**软件工程实验报告1**

**实验: project1**

**学号: U201517074**

**姓名: 王迪**

**班级: 软工1504班**

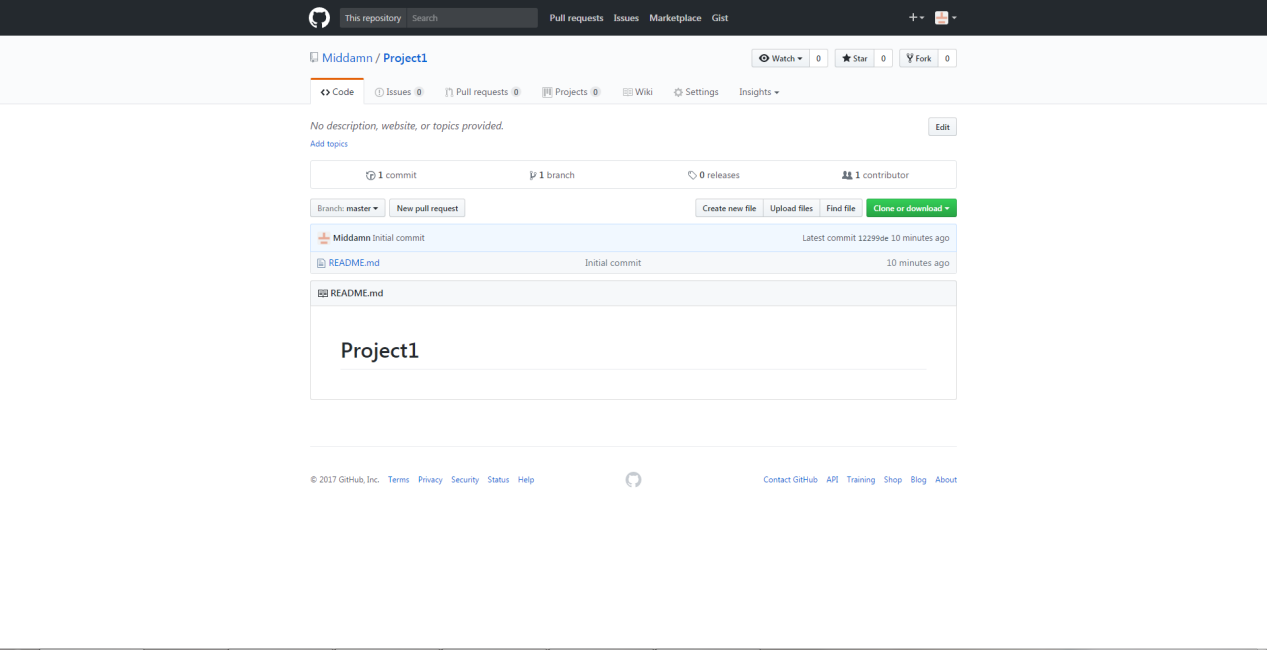
## 问题描述：

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

## Github

https://github.com/Middamn/Project1



## **算法**

将制定区域横纵分为若干份，以每个交点作为圆心，在制定圆的个数下通过枚举选取圆心，通过判断各个圆之间不相交、相容、出界后求取最大面积。

圆类：

class Circle

{

private float x;

private float y;

private float r;

public Circle(float \_x, float \_y, float \_r)

{

x = \_x;

y = \_y;

r = \_r;

}

public bool IsOut()

{

if (x + r > 2 || x - r < 0 || y + r > 2 || y - r < 0)

return true;

else return false;

}

public bool IsTrack(Circle otherCircle)

{

if (Math.Sqrt((x - otherCircle.x) \* (x - otherCircle.x) + (y - otherCircle.y) \* (y - otherCircle.y)) < r + otherCircle.r)

{

return true;

}

return false;

}

public float CaculateS()

{

return r \* r \* 3.14f;

}

}

}

## 测试用例

1).若只有一个圆，将变长为2的正方形横纵各分为20份。取每个节点作为圆心，半径由0.1，每次增加0.1，增加到1。每次存储计算出来的面积，存储最大值的面积。

代码：

namespace SoftWareCS

{

class Program

{

static float maxS = 0;

static int totalCircleNum = 0;

static void Main(string[] args)

{

for (float i = 0.1f; i < 2; i += 0.1f)

{

for (float j = 0.1f; j < 2; j += 0.1f)

{

for (float r = 0.1f; r <= 1.1f; r += 0.1f)

{

Console.WriteLine("i="+i+"j="+j+"r="+r);

Circle ciecle = new Circle(i, j, r);

if (!ciecle.IsOut())

{

Console.WriteLine(ciecle.CaculateS());

if (maxS < ciecle.CaculateS())

maxS = ciecle.CaculateS();

}

}

}

}

Console.WriteLine(maxS);

while(true)

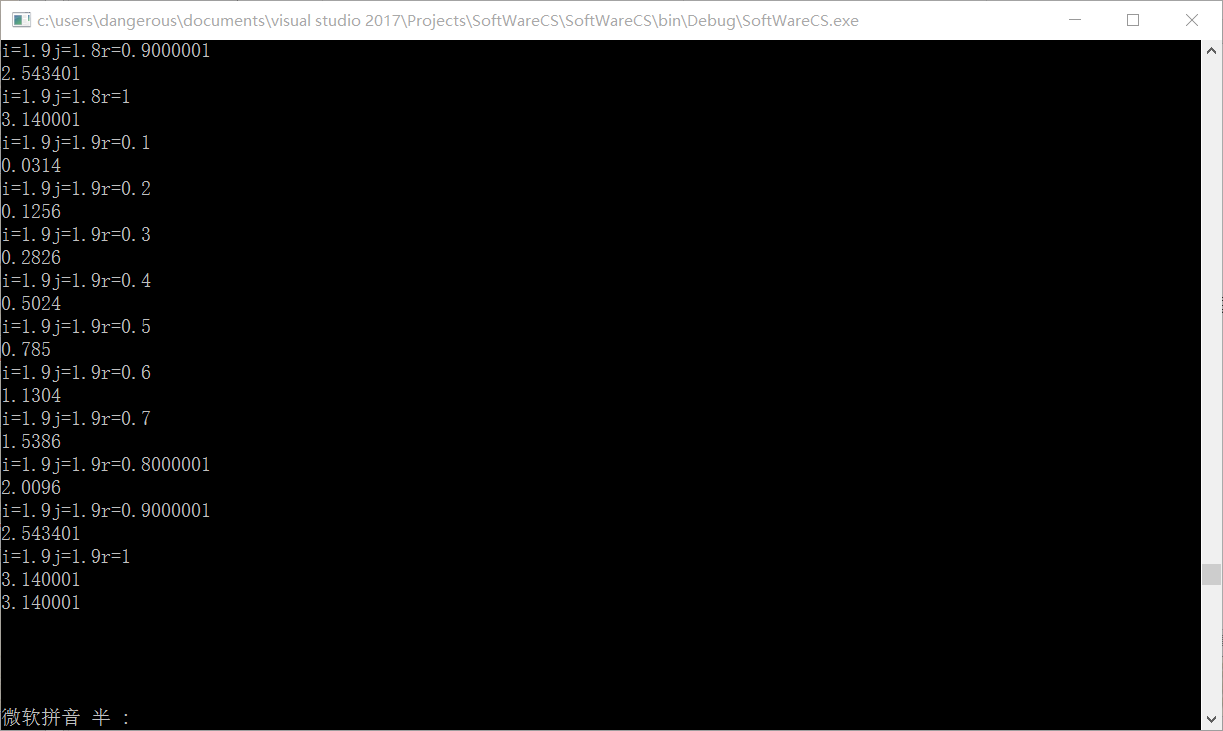
{

}

}

}

运行结果



最大值3.14。

2）若有两个或以上的圆，将区域横纵以0.1为跨度分区，以每个点为圆心，半径逐次增加，每次判断各个圆之间是否相交、出界，将符合条件的圆的位置和面积进行存储，求出最大值。

代码：

class Program

{

static float maxS = 0;

static void Main(string[] args)

{

Circle circle1 = new Circle();

Circle circle2 = new Circle();

Circle maxCiecle1 = new Circle();

Circle maxCiecle2 = new Circle();

for (circle1.x = 0.1f; circle1.x < 2; circle1.x += 0.1f)

for (circle1.y = 0.1f; circle2.y < 2; circle2.y += 0.1f)

for (circle1.r = 0.1f; circle1.r < 1.1; circle1.r += 0.1f)

{

for (circle2.x = 0.1f; circle2.x < 2; circle2.x += 0.1f)

for (circle2.y = 0.1f; circle2.y < 2; circle2.y += 0.1f)

for (circle2.r = 0.1f; circle2.r < 1.1; circle2.r += 0.1f)

{

if (circle1.IsOut() || circle2.IsOut() || circle1.IsTrack(circle2))

continue;

else

{

Console.WriteLine("ciecle1 x= " + circle1.x + "y= " + circle1.y + "r= " + circle1.r);

Console.WriteLine("ciecle2 x= " + circle2.x + "y= " + circle2.y + "r= " + circle2.r);

Console.WriteLine("maxs= " + maxS);

if (maxS < circle1.CaculateS() + circle2.CaculateS())

{

maxCiecle1 = new Circle(circle1.x, circle1.y, circle1.r);

maxCiecle2 = new Circle(circle2.x, circle2.y, circle2.r);

maxS = circle1.CaculateS() + circle2.CaculateS();

}

}

}

}

Console.WriteLine("Laster Result");

Console.WriteLine("ciecle1 x= " + maxCiecle1.x + "y= " + maxCiecle1.y + "r= " + maxCiecle1.r);

Console.WriteLine("ciecle2 x= " + maxCiecle2.x + "y= " + maxCiecle2.y + "r= " + maxCiecle2.r);

Console.WriteLine("maxS= "+maxS);

while(true)

{

}

}

}

运行结果：



## 总结

可以依次增加圆的个数，但暴力枚举效率太低。

## Gitlog

