Center for Global Design and Manufacturing

UNIVERSITY OF Cincinnati

Computational Methods of Additive Manufacturing – Spring 2020

STL SLICING ALGORITHM FOR ADDITIVE MANUFACTURING

Prof. Sam Anand

Background

B-Rep

- **B-Rep** is an abbreviation for Boundary representation
- Based on topological notion that an object is bounded by set of faces with given vertices and edges
- Supported auxiliary information can also include color, dimensions etc.

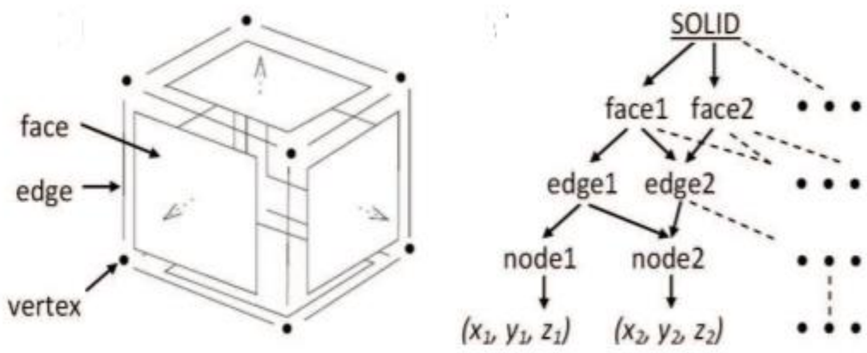


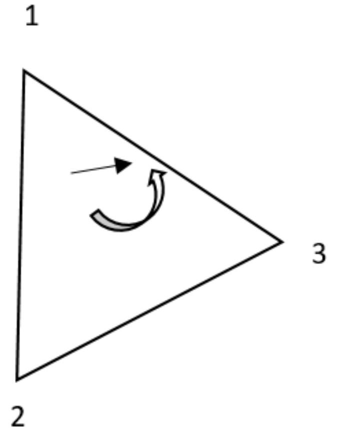
Illustration of B-Rep [1]

Stereolithography file for additive manufacturing is based on B-Rep (ASCII or Binary) code

[1] Ruding Lou. Modification of semantically enriched FE mesh models. Computer Aided Engineering. Arts et M´etiers ParisTech, 2011. English.

STL File Format

- STL is also called as standard tessellation language or standard triangulated language.
- STL is a B-Rep model and consists information of the faces, vertices and normal for triangular facets either in ASCII (readable) or Binary (bits) format.
- The stereolithography process was the first process where STL files were used as input CAD to transfer to machine. Following thereafter, all the AM machines used the similar format files to print parts.
- Surface details of the model are represented using STL with tessellated triangles (which is only required data for printing the parts layer wise)
- For machine to understand the necessary surface information and ease of converting data in G-code, STL is de-facto standard



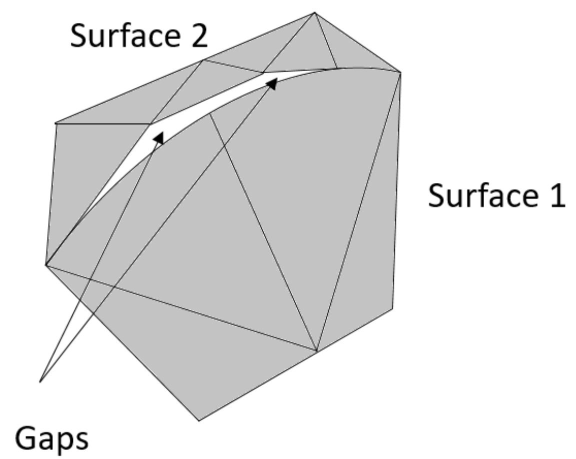

```
facet normal x y z
      outerloop
            Vertex x1 y1 z1
            Vertex x2 y2 z2
            Vertex x3 y3 z3
      endloop
endfacet
```

STL ASCII format for one triangular facet

STL Limitations

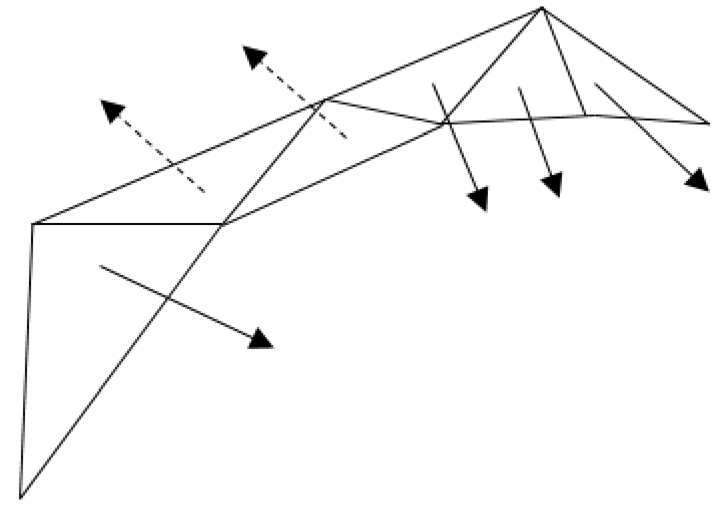
1. Gaps created in STL conversion

Due to triangular facets, the adjacent vertices of different facets may not join creating a gap and defect.



2. Inconsistent / Inverted Normals

The STL format specifies normal point outward according to the right-hand thumb rule, but the facets which direct inward result in inconsistent normal.



3. Strict Vertex to Vertex Rule

The adjacent faces should share two same vertices for water-tight solid model, but in some cases the points may be apart due to rounding error etc.

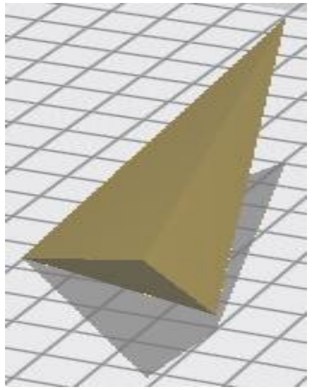
STL Limitations

4. Large file size of STL format:
The ASCII format is larger size compared to Binary format, even though other formats such as PLY or OBJ.

5. **Missing Information from STL:** Sometimes, the geometry primitive's connectivity is missing resulting triangles floating in 3D space.

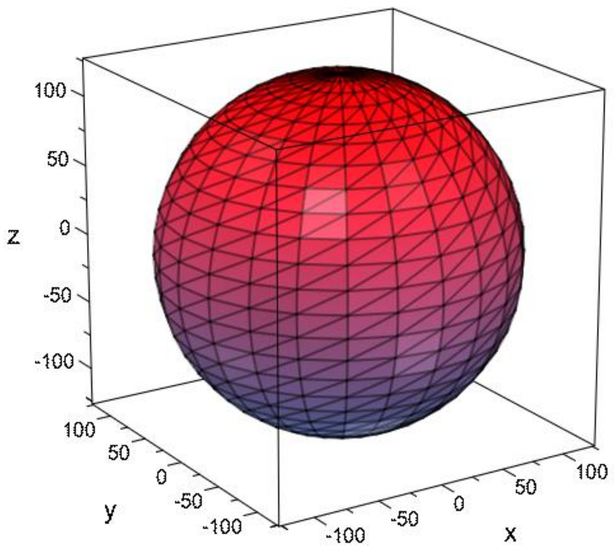
Methods such as adding adjacency matrix or adding vertex table / triangle connect table to record helps to retrieve the data.

	Vertex 1			Vertex 2			Vertex 3		
Triangle 1	X1	Y1	Z1	X2	Y2	Z2	X3	Y3	Z3
Triangle 2	X4	Y4	Z4	X5	Y5	Z5	X6	Y6	Z6
Triangle 3	…	…	…	…	…	…	…	…	…
Triangle 4	…	…	…	…	…	…	…	…	…

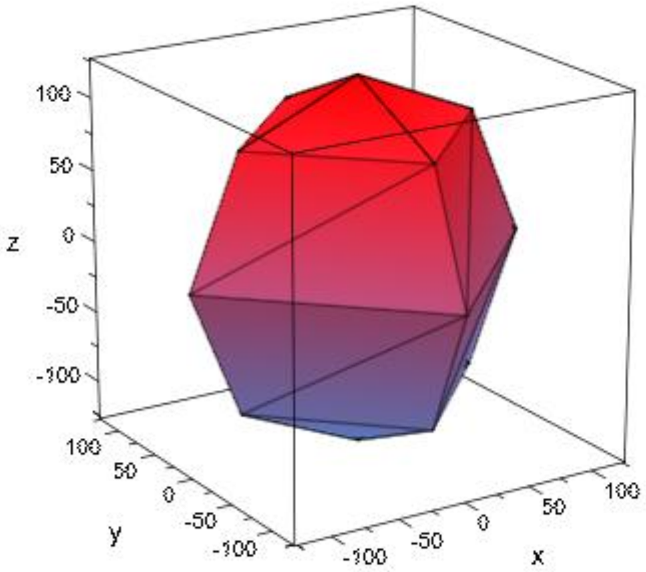


STL Limitations

6. Geometry distortion with change in number of triangular facets



STL for 10’’ diameter sphere with 1152 facets

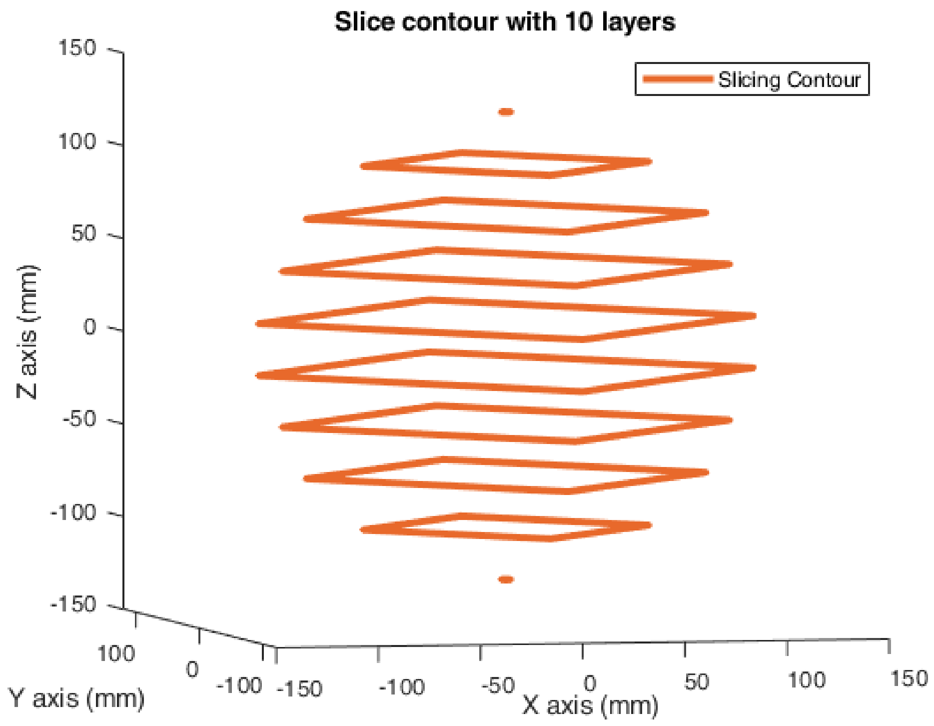


STL for 10’’ diameter sphere with 32 facets

STL Slicing

Key Steps:

- Reading and extracting all vertices and normals
  - In-built Function from programming language or from ASCII text file
- Finding faces intersecting the given slice plane
  - STL facets with non-repeated intersection: Plane passing through two edges of triangle
  - STL facets with repeated intersection: Plane passing through one vertex and opposite edge of triangle
  - STL facets with only one intersection: Plane passing through one vertex of each of the two triangles
- Contour construction algorithm



STL Slicing

Step 1

- Reading and extracting all vertices and normals
  - Read STL file generated from NX using "stlread" function in Matlab

```
[F,V,N] = stlread('part.stl');
```

Returns Faces, Vertices and Normals

F	142x3 double
N	142x3 double
V	426x3 double

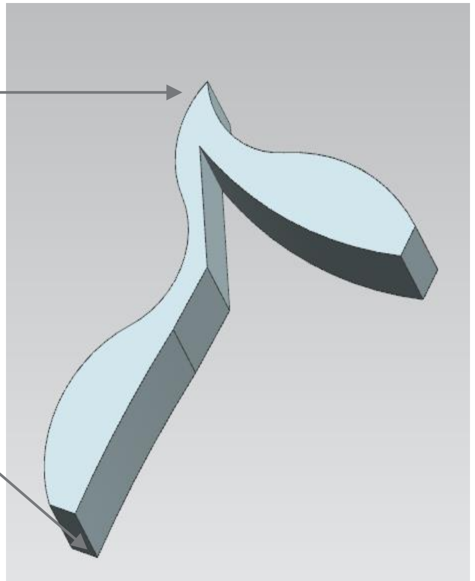
  - Output will be Faces,Vertices & Normals information.
  - For Python, numpy-stl library can be used

https://www.mathworks.com/matlabcentral/fileexchange/22409-stl-file-reader

STL Slicing

Step 2

- Find out the maximum and minimum values of Z among all the vertices to know the Z limits to slice the STL.
    - Z_max = max (STL Z co-ordinates)
    - Z_min= min(STL Z co-ordinates)



- The purpose is to find distance between each slicing layer
    - Distance = (Z_max-Z_min) / (Number of Layers)

- Take the Z values (between Z_min & Z_max) at which slicing is to be performed from user.

STL Slicing

Step 3

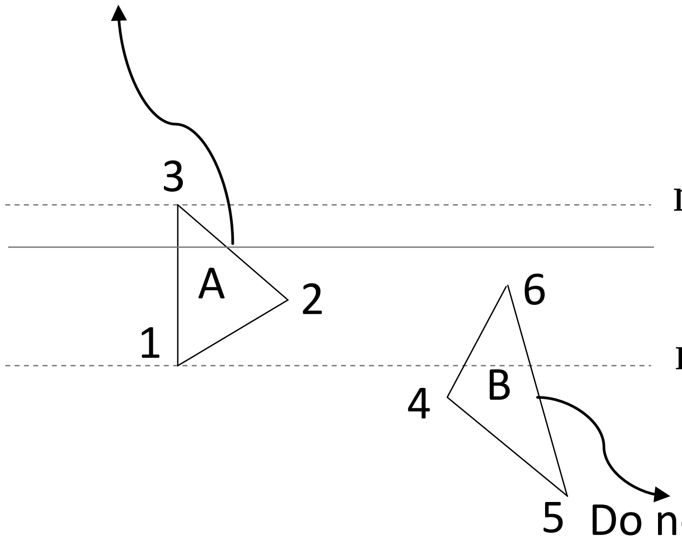
- Find all the facets for which the slicing height (Z-Coordinate of current slice) lies in between the Z-limits of 3 vertices.

→ for all STL facets

at given facet Z co-ordinates (Z_1, Z_2, Z_3)

if min (Z_1, Z_2, Z_3) <= Z_slice (current slicing height) < = max (Z_1, Z_2, Z_3)

consider this facet to check for slice intersection



STL Slicing

Step 4

Finding the intersection points of Slicing plane with the STL facets

- Check for intersection of all three edges of each facet lying at the Z height of the slicing plane.

- Z co-ordinate of the intersected points (Z-slicing plane) – Known

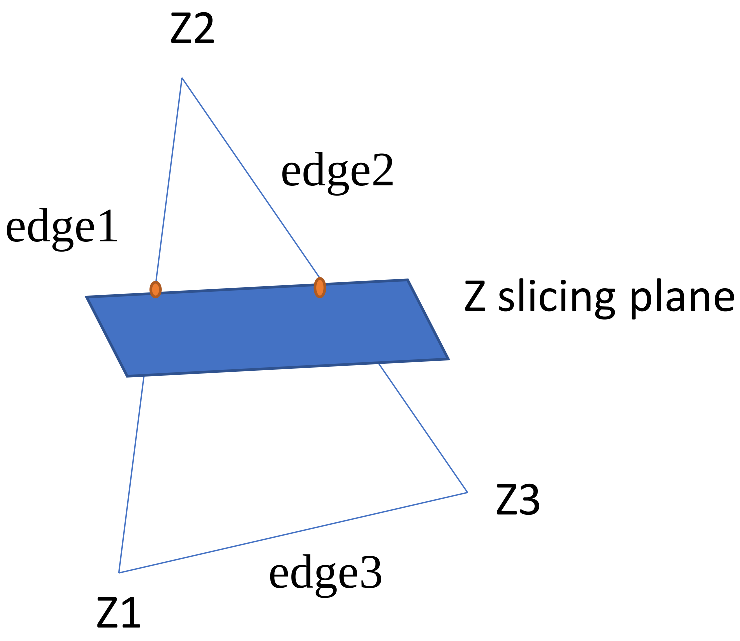
-- equation of line1 (edge1) is

r = vertex 1 + t_1*(vertex 2 – vertex 1)

$$Z_{\text{slicing plane}} = Z_1 + \boxed{t_1} * (Z_2 - Z_1)$$

✓ ✓ ✓ ✓

$$\rightarrow t_1 = (Z_{\text{slicing plane}} - Z_1) / (Z_2 - Z_1)$$



STL facet with non repeated intersections

Similarly t2 & t3 for other two edges of the each STL facet can be calculated.

STL Slicing

Step 5

Finding the intersection points of Slicing plane with the STL facets

After finding t1, t2 and t3; substitute value back in equation of line

For example Edge 1

Only if $t1>=0$ & $t1<=1$

equation of line1 (edge1) is $r = r_1 + t_1*(r_2 - r_1)$

$X_{\text{slicing plane edge 1}} = X_1 + t_1*(X_2 - X_1)$

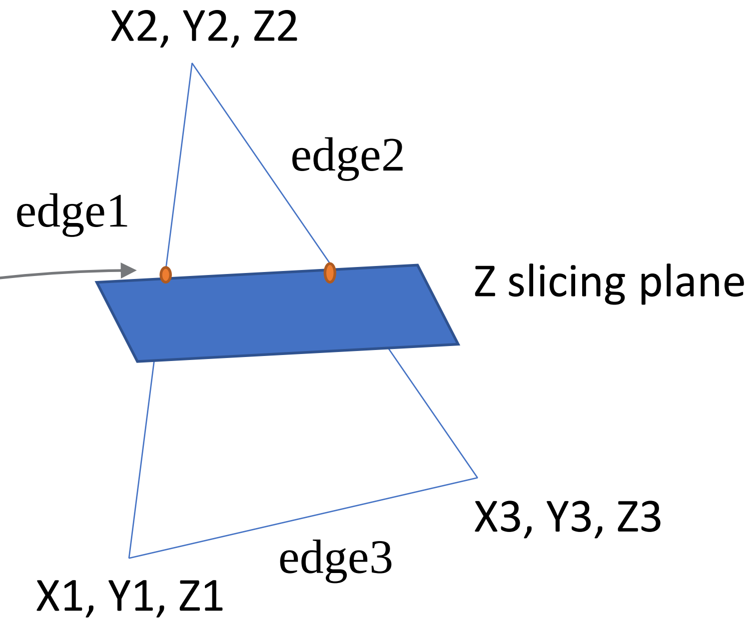
$Y_{\text{slicing plane edge 1}} = Y_1 + t_1*(Y_2 - Y_1)$

--Intersection point = r

$(X_{\text{slicing plane edge 1}}, Y_{\text{slicing plane edge 1}}, Z_{\text{slicing plane edge 1}})$

Similarly for Edge 2:

$(X_{\text{slicing plane edge 2}}, Y_{\text{slicing plane edge 2}}, Z_{\text{slicing plane edge 2}})$



Note, for this case t3 will not be between 0 and 1,
So intersection point for edge 3 is not calculated

STL Slicing

Step 6

Special case 1
Facets with repeated intersection points

Example: From Figure a,

edge 1 at t1 =0 (point 1), because
$Z_{\text{slicing plane 1}} = Z1 + t1\,(Z2-Z1)$
but $Z_{\text{slicing plane 1}} = Z1$, therefore t1=0

edge 2 at t2 = a (point 2)

edge 3 at t3 = 1 (point 3), because
$Z_{\text{slicing plane 3}} = Z3 + t3\,(Z1-Z3)$
but $Z_{\text{slicing plane 3}} = Z1$, therefore t3=1

Since point 1 is equal to point 3

So consider only points that are unique for each facet

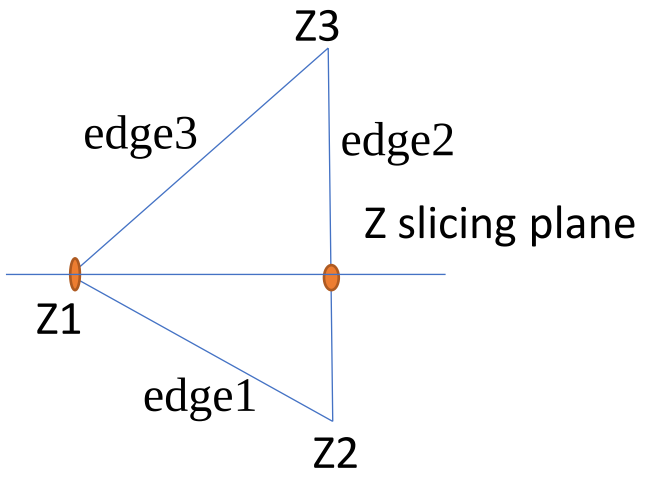


Fig a: STL facet with repeated intersections

STL Slicing

Step 6

Special case 2
STL facet with one intersection

- Consider only facets with intersections greater than 1 because for facets with only one intersection there will be adjacent facets having the same intersected point as shown in Fig b (intersection plane passing through vertices).

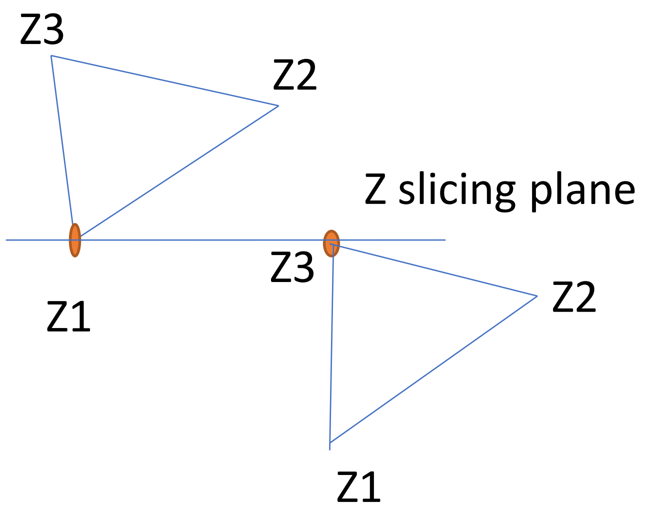


Fig b: STL facet with one intersection

STL Slicing

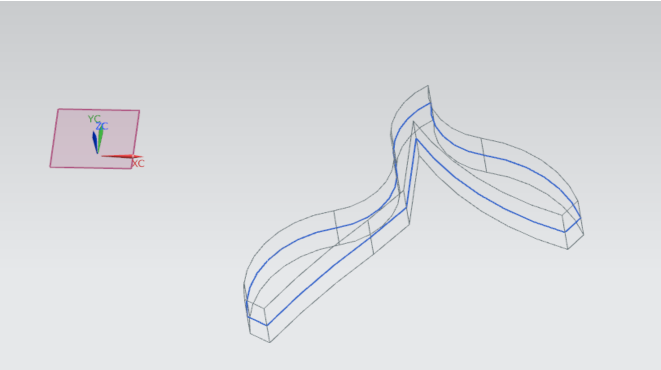
Step 7 Contour Construction

- Arrange all the intersected points in the sequence to form a contour.

-- if facet1 intersected points are point1 , point2
-- if facet2 intersected points are point3 , point4

- Compare last point of intersection in facet 1 with rest of facets

-- if any one point matches (say point 2 == point3)
-- Contour points = [point1 point2 point 4]



CAD slicing in NX

Example

Intersected points of facets at that Z slice plane

final intersected points =

[[-10,-16.8 ; -10,0]
[-10,-20 ; -10,-16.8]
[10,-3.2 ; 10,0]
[10,-20 ; 10,-3.2]
[6.8,-20 ; 10,-20]
[-10,-20 ; 6.8,-20]
[-10,0 ; 6.8,0]
[6.8,0 ; 10,0]]

final intersected points =

[[-10,-16.8 ; $\boxed{\text{-10,0}}$]
[-10,-20 ; -10,-16.8]
[10,-3.2 ; 10,0]
[10,-20 ; 10,-3.2]
[6.8,-20 ; 10,-20]
[-10,-20 ; 6.8,-20]
$\boxed{\text{[-10,0 ;}}$ 6.8,0]
[6.8,0 ; 10,0]]

1
2
3
4
5
6
7

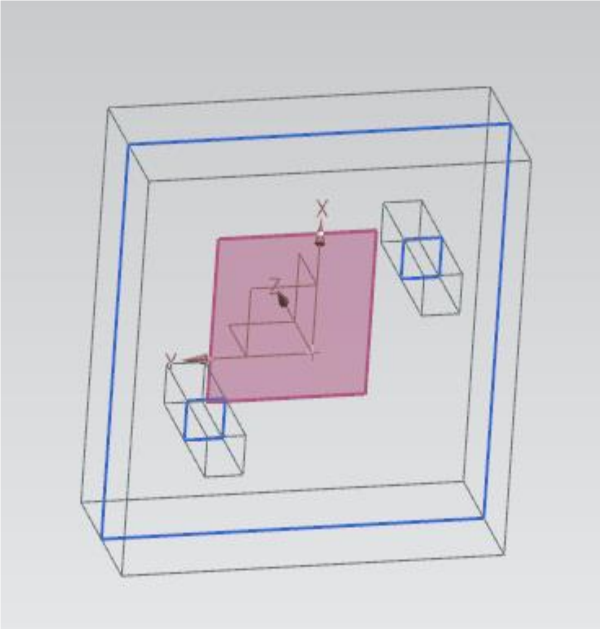
final contour points =

-10, -16.8
-10, 0
6.8, 0
10, 0
10, -3.2
10, -20
6.8, -20
-10, -20
-10, -16.8

STL Slicing

Parts with multiple contours

- If the last point in Contour points doesn't match with any of the intersected points
  - Create new array of contour points to form a new loop and follow the same procedure with the remaining facet intersections



CAD slicing in NX for part with multiple contours

STL Slicing

intersected points =

[0, 40; 0, 50]
7 [0, 0; 0, 40]
[0, 0; 10, 0]
[10, 0; 50, 0]
[50, 0; 50, 10]
[50, 10; 50, 50]
2 [40, 50; 50, 50]
[0, 50; 40, 50]

6
5
1
4
3

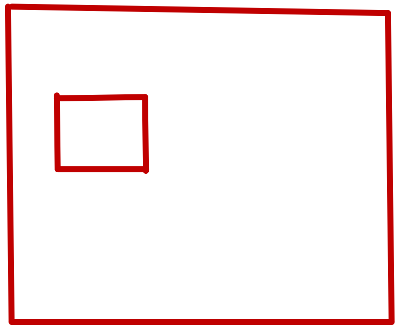
[5.5, 23.34; 5.5, 31.34]
7 [5.5, 21.34; 5.5, 23.34]
6 [5.5, 21.34; 13.5, 21.34]
[13.5, 21.34; 15.5, 21.34]
[15.5, 21.34; 15.5, 29.34]
[15.5, 29.34; 15.5, 31.34]
3 [5.5, 31.34; 13.5, 31.34]
[13.5, 31.34; 15.5, 31.34]

5
1
4
2

contour 1

0	40
0	50
40	50
50	50
50	10
50	0
10	0
0	0
0	40

No match with rest of contour points



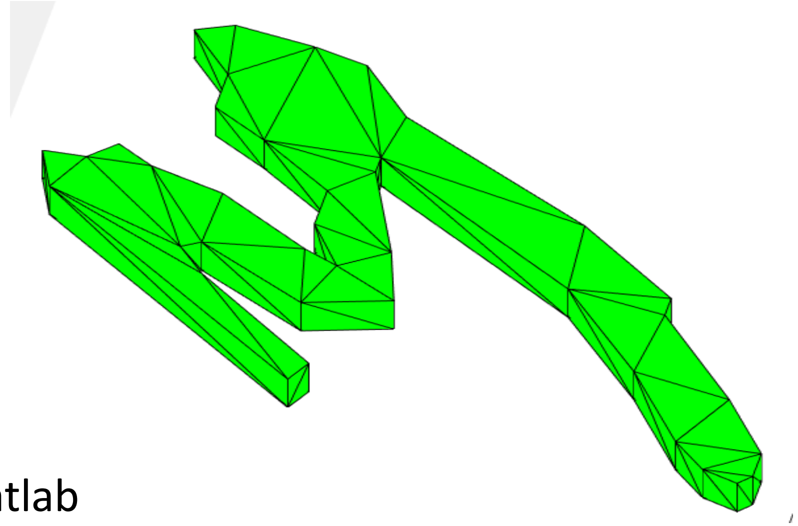
contour 2

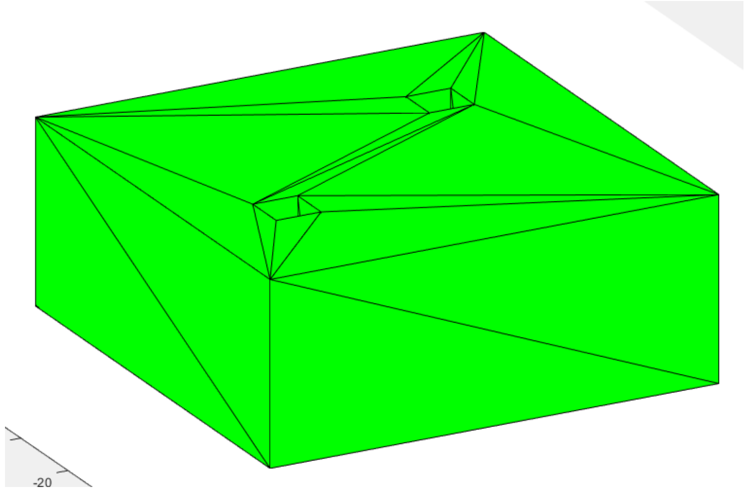
5.5,	23.34
5.5,	31.34
13.5,	31.34
15.5,	31.34
15.5,	29.34
15.5,	21.34
13.5,	21.34
5.5,	21.34
5.5,	23.34

New loop

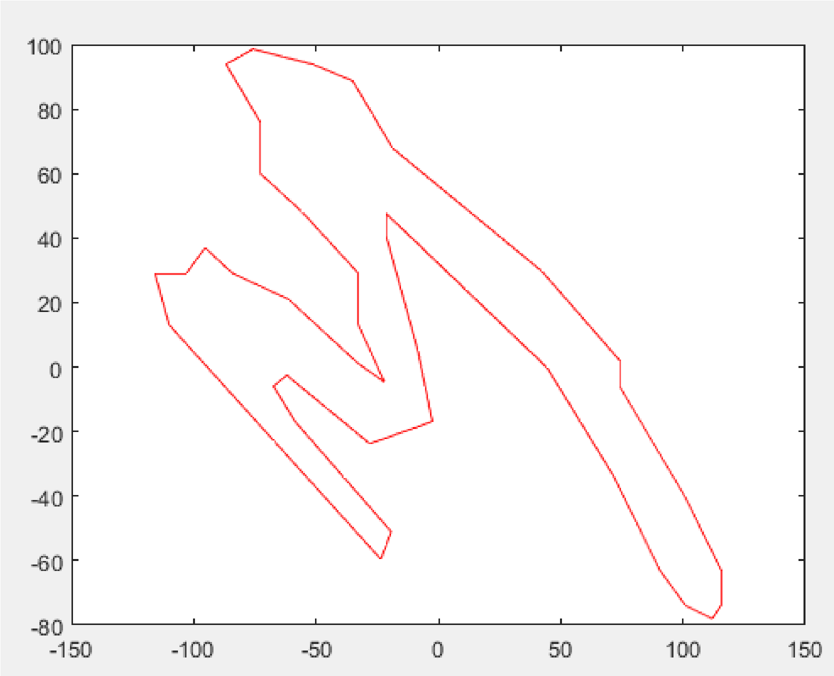
STL Slicing

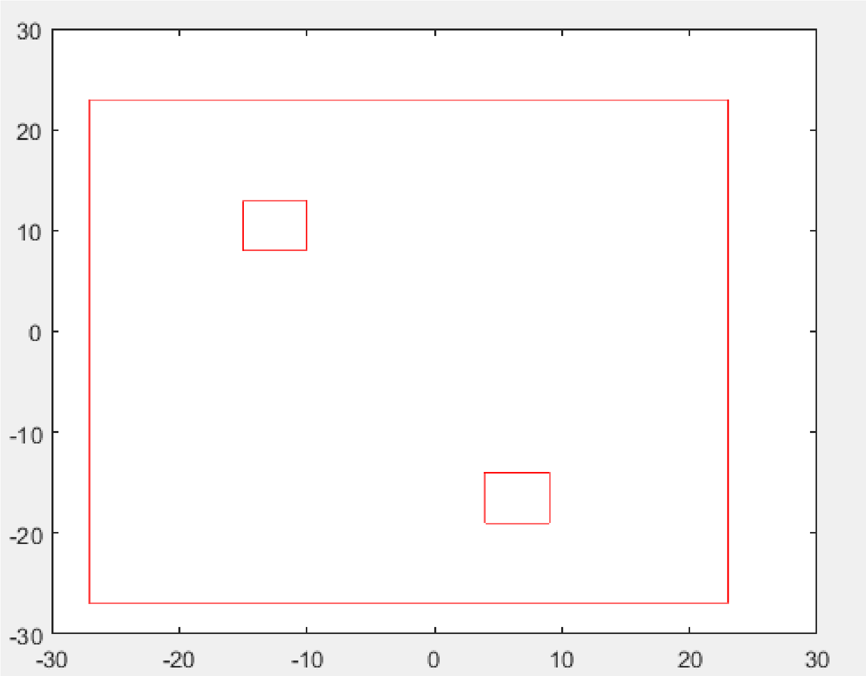
STL in Matlab





STL Slicing in Matlab

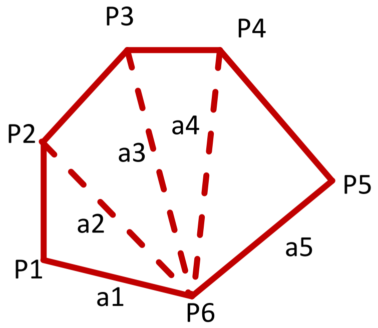




Calculation of Slice Area

For single contour :

Area of the slice can be calculated by using the area of polygon with all the points on the contour.



$SA_i = (1/2)\,\overline{n}\,(a_i \times a_{i+1})$ where $a_i = P_i - P_m$

(m=6 for this case)

Unit normal vector to the plane

Total Area = $\left| \sum_{i=1}^{m-2} SA_i \right|$

For multiple contours :

- Arrange all the points on the each of contours in anti clockwise direction when looking along the –Z axis (opposite to the build direction)
- Obtain the normal of one of the facets on the contour.
- Find the direction of the contour points (with any 2 consecutive points on contour)
- Perform the cross product of both (normal, direction)
- Calculate the unit vector of the cross product
- Product of unit vector with the respective contour areas will give the actual areas (solid +ve, hollow -ve).
- Sum of all the areas will result in final area of the layer