# ContourPolyline.cs

1. **using** System;
2. **using** System.Collections.Generic;
3. **using** System.Linq;
4. **using** System.Text;
5. **using** System.Threading.Tasks;
7. **namespace** AGIS\_work.DataStructure
8. {
9. **public** **class** ContourPolyline
10. {
11. **public** **int** PID { **get**; **private** **set**; }
12. **private** **static** **int** \_pid = 777777;
13. **public** List<DataPoint> PointList = **new** List<DataPoint>();
14. **public** ContourPolyline() { **this**.PID = \_pid++; }
15. **public** ContourPolyline(DataPoint[] points)
16. {
17. **this**.PointList.AddRange(points);
18. **this**.PID = \_pid++;
19. }
20. **public** **static** Object[] IntersectResult(ContourPolyline pl1, Edge edge)
21. {
22. List<ContourPolyline> sublineFromPL1 = **new** List<ContourPolyline>();
23. List<Edge> suEdgeFromEdge = **new** List<Edge>();
24. //对边上点排序
25. List<DataPoint> subEdgePoint = **new** List<DataPoint>();
26. subEdgePoint.Add(edge.StartPoint);
27. subEdgePoint.Add(edge.EndPoint);
28. edge.StartPoint.RelativeLoc = 0;
29. edge.EndPoint.RelativeLoc = 1;
30. //对折线上点排序
31. List<DataPoint> subLinePoint = **new** List<DataPoint>();
32. subLinePoint.Add(pl1.PointList[0]);
33. **for** (**int** i = 0; i < pl1.PointList.Count - 1; i++)
34. {
35. Edge pl1OneEdge = **new** Edge(pl1.PointList[i], pl1.PointList[i + 1]);
36. DataPoint intersectP = Edge.IntersectPoint(pl1OneEdge, edge);
37. **double** relativeLocOnLine = Edge.IntersectPointRelativeLoc(pl1OneEdge, edge);
38. **double** relativeLocOnEdge = Edge.IntersectPointRelativeLoc(edge, pl1OneEdge);
39. **if** (intersectP != **null**)
40. {
41. **if** (relativeLocOnEdge < 1 && relativeLocOnEdge > 0)
42. { intersectP.RelativeLoc = relativeLocOnEdge; subEdgePoint.Add(intersectP); }
43. **if** (relativeLocOnLine <= 1 && relativeLocOnLine > 0)
44. {
45. subLinePoint.Add(intersectP);
46. sublineFromPL1.Add(**new** ContourPolyline(subLinePoint.ToArray()));
47. subLinePoint = **new** List<DataPoint>();
48. subLinePoint.Add(intersectP);
49. }
50. }
51. subLinePoint.Add(pl1.PointList[i + 1]);
52. }
53. sublineFromPL1.Add(**new** ContourPolyline(subLinePoint.ToArray()));
54. subEdgePoint.Sort((x, y) => x.RelativeLoc.CompareTo(y.RelativeLoc));
55. **for** (**int** i = 0; i < subEdgePoint.Count - 1; i++)
56. {
57. suEdgeFromEdge.Add(**new** Edge(subEdgePoint[i], subEdgePoint[i + 1]));
58. }
59. **return** **new** Object[2] { sublineFromPL1, suEdgeFromEdge };
60. }
62. **public** **static** Object[] IntersectResult(ContourPolyline[] plineList, Edge edge)
63. {
64. List<ContourPolyline> sublineFromPLs = **new** List<ContourPolyline>();
65. List<Edge> suEdgeFromEdge = **new** List<Edge>();
66. //对边上点排序
67. List<DataPoint> subEdgePoint = **new** List<DataPoint>();
68. subEdgePoint.Add(edge.StartPoint);
69. subEdgePoint.Add(edge.EndPoint);
70. edge.StartPoint.RelativeLoc = 0;
71. edge.EndPoint.RelativeLoc = 1;
72. **for** (**int** k = 0; k < plineList.Length; k++)
73. {
74. //对折线上点排序
75. List<DataPoint> subLinePoint = **new** List<DataPoint>();
76. ContourPolyline curCpl = plineList[k];
77. subLinePoint.Add(curCpl.PointList[0]);
78. //选取一个等值线
79. **for** (**int** i = 0; i < curCpl.PointList.Count - 1; i++)
80. {
81. Edge pl1OneEdge = **new** Edge(curCpl.PointList[i], curCpl.PointList[i + 1]);
82. DataPoint intersectP = Edge.IntersectPoint(pl1OneEdge, edge);
83. **double** relativeLocOnLine = Edge.IntersectPointRelativeLoc(pl1OneEdge, edge);
84. **double** relativeLocOnEdge = Edge.IntersectPointRelativeLoc(edge, pl1OneEdge);
85. **if** (intersectP != **null**)
86. {
87. **if** (relativeLocOnEdge < 1 && relativeLocOnEdge > 0)
88. { intersectP.RelativeLoc = relativeLocOnEdge; subEdgePoint.Add(intersectP); }
89. **if** (relativeLocOnLine <= 1 && relativeLocOnLine > 0 )
90. {
91. **if** (subLinePoint.Count == 1 && subLinePoint[0].OID == intersectP.OID) { }
92. **else**
93. {
94. subLinePoint.Add(intersectP);
95. sublineFromPLs.Add(**new** ContourPolyline(subLinePoint.ToArray()));
96. subLinePoint = **new** List<DataPoint>();
97. subLinePoint.Add(intersectP);
98. }
99. }
100. }
101. subLinePoint.Add(curCpl.PointList[i + 1]);
102. }
103. **if** (subLinePoint.Count >= 2 && !(subLinePoint.Count == 2 && subLinePoint[0].OID == subLinePoint[1].OID))
104. sublineFromPLs.Add(**new** ContourPolyline(subLinePoint.ToArray()));
105. }
106. subEdgePoint.Sort((x, y) => x.RelativeLoc.CompareTo(y.RelativeLoc));
107. **for** (**int** i = 0; i < subEdgePoint.Count - 1; i++)
108. {
109. **if** (subEdgePoint[i].RelativeLoc != subEdgePoint[i + 1].RelativeLoc)
110. suEdgeFromEdge.Add(**new** Edge(subEdgePoint[i], subEdgePoint[i + 1]));
111. }
112. **return** **new** Object[2] { sublineFromPLs, suEdgeFromEdge };
113. }
115. **public** **override** **string** ToString()
116. {
117. **return** **string**.Format("CLid:{0},PtsCount:{1}", **this**.PID, **this**.PointList.Count);
118. }
119. }
120. }