软件工程第一次上机作业

姓名： 薛睿

班级： 软工1506班

学号： U201517136

问题描述：

In a box bounded by [-1, 1], given m balloons (they cannot overlap) with variable radio r and position mu, find the optimal value of r and mu which maximize

sum r^2

算法思想：

主要思想是通过计算出所有的情况来寻找最优解。

由于圆心x、y坐标的精度只能取到小数点后两位，所以圆的位置是有限的。区域内圆心的位置有10000个，通过计算每个位置满足条件的最大半径r，从而找出这些点的最大半径；

在寻找最大半径时，每个位置的最大半径的精度只能取到小数点后五位。把满足条件的最大圆将放入一个链表中，这样一来之后的圆在进行位置判断时可以与已放入的圆进行比较，从而判断其是否满足条件；

代码：

#include <stdlib.h>

#include <stdio.h>

#include <math.h>

//气球结构体

typedef struct ball{

double x; //圆心x坐标

double y; //圆心y坐标

double r; //圆半径

}Ball;

//用链表来储存已经放置的气球

typedef struct ballList{

struct ballList \* next;

Ball ball;

}BallList;

void insert(Ball ball);

double distance(Ball b1, Ball b2);

int judge(Ball b);

void putBall();

BallList \*head = NULL;

double step = 0.01; //改变气球位置的最小单位

int num = 0; //放置气球的个数

double sumr = 0; //用来记录r^2之和

void main(void){

int n=0;

printf("输入气球总数量: ");

scanf\_s("%d", &n);

printf("\n球编号\t x坐标\t y坐标\t 半径\t r^2之和\n");

int i;

for (i = 0; i < n; i++){

putBall();

}

printf("\nr^2之和最大为:\t %lf\n", sumr);

system("pause");

}

void insert(Ball ball){ //插入气球

BallList \* newBall = (BallList \*)malloc(sizeof(BallList));

newBall->ball = ball;

newBall->next = head;

head = newBall;

}

void putBall(){ //改变气球的初始位置，求的满足条件的气球

Ball ball = { -1 + step, -1 + step, 0 };

Ball maxBall = ball;

int i, j;

for (i = 0; ball.x < 1; ++i){

ball.x += step;

ball.y = -1 + step;

for (j = 0; ball.y < 1; ++j){

ball.y += step;

ball.r = 0;

double rstep = 0.1;

while (rstep > 0.00001){

if (ball.r > maxBall.r){

maxBall = ball;

}

ball.r += rstep;

if (!judge(ball)){ //此气球不合适，减小半径与步长重新尝试

ball.r -= rstep;

rstep /= 10;

}

}

}

}

if (judge(maxBall)){

insert(maxBall);

num++;

sumr += maxBall.r \* maxBall.r;

printf("%d\t %.3lf\t %.3lf\t %.3lf\t %lf \n", num, maxBall.x, maxBall.y, maxBall.r, sumr);

}

}

int judge(Ball b){ //判断新加入的气球是否符合规则

//将气球限制在[-1，1]内

if ((fabs(b.x) + b.r) > 1 || (fabs(b.y) + b.r) > 1){

return 0;

}

//依次比较气球b与已有气球是否相交

BallList \*tmp = head;

while (tmp){

Ball ball = tmp->ball;

//两个气球相交判断

if (distance(b, ball) < b.r + ball.r){

return 0;

}

tmp = tmp->next;

}

return 1;

}

//判断气球球心之间的距离

double distance(Ball b1, Ball b2){

double x1 = b1.x;

double y1 = b1.y; double x2 = b2.x;

double y2 = b2.y;

return pow((x1 - x2) \* (x1 - x2) + (y1 - y2) \* (y1 - y2), 0.5);

}

运行结果：

输入气球总数量: 3

球编号 x坐标 y坐标 半径 r^2之和

1 0.000 0.000 1.000 0.999800

2 -0.830 -0.830 0.170 1.028700

3 -0.830 0.830 0.170 1.057566

r^2之和最大为: 1.057566

输入气球总数量: 4

球编号 x坐标 y坐标 半径 r^2之和

1 0.000 0.000 1.000 0.999800

2 -0.830 -0.830 0.170 1.028700

3 -0.830 0.830 0.170 1.057566

4 0.830 -0.830 0.170 1.086432

r^2之和最大为: 1.086432