M.Tech Project Phase-I Report

On Analysis of Temple Architecture by Isothetic Slicing

Prasad Shetkar
Roll No.: 14CS60R44

Supervisor: Dr. Partha Bhowmick
Computer Science and Engineering Department
Indian Institute of Technology, Kharagpur

November 17, 2015

Certificate

This is to certify that the project entitled “ Analysis of Temple Architecture by Isothetic Slicing ” is a bonafide record of the work carried out by Mr. Prasad Shetkar (Roll No. 14CS60R44) under my supervision and guidance for the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science & Engineering during the academic session 2014-2016 in the Department of Computer Science & Engineering, Indian Institute of Technology, Kharagpur. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Dr. Partha Bhowmick
Associate Professor
Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur, India 721302

Abstract

An isothetic cover of a digital object not only specifies a simple representation of the object but also provides an approximate information about its structural content and geometric characteristics. If a set of horizontal and vertical grid lines is imposed on the object plane, then the outer (inner) isothetic cover is defined by a set of isothetic polygons, having their edges lying on the grid lines, such that the effective area corresponding to the object is minimized (maximized). Document focuses on the algorithm (free of backtracking) to get isothetic cover in linear time and then use orthogonal slicing along all 3 axes to enable us to run this algorithm on each slice and find out structure information in terms of isothetic polygons in each slice. Having done that, it proposes the algorithm to find the overall symmetry of 3D objects particularly Temple Architecture, algorithm to segment it to detect the different identical components of the object. Finally it covers the idea of door detection of temple object by considering combination of the information present in each slice.

Keywords: Connected component, Digital geometry, Digital object, Isothetic cover, Isothetic polygon, Shape analysis, Orthogonal slicing

Contents

1 Introduction

Architecture, as any compositional art, makes extensive use of symmetry. Across all cultures and in all time periods, architectural compositions are symmetrically arranged. With the research in 3D imaging, computer graphics, and computer vision, we can see abundant amount of work being done in shape analysis of the given 3D object. One way of shape analysis is by using orthogonal slicing as described in [2,3]. With kind of shape analysis we do , Objective of our work is to detect the geometrical component and other features of the given 3D Objects.Especially in case of temple object, find its symmetry, door detection etc. and use this analysis to reform the model.

1.1 Research topic and problems

Our research topic is centered around isothetic slicing, outer isothetic cover, shape analysis, and related problems in $\mathbb{Z}^3$. The first work briefed in Section 2 is focused on problem of finding the minimum area cover of 2D object in terms of the isothetic polygon as in def 4. By doing orthogonal slicing and finding the isothetic cover the major task is to combine slicing information along 3 axes and find out the algorithm which can efficiently combine this information to come up with concrete model to reform given model.

1.2 Existing work

Over the last few decades, a multitude of research work has been done in the area of shape analysis like one in [3]. Our problem is related to shape analysis but much beyond that. It is concerned with designing an efficient algorithm that can capture the shape information in such manner which can be utilized easily to analysis the architecture of given 3D object,finding its symmetry and finding the characteristics which can detect geometrical components of it.

1.3 Our work

Our goal is to break down entire 3D model into slices along all three axes.We divide the points set in 3D by using orthogonal slicing along each axes. We explain it in section 2.2 along y axis and same can be extended along x and z. our work is focused on constructing outer isothetic cover in 2D plane for each of the slices. Input specification is in real domain and output generates has only isothetic polygons whose co-ordinates are in integer domain. We concentrate on the problem of designing an efficient algorithm on the top of the output of the first step to extract the valuable information , which is used for segmentation step, find the overall symmetry of the object etc. Hence our work can be summarized in 3 steps for each of the axis

1. Orthogonal slicing

2. Find isothetic covers(isothetic polygon and holes) along each slice
3. Use the output of the second step to do segmentation and symmetry analysis
4. use the output of step 2 and 3 for feature detection.

The report is organized as follows. Section 2 presents the overview of the earlier work in finding isothetic cover of 2D object as described in [1] and its use in analysis of architecture of given object. Section 3 explains segmentation of object in along 3 axes. Section 4 contains the algorithm for finding the overall symmetry of the temple. Section 5 gives the most interesting insight on how we can use the information stored along each slice to detect any features and in particular emphasizes on solution for door location problem for given temple object. The results of all three algorithms are presented in their respective sections. Finally, we conclude in Section 6 with our future scope of work.

2 Orthogonal slicing

We do the slicing along each axis by sorting the given point set along every axis by using the input parameter and run the isothetic cover finding algorithm on each of them.

2.1 Input paratameters

- scale factor: First input parameter is scale factor and it is just to scale the object large enough for output visibility.
- slice size(s): slice size depicts the set of points taken while slicing along any axis. e.g. Let slice size is 5 along y axis .Lets say min y is 3. Then first slice will contain all points having y co-ordinate in [-42,-33).Here point to note is that for all such points in this slice y co-ordinate vanishes and we concentrate only on x and z co-ordinate in 2D plane.
- Grid factor(f): Factor by which we multiply isothetic distance (d) to get appropriate grid size. Let $(d_1, d_2 \ldots d_n)$ be the minimum isothetic distance with respect to n slices along the particular axis. d is $\min(d_1, d_2 \ldots d_n)$.

2.2 Isothetic cover

Following are the definitions of some terms we are going to refer in context of explanation of the all algorithms listed in this document.

Definition 1 (Isothetic distance). *The (isothetic) distance between two points, $P(i_p, j_p)$ and $Q(i_q, j_q)$, is given by $d_T(P, Q) = max(|i_p - i_q|, |j_q - j_p|)$*

Definition 2 (Digital grid)**.** *A digital grid (henceforth referred simply as a grid) $G : H; V$ consists of a set H of horizontal (digital) grid lines and a set V of vertical (digital) grid lines, where, $\{H = ...; l_H(j-2g), l_H(j-g), l_H(j), l_H(j+g), l_H(j+2g); ...\}$ and $\{V = ...; l_V(i-2g), l_H(i-g), l_H(j), l_V(i+g), l_V(i+2g); ...\}$ for a grid size, $g \in \mathbb{Z}^2$.Here, $l_H(i) = \{(i,y) : y \in \mathbb{Z}^2\}$ denotes horizontal grid line and $l_v(j) = \{(x,j) : x \in \mathbb{Z}^2\}$ denotes vertical grid line.*

Definition 3 (Unit Grid Block)**.** *Horizontal line divide $l_H(i)$ divides plane into $l_H^+(i) = \{(x,y) \in \mathbb{Z}^2 : y \geqslant i\}$ and ${l_H}^-(i) = \{(x,y) \in \mathbb{Z}^2 : y \leqslant i\}$ and vertical line divides plane into $l_v(j)$ divide plane into ${l_H}^+(j) = \{(x,y) \in \mathbb{Z}^2 : x \geqslant j\}$ and ${l_V}^-(j) = \{(x,y) \in \mathbb{Z}^2 : x \leqslant j\}$. UGB(i,j) is given by set ${l_H}^+(j) \cap {l_H}^-(j+1) \cap {l_V}^+(i) \cap {l_V}^-(i+1)$.*

Definition 4 (Isothetic Polygon)**.** *An isothetic polygon P is a simple polygon (i.e., with non-intersecting sides) of finite size in $\mathbb{Z}^2$ whose alternate sides are subsets of the members of H and V .The polygon P, hence given by a finite set of UGBs, is represented by the (ordered) sequence of its vertices, which are grid points.Refer Figure 2*

Definition 5 (Outer Isothetic Cover)**.** *The outer (isothetic) cover (OIC), is a set of outer polygons and (outer) hole polygons, such that the region, given by the union of the outer polygons minus the union of the interiors of the hole polygons, contains a UGB if and only if it has object occupancy (i.e., has a nonempty intersection with S).*

In context of our work, we need only to focus on the outer cover and detection of holes as in section 2.6.

2.3 Types of grid points

There are 3 types of grid points present in each polygon exits

1. Type 1: internal angle is 90^0
2. Type 2: internal angle is 180^0
3. Type 3: internal angle is 270^0

2.4 Filling grid

We impose hypothetical grid on the object in the consideration and create Filled grid object. Algorithm 1 explains slicing along y and filling the grid. Same idea can be used for x and z slicing.

For each point in the slice we divide its x and y co-ordinates by gridsize to form the index of the Unit Grid block(UGB) in which that point will reside. If cell(UGB) contains point then that cell is set to true. For each slice it takes operations equal to number of points in that slice and total number of UGBs in that filled Grid. Let S_i has filled grid

Algorithm 1: Fill grid

```
Input: 3D point set:$P$ with size $n$, slice size $s$ , gridsize $g$
Output: Array of Filled Grids(with respect to each slice)
sort $P$ along y axis
$u \leftarrow P[0].y + s$
$G \leftarrow \{\}$ (set of filled grids)
$S \leftarrow \{\}$
$n1 \leftarrow 0$, $n2 \leftarrow 0$
int $minx \leftarrow \infty$ , $maxx \leftarrow -\infty$ , $miz \leftarrow \infty$ , $maxz \leftarrow -\infty$
for each $p$ in $P$ do
    if $p.y < u$ then
        $S \leftarrow S \cup p$
        $minx \leftarrow min(minx, p.x)$
        $maxx \leftarrow max(maxx, p.x + 1)$
        $mizz \leftarrow min(minz, p.z)$
        $maxz \leftarrow max(maxz.p.z + 1)$
    end
    else
        int $n1 = maxx - minx + 1$ , $n2 = maxz - minz + 1$
        $g \leftarrow bool[n1][n2]$ ( $n1Xn2$ boolean array )
        for $i \leftarrow 0$ to $n1$ do
            for $j \leftarrow 0$ to $n2$ do
                $g[i][j] \leftarrow false$
            end
        end
        for each $p$ in $S$ do
            int $i \leftarrow p.x/g$, $j \leftarrow p.z/g$
            $g[i][j] = true$
        end
        $G \leftarrow G \cup g$ $S \leftarrow \{\}$
        $u \leftarrow u + s$
        $minx \leftarrow \infty$ , $maxx \leftarrow -\infty$ , $miz \leftarrow \infty$ , $maxz \leftarrow -\infty$
    end
end
return $G$
```

with total UGBs C_i.Let number of slices be m. Let n be the total number of points of the object.

Time complexity is $O(n) + \sum_{i=1}^{m} C_i$.

Algorithm 2: Outer Isothetic Cover

```
Input: Filled Grid $G$ and start point $s0$
Output: Isothetic polygon $P$ is list of grid points in anticlockwise manner
1  $P \leftarrow \{\}, d \leftarrow 1$
2  $p1 \leftarrow GETNEXT(G, p0, d)$ while $p1 \neq p$ do
3      if $p1.type = 1$ —— $p1.type = 3$ then
4          $P \leftarrow P \cup p1$
5      end
6      $p1 \leftarrow GETNEXT(G, p1, d)$
7      if $p1.type = 3$ then
8          $d \leftarrow (d + 1) mod 4$
9      else
10         if $p1.type = 1$ then
11             $d \leftarrow 4 + d - 1 mod 4$
12         end
13     end
14 end
15 return $S$
```

Algorithm 3: GETNEXT

```
Input: Filled Grid $G$ ,grid point $p$, direction $d$
Output: next grid point in anticlockwise traversal of polygon boundary
1  $dx \leftarrow \{0, 1, 0, -1\}$ , $dz \leftarrow \{1, 0, -1, 0\}$
2  $p1.x = p.x + dx[d], p2.z = p.z + dz[d]$
3  Determine type of p1 by finding all combinations of 4 UGBs incident on p1
4  return $p1$
```

2.5 Single connected object

For Finding the Outer isothetic cover, we used the following algorithm 2.We first find the start point in the slice while filling grid itself. It is the UGB where the bottommost and left most point of the object lies.

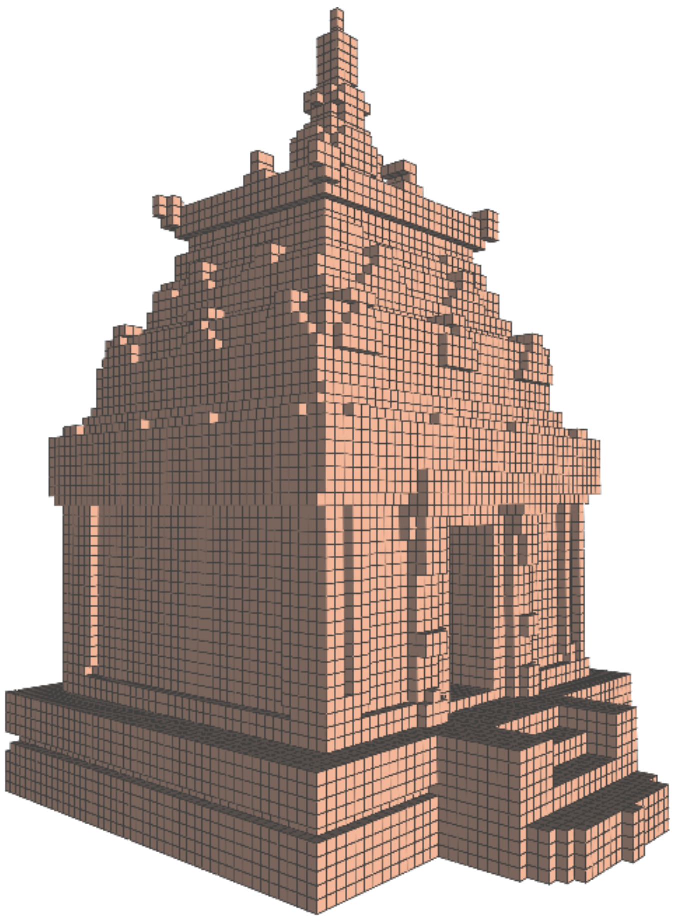

Figure 1: voxel-visualization of the point cloud(standard model) used as input

2.6 Finding holes and disconnected component

During the traversal in algorithm in section 2.5 we mark the cells along the path as visited. We start traversing entire filled grid from leftmost bottom most point of the grid. Each time we find the filled UGB and not visited, We traverse such unvisited hole or component by using Algorithm in figure 2. Each time we get the new polygon(hole or normal outer polygon) we store it by maintaining the structure for each slice s. We repeat the procedure till we reach uppermost rightmost UGB of the filled grid. Let there are m isothetic polygon including holes in the isothetic cover of particular slice. Let P_i be the perimeter of the i^{th} polygon. Let g be the grid size. Let C be the total UGBs of the filled grid with respect to particular slice(s). For that slice(s), it takes $(C + \sum_{i=1}^{m} P_i/g)$ time.Please refer the figure 2.

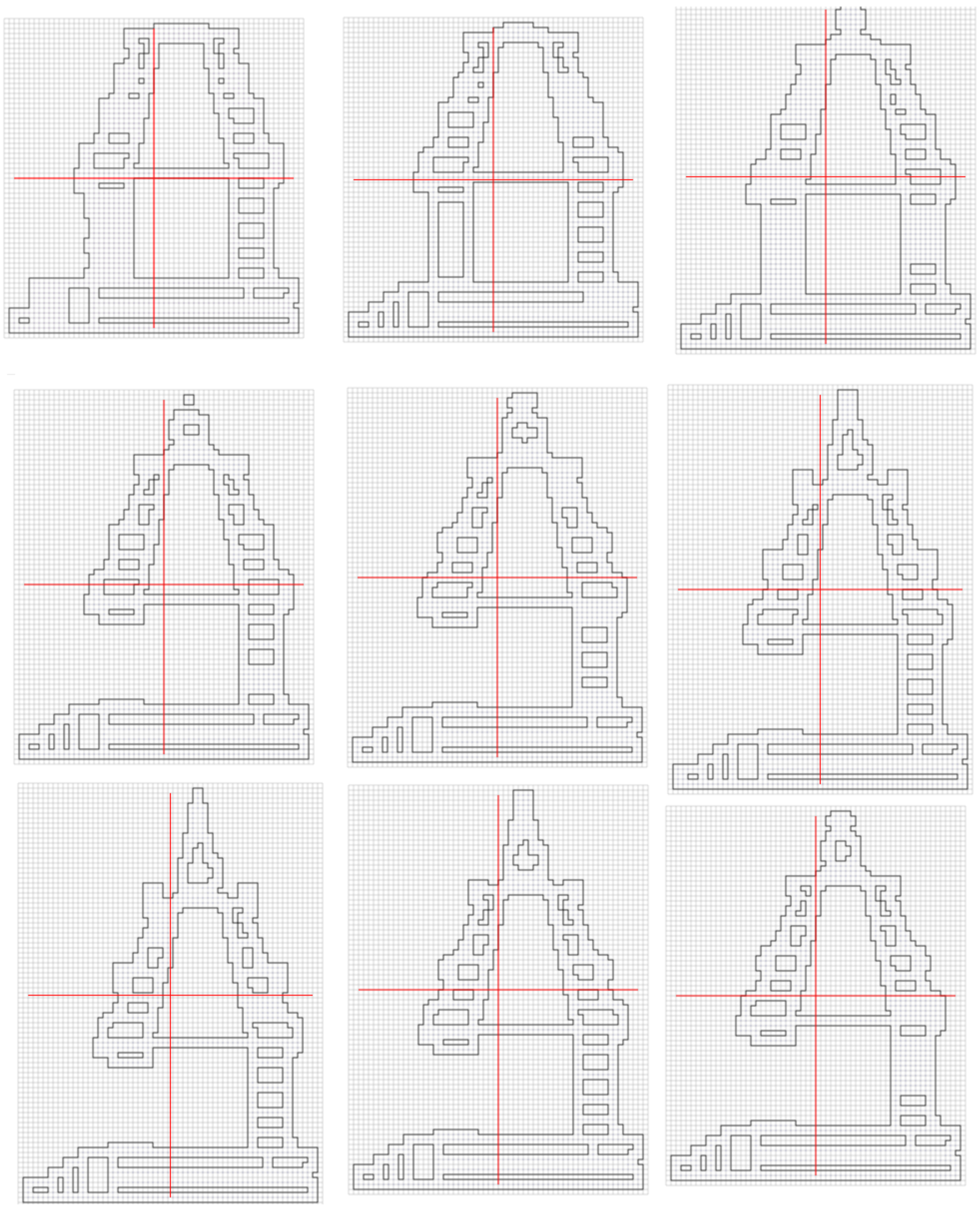

Figure 2: Slice along zy, grid size = 5, from left to right(and then moving down) 1. x∈[-42,-33)

2.7 Slice and polygon structure

Each slice s we are referring to, contains following information

1. List of all the polygons in that slice, denoted L.

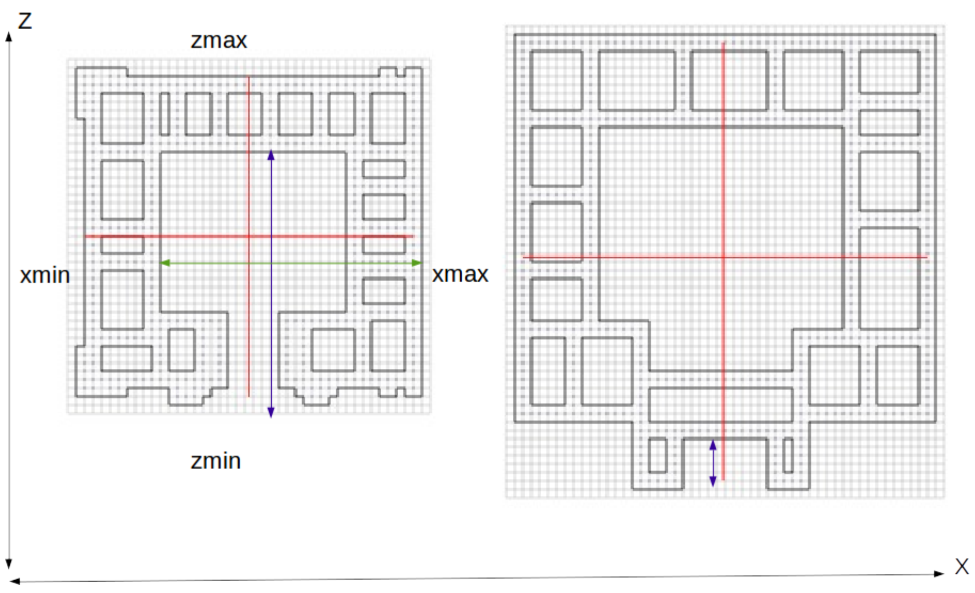

Figure 3: green line for xdepth and blue line zdepth, slice on right has no xdepth

2. Outer most polygon L_Poly

Each polygon P we are referring to, contains following information(explained with respect to y axis)

1. List of all grid points stored in anticlockwise direction.
2. Perimeter of polygon p
3. $xmin$ and $xmax$ are minimum and maximum x values along the boundary of polygon
4. $zmin$ and $zmax$ are minimum and maximum z values along the boundary of polygon
5. $xdepth$ refer figure 3 Find out 2 consecutive type 3(with same x)such that next point has x greater. Take all such pair and find the differences of x of such points and xmax. Find out 2 consecutive type 3(with same x) such that next point has x smaller. Take all such pair and find the differences of xmin and x of such points. Take the maximum of all such differences as $xdepth$
6. $zdepth$ refer figure 3. We can find it in same way as $xdepth$

Algorithm 4: Segmentation

```
Input: Set $G$ containing $n$ slices along y axis and factor $f$(for comparison)
Output: $S$ containing all segments along y
1  $S \leftarrow \{\}$ $s1 \leftarrow \{\}$ for each slice $s$ in $G$ do
2      if $s1$ is empty then
3          $s1 \leftarrow s1 \cup \{s\}$
4      end
5      else
6          $l$ be the last slice in $s1$
7          $r \leftarrow s.Lpoly.p/l.Lpoly.p$
8          $r1 \leftarrow s.Lpoly.xdepth/l.Lpoly.xdepth$
9          $r2 \leftarrow s.Lpoly.zdepth/l.Lpoly.zdepth$
10         if $(r \leqslant f \vee r \leqslant 1/f) \wedge ((r1 \leqslant f \vee r1 \leqslant 1/f) \wedge (r2 \leqslant f \vee r2 \leqslant 1/f))$ then
11             $s1 \leftarrow s1 \cup \{s\}$
12         end
13         else
14             $S \leftarrow S \cup \{s1\}$
15             $s1 \leftarrow \{\}$
16         end
17     end
18 end
19 return $S$
```

3 Segmentation

Segmentation in this context means grouping of the slices which have common geometrical structure. Such segmentation can enable us to determine the part of object where we can search for particular type of geometrical component of the object.Algorithm 4 explain the segmentation procedure for the given set of slices we get in section 2.6. Here the factor f is used enables us to get better segmentation. As we vary f results are bound to change and we have shown the result in figure 4 is with f set to 1.2.

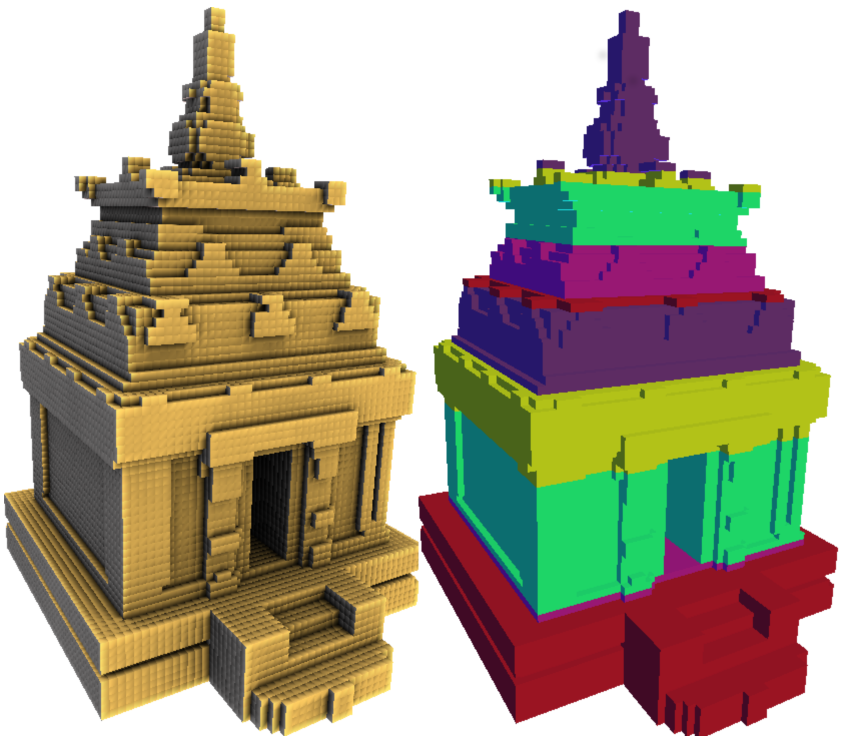

Figure 4: segmentation of standard model from left to right 1.model 2.along yz plane

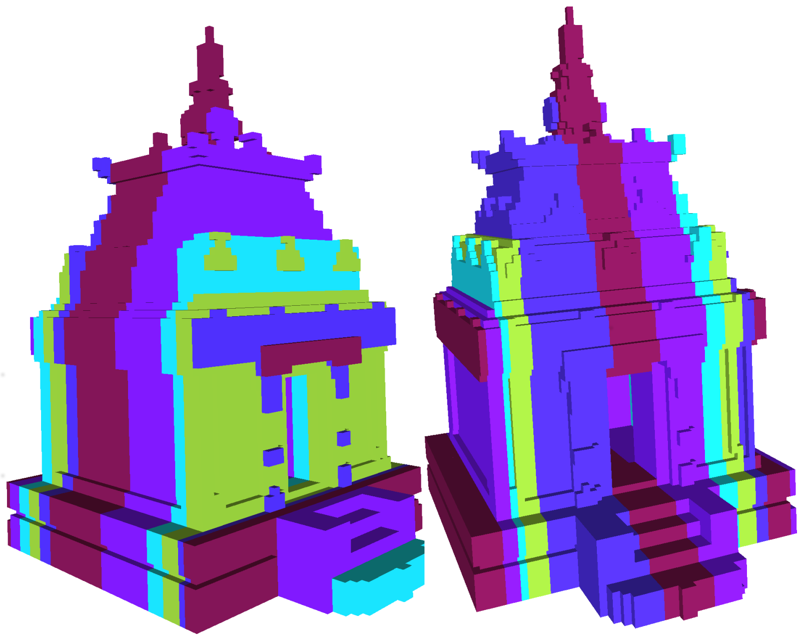

Figure 5: segmentation of real model from left to right along 1.xy 2.yz

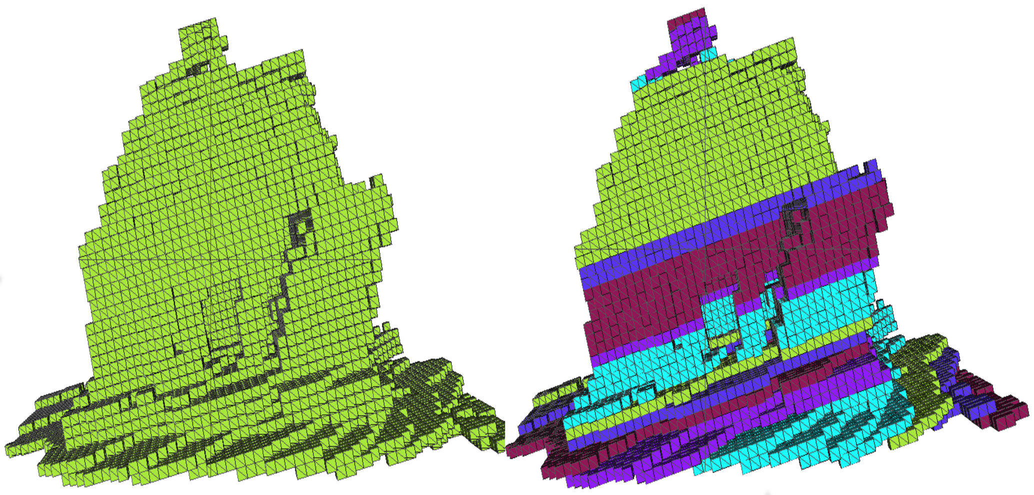

Figure 6: segmentation of real model from left to right 1.model 2. along xz plane

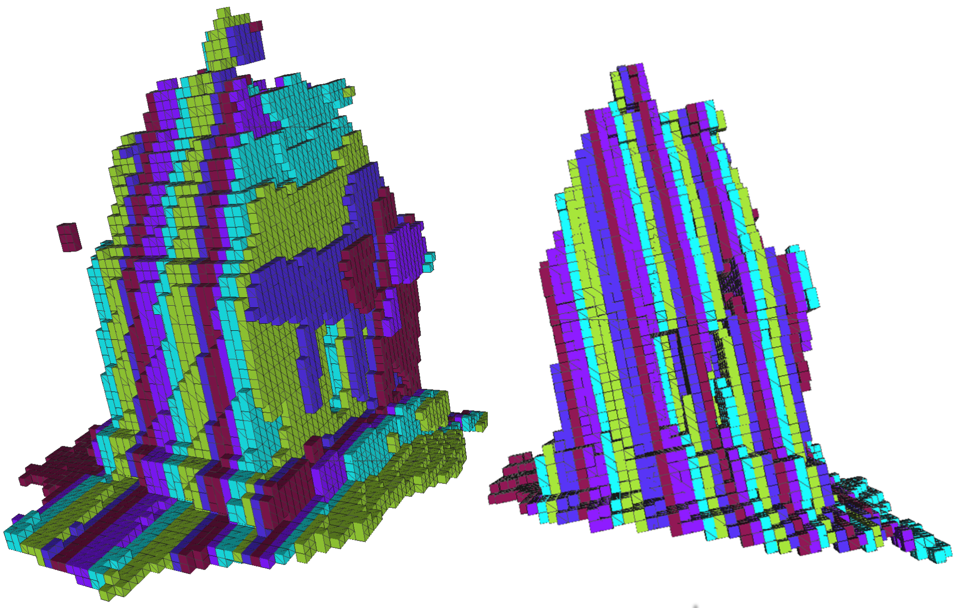

Figure 7: segmentation of real model from left to right along 1.xy 2.yz

Algorithm 5: Slice x-symmetry Detection

```
   Input: slice $s$ along y axis and parameter $c$
   Output: $xsym$ is x symmetry value for the slice $s$
1  Sort List $L$ of polygons in $s$ by non-increasing order of their perimeter $p$
2  $xsym \leftarrow 0$
3  Find first polygon $L[j]$ such that, $L[j].p < L[j-1].p/c$
4  $total \leftarrow 0$
5  for $i \leftarrow 1$ to $j-1$ do
6      $total \leftarrow total + L[i].p$
7  end
8  for $i \leftarrow 1$ to $j-1$ do
9      $xsym \leftarrow xsym + L[i].p * (L[i].xmax - L[i].xmin)/total$
10 end
11 $xsym \leftarrow xsym/j$
12 return $xsym$
```

4 Symmetry detection

Symmetry detection can be divided into 2 parts, first is to detect symmetry for each slice and then combine the symmetry of all slices to get the entire symmetry of temple. Specifically we use the slices along y axis to detect the symmetry of temple along x axis and z axis.

4.1 Slice symmetry detection

The idea is that we consider only those polygon in slice which are large enough in the current context that is in the correct slice and take weighted mean of the symmetries of only those polygons. The questions are what is large enough means and what is the weight.To answer the second question it is the perimeter of the polygon which contributes as weight , and we consider the polygons in the non-increasing order of their perimeter and whenever the perimeter of the current polygon is less than *half* of the previous we discard the significance of all the polygons from that polygon in the contribution of the symmetry.This half can also be customized parameter to be dependent upon particular object and hence we keep it as input parameter c. We have shown the results in figure 8 with $c = 2$. Algorithm 5 explains how to find the x symmetry point for slice along y. Same can be extended to find z symmetry.

4.2 Temple symmetry

Overall x symmetry of the temple can be determined by taking mean of the x symmetry values of all the slices calculated in section 4.1 and same for z symmetry.

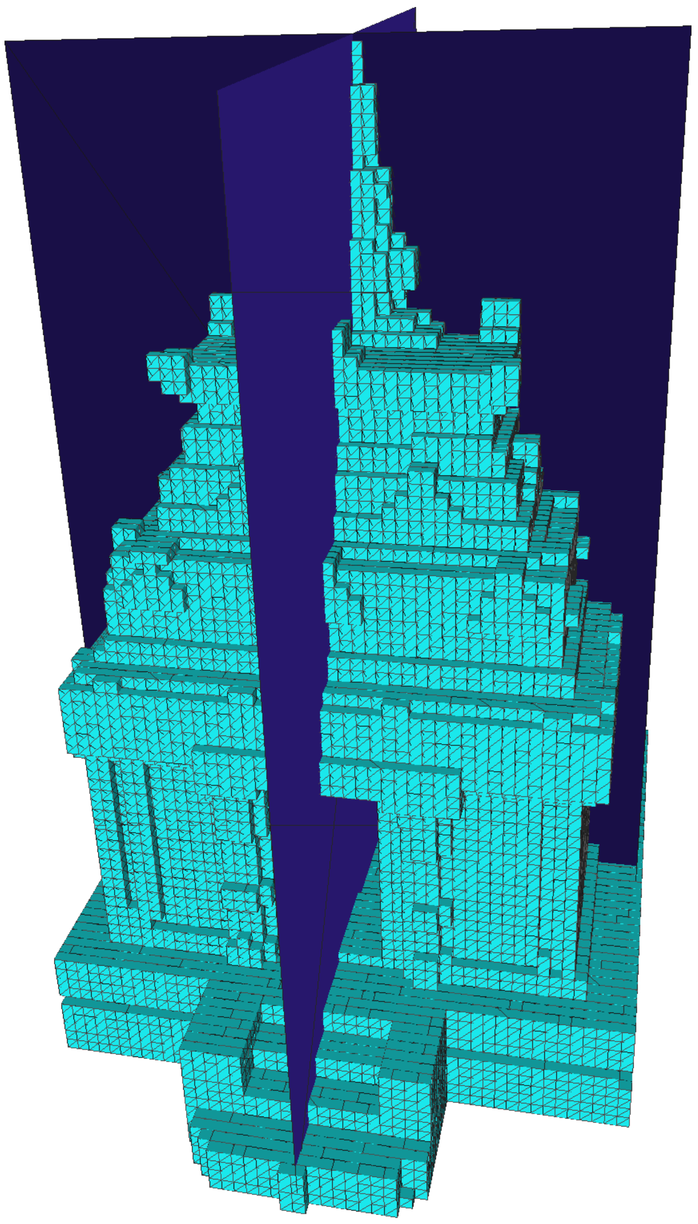
Figure 8: Symmetry Planes for x and z are shown in blue colors

5 Door detection

To locate the door in the temple object we use the algorithm 6. Results are shown in figure 9. We find the 2 consecutive 1 type grid points in the given slice along xz plane such that previous and next grid points have x either small(greater) than current points. We find such pairs where x is greatest(smallest). Such 2 points are probable candidates for door points. We find the continuous slices which have such pairs and combine them to

```
Algorithm 6: Door Points Detection
Input: Isothetic Polygon point Set:$P$ with size $n$
Output: Array containing indexes of doopoints in $P$
$u \leftarrow P[0].y + s$
$S1 \leftarrow \{\ \}$ , $S2 \leftarrow \{\ \}$ , $S \leftarrow \{\ \}$
for each $p$ in $P$ do
    all point in $P$ are stored in anticlockwise fashion
    if $p.type = 1 \wedge p.nxt.type = 1 \wedge p.x = p.nxt.x \wedge p.x < p.prv.x \wedge p.x < p.nxt.nxt.x$ then
        (/* finding minimum x point */)
        if $S1$ contains a point $p1$ and $p1.x > p.x$ then
            $S1 \leftarrow \{\}$, $S1 \leftarrow \{p\} \cup \{p.next\}$
        end
        if $p1.x = p.x$ then
            $S1 \leftarrow \{p\} \cup \{p.next\}$
        end
    end
    if $p.type = 1 \wedge p.nxt.type = 1 \wedge p.x = p.nxt.x \wedge p.x > p.prv.x \wedge p.x > p.nxt.nxt.x$ then
        (/finding maximum x points*/)
        if $S2$ contains a point $p1$ and $p1.x < p.x$ then
            $S2 \leftarrow \{\}$, $S2 \leftarrow \{p\} \cup \{p.next\}$
        end
        if $p1.x = p.x$ then
            $S2 \leftarrow \{p\} \cup \{p.next\}$
        end
    end
end
if $S1$ and $S2$ both contains more than 2 points then
    Add first and last point from both $S1$ and $S2$ in $S$
end
return $S$
```

get the door location.

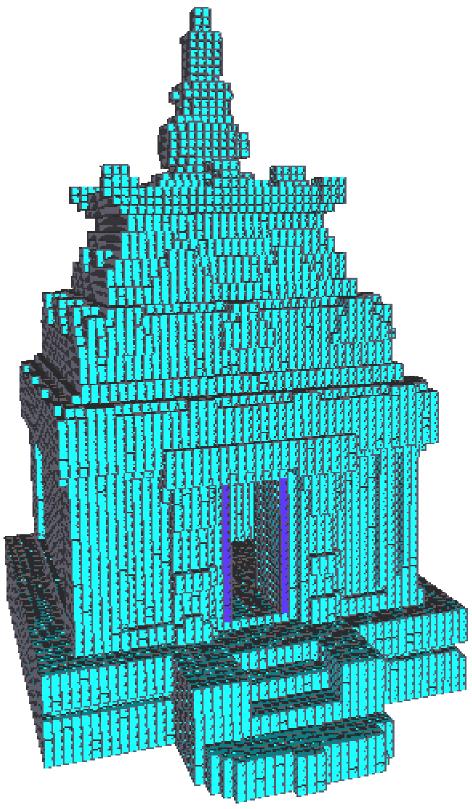

Figure 9: Door detected in standard model shown in blue color

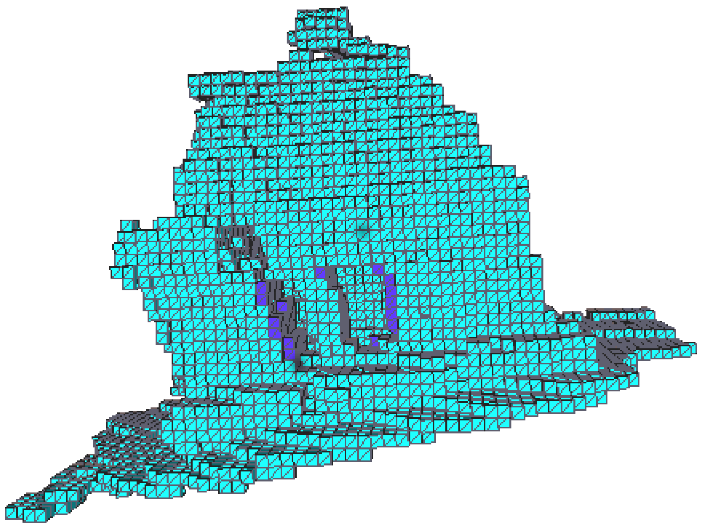

Figure 10: Door detected in real model

6 Future works and conclusion

With the kind of work we have done and results we have got are much more interesting.We are planning to move forward in the same direction and combine the results along all 3 dimensions to detect more features of the object. But challenges are there with respect to real world model as we have seen. So we will be working more on real data and try and build more accurate system.

Bibliography

[1] A. Biswas, P. Bhowmick, and B. B. Bhattacharya. Construction of isothetic covers of a digital object: A combinatorial approach. *Journal of Visual Communication and Image Representation*, 21(4):295–310, 2010.

[2] N. Karmakar, A. Biswas, P. Bhowmick, and B. B. Bhattacharya. A combinatorial algorithm to construct 3d isothetic covers. *International Journal of Computer Mathematics*, 90(8):1571–1606, 2013.

[3] B. B. Kristian and M. Alexa. Orthogonal slicing for additive manufacturing. In *Shape Modeling International (SMI) Conference 2013*, 2013.