{\pi-\alpha}) \tag{21}$$

单边光线照射比例P随α的变化如下图所示：

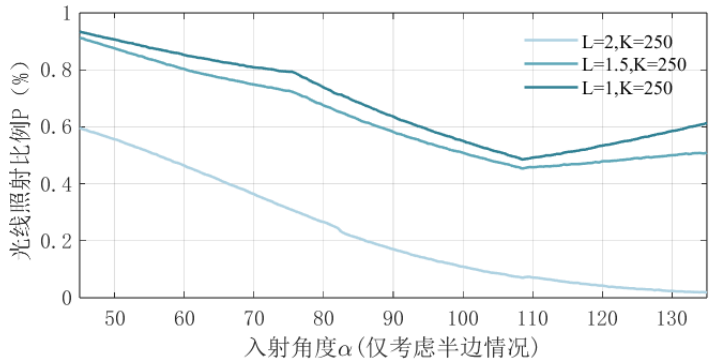

图 15 不同平面镜参数情况下单边光线照射比例随入射角度变化趋势

由图 15 可以看出，图像的整体趋势分为三段，在$\alpha=80°$与$\alpha=110°$左右出现明显转折。三段区域从左到右依次记为$\mathrm{Part}_i(i=1,2,3)$。记光线被遮挡比例为P_H。

对于Part_1，当入射角度较小时，光线不会因曲线EF遮挡而损失比例。随着角度的增大，第一次反射光线越容易被相邻反射镜遮挡，造成比例下降。

对于Part_2，部分光线开始被EF遮挡。随着α的增加，P_H逐渐增加，造成光线照射比例P的下降速率增加。

对于Part_3，光线照射比例P趋于稳定，表明曲线EF在地面上的投影已经完

全落在平面镜布置区域内。在镜面数量固定的情况下，L较大时，相邻镜面之间的遮挡比例会显著增加，直至几乎完全挡住第一次反射光线。

5.3.2 基于分组变步长优化策略的调整方案

问题二中光线的入射角度变化范围为45 — 90°。在本问题的调整方案中，针对五种入射角度进行考虑。基于前文经过修正的光线比例计算方法和问题一建立的光线比例优化模型，使用分组变步长搜索参数策略得到每种角度下的最佳布置策略。结果如表 5 所示（具体结果见附录文件）：

表 5 不同入射角度的平面镜分组参数调整方案

分组		-8	…	-1	1	…	8	P_{max}
45°	K_j	38	…	17	17	…	37	76.76%
	L_j	1.54	…	1.16	1.4	…	2.28	
…	K_j			…				…
	L_j			...				
81°	K_j	34	…	7	37	…	32	63.92%
	L_j	1.62	…	0.4	1.1	`…	1.84	

将上述结果可视化，如图 16-18 所示：

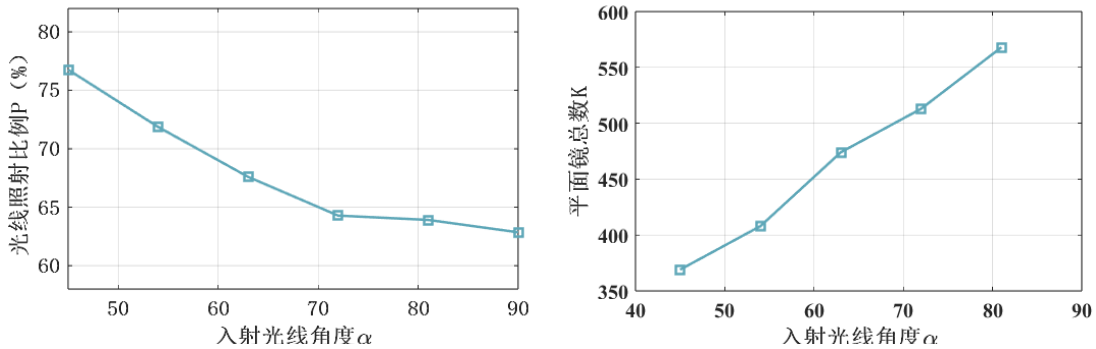

图 16 光线照射最高比例、镜面总数随入射光线角度变化图

当入射光线角度逐渐减小时，经过优化后的光线照射比例P会有所提高，所使用平面镜总数也随之减小。其可能原因是更小的入射光线角度对镜面倾角要求更高，导致相同镜面密度情况下，相邻镜面间遮挡的影响更加严重，从而使得经过优化后，镜面分布更加分散。

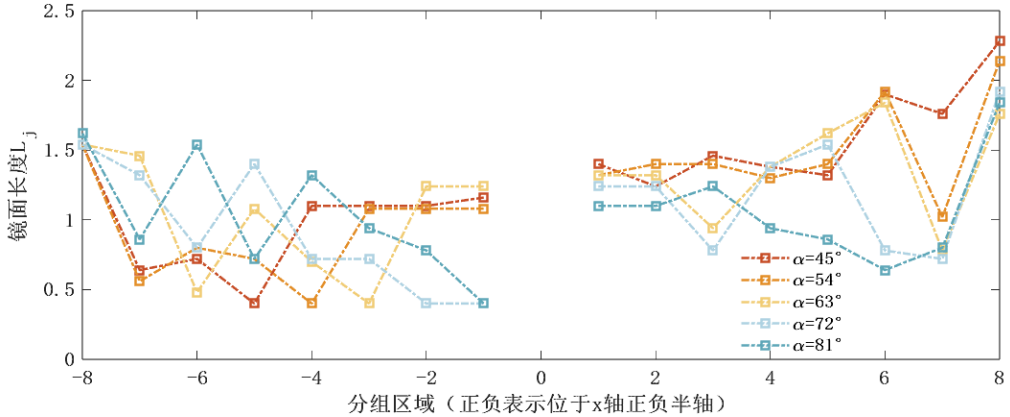

图 17 不同入射角度的镜面数量的分组布置最优策略

可以看到，尽管入射光线角度发生改变，各组安排定日镜数量由镜场内圈至外围边界处整体上仍呈现“疏—密—疏”的交替分布规律。随着入射系统光线的倾角在$45-90°$的区间内减小，镜面数量整体上呈现下降趋势，镜面数量较大的组由镜场内圈逐渐向外边界转移，在图像上呈现为曲线波峰向外侧移动。值得注意的是，曲线在x轴负半轴区域存在极小值，且该点随入射角减小向负半轴方向移动。其可能原因是入射光线减小带来的反射镜EF在水平面上投影移动。投影覆盖范围与极小值点移动规律正好吻合，证明该方案与优化策略存在很大程度上的合理性。

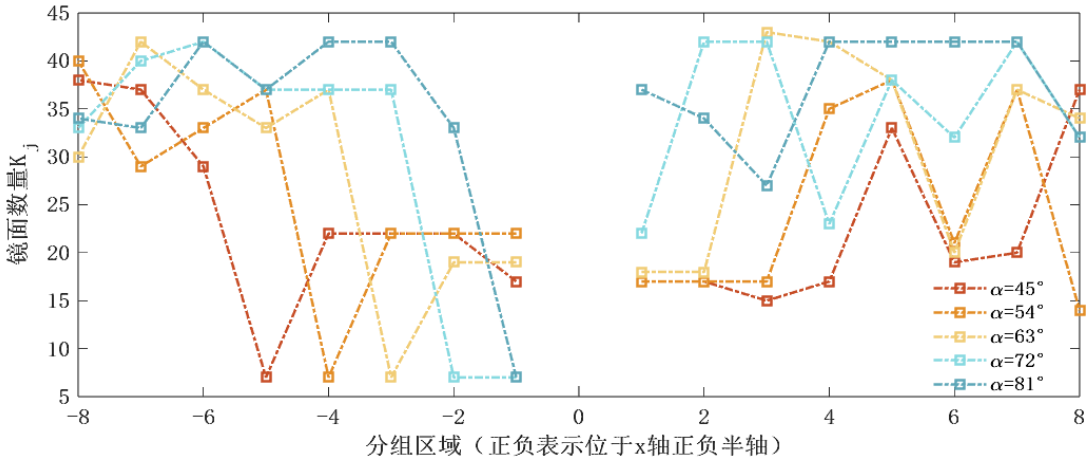

图 18 不同入射角度的镜面长度的分组布置最优策略

从图中可以看到，各组镜面长度在镜场中整体上由内圈向外边界呈现出增长趋势，具体变化与该组镜面数量在镜场中分布规律变化存在较强的负相关性，只在阴影遮挡对应组内二者均小（因为此时该区域内并无入射光线，镜面排布并无太大意义）。该规律与光线垂直入射时各组镜面长度与镜面数量分布规律基本吻合，证明方案存在合理性。

5.4 问题三模型建立与求解

5.4.1 光滑曲线反射镜设计

在太阳能利用领域，以圆锥曲面为代表的聚光型反射器是优化定日镜的主要方向，考虑到太阳光线的准平行性与实际聚光需求，抛物面型聚光器又是其中的主要对象。平行于轴线的光线经抛物线反射后汇聚于焦点，从焦点发出的光线经抛物线反射后平行于轴线，这是抛物线的反射特性。虽然实际运用中难以做到完全平行轴线或点光源，然而目前研究[6][7]指出，抛物面反射镜对倾斜入射光仍然具有一定的聚光作用，在聚光度要求不高时，它仍可以利用，如图 19 所示。故在此我们选择抛物线作为镜面的优化方向。

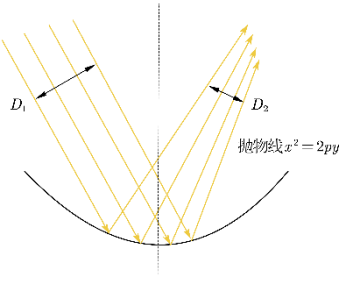

图 19 抛物线光学性质

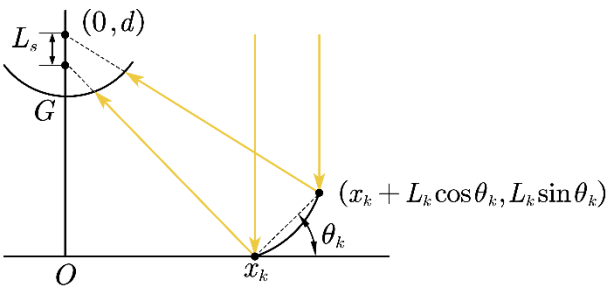

图 20 抛物反射镜汇聚比

考虑到不与抛物线对称轴平行的光线入射时反射光线不会完全交于其焦点，而是具有一定散焦性，因此需要定义某个参数用来评估抛物线镜面的汇聚作用。

如图 20 所示，出于简化模型考虑，我们假设所有经镜面反射的光均落在L_s区域内，故此处只用对光线经镜面左、右两端点反射与y轴交点两个边界条件进行计算。已知抛物线左端点反射光线汇聚于反射镜焦点，设经过抛物线汇聚后，经过镜面M_k右端点的反射光线与y轴交于$(0,d)$，则可定义汇聚比为

$$\xi = \frac{L_s}{L_k} = \frac{d - f_1 - f_2}{L_k} \tag{22}$$

假设第i根入射光线与镜面交点为(x_{1i}, y_{1i})，则根据汇聚比的定义，可得到其第一次反射光线的延长线与y轴交点为

$$Q\left(0, f_1 + f_2 + \frac{x_{1i} - x_k}{\cos\theta_k}\xi\right) \tag{23}$$

得到对应的第一次反射光线单位向量$\boldsymbol{r}_1'$和直线方程，后续计算步骤同问题一的光线比例计算模型。

然而值得注意的是，汇聚比并非越小越好。当汇聚比过小时，过强的汇聚能力对反射光线的角度偏移过大，镜面右端反射的部分光线反而被相邻反射镜遮挡，造成损失，如图 21(a)所示。当汇聚比过大的时候，第一次反射光线可能较为离散，受到双曲线中央反射镜EF光学性质的影响，部分光线经过第二次反射无法照射到吸热区域CD内，如图 21(b)所示。

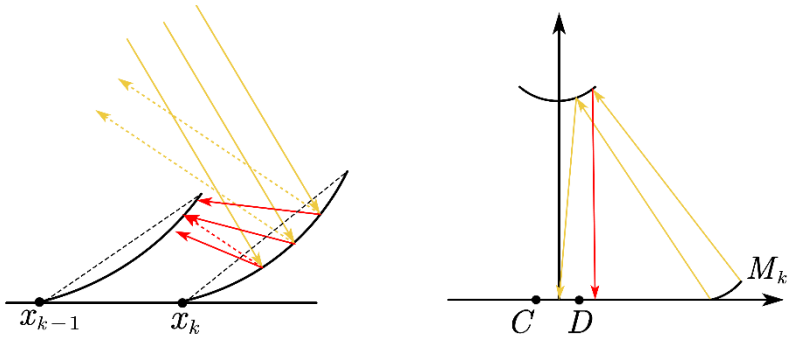

图 21 汇聚比过小或过大可能出现的情况

为寻找最合适的汇聚比，使用光线比例计算方法，计算地面反射镜均匀分布且参数相同情况下，光线照射比例随汇聚比的变化趋势，如图 22 所示。

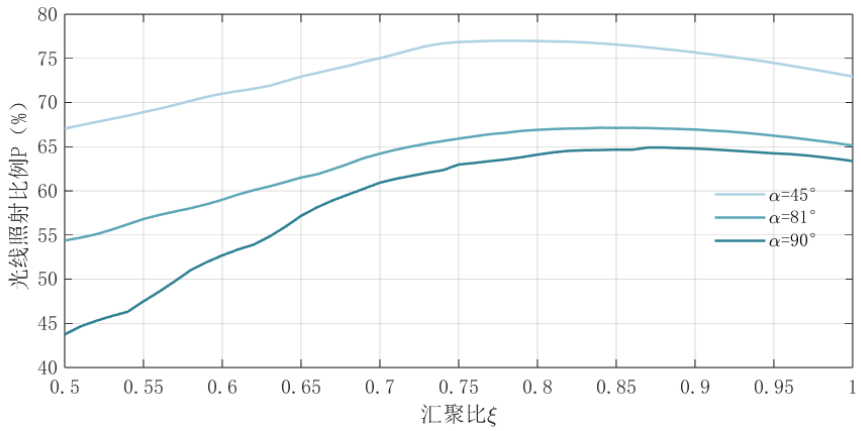

图 22 不同入射角度下光线照射比例随汇聚比ξ的变化趋势

可以看到，随着汇聚比的提高，光线照射比例呈现先增后减的趋势，且在增长阶段光线照射比例变化较快，与前文定性分析得到的结论和原因一致。同时，不同入射角度情况下，最佳汇聚比ξ会有差异。

沿用问题一的分区域规划策略，将镜面汇聚比ξ_k引入新的镜面参数，得到$M_k(L_k,\theta_k,x_k,\xi_k)$.为简化计算，假设在每个区域内的各参数相同，

5.4.2 基于抛物线地面反射镜的光线比例优化模型建立

本文建立了光线比例优化模型，在问题二求得的不同入射角度下最优布置策略的基础上，确定曲线镜场参数，使得入射光线以最大比例汇聚至吸热区域。

- **决策变量**

单组内的镜面汇聚比改变会影响该组的反射光线的遮挡情况和反射效率，从而影响到镜场的反射光线比例。因此决策变量设置为ξ_j、L_j、K_j.

- **目标函数**

问题的优化目标为光线进入CD区间内的比例最大。因此，目标函数为：

$$\max\ P=f(\xi_1,\xi_2,...,\xi_n,K_1,K_2,...,K_n,L_1,L_2,...L_n) \tag{24}$$

其中ξ_j、L_j、K_j分别表示第j组的曲线镜汇聚比、镜长与镜数量.

- **约束条件**

相较于光线比例优化模型的约束条件（式（13）至式（17））,发生改变的约束条件为：

1. 平面镜角度约束：

$$\theta_k=\frac{1}{2}(\alpha+\beta_k)-\frac{\pi}{4} \tag{25}$$

2. 汇聚比约束：

考虑到斜入射抛物线反射镜的光线并非完全交于一点，以及抛物线反射镜实际的加工精度，有汇聚比的范围为

$$\xi\geqslant 0.2 \tag{26}$$

综上，本文建立了基于抛物线地面反射镜的光线比例优化模型

$$\max\ P=f(\xi_1,\xi_2,...,\xi_n,K_1,K_2,...,K_n,L_1,L_2,...L_n)$$

$$s.t.\begin{cases} 0<L_k\leqslant 2.5 \\ 5\leqslant x_k\leqslant 200 \\ x_{k-1}\leqslant x_k \\ \theta_k=\dfrac{1}{2}(\alpha+\beta_k)-\dfrac{\pi}{4} \\ \dfrac{\pi}{4}\leqslant\alpha\leqslant\dfrac{\pi}{2} \\ \xi_k\geqslant 0.2 \\ \Delta x_j=\begin{cases}\dfrac{S_j}{K_j},j\neq 1,n \\ \dfrac{S_j}{K_j-0.5},j=1,n\end{cases}\end{cases} \tag{27}$$

5.4.3 问题三的求解与结果

与问题一、二相同，使用分组变步长参数搜索方法，寻找不同角度下的抛物线形反射镜参数和布置策略，搜索步长和遍历范围如表 6 所示：

表 6 抛物线形反射镜参数搜索步长和遍历范围

参数	镜面数量K_j	镜面长度L_j	镜面汇聚比ξ_j
初始步长	5	0.3	0.05
遍历范围	[10, 50]	[0.6, 2.4]	[0.6, 1.0]
缩短步长	1	0.08	0.01
遍历范围	$[K_j'-3, K_j'+$	$[L_j'-0.2, L_j'+0.2]$	$[\xi_j'-0.03, \xi_j'+0.03]$

将所得结果可视化，如图 23 所示：

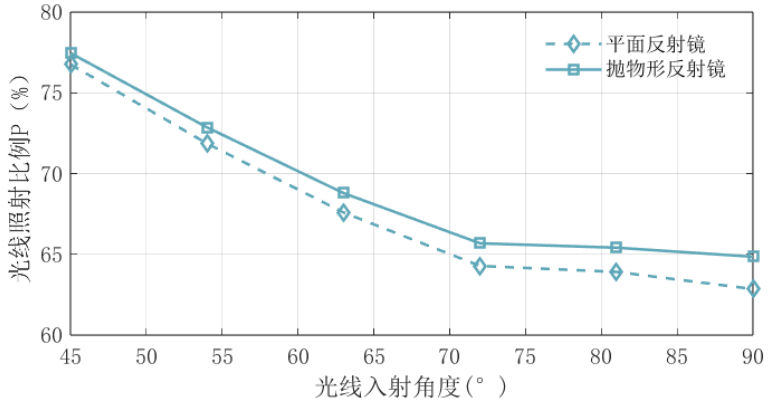

图 23 光线照射最高比例随入射角度变化趋势图

可以看到，**使用抛物形反射镜后，光线照射最大比例均有所提高，**且入射角度越大，比例提升越明显。

六、 灵敏度分析

6.1 模型对曲线 EF 参数设计的灵敏度

在上文中，假设中央反射镜EF参数为$f_2/f_1=4$。考虑实际情况，为探究中央反射镜EF参数改变对光线照射比例的影响程度，我们使参数$k=f_2/f_1$在$[3.8,4.2]$范围内，以0.1为步长进行调整，并计算光线照射比例的取值。

不同中央反射镜EF的参数$k=f_2/f_1$下光线照射比例随角度变化如图 24 所示：

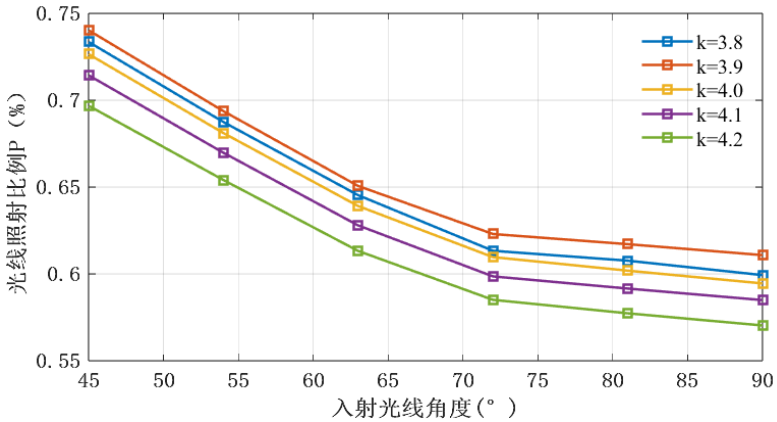

图 24 不同中央反射镜EF的参数$k=f_2/f_1$下光线照射比例随角度变化图

从图 24 中可以看到，随着中央反射镜参数$k=f_2/f_1$的改变，最大光线照射比例的变化在-2.4%~2.8%之间。且在计算范围内，在$k=3.9$时比例最高；而$k=3.8$和$k=4.0$时结果较接近，说明系统的鲁棒性较好。

6.2 模型对中央反射镜高度的灵敏度

在上文中，中央反射镜的高度为 100. 考虑到实际二次反射式集热装置的高度各有不同，为探究中央反射镜高度的改变对平面镜场的光线照射比例的影响程度，我们使中央反射镜高度在$[80,120]$范围，以10为步长进行调整，并计算光线照射比例的取值。

不同中央反射镜EF高度f_2下光线照射比例随角度变化如图 25 所示：

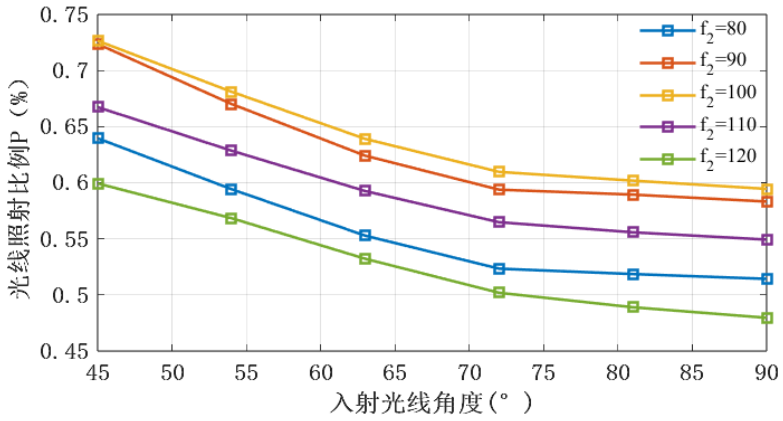

图 25 不同中央反射镜EF高度f_2下光线照射比例随角度变化图

从图 25 可以看出，随着中央反射镜的高度增加，光线照射比例呈现先增后减的趋势，且在$f_2=100$时光线照射比例最高。在进行定日镜场设计时，中央反射镜的高度是关键因素。

6.3 模型对最优布置策略中镜面长度的灵敏度

在上文中，得到不同角度下的镜面分组最优布置策略。考虑到实际情况中，镜面长度可能存在精度误差，为探究镜面长度的改变对平面镜场的光线照射比例的影响程度，我们使最优布置策略的每组镜面长度改变量在$[-0.2,\ 0.2]$范围，以0.1为步长进行调整，并计算光线照射比例的取值。

不同镜面长度改变量下光线照射比例随角度变化如图 26 所示：

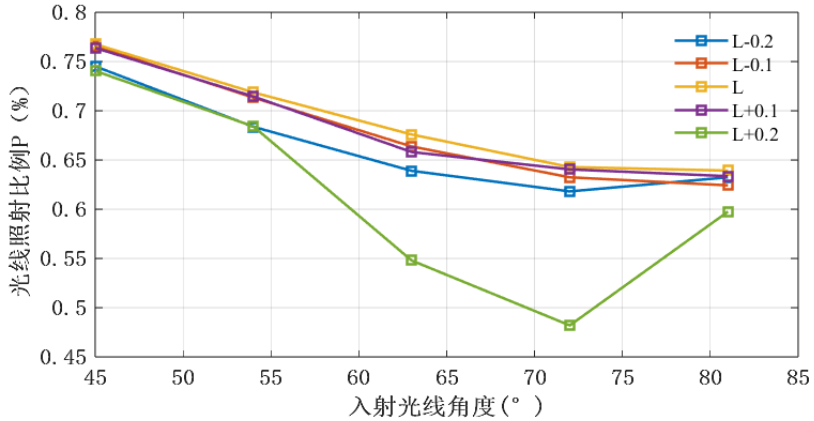

图 26 不同镜面长度改变量下光线照射比例随角度变化图

从图 26 可以看出，当镜面长度在小范围内波动时，对最终的光线照射比例影响较小，反映了该布置策略的鲁棒性较高，对实际情况中镜面长度的精度要求较低。而当镜面长度的改变量$\Delta L = 0.2$时，在部分角度下，光线照射比例会有显著下降，可能原因是在该种情况下，镜面长度的增加会产生更多遮挡。

七、 模型评价与推广

7.1 模型的优点

- 在曲线EF的设计上，在保证题目要求条件下，综合考虑了客观事实，使模型具有现实意义与使用价值；
- 使用光线追踪方法计算最终光线照射比例，总共模拟 2 万条光线，有效提高了计算精度，并有利于后续研究；
- 模型利用分组步进式收敛最优解算法，提高了优化效率以及优化精度。模型运行速度快，精度较高，且经过检验符合客观事实。

7.2 模型的缺点

- 在光线模拟过程中，未考虑光线的大气衰减，光线吸收等效率影响，仅考虑理想状态下的光线轨迹；
- 为加快计算速度，模型在进行分区域规划时仅考虑 8 组情况，且在步进优化时仅考虑相邻两组之间的影响，可能产生一定误差。

7.3 模型的推广

- 在算力允许情况下，增加分组数量，考察不相邻组之间的影响，适当增加光线模拟条数，以提高结果精度；
- 考虑更多客观损失，如大气衰减，余弦效率等，使模型更具有现实意义，提高模型精度；
- 推广到合适的三维模型，使模型更具有实际价值和普适性。

参考文献

[1] 宋永兴,姜希彤,王君,等. 新型太阳能集热装置的二次反射器研究[J]. 热力发电,2015(1):49-53.

[2] 沈焕波,付杰,李心,等. 基于二次反射的光学系统聚光特性研究[J]. 太阳能,2015(1):34-38.

[3] Akiba Segal，Michael Epstein. The optics of the solar tower reflector[J]. Solar Energy，2000，69: 229 － 241.

[4] 张海洪,樊仲维,郝沛明.两镜系统分析[J].量子电子学报,2003,(3): 19–23.

[5] 郑世旺.旋转双曲面成像问题再研究[J].河南科学,2003,(3): 17－20.

[6] 徐任学.抛物面反射镜对倾斜入射光的聚光作用[J].太阳能学报,1985(02):179-193.

[7] 程金铭.抛物线的一般反射规律[J].科技创业家,2012,(15):23+43.

附录

附录一 支撑材料列表

工作簿：不同角度分组最优排布方式.xlsx

程序列表：

程序 0 EF.m

程序 1 RayCross.m

程序 2 RayMirrorReflection.m

程序 3 checkIntersection.m

程序 4 FindIntersectionAndReflection.m

程序 5 Func.m

程序 6 optim.m

程序 7 Func2.m

程序 8 optim2.m

程序 9 Func3.m

程序 10 optim3.m

程序 11 RayMirrorReflection3.m

程序 12 temp.m

附录二 程序代码

Code 0 确定 EF *EF.m*

```
clc,clear
k_rm=zeros(3,8);
tic
for k=2:1:9
f1 = 100/k;
f2 = 100;
L=2;

x_b = 200;
theta = (1/2)*(atan(x_b/(f1+f2)));
k1 = -1/(tan(2*theta));
a = ((k-1)/2)*f1;
c = (f2-a);
b = sqrt(c*c-a*a);
m = f2-a;
A=k1^2-(a/b)^2;
B=-2*k1*(k1*x_b+m);
C=(k1*x_b+m)^2-a^2;
delta=B^2-4*A*C;
```

```
x=(-sqrt(delta)-B)/(2*A);
y=k1*(x-x_b);
A1=A;
B1=-2*k1*(k1*x_b+k1*L*cos(theta)-L*sin(theta)+m);
C1=(k1*x_b+k1*L*cos(theta)-L*sin(theta)+m)^2-a^2;
delta1=B1^2-4*A1*C1;
x1=(-sqrt(delta1)-B1)/(2*A1);%max radius
y1=k1*(x1-x_b-L*cos(theta))+L*sin(theta);
f=(x1/b^2)*a^2/(y1-m);
k12=-1/f;
n=[1/sqrt(1+k12^2) k12/sqrt(1+k12^2)];
in=[1/sqrt(1+k1^2) k1/sqrt(1+k1^2)];
out=in-2*(sum(in.*n))*n;
k2=out(2)/out(1);
x2=x1-y1/k2;
k_rm(1,k-1)=k;
k_rm(2,k-1)=x1;
k_rm(3,k-1)=x2/L;
end

figure;
yyaxis left;
plot(k_rm(1,:), k_rm(2,:), 'r-', 'DisplayName', 'Y1');
ylabel('半径');
yyaxis right;
plot(k_rm(1,:), k_rm(3,:), 'b--', 'DisplayName', 'Y2');
ylabel('放大率');
xlabel('f_2/f_1');
legend('Y1', 'Y2');
grid on;
toc
```

Code 1 判断光线是否与平面镜相交 ***RayCross.m***

```
function hasIntersection = RayCross(A, d, B, C)
    % A: 点 A 的坐标，2 维向量 [Ax, Ay]
    % d: 光线的方向向量，2 维向量 [dx, dy]
    % B, C: 线段 BC 的端点坐标，每个点是一个 2 维向量 [Bx, By], [Cx, Cy]

    % 计算向量 AB 和 AC
    AB = B - A;
    AC = C - A;
```

```
    % 计算叉乘
    cross_AB_d = AB(1) * d(2) - AB(2) * d(1);
    cross_AC_d = AC(1) * d(2) - AC(2) * d(1);

    % 如果两个叉乘结果符号相同，说明光线不与线段 BC 相交
    if sign(cross_AB_d) == sign(cross_AC_d)
        hasIntersection = false;
        return;
    end

    % 计算向量 BC 和 BA
    BC = C - B;
    BA = A - B;

    % 计算叉乘
    cross_BC_BA = BC(1) * BA(2) - BC(2) * BA(1);
    cross_BC_d = BC(1) * d(2) - BC(2) * d(1);

    % 如果两个叉乘结果符号相同，说明光线不与线段 BC 相交
    if sign(cross_BC_BA) == sign(cross_BC_d)
        hasIntersection = false;
        return;
    end

    % 光线与线段 BC 相交
    hasIntersection = true;
end
```

Code 2 计算光线与平面镜的交点，与反射光线

RayMirrorReflection.m

```
function reflection = RayMirrorReflection(A, d, B, C)
    % A: 点 A 的坐标，2 维向量 [Ax, Ay]
    % d: 光线的方向向量，2 维向量 [dx, dy]
    % B, C: 平面镜的两个端点坐标，每个点是一个 2 维向量 [Bx, By], [Cx, Cy]

    % 初始化返回值
    reflection = [0, 0, 0, 0];

    % 计算线段 BC 的方向向量
    v_BC = C - B;
```

```
    % 计算法向量（垂直于BC）
    normal = [v_BC(2), -v_BC(1)];
    normal = normal / norm(normal);  % 单位化法向量

    % 计算光线与直线BC的交点
    t = dot(B - A, normal) / dot(d, normal);

    % 判断光线是否与直线BC相交
    if t < 0
        return;  % 光线与直线BC不相交
    end

    % 计算交点P
    P = A + t * d;

    % 判断交点P是否在线段BC上
    s = dot(P - B, v_BC) / dot(v_BC, v_BC);
    if s < 0 || s > 1
        return;  % 交点不在线段BC上
    end

    % 计算反射方向向量
    d_reflected = d - 2 * dot(d, normal) * normal;

    % 返回反射点P和反射方向向量d_reflected
    reflection = [P, d_reflected];
end
```

Code 3 判断反射光线是否与 **EF** 相交 ***checkIntersection.m***

```
function isIntersect = checkIntersection(In, xk)
    a = 37.5;
    b = 50;
    m = 62.5;
    f1 = 25;
    f2 = 100;
    k1 = -(f1+f2)/xk;
    A = k1^2 - a^2/b^2;
    B = -2*k1*(k1*In(1) + m - In(2));
    C = (In(2) - m - k1*In(1))^2 - a^2;
    x = (-B-sqrt(B^2-4*A*C))/(2*A);
    if abs(x)  <= 32.0392
        isIntersect = true;
```

```
    else
        isIntersect = false;
    end
end
```

Code 4 判断光线与经过 EF 反射后是否落入 CD

FindIntersectionAndReflection.m

```
function isin = FindIntersectionAndReflection(In, xk)
a = 37.5;
b = 50;
m = 62.5;
f1 = 25;
f2 = 100;
k1 = -(f1+f2)/xk;
A = k1^2 - a^2/b^2;
B = -2*k1*(k1*In(1) + m - In(2));
C = (In(2) - m - k1*In(1))^2 - a^2;
x = (-B-sqrt(B^2-4*A*C))/(2*A);
if abs(x)  > 32.0392
    return
end
y = k1*(x - In(1))+In(2);
fp = x*a^2/(b^2 * (y-m));
k = -1/fp;
n = [1/sqrt(k^2+1), k/sqrt(k^2+1)];
a = [1/sqrt(k1^2+1), k1/sqrt(k1^2+1)];
b = a - 2*sum(a.*n)*n;
k2 = b(2)/b(1);
if ((x-y/k2) >= -5) && ((x-y/k2) <= 5)
    isin = true;
else
    isin = false;
end
% disp([x,y,n,k2,x-y/k2, a, b])
end
```

Code 5 第一问主函数 ***Func.m***

```
function pers = Func(x_left, l)
% 参数初始化
f1 = 25;
f2 = 100;
tot_ray = 20000;
t = 1;
theta = 0.5*atan(x_left/(f1+f2));
x_right = x_left + l.*cos(theta);
y_right = l.*sin(theta);
cnt = 0;

% 光线模拟
for i = 32:200/tot_ray:200
    if ~RayCross([i, 300], [0, -1], [x_left(t), 0], [x_right(t), y_ri
ght(t)])
        if ~RayCross([i, 300], [0, -1], [x_left(t+1), 0], [x_right(t+
1), y_right(t+1)])
            continue;
        end
        reflection = RayMirrorReflection([i, 300], [0, -1], [x_left(t
+1), 0], [x_right(t+1), y_right(t+1)]);
        t = t + 1;
        if RayCross(reflection(1:2), reflection(3:4), [x_left(t), 0],
 [x_right(t), y_right(t)])
            continue;
        end
        if ~checkIntersection(reflection(1:2),x_left(t))
            continue;
        end
        if findIntersectionAndReflection(reflection(1:2),x_left(t))
            cnt = cnt+1;
        end
    end
    reflection = RayMirrorReflection([i, 300], [0, -1], [x_left(t),
0], [x_right(t), y_right(t)]);
    if t > 1
        if RayCross(reflection(1:2), reflection(3:4), [x_left(t-1),
0], [x_right(t-1), y_right(t-1)])
            continue;
        end
    end
    if ~checkIntersection(reflection(1:2), x_left(t))
```

```
        continue;
    end
    FFF = FindIntersectionAndReflection(reflection(1:2),x_left(t));
    if FFF
        cnt = cnt+1;
    end
end
pers = cnt/tot_ray;
end


function isin = FindIntersectionAndReflection(In, xk)
a = 37.5;
b = 50;
m = 62.5;
f1 = 25;
f2 = 100;
k1 = -(f1+f2)/xk;
A = k1^2 - a^2/b^2;
B = -2*k1*(k1*In(1) + m - In(2));
C = (In(2) - m - k1*In(1))^2 - a^2;
x = (-B-sqrt(B^2-4*A*C))/(2*A);
if abs(x)  > 32.0392
    return
end
y = k1*(x - In(1))+In(2);
fp = x*a^2/(b^2 * (y-m));
k = -1/fp;
n = [1/sqrt(k^2+1), k/sqrt(k^2+1)];
a = [1/sqrt(k1^2+1), k1/sqrt(k1^2+1)];
b = a - 2*sum(a.*n)*n;
k2 = b(2)/b(1);
if ((x-y/k2) >= -5) && ((x-y/k2) <= 5)
    isin = true;
else
    isin = false;
end
% disp([x,y,n,k2,x-y/k2, a, b])
end
```

Code 6 第一问优化代码 ***optim.m***

```
%%
clear;
clc;

team = 8;
X = linspace(32,200,team+1);
X = [(X(1:team));(X(2:team+1));10*ones(1,team);ones(1,team);zeros(1,
team)];
X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X(3, 2:team-1));
x_r = [];
x_r = [x_r, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X(3,1))];
for i = 2:team-1
    x_r = [x_r, linspace(X(1,i) + 0.5*X(5,i), X(2,i) - 0.5*X(5,i), X
(3,i))];
end
x_r = [x_r, linspace(X(1,team) + 0.5*X(5,team), X(2,team), X(3,tea
m))];
%%
curr = 0;
bestresult = 0;
bestx = 10*ones(1,team);
bestl = 0.6*ones(1,team);
besti = 0;
bestj = 0;
bestII = [35,35,35,40,35,40,30,30];
bestJJ = [1.2, 1.2, 0.6, 1.2, 1.5, 0.9, 1.5, 1.5];
bestI = zeros(1,team);
bestJ = zeros(1,team);

for t = 1:team-1
    for i1 = bestII(t)-3:1:bestII(t)
        for i2 = bestII(t+1)-3:1:bestII(t+1)+3
            for j1 = bestJJ(t)-0.2:0.08:bestJJ(t)+0.2
                for j2 = bestJJ(t+1)-0.2:0.08:bestJJ(t+1)+0.2
                    team = 8;
                    X = linspace(32,200,team+1);
                    X = [(X(1:team));(X(2:team+1));bestx;bestl;zeros
(1,team)];
                    X(3,t) = i1;
                    X(3,t+1) = i2;
```

```
                    X(4,t) = j1;
                    X(4,t+1) = j2;
                    X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
                    X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
                    X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X
(3, 2:team-1));
                    x_r = [];
                    x_r = [x_r, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X
(3,1))];
                    for i = 2:team-1
                        x_r = [x_r, linspace(X(1,i) + 0.5*X(5,i), X(2,
i) - 0.5*X(5,i), X(3,i))];
                    end
                    x_r = [x_r, linspace(X(1,team) + 0.5*X(5,team), X
(2,team), X(3,team))];
                    l = [];
                    for i = 1:team
                        l = [l, X(4,i)*ones(1,X(3,i))];
                    end
                    curr = Func(x_r,l);
                    if curr > bestresult
                        bestresult = curr;
                        bestx = X(3, :);
                        bestl = X(4, :);
                        besti = [i1,i2];
                        bestj = [j1,j2];
                    end
                end
            end
        end
    end
    bestI(t) = besti(1);
    bestJ(t) = bestj(1);
    disp([besti, bestj,bestresult]);
end
bestI(t+1) = besti(2);
bestJ(t+1) = bestj(2);
%%
x = [];
x = [x, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X(3,1))];
for i = 2:team-1
    x = [x, linspace(X(1,i) + 0.5*X(5,i), X(2,i) - 0.5*X(5,i), X(3,
i))];
end
```

```
x = [x, linspace(X(1,team) + 0.5*X(5,team), X(2,team), X(3,team))];
l = [];
for i = 1:team
    l = [l, X(4,i)*ones(1,X(3,i))];
end
f1 = 25;
f2 = 100;
tot_ray = 8000;
t = 1;
theta = 0.5*atan(x/(f1+f2));
x_r = x + l.*cos(theta);
y_right = l.*sin(theta);
hold on
scatter(x(200:1:300),0, 'red');
scatter(x_r(200:1:300), y_right(200:1:300), 'blue')
for i = 200:1:300
    plot([[x(i), 0], [x_r(i), y_right(i)]])
end
```

Code 7 第二问主函数 *Func2.m*

```
function [pers, tot_ray] = Func2(x_left, l, Alpha)
% 参数初始化
f1 = 25;
f2 = 100;
tot_ray = 10000;
t = 1;
beta = atan(x_left/(f1+f2));
theta = 0.5*(Alpha + beta)-pi/4;
Key_index = 0;
for i = 1:length(x_left)-1
    if theta(i)*theta(i+1) < 0
        Key_index = i;
        break;
    end
end
theta(1:Key_index) = theta(1:Key_index) + pi;
x_right = x_left + l.*cos(theta);
y_right = l.*sin(theta);
cnt = 0;

% 光线模拟
for i = 0:200/tot_ray:200
```

```
    % if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
    %     continue;
    % end
    if t > Key_index
        if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin(Al
pha)], [x_left(t), 0], [x_right(t), y_right(t)])
            if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin
(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
            reflection = RayMirrorReflection([i + 110/tan(Alpha), 11
0], [-cos(Alpha), -sin(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_ri
ght(t+1)]);
            t = t + 1;
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t),
0], [x_right(t), y_right(t)])
                continue;
            end
            if ~checkIntersection(reflection(1:2),x_left(t))
                continue;
            end
            if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
                continue;
            end
            if findIntersectionAndReflection(reflection(1:2),x_left
(t))
                cnt = cnt+1;
            end
        end
        reflection = RayMirrorReflection([i + 110/tan(Alpha), 110],
[-cos(Alpha), -sin(Alpha)], [x_left(t), 0], [x_right(t), y_right
(t)]);
        if t > 1
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t-
1), 0], [x_right(t-1), y_right(t-1)])
                continue;
            end
        end
        if ~checkIntersection(reflection(1:2), x_left(t))
            continue;
        end
        if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
            continue;
        end
```

```
        FFF = FindIntersectionAndReflection(reflection(1:2),x_left
(t));
        if FFF
            cnt = cnt+1;
        end
    else
        if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin(Al
pha)], [x_left(t), 0], [x_right(t), y_right(t)])
            if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin
(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
            reflection = RayMirrorReflection([i + 110/tan(Alpha), 11
0], [-cos(Alpha), -sin(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_ri
ght(t+1)]);
            t = t + 1;
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t+
1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
            if ~checkIntersection(reflection(1:2),x_left(t))
                continue;
            end
            if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
                continue;
            end
            if findIntersectionAndReflection(reflection(1:2),x_left
(t))
                cnt = cnt+1;
            end
            continue
        end
        if RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin(Alp
ha)], [x_left(t), 0], [x_right(t), y_right(t)])
            if RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin
(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_right(t+1)])
                reflection = RayMirrorReflection([i + 110/tan(Alpha),
110], [-cos(Alpha), -sin(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_
right(t+1)]);
                t = t + 1;
                if RayCross(reflection(1:2), reflection(3:4), [x_left
(t+1), 0], [x_right(t+1), y_right(t+1)])
                    continue;
                end
```

```
                if ~checkIntersection(reflection(1:2),x_left(t))
                    continue;
                end
                if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) <
32)
                    continue;
                end
                if findIntersectionAndReflection(reflection(1:2),x_lef
t(t))
                    cnt = cnt+1;
                end
                continue
            end
        end
        reflection = RayMirrorReflection([i + 110/tan(Alpha), 110],
[-cos(Alpha), -sin(Alpha)], [x_left(t), 0], [x_right(t), y_right
(t)]);
        if t > 1
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t+
1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
        end
        if ~checkIntersection(reflection(1:2), x_left(t))
            continue;
        end
        if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
            continue;
        end
        FFF = FindIntersectionAndReflection(reflection(1:2),x_left
(t));
        if FFF
            cnt = cnt+1;
        end
    end
end
pers = cnt/tot_ray;

end
```

Code 8 第二问优化代码 ***optim2.m***

```
clear;
clc;

team = 8;
X = linspace(5,200,team+1);
X = [(X(1:team));(X(2:team+1));20*ones(1,team);ones(1,team);zeros(1,
team)];
X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X(3, 2:team-1));
x_r = [];
x_r = [x_r, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X(3,1))];
for i = 2:team-1
    x_r = [x_r, linspace(X(1,i) + 0.5*X(5,i), X(2,i) - 0.5*X(5,i), X
(3,i))];
end
x_r = [x_r, linspace(X(1,team) + 0.5*X(5,team), X(2,team), X(3,tea
m))];

curr = 0;
bestresult = 0;
bestx = 10*ones(1,team);
bestl = 0.6*ones(1,team);
besti = 0;
bestj = 0;
Alpha = 135*pi/180;
bestII = [20  20  20  20  10  30  35  35];
bestJJ = [1.2 0.9 0.9 0.9 0.6 0.6 0.6 1.5];
bestI = zeros(1,team);
bestJ = zeros(1,team);

for t = 1:team-1
    for i1 = bestII(t)-3:1:bestII(t)+3
        for i2 = bestII(t+1)-3:1:bestII(t+1)+3
            for j1 = bestJJ(t)-0.2:0.08:bestJJ(t)+0.2
                for j2 = bestJJ(t+1)-0.2:0.08:bestJJ(t+1)+0.2
                    team = 8;
                    X = linspace(5,200,team+1);
                    X = [(X(1:team));(X(2:team+1));bestx;bestl;zeros
(1,team)];
                    X(3,t) = i1;
                    X(3,t+1) = i2;
```

```
                    X(4,t) = j1;
                    X(4,t+1) = j2;
                    X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
                    X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
                    X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X
(3, 2:team-1));
                    x_r = [];
                    x_r = [x_r, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X
(3,1))];
                    for i = 2:team-1
                        x_r = [x_r, linspace(X(1,i) + 0.5*X(5,i), X(2,
i) - 0.5*X(5,i), X(3,i))];
                    end
                    x_r = [x_r, linspace(X(1,team) + 0.5*X(5,team), X
(2,team), X(3,team))];
                    l = [];
                    for i = 1:team
                        l = [l, X(4,i)*ones(1,X(3,i))];
                    end
                    [curr, ~] = Func2(x_r,l, Alpha);
                    if curr > bestresult
                        bestresult = curr;
                        bestx = X(3, :);
                        bestl = X(4, :);
                        besti = [i1,i2];
                        bestj = [j1,j2];
                    end
                end
            end
        end
    end
    bestI(t) = besti(1);
    bestJ(t) = bestj(1);
    disp([besti, bestj,bestresult]);
end
bestI(t+1) = besti(2);
bestJ(t+1) = bestj(2);
```

Code 9 第三问主函数 *Func3.m*

```
function [pers, tot_ray] = Func3(x_left, l, Alpha, k)
% 参数初始化
f1 = 25;
f2 = 100;
```

```
tot_ray = 10000;
t = 1;
beta = atan(x_left/(f1+f2));
theta = 0.5*(Alpha + beta)-pi/4;
Key_index = 0;
for i = 1:length(x_left)-1
    if theta(i)*theta(i+1) < 0
        Key_index = i;
        break;
    end
end
theta(1:Key_index) = theta(1:Key_index) + pi;
x_right = x_left + l.*cos(theta);
y_right = l.*sin(theta);
cnt = 0;

% 光线模拟
for i = 0:200/tot_ray:200
    % if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
    %     continue;
    % end
    if t > Key_index
        if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin(Al
pha)], [x_left(t), 0], [x_right(t), y_right(t)])
            if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin
(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
            reflection = RayMirrorReflection3([i + 110/tan(Alpha), 11
0], [-cos(Alpha), -sin(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_ri
ght(t+1)], x_left(t+1), k(t+1));
            t = t + 1;
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t),
0], [x_right(t), y_right(t)])
                continue;
            end
            if ~checkIntersection(reflection(1:2),x_left(t))
                continue;
            end
            if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)

                continue;
            end
            if FindIntersectionAndReflection(reflection(1:2),x_left
```

```
(t))
                cnt = cnt+1;
            end
        end
        reflection = RayMirrorReflection3([i + 110/tan(Alpha), 110],
[-cos(Alpha), -sin(Alpha)], [x_left(t), 0], [x_right(t), y_right(t)],
 x_left(t), k(t));
        if t > 1
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t-
1), 0], [x_right(t-1), y_right(t-1)])
                continue;
            end
        end
        if ~checkIntersection(reflection(1:2), x_left(t))
            continue;
        end
        if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
            continue;
        end
        FFF = FindIntersectionAndReflection(reflection(1:2),x_left
(t));
        if FFF
            cnt = cnt+1;
        end
    else
        if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin(Al
pha)], [x_left(t), 0], [x_right(t), y_right(t)])
            if ~RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin
(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
            reflection = RayMirrorReflection3([i + 110/tan(Alpha), 11
0], [-cos(Alpha), -sin(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_ri
ght(t+1)], x_left(t+1), k(t+1));
            t = t + 1;
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t+
1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
            if ~checkIntersection(reflection(1:2),x_left(t))
                continue;
            end
            if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
```

```
                continue;
            end
            if FindIntersectionAndReflection(reflection(1:2),x_left
(t))
                cnt = cnt+1;
            end
            continue
        end
        if RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin(Alp
ha)], [x_left(t), 0], [x_right(t), y_right(t)])
            if RayCross([i + 110/tan(Alpha), 110], [-cos(Alpha), -sin
(Alpha)], [x_left(t+1), 0], [x_right(t+1), y_right(t+1)])
                reflection = RayMirrorReflection3([i + 110/tan(Alpha),
 110], [-cos(Alpha), -sin(Alpha)], [x_left(t+1), 0], [x_right(t+1), y
_right(t+1)], x_left(t+1), k(t+1));
                t = t + 1;
                if RayCross(reflection(1:2), reflection(3:4), [x_left
(t+1), 0], [x_right(t+1), y_right(t+1)])
                    continue;
                end
                if ~checkIntersection(reflection(1:2),x_left(t))
                    continue;
                end
                if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) <
32)
                    continue;
                end
                if FindIntersectionAndReflection(reflection(1:2),x_lef
t(t))
                    cnt = cnt+1;
                end
                continue
            end
        end
        reflection = RayMirrorReflection3([i + 110/tan(Alpha), 110],
[-cos(Alpha), -sin(Alpha)], [x_left(t), 0], [x_right(t), y_right(t)],
 x_left(t), k(t));
        if t > 1
            if RayCross(reflection(1:2), reflection(3:4), [x_left(t+
1), 0], [x_right(t+1), y_right(t+1)])
                continue;
            end
        end
        if ~checkIntersection(reflection(1:2), x_left(t))
```

```
            continue;
        end
        if (i + 110/tan(Alpha) > -32) && (i + 110/tan(Alpha) < 32)
            continue;
        end
        FFF = FindIntersectionAndReflection(reflection(1:2),x_left
(t));
        if FFF
            cnt = cnt+1;
        end
    end
end
KK = sum(k)/length(k);
pers = (cnt/tot_ray)*exp(-(KK-0.7)^2)/exp(-(1-0.7)^2);

end
```

Code 10 第三问优化代码 *optim3.m*

```
%%
clear;
clc;
tic

team = 8;
X = linspace(5,200,team+1);
X = [(X(1:team));(X(2:team+1));[17 17  15  17  33  19  20  37];[1.4   1.
24  1.46   1.38   1.32   1.9 1.76   2.28];zeros(1,team)];
X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X(3, 2:team-1));
x_r = [];
x_r = [x_r, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X(3,1))];
for i = 2:team-1
    x_r = [x_r, linspace(X(1,i) + 0.5*X(5,i), X(2,i) - 0.5*X(5,i), X
(3,i))];
end
x_r = [x_r, linspace(X(1,team) + 0.5*X(5,team), X(2,team), X(3,tea
m))];
l = [];
for i = 1:team
    l = [l, X(4,i)*ones(1,X(3,i))];
end
```

```
team = 8;
X1 = linspace(5,200,team+1);
X1 = [(X1(1:team));(X1(2:team+1));[17  22  22  22  7   29  37  38];[1.16
    1.1 1.1 1.1 0.4 0.72   0.64   1.54];zeros(1,team)];
X1(5,1) = (X1(2,1)-X1(1,1))/(0.5+X1(3,1));
X1(5,team) = (X1(2,team)-X1(1,team))/(0.5+X1(3,team));
X1(5,2:team-1) = (X1(2,2:team-1)-X1(1,2:team-1))./(X1(3, 2:team-1));
x_r1 = [];
x_r1 = [x_r1, linspace(X1(1,1), X1(2,1) - 0.5*X1(5,1), X1(3,1))];
for i = 2:team-1
    x_r1 = [x_r1, linspace(X1(1,i) + 0.5*X1(5,i), X1(2,i) - 0.5*X1(5,
i), X1(3,i))];
end
x_r1 = [x_r1, linspace(X1(1,team) + 0.5*X1(5,team), X1(2,team), X1(3,
team))];
l1 = [];
for i = 1:team
    l1 = [l1, X1(4,i)*ones(1,X1(3,i))];
end
%%
curr = 0;
bestresult = 0;
currk1 = 0.8*ones(1,team);
bestk1 = 0.8*ones(1,team);
currk2 = 0.8*ones(1,team);
bestk2 = 0.8*ones(1,team);
Alpha = 45*pi/180;


for t = 1:team
    for tt = 1:team
    for i = 0.8:0.05:1
        for j = 0.6:0.05:1
        currk1(t) = i;
        currk2(tt) = j;
        K1 = [];
        for m = 1:team
        K1 = [K1, currk1(m)*ones(1,X(3,m))];
        end
        K2 = [];
        for m = 1:team
        K2 = [K2, currk2(m)*ones(1,X1(3,m))];
        end
```

```
        curr = (Func3(x_r, l, Alpha, K1)+Func3(x_r1, l1, pi-Alpha, K
2))/2;
        if curr > bestresult
            bestk1 = currk1;
            bestk2 = currk2;
            bestresult = curr;
        end
        end
    end
    disp(bestresult)
    end
end
toc
```

Code 11 模型三反射角函数 *RayMirrorReflection3.m*

```
function reflection = RayMirrorReflection3(A, d, B, C, xk, k)
    % A: 点A的坐标，2维向量 [Ax, Ay]
    % d: 光线的方向向量，2维向量 [dx, dy]
    % B, C: 平面镜的两个端点坐标，每个点是一个2维向量 [Bx, By], [Cx, Cy]

    f1 = 25;
    f2 = 100;
    % 初始化返回值
    reflection = [0, 0, 0, 0];

    % 计算线段BC的方向向量
    v_BC = C - B;

    % 计算法向量（垂直于BC）
    normal = [v_BC(2), -v_BC(1)];
    normal = normal / norm(normal);  % 单位化法向量

    % 计算光线与直线BC的交点
    t = dot(B - A, normal) / dot(d, normal);

    % 判断光线是否与直线BC相交
    if t < 0
        return;  % 光线与直线BC不相交
    end

    % 计算交点P
    P = A + t * d;
```

```
    % 判断交点 P 是否在线段 BC 上
    s = dot(P - B, v_BC) / dot(v_BC, v_BC);
    if s < 0 || s > 1
        return;  % 交点不在线段 BC 上
    end

    % 计算反射方向向量
    d_reflected = d - 2 * dot(d, normal) * normal;
    d_reflected(2) = d_reflected(2)-(1-k)*(P(1)-xk);
    % 返回反射点 P 和反射方向向量 d_reflected
    reflection = [P, d_reflected];
end
```

Code 12 综合求值函数 *temp.m*

```
function output = temp(x1,l1, x2, l2, alpha)
team = 8;
X = linspace(5,200,team+1);
X = [(X(1:team));(X(2:team+1));x1;l1;zeros(1,team)];
X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X(3, 2:team-1));
x_r1 = [];
x_r1 = [x_r1, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X(3,1))];
for i = 2:team-1
    x_r1 = [x_r1, linspace(X(1,i) + 0.5*X(5,i), X(2,i) - 0.5*X(5,i),
X(3,i))];
end
x_r1 = [x_r1, linspace(X(1,team) + 0.5*X(5,team), X(2,team), X(3,tea
m))];
l1 = [];
for i = 1:team
    l1 = [l1, X(4,i)*ones(1,X(3,i))];
end

X = linspace(5,200,team+1);
X = [(X(1:team));(X(2:team+1));x2;l2;zeros(1,team)];
X(5,1) = (X(2,1)-X(1,1))/(0.5+X(3,1));
X(5,team) = (X(2,team)-X(1,team))/(0.5+X(3,team));
X(5,2:team-1) = (X(2,2:team-1)-X(1,2:team-1))./(X(3, 2:team-1));
x_r2 = [];
x_r2 = [x_r2, linspace(X(1,1), X(2,1) - 0.5*X(5,1), X(3,1))];
```

```
for i = 2:team-1
    x_r2 = [x_r2, linspace(X(1,i) + 0.5*X(5,i), X(2,i) - 0.5*X(5,i),
X(3,i))];
end
x_r2 = [x_r2, linspace(X(1,team) + 0.5*X(5,team), X(2,team), X(3,tea
m))];
l2 = [];
for i = 1:team
    l2 = [l2, X(4,i)*ones(1,X(3,i))];
end
curr1 = 0;
curr2 = 0;
[curr1, ~] = Func2(x_r1,l1, alpha);
[curr2, ~] = Func2(x_r2, l2, pi-alpha);
output = (curr1+curr2)/2;
end
```