**目录**

[一． 界面设计 1](#_Toc138807255)

[二． 功能设计 7](#_Toc138807256)

[三． 程序设计 8](#_Toc138807257)

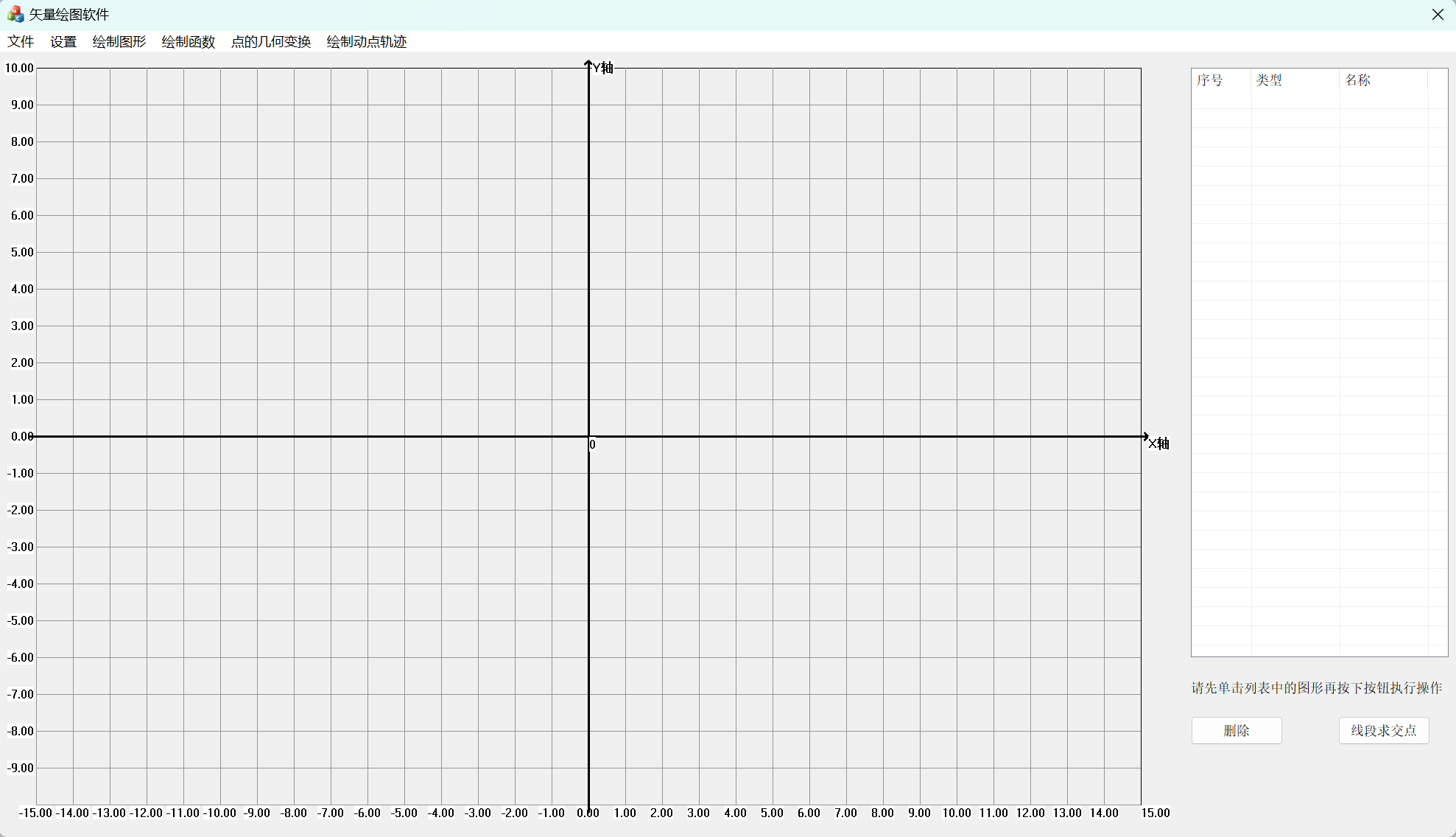
[四． 程序测试用例 19](#_Toc138807258)

[五． 总结 36](#_Toc138807259)

## 界面设计

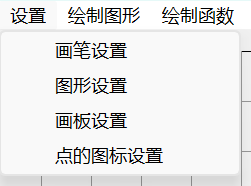
1. **主界面**

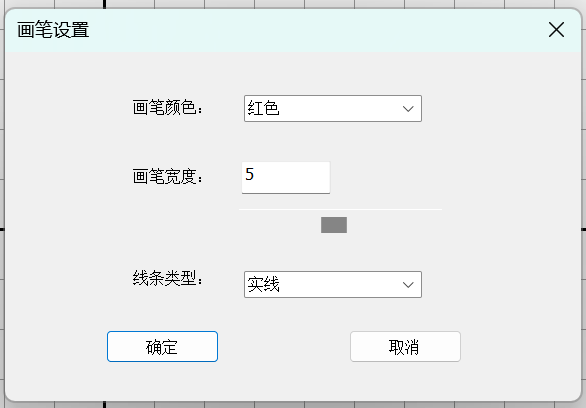
MFC程序采用对话框架构，主界面左侧为绘图区域，右侧为列表，用来显示当前绘制的图形/函数，底部的按钮可以对图形/函数进行操作，上侧有菜单，用于完成一系列绘图、设置操作



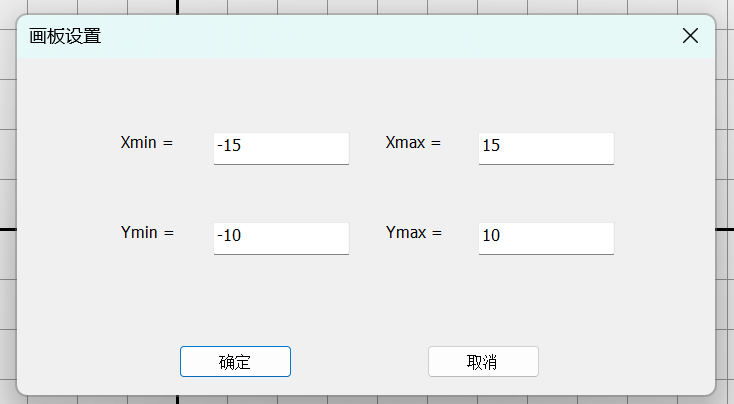
1. **设置界面**

设置界面由四个对话框组成，分别为画笔、图形、画板、点的图标设置。其中画笔、图形设置对话框用来设置当前绘图的颜色、线条粗细、线条样式等。画板设置对话框用来设置画板的X、Y的范围。点的图标设置对话框用来设置点的显示样式。





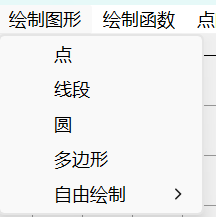


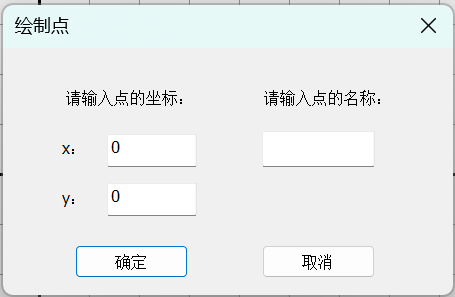


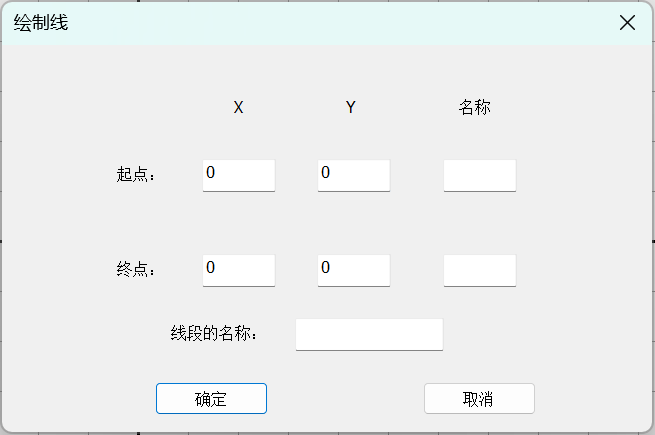


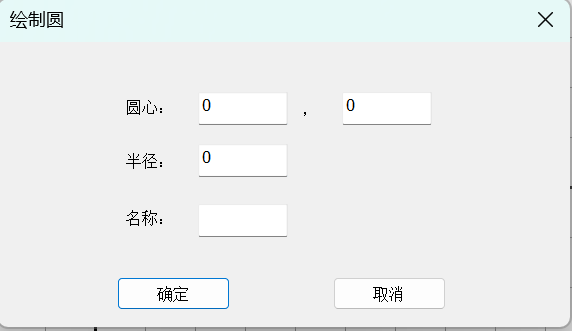
1. **图形绘制界面**

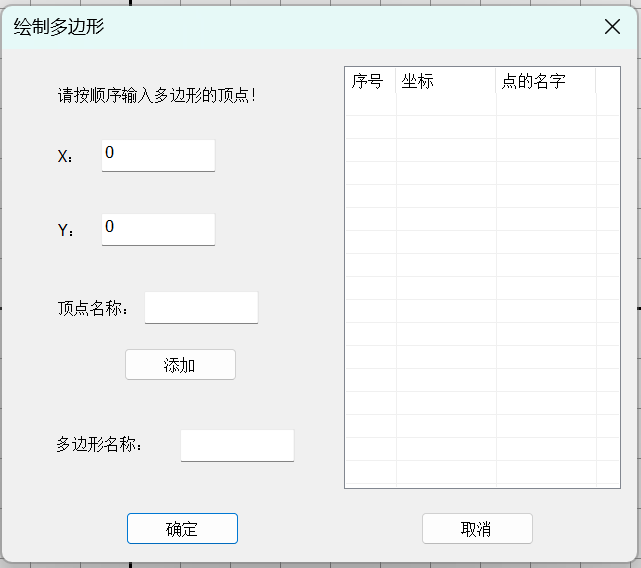
图形绘制界面由四个对话框组成，分别为点、线段、圆、多边形对话框。用户可根据对话框的提示绘制相应的图形。





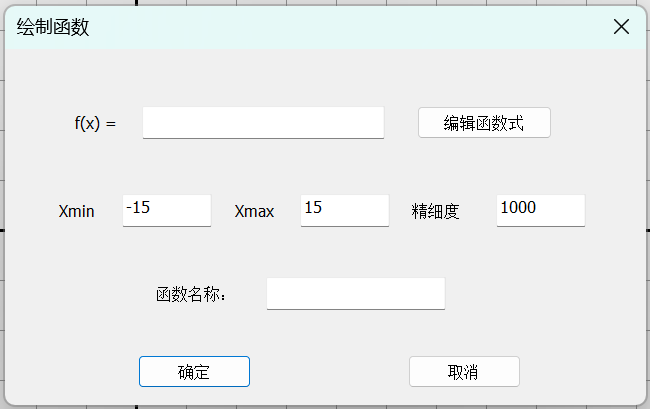






1. **函数绘制界面**

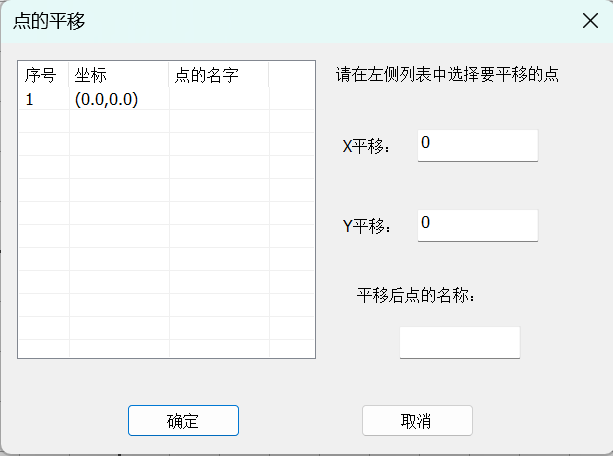
函数绘制界面由绘制函数对话框和编辑函数对话框组成，绘制函数对话框可以输入函数的表达式，设置XY的范围。编辑函数对话框可以使用户更方便地编辑函数

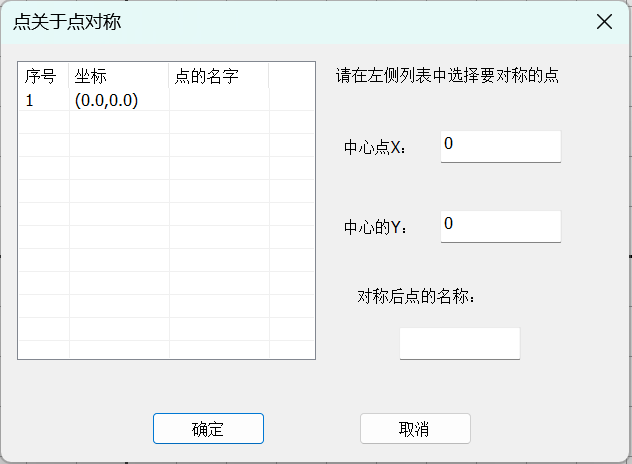
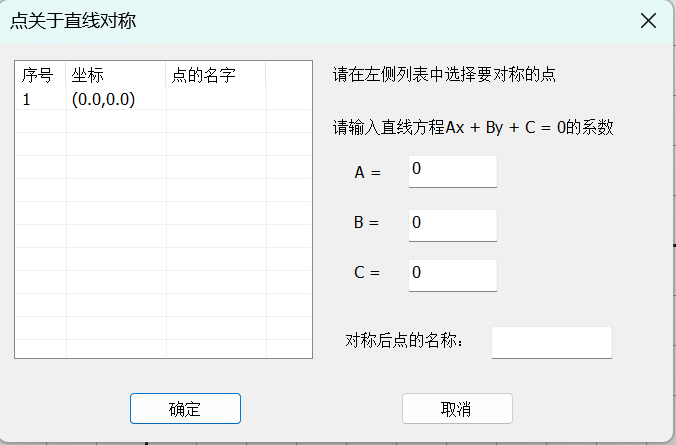
****



1. **点的几何变换界面**

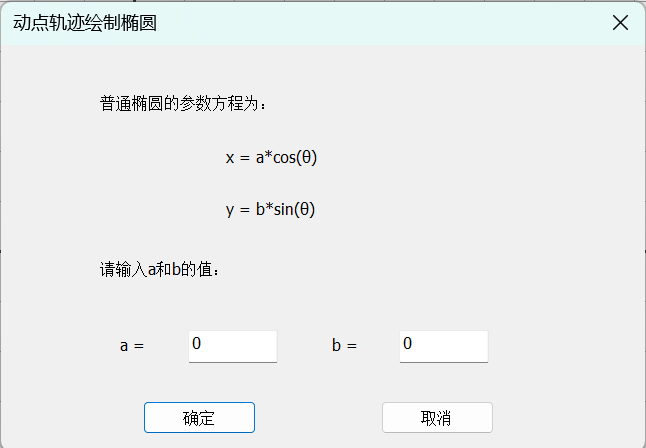
点的几何变换界面由点的平移、关于直线对称、关于点对称三个对话框组成，使用者根据提示输入信息即可计算出对称点

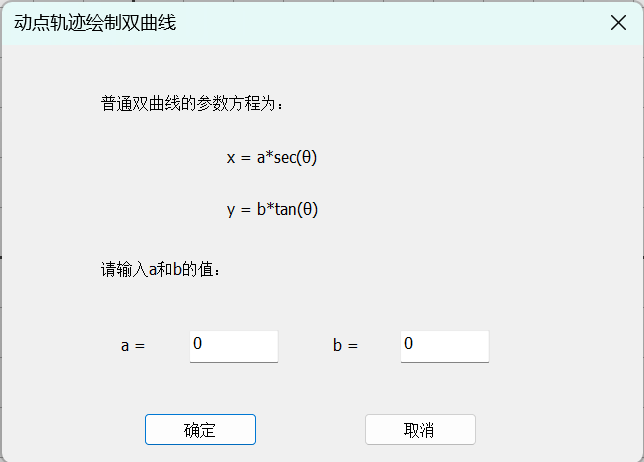


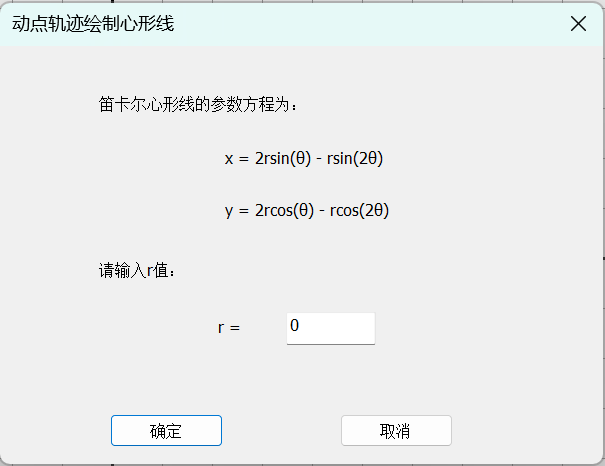


1. **动点绘制界面**

动点绘制界面由椭圆、双曲线、心形线绘制三个对话框自称，使用者可以根据提示信息输入参数方程的值完成动点的绘制







## 功能设计

1. **显示点的坐标**

用户双击绘图界面随便一个点可以显示他的坐标

1. **画笔设置**

可以设置画笔宽度，画笔颜色,轮廓颜色，填充颜色可以随意绘制曲线。画笔类型，点线、实线、虚线等。

1. **坐标网格绘制**
2. **点的绘制**
3. **线段绘制**
4. **多边形绘制**
5. **对象求交点**

线段可以与线段、图形之间求交点并显示

1. **点的几何变化**

包括点的平移、对称（关于点、线）

1. **方程的输入与曲线的生成**

可以绘制基本的函数，并显示曲线于绘图区域

1. **对象属性设置**

可以设置点的样式，提供五种样式

1. **动点轨迹绘制**

可以绘制椭圆、双曲线、笛卡尔心形线的轨迹

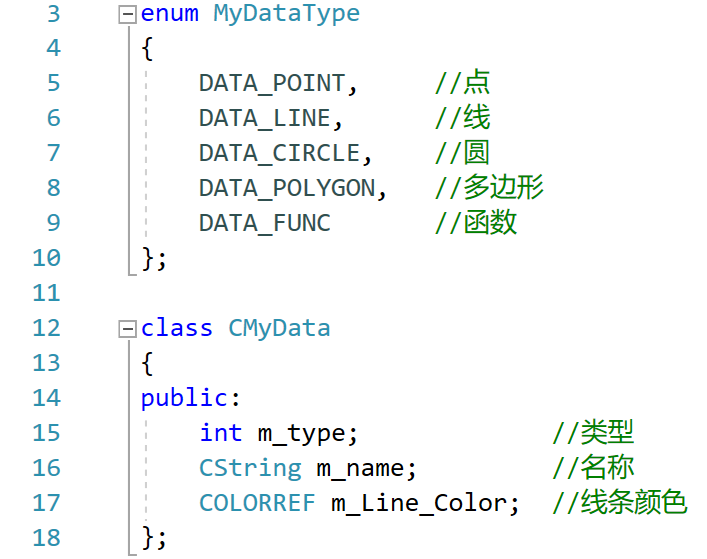
1. **导出图片**

可以将当前对话框的内容以图片形式导出

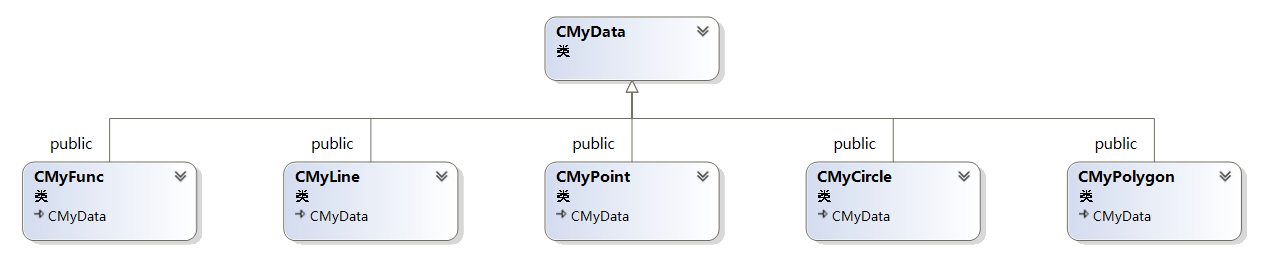
## 程序设计

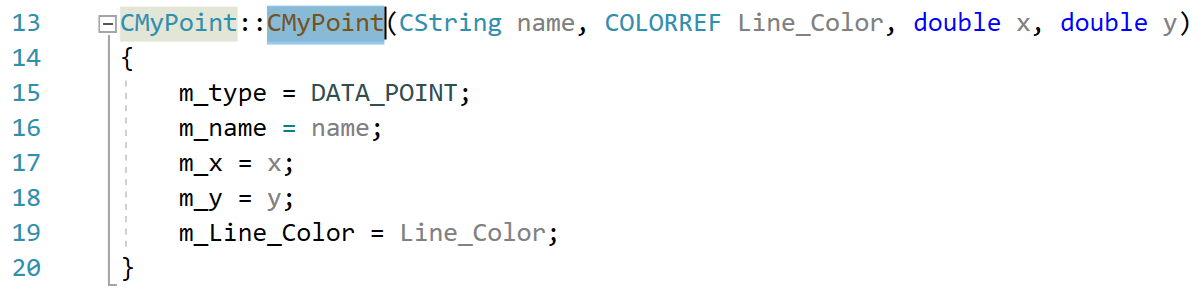
1. **数据结构设计**

根据第二章提到的绘图功能需求，本程序绘制的图形包括：点、线、圆、多边形、函数五大类。这里使用C++继承的特性，首先定义CMyData类作为父类，成员变量有类型、名称和颜色，这是所有图形类都具备的特征。然后声明枚举类来枚举图形的类型，将五种图形类型写入枚举类中。

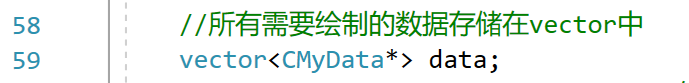


然后分别定义点、线、圆、多边形、函数五个类，它们继承CMyData父类，并在子类的构造函数中声明自己的类型，方便之后多态的判断。

****

****

定义完类之后，在View.h中定义动态数组，这里使用STL中的vector。（为什么不使用list、stack或者queue？因为vector可以根据索引方便地取出想要的元素，而其他数据结构则需要遍历。）这里使用C++多态的特性，在vector容器中储存CMyData类型的指针，这使得所有图形类型均可存入vector中，方便了数据的操作。

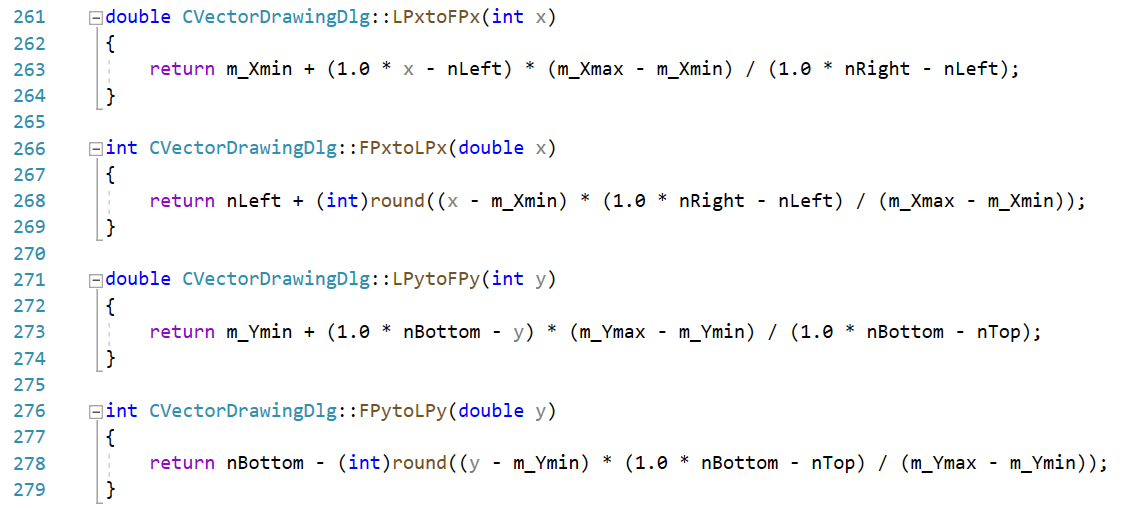
****

多态的好处体现在绘图中，在绘图时，依次遍历vector中的所有元素，根据它们的m\_type变量来判断数据的类型，分别执行对应的操作即可。

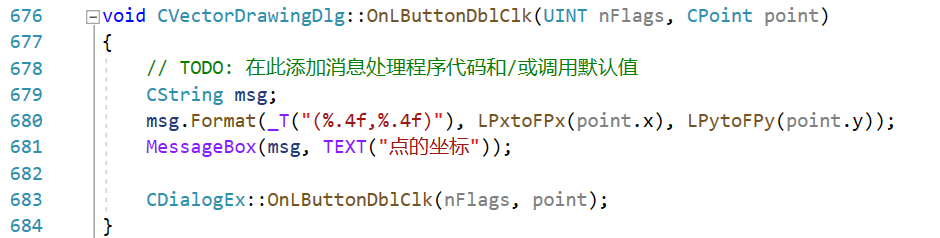
****

1. **显示点坐标的实现**

首先要定义一个映射函数，将绘图区域的坐标与对话框实际的坐标相互映射。



然后添加双击左键的消息，在消息处理函数中，根据传来的对话框坐标，使用映射函数转换为绘图坐标即可。

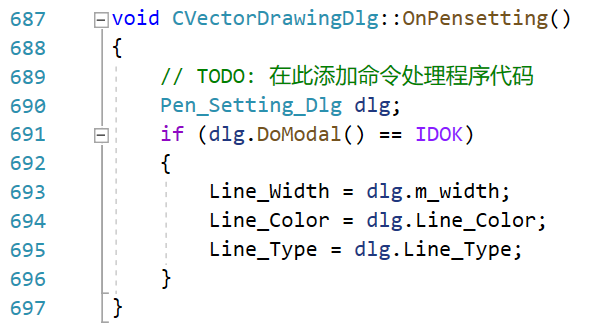


1. **画笔设置的实现**

首先定义绘图时用到的图形参数，包括：是否填充、线型、线宽、线条颜色、填充颜色、点的样式等。

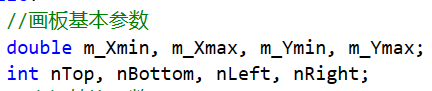


用户在点击设置样式时，弹出响应的对话框，用对话框中用户设置的参数覆盖掉当前的参数即可完成设置。

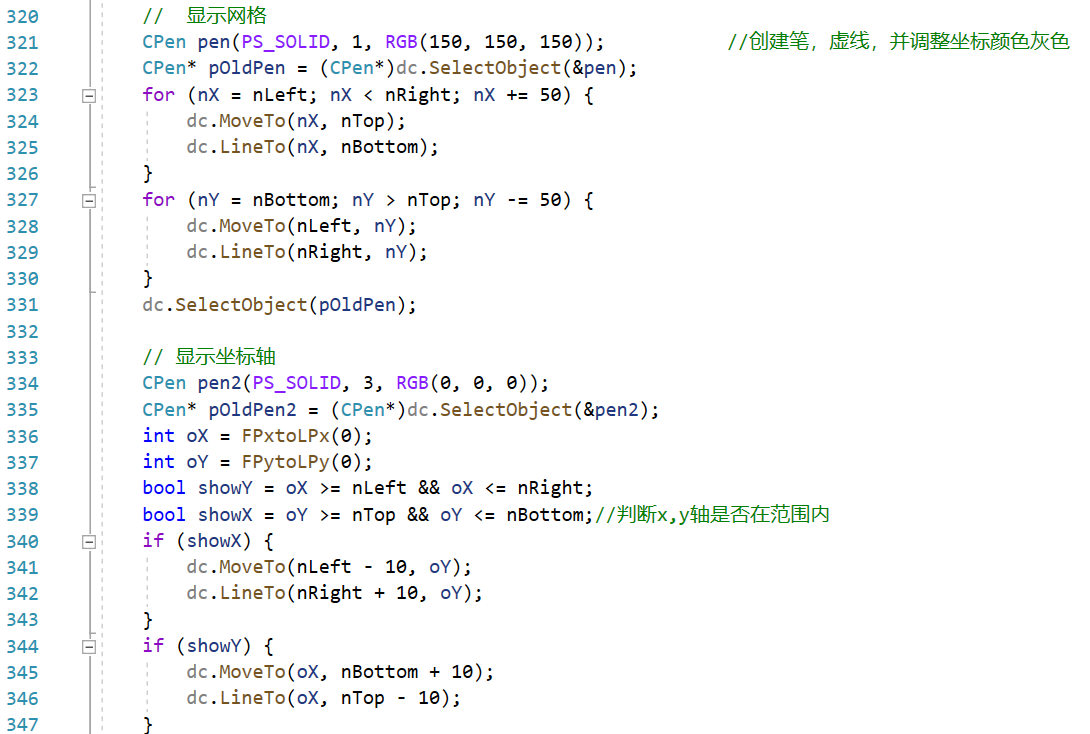


1. **坐标网络的实现**

首先定义画板的基本参数：XY的范围以及绘图区域的位置

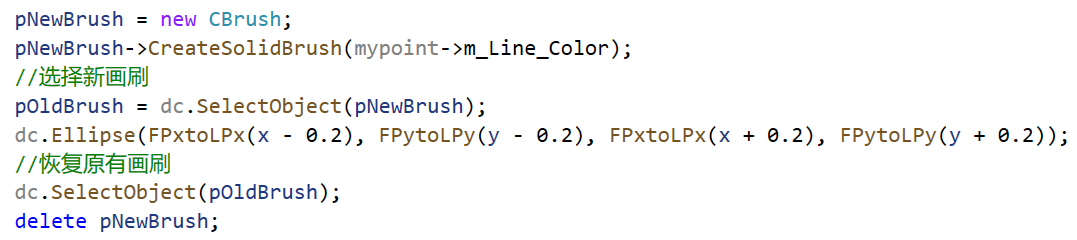


然后根据参数绘制坐标和网络，使用MoveTo和LineTo绘制直线即可



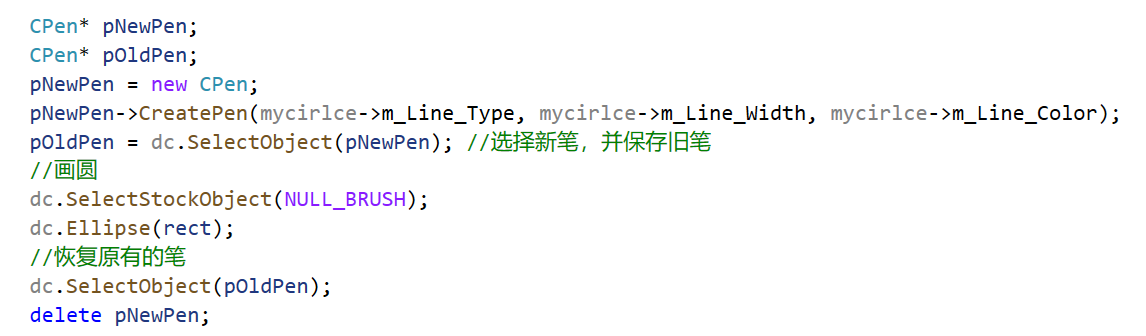
1. **点的绘制的实现**

在绘制点时，因为要考虑到视觉效果，选择绘制一个圆形来代替像素。定义绘制的画刷，选择颜色，以0.2为半径绘制一个圆



1. **线段的绘制的实现**

在绘制线段时，首先定义CPen类型的指针，设置画笔的颜色、线型、宽度等参数，然后使用MoveTo和LineTo绘制线段



1. **多边形的绘制的实现**

在绘制多边形时，首先将多边形的顶点传入一个点集数组p中，然后使用Polygon函数绘制



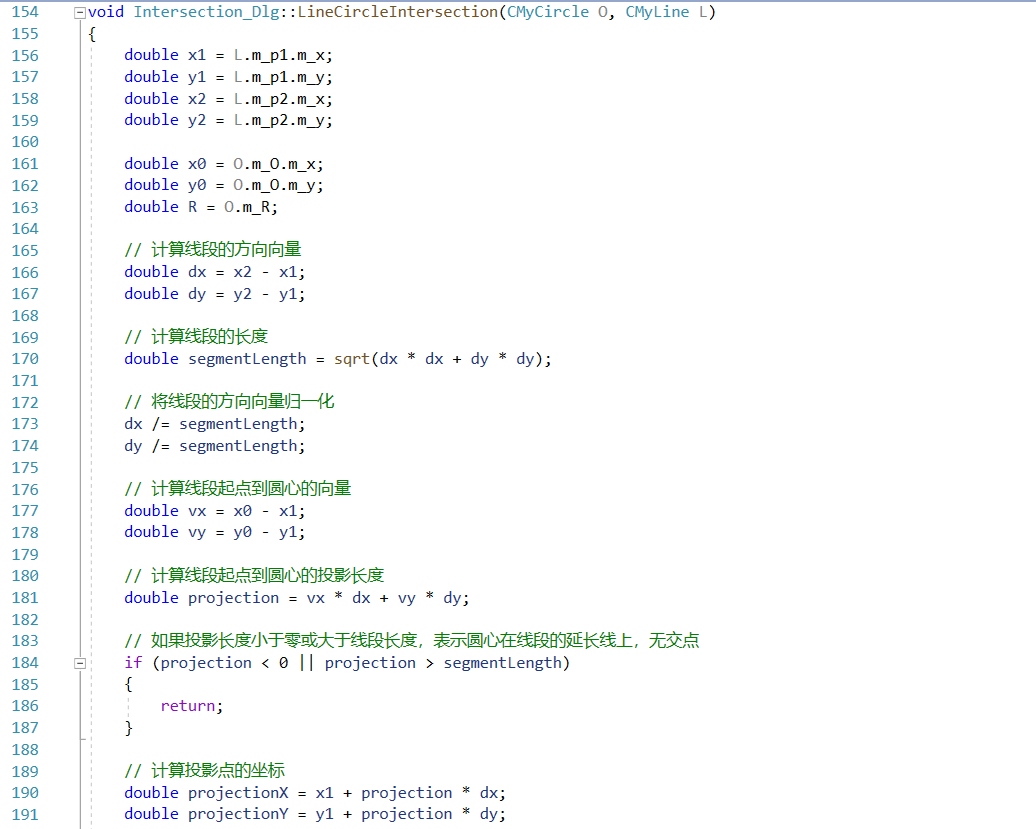
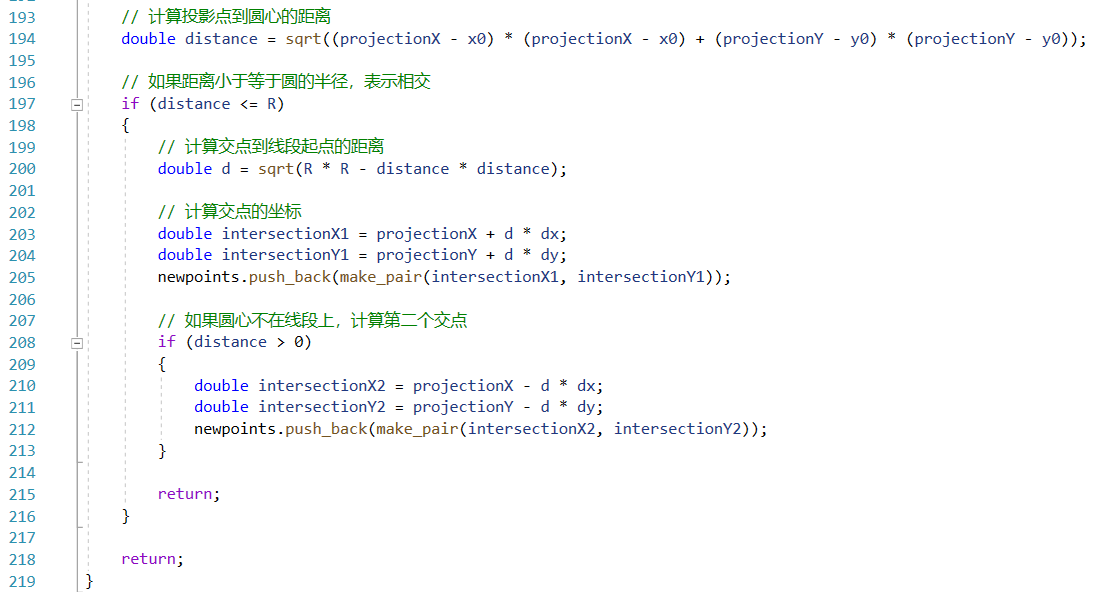
1. **对象求交点的实现**

在求线段与线段的交点时，首先计算两条线段的方向向量，然后使用向量的叉乘判断它们是否相交。如果两个叉乘的乘积都小于零，表示线段相交。然后，计算交点的坐标。

计算线段与多边形是否相交时，只需要利用上述函数，逐一判断多边形的每条边与线段是否相交即可。



在计算线段与圆是否相交时，首先计算线段的方向向量，并将其归一化。然后，计算线段起点到圆心的投影长度，并检查投影点是否在线段上。如果投影点不在线段上，表示圆心在线段的延长线上，因此没有交点。如果投影点在线段上，计算投影点到圆心的距离。如果距离小于等于圆的半径，表示相交，计算交点的坐标。

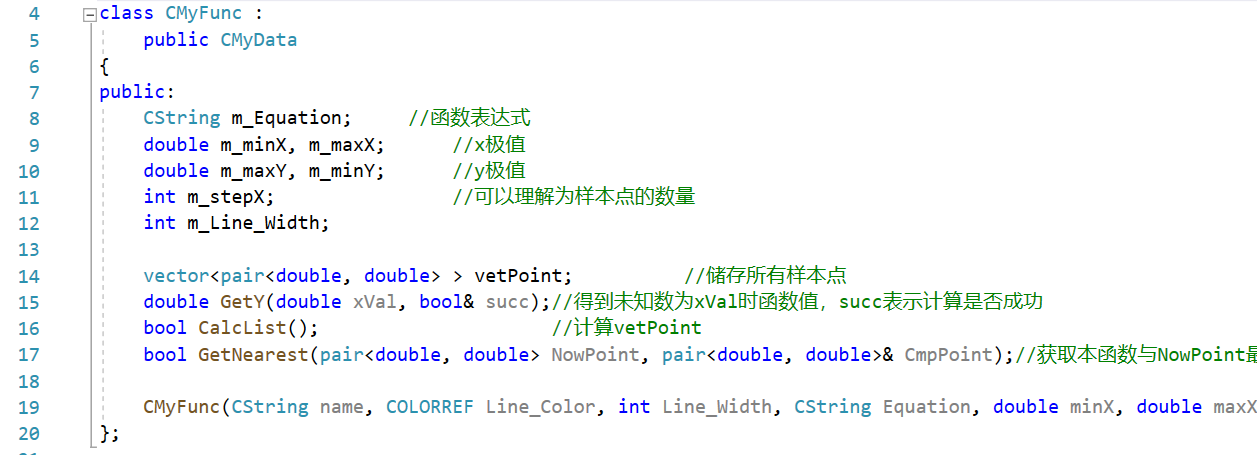
 

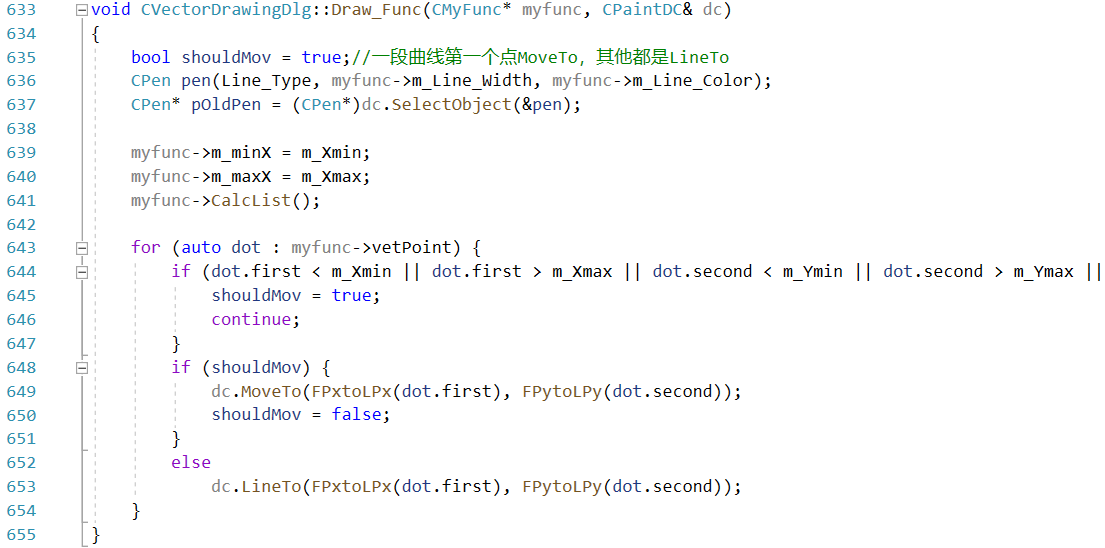
1. **函数绘制的实现**

在绘制函数时，使用逆波兰表达式（Reverse Polish Notation, RPN）算法。该算法用于将中缀表达式转换为后缀表达式，方便进行计算。在函数中，通过将数字和运算符按照一定规则入栈和出栈的方式，实现了对中缀表达式的转换和计算。函数中使用了两个栈，分别为数字栈（num）和表达式栈（opt）。通过栈的先进后出（LIFO）特性，实现了逆波兰表达式的计算过程。



绘制函数时，首先计算函数中的所有样本点，函数类中使用vector来储存所有的样本点，CalcList()方法可以计算所有样本点的值。然后遍历函数中的所有样本点，将其依次连接即可显示函数。





1. **对象属性设置的实现**

首先在资源视图中绘制五个位图，分别为实心圆、空心圆、正方形、三角形与十字形位图。

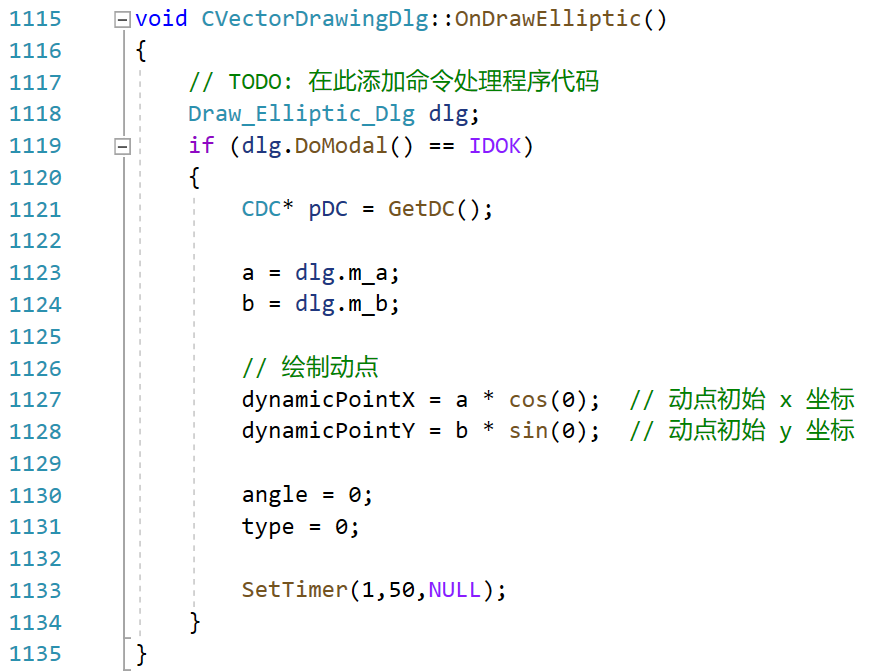


然后将位图插入到对话框中，使用radio button空间为用户提供选择交互。然后根据用户的样式绘制响应的图案即可。

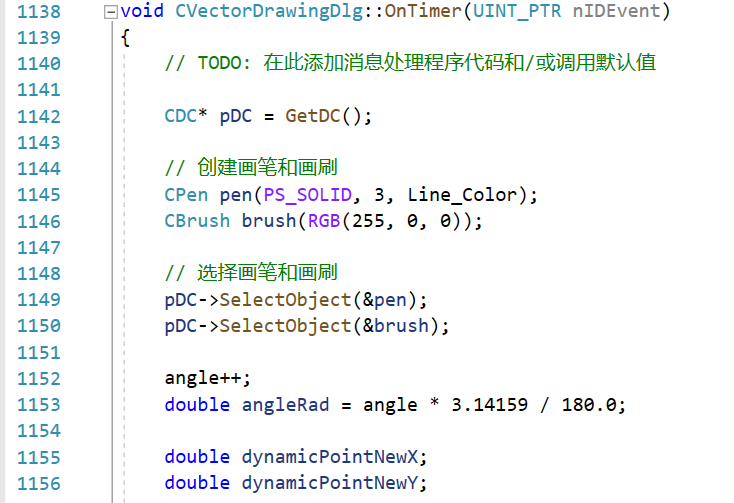
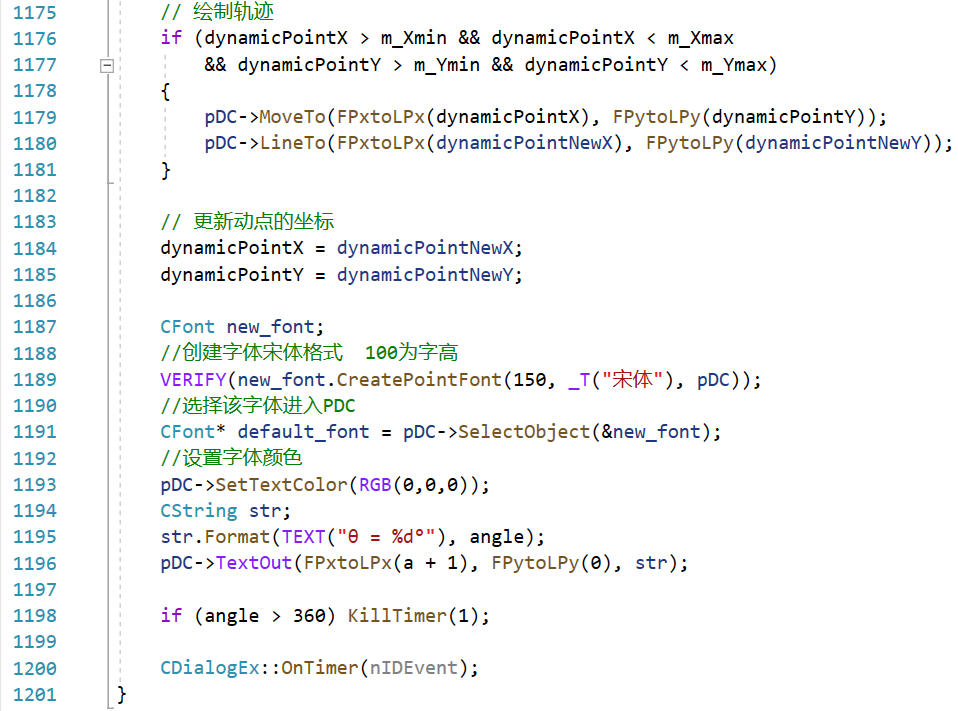


1. **动点轨迹绘制的实现**

首先查找图形的参数方程，在参数设置完成后，使用SetTimer函数启动定时器，这里设置为50ms响应一次。



在定时器的响应函数中，首先更新角度的值，然后根据参数方程计算当前动点的位置，将更新的点与之前的点相连，这样即可绘制动点的轨迹。角度大于360°后，使用KillTimer函数关闭定时器，绘制完成

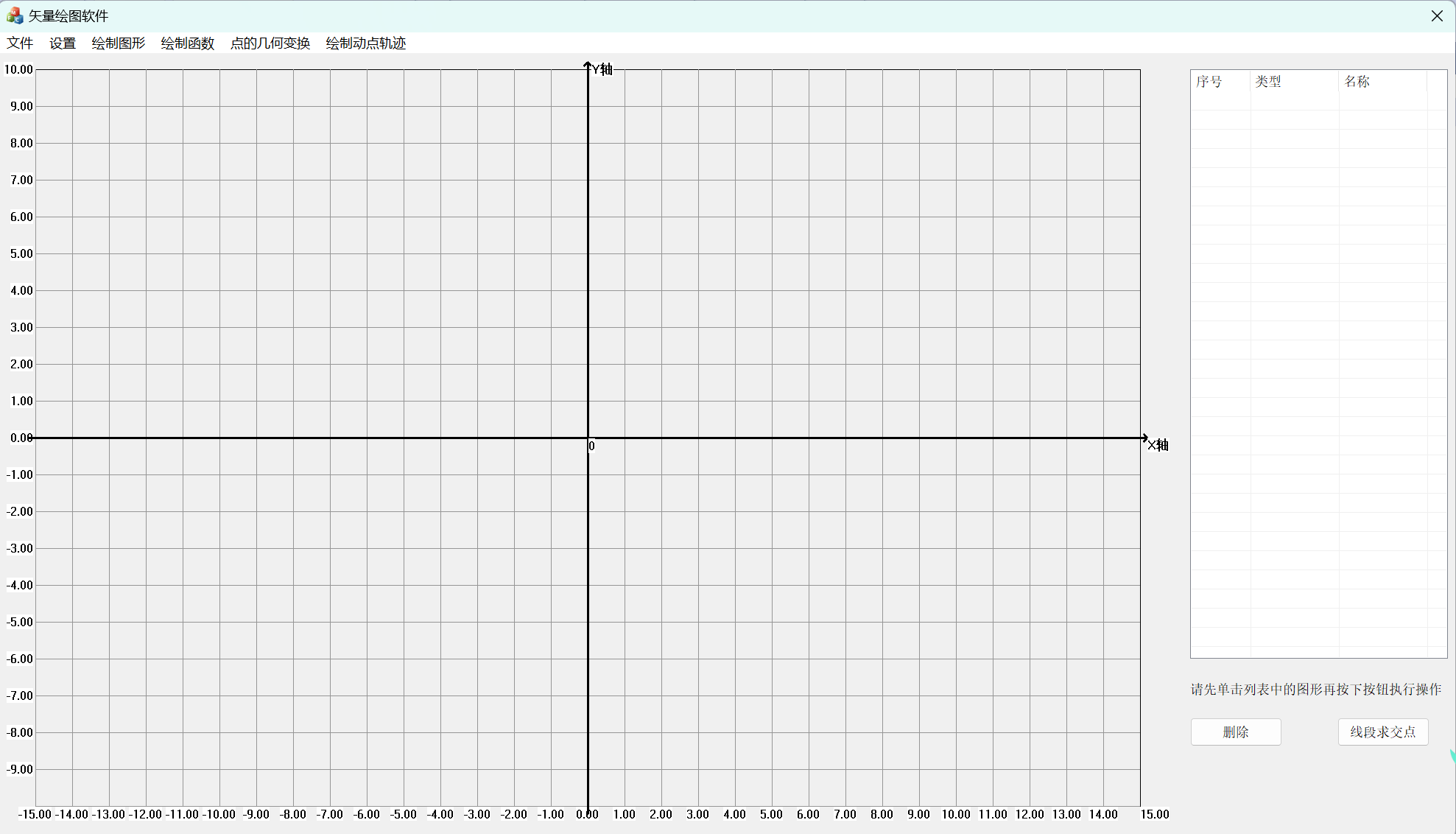
1. **导出图片的实现**

导出图片时，首先获取客户区，创建兼容性位图。创建完毕后，将位图转换为一般图像，然后创建文件选择对话框，记录保存的路径与图片格式，写入图片文件即可。

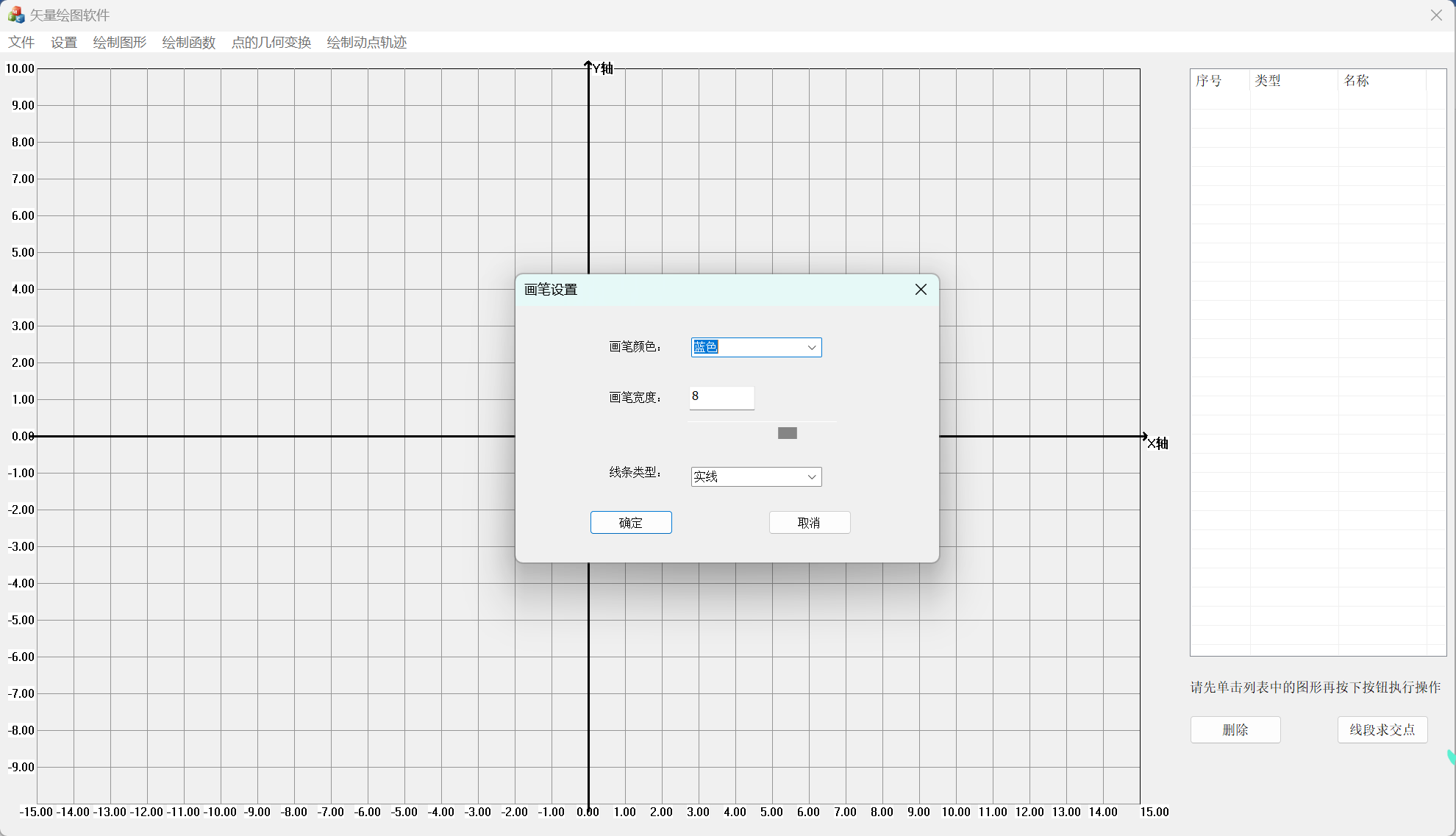


## 程序测试用例

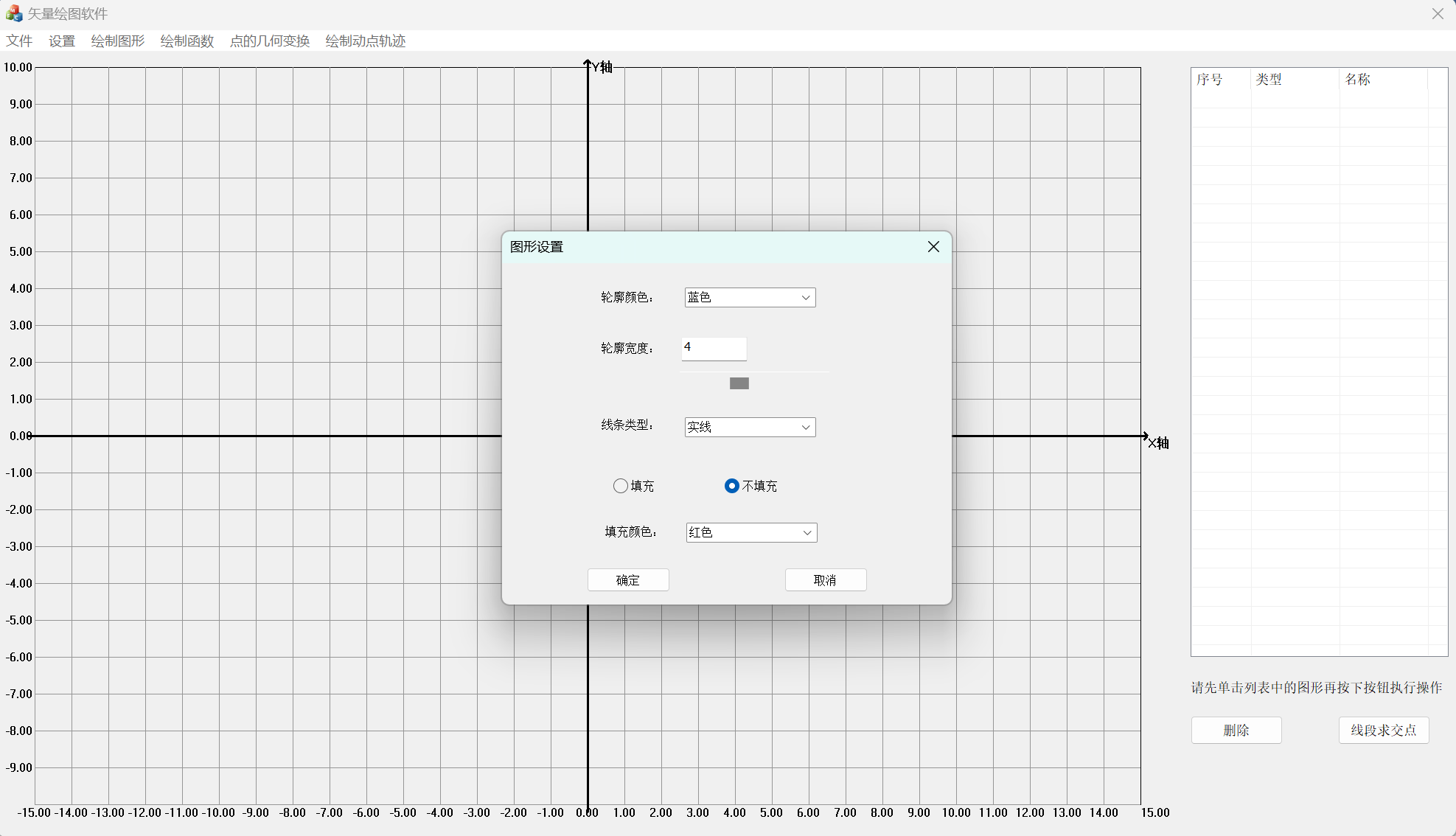
首先打开程序



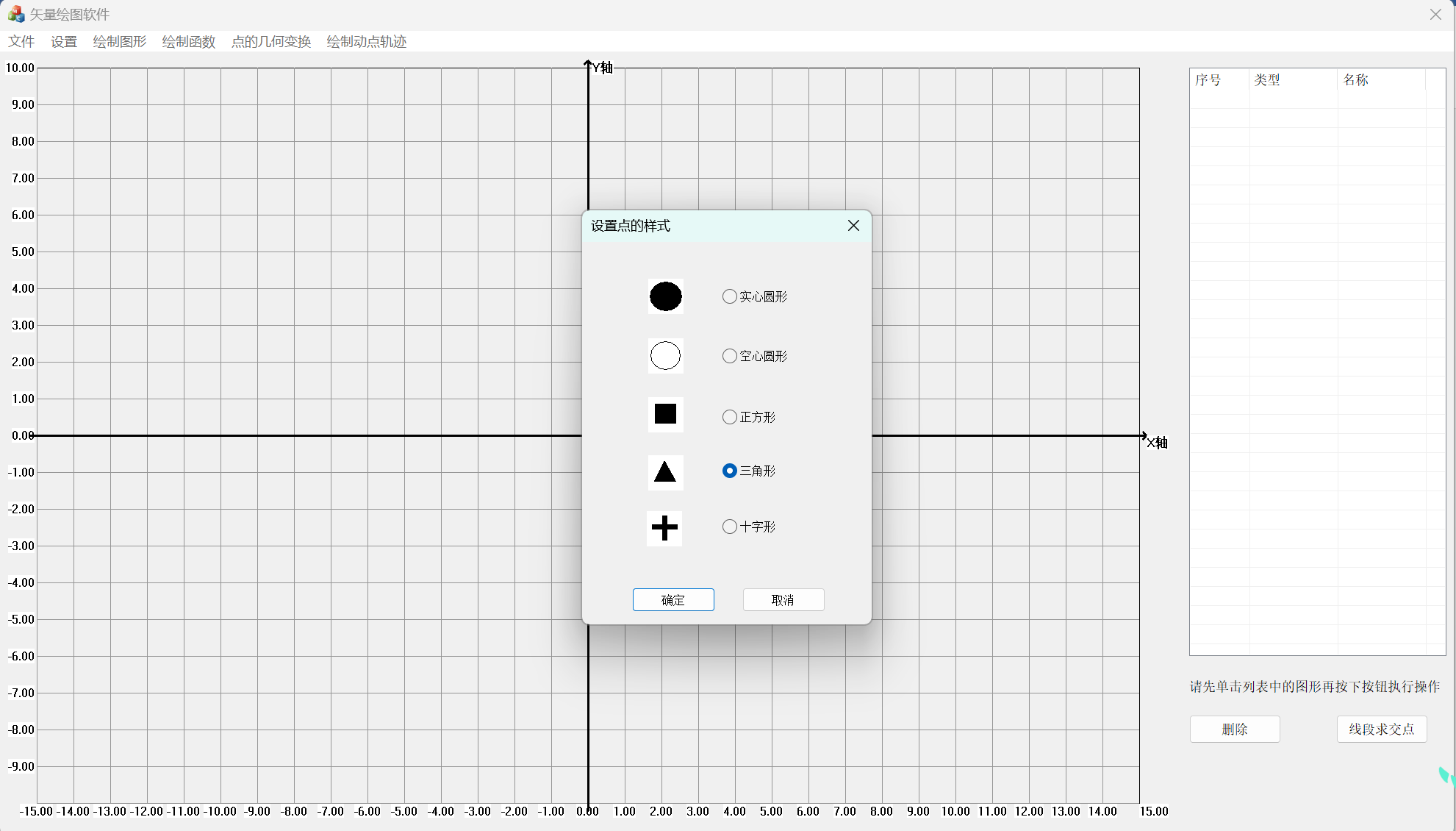
设置画笔颜色、画笔宽度、线条类型



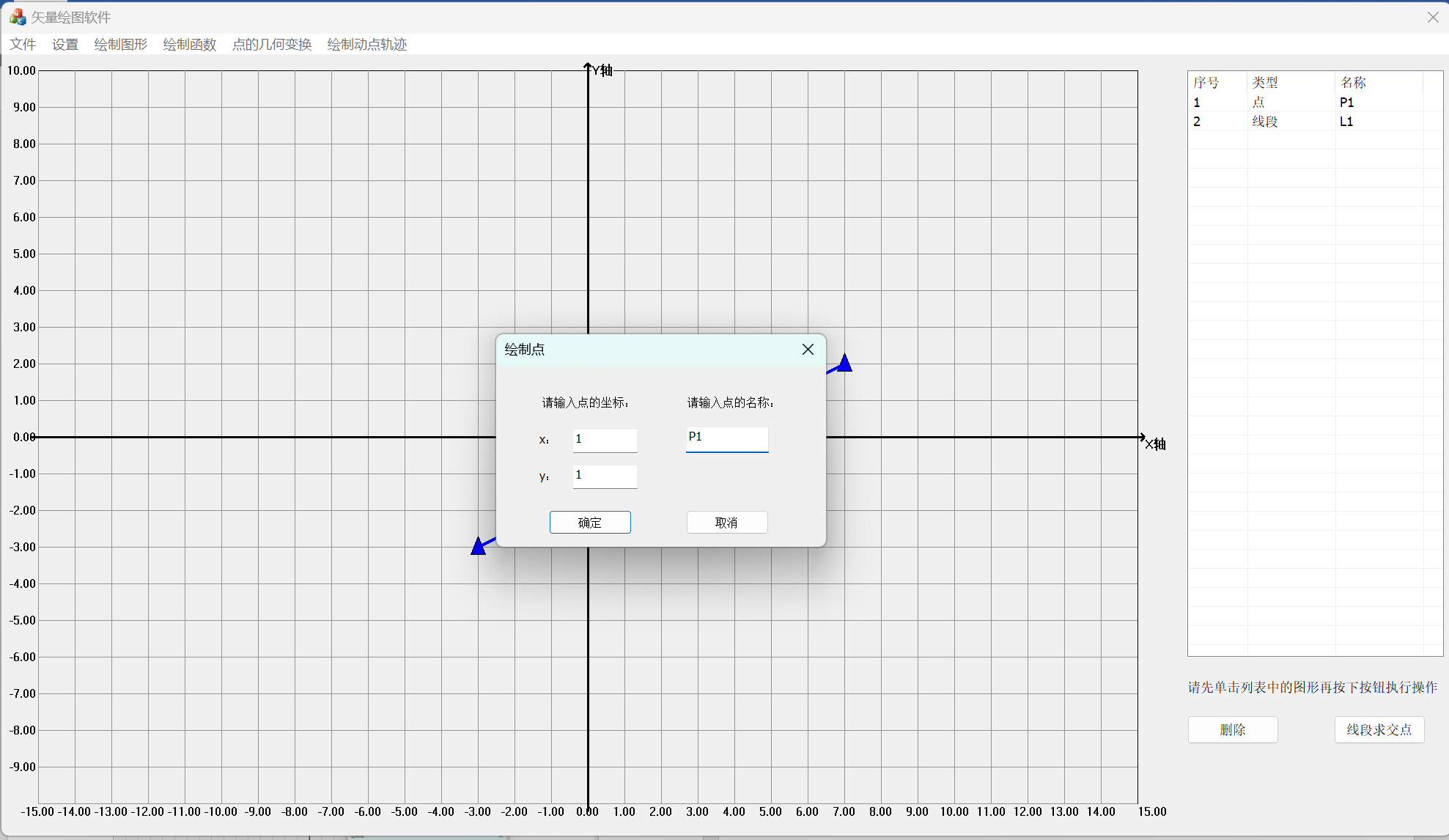
设置轮廓颜色、轮廓宽度、线条类型，是否填充、填充颜色



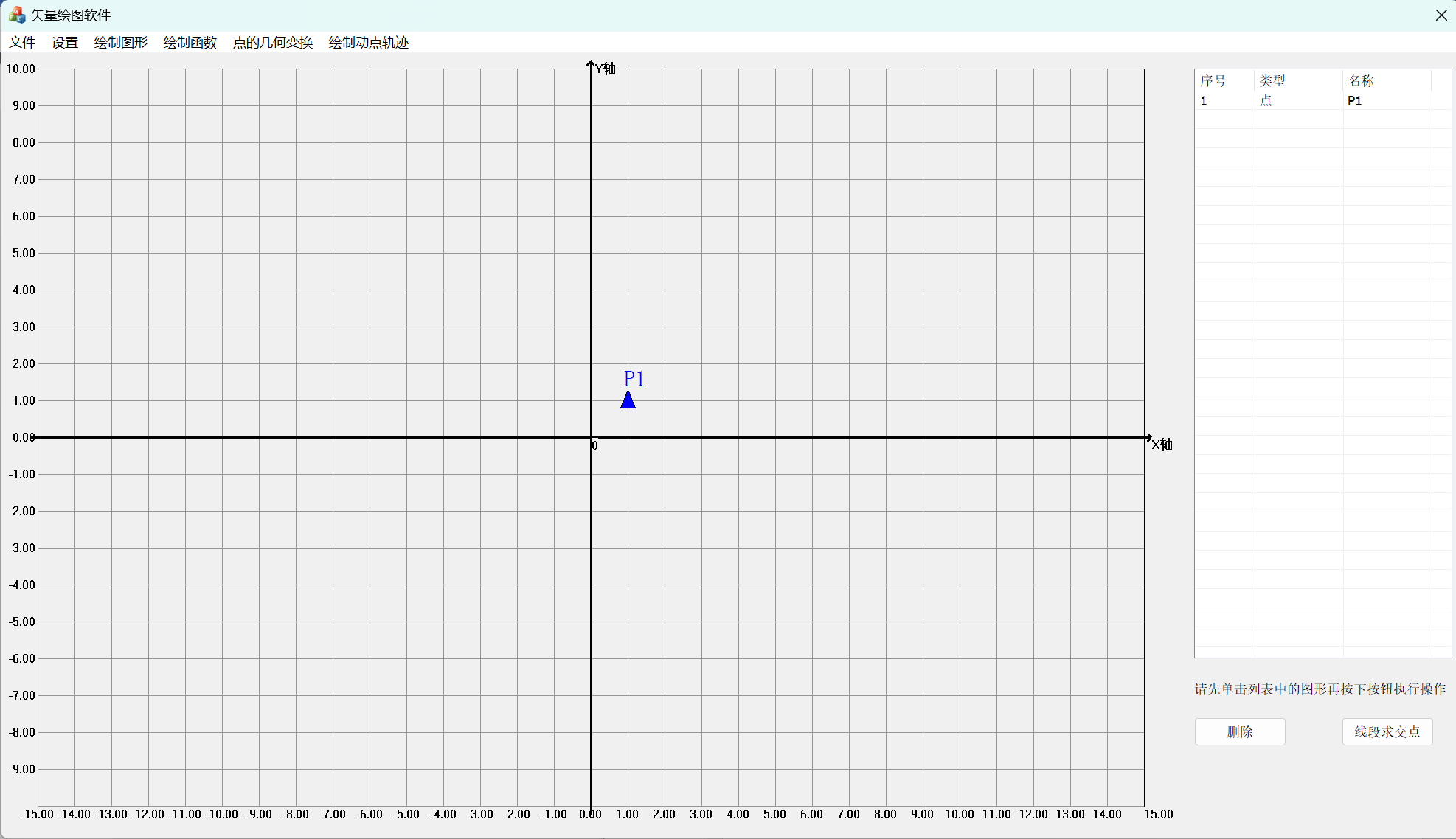
设置点的样式，这里选择三角形



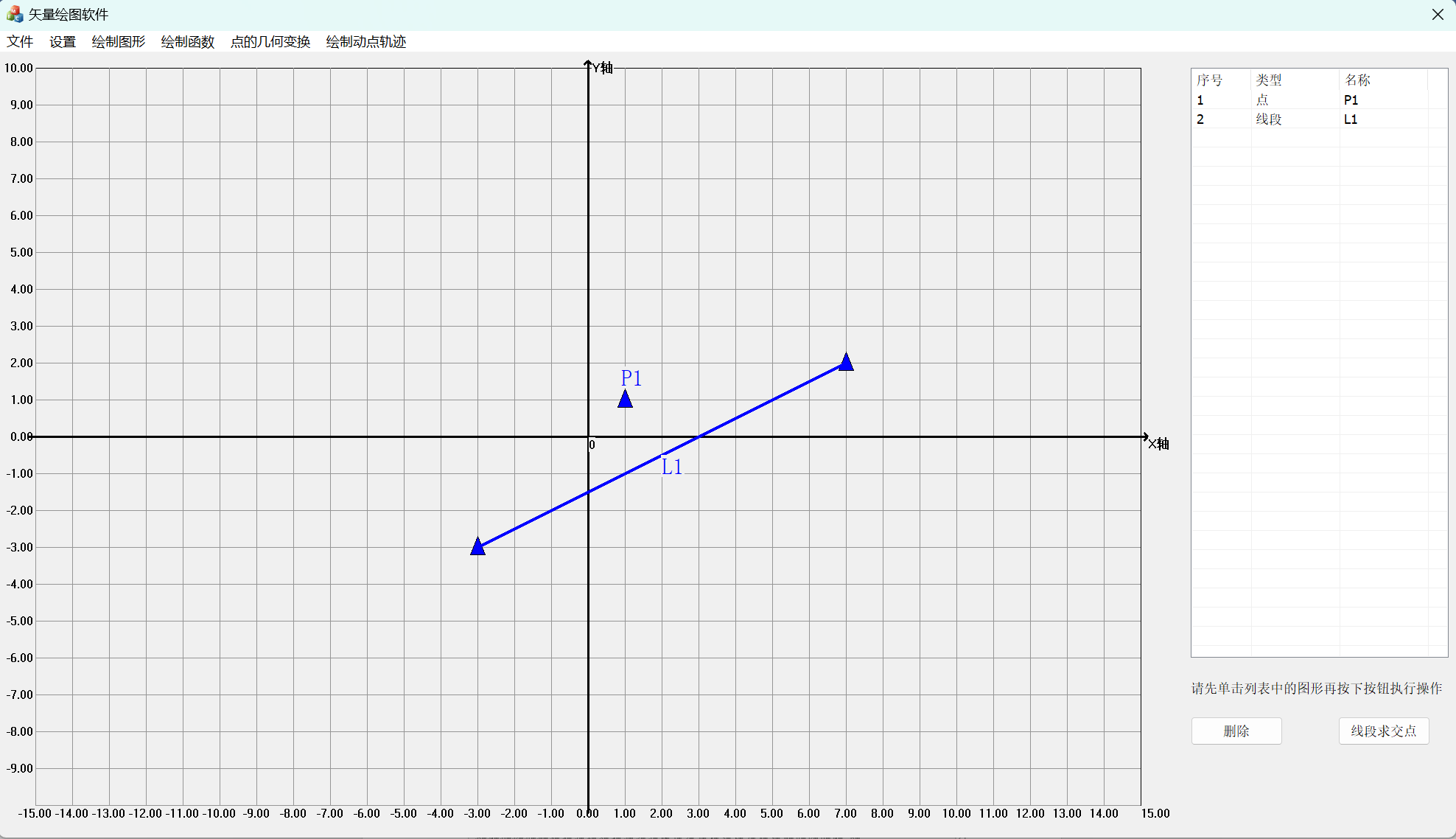
首先绘制一个点，坐标为（1，1），名称为P1



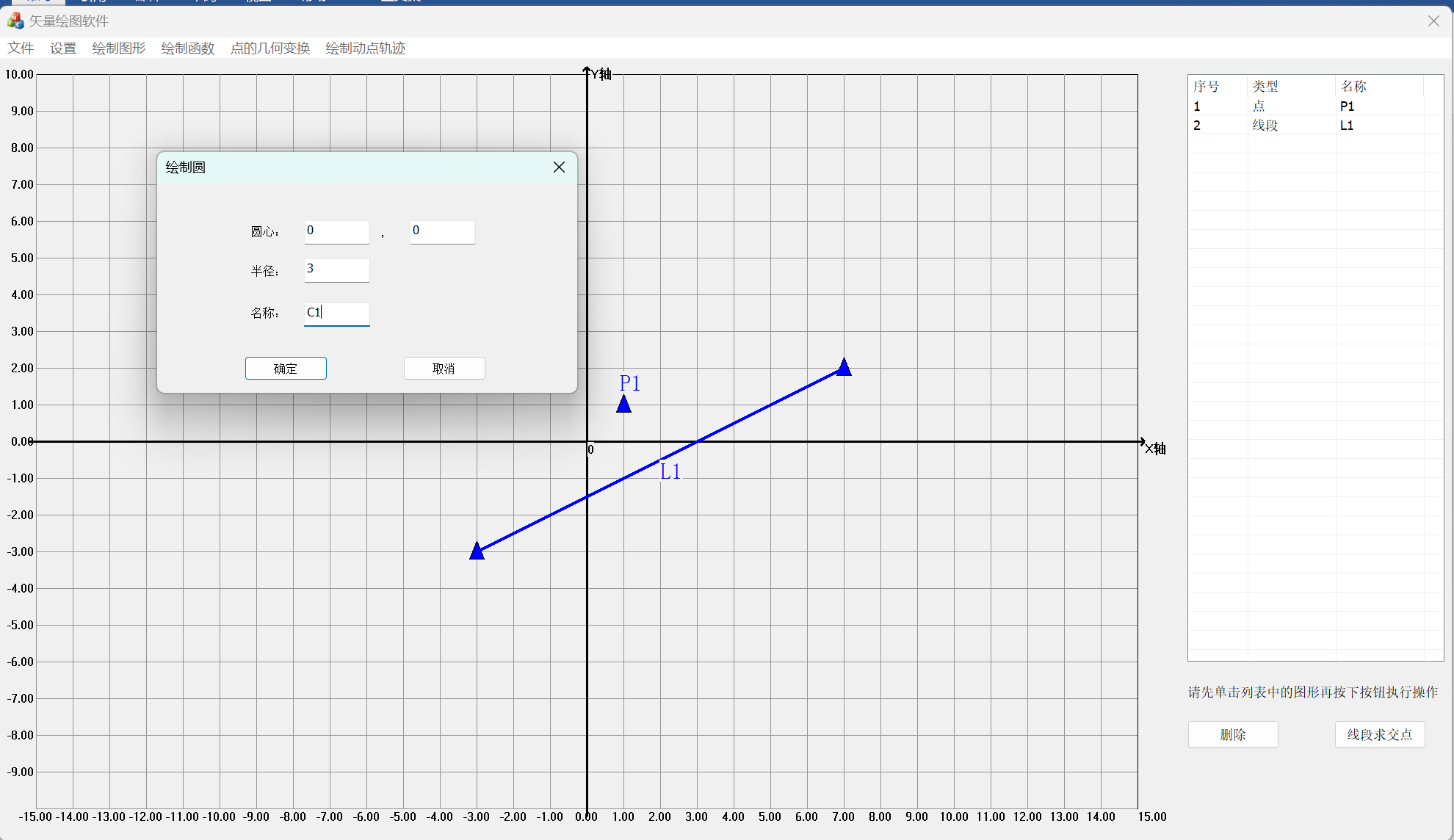
P1已成功绘制



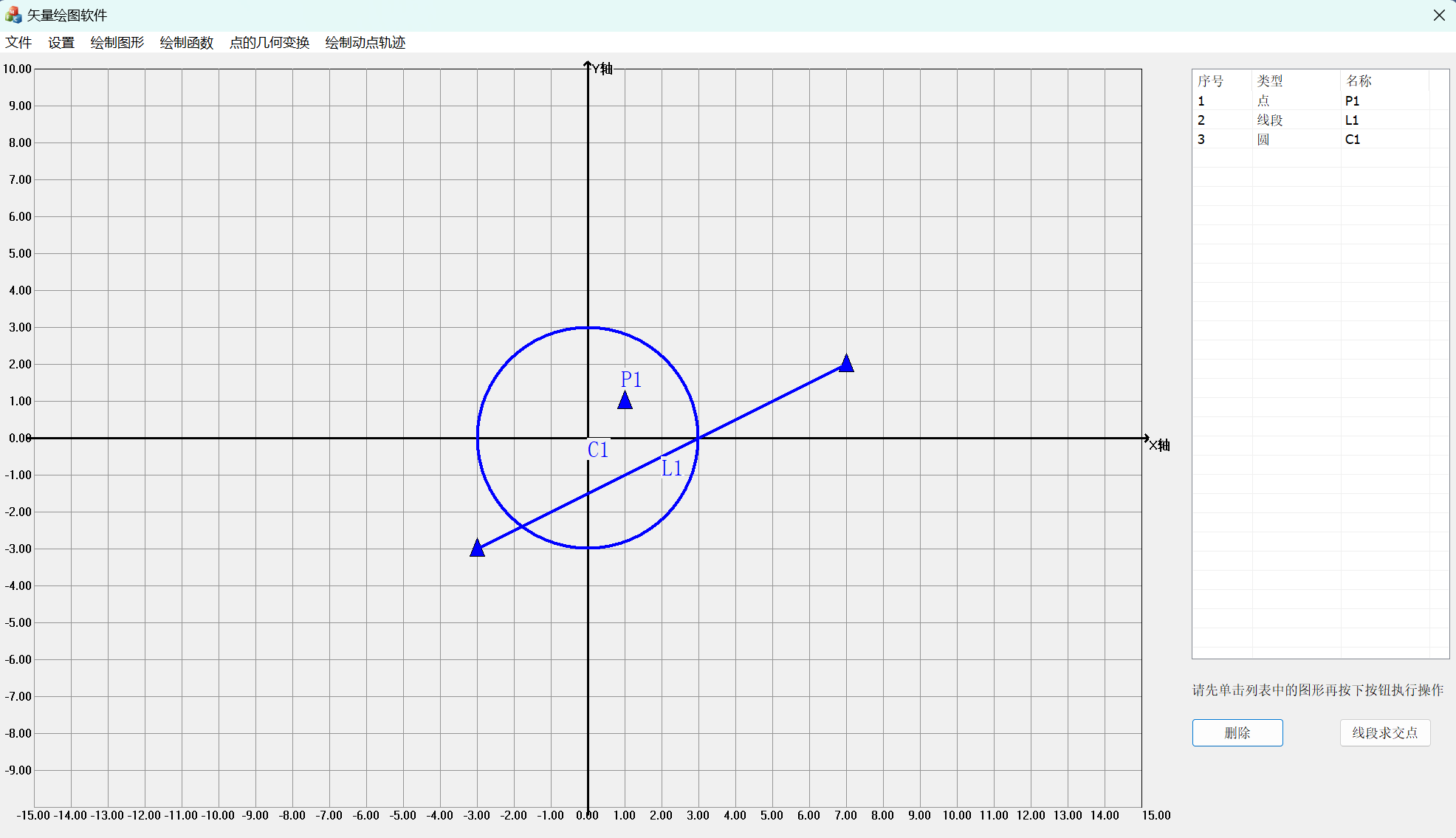
绘制直线L1，起点为（-3，-3），终点为（7，3），绘制成功



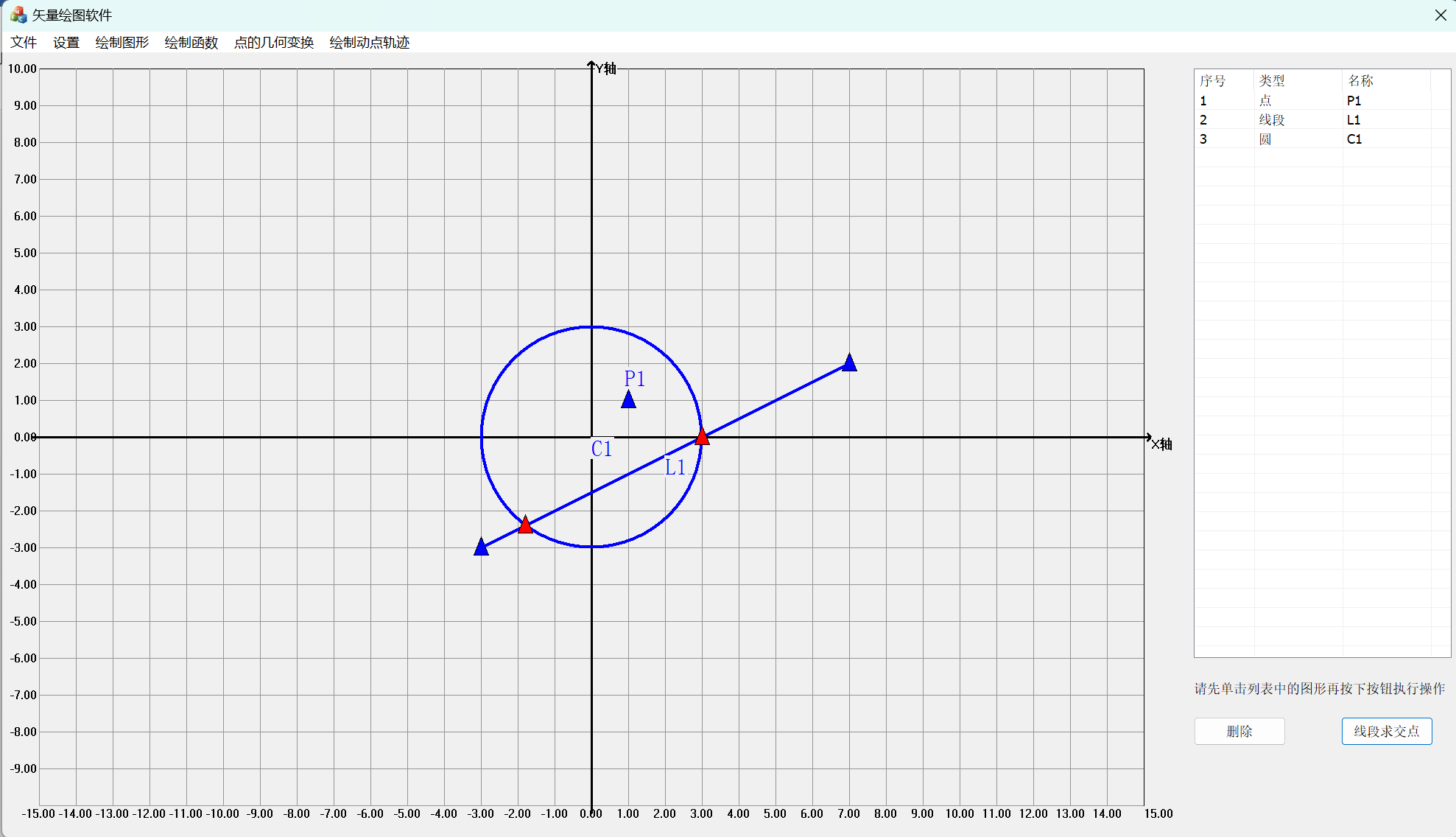
绘制一个圆，圆心在（0，0），半径为3，名称为C1



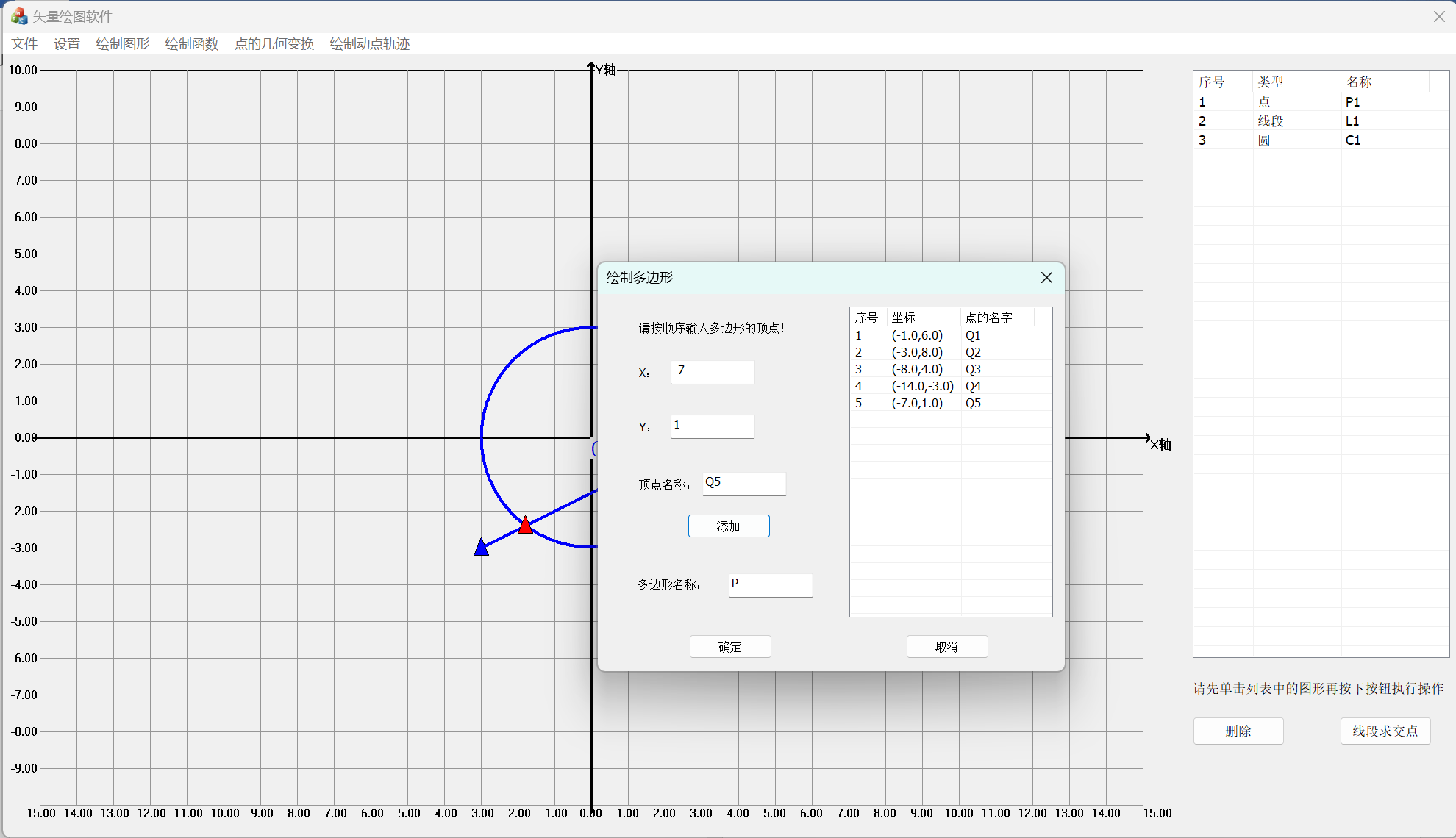
圆绘制成功

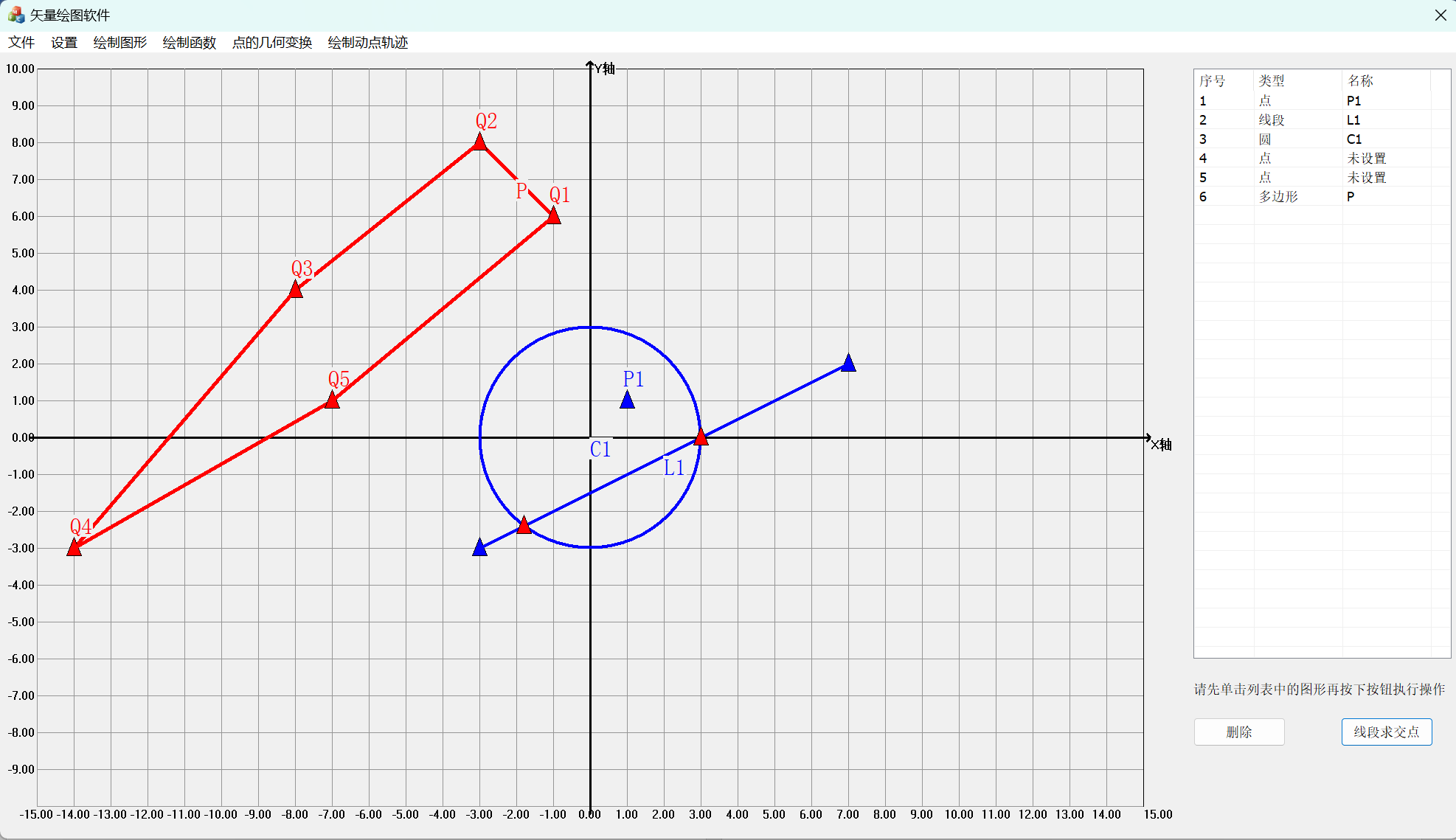


计算线段L1与圆C1的交点

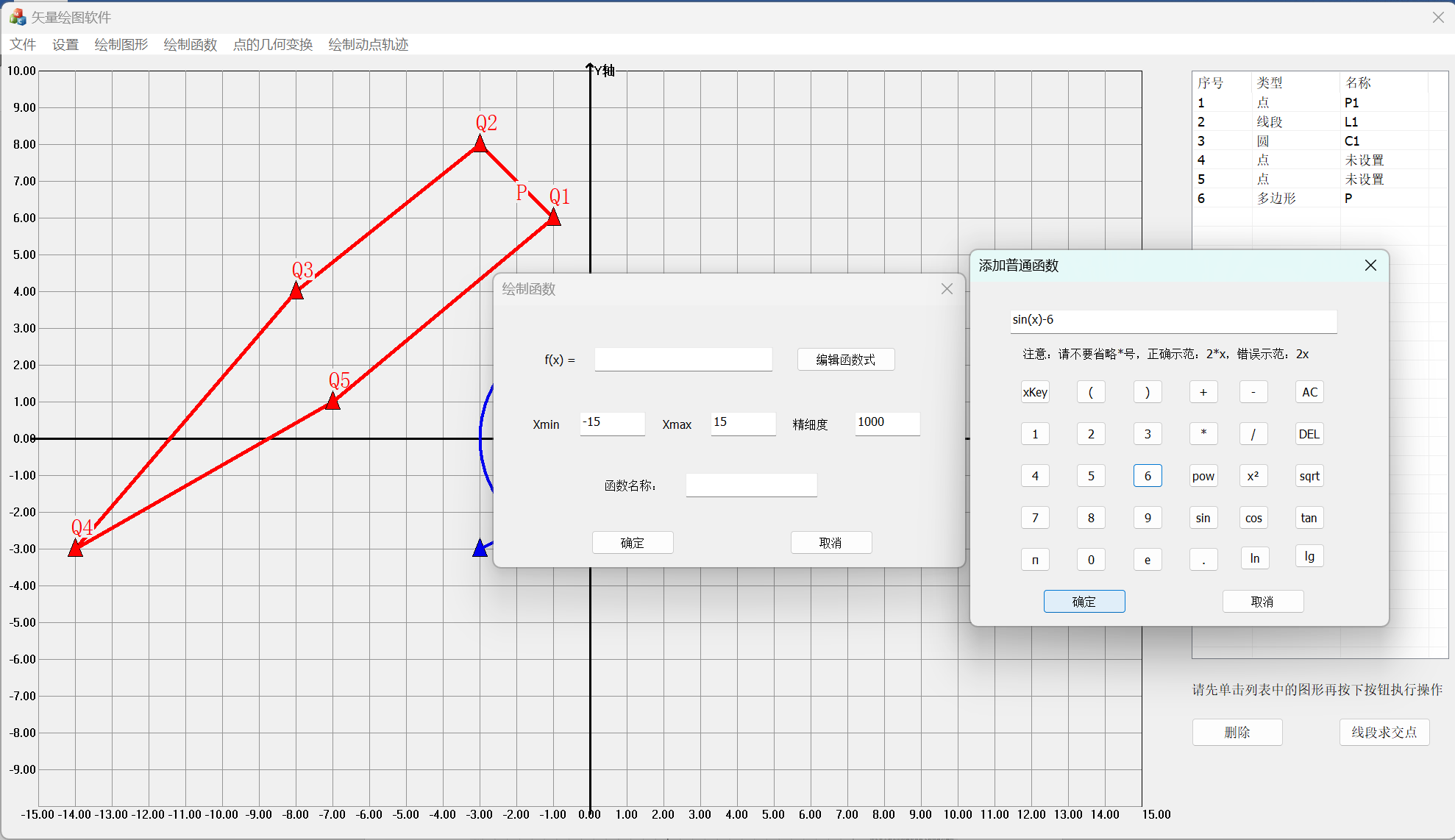
圆与线段存在两个交点，已成功绘制

绘制多边形，在对话框中依次输入顶点的坐标与名字



多边形绘制成功

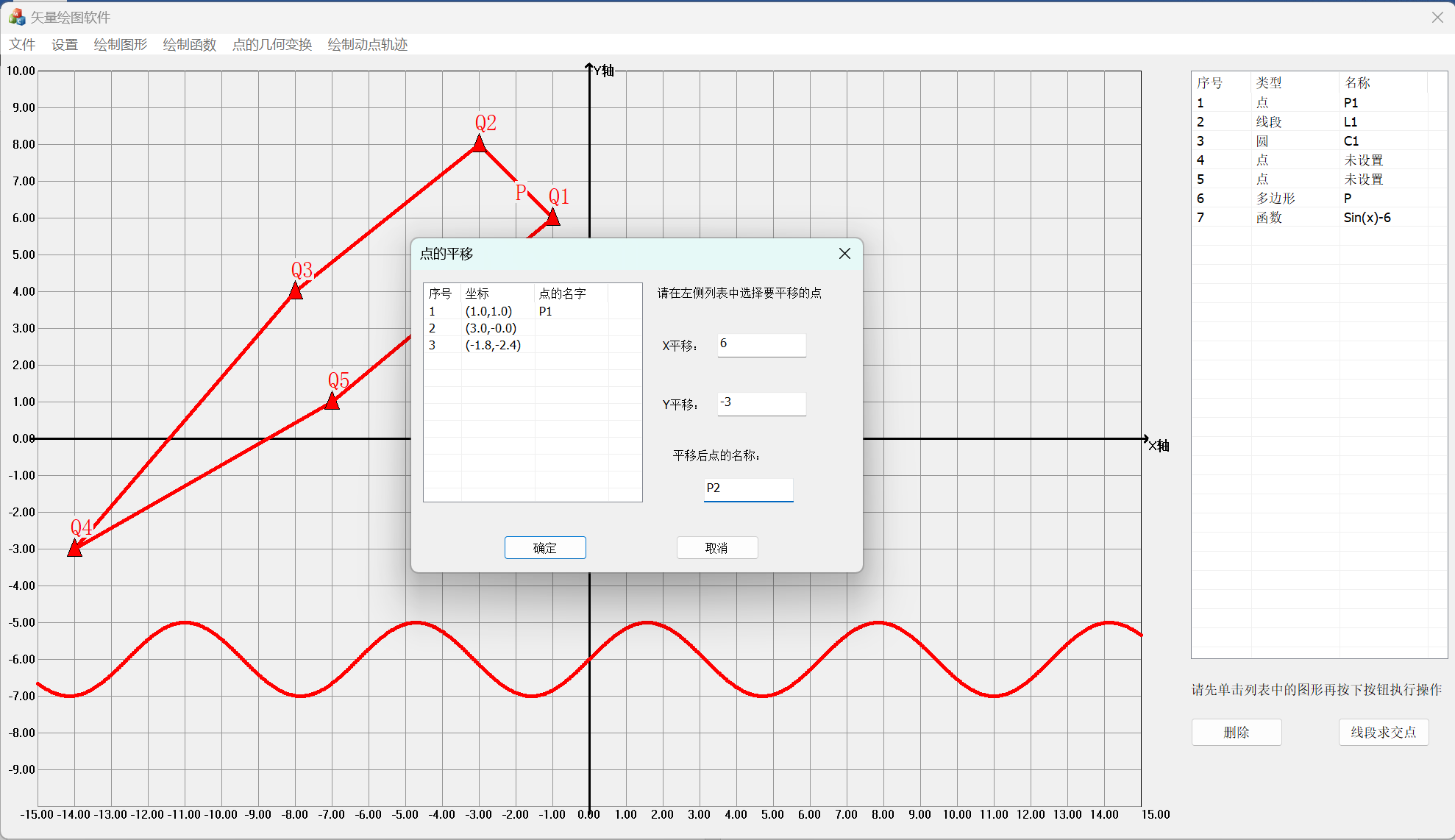
绘制函数，函数表达式为sin(x)-6



函数绘制成功



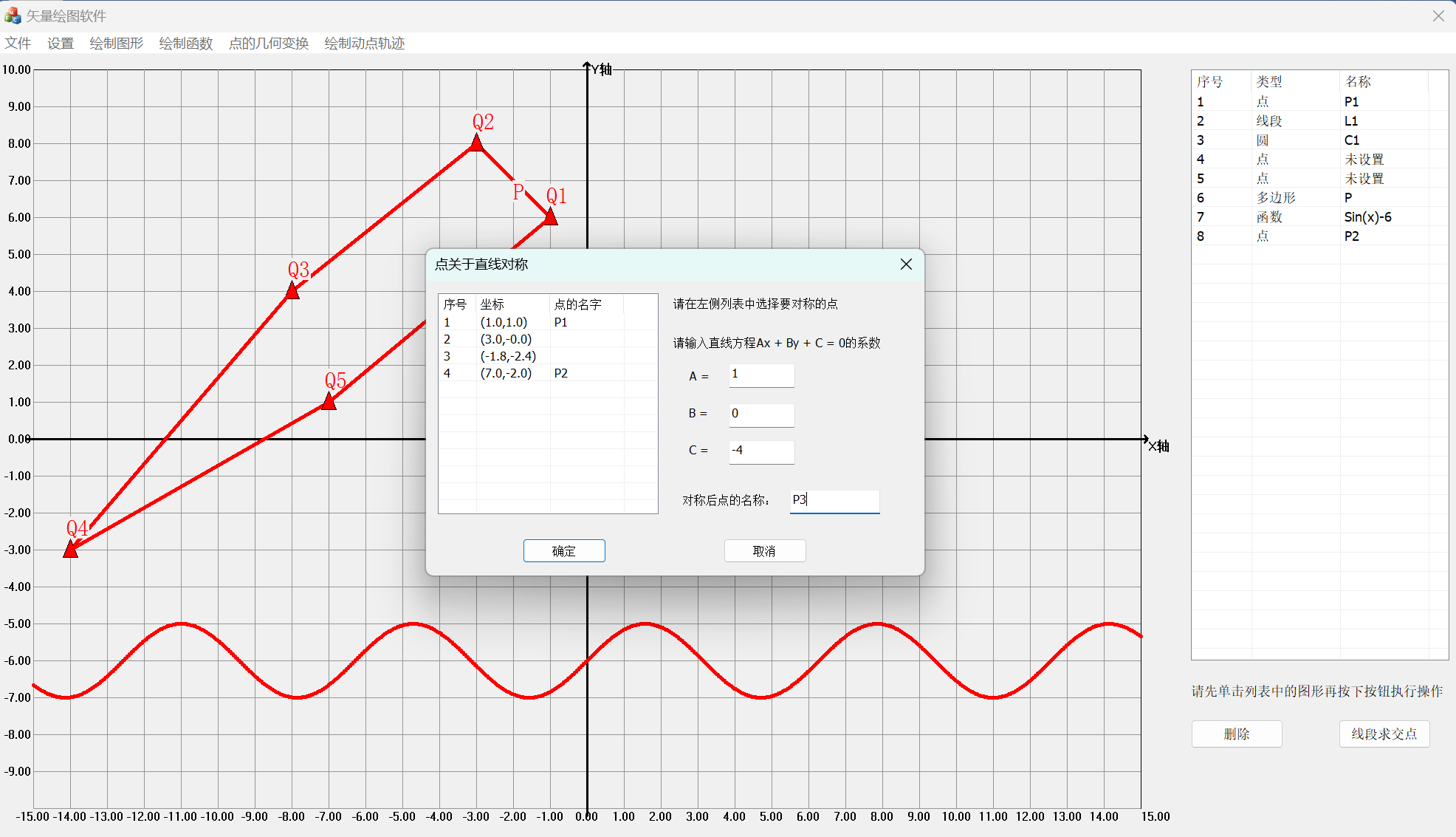
点的几何变化，首先时平移，将点P1，向左平移6个单位，向下移3个单位，平移后的点为P2



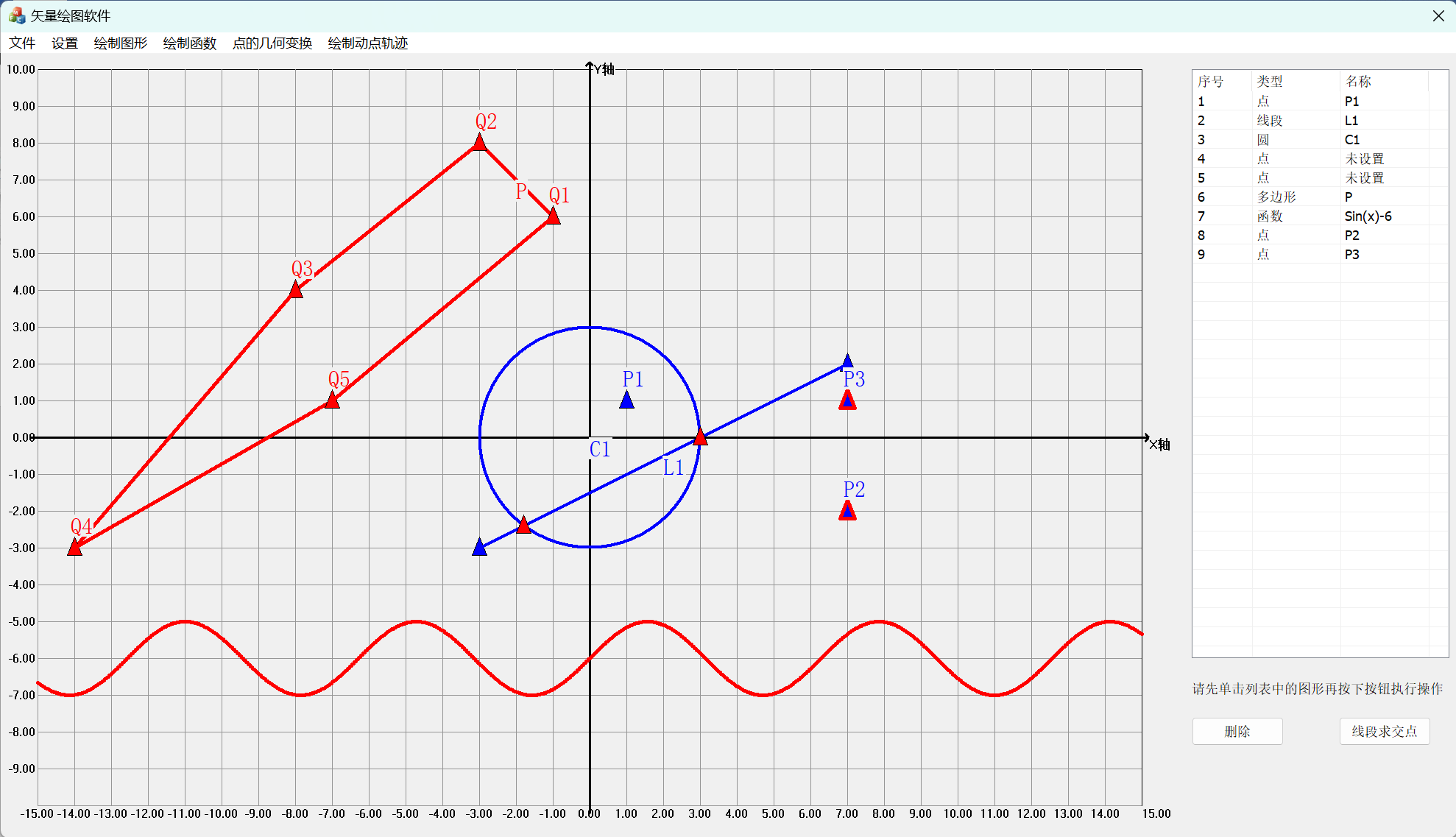
点平移成功，p2显示在画面上



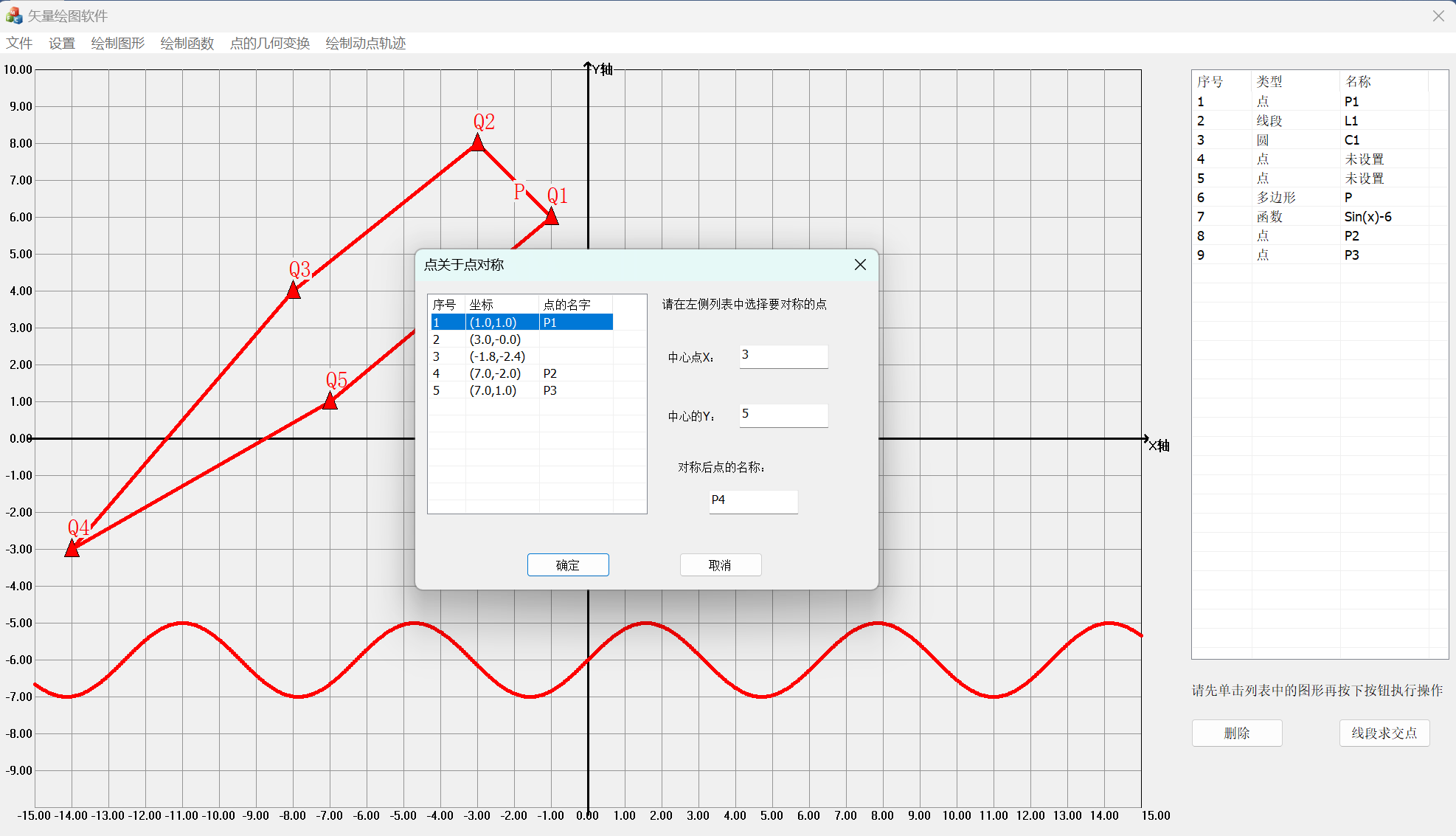
求点P1关于直线x=4的对称点，新点为P3



P3绘制成功

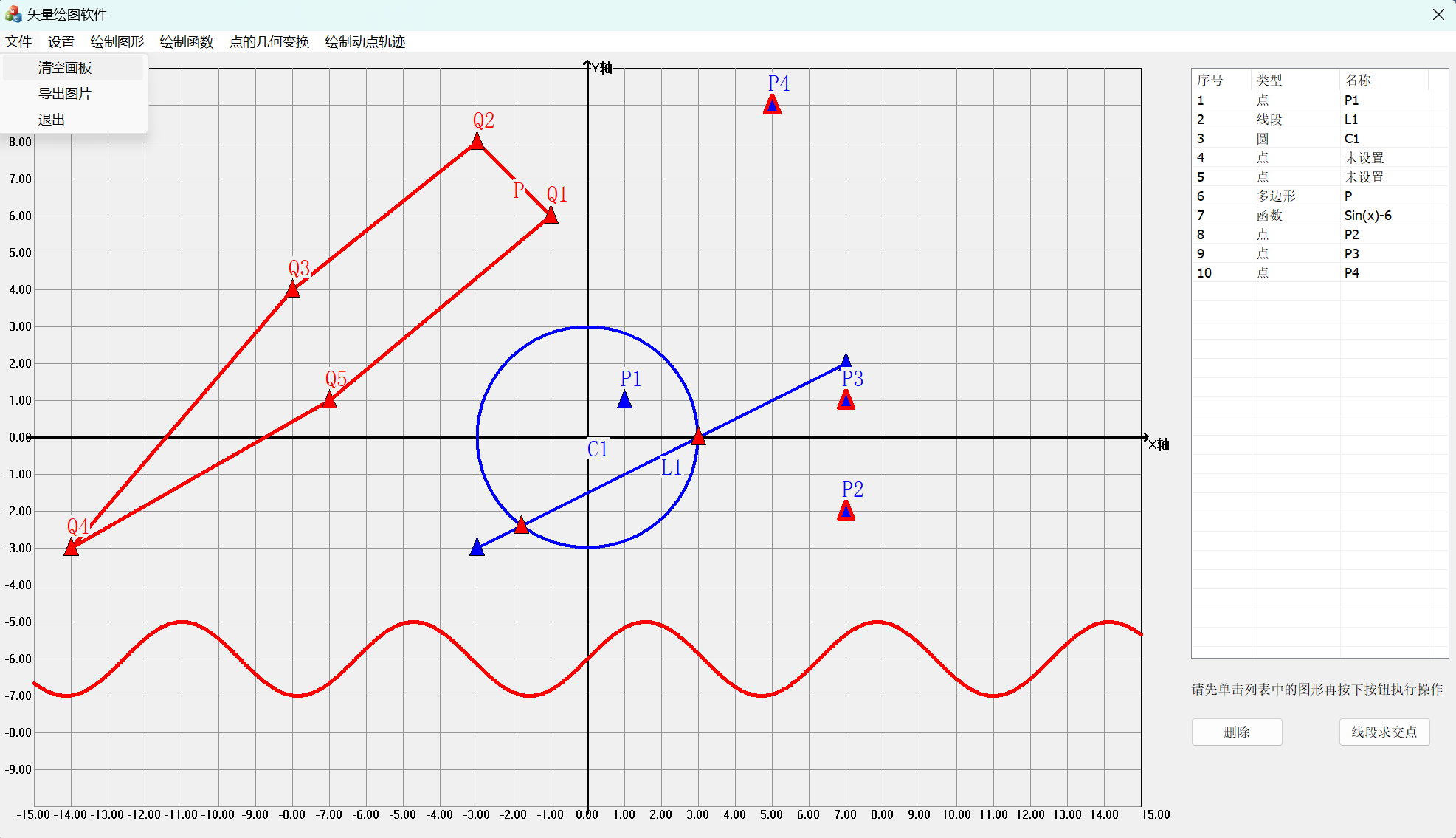


求点P1关于点（3，5）的对称点，新点为P4

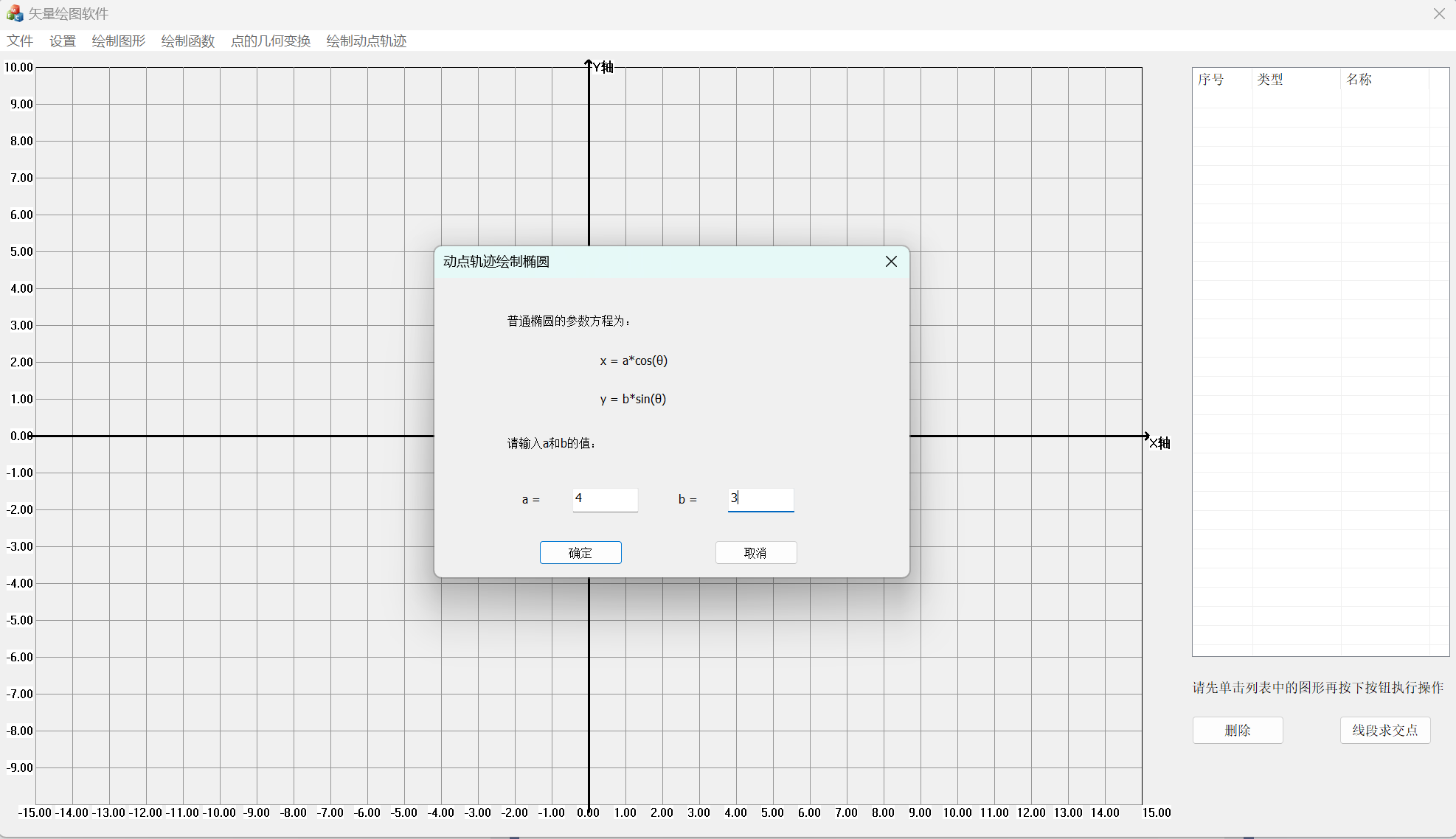


P4绘制成功



点击“清空画板”

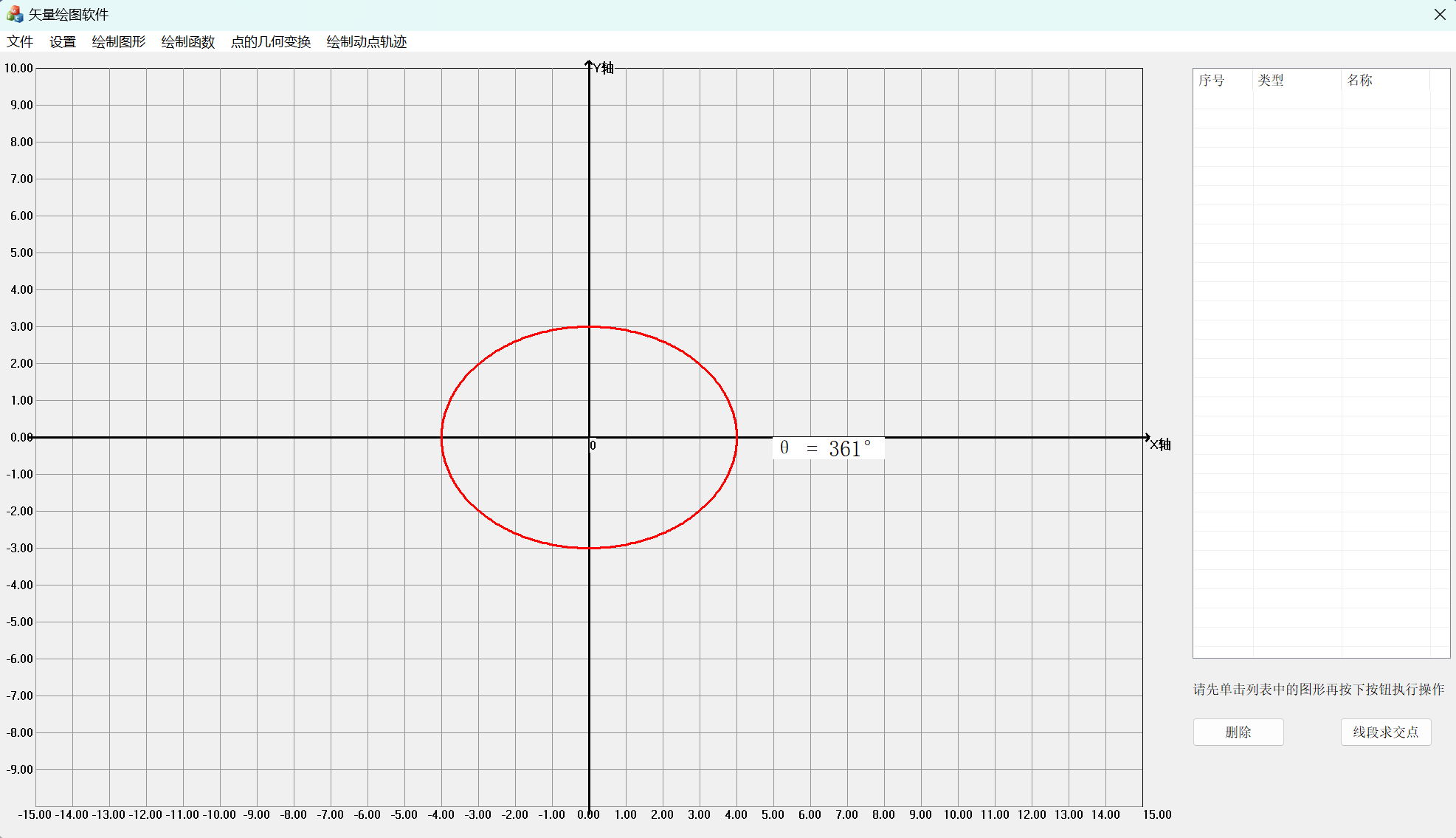
画板清空后，绘制椭圆的动点轨迹

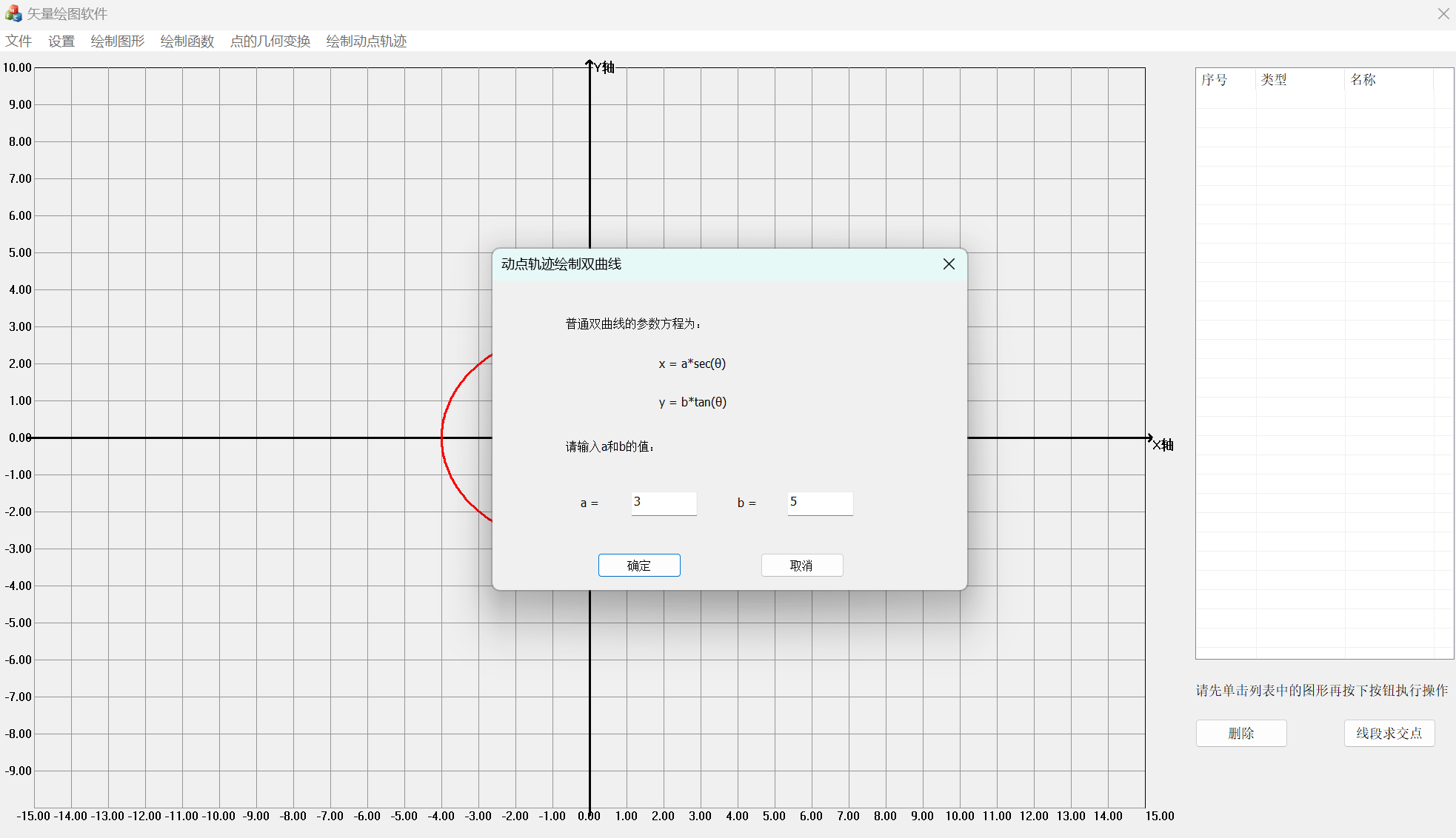


正在绘制椭圆

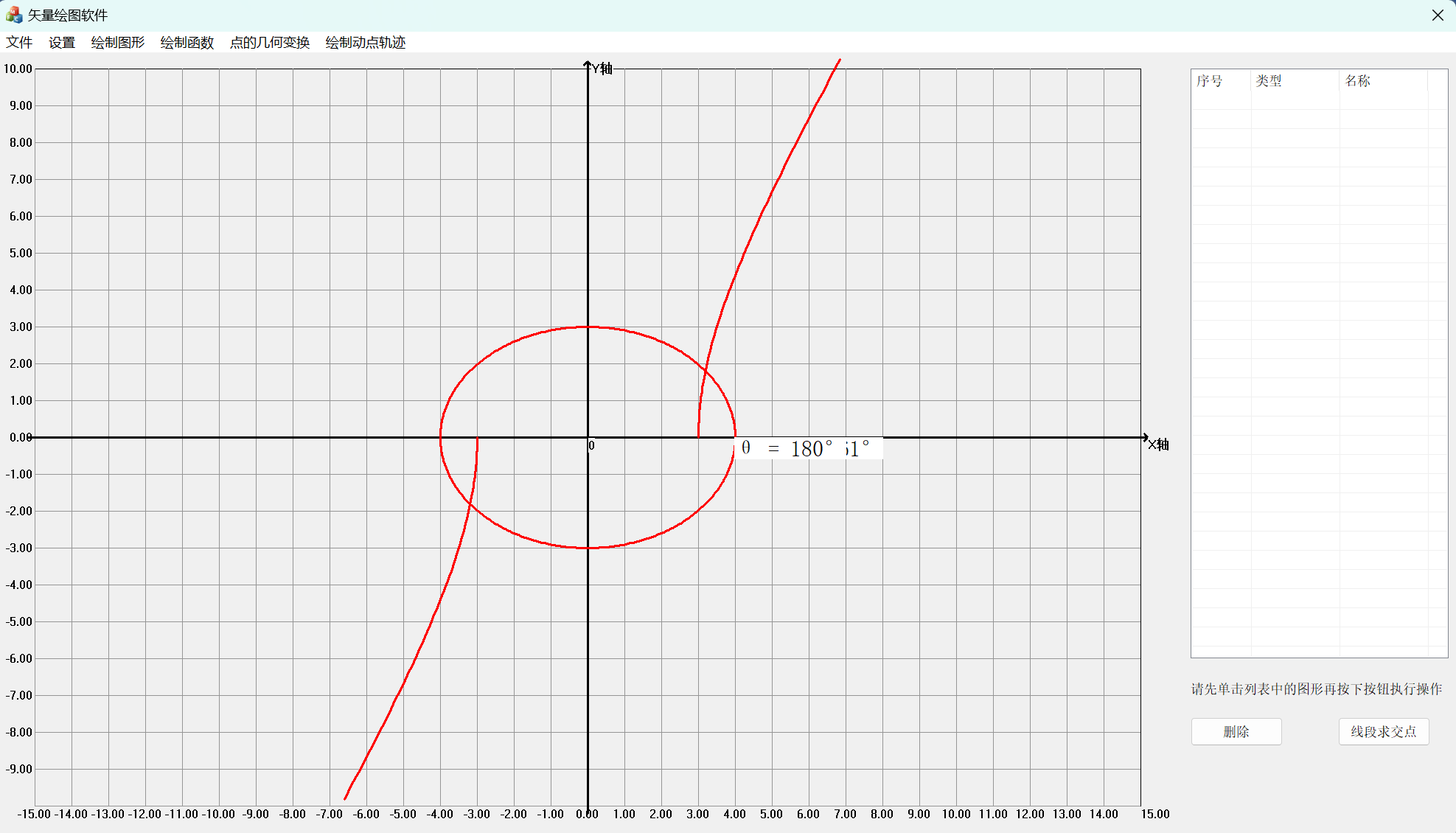


椭圆绘制成功

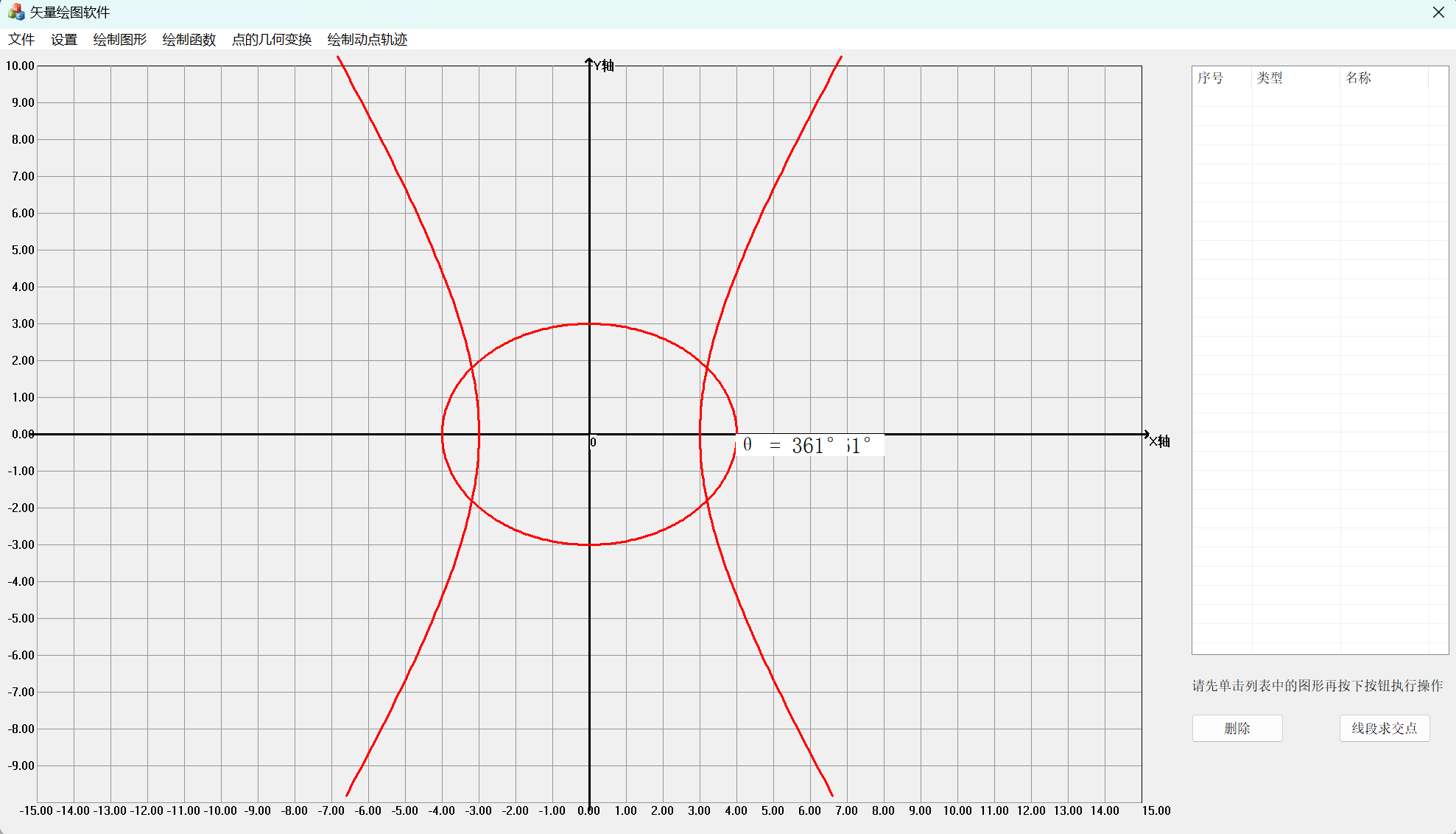


绘制双曲线的动点轨迹

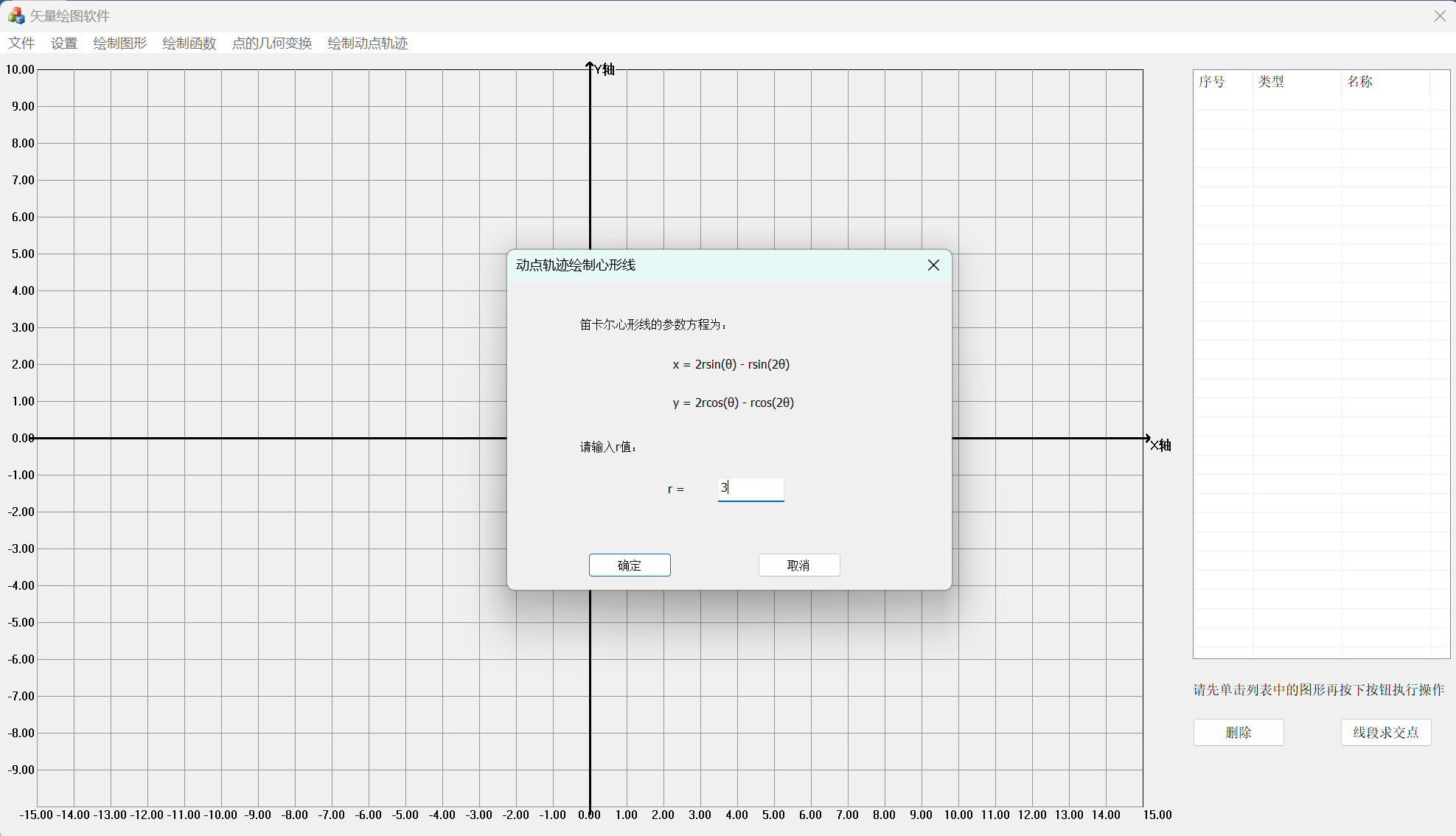
正在绘制双曲线



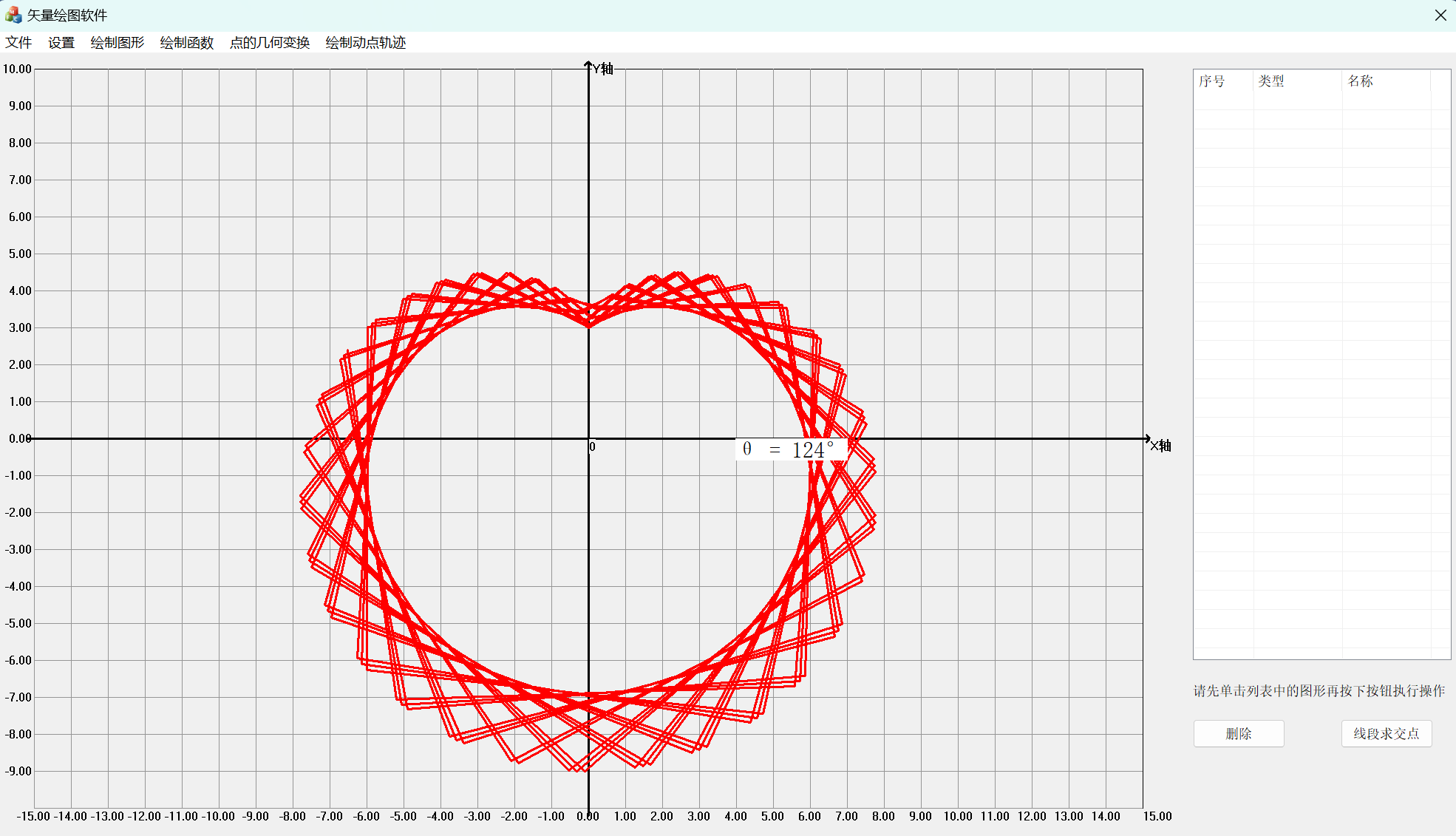
双曲线绘制成功



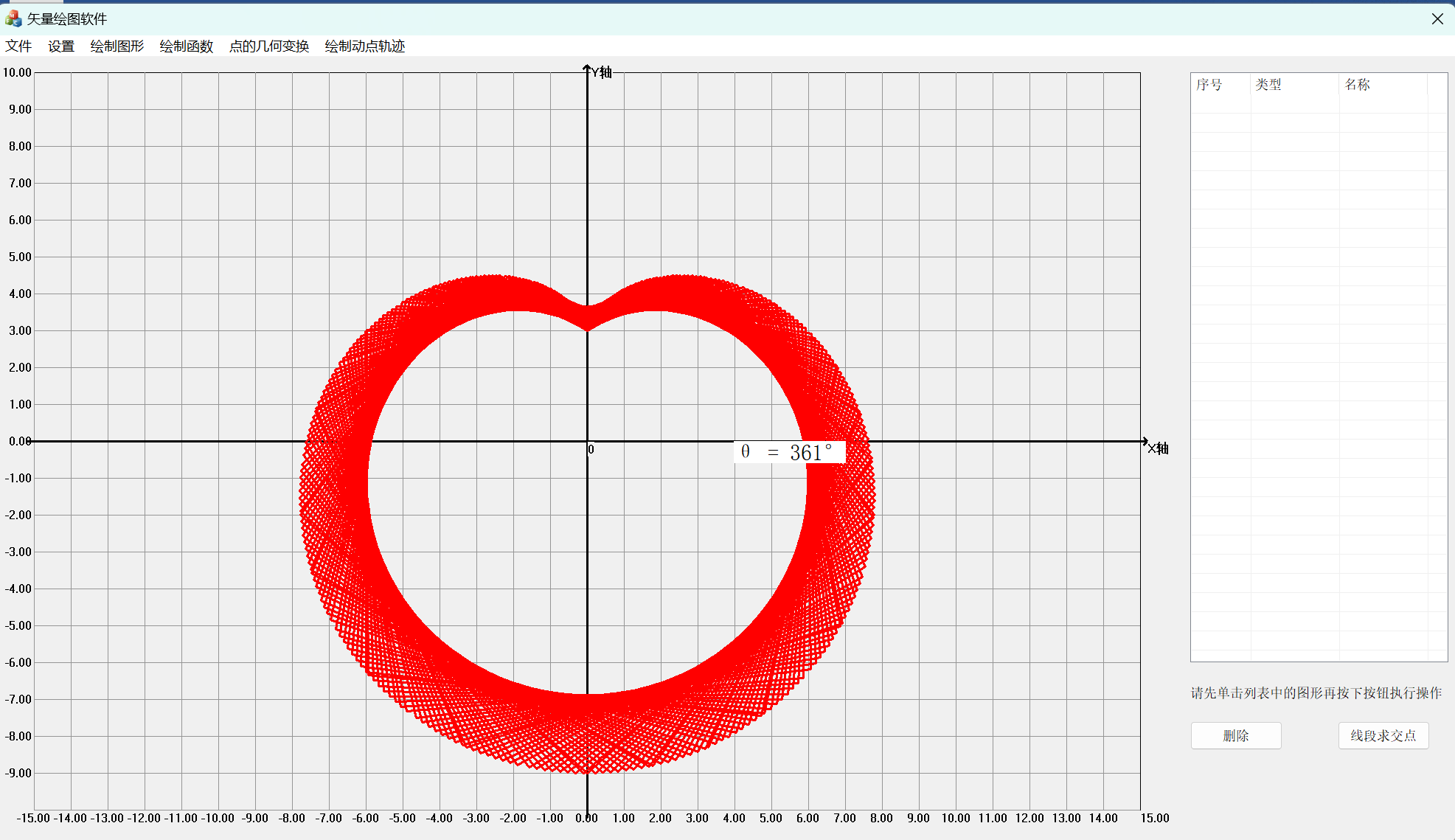
绘制心形线的动点轨迹



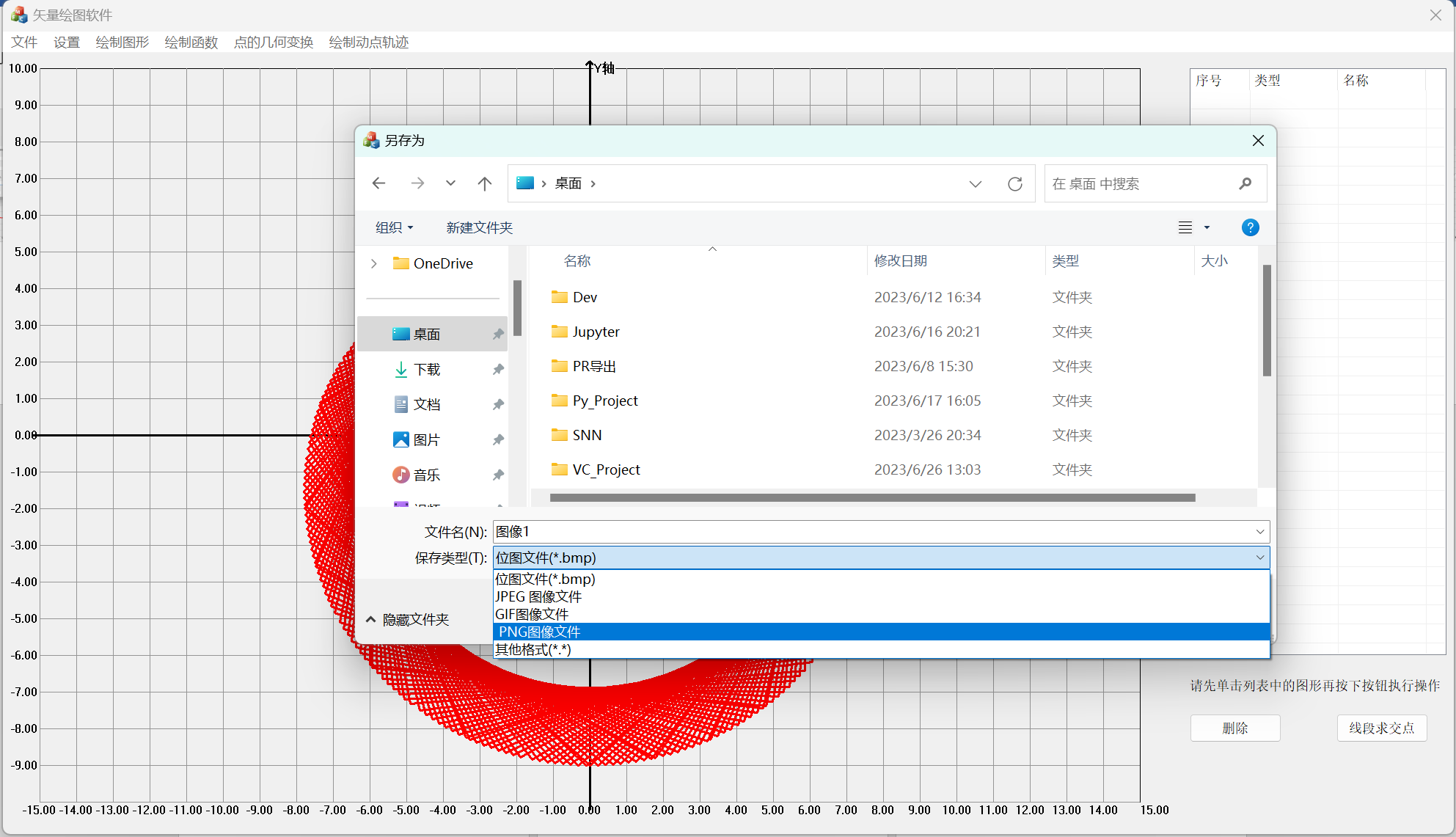
正在绘制心形线

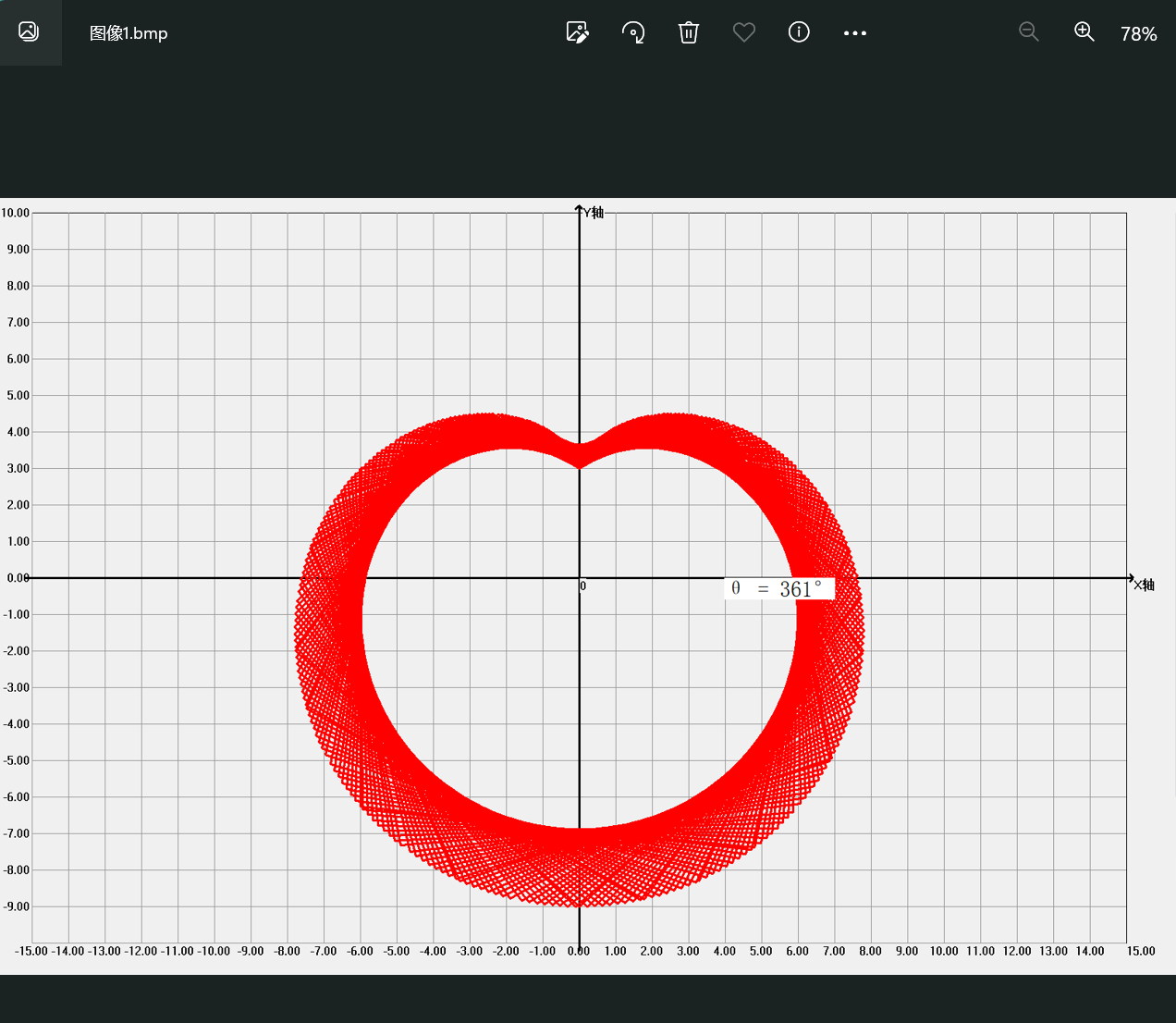


心形线绘制成功



将整个绘图区域导出图片到桌面，名称为图像1，格式为png



图片导出成功

## 总结

在本次的课程设计中，我设计了一个基于MFC的绘图软件，具备以下主要功能：

* 显示点的坐标：当用户双击绘图界面的任意点时，可以显示该点的坐标。
* 画笔设置：用户可以设置画笔的宽度、颜色、轮廓颜色和填充颜色。还可以选择画笔类型，如点线、实线、虚线等。
* 坐标网格绘制：绘图界面可以显示坐标网格，帮助用户定位和绘制图形。
* 点的绘制：用户可以绘制点，并可以根据设置的属性（如颜色、样式等）显示点。
* 线段绘制：用户可以绘制线段，通过指定起点和终点来绘制直线，并可以根据设置的属性进行显示。
* 多边形绘制：用户可以绘制多边形，通过连接多个点来创建自定义形状。
* 对象求交点：线段可以与线段、图形之间求交点，并在界面上显示交点的坐标。
* 点的几何变化：用户可以对点进行平移和关于点、线的对称变换操作。
* 方程的输入与曲线的生成：用户可以输入方程并生成相应的曲线，支持常见的数学函数和曲线方程。
* 对象属性设置：用户可以设置点的样式，提供了五种样式供选择，还可以自定义样式。
* 动点轨迹绘制：用户可以绘制椭圆、双曲线和笛卡尔心形线的轨迹。
* 导出图片：用户可以将绘制的图形以图片形式导出。

在数据结构设计方面，我使用了C++的继承和多态特性。定义了CMyData类作为父类，其中包含了图形的共有属性，如类型、名称和颜色。然后通过枚举类列举了五种图形类型，并定义了点、线、圆、多边形、函数五个类，它们继承自CMyData类，各自在构造函数中声明自己的类型。使用vector容器来存储CMyData类型的指针，实现了对各种图形数据的存储和操作。

绘图操作利用多态的特性，遍历存储图形数据的vector容器，根据每个元素的类型执行相应的绘图操作。

在设计过程中，考虑到扩展性和异常处理。设计具有良好的扩展性，可以方便地添加新的图形类型或功能模块。并在处理用户输入和图形操作时加入了适当的异常处理机制，以处理可能出现的错误情况。

综上所述，本次设计的基于MFC的绘图软件具备了丰富的功能和灵活的数据结构，可以满足用户的绘图需求，并提供良好的用户体验。