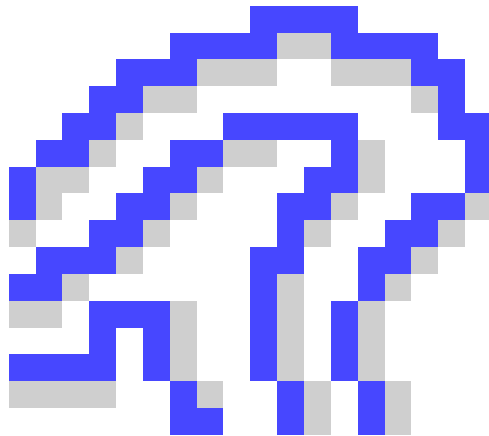


ContourOCX 5.52

User's Manual



Sep. 7, 2007

Introduction: This OCX Component can create contours and polygons from a series of points and convert them to SHP format files. It also can cut or mask contours or polygons in order to show only the part in which you interest. TIN with customized borders can be generated, too.

Attention:

*When the number of points exceeds 20, the ABOUT Dialog will appear in the trial version.

Member Procedures of ContourOCX

I . The Contour Lines Section	1
1. To Create Contour Lines from Regularly Distributed Points	1
1.1 Data Initiation Methods.....	1
1.2.to Calculate, Patch data and Create Contours	2
2.to Create Contour Lines from Scattered Points.....	3
2. 1 Initialization Methods	3
2.2 to Add Scattered Points	5
2.3 to Create Contour Lines From Scattered Points	5
3. to Get Attributes of Contour Lines.....	6
4.to Update Lines' Attributes	6
II the Contour Surface Section.....	7
1 to Create Contour surfaces	7
2. Contour Surface Attributes.....	8
3. the Manipulation of Surfaces	9
III. Conversion to SHP Format File	9
IV to Cut Contours and Polygons	10
1.Initialization	10
2.to Add Frames	10
3.to Start Cutting	11
4. Attributes of Contour or Polygon After Cutting.....	12
4.1 Attributes of cut contours	12
4.2 to Get Attributes of a Polygon After Cutting.....	12
5.to Draw After Cutting	13
V. to Mask Lines or Polygons Locally	14
1.to Start Masking	14
2.to Draw Locally Masked Lines or Polygons.....	15
VI to Generate Triangle Irregular Network (TIN).....	16
1 Ordinary TIN.....	16
2 TIN With Customized Borders.....	16
3 to Get Attributes of TIN	18
4 to Convert TIN to Contours	19
VII 2D Vector Field	19
1 to Create Stream Lines	19
2 Properties of the Vector Field.....	21
3 to Draw Vector Field by Stream Lines	22
4 to Mask Stream Lines.....	23
5 to Draw Vector Field by Arrows	23
VIII. to Free Memory.....	24
IX Other Procedures.....	25
1. to Relocate the Position of a Map On the Screen.....	25

2. to set the color class of a map	25
3 to Make polygons transparent	26
4 to Export the Interpolated Data to a Binary File	26
5 to Remove the Exception When Using the ContourOCX in Delphi	28
6 the Abnormal Color Change in VB After Masking	28
X Contact	29

I . The Contour Lines Section

1. To Create Contour Lines from Regularly Distributed Points

1.1 Data Initiation Methods

(1) `void Initial(long row, long col, double step, short InterposeValue, short RestrictInterposedValue);`

function: to initialize the memory

parameters:

row: the row number of data

col: the column number of data

step: the value of contour line interval

InterposeValue:

0: Do not interpose values

1: Interpose values

RestrictInterposedValue:

0: interposed values are not restricted.

1: interposed values are restricted.

(2) `void AddPoint(long row, long col, double x, double y, double value);`

function: Add a point

parameters:

row: the row index(starting from 0)

col: the column index(starting from 0)

x,y: point coordinates (real position of a point)

value: the value of a point (Z Value)

code example:

float row=10;

foat col=12;

```

float step=1;
CContourOCX m_sst;
.....
m_sst.Initial(row,col,step);
for(int i=0;i<row;i++)
{
    for(int j=0;j<col;j++)
    {
        float randomValue(rand()%10+rand()%10/10.0);
        m_sst.AddPoint(i,j,j*20,i*20,randomValue);
    }
}
}

```

(3) `void void AddCustomedStep(float x);`

Function: to draw lines with specified values. This function will disable the variable named *step* in [void Initial(long row, long col, double step)].

1.2.to Calculate, Patch data and Create Contours

There are three methods to create contours.

(1) `void Calculate(long Compensate, long CompensateValue);`

parameters:

Compensate: 1-to patch values ; 0-do not patch values

CompensateValue: the value to be patched

For example:

m_ssst.Calculate(1, 9999); //the point whose value is not less than 9999 will be patched.

(2) `void Calculate2(long Compensate, long CompensateValue, float PointStep);`

parameters:

Compensate: 1-do patch points; 0-do not patch points

CompensateValue: *the point whose value is not less than 9999 will be patched.*

PointStep: lines will be smoother when the value is small. In general, the value can be set to 1.

(3) `void Calculate3(long Compensate, long CompensateValue, int detail);`

parameters:

Compensate: 1-do patch points; 0-do not patch points
 CompensateValue: the value not less than which be patched
 detail: the lines will be smoother if the value is large. However, it will
 take more time to complete the calculation.

2.to Create Contour Lines from Scattered Points

2.1 Initialization Methods

- (1) `void InitialRandomIIDW(long insertLineCount, long nearPointCount, float Step, long Smooth);`

Function: Initialization of interpolation method named IDW(Inverse Distance Weighted).Search nearest points until the appointed number of points is reached.

Parameters: insertLineCount: the amount of interpolated lines larger than 0. if given -1 the default value(100) will be used to interpolate values

NearPointCount: the amount of nearest points to search. Larger than 0. if given -1 the default value will be used to search nearest points.

Step: the value of contour line interval

Smooth: the lines smooth parameter, which is larger than 0, if given -1 the default value will be used to make lines smooth.

For example:

ContourOCX1.InitialRandomIIDW -1, -1, 1, -1

- (2) `void InitialRandomIDW2(long insertLineCount, float distance, float DistPowerExponent, float Step, long smooth, long AtLeastOnePointFound, float ValueWhenNoFound);`

function: Initialization of interpolation method named IDW(Inverse Distance Weighted).Search nearest points within a appointed distance(radius)

parameters:

insertLineCount: the amount of interpolated lines larger than 0. if given -1 the default value(100) will be used to interpolate values

distance : the maximum distance to search for.

DistPowerExponent: the power that distance raised to.

Step: the value of contour line interval

Smooth: the line smooth parameter, which is larger than 0, if given -1 the default value will be used to make lines smooth.

AtLeastOnePointFound: Go on searching until at least one point is found if there is no point within the appointed distance. 1-Yes.0-No

ValueWhenNoFound: the value to fill in if there is no neighbor point found.

For example:

ContourOCX1.InitialRandomIDW2 -1, 150, 2, linestep, Smooth, 0, 0

(3)void InitialRandomCFWAI(long insertLineCount, float Step, long Smooth);

Function : Initialization of interpolation method named CFWAI(Core Function Whole Area Interpolation).

Parameters:

insertLineCount: the number of interpolated lines larger than 0. if given -1 the default value(100) will be used to interpolate values

Step: the value of contour lines interval

Smooth: line-smoothing parameter .larger than 0, if given -1 the default value will be used to make lines smooth.

For example:

ContourOCX1.InitialRandomCFWAI -1, 1, -1

(4) void InitialRandomCFPAI(long insertLineCount, long nearPointCount, float Step, long Smooth, float Power);

Function : Initialization of interpolation method named CFPai(Core Function Part Area Interpolation).

Parameters:

insertLineCount: the number of interpolated lines larger than 0. if given -1 the default value(100) will be used to interpolate values

nearPointCount: the number of the nearest points to search for. It is larger than 0. if given -1 the default value will be used to search for the nearest points.

Step: the value of contour lines interval

Smooth: line-smoothing parameter .larger than 0, if given -1 the default value will be used to make lines smooth.

Power: the power that distance raised to

For example:

ContourOCX1.InitialRandomCFPAI -1, 30, 1, -1, -1

(5)void InitialRandomKrigingOK(long insertLineCount, long nearPointCount, float Step, long Smooth, long ReservedValue);;

Function : Initialization of interpolation method named Ordinary Kriging (OK).Linear model is applied and nugget is 0.

Parameters:

insertLineCount: the number of interpolated lines. larger than 0. If given -1 the default value(100) will be used to interpolate lines.

NearPointCount: the number of the nearest points to search for. It is larger than 0. if given -1 the default value will be used to search for the nearest points.

Step: the value of contour lines interval

Smooth: the line-smoothing parameter, which is larger than 0. If given -1 the default value will be used to make lines smooth.

ReservedValue: Reserved value

For example:

ContourOCX1.InitialRandomKrigingOK -1, 30, 1, 2, -1

(6) `void InitialRandomNN(float Step, long Smooth, long para1);`

Function: Initialization of interpolation method named Nature Neighbor.◦

Parameters:

Step: the value of contour lines interval

Smooth: line-smoothing parameter . larger than 0, if given - 1 the default

para1: unused.

For example:

ContourOCX1.InitialRandomNN 1.0, 2, 1

2.2 to Add Scattered Points

There are two ways to add scattered points

(1) `void AddPointRandom(double x, double y, double value);`

Function: to add points one by one

Parameters:

x,y,Value: the position and Z value of a point

(2) `long AddRandomPointsFromFile(BSTR* path);`

Function: to import data from a file directly

Parameter:

path: The path of the data file that is in TXT format. ◦ Values of one point should be separated by character(s) such as a blank or ‘,’.For example:

x1 y1 value1		x1 ,y1,value1		x1, y1 value1
x2 y2 value2		x2, y2, value2		x2 y2,, value2
.....	or	or
xn yn valuen		xn, yn, valuen		xn yn valuen

2.3 to Create Contour Lines From Scattered Points

`void CalculateRandom();`

Function : to create contours via scattered points.

3. to Get Attributes of Contour Lines

(1) `void GetLineCount (long* LineCount);`

Function: to get the number of contour lines

parameters:

* LineCount: the line number

(2) `void GetLineValue(long LineIndex, float* value);`

Function : to get the value of a contour line

parameters:

LineIndex: the line index

*value: the line value

(3) `void GetCtrlPointCount(long LineIndex, long*PointCount);`

Function : to get control point number in a contour line

parameters:

LineIndex: the line index

*PointCountt: the control point number

(4) `void GetCtrlPoint(long LineIndexi, long PointIndex, double*X, double*Y, double*Value);`

Function: to get attributes of a point

parameters:

LineIndex: the line index

PointIndex: the control point index

*X, *Y, *Value: coordinates and the value of a line

4.to Update Lines' Attributes

(1) `void SetLineSize(long LineIndex, short LineSize);`

function: to specify the width of the line

parameters:

LineIndex: the line index

LineSize: the width of the line.

(2) `void SetLineColor(long LineIndex, long LineColor);`

function: to specify the color of the line

parameters:

LineIndex: the line index

LineColor: the color of the line.

(3) `void ResetContoursColor();`

function: to specify the color of all lines

parameter(s): none

For example(in VB):

```
' change color from red to green, then from green to blue
ContourOCX1.ClearColorClass

ContourOCX1.AddNewElementToColorClass RGB(255, 0, 0)
ContourOCX1.AddNewElementToColorClass RGB(0, 255, 0)
ContourOCX1.AddNewElementToColorClass RGB(0, 0, 255)
ContourOCX1.AddNewElementToColorClass RGB(255, 0, 255)
' ContourOCX1.SetDefaultZValueRange 1, 10
ContourOCX1.ResetContoursColor
```

(4) `void ResetContourPosition();`

function: to reset the position of contours

parameter(s): none

For Example:

```
ContourOCX1.SetDefaultPositionValus Picture1.Width, Picture1.Height, 0, 500, 0,
500
ContourOCX1.ResetContourPosition
ContourOCX1.DrawContours Picture1.hDC
```

(5) `void DrawContours(long dc);`

Function: to draw all contours to a device context handle.

parameter:

dc: the device context handle

II the Contour Surface Section

Before creating contour surfaces, please ensure that contour lines were already created..

1 to Create Contour surfaces

(1) `void ConvertToPolygon(long* Succeed);`

function : to create contour surfaces

parameter:

Succeed: If the procedure fails , it is equal or less to 0. Otherwise , it is larger than 0.

2. Contour Surface Attributes

(1) `void GetPolygonCount(long* PolygonCount);`

function : to get the number of contour surfaces

parameter:

*PolygonCount : the number of contour surfaces

(2) `void GetPolygonPointCountValueArea(long index, long* PointCount, float* MinValue, float* MaxValue, float* AreaWithoutHoles, float* AreaWithHoles);`

function: to get control point amount in a polygon, as well as value range (MinValue, MaxValue) and area(AreaWithoutHoles, AreaWithHoles);

parameters:

index: the polygon index

*PointCount: the the number of control points in the edge of a polygon

*MinValue、maxValue: the value range

AreaWithoutHoles、AreaWithHoles: areas of a polygon.

`void GetPolygonPointXY(long polygonIndex, long pointIndex, float* X, float* Y);`

function: to get the coordinates of a control point

parameters:

polygonIndex: polygon index

pointIndex: index of a control point

* X, * Y: coordinates of a control point

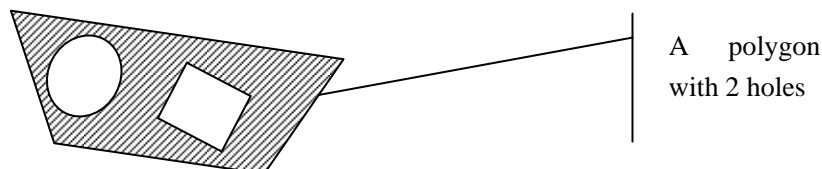
(2) `void GetPolygonHoleCount(long polygonIndex, long* holeCount);`

function: to get the number of holes in a polygon

parameters:

polygonIndex: the polygon index

*holeCount: the number of holes



(3) `void GetPolygonIndexFromHoleIndex(long polygonIndex, long holeIndex, long* PolygonIndexInList);`

function : to get the polygon index of a hole. In fact, holes are polygons ,too.

parameters:

polygonIndex: the polygon index

holeIndex: the hole index

* PolygonIndexInList: as a polygon, the index of the hole

3. the Manipulation of Surfaces

(1)void ResetPolyPostion();

function: to convert real coordinates to screen coordinates .The Real coordinates will not be changed after the procedure is called.

(2)void ResetOnePolygonColor(long polygonIndex, long fillColor, long borderColor);

Function: to set one polygon's color

Parameters:

 polygonIndex: the polygon index

 fillColor: the color to fill the polygon

 borderColor: the color of the polygon border

(3)void ResetPolygonsColor(long BorderColor);

function: to specify the color of all polygons

parameter(s) :

 BorderColor : the color of the border line.

For example(in VB):

```
-----  
' change color from red to green, then from green to blue  
ContourOCX1.ClearColorClass  
ContourOCX1.AddNewElementToColorClass RGB(255, 0, 0)  
ContourOCX1.AddNewElementToColorClass RGB(0, 255, 0)  
ContourOCX1.AddNewElementToColorClass RGB(0, 0, 255)  
ContourOCX1.AddNewElementToColorClass RGB(255, 0, 255)  
' ContourOCX1.SetDefaultZValueRange 1, 10  
ContourOCX1.ResetPolygonsColor RGB(255, 255, 0)  
-----
```

(4)void DrawAllPolygons(long dc);

Function: to draw all polygons to a device context handle。

parameter:

dc: a device context handle

III. Conversion to SHP Format File

(1) void InitializeSHPFile(BSTR* Path, BSTR* SHPType);

Function: : to initialize the Shape file path and shape type.

Path: the path to save a SHP file;

SHPTYPE: the Shape type. There are two constant strings, "line" and "polygon", can be chosen. to create SHP file: .

(2) `void AddAppendedField(BSTR* ShpType,BSTR*FieldName,BSTR*FieldType,long CharLength);`

Function : to add additional fields to the DBF file .

SHPTYPE: the Shape type. There are two constant strings, "line" and "polygon", can be chosen :

FieldName: the field name.

FieldType: the data type of the field. There are 3 types of data can be used for the field, including "integer","float" and "string"

CharLength: the length of the datum in the field;

For the string type field, it equals to the length of the string.

For the integer type field, it equals to number of the digitals in the value

For the float type field, it equals to the total number of digitals in the integral part and the decimal part;

(3)`void CreateShapeFile();`

To Create a Shape File.

TIP: As a auxiliary file accompanying a SHP file, DBF file has following fields if the procedure AddAppendedField was not called .

Field I: MinValue: float ,the minimum value of a line or polygon

Field II: MaxValue: float ,the maximum value of a line or polygon. When the shape type is line,the MinValue is equal MaxValue.

Field III: IDNo: integer,the index of a line or polygon

IV to Cut Contours and Polygons

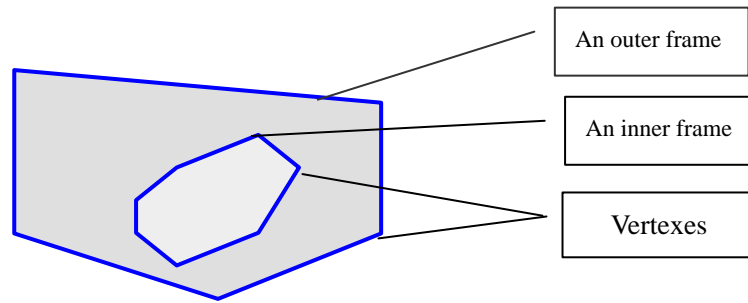
1.Initialization

`void InitializeForCut();`

to initialize data before cutting

2.to Add Frames

There are two types of frame: the inner frame (which is called hole) and the outer frame(which is called border)。



(1) void AddNewFrame(BSTR* BorderOrHole);

function: Add a new frame

parameter:

* BorderOrHole: border type ("border" or "hole")

(2) void AddFrmCtrlPoint(double x, double y);

功能: to add a new vertex to the frame

parameter:

x,y: the position of a vertex

For example(VB) :

'circle border

s = "border"

Dim i As Integer

Dim angle As Double

ContourOCX1.AddNewFrame s

For i = 1 To 71

angle = 3.1415926 * 2 / 72 * i

ContourOCX1.AddFrmCtrlPoint 110 + Cos(angle) * 70, 120 + Sin(angle) * 100

Next

'add a hole in the circle area

s = "hole"

ContourOCX1.AddNewFrame s

For i = 1 To 71

angle = 3.1415926 * 2 / 72 * i

ContourOCX1.AddFrmCtrlPoint 120 + Cos(angle) * 50, 120 + Sin(angle) * 50

Next

3.to Start Cutting

To cut current contours or polygons

(1) void CutLinesAlongFrame();

function: to cut contours

parameters: none

(2) `void CutPolygonAlongFrame();`

function: to cut polygons

parameters: none

4. Attributes of Contour or Polygon After Cutting

4.1 Attributes of cut contours

(1) `void GetLineCountAfterCut(long* LineCount);`

function: to get the number of contours after cutting

parameter(s):

*LineCount: the contour number after cutting

(2) `void GetCtrlPntCountAndValueAfterCut(long LineIndex, long* PointCount, double* Value);`

function: to get the control point number and contour value after cutting

parameters:

LineIndex: the contour index

*PointCount: the control point number

*Value: the contour value

(3) `void GetCtrlPointAfterCut(long LineIndex, long PointIndex, double* x, double* y);`

function: to get attributes of a point after cutting

parameters:

LineIndex: the contour index

PointIndex: the control point index

*X, *Y, *Value: the attributes of a point

4.2 to Get Attributes of a Polygon After Cutting

(1) `void GetPolygonCountAfterCut(long* PolygonCount);`

function: to get the polygon number after cutting

parameter:

*PolygonCount: the polygon number after cutting

(2) `void GetPolygonPntCountValueAfterCut (long polyIndex, long* PointCount, float* MinValue, float* MaxValue);`

function: to get the polygon number and value range (MinValue, MaxValue) after cutting

parameters:

index: the polygon index
*PointCount: the vertex number
*MinValue、maxValue: the value range

(3) `void GetPolygonPointXYAfterCut(long polygonIndex, long pointIndex, float * X, float * Y);`

function: to get vertex attributes of a polygon after cutting

parameters:

polygonIndex: the polygon index
pointIndex: the vertex index in a polygon
* X, * Y: the position of the vertex

5.to Draw After Cutting

(1)`void ResetContourPositionAfterCut();`

function: to reset the position of contours after cutting

parameter(s): none

(2)`void SetLineColorAfterCut(long LineIndex, long LineColor);`

function: to specify the color of the line after cutting

parameters:

LineIndex: the line index
LineColor: the color of the line.

(3)`void ResetContoursColorAfterCut();`

function: to specify the color of all lines

parameter(s): none

(4) `void DrawContoursAfterCut(long dc);`

function: to draw all contours to a device context handle after cutting

parameter:

dc: a device context handle

(5) `void ResetPolyPostionAfterCut();`

function: to convert real coordinates to screen coordinates .The Real coordinates will not change after the calling

Parameters: none

(6) `void ResetOnePolygonColorAfterCut(long polygonIndex, long fillColor, long borderColor);`

Function: to set polygon color after cutting

Parameters:

 polygonIndex: the polygon index

 fillColor: the color to fill the polygon

 borderColor: the color of the polygon border

(7) void ResetPolygonsColorAfterCut(long BorderColor);

 function: to specify the color of all polygons

 parameter(s) :

 BorderColor : the color of the border line.

(8) void DrawAllPolygonsAfterCut(long dc);

 function: to draw all polygons to a device context handle after cutting

 parameter:

 dc: a device context handle

V. to Mask Lines or Polygons Locally

1.to Start Masking

(1)void InitializeForMask();

 function: to initialize the data before masking

 parameters: none

(2)void MaskLinesByFrame(long dc,long width,long height,float x1,float x2,float y1,float y2);

 function: to mask contours

 parameters:

 dc: the device context for the specified window's client area.

 width,height: pixel numbers of the width and height of window's client area

 x1, x2, y1, y2: x1,x2,y1,y2: the minimum and maximum of the

x-coordinate and y-coordinate in the window's client area

(3)void MaskPolygonsByFrame(long dc,long width,long height,float x1,float x2,float y1,float y2);

 function: to mask polygons

 parameters:

 dc: the device context for the specified window's client area.

 width,height: pixel numbers of the width and height of window's client area

 x1, x2, y1, y2: x1,x2,y1,y2: the minimum and maximum of the

x-coordinate and y-coordinate in the window's client area

For example (in VB)

```
-----  
ContourOCX1.InitializeForMask  
Call addFrames 'add frames  
Dim lineCount As Long, pointCount As Long, Value As Single, color As Variant  
ContourOCX1.GetLineCount lineCount  
...  
ContourOCX1.MaskLinesByFrame Picture2.hDC, Picture2.Width, Picture2.Height,  
10, 500, 0, 500  
ContourOCX1.DrawMaskedLinesOrPolygons Picture2.hDC  
-----
```

2.to Draw Locally Masked Lines or Polygons

(1) `void DrawMaskedLines(long dc);`

function: to draw partly masked contours

parameter:

dc: the device context for the specified window's client area.

(2) `void DrawMaskedPolygons(long dc);`

function: to draw partly masked polygons

parameter:

dc: the device context for the specified window's client area.

(3) `void DrawMaskedLinesOrPolygons(long dc);`

function: to draw current masked contours or polygons

parameter:

dc: the device context for the specified window's client area.

For example:

VB:

```
Picture2.Cls  
ContourOCX1.DrawMaskedLinesOrPolygons Picture2.hDC
```

VC:

```
HDC dc=: : GetDC(m_Picture2.m_hWnd );  
ContourOCX1.DrawMaskedLinesOrPolygons((long)dc);  
: : ReleaseDC(m_Picture2.m_hWnd,dc);/release dc
```

Delphi:

```
dc:=GetDC(Picture2.Handle);  
ContourOCX1.DrawMaskedLinesOrPolygons(dc);  
ReleaseDC(Picture2.Handle,dc);
```

VI to Generate Triangle Irregular Network (TIN)

1 Ordinary TIN

```
(1) void InitialRandom (float Step) ;
```

Function: Initialization

parameter:

step: conserved value

```
(2) void DrawTIN(long dc);
```

function: to draw TIN

parameter:

dc: the device context for the specified window's client area.

For example:

VB:

```
ContourOCX1.InitialRandomTIN 1  
Call AddRandomPoint  
ContourOCX1.CalculateRandom  
dc = GetDC(Picture1.hwnd)  
ContourOCX1.DrawTIN dc  
ReleaseDC Picture1.hwnd, dc
```

2 TIN With Customized Borders

```
(1) void InitialRandomTINconstrain(long deleteWhenOutOfBorder);
```

function : Initialization for TIN with constraint

parameter:

deleteWhenOutOfBorder : if the triangle is out of the border ,it should be deleted or not.

1 — to be deleted

0 — to be conserved

(2) There are 2 methods to add Customized borders

a.

```
void AddFromCtrlPointForTIN(double x, double y, double value);
```

function: to insert points one by one

parameters:

x,y,value: the position and value of a point

For example:

VB:

```
-----  
ContourOCX1.InitializeForMask  
Dim s As String  
s = "border"  
ContourOCX1.AddNewFrame s  
ContourOCX1.AddFrmCtrlPointForTIN 120, -40, 2  
ContourOCX1.AddFrmCtrlPointForTIN 128, -30, 3  
ContourOCX1.AddFrmCtrlPointForTIN 140, -41, 4
```

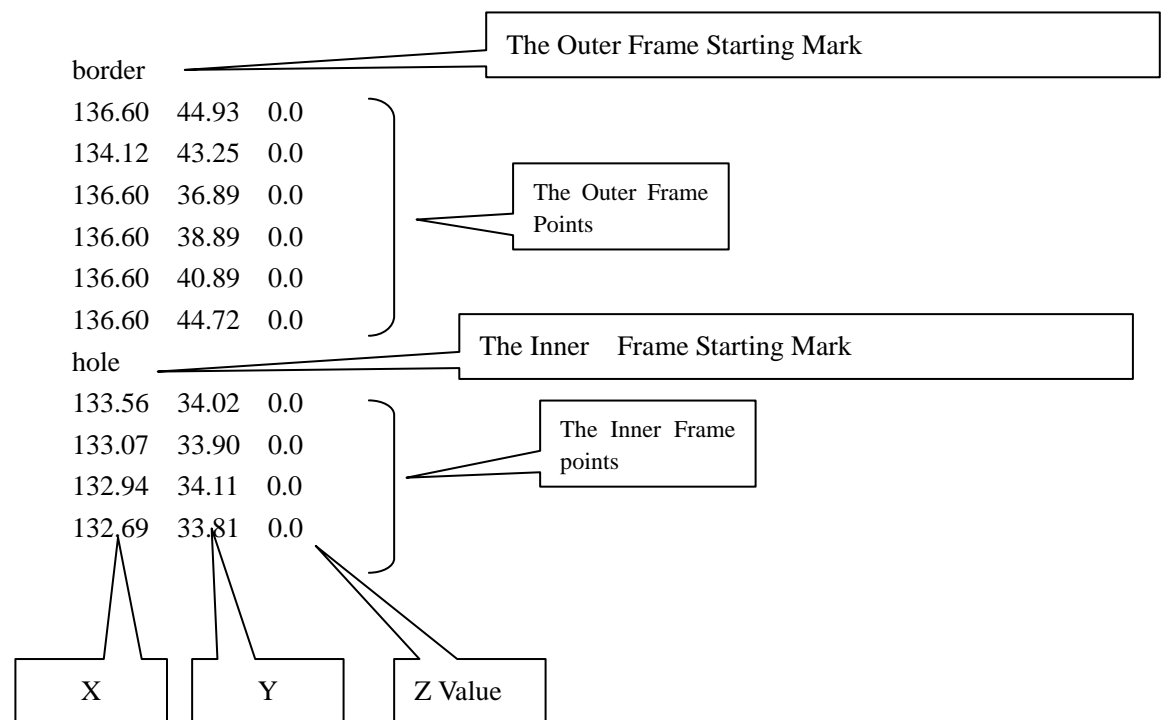
b. `void AddFrmCtrlPointFromTXTFile(BSTR path);`

function: to add frame points from a TXT file

parameter:

path: file path

The file format is like following :



For example:

```
-----  
path = App.path + "\\Australia.txt"  
ContourOCX1.AddFrmCtrlPointFromTXTFile path
```

3 to Get Attributes of TIN

```
(1) void GetTriangleCount(long* count);
```

function: to get triangle number in the TIN

parameter:

count: the number of triangles

```
(2) void GetTriangleEdgeCount(long* EdgeCount);
```

function: to Get Edge number in the TIN

parameter:

EdgeCount: edge count

```
(3) void GetTrianglePoint(long TriangleIndex, long PointIndex, float* x, float* y, float* value);
```

function : to get one vertex of a triangle

parameters:

TriangleIndex: the index of the triangle in the TIN

PointIndex: the vertex index of the triangle

x,y,value: the position and value of a vertex

```
(4) void GetTriangleEdges(long TriangleIndex, long* EdgeIndex1, long* EdgeIndex2, long* EdgeIndex3);
```

function: to get 3 indexes of edges on a triangle

parameters:

TriangleIndex: the index of a triangle

EdgeIndex1, EdgeIndex2, EdgeIndex3: 3 indexes of edges on a triangle

```
(5) void GetTriangleEdgeEndPoints(long EdgeIndex, float* x1, float* y1, float* x2, float* y2);
```

function: to get 2 endpoints of an edge

parameters: EdgeIndex: the index of an edge

x1,y1: an endpoint

x2,y2: the other endpoint

```
(6) void GetTriangleNeighboursViaEdge(long EdgeIndex, long* TriangleIndex1, long* TriangleIndex2);
```

function: to get two indexes of triangles sharing the same edge

parameters: EdgeIndex: the index of an edge

TriangleIndex1, TriangleIndex2: indexes of triangles

```
(7) void GetTriangleNeighborsViaSelf(long SelfIndex, long* TriangleIndex1, long* TriangleIndex2, long* TriangleIndex3);
```

function: to get three neighbors of a triangle

parameters: SelfIndex: the index of a triangle

TriangleIndex1, TriangleIndex2, TriangleIndex3: 3 triangle indexes

4 to Convert TIN to Contours

```
void TIN2Contours(float SimpleStep, BSTR CustomizeSteps);
```

function: to Convert TIN to Contours

parameters: SimpleStep: the fixed line interval .

CustomizeSteps: the customized line interval .

VII 2D Vector Field

1 to Create Stream Lines

```
( 1 ) void InitialVector2D(long insertLineCount, long nearPointCount, float  
IDW2Dist,float DeltaH, long RKType, long SpeadLineDensity,long  
DistBetweenLines,long DistBetweenArrows,BSTR InterpolateMethod);
```

Function: Initialization before generating stream lines

Parameters:

insertLineCount: the amount of interpolated lines larger than 0. if given -1 the default value(100) will be used to interpolate values

NearPointCount: the amount of nearest points to search. Larger than 0. If given -1 the default value will be used to search nearest points.

IDW2Dist: the maximum distance to search for. Only be available when InterpolateMethod is "IDW2".

DeltaH: -1 is recommended.

RKType: reserved.

SpeadLineDensity: the density of stream lines.

DistBetweenLines: the distance between lines

DistBetweenArrows: the distance . Not less than 0.

InterpolateMethod: the method of Interpolation . There are 6 methods,IDW1,IDW2,Kriging,CFWAI,CFPAI and NN.

For example:

```
ContourOCX1.InitialVector2D -1, 30, 20, -1, -1, 30, 1, 60,"kiriging"
```

```
(2) void AddCustomedVectorLnSeed(float x, float y);
```

Function: to create custom-made vector lines.

Parameters : x,y: the position of vector line' seed

For example(in VB):

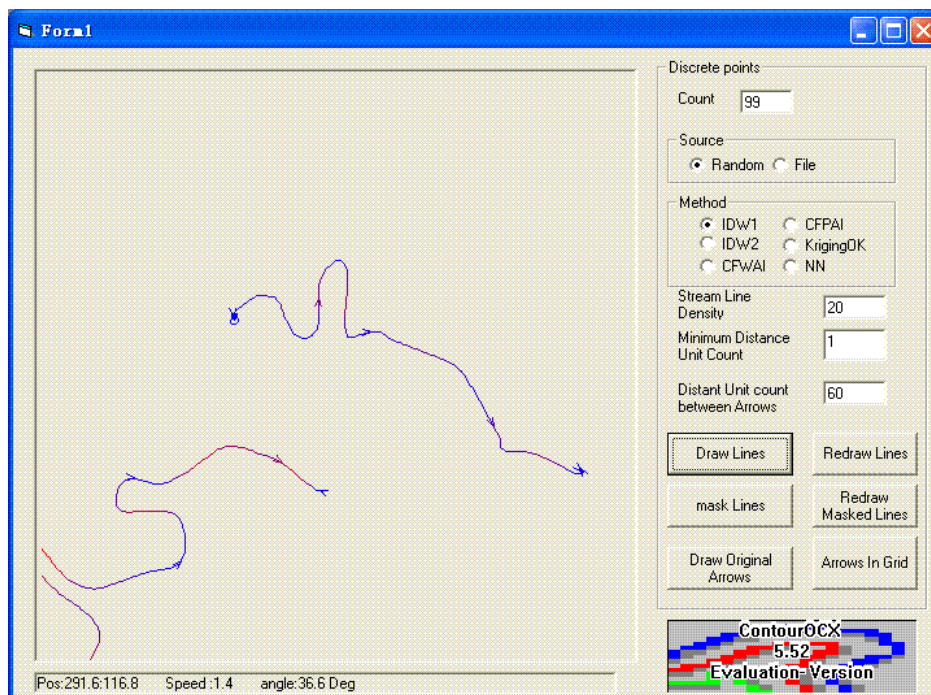
```

ContourOCX1.InitialVector2D -1, 30, 20, -1, -1, CInt(LineDensityEdit.Text),
CInt(distBetweenLinesEdit.Text), CInt(DistBtwnArrowsEdit.Text), m
ContourOCX1.AddCustomedVectorLnSeed 30, 20
ContourOCX1.AddCustomedVectorLnSeed 40, 80
ContourOCX1.AddCustomedVectorLnSeed 140, 180
Dim i As Integer
    Dim X As Double, Y As Double, u As Double, v As Double
If DataSourceIndex = 0 Then
    Call AddRandomPoints
Else
    Dim path As String
    'path = App.path + "\"vectorPoint.txt"
    path = App.path + "\"out3.txt"

    Dim count As Integer
    count = ContourOCX1.AddRandomVectorPointsFromFile(path)
End If
    ContourOCX1.CalculateVector
Call SetStreamLinesColor
ContourOCX1.ResetVector2DLinePosition
ContourOCX1.DrawVectorLines Picture1.hdc

```

The image with 3 vector lines is:



```
(3) void AddVectorPointRandom(double x, double y, double uValue, double vValue);
```

Function: to add points one by one

Parameters:

x,y: the position of a vector point

u,v: vector(such as the speed)'s magnitudes in X and Y directions

```
(4) long AddRandomVectorPointsFromFile(BSTR* path);
```

Function: to import data from a file directly

Parameters:

path: The path of the data file that is in TXT format. 。 Values of one point should be separated by character(s) such as a blank or ‘,’.For example:

x1 y1 u1 v1	x1, y1, u1, v1
x2 y2 u2 v2	x2 ,y2 ,u2 ,v2
.....	或
xn yn un,vn	xn ,yn ,un,,vn

The number of imported points will be returned by the function.

```
(5) void CalculateVector();
```

Function: to calculate to generate stream lines

Parameters: None

2 Properties of the Vector Field

```
(1) void GetVectorLineCount(long* lineCount);
```

Function: to get the number of stream lines

Parameters: lineCount: the number of stream lines

```
(2) void GetVectorLinePointCount(long lineIndex, long* pointCount);
```

Function: to get the number of control points in a stream line

Parameters: lineIndex: the index of the stream line

pointCount: the number of control points

```
(3) void GetVectorLinePointPosAndValue(long lineIndex, long pointIndex, float* x, float* y, float* u, float* v);
```

Function: to get properties of a control point

Parameters:

lineIndex: the index of a stream line

pointIndex: the index of a control point

x,y: position

u,v: vector(such as the speed)'s magnitudes in X and Y directions

```
(4) void GetVectorPointValues(float x, float y, float* u, float* v, float* angle);
```

Function: to get vector properties at a certain point

Parameters:

x,y: position

u,v: vector(such as the speed)'s magnitudes in X and Y directions

angle: the direction of the vector

3 to Draw Vector Field by Stream Lines

```
(1) void ResetVector2DLinePosition();
```

Function: to set stream lines' position .It is should be called after the procedure named SetDefaultPositionValus.

Parameter: none

```
(2) void ResetVectorLinesColor();
```

Function: to set stream lines' color.

Parameter: None

```
(3) void DrawVectorLines(long dc);
```

Function: to draw stream lines

Parameter:

dc: the device context handle

For example:

.....

ContourOCX1.CalculateVector

ContourOCX1.ClearColorClass

ContourOCX1.AddNewElementToColorClass RGB(0, 0, 255)

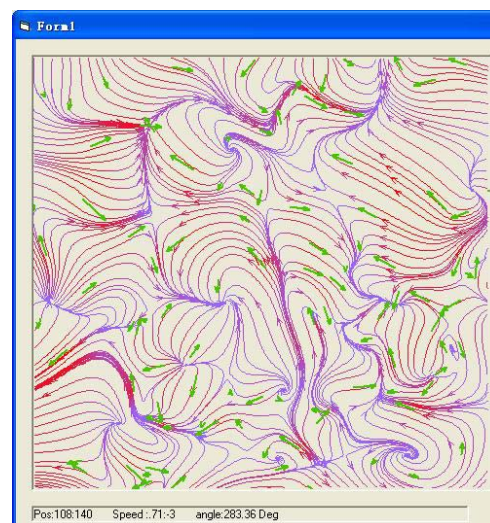
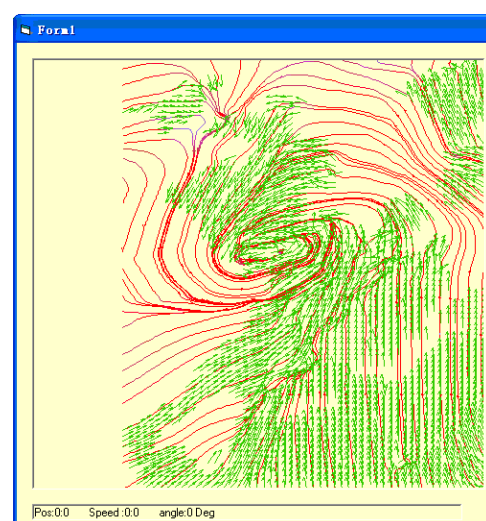
ContourOCX1.AddNewElementToColorClass RGB(0, 255, 0)

ContourOCX1.AddNewElementToColorClass RGB(255, 0, 0)

ContourOCX1.ResetVectorLinesColor

ContourOCX1.DrawVectorLines Picture1.hdc

.....



4 to Mask Stream Lines

```
(1) void InitializeForMaskVectorLines();
```

Function: the initialization before masking stream lines

Parameters: None

```
(2) void MaskVectorLinesByFrame(long dc);
```

Function: to mask stream lines partly

Function:

dc: the device context handle

```
(3) void DrawMaskedVectorLines(long dc);
```

Function: to draw stream lines after masking

Parameter:

dc: the device context handle

For exaple:

```
-----  
ContourOCX1.InitializeForMaskVectorLines  
Dim s As String  
s = "border"  
ContourOCX1.AddNewFrame s  
ContourOCX1.AddFrmCtrlPoint 130, 21  
ContourOCX1.AddFrmCtrlPoint 130, 35  
ContourOCX1.AddFrmCtrlPoint 121, 30  
Dim dc As Long  
dc = GetDC(Picture1.hwnd)  
ContourOCX1.MaskVectorLinesByFrame dc  
ContourOCX1.DrawMaskedVectorLines dc  
ReleaseDC Picture1.hwnd, dc  
-----
```

5 to Draw Vector Field by Arrows

```
(1) void SetArrowLengthRange(long minLength, long maxLength);
```

Function: the length range of arrows

Parameters: minLength, maxLength: the minimum and maximum length of arrows

For example:

```
-----  
ContourOCX1.SetArrowLengthRange 1, 52  
-----
```

```
(2) void SetArrowSize(long size);
```

Function: to set the size of arrows

Parameters: size: the size of arrows

```
(3) void SetArrowColor(long color);
```

Function: to set the color of all arrows

Parameters: color: the color of arrows

```
(4) void DrawOneArrow(long hdc, float x, float y, long color, long size, long length);
```

Function: to draw an arrow

Parameters:

dc: the device context handle

x,y: position

color: the color of the arrow

size: the size of the arrow

length: the length of the arrow.

For Example:

```
-----  
ContourOCX1.DrawOneArrow Picture1.hdc, j * 20, i * 20, RGB(128, 128, 128), 1, -1  
-----
```

```
(5) void DrawOriginalVectors(long hdc);
```

Function: to draw original points by arrows

Parameter:

dc: the device context handle

For example:

```
-----  
ContourOCX1.SetArrowColor RGB(255, 0, 0)  
ContourOCX1.SetArrowLengthRange 1, 20  
ContourOCX1.DrawOriginalVectors Picture1.hdc  
-----
```

VIII. to Free Memory

```
void FreeData();
```

function: to release memory

this procedure should be called before closing the application in order to RELEASE memory.

IX Other Procedures

1. to Relocate the Position of a Map On the Screen

(1) `void SetDefaultPositionValus(long width, long height, float x1, float x2, float y1, float y2);`

function: to fit contours or polygons to the client area

parameters:

width,height: the width and height of the window's client area.

x1,x2,y1,y2: the minimum and maximum of the x-coordinate and y-coordinate in the window's client area

(2) `void GetScrPosFromRealPos(float RealX, float RealY, long* ScrX, long* ScrY);`

function: to get coordinates on the screen from coordinates on the map.

Parameters:

RealX, RealY: coordinates on the map

ScrX,ScrY: coordinates on the screen

(3) `void GetRealPosFromScrPos(long ScrX, long ScrY, float* RealX, float* RealY);`

function: to get coordinates on the map from coordinates on the screen.

Parameters:

ScrX,ScrY: coordinates on the screen

RealX, RealY: coordinates on the map

2. to set the color class of a map

(1) `void ClearColorClass();`

function: to clear color class

Parameters: None

(2) `void SetDefaultZValueRange(float MinValue, float MaxValue);`

function: to specify the minimum and maximum of Z value

parameters:

MinValue, MaxValue: the minimum and maxmum Z value

(3) `void AddNewElementToColorClass(long ColorElement);`

function: to add a new color to the color list

parameters:

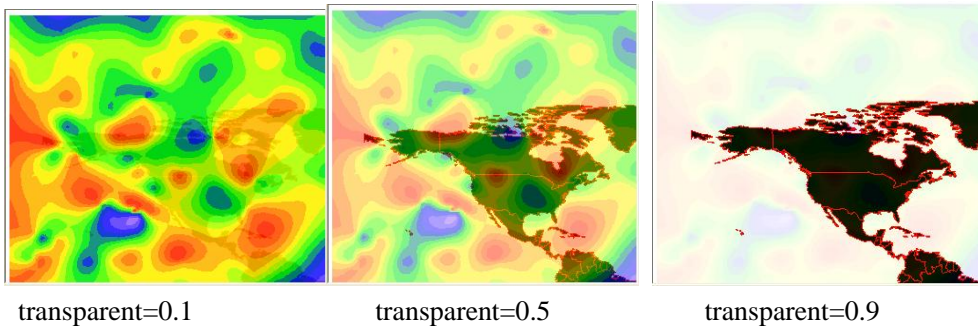
ColorElement: a new color element
 (4) `long GetValueColorInColorClass(float value);`
 function : to return a color
 parameters:
 value: a Z value

3 to Make polygons transparent

`void SetPolygonsTransparent(float transparent);`
 function: . to Make polygons transparent
 parameters:
 transparent: the coefficient of polygon transparency

For example:

```
-----
ContourOCX1.SetPolygonsTransparent 0.5
Dim dc As Long
dc = GetDC(Picture2.hwnd)
Picture2.Cls
ContourOCX1.DrawAllPolygons dc
ReleaseDC Picture2.hwnd, dc
-----
```



4 to Export the Interpolated Data to a Binary File

`void SetOutputDirectionToDataFile(BSTR Path, float XStart, float XEnd, float XInterval, float YStart, float YEnd, float YInterval, long DataAndContour);`
 function: to export interpolated data
 parameters:
 Path: the data file path
 XStart, XEnd, YStart, YEnd : the coordinates range in X and Y direction
 XInterval, Yinterval: point steps in x direction y direction
 DataAndContour: if it is equal to 1 , both data file and contours are generated.

Otherwise , only data file is created

For example:

ContourOCX1.SetOutputDirectionToDataFile "d: \xxx.xyz", 1, 100, 1, 1, 100, 1, 1

ContourOCX1.CalculateRandom

The Format of the data file to be exported is like:

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	
00000000h:	69	6E	74	65	72	70	00	00	64	00	00	00	00	00	00	00	interp..d...d...
00000010h:	00	00	80	3F	00	00	80	3F	00	00	80	3F	00	00	80	3F	..e?..e?..e?..e?
00000020h:	0D	01	00	00	DB	00	00	00	0C	01	00	00	DB	00	00	00?.....?..
00000030h:	0C	01	00	00	DC	00	00	00	0B	01	00	00	DC	00	00	00?.....?..
00000040h:	0A	01	00	00	DC	00	00	00	08	01	00	00	DB	00	00	00?.....?..
00000050h:	08	01	00	00	DB	00	00	00	08	01	00	00	DB	00	00	00?.....?..
00000060h:	08	01	00	00	DB	00	00	00	06	01	00	00	DA	00	00	00?.....?..
00000070h:	05	01	00	00	DA	00	00	00	05	01	00	00	D9	00	00	00?.....?..
00000080h:	04	01	00	00	D9	00	00	00	03	01	00	00	D8	00	00	00?.....?..
00000090h:	03	01	00	00	D8	00	00	00	02	01	00	00	D7	00	00	00?.....?..
000000a0h:	01	01	00	00	D7	00	00	00	01	01	00	00	D6	00	00	00?.....?..
000000b0h:	00	01	00	00	D5	00	00	00	FF	00	00	00	D5	00	00	00?.. ..?..
000000c0h:	FF	00	00	00	D4	00	00	00	FF	00	00	00	D3	00	00	00?.. ..?..
000000d0h:	FF	00	00	00	D3	00	00	00	FE	00	00	00	D2	00	00	00?..?..?..
000000e0h:	FE	00	00	00	D1	00	00	00	FE	00	00	00	D0	00	00	00	?..?..?..?..
000000f0h:	FE	00	00	00	D0	00	00	00	FE	00	00	00	CF	00	00	00	?..?..?..?..
00000100h:	EE	A6	E5	40	EE	A6	E5	40	EE	A6	E5	40	EE	A6	E5	40	香銻香銻香銻香銻
00000110h:	E6	9F	E5	40	42	95	E5	40	D6	86	E5	40	63	72	E5	40	鎢銻B嗎@請銻cr銻
00000120h:	F0	55	E5	40	05	2F	E5	40	D3	7A	E4	40	65	B6	E4	40	鐵銻./銻愈銻e朵@
00000130h:	9C	5E	E4	40	4D	F0	E3	40	2F	68	E3	40	19	C3	E2	40	衆銻M担@/h站.免@

After Installation of ContourOCX in version 5.23 or a version above it, there is a new class library named InterpolateDataBlock will be registered in the system automatically, via which the data file in binary format can be read .

For example:

```

Dim f As New InterpolateDataBlock
f.Read "d: \xxx.xyz"
Dim row As Long, col As Long
Dim x As Single, y As Single, z As Single
f.GetRowCol row, col
For i = 0 To row - 1
  For j = 0 To col - 1
    f.GetPoint i, j, x, y, z
    Debug.Print x, y, z
  Next
Next

```

5 to Remove the Exception When Using the ContourOCX in Delphi

If you are a Delphi developer, you might meet a floating point error like following.



Therefore , Some measures should be taken to remove the exception:

1.to declare a global variable Saved8087CW

```
var  
Saved8087CW: Word;
```

2.Before starting the application , (for example , in OnCreate()), save the default value of 8087CW, please

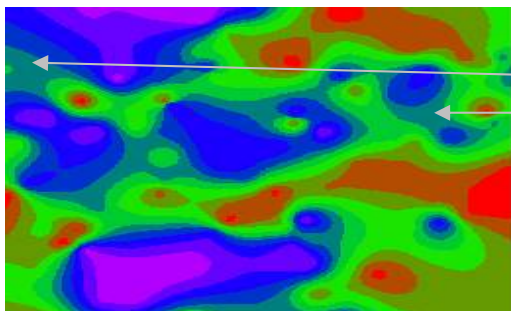
```
Saved8087CW := Default8087CW;  
Set8087CW($133f);
```

3.Before ending the application(for example, in OnClose) ,restore the default value of 8087CW, please

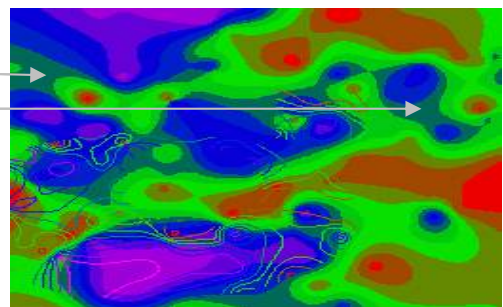
```
Set8087CW(Saved8087CW);
```

6 the Abnormal Color Change in VB After Masking

After mask operations on contours or surfaces, the color of the image may get bizarre.

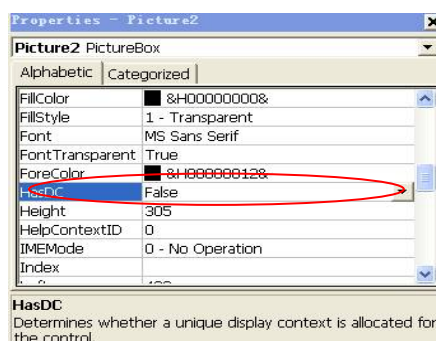


the color before masking



the color after masking

To solve the problem,first ,set HasDC of PictureBox properties to False



Second, to get PictureBox context by APIs. For example

For example:

'draw

Dim dc As Long

dc = GetDC(Picture2.hwnd)

ContourOCX1.MaskLinesByFrame dc

ContourOCX1.DrawMaskedLines dc

ReleaseDC Picture2.hwnd, dc

X Contact

If you are interested in this component or intend to buy it ,please send email to cui1012@sh163.net. or cuixuesen@eastfishery.ac.cn .Thank you.

Our website is <http://snowedforest.brinkster.net/>

Do not hesitate to send mail in case of problems please.