[解题报告] POJ3301

[Source] http://poj.org/problem?id=3301

[Description]

给出平面上 n(n<=30)个点，要求用最小的正方形把这 n 个点圈住，求最小正方形的面积（正方形可以不平行于坐标轴，即任意方向）。

[Solution]

三分法。题意很简单，数据范围也很小，最多只有 30 个点，但是需要好的想法。想到如果正方形只能平行于坐标轴的话，题目就变得很简单了。然而正方形的方向是任意的，思路就是把坐标轴旋转，既进行坐标变换，对变换坐标后的点求平行坐标轴的最小正方形。于是正方形的最小边长成了坐标轴旋转角 a 的函数，目标是求该函数的最小值，三分法即可。
对于二维直角坐标系的坐标变换，有如下公式：
x'=x*cos(a)-y*sin(a);
y'=y*cos(a)+x*sin(a);
a 为坐标顺时针旋转的角度。

[Code]

```
#include<cstdio>
#include<cmath>
#include<algorithm>
#define eps 1e-8
#define oo 1e10
#define PI acos(-1.0)
#define N 50
using namespace std;

struct Point
{
    double x,y;
};

Point pt[N];
int n;

double calc(double a)
```

```
{
    double max_x,max_y,min_x,min_y,res;
    Point p;
    int i;

    max_x=max_y=-oo;
    min_x=min_y=oo;

    for (i=0;i<n;i++)
    {
        p.x=pt[i].x*cos(a)-pt[i].y*sin(a);
        p.y=pt[i].y*cos(a)+pt[i].x*sin(a);

        if (max_x<p.x)
            max_x=p.x;
        if (min_x>p.x)
            min_x=p.x;

        if (max_y<p.y)
            max_y=p.y;
        if (min_y>p.y)
            min_y=p.y;
    }

    res=max(max_x-min_x,max_y-min_y);

    return res;
}


int main()
{
    double ans,tmp,l,r,mid,mid2,mid_v,mid2_v;
    int t,i;

    scanf("%d",&t);
    while (t--)
    {
        scanf("%d",&n);
        for (i=0;i<n;i++)
            scanf("%lf%lf",&pt[i].x,&pt[i].y);

        l=0;
        r=PI;
```

```
        while (r-l>eps)
        {
            mid=(l+r)/2;
            mid2=(l+mid)/2;

            mid_v=calc(mid);
            mid2_v=calc(mid2);

            if (mid_v<mid2_v)
                l=mid2;
            else
                r=mid;
        }

        ans=calc(l);
        ans*=ans;

        printf("%.2f\n",ans);
    }

    return 0;
}
```