3DCCOM Polygon Reduction Algorithm in Presence of Obstacles, Facilators and Constrains

Mamta Malik
Research Scholar
DCRUST, Murthal

Dr.A.K.Sharma
Professor & Dean
YMCA University of Science & Technology,
Faridabad

ABSTRACT
Clustering of spatial data in the presence of obstacles, facilitator and constraints has the very strong practical value, and becomes to an important research issue. Most of the existing spatial clustering algorithm in presence of obstacles and constraints can't cluster with irregular obstacles. In this paper a 3DCCOM Polygon Reduction Algorithm is proposed. The advantage of this clustering algorithm is to reduce polygon edges, memory would be very less than the matrix approach contains reduction line .We are here going to use Set of reduction lines than matrix approach. Further with help of Polygon Reduction Algorithm, A novel 3DCCOM (3 Dimensional Clustering with Constraints and Obstacle Modeling) algorithm is proposed. 3DCCOM takes into account the problem of clustering in the presence of physical obstacles while modeling the obstacles by Reentrant Polygon Reduction Algorithm. The 3DCCOM algorithm processes arbitrary shape obstacle and finds arbitrary shape clusters efficiently. Meanwhile, the 3DCCOM algorithm used to reduce the complexity of clustering in presence of obstacles, facilators and constraints and the operation efficiency of algorithm is improved. The results of experiment show that 3DCCOM algorithm can process spatial clustering in presence of obstacles, facilitator and constraints and has higher clustering quality and better performance.

Keywords
Polygon Reduction, Concave or Reentrant Polygon, Clustering Algorithms

1. INTRODUCTION
Clustering is one of the most useful tasks in data mining process which can partition objects of a data set into distinct groups such that two objects from one cluster are similar to each other, whereas two objects from distinct clusters are not [1]. Spatial clustering, aiming to identify clusters, or densely populated regions in a large spatial dataset, serves as an important task of spatial data mining. At present, the research of spatial clustering is very active. The main acceptable methods of this domain include algorithm based on spatial dataset partition, algorithm based on hierarchical, algorithm based on density, algorithm based on grid, etc. The precondition of these algorithms is that the direct accessibility is defined between spatial data samples. Most of these traditional clustering algorithms ignore the presence of obstacles, facilators and constrains. Euclid distance is used to measure the material comparability measurement between data samples and clusters. But the spatial data samples are always obstructed by obstacles such as hills, rivers, lakes, enclosed roads which destroy the connectivity of spatial data space in actual application whereas underpass, bridges etc. are the facilitators in actual space. The presence of obstacles results in the meaningless and impractical spatial cluster result which shows in Fig.1. The problem of spatial clustering in presence of obstacles, Facilators and constrains is highly interested recently.

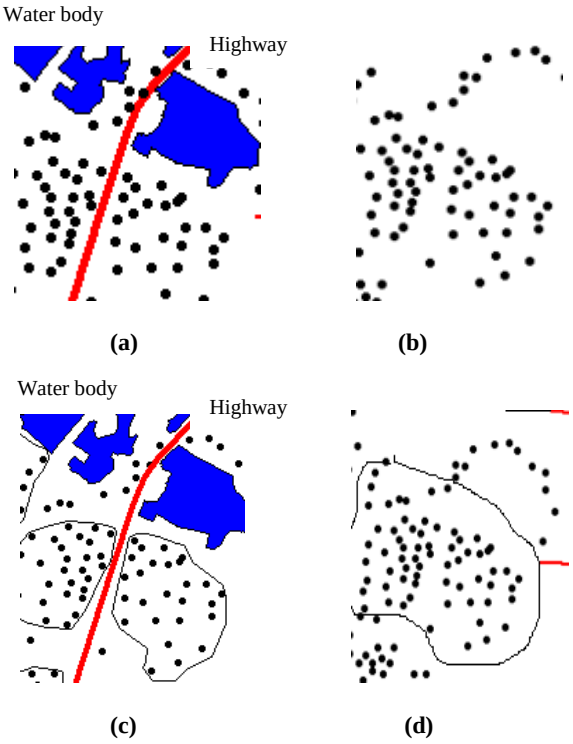


Figure 1 (a) Original Dataset with obstacles water body and highway etc., (b) Dataset without obstacles, (c) Cluster with considering obstacles, (d) cluster when ignoring obstacles

At present, the typical clustering algorithms of spatial data sample in presence of obstacle include COD-CLARANS, AUTOCLUST+, DBCLUC and DBRS+. Each method has its own advantage and disadvantage. COD-CLARANS [2] is the earliest clustering algorithm with obstacles which is proposed by Tung. It based on partitioning approach and defines obstacle-distance firstly, and computed the distance with global obstacle-culminations visual graphic algorithm. The operation cost of this clustering algorithm is higher. AUTOCLUST+ [3] is a Voronoi

graphic and Delaunay triangulation-based spatial clustering algorithm with obstacles which is proposed based on AUTOCLUST [4]. It ensures the accuracy of clustering result to certain extent by without any parameter. However, the cost of constructing Delaunay graphic and triangulation is higher and it lacks of flexibility of polygon obstacles treatment. DBCLUC [5] uses obstacle-line method to make the invariant spatial space to reduce processing time. But the parameters of this algorithm are complex and the algorithm can't process with reentrant polygon obstacles [15] effectively. DBRS+ [6] which is based on DBRS [7] defines minimum boundary region to reduce the affection of obstacle of spatial space. The cluster result is sensible with value of parameters of algorithm, and the cost of graphic pre-treatment of obstacles is too high. Therefore, Polygon Reduction clustering algorithm in presence of obstacles, facilitator and constrains which is abbreviated as PRC further extended as a novel 3DCCOM (3 Dimensional Clustering with Constraints and Obstacle Modeling). 3DCCOM takes into account the problem of clustering in the presence of physical obstacles while modeling the obstacles by Reentrant Polygon Reduction. As the same time, this algorithm adopts hierarchical idea to cluster spatial data space in presence of obstacles [15]. It divides the whole data space into lots of region without obstacle by raster extension line of obstacle polygon boundary. The Reentrant Polygon Reduction clustering is used to cluster in these regions without obstacle first, with obstacle and facilitator second.

2. BACKGROUND CONCEPTS

2.1 Spatial clustering in presence of obstacles and constrains

The definition of clustering in the presence of obstacles and constrains is proposed in paper [2]:

Definition 2.1.1 clustering with obstacle: (1) Dataset P = {p_1, p_2... p_n} includes n data points; (2) the disjoined obstacles set O = {O_1, $O_2 \ldots O_m$} in 2-dimensional data space R. Each obstacle O_i is described as a simple polygon. The distance between two points in the space is Euclid distance d (pk, pj) when the obstacles are ignored. When the obstacles are taken account of, the distance between pk and pj recording as d (pk, pj) is the shorted path which can't be cut by any obstacle. The spatial clustering in presence of obstacle is the processing which divides the whole data space into k clusters C_{L1}, C_{L2},.. .. .. ,C_{Lk} to make every data point most closed to its cluster centre. That is equivalent to make the value of E minimum in formula (1) Ci is the cluster centre of C_{Li}.

$$E = \sum_{i=1}^{k} \sum_{p \in CLi} (d'(p, c_i))^2 \qquad (1)$$

2.2 Polygon-description of obstacles and preprocessing

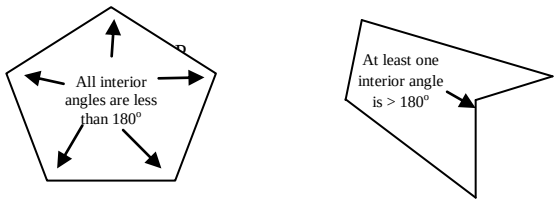
To estimate the affection of obstacle in the clustering process of spatial data samples, the suitable description of obstacle is needed first. The obstacles, facilitator and constrains in reality are in different shape and character. Water body and highway shows as stripe polygon while hill, bridge, underpass may be curve in shape, lake and park shows as plane polygon in 2-dimensional space. To facilitate the storage and management of obstacle in computer program, the vector spatial data structure [8] is used to describe each obstacle polygon. Obstacles-polygons set is Y = {Y_1, Y_2... ……, Y_n}, a series of ordered space points from the polygon Y_k= {Y_{k1}, Y_{k2}.....,Y_{ks}} where ($1 \le k \le n$). The coordinate of space point is

$$Y_{km} = (Y_{km1}, Y_{km2})^T \in R^2 (1 \le m \le s)$$

2.3 Obstacle Modeling

Almost all physical obstacles like rivers, hills, and highways etc. can be modelled using simple polygons [12]. All the polygons can be divided into two types: simple polygons and crossing polygons. A simple polygon is the polygon in which every edge in the polygon is not intersected with any other edge in the polygon and a crossing polygon is the polygon in which at least one edge is intersected with any other edge in the polygon. We take into account simple polygons because almost all obstacles can be represented using only simple polygons. Simple polygons can be further divided into two types: convex and concave as shown in figure 2(a) (b). A polygon is a convex polygon if all vertices of the polygon make the same directional turn whether clockwise or anticlockwise. Suppose a polygon P does not follow the claim. It then is obvious that P is not a convex. All other polygons, which don't satisfy this condition, are said to be concave. In order to test a turning direction for 3 consecutive vertices, the sign of the triangle area of 3 points is examined via a determinant. As a result, the sign of the determinant evaluates the turning direction either a clockwise or a counterclockwise.

Note that we assume that all points in a polygon are enumerated in an order either clockwise or a counterclockwise. Hence, we can easily identify a type of a polygon as well as a type of each vertex from the polygon in a linear time O (n), where n is the number of points in a polygon.



(a) Convex polygon **(b) Concave/ Reentrant polygon**

Figure 2(a) in convex polygon all interior angles are less than 180 degree; (b) in concave polygon at least one interior angle is greater than 180 degree.

3. PROPOSED REENTRANT POLYGON REDUCTION ALGORITHM

In any clustering algorithms, when obstacles are considered, the visibility of data objects with each other is checked via the line segments or edges of the obstacle. The number of line segments to check is the number of edges of the polygons, which is large in number for a large data space. The number of lines to check can be reduced to actual one by our proposed polygon-edge reduction algorithm but memory would be much less then the matrix approach contains reduction lines. We are here to going to use set of reduction lines. Let us call the reduced number of lines as reduction lines. The algorithm assumes the following definition of a polygon.

Definition: Polygon

A simple polygon is denoted by an undirected graph P (V, E) where V is a set of k vertices: $V = \{v_1, v_2 \ldots, v_k\}$ and E is a set of k edges: $E = \{e_1, e_2 \ldots e_k\}$ where ei is a line segment joining v_i and v_{i+1}, $1 \le i \le k$. i+1=1 if i+1>k. First all the convex vertices of the polygon are extracted because only convex vertices are considered to find the visibility between two data objects. Assume that a polygon P (V, E) of n convex vertices is stored in the form of adjacency matrix A of order n×n where A [I,J]=1 if edge (I,J) exists between vertices I and J i.e.(I,J) ε E.

A [I, J] =0 if (I, J) not ε E.

The algorithm returns the output ordered set O.

$O = \{(I, J) : I, J \in V$, pair (I, J) is a reduction line$\}$

It first identifies the convex vertices in the polygon by turning direction approach and by checking the triangle area of three consecutive points via its determinant. After finding all the n convex vertices, a matrix A of order n×n stores the link information about polygon. The entries in the upper half of matrix A are checked so as to avoid the repetition because the polygon is undirected graph.

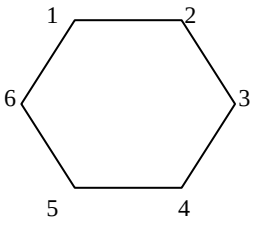
Proposed Reentrant Polygon Reduction Algorithm

```
Algorithm: Reentrant_polygon_reduction (P)
//P is given polygon with V vertices and E edges
Output: A set of obstruction lines (I, J) in ordered set O.
  Identify the convex and concave vertices. Let convex vertices be n;
  Store the link information of convex vertices in A taking them in
order;
Flag=0; k=0;
FOR (I=1; I<=n; I++) {
    FOR  (M=0; M<=k; M++) {    // k is always<=n
      IF(I= = B[M])
      {     // B is matrix for storing row numbers
          Flag:=1;
       }
       FOR (J=I; J<=n; J++)
       {
              IF( ( A[I,J]= =1) OR ( (A[I,J]= =0) AND ( (I,J) is interior
      to P) ))
              {
               Push(O,I,J);// Insert (I,J) into ordered set O
                     B[k]:=J;   k++;
          }
          IF(Flag = = 1)
            {
               A[I,J]:=0;
          }
       }
      Flag:=0;
    }
  }
Return O i.e. Reduction lines L;   //END polygon_reduction;
```

All reduction lines should be interior to polygon P and each convex vertex should've at least one reduction line from it. The number of reduction lines must be at least equal to the number of convex vertices to allow the correct visibility between the data points.

EXAMPLE showing polygon reduction

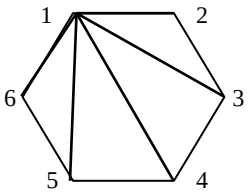
Take the example convex polygon shown in figure 3(a) that has six convex vertices and six edges. Corresponding to these six convex vertices, the input matrix A becomes as shown in figure 3(b). As a result of the application of the polygon_reduction algorithm, the output ordered set O is shown in figure 3(d). The output-reduced polygon is constructed according to output ordered set O and is shown in figure 3(c), which contains five instead of six reduced lines.



(a) Input Polygon

	1	2	3	4	5	6
1	0	1	0	0	0	1
2	1	0	1	0	0	0
3	0	1	0	1	0	0
4	0	0	1	0	1	0
5	0	0	0	1	0	1
6	1	0	0	0	1	0

(b) Input matrix A



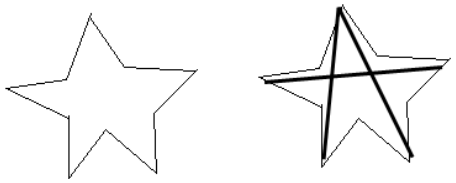
(c) Output Matrix

Set(O)={ (1,2) (1,3) (1,4) (1,5) (1,6) }

(d) Output ordered set O

Figure 3(a) Input Polygon, (b) Input Matrix A having replicated data, (c) Output Matrix, (d) Output ordered set O with no replicated data

This is the case where significant improvement is not achieved but in the case of concave polygons, a remarkable improvement can be obtained as is shown in figure 4(a), (b) where ten lines are being reduced to only three reduction lines. On an average, the number of reduction lines obtained by the polygon reduction algorithm is approximately half the number of edges of the polygon.



(a) (b)

Figure 4(a) a concave polygon, (b) resulting reduction lines

So, in a large dataset, where the number of obstacles can be large in number and hence the number of edges to test is also large in number, the polygon_reduction algorithm can be applied to reduce the number of lines to test during the clustering procedure.

3.1 Performance Evaluation

Complexity of polygon_reduction is much less than O (k.n2), k≤n is any positive integer. The reduction algorithm can be used either as a preprocessing step for the clustering algorithm where its complexity is not considered as a part of the complexity of main clustering algorithm, or can be considered as a part of clustering algorithm where its complexity amounts to the complexity of main clustering algorithm. Although the complexity of the reduction algorithm costs a little bit, but still benefits are more than pitfalls. The visibility of the corresponding pair of points can be tested easily and at a faster rate, reducing the overall running time of the clustering procedure. Let N be the number of points of a polygon P, and m and n are the number of concave points and the number of convex points respectively with N= n +m. The convexity test for P requires O (N). The polygon reduction algorithm requires a weighted graph to replace a non admissible obstruction line segment with a set of admissible line segments. The complexity in the replacement is in worst-case O (N log N). Polygon reduction algorithm for P requires O (N log N + k.∑n) in the worst case. The complexity of the polygon reduction is on average in the order of ≤ O (n2), n≤ N.

4. EXTENDED 3DCCOM ALGORITHM

The algorithm 3DCCOM (3 Dimensional Clustering with Constraints and Obstacle Modeling) pronounced as 3DCCOM takes into account the problem of clustering in the presence of physical obstacles while modeling the obstacles by polygon reduction algorithm.

4.1 3DCCOM Algorithm

The algorithm is based on the concept of density based clustering and hence on the algorithms DBSCAN [13], DBCLuC (Density Based Clustering with Constraints) [14] and DBCCOM. Some other definitions related to the algorithm 3DCCOM, besides those in density based clustering, are further added to have a clear idea about the algorithm 3DCCOM. The input parameters Eps and Minpts are the same as that in DBSCAN [13] algorithm.

Definition 4.1.1 Obstacle

An obstacle can be thought of a simple polygon P (V, E) where V= {v1, v2…vk} is the set of vertices and E= {e1, e2…ek} is the set of edges of the polygon where ei is the edge between nodes v_i and v_{i+1}. 1≤i≤k. i+1=1 if i+1>k.
So, an obstacle is a simple polygon represented as an undirected graph.

Definition 4.1.2 Visibility

- If polygon reduction is not used: Let D= {d1, d2…dn} is the dataset of n points. Visibility is a relation between two data points. Two points di and dj; i ≠ j are visible to each other if the line segment joining di and dj is not intersected with any polygon edge el; 1≤ l≤ k.
- If polygon reduction is used: Let L= {l1, l2…lm} be a set of m reduction lines produced by a polygon reduction algorithm. Two points d_i and d_j; i ≠ j are visible to each other if the line segment joining di and dj is not intersected with any reduction line li; 1≤ i≤ m.

Definition 4.1.3 Visible Space

Given a set D of n data points D = {d_1, d_2…d_n}. A visible space is a set S of k data points S = {s_1, s_2…s_k} such that ∀si, sj ε S, si and sj are visible to each other, while s_k is not visible to s_k's in S', where S' is a visible space such that S'∩S=ϕ; S, S'⊆D; i≠j; i, j ε [1..n].

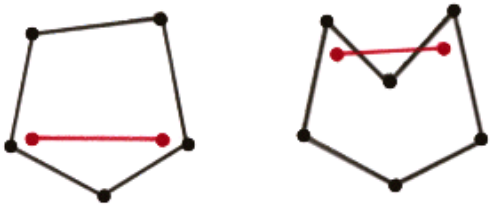
In the presence of obstacles, it becomes necessary to check the visibility of two points. So, the new definition of cluster becomes:

Definition 4.1.4 Cluster

A cluster C with respect to Eps and Minpts is a non-empty subset of D satisfying following conditions:

1. Maximality: ∀ p,q ε D, if p ε C and q is density reachable from p wrt Eps and Minpts, then q ε C.
2. Connectivity: ∀ p,q ε C, p and q are density connected to each other wrt Eps and Minpts.
3. Visibility: ∀ ci, cj ε C, there exists a chain of zero or more point's c1, c2…ck, k≤n such both ci and cj are visible to each other either directly or indirectly through the pair wise chain of k points. For every ciεC there exists at least one point cjεC, such that ci, cj are visible to each other.

Figure 5(a), (b) illustrates the concept of cluster condition 3 when considering obstacles.



(a) (b)

(a) Direct visibility between two points, (b) Indirect visibility between two points

Figure 5 Visibility between points of a cluster

The number of visible spaces in a convex polygon is same as the number of convex points since each line segments from the convex vertex blocks the visibility of its neighbor visible spaces. In contrast a concave point does not create two visible spaces but uses a visible space created by its nearest convex point. There should be at least one reduction line from a convex vertex in the resulting reduced polygon so that the concept of visible spaces is not compromised. Also each of the produced reduction line should be inside the polygon or it may coincide with any edge of the polygon only then it is admissible reduction line. The main clustering algorithm 3DCCOM is described below where polygon_reduction is taken into consideration.

The clustering procedure in 3DCCOM is quite similar to DBSCAN. A database D is a set of data points to be clustered in the algorithm. First all the reduction lines from the set of obstacles are extracted by an iterative call to Reentrant polygon_reduction (). After that clustering procedure is initiated. The Expand-Cluster is described next and it may seem similar to the function in the DBSCAN. However, the distinction is that obstacles are considered in RetrieveNeighbours (Point, Eps, L) illustrated in last.

```
3DCCOM (Database D, Obstacles O)
Output: A set of Clusters.
//Start clustering
 L:=ϕ;       // L is the set of reduction lines
  FOR (obstacle ε O) do
        Lines:=Reentrant_Polygon_reduction(obstacle);
        L:=L ∪ lines;
  END FOR;
 ClusterId:=nextId(noise);
 FOR (pt ε D) do
      IF (pt.ClusterId= unclassified) THEN
          IF ExpandCluster(D,pt,ClusterId,Eps,Minpts,L) THEN
                ClusterId:= nextId(ClusterId);
          END IF;
      END IF;
END FOR;
RETURN cluster Ids; //END 3DCCOM;
```

Given a query point, neighbors of the query point are retrieved using SR tree. In 3DCCOM, we adopt the range neighbor query approach instead of the nearest neighbor query approach from SR-tree, since it is extremely difficult for the latter to expand a set of clusters if a density of data objects is high. Its average run time of a neighbor query is O (log N) where N is the number of data objects.

```
ExpandCluster(D,pt,ClusterId,Eps,Minpts,L)
Input: Database, a data point Point, ClusterId, Eps, MinPts, and
reduction lines L
Output: True or False
 SEED := RetrieveNeighbours (Point, Eps, L);
 IF (SEED.SIZE< MinPts) THEN
       Classify Point as NOISE;
       Return False;
 END IF;
 Change clustered of all elements in SEED into ClusterId;
 Delete Point from SEED;
 WHILE (SEED.SIZE>0) do
       Current-Point: = SEED. First ();
       RESULT: = RetrieveNeighbours (Current Point, Eps, L);
       IF (RESULT.SIZE>=MinPts) THEN
            FOR (element ε RESULT) do
                 IF (element is UNCLASSFIED) THEN
                      Put it into SEED;
                      Set its cluster id to ClusterId;
                 END IF;
                 IF (element is NOISE) THEN
                      Set its cluster id to ClusterId;
                 END IF;
            END FOR;
       END IF;
       Delete Current-Point from SEED;
  END WHILE;
Return True;
```

The visibility between two data objects in the presence of obstacles is computed using a line segment whose endpoints are the two data objects in question. If this line is intersected with any reduction line, then the two data points are not grouped together, since they are not visible to each other. Those accepted neighbors defined as the SEED that are retrieved by RetrieveNeighbours continue to expand a cluster from elements of the SEED, if the number of elements in the SEED is not less than MinPts. A data object is labeled by a proper cluster id, if retrieved neighbors are satisfied with the parameter MinPts discriminating outliers. The "RESULT" in Algorithm below is a set of data objects that are neighbors of a given query object.

```
RetrieveNeighbours (Point, Eps, L)
Input: a data object Point, Eps, and reduction lines L
Output: A set of data points
  RESULT: = GetNeighbour (Point, Eps);
  FOR (element ε RESULT) do
      IF Check-Visibility-with (Point, element, L)=FALSE THEN
           RESULT. Delete (element);
      END IF;
  END FOR;
 Return RESULT;
```

The RESULT elements are constructed by removing data objects that are not visible to the sample point because of the blockage of obstacles. This algorithm retrieves neighbors of a given query point using SR tree.

5. RESULTS AND ANALYSIS OF EXPERIMENT

Fundamentally, 3DCCOM is a density based clustering algorithm. DBCLUC is the mostly acceptance density-based clustering algorithm in presence of obstacle. To demonstrate the advantage of 3DCCOM, it will compare with DBCLUC in this experiment. To facilitate the comparison of two algorithms, the data set which is generated by the improved DatGen [11] is used and compares it with new dataset. For simplicity, the synthetic spatial data set is 3-dimesional spatial data. The data set and obstacles are showed as Fig.6. The best results of two algorithms with a broad range of parameter settings are selected. Clustering result of this spatial data space is showed by Fig.7 when the obstacles are ignored. The different cluster of data space is described by different color.

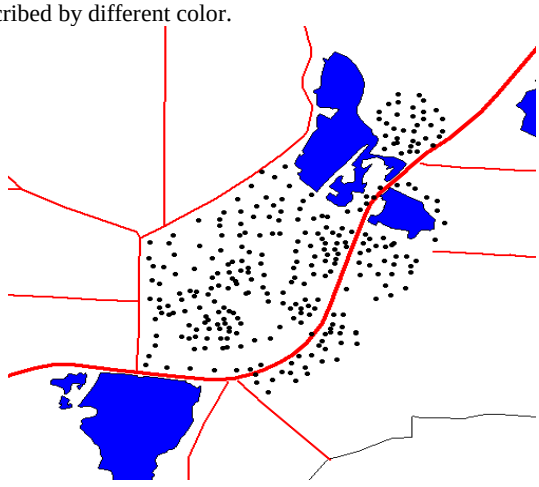


Figure 6. Spatial dataset with obstacles

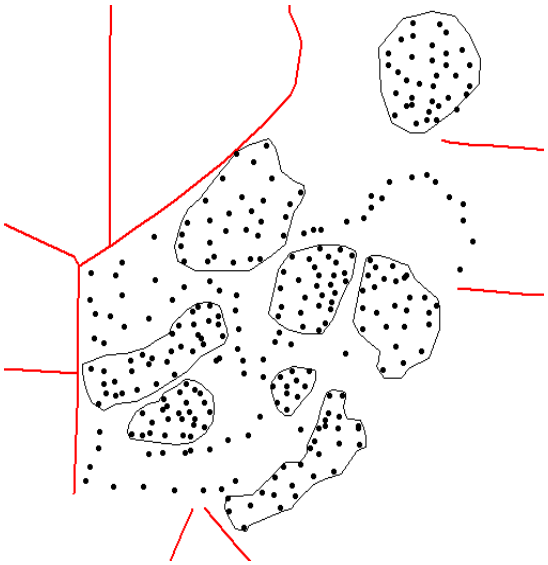


Figure 7. Clusters without obstacles and constraints

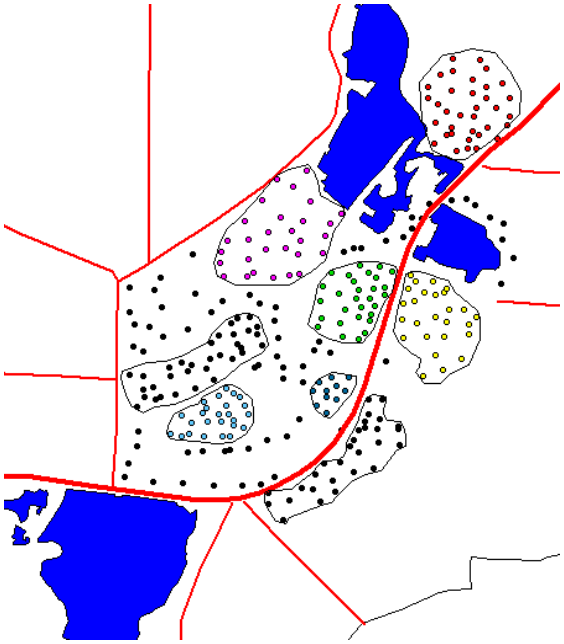


Figure 8. Clusters with obstacles and constraints

The connectivity of spatial data space is destroyed by existing obstacles when they are considered. Cluster results of two cluster algorithm in presence of obstacles and constrains DBCLUC and 3DCCOM are showed by Fig.7, Fig. 8 and Fig.9. Through comparison and analysis of cluster results of two algorithms, the conclusion is that 3DCCOM can get better cluster result in presence of cluster than DBCLUC. DBCLUC can cluster spatial data space with obstacles, but the process of obstacles in arbitrary shape is ideal insufficiently. There are many scattered meaningless cluster in the final cluster result of

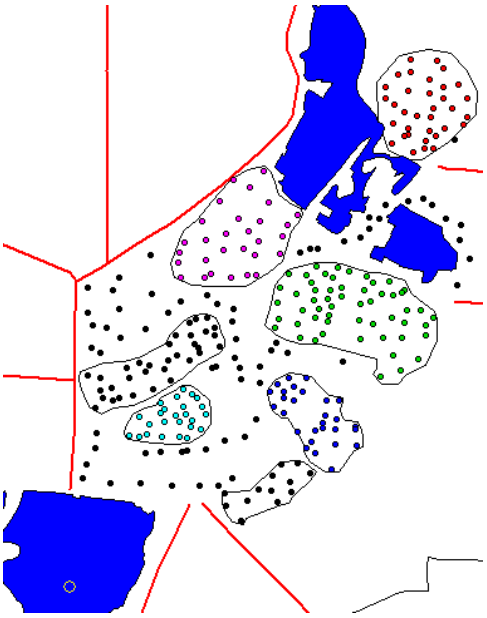


Figure 9. Cluster results of 3DCCOM

DBCLUC. 3DCCOM algorithm proposed in this paper inherits advantage of polygon reduction and density cluster algorithm and it can operate obstacle polygons and find clusters in arbitrary shape to avoid scattered meaningless cluster in final result. So the cluster result of 3DCCOM is more accurate and more practical. At the same time, the execution space-time cost of 3DCCOM is very less than DBCLUC because of the adoption of reentrant polygon reduction strategy. Clustering process in presence operates upon cluster representative points of first order cluster instead of the whole data space in 3DCCOM. Comparison with other cluster algorithm in presence of obstacles, facilitator and constrains, 3DCCOM is more applicable to operate on large scale spatial dataset.

There are a number of areas into which the proposed work can be extended or improved. The work shows how to consider obstacles in the clustering process and how to model the physical obstacles using the reduction algorithm, but no indexing scheme is used for obstacles. In the absence of any indexing scheme, all produced reduction lines are checked for visibility of a data point. By using an indexing scheme, only lines in the neighborhood of a particular data point can be checked instead of all the lines. With such a scheme, the complexity can be reduced to *O (N log N)* which would be a significant improvement over the proposed algorithm.

6. ACKNOWLEDGMENTS

The authors would like to thank the YMCA University of Science and Technology and DCRUST Sonepat to support us in our research and allow us to use laboratory for various software like MapInfo, ArcGIS, clustering tools, allow accessing various research reports and off course for their valuable suggestions.

7. REFERENCES

[1] Han Jiawei, Kamber M, 2001 "Data Mining: Concepts and Techniques," Academic Press.

[2] Tung A K H, Hou Jean, Han Jiawei, 2001, "Spatial Clustering in the Presence of Obstacles," Int. Conf. on Data Engineering, pp. 359-367.

[3] V. Estivill Castro, I. J. Lee, 2000 "AutoClust+: Automatic Clustering of Point-Data Sets in the Presence of Obstacles," Int. Workshop on Temporal, Spatial and Spatio-Temporal Data Mining, pp. 133-146.

[4] V. Estivill Castro, I. J. Lee, 2000, "AutoClust: Automatic Clustering via Boundary Extraction for Massive Point-Data Sets," Int. Conf. on Geo-computation'00, pp. 23-25

[5] O. R. Zaiane, C. H. Lee, 2002, "Clustering Spatial Data when Facing Physical Constraints," The IEEE International Conf. on Data Mining, pp. 737-740

[6] X. Wang, H. J. Hamilton, 2004, "Density-Based Spatial Clustering in the Presence of Obstacles and Facilitators," Int. the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 446-458

[7] X. Wang, H. J. Hamilton, 2003, "DBRS: A Density- Based Spatial Clustering Method with RandomSampling," The 7th International Conference on PAKDD, pp. 563-575

[8] Chen Shupeng, Lu Xuejun, Zhou Chenhu, 1999, "Geographic information introduction," Beijing: Machinery Industry Press.

[9] Zhou Lihua,Wang Lizhen, Chen Keping, 2004, "Spatial Hierarchical Clustering in the presence of obstacle," Computer Science, Vol.33, No.5,2006, pp. 182-185.

[10] Sun Yuqing, Zhao Rui, Yao Qing, 2006, "A mesh-based clustering algorithm in the presence of obstacles," Journal of Shan Dong University (engineering science), Vol.36, No.3, pp.86-90

[11] Melli G, Dataset generator (DatGen), http://www.datasetgenerator.com

[12] C.-H. Lee and O. R. Za¨ıane. 2002," Polygon reduction: algorithm for minimum line representation for polygons. In Submitted to 14th Canadian Conf. on Computational Geometry.

[13] Ester M., Kriegel H.-P., Sander J., Xu X., 1996: "A Density Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining, Portland, OR, AAAI Press, pp. 226-231.

[14] Xin Wang, Camilo Rostoker & Howard J. Hamilton, 2004-9, Technical Report "Density-Based Spatial Clustering in thePresence of Obstacles and Facilitators", CS-2004-9.

[15] Yue Yang, Jian-pei Zhang, Jing Yang, 2008, Grid-based Hierarchical Spatial Clustering Algorithm in Presence of Obstacle and Constraints" at International Conference on Internet Computing in Science and Engineering.