SIGGRAPH2013

OpenFab

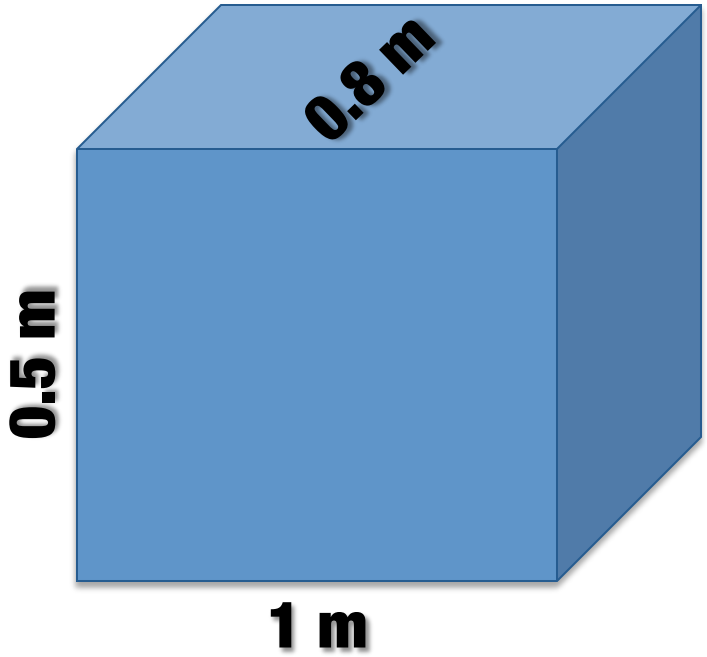
A Programmable Pipeline for Multi-Material Fabrication

Kiril Vidimče
Szu-Po Wang
Jonathan Ragan-Kelley
Wojciech Matusik
Computational Fabrication Group
Massachussetts Institute of Technology

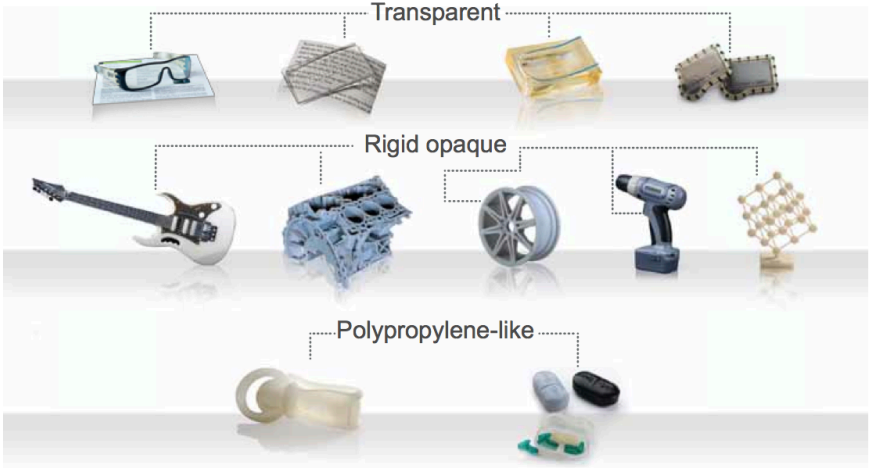
State of the Art of Multi-Material Fabrication

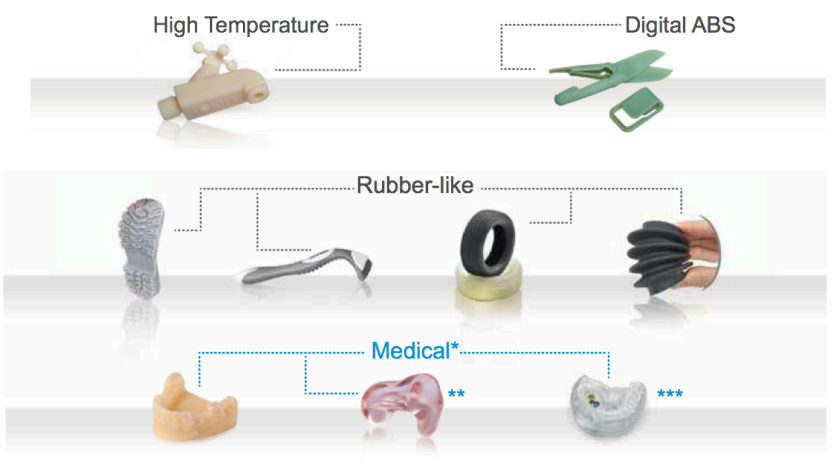
Multi-Material 3D Printers

Large Build Volume



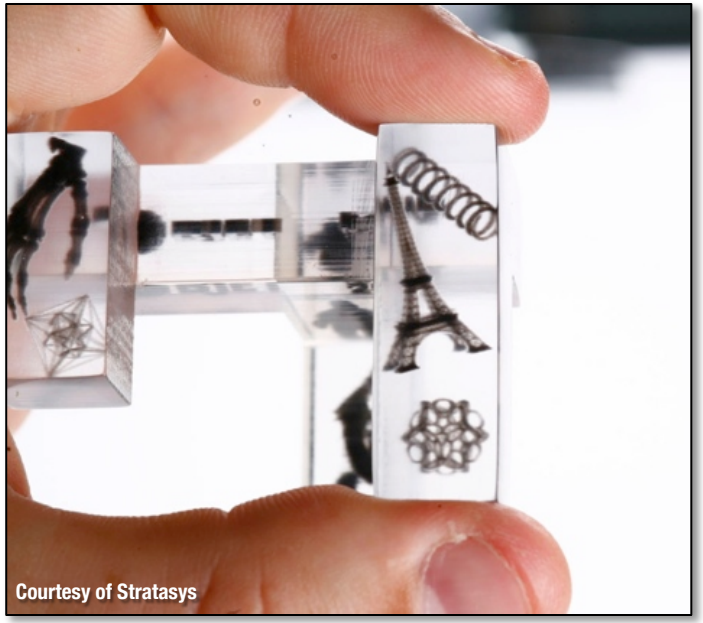
Large Material Library

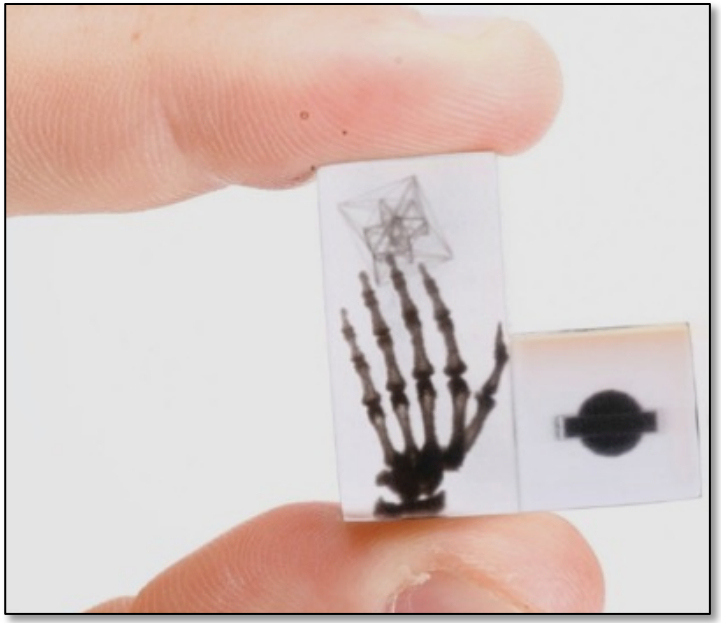




Courtesy of Stratasys

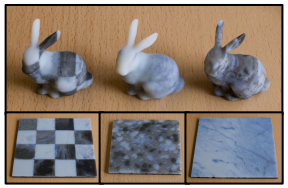
High-Resolution: 600 DPI

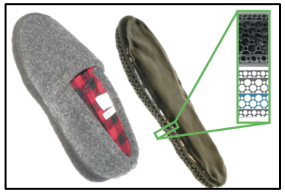
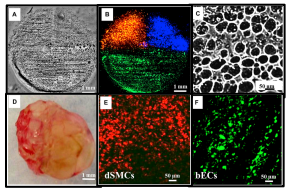


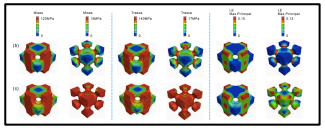


Recent Research Results

- BSSDRF 2010 [Hašan 2010]
- Deformation [Bickel 2010]
- Co-Continuous Polymers [Wang 2011]
- FGM Prototyping [Oxman 2011]
- Tissue Constructs [Xu 2012]
- Tough Composites [Dimas 2013]
- Actuated Characters [Skouras 2013]
- Lenticulars [Tompkin 2013]
- Printed Optics [Willis 2013]



Current Industry Use

One Material Per Part

Why?

Why One Material Per Part?

- Traditional constraints of manufacturing
- Poor specification methods
- Lack of scalable software architectures

Improve Specification Methods

- Functional specification
  - ✓ Spec2Fab
- Direct specification
  - ➡ OpenFab

Software Architecture Challenges

- Giga voxels/inch^3, Tera voxels/foot^3
- Continuous gradation between materials
- Reusable material definitions
- Resolution and printer independence

OpenFab

First Programmable and Scalable Fabrication Pipeline

OpenFab

- Inspired by rendering pipelines
- Fixed stages and programmable stages
- Procedural surface and material definitions
- Resolution independence
- Streaming architecture

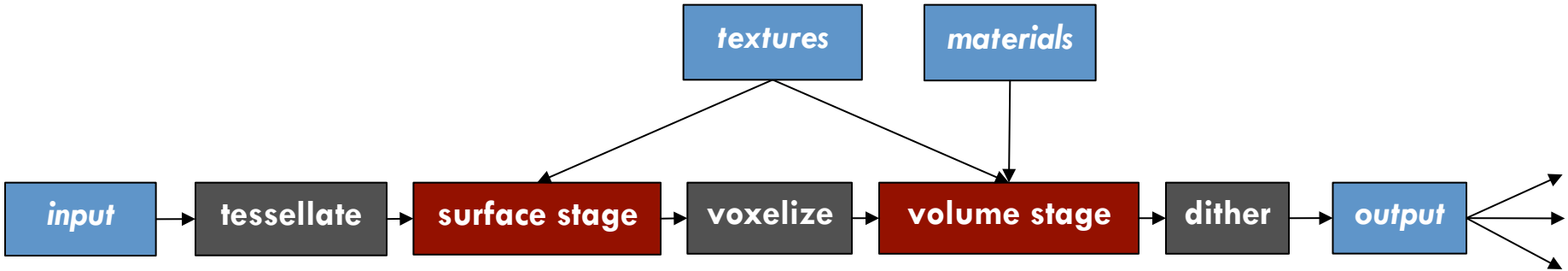
Outline

Outline

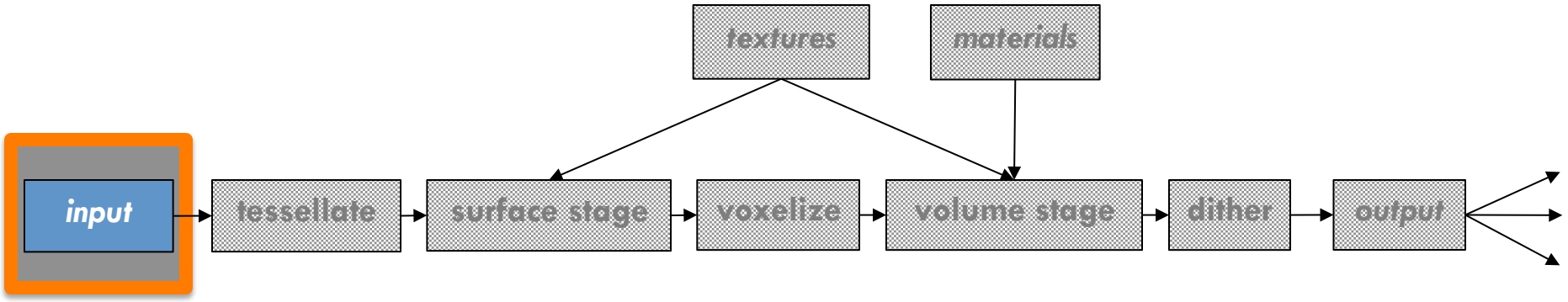
- OpenFab programming model
- OpenFL and *fablets*
- Architecture
- Results

The OpenFab Programming Model

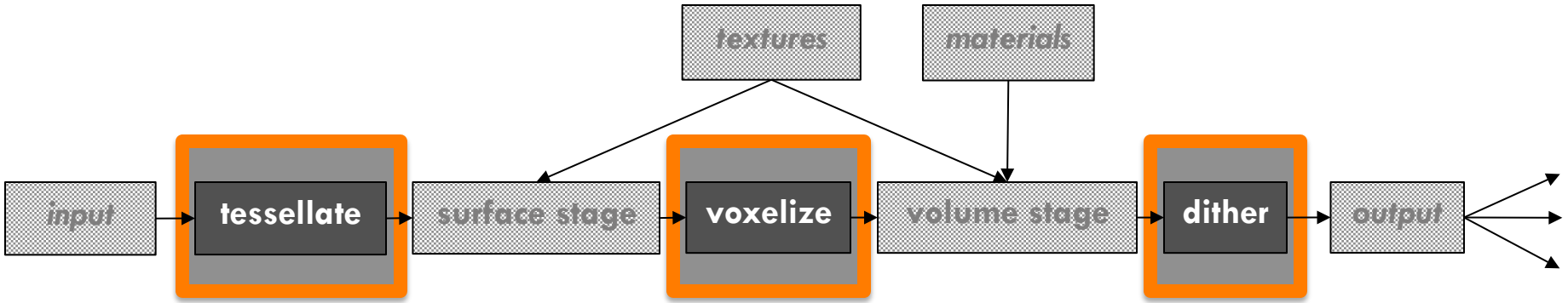
The OpenFab Programming Model



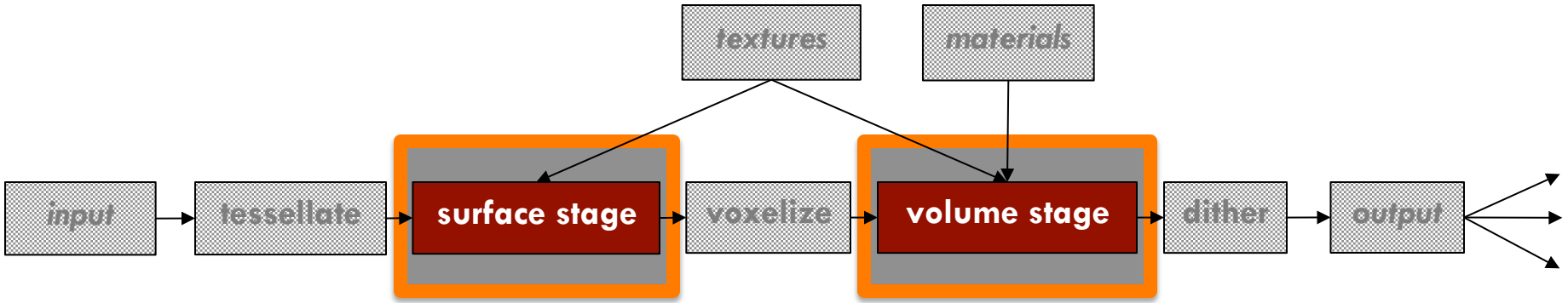
Input


textures
materials
input
tessellate
surface stage
voxelize
volume stage
dither
output

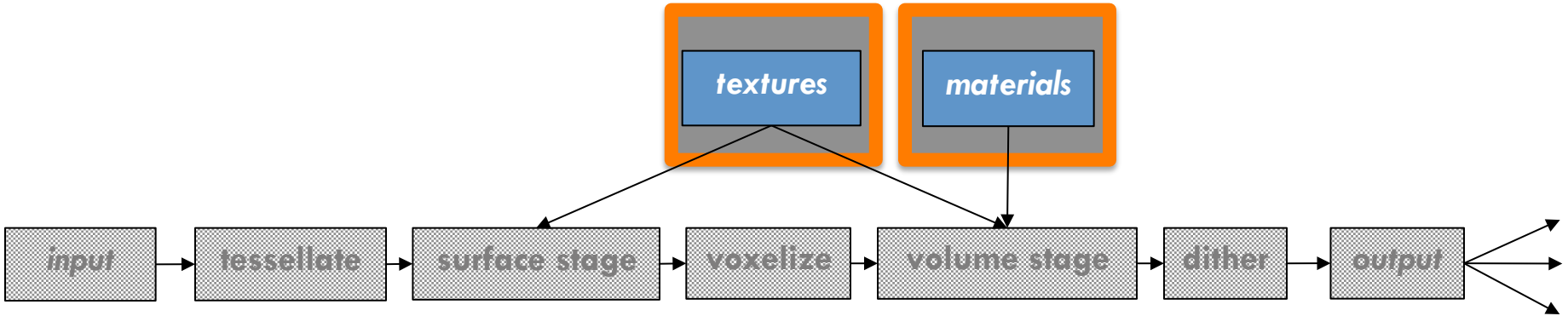
Fixed-Function Stages



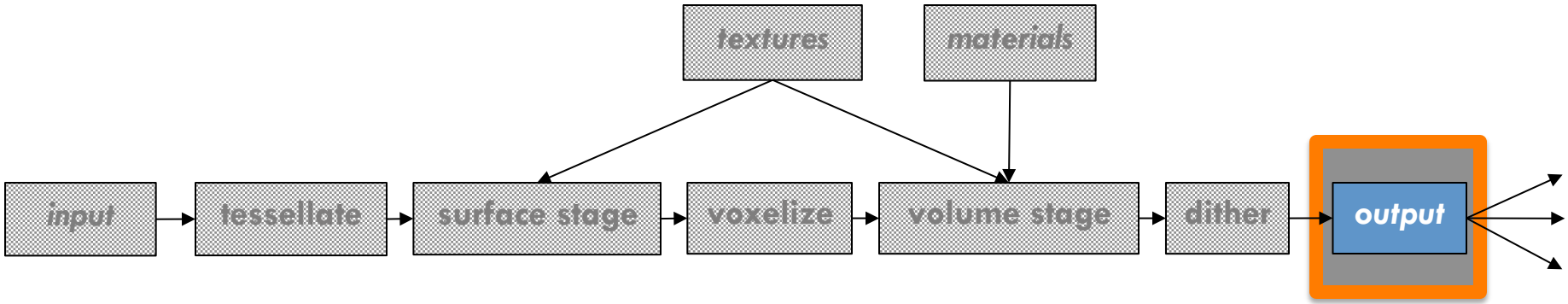
Programmable Stages



External Resources

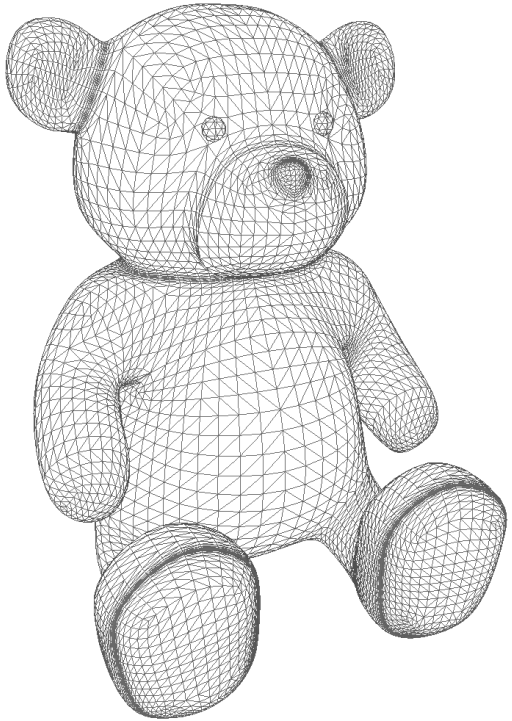


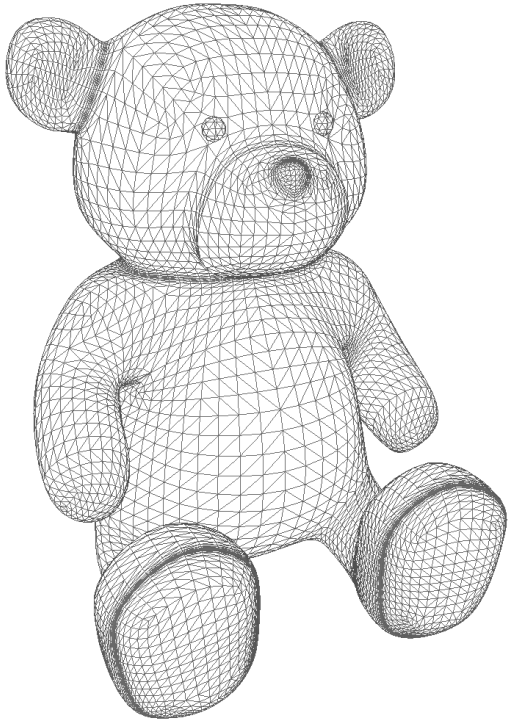
Output


textures
materials
input
tessellate
surface stage
voxelize
volume stage
dither
output


input
tessellate
surface stage
voxelize
volume stage
dither
output

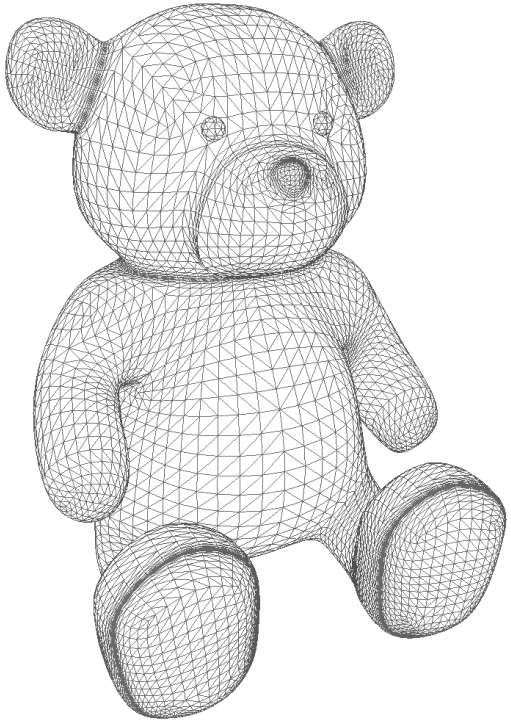

input
tessellate
surface stage
voxelize
volume stage
dither
output

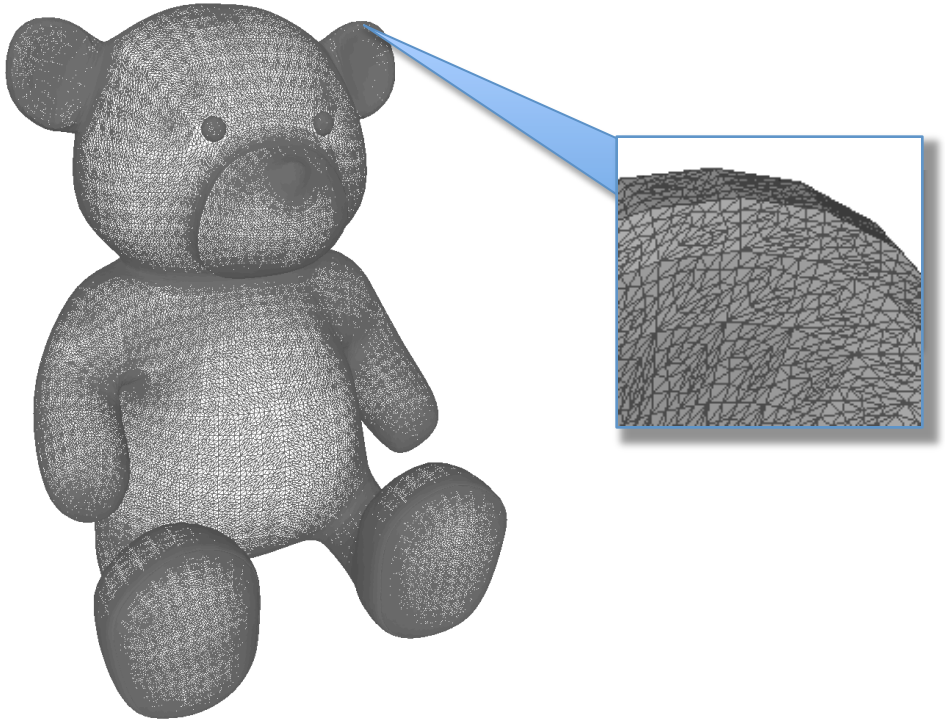




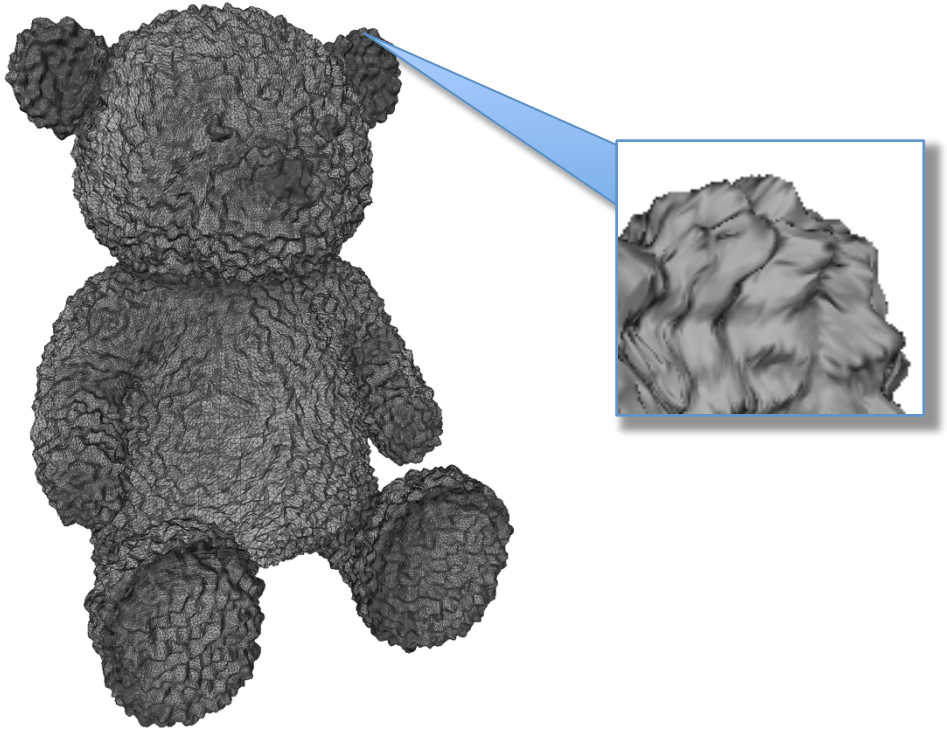
- Shapes (boundary representation)
- Shape priorities
- Fablets
- Resources
  - Textures
  - Materials


input
tessellate
surface stage
voxelize
volume stage
dither
output

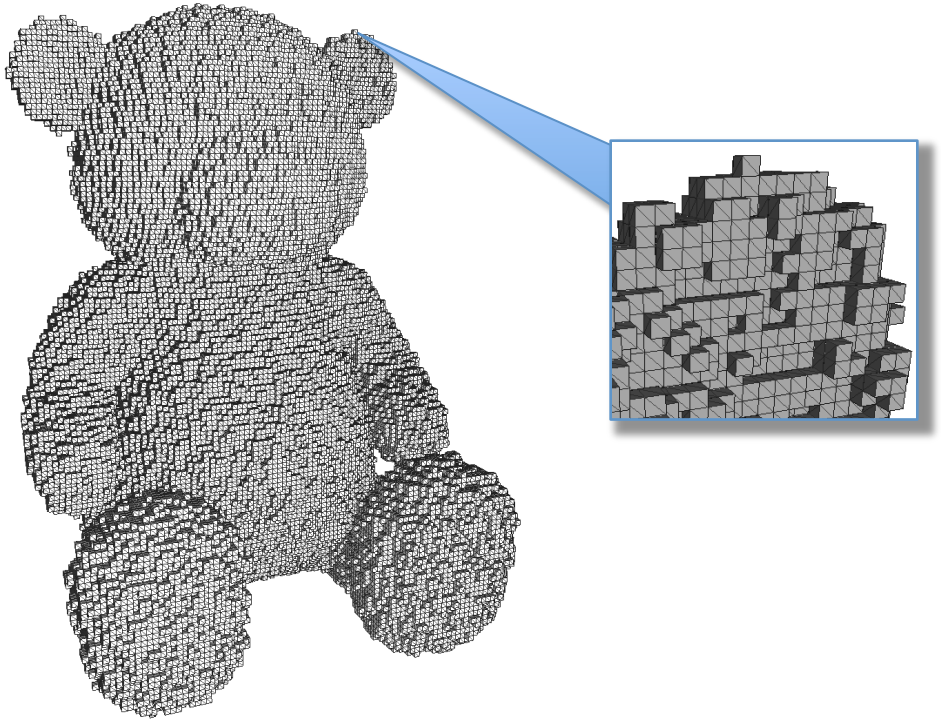




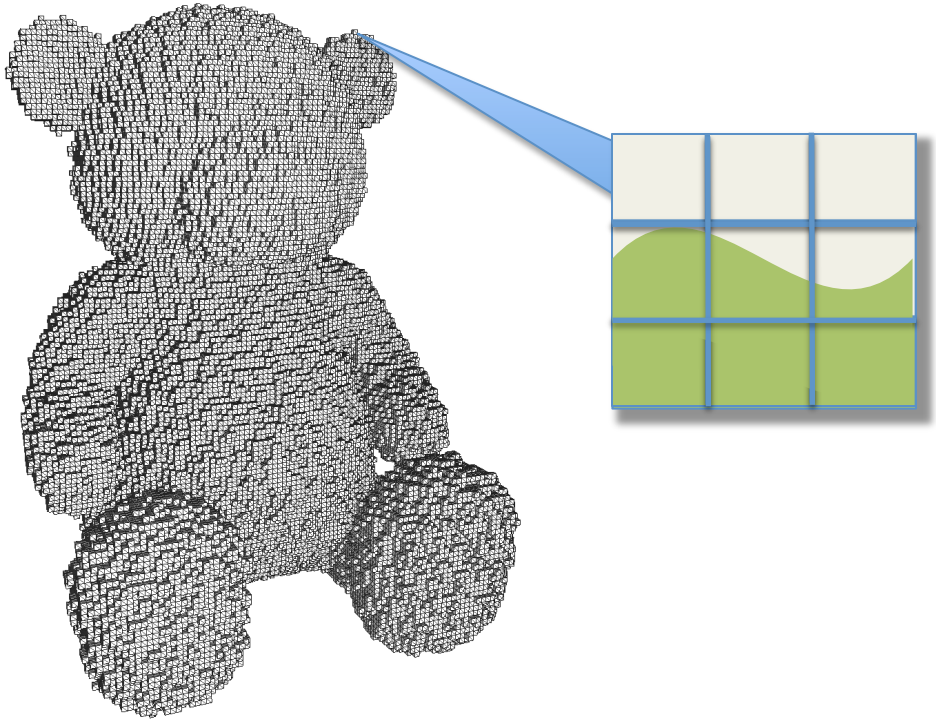

input
tessellate
surface stage
voxelize
volume stage
dither
output



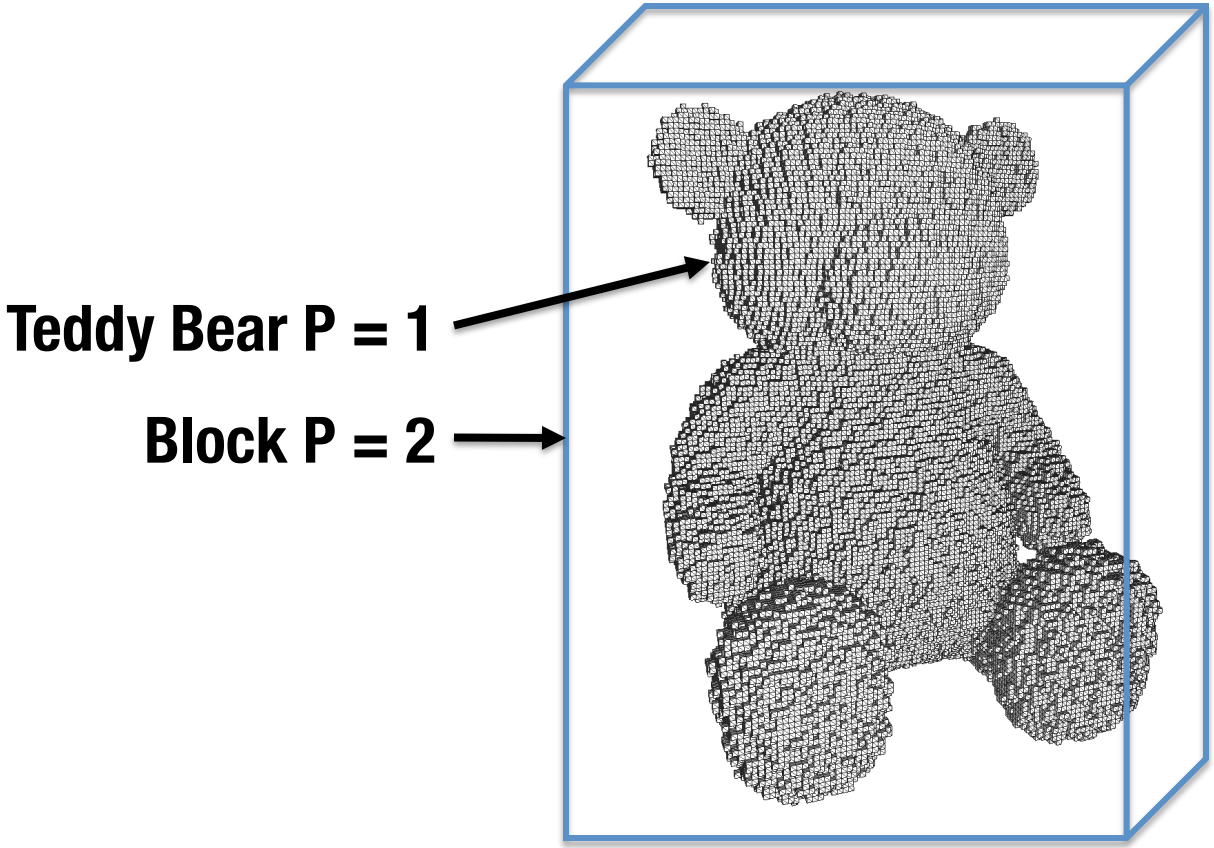

input
tessellate
surface stage
voxelize
volume stage
dither
output



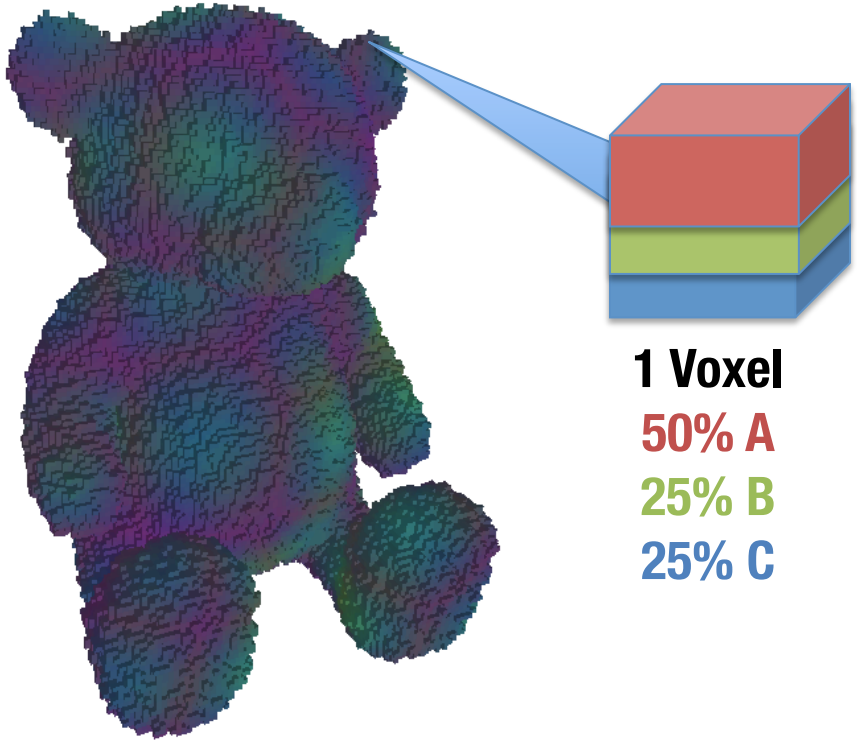

input
tessellate
surface stage
voxelize
volume stage
dither
output



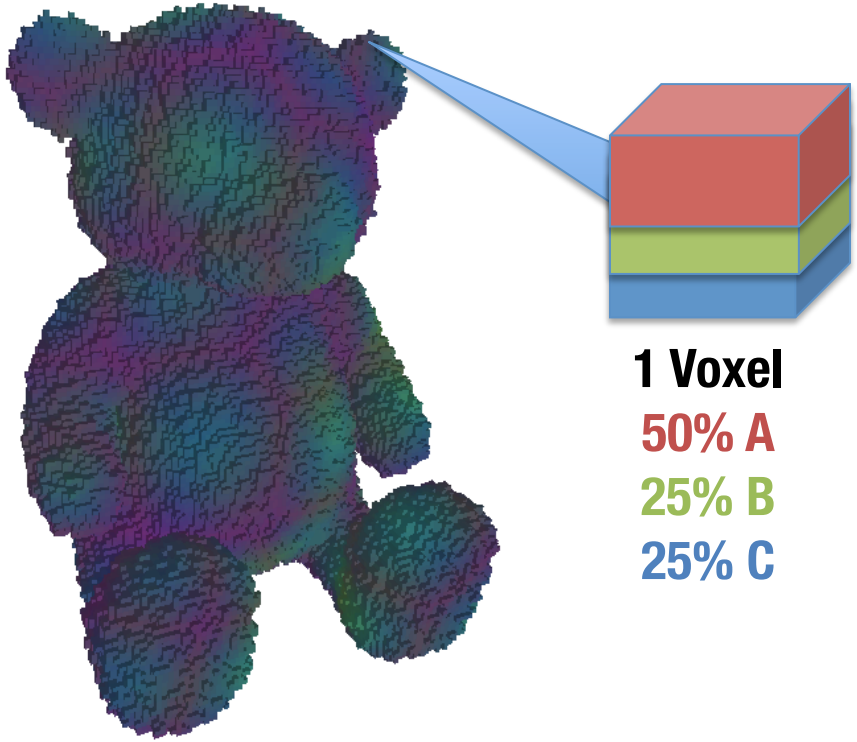

input
tessellate
surface stage
voxelize
volume stage
dither
output


Teddy Bear P = 1
Block P = 2

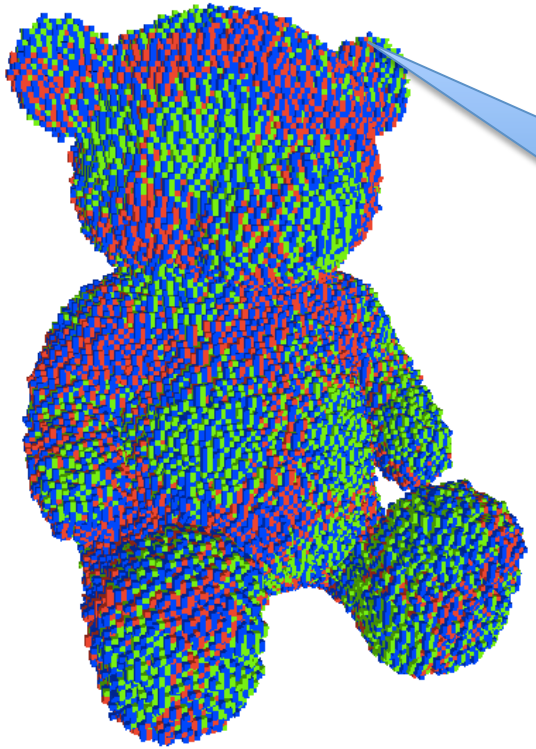

input
tessellate
surface stage
voxelize
volume stage
dither
output

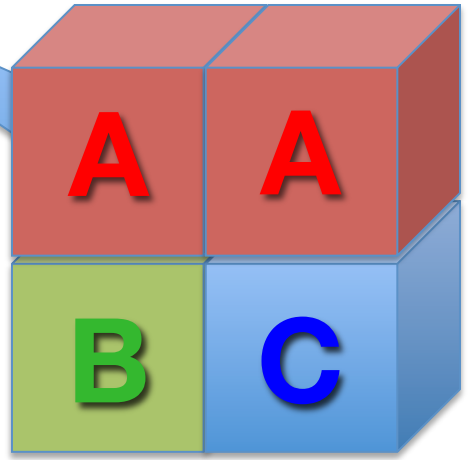

1 Voxel
50% A
25% B
25% C


input
tessellate
surface stage
voxelize
volume stage
dither
output


1 Voxel
50% A
25% B
25% C


input
tessellate
surface stage
voxelize
volume stage
dither
output

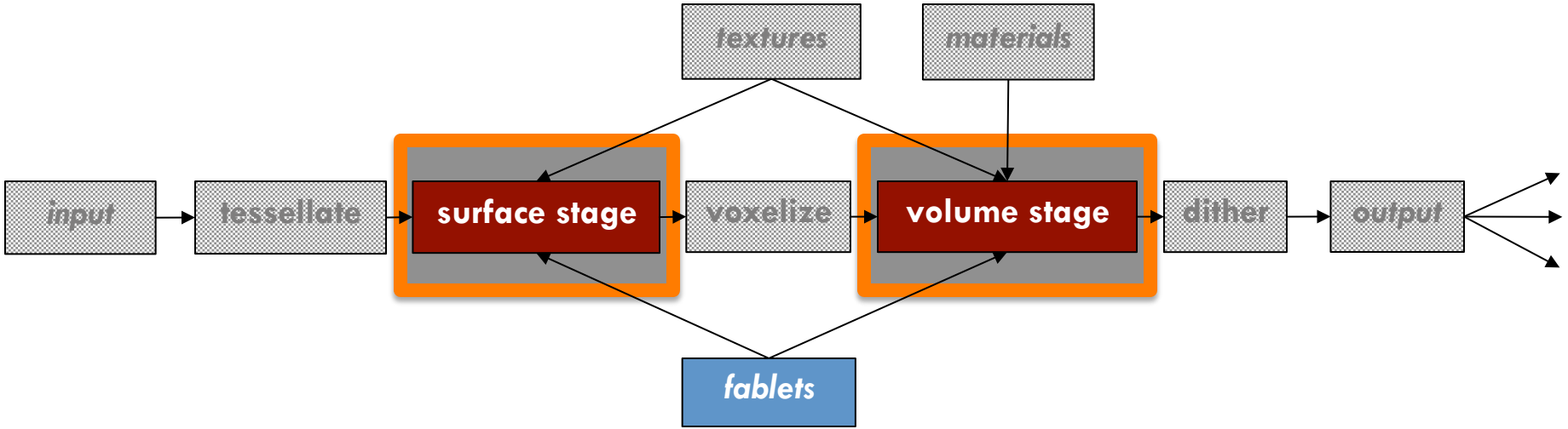



A
A
B
C

4 Voxels

OpenFL and Fablets

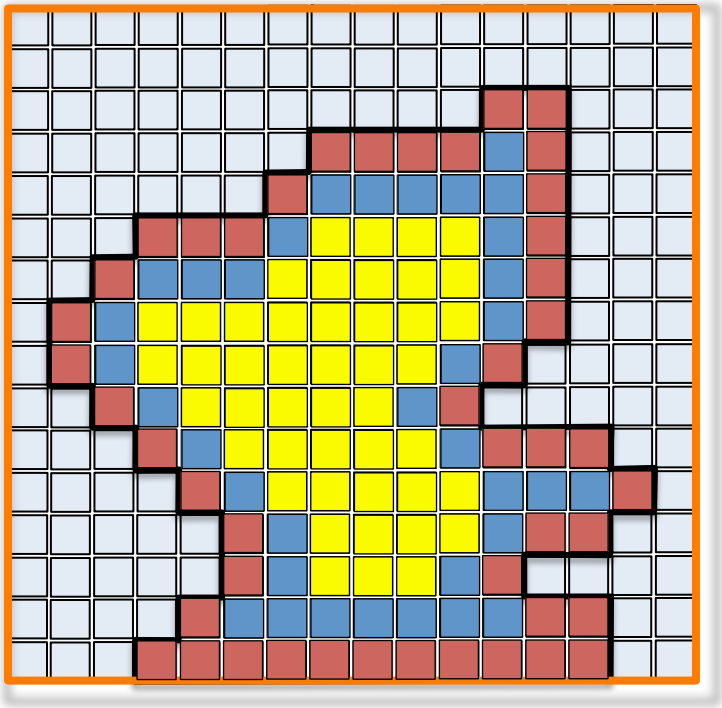
Programmable Stages



OpenFL: Domain-Specific Language

- C/C++ like language
- Built-in vector, matrix, texture, material types
- Modest OO features
- Pointwise (kernel) programming model
- Standard library of math functions
- Global queries

Volume Fablet: Global Queries



```
fablet MyFablet {
  @uniform Material red, blue, yellow;

  @Surface(…) {
      return double3(0, 0, 0); // no displacement
  }

  @Volume(@varying double3 voxelCenter) {
      MaterialComposition mc;
      const double layerThickness = 1;
      double dist = distance();
      if (dist <= layerThickness) {
          mc.Set(red, 1);
      } else if (dist <= layerThickness * 2) {
          mc.Set(blue, 1);
      } else {
          mc.Set(yellow, 1);
      }
      return mc;
  }
}
```

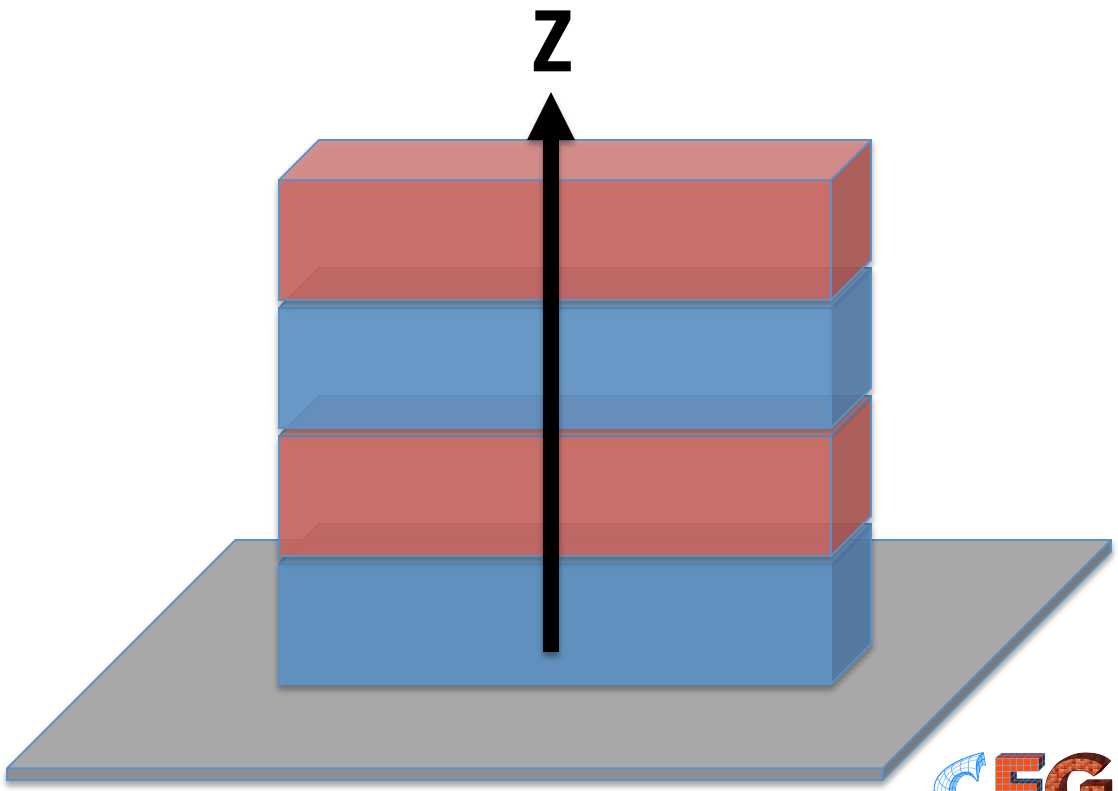
Why DSL?

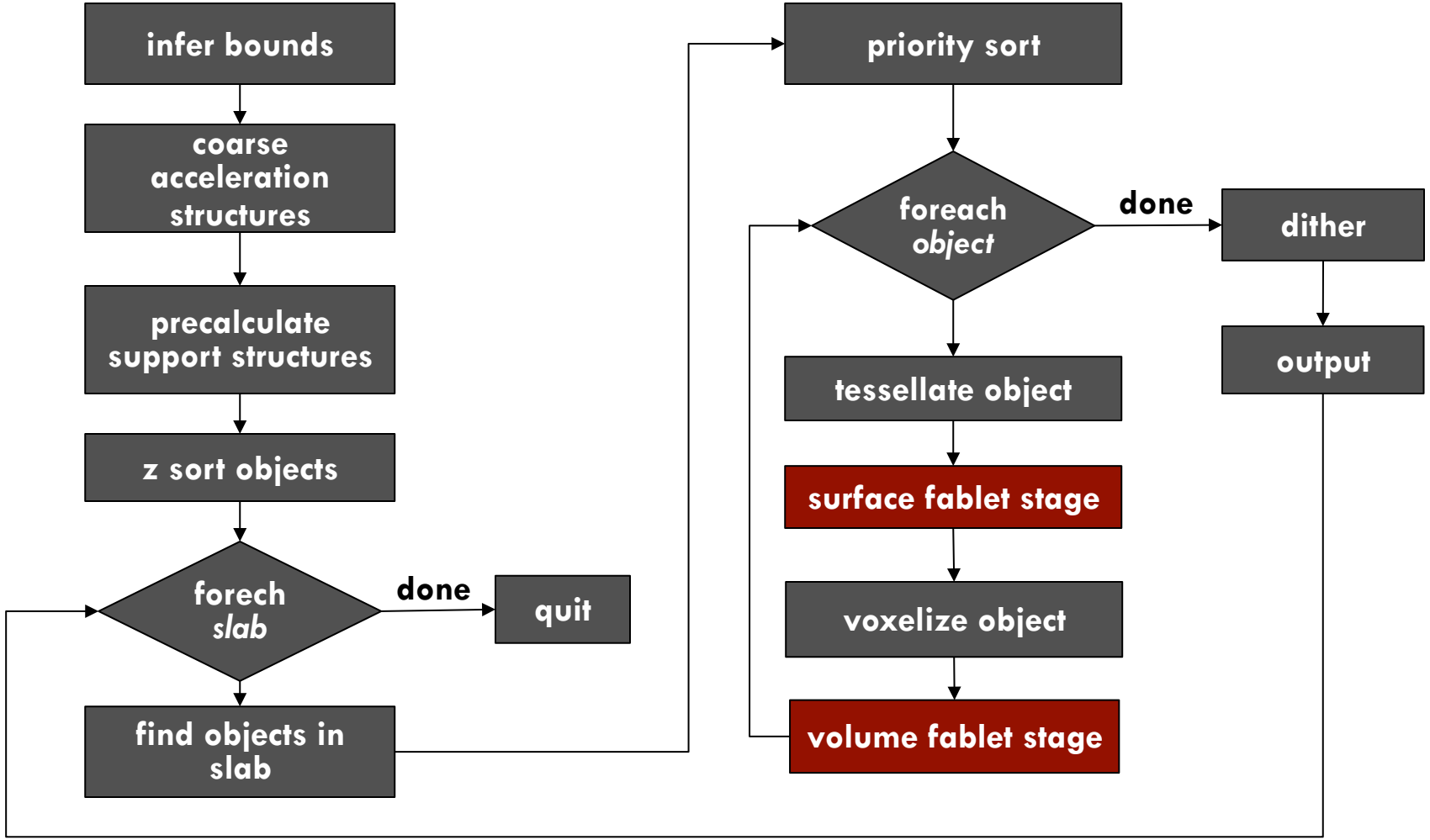
- Full control over programming model
- Analysis opportunities
- Optimizations
- Retargeting
- Sand-boxing

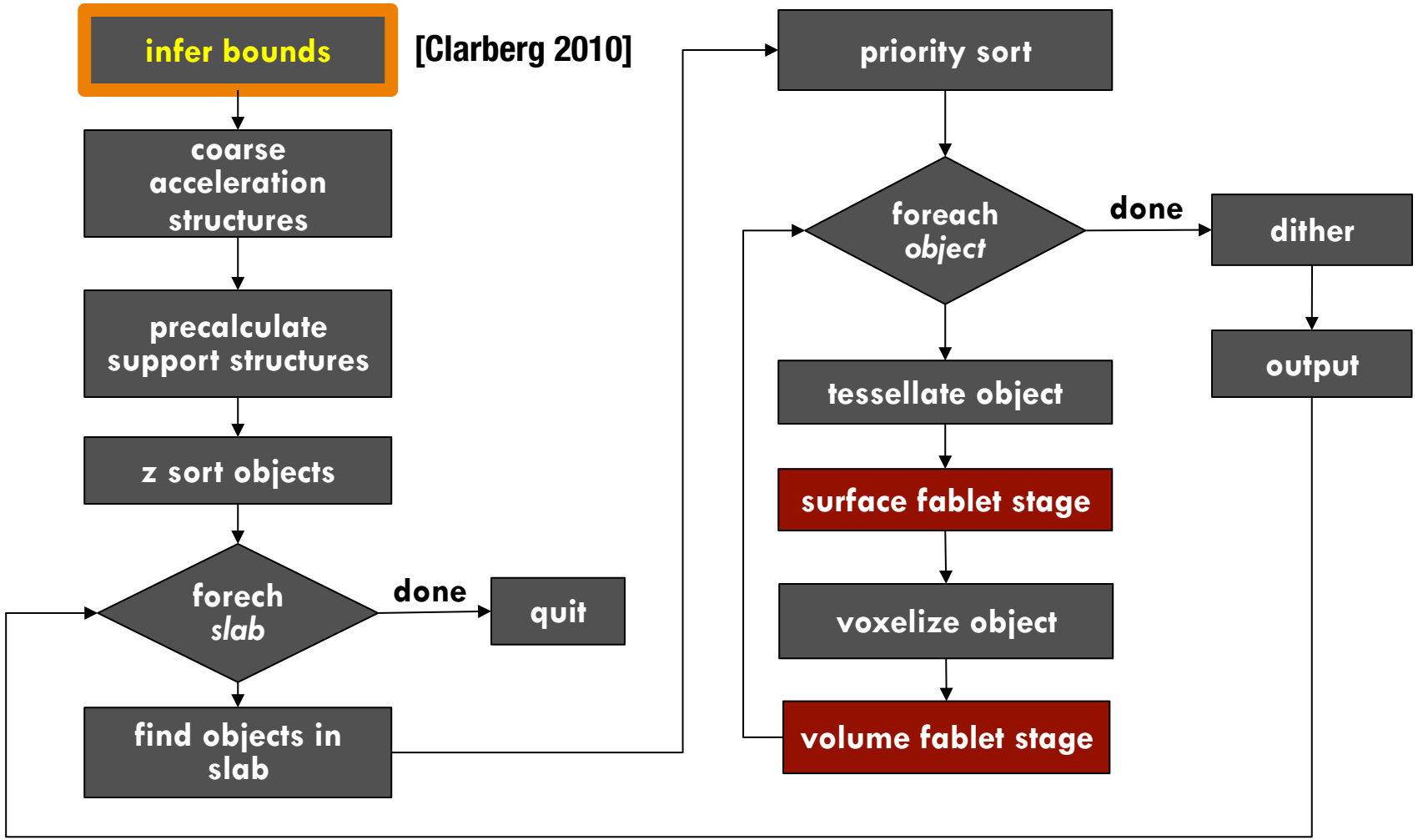
The OpenFab Architecture

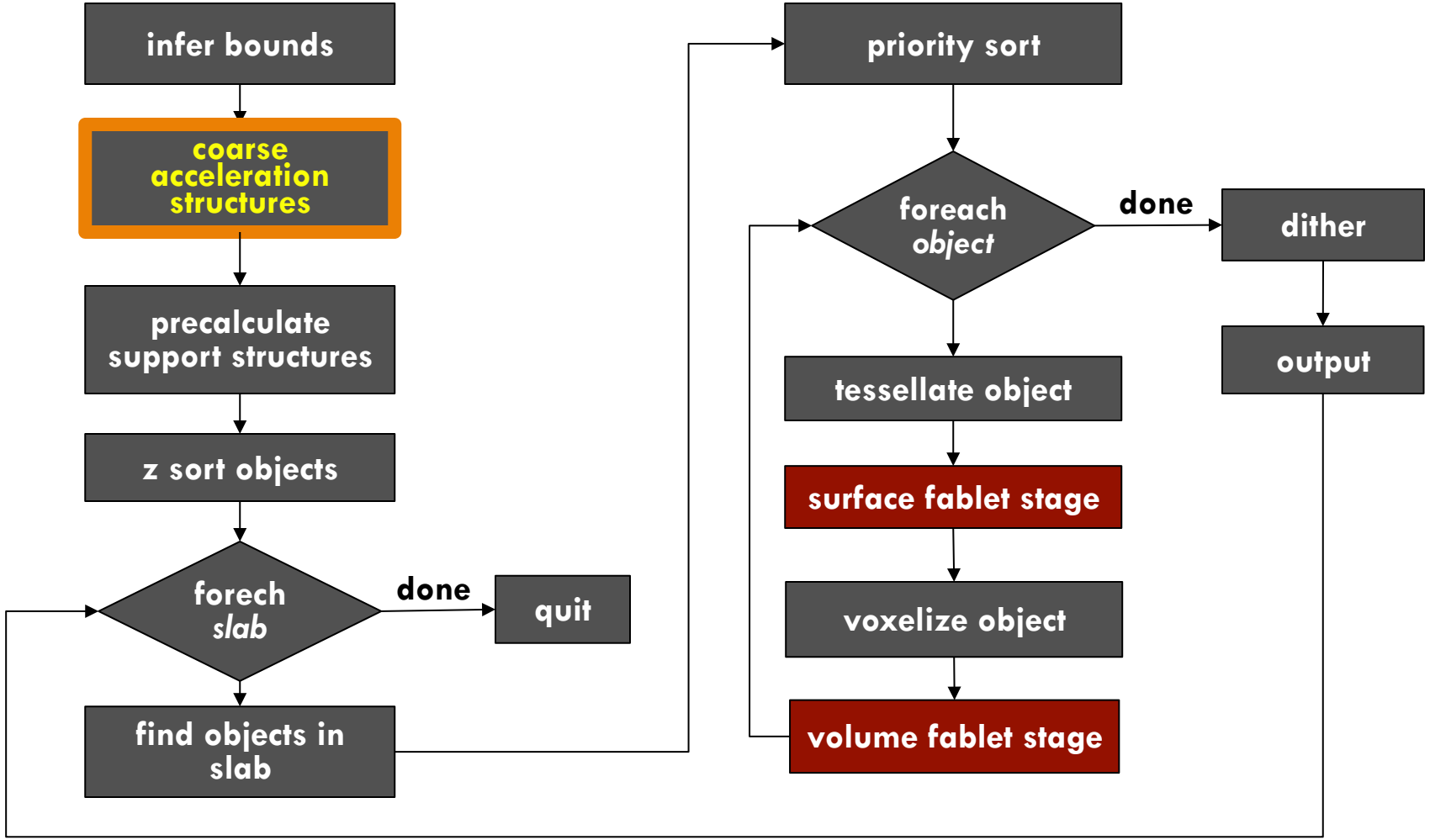
Scalable Architecture

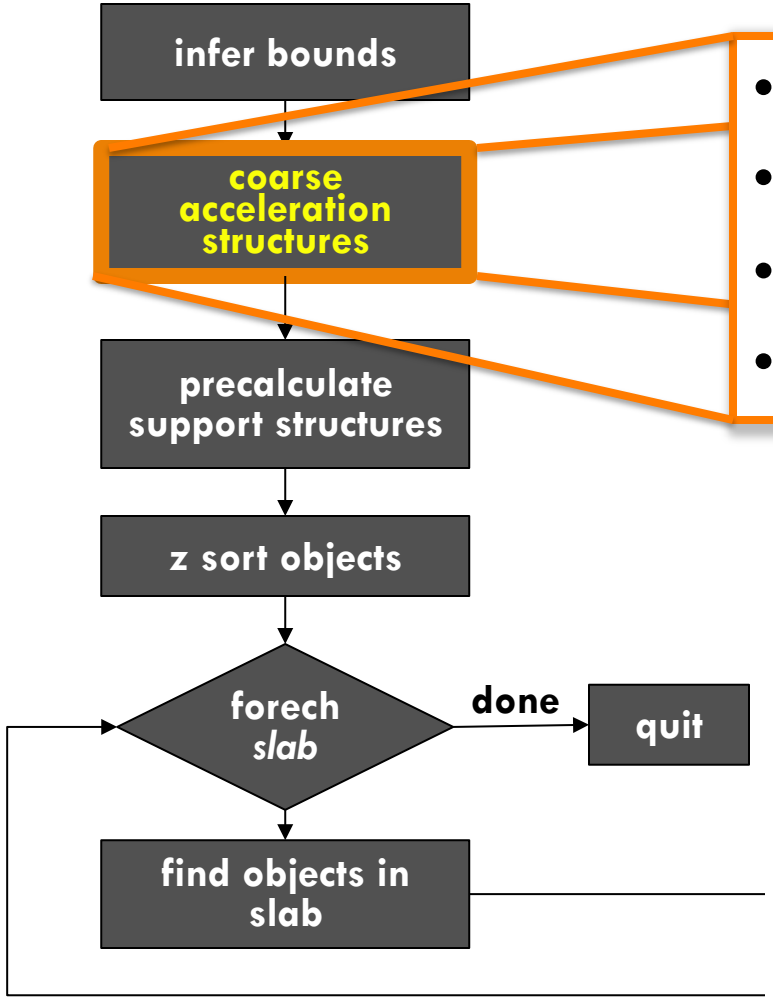
- Fast start-up
- Streaming
- Fixed memory



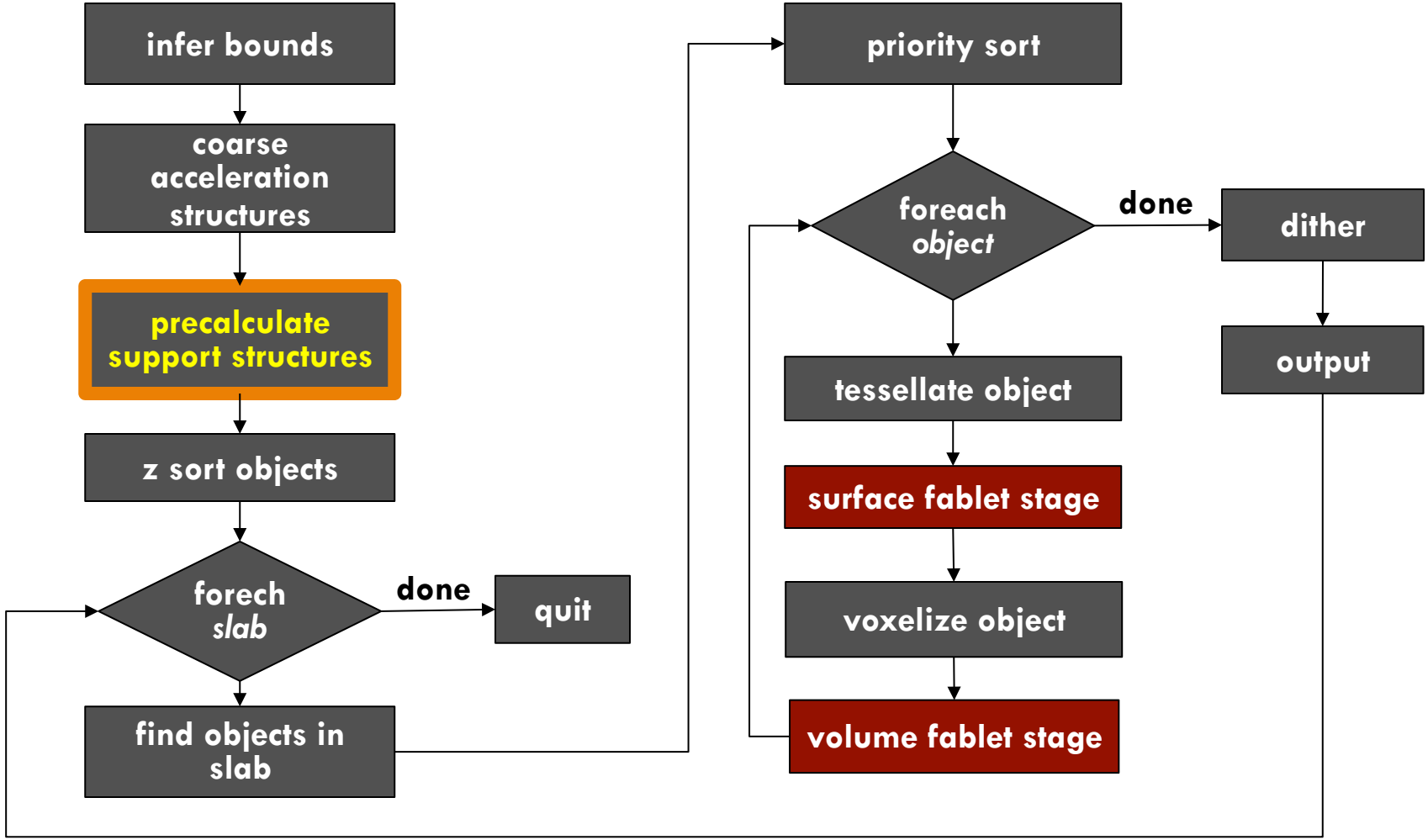

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

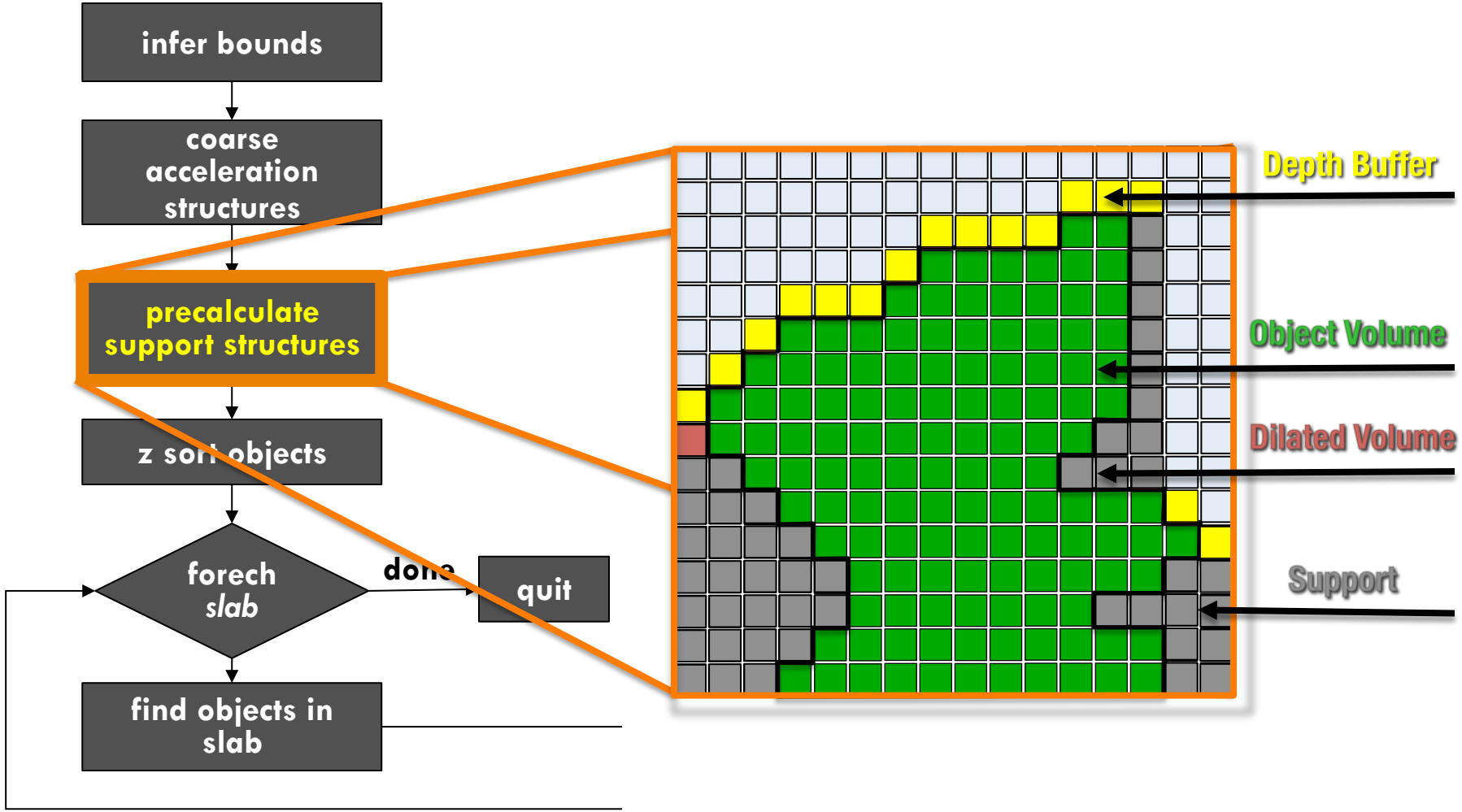

infer bounds
[Clarberg 2010]
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

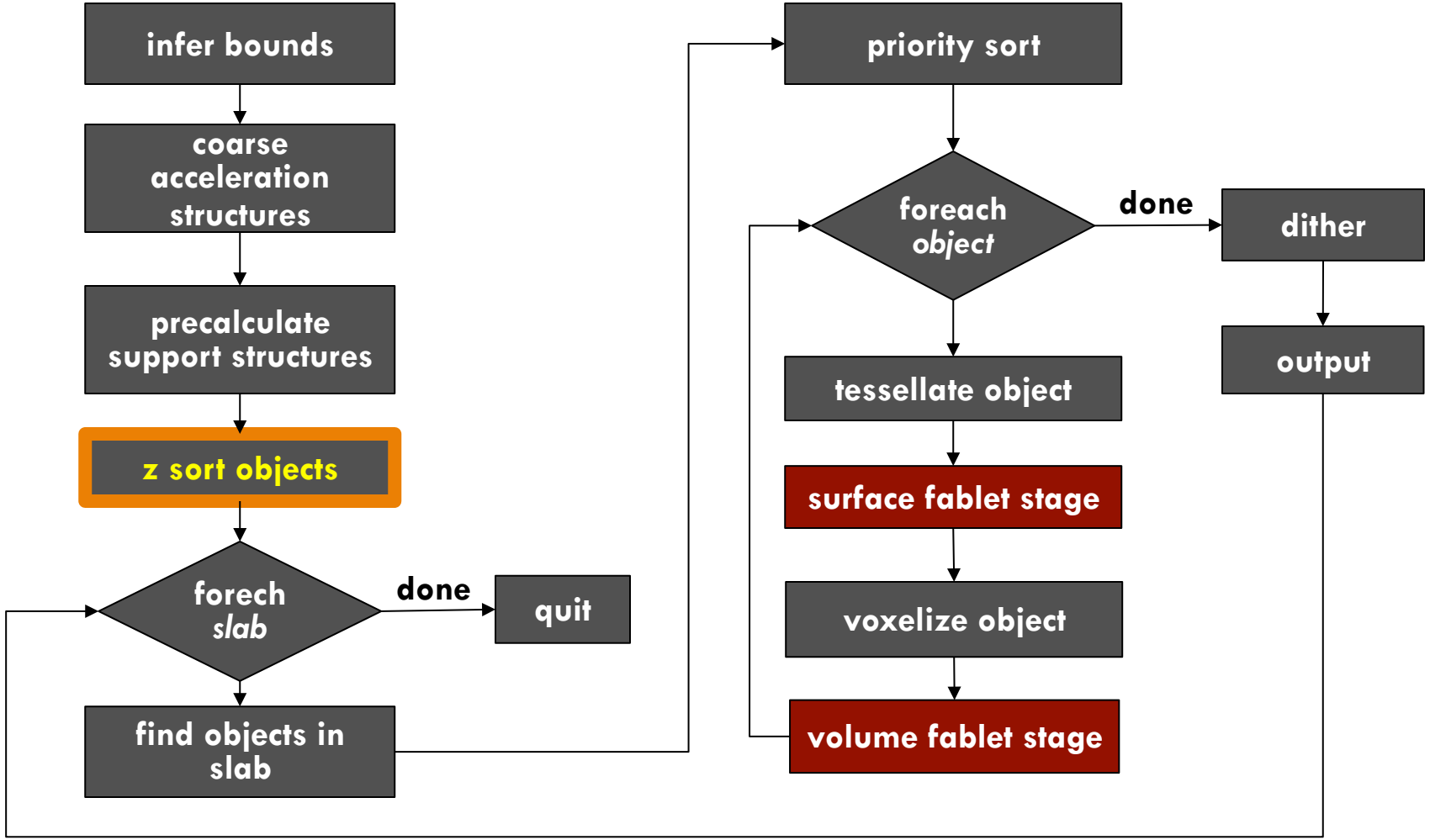

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

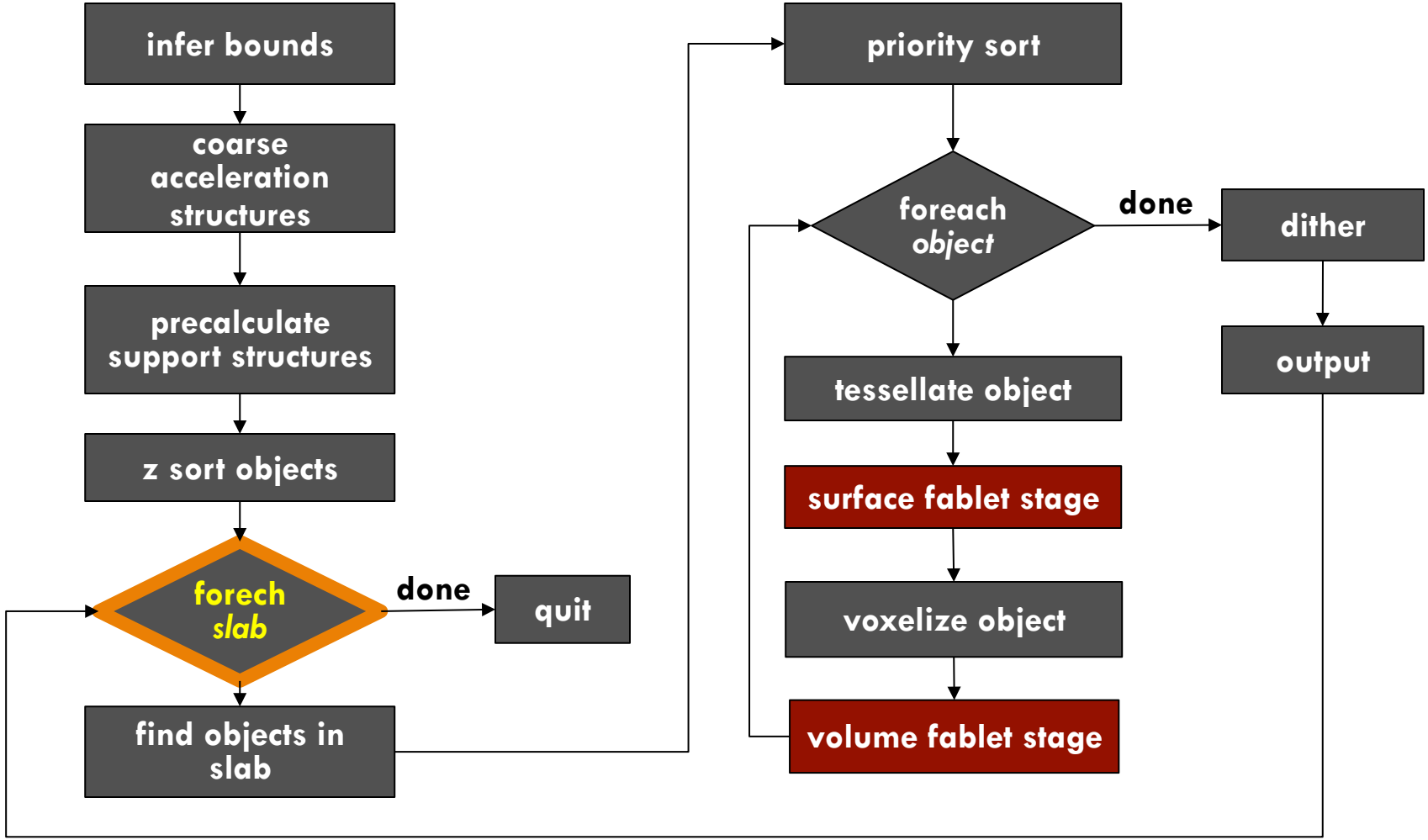


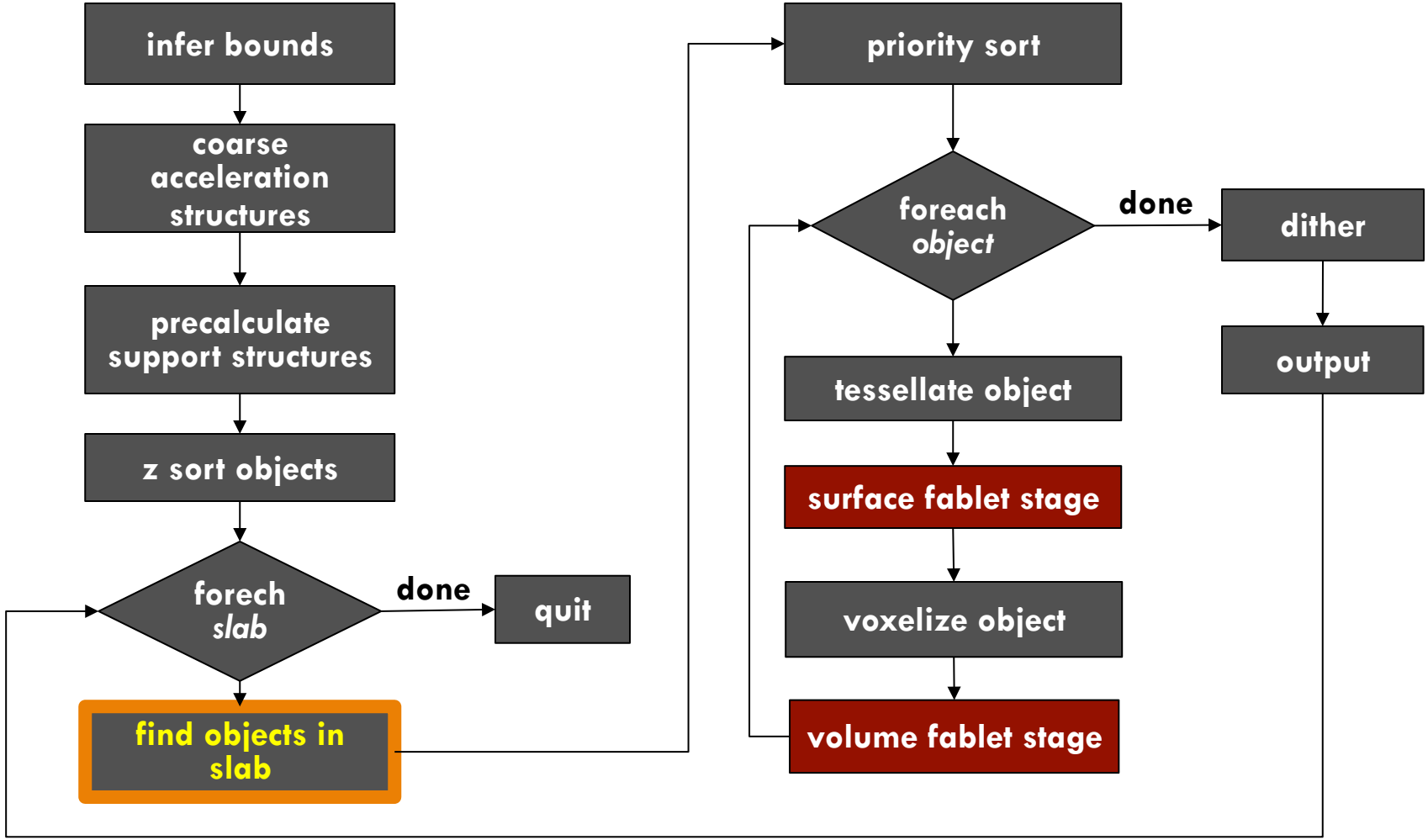
- Build coarse octree
- On query, build second-level octree
- Evaluate surface stage of fablet
- Cache result in the LRU cache

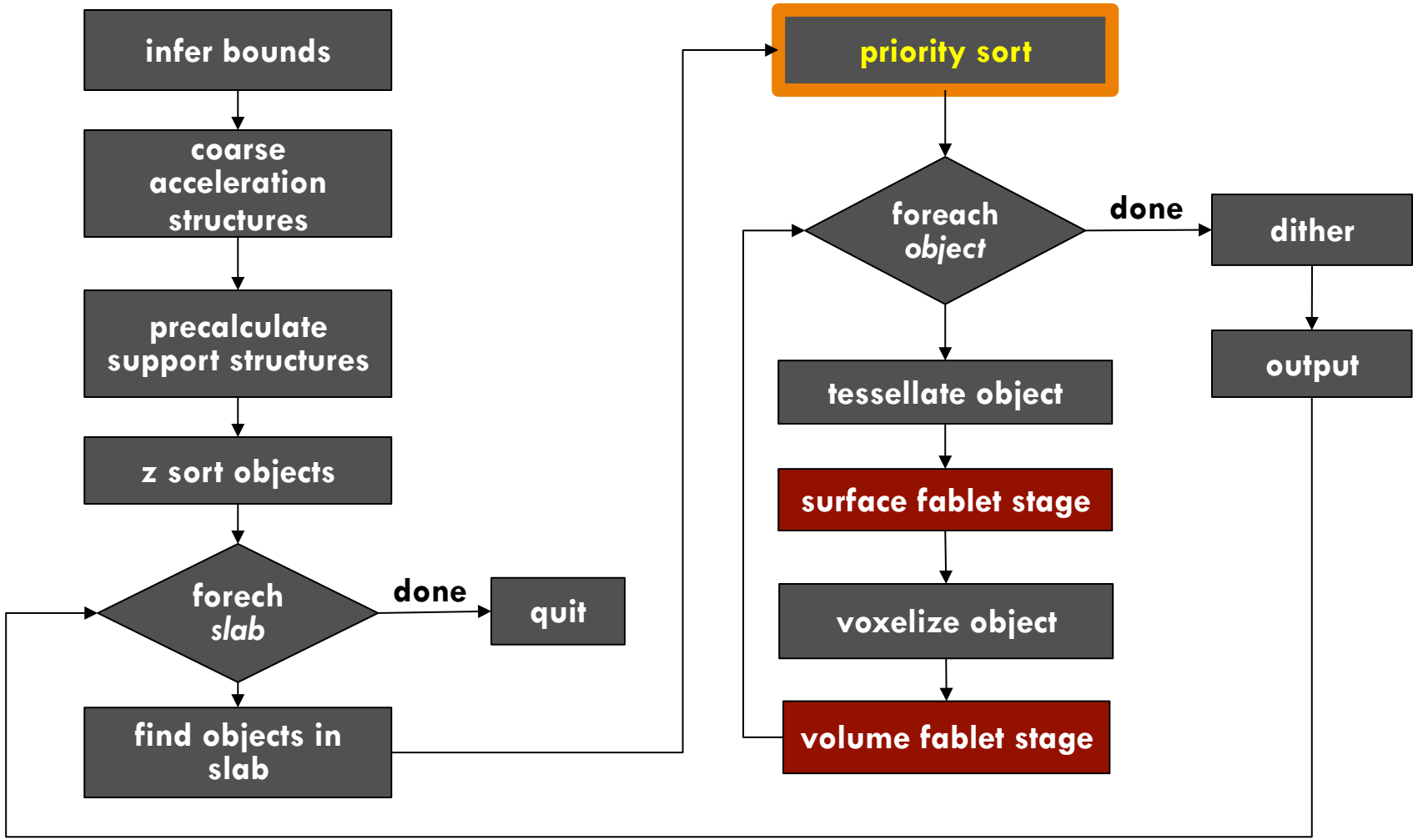

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

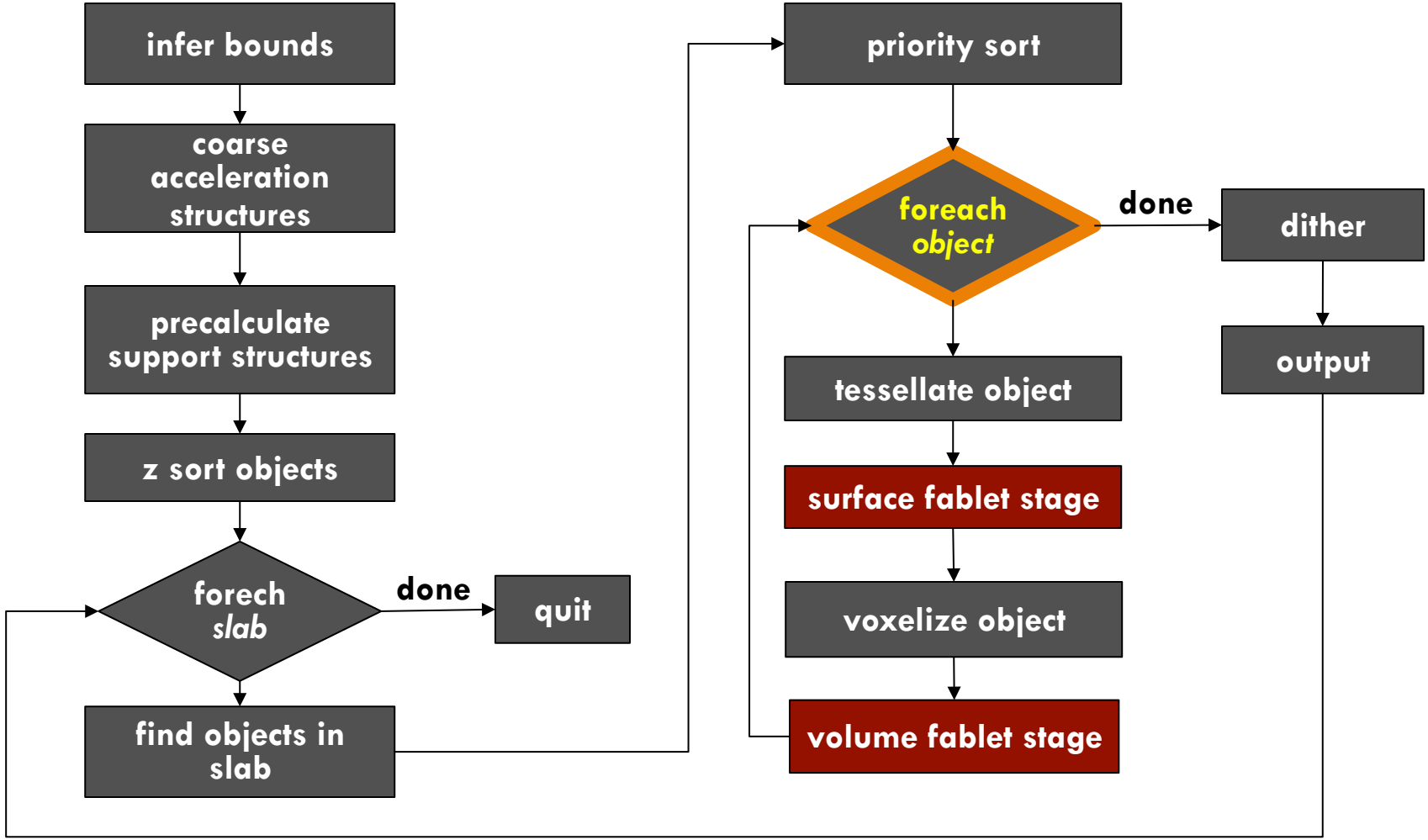

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
Depth Buffer
Object Volume
Dilated Volume
Support

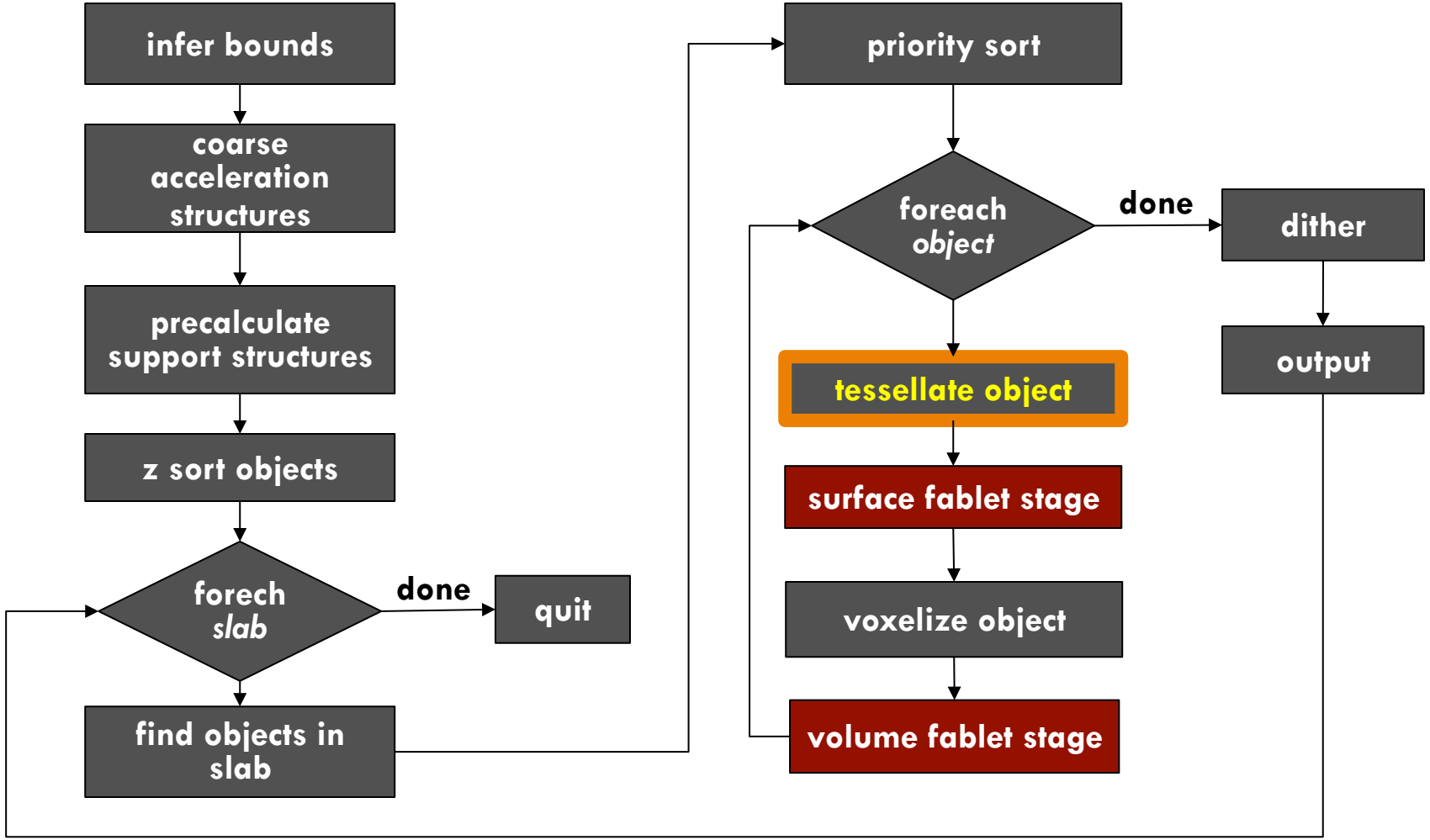

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

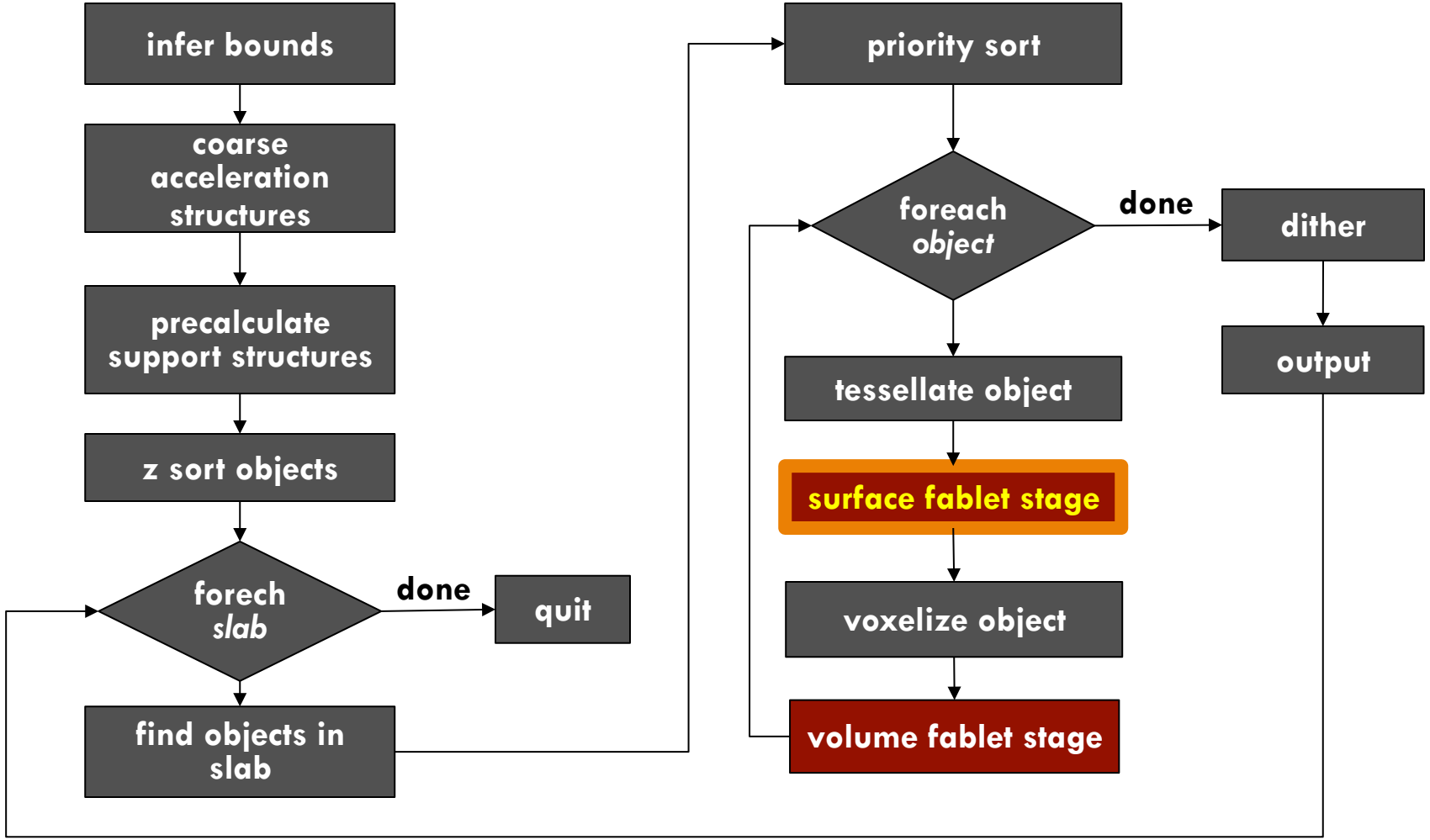

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

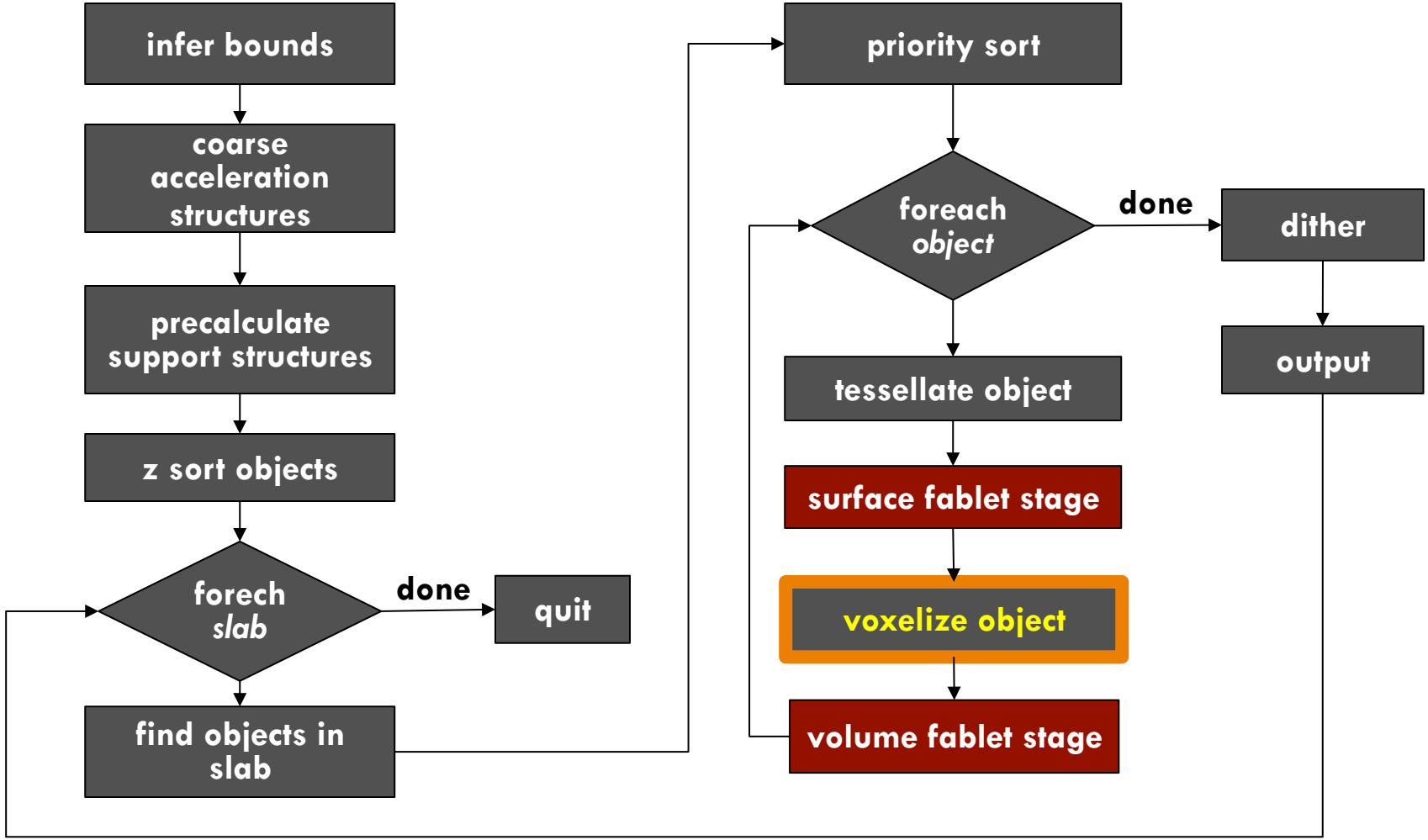

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

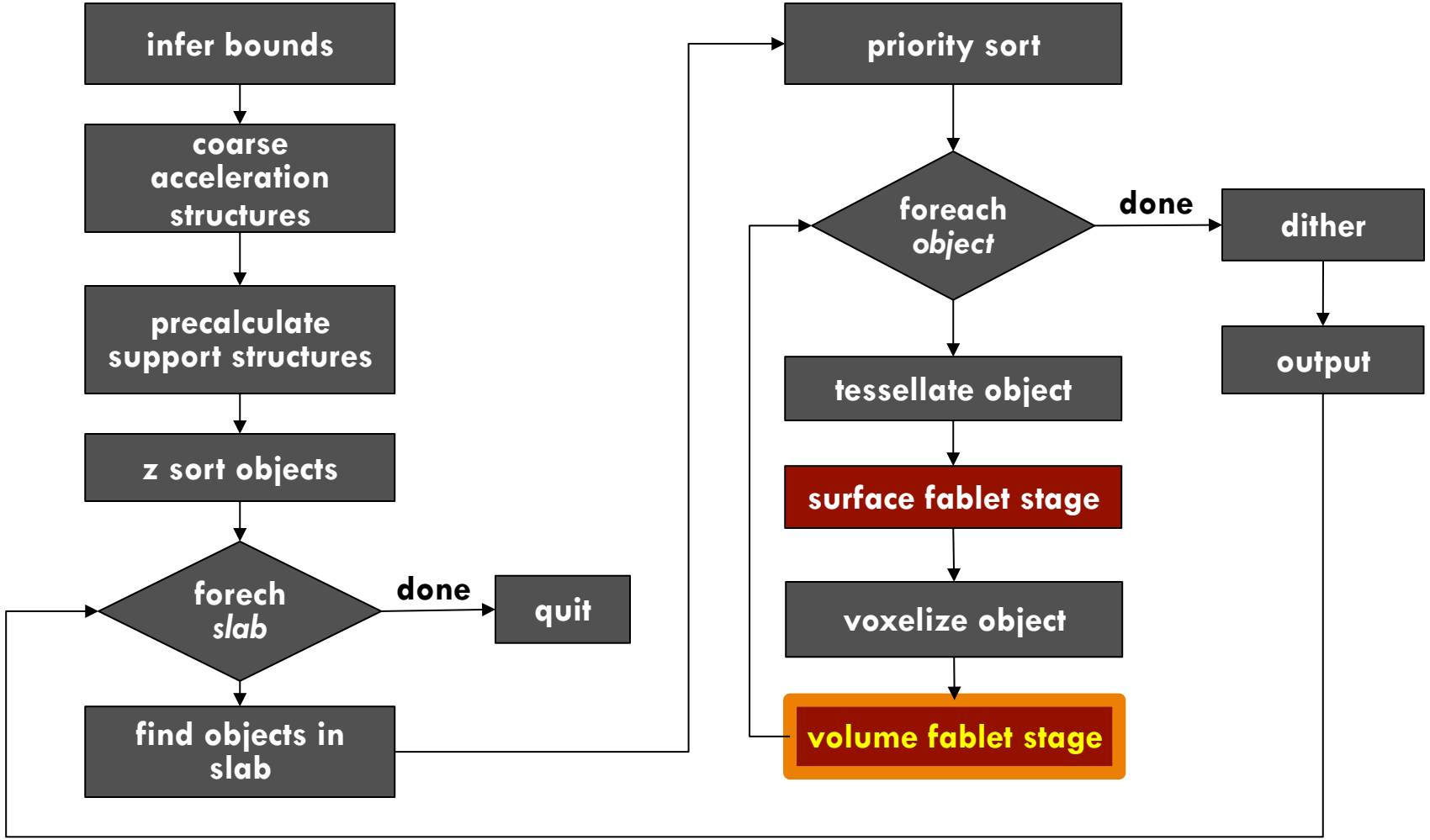

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

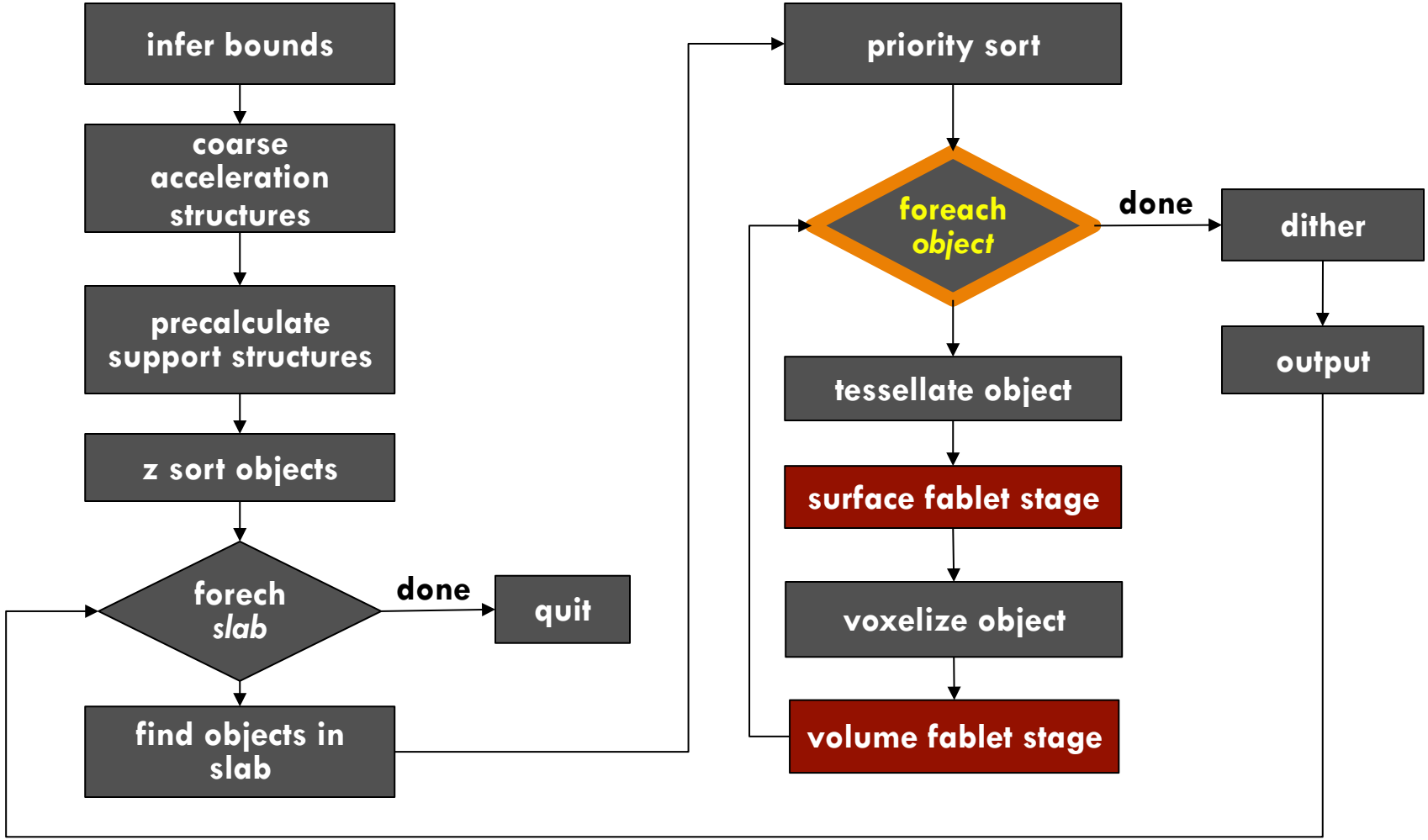

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

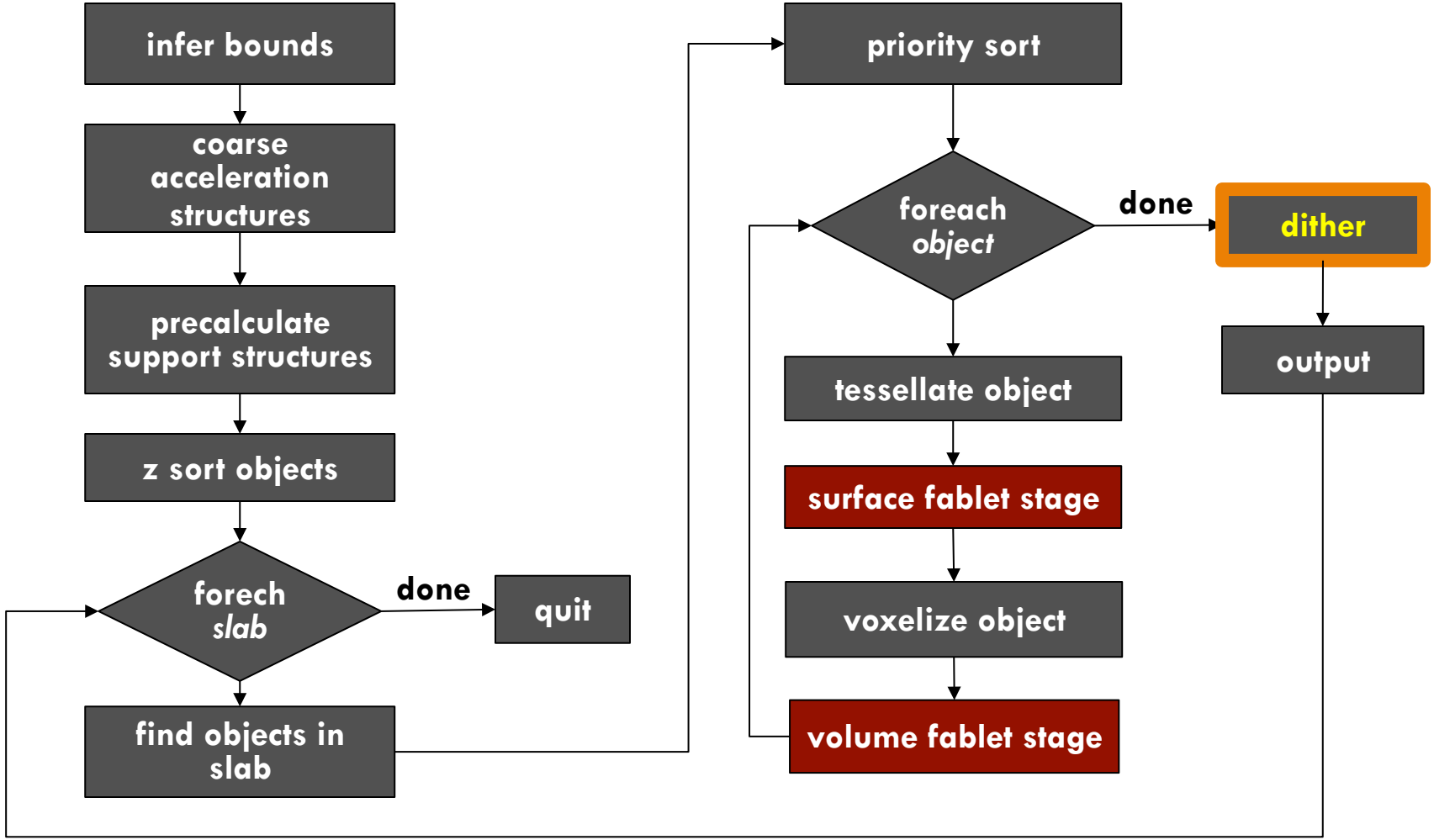

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

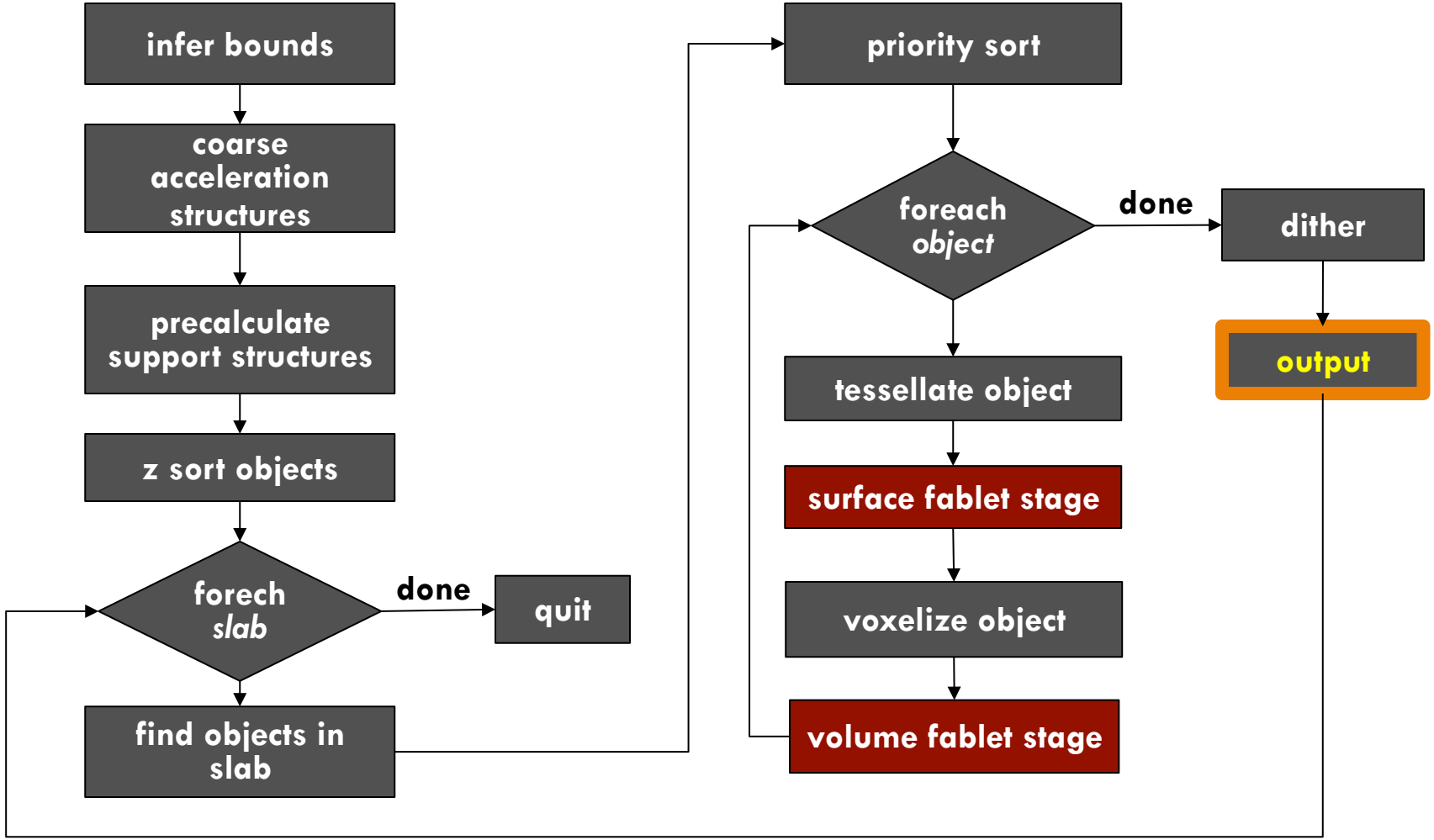

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

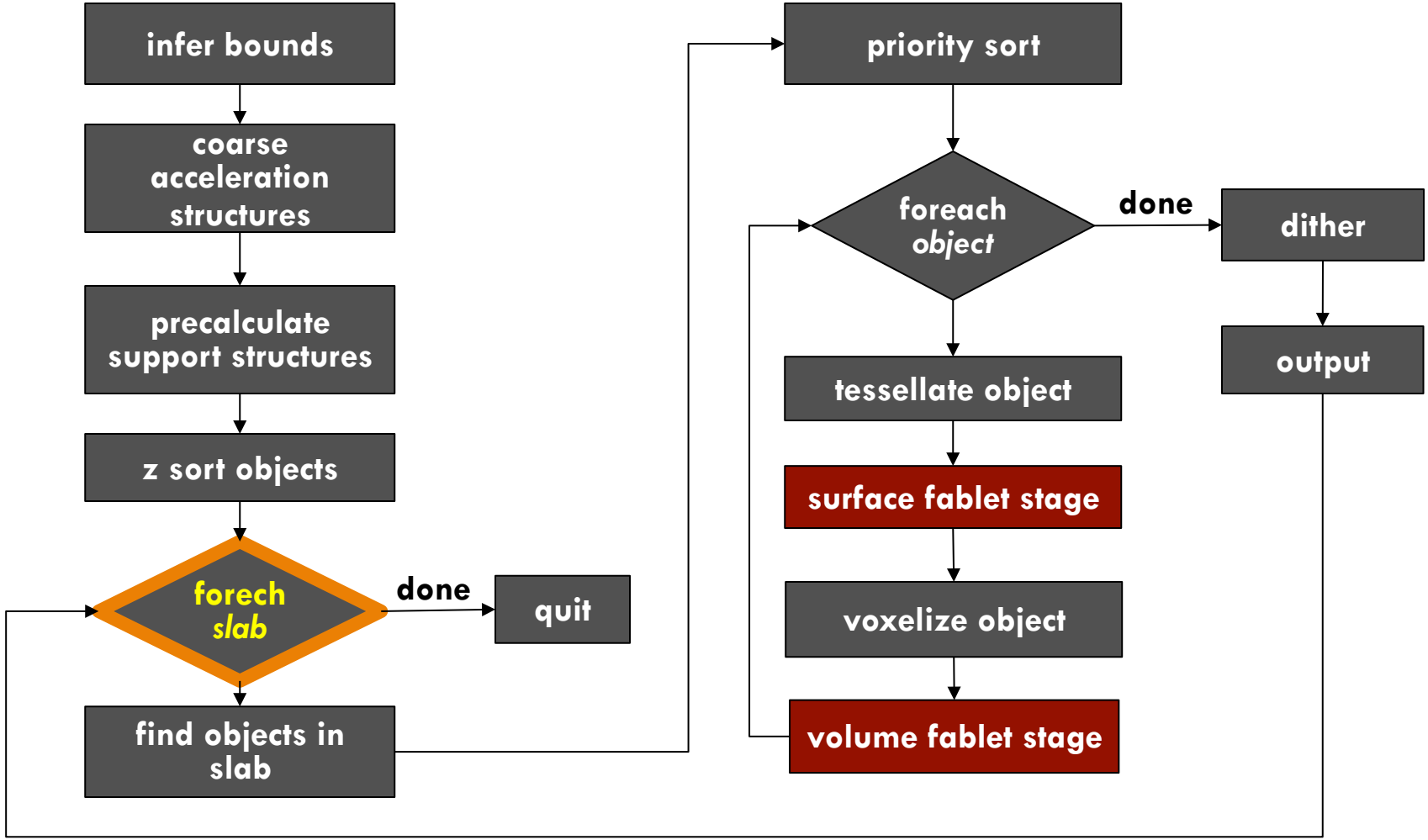

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

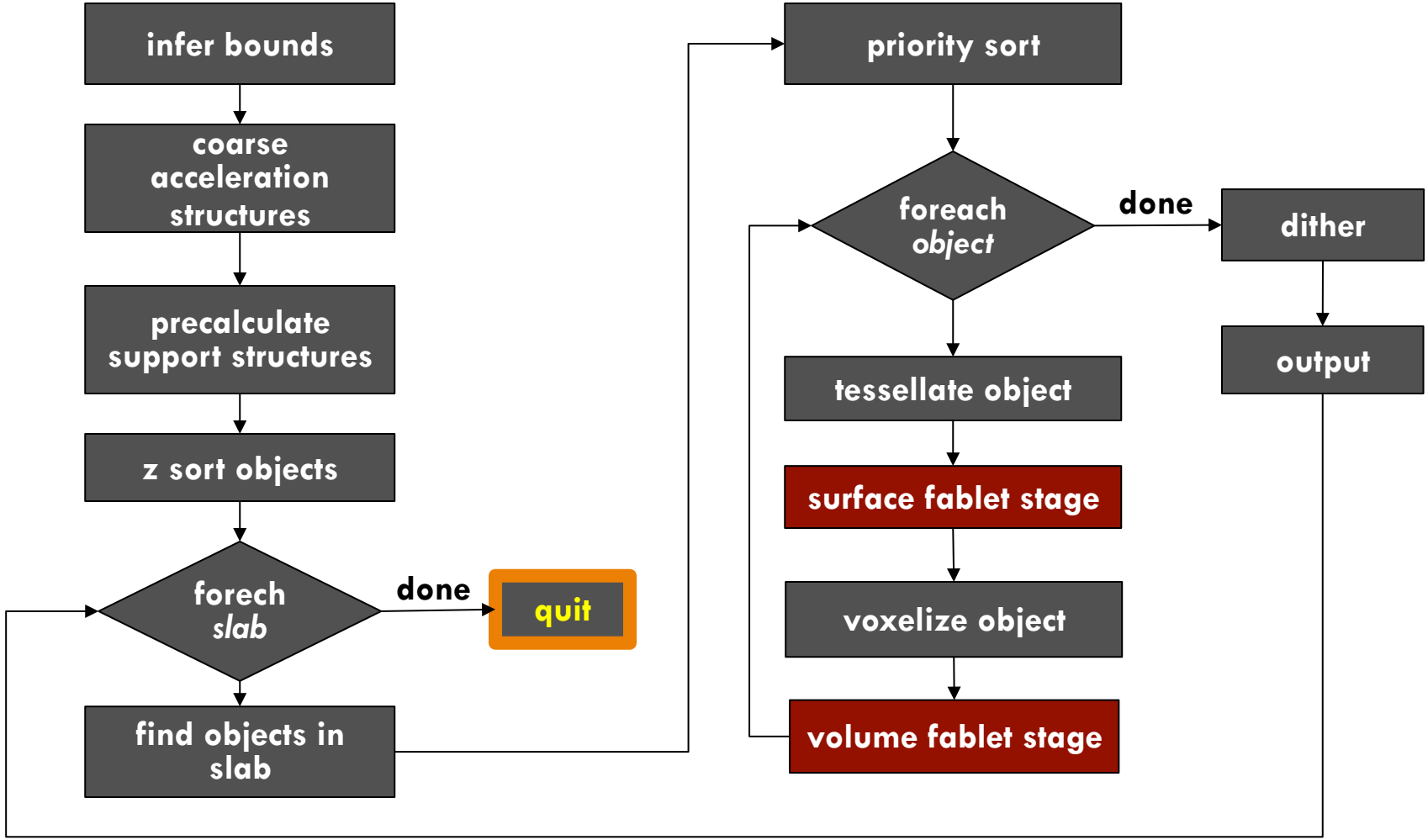

infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage


infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage


infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage


infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage


infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage


infer bounds
coarse acceleration structures
precalculate support structures
z sort objects
forech slab
done
quit
find objects in slab
priority sort
foreach object
done
dither
output
tessellate object
surface fablet stage
voxelize object
volume fablet stage

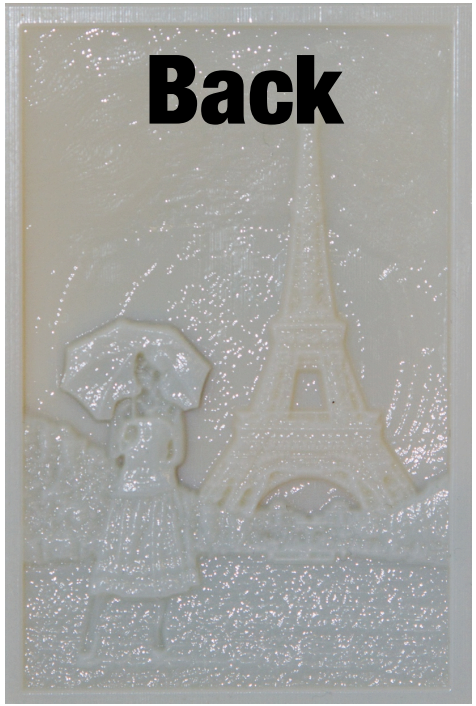
Results

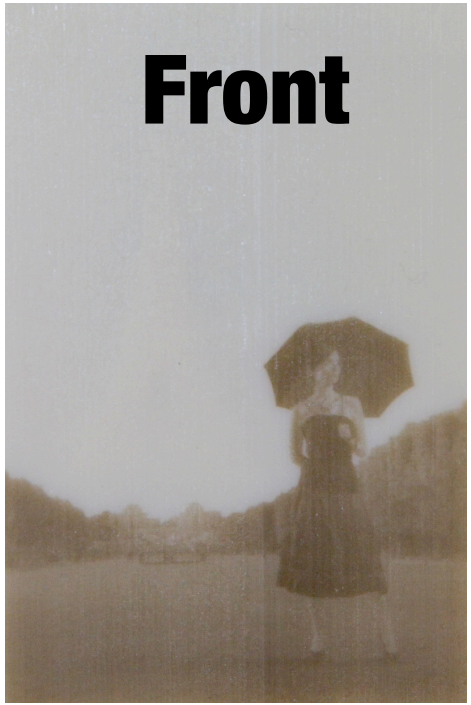
Material Decoupling

Material Decoupling

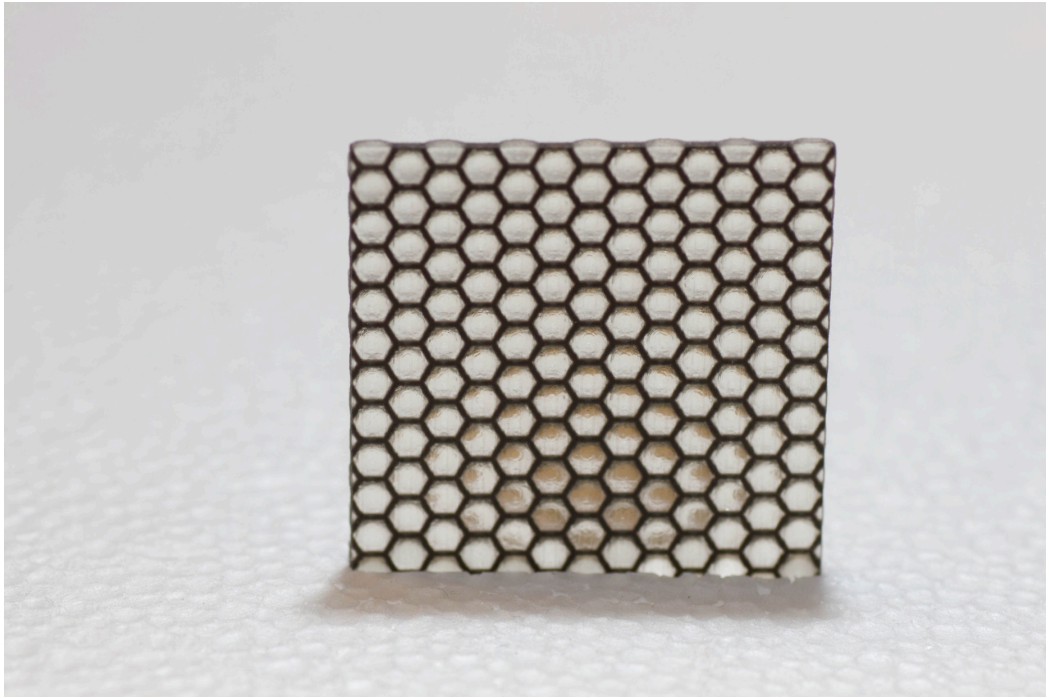
Material Decoupling

Lithopane





Procedural Surfaces



Procedural Volumes

Shape Priority

Conclusion

- First programmable pipeline for fabrication
- New programming model
- Domain-specific language
- Scalable architecture

Software Release

http://openfab.mit.edu/

Open sourcing the OpenFab API (BSD license)
Binary release of the fabricator and compiler

Thanks

- Mark Leone
- Jaakko Lehtinen
- Frédo Durand
- Pitchaya Sitthi-Amorn
- Ye Wang
- Desai Chen
- Moira Forberg
- Justin Lan
- David Levin
- James Tompkin
- Thiago Pereira
- Shinjiro Sueda

Funding: NSF, NSF GRP, DARPA, and MIT UROP

SIGGRAPH2013