Algorithm for Constructing Complete Distributive Lattice of Polyhedrons Defined over Three Dimensional Rectangular Grid- Part II

G. Ramesh Chandra* and E.G. Rajan

Pentagram Research Centre Pvt. Ltd.,
Mehdipatnam, Hyderbad, India
rameshchandra_g@vnrvjiet.in, rajan_eg@yahoo.co.in

Abstract. This paper initially discusses how Geometric Filters (G-Filters) are useful as an efficient shape filter when compared to other shape filter such as mathematical morphology. The algorithm for constructing complete distributive lattice of polyhedrons for three dimensional rectangular grid is divided in to two algorithms. The first algorithm proposes a new way automatic construction of 256 convex polygons by removing the duplicate subsets. The second algorithm proposes a way for hierarchical path enumeration in visualizing the relationships between convex polyhedron sets and their corresponding subsets. The final lattice is generated based on the information provided by these two algorithms.

Keywords: Geometric Filters, Automatic Convex Polyhedron Construction, lattice, 3D rectangular grid, Hierarchical Path Enumeration, 3-D image processing.

1 Introduction

In very broad set-theoretic terms, a system is a filter which has a separate, to partition a given set of entities in to disjoint subsets, based on whether they have or do not have a stipulated property. One does, in principle, conceive of a procedure for such a partitioning, one that is a function of time and space, and which depends for its realization on physical device, be it a sieve, or on electronic circuit, or a computer.

In the study and design of such filtering procedures for signals, one crucial underlying principle seems to be the following: *System models of signal processing operations and those of sources from which signals originate should belong to the same class.* This principle was introduced by Rajan and Sinha , and is referred here as R-S Principle(RSP). It is implied here that signals are to be thought of as outputs of sources, and that their description is not really complete unless we specify source models.

* AssociateProfessor, Computer Science Department, V.N.R. Vignana Jyothi Institute of Engineering and Technology, Bachupally, Hyderabad, Andhra Pradesh, India.

N. Meghanathan et al. (Eds.): CCSIT 2012, Part III, LNICST 86, pp. 202–210, 2012.

The existing time invariant systems which can serve as filters, are governed by differential equations do not serve as canonical models. These systems of filter theory cannot be able to serve the purpose of efficient image processing operations. In general, complex systems that are traditionally modelled in terms of differential equations and partial differential equations could be ideally modelled and studied in the framework of cellular automata[2].

Alternatively, one can create generalized spatial domain techniques for processing images and pictures, which make use of shape based filters. Morphological image processing is one such technique[1][7][8][9][10], which deals with processing of form and structure of digital images. Yet this technique does not measure up to fulfil the requirements of shape based filtering similar to traditional frequency based filtering. The question that arises now is whether it is possible to develop a technique or at least set a trend which leads to the formulation of a theory for shape based filtering. The theoretical foundations for such shape based filters are studied in the framework of Geometric filters also called G-Filters[2][4][5][6]. The G-Filters are practically implemented in the processing of 2-D images to effectively and efficiently identifying the shape features such as corners, curves, lines, dots and skeleton[2][4][5]. The Logical Image processing system (LIPS) version 3.0 implemented these G-Filters for efficient shape feature detection for 2-D images.

There are no such techniques formulated till now for the 3-D images under the rectangular grid. The algebraic intricacies and algorithms development to simulate these theoretical foundations of 3D G-filters are separated in to two papers.

The paper (Part I) [11] discusses the theoretical foundations of 2-D and 3-D G-Filters over the rectangular grid. The same paper initially discusses the 2-D Geometrical filters, and then proposes the 3-D Geometrical filters (3-D G-Filters) and their algebra.

In this paper we deal with the implementation and algorithms related to 3-D G-Filters. The algorithm in the section 2 discusses automatic construction of 255 convex polyhedrons used for constructing complete distributive lattice[3] of polyhedrons for rectangular grid. In Section 3, we discuss the algorithm for hieararchical path enumeration of convex polyhedron set and their relationships formed in hierarchical fashion. Section 4 discusses the results.

2 Algorithm for Automatic Construction of Convex Polyhedron Sets and Their Corresponding Subsets over Rectangular Grid

Consider a 3x3x3 rectangular grid, in which there are 27 neighbourhood elements including central pixel. The central idea is to construct 256 unique convex polyhedrons in a 3-D rectangular grid(3x3x3 array of cells), by dropping one corner, two corners, three corners, four corners, fiver corners, six corners, seven corners and eight corners at a time with different probable combinations. The initial set A contains all the eight different corner voxels of the 3x3x3 voxel grid. (for more details on numbering the corner voxels refer the paper [11] published in the same conference) i.e.

A= {1, 3, 7, 9, 19, 21, 25, 27}
By dropping one voxel at a time we can construct 8 different subsets as follows:
B1= {3, 7, 9, 19, 21, 25, 27}
B3= {1, 7, 9, 19, 21, 25, 27}
B7= {1, 3, 9, 19, 21, 25, 27}
B9= {1, 3, 7, 19, 21, 25, 27}
B19= {1, 3, 7, 9, 21, 25, 27}
B21= {1, 3, 7, 9, 19, 25, 27}
B25= {1, 3, 7, 9, 19, 21, 27}
B27= {1, 3, 7, 9, 19, 21, 25}

The dropped voxel can be easily identified by looking at the subset index. For example, the voxel 1 is dropped from the set of corners i.e. A and formed a subset B1, here one can observe that the dropped voxel is considered as a index of the new subset B. For complete list of the 256 convex polyhedrons refer to Paper (Part I)[11].

For reducing complexity in finding subsets from the set, only one element is removed from the set at each level. For example by removing one element from the set A (level 1) we found subsets B1, B3, B7, B9, B19, B21, B25, B27 (level 2). In the similar way by removing one element from B1 we found C1,3, C1,7, C1,9, C1,19, C1,21, C1,25, C1,27 (level 3). In the level 3 we found 28 such unique C subsets. In the similar way we have 56 D subsets in level 4 by discarding one element from C sets, 70 E subsets in level 5 by discarding one element from D sets, 56 F subsets in level 6 by discarding one element from E sets, 28 G subsets in level 7 by discarding one element from F sets, 8 H subsets in level 8 by discarding one element from G sets, and 1 E subset in level 9 by discarding one element from H sets. When all the subsets are properly arranged we get a complete distributive lattice structure as shown in Figure 3. The total number of convex polyhedrons is calculated by adding all the subsets i.e. 256. The list of all the subsets and their corresponding elements are discussed in Paper (Part I)[11].

So this section discusses the algorithm for automatic generation of convex polyhedron subsets from the set A.

Algorithm 1: Automatic Generation of Convex Polyhedrons

```
Input: Set A. i.e. A= {1, 3, 7, 9, 19, 21, 25, 27}
Output: 255 subsets
Steps:
Step1: Declare a array a[ ] and initialize it with
    {1, 3, 7, 9, 19, 21, 25, 27}

Step 2:
```

```
For Subset Group B:
for ( int i = 0; i <8; i++)
  {
    for (int j = 0; j <8; j++)
      {  Print B+ a[i]+{=
           if ((j == i) )
           {
              //Dont do anything
           }
       else
            {
              Print a[j] + ,
          }
} print }\n
}
Group C:
for(int i=0;i<=7;i++)
{
  for (int j = i + 1; j <= 7; j++)
  {
    Print C+a[i]+a[j]+{=
    for (int d = 0; d <= 7; d++)
    {
      if (d == i || d == j)
      { //Dont do anything
      }
           else
           {
             Print a[d]+,
           }
    }
   } print }\n
}
Similarly the group D subset can be generated with four for
loops. Expect out for loop  and inner most for loop every for
loop is intialized to its previous for loop variable+1.
For Group E 5 Nested for loops
    Group F 6 Nested for loops
    Group G 7 Nested for loops
    Group H 8 Nested for loops
    Group I 9 Nested for loops
```

The output of the above said algorithm is discussed in the results section i.e. Section 4

3 Algorithm for Constructing Hierarchical Relationships among Convex Polyhedron Sets and Their Corresponding Subsets

The purpose of this algorithm is to make hierarchical path enumeration among convex polyhedron sets and their subsets. As discussed in the previous section the sets C1,3, C1,7, C1,9, C1,19, C1,21, C1,25, C1,27 are subsets of B1. This algorithm will construct a tree view of the relationships mention above such sets. The algorithm present in this section will form a hierarchical relationship among sets and subsets.

Algorithm 2: Hierarchical Path Enumeration

```
Input: Set A. i.e. A= {1, 3, 7, 9, 19, 21, 25, 27}
Output: Tree view of convex polyhedron sets and their
        corresponding polyhedron subsets
Steps:
Step1: Declare a array a[ ] and initialise it with
    {1, 3, 7, 9, 19, 21, 25, 27}
Step 2:
For Subset Group B:
Create Empty tree
Add set A as root node to the tree
for ( int i = 0; i <8; i++)
{
    Create Empty BNode
    Name node as  "B" + a[i];
    Add the node generated to the
       Root node i.e. A.
    for (int j = 0; j <8; j++)
    {
        // Logic for Subset Creation
    }
}
Group C:
for(int i=0;i<=7;i++)
{
  for (int j = i + 1; j <= 7; j++)
  {
    Create Empty CNode
    Name node as  C+a[i]+,+a[j]
    Add the node generated to the
    Parent node as follows:
    mainNode.Nodes[i].Nodes.Add(CNode);
    for (int d = 0; d <= 7; d++)
```

```
    {
    // Logic for Subset Creation
    }
  }
}
Group D:
for (int i = 0; i <= 7; i++)
{
   for (int k = i + 1; k <= 7; k++)
   {
     for (int j = k + 1; j <= 7; j++)
     {
      Create Empty Node
      Name node as D+a[i]+a[k]+a[j]
      Add the node generated to the
      Parent node as follows:
        m = ((k - 1) - i);
      mainNode.Nodes[i].Nodes[m].
      Nodes.Add(DNode);
      for (int d = 0; d <= 7; d++)
      {
    // Logic for Subset Creation
      }
     }
    }
}
Similarly the group E subset can be generated with five for loops.
For Group F 6 Nested for loops
    Group G 7 Nested for loops
    Group H 8 Nested for loops
    Group I 9 Nested for loops
```

The output of the above said algorithm is discussed in the results section i.e. Section 4

4 Results

The algorithms discussed in the section 2 and 3 are implemented in C#.NET language. The results of the convex polyhedron construction are shown in the Fig 1. The Fig 1(a) shows results of some of the convex polyhedrons. The other subsets ending with F, G, H and I are shown in the Fig 1(b). The algorithm discussed in the section 2 is able to generate all 255 convex polyhedrons. The complete list of convex polyhedrons can not be listed here as output. So the partial list is shown in the Fig 1(a) & 1(b). The results of algorithm discussed in the section 3 is shown in Fig 2. This figure shows the relationships among polyhedron sets and their corresponding subsets. The Fig 2(b) shows the lower part of the some more results of the algorithm for hierarchical subset construction. By looking at the output one can clearly understand that H is subset of

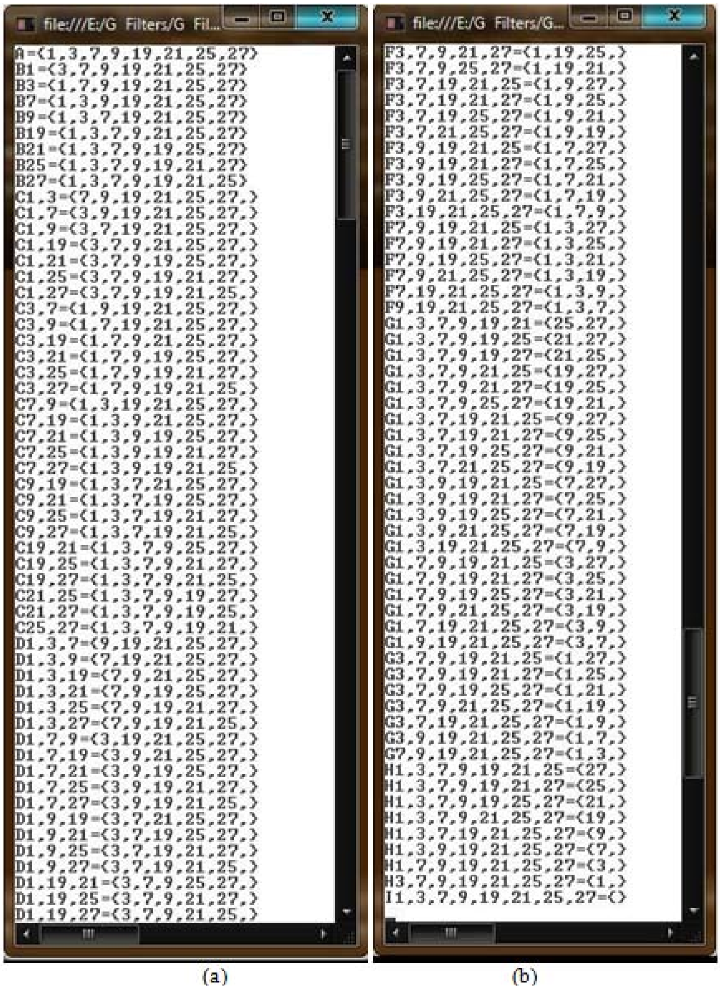

Fig. 1. (a) Results window showing subsets from A to D1, 19, 27(b) Results of subsets showing F, G, H and I

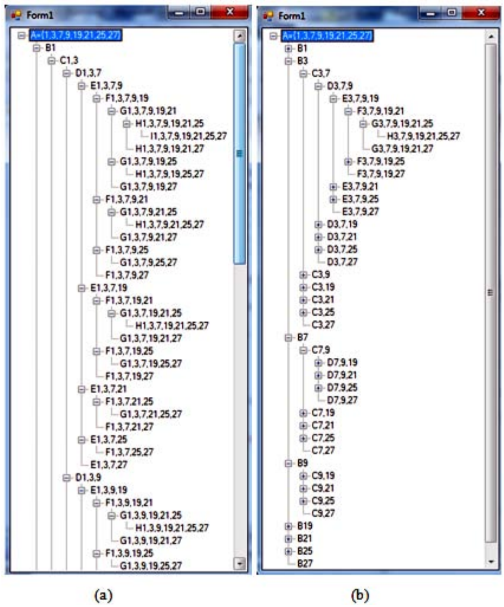

Fig. 2. (a)The hierarchical relationships showing upper part of the results(b)The hierarchical relationships showing lower part of the results

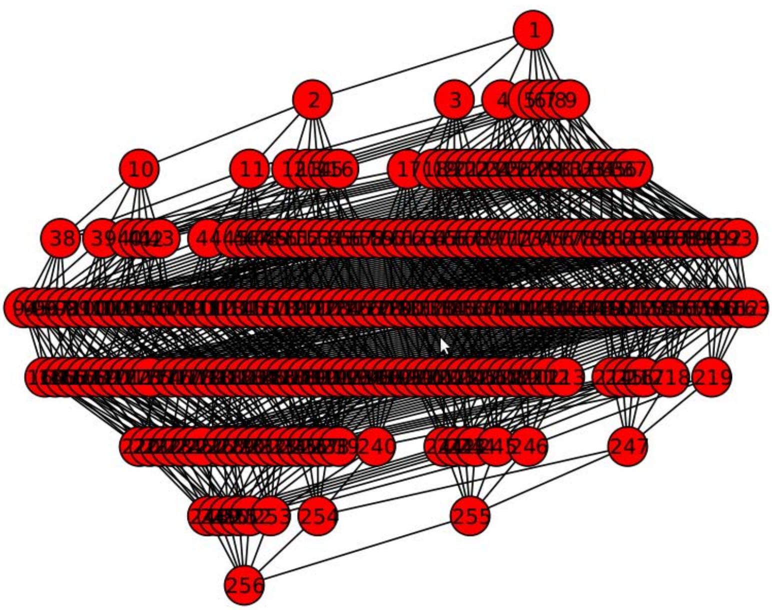

Fig. 3. Lattice Formed By 256 Convex Polyhedrons

G, G is subset of F, F is subset of E, E is subset of D, D is subset of C, C is subset of B and finally B is subset of A. The final lattice generated by using these hierarchical relations, including ambiguous relations is shown in figure 3.

5 Conclusion

Manual construction of 256 convex polyhedrons in a 3X3X3 rectangular grid is cumbersome and highly error prone. One may end up with constructing subsets repeatedly and above all it has been empirically verified that constructing all 256 convex polyhedrons does consume 48 man hours. To overcome this problem, we proposed a new algorithm automatic construction of convex polyhedrons. This algorithm is discussed in the section 2, which generates 255 polyhedron subsets in a fraction of a second.

Secondly, manual construction of paths in the lattice from supremum to infimum is almost impossible task. Each path defines a GE(Evolutionary G-Filter) filter. Each GE filter is of some use in processing 3-D images. So, one comes across the usual problem of enumerating potentially very large GE filters defined over a finite set. To overcome this problem, we proposed in this paper a novel algorithm for Hierarchical Path Enumeration of 256 convex polyhedrons.

Acknowledgments. We sincerely thank Mrs. G.Chitra, Managing Director, Pentagram Research Centre Pvt. Ltd.for her support to carry out research work.

References

1. Gonzalez, R.C., Woods, R.E.: Digital Image processing, 2nd edn. PHI
2. Rajan, E.G.: Symolic Computjng Signal & image processing. B.S. Publications, India
3. Choquet, G.: Topology. Academic Press, New York (1966)
4. Rajan, E.G.: The notion of geometric filters and their use in computer vision. In: IEEE International Conference on Systems, Man and Cybernetics, Vancouver, B.C., Canada, pp. 4250–4255 (1995)
5. Rajan, E.G.: On the notion of a geometric filter and its relevance in the neighbourhood processing of digital imagesin hexagonal grids. In: 4th International Conference on Control, Automation, Robotics and Vision, Westim Stamford, Singapore (1996)
6. Vasantha, N., Rajan, E.G.: On the notion of geometric filters. National Conference organized by the Institution of Engineers, India and Annamalai University, SURGE 1994 (1994)
7. Matheron, G.: Random sets and Integral Geometry. Wiley, New York (1975)
8. Serra, J.: Image Analysis and Mathematical Morphology. Acadamic, New York (1982)
9. Maragos, P., Schafer, R.W.: Applications of morphological filtering to image processings and analysis. In: Proc. 1986 IEEE Int. Conf. Acoust., Speech, Signal Processing, Tokyo, Japan, pp. 2067–2070 (April 1986)
10. Sonka, M., Hlavac, V., Boyle, R.: Image Processing, Analysis and Machine Vision, 2nd edn.
11. Sultana, T., Rajan, E G.: Algebra of Geometric Filters Defined over Three Dimensional Rectangular Lattice-Part I. In: Paper Submitted to This Conference Along with This Paper