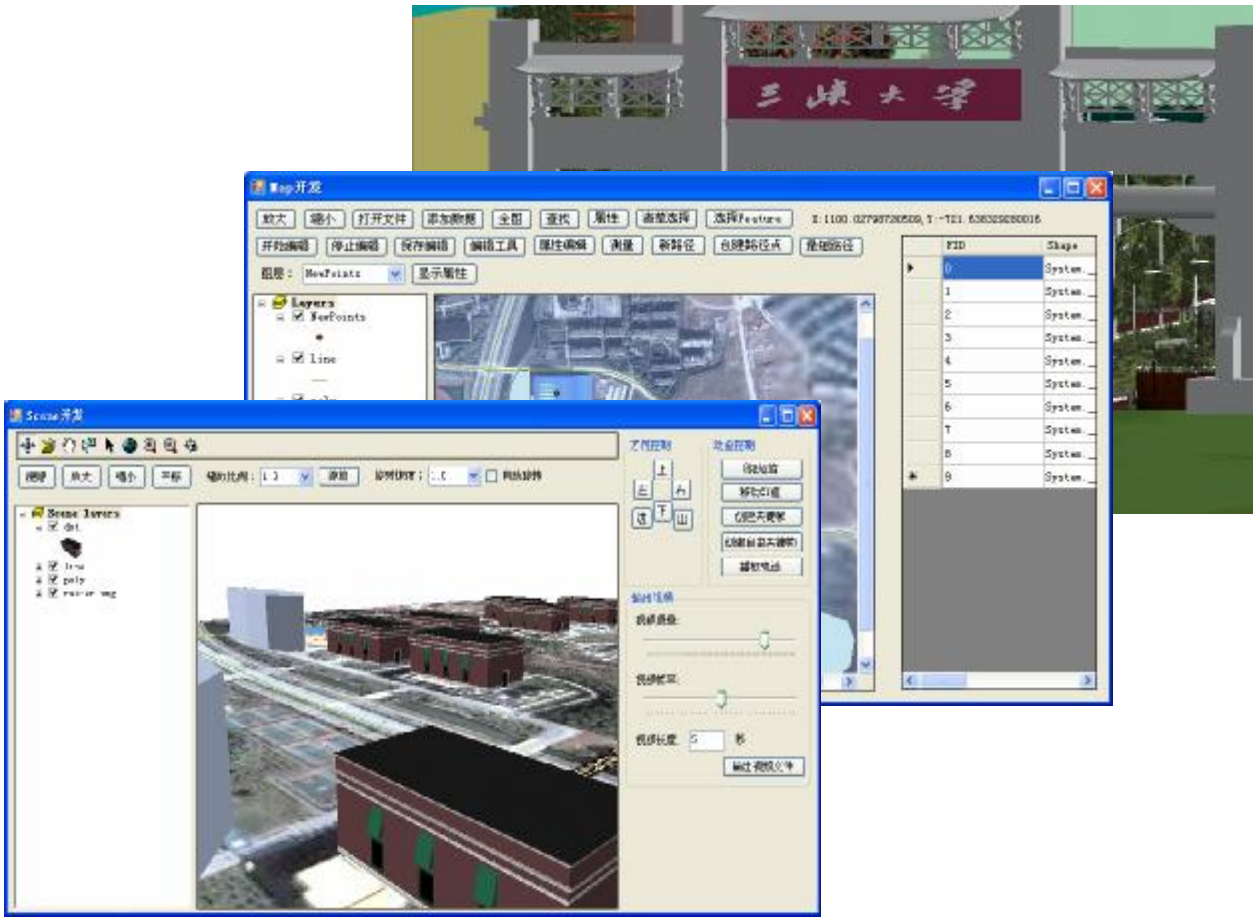
三峡大学

ArcGIS Engine 开发实例教程

三峡大学土木水电学院　肖泽云



二〇〇九年 三月

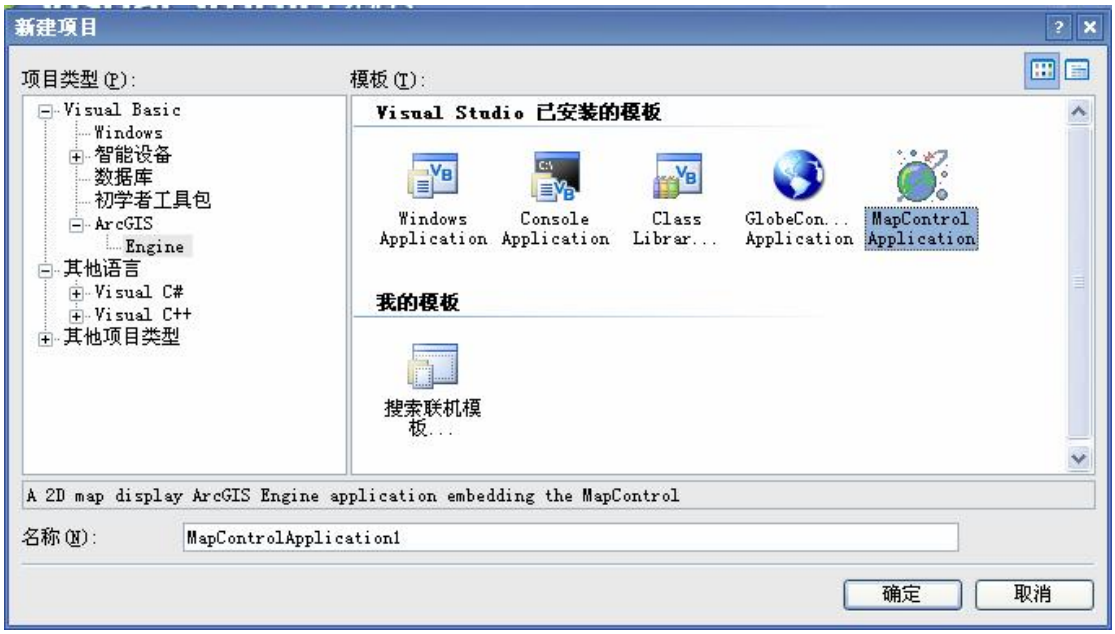
很高兴和大家分享我的心得！ArcGIS Engine 是专门用于开发 GIS 系统的开发包，目前最新版本是 9.3。基于 ArcGIS Engine，结合程序开发语言可以开发具有很强专业性的 GIS 系统。而且所开发的系统不需要客户端安装 ArcGIS 软件，只需要安装 ArcGIS Engine Runtime 及相应的许可就可以使用开发的 GIS 系统。本教程是本人原创作品，其主要目的是能让读者在学习完本教程后，对 ArcGIS Engine 开发有基础的认识，对 ArcGIS Engine 主要的库有一定了解，熟悉 GIS 开发的基本流程，以期得到抛砖引玉的效果。本教程中所有的程序代码都是作者一句一句编写的，希望广大读者、特别是初学者一定要仔细阅读、认真思考、反复斟酌，才会达到事半功倍的效果。限于作者水平及时间关系，其中的错误在所难免，望广大读者提出宝贵的意见和建议！相互交流，共同提高！

—— 肖泽云

作者联系方式：
地址：湖北省宜昌市三峡大学土木水电学院
邮编：443002
Email：xwebsite@163.com
QQ：289700062

软件建议配置要求：

由于在本教程中主要以 Visual Basic.Net 语言开发为主，所以要求读者在使用本教程之前先安装 Visual Studio2005。然后安装 ArcGIS Engine for .Net，安装完 ArcGIS Engine 后安装 Engine 的许可文件，在启动 Visual Studio 时会出现如下所示界面：



其中 MapControl Application 是一个典型的二维地图模版。

目　录

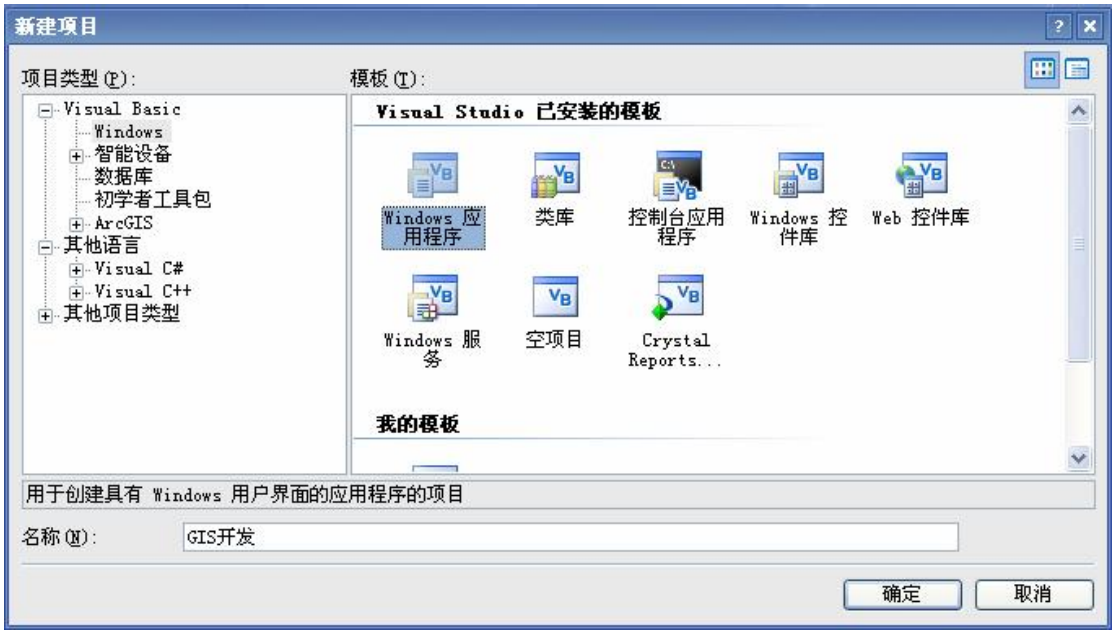
第一篇 Map 地图开发.....1
1、新建一个 GIS 项目.....1
2、导入 Map 文档.....5
3、设置 TOCControl 和 ToolbarControl 控件的链接.....7
4、添加 Shapefile 数据文件.....7
5、添加图层.....8
6、其他常用属性及方法.....9
7、地图放大缩小等.....10
8、创建数据属性表格.....12
9、缓冲分析.....15
10、CAD 与 Shapfile 格式之间的相互转换.....20
11、将 CAD 导出成 Shp 格式.....23
12、创建符号选择窗体 SymbologyControl.....23
13、指定对象的符号.....27
14、三维分析.....31
15、获取 TIN 图层.....40
16、选择对象.....42
17、获取选择数据.....45
18、获取剖面图.....49
19、闪烁动画并缩放至数据.....62
20、更改点的显示符号.....67
21、更改线的显示符号.....69
22、自定义点的符号.....71
23、打开个人数据库.....72
24、打开 CAD 数据.....74
25、计算 Raster 的绝对值.....77
26、Raster 的复杂计算.....79
27、生成表面 Raster.....82

28、生成坡度 Raster.....83
29、生成等高线.....84
30、生成曲率 Raster.....85
第二篇 **Scene** 三维开发.....87
1、创建一个简单的三维场景.....87
2、在 SceneControl 控件中添加 Scene 文件.....88
3、旋转/移动摄像机动画.....88
4、常用浏览功能.....88
5、更改摄像机位置.....90
6、输出 AVI 动画.....91
7、创建动画关键帧.....92
8、在三维控件 SceneControl 中添加 Shapfile、jpg、tif 等格式文件.....95
9、保存场景图片.....96
10、添加 Raster 数据.....96
11、生成 TIN.....97
12、设置图层基准高程.....107
13、点击查询.....115
14、转换 TIN 成 MultiPatches.....120
15、获取 TIN 中对象.....121
16、输出 TIN 节点坐标.....124
17、设置点的模型.....127

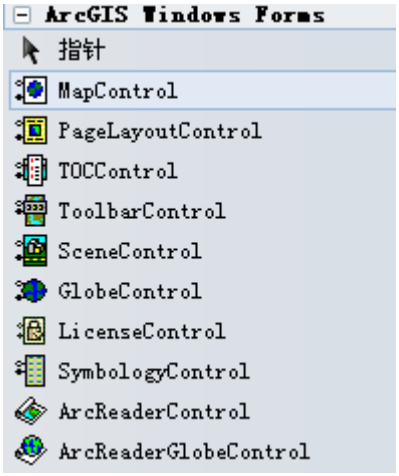
第一篇 Map 地图开发

1、新建一个 GIS 项目

1）启动 Visual Studio，在 Visual Basic 下面的 Windows 模版下选择 Windows 应用程序，并命名项目名称为“GIS 开发”，如下图所示。



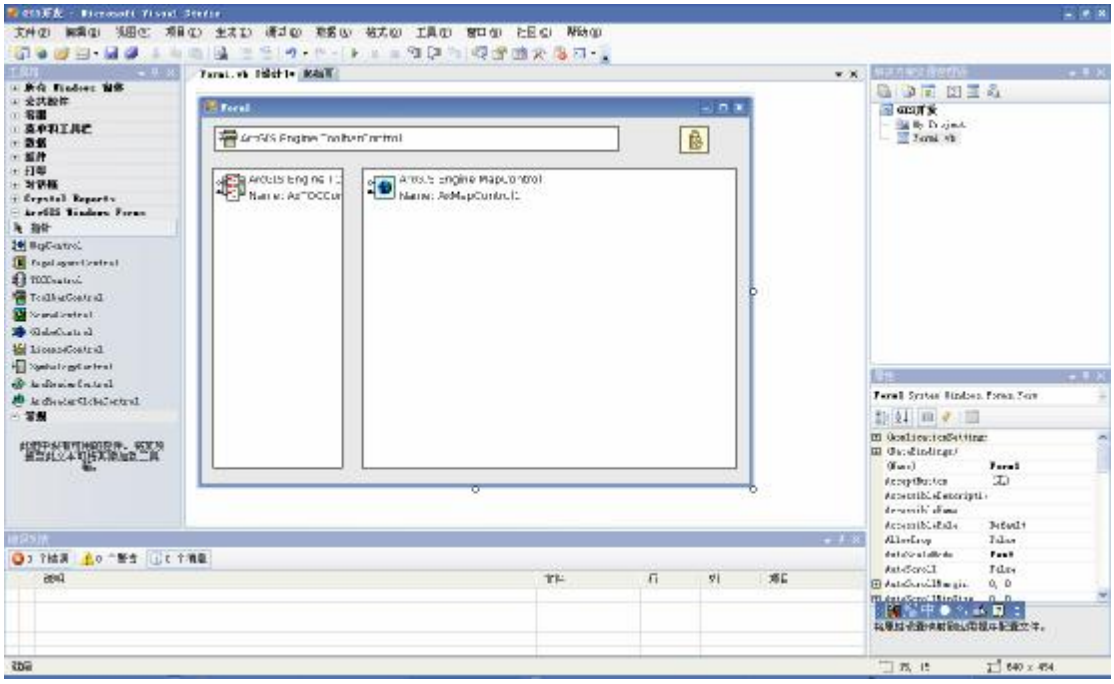
2）点击确定按钮，创建一个 Windows 应用程序项目。在工具栏中的 ArcGIS Windows Forms 选项下有一些常用的 GIS 控件，如下图所示：



其中，MapControl 就是 Map 地图控件，PageLayoutCOntrol 是布局地图控件，TOCControl 是目录控件，ToolbarControl 是 GIS 工具栏控件，SceneControl 是 Scene 三维场景控件，GlobeControl 是 Globe 控件，LicenseControl 是许可控件，SymbologyControl 是符号选择器控件，ArcReaderControl 是 ArcReader 控件，ArcReaderGlobeControl 是 ArcReaderGlobe 控件。常用的有 MapControl、SceneControl、ToolbarControl 、TOCControl 和 LicenseControl。LicenseControl 是许可控件，一般 GIS 系统中都必须添加，否则无法使用。

3）下面我们添加 MapControl、ToolbarControl 、TOCControl 和 LicenseControl

这四个控件，如下图所示：



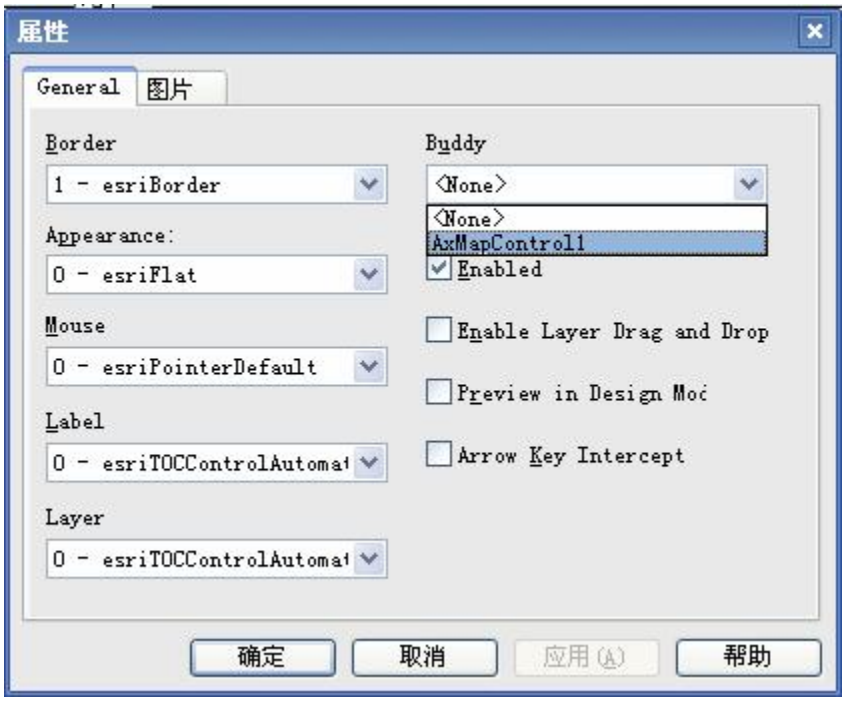
4）这些控件也有一些属性性质，可以直接在控件上点击右键打开控件的属性窗体，如下图所示：



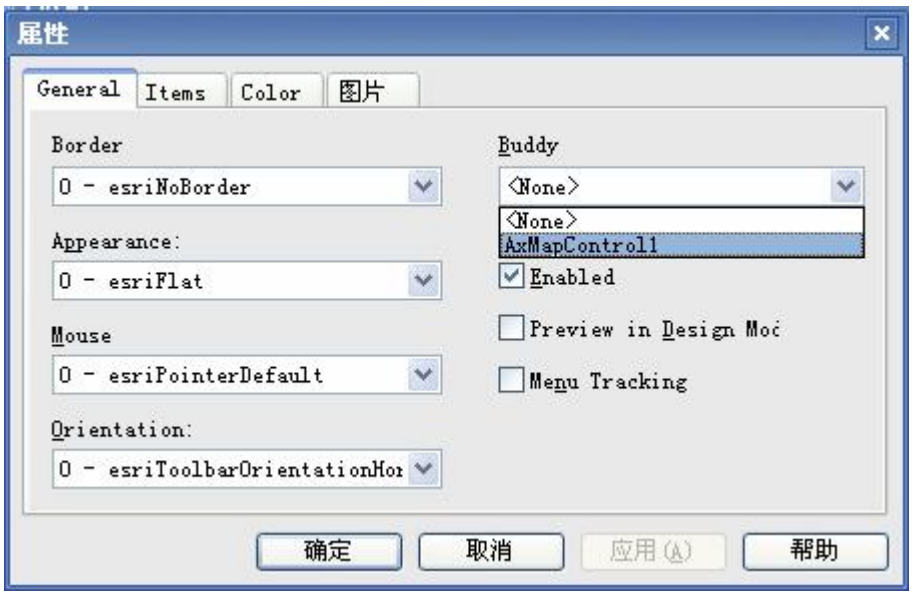
5）打开 MapControl 控件的属性，在 Map Document 后可以浏览 Map 文件的地址，在启动程序后将会打开该地图。如下图所示：



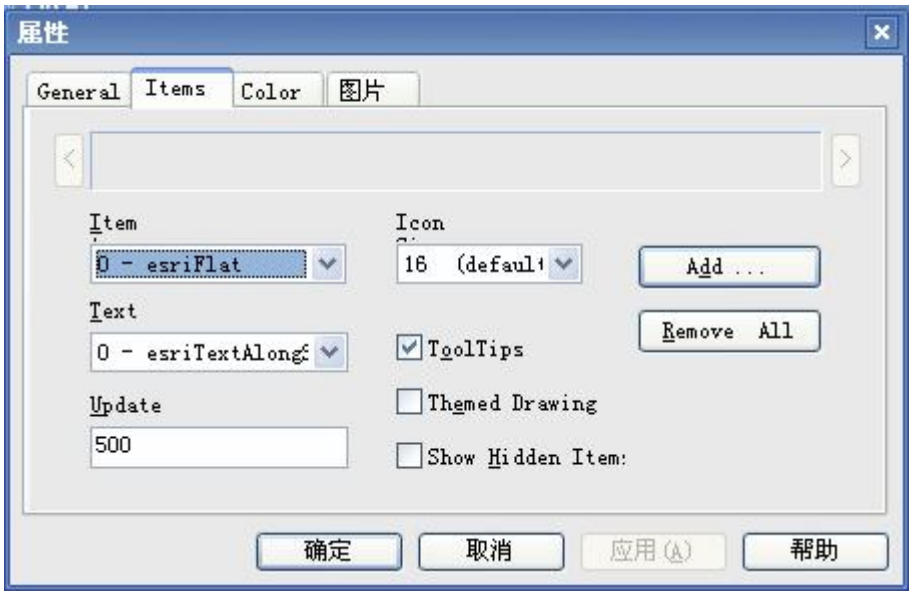
6）点击确定按钮。然后打开 TOCControl 控件的属性窗体，在 Buddy 下设置其链接到 AxMapControl 控件，如下图所示，表示该目录对应的是 AxMapControl1 中地图的数据文件列表。



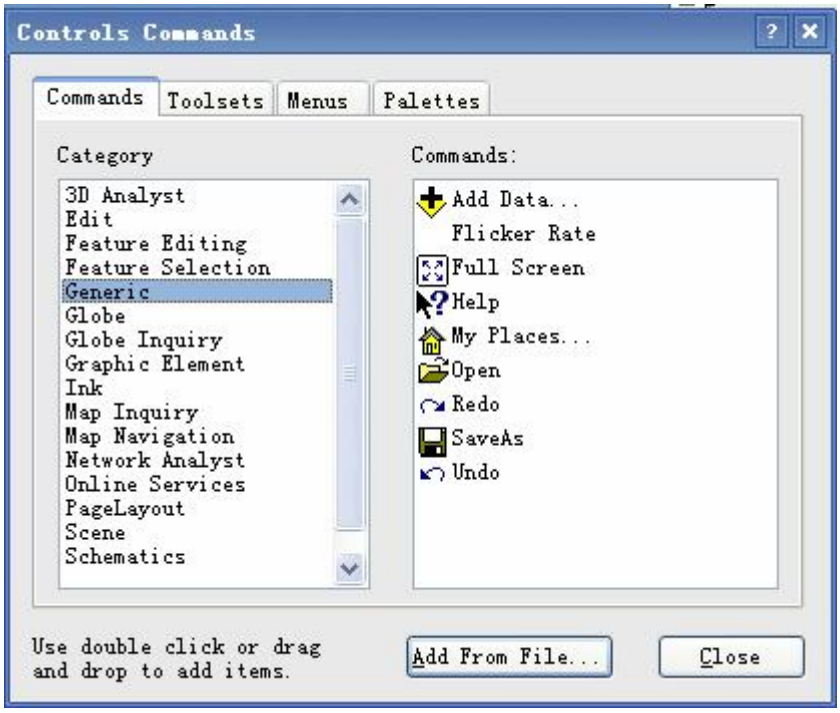
7）点击确定按钮。然后打开 ToolbarControl 控件的属性窗体，同样在 Buddy 下设置其链接到 AxMapControl 控件，如下图所示：



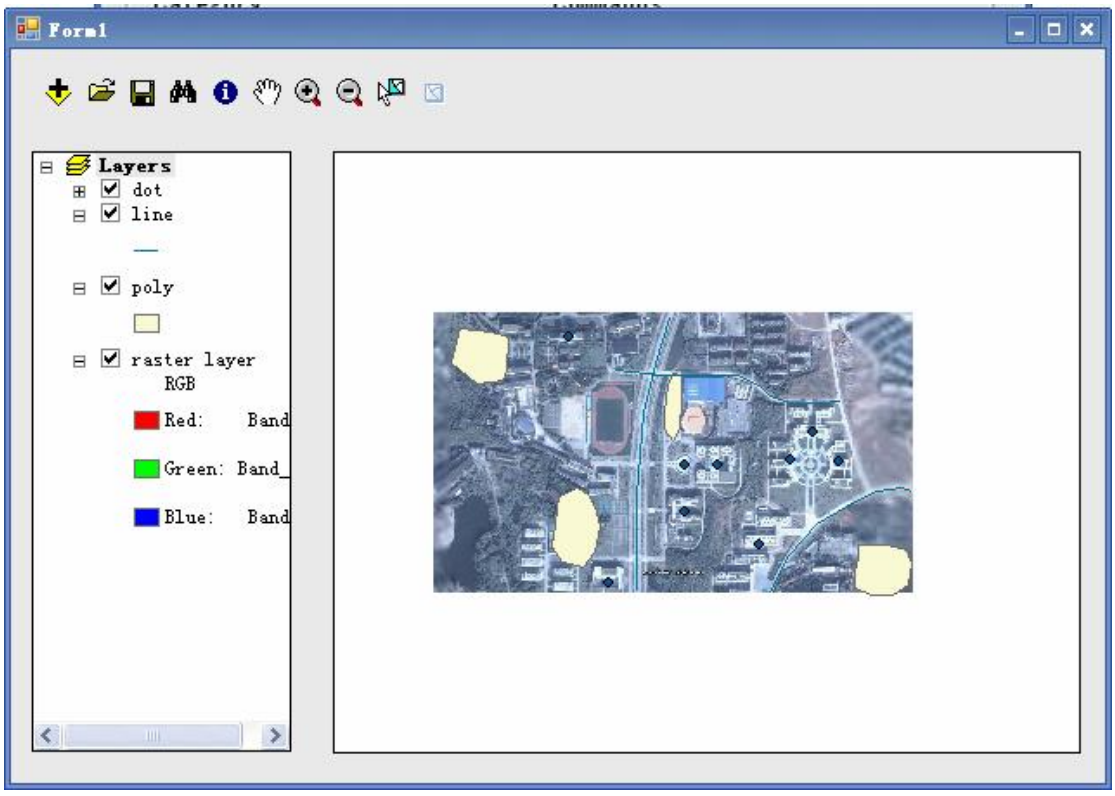
8）点击 Items 标签，在此可以添加工具，如下图所示，点击 Add 按钮开始添加工具。



9）如下图所示，在添加工具窗体上可以选择需要添加的工具，若需要添加某个工具，在双击该工具。



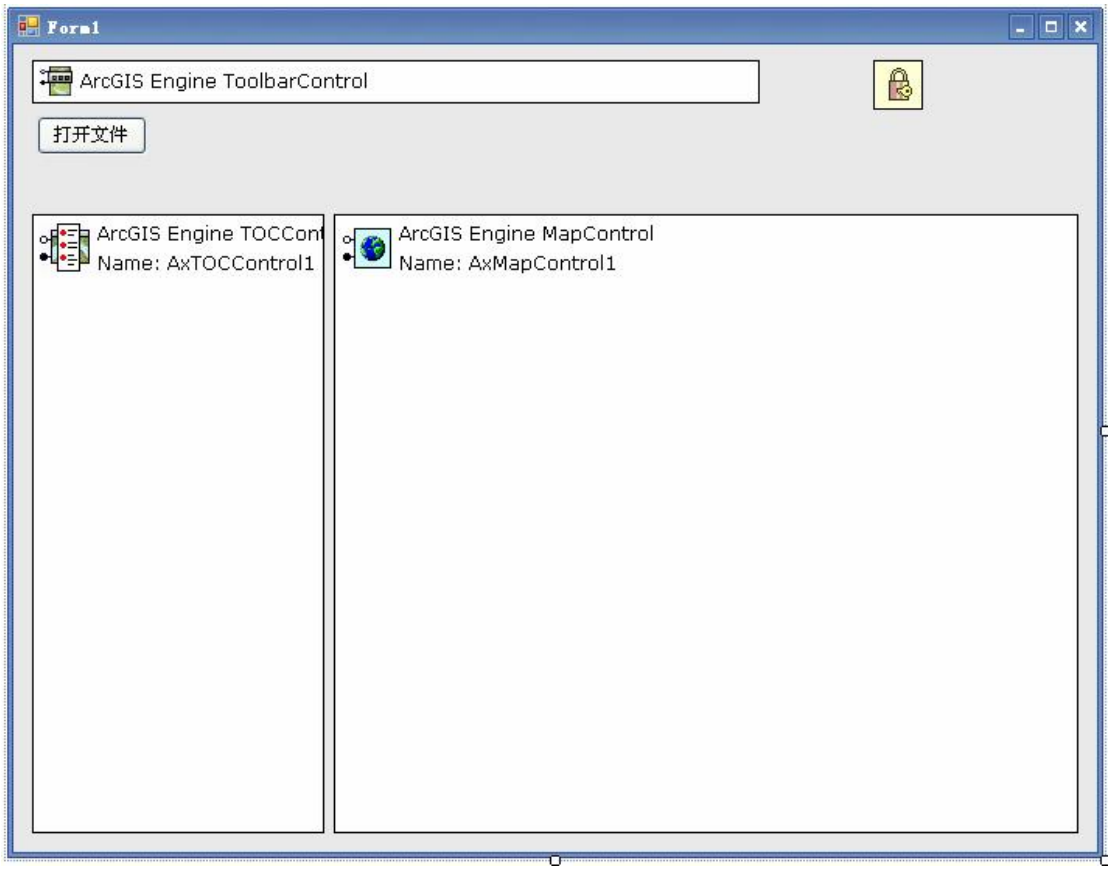
10）添加完工具后关闭选择工具窗体，在 ToolbarControl 控件的属性窗体上点击确定按钮。按 F5 开始运行程序，其现实结果如下图所示：



至此，一个简单的 GIS 系统就形成了。该系统没有编写任何代码就实现了，可见 ArcGIS Engine 开发还是比较人性化的。接下来，我们基于这个项目继续开发。

2、导入 Map 文档

1）在 Form1 窗体上添加一个按钮控件，并设置其 Text 为“打开文件”，如下图所示：



2）双击该按钮，为该按钮添加事件，其代码如下：

```
        Dim openFileDialog1 As OpenFileDialog = New OpenFileDialog()
        openFileDialog1.Title = "打开Map 地图"
        openFileDialog1.Filter = "Map Documents (*.mxd, *.mxt,
*.pmf)|*.mxd;*.mxt;*.pmf"
        openFileDialog1.ShowDialog()

        Dim sFilePath As String
        sFilePath = openFileDialog1.FileName

        If sFilePath = "" Then Exit Sub

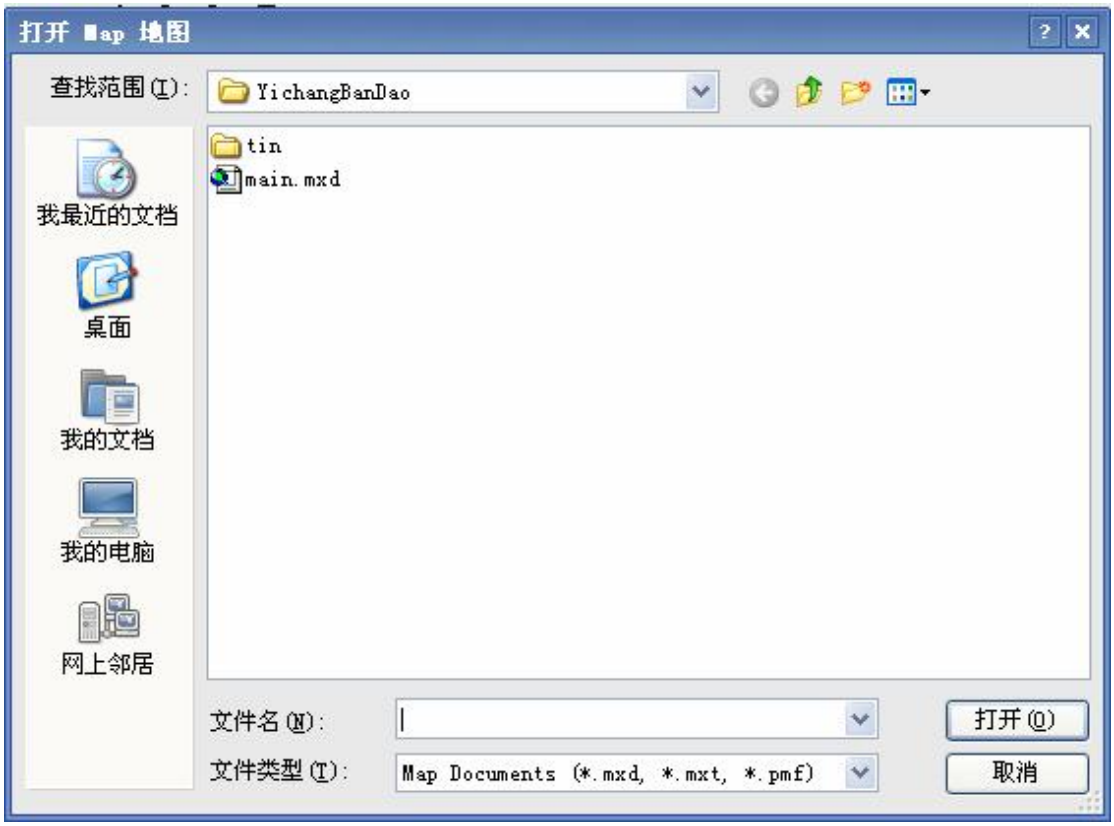
        On Error GoTo ErrorHandler_LoadMxFile
        AxMapControl1.LoadMxFile(sFilePath)
        Exit Sub

ErrorHandler_LoadMxFile:
        MsgBox(sFilePath & "不能导入！" + vbCrLf + "错误:" +
Err.Description)
```

其中该代码中最主要的是 AxMapControl1.LoadMxFile(sFilePath)，MapControl 的 LoadMxFile 方法可以将指定路径 sFilePath 的 Map 文件导入到

MapControl 控件中去。

3) 运行程序，点击按钮，其运行结果如下图所示：



3、设置 TOCControl 和 ToolbarControl 控件的链接

TOCControl 和 ToolbarControl 控件分别是用于链接 Map 或 Scene 控件的目录控件和工具栏控件，它们与 Map 或 Scene 控件链接的方法除了前面的设置其属性外，还可以通过在程序中设置它们的 SetBuddyControl 属性。一般都是在程序启动或窗体初始化时就设置，如下代码：

```
AxTOCControl1.SetBuddyControl(AxMapControl1)
AxToolbarControl1.SetBuddyControl(AxMapControl1)
```

这就是设置 TOCControl 和 ToolbarControl 控件链接到 Map 控件上的代码。

4、添加 Shapefile 数据文件

1）添加一个按钮控件，并设置其 Text 属性为“添加 Shp 文件”。

2）双击该按钮，为其添加事件，其代码如下:

```
Dim addDataDialog1 As OpenFileDialog = New OpenFileDialog()
addDataDialog1.Title = "添加Shapefile数据文件"
addDataDialog1.Filter = "Shapfiles (*.shp)|*.shp"
addDataDialog1.ShowDialog()

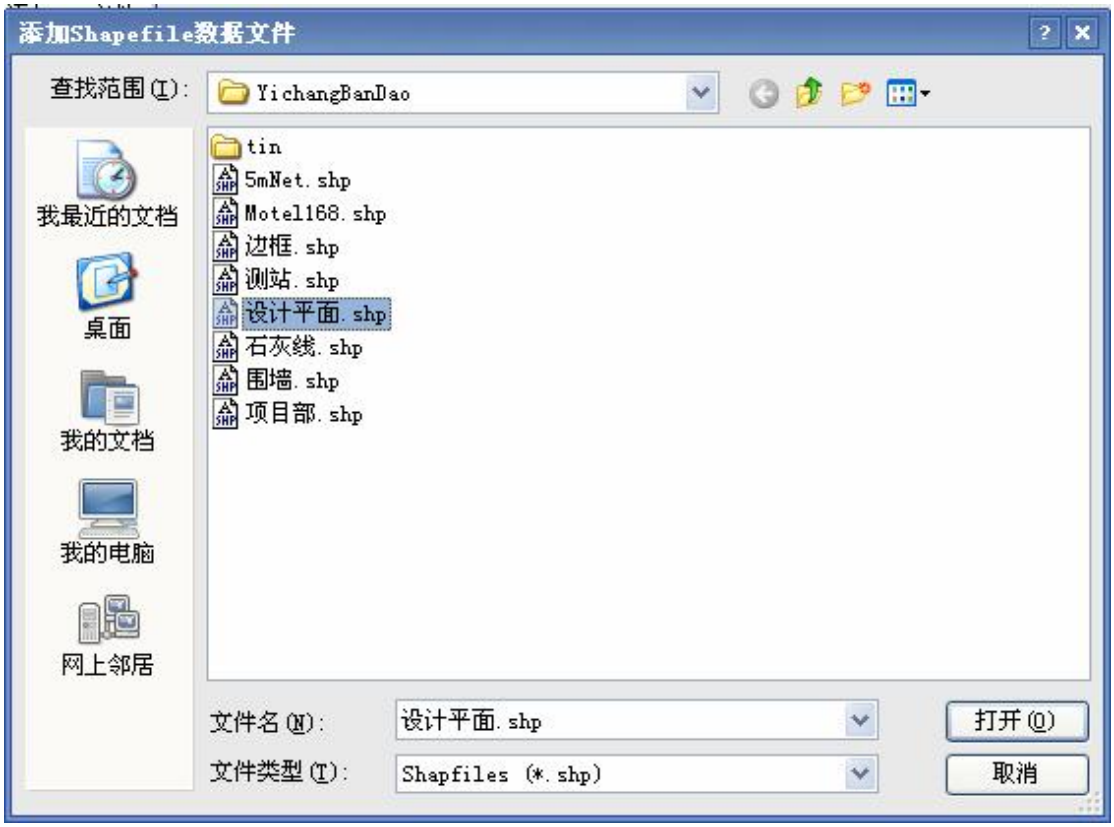
Dim sFilePath As String
sFilePath = addDataDialog1.FileName
```

```
        If sFilePath = "" Then Exit Sub
        On Error GoTo ErroHandle_AddData
        Dim i As Integer
        i = sFilePath.LastIndexOf("\")

AxMapControl1.AddShapeFile(Microsoft.VisualBasic.Left(sFilePath, i),
Microsoft.VisualBasic.Right(sFilePath, sFilePath.Length - i - 1))
        Exit Sub
ErroHandle_AddData:
        MsgBox(sFilePath & "不能导入！" + vbCrLf + "错误:" +
Err.Description)
```

其中 AxMapControl1.AddShapeFile 是添加 shapefile 的方法，后面的是获取该文件的路径和名称。

3）运行程序，其结果如下图所示：



5、添加图层

1）添加一个按钮控件，并设置其 Text 属性为“添加图层”。

2）双击该按钮，为其添加事件，其代码如下:

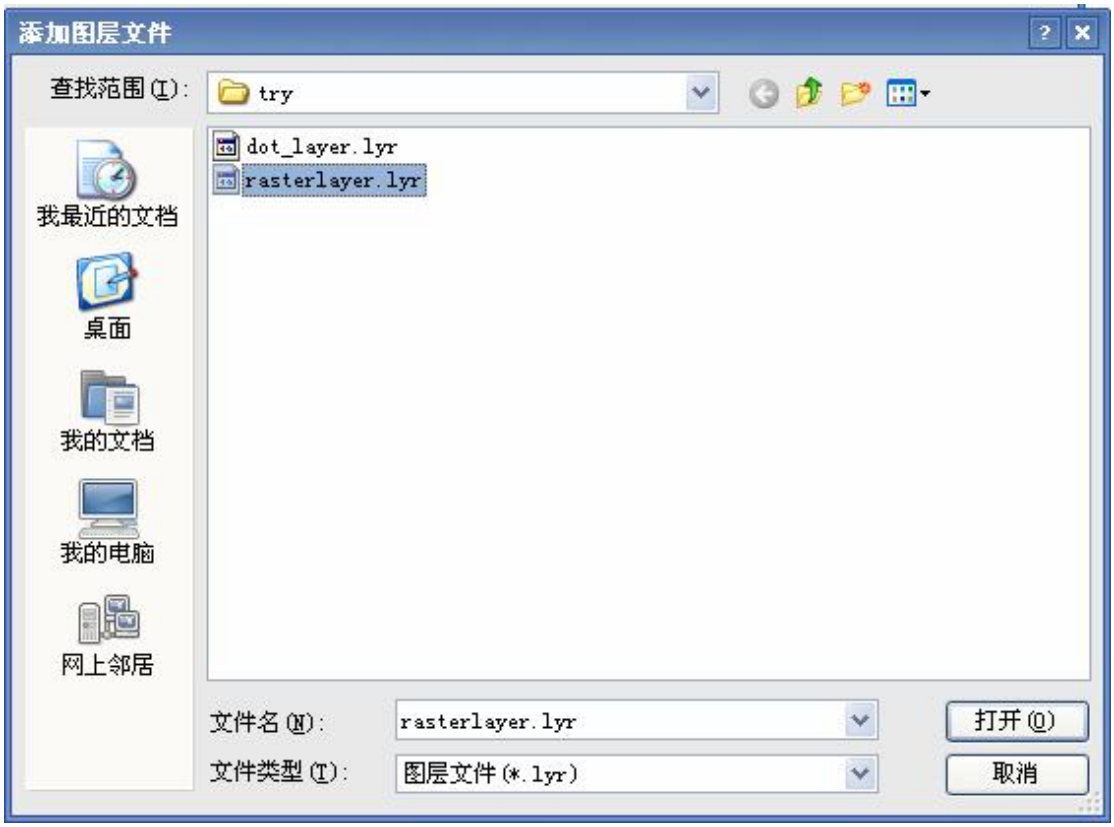
```
        Dim addDataDialog1 As OpenFileDialog = New OpenFileDialog()
        addDataDialog1.Title = "添加图层文件"
        addDataDialog1.Filter = "图层文件(*.lyr)|*.lyr"
        addDataDialog1.ShowDialog()
```

```
        Dim sFilePath As String
        sFilePath = addDataDialog1.FileName

        If sFilePath = "" Then Exit Sub
        On Error GoTo ErroHandle_AddData
        AxMapControl1.AddLayerFromFile(sFilePath)
        Exit Sub
ErroHandle_AddData:
        MsgBox(sFilePath & "不能导入！" + vbCrLf + "错误:" +
Err.Description)
```

其中 AxMapControl1.AddLayerFromFile 为添加图层代码。

3）运行程序，其最终结果如下图所示：



6、其他常用属性及方法

1）获取图层数目

```
MsgBox("共有：" + AxMapControl1.LayerCount.ToString() + "个图层")
```

其中 AxMapControl1.LayerCount 是获取 Map 中图层的个数。

2）控制鼠标滚轮缩放

```
        If AxMapControl1.AutoMouseWheel Then
            AxMapControl1.AutoMouseWheel = False
        Else
            AxMapControl1.AutoMouseWheel = True
```

```
        End If
```

3）删除图层

```
AxMapControl1.DeleteLayer(1)
```

4）移动图层顺序

```
AxMapControl1.MoveLayerTo(1, 2)
```

5）平移地图

```
AxMapControl1.Pan()
```

6）获取到地图中的图层

```
        Dim player As ESRI.ArcGIS.Carto.ILayer
        player = AxMapControl1.get_Layer(0)
```

7）获取鼠标当前坐标

在 Map 控件的 OnMouseMove 事件中可以获取，如 AxMapControl1_OnMouseMove 事件中添加如下代码：

```
Label1.Text = "X:" + e.mapX.ToString() + ",Y:" + e.mapY.ToString()
```

值得注意的是：mapX 及 mapY 代表地图坐标，x 和 y 代表窗口坐标。

8）拖选视图范围

在 Map 控件的 OnMouseDown 事件中可以获取，如 AxMapControl1_OnMouseDown 事件中添加如下代码：

```
            Dim pEn As ESRI.ArcGIS.Geometry.IEnvelope = New
ESRI.ArcGIS.Geometry.Envelope()
        pEn = AxMapControl1.TrackRectangle()
        AxMapControl1.Extent = pEn
```

则可以设置视图的范围为拖选的矩阵范围。

9）设置视图中心

```
        Dim i As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
        i.PutCoords(100, 100)
        AxMapControl1.CenterAt(i)
        AxMapControl1.Refresh()
```

其中 i.PutCoords(100, 100)是设置中心在（100，100），若该点为某个数据的中心，则该功能就是以该数据为中心。

7、地图放大缩小等

放大、缩小、打开文件、保存文件等 ArcMap 中长用的命令在 ArcGIS Engine 中也可以找到对应的命令，这些命令都在 controls 库中。所以首先必须引用 ESRI.ArcGIS.Controls。下面以放大为例：

在项目中添加 ESRI.ArcGIS.Controls 的引用，然后在全局定义前引用，如下：

```
Imports ESRI.ArcGIS.Controls
```

在放大按钮上添加如下代码：

```
 Private Sub ZoomIn_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles ZoomIn.Click
        Dim zoomin As ControlsMapZoomInTool = New
```

```
ControlsMapZoomInTool()
        zoomin.OnCreate(AxMapControl1.Object)
        AxMapControl1.CurrentTool = zoomin
       End Sub
```

其中 Dim zoomin As ControlsMapZoomInTool 是定义地图放大命令，zoomin.OnCreate 是创建放大命令，AxMapControl1.CurrentTool = zoomin 是使 Map 控件的当前命令按钮是 zoomin。

同理，还可以创建缩小、打开文件、保存文件等等。值得注意的是，有的时候是需要点击命令，有时候是使 Map 控件的当前命令为该命令。下面是各主要命令所对应的类：

功能	类	事件
放大	ControlsMapZoomInTool	设置 CurrentTool
缩小	ControlsMapZoomOutTool	设置 CurrentTool
打开文件	ControlsOpenDocCommand	OnClick()
添加数据	ControlsAddDataCommand	OnClick()
全图	ControlsMapFullExtentCommand	OnClick()
查找	ControlsMapFindCommand	OnClick()
属性工具	ControlsMapIdentifyTool	设置 CurrentTool
选择 Feature	ControlsSelectFeaturesTool	设置 CurrentTool
清除选择	ControlsClearSelectionCommand	OnClick()
开始编辑	ControlsEditingStartCommand	OnClick()
保存编辑	ControlsEditingSaveCommand	OnClick()
停止编辑	ControlsEditingStopCommand	OnClick()
编辑工具	ControlsEditingEditTool	设置 CurrentTool
属性编辑命令	ControlsEditingAttributeCommand	OnClick()
测量工具	ControlsMapMeasureTool	设置 CurrentTool
创建路径（网络分析）	ControlsNetworkAnalystRouteCommand	OnClick()
创建路径点	ControlsNetworkAnalystCreateLocationTool	设置 CurrentTool
最短路径查询	ControlsNetworkAnalystSolveCommand	OnClick()
Scene 缩小	ControlsSceneExpandFOVCommand	OnClick()

当然，还有很多很多其他的命令。从上面的列表比较我们也可以看出，一般来说，如果类的后缀是 Command，则用 OnClick 方法；如果是 Tool，则设置 Map 的 CurrentTool 属性为该工具。

8、创建数据属性表格

首先定义一个属性数据表格显示函数 ShowAttirbuteGrid（），其代码如下：

```
    Private Sub ShowAttirbuteGrid(ByVal pFeatureLayer As
ESRI.ArcGIS.Carto.IFeatureLayer, ByVal myDataGridView As DataGridView)
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass
        pFeatureClass = pFeatureLayer.FeatureClass

        Dim i As Integer
        Dim pQuery As ESRI.ArcGIS.Geodatabase.IQueryFilter = New
ESRI.ArcGIS.Geodatabase.QueryFilter()
        For i = 1 To pFeatureLayer.FeatureClass.Fields.FieldCount
            Dim s As String = pFeatureLayer.FeatureClass.Fields.Field(i
- 1).Name.ToString()
            myDataGridView.Columns.Add(s, s)
        Next

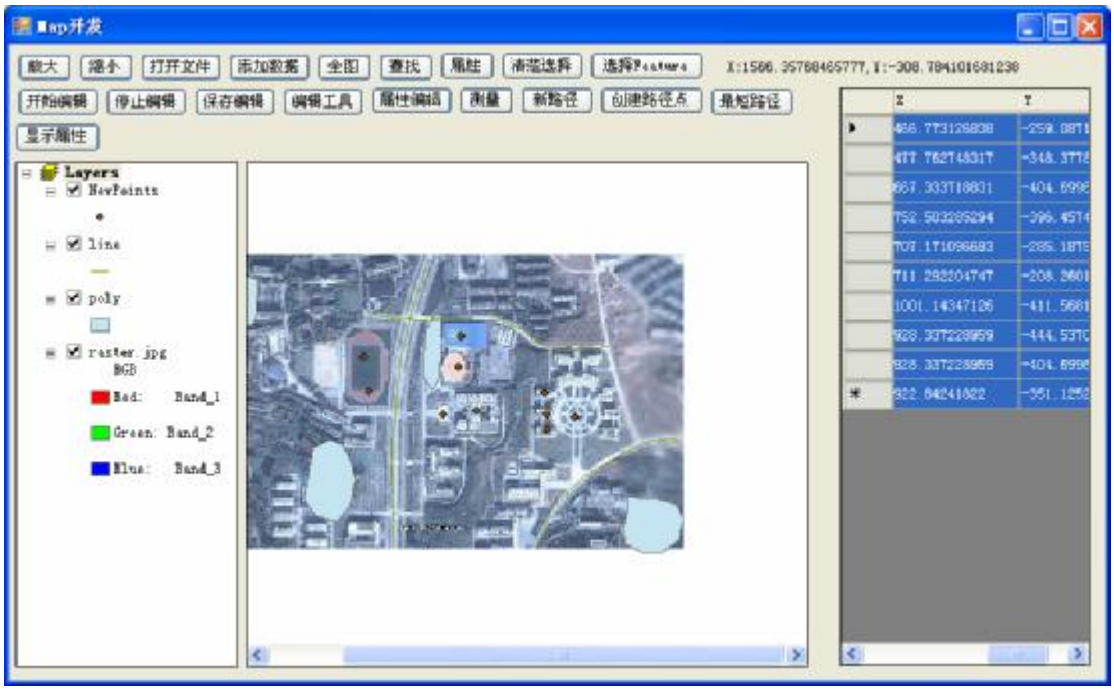

myDataGridView.Rows.Add(pFeatureLayer.FeatureClass.FeatureCount(pQuer
y) - 1)
        Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature
        Dim j As Integer
        For i = 0 To myDataGridView.Rows.Count - 1
            pFeature = pFeatureClass.GetFeature(i)
            For j = 0 To myDataGridView.Columns.Count - 1
                myDataGridView.Rows(i).Cells(j).Value =
pFeature.Value(j)
            Next
        Next
    End Sub
```

其中，需要说明的是：要获取某个数据文件中的某个数据的某个字段，首先获取到该数据层 Layer 或 FeatureLayer；然后定义 FeatureClass 数据集；通过 FeatureLayer 的 FeatureClass 属性获得 FeatureClass 数据集；在 FeatureClass 中包含了每个 Feature，通过 FeatureClass 的 GetFeature 属性就可以获得某个 ID 的 Feature；Feature 的属性中就包括有 Fields 属性，通过 Fields 可以找到每个 Field，Field 的 name 属性就是属性字段的名称；通过 Feature 的 Value 属性就可以获得某个字段值。

在按钮的 Click 事件中调用前面的函数，并添加一个 DataGridView 控件，代码如下：

```
        Dim player As ESRI.ArcGIS.Carto.ILayer
        player = AxMapControl1.Map.Layer(0)
        Dim pFeatureLayer As ESRI.ArcGIS.Carto.IFeatureLayer
        pFeatureLayer = player
        ShowAttirbuteGrid(pFeatureLayer, DataGridView1)
```

运行，最终显示结果如下图所示：



此外，还可以改进一下，添加一个 ComboBox 控件，在控件上添加图层的名称，直接点击显示属性按钮就可以显示 ComboBox 控件中相应的图层属性表格。首先需要更改一下 ShowAttirbuteGrid（）函数，如下：

```
    Private Sub ShowAttirbuteGrid(ByVal pFeatureLayer As
ESRI.ArcGIS.Carto.IFeatureLayer, ByVal myDataGridView As DataGridView)
        myDataGridView.Rows.Clear()
        myDataGridView.Columns.Clear()
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass
        pFeatureClass = pFeatureLayer.FeatureClass

        Dim i As Integer
        Dim pQuery As ESRI.ArcGIS.Geodatabase.IQueryFilter = New
ESRI.ArcGIS.Geodatabase.QueryFilter()
        For i = 1 To pFeatureLayer.FeatureClass.Fields.FieldCount
            Dim s As String = pFeatureLayer.FeatureClass.Fields.Field(i
- 1).Name.ToString()
            myDataGridView.Columns.Add(s, s)
        Next


myDataGridView.Rows.Add(pFeatureLayer.FeatureClass.FeatureCount(pQuer
y) - 1)
        Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature
        Dim j As Integer
        For i = 0 To myDataGridView.Rows.Count - 1
            pFeature = pFeatureClass.GetFeature(i)
```

```
            For j = 0 To myDataGridView.Columns.Count - 1
                myDataGridView.Rows(i).Cells(j).Value =
pFeature.Value(j)
            Next
        Next
    End Sub
```

然后，在 Form 的 Load 事件中先添加 ComboBox 的 Item，如下：

```
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim i As Integer
        For i = 0 To AxMapControl1.LayerCount - 1

ComboBox1.Items.Add(AxMapControl1.Map.Layer(i).Name.ToString())
        Next
        ComboBox1.Text = ComboBox1.Items(0)
    End Sub
```

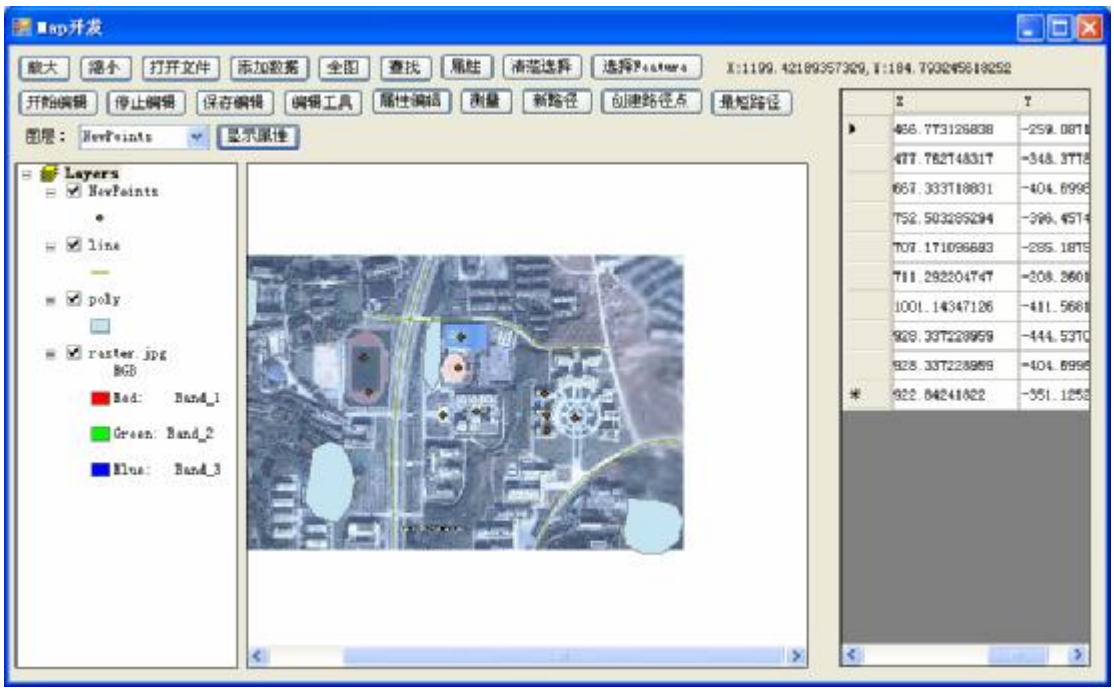
最后，更改显示属性按钮的 Click 事件，如下：

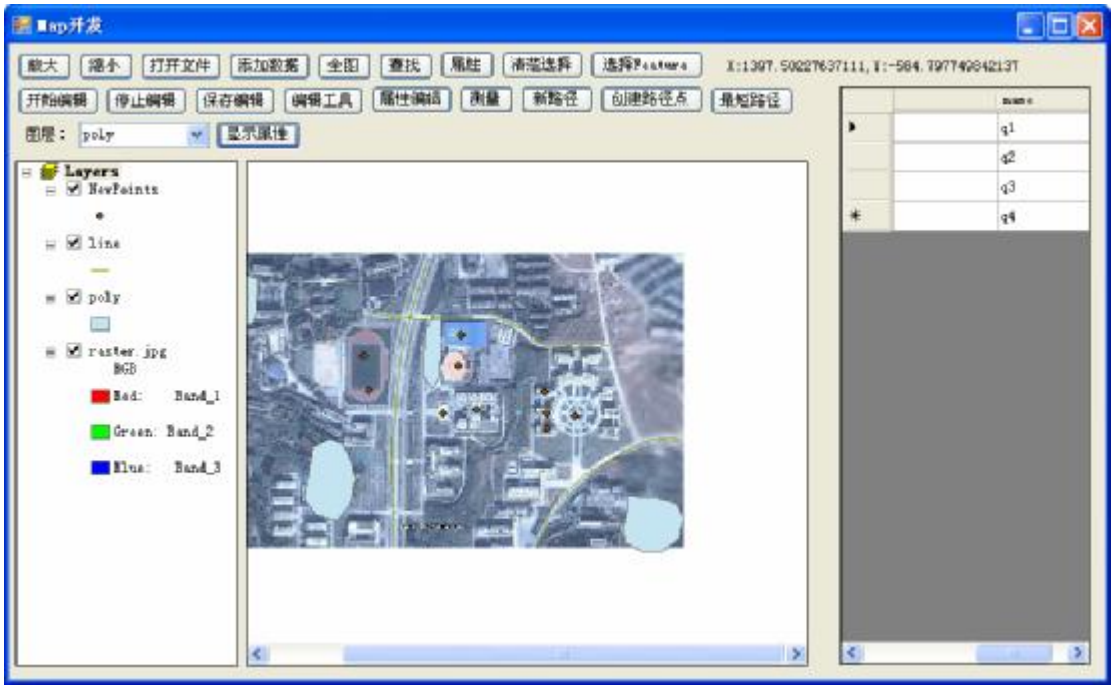
```
        Dim player As ESRI.ArcGIS.Carto.ILayer

        Dim i As Integer
        For i = 0 To AxMapControl1.LayerCount - 1
            If AxMapControl1.Map.Layer(i).Name = ComboBox1.Text Then
                player = AxMapControl1.Map.Layer(i)
                Exit For
            End If
        Next

        Dim pFeatureLayer As ESRI.ArcGIS.Carto.IFeatureLayer
        pFeatureLayer = player
        ShowAttirbuteGrid(pFeatureLayer, DataGridView1)
```

运行程序，最终显示结果如下所示：

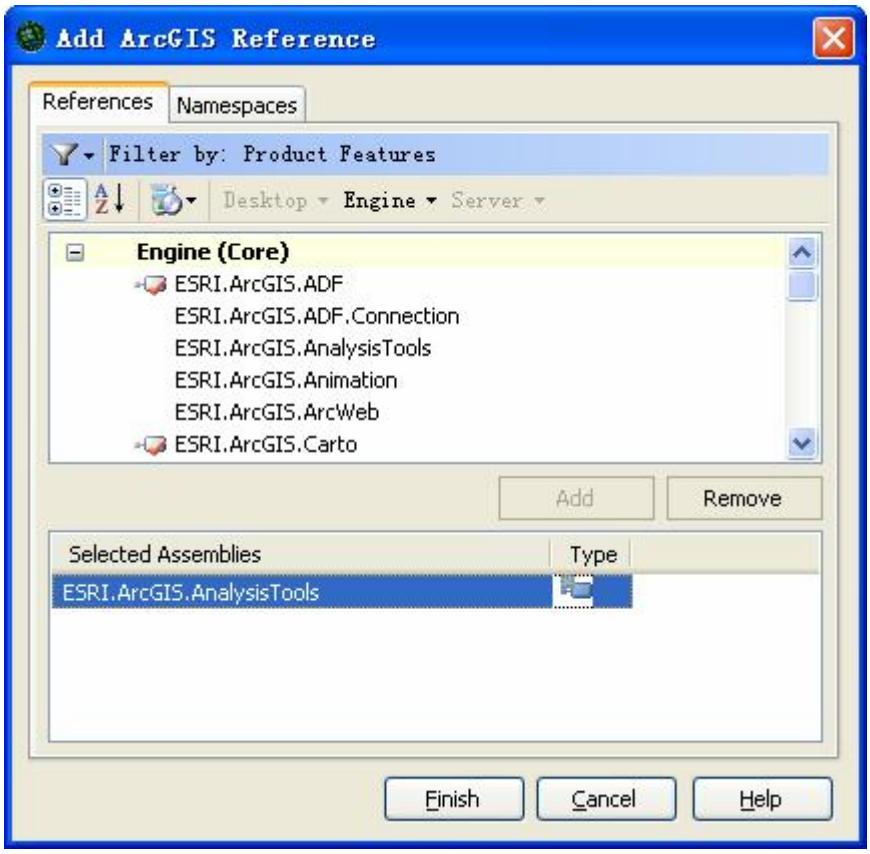


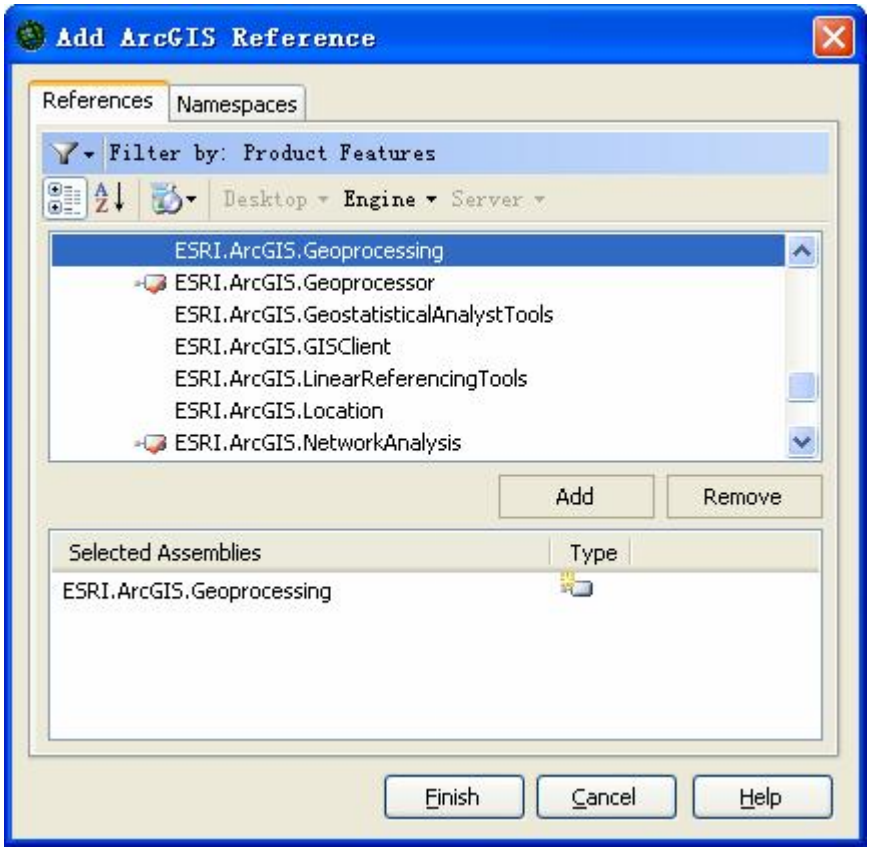


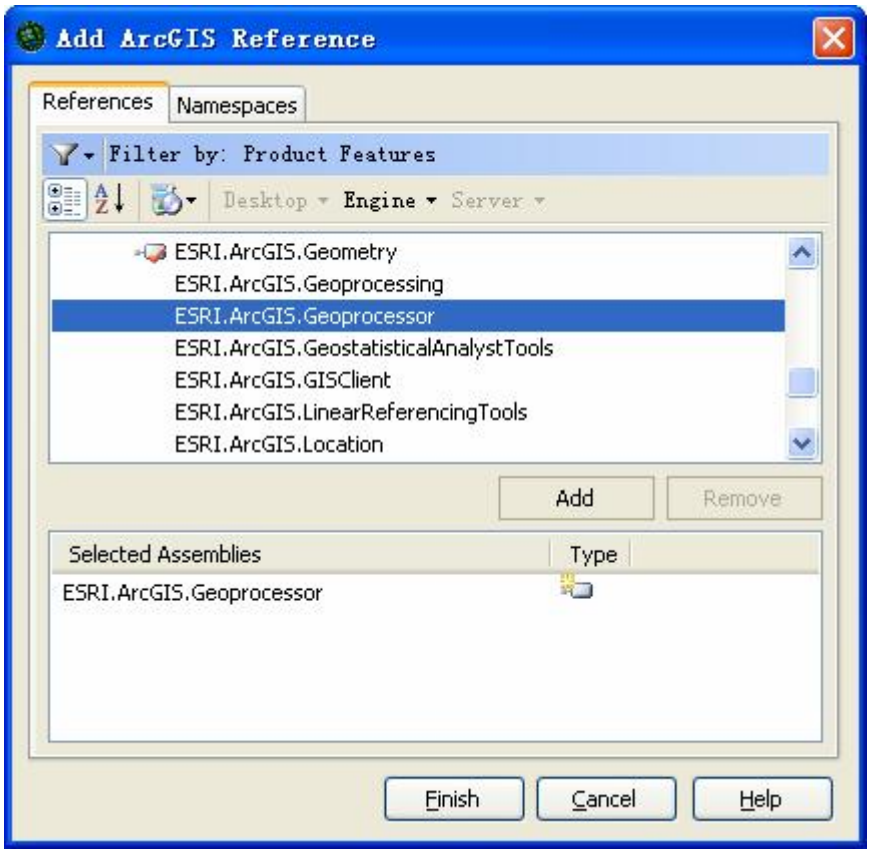
9、缓冲分析

本例主要用到了 Geoprocessor 处理工具。

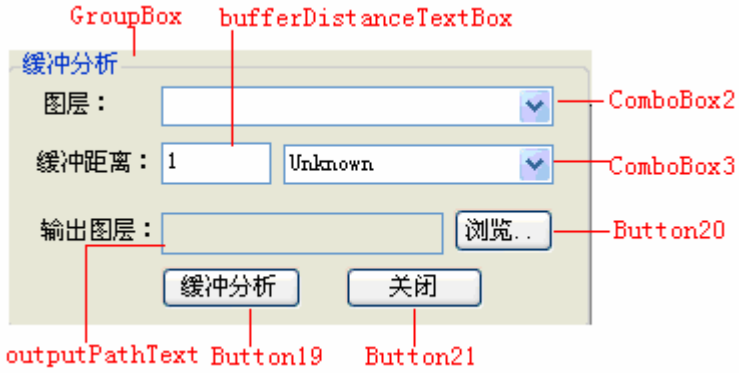
1）首先，添加引用 AnalysisTools、Geoprocessing 和 Geoprocessor 库，如下图所示：


Add ArcGIS Reference
References
Namespaces
Filter by: Product Features
Desktop
Engine
Server
Engine (Core)
ESRI.ArcGIS.ADF
ESRI.ArcGIS.ADF.Connection
ESRI.ArcGIS.AnalysisTools
ESRI.ArcGIS.Animation
ESRI.ArcGIS.ArcWeb
ESRI.ArcGIS.Carto
Add
Remove
Selected Assemblies
Type
ESRI.ArcGIS.AnalysisTools
Finish
Cancel
Help


Add ArcGIS Reference
References
Namespaces
Filter by: Product Features
Desktop
Engine
Server
ESRI.ArcGIS.Geoprocessing
ESRI.ArcGIS.Geoprocessor
ESRI.ArcGIS.GeostatisticalAnalystTools
ESRI.ArcGIS.GISClient
ESRI.ArcGIS.LinearReferencingTools
ESRI.ArcGIS.Location
ESRI.ArcGIS.NetworkAnalysis
Add
Remove
Selected Assemblies
Type
ESRI.ArcGIS.Geoprocessing
Finish
Cancel
Help



2）添加一个按钮用于缓冲分析，两个 Combox 控件，分别用来列出图层和缓冲数据的单位，添加一个 TextBox 用于输入缓冲距离，还有一个不可写的 TextBox 用于显示缓冲数据保存的路径，还有一个对应的浏览按钮用于浏览缓冲数据保存的路径，最后添加一个按钮用于关闭该缓冲的面板 GroupBox。如下图所示：



3）先为 ComboBox2 添加图层列表，在 Form 的 Load 事件中就要先为它添加，代码如下：

```
        Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim i As Integer
        For i = 0 To AxMapControl1.LayerCount - 1

ComboBox2.Items.Add(AxMapControl1.Map.Layer(i).Name.ToString())
        Next
```

```
        ComboBox2.Text = ComboBox1.Items(0)
    End Sub
```

4）为 ComboBox3 添加缓冲距离单位列表，其为：

```
Unknown
Inches
Points
Feet
Yards
Miles
NauticalMiles
Millimeters
Centimeters
Meters
Kilometers
DecimalDegrees
Decimeters
```

并设置其 Text 属性为 Unknown，即默认值为 Unknown。

5）为浏览按钮 Button20 添加 Click 事件，代码如下：

```
    Private Sub Button20_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button20.Click
        Dim saveDlg As SaveFileDialog = New SaveFileDialog()
        saveDlg.CheckPathExists = True
        saveDlg.Filter = "Shapefile (*.shp)|*.shp"
        saveDlg.OverwritePrompt = True
        saveDlg.Title = "保存缓冲数据"
        saveDlg.RestoreDirectory = True
        saveDlg.FileName = CStr(ComboBox2.Text) & "_buffer.shp"

        Dim dr As DialogResult = saveDlg.ShowDialog()
        If dr = System.Windows.Forms.DialogResult.OK Then
            outputPathText.Text = saveDlg.FileName
        End If
        End Sub
```

6）为缓冲分析按钮 Button19 添加 Click 事件，代码如下：

```
    Private Sub Button19_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button19.Click
        Dim gp As ESRI.ArcGIS.Geoprocessor.Geoprocessor = New
ESRI.ArcGIS.Geoprocessor.Geoprocessor()
        Dim bufferDistance As Double
        bufferDistance = bufferDistanceTextBox.Text
        Dim player As ESRI.ArcGIS.Carto.ILayer

        Dim i As Integer
        For i = 0 To AxMapControl1.LayerCount - 1
```

```
            If AxMapControl1.Map.Layer(i).Name = ComboBox2.Text Then
                player = AxMapControl1.Map.Layer(i)
                Exit For
            End If
        Next

        Dim buffer As ESRI.ArcGIS.AnalysisTools.Buffer = New
ESRI.ArcGIS.AnalysisTools.Buffer(player, outputPathText.Text,
CStr(bufferDistance) & " " & CStr(ComboBox3.SelectedItem))
        Dim results As ESRI.ArcGIS.Geoprocessing.IGeoProcessorResult =
CType(gp.Execute(buffer, Nothing),
ESRI.ArcGIS.Geoprocessing.IGeoProcessorResult)

        If results.Status <> 4 Then
            MsgBox("缓冲分析失败！")
        Else
            MsgBox("缓冲分析成功！")
            Dim j As Integer
            Dim bufferShpPath As String = outputPathText.Text
            j = bufferShpPath.LastIndexOf("\")

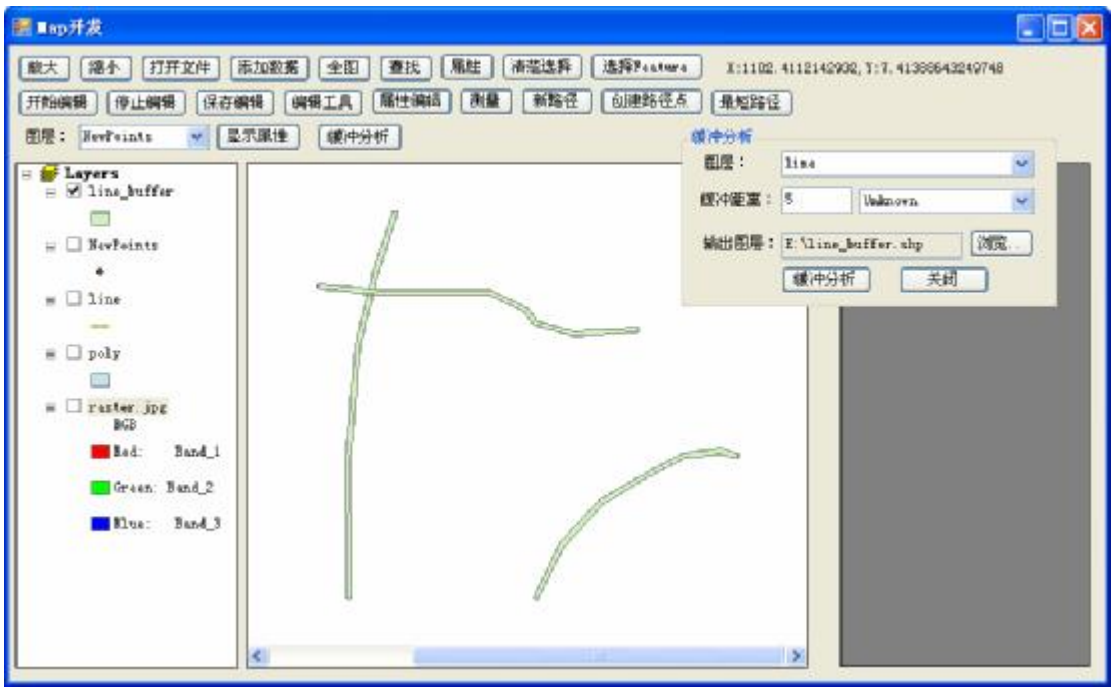
AxMapControl1.AddShapeFile(Microsoft.VisualBasic.Left(bufferShpPath,
j), Microsoft.VisualBasic.Right(bufferShpPath, bufferShpPath.Length - j
- 1))
        End If
        End Sub
```

其中 MsgBox("缓冲分析成功！")后为添加缓冲数据到 Map 控件中。

7）最后，为关闭按钮 Button21 添加 Click 事件，用于隐藏 GroupBox1，代码如下：

```
    Private Sub Button21_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button21.Click
        GroupBox1.Visible = False
       End Sub
```

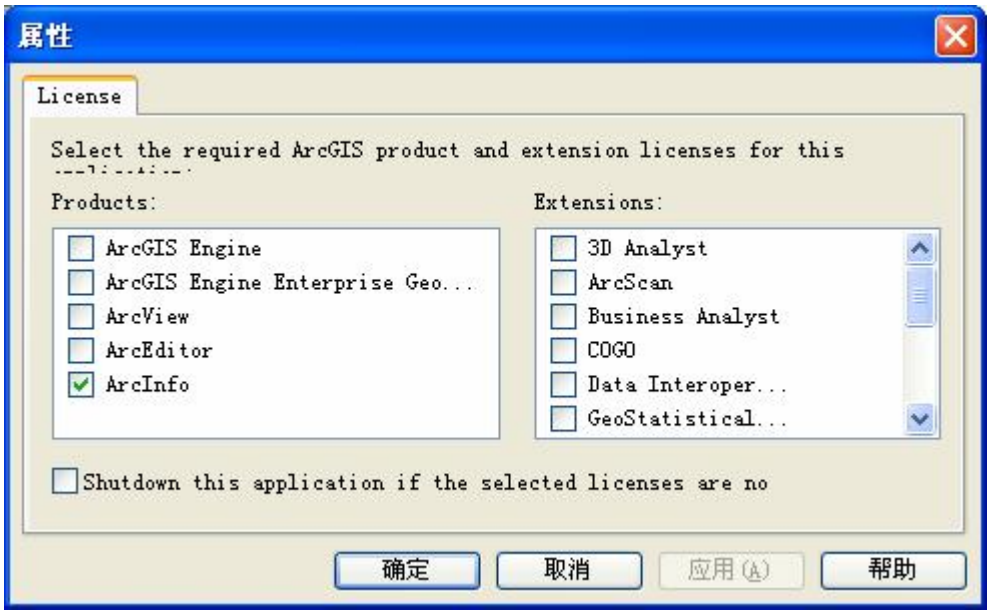
运行程序，其结果如下图所示：



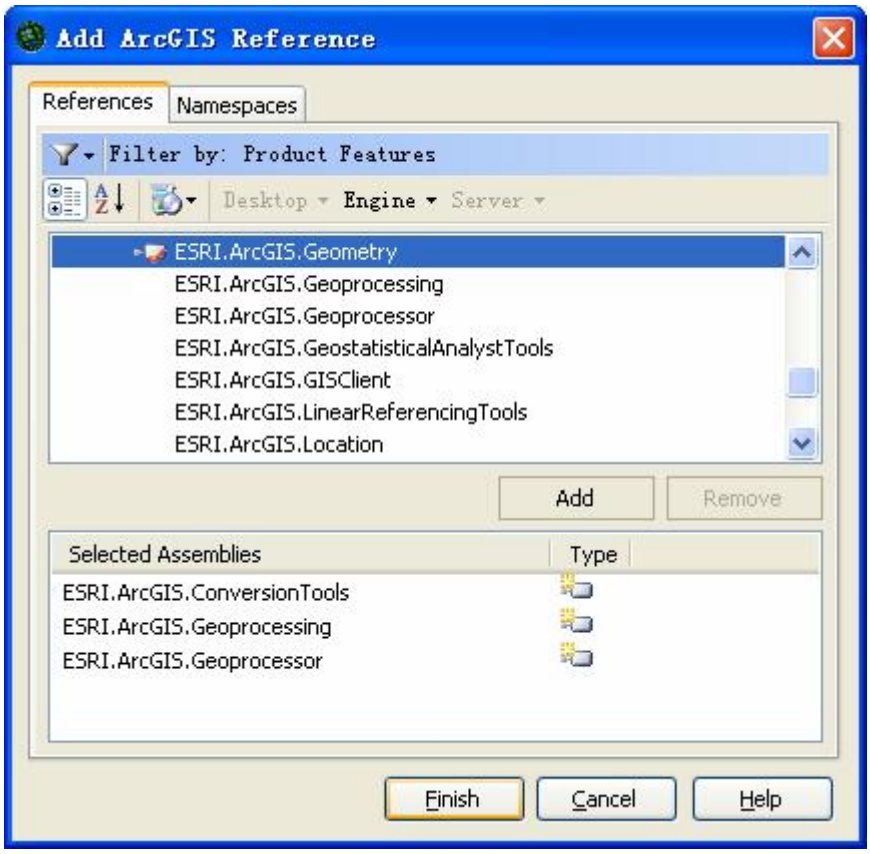
10、CAD 与 Shapfile 格式之间的相互转换

C#中：

1）首先，添加一个 LicenseControl，如图所示，。在上面点击右键选择属性，在弹出的属性窗体上选择如下：选中 **ArcInfo** 即可，其余都不选。右侧的 Extention 可选。



2) 然后再在菜单栏上选择“项目”——“Add ArcGIS Reference…”，在弹出的窗体上添加引用 ConversionTools,Geoprocessing,Geoprocessor，如下图所示：



3) 在窗体上添加一个按钮，在窗体的代码窗口上输入下列代码：

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using ESRI.ArcGIS.ConversionTools;
using ESRI.ArcGIS.Display;
using ESRI.ArcGIS.esriSystem;
using ESRI.ArcGIS.Geoprocessor;
using ESRI.ArcGIS.Geoprocessing;
using ESRI.ArcGIS.Geometry;
using ESRI.ArcGIS.Output;


namespace WindowsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
```

```
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Geoprocessor GP = new Geoprocessor();

            ExportCAD GPexport = new ExportCAD();
            GPexport.Output_File = @"F:\XZY\Test\D111.dxf";   //输出的
文件名
            GPexport.Output_Type = "DXF_R2000";  //输出的文件类型
            GPexport.in_features = @"F:\XZY\Test\q.shp";   //源文件
            GP.Execute(GPexport, null);

            MessageBox.Show("转换完成！");

        }
    }
}
```

本机安装有 VS2005，ArcInfo 9.2，ArcEngine 9.2 for .Net，在本机上测试运行成功！

VB.Net 中：

第一步和第二步同 C#相同，只是代码不同，如下：

```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click


        Dim gp As ESRI.ArcGIS.Geoprocessor.Geoprocessor = New
ESRI.ArcGIS.Geoprocessor.Geoprocessor()

        Dim pExptoCAD As ExportCAD = New ExportCAD()
        pExptoCAD.in_features = "F:\XZY\Test\q.shp"
        pExptoCAD.Output_File = "F:\XZY\Test\VB1.dwg"
        pExptoCAD.Output_Type = "DWG_R2004"
        gp.Execute(pExptoCAD, Nothing)
        MsgBox("OK!")

    End Sub
End Class
```

值得注意的是：Geoprocessor 这个类在 ESRI.ArcGIS.Geoprocessor 和

ESRI.ArcGIS.Geoprocessing 中都有，如果不指定的话，在 VB.Net 中默认为 ESRI.ArcGIS.Geoprocessing，将无法实现。在 C# 中也是同样的问题，但是在 C# 中可以在执行方法的时候人为的指定它的重载函数。

11、将 CAD 导出成 Shp 格式

基本同前，代码如下：

```
Imports ESRI.ArcGIS.ConversionTools
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        '添加引用ESRI.ArcGIS.Geoprocessor，ESRI.ArcGISConversionTools
        Dim gp As ESRI.ArcGIS.Geoprocessor.Geoprocessor = New
ESRI.ArcGIS.Geoprocessor.Geoprocessor()

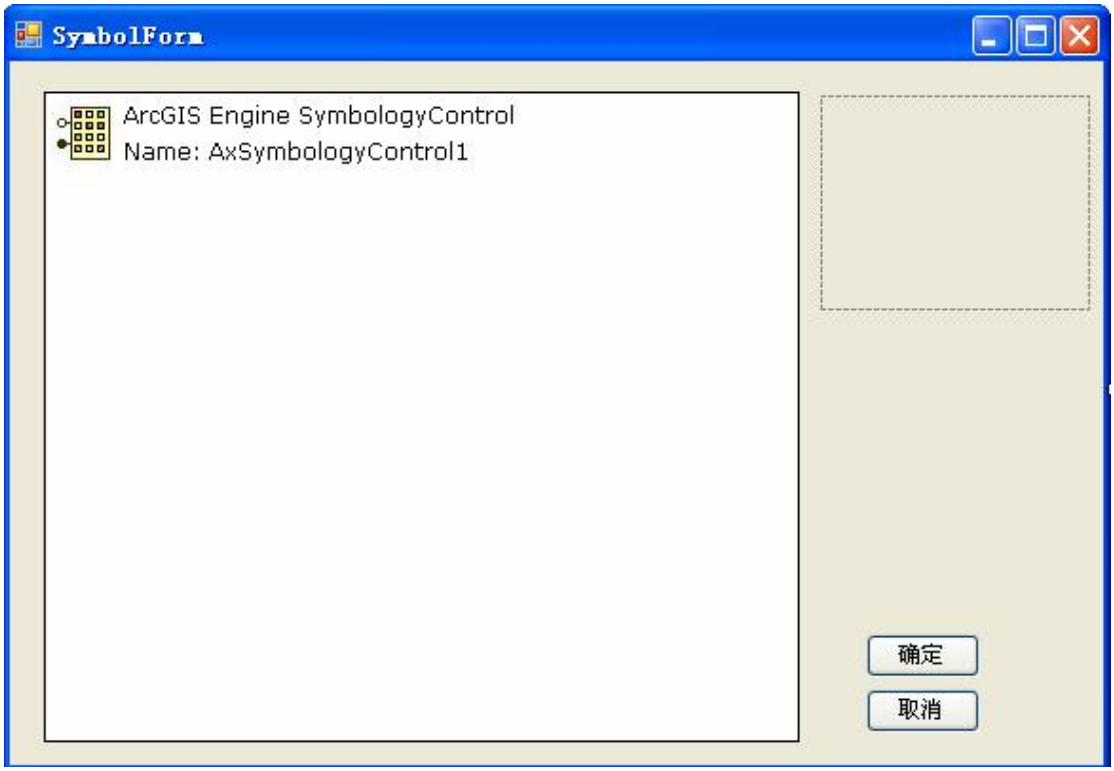
        Dim pExptoCAD As ExportCAD = New ExportCAD()

        Dim aa As FeatureClassToShapefile = New
FeatureClassToShapefile()
        aa.Input_Features = "F:\XZY\Test\D.dwg\MultiPatch"
        aa.Output_Folder = "F:\XZY\Test\"

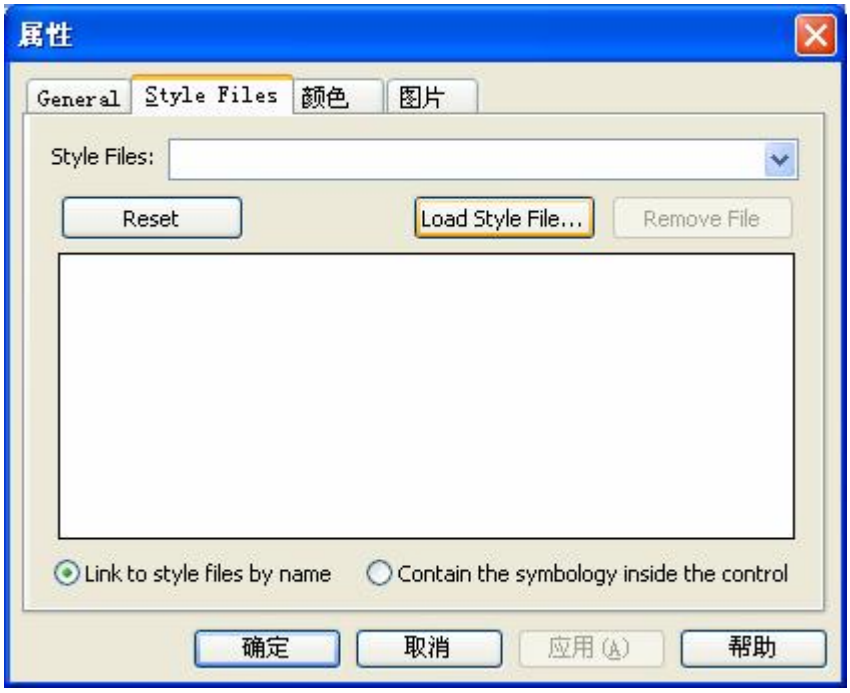
        gp.Execute(aa, Nothing)
        MsgBox("OK!")
    End Sub
End Class
```

12、创建符号选择窗体 SymbologyControl

1）SymbologyControl 是 ArcGIS Engine 中的符号显示、选择控件。首先在 Map 开发窗体上添加一个符号选择按钮。然后新建一个 Windows 窗体，并命名为 SymbolForm，在 SymbolForm 上添加一个 AxSymbologyControl1 控件，添加一个 PictureBox1 控件用于预览选择的符号，再添加两个按钮，如下图所示：



2）在 AxSymbologyControl1 控件上点击右键打开属性窗体，在属性窗体上点击 Style Files 标签，如下图所示，通过点击 Load Style File 按钮可以浏览 Style 文件。



同样，还可以通过程序设计来实现，如下在窗体启动时就加载符号文件：

```
Private Sub SymbolForm_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Dim regKey As Microsoft.Win32.RegistryKey =
Microsoft.Win32.Registry.LocalMachine.OpenSubKey("SOFTWARE\\ESRI\\Cor
```

```
eRuntime", True)

AxSymbologyControl1.LoadStyleFile(regKey.GetValue("InstallDir") &
"\\Styles\\3D Buildings.ServerStyle")
        End Sub
```

其中 regKey 变量是获取的 ESRI 系统注册键，通过 regKey.GetValue("InstallDir")属性就可以获取 ArcGIS 的安装路径。当然也可以自己指定 Style 文件的路径，值得注意的是：在此引用的 Style 文件是*.ServerStyle 文件，而不是*.style 文件。

3）在全局变量中定义选择的符号，如下：

```
Public m_styleGalleryItem As ESRI.ArcGIS.Display.IStyleGalleryItem
```

4）定义 PreviewImage()函数，用于预览选中的符号，代码如下：

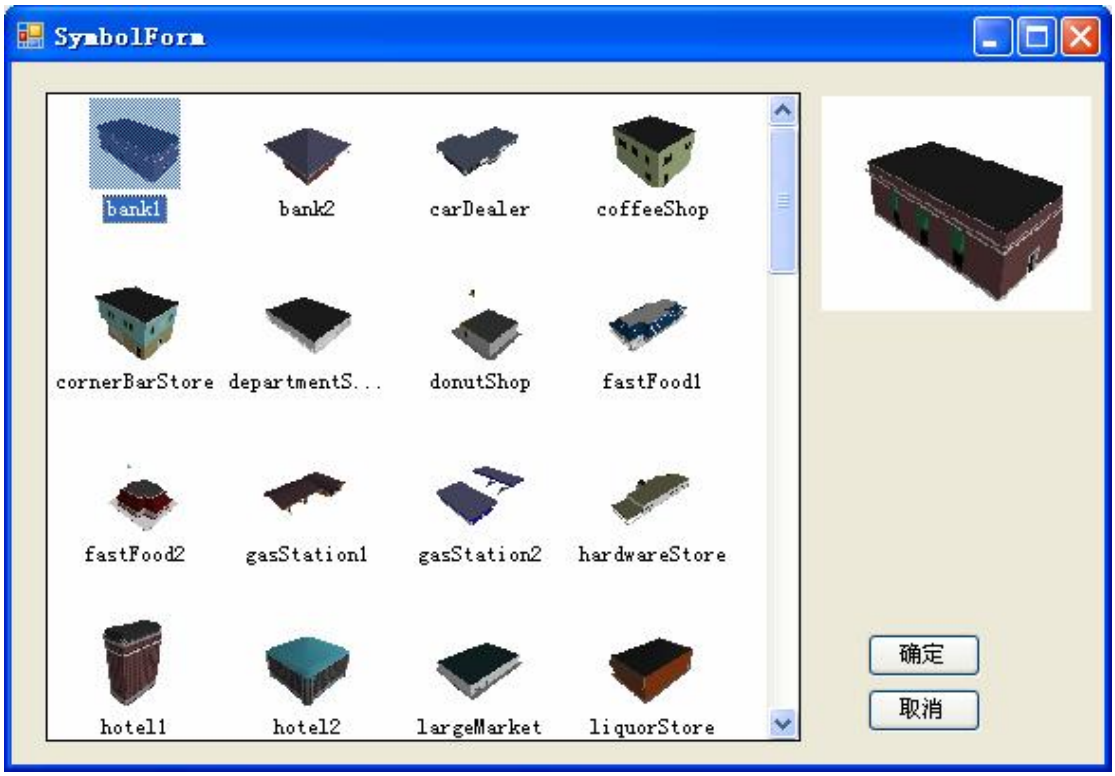
```
 Private Sub PreviewImage()
        Dim symbologyStyleClass As
ESRI.ArcGIS.Controls.ISymbologyStyleClass
        symbologyStyleClass =
AxSymbologyControl1.GetStyleClass(AxSymbologyControl1.StyleClass)
        Dim picture As stdole.IPictureDisp
        picture = symbologyStyleClass.PreviewItem(m_styleGalleryItem,
PictureBox1.Width, PictureBox1.Height)
        Dim image As System.Drawing.Image
        image = System.Drawing.Image.FromHbitmap(New
System.IntPtr(picture.Handle))
        PictureBox1.Image = image
        End Sub
```

其中，symbologyStyleClass 包括加载的 Style 文件中所有的符号，通过 symbologyStyleClass 的 PreviewItem 属性可以预览选中的符号，picture 的类型是根据 symbologyStyleClass 的 PreviewItem 属性所对应的类型而定。System.IntPtr 结构是用于表示指针或句柄的平台特定类型，其指到 picture.Handle 的句柄。

5) 最后，在 AxSymbologyControl1 控件的选择项目事件 OnItemSelected 上添加指定选择符号对象并调用预览符号事件，代码如下：

```
 Private Sub AxSymbologyControl1_OnItemSelected(ByVal sender As
System.Object, ByVal e As
ESRI.ArcGIS.Controls.ISymbologyControlEvents_OnItemSelectedEvent)
Handles AxSymbologyControl1.OnItemSelected
        m_styleGalleryItem = e.styleGalleryItem
        PreviewImage()
        End Sub
```

运行程序，其结果如下图所示：



至此，整个 SymbolForm 窗体的所有程序代码如下：

```
Public Class SymbolForm
    Public m_styleGalleryItem As ESRI.ArcGIS.Display.IStyleGalleryItem

    Private Sub SymbolForm_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Dim regKey As Microsoft.Win32.RegistryKey =
Microsoft.Win32.Registry.LocalMachine.OpenSubKey("SOFTWARE\\ESRI\\Cor
eRuntime", True)

AxSymbologyControl1.LoadStyleFile(regKey.GetValue("InstallDir") &
"\\Styles\\3D Buildings.ServerStyle")
    End Sub

    Private Sub PreviewImage()
        Dim symbologyStyleClass As
ESRI.ArcGIS.Controls.ISymbologyStyleClass
        symbologyStyleClass =
AxSymbologyControl1.GetStyleClass(AxSymbologyControl1.StyleClass)
        Dim picture As stdole.IPictureDisp
        picture = symbologyStyleClass.PreviewItem(m_styleGalleryItem,
PictureBox1.Width, PictureBox1.Height)
        Dim image As System.Drawing.Image
        image = System.Drawing.Image.FromHbitmap(New
System.IntPtr(picture.Handle))
```

```
        PictureBox1.Image = image
    End Sub

    Private Sub AxSymbologyControl1_OnItemSelected(ByVal sender As
System.Object, ByVal e As
ESRI.ArcGIS.Controls.ISymbologyControlEvents_OnItemSelectedEvent)
Handles AxSymbologyControl1.OnItemSelected
        m_styleGalleryItem = e.styleGalleryItem
        PreviewImage()
    End Sub
End Class
```

13、指定对象的符号

结合前面的符号选择窗体，下面继续对某个数据进行符号修改。首先我们来了解一下这个程序总的设计思路：先通过在 TOCControl 控件上点击右键弹出菜单；在菜单上点击更改对象符号的菜单；通过鼠标点击 TOCControl 控件上的位置获取到所选的图层；通过图层的类型判断图层数据的类型，如点、线、面；已知图层数据的类型，然后启动符号选择窗体，并将选择的符号返回之主程序；将返回的符号赋予图层数据文件，刷新 Map，完成。下面开始具体的实际操作：

1）添加一个 ContextMenuStrip 控件，用于 TOCControl 控件上的右键菜单显示，并添加一个菜单，如下图所示：



2）在全局变量中定义当前鼠标在 TOCControl 控件上点击的位置，代码如下：

```
    Public currentMouseX, currentMouseY As Integer
```

3）在 AxTOCControl2 控件的 OnMouseDown 事件中添加代码，如果点击右键时弹出前面定义的 ContextMenuStrip 控件，代码如下：

```
    Private Sub AxTOCControl2_OnMouseDown(ByVal sender As System.Object,
ByVal e As ESRI.ArcGIS.Controls.ITOCControlEvents_OnMouseDownEvent)
Handles AxTOCControl2.OnMouseDown
        If e.button = 2 Then
            TOCControlContextMenu.Show(AxTOCControl2, e.x, e.y)
            currentMouseX = e.x
            currentMouseY = e.y
        End If
    End Sub
```

其中，TOCControlContextMenu.Show(AxTOCControl2, e.x, e.y)表示显示 TOCControlContextMenu 菜单在 AxTOCControl2 控件的 x、y 处。

4）定义更改符号的函数 ChangeSymbol（），其参数有当前鼠标的位置 X 和 Y，代码如下所示：

```
    Private Sub ChangeSymbol(ByVal mouseX As Integer, ByVal mouseY As
```

```
Integer)
        Dim map As ESRI.ArcGIS.Carto.IMap = AxMapControl1.Map
        Dim layer As ESRI.ArcGIS.Carto.ILayer = New
ESRI.ArcGIS.Carto.FeatureLayer()
        Dim other As Object = New Object
        Dim index As Object = New Object
        Dim item As esriTOCControlItem

        '获取AxTOCControl2中当前位置的图层
        AxTOCControl2.HitTest(mouseX, mouseY, item, map, layer, other,
index)

        If layer Is Nothing Then Exit Sub
        If Not TypeOf layer Is ESRI.ArcGIS.Carto.IFeatureLayer Then Exit
Sub

        Dim featureLayer As ESRI.ArcGIS.Carto.IFeatureLayer = layer
        Dim geoFeatureLayer As ESRI.ArcGIS.Carto.IGeoFeatureLayer =
featureLayer
        Dim simpleRenderer As ESRI.ArcGIS.Carto.ISimpleRenderer =
geoFeatureLayer.Renderer

        '调用SymbolForm窗体
        Dim symbolForm1 As New SymbolForm

        Dim styleGalleryItem As ESRI.ArcGIS.Display.IStyleGalleryItem =
Nothing
        '根据数据的类型选定esriSymbologyStyleClass符号样式
        Select Case featureLayer.FeatureClass.ShapeType
            Case
ESRI.ArcGIS.Geometry.esriGeometryType.esriGeometryPoint
                styleGalleryItem =
symbolForm1.GetItem(ESRI.ArcGIS.Controls.esriSymbologyStyleClass.esri
StyleClassMarkerSymbols, simpleRenderer.Symbol)
            Case
ESRI.ArcGIS.Geometry.esriGeometryType.esriGeometryPolyline
                styleGalleryItem =
symbolForm1.GetItem(ESRI.ArcGIS.Controls.esriSymbologyStyleClass.esri
StyleClassLineSymbols, simpleRenderer.Symbol)
            Case
ESRI.ArcGIS.Geometry.esriGeometryType.esriGeometryPolygon
                styleGalleryItem =
symbolForm1.GetItem(ESRI.ArcGIS.Controls.esriSymbologyStyleClass.esri
StyleClassFillSymbols, simpleRenderer.Symbol)
```

```
        End Select

        SymbolForm.Dispose()
        Me.Activate()
        If styleGalleryItem Is Nothing Then Exit Sub

        '创建一个渲染
        simpleRenderer = New ESRI.ArcGIS.Carto.SimpleRendererClass
        '设置它的符号从styleGalleryItem来
        simpleRenderer.Symbol = styleGalleryItem.Item
        '吧渲染应用到geoFeatureLayer图层上
        geoFeatureLayer.Renderer = simpleRenderer

        AxMapControl1.Refresh()
       End Sub
```

其中，AxTOCControl2.HitTest（）是获取在当前位置的图层。

5）在 SymbolForm 窗体的程序代码中定义函数 GetItem（），其代码如下：

```
    Public Function GetItem(ByRef styleClass As
ESRI.ArcGIS.Controls.esriSymbologyStyleClass, ByVal symbol As
ESRI.ArcGIS.Display.ISymbol) As ESRI.ArcGIS.Display.IStyleGalleryItem
        m_styleGalleryItem = Nothing
        '获取符号样式集symbologyStyleClass
        AxSymbologyControl1.StyleClass = styleClass
        Dim symbologyStyleClass As
ESRI.ArcGIS.Controls.ISymbologyStyleClass =
AxSymbologyControl1.GetStyleClass(styleClass)
        Dim styleGalleryItem As New
ESRI.ArcGIS.Display.ServerStyleGalleryItem
        styleGalleryItem.Item = symbol
        styleGalleryItem.Name = "mySymbol"

        symbologyStyleClass.AddItem(styleGalleryItem, 0)
        symbologyStyleClass.SelectItem(0)

        Me.ShowDialog()
        Return m_styleGalleryItem
       End Function
```

6）更改 SymbolForm 窗体的程序代码中的 SymbolForm_Load（）函数，初始化时导入 ESRI.ServerStyle 文件，其代码如下所示：

```
    Private Sub SymbolForm_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        Dim regKey As Microsoft.Win32.RegistryKey =
Microsoft.Win32.Registry.LocalMachine.OpenSubKey("SOFTWARE\\ESRI\\Cor
eRuntime", True)
```

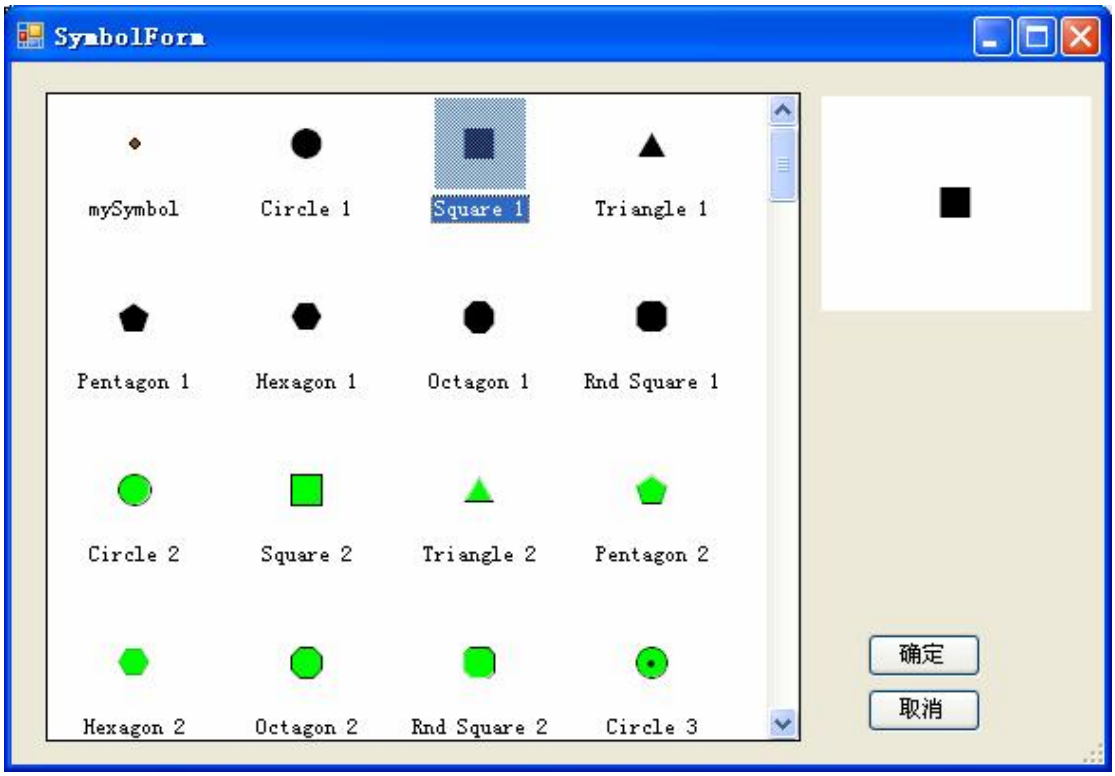
```
AxSymbologyControl1.LoadStyleFile(regKey.GetValue("InstallDir") &
"\\Styles\\ESRI.ServerStyle")
        End Sub
```

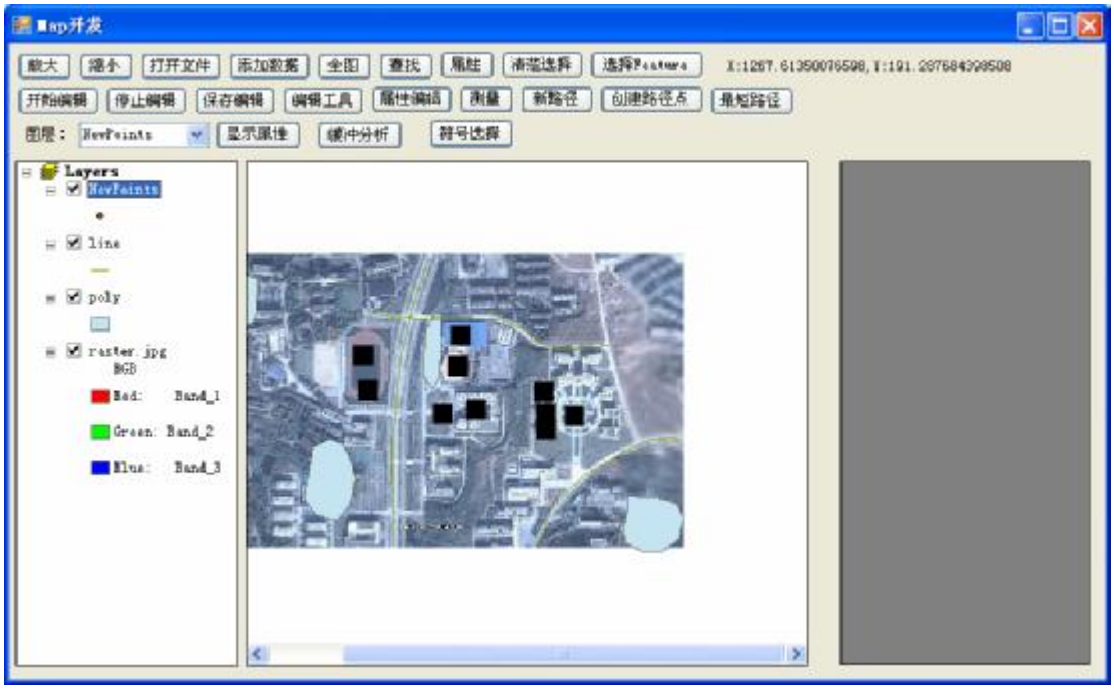
7）为 ContextMenuStrip 菜单控件中的更改显示符号菜单添加事件，其代码如下所示：

```
    Private Sub 更改显示符号SToolStripMenuItem_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles 更改显示符号
SToolStripMenuItem.Click
        ChangeSymbol(currentMouseX, currentMouseY)
        End Sub
```

8）允许程序，其最终显示效果如下图所示：







14、三维分析

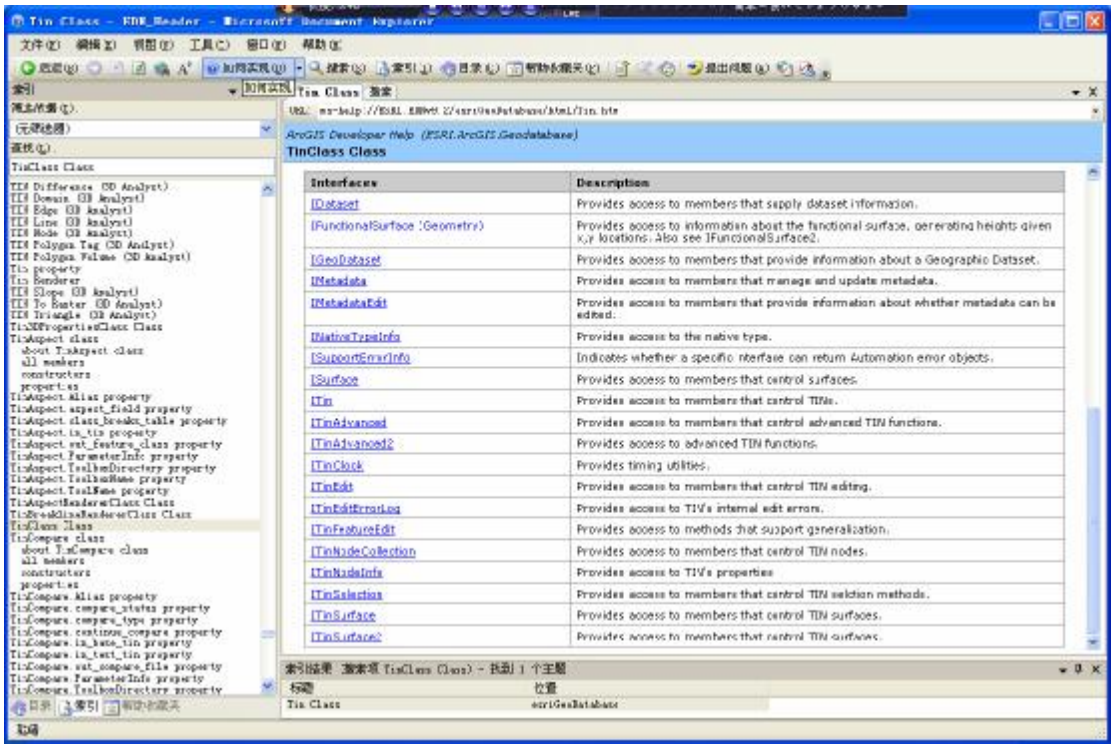
三维分析是 GIS 分析中的一个重要部分，主要对 TIN 数据进行查询、分析等。下面介绍几种常用的三维分析：

1）获取到 TIN 图层

```
Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
```

```
ESRI.ArcGIS.Carto.TinLayer()
        j = AxMapControl1.Map.Layer(7)
        pTIN = j.Dataset
```

其中 Map.Layer(7)是该 Map 文件中的第 8 个文件，该文件为 TIN 数据文件。pTIN 是 TIN 类中的一个接口，它可以与 ISurface、ITinEdit、ITinSurface 等接口连接，它们都同属于 TIN 类，如下图所示：



此外，常用的很多三维分析，如查询高程点、坡度、等高线、面积、体积等都是通过 **ISurface** 接口来实现的。

2）获取点击位置的高程值

该功能主要通过 **ISurface** 接口的 GetElevation 属性来实现。由于点击鼠标事件需要在 MapControl 控件的 OnMouseDown 事件里面进行，所以这些代码都必须添加到 MapControl 的 OnMouseDown 事件中。

首先，添加一个获取 TIN 高程值的按钮，然后在全局变量中定义如下，看是否现在鼠标在 MapControl 控件上按下的时候获取点击位置高程值（在多功能集合在一起的时候，需要区分现在使用的是具体某个功能，所以先在全局变量中定义该功能是否可以用）：

```
Public TINIdentifyElevation As Boolean
```

然后，为获取 TIN 高程值的按钮添加 Click 事件，代码如下：

```
Private Sub Button24_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button24.Click
        TINIdentifyElevation = True
    End Sub
```

在 MapControl 控件的 OnMouseDown 事件中添加如下代码：

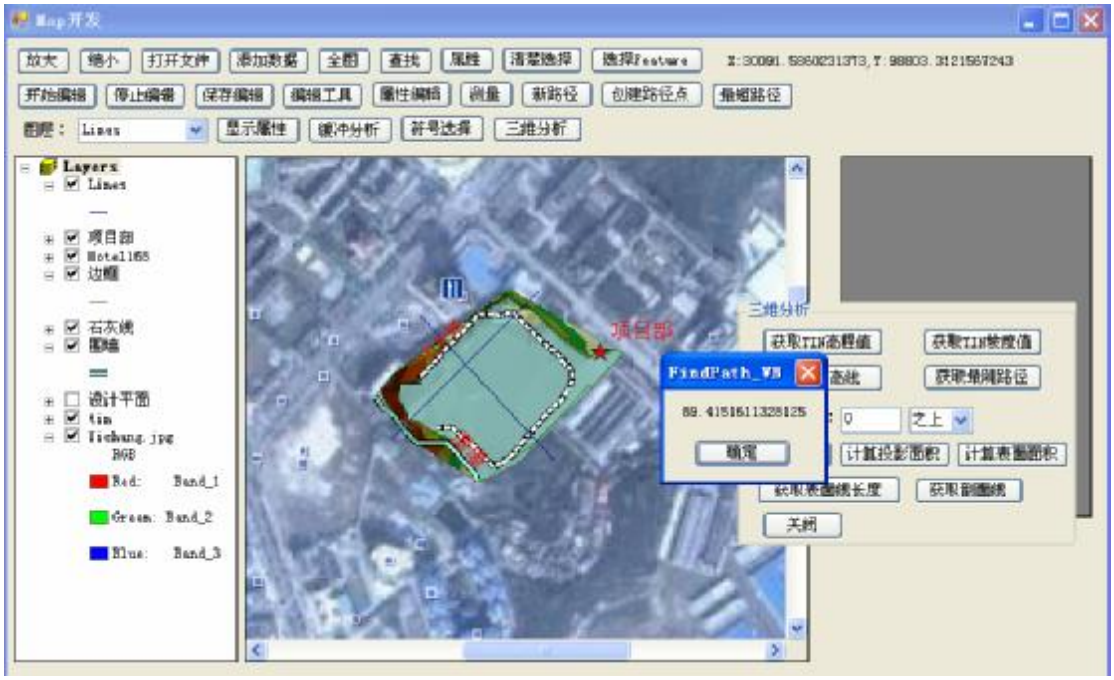
```
Private Sub AxMapControl1_OnMouseDown(ByVal sender As System.Object,
ByVal e As ESRI.ArcGIS.Controls.IMapControlEvents2_OnMouseDownEvent)
Handles AxMapControl1.OnMouseDown
```

```
        If e.button = 1 Then
            Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
            Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
            j = AxMapControl1.Map.Layer(7)
            pTIN = j.Dataset

            Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
            pSurface = pTIN
            Dim pPoint As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
            pPoint.X = e.mapX
            pPoint.Y = e.mapY

            If TINIdentifyElevation Then
                MsgBox(pSurface.GetElevation(pPoint))
            End If
        End If
       End Sub
```

其中，当鼠标左键点击（e.button = 1）时，先获取 TIN 图层，然后将 ISurface 接口连接到 ITIN 接口上。设置鼠标点击位置为 pPoint，通过 pSurface 的 GetElevation 属性获取 pPoint 处的高程值。运行程序，其显示结果如下图所示：



3）获取 TIN 坡度值

获取 TIN 坡度值也是通过 **ISurface** 接口的 GetSlopeDegrees 属性来实现。同样也是通过 MapControl 控件的 OnMouseDown 事件进行。

首先，在全局变量中定义如下：

```
Public TINIdentifyDegree As Boolean
```

然后，在获取 TIN 坡度值按钮上添加事件，代码如下：

```
Private Sub Button25_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button25.Click
        TINIdentifyDegree = True
        TINIdentifyElevation = False
       End Sub
```

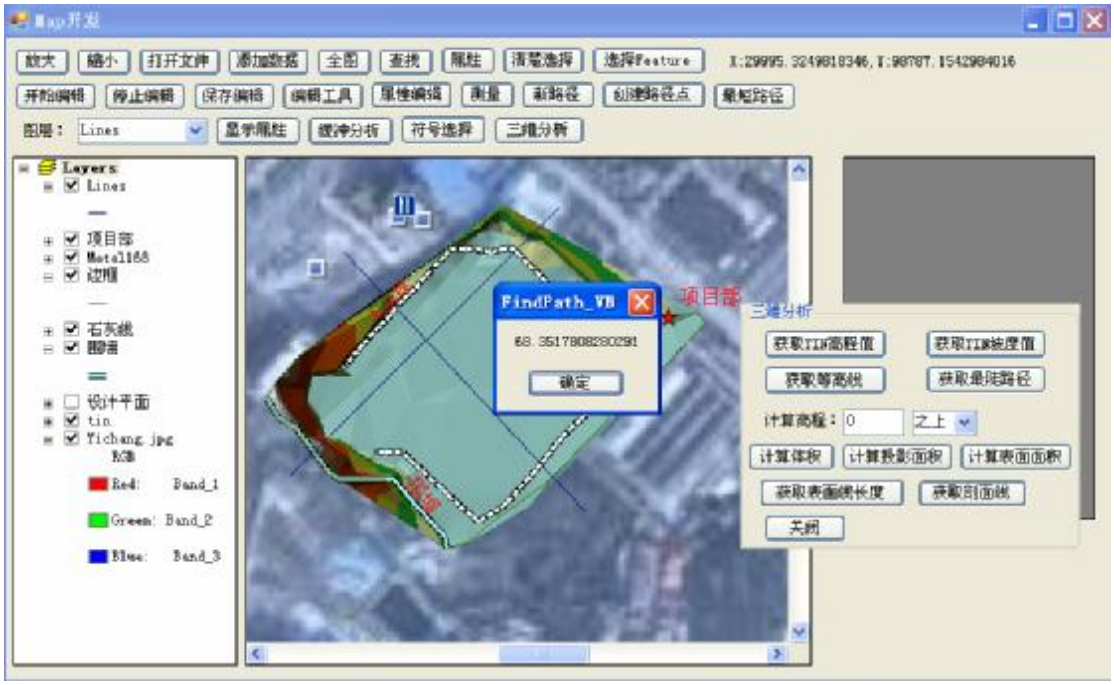
将 MapControl 控件 OnMouseDown 事件中的代码：

```
        If TINIdentifyElevation Then
            MsgBox(pSurface.GetElevation(pPoint))
        End If
```

更改为：

```
        If TINIdentifyElevation Then
            MsgBox(pSurface.GetElevation(pPoint))
        ElseIf TINIdentifyDegree Then
               MsgBox(pSurface.GetSlopeDegrees(pPoint))
        End If
```

运行程序，其显示结果如下图所示：



4）获取等高线

首先添加一个获取等高线按钮，在全局变量中定义：

```
Public  TINGetContour  As Boolean
```

在获取等高线按钮的 Click 事件中添加如下代码：

```
Private Sub Button27_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button27.Click
        TINGetContour = True
        TINIdentifyDegree = False
        TINIdentifyElevation = False
```

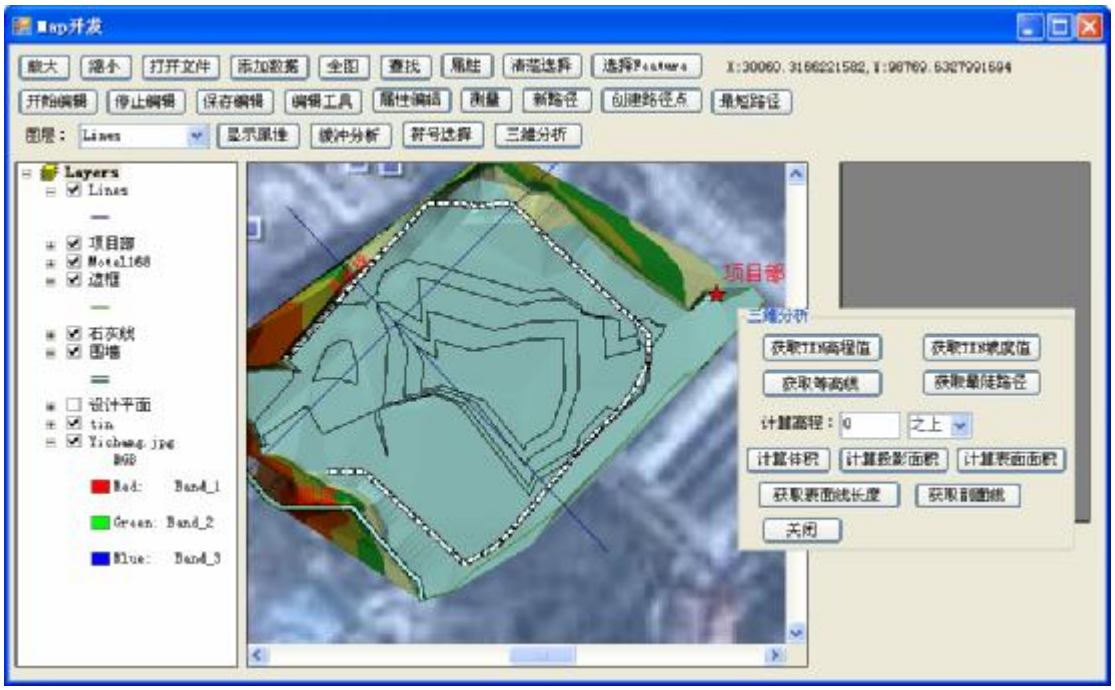
```
        End Sub
```

同样是通过 **ISurface** 接口的 GetContour 方法获得等高线，但此处除了获取到等高线外，还需要绘制出等高线。在 MapControl 控件的 OnMouseDown 事件中 MsgBox(pSurface.GetSlopeDegrees(pPoint)) 和 End If 之间添加如下代码：

```
ElseIf TINGetContour Then
                Dim pElevation As Double =
pSurface.GetElevation(pPoint)
                Dim ppcontour As ESRI.ArcGIS.Geometry.IPolyline = New
ESRI.ArcGIS.Geometry.Polyline
                    pSurface.GetContour(pPoint, ppcontour, pElevation)
                    AxMapControl1.DrawShape(ppcontour)
```

其中，AxMapControl1.DrawShape(ppcontour)就是在 AxMapControl1 中绘制获取到的等高线线条。

运行程序，其显示结果如下图所示：

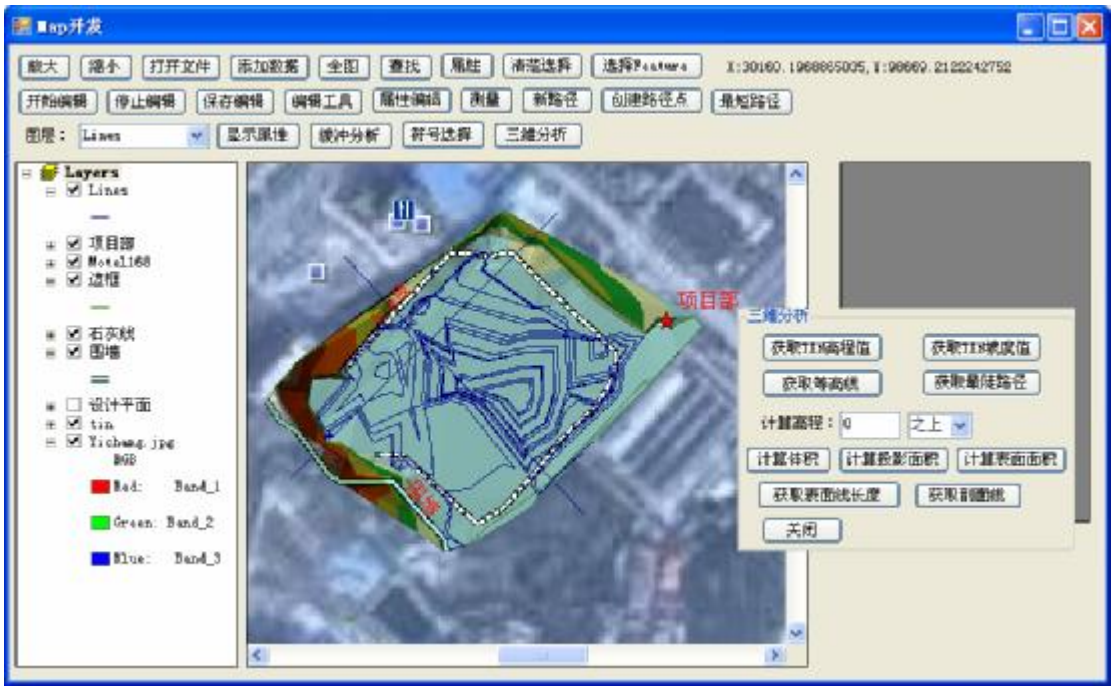


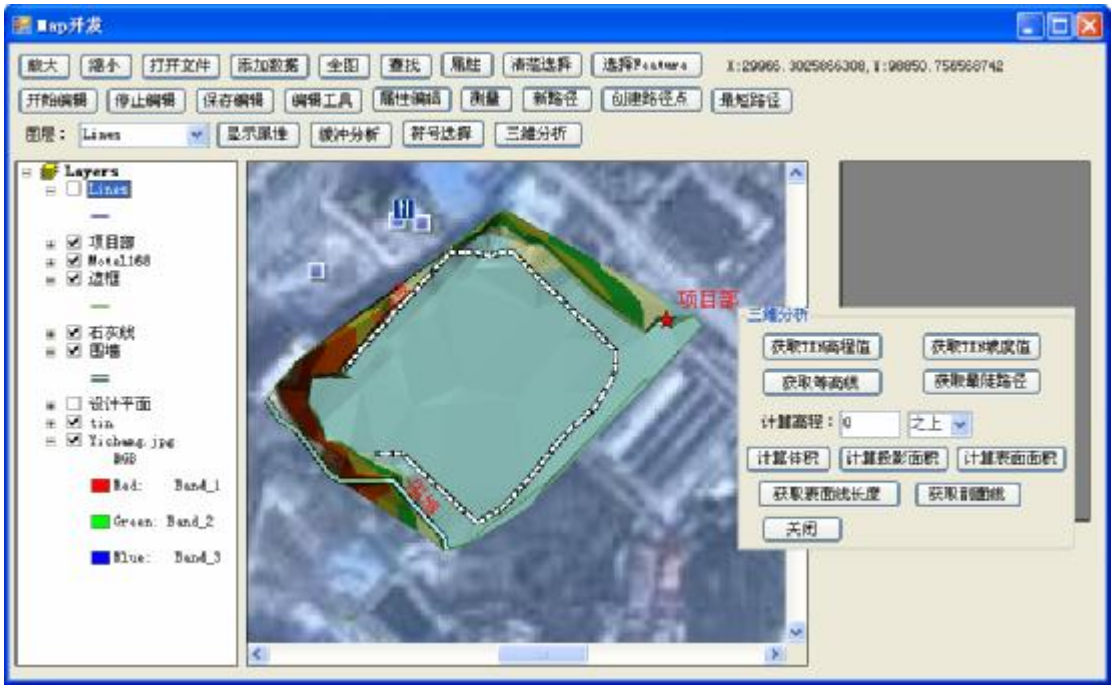
同样，还可以把获取的等高线数据保存值线数据文件，如上图中图层 Lines 就是一个线数据文件。在前面的代码“AxMapControl1.DrawShape(ppcontour)”后添加如下代码，即可实现：

```
                Dim player As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.FeatureLayer()
                player = AxMapControl1.Map.Layer(0)
                Dim pFeatureClass As
ESRI.ArcGIS.Geodatabase.IFeatureClass = player.FeatureClass
                Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature =
pFeatureClass.CreateFeature()
                pFeature.Shape = ppcontour
                pFeature.Store()
```

其中，pFeature.Shape = ppcontour 是设置新建的 IFeature 的几何形状是

前面获取的等高线，然后通过 pFeature.Store()保存形状（注：一定要使用本代码，否则将无法保存获取的等高线至 Line 数据文件）。运行程序，其显示结果如下图所示：





5）获取最陡路径

在全局变量中定义如下：

```
Public TINIdentifyElevation, TINIdentifyDegree, TINGetContour, TINGetSteepPath As Boolean
```

然后在获取最陡路径按钮的 Click 事件中添加如下代码：

```
Private Sub Button32_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button32.Click
```

```
        TINGetSteepPath = True
        TINGetContour = False
        TINIdentifyDegree = False
        TINIdentifyElevation = False
    End Sub
```

在 MapControl 控件的 OnMouseDown 事件中添加 ISurface.GetSteepestPath 事件。至此，整个 MapControl 控件 OnMouseDown 事件的代码如下：

```
    Private Sub AxMapControl1_OnMouseDown(ByVal sender As System.Object,
ByVal e As ESRI.ArcGIS.Controls.IMapControlEvents2_OnMouseDownEvent)
Handles AxMapControl1.OnMouseDown
        If e.button = 1 Then
            Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
            Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
            j = AxMapControl1.Map.Layer(7)
            pTIN = j.Dataset

            Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
            pSurface = pTIN
            Dim pPoint As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
            pPoint.X = e.mapX
            pPoint.Y = e.mapY

            If TINIdentifyElevation Then
                MsgBox(pSurface.GetElevation(pPoint))
            ElseIf TINIdentifyDegree Then
                MsgBox(pSurface.GetSlopeDegrees(pPoint))
            ElseIf TINGetContour Then
                Dim pElevation As Double =
pSurface.GetElevation(pPoint)
                Dim ppcontour As ESRI.ArcGIS.Geometry.IPolyline = New
ESRI.ArcGIS.Geometry.Polyline
                pSurface.GetContour(pPoint, ppcontour, pElevation)
                AxMapControl1.DrawShape(ppcontour)

                Dim player As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.FeatureLayer()
                player = AxMapControl1.Map.Layer(0)
                Dim pFeatureClass As
ESRI.ArcGIS.Geodatabase.IFeatureClass = player.FeatureClass
                Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature =
pFeatureClass.CreateFeature()
```

```
                pFeature.Shape = ppcontour
                pFeature.Store()

            ElseIf TINGetSteepPath Then
                Dim pSteepestPath As ESRI.ArcGIS.Geometry.IPolyline =
New ESRI.ArcGIS.Geometry.Polyline
                pSteepestPath = pSurface.GetSteepestPath(pPoint)
                AxMapControl1.DrawShape(pSteepestPath)
            End If
        End If
       End Sub
```

6）计算体积

首先，添加如下图所示控件，主要用于设置计算面积及体积的计算高程以及方向（在计算高程上还是在计算高程下？）



然后，在计算体积按钮的 Click 事件中添加如下代码：

```
 Private Sub Button26_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button26.Click
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
        Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        j = AxMapControl1.Map.Layer(7)
        pTIN = j.Dataset

        Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
        pSurface = pTIN

        If ComboBox4.Text = "之上" Then
            MsgBox("在高程" & TextBox1.Text & ComboBox4.Text & "的体积
是：" & pSurface.GetVolume(TextBox1.Text,
Geodatabase.esriPlaneReferenceType.esriPlaneReferenceAbove))
        Else
            MsgBox("在高程" & TextBox1.Text & ComboBox4.Text & "的体积
是：" & pSurface.GetVolume(TextBox1.Text,
Geodatabase.esriPlaneReferenceType.esriPlaneReferenceBelow))
        End If

       End Sub
```

其中，计算体积就是通过 pSurface.GetVolume 来实现的。运行程序，其结果如下图所示：



7）计算表面面积

在计算表面面积按钮的 Click 事件中添加如下代码：

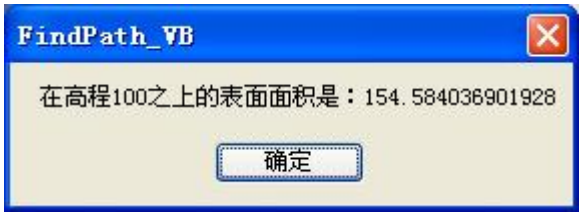
```
 Private Sub Button29_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button29.Click
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
        Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        j = AxMapControl1.Map.Layer(7)
        pTIN = j.Dataset

        Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
        pSurface = pTIN

        If ComboBox4.Text = "之上" Then
            MsgBox("在高程" & TextBox1.Text & ComboBox4.Text & "的表面
面积是：" & pSurface.GetSurfaceArea(TextBox1.Text,
Geodatabase.esriPlaneReferenceType.esriPlaneReferenceAbove))
        Else
            MsgBox("在高程" & TextBox1.Text & ComboBox4.Text & "的表面
面积是：" & pSurface.GetSurfaceArea(TextBox1.Text,
Geodatabase.esriPlaneReferenceType.esriPlaneReferenceBelow))
        End If

      End Sub
```

运行程序，其结果如下图所示：



8）计算投影面积

在计算投影面积按钮上添加如下代码：

```
 Private Sub Button28_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button28.Click
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
        Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        j = AxMapControl1.Map.Layer(7)
        pTIN = j.Dataset

        Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
        pSurface = pTIN

        If ComboBox4.Text = "之上" Then
            MsgBox("在高程" & TextBox1.Text & ComboBox4.Text & "的投影
面积是：" & pSurface.GetProjectedArea(TextBox1.Text,
Geodatabase.esriPlaneReferenceType.esriPlaneReferenceAbove) & "平方" &
AxMapControl1.Map.MapUnits.ToString())
        Else
            MsgBox("在高程" & TextBox1.Text & ComboBox4.Text & "的投影
面积是：" & pSurface.GetProjectedArea(TextBox1.Text,
Geodatabase.esriPlaneReferenceType.esriPlaneReferenceBelow) & "平方" &
AxMapControl1.Map.MapUnits.ToString())
        End If
       End Sub
```

其中，AxMapControl1.Map.MapUnits.ToString()是获取的。运行程序，其结果如下图所示：

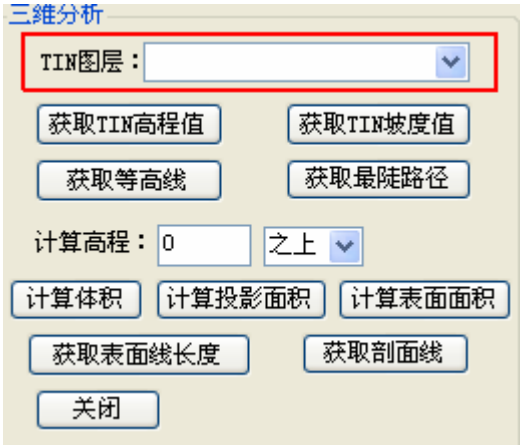


当然，后面的 esriMeters 可以通过一个 Switch...Case 语句来设置。

15、获取 TIN 图层

前面的三维分析都是在 TIN 图层上进行的，但是如何让系统自动获取到 TIN 图层，则主要通过 TIN 图层设置与是否出现错误来判断该图层是否为 TIN 图层。

如下图所示，添加一个 TIN 图层的 ComboBox（该控件名称为 ComboBox5）：



以后凡是要用到 TIN 图层的都直接从 TIN 图层的 ComboBox5 的 Text 属性中获得该 TIN 图层的名称，下面就是寻找 TIN 图层位置的函数，代码如下：

```
    Private Sub FindTINLayer(Optional ByRef TINLayerIndex As Integer =
0)
        On Error Resume Next
        Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        Dim i As Integer

        For i = 0 To AxMapControl1.Map.LayerCount - 1
            j = AxMapControl1.Map.Layer(i)
            If Err.Number Then
                Err.Clear()
            Else
                ComboBox5.Items.Add(AxMapControl1.Map.Layer(i).Name)
                ComboBox5.Text = AxMapControl1.Map.Layer(i).Name
                TINLayerIndex = i
                Exit Sub
            End If
        Next
       End Sub
```

该函数可以在 Form_Load 事件中就调用。系统启动时就会自动搜索 TIN 图层，此外，在添加新的数据后也可以调用它来搜索新的 TIN 图层及位置。

同时，在前面的获取高程值、获取坡度值、获取等高线、获取最陡路径、计算体积、计算表面面积、计算投影面积等的 TIN 图层不再是“AxMapControl1.Map.Layer(7)”，而要通过 TIN 图层的 ComboBox5 的 Text 属性来获得 TIN 图层。此外，通过 Text 来获得图层的序号还需要用一个函数查找搜索以下，该函数就是通过图层的名称获得该图层的序号：

```
    Private Function FindLayerIndex(ByVal LayerName As String) As
Integer
        Dim i As Integer
        For i = 0 To AxMapControl1.Map.LayerCount - 1
```

```
            If LayerName = AxMapControl1.Map.Layer(i).Name Then
                FindLayerIndex = i
                Exit Function
            End If
        Next
    End Function
```

在 AxMapControl1_OnMouseDown 事件中将“AxMapControl1.Map.Layer(7)”改为“AxMapControl1.Map.Layer(FindLayerIndex(ComboBox5.Text))”，在 Button26_Click 、 Button28_Click 、 Button29_Click 事件中将“AxMapControl1.Map.Layer(7)”改为“AxMapControl1.Map.Layer(FindLayerIndex(ComboBox5.Text))”。运行程序，其功能和以前一样，但在寻找 TIN 图层时将会比以前更加方便。

16、选择对象

选择对象在 GIS 开发中非常重要，通常要对数据进行编辑、分析等都要用到。在 Map 开发中，选择数据的方式也有很多，大体可分为两类：第一类就是通过 ArcGIS Engine 自带的工具栏来实现；第二类就是通过程序代码自己设计。

1）通过自带的工具栏选择数据

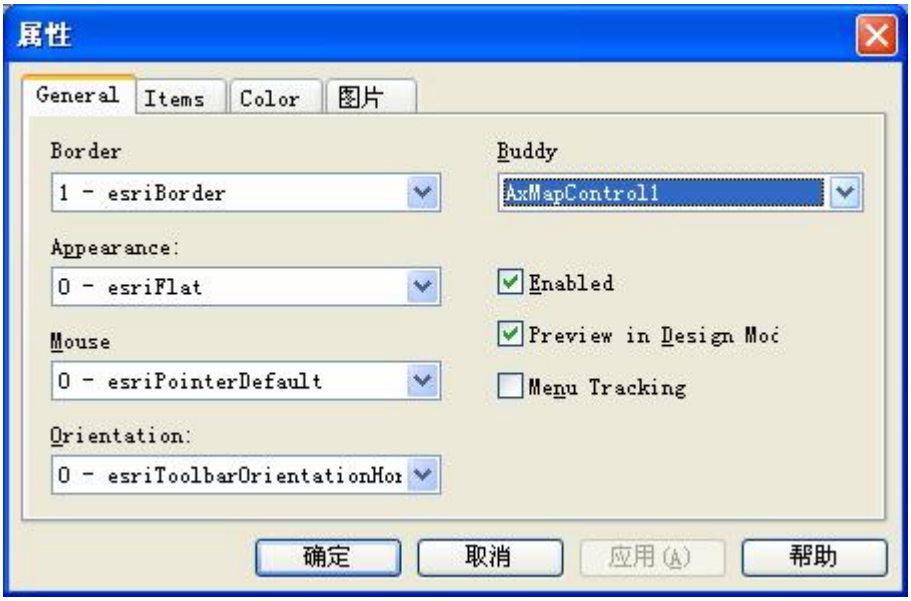
它主要通过 ControlsSelectFeaturesTool 工具来实现，代码如下：

```
    Dim i As ControlsSelectFeaturesTool = New
ControlsSelectFeaturesTool()
    i.OnCreate(AxMapControl1.Object)
    AxMapControl1.CurrentTool = i
```

此外，还可以通过添加一个工具栏，并在工具栏上添加一些常用的按钮，其中就包括选择 Feature 按钮，假如该按钮在工具栏上的序号为 4（序号从 0 开始计算），如下图所示：



且设置其 Buddy 为 AxMapControl1，如下图所示：



通过点击按钮实现该按钮同样的功能，只需要在按钮的 Click 事件中添加如下代码：

```
AxToolbarControl1.GetItem(4).Command.OnCreate(AxMapControl1.Object)
        AxToolbarControl1.CurrentTool =
AxToolbarControl1.GetItem(4).Command
```

其功能和前面一样，都是在 AxToolbarControl1 上选择数据。

2）通过程序设计实现

在 Map 中选择数据主要是通过形状来选择，这些形状主要有矩形、线条、点等，其核心程序就是 AxMapControl1.Map.SelectByShape（）。由于通过点或线得形状来选择数据效率比较低，很多时候很难选择到数据，所以在此以矩形选择为例。其中选择矩形是通过在 AxMapControl1 上点击获得矩形的第一个点，然后通过拖动后再释放鼠标的时候获得矩形的第二个点，所以获得矩形的位置需要通过 AxMapControl1 的 OnMouseDown 事件和 OnMouseUp 事件，此外，当鼠标按下移动时，还需要绘制选择的矩形，所以还要在 AxMapControl1 的 OnMouseMove 事件中添加绘制矩形的代码。

（1）首先，在全局变量中定义

```
Public SelectFeatureOrNot, SelectFeatureDrawShape As Boolean
Private SelectionRecPoint(2) As Double
```

其中，SelectFeatureOrNot 表示是否执行选择数据；SelectFeatureDrawShape 表示是否绘制形状，这个主要用于 AxMapControl1 的 OnMouseMove 事件中。SelectionRecPoint（）用于定义选择矩形的第一个顶点数组。

（2）在 AxMapControl1 的 OnMouseDown 事件中添加如下代码：

```
        If e.button = 1 Then
         If SelectFeatureOrNot Then
             SelectionRecPoint(0) = e.mapX
             SelectionRecPoint(1) = e.mapY
             SelectFeatureDrawShape = True
         End If
```

```
            End If
```

即，如果点击鼠标左键，而且现在开始执行选择数据命令，则设置当前点击的地图位置为选择矩形的第一个顶点。

（3）在 AxMapControl1 的 OnMouseUp 事件中添加如下代码：

```
    Private Sub AxMapControl1_OnMouseUp(ByVal sender As System.Object,
ByVal e As ESRI.ArcGIS.Controls.IMapControlEvents2_OnMouseUpEvent)
Handles AxMapControl1.OnMouseUp
        If SelectFeatureOrNot Then
            Dim pEnv As ESRI.ArcGIS.Carto.ISelectionEnvironment = New
ESRI.ArcGIS.Carto.SelectionEnvironment()
            pEnv.AreaSelectionMethod =
Geodatabase.esriSpatialRelEnum.esriSpatialRelIntersects

            Dim selectionRec As ESRI.ArcGIS.Geometry.IEnvelope = New
ESRI.ArcGIS.Geometry.Envelope()
            selectionRec.XMax = SelectionRecPoint(0)
            selectionRec.YMax = SelectionRecPoint(1)
            selectionRec.XMin = e.mapX
            selectionRec.YMin = e.mapY

            AxMapControl1.Map.SelectByShape(selectionRec, pEnv, False)

            If AxMapControl1.Map.SelectionCount = 0 Then
                MsgBox("没有选中！")
            Else
                SelectFeatureOrNot = False
                AxMapControl1.Refresh()
            End If
            SelectFeatureDrawShape = False
        End If
       End Sub
```

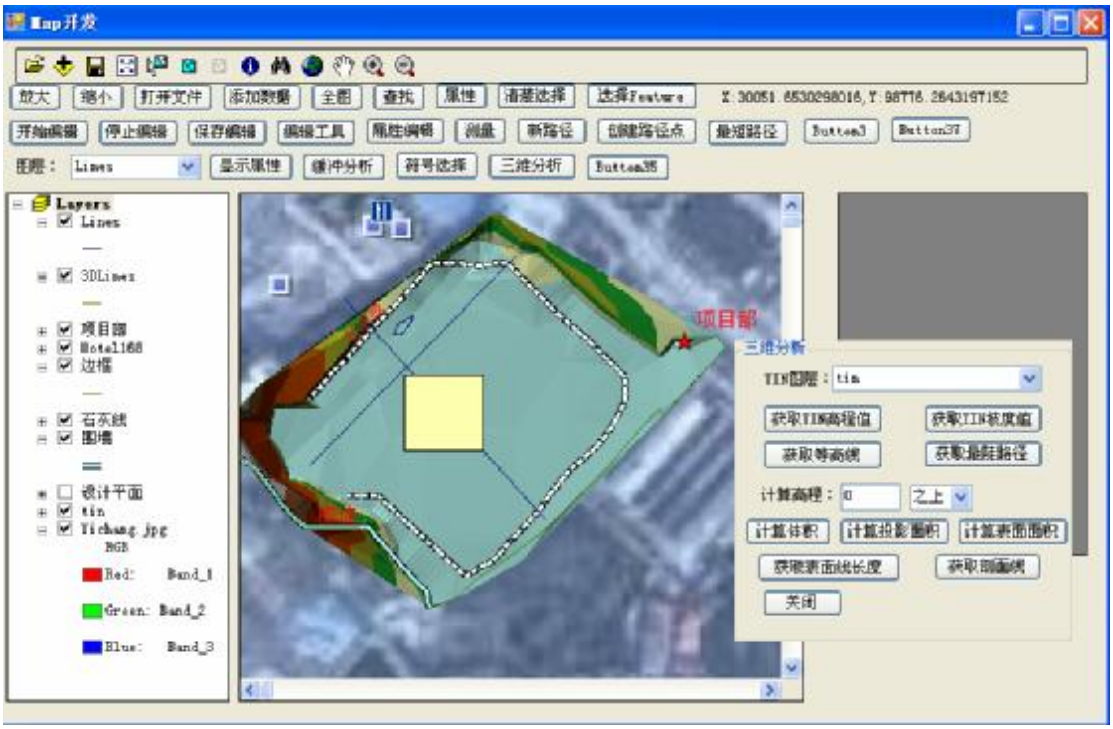
其中，Dim pEnv As ESRI.ArcGIS.Carto.ISelectionEnvironment 是用于定义选择环境，可以设置选择时候的选择模式，如相交或包含等关系时选择数据，还可以通过它的 DefaultColor 属性来设置选择数据的颜色。Dim selectionRec As ESRI.ArcGIS.Geometry.IEnvelope 是定义选择矩阵。如果每次只选择一个数据，则将 AxMapControl1.Map.SelectByShape(selectionRec, pEnv, False)改为 AxMapControl1.Map.SelectByShape(selectionRec, pEnv, True)。

（4）在 AxMapControl1 的 OnMouseMove 事件中添加如下代码：

```
        If SelectFeatureDrawShape Then
            Dim selectionRec As ESRI.ArcGIS.Geometry.IEnvelope = New
ESRI.ArcGIS.Geometry.Envelope()
            selectionRec.XMax = SelectionRecPoint(0)
            selectionRec.YMax = SelectionRecPoint(1)
            selectionRec.XMin = e.mapX
```

```
            selectionRec.YMin = e.mapY
            AxMapControl1.DrawShape(selectionRec)
        End If
```

主要用于在选择数据时显示选择矩阵的大小，如下图所示：



（5）在选择数据按钮上添加事件，其代码如下：

```
    Private Sub Button36_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button36.Click
        SelectFeatureOrNot = True
    End Sub
```

17、获取选择数据

前面是介绍如何选择数据，本节将介绍如何从选择集中获取选择的数据对象。下面添加一些控件，如下图所示，在前面已经建立的“选择数据”按钮基础上，继续添加一个“选择单个数据”的 CheckBox1，添加一个 ComboBox6 用于设置选择的数据所在的图层（因为在选择数据时会选择很多不同图层的数据，但是在调用这些选择的数据的时候需要区分对待，特别是不同类型的数据），添加一个“获取选择数据”按钮：



1）首先，在 AxMapControl1 的 OnMouseUp 事件中将“AxMapControl1.Map.SelectByShape(selectionRec, pEnv, False)”改为

“ AxMapControl1.Map.SelectByShape(selectionRec, pEnv, CheckBox1.Checked)”。

2）然后，在 Form1 的 Load 事件中添加为 ComboBox6 自动添加项目的程序，至此，Form1_Load 的整个程序代码如下：

```
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Dim i As Integer
        For i = 0 To AxMapControl1.LayerCount - 1

ComboBox1.Items.Add(AxMapControl1.Map.Layer(i).Name.ToString())

ComboBox2.Items.Add(AxMapControl1.Map.Layer(i).Name.ToString())

ComboBox6.Items.Add(AxMapControl1.Map.Layer(i).Name.ToString())
        Next
        ComboBox1.Text = ComboBox1.Items(0)
        ComboBox2.Text = ComboBox1.Items(0)
        ComboBox6.Text = ComboBox1.Items(0)
        FindTINLayer()
       End Sub
```

3）为“获取选择数据”按钮添加 Click 事件，其代码如下所示：

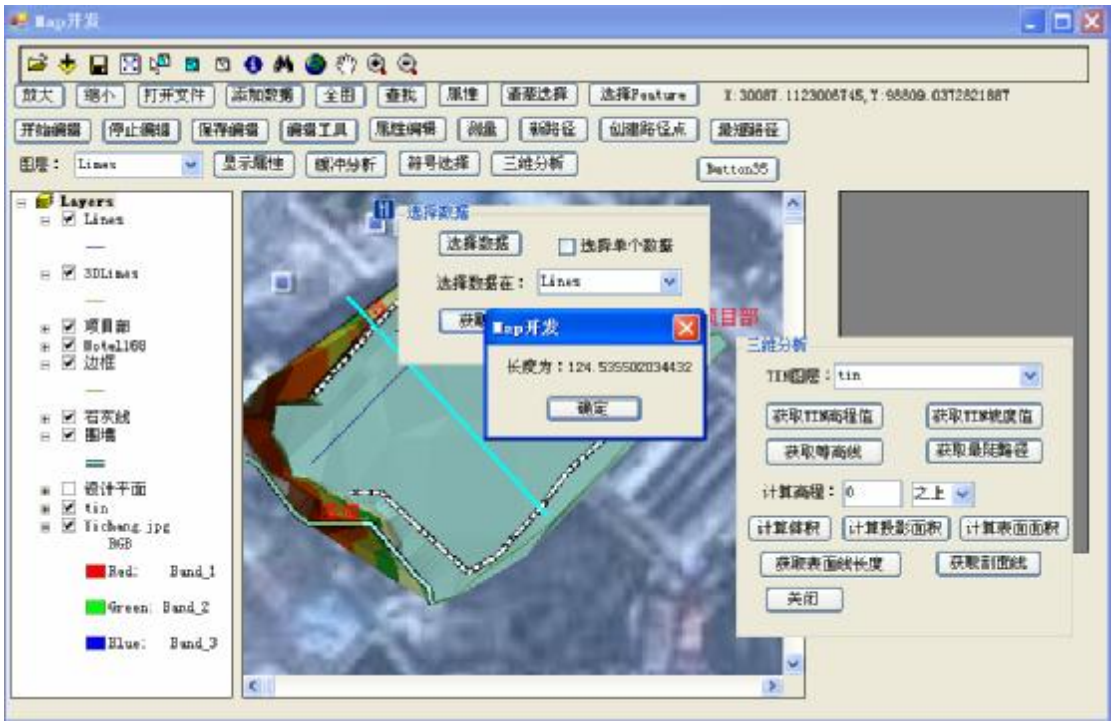
```
    Private Sub Button37_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button37.Click
        Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature = New
ESRI.ArcGIS.Geodatabase.Feature()
        Dim pFeatureCursor As ESRI.ArcGIS.Geodatabase.IFeatureCursor
        Dim pFeatureLayer As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(FindLayerIndex(ComboBox6.Text))
        Dim pFeatureSelection As ESRI.ArcGIS.Carto.IFeatureSelection
        Dim pSelectionSet As ESRI.ArcGIS.Geodatabase.ISelectionSet

        pFeatureSelection = pFeatureLayer
        pSelectionSet = pFeatureSelection.SelectionSet
        If pSelectionSet.Count = 0 Then Exit Sub
        pSelectionSet.Search(Nothing, False, pFeatureCursor)
        pFeature = pFeatureCursor.NextFeature

        Select Case pFeatureLayer.FeatureClass.ShapeType.ToString()
            Case "esriGeometryPolyline"
                Dim pFeatShape As ESRI.ArcGIS.Geometry.IPolyline
                pFeatShape = pFeature.Shape
                MsgBox("长度为： " + pFeatShape.Length.ToString())
            Case "esriGeometryPoint"
                Dim pFeatShape As ESRI.ArcGIS.Geometry.IPoint
```

```
                pFeatShape = pFeature.Shape
                MsgBox("X=" + pFeatShape.X.ToString() + ",Y=" +
pFeatShape.Y.ToString())
            Case "esriGeometryPolygon"
                Dim pFeatShape As ESRI.ArcGIS.Geometry.IPolygon
                pFeatShape = pFeature.Shape
                MsgBox("周长为：" + pFeatShape.Length.ToString())
        End Select
    End Sub
```

其中用到了前面定义的通过图层名称查找图层序号的函数 FindLayerIndex 寻找 ComboBox6.Text 所对应的图层。本例主要是通过 pFeatureCursor 游标找到 pSelectionSet 中的选择了的对象。如果 pSelectionSet 中选择个数为 0，即没有选择数据，将退出程序。本例为当只选择一个 Feature 时的情况，如果选择了两个及以上的对象时，可以用 pFeature = pFeatureCursor.NextFeature 来获得下一个选择对象，当然这些选择的对象都是指在同一个数据图层中的数据。如果要获取所有选择对象，则需要在不同的数据图层中查找这些对象，用错误试探语句退出。运行程序，其结果显示如下图所示：

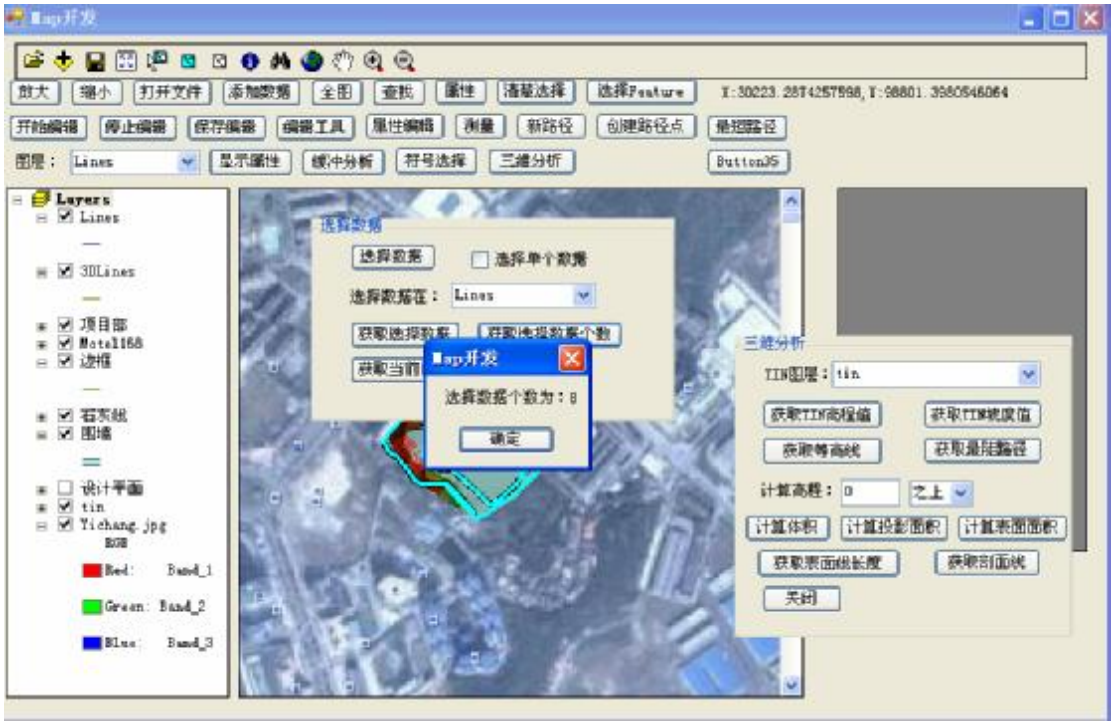


4）获取选择数据个数

获取所有选择数据的个数，其代码如下图所示：

```
        MsgBox("选择数据个数为：" +
AxMapControl1.Map.SelectionCount.ToString())
```

运行程序，其显示结果如下图所示：

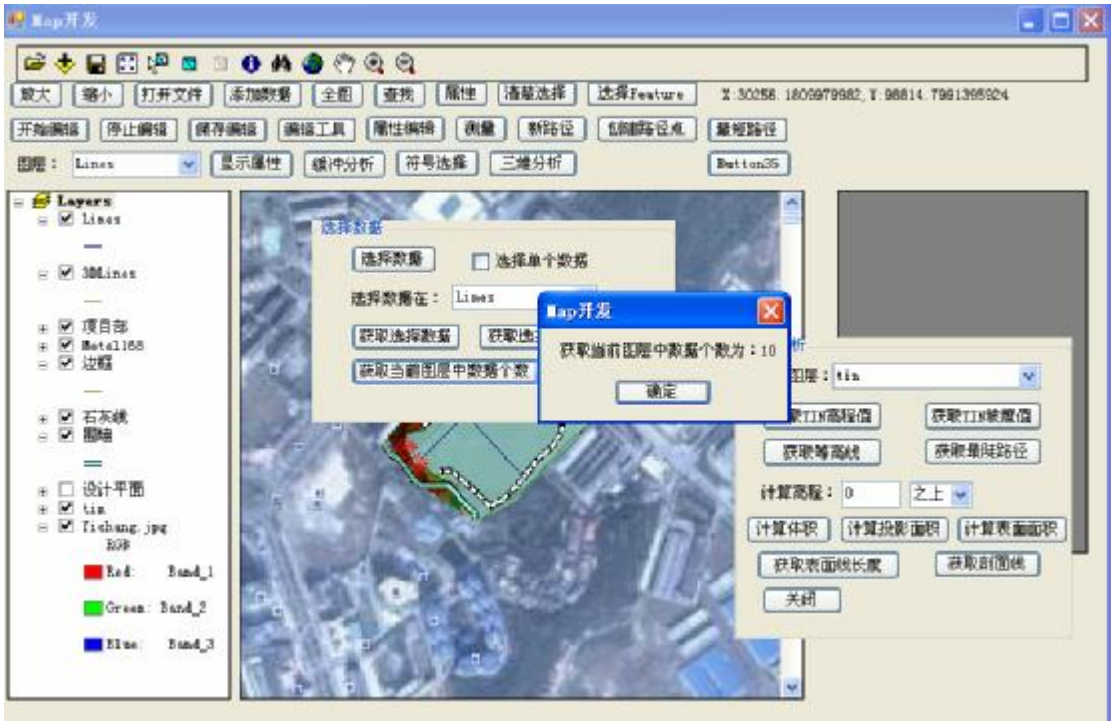


5）获取当前图层中数据个数

获取当前图层中数据的个数，主要用到图层的 FeatureClass 中的 FeatureCount 属性，其代码如下图所示：

```
    Private Sub Button39_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button39.Click
        Dim pFeatureLayer As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(FindLayerIndex(ComboBox6.Text))
        Dim pQueryfilter As ESRI.ArcGIS.Geodatabase.IQueryFilter
        MsgBox("获取当前图层中数据个数为：" +
pFeatureLayer.FeatureClass.FeatureCount(pQueryfilter).ToString())
    End Sub
```

运行程序，其显示结果如下图所示：



18、获取剖面图

获取剖面图主要分为两个部分：第一个部分就是线条的获取；第二部分就是通过获取到的线条根据 TIN 图层绘制剖面线。

1）获取线条

（1）从数据图层中直接获取

平面剖面线可以从已有的线型数据图层中获得，如下代码就是从 Map 中的第一个图层中获得第一条线为平面的剖面线：

```
        Dim player As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.FeatureLayer()
        player = AxMapControl1.Map.Layer(0)
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass =
player.FeatureClass
        Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature =
pFeatureClass.GetFeature(0)
        Dim    pPolyline    As    ESRI.ArcGIS.Geometry.IPolyline    =
pFeature.Shape
```

（2）从选择集中获取线条

结合上一节中介绍的从选择数据中获取数据，设置选择数据的条件，如只能在某个线型数据图层等，其中 pFeature 就是要获取的线条。

（3）在 Map 中绘制线条

在 Map 中绘制线条主要通过在 AxMapControl1 控件上点击两次就可以获取线条的端点，具体在 AxMapControl1 的 OnMouseDown 事件中进行。

首先在全局变量中定义是否绘制线 DrawPolylineOrNot（由于有很多事件都是在 AxMapControl1 的 OnMouseDown 事件中进行的，为了区分不同的功能，需要

在全局变量中区别定义），定义鼠标点击次数 rawPolylineClickCount，定义绘制的线条对象 pDrawPolyline，代码如下：

```
    Public DrawPolylineOrNot As Boolean
    Public DrawPolylineClickCount As Integer
    Public pDrawPolyline As ESRI.ArcGIS.Geometry.IPolyline = New
ESRI.ArcGIS.Geometry.Polyline()
```

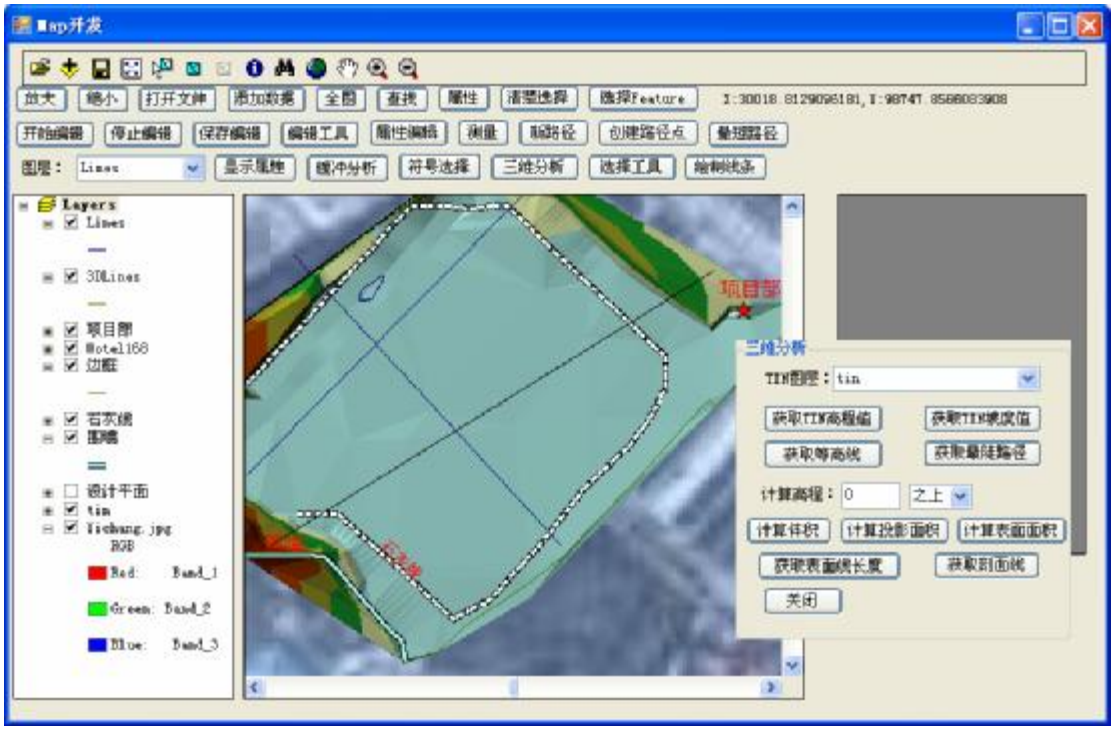
然后再在 AxMapControl1 的 OnMouseDown 事件中添加代码：

```
        If e.button = 1 Then
         If DrawPolylineOrNot Then
             If DrawPolylineClickCount = 0 Then
                 pDrawPolyline.FromPoint = pPoint
                 DrawPolylineClickCount = DrawPolylineClickCount + 1
             ElseIf DrawPolylineClickCount = 1 Then
                 pDrawPolyline.ToPoint = pPoint
                 DrawPolylineClickCount = 0
                 DrawPolylineOrNot = False
                 AxMapControl1.DrawShape(pDrawPolyline)
             End If
            End If
        End If
```

添加一个“绘制线条”按钮，并为该按钮添加事件，其代码如下所示：

```
Private Sub Button35_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button35.Click
        DrawPolylineOrNot = True
End Sub
```

运行程序，其显示结果如下图所示：



为了便于实际操作，在本例中选择第三种方式，即在 Map 中绘制线条来获得绘制剖面的线条。

2）绘制剖面线

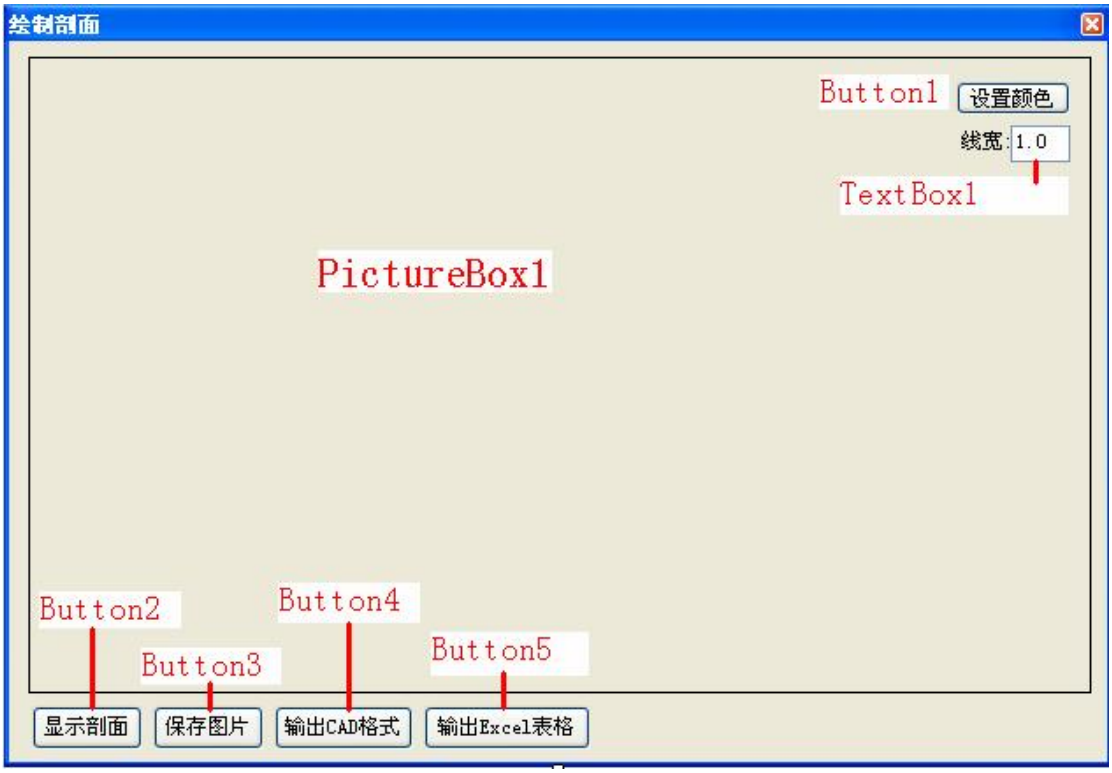
绘制剖面线主要用 ISurface 接口的 GetProfile 属性。获取到的新的线条（剖面线）又可以通过 GetSubcurve 属性来分隔，得到随着它的距离变化高程也随之变化的两个数组（或是一个二维数组）。通过这两个数组就可以在其它控件上绘制剖面图，包括输出常用的数据格式。

（1）首先，添加一个“获取剖面线”按钮，并为按钮添加如下所示代码：

```
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
        Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        j = AxMapControl1.Map.Layer(FindLayerIndex(ComboBox5.Text))
        pTIN = j.Dataset
        Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
        pSurface = pTIN
        Dim pnewPolyline As ESRI.ArcGIS.Geometry.IPolyline = New
ESRI.ArcGIS.Geometry.Polyline()
        pSurface.GetProfile(pDrawPolyline, pnewPolyline)
        Dim i As Integer
        Dim DistanceValues(), ZValues() As Double
        ReDim Preserve DistanceValues(0) : ReDim Preserve ZValues(0)
        DistanceValues(0) = 0
        ZValues(0) = pnewPolyline.ToPoint.Z
        For i = 1 To Int(pnewPolyline.Length)
            ReDim Preserve DistanceValues(i)
            ReDim Preserve ZValues(i)
            DistanceValues(i) = i
            Dim pCurve As ESRI.ArcGIS.Geometry.ICurve
            pnewPolyline.GetSubcurve(0, DistanceValues(i), False,
pCurve)
            ZValues(i) = pCurve.ToPoint.Z
        Next
        ReDim Preserve DistanceValues(i + 1)
        ReDim Preserve ZValues(i + 1)
        DistanceValues(i) = pnewPolyline.Length
           ZValues(i) = pnewPolyline.ToPoint.Z
```

其中，数组 DistanceValues（）就是距离数组，数组 ZValues（）是高程数组，要绘制剖面图也只需要这两个数据和数组的长度即可。

（2）新建一个绘图剖面的 Windows 窗体，主要用于显示剖面图及剖面图数据格式输出，该窗体中所要添加的主要控件如下图所示：



共有一个

该 PictureBox 控件，5 个 Button 控件，一个 TextBox 控件，一个 Label 控件。该窗体中所有的程序代码如下所示：

```
Public Class 绘制剖面
    Public x() As Double
    Public y() As Double
    Public count As Integer
    Private DrawLineColor As System.Drawing.Color = Color.Black
    Private DrawLineWidth As Double = 1.0
    Public Sub DrawLineControl(ByVal xx() As Double, ByVal yy() As Double,
ByVal count As Integer, Optional ByVal coordinateX As Double = 20,
Optional ByVal coordinateY As Double = 20)
        Dim MyImage1 As System.Drawing.Image = New
System.Drawing.Bitmap(PictureBox1.Width, PictureBox1.Height)
        Dim g As Graphics = System.Drawing.Graphics.FromImage(MyImage1)
        Dim precf As System.Drawing.RectangleF
        precf.X = 0 : precf.Y = 0
        precf.Width = PictureBox1.Width : precf.Height =
PictureBox1.Height

        g.FillRectangle(Brushes.White, precf)

        Dim pWidth, pHeight As Double
        pWidth = PictureBox1.Width - coordinateX - 20
        pHeight = PictureBox1.Height - coordinateY - 20
```

```
        Dim Xmax, Xmin, Ymax, Ymin As Double
        Xmax = xx(0)
        Xmin = xx(0)
        Ymax = yy(0)
        Ymin = yy(0)
        Dim maxValueIndex, minValueIndex As Integer
        Dim i As Integer
        For i = 0 To count - 1
            If Xmax < xx(i) Then
                Xmax = xx(i)
            End If
            If Xmin > xx(i) Then
                Xmin = xx(i)
            End If
            If Ymax < yy(i) Then
                Ymax = yy(i)
                maxValueIndex = i
            End If
            If Ymin > yy(i) Then
                Ymin = yy(i)
                minValueIndex = i
            End If
        Next
        If Ymax = Ymin Or Xmax = Xmin Then Exit Sub

        Dim pf(count - 1) As System.Drawing.PointF

        For i = 0 To count - 1
            pf(i).X = xx(i) * pWidth / (Xmax - Xmin) + coordinateX
            pf(i).Y = pHeight - (yy(i) - Ymin) * pHeight / (Ymax - Ymin)
+ coordinateY
        Next
        Dim textFont As System.Drawing.Font = New System.Drawing.Font("
宋体", 10, FontStyle.Bold)

        Dim LinePen As System.Drawing.Pen = New
System.Drawing.Pen(DrawLineColor)
        LinePen.Width = DrawLineWidth

        g.DrawCurve(LinePen, pf)
        g.DrawLine(Pens.Black, CSng(coordinateX), CSng(20.0),
CSng(coordinateX), CSng(PictureBox1.Height - coordinateY))  '绘制Y轴
        g.DrawLine(Pens.Black, CSng(coordinateX), CSng(20.0),
CSng(coordinateX - 5), CSng(30.0))  '绘制Y轴箭头
```

```
        g.DrawLine(Pens.Black, CSng(coordinateX), CSng(20.0),
CSng(coordinateX + 5), CSng(30.0))  '绘制Y轴箭头
        g.DrawString("高程", textFont, Brushes.Red, CSng(coordinateX -
10), CSng(5.0)) '绘制Y轴标签


        g.DrawLine(Pens.Black, CSng(coordinateX),
CSng(PictureBox1.Height - coordinateY), CSng(PictureBox1.Width -
coordinateX), CSng(PictureBox1.Height - coordinateY)) '绘制X轴
        g.DrawLine(Pens.Black, CSng(PictureBox1.Width - coordinateX),
CSng(PictureBox1.Height - coordinateY), CSng(PictureBox1.Width -
coordinateX - 10), CSng(PictureBox1.Height - coordinateY + 5)) '绘制X
轴
        g.DrawLine(Pens.Black, CSng(PictureBox1.Width - coordinateX),
CSng(PictureBox1.Height - coordinateY), CSng(PictureBox1.Width -
coordinateX - 10), CSng(PictureBox1.Height - coordinateY - 5)) '绘制X
轴
        g.DrawString("距离", textFont, Brushes.Red,
CSng(PictureBox1.Width - coordinateX - 10), CSng(PictureBox1.Height -
coordinateY - 20)) '绘制X轴标签

        g.DrawString("原点", textFont, Brushes.Red, CSng(coordinateX -
10), CSng(PictureBox1.Height - coordinateY)) '绘制X轴标签

        g.DrawString(Ymax.ToString(), textFont, Brushes.Red,
pf(maxValueIndex)) '绘制最大值
        g.DrawString(Ymin.ToString(), textFont, Brushes.Red,
pf(minValueIndex)) '绘制最小值
        PictureBox1.Image = MyImage1
        g.Dispose()
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim colorDg As ColorDialog = New ColorDialog()
        colorDg.ShowDialog()
        DrawLineColor = colorDg.Color
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        DrawLineControl(x, y, count)
    End Sub
```

```
    Private Sub TextBox1_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles TextBox1.TextChanged
        DrawLineWidth = TextBox1.Text
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        On Error GoTo handle01
        Dim pSavaImage As SaveFileDialog = New SaveFileDialog()
        pSavaImage.Title = "保存剖面图"
        pSavaImage.Filter = "Jpeg格式 (*.jpg)|*.jpg|Bmp格式
(*.bmp)|*.bmp|Png格式 (*.png)|*.png|TIFF格式 (*.tif)|*.tif|All files
(*.*)|*.*"
        pSavaImage.ShowDialog()
        Dim ImagesaveFile As String
        ImagesaveFile = pSavaImage.FileName

        Select Case pSavaImage.FilterIndex
            Case 1
                PictureBox1.Image.Save(ImagesaveFile,
System.Drawing.Imaging.ImageFormat.Jpeg)
            Case 2
                PictureBox1.Image.Save(ImagesaveFile,
System.Drawing.Imaging.ImageFormat.Bmp)
            Case 3
                PictureBox1.Image.Save(ImagesaveFile,
System.Drawing.Imaging.ImageFormat.Png)
            Case 4
                PictureBox1.Image.Save(ImagesaveFile,
System.Drawing.Imaging.ImageFormat.Tiff)
        End Select
        MsgBox("保存图像文件成功！")
        Exit Sub
handle01:
        MsgBox(Err.Description)
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
        On Error GoTo handle01
        Dim AcadApp As Object
        On Error Resume Next
        AcadApp = GetObject(, "AutoCAD.Application")
        If Err.Number Then
```

```
            Err.Clear()
            AcadApp = CreateObject("AutoCAD.Application")

        End If

        Dim dg As New SaveFileDialog
        dg.Filter = "CAD文件(*.dwg)|*.dwg"
        dg.ShowDialog()
        Dim savePathname As String
        savePathname = dg.FileName


'=====================================================================
        Dim i As Integer
        Dim points(2 * (count - 1) + 1) As Double
        For i = 0 To count - 1
            points(i * 2) = x(i)
            points(i * 2 + 1) = y(i)
        Next

        Dim Ymax As Double = y(0)
        For i = 0 To count - 1
            If Ymax < y(i) Then
                Ymax = y(i)
            End If
        Next


AcadApp.ActiveDocument.ModelSpace.AddLightWeightPolyline(points)

        '绘制X轴
        Dim pointsXAxis(3), XAxisTextCord(2), XAxisArrowCord(5) As
Double
        pointsXAxis(0) = 0 : pointsXAxis(1) = 0
        pointsXAxis(2) = x(count - 1) + 10 : pointsXAxis(3) = 0
        XAxisTextCord(0) = pointsXAxis(2) : XAxisTextCord(1) =
pointsXAxis(3)
        XAxisArrowCord(0) = x(count - 1) : XAxisArrowCord(1) = 5
        XAxisArrowCord(2) = x(count - 1) + 10 : XAxisArrowCord(3) = 0
        XAxisArrowCord(4) = x(count - 1) : XAxisArrowCord(5) = -5

AcadApp.ActiveDocument.ModelSpace.AddLightWeightPolyline(pointsXAxis)
        AcadApp.ActiveDocument.ModelSpace.AddText("距离",
XAxisTextCord, 6)
```

```
AcadApp.ActiveDocument.ModelSpace.AddLightWeightPolyline(XAxisArrowCo
rd)

        '绘制Y轴
        Dim pointsYAxis(3), YAxisTextCord(2), YAxisArrowCord(5) As
Double
        pointsYAxis(0) = 0 : pointsYAxis(1) = 0
        pointsYAxis(2) = 0 : pointsYAxis(3) = Ymax + 20
        YAxisTextCord(0) = pointsYAxis(2) : YAxisTextCord(1) = Ymax + 20
        YAxisArrowCord(0) = -5 : YAxisArrowCord(1) = Ymax + 10
        YAxisArrowCord(2) = 0 : YAxisArrowCord(3) = Ymax + 20
        YAxisArrowCord(4) = 5 : YAxisArrowCord(5) = Ymax + 10

AcadApp.ActiveDocument.ModelSpace.AddLightWeightPolyline(pointsYAxis)
        AcadApp.ActiveDocument.ModelSpace.AddText("高程",
YAxisTextCord, 6)

AcadApp.ActiveDocument.ModelSpace.AddLightWeightPolyline(YAxisArrowCo
rd)

        '绘制原点
        Dim XYAxisTextCord(2) As Double
        XYAxisTextCord(0) = -10 : XYAxisTextCord(1) = -10
        AcadApp.ActiveDocument.ModelSpace.AddText("0,0",
XYAxisTextCord, 6)

'=====================================================================

        AcadApp.ActiveDocument.SaveAs(savePathname)
        AcadApp.Quit()
        MsgBox("保存CAD文件成功！")
        Exit Sub
handle01:
        MsgBox(Err.Description)
    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
        On Error GoTo handle01
        Dim oExcel As Object
        Dim oBook As Object
        Dim oSheet As Object
```

```
        oExcel = CreateObject("Excel.Application")
        oBook = oExcel.Workbooks.Add

        oSheet = oBook.Worksheets(1)
        Dim i As Integer
        Dim s1, s2 As String
        oSheet.Range("A1") = "距离"
        oSheet.Range("B1") = "高程"
        For i = 0 To count - 1
            s1 = Chr(65) + (i + 2).ToString()  '即A1
            s2 = x(i)
            oSheet.Range(s1) = s2
        Next
        For i = 0 To count - 1
            s1 = Chr(66) + (i + 2).ToString()  '即B1
            s2 = y(i)
            oSheet.Range(s1) = s2
        Next

        Dim dg As New SaveFileDialog
        dg.Filter = "Excel表格(*.xls)|*.xls"
        dg.ShowDialog()
        Dim savePathname As String
        savePathname = dg.FileName
        oBook.SaveAs(savePathname)
        oExcel.Quit()
        MsgBox("保存Excel文件成功！")
        Exit Sub
handle01:
        MsgBox(Err.Description)
    End Sub
   End Class
```

（3）在“获取剖面线”按钮的 Click 事件中添加绘制剖面的代码，如下所示：

```
        Dim newform As 绘制剖面= New 绘制剖面()
        newform.x = DistanceValues
        newform.y = ZValues
        newform.count = i + 1
           newform.Show()
```

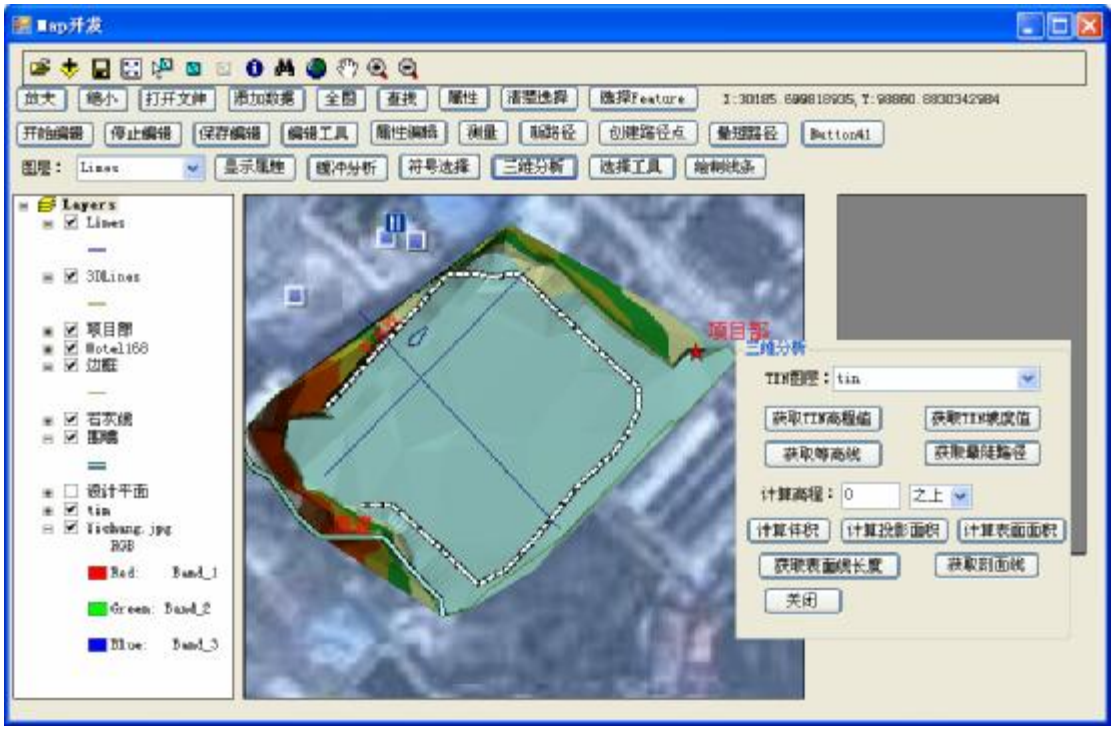
至此，“获取剖面线”按钮的 Click 事件中所有的代码如下：

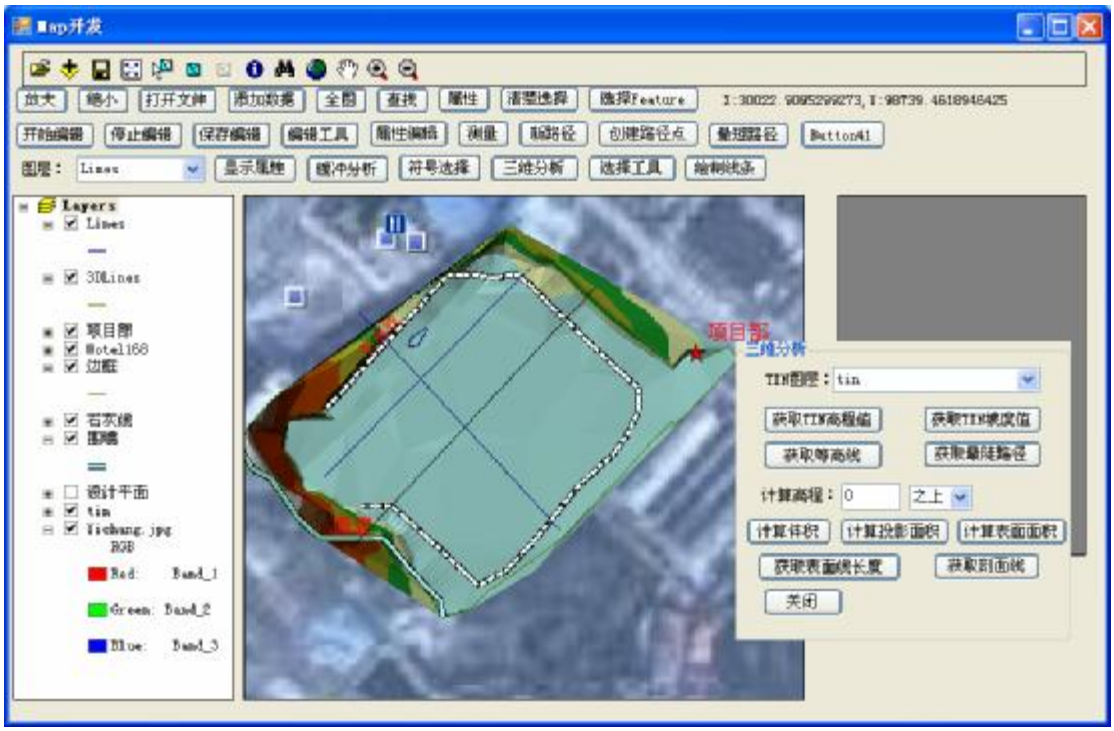
```
    Private Sub Button34_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button34.Click
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITin
        Dim j As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
```

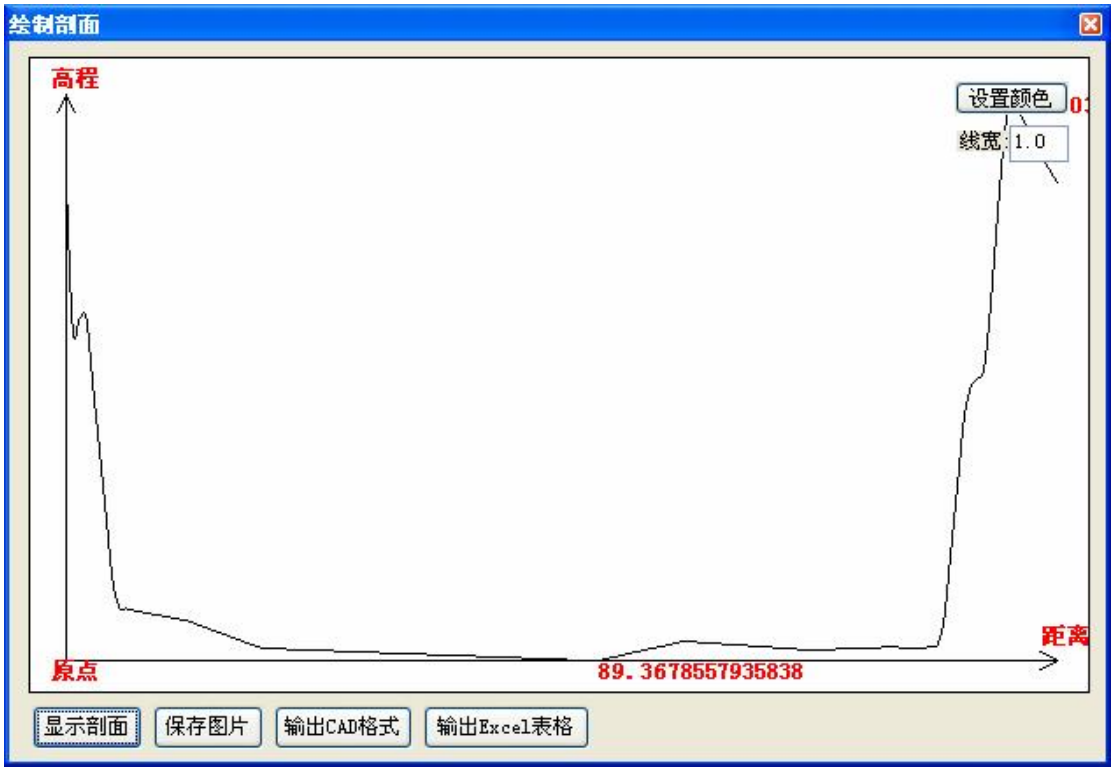
```
        j = AxMapControl1.Map.Layer(FindLayerIndex(ComboBox5.Text))
        pTIN = j.Dataset
        Dim pSurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
        pSurface = pTIN
        Dim pnewPolyline As ESRI.ArcGIS.Geometry.IPolyline = New
ESRI.ArcGIS.Geometry.Polyline()
        pSurface.GetProfile(pDrawPolyline, pnewPolyline)
        Dim i As Integer
        Dim DistanceValues(), ZValues() As Double
        ReDim Preserve DistanceValues(0) : ReDim Preserve ZValues(0)
        DistanceValues(0) = 0
        ZValues(0) = pnewPolyline.ToPoint.Z
        For i = 1 To Int(pnewPolyline.Length)
            ReDim Preserve DistanceValues(i)
            ReDim Preserve ZValues(i)
            DistanceValues(i) = i
            Dim pCurve As ESRI.ArcGIS.Geometry.ICurve
            pnewPolyline.GetSubcurve(0, DistanceValues(i), False,
pCurve)
            ZValues(i) = pCurve.ToPoint.Z
        Next
        ReDim Preserve DistanceValues(i + 1)
        ReDim Preserve ZValues(i + 1)
        DistanceValues(i) = pnewPolyline.Length
        ZValues(i) = pnewPolyline.ToPoint.Z

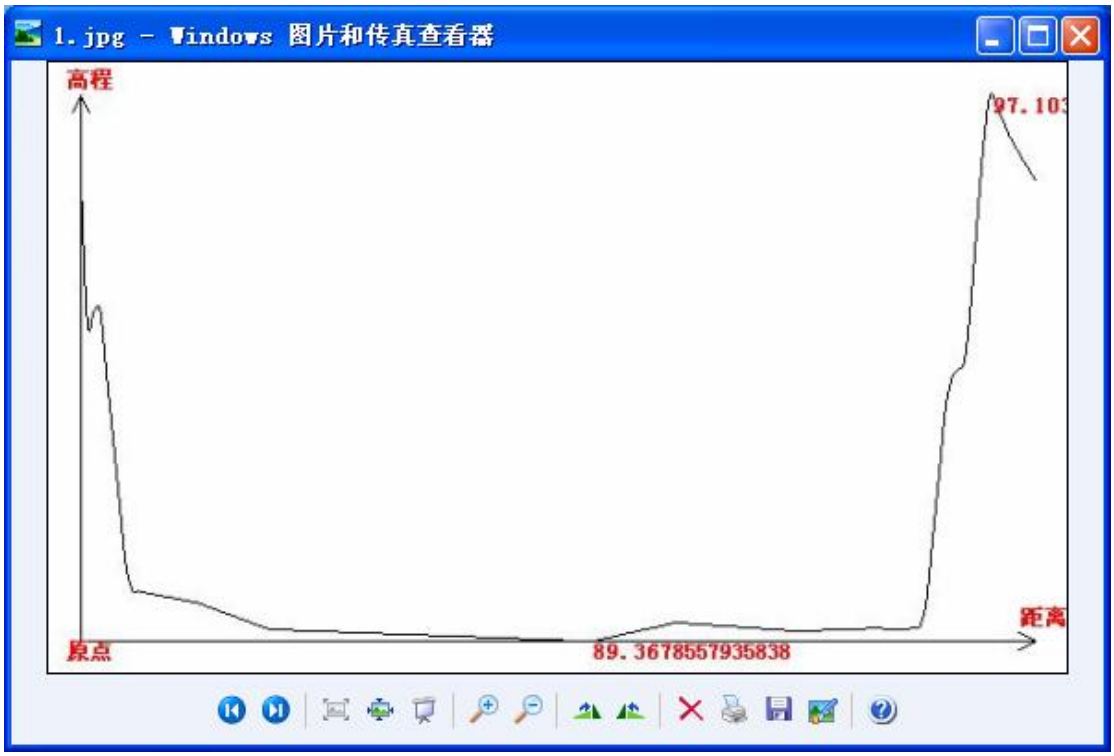
        Dim newform As 绘制剖面= New 绘制剖面()
        newform.x = DistanceValues
        newform.y = ZValues
        newform.count = i + 1
        newform.Show()
    End Sub
```

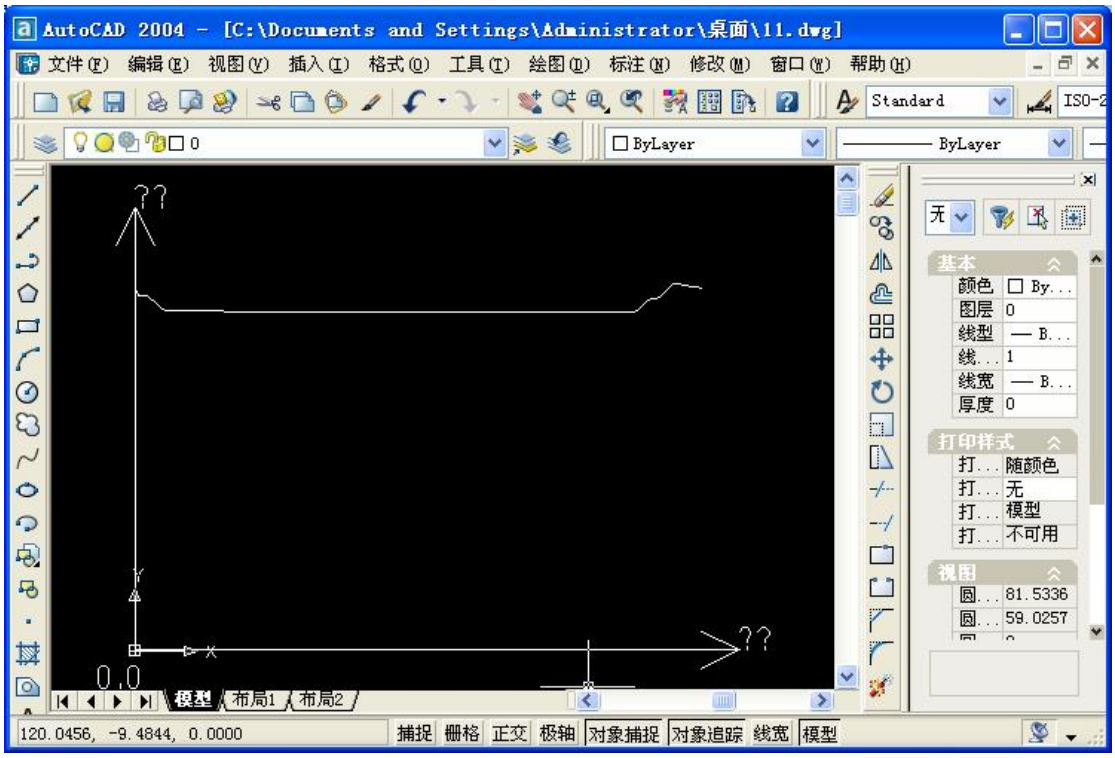
运行程序，整个过程如下图所示：

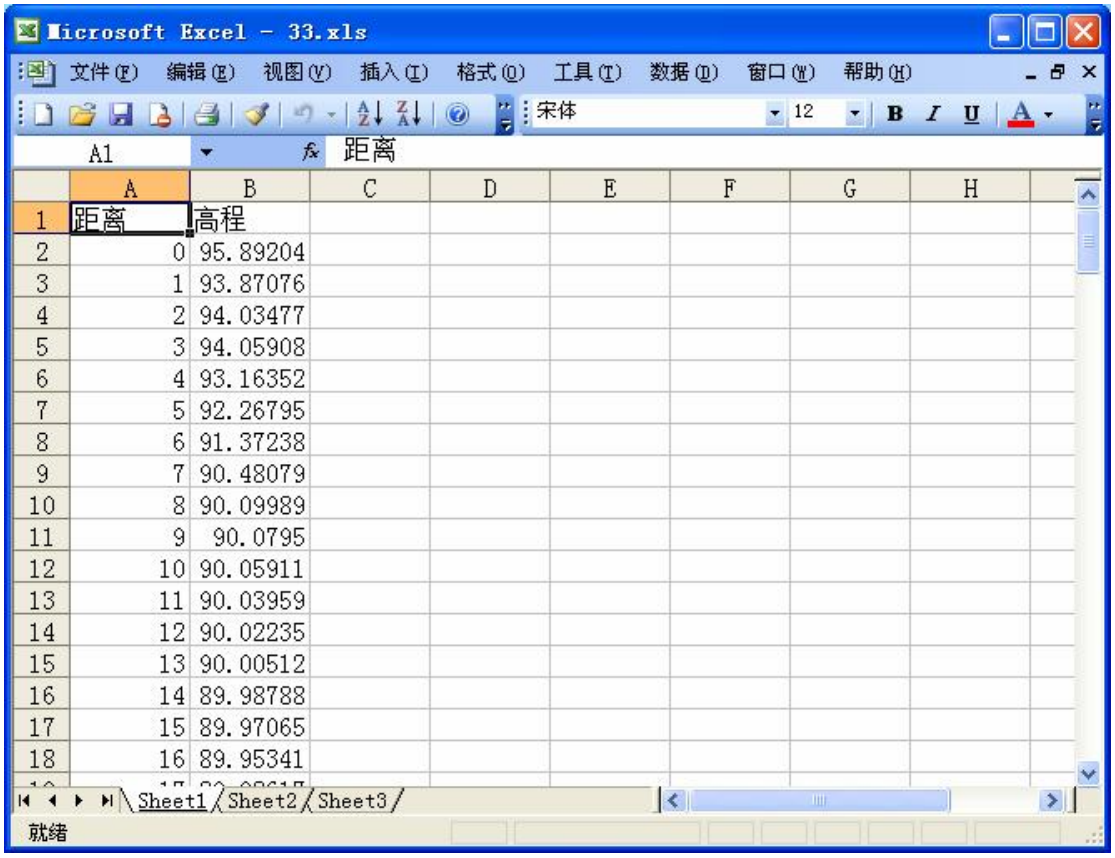

Map开发
放大
缩小
打开文件
添加数据
全图
查找
属性
清除选择
选择Feature
开始编辑
停止编辑
保存编辑
编辑工具
属性编辑
测量
创建路径
创建路径点
量测路径
Button41
图层：
Lines
显示属性
缓冲分析
符号选择
三维分析
选择工具
Layers
Lines
3DLines
项目部
Hotel168
边框
等高线
图廓
设计平面
tin
Yichang.jpg
RGB
Red: Band_1
Green: Band_2
Blue: Band_3
三维分析
TIN图层：tin
获取TIN高程值
获取TIN坡度值
获取等高线
获取最短路径
计算高程：0
之上
计算体积
计算投影面积
计算表面面积
获取表面线长度
获取剖面线
关闭


Map开发
放大
缩小
打开文件
添加数据
全图
查找
属性
清除选择
选择Feature
开始编辑
停止编辑
保存编辑
编辑工具
属性编辑
测量
创建路径
创建路径点
量测路径
Button41
图层：
Lines
显示属性
缓冲分析
符号选择
三维分析
选择工具
Layers
Lines
3DLines
项目部
Hotel168
边框
等高线
图廓
设计平面
tin
Yichang.jpg
RGB
Red: Band_1
Green: Band_2
Blue: Band_3
三维分析
TIN图层：tin
获取TIN高程值
获取TIN坡度值
获取等高线
获取最短路径
计算高程：0
之上
计算体积
计算投影面积
计算表面面积
获取表面线长度
获取剖面线
关闭


绘制剖面
高程
设置颜色
线宽:
1.0
距离
原点
89.3678557935838
显示剖面
保存图片
输出CAD格式
输出Excel表格


1.jpg - Windows 图片和传真查看器
高程
97.10
距离
原点
89.3678557935838





	A	B
1	距离	高程
2	0	95.89204
3	1	93.87076
4	2	94.03477
5	3	94.05908
6	4	93.16352
7	5	92.26795
8	6	91.37238
9	7	90.48079
10	8	90.09989
11	9	90.0795
12	10	90.05911
13	11	90.03959
14	12	90.02235
15	13	90.00512
16	14	89.98788
17	15	89.97065
18	16	89.95341

19、闪烁动画并缩放至数据

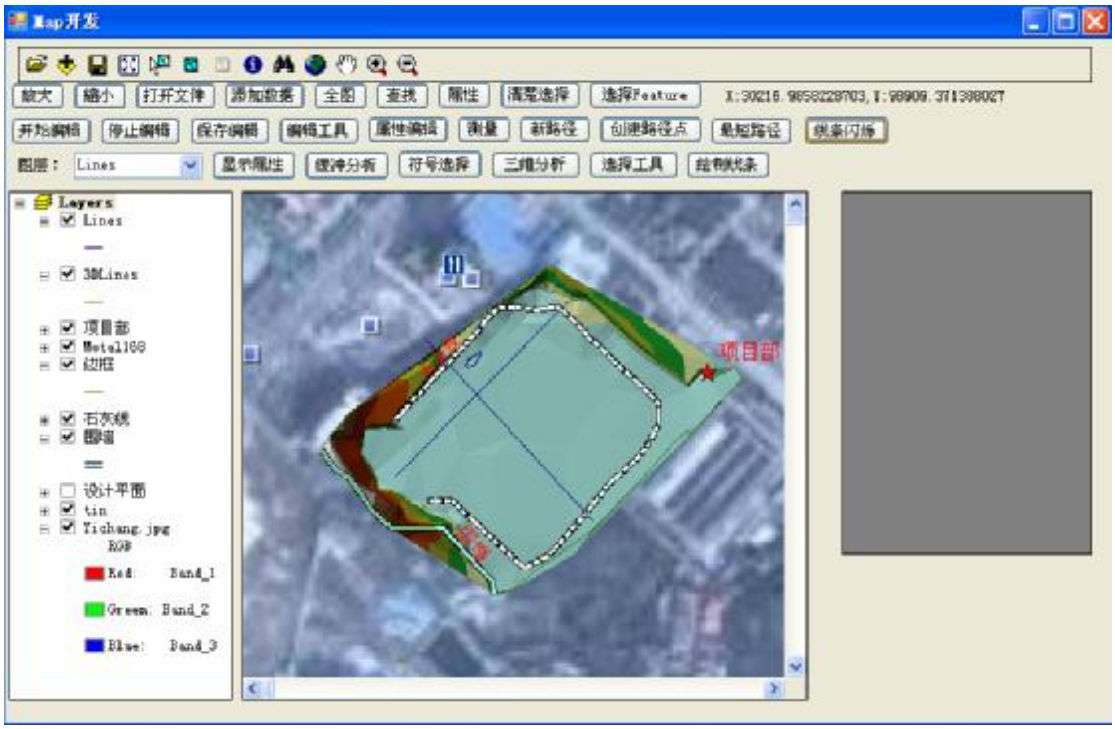
缩放至数据主要用到 AxMapControl 的 Extent 属性和 IGeometry 的 Envelope 属性，而闪烁动画则主要用到 AxMapControl 的 FlashShape（）方法。

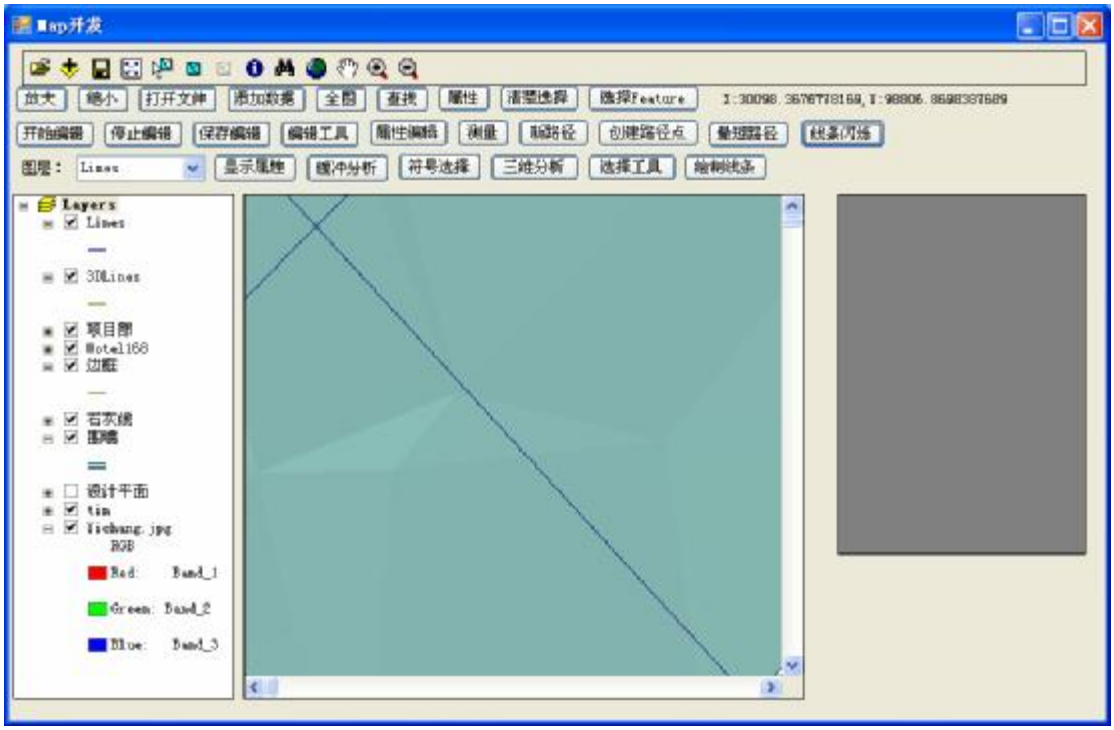
1）线型数据的闪烁

添加一个“线条闪烁”的按钮，为该按钮添加如下所示代码：

```
 Private Sub Button41_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button41.Click
        Dim pLineSymbol As ESRI.ArcGIS.Display.ILineSymbol = New
ESRI.ArcGIS.Display.SimpleLineSymbolClass()
        pLineSymbol.Width = 5
        Dim pGeometry As ESRI.ArcGIS.Geometry.IGeometry
        pGeometry = pDrawPolyline
        AxMapControl1.FlashShape(pGeometry, 5, 100, pLineSymbol)
        AxMapControl1.Extent = pDrawPolyline.Envelope
        AxMapControl1.Refresh()
      End Sub
```

该代码中使用的线条几何对象就是上一节中介绍的在 Map 中绘制线条。运行程序，其显示结果如下图所示：



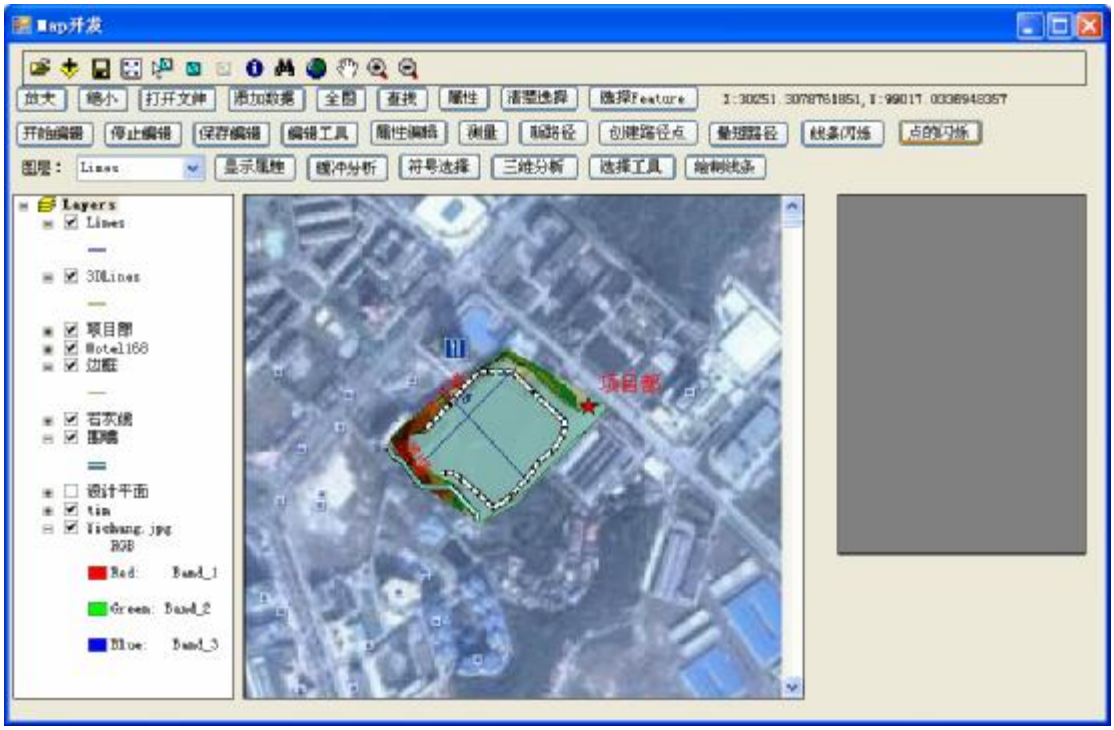


2）点数据的闪烁

添加一个“点的闪烁”的按钮，为该按钮添加如下所示代码：

```
 Private Sub Button42_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button42.Click
        Dim pGeometry As ESRI.ArcGIS.Geometry.IGeometry
        Dim pLayer As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(2)
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass
        Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature
        pFeatureClass = pLayer.FeatureClass
        pFeature = pFeatureClass.GetFeature(0)
        pGeometry = pFeature.Shape
        Dim pMarkerSymbol As ESRI.ArcGIS.Display.IMarkerSymbol = New
ESRI.ArcGIS.Display.SimpleMarkerSymbolClass()
        AxMapControl1.FlashShape(pGeometry, 5, 100, pMarkerSymbol)
       End Sub
```

该代码中使用的点几何对象来自于 Map 中的第三个图层的第一个数据。运行程序，其显示结果如下图所示：

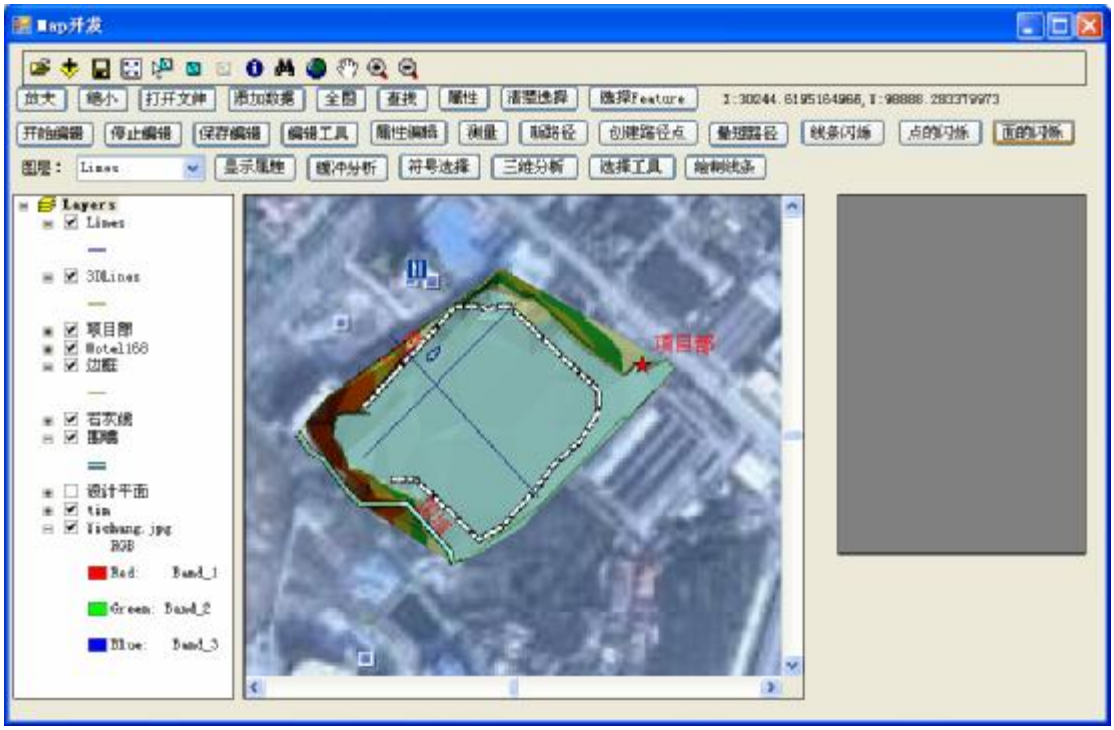


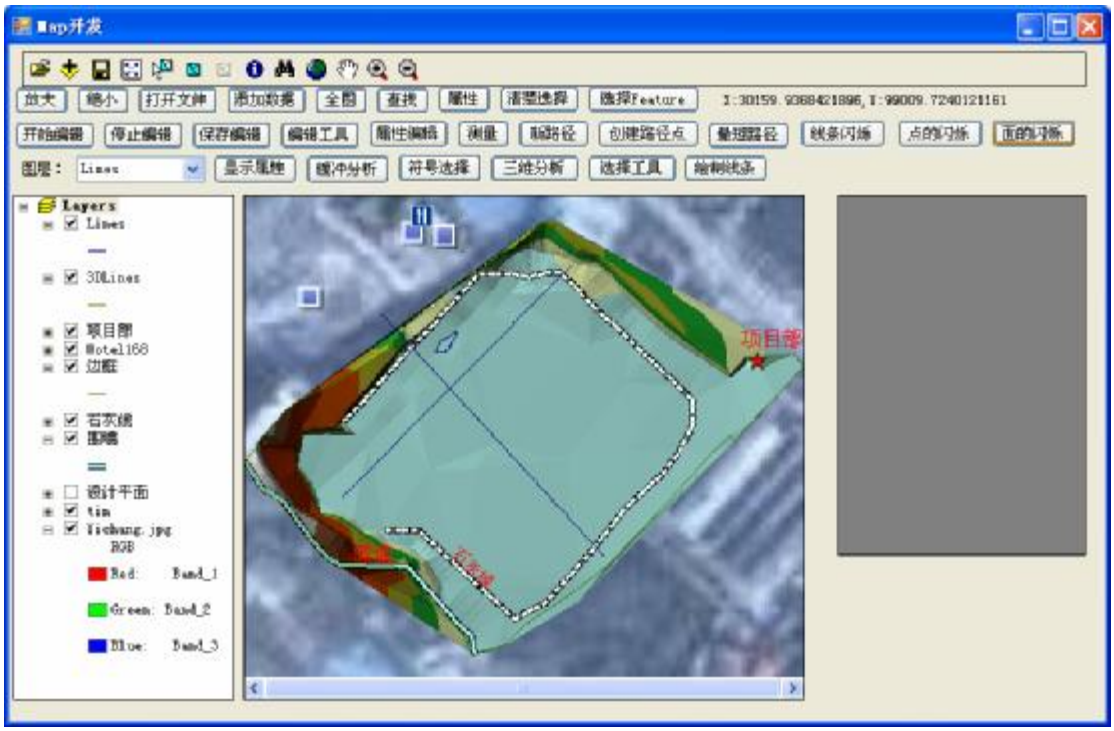
3）面数据的闪烁

添加一个“面的闪烁”的按钮，为该按钮添加如下所示代码：

```
 Private Sub Button43_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button43.Click
        Dim pGeometry As ESRI.ArcGIS.Geometry.IGeometry
        Dim pLayer As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(7)
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass
        Dim pFeature As ESRI.ArcGIS.Geodatabase.IFeature
        pFeatureClass = pLayer.FeatureClass
        pFeature = pFeatureClass.GetFeature(0)
        pGeometry = pFeature.Shape
        Dim pFillSymbol As ESRI.ArcGIS.Display.IFillSymbol = New
ESRI.ArcGIS.Display.SimpleFillSymbolClass()
        pFillSymbol.Outline.Width = 3
        AxMapControl1.FlashShape(pGeometry, 5, 100, pFillSymbol)
        AxMapControl1.Extent = pGeometry.Envelope
        AxMapControl1.Refresh()
       End Sub
```

该代码中使用的点几何对象来自于 Map 中的第八个图层的第一个数据。运行程序，其显示结果如下图所示：





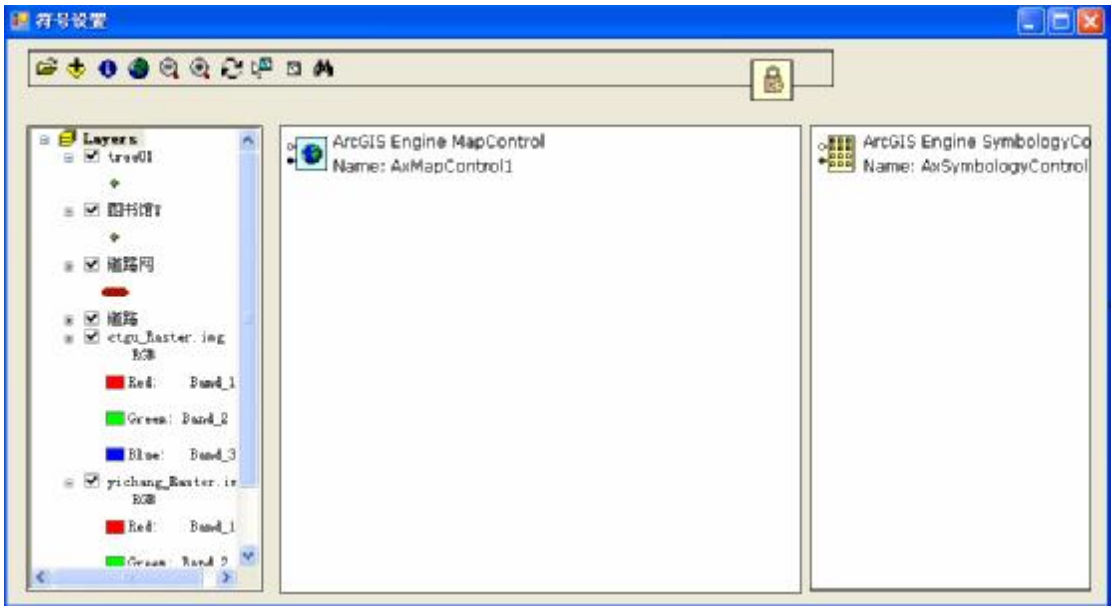
4）此外，可以通过定义一个函数将闪烁各类数据包含，其代码如下所示：

```
 Public Sub DoFlashShape(ByVal pGeometry As IGeometry)
      pGeometry.SpatialReference = axMap1.SpatialReference
      pGeometry.SnapToSpatialReference()
      Select Case pGeometry.GeometryType
          Case esriGeometryType.esriGeometryPoint
              Dim pMarkerSymbol As IMarkerSymbol = New
SimpleMarkerSymbolClass()
```

```
                axMap1.FlashShape(pGeometry, 3, 500, pMarkerSymbol)
                Exit Select
            Case esriGeometryType.esriGeometryPolyline
                Dim pLineSymbol As ILineSymbol = New
SimpleLineSymbolClass()
                axMap1.FlashShape(pGeometry, 3, 500, pLineSymbol)
                Exit Select
            Case esriGeometryType.esriGeometryPolygon
                Dim pFillSymbol As IFillSymbol = New
SimpleFillSymbolClass()
                pFillSymbol.Outline.Width = 3
                axMap1.FlashShape(pGeometry, 3, 500, pFillSymbol)
        End Select
       End Sub
```

20、更改点的显示符号

整个实现过程就是：MapControl——IFeatureLayer——IGeoFeatureLayer——ISimpleRenderer——IStyleGalleryItem——ISymbologyStyleClass——SymbologyControl，通过这个过程即可实现 MapControl 中某个图层的符号改变。首先新建一个工程，并添加一个 ToolbarControl 控件、TOCControl 控件、MapControl 控件、SymbologyControl 控件和 LicenseControl 控件如下图所示控件：



1）首先，添加一个“导入符号库”的按钮，其 Click 事件的程序代码如下：

```
 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim regKey As Microsoft.Win32.RegistryKey =
Microsoft.Win32.Registry.LocalMachine.OpenSubKey("SOFTWARE\\ESRI\\Cor
eRuntime", True)
```

```
AxSymbologyControl1.LoadStyleFile(regKey.GetValue("InstallDir") &
"\\Styles\\ESRI.ServerStyle")
        End Sub
```

其中 regKey.GetValue 是获取到 ArcGIS 符号库所在目录。ESRI.ServerStyle 是符号样式文件。

2）添加一个“设置点的符号”的按钮，其 Click 事件的程序代码如下：

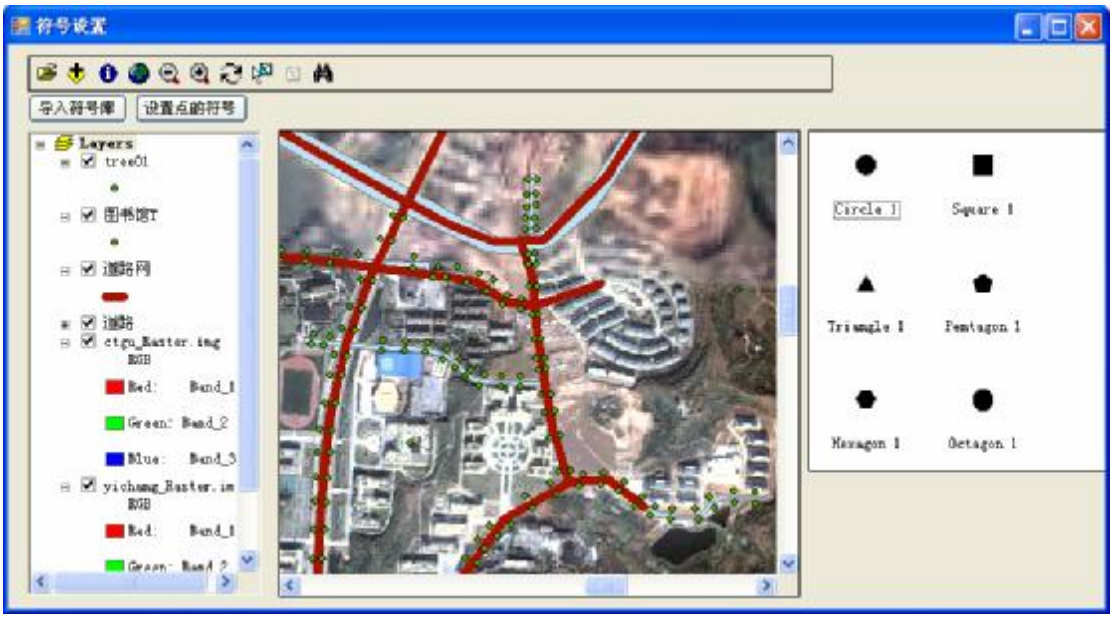
```
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim featureLayer As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(0)
        Dim geoFeatureLayer As ESRI.ArcGIS.Carto.IGeoFeatureLayer =
featureLayer
        Dim simpleRenderer As ESRI.ArcGIS.Carto.ISimpleRenderer =
geoFeatureLayer.Renderer

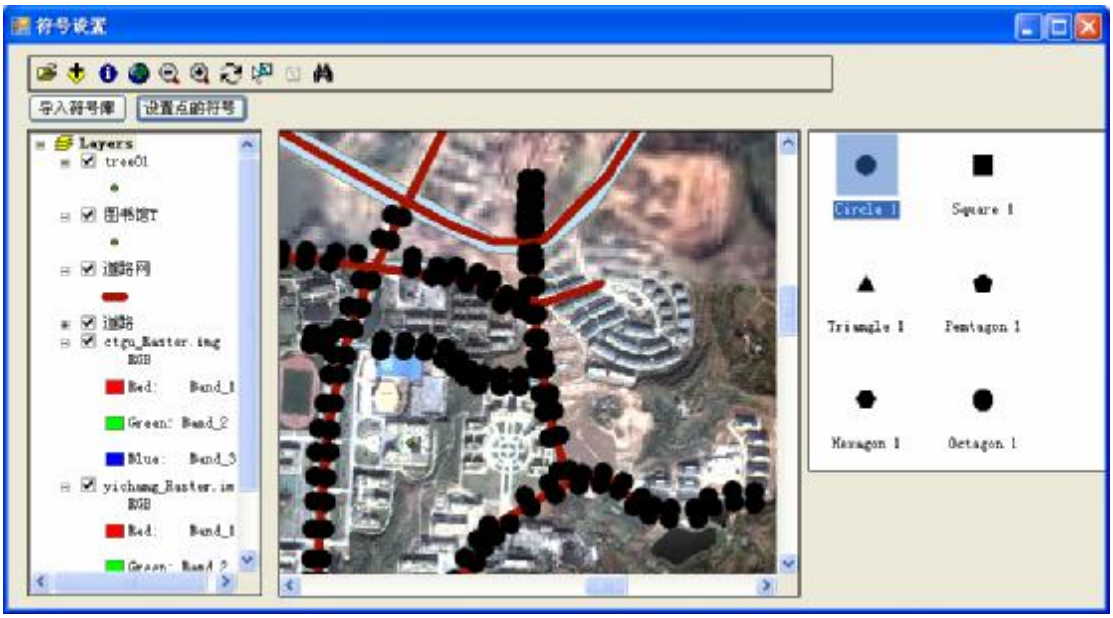
        Dim pStyleGalleryItem As ESRI.ArcGIS.Display.IStyleGalleryItem
        Dim symbologyStyleClass As
ESRI.ArcGIS.Controls.ISymbologyStyleClass =
AxSymbologyControl1.GetStyleClass(ESRI.ArcGIS.Controls.esriSymbologyS
tyleClass.esriStyleClassMarkerSymbols)

        pStyleGalleryItem = symbologyStyleClass.GetSelectedItem
        '创建一个渲染
        simpleRenderer = New ESRI.ArcGIS.Carto.SimpleRendererClass
        '设置它的符号从styleGalleryItem来
        simpleRenderer.Symbol = pStyleGalleryItem.Item
        '吧渲染应用到geoFeatureLayer图层上
        geoFeatureLayer.Renderer = simpleRenderer
        AxMapControl1.Refresh()
       End Sub
```

其中，定义 geoFeatureLayer 图层是为了获取到 Renderer 对象。AxSymbologyControl1.GetStyleClass（）是获取 AxSymbologyControl1 中的点符号。

3）运行程序，其显示结果如下图所示：





21、更改线的显示符号

其设计思路同前面一样，接着上面的例子继续。

1）首先，在“导入符号库”按钮的 Click 事件中添加如下程序代码：

```
AxSymbologyControl1.StyleClass = ESRI.ArcGIS.Controls.esriSymbologyStyleClass.esriStyleClassLineSymbols
```

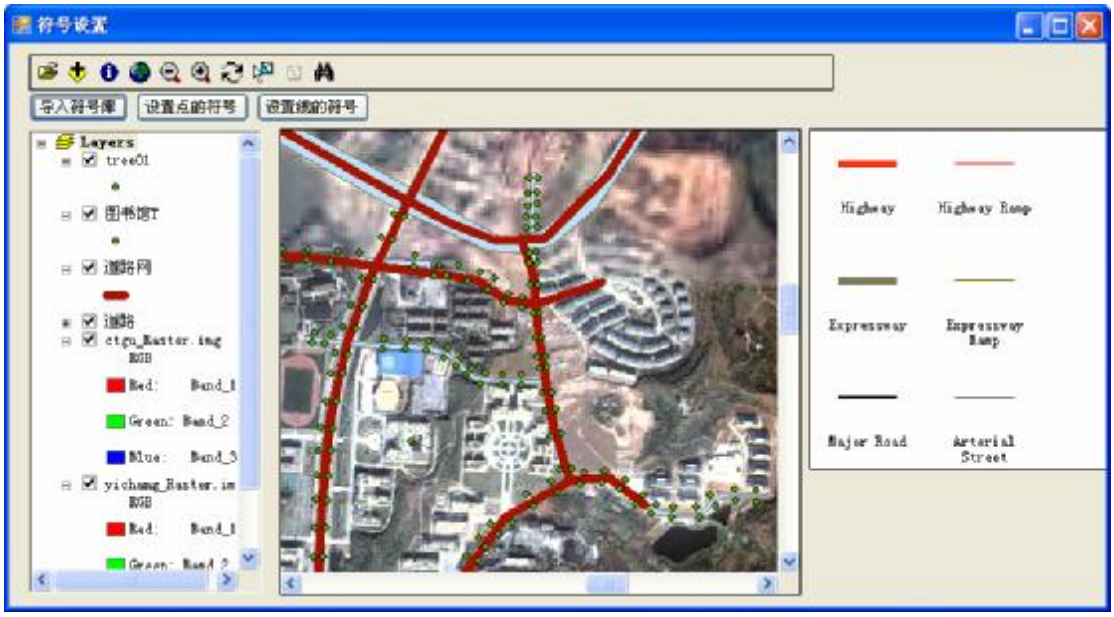
这主要用于显示线型符号。一个 ServerStyle 文件中包含很多种类型数据的显示符号，通过 SymbologyControl 的 StyleClass 属性可以指定只显示某种类型数据的符号。

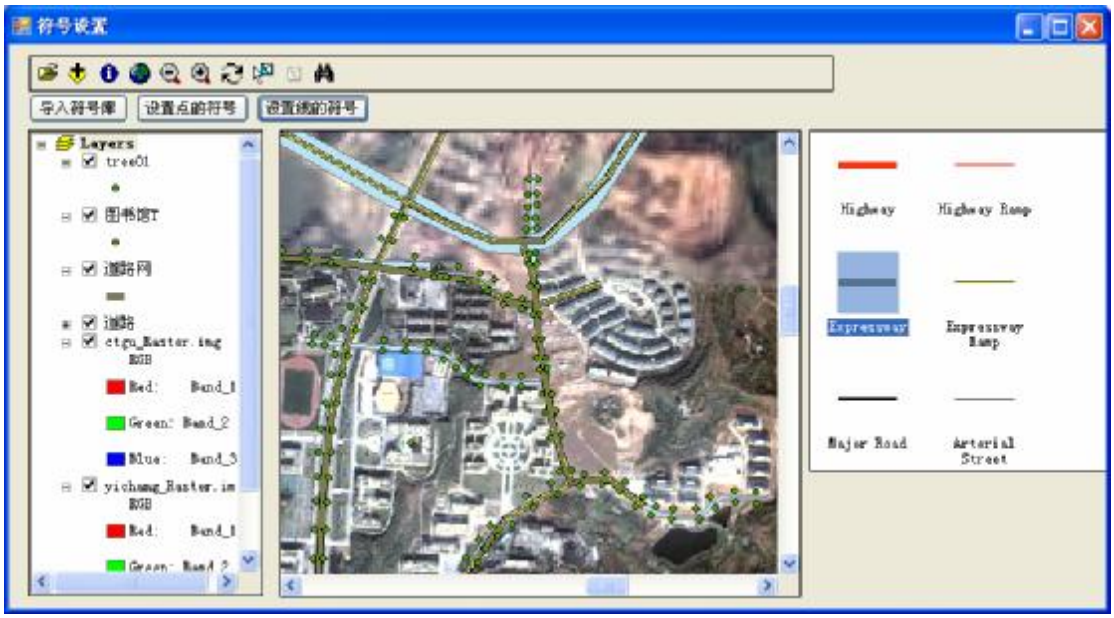
2）添加一个“设置线的符号”的按钮，其 Click 事件的程序代码如下：

```
Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
```

```
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(2)
        Dim pGeoFeatureLyr As ESRI.ArcGIS.Carto.IGeoFeatureLayer =
pFeatureLyr
        Dim pRender As ESRI.ArcGIS.Carto.ISimpleRenderer =
pGeoFeatureLyr.Renderer
        Dim pStyleGalleryItem As ESRI.ArcGIS.Display.IStyleGalleryItem
        Dim pSymbologyStyleClass As
ESRI.ArcGIS.Controls.ISymbologyStyleClass =
AxSymbologyControl1.GetStyleClass(ESRI.ArcGIS.Controls.esriSymbologyS
tyleClass.esriStyleClassLineSymbols)
        pStyleGalleryItem = pSymbologyStyleClass.GetSelectedItem()
        pRender.Symbol = pStyleGalleryItem.Item
        pGeoFeatureLyr.Renderer = pRender
        AxMapControl1.Refresh()
      End Sub
```

3）运行程序，其显示结果如下图所示：





22、自定义点的符号

在很多时候需要自定义点的显示符号，这些显示符号主要是三维模型，其中又以 3ds 格式的模型文件居多。下面就以导入外部 3ds 格式模型文件到 Map 为例。

1）添加一个“自定义点的符号”的按钮，其 Click 事件的程序代码如下：

```
 Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
        Dim pOpenDg As OpenFileDialog = New OpenFileDialog()
        pOpenDg.Title = "打开模型文件"
        pOpenDg.Filter = "模型文件(*.3ds)|*.3ds"
        pOpenDg.ShowDialog()
        Dim marker3DSymbolfile As String
        marker3DSymbolfile = pOpenDg.FileName
        Dim p As ESRI.ArcGIS.Analyst3D.IMarker3DSymbol = New
ESRI.ArcGIS.Analyst3D.Marker3DSymbol()
        p.CreateFromFile(marker3DSymbolfile)

        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxMapControl1.Map.Layer(0)
        Dim pGeoFeatureLyr As ESRI.ArcGIS.Carto.IGeoFeatureLayer =
pFeatureLyr
        Dim pRender As ESRI.ArcGIS.Carto.ISimpleRenderer =
pGeoFeatureLyr.Renderer

        pRender.Symbol = p
        pGeoFeatureLyr.Renderer = pRender
        AxMapControl1.Refresh()
       End Sub
```

其主要思路就是通过 IMarker3DSymbol 接口获取外部的模型文件，然后通过 IGeoFeatureLayer 接口设置图层的 Renderer，通过 ISimpleRenderer 接口获得 Symbol 为前面定义了已经导入模型的 IMarker3DSymbol，设置 IGeoFeatureLayer 的 Renderer。

2）运行程序，其显示结果如下图所示：

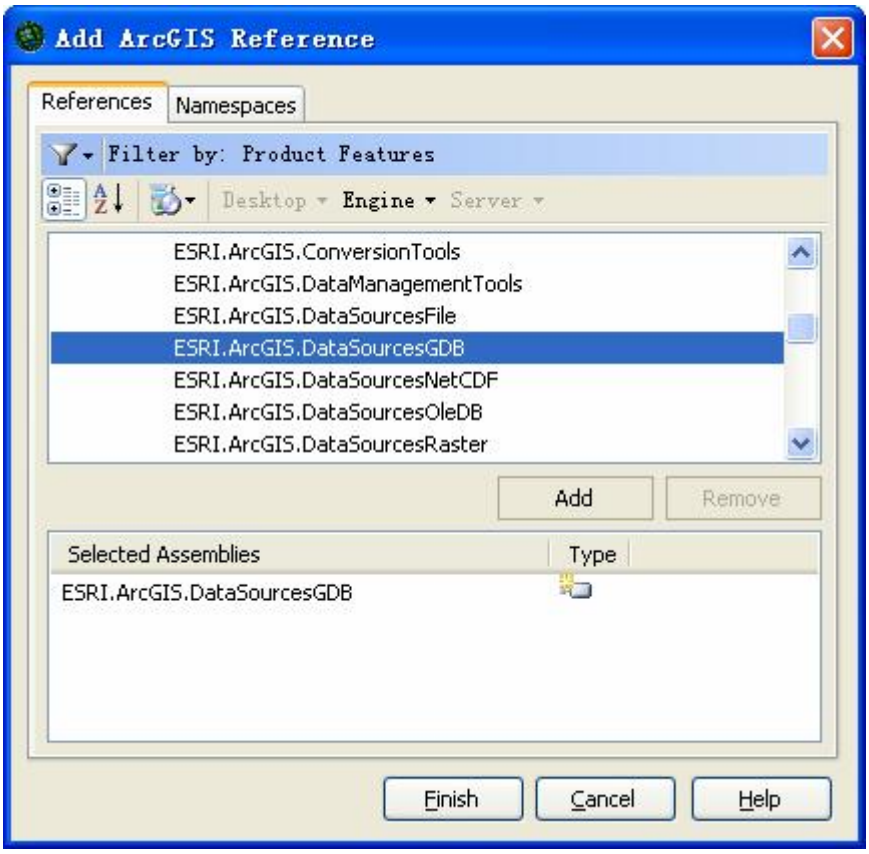




23、打开个人数据库

打开个人数据库主要通过 AccessWorkspaceFactory 类通过 OpenFromFile 方法获得 IFeatureWorkspace。

1）首先，添加引用 ESRI.ArcGIS.DataSourcesGDB 库，如下图所示：

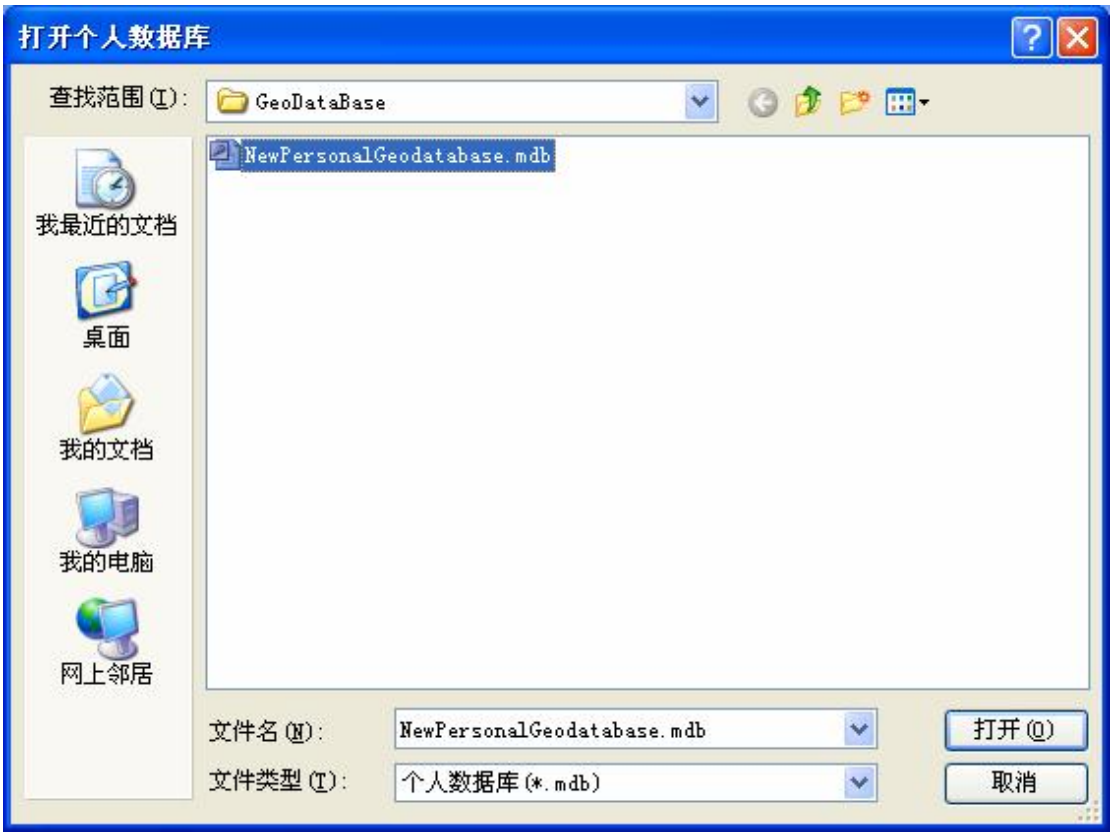


2）添加一个“打开个人数据库”按钮，其 Click 事件的程序代码如下所示：

```
Private Sub Button44_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button44.Click
        Dim pOpenAccDG As OpenFileDialog = New OpenFileDialog()
        pOpenAccDG.Title = "打开个人数据库"
        pOpenAccDG.Filter = "个人数据库(*.mdb)|*.mdb"
        pOpenAccDG.ShowDialog()
        If pOpenAccDG.FileName = "" Then Exit Sub
        Dim pAccessWF As New
ESRI.ArcGIS.DataSourcesGDB.AccessWorkspaceFactory
        Dim pWorkspace As ESRI.ArcGIS.Geodatabase.IFeatureWorkspace
        pWorkspace = pAccessWF.OpenFromFile(pOpenAccDG.FileName, 0)
        Dim pFeartureLyr As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.FeatureLayer()
        pFeartureLyr.FeatureClass =
pWorkspace.OpenFeatureClass("Points01")
        pFeartureLyr.Name = "Points01"
        AxMapControl1.Map.AddLayer(pFeartureLyr)
       End Sub
```

其中 pFeartureLyr.FeatureClass = pWorkspace.OpenFeatureClass("Points01") 是获取到的数据库中名称为 Points01 的 FeatureClass。

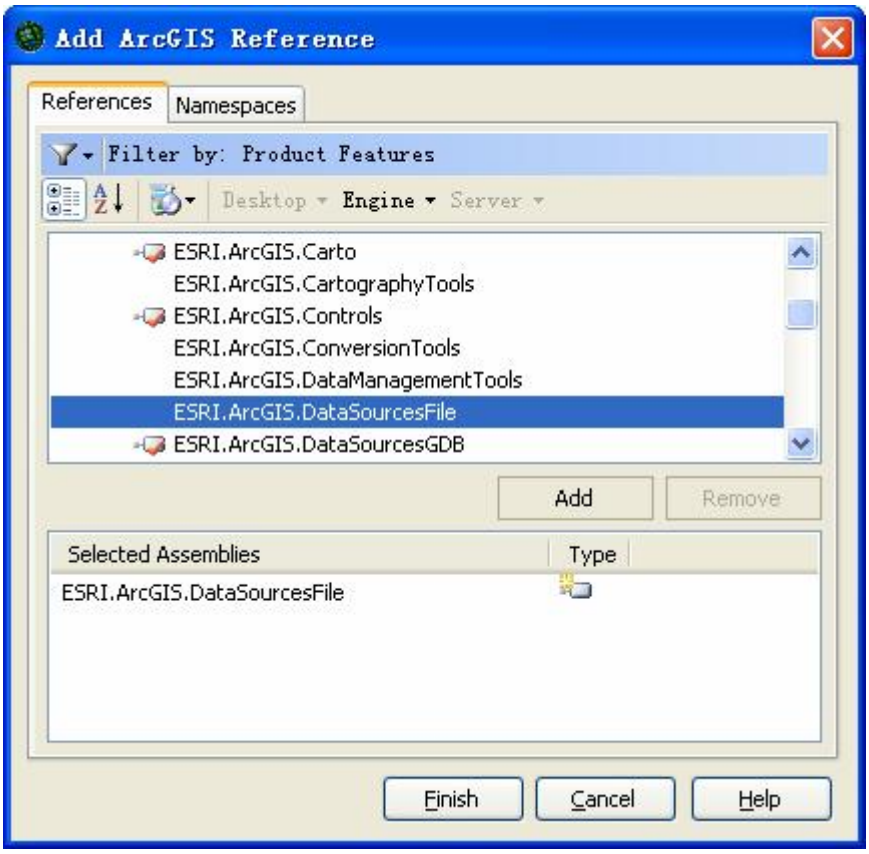
3）运行程序，其结果如下图所示：



24、打开 CAD 数据

打开 CAD 数据类似于前面的打开个人数据库，也是通过 CadWorkspaceFactory 类来创建一个 IFeatureWorkspace。

1）添加引用 ESRI.ArcGIS.DataSourcesFile 库，如下图所示：



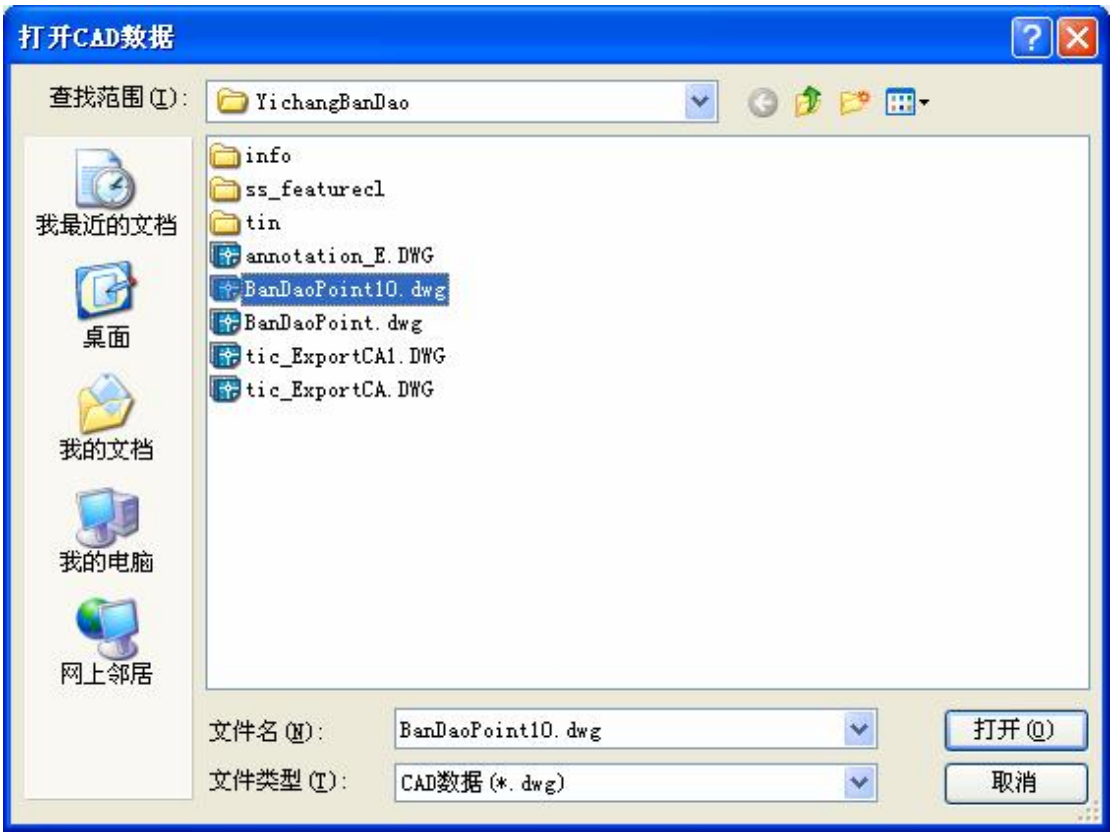
2）添加一个“打开 CAD 数据”按钮，其 Click 事件的程序代码如下所示：

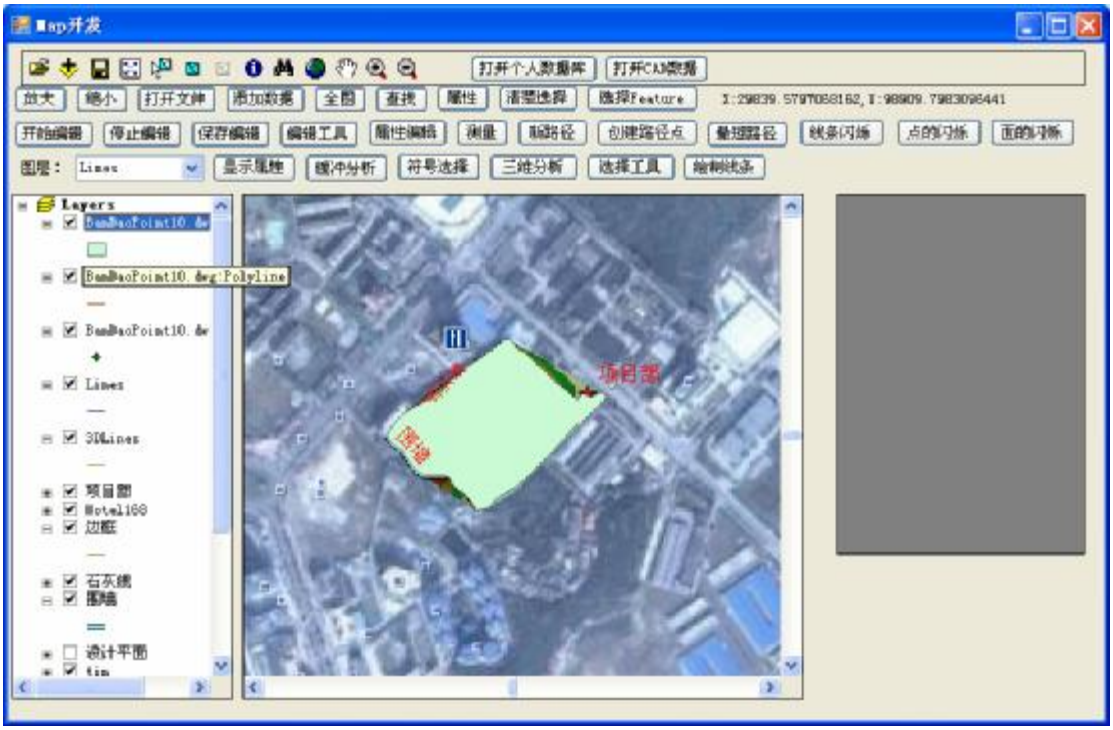
```
    Private Sub Button45_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button45.Click
        On Error GoTo handle01
        Dim pOpenCADDG As OpenFileDialog = New OpenFileDialog()
        pOpenCADDG.Title = "打开CAD数据"
        pOpenCADDG.Filter = "CAD数据(*.dwg)|*.dwg"
        pOpenCADDG.ShowDialog()
        Dim CADFilePath As String = pOpenCADDG.FileName
        If CADFilePath = "" Then Exit Sub
        CADFilePath = Microsoft.VisualBasic.Left(pOpenCADDG.FileName,
pOpenCADDG.FileName.LastIndexOf("\"))
        Dim CADFileName As String =
Microsoft.VisualBasic.Right(pOpenCADDG.FileName,
pOpenCADDG.FileName.Length - 1 - pOpenCADDG.FileName.LastIndexOf("\"))

        Dim pCADWF As New
ESRI.ArcGIS.DataSourcesFile.CadWorkspaceFactory
        Dim pWorkspace As ESRI.ArcGIS.Geodatabase.IFeatureWorkspace
        pWorkspace = pCADWF.OpenFromFile(CADFilePath, 0)
        '添加CAD中的点数据
        Dim pPointsLayer As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.CadFeatureLayer()
```

```
        Dim pFeatPoints As ESRI.ArcGIS.Geodatabase.IFeatureClass =
pWorkspace.OpenFeatureClass(CADFileName + ":Point")
        pPointsLayer.FeatureClass = pFeatPoints
        pPointsLayer.Name = CADFileName + ":Point"
        AxMapControl1.Map.AddLayer(pPointsLayer)
        '添加CAD中的线数据
        Dim pFeatPolyline As ESRI.ArcGIS.Geodatabase.IFeatureClass =
pWorkspace.OpenFeatureClass(CADFileName + ":Polyline")
        Dim pPolylineLayer As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.CadFeatureLayer()
        pPolylineLayer.FeatureClass = pFeatPolyline
        pPolylineLayer.Name = CADFileName + ":Polyline"
        AxMapControl1.Map.AddLayer(pPolylineLayer)
        '添加CAD中的面数据
        Dim pFeatPolygon As ESRI.ArcGIS.Geodatabase.IFeatureClass =
pWorkspace.OpenFeatureClass(CADFileName + ":Polygon")
        Dim pPolygonLayer As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.CadFeatureLayer()
        pPolygonLayer.FeatureClass = pFeatPolygon
        pPolygonLayer.Name = CADFileName + ":Polygon"
        AxMapControl1.Map.AddLayer(pPolygonLayer)
        Exit Sub
handle01:
        MsgBox(Err.Description)
       End Sub
```

3）运行程序，其结果如下图所示：



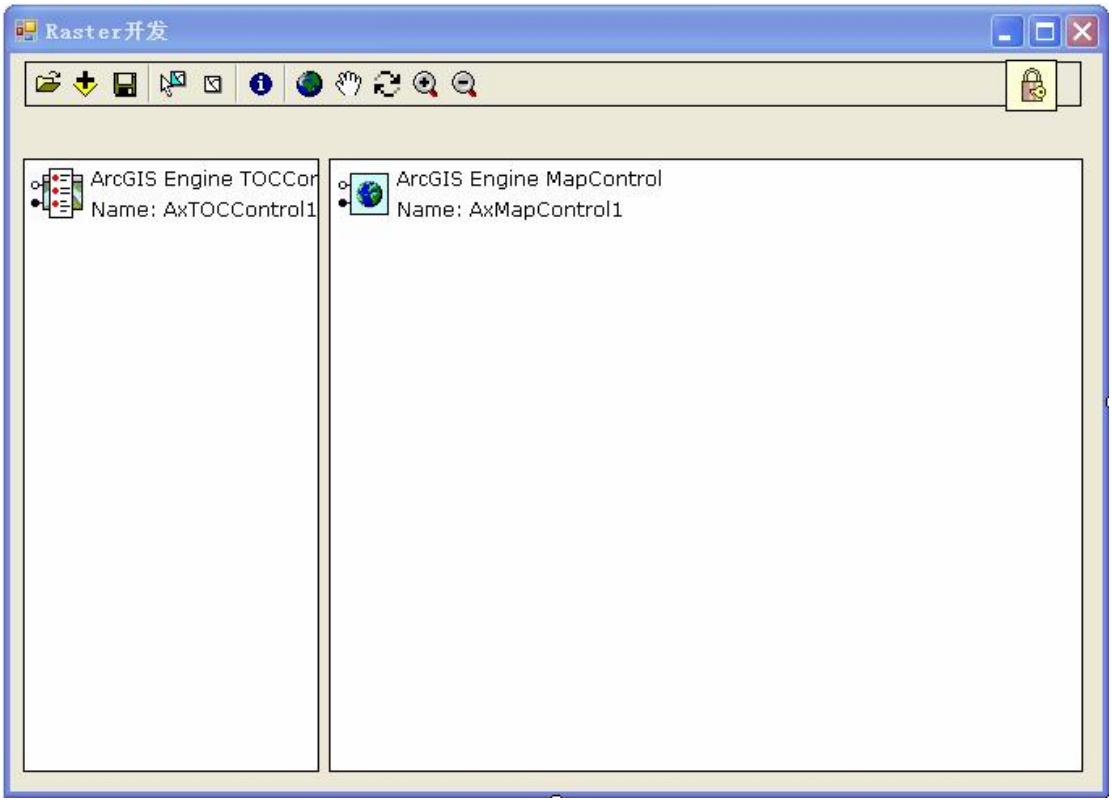


25、计算 Raster 的绝对值

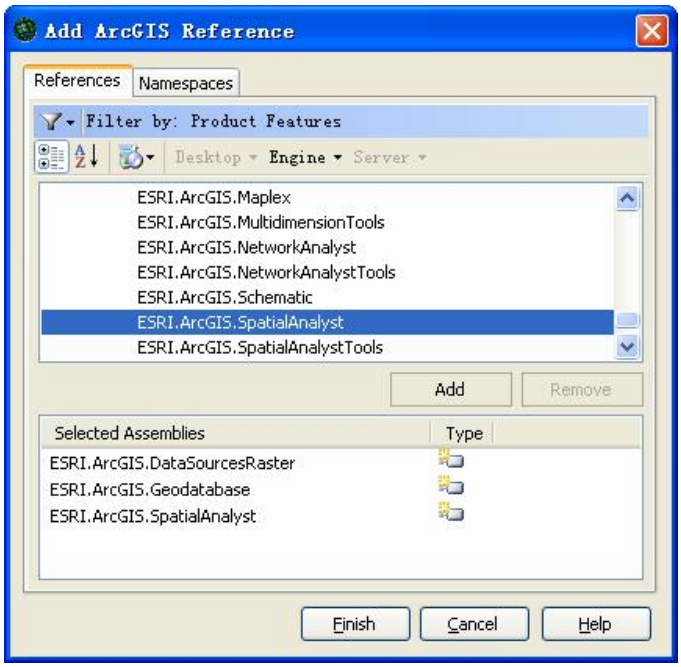
Raster 的计算主要是通过 IMathOp 接口、ILogicalOp 接口和 ITrigOp 接口完成的，它们分别用于 Raster 的数学运算、逻辑运算以及三角函数运算。

1）首先，新建一个项目，名称为“Raster 开发”，并添加如下图所示的控

件：



2）添加引用 ESRI.ArcGIS.DataSourcesRaster、ESRI.ArcGIS.Geodatabase、ESRI.ArcGIS.SpatialAnalyst 和 ESRI.ArcGIS.GeoAnalyst，如下图所示：



3）添加一个“计算绝对值”按钮，其 Click 事件的代码如下：

```
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim pRasterLayer As ESRI.ArcGIS.Carto.IRasterLayer = New
```

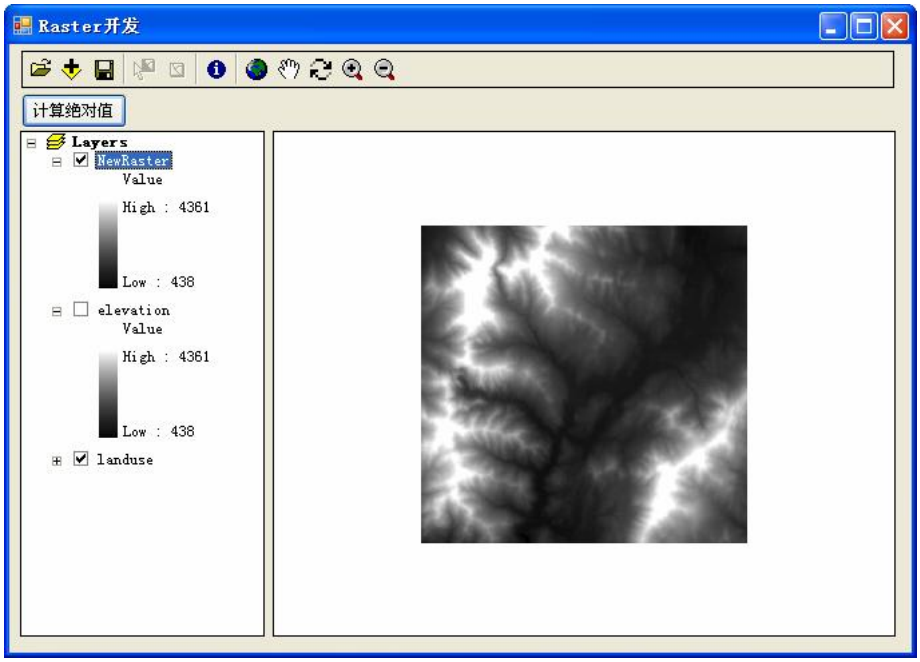
```
ESRI.ArcGIS.Carto.RasterLayer()
        pRasterLayer = AxMapControl1.Map.Layer(0)
        Dim pIRaster As ESRI.ArcGIS.Geodatabase.IRaster =
pRasterLayer.Raster
        Dim pGeoIGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset
        pGeoIGeoDataset = pIRaster

        Dim pp As ESRI.ArcGIS.SpatialAnalyst.IMathOp = New
ESRI.ArcGIS.SpatialAnalyst.RasterMathOpsClass()
        Dim p2 As ESRI.ArcGIS.Geodatabase.IGeoDataset
        p2 = pp.Abs(pGeoIGeoDataset)

        Dim pRasterLayer2 As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        Dim praster As ESRI.ArcGIS.Geodatabase.IRaster = New
ESRI.ArcGIS.DataSourcesRaster.Raster()
        praster = p2

        pRasterLayer2.CreateFromRaster(praster)
        pRasterLayer2.Name = "NewRaster"
        AxMapControl1.Map.AddLayer(pRasterLayer2)
         End Sub
```

4）运行程序，其效果如下图所示：

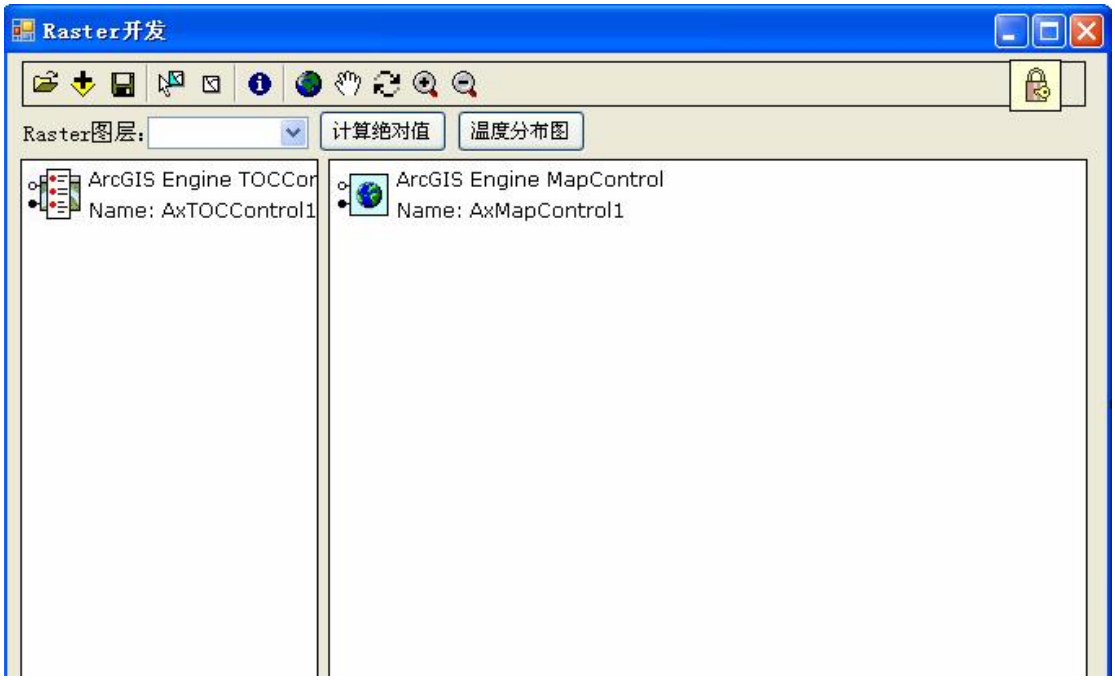


26、Raster 的复杂计算

通过通过 IMathOp 接口、ILogicalOp 接口和 ITrigOp 接口可以实现 Raster 数据的复杂计算，只要数学模型确定即可通过这些接口的组合来实现。如现在要

获取整个地区的温度，假定：温度=39-高度的平方*0.000002

1）添加一个 ComboBox 控件和一个“温度分布图”按钮，如下图所示：



2）定义函数 RefreshRasterLayer()用于刷新 Raster 图层，其代码如下所示：

```
    Private Sub RefreshRasterLayer()
        ComboBox1.Items.Clear()
        Dim i As Integer
        For i = 0 To AxMapControl1.Map.LayerCount - 1
            If TypeOf AxMapControl1.Map.Layer(i) Is
ESRI.ArcGIS.Carto.RasterLayer Then
                ComboBox1.Items.Add(AxMapControl1.Map.Layer(i).Name)
            End If
        Next
        ComboBox1.Text = ComboBox1.Items(0)
         End Sub
```

3）在 Form 的 Load()事件中，Button1 的 Click 事件以及 Button2 的 Click 事件中添加引用函数 RefreshRasterLayer()。

4）为“温度分布图”按钮添加 Click 事件，其代码如下：

```
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim pRasterLayer As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        pRasterLayer =
AxMapControl1.Map.Layer(ComboBox1.SelectedIndex)
        Dim pIRaster As ESRI.ArcGIS.Geodatabase.IRaster =
pRasterLayer.Raster
        Dim pGeoIGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset
```

```
        pGeoIGeoDataset = pIRaster
        Dim p1 As ESRI.ArcGIS.Geodatabase.IGeoDataset

        Dim pp As ESRI.ArcGIS.SpatialAnalyst.IMathOp = New
ESRI.ArcGIS.SpatialAnalyst.RasterMathOpsClass()
        Dim pp2 As ESRI.ArcGIS.SpatialAnalyst.ILogicalOp = New
ESRI.ArcGIS.SpatialAnalyst.RasterMathOpsClass()
        Dim p2 As ESRI.ArcGIS.Geodatabase.IGeoDataset
        p1 = pp.Power(pp.Exp2(pp2.EqualTo(pGeoIGeoDataset,
pGeoIGeoDataset)), 5.2854)
        p2 = pp.Power(pp.Exp2(pp2.EqualTo(pGeoIGeoDataset,
pGeoIGeoDataset)), -19)
        p2 = pp.Times(p2, pp.Square(pGeoIGeoDataset))
        p1 = pp.Minus(p1, p2)

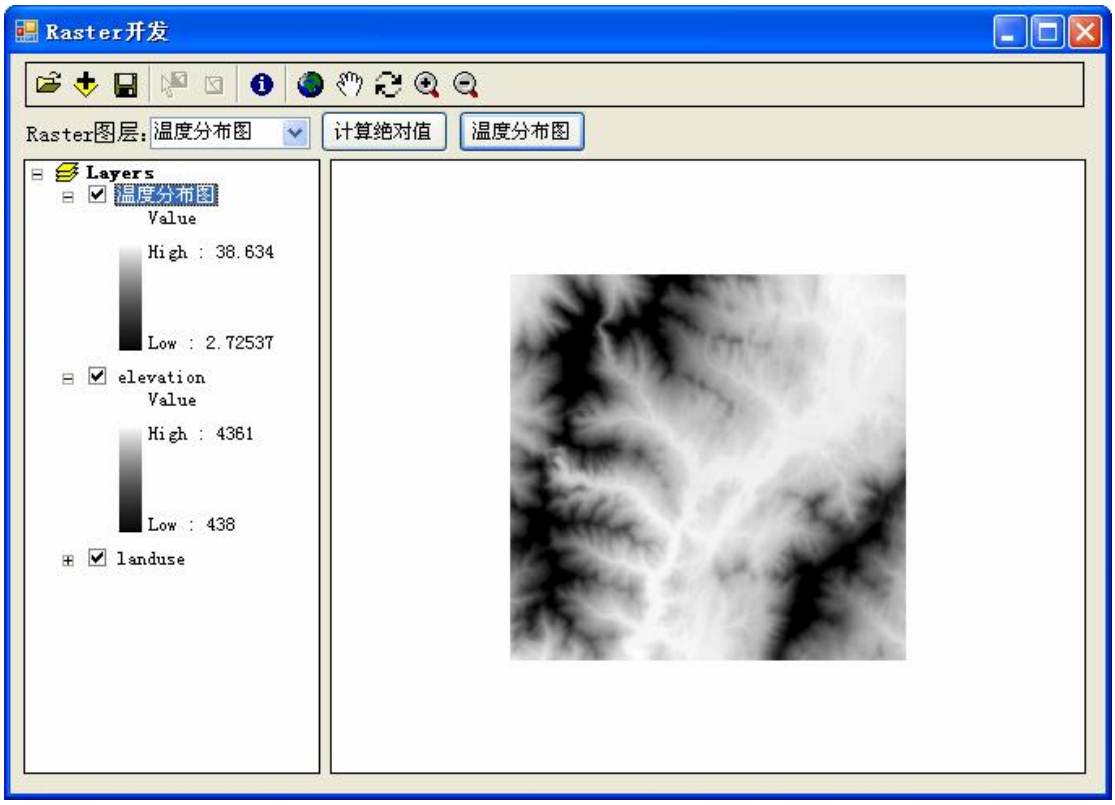
        Dim pRasterLayer2 As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        Dim praster As ESRI.ArcGIS.Geodatabase.IRaster = New
ESRI.ArcGIS.DataSourcesRaster.Raster()
        praster = p1

        pRasterLayer2.CreateFromRaster(praster)
        pRasterLayer2.Name = "温度分布图"

        AxMapControl1.Map.AddLayer(pRasterLayer2)
        Call RefreshRasterLayer()
         End Sub
```

由于 IMathOp 中没有 Raster 与数值相乘的方法，所以在此使用 pp.Power(pp.Exp2(pp2.EqualTo(pGeoIGeoDataset, pGeoIGeoDataset)), -19) 获得约为 0.000002 的 Raster。pp.Power(pp.Exp2(pp2.EqualTo(pGeoIGeoDataset, pGeoIGeoDataset)), 5.2854) 是获取的是值为 39 的 Raster。其中，pp2.EqualTo(pGeoIGeoDataset, pGeoIGeoDataset)获取到的就是一个值为 1 的 Raster。

5）运行程序，其结果如下图所示：



27、生成表面 Raster

无论是 TIN 模型还是 Raster 模型，都可以通过 ISurfaceOp 接口来实现表面 Raster、坡度 Raster、等高线、可视区域等。

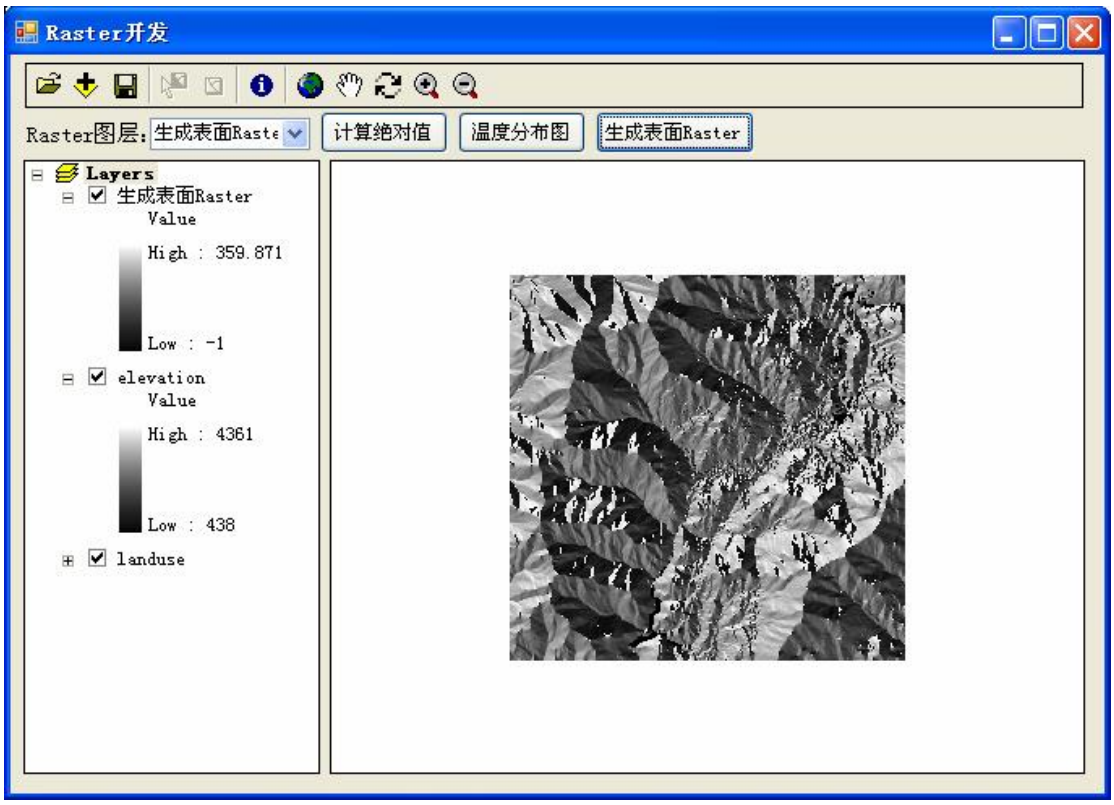
1）首先添加一个“生成表面 Raster”按钮，其 Click 事件的代码如下：

```
    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        Dim pRasterLayer As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        pRasterLayer =
AxMapControl1.Map.Layer(ComboBox1.SelectedIndex)
        Dim pIRaster As ESRI.ArcGIS.Geodatabase.IRaster =
pRasterLayer.Raster
        Dim pSurfaceOp As ESRI.ArcGIS.GeoAnalyst.ISurfaceOp = New
ESRI.ArcGIS.GeoAnalyst.RasterSurfaceOp()
        Dim pGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset
        pGeoDataset = pSurfaceOp.Aspect(pIRaster)
        Dim pRasterLayer1 As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        Dim praster As ESRI.ArcGIS.Geodatabase.IRaster = New
ESRI.ArcGIS.DataSourcesRaster.Raster()
        praster = pGeoDataset
        pRasterLayer1.CreateFromRaster(praster)
```

```
        pRasterLayer1.Name = "生成表面Raster"

        AxMapControl1.Map.AddLayer(pRasterLayer1)
        Call RefreshRasterLayer()
         End Sub
```

2）运行程序，其结果如下图所示：



28、生成坡度 Raster

1）首先添加一个“生成坡度 Raster”按钮，其 Click 事件的代码如下：

```
    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
        Dim pRasterLayer As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        pRasterLayer =
AxMapControl1.Map.Layer(ComboBox1.SelectedIndex)
        Dim pIRaster As ESRI.ArcGIS.Geodatabase.IRaster =
pRasterLayer.Raster
        Dim pSurfaceOp As ESRI.ArcGIS.GeoAnalyst.ISurfaceOp = New
ESRI.ArcGIS.GeoAnalyst.RasterSurfaceOp()
        Dim pGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset
        pGeoDataset = pSurfaceOp.Slope(pIRaster,
ESRI.ArcGIS.GeoAnalyst.esriGeoAnalysisSlopeEnum.esriGeoAnalysisSlopeD
egrees)
```

```
        Dim pRasterLayer1 As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        Dim praster As ESRI.ArcGIS.Geodatabase.IRaster = New
ESRI.ArcGIS.DataSourcesRaster.Raster()
        praster = pGeoDataset
        pRasterLayer1.CreateFromRaster(praster)
        pRasterLayer1.Name = "生成坡度Raster"

        AxMapControl1.Map.AddLayer(pRasterLayer1)
        Call RefreshRasterLayer()
         End Sub
```

2）运行程序，其结果如下图所示：



29、生成等高线

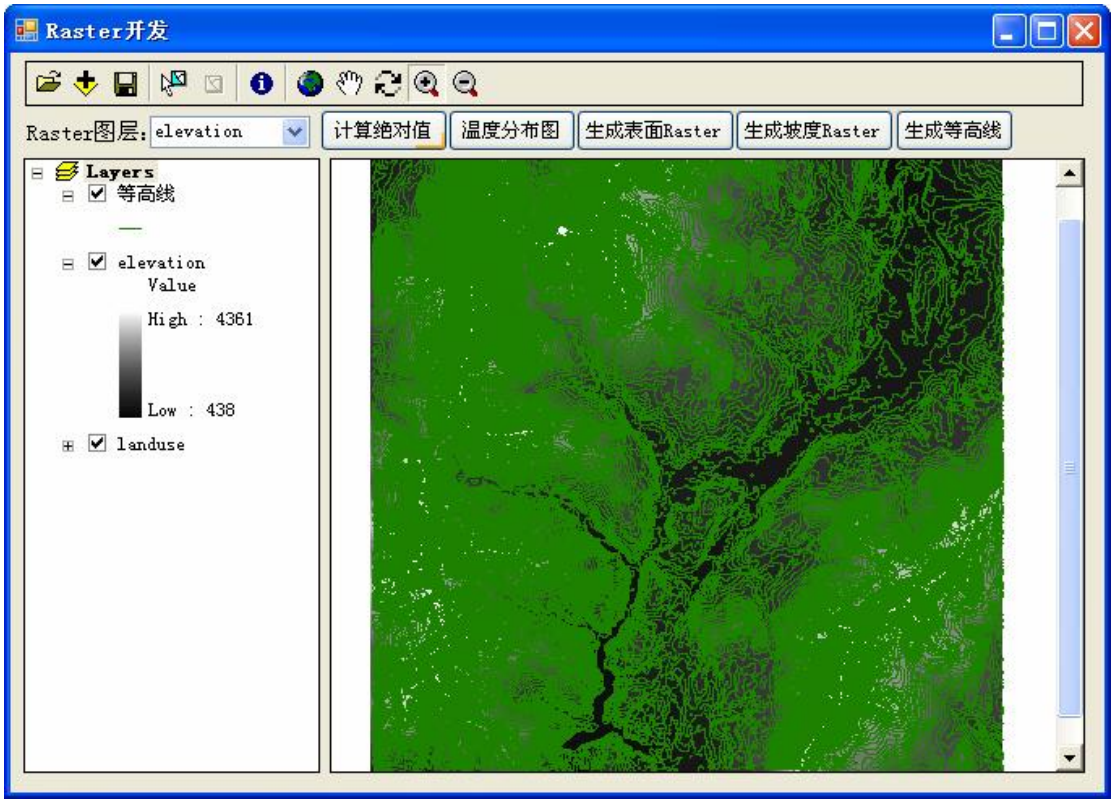
1）首先添加一个“生成等高线”按钮，其 Click 事件的代码如下：

```
    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
        Dim pRasterLayer As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        pRasterLayer =
AxMapControl1.Map.Layer(ComboBox1.SelectedIndex)
        Dim pIRaster As ESRI.ArcGIS.Geodatabase.IRaster =
pRasterLayer.Raster
```

```
        Dim pSurfaceOp As ESRI.ArcGIS.GeoAnalyst.ISurfaceOp = New
ESRI.ArcGIS.GeoAnalyst.RasterSurfaceOp()
        Dim pGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset
        pGeoDataset = pSurfaceOp.Contour(pIRaster, 50.0)
        Dim pFeatureLayer As ESRI.ArcGIS.Carto.IFeatureLayer = New
ESRI.ArcGIS.Carto.FeatureLayer()
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass
        pFeatureClass = pGeoDataset
        pFeatureLayer.Name = "等高线"
        pFeatureLayer.FeatureClass = pFeatureClass
        AxMapControl1.Map.AddLayer(pFeatureLayer)
        Call RefreshRasterLayer()
         End Sub
```

其中，pSurfaceOp.Contour(pIRaster, 50.0)是获取等高线数据，50 位等高线的间距。

2）运行程序，其结果如下图所示：



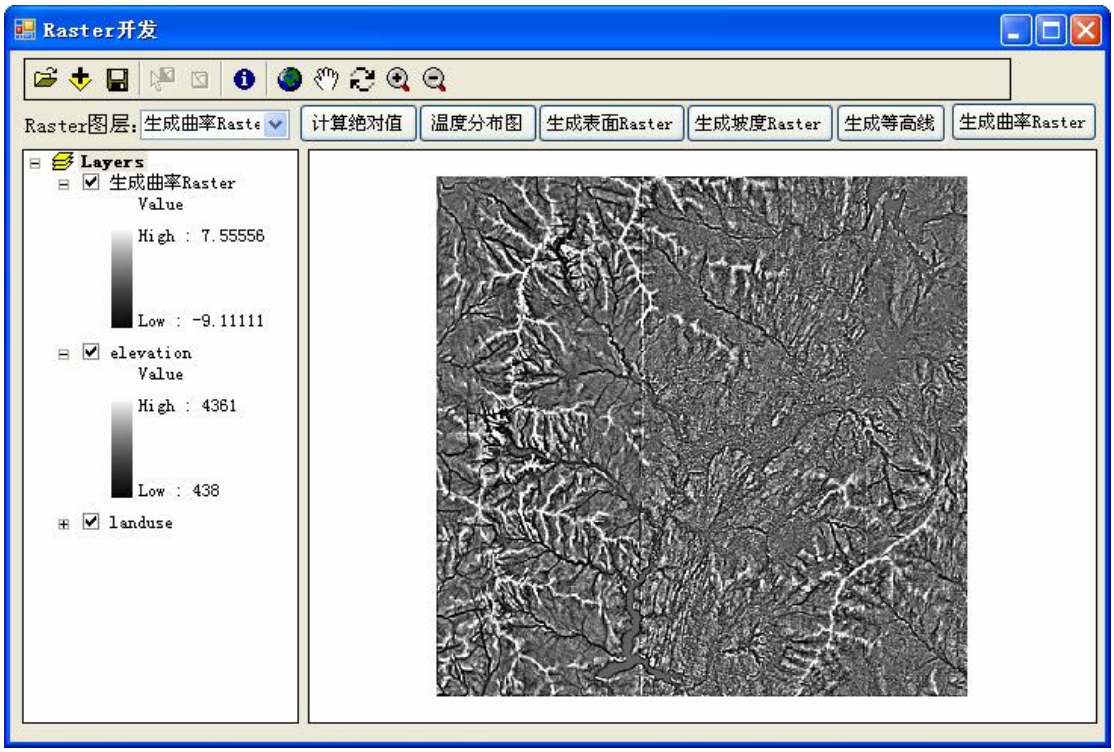
30、生成曲率 Raster

1）首先添加一个“生成曲率 Raster”按钮，其 Click 事件的代码如下：

```
    Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
        Dim pRasterLayer As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
```

```
        pRasterLayer =
AxMapControl1.Map.Layer(ComboBox1.SelectedIndex)
        Dim pIRaster As ESRI.ArcGIS.Geodatabase.IRaster =
pRasterLayer.Raster
        Dim pSurfaceOp As ESRI.ArcGIS.GeoAnalyst.ISurfaceOp = New
ESRI.ArcGIS.GeoAnalyst.RasterSurfaceOp()
        Dim pGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset
        pGeoDataset = pSurfaceOp.Curvature(pIRaster, False, False)
        Dim pRasterLayer1 As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        Dim praster As ESRI.ArcGIS.Geodatabase.IRaster = New
ESRI.ArcGIS.DataSourcesRaster.Raster()
        praster = pGeoDataset
        pRasterLayer1.CreateFromRaster(praster)
        pRasterLayer1.Name = "生成曲率Raster"
        AxMapControl1.Map.AddLayer(pRasterLayer1)
        Call RefreshRasterLayer()
         End Sub
```

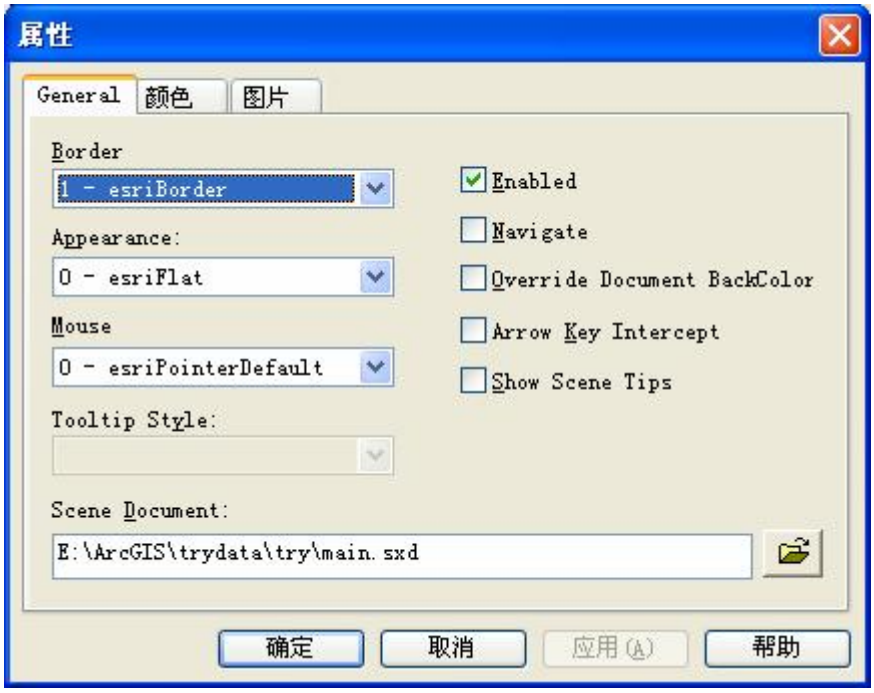
2）运行程序，其结果如下图所示：



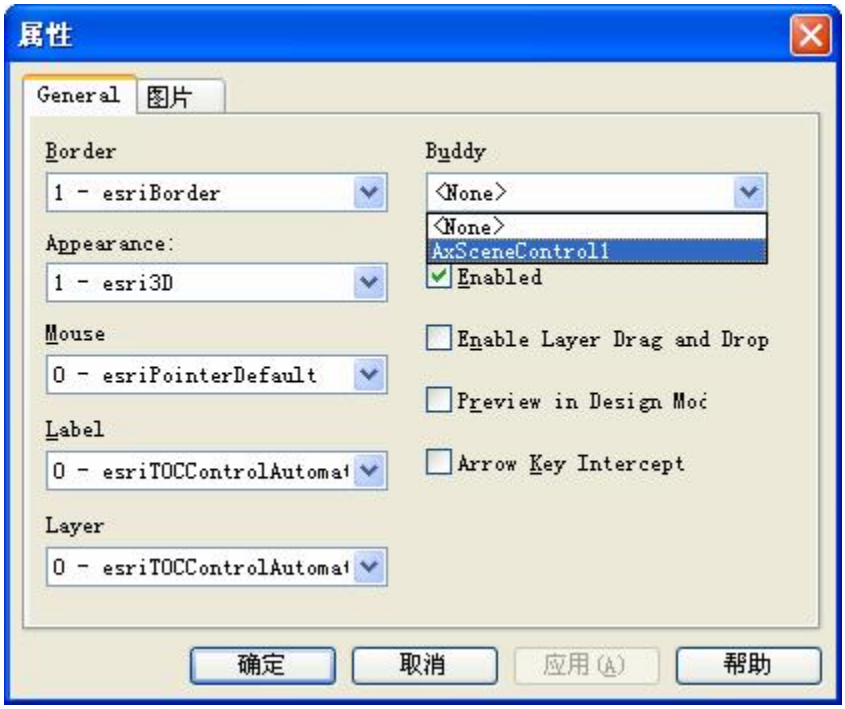
第二篇 Scene 三维开发

1、创建一个简单的三维场景

创建一个三维场景中必须包括 SceneControl 控件，另外根据需要添加 TOCControl 控件和 ToolbarControl 控件。同样，可以设置 SceneControl 控件的属性，并在属性窗体的 Scene Document 下设置 Scene 文件的路径，如下图所示：



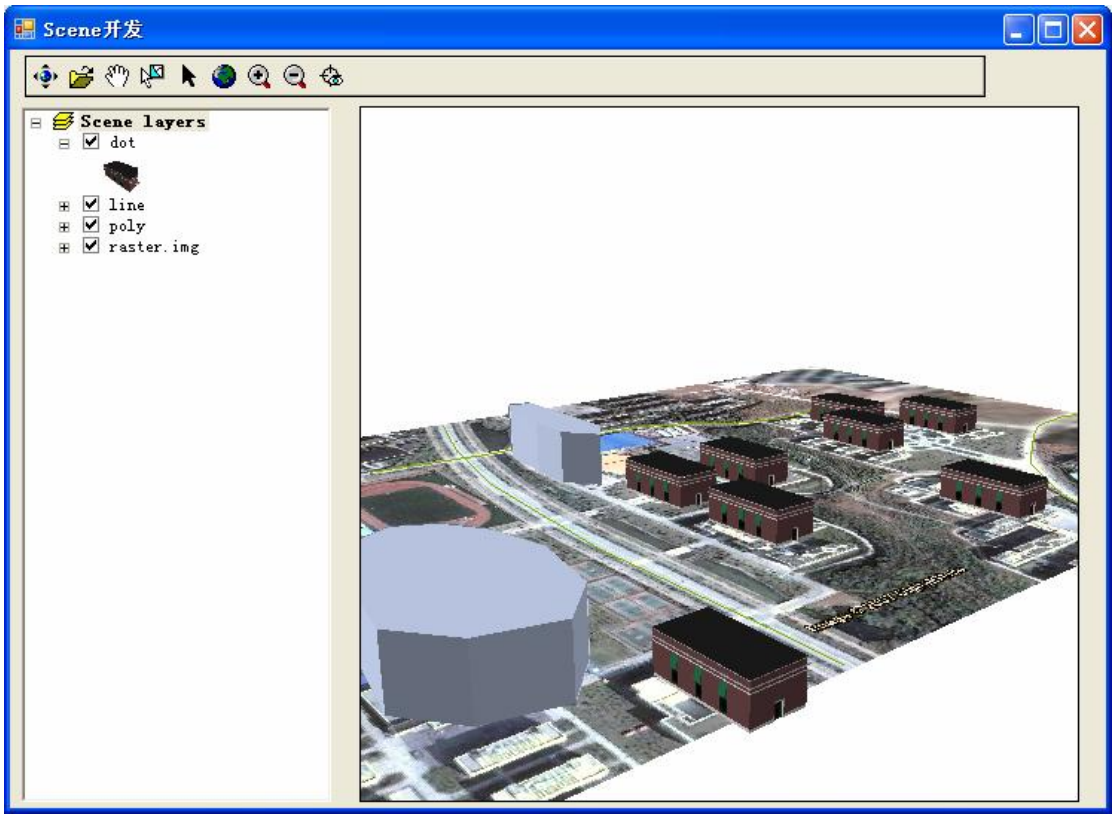
同样在 TOCControl 控件上点击右键选择属性，打开属性窗体，在 Buddy 后设置其所链接的控件，如下图所示。



当然，也可以通过程序来实现链接，如下：

```
AxTOCControl1.SetBuddyControl(AxSceneControl1)
```

运行程序，如下图所示：



2、在 SceneControl 控件中添加 Scene 文件

采用代码：

```
AxSceneControl1.LoadSxFile filestring
```

但是需要注意的是：该 Scene 文件（即 sxd 文件）中不能包含动画 Animation，否则在导入时会出错。

3、旋转/移动摄像机动画

主要用到 Timer 控件和 AxSceneControl 的 Camera 类。在 Timer 的 Tick 函数中添加如下：

```
AxSceneControl1.Camera.Rotate(0.5)
AxSceneControl1.Refresh()
```

注意：动画变化后要每次都刷新，否则看不到明显的动画效果。同样的方法，还可以平移摄像机。

`AxSceneControl1.Camera.ViewFieldAngle` 更改摄像机的视角范围。

`AxSceneControl1.Camera.Zoom` 摄像机缩放，其中大于 1 表示放大可视范围。

`AxSceneControl1.ShowPropertyPages()` 显示控件的属性窗体

4、常用浏览功能

1）漫游

```
AxSceneControl1.Navigate = True
```

2）放大

```
AxSceneControl1.Camera.Zoom(0.9)
AxSceneControl1.Refresh()
```

3）缩小

```
AxSceneControl1.Camera.Zoom(1.1)
AxSceneControl1.Refresh()
```

4）自动旋转

这个操作分两个部分，第一部分就是按钮事件，第二个部分是 Timer 的动画操作。

```
Private Sub CheckBox1_CheckedChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles CheckBox1.CheckedChanged
        If CheckBox1.Checked Then
            Timer1.Enabled = True
        Else
            Timer1.Enabled = False
        End If
    End Sub

    Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer1.Tick
        AxSceneControl1.Camera.Rotate(1.0)
        AxSceneControl1.Refresh()
       End Sub
```

其中 AxSceneControl1.Camera.Rotate（）就是旋转摄像机，1.0 指旋转的角度。

5）上移

```
AxSceneControl1.Camera.Move(ESRI.ArcGIS.Analyst3D.esriCameraMo
vementType.esriCameraMoveUp, 0.1)
        AxSceneControl1.Refresh()
```

6）下移

```
AxSceneControl1.Camera.Move(ESRI.ArcGIS.Analyst3D.esriCameraMo
vementType.esriCameraMoveDown, 0.1)
        AxSceneControl1.Refresh()
```

7）左移

```
AxSceneControl1.Camera.Move(ESRI.ArcGIS.Analyst3D.esriCameraMovementT
ype.esriCameraMoveLeft, 0.1)
        AxSceneControl1.Refresh()
```

8）右移

```
AxSceneControl1.Camera.Move(ESRI.ArcGIS.Analyst3D.esriCameraMovementT
ype.esriCameraMoveRight, 0.1)
        AxSceneControl1.Refresh()
```

9）缩进

```
AxSceneControl1.Camera.Move(ESRI.ArcGIS.Analyst3D.esriCameraMovementT
ype.esriCameraMoveAway, 0.1)
        AxSceneControl1.Refresh()
```

10）缩出

```
AxSceneControl1.Camera.Move(ESRI.ArcGIS.Analyst3D.esriCameraMovementT
ype.esriCameraMoveToward, 0.1)
        AxSceneControl1.Refresh()
```

11）平移

在 Scene 中平移主要是通过移动摄像机来实现。

首先进行全局变量定义：

```
    Public scenePan As Boolean = False
    Public clickSceneTime As Integer = 0
    Public scenePanPoints1 As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
    Public scenePanPoints2 As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
```

在按钮事件中添加代码：

```
    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
        scenePan = True
    End Sub
```

在 SceneControl 控件的 OnMouseDown 事件中添加获取移动的起点和终点以及移动这一事件，代码如下：

```
    Private Sub AxSceneControl1_OnMouseDown(ByVal sender As
System.Object, ByVal e As
ESRI.ArcGIS.Controls.ISceneControlEvents_OnMouseDownEvent) Handles
AxSceneControl1.OnMouseDown
        If scenePan = True Then
            If clickSceneTime = 0 Then
                scenePanPoints1.PutCoords(e.x, e.y)
                clickSceneTime = 1
            ElseIf clickSceneTime = 1 Then
                scenePanPoints2.PutCoords(e.x, e.y)
                AxSceneControl1.Camera.Pan(scenePanPoints2,
scenePanPoints1)
                AxSceneControl1.Refresh()
                clickSceneTime = 0
                scenePan = False
            End If
        End If
    End Sub
```

5、更改摄像机位置

摄像机包括两个重要的部分，即观察处和目标处。通过更改摄像机的观察处和目标处，结合 Timer 时间空间就可以制作动画过程。如下，就是一个简单的设置摄像的观察处为（200，200，0）目标处为（0，0，0）：

```
        Dim p1 As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
        Dim p2 As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
        p1.X = 0 : p1.Y = 0 : p1.Z = 0
        p2.X = 200 : p2.Y = 200 : p2.Z = 0
        AxSceneControl1.Camera.Target = p1
        AxSceneControl1.Camera.Observer = p2
       AxSceneControl1.Scene.SceneGraph.RefreshViewers()
```

下面以移动 X 值为例，说明动画制作的简单过程：

首先，在全局变量中定义：

```
    Public CameraObs As ESRI.ArcGIS.Geometry.IPoint = New
ESRI.ArcGIS.Geometry.Point()
```

然后添加一个 Timer，并在按钮事件中添加如下代码：

```
    Private Sub Button12_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button12.Click
        If Timer2.Enabled = False Then
            Timer2.Enabled = True
            CameraObs = AxSceneControl1.Camera.Observer
        Else
            Timer2.Enabled = False
        End If
    End Sub
```

在 Timer 的 Tick 事件中添加如下代码：

```
    Private Sub Timer2_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer2.Tick
        CameraObs.X = CameraObs.X + 1
        AxSceneControl1.Camera.Observer = CameraObs
        AxSceneControl1.Scene.SceneGraph.RefreshViewers()
    End Sub
```

运行程序，摄像机的观察处 X 值将随着时间变化。根据该原理，还可以设置沿着某条线移动摄像机等等。

6、输出 AVI 动画

输出 AVI 视频动画主要用到 ESRI.ArcGIS.Analyst3D.ISceneExporter3d 接口和 ESRI.ArcGIS.Analyst3D.IAVIExporter，如下所示：

```
       Dim saveVedioFile As SaveFileDialog = New SaveFileDialog()
        saveVedioFile.Filter = "视频文件(*.avi)|*.avi"
        saveVedioFile.Title = "输出AVI文件"
        saveVedioFile.ShowDialog()
```

```
        Dim p3DExporter As ESRI.ArcGIS.Analyst3D.ISceneExporter3d = New
ESRI.ArcGIS.Analyst3D.AVIExporter() ' ISceneExporter3d
        p3DExporter.ExportFileName = saveVedioFile.FileName

        Dim pExporter As ESRI.ArcGIS.Analyst3D.ISceneVideoExporter
        pExporter = p3DExporter

        pExporter.Viewer =
AxSceneControl1.Scene.SceneGraph.ActiveViewer

        pExporter.VideoDuration = 2.0
        pExporter.FrameRate = 15

        Dim pAVIExporter As ESRI.ArcGIS.Analyst3D.IAVIExporter
        pAVIExporter = p3DExporter
        pAVIExporter.Quality = 100

        p3DExporter.ExportScene(AxSceneControl1.Scene)
        msgbox("输出 AVI 视频完成！")
```

7、创建动画关键帧

Scene 开发中允许用户创建动画关键帧，并可以播放关键帧。首先要添加引用 Animation 库。然后添加如下代码：

```
        Dim pScene As ESRI.ArcGIS.Analyst3D.IScene
        pScene = AxSceneControl1.Scene
        Dim tr1 As ESRI.ArcGIS.Analyst3D.IAnimationTrack = New
ESRI.ArcGIS.Analyst3D.AnimationTrack()
        Dim k1 As ESRI.ArcGIS.Analyst3D.IKeyframe = New
ESRI.ArcGIS.Analyst3D.Bookmark3D()
        k1.CaptureProperties(pScene,
pScene.SceneGraph.ActiveViewer.Camera)

        tr1.InsertKeyframe(k1, 0)

        AxSceneControl1.Camera.Zoom(1.1)
        AxSceneControl1.Refresh()
        k1.CaptureProperties(pScene,
pScene.SceneGraph.ActiveViewer.Camera)
        tr1.InsertKeyframe(k1, 1)

        AxSceneControl1.Camera.Zoom(1.1)
        AxSceneControl1.Refresh()
```

```
        k1.CaptureProperties(pScene,
pScene.SceneGraph.ActiveViewer.Camera)
        tr1.InsertKeyframe(k1, 2)
        tr1.Name = "track01"

        Dim trs As ESRI.ArcGIS.Analyst3D.IAnimationTracks
        trs = AxSceneControl1.Scene
        trs.AddTrack(tr1)
```

当然，如果要创建自由的关键帧则要在 SceneControl 的 OnMouseMove 事件下，而且要求 ScenControl 的当前工具是漫游工具。下面就创建关键帧、播放关键帧等进行说明。

首先在全局变量中定义：

```
    Public Playtr1 As ESRI.ArcGIS.Analyst3D.IAnimationTrack = New
ESRI.ArcGIS.Analyst3D.AnimationTrack()
    Public CreateKeyFrameSwitch As Boolean = False
    Public KeyIndex As Integer
```

定义函数 CreateKeyFrame（）用于创建新的关键帧，代码如下：

```
    Private Sub CreateKeyFrame(ByVal pScene As
ESRI.ArcGIS.Analyst3D.IScene, ByVal animationtrack1 As
ESRI.ArcGIS.Analyst3D.IAnimationTrack, ByVal keyIndex As Integer)
        Dim k1 As ESRI.ArcGIS.Analyst3D.IKeyframe = New
ESRI.ArcGIS.Analyst3D.Bookmark3D()
        k1.CaptureProperties(pScene,
pScene.SceneGraph.ActiveViewer.Camera)
        animationtrack1.InsertKeyframe(k1, keyIndex)
        End Sub
```

在 AxSceneControl 控件的 OnMouseDown 事件中添加如下代码：

```
        If CreateKeyFrameSwitch = True Then
            CreateKeyFrame(AxSceneControl1.Scene, Playtr1, KeyIndex)
            KeyIndex = KeyIndex + 1
            End If
```

在创建自由关键帧按钮 Ckick 事件中添加如下代码，运行 AxSceneControl 控件的 OnMouseDown 事件中添加关键帧：

```
    Private Sub Button16_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button16.Click
        CreateKeyFrameSwitch = True
        AxSceneControl1.Navigate = True
        End Sub
```

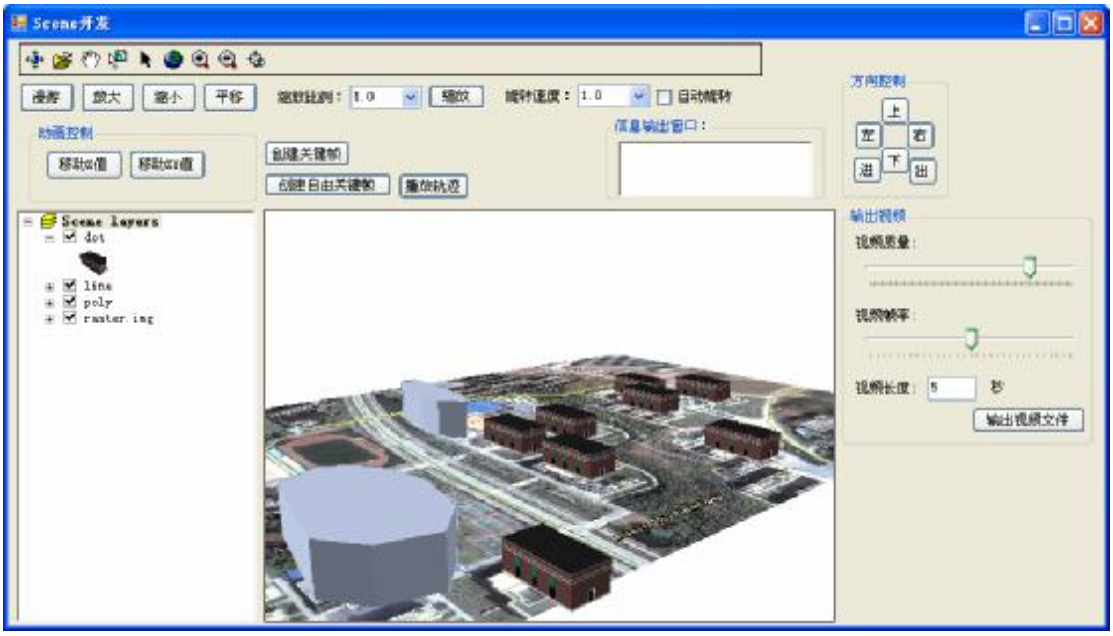
在播放轨迹按钮 Click 事件中添加播放动画轨迹的代码，如下：

```
    Private Sub Button17_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button17.Click
        Dim ptracks As ESRI.ArcGIS.Analyst3D.IAnimationTracks
        ptracks = AxSceneControl1.Scene
        ptracks.AddTrack(Playtr1)
```

```
        Dim i As Integer
        For i = 0 To 300
ptracks.ApplyTracks(AxSceneControl1.Scene.SceneGraph.ActiveViewer, i,
300)
            AxSceneControl1.Scene.SceneGraph.RefreshViewers()
        Next
        End Sub
```

其中 300 为动画播放的持续时间，Playtr1 是前面创建了关键帧的动画轨迹，然后利用 ptracks.AddTrack(Playtr1)将轨迹添加到轨迹集上去。

运行程序，先点击创建自由关键帧按钮，然后在 SceneControl 控件中创建几个关键帧，再点击播放轨迹按钮，即开始播放动画。结合前面的输出 AVI 视频，还可以输出动画视频。如下图所示动画的演示过程：







8、在三维控件 SceneControl 中添加 Shapfile、jpg、tif 等格式文件

其代码如下所示：

```
Public Sub addData()
        Dim pOpenfile As OpenFileDialog = New OpenFileDialog
        pOpenfile.ShowDialog()

        Dim pLayerFactoryHelper As
ESRI.ArcGIS.Carto.ILayerFactoryHelper = New
ESRI.ArcGIS.Carto.LayerFactoryHelper
        Dim filename As ESRI.ArcGIS.esriSystem.IFileName =
New FileNameClass
        filename.Path = pOpenfile.FileName
        Dim enumlayer As ESRI.ArcGIS.Carto.IEnumLayer =
pLayerFactoryHelper.CreateLayersFromName(filename)
        Dim layer As ESRI.ArcGIS.Carto.ILayer
        enumlayer.Reset()
        layer = enumlayer.Next()
        While (layer IsNot Nothing)
            AxSceneControl1.SceneGraph.Scene.AddLayer(layer,
False)
            layer = enumlayer.Next()
            AxSceneControl1.SceneGraph.RefreshViewers()
        End While

    End Sub
```

9、保存场景图片

保存场景图片主要用到 AxSceneControl1.SceneViewer.GetScreenShot（）方法。添加一个按钮控件，在其 Click 事件中添加如下代码：

```
    Private Sub Button18_Click_1(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button18.Click
        On Error GoTo handle01
        Dim sFileName As String
        Dim pSaveFile As SaveFileDialog = New SaveFileDialog()
        pSaveFile.Title = "保存图片"
        pSaveFile.Filter = "Bmp图片(*.bmp)|*.bmp|Jpeg图片
(*.jpg)|*.jpg"

        pSaveFile.ShowDialog()
        sFileName = pSaveFile.FileName
        If pSaveFile.FilterIndex = 1 Then

AxSceneControl1.SceneViewer.GetScreenShot(ESRI.ArcGIS.Analyst3D.esri3
DOutputImageType.BMP, sFileName)
        ElseIf pSaveFile.FilterIndex = 2 Then

AxSceneControl1.SceneViewer.GetScreenShot(ESRI.ArcGIS.Analyst3D.esri3
DOutputImageType.JPEG, sFileName)
        End If
        MsgBox("成功保存图片至：" + sFileName)
        Exit Sub
handle01:
        MsgBox(Err.Description)
       End Sub
```

10、添加 Raster 数据

添加 Raster 数据主要用的 ESRI.ArcGIS.Carto.IRasterLayer 接口，除了可以添加 Img 格式的数据，还可以添加 Bmp 格式、Jpg 格式等。在使用 ESRI.ArcGIS.Carto.RasterLayerClass()类时需要引用，如下图所示：

。添加一个“添加 Raster 数据”按钮，其程序代码如下：

```
    Private Sub Button19_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button19.Click
        On Error GoTo handle01
        Dim pOpenDg As OpenFileDialog = New OpenFileDialog()
        pOpenDg.Title = "添加Raster数据"
        pOpenDg.Filter = "Img格式(*.img)|*.img|Bmp格式
(*.bmp)|*.bmp|Jpeg格式(*.jpg)|*.jpg|TIFF格式(*.tif)|*.tif"
        pOpenDg.ShowDialog()
```

```
        Dim sFilePath As String
        sFilePath = pOpenDg.FileName
        Dim praster As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayerClass()
        praster.CreateFromFilePath(sFilePath)
        AxSceneControl1.Scene.AddLayer(praster, True)
        Exit Sub
handle01:
        MsgBox(Err.Description)
       End Sub
```

11、生成 TIN

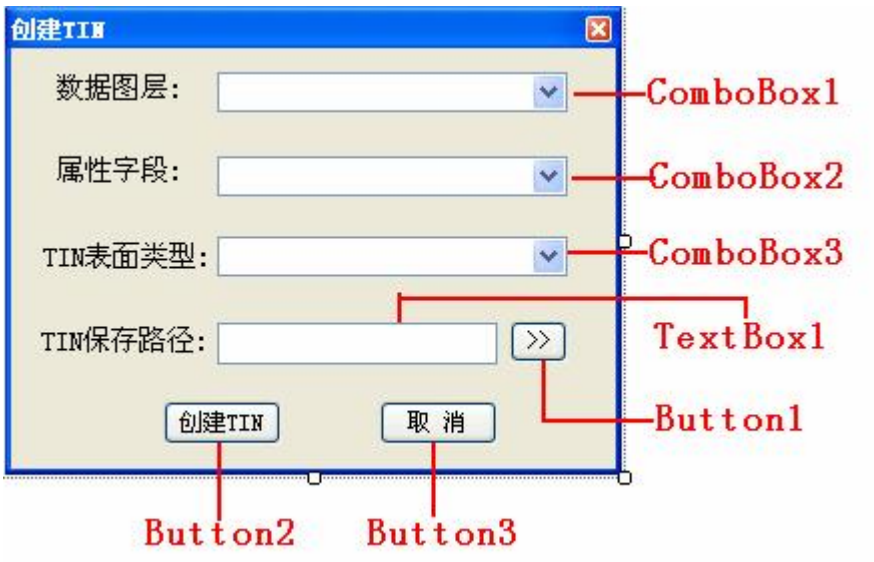
TIN 是 GIS 中常用的一种数字地形模型，它可以通过各种矢量数据生成。生成 TIN 的数据类型主要分为点和线。在 ArcGIS Engine 中主要是通过 ITinEdit 的 AddFromFeatureClass 方法来实现从点或线数据来生成 TIN。

一般来说，生成 TIN 主要分为四个步骤：一、设置 SetSpatialReference；二、初始化 TIN 的范围 InitNew(pEnvelope)；三、添加生成 TIN 的数据 AddFromFeatureClass 方法；四、保存 TIN 数据 SaveAs（）方法。

1）首先，添加一个新的窗体，并命名为“创建 TIN”，如下图所示：



2）在新的创建 TIN 窗体上添加如下图所示控件：



3）在窗体启动时，自动添加图层名称到 ComboBox1 上面。所以，在创建 TIN 窗体的全局变量中添加如下代码：

```
Public AxSceneControl As ESRI.ArcGIS.Controls.AxSceneControl = Form1.AxSceneControl1
```

在创建 TIN 窗体中所有的代码引用 GIS 对象都将直接从 AxSceneControl 变量来引用。然后创建一个函数，用于刷新自动添加图层名称到 ComboBox1 上面，其代码如下所示：

```
Private Sub RefreshLayer()
    Dim i As Integer
    For i = 0 To AxSceneControl.Scene.LayerCount - 1
        ComboBox1.Items.Add(AxSceneControl.Scene.Layer(i).Name)
    Next
    ComboBox1.Text = ComboBox1.Items(0)
End Sub
```

在创建 TIN 的 Load 事件中调用函数 RefreshLayer()，如下：

```
Private Sub 创建TIN_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Call RefreshLayer()
End Sub
```

运行程序，如下图所示：



4）自动添加属性字段。属性字段主要根据图层而定，下面定义一个函数 AddFieldName（），其代码如下：

```
    Private Sub AddFieldName(ByVal LayerName As String)
        ComboBox2.Items.Clear()
        Dim i As Integer
        Dim pLayer As ESRI.ArcGIS.Carto.IFeatureLayer
        Dim pFields As ESRI.ArcGIS.Geodatabase.IFields
        For i = 0 To AxSceneControl.Scene.LayerCount - 1
            If AxSceneControl.Scene.Layer(i).Name = LayerName Then
                On Error Resume Next
                pLayer = AxSceneControl.Scene.Layer(i)
                Exit For
            End If
        Next
        pFields = pLayer.FeatureClass.Fields
        For i = 0 To pFields.FieldCount - 1
            ComboBox2.Items.Add(pFields.Field(i).Name)
        Next
        ComboBox2.Text = ComboBox2.Items(0)
       End Sub
```

在创建 TIN 的 Load 事件中就先调用该函数，然后再在 ComboBox1 的 SelectedIndexChanged 事件中也调用该函数，分别如下：

```
    Private Sub 创建TIN_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Call RefreshLayer()
        AddFieldName(ComboBox1.Text)
       End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox1.SelectedIndexChanged
        AddFieldName(ComboBox1.Text)
```

```
    End Sub
```

运行程序，如下图所示：



5）TIN 表面类型常用的主要有 esriTinMassPoint、esriTinHardLine 和 esriTinSoftLine 这三种，在帮助文档地址 ms-help://ESRI.EDNv9.2/esriGeoDatabase/html/esriTinSurfaceType.htm 中介绍了其他几种表面类型。在此，我们只以 esriTinMassPoint、esriTinHardLine 和 esriTinSoftLine 这三种为例，在 ComboBox3 的 Items 中添加点、直线和光滑线，分别对应于 esriTinMassPoint、esriTinHardLine 和 esriTinSoftLine，如下图所示：



并设置 ComboBox3 的 Text 属性为点。

6）设置 TIN 保存路径。在 Button1 的 Click 事件中添加如下代码：

```
 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim pSavefileDG As SaveFileDialog = New SaveFileDialog()
        pSavefileDG.Title = "保存TIN数据"
        pSavefileDG.ShowDialog()
        If pSavefileDG.FileName = "" Then
            MsgBox("无效的保存路径！")
```

```
            Exit Sub
        End If
        TextBox1.Text = pSavefileDG.FileName
    End Sub
```

运行程序，如下图所示：



7）创建 TIN，在 Button2 的 Click 事件中添加如下代码：

```
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITinEdit = New
ESRI.ArcGIS.Geodatabase.Tin()
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxSceneControl.Scene.Layer(ComboBox1.SelectedIndex)
        Dim pEnvelope As ESRI.ArcGIS.Geometry.IEnvelope = New
ESRI.ArcGIS.Geometry.Envelope()
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass =
pFeatureLyr.FeatureClass

        Dim pFilter As ESRI.ArcGIS.Geodatabase.IQueryFilter = New
ESRI.ArcGIS.Geodatabase.QueryFilter()
        Dim pFiled As ESRI.ArcGIS.Geodatabase.IField
        pFiled =
pFeatureClass.Fields.Field(pFeatureClass.Fields.FindField(ComboBox2.T
ext))

        If pFiled.Type =
ESRI.ArcGIS.Geodatabase.esriFieldType.esriFieldTypeDouble Or
pFiled.Type =
ESRI.ArcGIS.Geodatabase.esriFieldType.esriFieldTypeInteger Or
pFiled.Type =
ESRI.ArcGIS.Geodatabase.esriFieldType.esriFieldTypeSingle Then
            Dim pGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset =
New ESRI.ArcGIS.Carto.FeatureLayer()
```

```
            pGeoDataset = pFeatureLyr
            pEnvelope = pGeoDataset.Extent

            Dim pSpatialReference As
ESRI.ArcGIS.Geometry.ISpatialReference
            pSpatialReference = pGeoDataset.SpatialReference

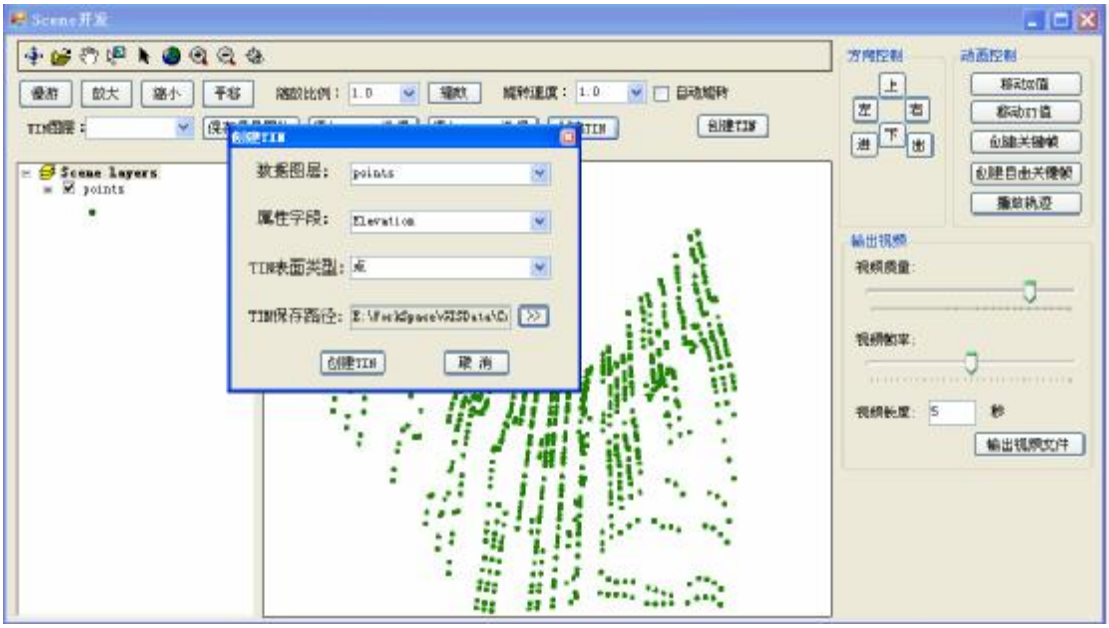
            Dim pSurfaceTypeCount As Integer = 18
            Select Case ComboBox3.Text
                Case "点"
                    pSurfaceTypeCount = 18
                Case "直线"
                    pSurfaceTypeCount = 9
                Case "光滑线"
                    pSurfaceTypeCount = 1
            End Select

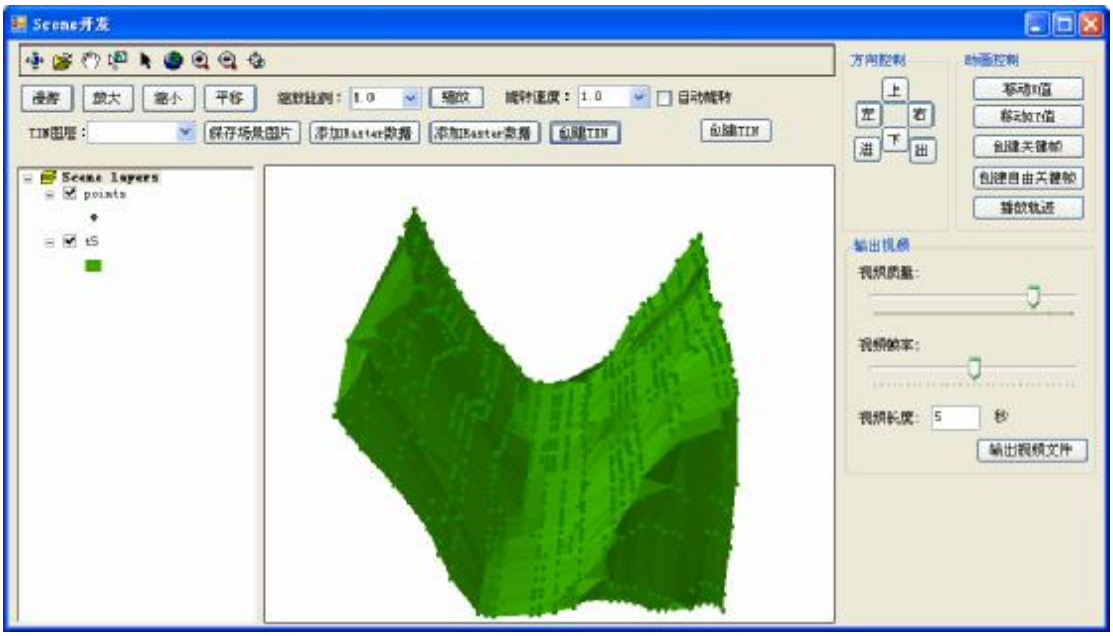
            pTIN.InitNew(pEnvelope)
            pTIN.AddFromFeatureClass(pFeatureClass, pFilter, pFiled,
pFiled, pSurfaceTypeCount)
            pTIN.SetSpatialReference(pGeoDataset.SpatialReference)

            If TextBox1.Text <> "" Then
                pTIN.SaveAs(TextBox1.Text)
            End If

            Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
            pTINLyr.Dataset = pTIN
            If TextBox1.Text = "" Then
                pTINLyr.Name = "TIN" + ComboBox1.Text
            Else
                pTINLyr.Name =
Microsoft.VisualBasic.Right(TextBox1.Text, TextBox1.Text.Length -
TextBox1.Text.LastIndexOf("\") - 1)
            End If
            AxSceneControl.Scene.AddLayer(pTINLyr)
        Else
            MsgBox("数据属性字段无效！")
        End If
```

该段代码中包含了对数据属性字段有无效的判断、添加 TIN 数据到场景中、保存 TIN 数据等操作。运行程序，如下图所示：





8）为取消按钮的 Click 事件添加如下代码：

```
    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        Me.Dispose(True)
    End Sub
```

至此，创建 TIN 窗体中所有的程序代码如下：

```
Public Class 创建TIN
    Public AxSceneControl As ESRI.ArcGIS.Controls.AxSceneControl =
Form1.AxSceneControl1

    Private Sub RefreshLayer()
        Dim i As Integer
        For i = 0 To AxSceneControl.Scene.LayerCount - 1
```

```
            ComboBox1.Items.Add(AxSceneControl.Scene.Layer(i).Name)
        Next
        ComboBox1.Text = ComboBox1.Items(0)
    End Sub


    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox1.SelectedIndexChanged
        AddFieldName(ComboBox1.Text)
    End Sub

    Private Sub 创建TIN_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        Call RefreshLayer()
        AddFieldName(ComboBox1.Text)
    End Sub

    Private Sub AddFieldName(ByVal LayerName As String)
        ComboBox2.Items.Clear()
        Dim i As Integer
        Dim pLayer As ESRI.ArcGIS.Carto.IFeatureLayer
        Dim pFields As ESRI.ArcGIS.Geodatabase.IFields
        For i = 0 To AxSceneControl.Scene.LayerCount - 1
            If AxSceneControl.Scene.Layer(i).Name = LayerName Then
                On Error Resume Next
                pLayer = AxSceneControl.Scene.Layer(i)
                Exit For
            End If
        Next
        pFields = pLayer.FeatureClass.Fields
        For i = 0 To pFields.FieldCount - 1
            ComboBox2.Items.Add(pFields.Field(i).Name)
        Next
        ComboBox2.Text = ComboBox2.Items(0)
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim pTIN As ESRI.ArcGIS.Geodatabase.ITinEdit = New
ESRI.ArcGIS.Geodatabase.Tin()
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxSceneControl.Scene.Layer(ComboBox1.SelectedIndex)
        Dim pEnvelope As ESRI.ArcGIS.Geometry.IEnvelope = New
```

```
ESRI.ArcGIS.Geometry.Envelope()
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass =
pFeatureLyr.FeatureClass

        Dim pFilter As ESRI.ArcGIS.Geodatabase.IQueryFilter = New
ESRI.ArcGIS.Geodatabase.QueryFilter()
        Dim pFiled As ESRI.ArcGIS.Geodatabase.IField
        pFiled =
pFeatureClass.Fields.Field(pFeatureClass.Fields.FindField(ComboBox2.T
ext))

        If pFiled.Type =
ESRI.ArcGIS.Geodatabase.esriFieldType.esriFieldTypeDouble Or
pFiled.Type =
ESRI.ArcGIS.Geodatabase.esriFieldType.esriFieldTypeInteger Or
pFiled.Type =
ESRI.ArcGIS.Geodatabase.esriFieldType.esriFieldTypeSingle Then
            Dim pGeoDataset As ESRI.ArcGIS.Geodatabase.IGeoDataset =
New ESRI.ArcGIS.Carto.FeatureLayer()
            pGeoDataset = pFeatureLyr
            pEnvelope = pGeoDataset.Extent

            Dim pSpatialReference As
ESRI.ArcGIS.Geometry.ISpatialReference
            pSpatialReference = pGeoDataset.SpatialReference

            Dim pSurfaceTypeCount As Integer = 18
            Select Case ComboBox3.Text
                Case "点"
                    pSurfaceTypeCount = 18
                Case "直线"
                    pSurfaceTypeCount = 9
                Case "光滑线"
                    pSurfaceTypeCount = 1
            End Select

            pTIN.InitNew(pEnvelope)
            pTIN.AddFromFeatureClass(pFeatureClass, pFilter, pFiled,
pFiled, pSurfaceTypeCount)
            pTIN.SetSpatialReference(pGeoDataset.SpatialReference)

            If TextBox1.Text <> "" Then
                pTIN.SaveAs(TextBox1.Text)
            End If
```

```
            Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
            pTINLyr.Dataset = pTIN
            If TextBox1.Text = "" Then
                pTINLyr.Name = "TIN" + ComboBox1.Text
            Else
                pTINLyr.Name =
Microsoft.VisualBasic.Right(TextBox1.Text, TextBox1.Text.Length -
TextBox1.Text.LastIndexOf("\") - 1)
            End If
            AxSceneControl.Scene.AddLayer(pTINLyr)
        Else
            MsgBox("数据属性字段无效！")
        End If
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        Me.Dispose(True)
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim pSavefileDG As SaveFileDialog = New SaveFileDialog()
        pSavefileDG.Title = "保存TIN数据"
        pSavefileDG.ShowDialog()
        If pSavefileDG.FileName = "" Then
            TextBox1.Text = ""
            MsgBox("无效的保存路径！")
            Exit Sub
        End If
        TextBox1.Text = pSavefileDG.FileName
    End Sub
   End Class
```

9）在主程序中调用这个窗体，添加一个“创建 TIN”按钮，其 Click 事件的代码如下：

```
    Private Sub Button25_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button25.Click
        Dim p As New 创建TIN
        p.Show()
       End Sub
```

12、设置图层基准高程

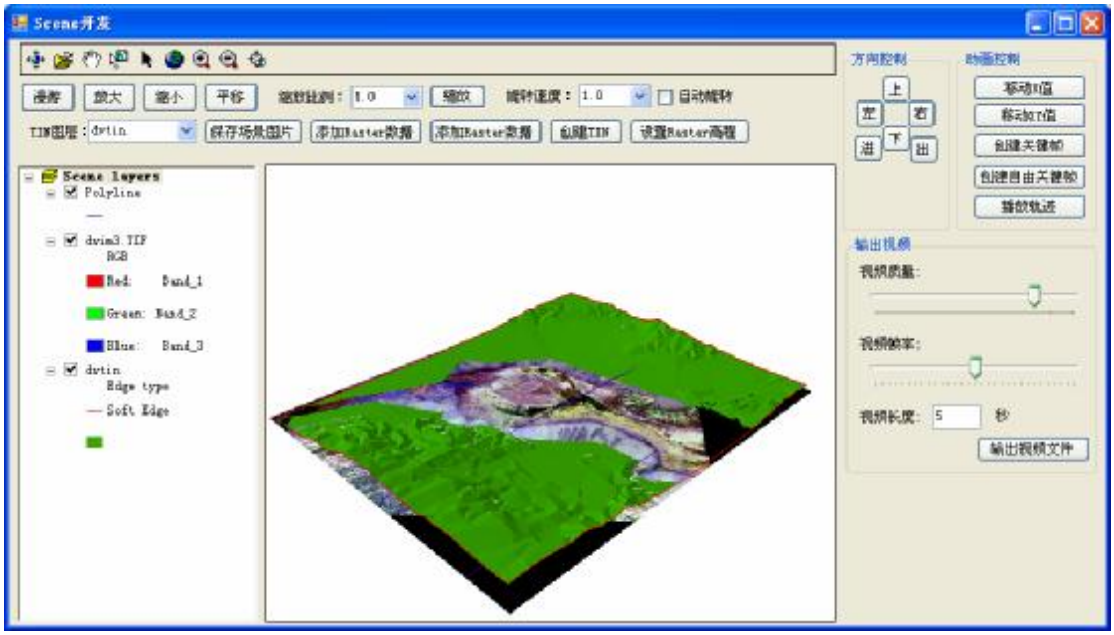
设置图层基准高程在 GIS 的三维场景中非常重要，如要将地物赋予地形表面上。在 ArcGIS Engine 中主要通过 ISurface——I3DProperties——ILayerExtensions 来实现。其中 ISurface 用于获取地形的表面，如 TIN 或 Raster 的表面，I3DProperties 是一种扩展的三维属性，通过 ILayerExtensions 可以将 I3DProperties 和需要设置高程的图层连接。

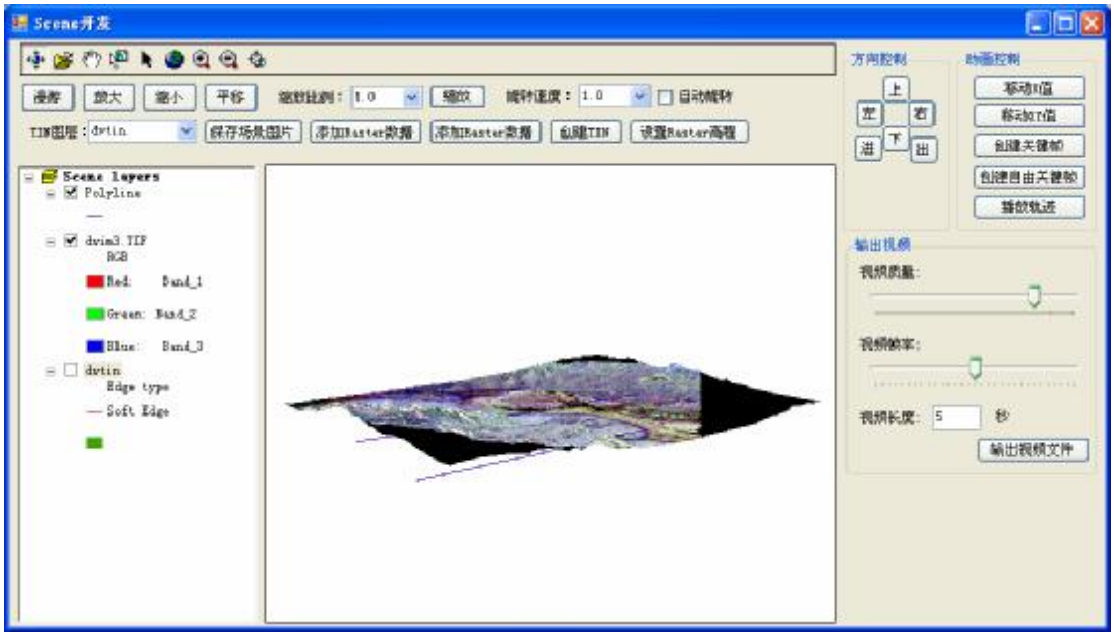
1）添加一个“设置 Raster 高程”按钮，其 Click 事件的代码如下：

```
 Private Sub Button24_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button24.Click
        Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        Dim pRasterLyr As ESRI.ArcGIS.Carto.IRasterLayer = New
ESRI.ArcGIS.Carto.RasterLayer()
        pTINLyr = AxSceneControl1.Scene.Layer(2)
        pRasterLyr = AxSceneControl1.Scene.Layer(1)
        Dim pI3DProperties As ESRI.ArcGIS.Analyst3D.I3DProperties = New
ESRI.ArcGIS.Analyst3D.Feature3DProperties()
        Dim pISurface As ESRI.ArcGIS.Geodatabase.ISurface = New
ESRI.ArcGIS.Geodatabase.Tin()
        pISurface = pTINLyr.Dataset
        pISurface.ZFactor = 1
        Dim p As ESRI.ArcGIS.Carto.ILayerExtensions = pRasterLyr
        Dim pp As Object
        Dim i As Integer
        For i = 0 To p.ExtensionCount - 1
            pp = p.Extension(i)
            If pp IsNot Nothing Then
                pI3DProperties = p.Extension(i)
                Exit For
            End If
        Next
        pI3DProperties.BaseOption =
ESRI.ArcGIS.Analyst3D.esriBaseOption.esriBaseSurface
        pI3DProperties.BaseSurface = pISurface
        pI3DProperties.Apply3DProperties(pRasterLyr)
        AxSceneControl1.SceneGraph.RefreshViewers()
       End Sub
```

其中，pISurface.ZFactor = 1是设置地形表面的高程倍数因子，通过业可以通过pI3DProperties.ZFactor 来设置。

2）运行程序，其结果如下图所示：



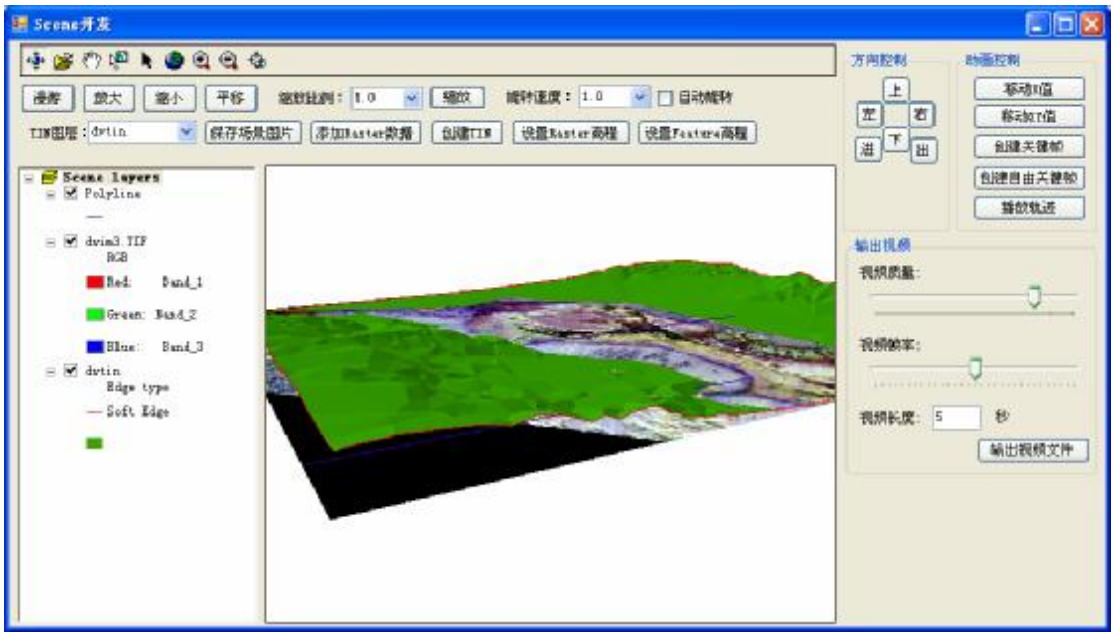


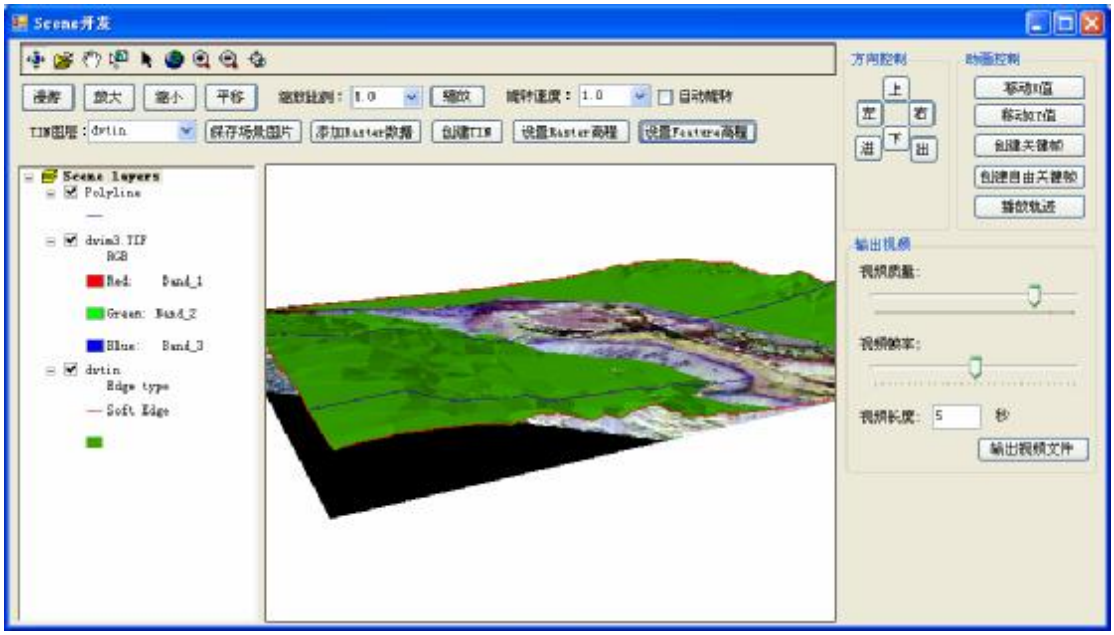
3）前面是设置 Raster 的基准高程，还可以设置其他数据，如 Feature 的基准高程。添加一个“设置 Feature 高程”按钮，其 Click 事件代码如下：

```
    Private Sub Button26_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button26.Click
        Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        pTINLyr = AxSceneControl1.Scene.Layer(2)
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxSceneControl1.Scene.Layer(0)
        Dim pI3DProperties As ESRI.ArcGIS.Analyst3D.I3DProperties = New
ESRI.ArcGIS.Analyst3D.Feature3DProperties()
        Dim pISurface As ESRI.ArcGIS.Geodatabase.ISurface = New
```

```
ESRI.ArcGIS.Geodatabase.Tin()
        pISurface = pTINLyr.Dataset
        pISurface.ZFactor = 1
        Dim p As ESRI.ArcGIS.Carto.ILayerExtensions = pFeatureLyr
        Dim pp As Object
        Dim i As Integer
        For i = 0 To p.ExtensionCount - 1
            pp = p.Extension(i)
            If pp IsNot Nothing Then
                pI3DProperties = p.Extension(i)
                Exit For
            End If
        Next
        pI3DProperties.BaseOption =
ESRI.ArcGIS.Analyst3D.esriBaseOption.esriBaseSurface
        pI3DProperties.BaseSurface = pISurface
        pI3DProperties.Apply3DProperties(pFeatureLyr)
        AxSceneControl1.SceneGraph.RefreshViewers()
```

4）运行程序，其结果如下图所示：





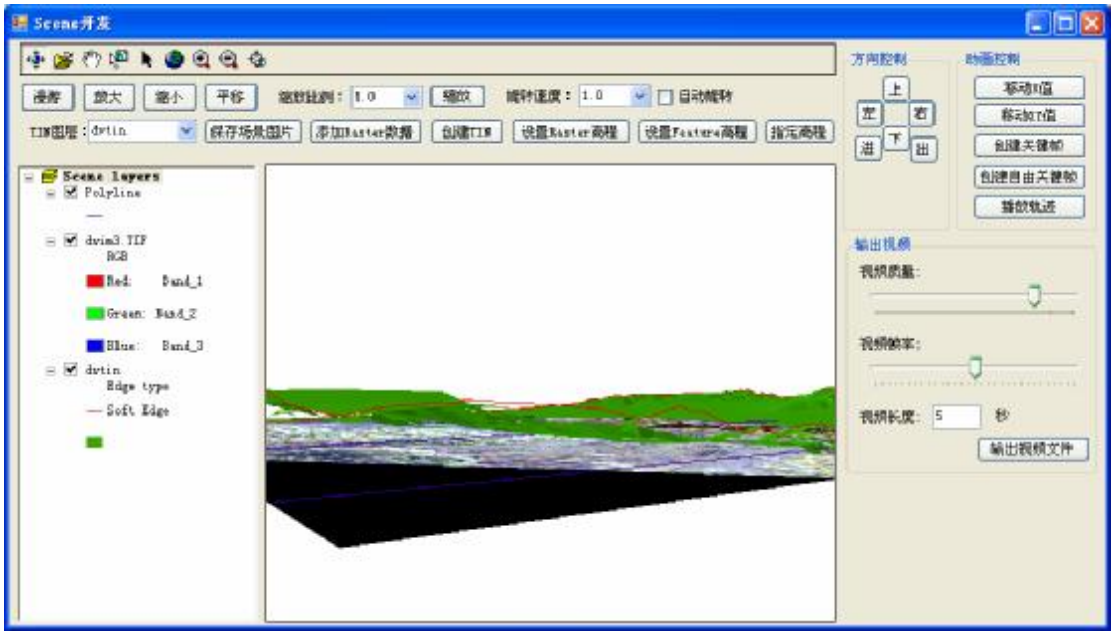
5）自定义高程。自定义高程在三维场景中也用的比较多，如设置水面的高度，其思路和前面的大同小异。添加一个“指定高程”的按钮，其 Click 事件代码如下：

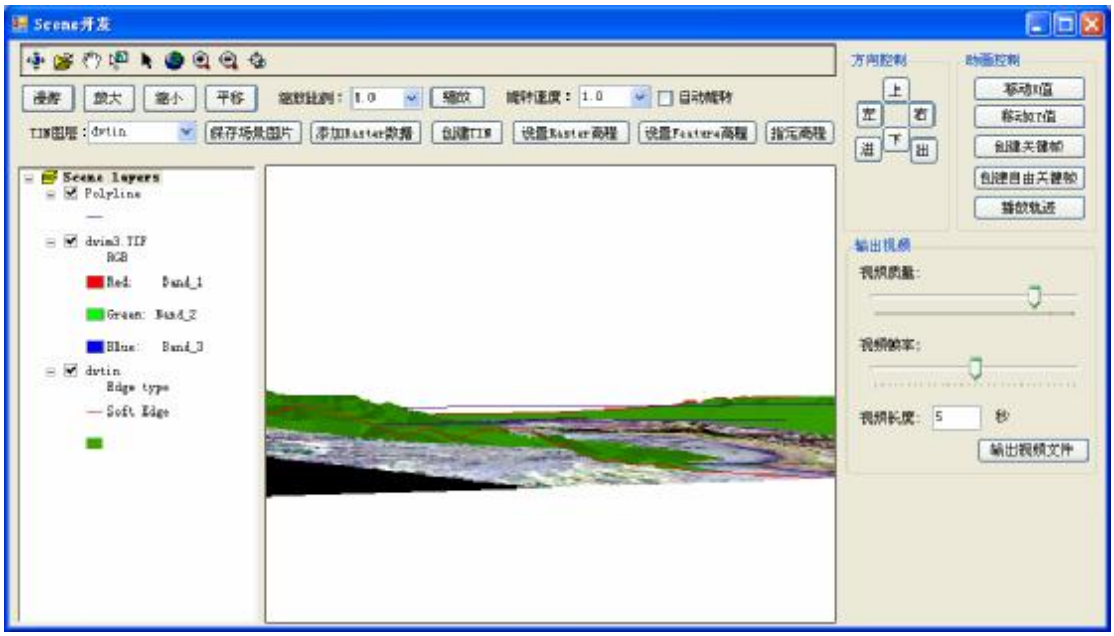
```
    Private Sub Button27_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button27.Click
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxSceneControl1.Scene.Layer(0)
        Dim pI3DProperties As ESRI.ArcGIS.Analyst3D.I3DProperties = New
ESRI.ArcGIS.Analyst3D.Feature3DProperties()
        Dim p As ESRI.ArcGIS.Carto.ILayerExtensions = pFeatureLyr
        Dim pp As Object
        Dim i As Integer
        For i = 0 To p.ExtensionCount - 1
            pp = p.Extension(i)
            If pp IsNot Nothing Then
                pI3DProperties = p.Extension(i)
                Exit For
            End If
        Next
        pI3DProperties.BaseOption =
ESRI.ArcGIS.Analyst3D.esriBaseOption.esriBaseExpression
        pI3DProperties.BaseExpressionString = "1000"
        pI3DProperties.Apply3DProperties(pFeatureLyr)
        AxSceneControl1.SceneGraph.RefreshViewers()
      End Sub
```

其中，pI3DProperties.BaseOption = ESRI.ArcGIS.Analyst3D.esriBaseOption.esriBaseExpression是指定其设置基准高程的方式为表达式。pI3DProperties.BaseExpressionString = "1000"就是

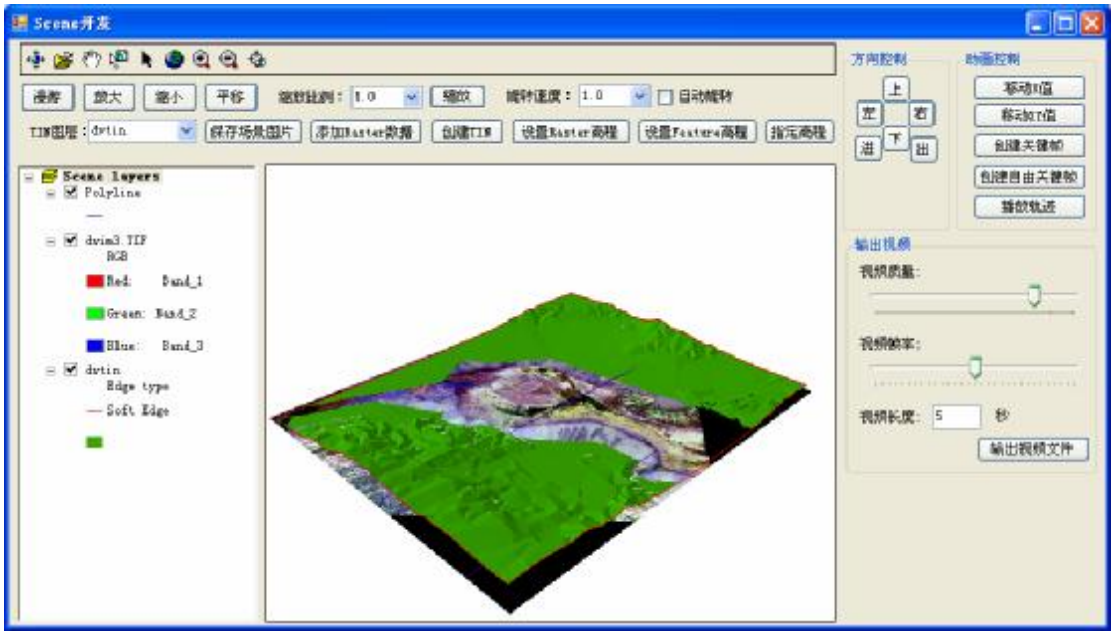
设置其高度为1000。

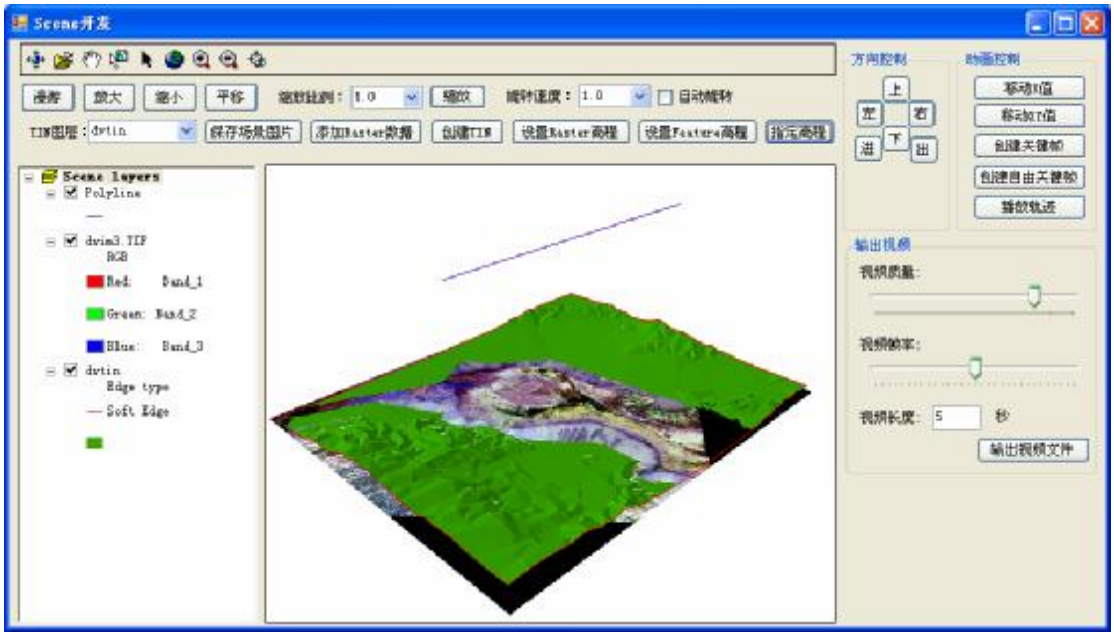
6）运行程序，其结果如下图所示：





7）同样可以通过表达式来设置其基准高程，如将前面的代码“pI3DProperties.BaseExpressionString = "1000" ”改为“pI3DProperties.BaseExpressionString = "[length]"”，运行程序，其结果如下图所示：





当然，要求设置的数据具有这个字段。

8）设置偏移值。通过 pI3DProperties.OffsetExpressionString = "-1000" 即可设置其偏移值为 1000。

9）通过添加一个 Timer 控件，结合设置 pI3DProperties.BaseExpressionString 即可实现水位上升的动画。如添加一个“Timer4”的时间控件，添加一个“水位上升动画”的按钮。

首先在全局变量中定义如下：

```
Public startHeight, endHeight, nowHeight As Double
```

然后在“水位上升动画”按钮的 Click 事件中添加如下代码：

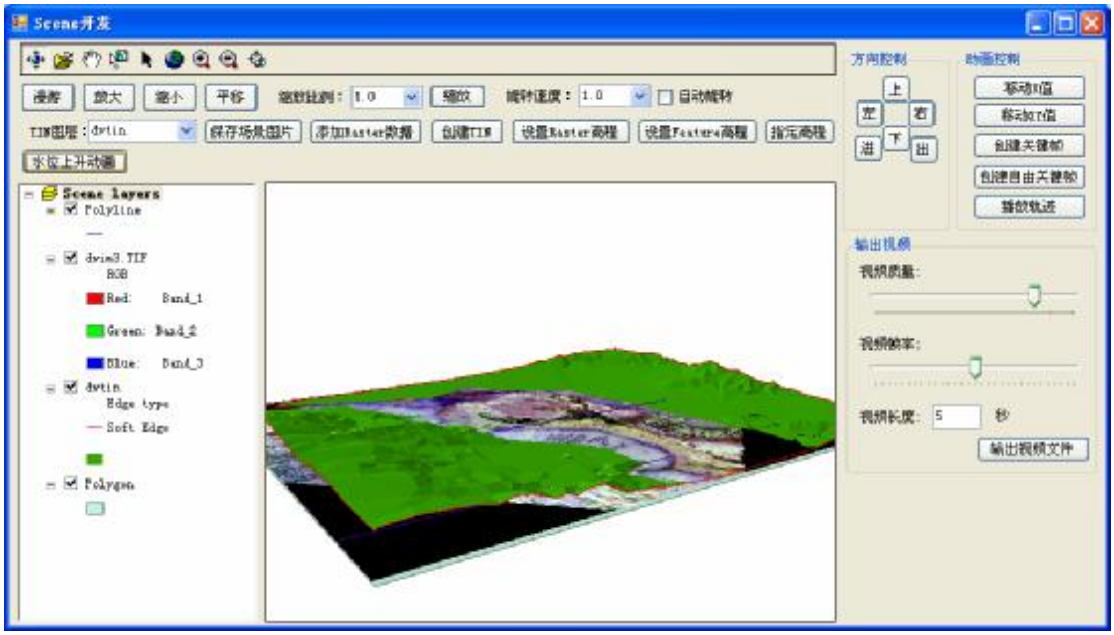
```
Private Sub Button28_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button28.Click
        startHeight = -300
```

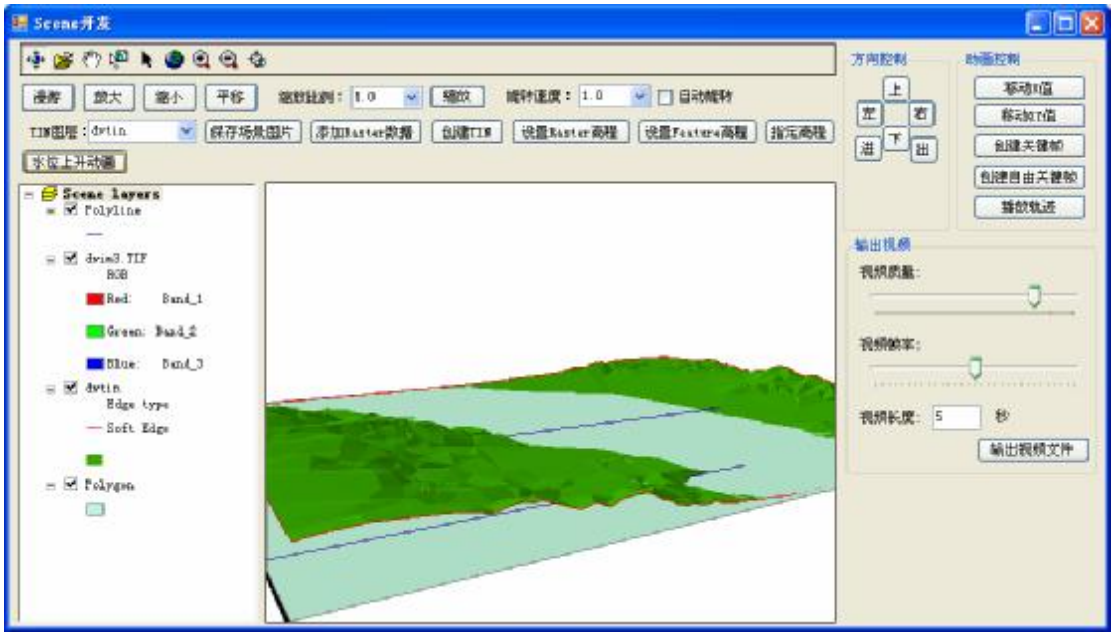
```
        endHeight = 1000
        nowHeight = startHeight
        Timer4.Enabled = True
    End Sub
```

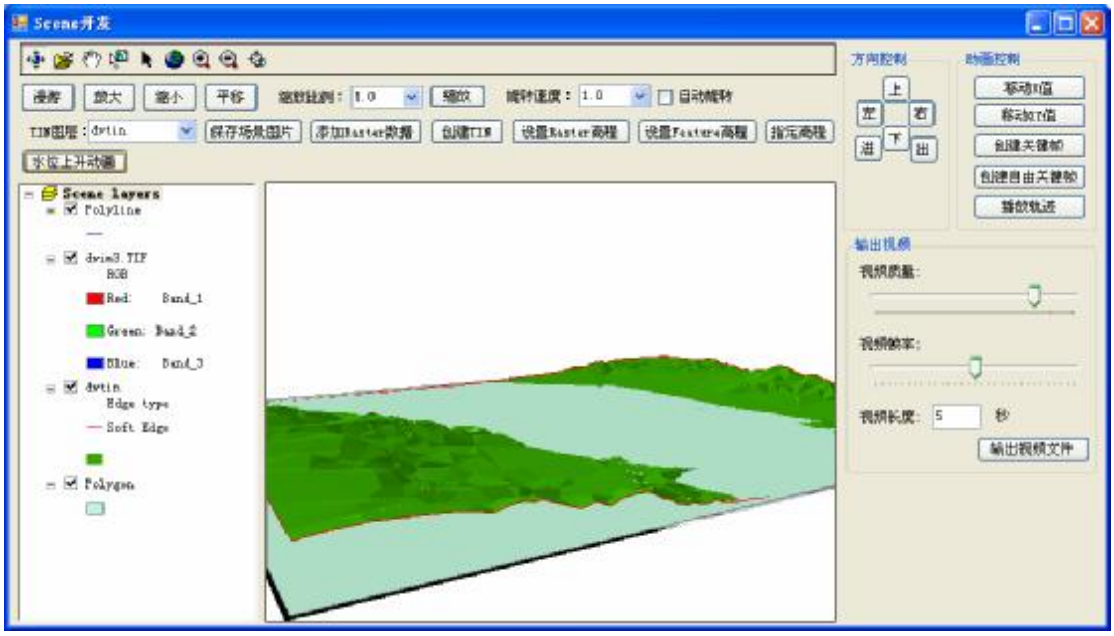
“Timer4”的时间控件 Tick 事件中程序代码如下：

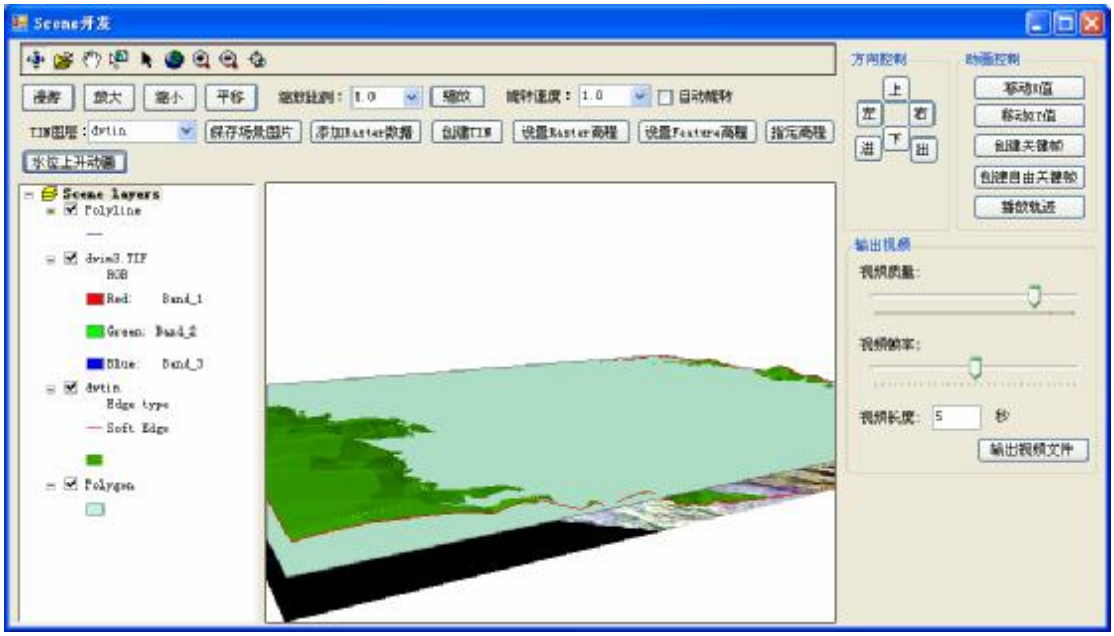
```
    Private Sub Timer4_Tick(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Timer4.Tick
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxSceneControl1.Scene.Layer(3)
        Dim pI3DProperties As ESRI.ArcGIS.Analyst3D.I3DProperties = New
ESRI.ArcGIS.Analyst3D.Feature3DProperties()
        Dim p As ESRI.ArcGIS.Carto.ILayerExtensions = pFeatureLyr
        Dim pp As Object
        Dim i As Integer
        For i = 0 To p.ExtensionCount - 1
            pp = p.Extension(i)
            If pp IsNot Nothing Then
                pI3DProperties = p.Extension(i)
                Exit For
            End If
        Next
        pI3DProperties.BaseOption =
ESRI.ArcGIS.Analyst3D.esriBaseOption.esriBaseExpression
        nowHeight = nowHeight + 10
        pI3DProperties.BaseExpressionString = nowHeight.ToString()
        pI3DProperties.Apply3DProperties(pFeatureLyr)
        AxSceneControl1.SceneGraph.RefreshViewers()
        If nowHeight >= endHeight Then Timer4.Enabled = False
    End Sub
```

运行程序，其结果如下图所示：


Scene开发
漫游
放大
缩小
平移
缩放
自动旋转
保存场景图片
方向控制
动画控制
输出视频
视频质量:
视频帧率:
视频长度:
秒
输出视频文件


Scene开发
漫游
放大
缩小
平移
缩放
自动旋转
保存场景图片
方向控制
动画控制
输出视频
视频质量:
视频帧率:
视频长度:
秒
输出视频文件





13、点击查询

通过在 SceneControl 控件中点击来查询对象是三维场景开发中的一个重要组成部分。它主要通过 AxSceneControl 的 SceneGraph.LocateMultiple 获得点击到的对象 IHit3DSet，然后通过 IHit3D 获取到 IHit3DSet 的元素。

1）首先，添加一个“点击查询”按钮，并在全局变量中定义如下：

```
Public pIdnetifyIsOrNot As Boolean
```

然后在点击查询”按钮的 Click 事件中添加如下代码：

```
Private Sub Button21_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button21.Click
        pIdnetifyIsOrNot = True
```

```
        End Sub
```

2）由于鼠标点击事件是在 SceneControl 控件中进行的，所以在 AxSceneControl1 的 OnMouseDown 事件中添加如下代码：

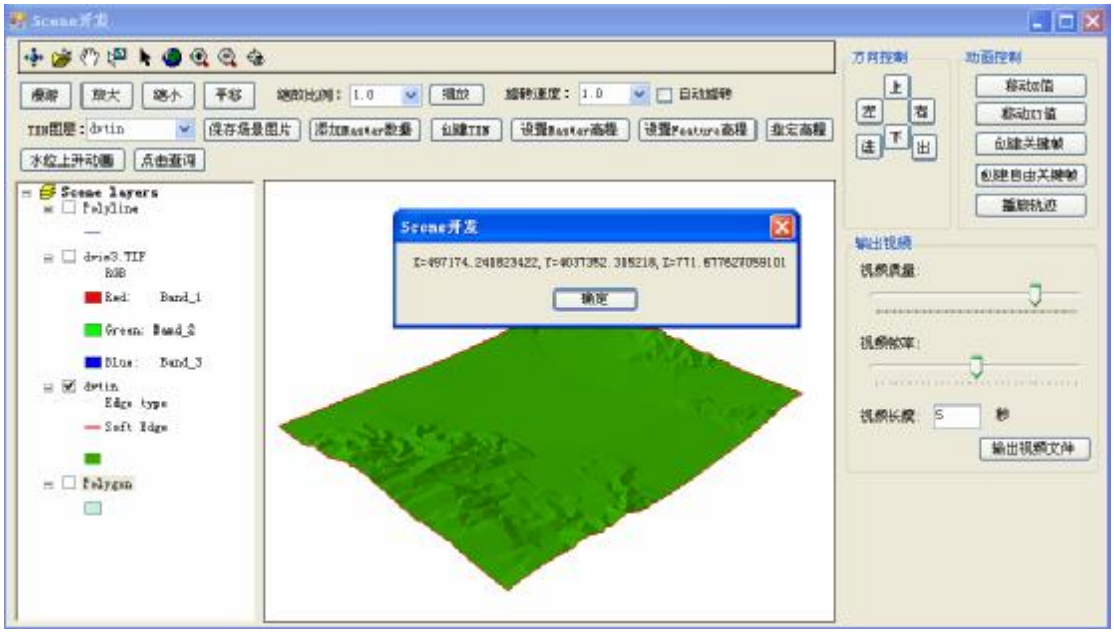
```
        If pIdnetifyIsOrNot = True Then
            Dim pHit3DSet As ESRI.ArcGIS.Analyst3D.IHit3DSet

AxSceneControl1.SceneGraph.LocateMultiple(AxSceneControl1.SceneGraph.
ActiveViewer, e.x, e.y,
ESRI.ArcGIS.Analyst3D.esriScenePickMode.esriScenePickAll, False,
pHit3DSet)
            pHit3DSet.OnePerLayer()
            If pHit3DSet Is Nothing Then
                MsgBox("没有点击到对象！")
                Exit Sub
            End If

            Dim pHit3D As ESRI.ArcGIS.Analyst3D.IHit3D
            pHit3D = pHit3DSet.Hits.Element(0)
            pIdnetifyIsOrNot = False
            MsgBox("X=" + pHit3D.Point.X.ToString + ",Y=" +
pHit3D.Point.Y.ToString + ",Z=" + pHit3D.Point.Z.ToString)
          End If
```

其中，pHit3D = pHit3DSet.Hits.Element(0)是指获取的 pHit3DSet 中点击的第一个元素。

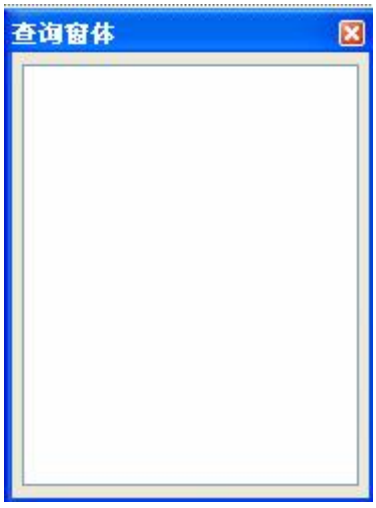
3）运行程序，如下图所示：



4）除了可以显示点击点的三维坐标，还可以获取到对象所在的图层。如下图所示，添加一个“查询窗体”：



在查询窗体上添加一个 TreeView 控件，如下图所示：



5）查询窗体中所有的程序代码如下：

```
Public Class 查询窗体
    Public pHit3DSet As ESRI.ArcGIS.Analyst3D.IHit3DSet

    Public Sub RefreshTreeView()
        TreeView1.BeginUpdate()
        TreeView1.Nodes.Clear()
        Dim pHit3D As ESRI.ArcGIS.Analyst3D.IHit3D
        Dim i As Integer
        For i = 0 To pHit3DSet.Hits.Count - 1
            pHit3D = pHit3DSet.Hits.Element(i)
            If TypeOf pHit3D.Owner Is ESRI.ArcGIS.Carto.ILayer Then
                Dim pLayer As ESRI.ArcGIS.Carto.ILayer
                pLayer = pHit3D.Owner
                TreeView1.Nodes.Add(pLayer.Name.ToString())
```

```
                TreeView1.Nodes(i).Nodes.Add("X=" +
pHit3D.Point.X.ToString)
                TreeView1.Nodes(i).Nodes.Add("Y=" +
pHit3D.Point.Y.ToString)
                TreeView1.Nodes(i).Nodes.Add("Z=" +
pHit3D.Point.Z.ToString)
            End If
        Next
        TreeView1.EndUpdate()
    End Sub
End Class
```

6）在 AxSceneControl1 的 OnMouseDown 事件中将前面的代码改为如下代码：

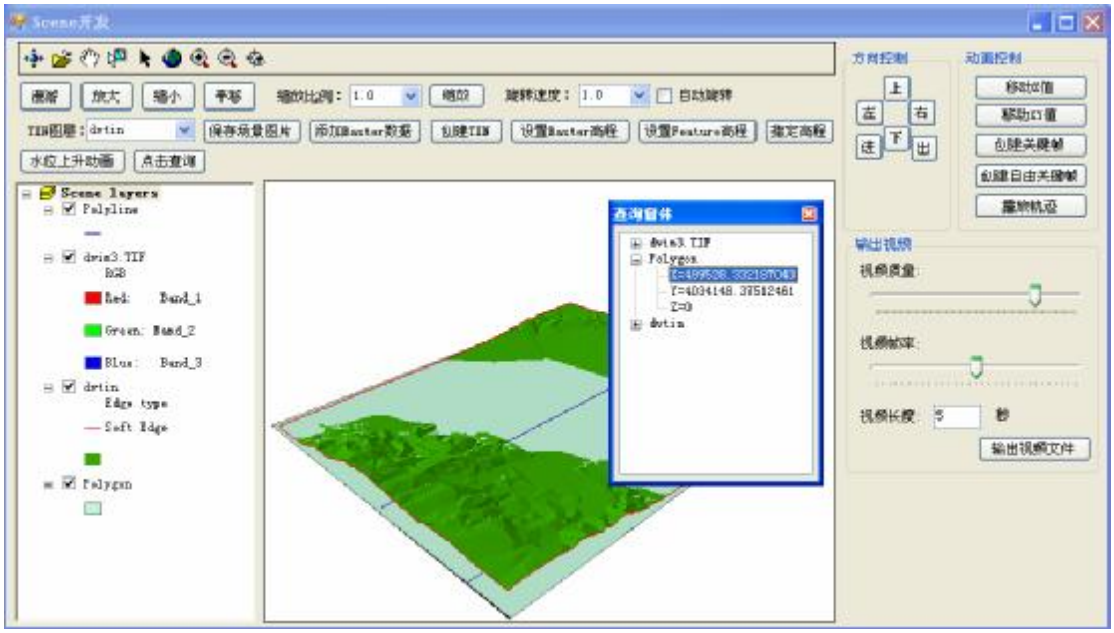
```
        If pIdnetifyIsOrNot = True Then
            Dim pHit3DSet As ESRI.ArcGIS.Analyst3D.IHit3DSet

AxSceneControl1.SceneGraph.LocateMultiple(AxSceneControl1.SceneGraph.
ActiveViewer, e.x, e.y,
ESRI.ArcGIS.Analyst3D.esriScenePickMode.esriScenePickAll, False,
pHit3DSet)
            pHit3DSet.OnePerLayer()
            If pHit3DSet Is Nothing Then
                MsgBox("没有点击到对象！")
                Exit Sub
            End If

            Dim ShowIdentifyForm As New 查询窗体
            ShowIdentifyForm.pHit3DSet = pHit3DSet
            ShowIdentifyForm.RefreshTreeView()
            ShowIdentifyForm.Show()

            pIdnetifyIsOrNot = False
           End If
```

7）运行程序，其显示结果如下图所示：



8）显示选择数据的属性。如果选择的数据是 Feature，还可以显示其属性，其主要是通过 IHit3D 接口的 Object 对象来实现。在查询窗体的 RefreshTreeView()中添加如下代码：

```
If Not pHit3D.Object Is Nothing Then
                    If TypeOf pHit3D.Object Is
ESRI.ArcGIS.Geodatabase.IFeature Then
                        Dim pFeature As
ESRI.ArcGIS.Geodatabase.IFeature
                        pFeature = pHit3D.Object
                        Dim j As Integer
                        For j = 0 To pFeature.Fields.FieldCount - 1

TreeView1.Nodes(i).Nodes.Add(pFeature.Fields.Field(j).Name.ToString()
+ ":" + pFeature.Value(j).ToString())
                        Next
                    End If
```

至此，整个 RefreshTreeView()函数的程序代码如下：

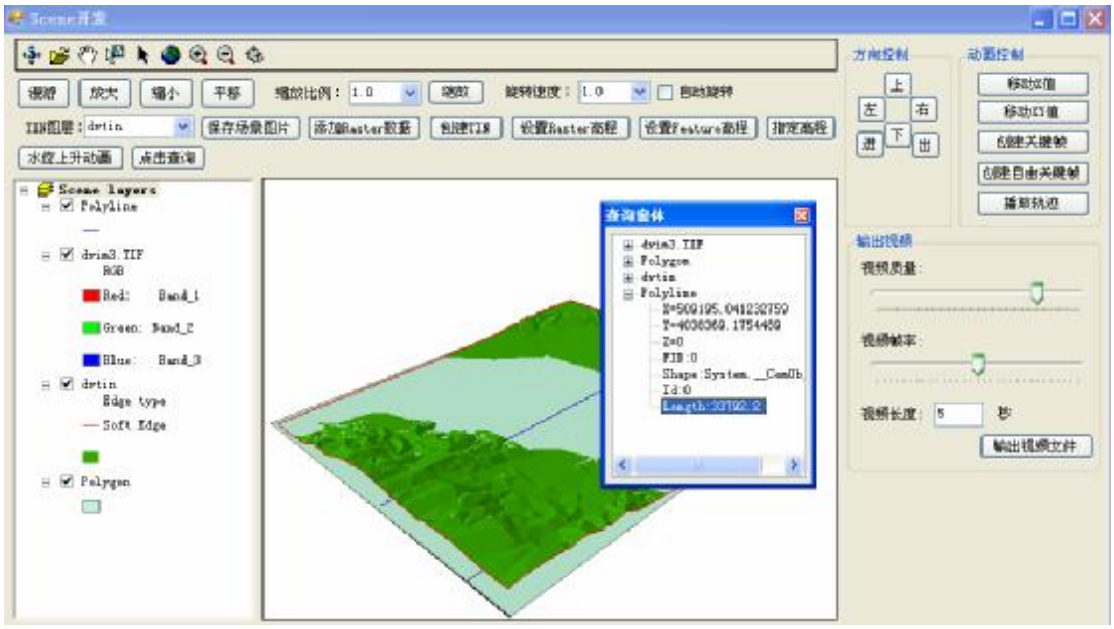
```
 Public Sub RefreshTreeView()
        TreeView1.BeginUpdate()
        TreeView1.Nodes.Clear()
        Dim pHit3D As ESRI.ArcGIS.Analyst3D.IHit3D
        Dim i As Integer
        For i = 0 To pHit3DSet.Hits.Count - 1
            pHit3D = pHit3DSet.Hits.Element(i)
            If TypeOf pHit3D.Owner Is ESRI.ArcGIS.Carto.ILayer Then
                Dim pLayer As ESRI.ArcGIS.Carto.ILayer
                pLayer = pHit3D.Owner
                TreeView1.Nodes.Add(pLayer.Name.ToString())
```

```
                TreeView1.Nodes(i).Nodes.Add("X=" +
pHit3D.Point.X.ToString)
                TreeView1.Nodes(i).Nodes.Add("Y=" +
pHit3D.Point.Y.ToString)
                TreeView1.Nodes(i).Nodes.Add("Z=" +
pHit3D.Point.Z.ToString)

                If Not pHit3D.Object Is Nothing Then
                    If TypeOf pHit3D.Object Is
ESRI.ArcGIS.Geodatabase.IFeature Then
                        Dim pFeature As
ESRI.ArcGIS.Geodatabase.IFeature
                        pFeature = pHit3D.Object
                        Dim j As Integer
                        For j = 0 To pFeature.Fields.FieldCount - 1

TreeView1.Nodes(i).Nodes.Add(pFeature.Fields.Field(j).Name.ToString()
+ ":" + pFeature.Value(j).ToString())
                        Next
                    End If
                End If
            End If
        Next
        TreeView1.EndUpdate()
    End Sub
```

运行程序，其显示结果如下图所示：

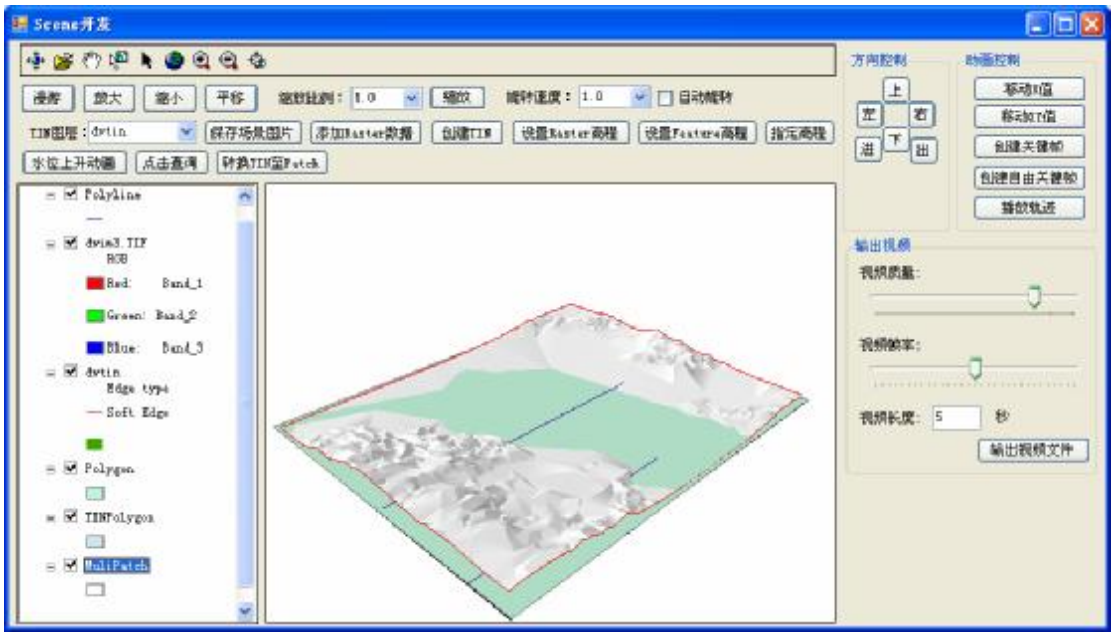


14、转换 TIN 成 MultiPatches

将 TIN 转换成 MultiPatches 的关键在于使用 ITinSurface2 接口的 ConvertToMultiPatches 方法。添加一个“转换 TIN 至 Patch”的按钮，其 Click 事件的程序代码如下：

```
Private Sub Button20_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button20.Click
        Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New ESRI.ArcGIS.Carto.TinLayer()
        pTINLyr = AxSceneControl1.Scene.Layer(2)
        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer = AxSceneControl1.Scene.Layer(5)
        Dim pFeatureClass As ESRI.ArcGIS.Geodatabase.IFeatureClass = pFeatureLyr.FeatureClass
        Dim pTinSurface2 As ESRI.ArcGIS.Geodatabase.ITinSurface2
        pTinSurface2 = pTINLyr.Dataset
        pTinSurface2.ConvertToMultiPatches(pFeatureClass, 100, 10)
    End Sub
```

运行程序，其显示结果如下图所示：



15、获取 TIN 中对象

通过获取 TIN 中对象主要通过 ITinAdvanced2 接口的相关方法或属性来实现。如通过 NodeCount 可以获得节点数目、通过 TriangleCount 可以获得三角形的数目、通过 GetTriangle 可以获得三角形等等，下面只简单介绍一些常用的属性和方法：

1）添加一个“获取 TIN 中对象”按钮，并在全局变量中添加如下定义：

```
Public FindTriangleIndexIsNot As Boolean
```

“获取 TIN 中对象”按钮的 Click 事件中添加如下代码：

```
Private Sub Button23_Click(ByVal sender As System.Object, ByVal e As
```

```
System.EventArgs) Handles Button23.Click
        FindTriangleIndexIsNot = True
       End Sub
```

2）由于点击事件要在 SceneControl 控件中进行的，所以在 AxSceneControl1 的 OnMouseDown 事件中添加如下代码：

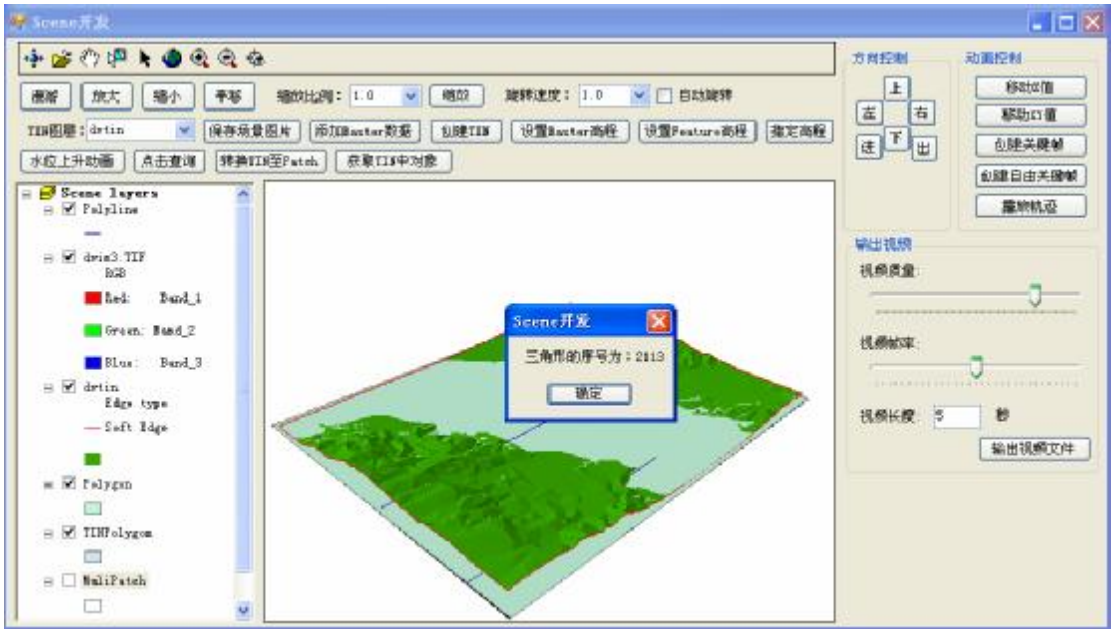
```
If e.button = 1 Then
            If FindTriangleIndexIsNot Then
                Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
                pTINLyr = AxSceneControl1.Scene.Layer(2)
                Dim pTINAD As ESRI.ArcGIS.Geodatabase.ITinAdvanced2 =
New ESRI.ArcGIS.Geodatabase.Tin()
                pTINAD = pTINLyr.Dataset

                Dim pHit3DSet As ESRI.ArcGIS.Analyst3D.IHit3DSet

AxSceneControl1.SceneGraph.LocateMultiple(AxSceneControl1.SceneGraph.
ActiveViewer, e.x, e.y,
ESRI.ArcGIS.Analyst3D.esriScenePickMode.esriScenePickAll, False,
pHit3DSet)
                pHit3DSet.OnePerLayer()
                If pHit3DSet Is Nothing Then
                    MsgBox("没有点击到对象！")
                    Exit Sub
                End If
                Dim pHit3D As ESRI.ArcGIS.Analyst3D.Hit3D =
pHit3DSet.Hits.Element(0)
                MsgBox("三角形的序号为：" +
pTINAD.FindTriangleIndex(pHit3D.Point).ToString())
                FindTriangleIndexIsNot = False
            End If
          End If
```

此处又用到了 IHit3DSet 接口和 Hit3D 类，其主要原因是在 SceneControl 中点击的点要转换成三维空间的点。MsgBox(pTINAD.FindTriangleIndex(pHit3D.Point))就是获取到点击的三角形的序号。

运行程序，其显示结果如下图所示：



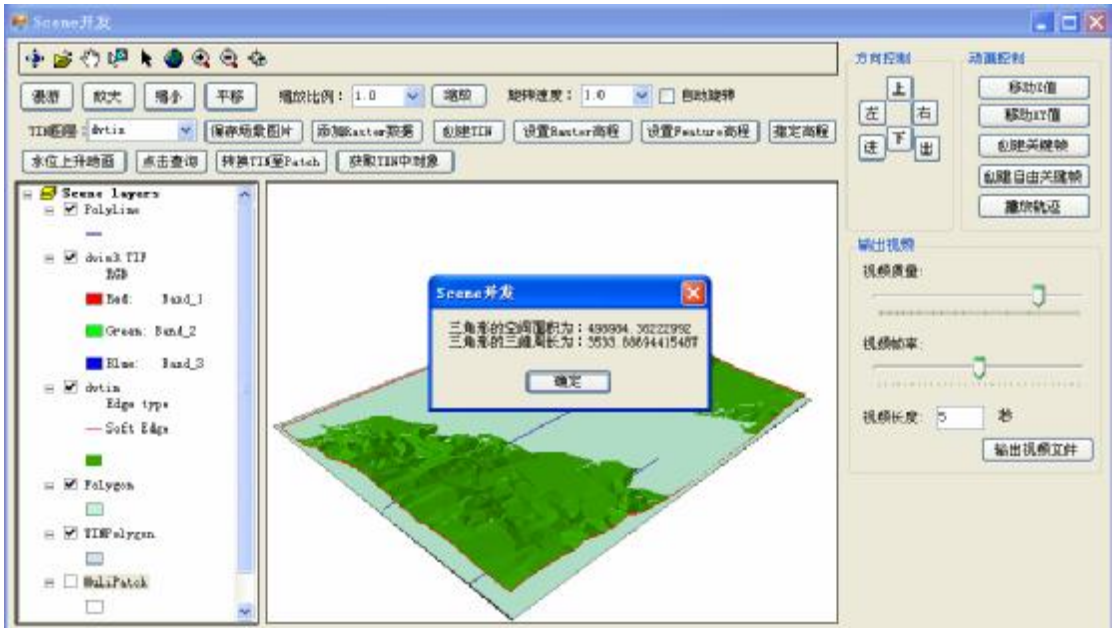
3）获取三角形的面积及周长。

获取三角形的面积及周长主要通过 ITinTriangle 接口的 Area3D 属性和 Perimeter3D 属性获得，当然它还有许多其他的属性，如平面周长、平面面积、坡度、法线、节点等等。

在 AxSceneControl1 的 OnMouseDown 事件中添加如下代码：

```
MsgBox("三角形的空间面积为：" +
pTINAD.GetTriangle(pTINAD.FindTriangleIndex(pHit3D.Point)).Area3D.ToString() + Chr(13) + "三角形的三维周长为：" +
pTINAD.GetTriangle(pTINAD.FindTriangleIndex(pHit3D.Point)).Perimeter3D.ToString())
```

运行程序，其显示结果如下图所示：



4）获得节点

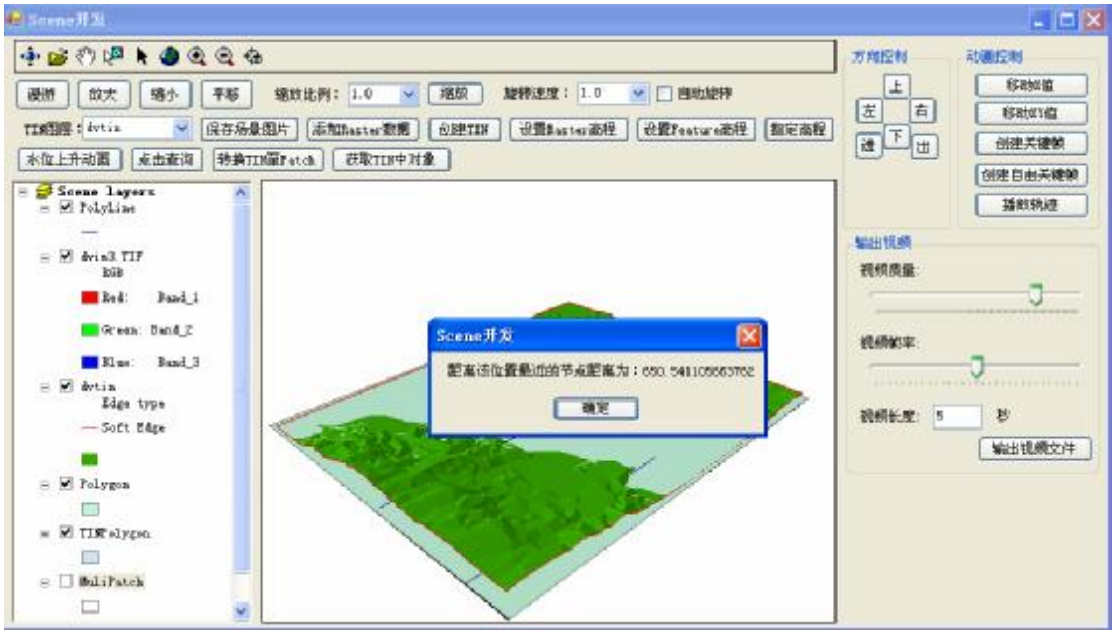
获得节点主要是通过 GetNode、QueryNode 等方法获得节点对象，获取到节点

后，通过这些节点也可以实现许多功能。下面以查询距离点击位置最近的节点。

在 AxSceneControl1 的 OnMouseDown 事件中添加如下代码：

```
Dim pNode As ESRI.ArcGIS.Geodatabase.ITinNode = New
ESRI.ArcGIS.Geodatabase.TinNode()
Dim nearestDistance As Double
pTINAD.QueryNearestNode(pHit3D.Point, pNode,
nearestDistance)
MsgBox("距离该位置最近的节点距离为：" +
nearestDistance.ToString())
```

运行程序，其显示结果如下图所示：



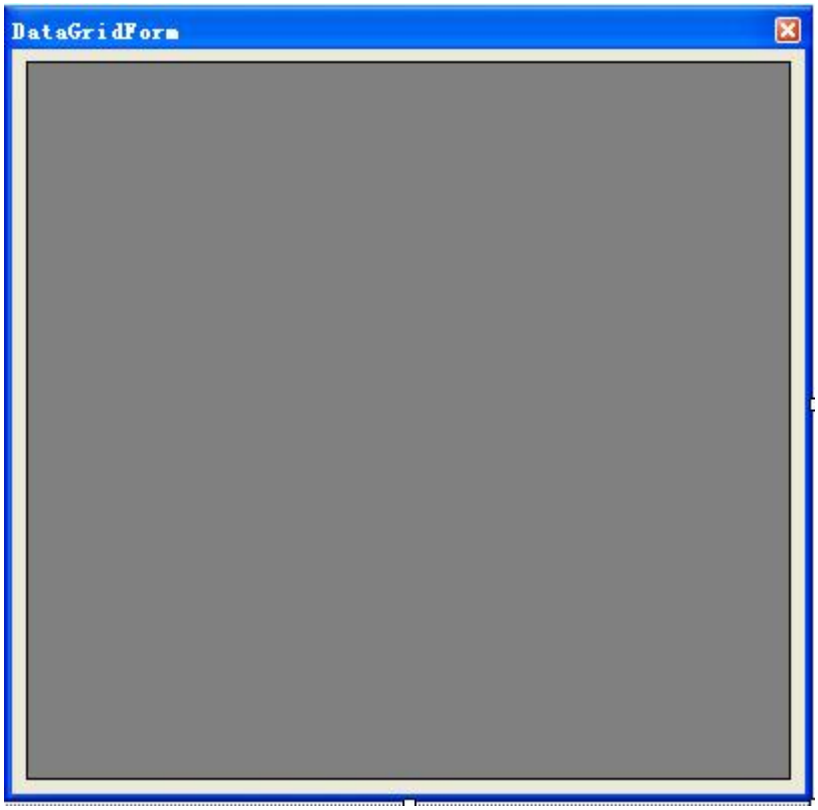
16、输出 TIN 节点坐标

输出 TIN 节点坐标主要是通过 ITinNode 接口的 GetNode 方法来获得节点对象，然后通过节点的 X、Y 和 Z 值得到坐标值。

1）新建一个“DataGridForm”窗体用于显示数据，如下图所示：



并在 DataGridForm 窗体中添加一个 DataGridView1 控件，如下图所示：



2）DataGridForm 窗体中所有的程序代码如下：

```
Public Class DataGridForm
    Public x(), y(), z() As Double
    Public count As Integer
    Public Sub CreateGrid()
        DataGridView1.ColumnCount = 4
        DataGridView1.Columns(0).Name = "序号"
```

```
        DataGridView1.Columns(1).Name = "X"
        DataGridView1.Columns(2).Name = "Y"
        DataGridView1.Columns(3).Name = "Z"

        Dim i As Integer
        For i = 0 To count - 1
            DataGridView1.Rows.Add(1)
            DataGridView1.Rows(i).Cells(0).Value = (i + 1).ToString()
            DataGridView1.Rows(i).Cells(1).Value = x(i)
            DataGridView1.Rows(i).Cells(2).Value = y(i)
            DataGridView1.Rows(i).Cells(3).Value = z(i)
        Next
    End Sub
End Class
```

其中函数 CreateGrid()用于创建 DataGridView1 控件中的数据列表。

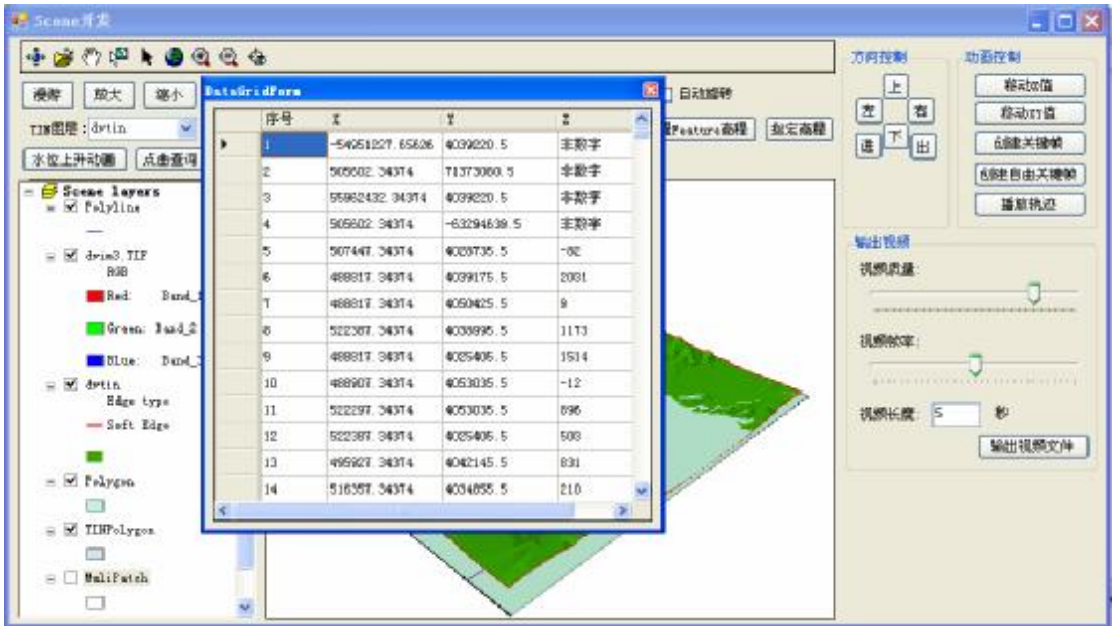
3）创建一个“输出 TIN 节点坐标”按钮，其 Click 事件代码如下所示：

```
    Private Sub Button29_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button29.Click
        Dim x(), y(), z() As Double
        Dim count As Integer
        Dim pTINLyr As ESRI.ArcGIS.Carto.ITinLayer = New
ESRI.ArcGIS.Carto.TinLayer()
        pTINLyr = AxSceneControl1.Scene.Layer(2)
        Dim pTINAD As ESRI.ArcGIS.Geodatabase.ITinAdvanced2 = New
ESRI.ArcGIS.Geodatabase.Tin()
        pTINAD = pTINLyr.Dataset
        Dim i As Integer
        Dim pNode As ESRI.ArcGIS.Geodatabase.ITinNode = New
ESRI.ArcGIS.Geodatabase.TinNode()

        For i = 1 To pTINAD.NodeCount
            ReDim Preserve x(i)
            ReDim Preserve y(i)
            ReDim Preserve z(i)
            pNode = pTINAD.GetNode(i)
            x(i - 1) = pNode.X
            y(i - 1) = pNode.Y
            z(i - 1) = pNode.Z
        Next
        count = i - 1
        Dim pDataGridForm As New DataGridForm()
        DataGridForm.x = x
        DataGridForm.y = y
        DataGridForm.z = z
```

```
        DataGridForm.count = count
        DataGridForm.CreateGrid()
        DataGridForm.Show()
    End Sub
```

4）运行程序，其显示结果如下图所示：



17、设置点的模型

在 SceneControl 中设置点的模型和在 MapControl 中自定义点的符号的方式差不多，都是通过获取到图层，然后通过 ISimpleRenderer 来渲染。

1）添加一个“设置点的模型”按钮，其 Click 事件的代码如下：

```
Private Sub Button30_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button30.Click
        Dim pOpenDg As OpenFileDialog = New OpenFileDialog()
        pOpenDg.Title = "打开模型文件"
        pOpenDg.Filter = "模型文件(*.3ds)|*.3ds"
        pOpenDg.ShowDialog()
        Dim marker3DSymbolfile As String
        marker3DSymbolfile = pOpenDg.FileName
        Dim p As ESRI.ArcGIS.Analyst3D.IMarker3DSymbol = New
ESRI.ArcGIS.Analyst3D.Marker3DSymbol()
        p.CreateFromFile(marker3DSymbolfile)

        Dim pFeatureLyr As ESRI.ArcGIS.Carto.IFeatureLayer =
AxSceneControl1.Scene.Layer(1)
        Dim pGeoFeatureLyr As ESRI.ArcGIS.Carto.IGeoFeatureLayer =
pFeatureLyr
        Dim pRender As ESRI.ArcGIS.Carto.ISimpleRenderer =
pGeoFeatureLyr.Renderer
```

```
        pRender.Symbol = p
        pGeoFeatureLyr.Renderer = pRender
        AxSceneControl1.SceneGraph.RefreshViewers()
    End Sub
```

2）运行程序，其结果如下图所示：

