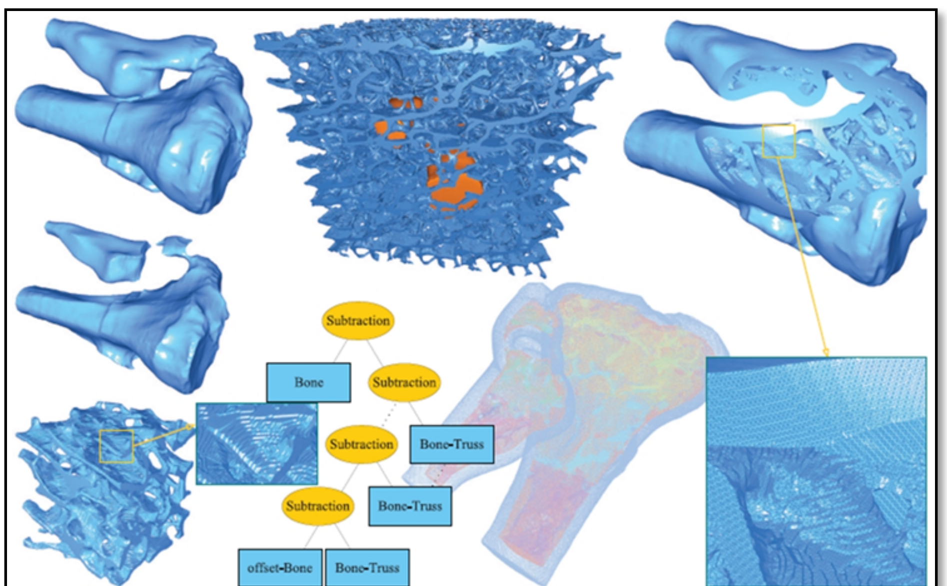
L11 – GPU-based Solid Modeling for Manufacturing

- Introduction
- Related Work
- Structured Point Representation
- Boolean Operations
- Offsetting by Super-Union of Balls
- Solid Modeling for Rapid Fabrication
- Fast Solid Modeling for Feed-rate Optimization

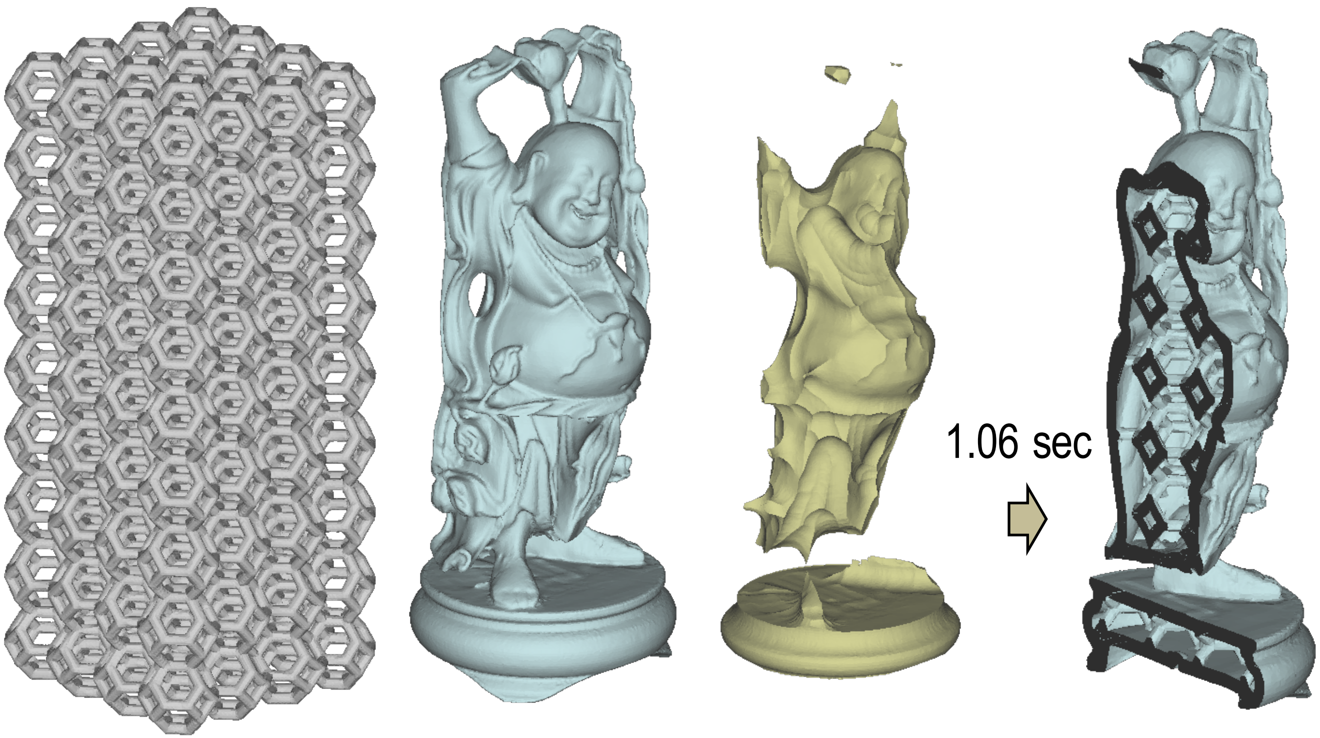
Solid Modeling for Fabrication

- **Framework:** *GPU-based Solid Modeler for Complex Objects*
- **Purpose:** using the computational power on GPU to speed up solid modeling operations in Layered Depth-Normal Images (LDNI) rep.
- Models in many applications have very complex shape and topology (e.g., microstructure design, rapid prototyping, etc.)

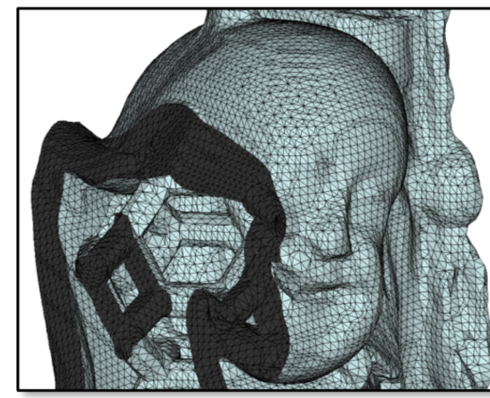


Boolean Operations

- Boolean operations on models with massive number of triangles (Wang et al., 2010)



941.9k Faces 497.7k Faces 213.3k Faces 780.4k Faces



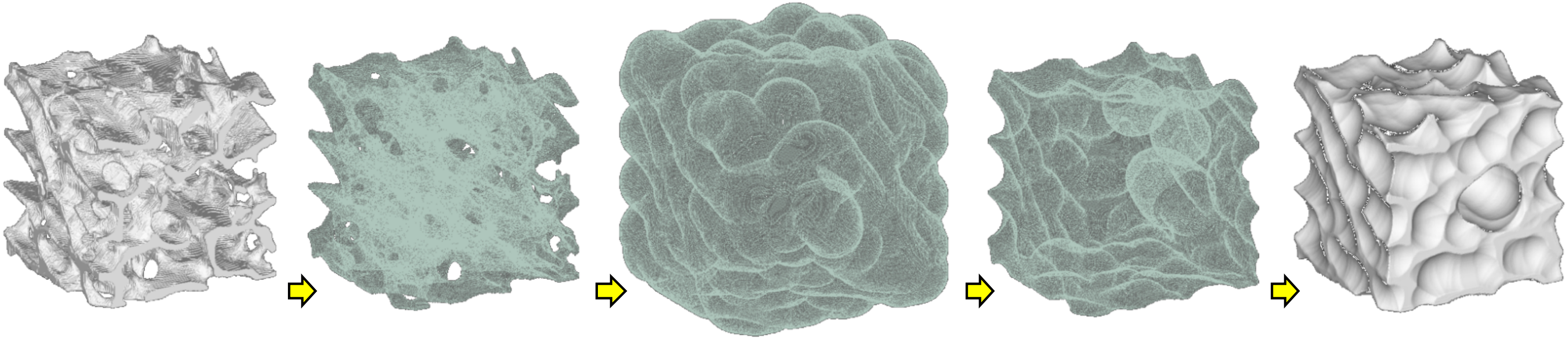
On GeForce GTX 580

Offsetting

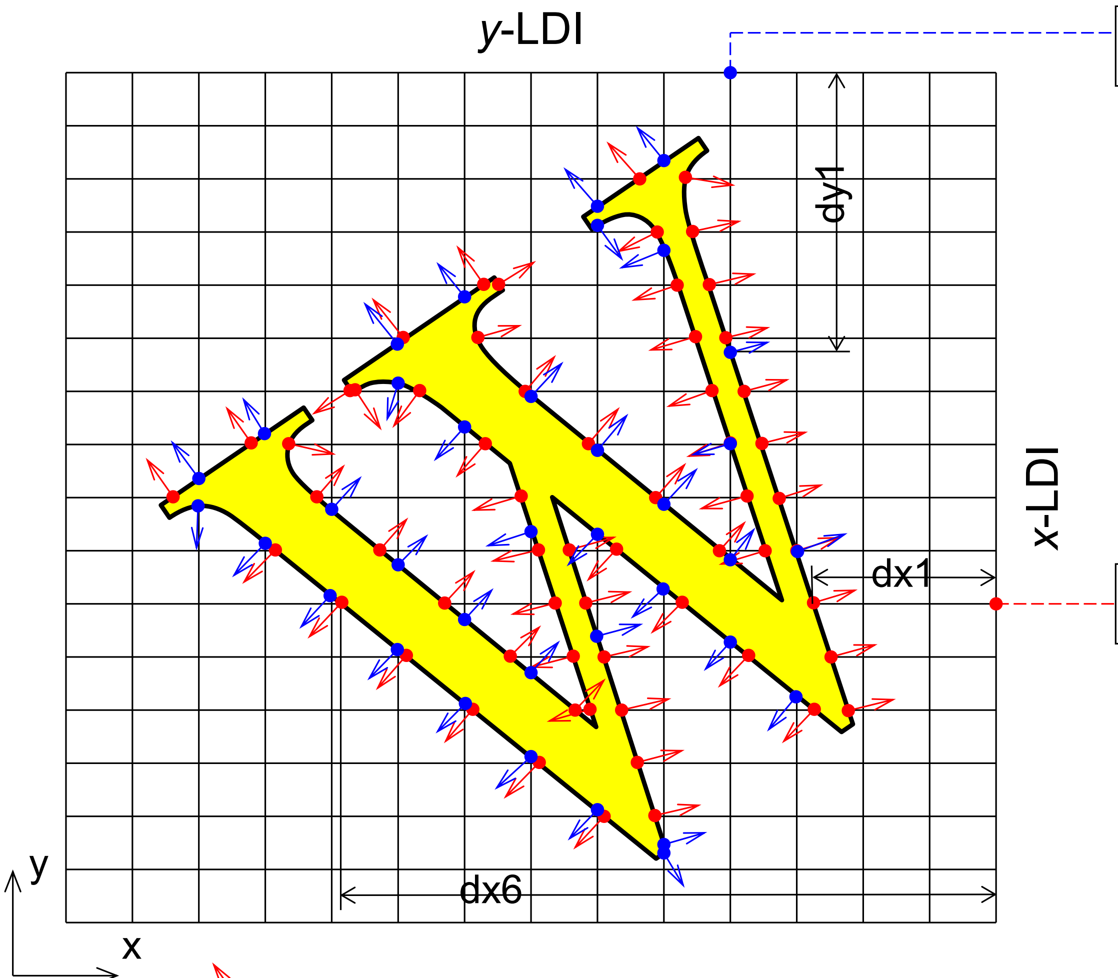
- **Problem Definition:**

  Given a solid model H with its boundary surface approximated by the set P_H of sample points in LDNI-representation, we compute the boundary surface of exterior offset (or interior offset) and represent it by a point set in LDNI-representation.

- **Our Approach:**
  - Fast Approximate Offsetting
  - Directly offsetting solid models in LDNI representation
  - Highly parallel algorithm



Structured Point Representation - LDI



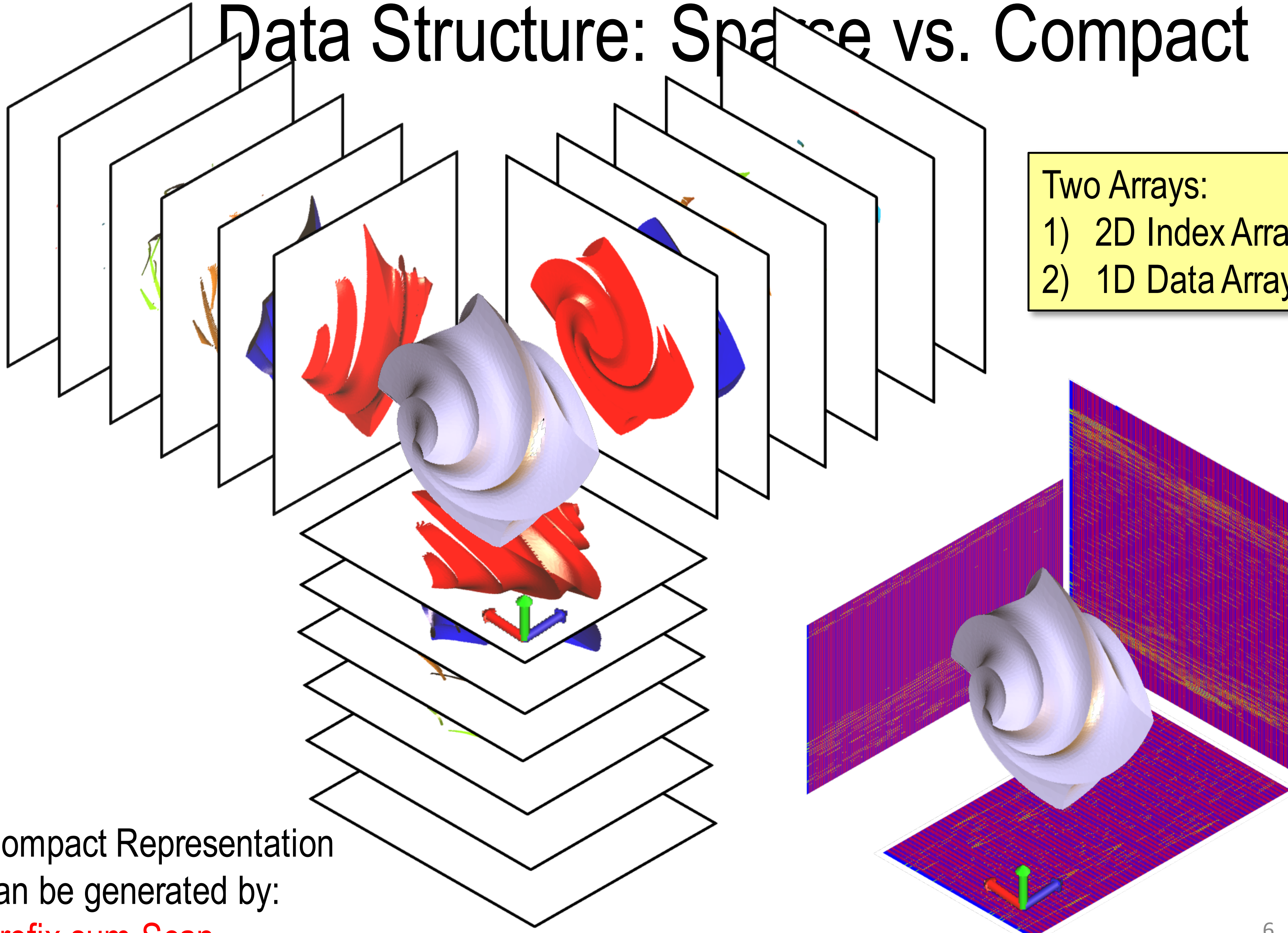
4				
	dy1	$n1_x$	$n1_y$	$n1_z$
	dy2	$n2_x$	$n2_y$	$n2_z$
	dy3	$n3_x$	$n3_y$	$n3_z$
	dy4	$n4_x$	$n4_y$	$n4_z$

6				
	dx1	$n1_x$	$n1_y$	$n1_z$
	dx2	$n2_x$	$n2_y$	$n2_z$
	dx3	$n3_x$	$n3_y$	$n3_z$
	dx4	$n4_x$	$n4_y$	$n4_z$
	dx5	$n5_x$	$n5_y$	$n5_z$
	dx6	$n6_x$	$n6_y$	$n6_z$

Sampled depth on *x*-LDNI coupled with surface normal

Sampled depth on *y*-LDNI coupled with surface normal

Data Structure: Sparse vs. Compact



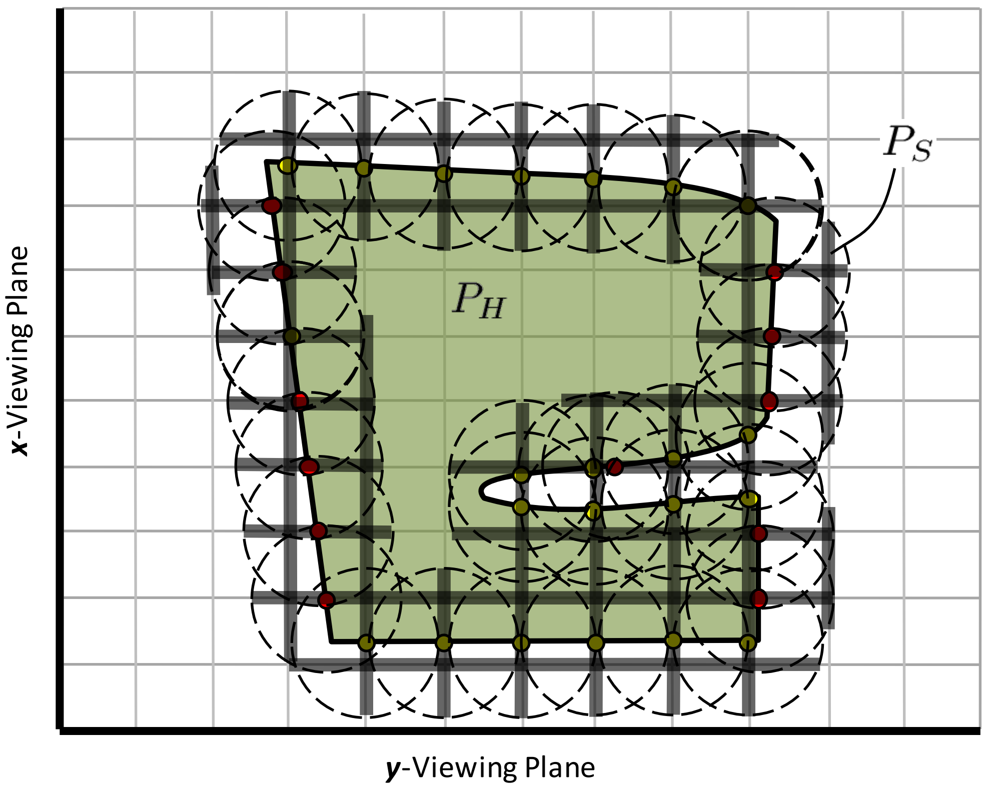
Two Arrays:
1) 2D Index Array
2) 1D Data Array

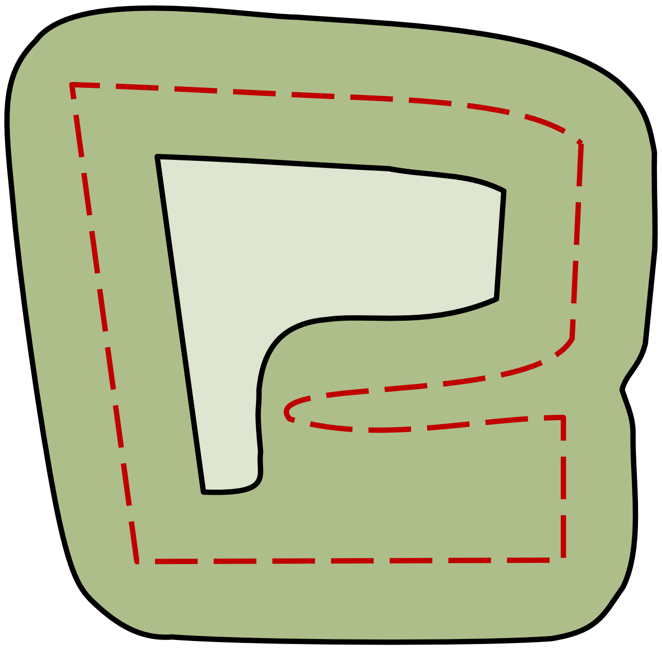
Compact Representation
can be generated by:
Prefix-sum Scan

Offsetting by Super-Union of Balls

- Boolean operations on LDNI solids – Boolean on 1D rays
  - Highly parallel and robust
- Offsetting shell P_S – by **union** many balls

Offsetting Result:
Exterior Offset = $P_H \cup P_S$
Interior Offset = $P_H - P_S$





Offsetting by Super-Union (Cont.)

- Ray-based Computation
- Two Groups

I) By rays in the same view

II) By rays in different views

- Super-union

Status update by

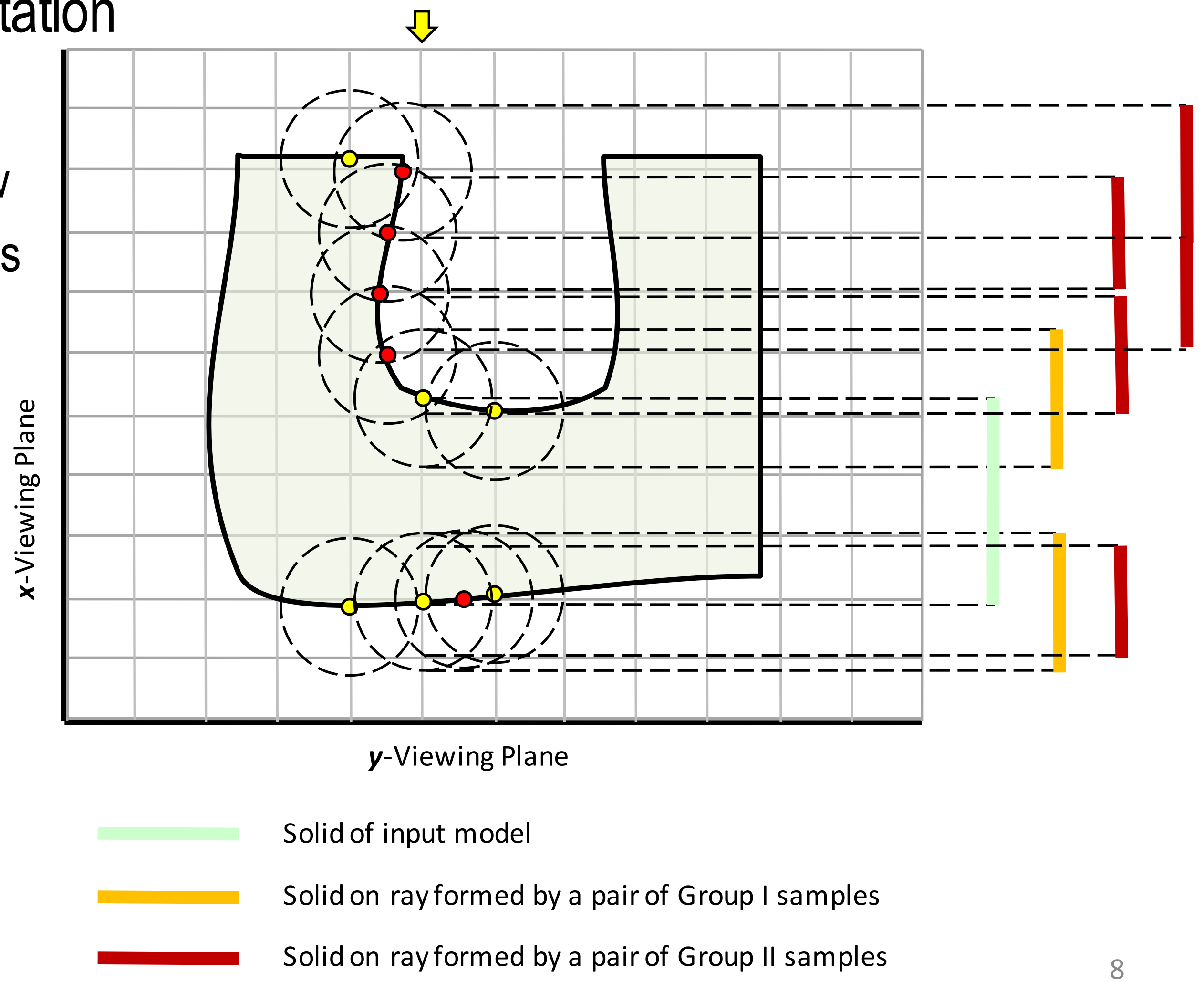
entering / **leaving**

samples

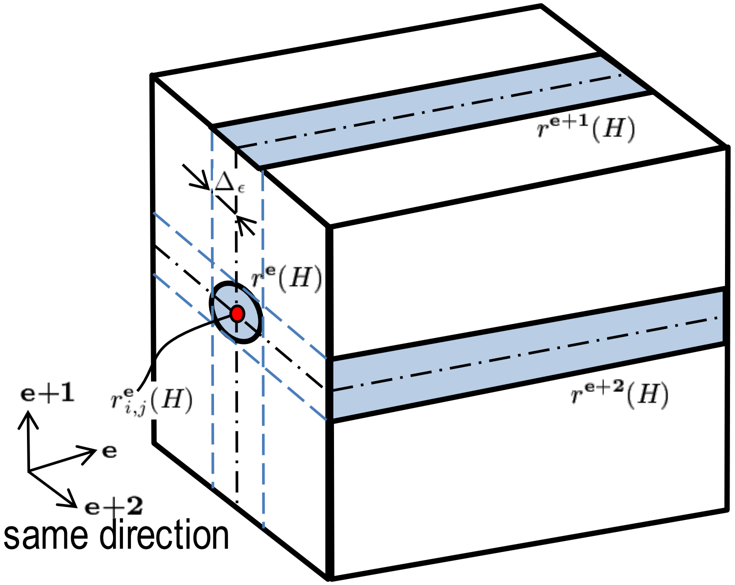
- Problem:

Efficient ray-sphere

intersection detection



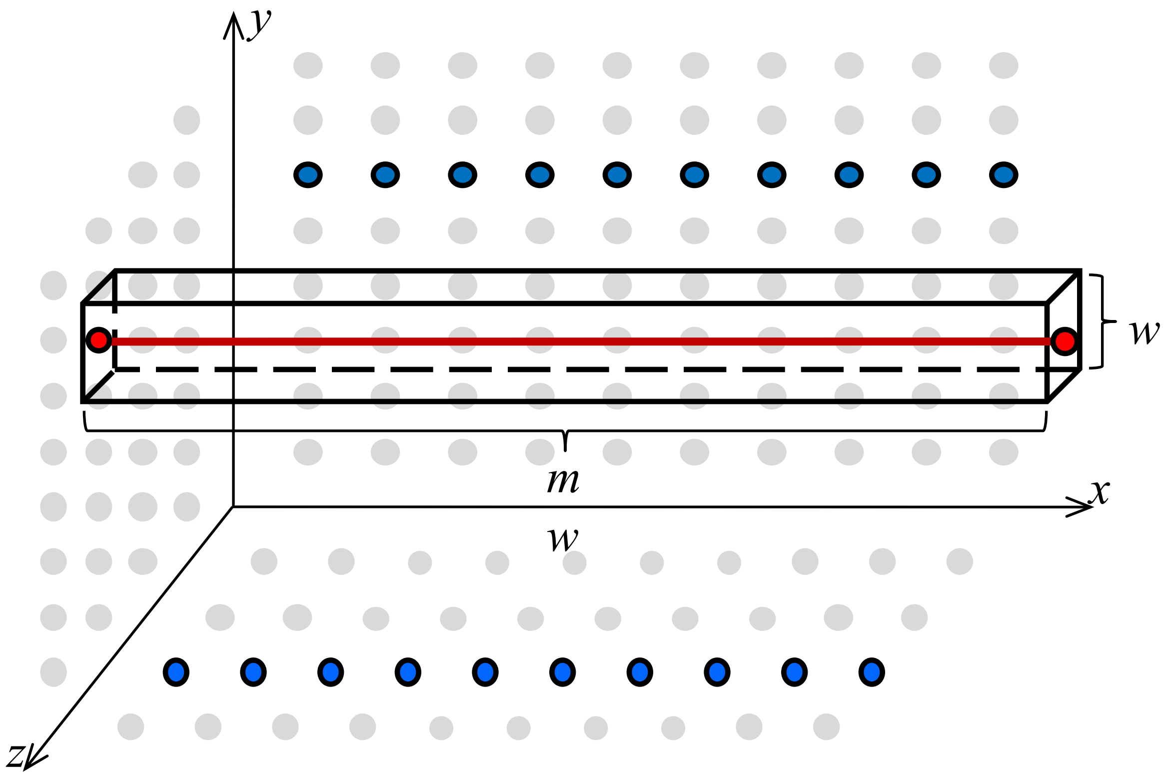
Primary GPU Scheme



- Three steps algorithm:
  1) For each ray in one direction in parallel
     - Search the intersections between this rays and spheres generated by samples on the rays in the same direction
     - Merging intersected 1D solids
     - Storing the result in a global data buffer array
  2) For each ray in one direction in parallel
     - Search the intersections between this ray and spheres centered at the rays in other directions
     - Merging intersected 1D solids into the existing 1D solid on this ray
  3) Rebuild the index array and the resultant data array (by Prefix-sum Scan)
- Reconstruct normal vectors on the resultant samples
  - Orientation-aware *Principal Component Analysis* (PCA)
  - Carry on the neighborhoods of a sample

GPU-based Algorithm: Spatial Hashing

- Bottleneck of primary GPU algorithm – Step 2) taking 80%-85% time
- Searching too many rays in other directions: $(2m) \times (2r / w)$
- Redundancy: not every ray has sample fall in the range



Solution:
- Sorting samples from other rays by their coordinate in the *yoz* plane
- Building spatial hashing bins around ray in *x* direction
- Step 2) can be conducted by only searching samples in these bins

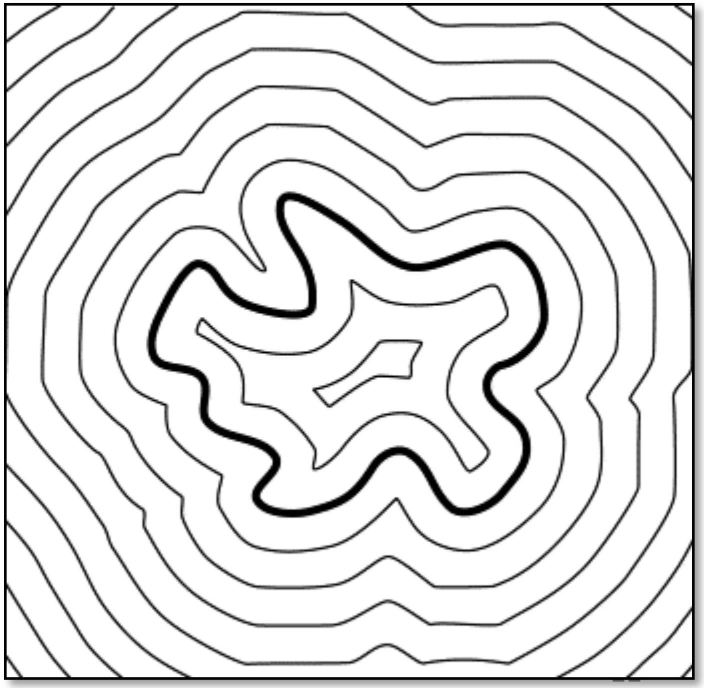
Result: search only (2r / w) x (2r / w) bins

Successive Offsetting for Large Offset

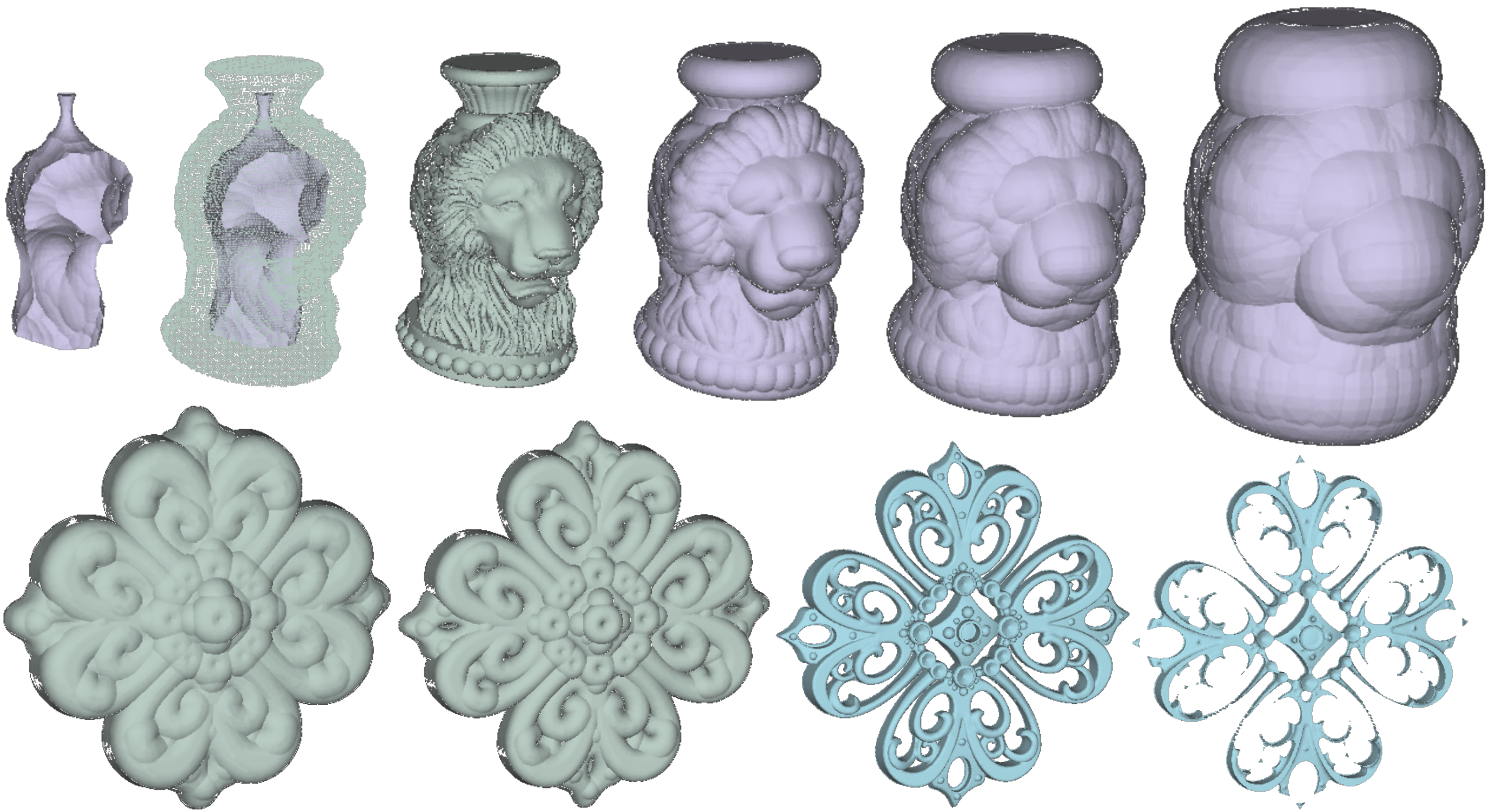
- Computation cost on each ray: O($(2r/w)^2$) – the search range
- Slow, when r is very large
- Offsetting with large distance r can be decomposed into n successive offsetting with smaller distance r_i where $r = nr_i$ (Rossignac & Requicha, 1986)
- Computational cost is reduced to $1/n$

  O($n(2r_i/w)^2$) = O($(2r/w)^2$)/n
- At the downside, performing offsets too many times in succession

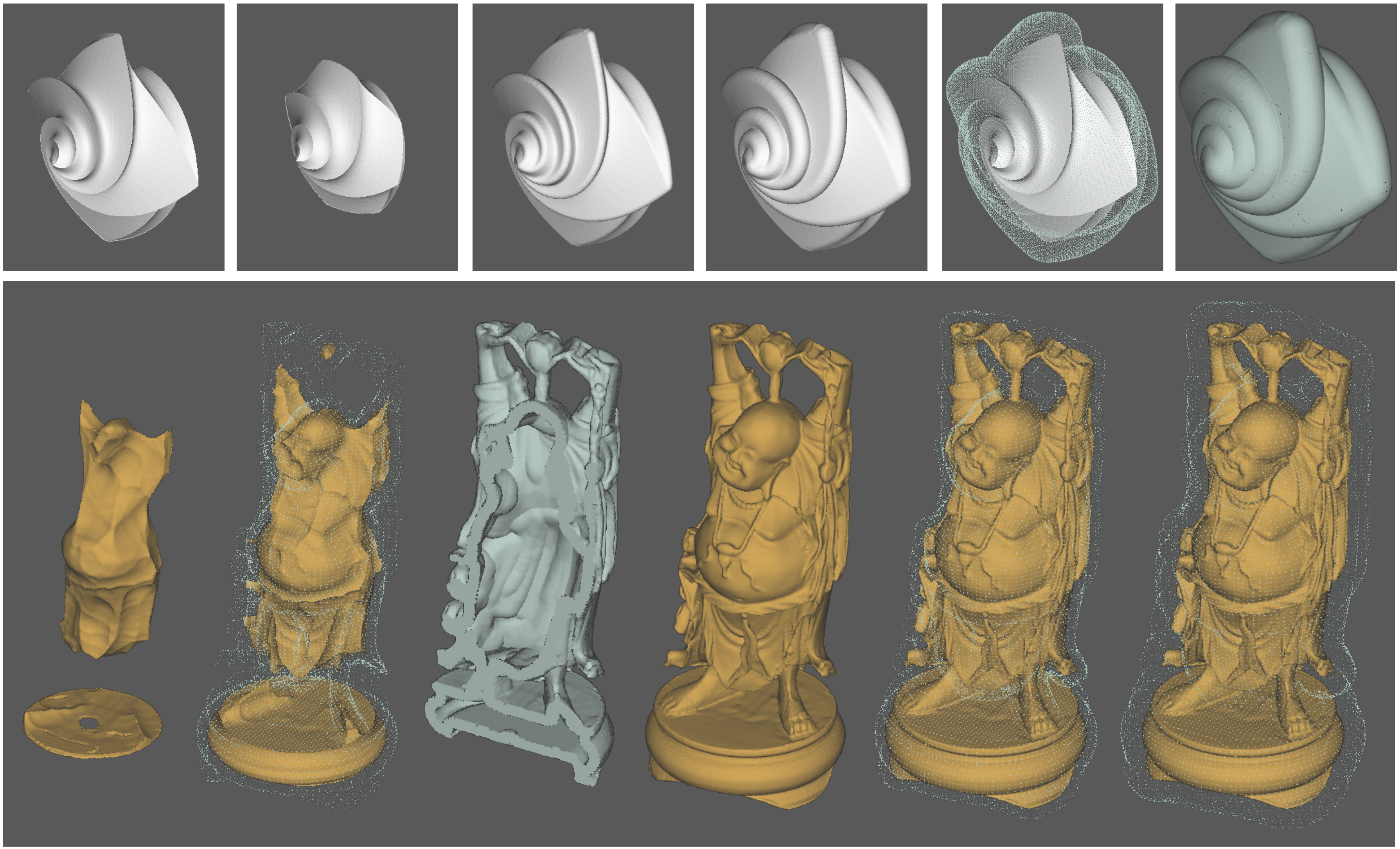
  => Large approximation error



Offsetting Results

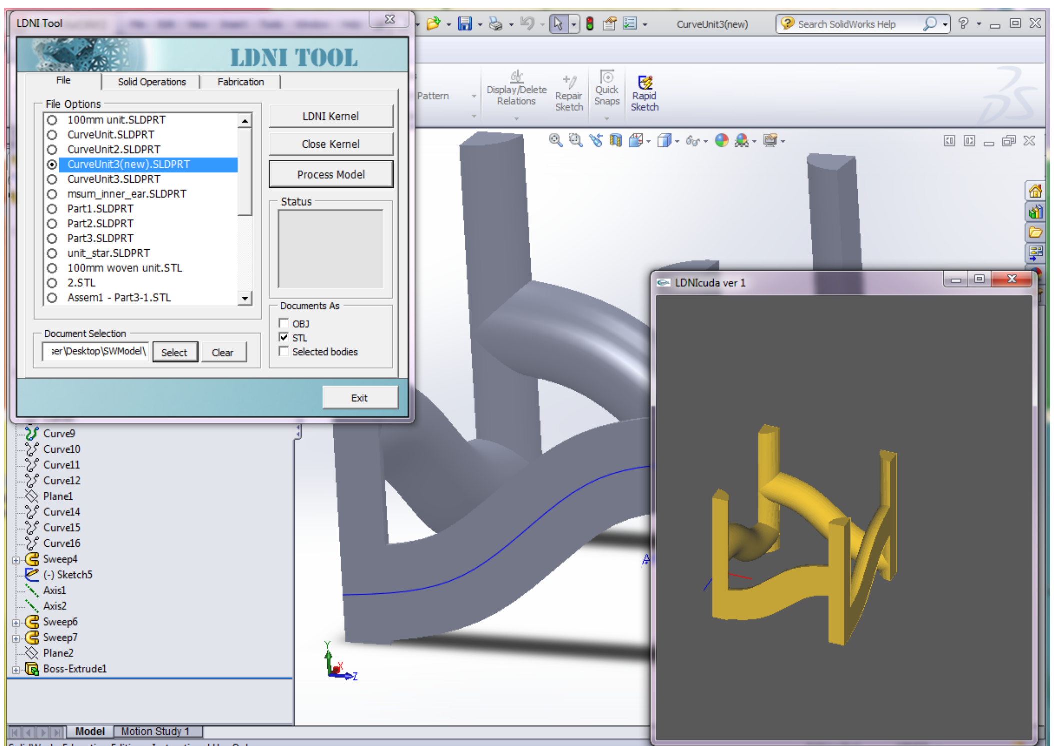


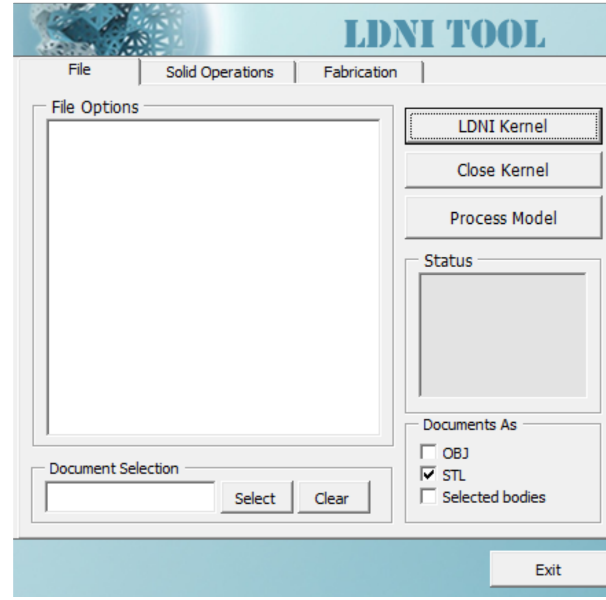
Offsetting on Different Models

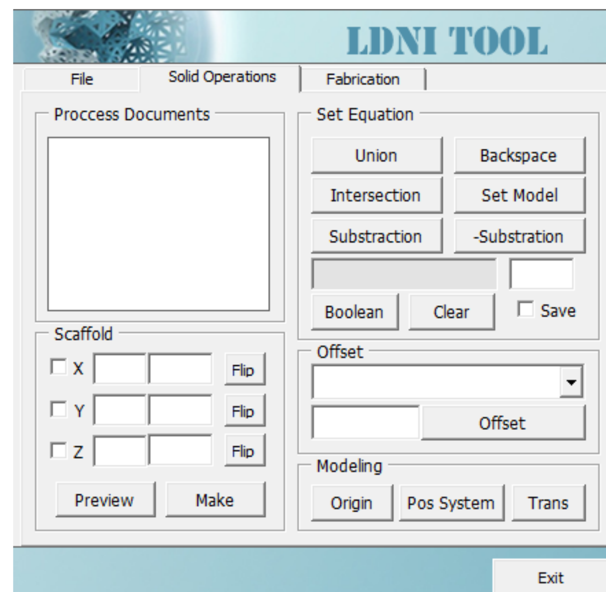


Current Development

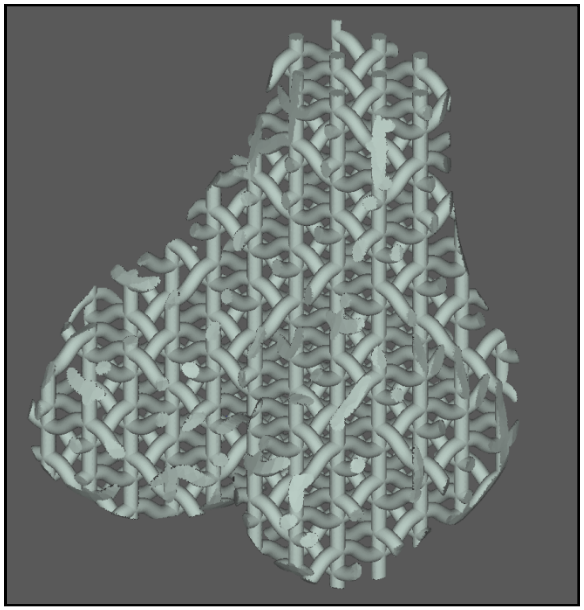
- Not only the framework of our kernel, we also develop an interface for users to interact between the SolidWorks (a commercial CAD tool) and our framework
- Increase the utility of our work

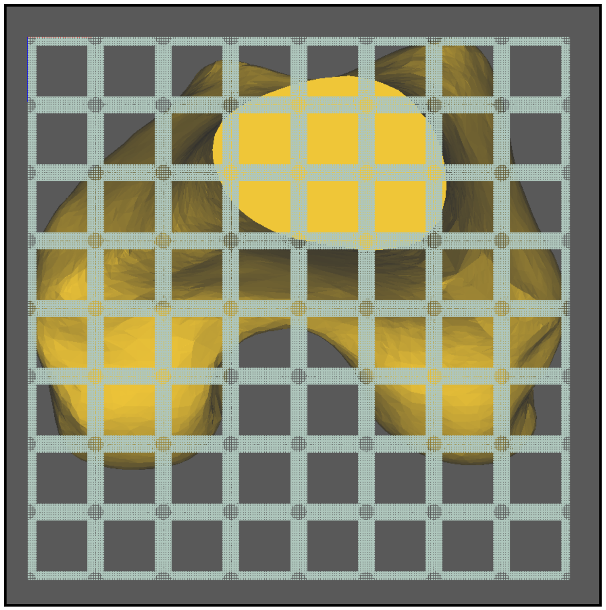


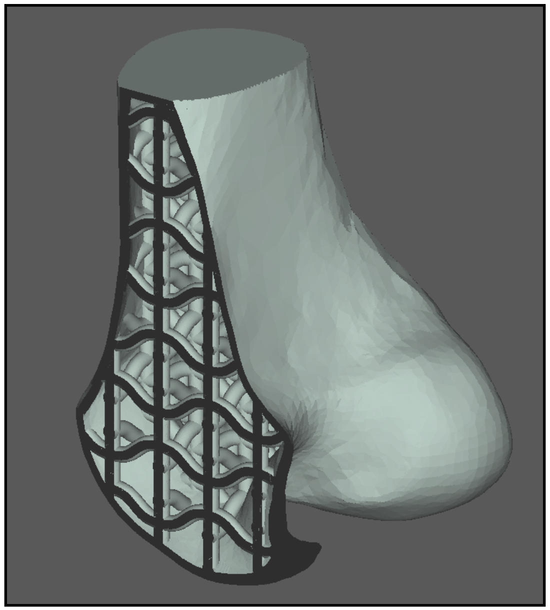




Boolean Operations







LDNI Resolution	1024 × 1024
Models	Femur ∩ Scaffold (832 cells)
GPU Memory Usage	27.3MB
*Operation Time (sec)	3.71s

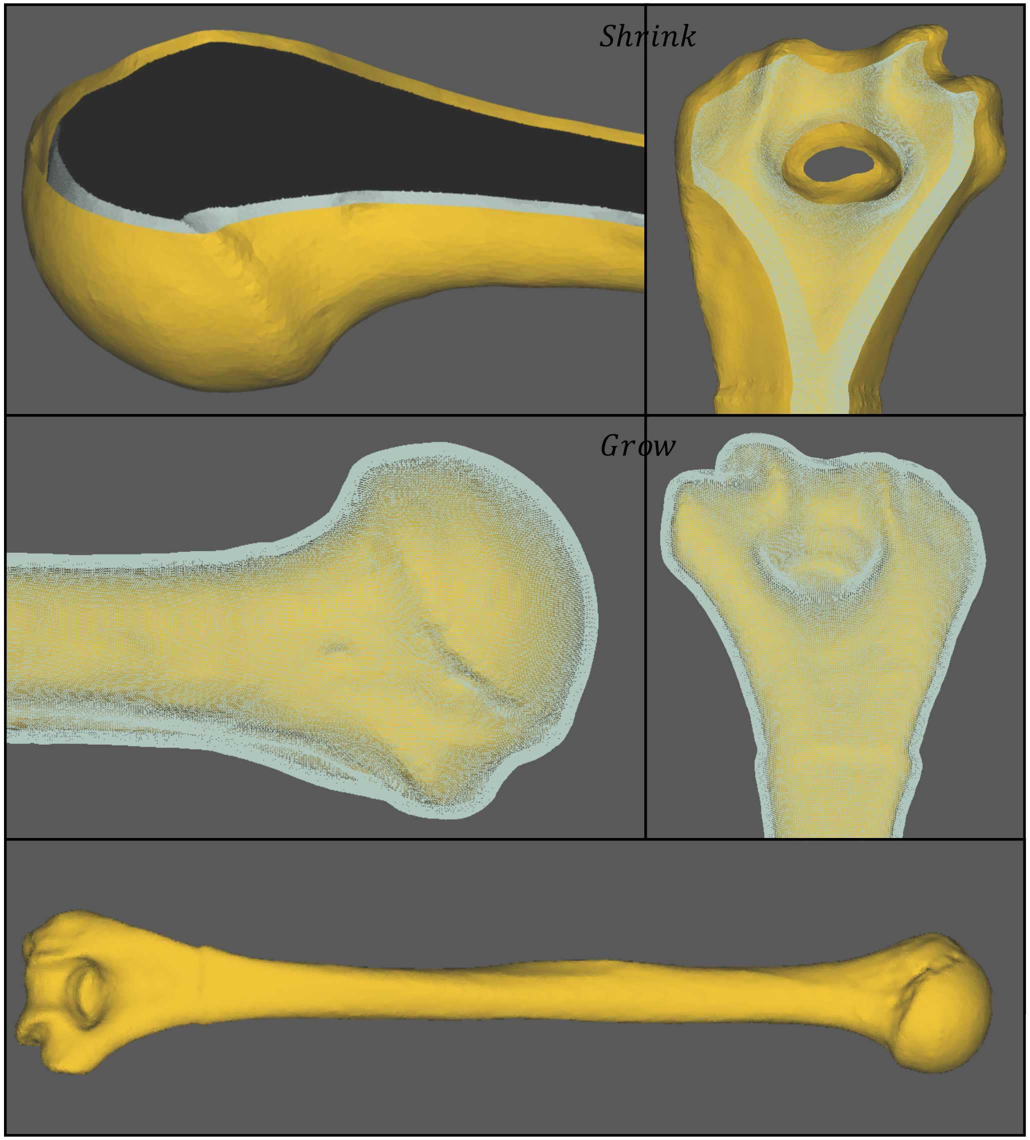
LDNI Resolution	1024 × 1024
Models	Femur / (FemurOff / Scaffold)
GPU Memory Usage	49.6MB
*Operations Time (sec)	4.07s

*Included scaffolding and sampling time

Offsetting

Shrinking Offset	
LDNI Resolution	2048×2048
Offset value	$-15 \times \varepsilon$
Face Num.	70K
GPU Memory Usage	84.1 MB
Operation Time (sec)	8.14s

Growing Offset	
LDNI Res	2048×2048
Offset value	$10 \times \varepsilon$
Face Num.	70K
GPU Memory Usage	32.8 MB
Operation Time (sec)	4.602s

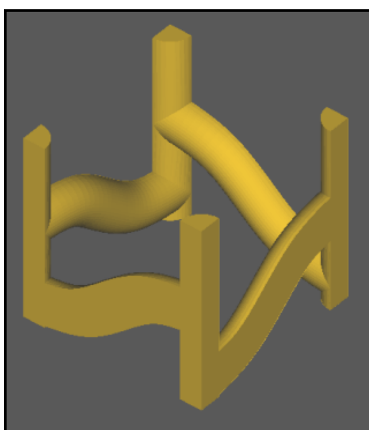


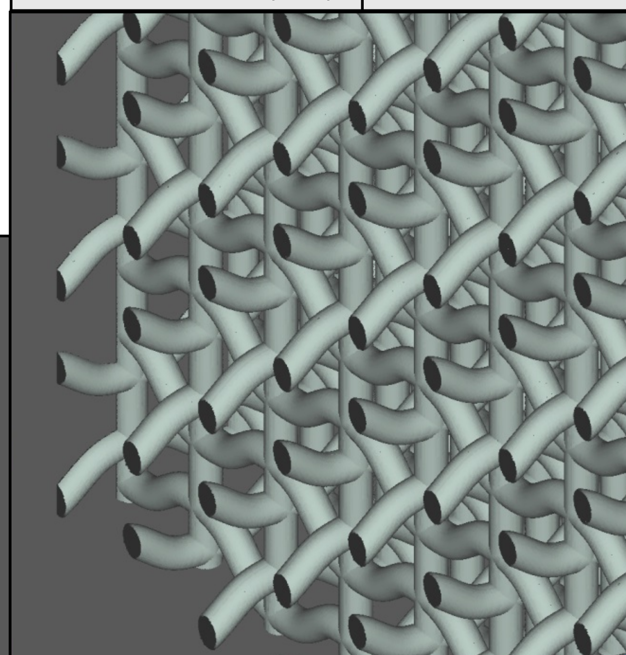
$^{*}\varepsilon = sampling\ distance\ width$

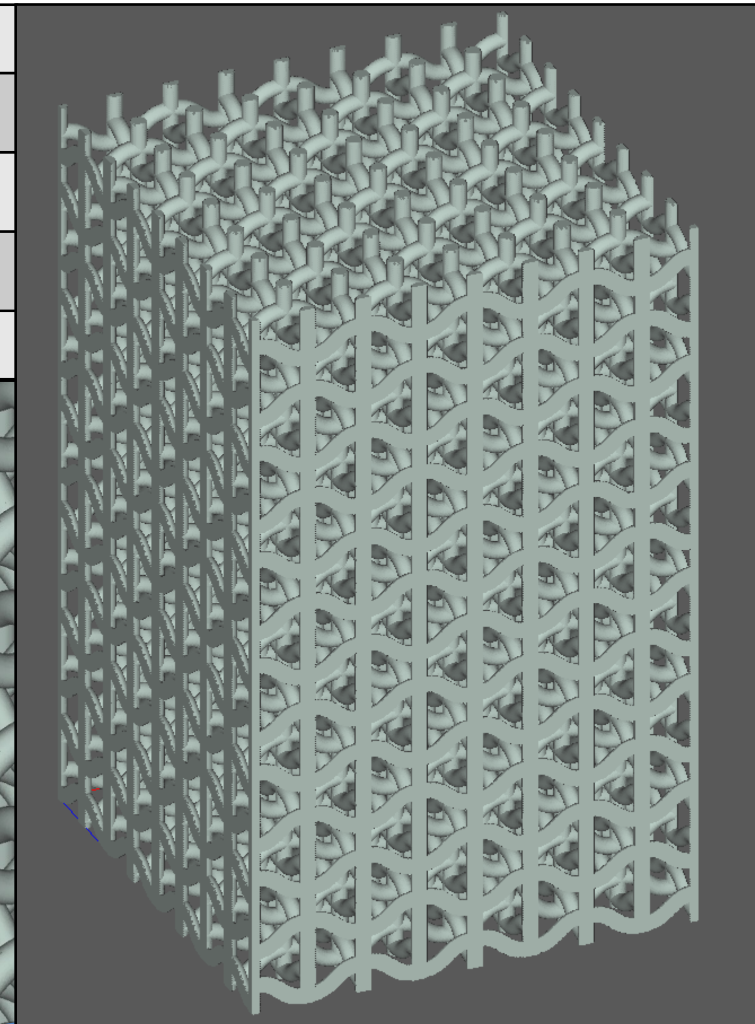
Scaffolding

- Union operations applied on instances of a model at the same time

LDNI Res	1024 × 1024
Cell Num.	8 x 13 x 8 (832)
Face Num.	7.6K/per cell
GPU Memory Usage	151 MB
Operation Time (sec)	3.46s



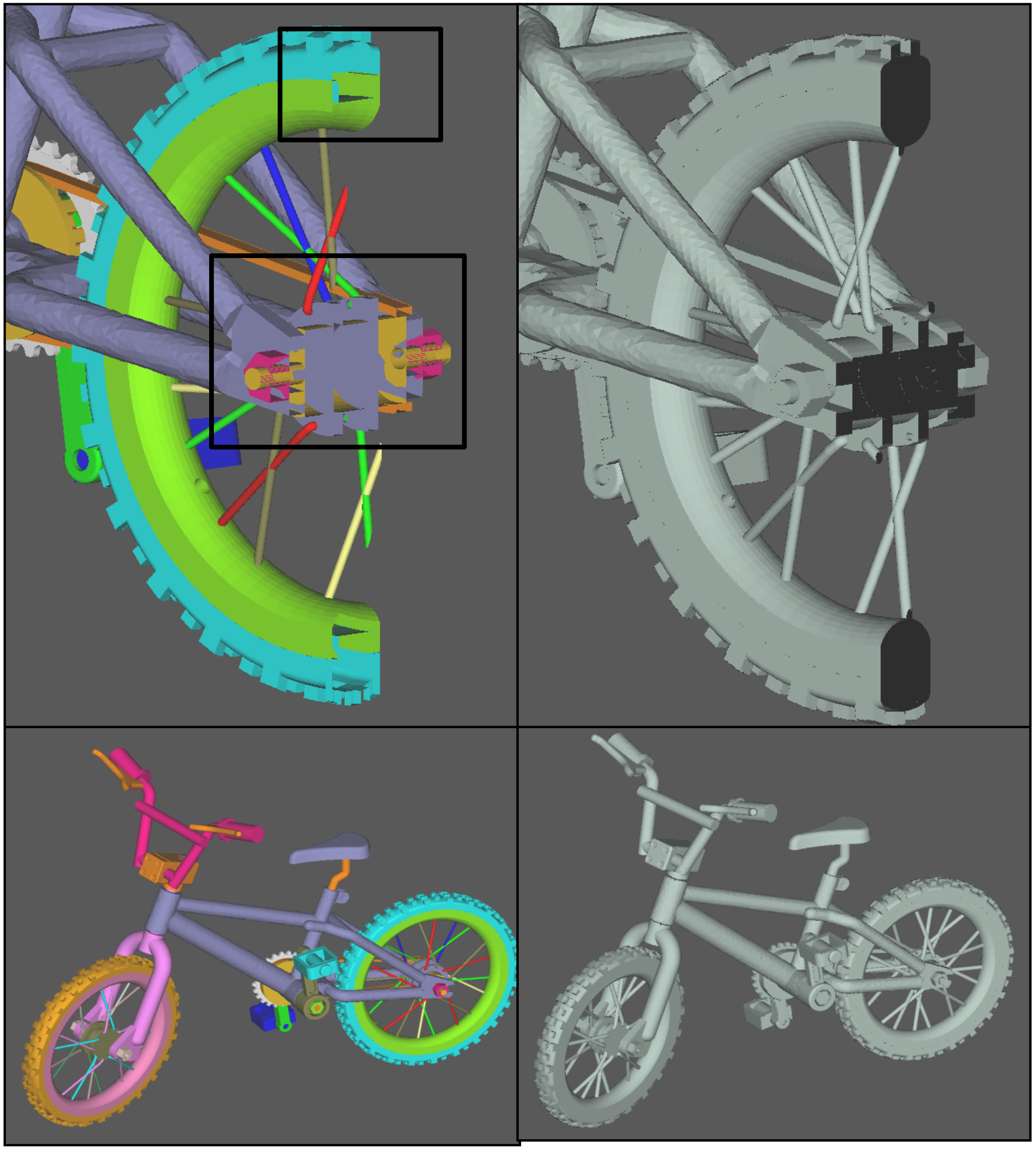




Super-Union

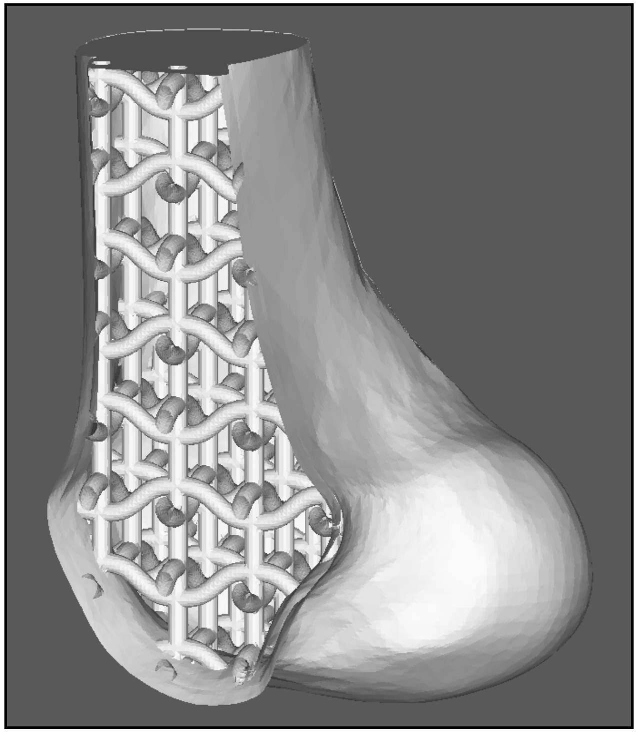
- Union operations applied on multiple different models at the same time
- Overlapped or intersected objects can be converted into one solid

Components	32
Total Face Num.	161K
Resolution	4096 × 4096
GPU Memory Usage	232.6MB
Operation Time (sec)	1.88s

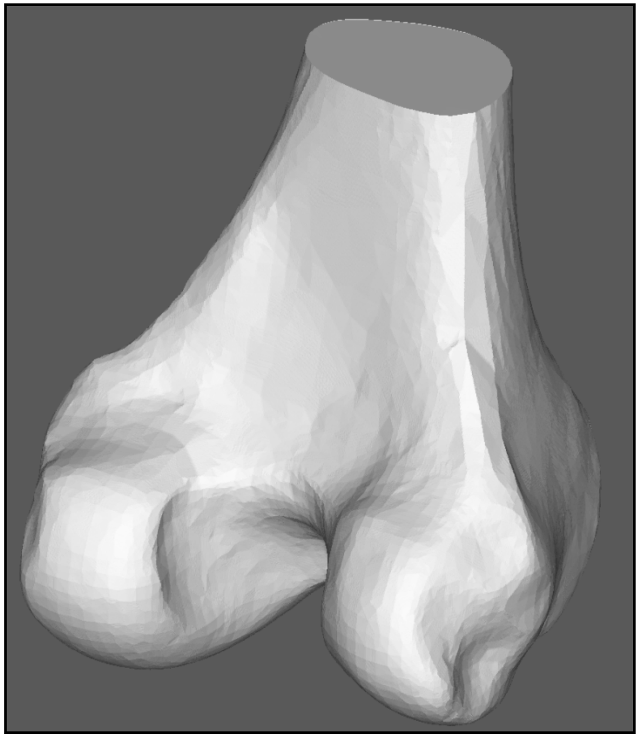


Contouring

- Convert LDNI back to B-rep representation
- For further operations that require boundary information



Boolean of LDNI → Mesh	
LDNI Res	1024 × 1024
Face Num.	513K
Time (sec)	0.23s

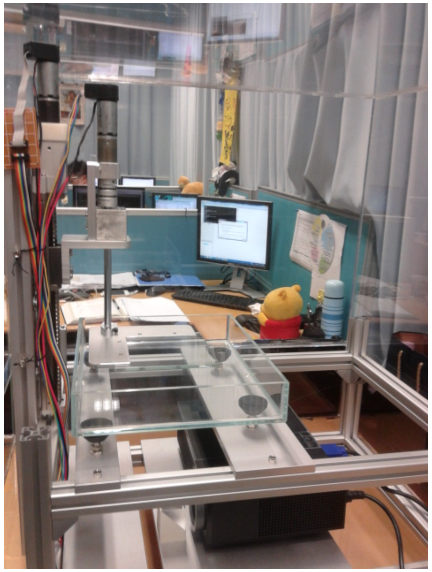


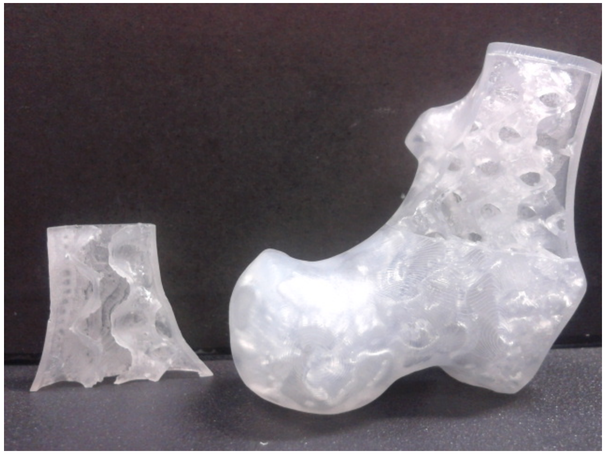
Offset in LDNI → Mesh	
LDNI Res	1024 × 1024
Face Num.	338K
Time (sec)	0.11s

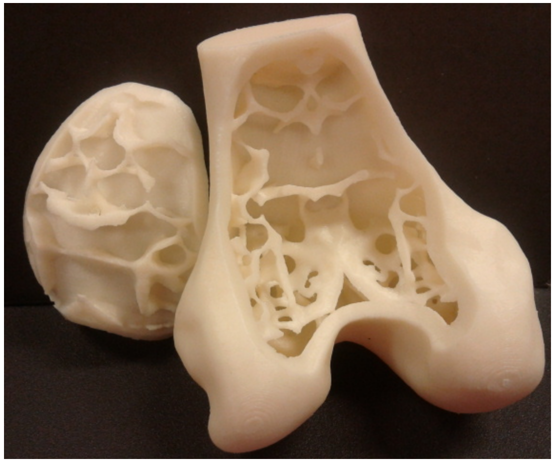
[Link of source code]

Downstream Apps

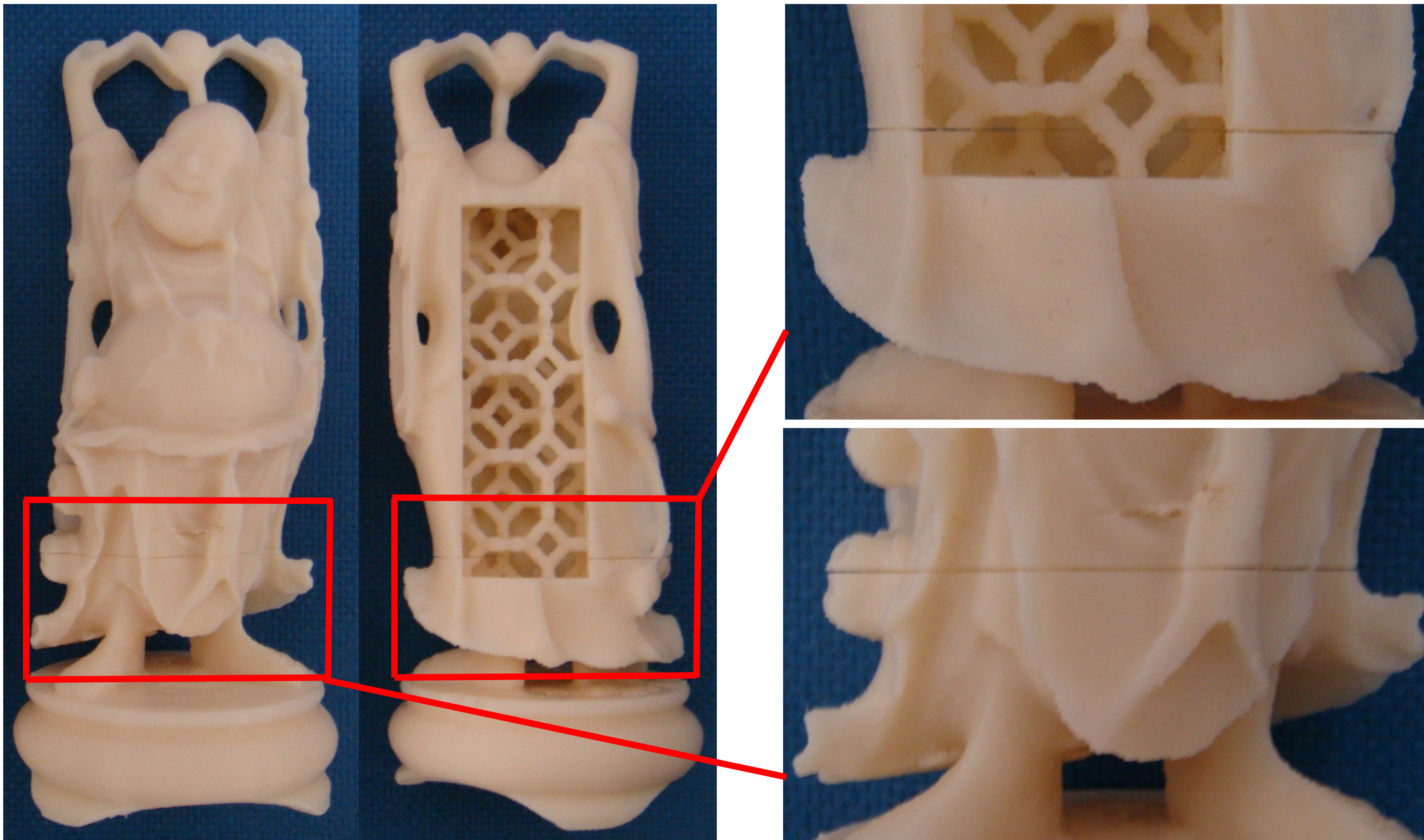
- Fused Deposition Modeling (FDM)
- StereoLithography Apparatus (SLA)
  - Contours are needed
- Mask-projection SLA
  - Direct binary image projection

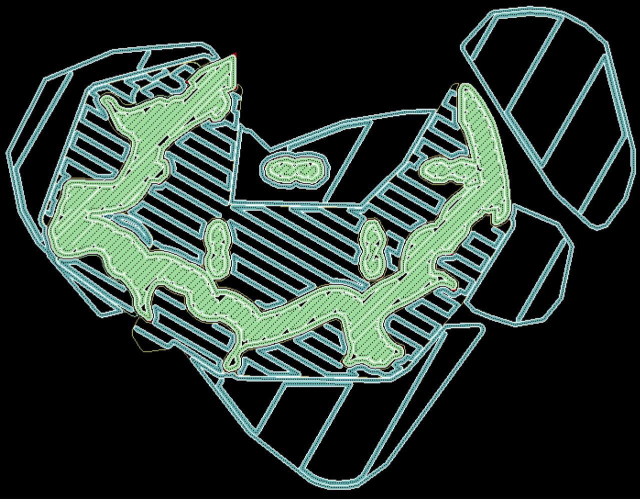




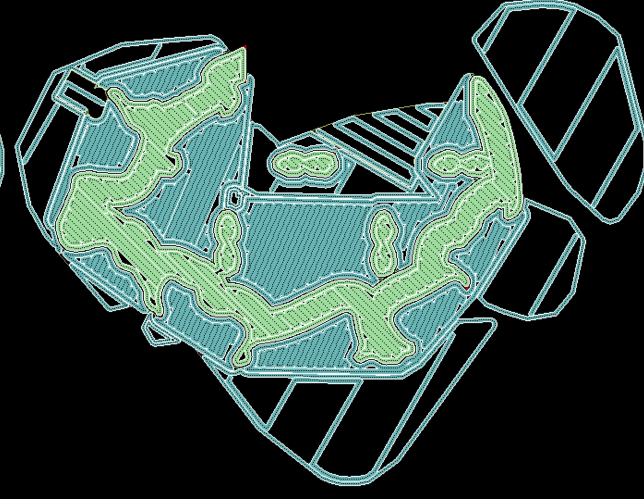


Problem with Existing Approaches (by B-rep)

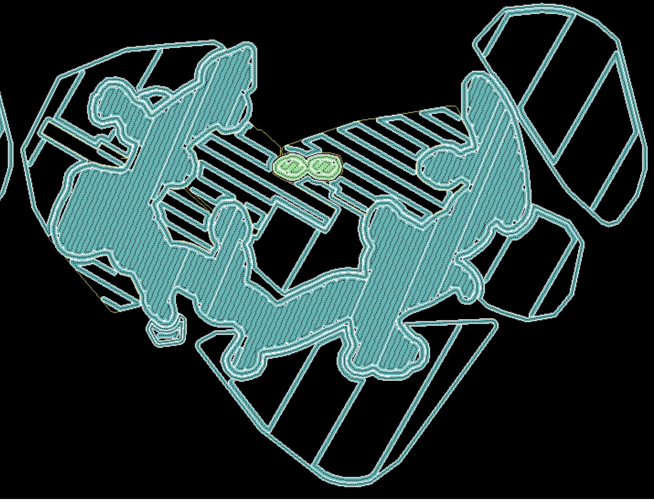




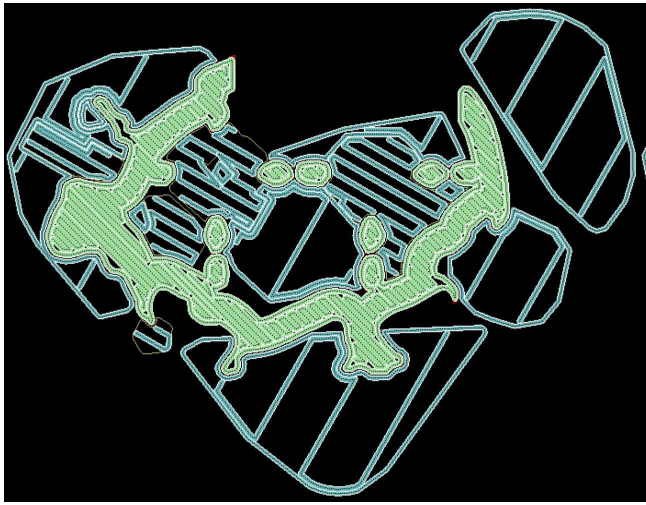
Height = 1.77 inch



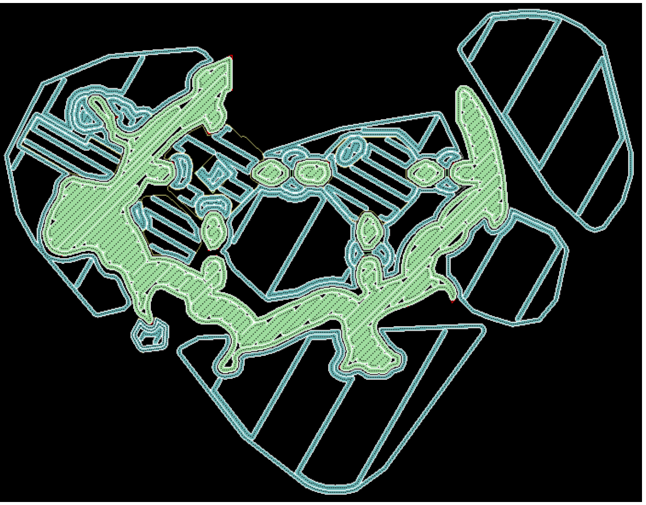
Height = 1.78 inch



Height = 1.79 inch



Height = 1.80 inch

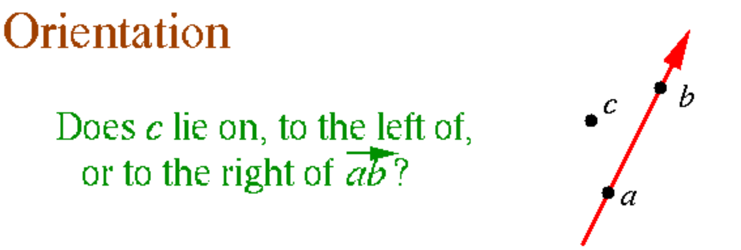


Height = 1.81 inch

Generated by Commercial Software for FDM

Problem of Conventional B-rep Modeler

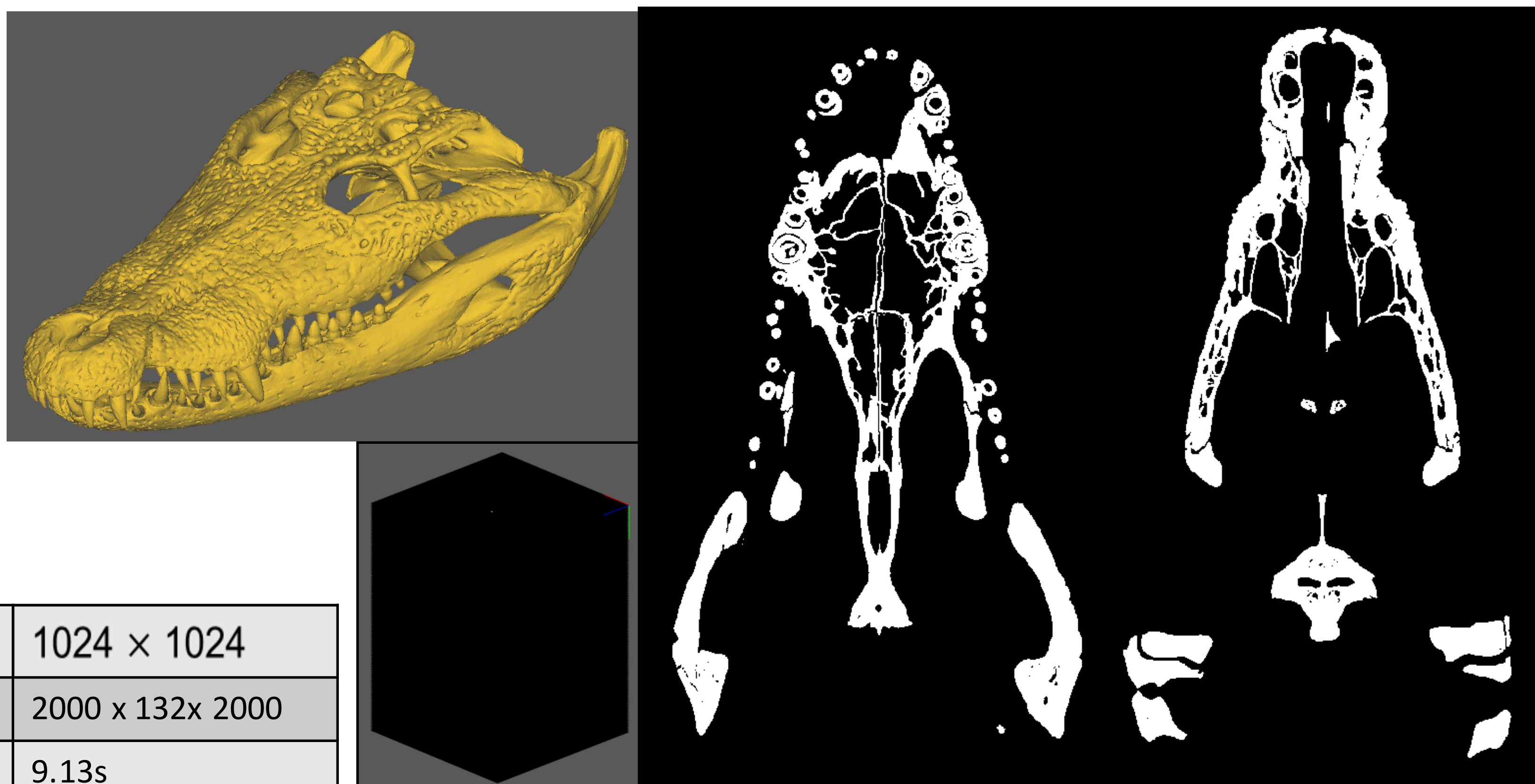
- Numerical robustness
- Computation in IEEE arithmetic
  - Limited precision of floating-point arithmetic
- Geometry becomes inexact after intersection
- Geometric predicates
  - Correct?
  - Intersected models?
  - Membership classification?
- Exact representation?
  - Multiple precision arithmetic library
  - Plane-based representation



$$\begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = \begin{vmatrix} a_x - c_x & a_y - c_y \\ b_x - c_x & b_y - c_y \end{vmatrix}$$

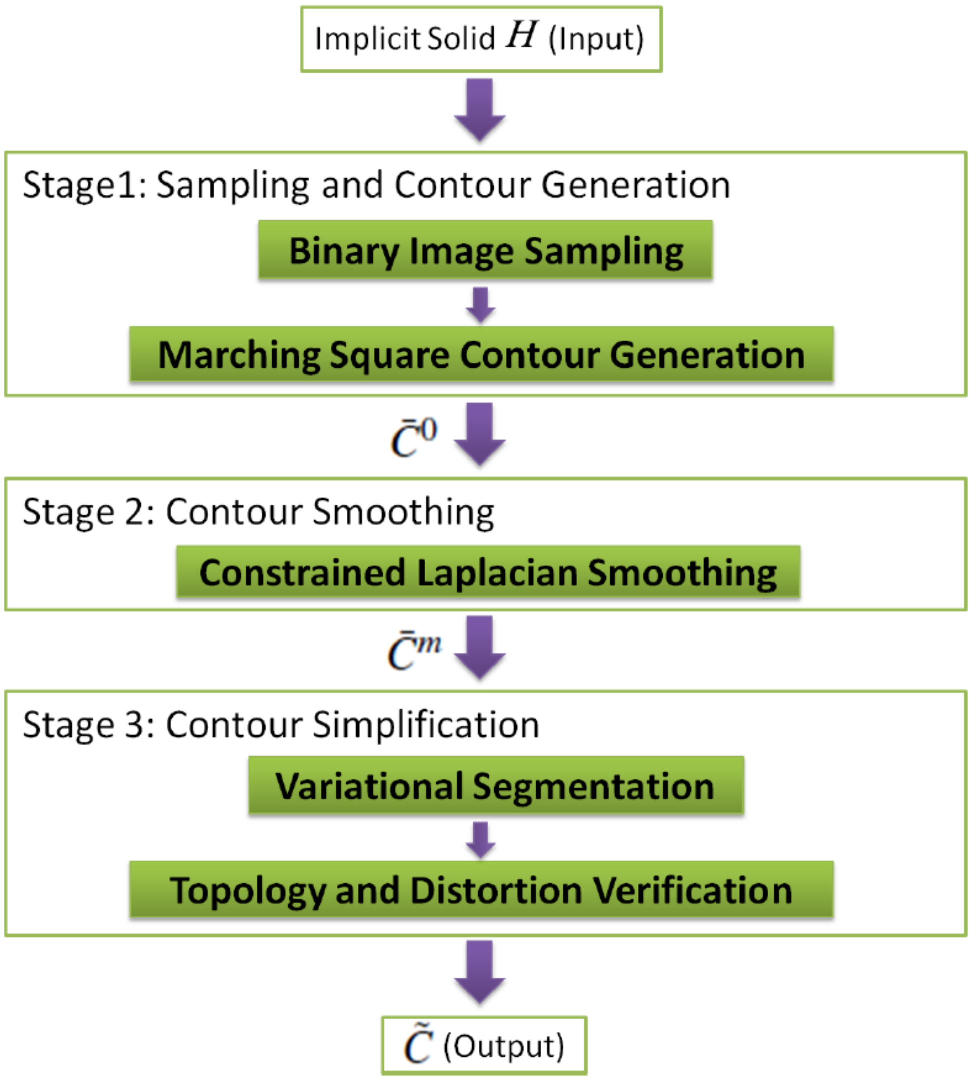
How to provide reliable information for fast fabrication?

- Slicing or Modeling (by LDNI-rep) in image space
- Fabrication in image space – Mask-Projection based SLA



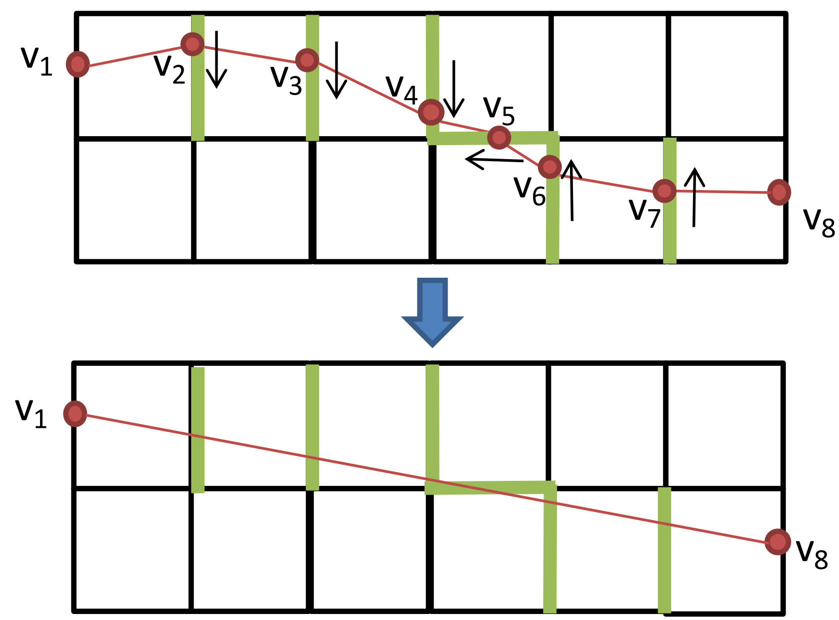
LDNI Res	1024 × 1024
Image Size	2000 x 132x 2000
Time (sec)	9.13s

Reliable Slicing in Image Space



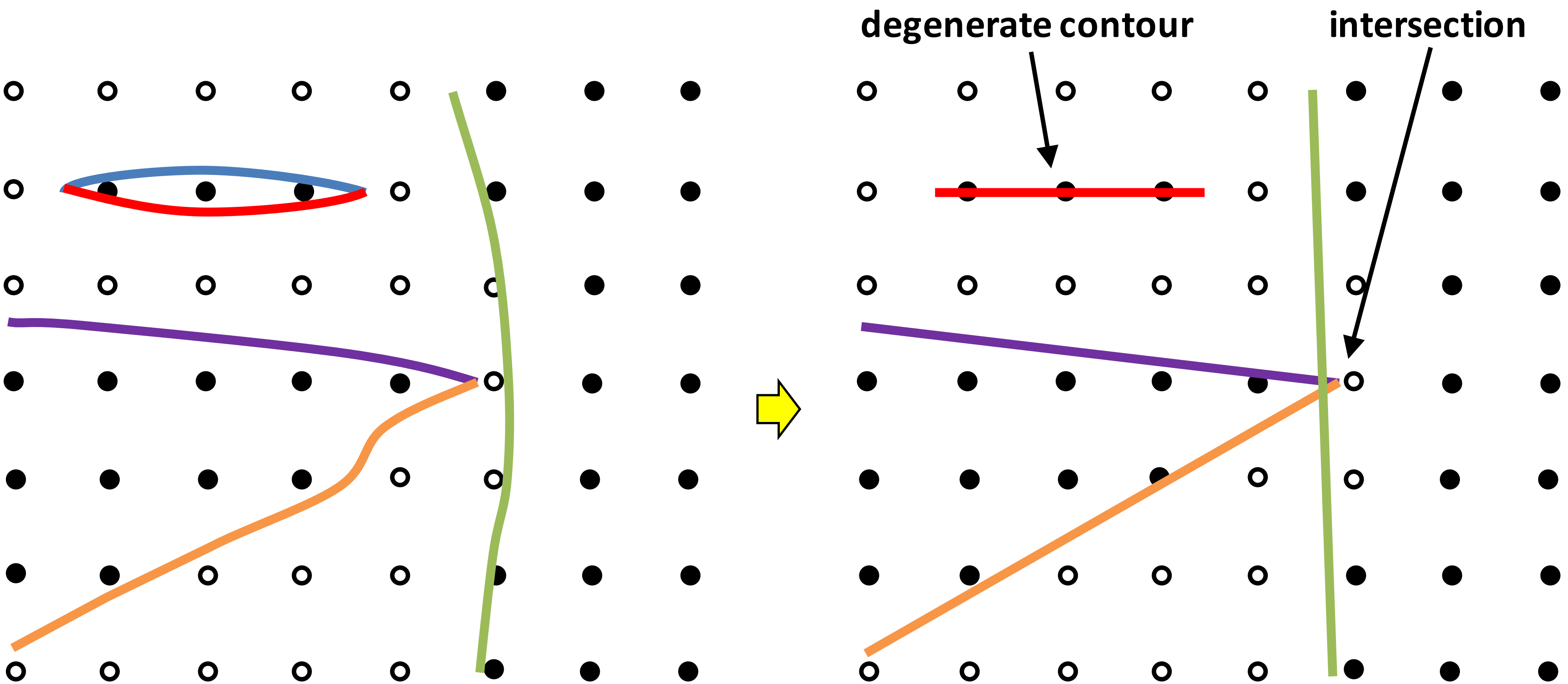
Binary Image Sampling by using the concept of r-regular to guarantee the topological faithful

In the Stages 2 and 3, the self-intersection must be prevented by the stick-concept when sliding on the edges

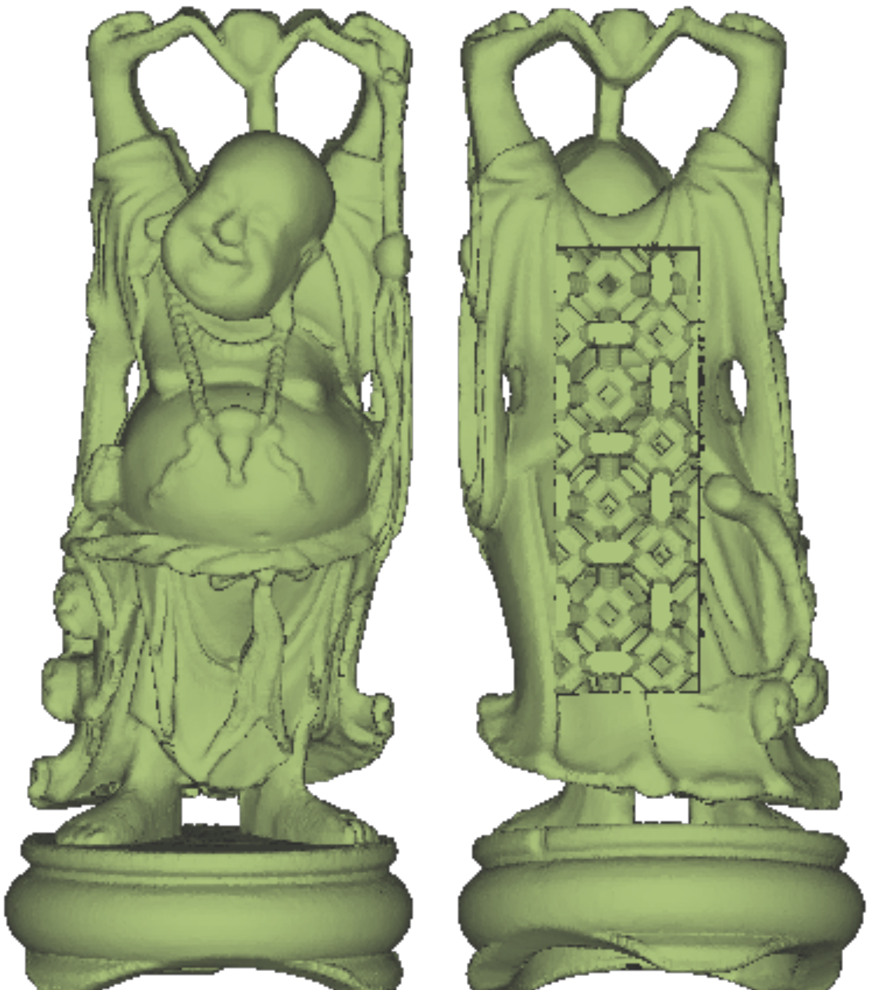


Self-intersection-free Contours

- Without snapping the contours on the edge-sticks, self-intersection happens

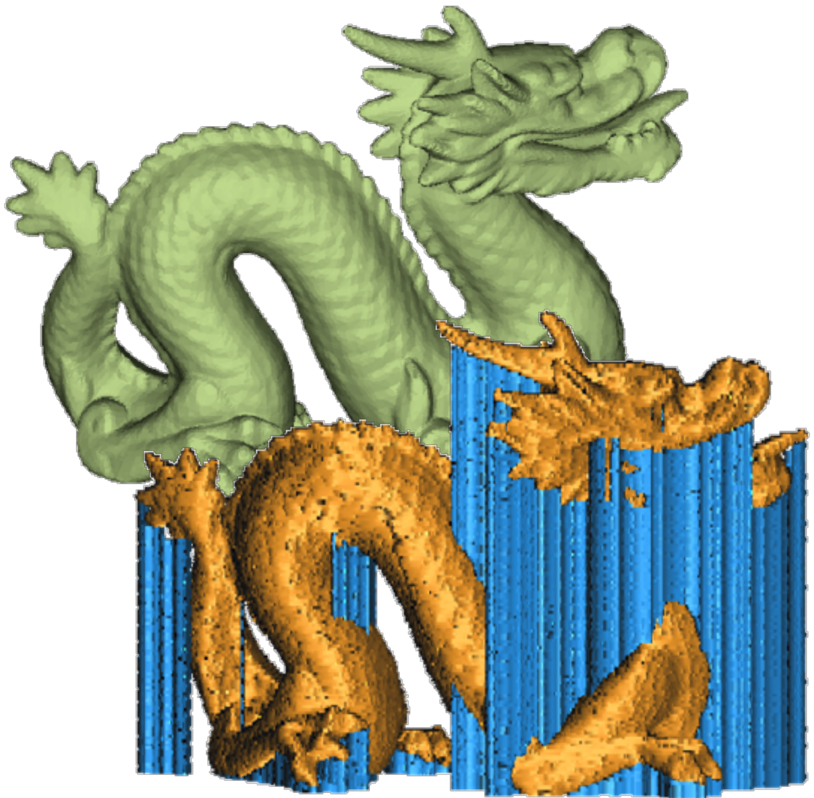


Topological Faithful Contouring Result

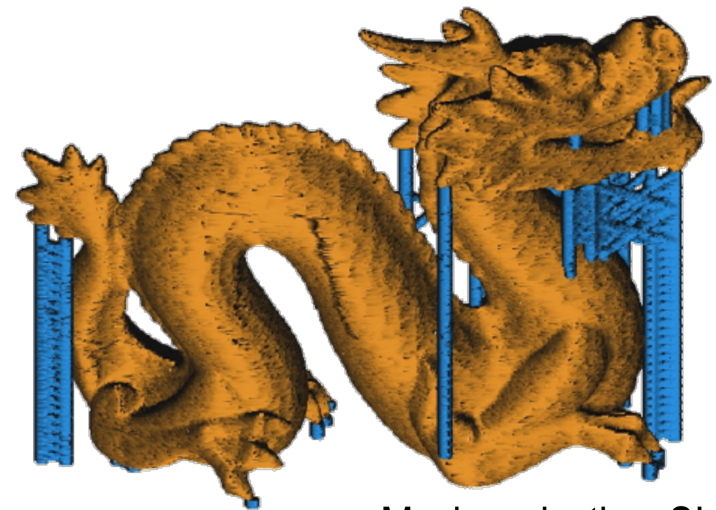


Pu Huang, Charlie C.L. Wang, and Yong Chen, "Intersection-free and topologically faithful slicing of implicit solid", ASME Journal of Computing and Information Science in Engineering, vol.13, no.2, 021009 (13 pages), June 2013.

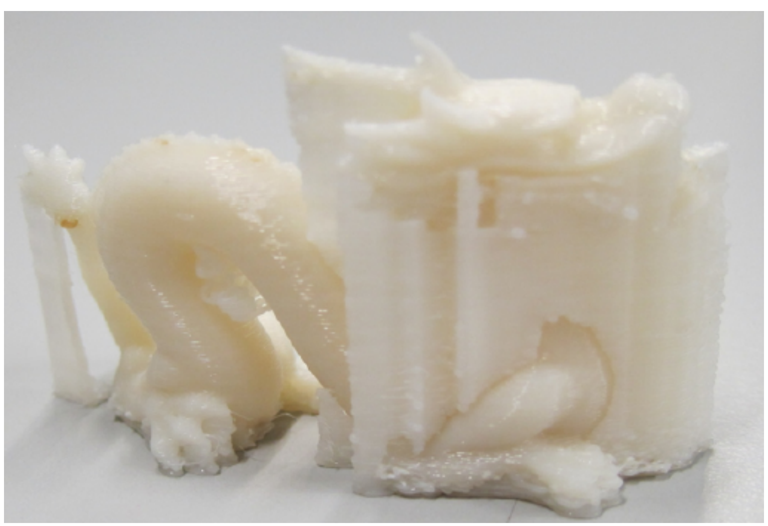
Supporting Structure?

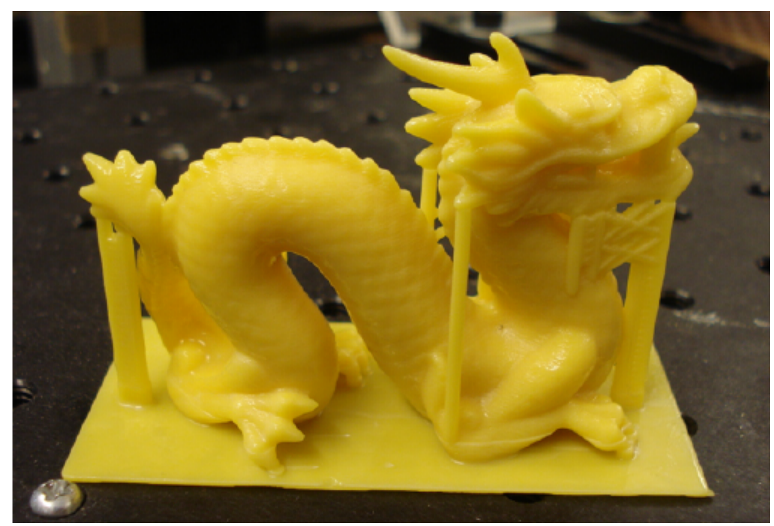


FDM



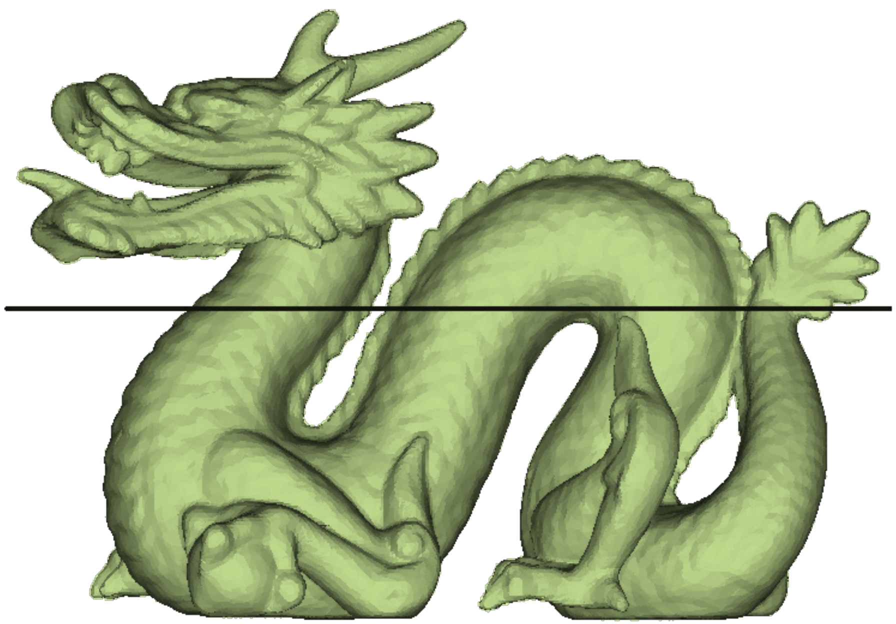
Mask-projection SLA

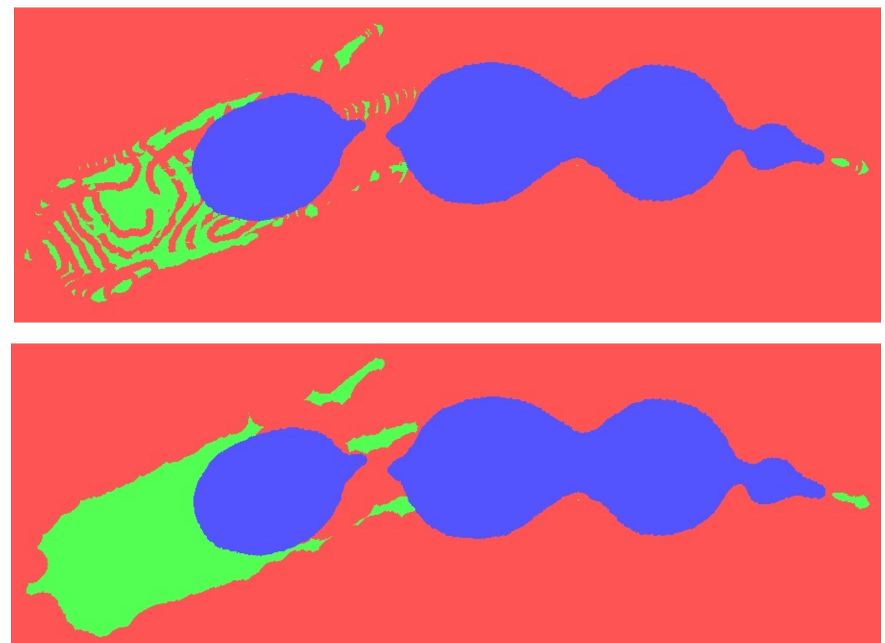




Algorithms for Generating Supporter

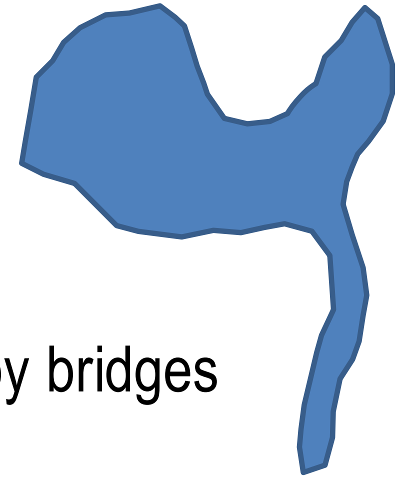
- FDM's supporter is based on *Reliable & Robust Region Subtraction*
- Dilation and erosion must be applied to remove those self-supported regions
- Numerical pruning as a post-processing step is needed

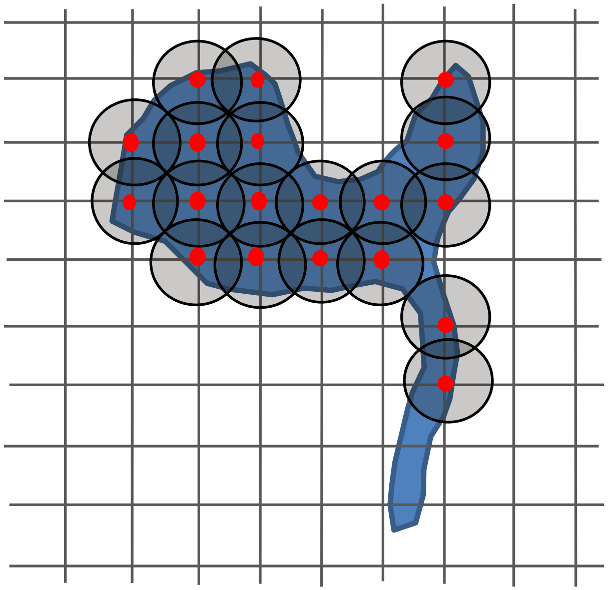


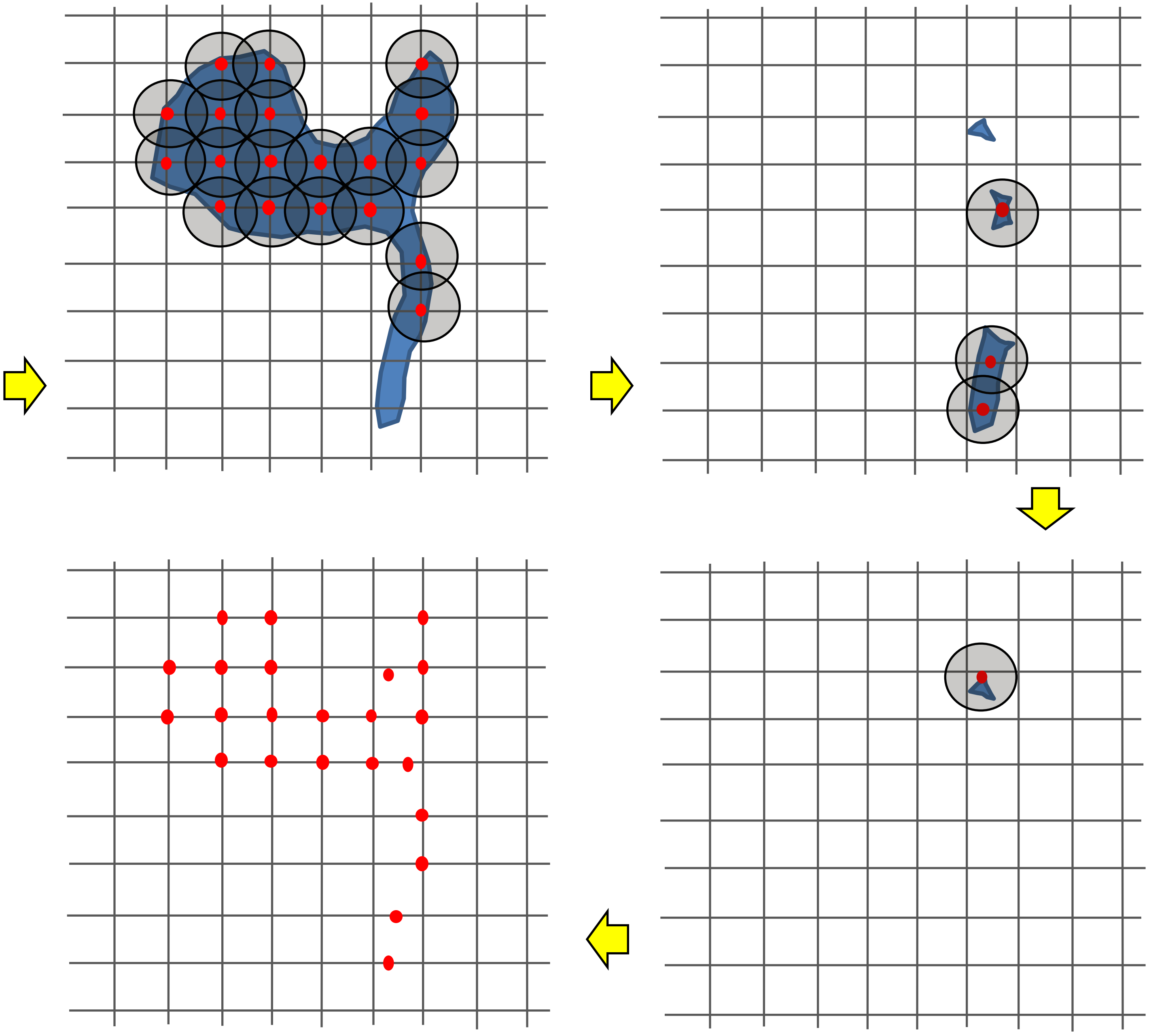


Algorithms for Generating Supporter

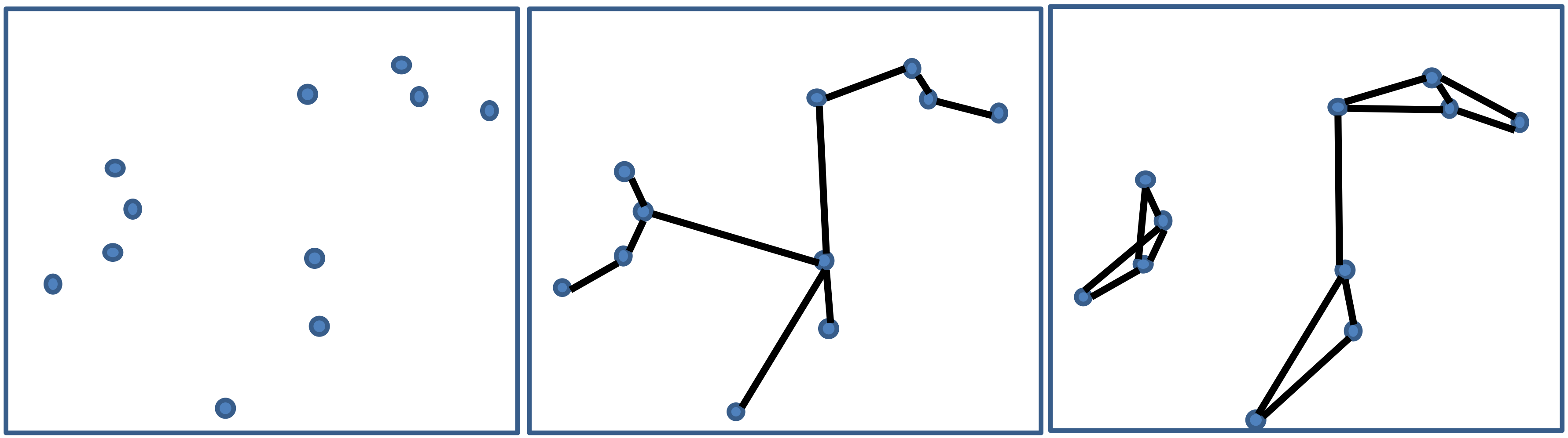
- SLA is based on *Region Subtraction* but using *Anchor Maps*
- Anchor maps are used to represent regions and also take the region subtraction
- Scanning orders:
  - Grid Nodes
  - Grid edges
  - Remaining region
- Linking anchor points by bridges







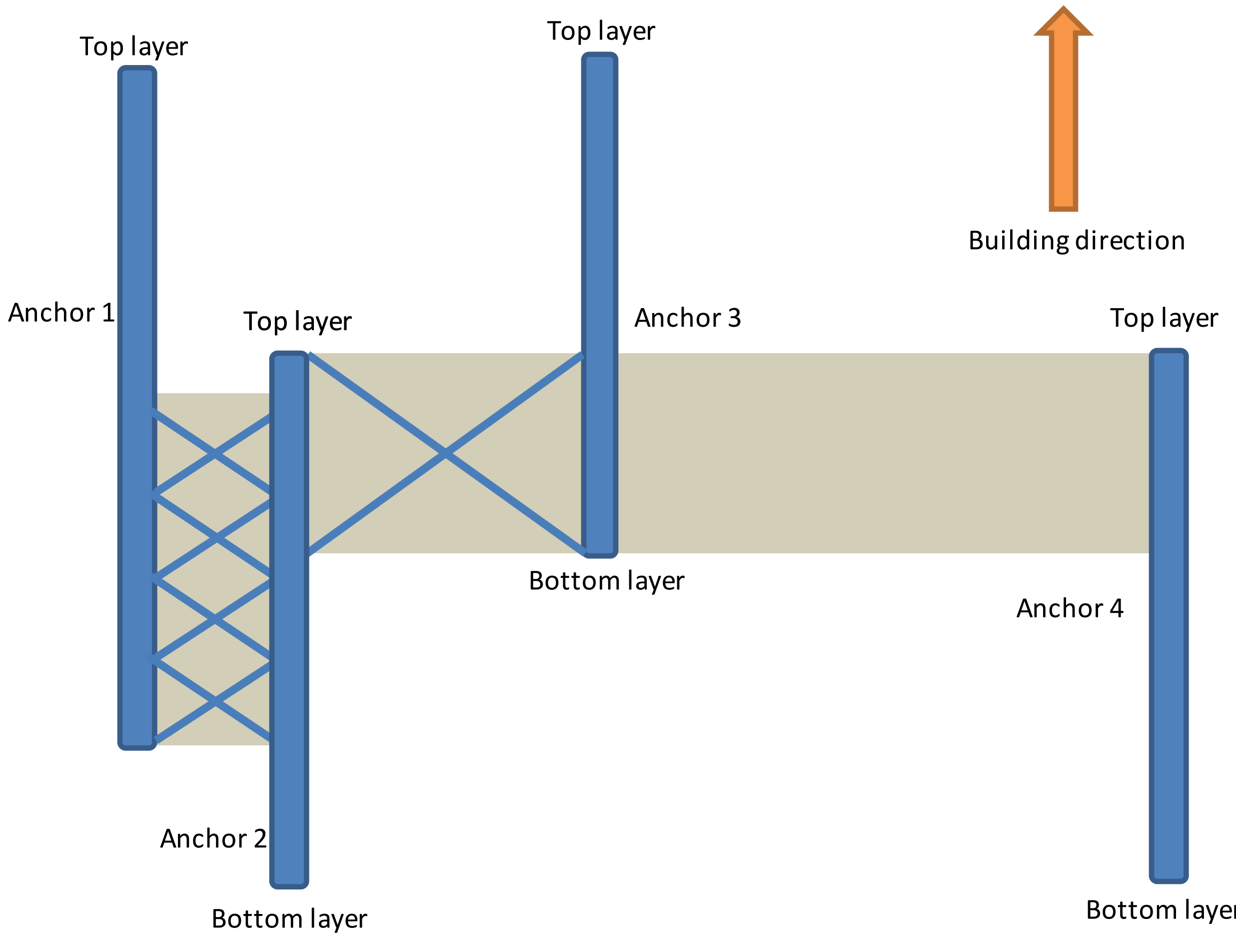
Linking Anchors by Bridges

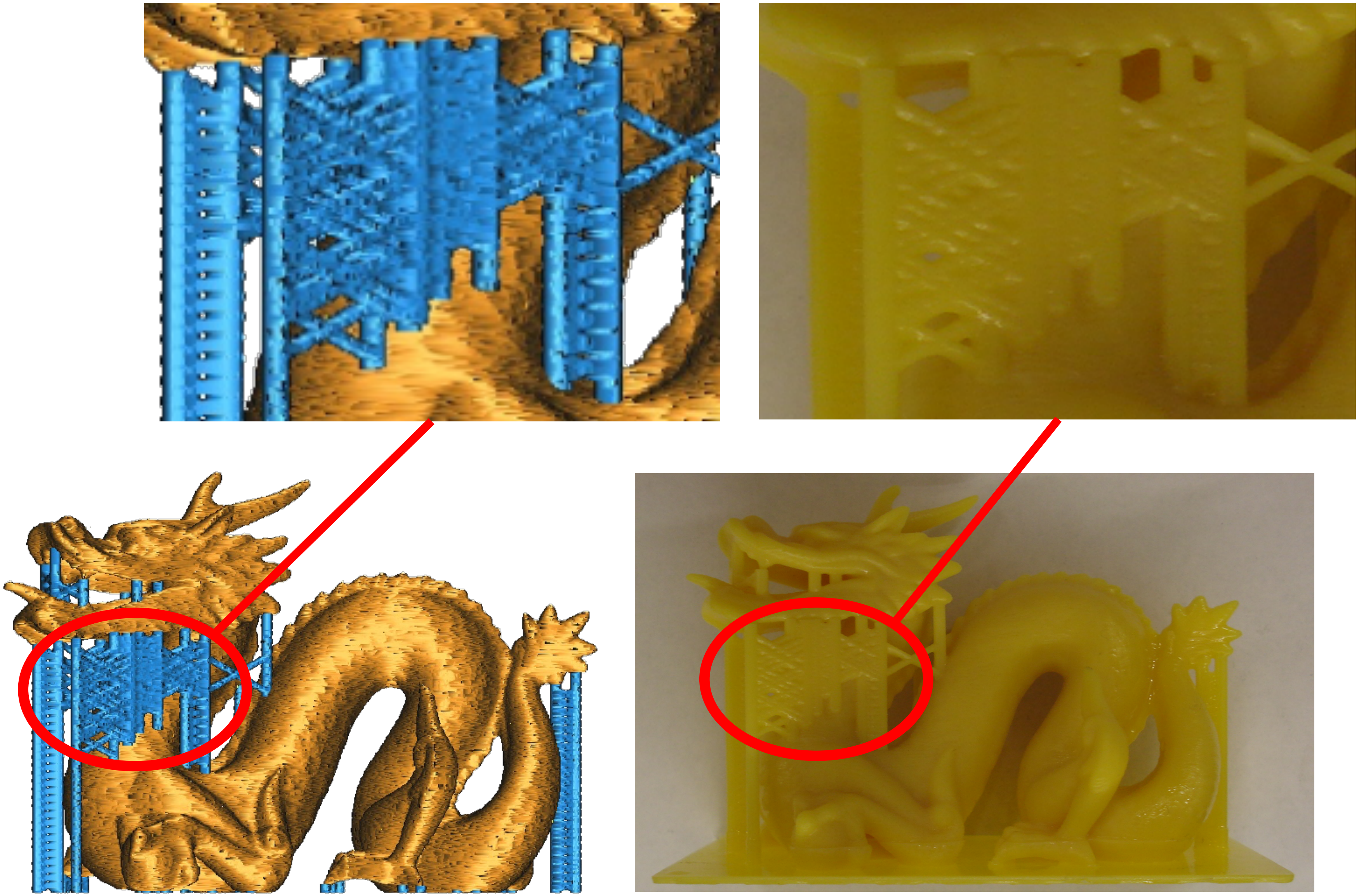


- One approach is based on *Minimal Spanning Tree* (MST)
- Another is based on closest neighbor search
- Which is better? The latter one.
- For building a long bridge, the mechanical stiffness is not good.

Linking Anchors by Bridges (cont.)

- Anchors are located in different heights





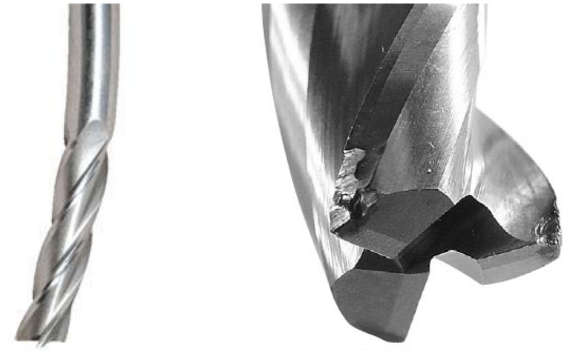
Pu Huang, Charlie C.L. Wang, and Yong Chen, "Algorithms for layered manufacturing in image space", Book Chapter, ASME Advances in Computers and Information in Engineering Research, 2014.

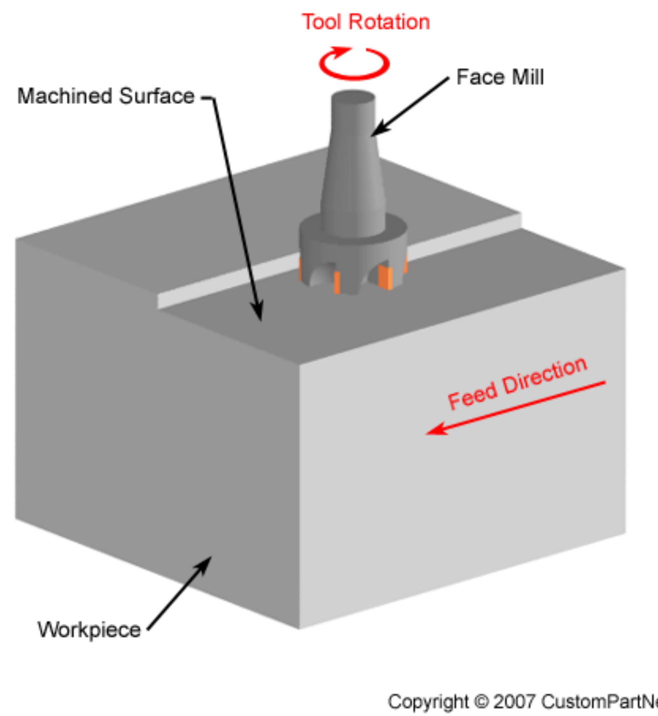
LDNI-Based Solid Modeling

http://ldnibasedsolidmodeling.sourceforge.net

Problem in CNC Based Mass Production

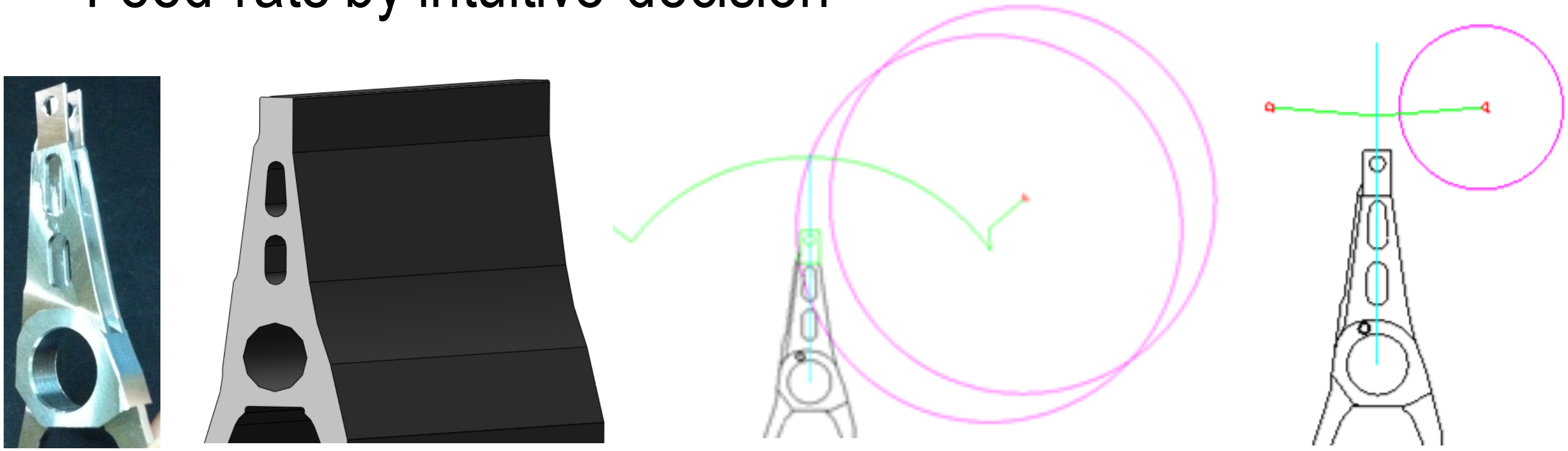
- Mostly *NOT* in an open-architecture
  - Difficult to online monitoring / adapting
- Too fast or Too slow
  - Damage on surface / tool?
  - Inefficient?
- Tool-path has been given
- Remained variables to tune:
  - Feed-rate of cutter engagement
  - Tuning speed





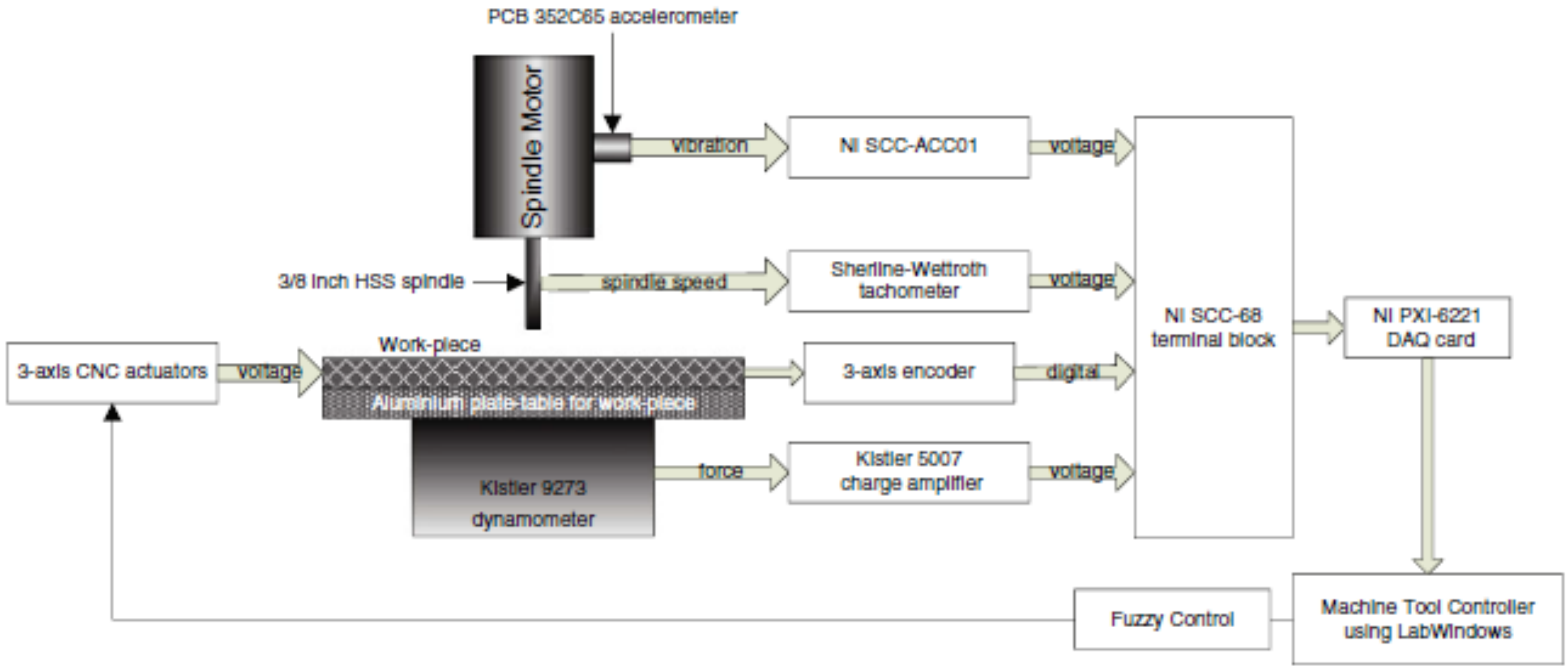
Problem in CNC Machining (Cont.)

- Real Scenario
  - High precision parts
  - CNC machine with closed system
  - Big cutter – large volume removal
  - Weak stiffness at spindle
- Tool-path by designed shape
- Feed-rate by intuitive decision



Possible Solutions

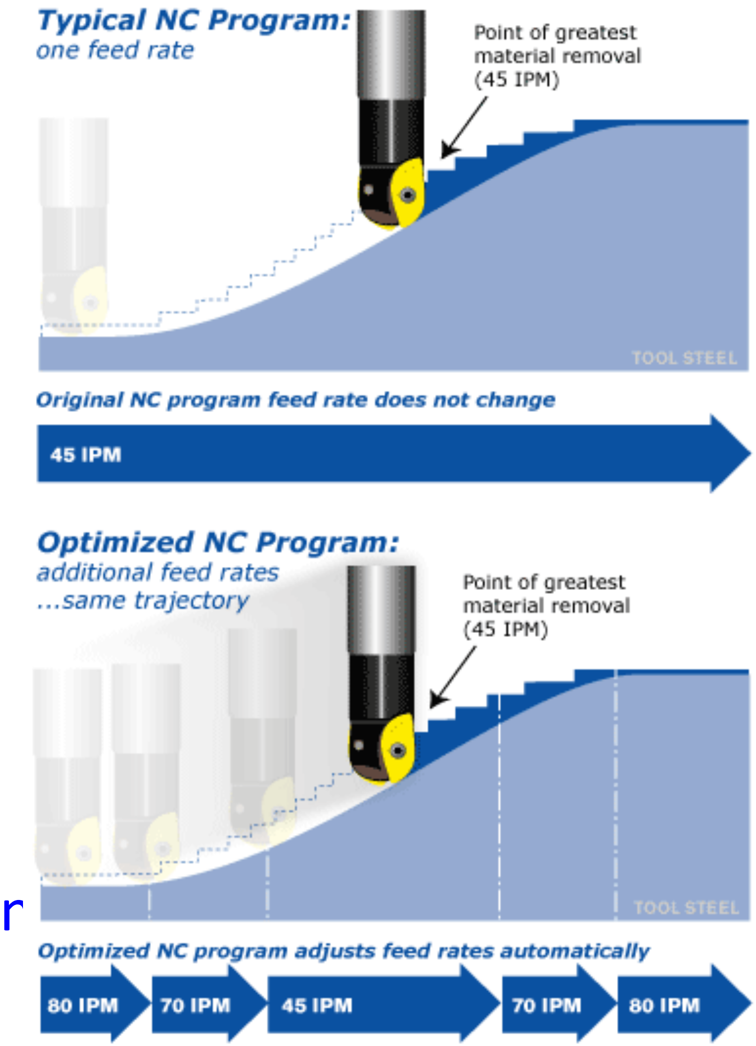
- Online adaptive control



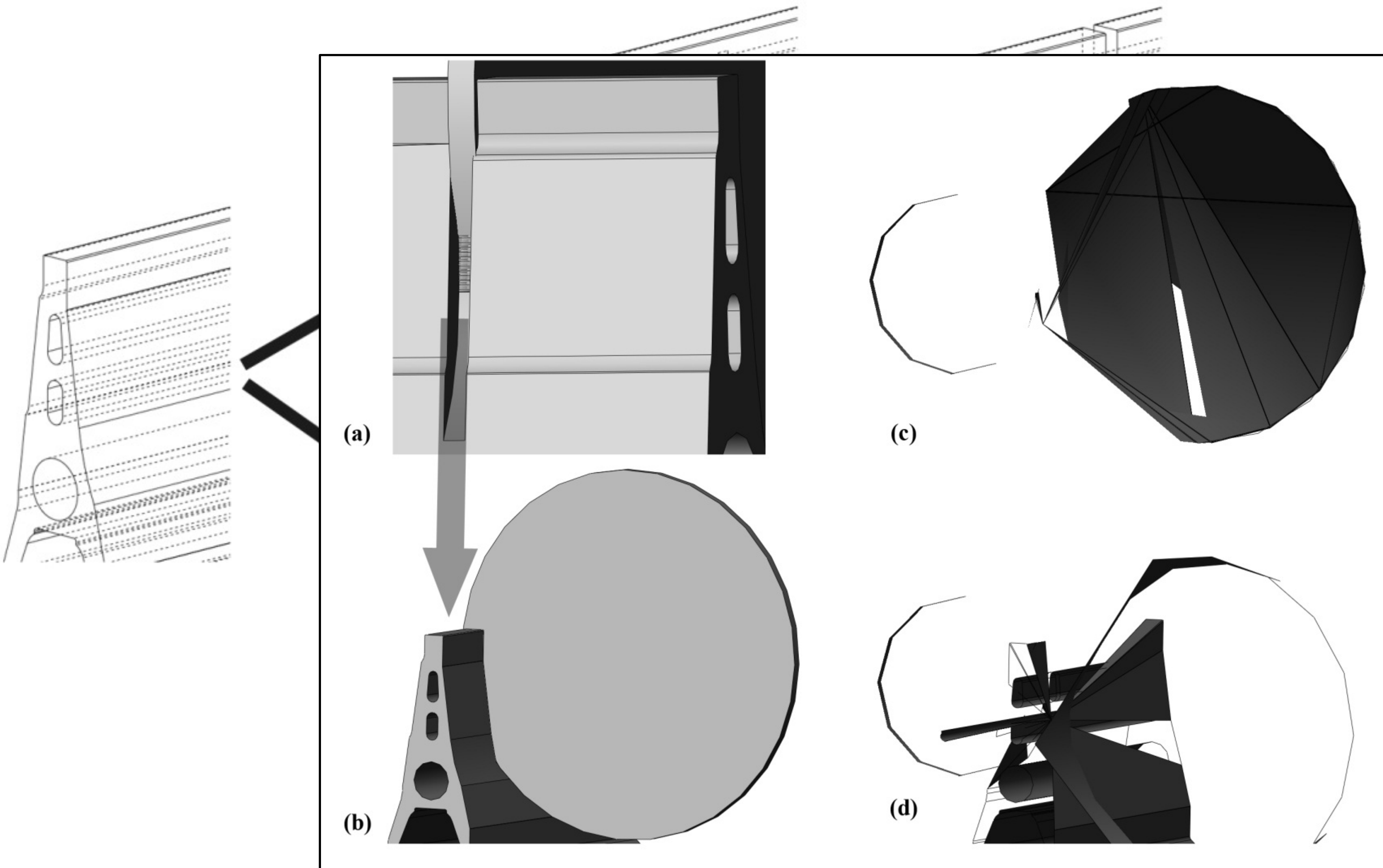
- Additional setup + need to be open-architecture

Possible Solutions

- Offline simulation
- Adjusting the feed-rates
- Could based on:
  - Chip thickness
  - Material removal rate (MRR)
  - Maximal acceleration
  - Force-model
- MRR is employed here
  - A simple but general solution
  - In the past, very coarse level
  - Lacks of MRR at high resolutior



Problem of Robustness (B-rep Modeler)

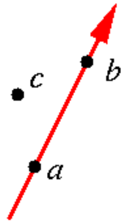


Problem of Conventional B-rep Modeler

- Numerical robustness
- Computation in IEEE arithmetic
  - Limited precision of floating-point arithmetic
- Geometry becomes inexact after intersection
- Geometric predicates
  - Correct?
  - Intersected models?
  - Membership classification?
- Exact representation?
  - Multiple precision arithmetic library
  - Plane-based representation

Orientation

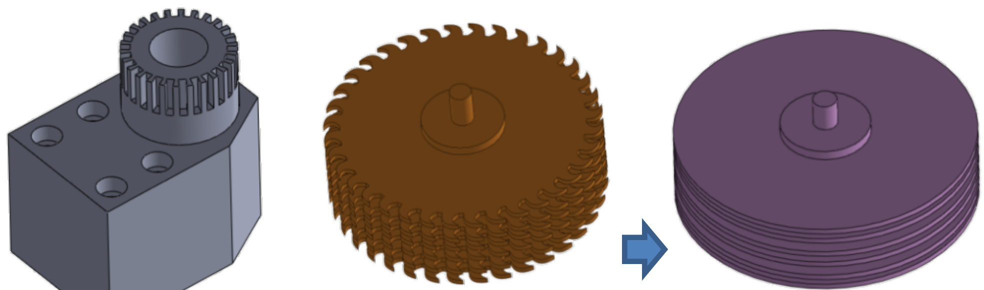
Does c lie on, to the left of, or to the right of $\overrightarrow{ab}$?



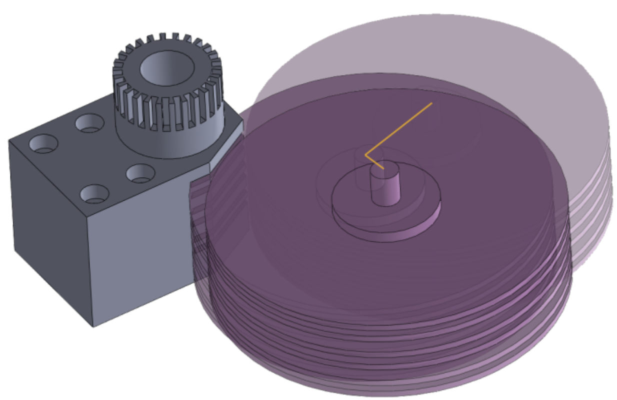
$$\begin{vmatrix} a_x & a_y & 1 \\ b_x & b_y & 1 \\ c_x & c_y & 1 \end{vmatrix} = \begin{vmatrix} a_x - c_x & a_y - c_y \\ b_x - c_x & b_y - c_y \end{vmatrix}$$

Based on Efficient Boolean Operation

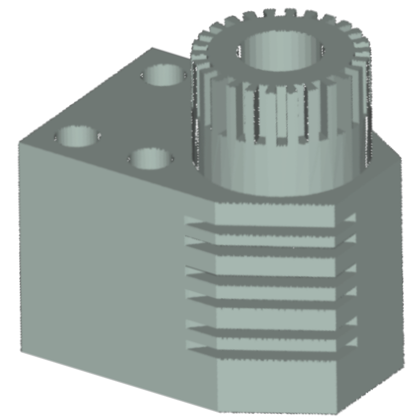
- GPU-based Solid Modeling Kernel



Appropriate Simplification



Procedural Simulation

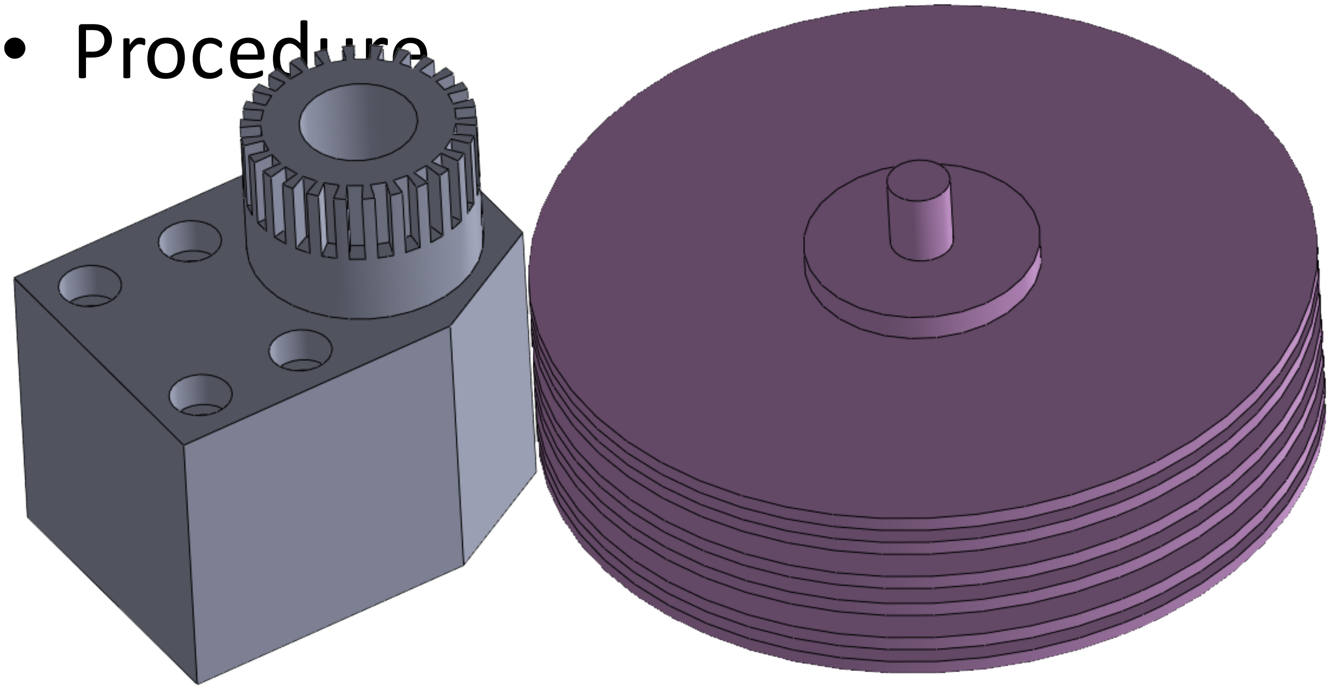


Simulation Result

http://ldnibasedsolidmodeling.sourceforge.net

Based on Efficient Boolean Operation

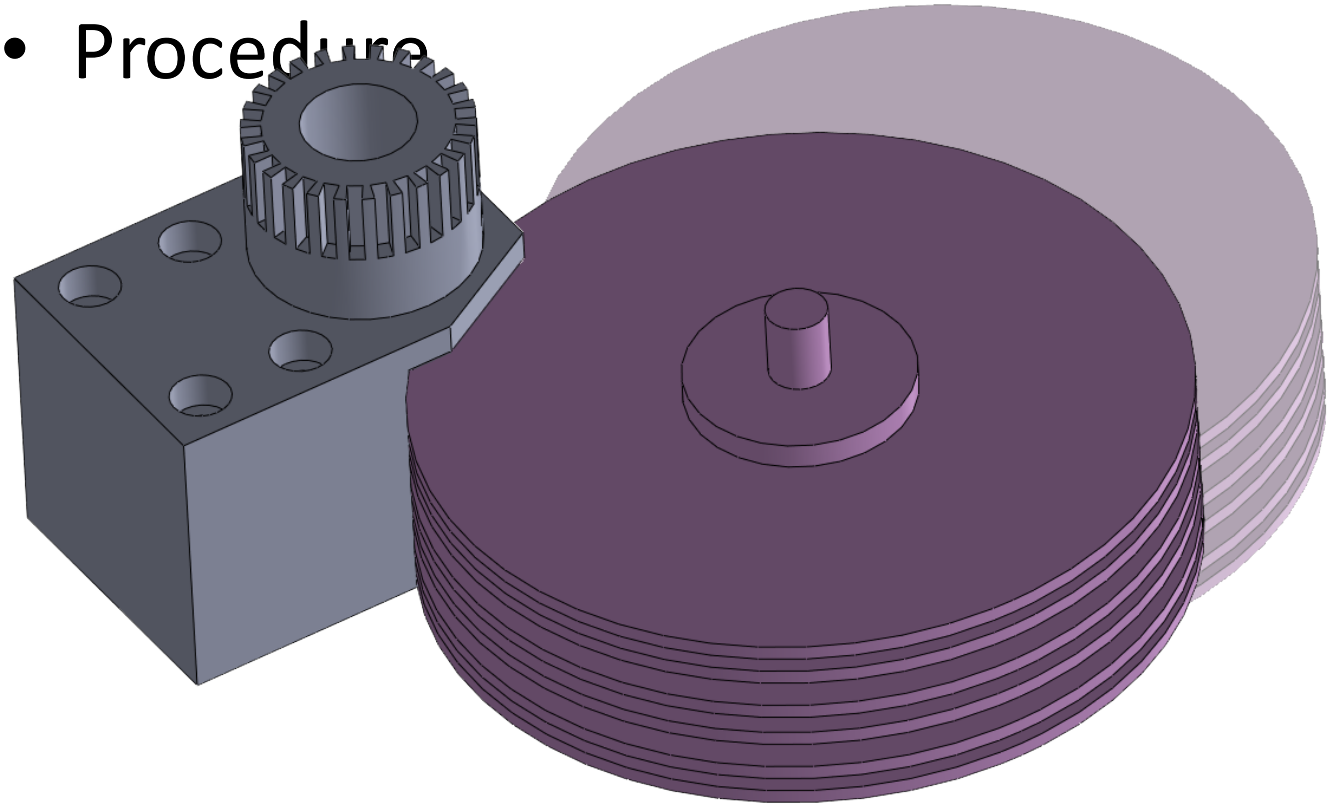
- Procedure



```
(T10D100X2.2/3.7 SAW)
G00X40Y58M03S10000
G43H10Z-12.5
G00Z-12.5
G01X40Y58F600
G01X52Y32
G01X60Y32
G00Z15.0
M99
%
```

Based on Efficient Boolean Operation

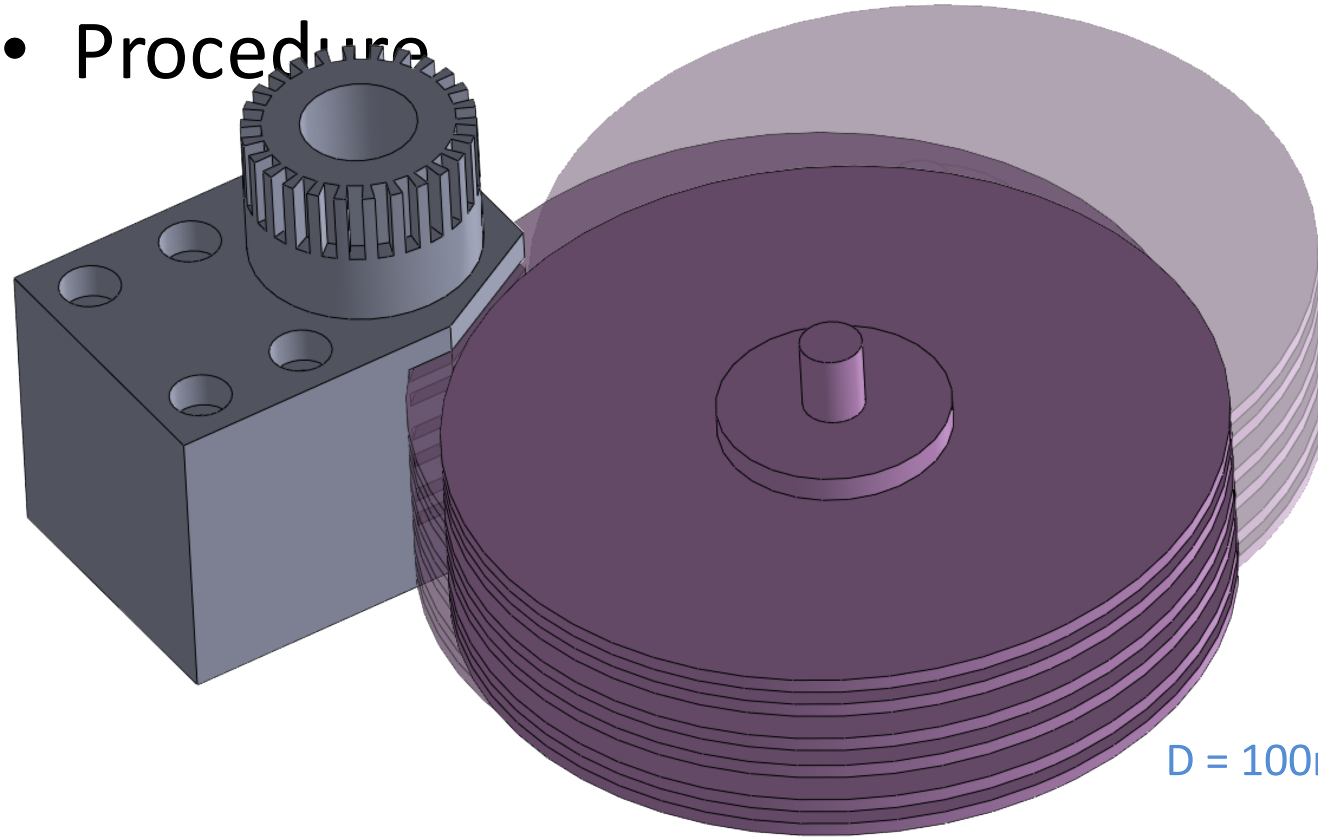
- Procedure



```
(T10D100X2.2/3.7 SAW)
G00X40Y58M03S10000
G43H10Z-12.5
G00Z-12.5
G01X40Y58F600
G01X52Y32
G01X60Y32
G00Z15.0
M99
%
```

Based on Efficient Boolean Operation

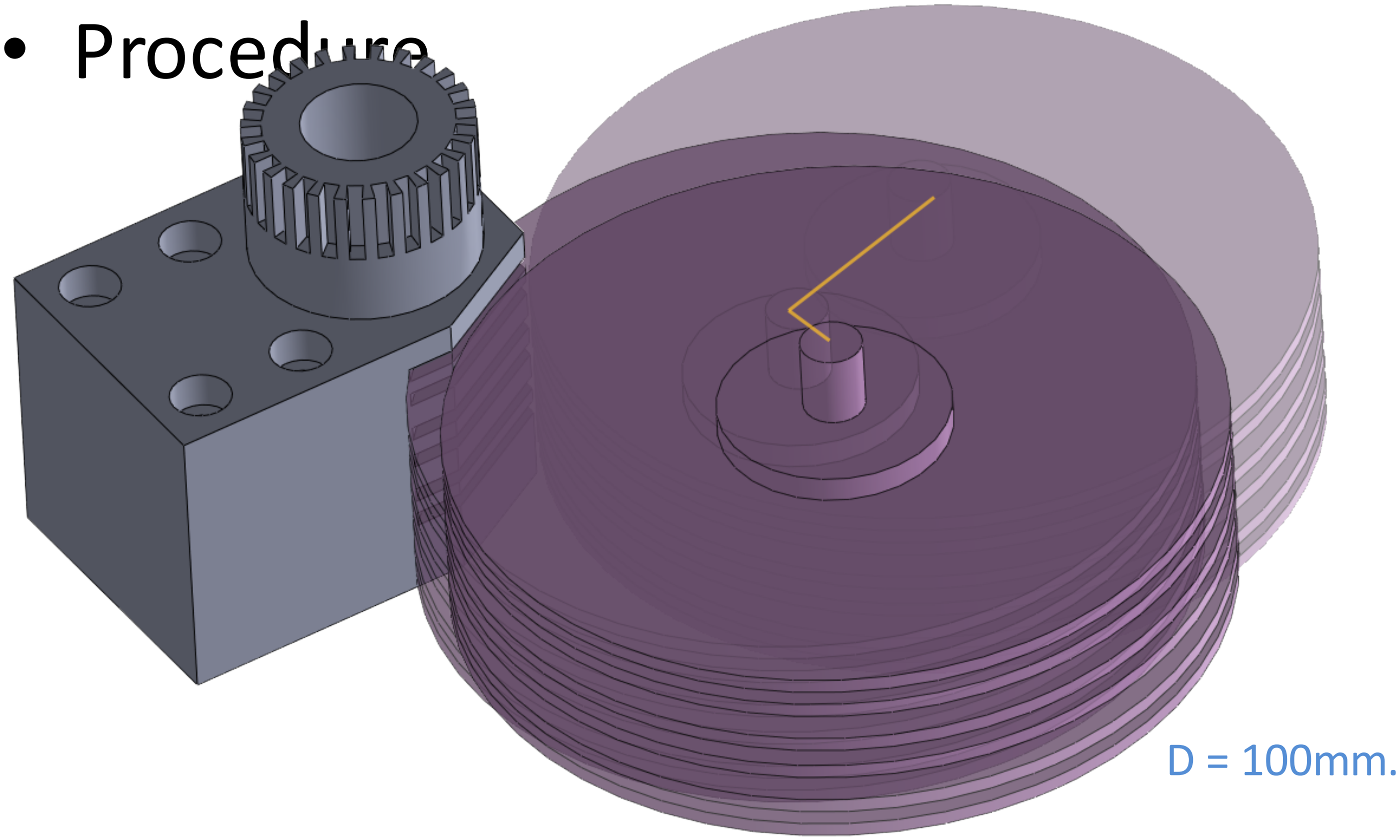
- Procedure




```
(T10D100X2.2/3.7 SAW)
G00X40Y58M03S10000
G43H10Z-12.5
G00Z-12.5
G01X40Y58F600
G01X52Y32
G01X60Y32
G00Z15.0
M99
%
```

Based on Efficient Boolean Operation

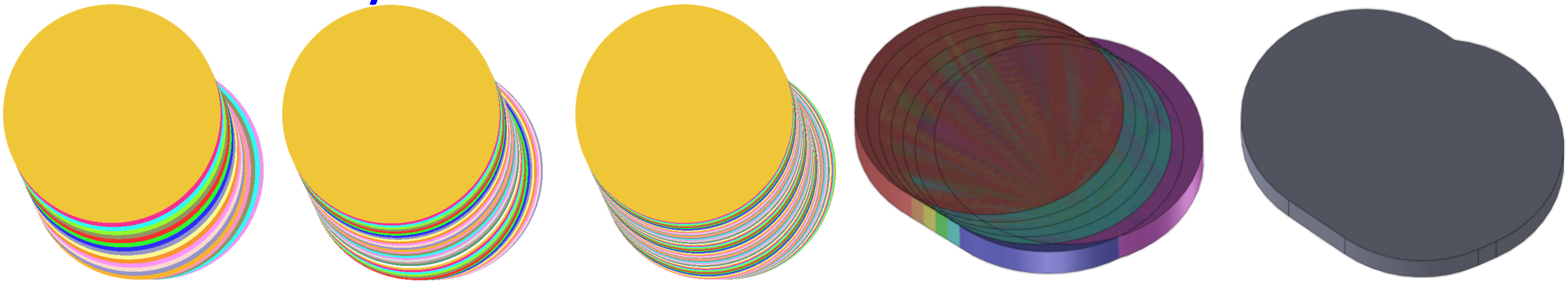
- Procedure




```
(T10D100X2.2/3.7 SAW)
G00X40Y58M03S10000
G43H10Z-12.5
G00Z-12.5
G01X40Y58F600
G01X52Y32
G01X60Y32
G00Z15.0
M99
%
```

Using Dense Sampling for Sweeping

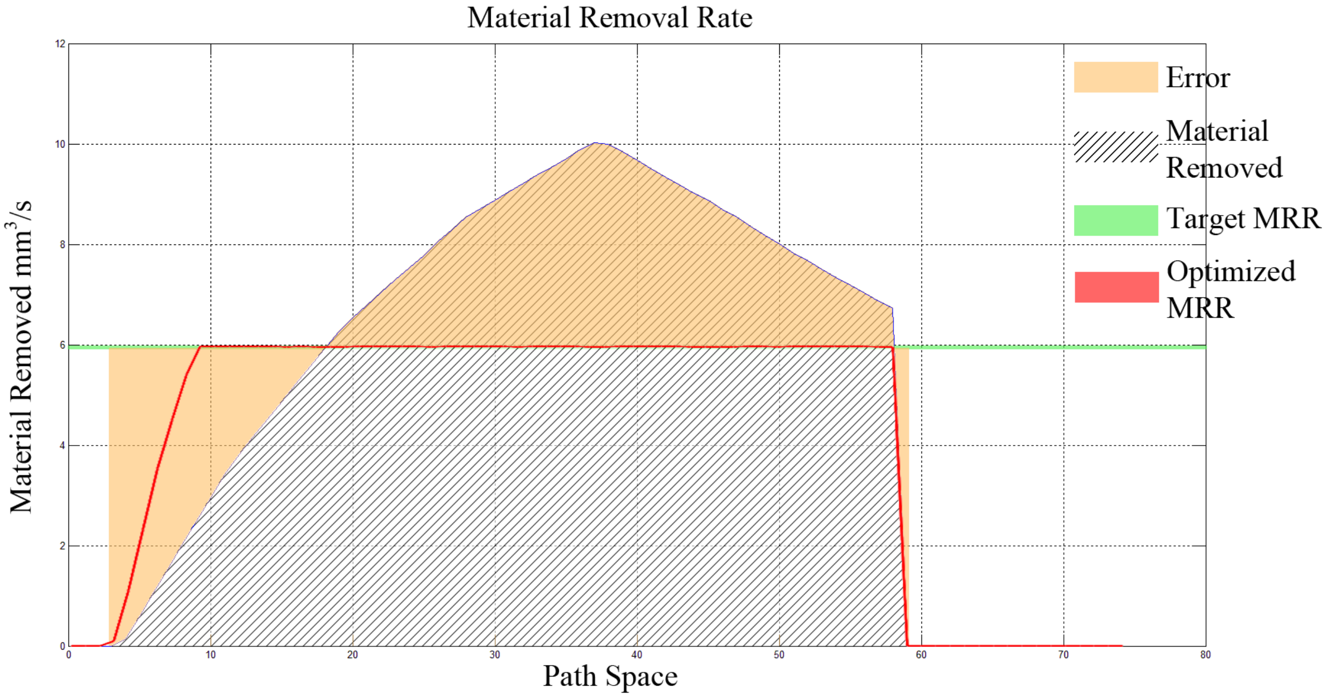
- More dense sampling results in better accuracy



D(mm)	Calculated volume	Analytic Volume	Error %:
2.0	30785.7	30801.4	0.0509%
1.0	30786.5	“”	0.0482%
0.8	30786.8	“”	0.0473%
0.5	30787.0	“”	0.0467%

Challenge of Feed-Rate Variation

- High resolution MRR histogram – cannot change feed-rate too frequently
- Constant MRR can only be realized when goes to infinity



Computed in 109 sec. with the help of GPU-based solid modeling

MRR-based Feed-Rate Optimization

- **Objective I** – The MRR, $R(t)$, is controlled within $[R_{min}, R_{max}]$ during the machining.
- **Objective II** – While achieving a bounded MRR in the range mentioned above, the number of variations of feed-rates must be minimized (Crucial & Not Guaranteed in Prior Research).
- To meet these objectives by a progressive segmentation
  - Generate the histogram of MRR at very high resolution;
  - A hybrid subdivision algorithm is developed to meet the demand of *Objective II*;
    - The given tool-path is segmented into sub-regions (Greedy);
    - Different feed-rates are assigned to each sub-regions (Objective I).

Segmentation Algorithm

- Mainly three steps:
  1) Selecting the next group of engagement to divide;
  2) Locating a best place to conduction the subdivision;
  3) Assigning feed-rates to the newly created groups of engagement.
- These three steps are repeatedly applied until
  - TC1: MRRs in all engagement fall into the range of [R_{min}, R_{max}]
  - or TC2: the number of groups has reached the allowed maximum
    - This is favorable to the old CNC machines cannot process too many blocks in G-code
    - Also, too frequently change feed-rates will result in bad dynamic performance during machining

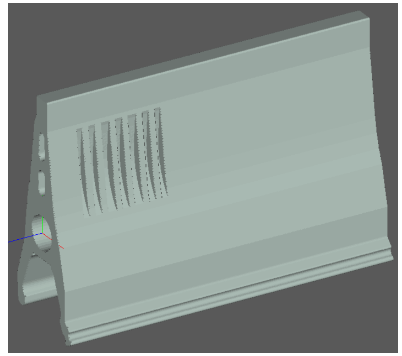
PROGRESSIVE SEGMENTATION FOR MRR-BASED FEED-RATE OPTIMIZATION IN CNC MACHINING

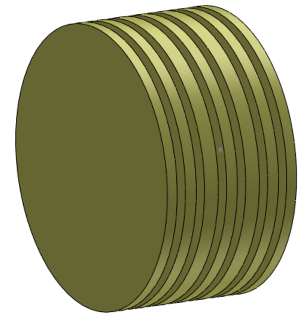
Ka-Chun Chan and Charlie C. L. Wang

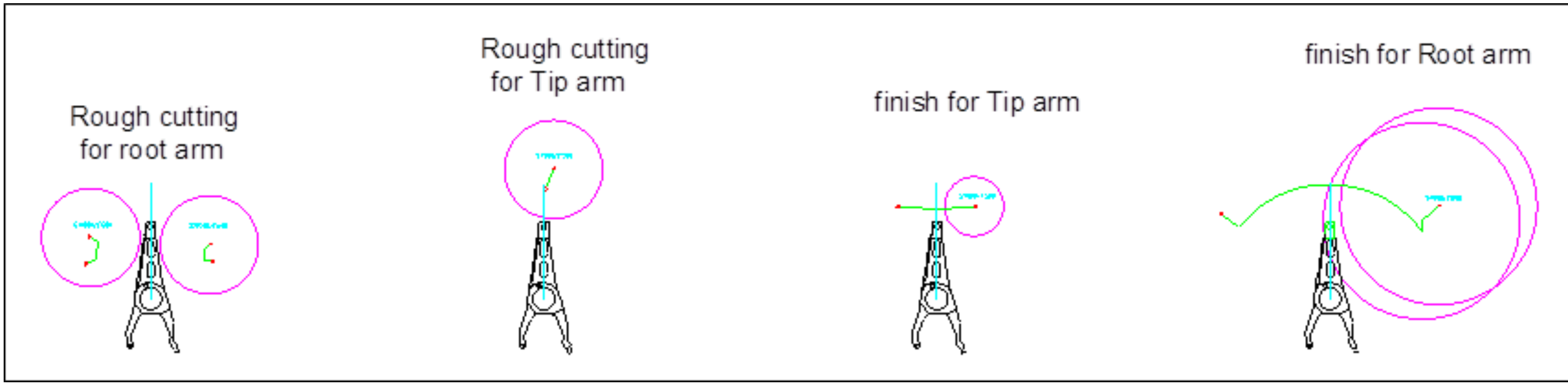
Department of Mechanical and Automation Engineering

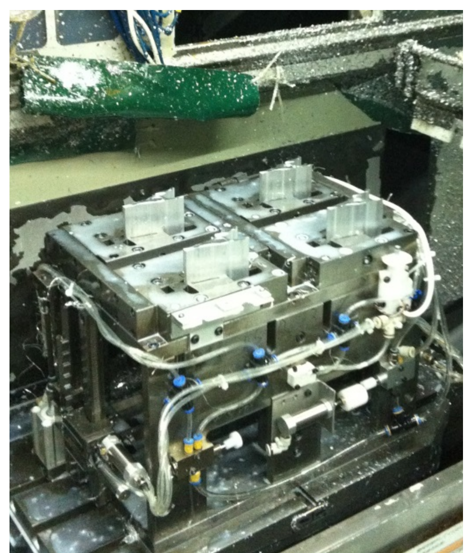
The Chinese University of Hong Kong

Case Study



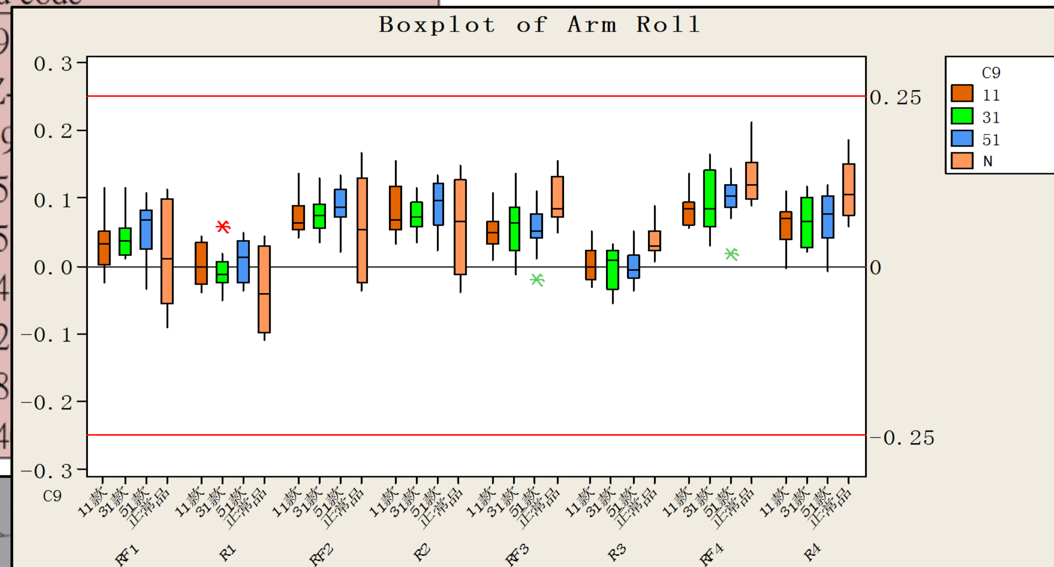


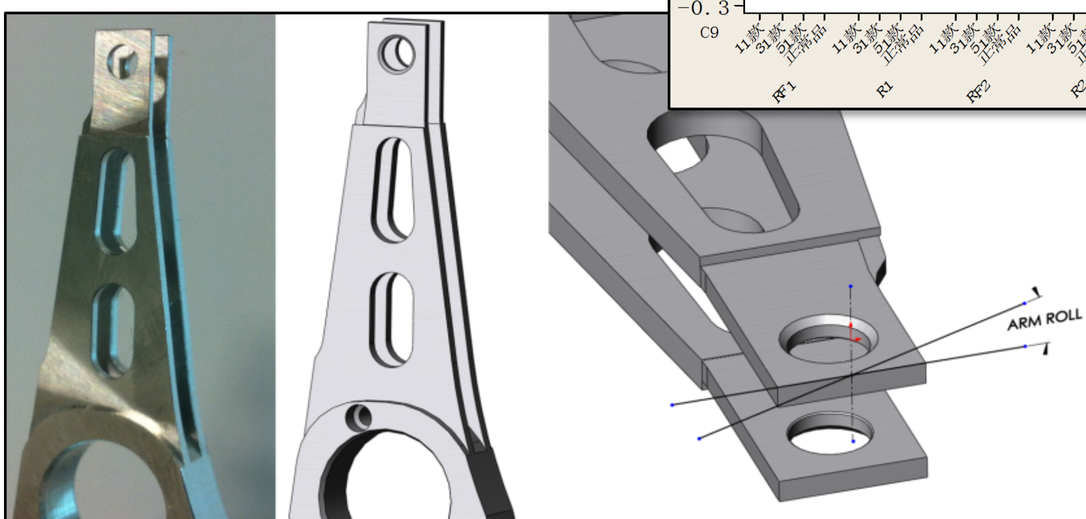

Rough cutting
for root arm
Rough cutting
for Tip arm
finish for Tip arm
finish for Root arm



Case Study

Original code	Optimized code
G00X29.9Y27.45Z0.0M04S4250	G00X29.9
G43H11Z-24.635	G43H11Z-
G00Z-24.900F5000	G00Z-24.9
	G01X28.5
	G01X28.5
	G01X28.4
	G01X28.2
	G01X27.8
G01X27.400Y25.100F159	G01X27.4





Smaller deviation

More trials towards *zero* arm-roll

Ka-Chun Chan, and Charlie C.L. Wang, "Progressive segmentation for MRR-based feed-rate optimization in CNC machining", 2015 IEEE International Conference on Automation Science and Engineering (CASE 2015)