习题

设 $p(\mathbf{x}|\Sigma) \sim N(\mu, \Sigma)$ 这里 μ 已知，Σ 未知。 如果 Σ 的最大似然估计为 $\widehat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}(x_k - \mu)(x_k - \mu)^t$，证明以下论述：

1) 证明矩阵等式$a^t Aa = tr[Aaa^t]$，这里矩阵的迹$tr[A]$表示$n \times n$维矩阵A的对角线元素之和，a为一个向量。

3）设 $A=\Sigma^{-1}\hat{\Sigma}$ ，$\lambda_1,\ldots,\lambda_d$ 为A的特征值，证明前面第(2)小题中的概率式

$$p(\mathbf{x_1},\ldots,\mathbf{x_n}\mid\Sigma)=\frac{1}{(2\pi)^{nd/2}}\mid\Sigma^{-1}\mid^{n/2}\ exp\left[-\frac{1}{2}tr\left[\Sigma^{-1}\sum_{k=1}^{n}(\mathbf{x_k}-\mu)(\mathbf{x_k}-\mu)^t\right]\right]$$

可写为

$$p(\mathbf{x_1},\ldots,\mathbf{x_n}\mid\Sigma)=\frac{1}{(2\pi)^{nd/2}\mid\hat{\Sigma}\mid^{n/2}}(\lambda_1\ldots\lambda_d)^{n/2}exp\left[-\frac{n}{2}(\lambda_1+\ldots+\lambda_d)\right]$$

提示：

$$\mid A\mid=\mid\Sigma^{-1}\mid\mid\hat{\Sigma}\mid$$

$$tr(A)=\sum(A_{ii})=\sum(\lambda_i)$$

$$\mid A\mid=\prod\lambda_i$$

4）证明概率 $p(\mathbf{x_1},...,\mathbf{x_n}\,|\,\Sigma)=\dfrac{1}{(2\pi)^{nd/2}\,|\hat{\Sigma}|^{n/2}}(\lambda_1...\lambda_d)^{n/2}exp\left[-\dfrac{n}{2}(\lambda_1+...+\lambda_d)\right]$

在 $\lambda_1=1...\lambda_d=1$ 时达到最大值。

数据建模与分析
数字几何与制造

姚 远
yaoyuan@shu.edu.cn

快速制造工程中心

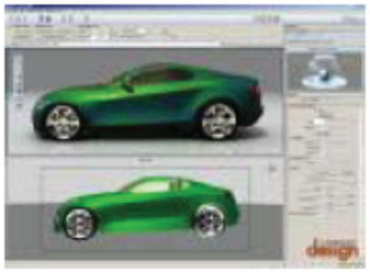
2020/10/16

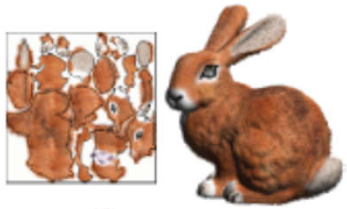
课程内容

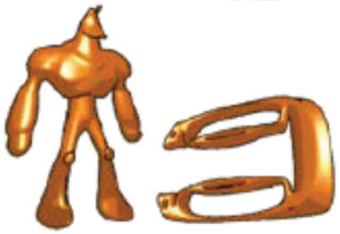
- 数字几何的概述
- 数字几何基本问题
- 几何模型表示方法
- 表面网格的处理方法
- 3D打印中的应用

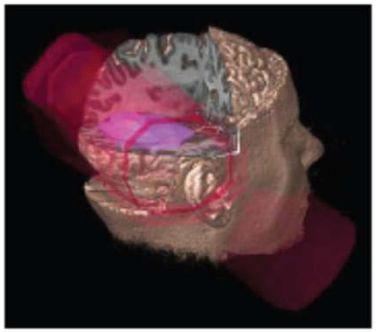
数字几何概述

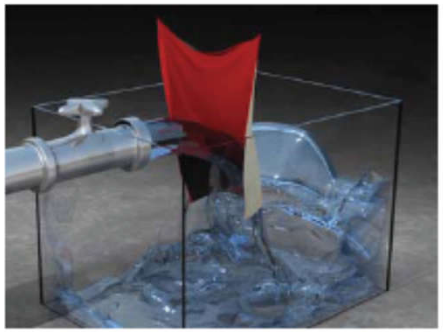
几何数据的表达无处不在











数字几何概述

- (Geometry)= “Geo”+“metron”，

  大地　　测量

- 数字几何模型是对各种物理实体的数字化表达，即数字化物理实体。
- 数字几何模型是构筑虚拟世界、仿真模型的基本要素。

数字几何概述

- 1963，Boeing公司，Ferguson，曲线曲面的参数表示
- 1964-1967，MIT，Coons，构造插值给定边界及导矢的曲面，Coons曲面

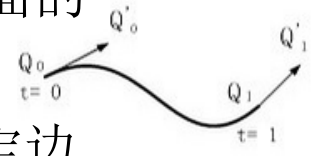


图 Ferguson曲线

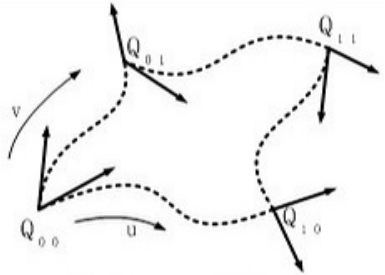


图 Ferguson曲面

-Steven Anson Coons 奖是计算机图形学领域的最高奖

- 1971，Rénault公司，Bézier，由控制多边形定义的曲线；Citroën，de Casteliau，独立发展了类似方法。

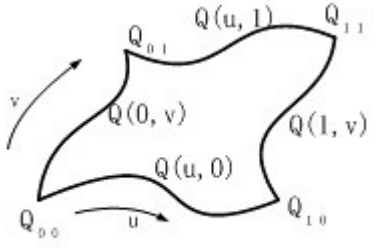
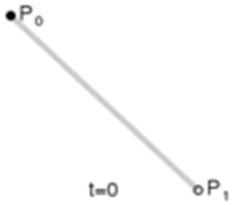


图 Coons曲面

数字几何概述

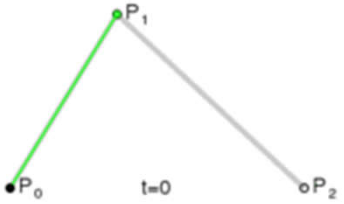
- Bezier曲线实例
  - 一次

$$\mathbf{B}(t) = \mathbf{P}_0 + (\mathbf{P}_1 - \mathbf{P}_0)t = (1-t)\mathbf{P}_0 + t\mathbf{P}_1,\ t \in [0,1]$$



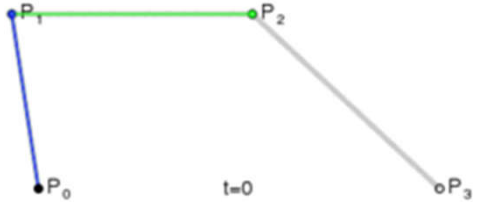
  - 二次

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2t(1-t)\mathbf{P}_1 + t^2\mathbf{P}_2,\ t \in [0,1]$$



  - 三次

$$\mathbf{B}(t) = \mathbf{P}_0(1-t)^3 + 3\mathbf{P}_1 t(1-t)^2 + 3\mathbf{P}_2 t^2(1-t) + \mathbf{P}_3 t^3,\ t \in [0,1]$$



数字几何概述

- 1973剑桥大学的Ian Braid完成了实体表达的博士论文
  - 与导师Charles共同创建了Shape Data公司推出Romulus引擎
  - Parasolid
  - ACIS
- 1974, Barnhill and Riesenfeld。
  - 第一届CAD会议
  - 在计算机环境下的曲线曲面的表示与逼近
  - CAD中的几何问题与数学描述
- 1975-1990, Syracuse大学的Versprill, Piegl, Tiller
  - 有理B样条方法
  - 非均匀有理B样条方法（即NURBS）

- UG/SolidEdge/Solidworks/Anysis Comos/EMAP
- AutoCAD/Abaqus/Flunt/Nastran

数字几何概述

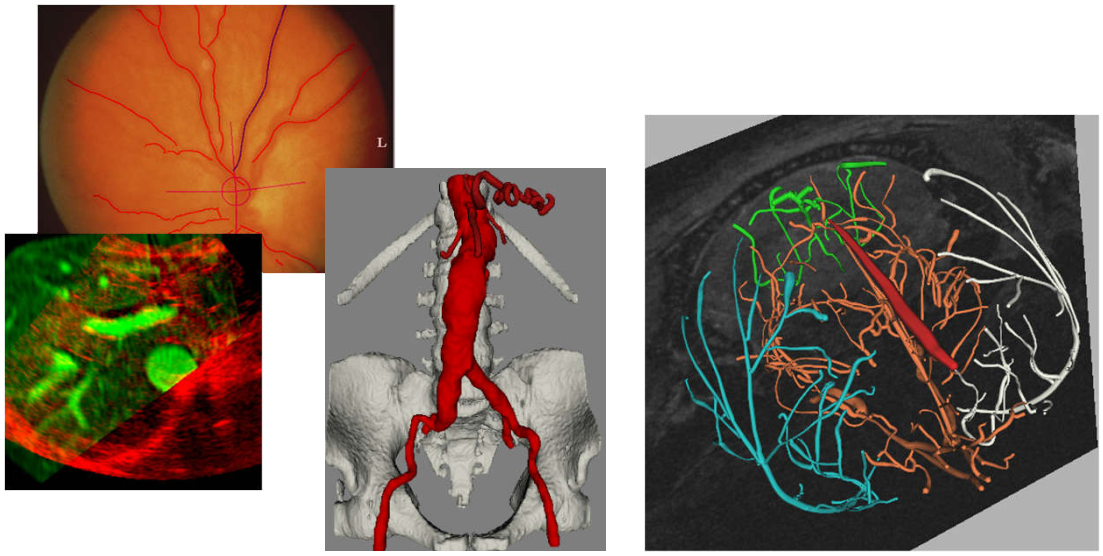
- 1990, Dyn，butterfly插值细分**曲面**
- 1996，Kobbelt，四边形**网格**插值细分格式
- 1990~，Hoppe等，曲面重建，实体造型，逆向工程…

数字几何概述

- 来自娱乐业的驱动

数字几何概述

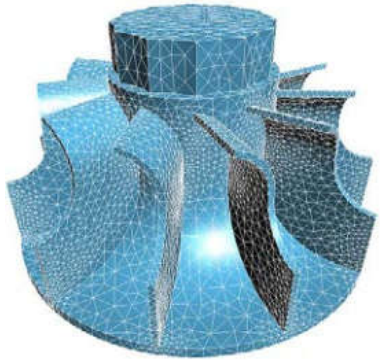
- 来自消费和医疗领域的驱动

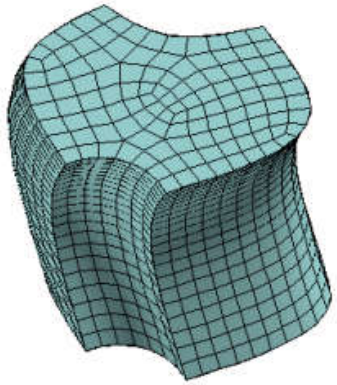


- *3D-Doctor*
- *Amira*
- *Mimics*
- *ScanIP, ScanFE, ScanRP*
- *GStudio MAX*
- *Imageware*
- *Geomagic Studio*
- *RapidForm*

数字几何概述

- 数字几何主要关注多边形网格和体结构数据的处理
  - 表达方法
  - 采集方法
  - 处理方法



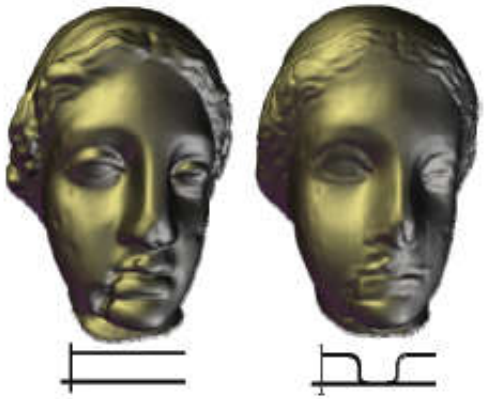


课程内容

- 数字几何的概述
- 数字几何基本问题
- 几何模型表示方法
- 表面网格的处理方法
- 3D打印中的应用

数字几何处理的基本问题

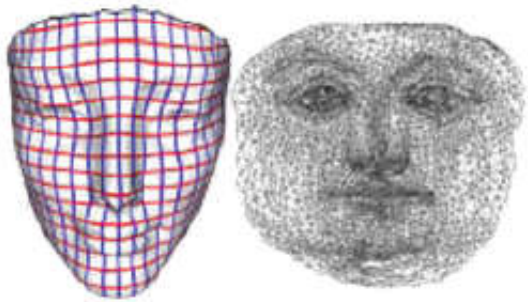
模型获取



几何滤波

几何分析

模型编辑

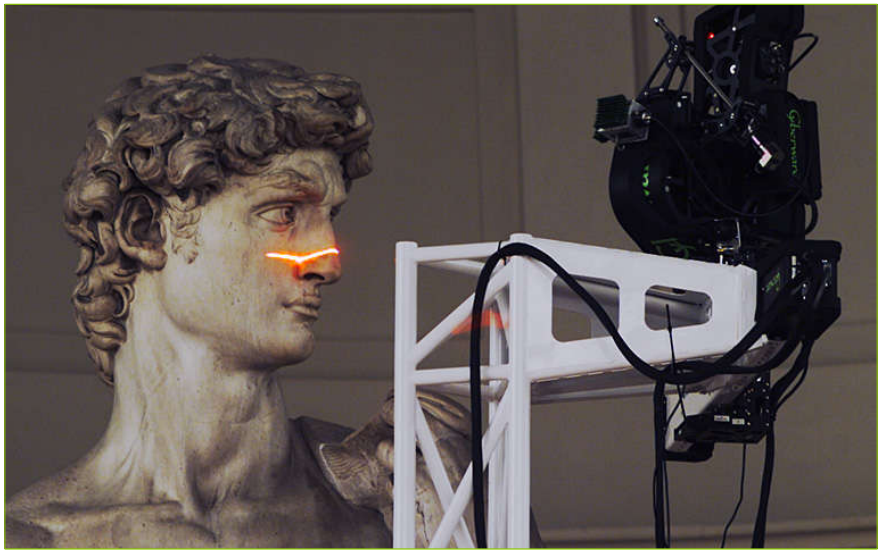


参数化

模型检索

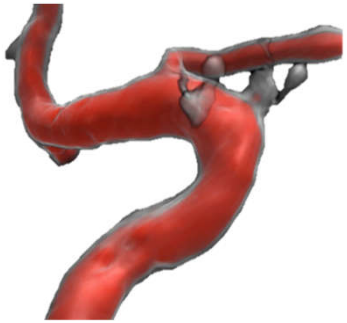
1.模型获取

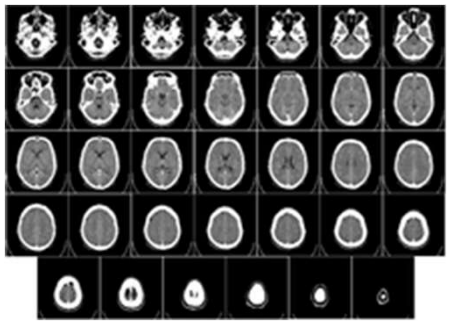
- 3D扫描
  - 依赖于激光、结构光、立体视觉硬件
  - 依赖ICP、Marching Cubes等算法。

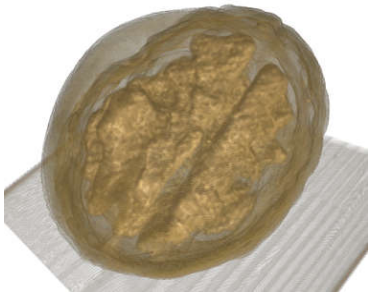


1.模型获取

- 医疗影像重建
  - CT（计算机断层扫描）
  - MRI（核磁共振）
  - 其他方法

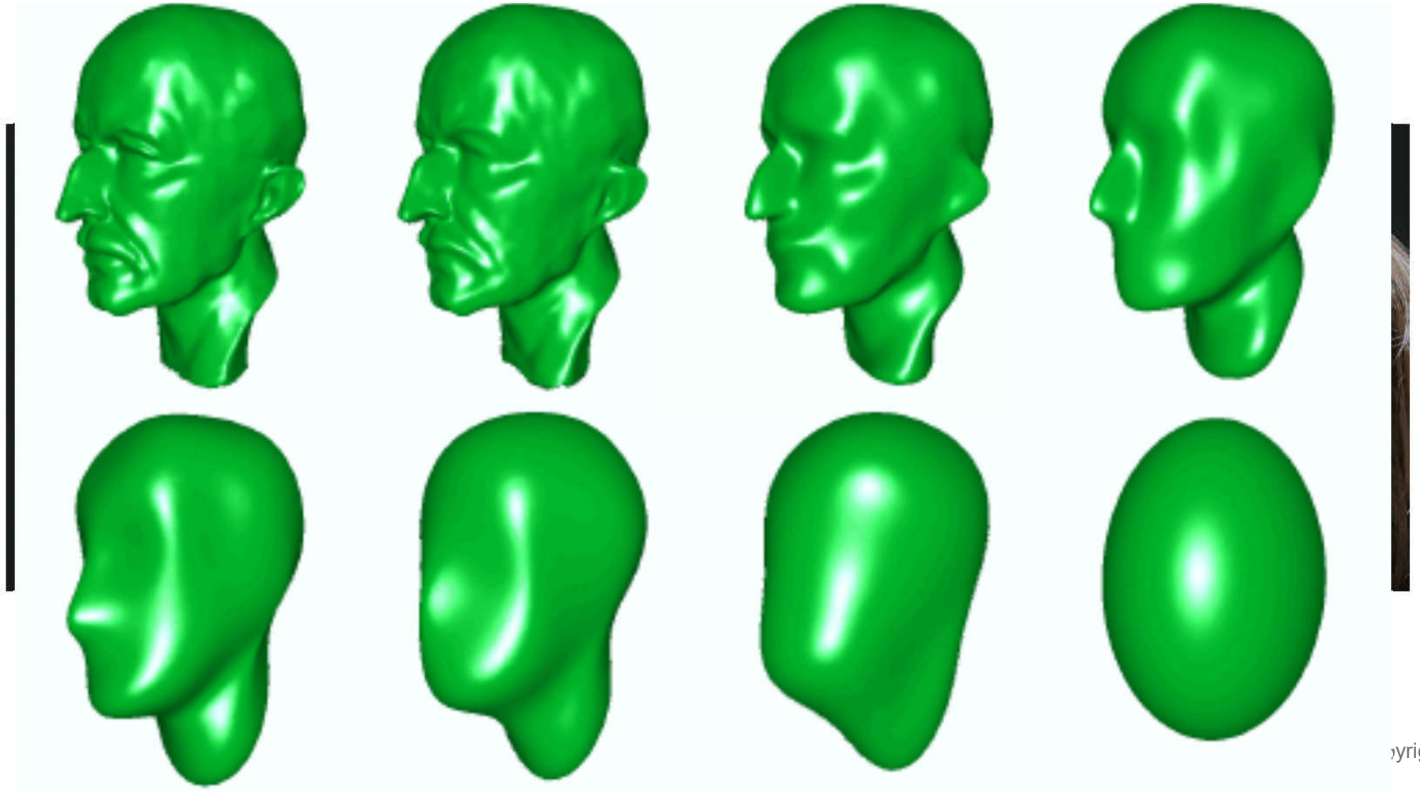






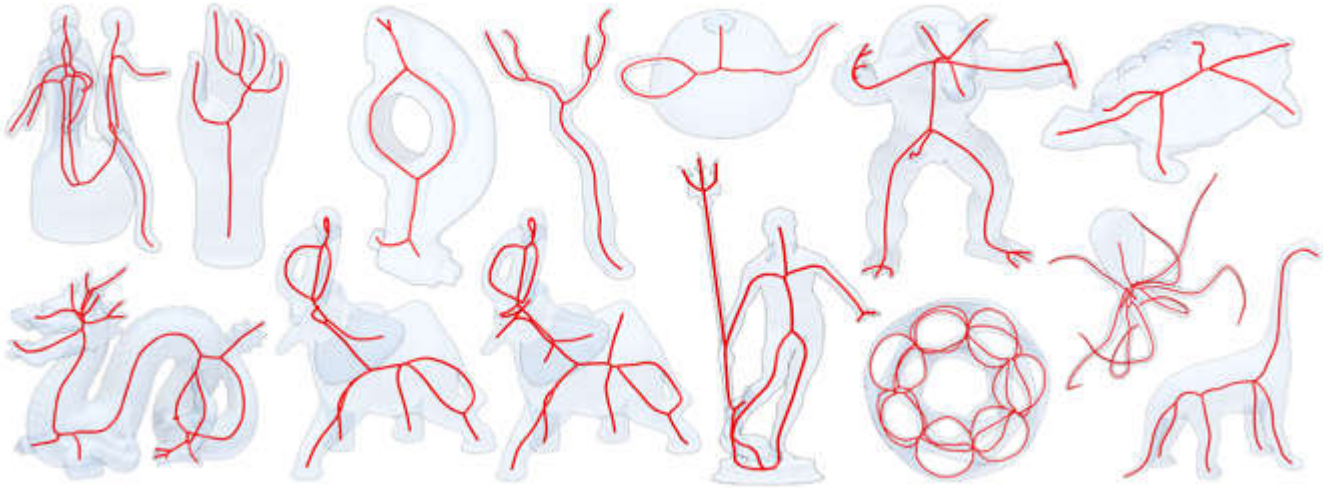
2. 几何滤波

- 变换：离散Fourier变换、小波变换、双边滤波、基于曲率流的滤波技术和Wienner滤波技术
- 从二维到三维



3. 几何分析

- **拓扑分析**
- 数据压缩
- 特征提取



中线提取[Andrea 2012]

3. 几何分析

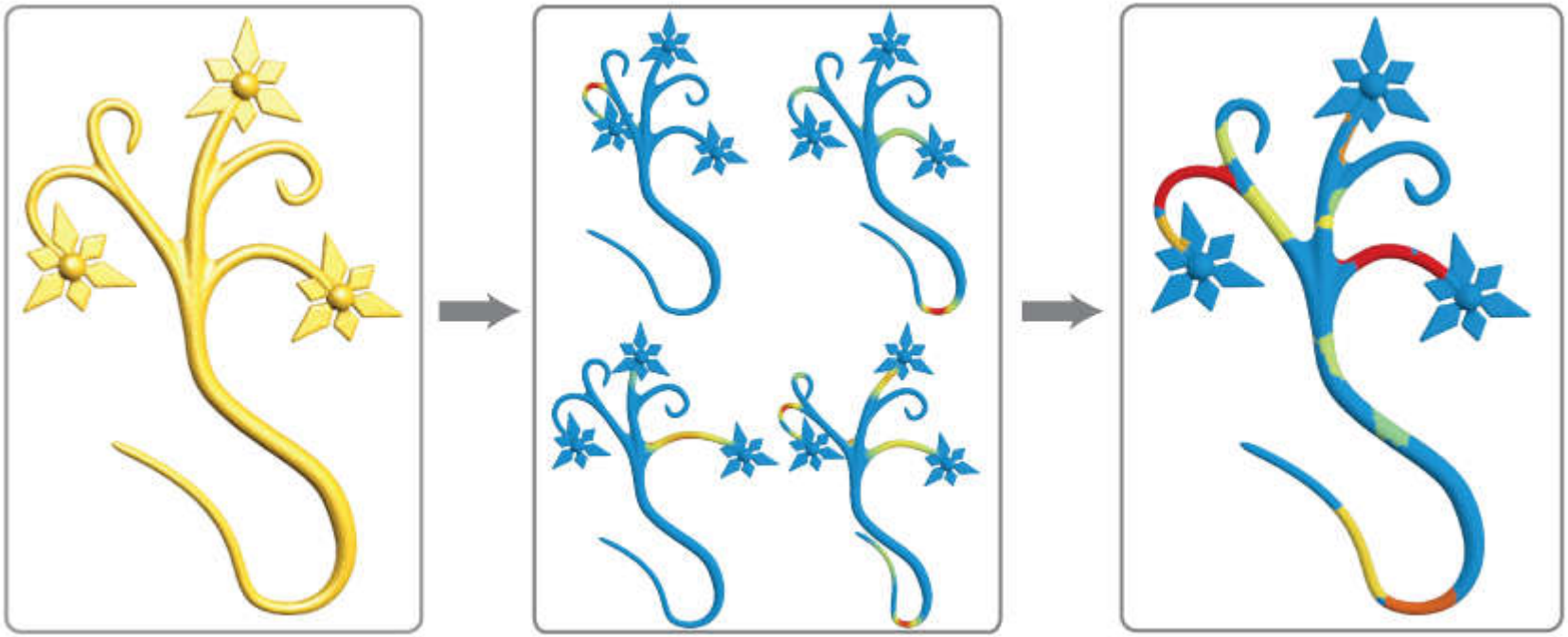
- 拓扑分析
- **数据压缩**
- 特征提取

Zip->多尺度表达->特征映射

3. 几何分析

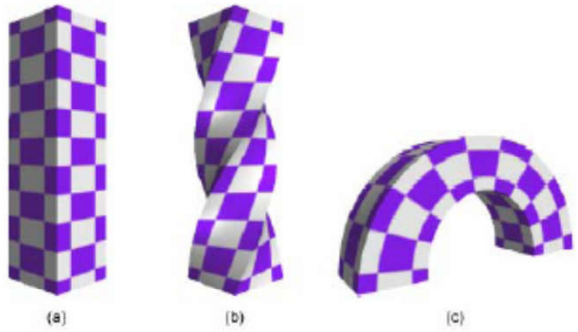
- 拓扑分析
- 数据压缩
- **特征提取**

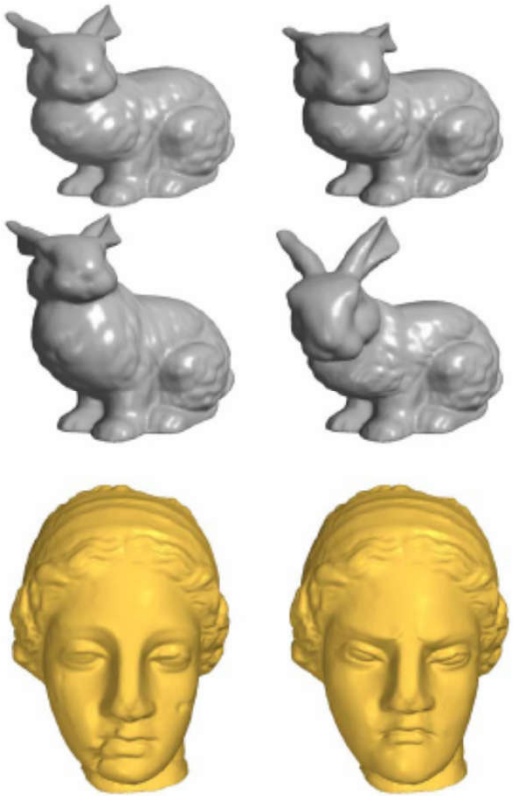
局部应力分布检测 [Zhou 2013]



4.数据编辑-几何变形

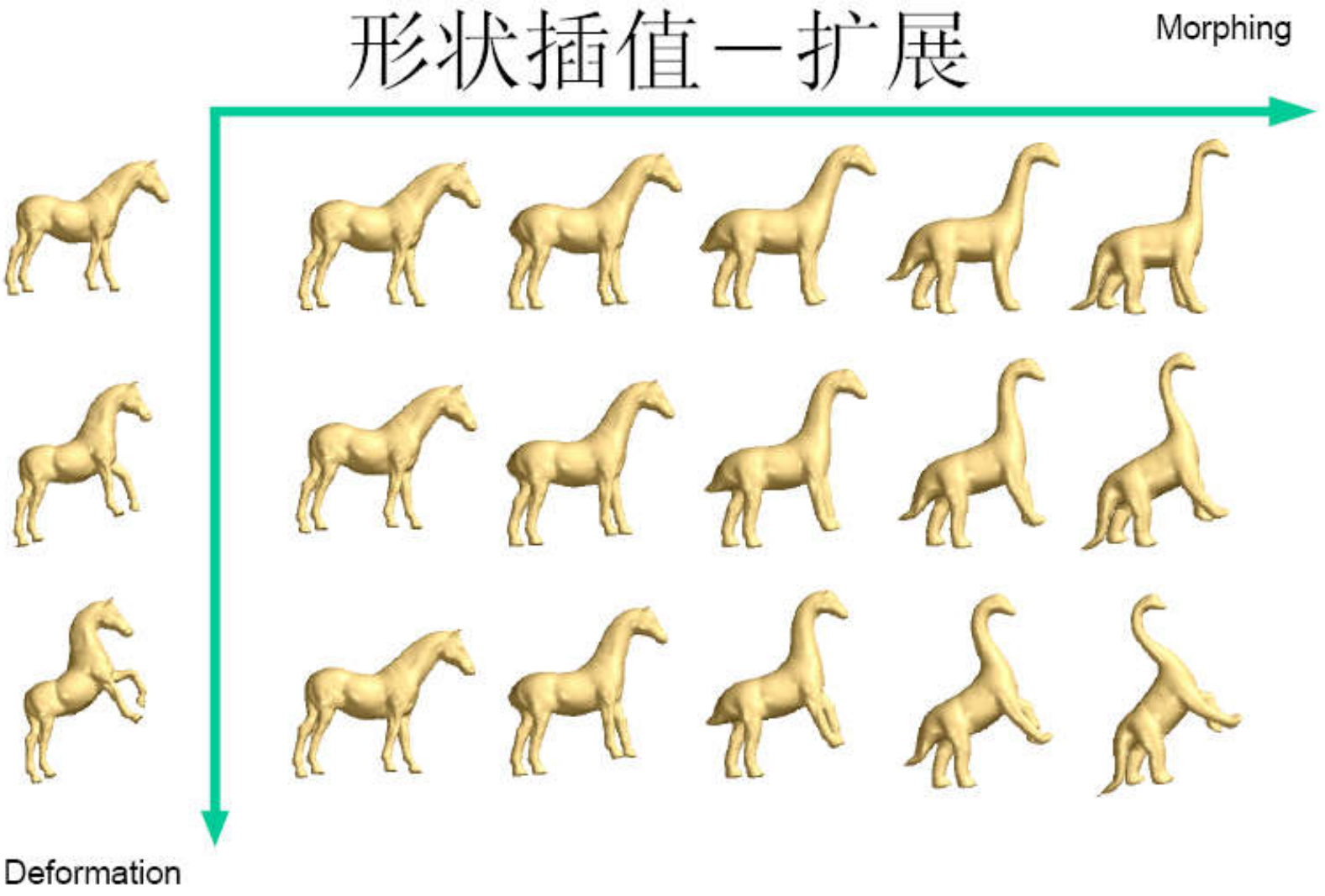
- 特效和**计算机动画**的生成。
  - 提高变形结果的质量
  - 降低用户交互工作量。





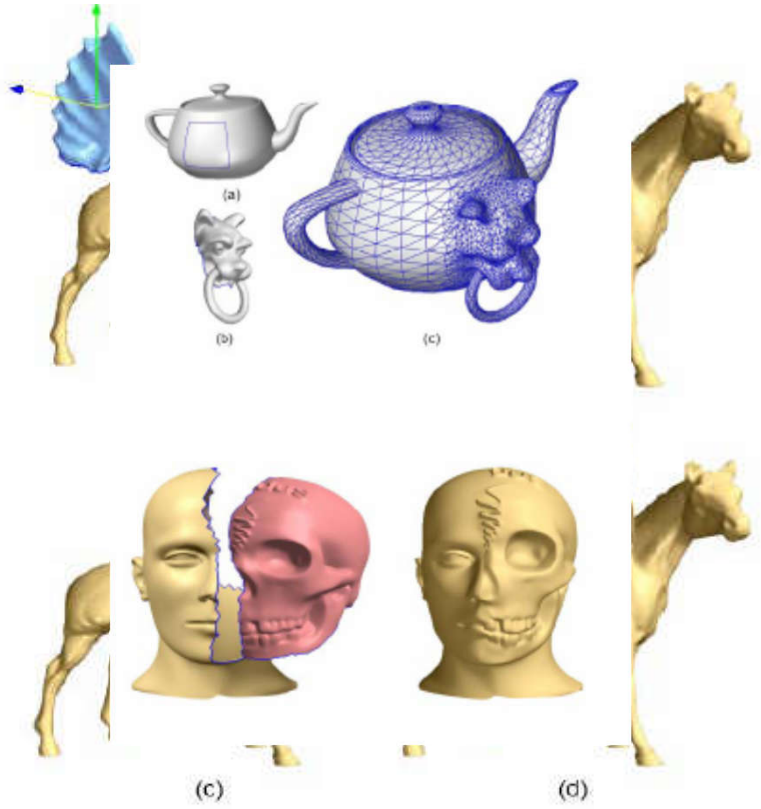
4.数据编辑-几何插值

- 形状插值—扩展



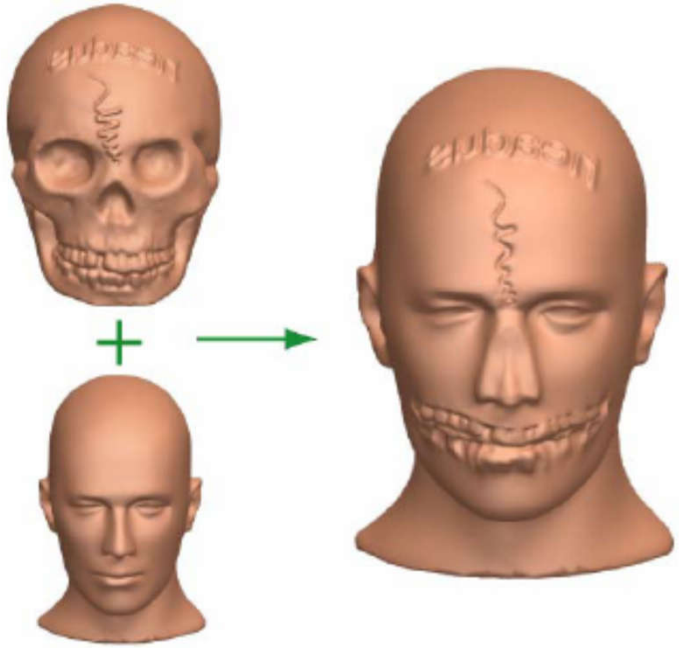
4.数据编辑-几何融合

- 将来自不同模型多个部分组件融合成为一个新的整体，长用于原型数据的快速获取。问题的难点在于如何根据融合边界自适应的调整各个组件的大小和形状，以保持融合边界能够自然的过渡。几何融合既可以用于图形生成，也可用于图形修补。



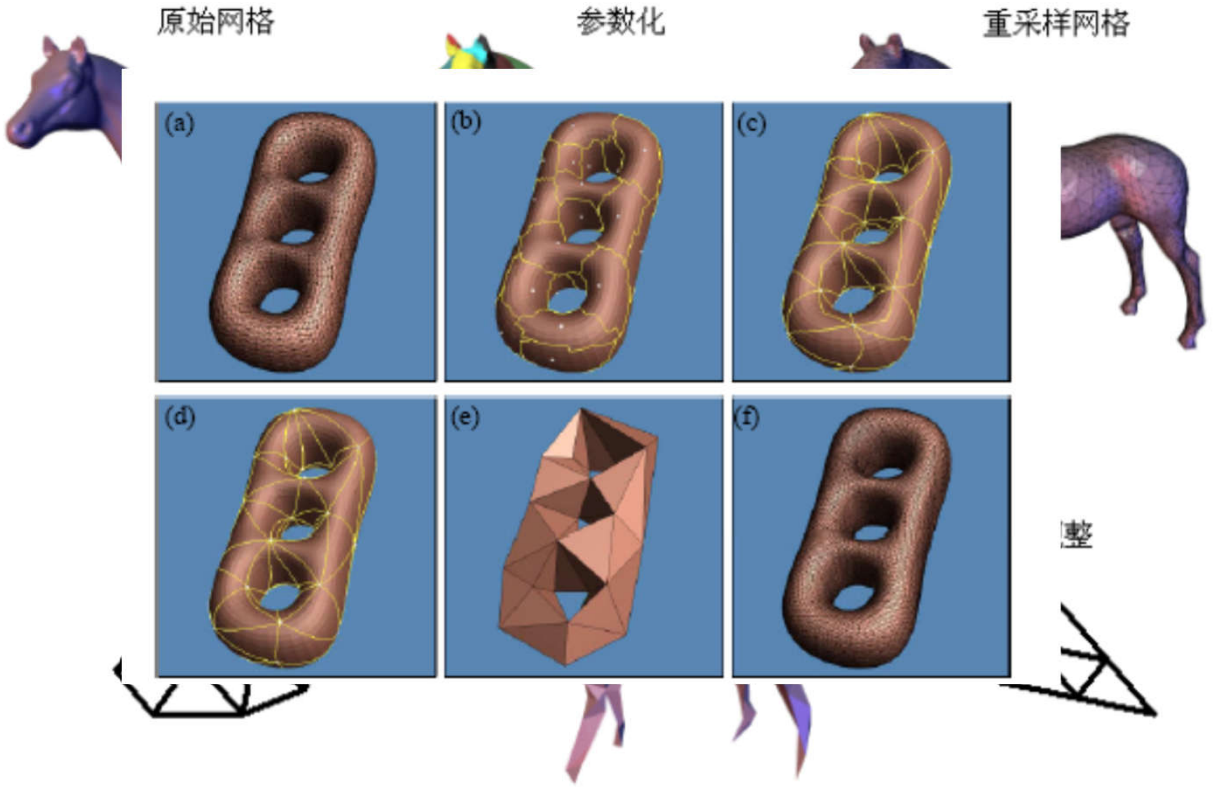
4.数据编辑-几何融合

- 将源模型上的几何信息，如几何细节特征或者变形信息重用于目标模型之上，常用于数据驱动的快速建模。问题的难点在于从源模型中抽取的几何信息自适应的调整，以保证能够适用于可能与源模型几何相差较大的目标模型之上，并得到自然重用。在这里常常隐含的要求源模型与目标模型基本相似。



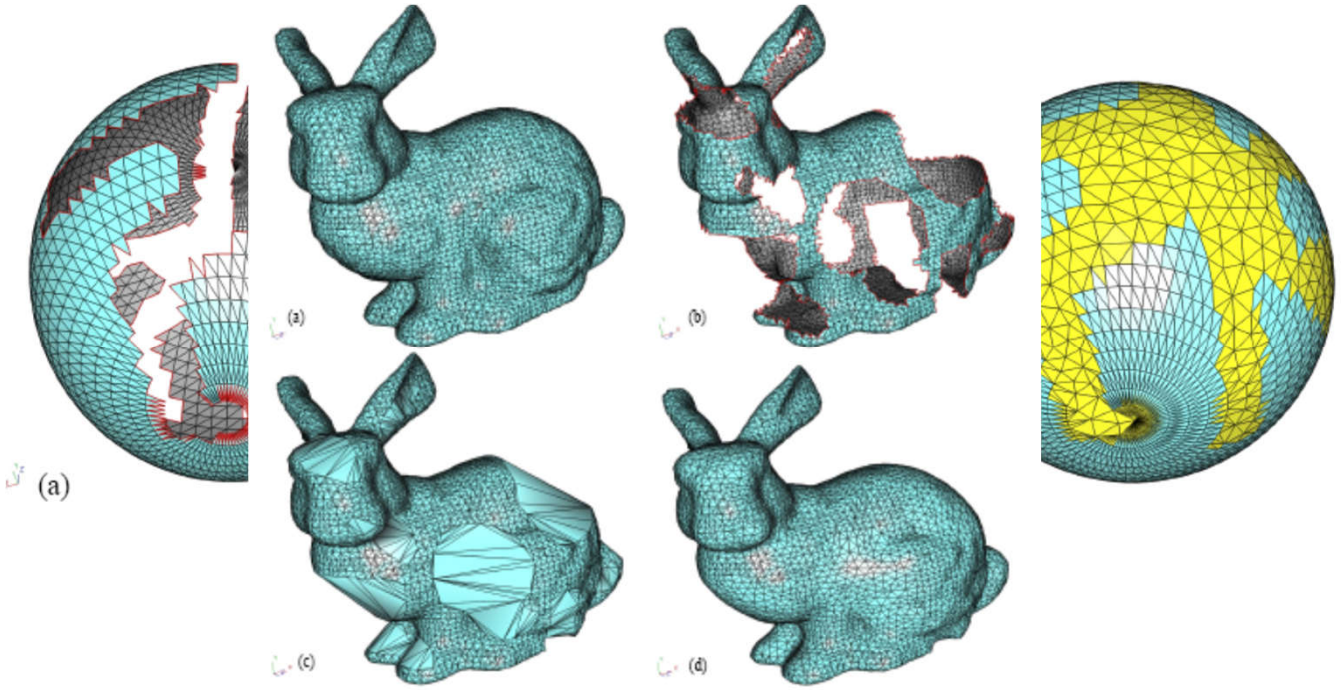
4.数据编辑-几何重采样

- 由于原始模型的采用常常不能满足特定的编辑需求，为此，需要根据具体应用的要求对原始模型进行重新采样。常见的重采样要求包括采样密度均匀，采样密度随表面曲率自适应的调整，顶点之间满足子分连接关系。问题的难点在于如何尽可能的降低采样误差，以及如何有效的保持模型表面的尖锐关系等。



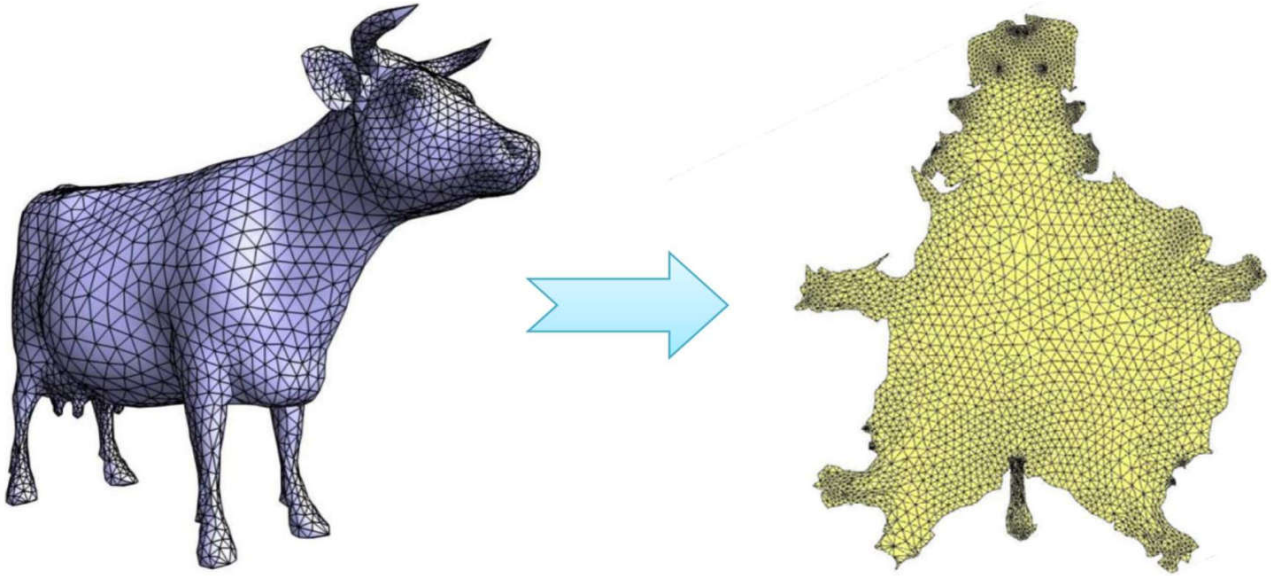
4.数据编辑-几何重构

- 根据原始模型中常常会因为扫描过程的错误产生缺憾，或者缺损直接是物理模型本身损坏造成的，这就需要根据模型本身特征进行补缺。常用的技术有曲面重够、对称修补和实体重构。



5.参数化

- 在数字几何领域，参数化的意思是把一个高维表面，展成一个低维表面



- 2D到1D

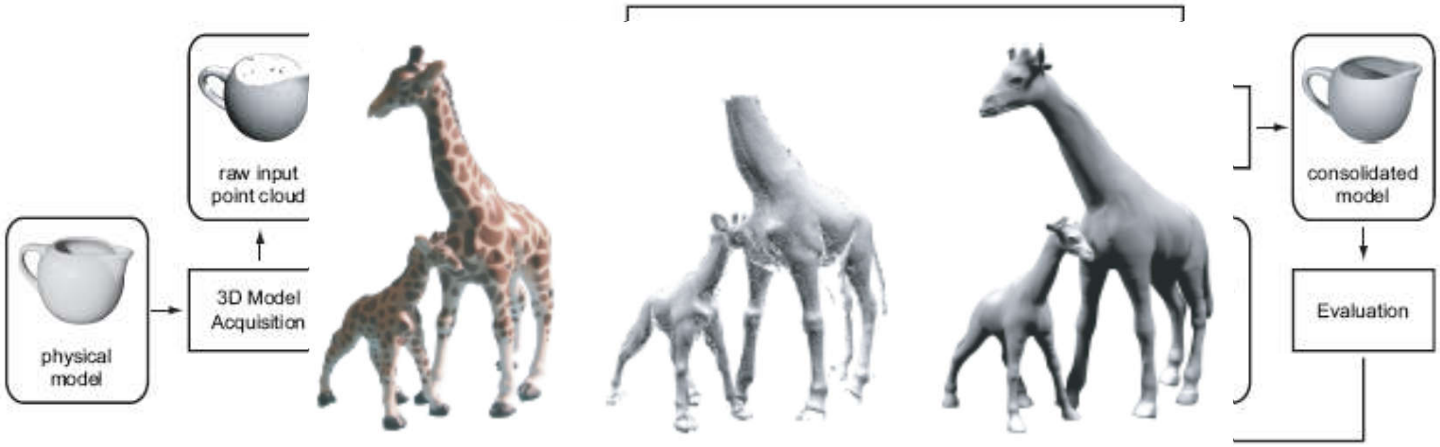
$$\begin{cases} x = x(t), \\ y = y(t), \end{cases} \quad t \in [a, b],$$

- 3D到2D

$$\begin{cases} x = x(u, v), \\ y = y(u, v), \\ z = z(u, v), \end{cases} \quad (u, v) \in [a, b] \times [c, d]$$

6.数据检索

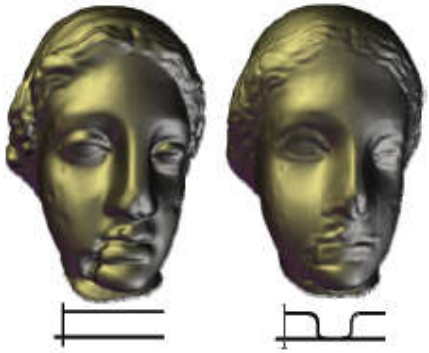
- 随着各种建模方法的发展和建模软件的普及，工业界和学术界每天都在创建新的几何模型，其数量快速增长，已经达到了相当的规模。



(Mark P. 2005)

数字几何研究

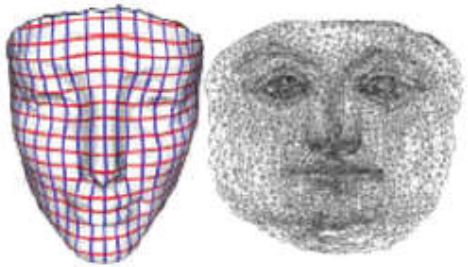
模型获取



几何滤波

几何分析

模型编辑



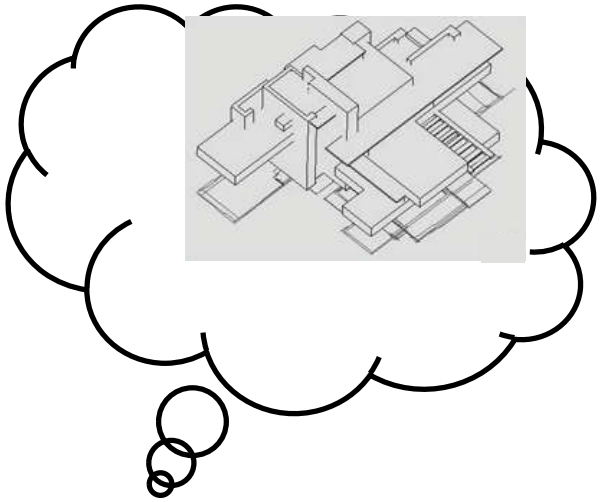
参数化

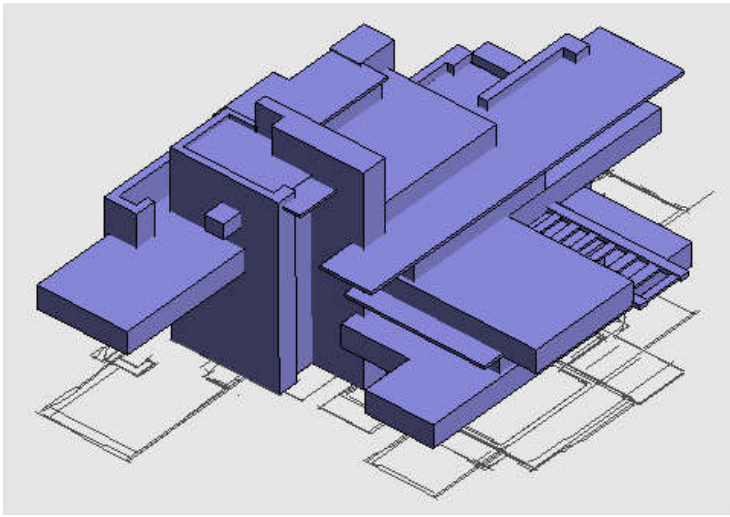
模型检索

- 一般而言数字几何研究的问题包括上述6个方面，而狭义的数字几何处理则特指几何数据编辑。

数字几何模型的表示方法

- 表达形式



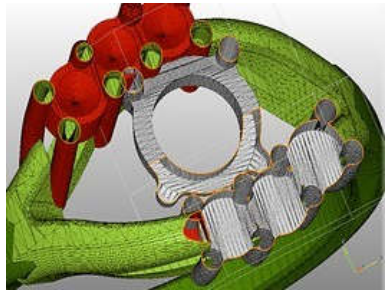


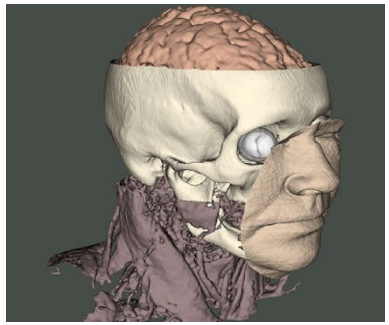
课程内容

- 数字几何的概述
- 数字几何基本问题
- 几何模型表示方法
- 表面网格的处理方法
- 工业应用

数字几何概述

越来越多的表达方式





- （1）多边形网格、隐曲面、边界表达…
- （2）实体造型、体素、粒子…

数字几何模型的表示方法

- 表达形式
  - 曲线与曲面
  - 点云
  - 隐式曲面
  - 体数据
  - 实体表示
  - 边界表达

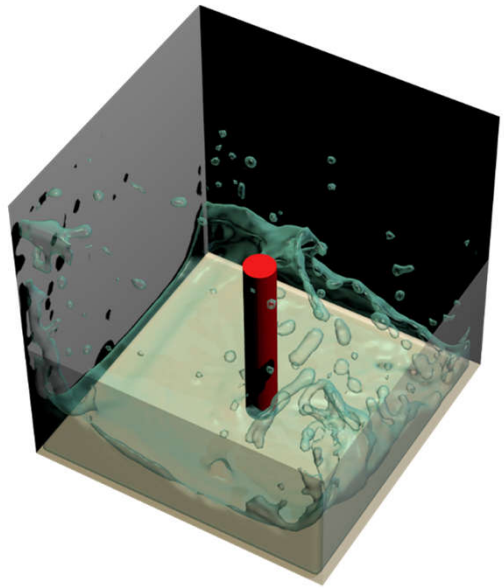
数字几何模型的表示方法

- 隐曲面（Implicit Surface）

  •不采用直接参数形式表示，
  如：$c(t) = (x(t), y(t))$

  •形如$F(x, y) = 0$定义的曲面

  •将参数化引入到函数中
  $F(x(t), y(t)) = 0$



数字几何模型的表示方法

- 为什么需要隐曲面?

参数曲面

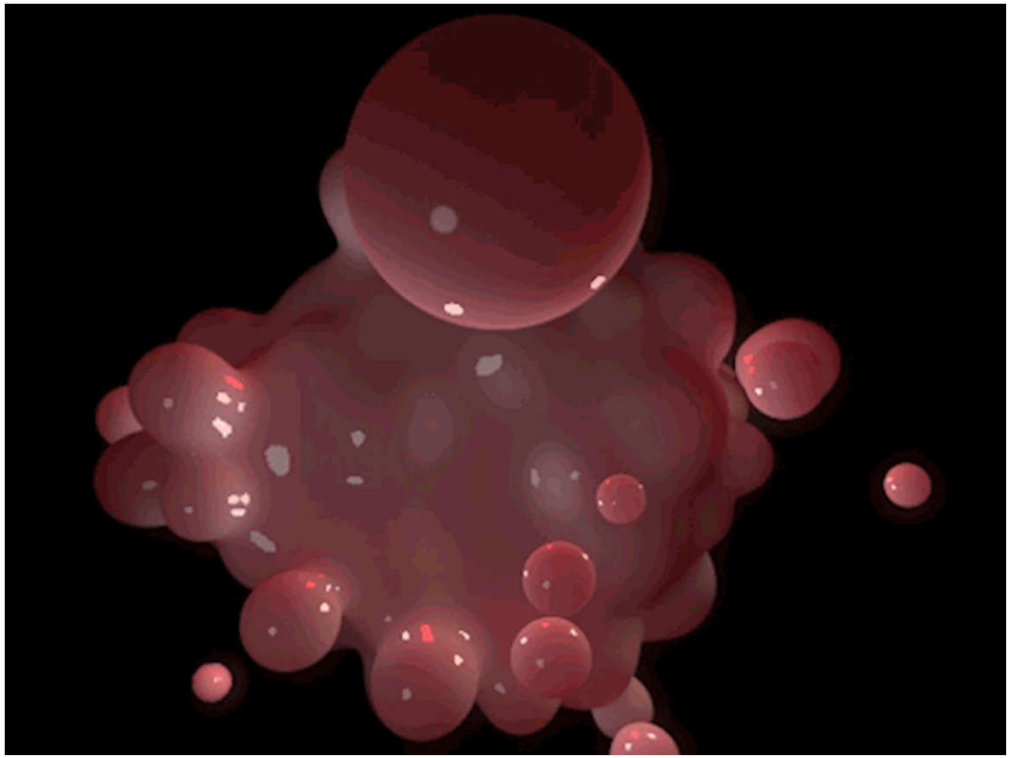
- 容易得到点坐标
  For curve/surface $c(t)$ $= \big(x(t), y(t)\big)$:
  choose $t = \tau$ and evaluate $c(\tau)$.
- 不易判定点是否在曲面/曲线上
  Requires solving
  $c(t) = (x, y)$ for t .

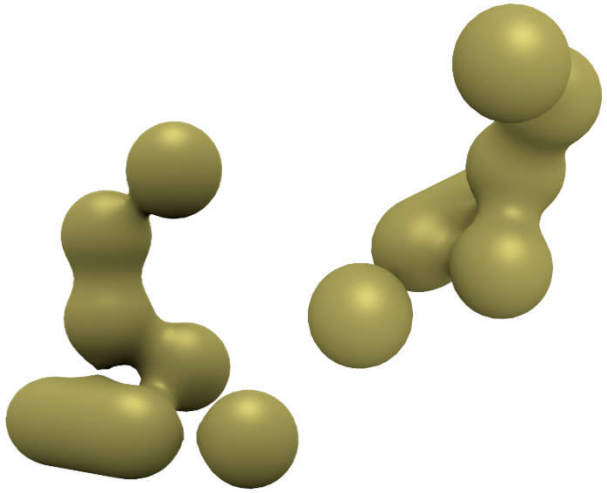
隐曲面

- 易判定点是否在曲面/曲线上
  For curve $F(x, y) = 0$:
  Point $\boldsymbol{P} = (x, y)$ is on curve
  if $F(\boldsymbol{P}) = 0$.
- 难于直接得到点坐标
  需要求解
  $$F(x, y) = 0$$

数字几何模型的表示方法

- 隐曲面-容易生成但难于交互编辑

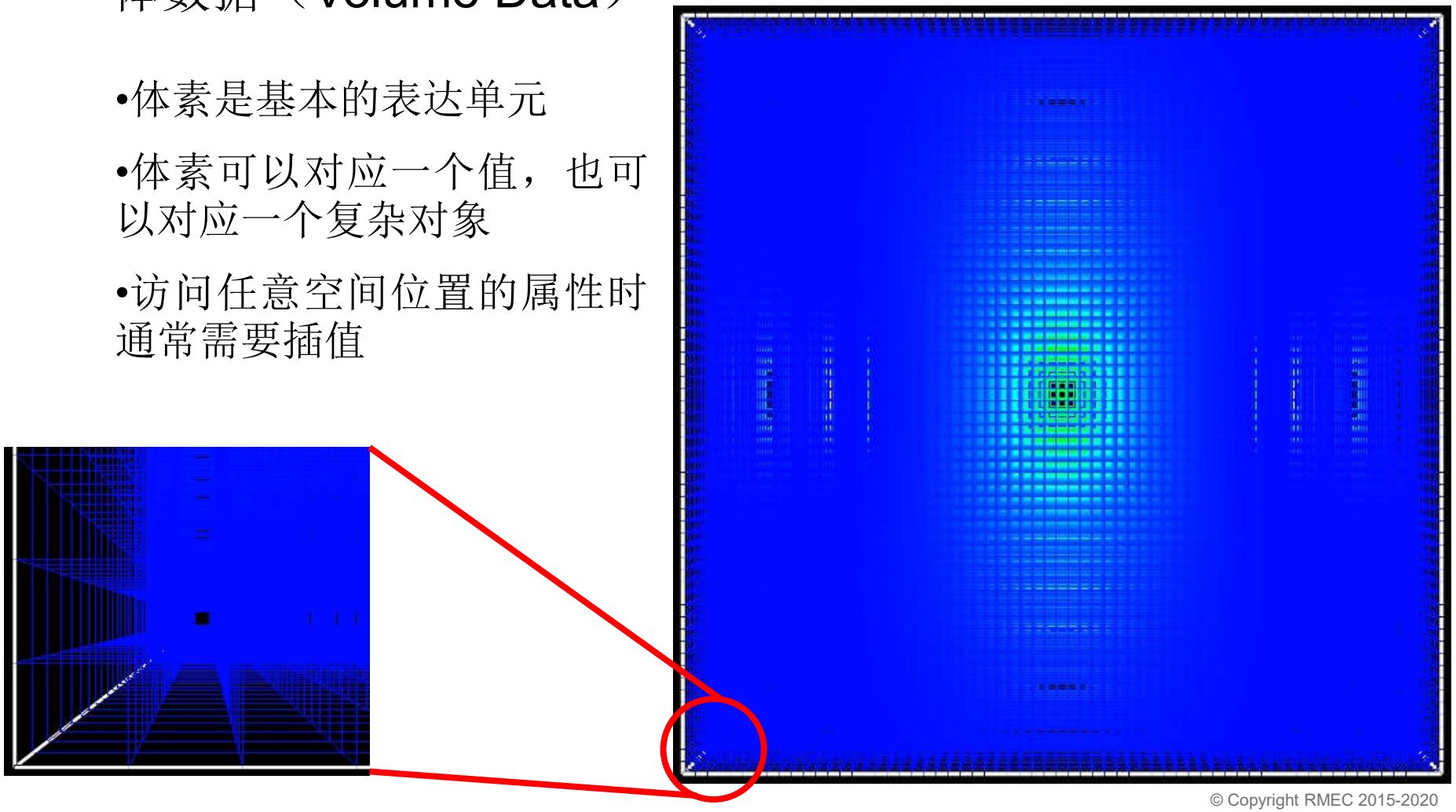




数字几何模型的表示方法

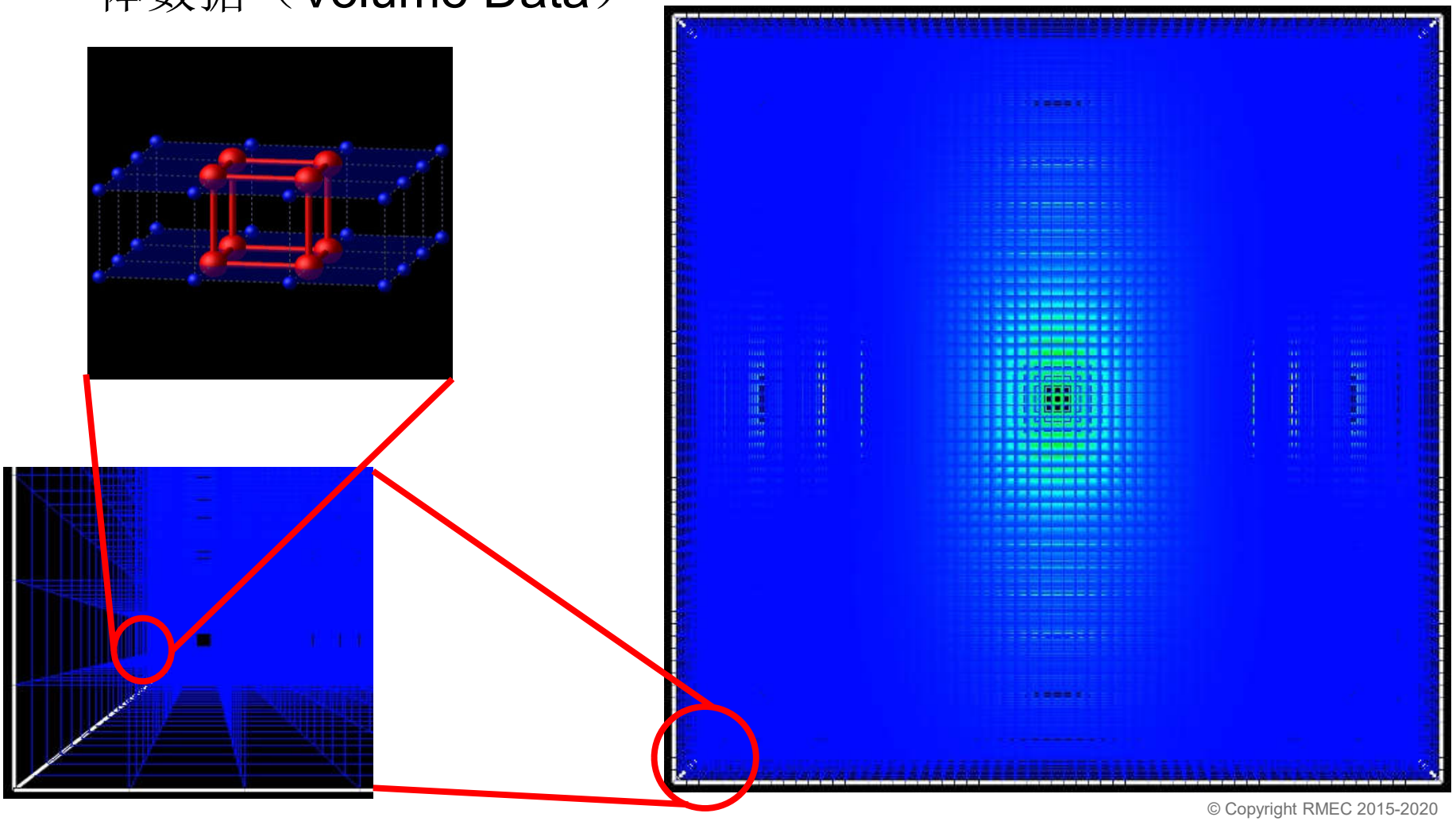
- 体数据（Volume Data）

  - 体素是基本的表达单元
  - 体素可以对应一个值，也可以对应一个复杂对象
  - 访问任意空间位置的属性时通常需要插值



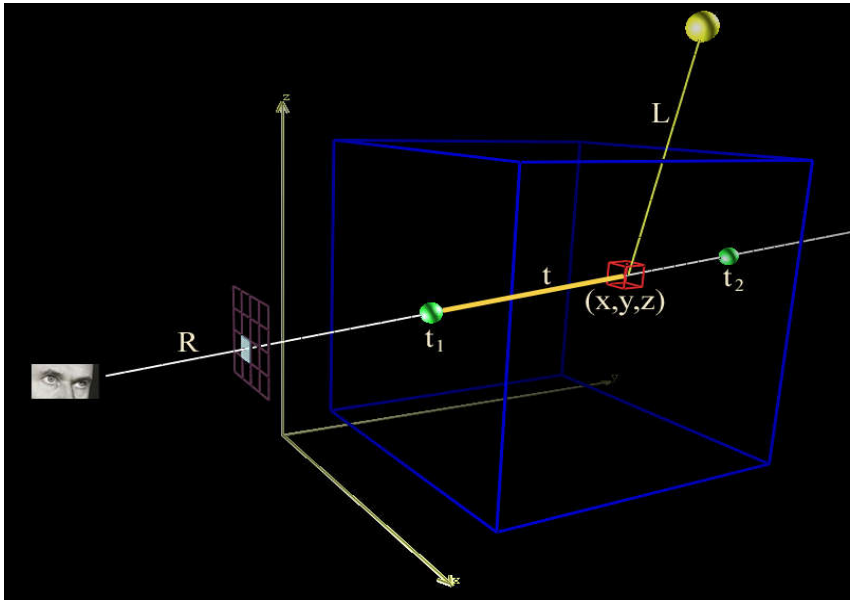
数字几何模型的表示方法

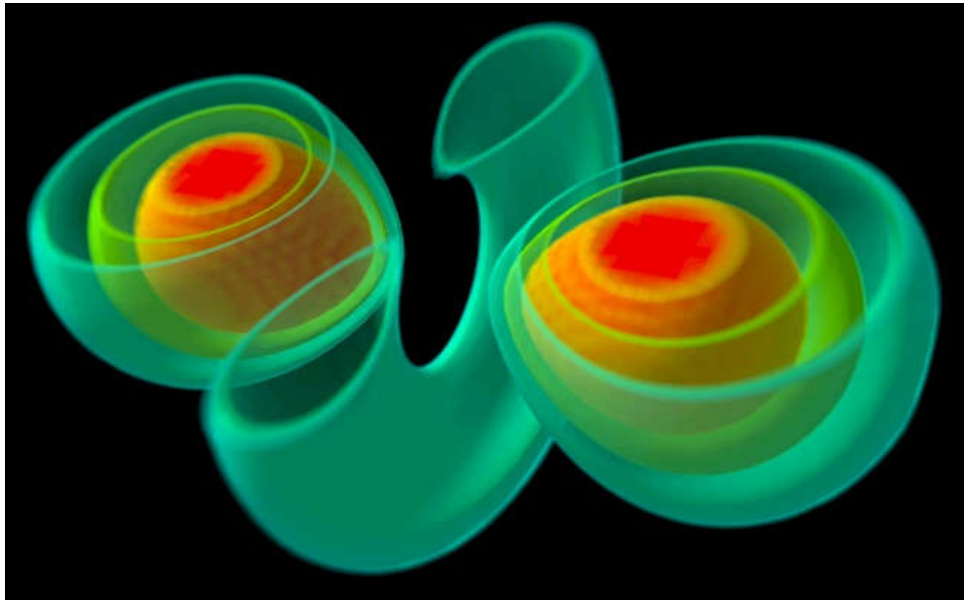
- 体数据（Volume Data）



数字几何模型的表示方法

- 体数据（Volume Data）





•显示方法

-光线跟踪

数据量大

难于交互编辑

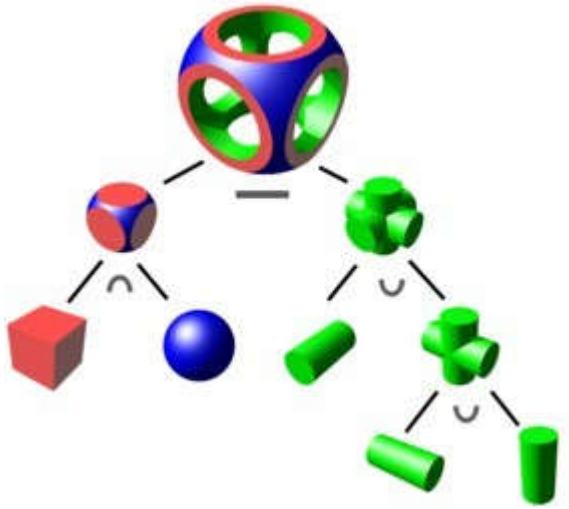
数字几何模型的表示方法

- 实体表示（Constructive Solid Geomery,简称CSG）

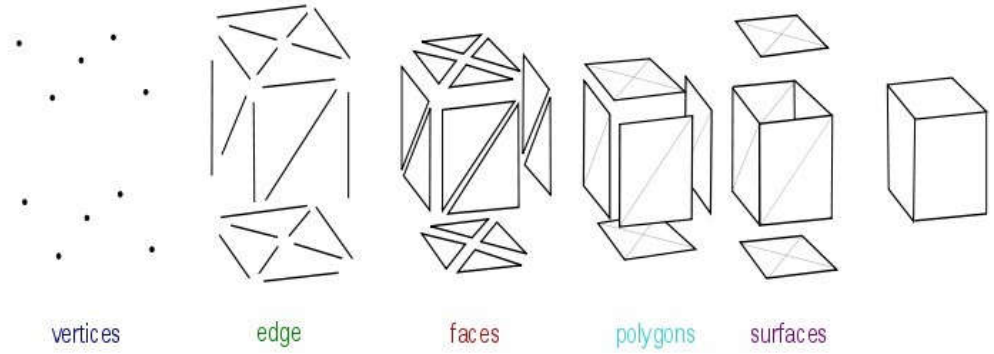
  •体素可以是立方体、圆柱、圆锥等

  •其运算变换为并、交、差等正则集合运算

  •可采用一颗有序的二叉树来表示

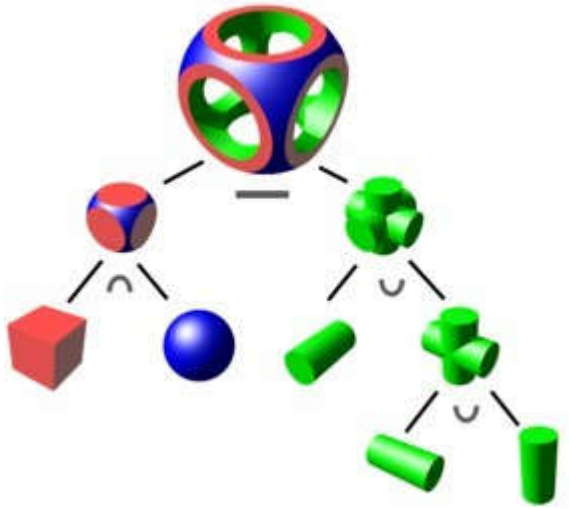


- 边界表示(Boundary Representation, 简称B-rep)



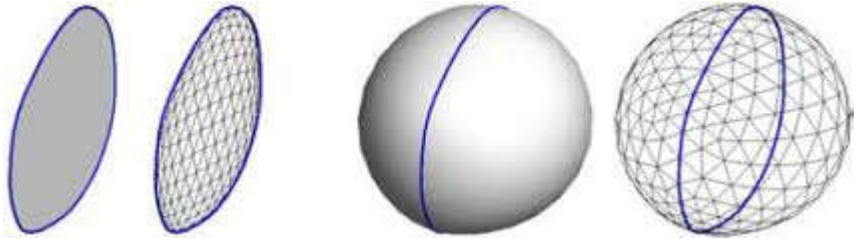
评价

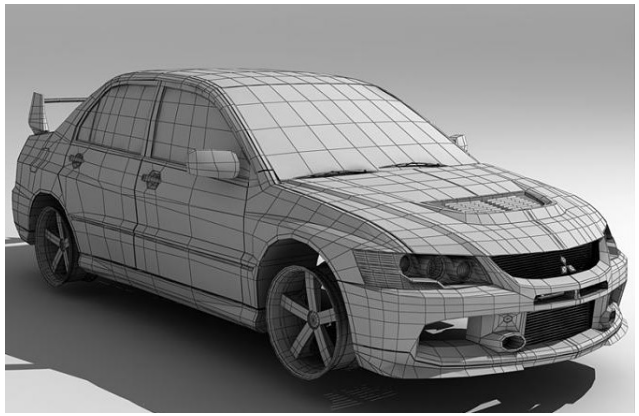
- CSG方法
  - 优点：（1）数据结构简单
    （2）易于转换成边界表示
    （3）易于修改
  - 缺点：（1）形体的表示收体素种类和体素操作类型的限制
    （2）形体的局部操作不易实现
    （3）难于绘制；



评价

- B-rep方法
  - 优点：（1）几何元素显示给出，易于检索
    （2）方便对物体进行各种局部操作
    （3）便于在数据结构上附加各种非几何信息
  - 缺点：（1）数据结构复杂导致维护困难
    （2）需要运用欧拉操作来保证所表示形体的有效性。





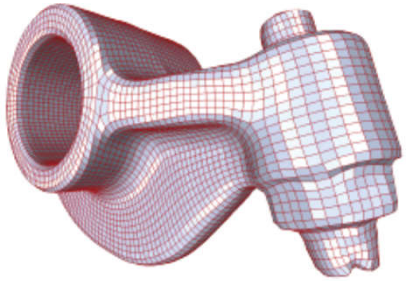
典型B-REP——多边形网格

- 多边形网格是一种典型的边界表示，它将物体的表面表示成一系列面片的交集。
- 特点：

（1）可以任意精度来表示任意拓扑的物体表面。

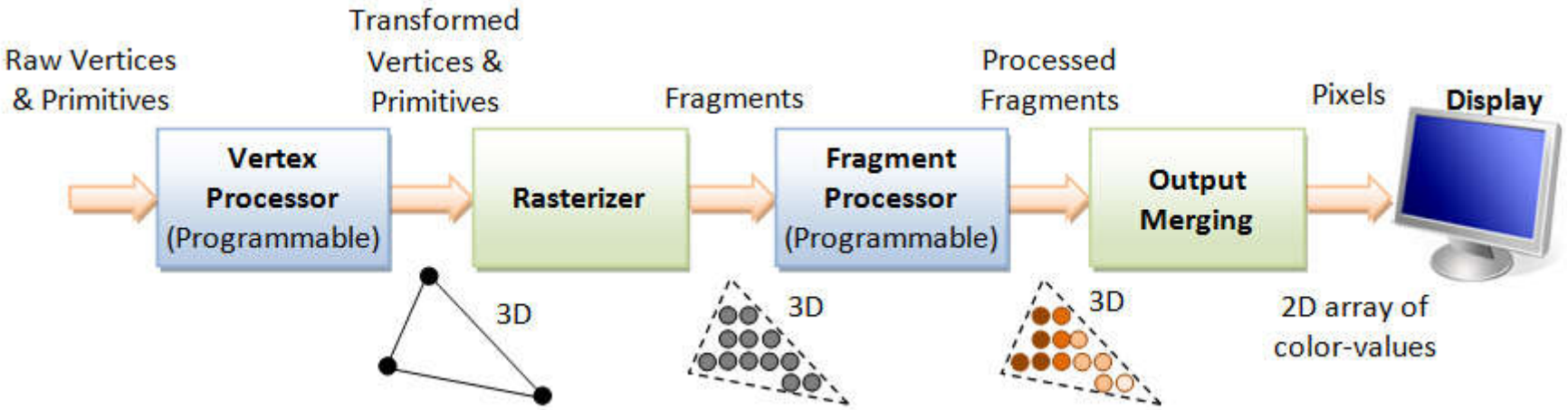
（2）三角形网格表示适合于现代渲染流水线硬件的设计。

（3）四角形网格易于附加纹理。



- 在这里我们主要介绍的都是指三角形网格，简称三角网格。通常的文件格式包括STL, VRML, DXF, OBJ, PLY, ZPR 和VFX等

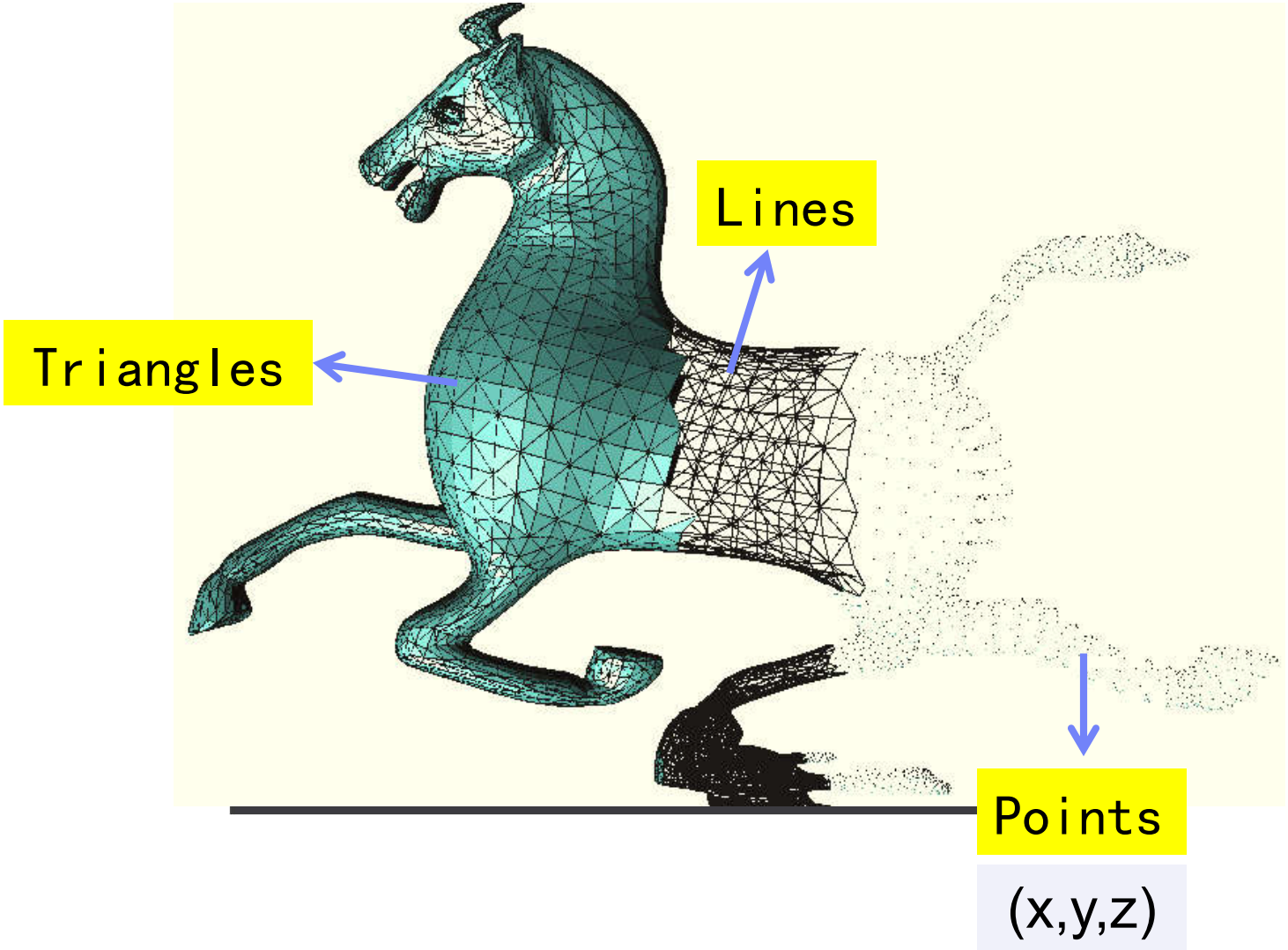
渲染管道



- 渲染管道指计算机中将3D几何输入转化为2D屏幕输出的流程
- 像素化的操作是通过固化在硬件中的算法实现的
- 三角网格模型被普遍使用

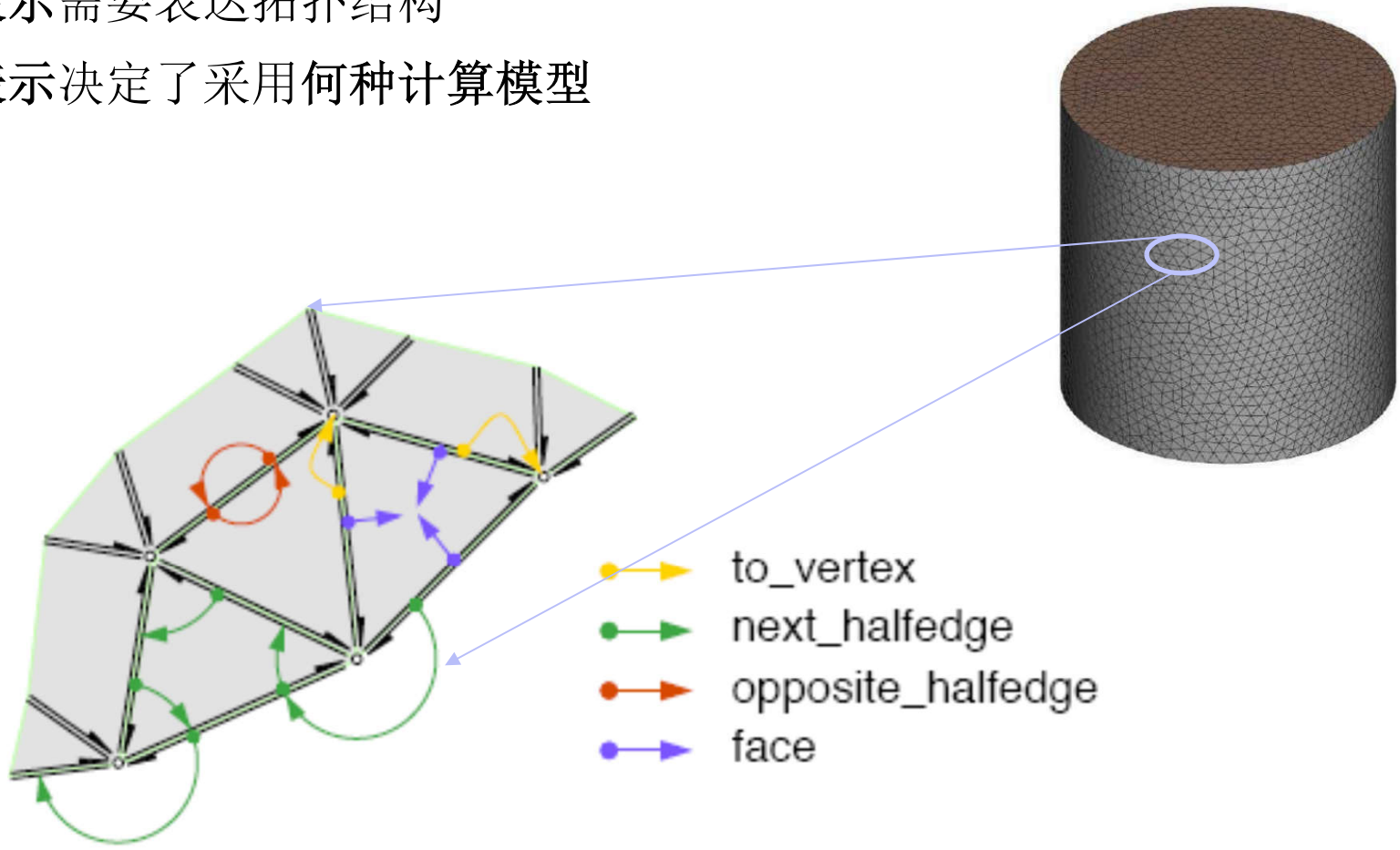
典型的多边形网格-三角网格

一组三角形“汤”



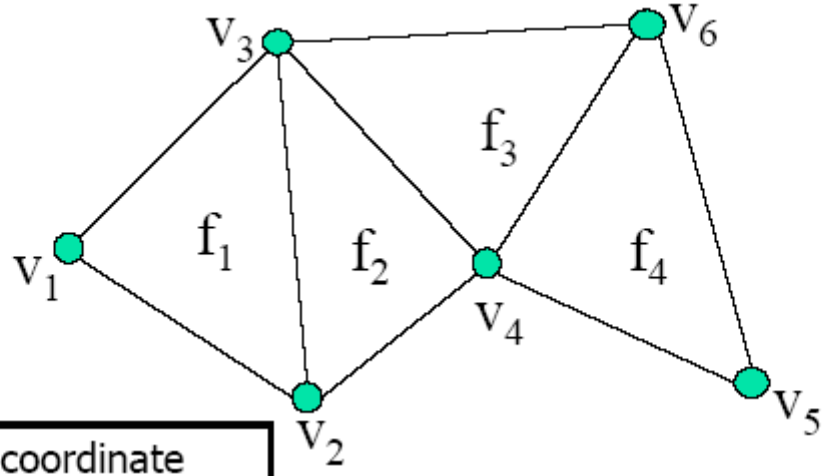
典型的多边形网格-三角网格

- **数据表示**需要表达拓扑结构
- **数据表示**决定了采用**何种计算模型**



典型的多边形网格-三角网格

- 三角面列表



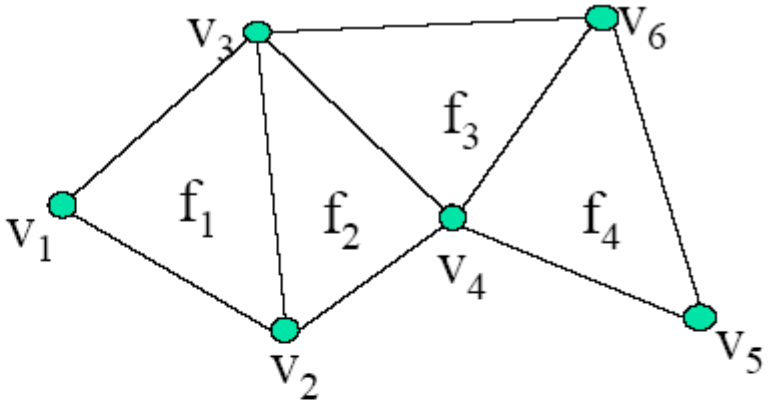
vertex	coordinate
v_1	(x_1,y_1,z_1)
v_2	(x_2,y_2,z_2)
v_3	(x_3,y_3,z_3)
v_4	(x_4,y_4,z_4)
v_5	(x_5,y_5,z_5)
v_6	(x_6,y_6,z_6)

face	vertices (ccw)
f_1	(v_1, v_2, v_3)
f_2	(v_2, v_4, v_3)
f_3	(v_3, v_4, v_6)
f_4	(v_4, v_5, v_6)

典型的多边形网格-三角网格

vertex	coordinate
v_1	(x_1,y_1,z_1)
v_2	(x_2,y_2,z_2)
v_3	(x_3,y_3,z_3)
v_4	(x_4,y_4,z_4)
v_5	(x_5,y_5,z_5)
v_6	(x_6,y_6,z_6)

face	vertices (ccw)
f_1	(v_1, v_2, v_3)
f_2	(v_2, v_4, v_3)
f_3	(v_3, v_4, v_6)
f_4	(v_4, v_5, v_6)



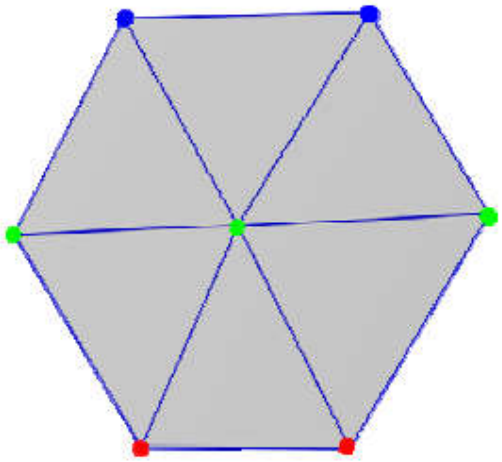
	v_1	v_2	v_3	v_4	v_5	v_6
v_1		1	1			
v_2	1		1	1		
v_3	1	1		1		1
v_4		1	1		1	1
v_5				1		1
v_6			1	1	1	

课程内容

- 数字几何的概述
- 数字几何基本问题
- 几何模型表示方法
- 表面网格的处理方法
- 3D打印中的应用

三角网格的基本定义

- 网格的数学表达



任意一个网格可以表达为一个二元组

$M=\{P,K\}$

其中P是N个元素的三维欧氏空间中的点集

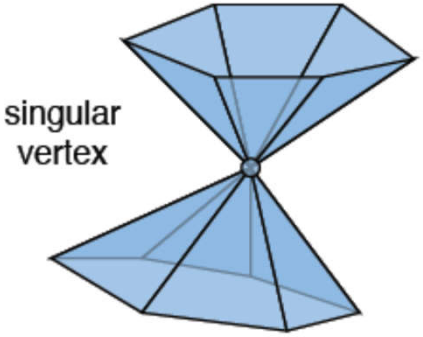
其中$P=\{p_i \in R^3\}$

K是一个抽象单纯复形，描述了网格的全部连接信息。

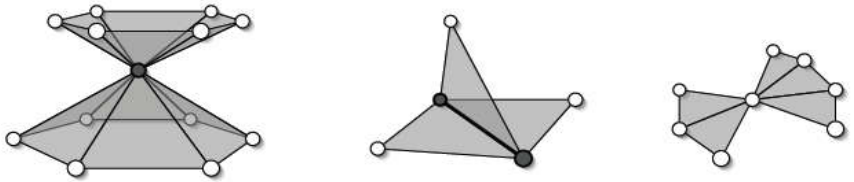
$K=V \cup E \cup F$

基本定义

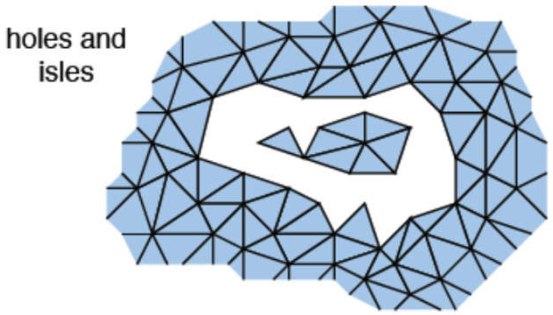
- 退化与非退化网格



如果网格M中出现重合的顶点或者重合的边，计算出将会出现**退化**现象。



- 连通与非连通网格

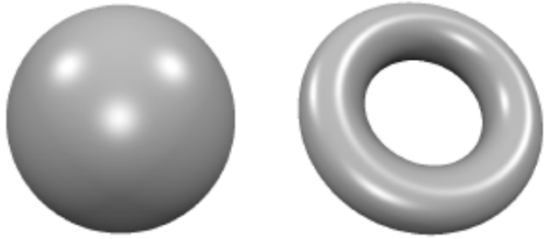


如果网格M中任意两个顶点之间，均存在一条由网格上的边组成的通路，我们称M为**连通网格**，否则为**非连通网格**。

连通网格的**连通分量**为1

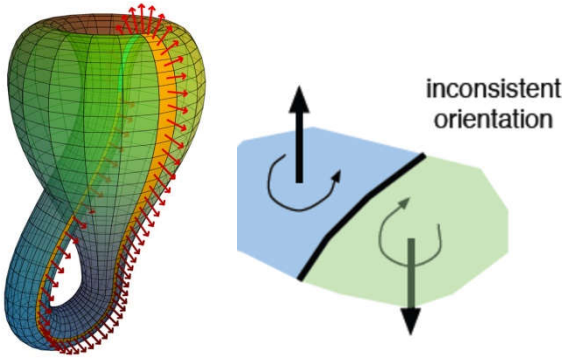
基本定义

- 单连通



连通是拓扑学的概念。 设X是拓扑空间，如果X中任何一个点的回路都可以连续地收缩成这个点，那么就称X为单连通的。

- 朝向

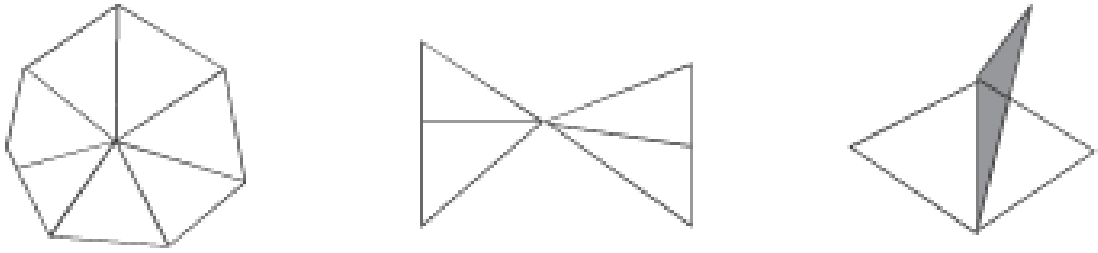


网格M中每个面片的顶点顺序成为**朝向**，表面网格只有两种朝向，分别以顺时针和逆时针为准。我们可以通过排列顶点的顺序，使得网格中所有面片都朝向一个方向，此时网格M成为可定向的,否则为不可定向网格。

二维流形

数字几何算法中一般要求处理的对象（表示曲面的网格）要满足**二维流形**的要求。

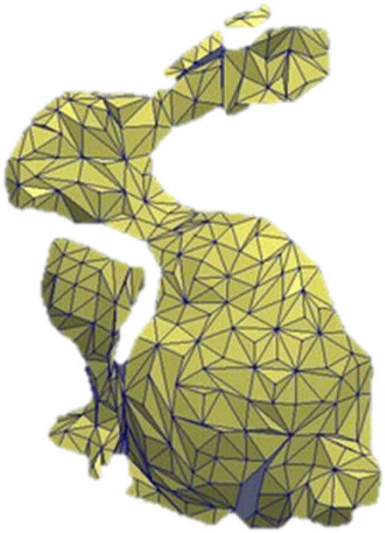
所谓**二维流形**是指曲面上任意顶点v为中心的邻域L(v)，若v位于曲面内部，则L(v)与一个圆盘同坯，若v位于边界，则要与一个半圆同坯。



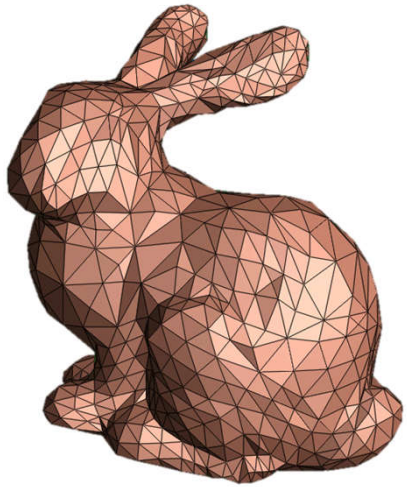
Manifold　　Non-Manifold　　Non-Manifold

开(open)网格与闭(close)网格

如果网格中不存在边界边，则称网格式封闭（Close）的，否则成网格为开（Open）的。



Open

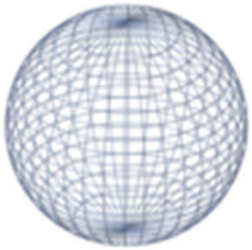


Close

如果一个封闭网格式“可制造的”，那么它是一个二维流形

网格的亏格（genus）

网格 中沿闭简单曲线切开但不切断曲面的最大曲线条数称为的**亏格**，也可以形象的理解为 中洞的个数。



亏格=0

亏格=1

亏格=3

欧拉公式

如何一个网格是一个二维流形，它满足欧拉公式

$$v+f-e = 2(c-g)-b$$

v：顶点数　　*f*：三角面数　　*e*：边数

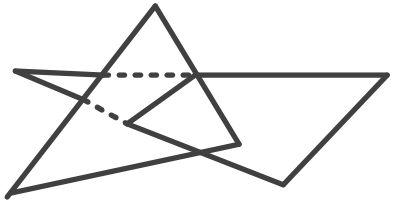
c：*连通分量个数（对于封闭的连通网格为1）*

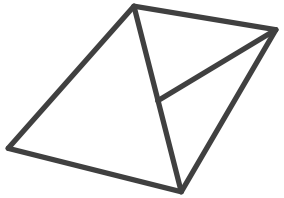
g：*亏格个数*

b：*边界数（在连通网格中为0）*

应用

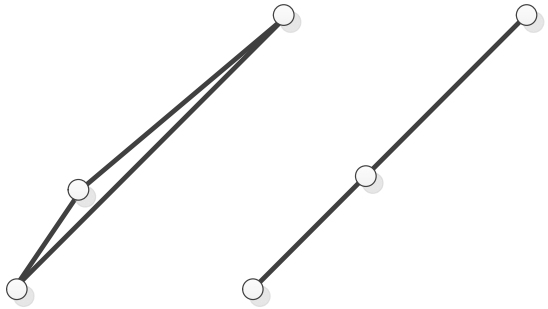
- 网格模型的检测
  - 拓扑错误

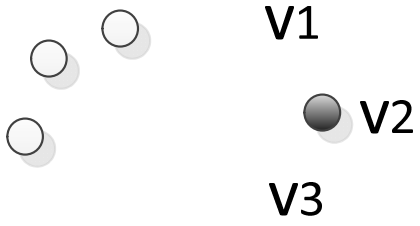




$v+f-e == 2(c-g)-b$ ？

  - 退化面片





问题

- 隐曲面表达的优点是什么？

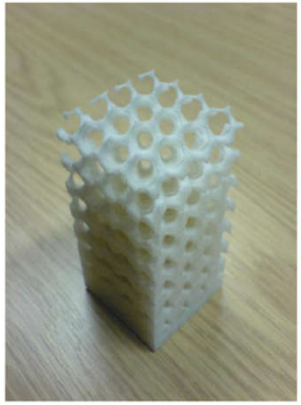
- 3D打印的中的stl模型是典型的三角网格表达，如何是否完整？

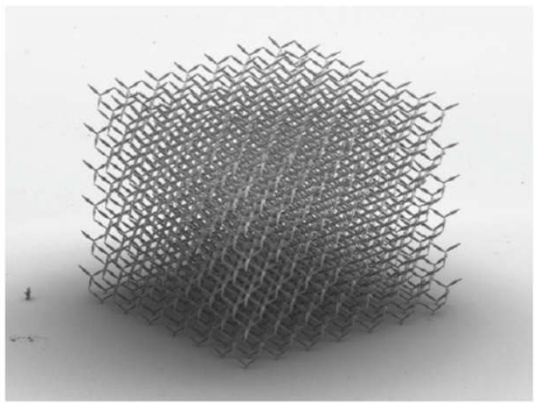
课程内容

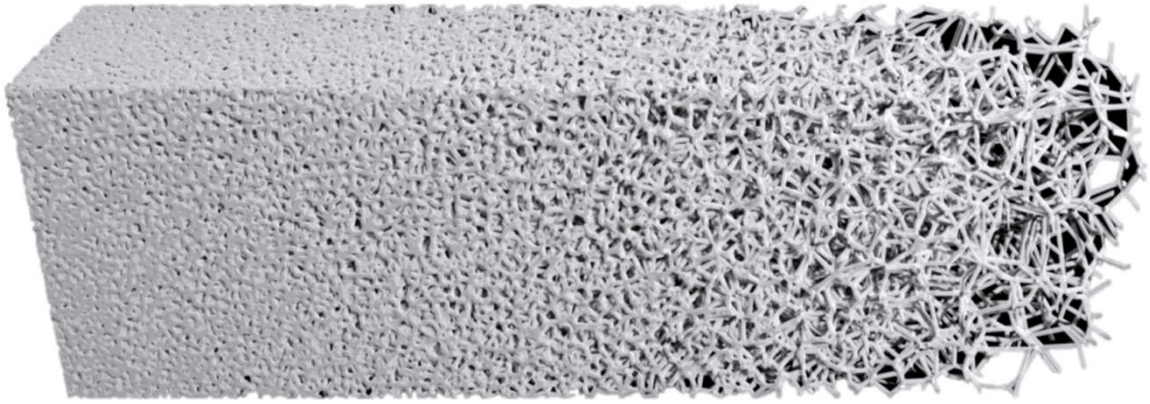
- 数字几何的概述
- 数字几何基本问题
- 几何模型表示方法
- 表面网格的处理方法
- 3D打印中的应用

3D打印的基础

- 三角网格模型







- STL文件
- SLC文件
- AMF文件
- 3MF文件

- STL是事实上的工业标准

3D打印流程

▪围绕三角网格的处理展开

- 输入模型
- 打印方向
- 支撑生成
- 切片
- 路径规划
- 面向工艺的G-Code生成

事实上的工业标准-STL

STL格式的优点

三角形是最简单的表示方法

支持设备对模型的快速分层

STL文件包含4大信息：实体名称、三角面片个数、三角形的法矢量、顶点坐标值

Ascii格式

- Solid ***name***
- facet normal n_i n_j n_k
- outer loop
- vertex $v1_x$ $v1_y$ $v1_z$
- vertex $v2_x$ $v2_y$ $v2_z$
- vertex $v3_x$ $v3_y$ $v3_z$
- endloop
- endfacet
- endsolid **name**

事实上的工业标准-STL

STL格式的格式

Binary格式

- UINT8[80] – 文件头，通常被忽略，可以存储额外的描述信息.
- UINT32 – Number of triangles

- foreach triangle
- REAL32[3] – Normal vector
- REAL32[3] – Vertex 1
- REAL32[3] – Vertex 2
- REAL32[3] – Vertex 3
- UINT16 – Attribute byte count
- end

Ascii格式

- Solid ***name***
- facet normal n_i n_j n_k
- outer loop
- vertex $v1_x$ $v1_y$ $v1_z$
- vertex $v2_x$ $v2_y$ $v2_z$
- vertex $v3_x$ $v3_y$ $v3_z$
- endloop
- endfacet
- endsolid **name**

事实上的工业标准-STL

▪三维**CAD**
Siemens NX
CATIA
CAXA
SolidWorks
Solid Edge
Autodesk Inventor
OpenSCAD(开源系统)

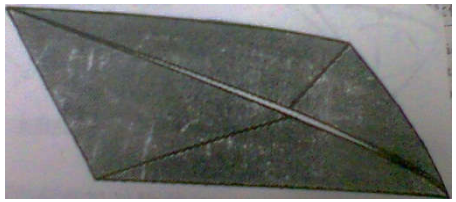
▪工业造型设计
Think Design
Alias Design
Rhino犀牛
Blender

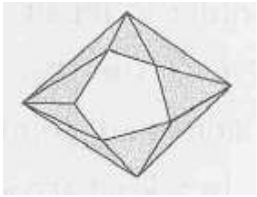
▪**CAM**系统
VERICUT
Cimatron E(中档CAM软件的代表,集成CAD)
CAXA制造工程师(本土)
MASTERCAM
Powermill(应用面较宽，鞋类行业几乎没存竞争对手)
HyperMILL(基于特征加工的系统)
Edgecam (数控加工系统)
SolidCAM(来自以色列，整合Solidworks的CAM系统)
ESPRIT(扩展性好)
Magics（比利时）

▪几乎所有CAD系统都支持STL格式的输出

STL的缺点

- 出现裂缝或空洞。





- 没有整体性的定义
- 不必要的冗余：对于不同三角面共用的点和线，重复存储，造成数据大量冗余，并且容易造成歧义。
- 不能表示颜色，多种材料等。
- 缺少必要的辅助信息。

SLC文件

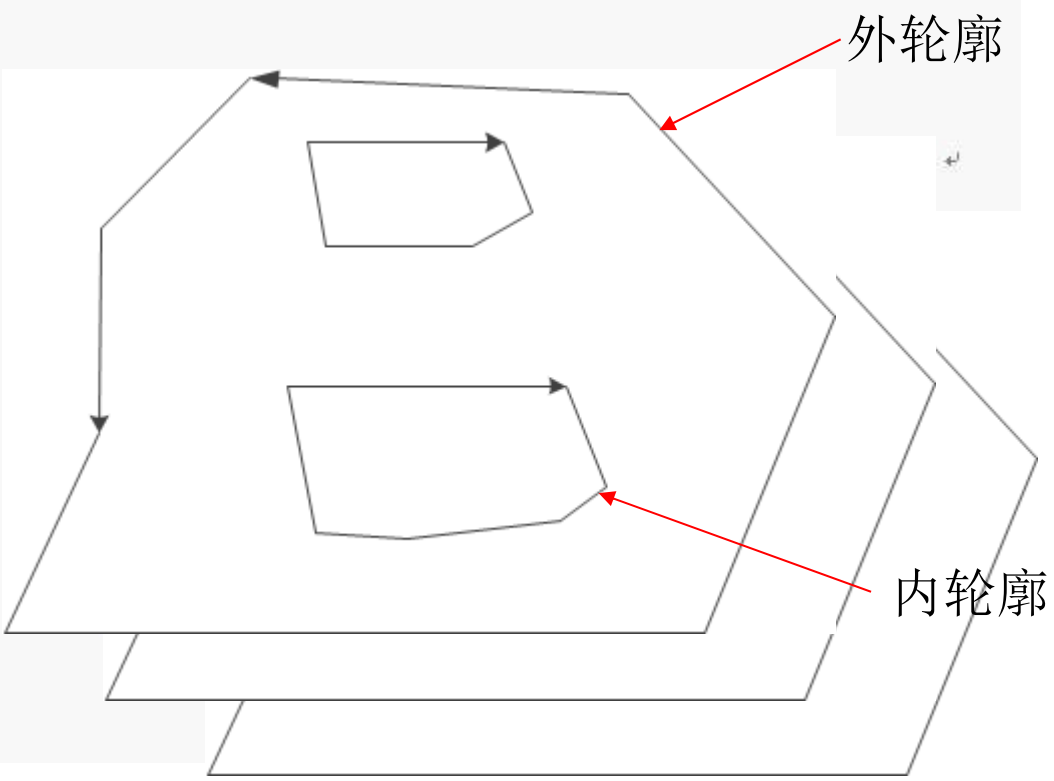
- SLC(StereoLithography Contour)文件是由2.5维的轮廓模型组成

```
SLICES name
FILE(Filename)
SLICES()
Z z dz                  #Z is a label, z=absolute z of a layer
x y                     #dz is layer thickness
x y                     #(x,y) is vertex
x y
x y
C                       #End of a contour
Z z dz                  #Begin a new layer
x y                     #New contour on a new level
x y                     #(x,y) is vertex
…
C
```

SLC文件

- SLC文件是由2.5维的轮廓模型组成。

```
SLICES name
FILE(Filename)
SLICES()
Z z dz
x y
x y
x y
x y
C
Z z dz
x y
x y
...
C
```



SLC文件

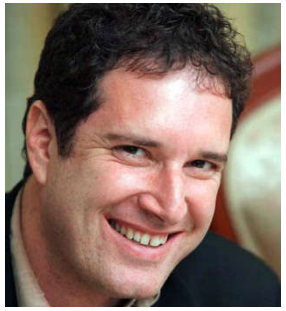
- SLC文件是由2.5维的轮廓模型组成。

-直接在快速成型机上使用

-缺少信息，使用SLC文件很难生成支撑结构

工业标准STL2.0-AMF格式

- AMF（Additive Manufacturing File Format）
- 2010年由Hod Lipson提出
- 2012年由成为ASTM subcommittee的RP文件新标准



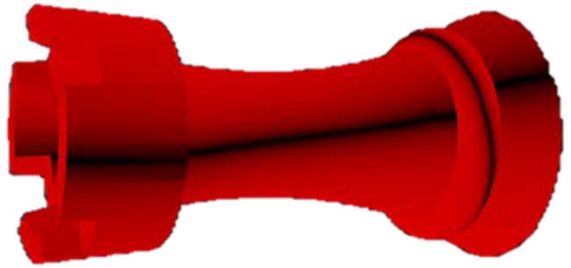
Hod Lipson

目的？

AMF文件-STL 2.0

- 基于XML文件的格式，具有简单，灵活，可扩展。

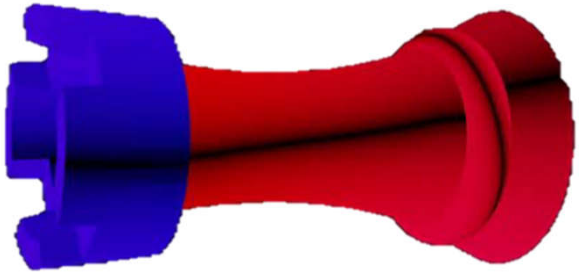
```
<?xml version="1.0"?>   //XML版本
<AMF> //AMF文件
    <Object PrintID = "0" units = "mm">   //定义一个模型，单位mm
        <Mesh>//三维网格划分
            <Vertices>
                <Vertex VertexID="0"> //第一个顶点ID
                <VertexLocation x="0" y="1.332" z="3.715"/> //第一个顶点坐标
                </Vertex>
                <Vertex VertexID="1"> //第二个顶点ID
                    <VertexLocation x="0" y="1.269" z="3.715"/>
                </Vertex>
            ...
            </Vertices>  //列出了所有顶点
            <Region FillMaterialID = "0">
                <Triangle V1 = "0" V2 = "1" V3 = "3"/>
                <Triangle V1 = "0" V2 = "1" V3 = "4"/>
                ...
            </Region>
        </Mesh>
    </Object>
</AMF>
```



AMF文件-STL 2.0

- 基于XML文件的格式，可以表达不同的材料。

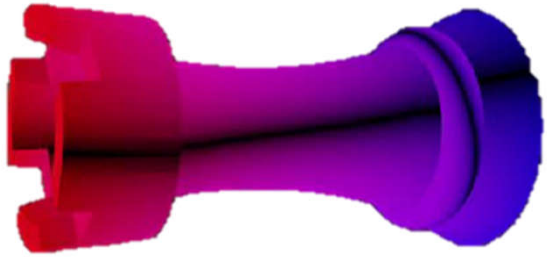
```
<?xml version="1.0"?>
<AMF>
    <Palette> //" 调和板" 列出多种材料
        <Material MaterialID = "0"> //第一种材料ID
            <Name>StiffMaterial</Name> //第一种材料名称：刚性材料
        </Material>
        <Material MaterialID = "1"> //第二种材料ID
            <Name>FlexibleMaterial</Name> //第二种材料名称：柔性材料
        </Material>
    </Palette>

    <Object PrintID = "0" units = "mm">
        <Mesh>
            <Vertices>
            ...
            </Vertices>

            <Region FillMaterialID = "0"> //第一种材料的分布区域
            ...
            </Region>
            <Region FillMaterialID = "1"> //第一种材料的分布区域
                <Triangle V1 = "5" V2 = "6" V3 = "7"/>
                <Triangle V1 = "5" V2 = "7" V3 = "9"/>
                ...
            </Region>
        </Mesh>
    </Object>
</AMF>
```



AMF文件-STL 2.0

- 基于XML文件的格式，可以表达渐变的颜色。

```
<?xml version="1.0"?>
    <AMF>
        <Palette>
            <Material MaterialID = "0"> //第一种材料
                <Name>StiffMaterial</Name>
            </Material>
            <Material MaterialID = "1"> //第二种材料
                <Name>FlexibleMaterial</Name>
            </Material>
            <Material MaterialID = "2"> //第三种材料
                <Name>GradientMaterial</Name>
                <Equation UseMaterialID = "0">0.30*X</Equation>
                <Equation UseMaterialID = "1">0.30*(1-X)</Equation>
                //模型材料的性能按照关于x的函数分布
            </Material>
        </Palette>
        <Object PrintID = "0" units = "mm">
        ...
        </Object>
</AMF>
```



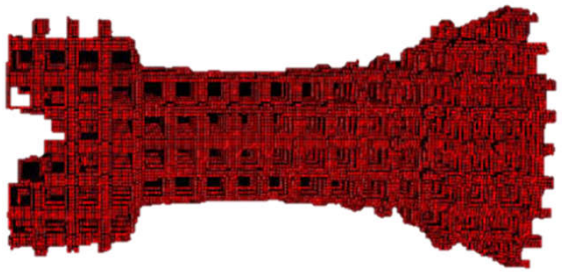
调色板

实体定义

AMF文件-STL 2.0

- 基于XML文件的格式，可以表达模型的内部特征。

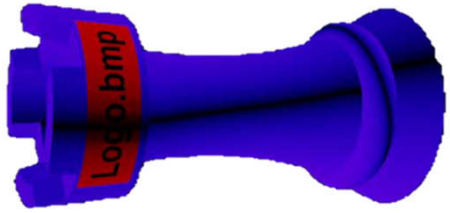
```
<?xml version="1.0"?>
    <AMF>
        <Palette>
            <Material MaterialID = "0">
                <Name>StiffMaterial</Name>
            </Material>
            <Material MaterialID = "1">
                <Name>FlexibleMaterial</Name>
            </Material>
            <Material MaterialID = "2">
                <Name>MesoStructureMaterial</Name>
                <Lattice> // 将模型划分为很多小格
                ...
                </Lattice>
                <Structure> //三维像素体特征、分布
                    <X_Voxels>10</X_Voxels> // 三维像素体的大小
                    <Y_Voxels>10</Y_Voxels>
                    <Z_Voxels>10</Z_Voxels>
                    <Data Layer = "1">0000110000...</Data>  //三维像素体的分布
                    <Data Layer = "2">0001111000...</Data>
                    ...
                </Structure>
            </Material>
```



AMF文件-STL 2.0

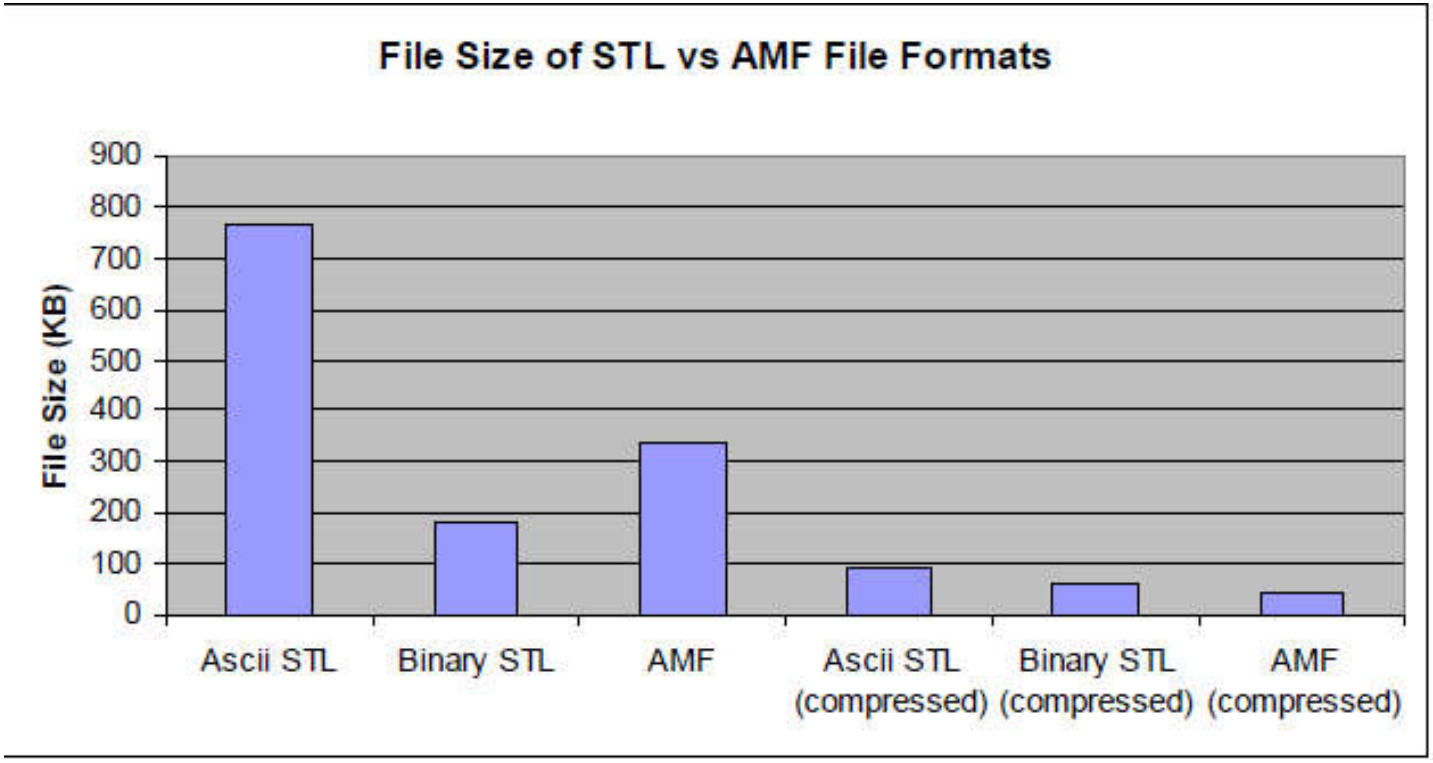
- 基于XML文件的格式，可以表达模型不同顶点的颜色。

```
<?xml version="1.0"?>
<AMF>
    <Object PrintID = "0" units = "mm">
        <Mesh>
            <ColorFile MapID="0"> // 定义颜色分布的ID
                <File>Logo.bmp</File>
            </ColorFile>
            <Vertices>
                <Vertex VertexID="0">
                    <VertexLocation x="0" y="1.332" z="3.715"/>
                    <VertexMap UseMapID="0" MapXPixel="65" MapYPixel="87"/>
                </Vertex>
                <Vertex VertexID="1">
                    <VertexLocation x="0" y="1.269" z="3.715"/>
                    <VertexMap UseMapID="0" MapXPixel="64" MapYPixel="87"/>
                </Vertex> //把空间中的点与三维像素联系起来
                    <Vertex VertexID="2">
                        <VertexLocation x="0" y="1.310" z="3.587"/>
                        <VertexMap UseMapID="0" MapXPixel="32" MapYPixel="10"/>
                </Vertex>
                ...
                </Vertices>

                <Region FillMaterialID = "0">
                    <Color R = "0" G = "0" B = "0.5"/> //赋予不同区域不同的RGB值
                    <Triangle V1 = "0" V2 = "1" V3 = "2"/>
                    <Triangle V1 = "0" V2 = "1" V3 = "4"/>
                    ...
                </Region>
            </Mesh>
    </Object>
</AMF>
```



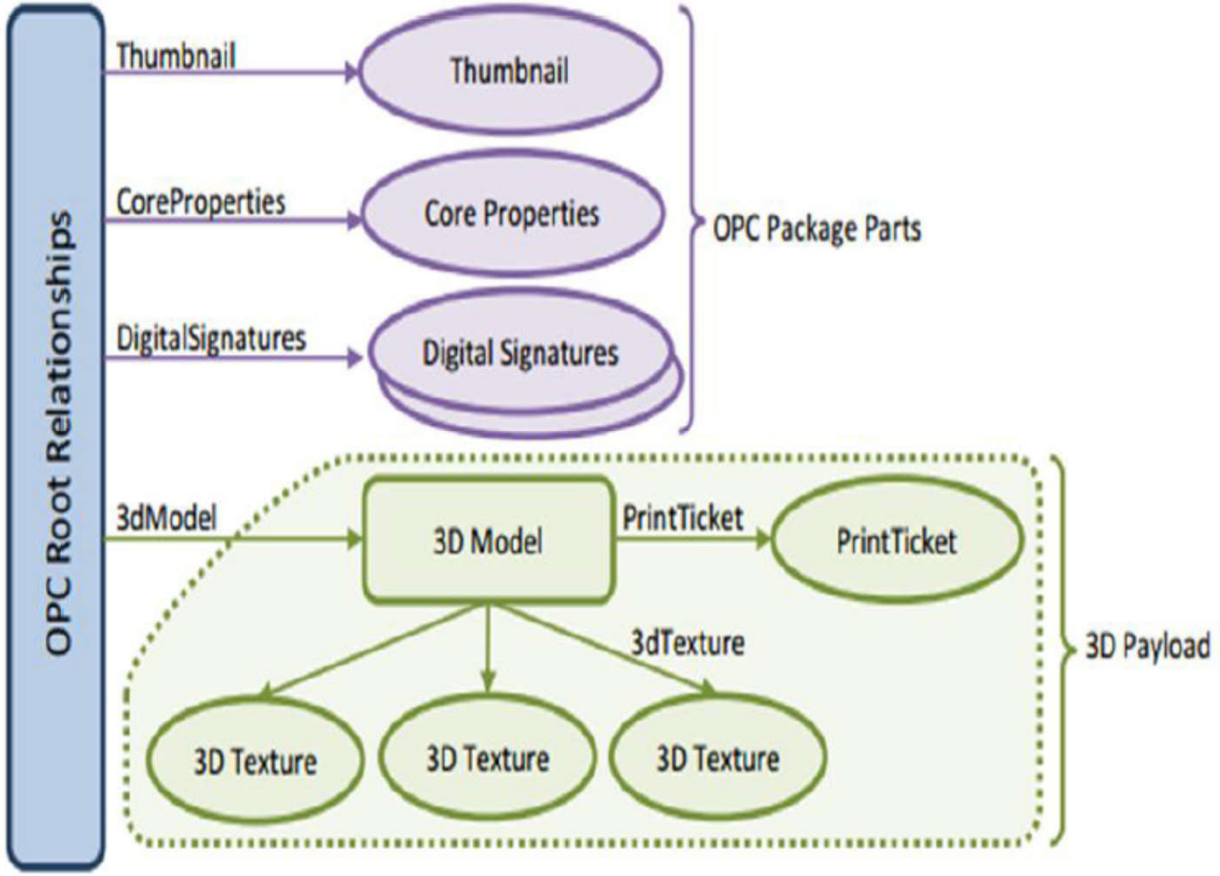
AMF文件-STL 2.0

- 基于XML文件的格式，可以获得更小的压缩格式。



3MF文件-STL 3.0?

- Windows集成3D打印格式（http://www.3mf.io）。



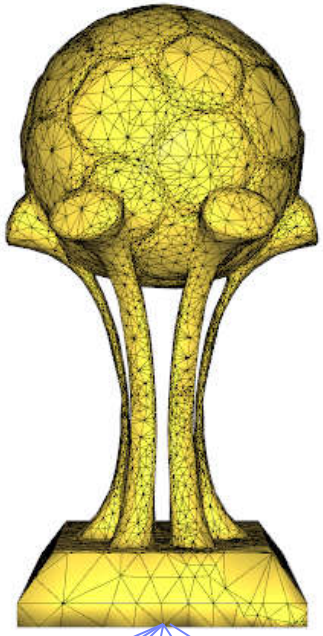
<model> is the root element of the 3MF document

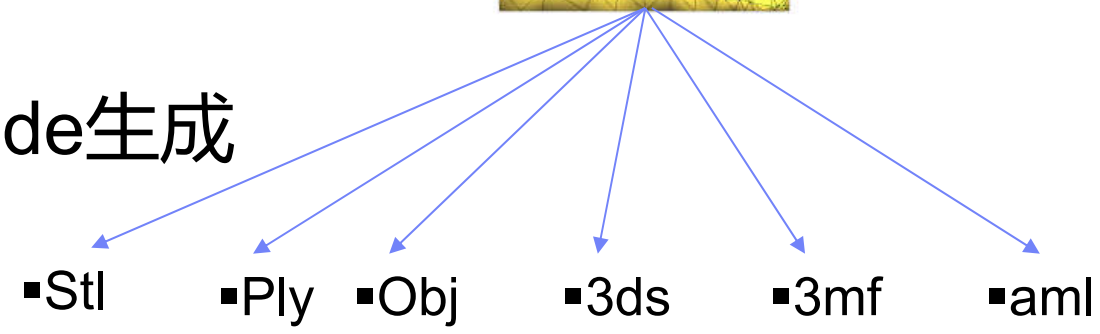
```
<model>
 :
 :
 :
 :
</model>
```

目标：在不同平台和服务商之间传递模型的所有信息

3D打印流程

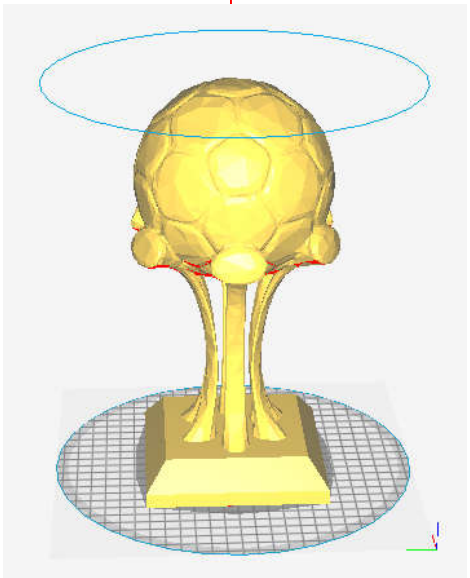
- 输入模型
- 打印方向
- 支撑生成
- 切片
- 路径规划
- 面向工艺的G-Code生成

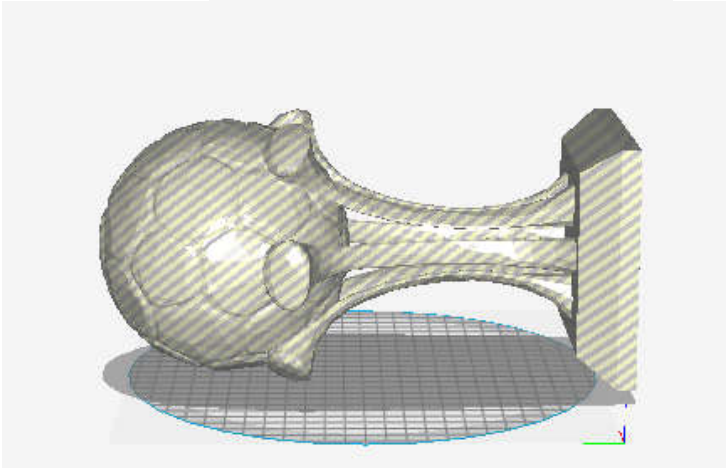




3D打印流程

- 输入模型
- 打印方向
- 支撑生成
- 切片
- 路径规划
- 面向工艺的G-Code生成

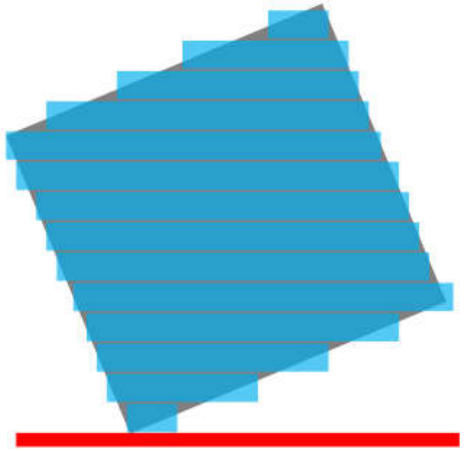




打印方向

- 考虑因素
  - 表面精度
  - 打印时间
  - 支撑体积
  - 支撑面积
  - 力学特性

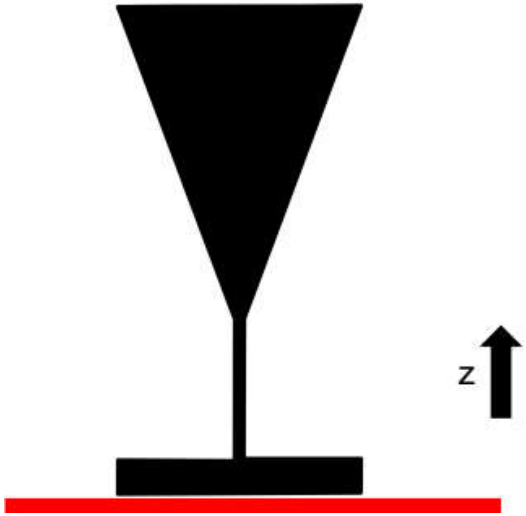
- 需要对工件综合分析



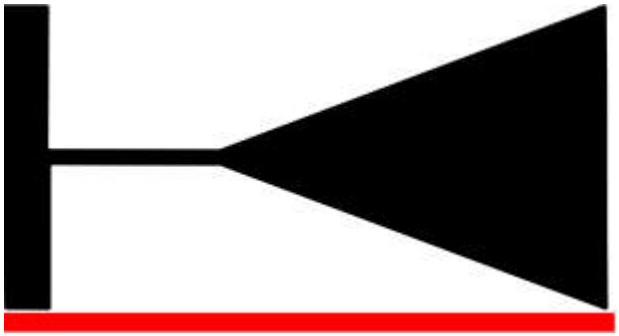
▪影响表面精度

打印方向

- 考虑因素
  - 表面精度
  - 打印时间
  - 支撑体积
  - 支撑面积
  - 力学特性



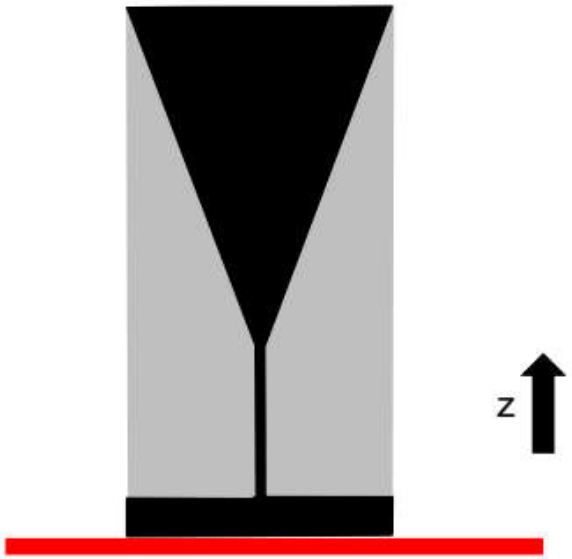
▪更长的时间



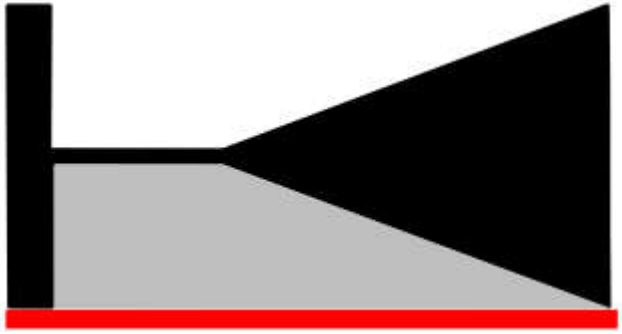
▪更短的时间

打印方向

- 考虑因素
  - 表面精度
  - 打印时间
  - 支撑体积
  - 支撑面积
  - 力学特性



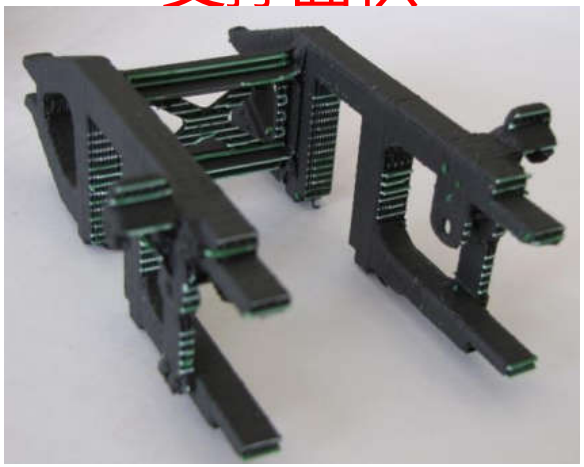
▪更大的支撑体积

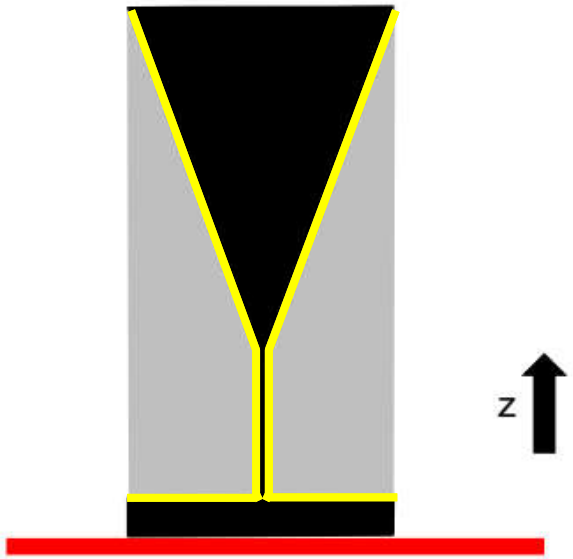


▪更少的支撑体积

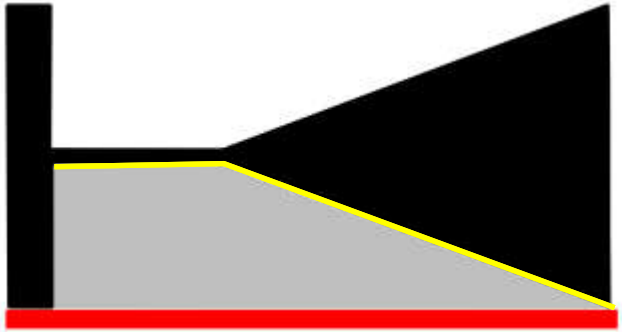
打印方向

- 考虑因素
  - 表面精度
  - 打印时间
  - 支撑体积
  - 支撑面积





▪更大的支撑面积



▪更少的支撑面积

打印方向

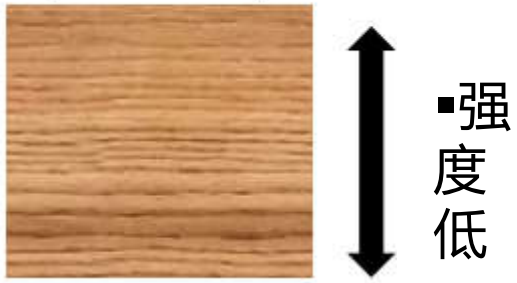
- 考虑因素
  - 表面精度
  - 打印时间
  - 支撑体积
  - 支撑面积
  - 力学特性

▪例如:

▪ * 在不同方向上具有不同的强度

▪ * 在纤维路径上具有更好的强度/刚度

▪强度高



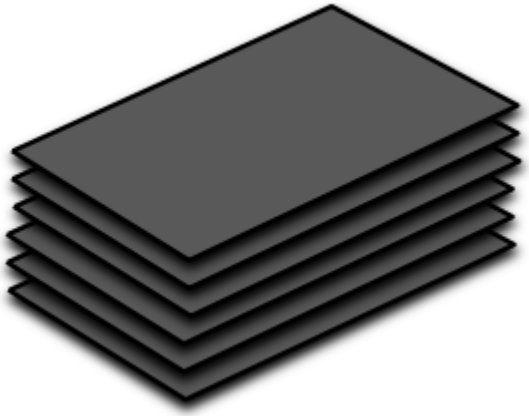
▪强度低

▪轴向截面

▪径向截面

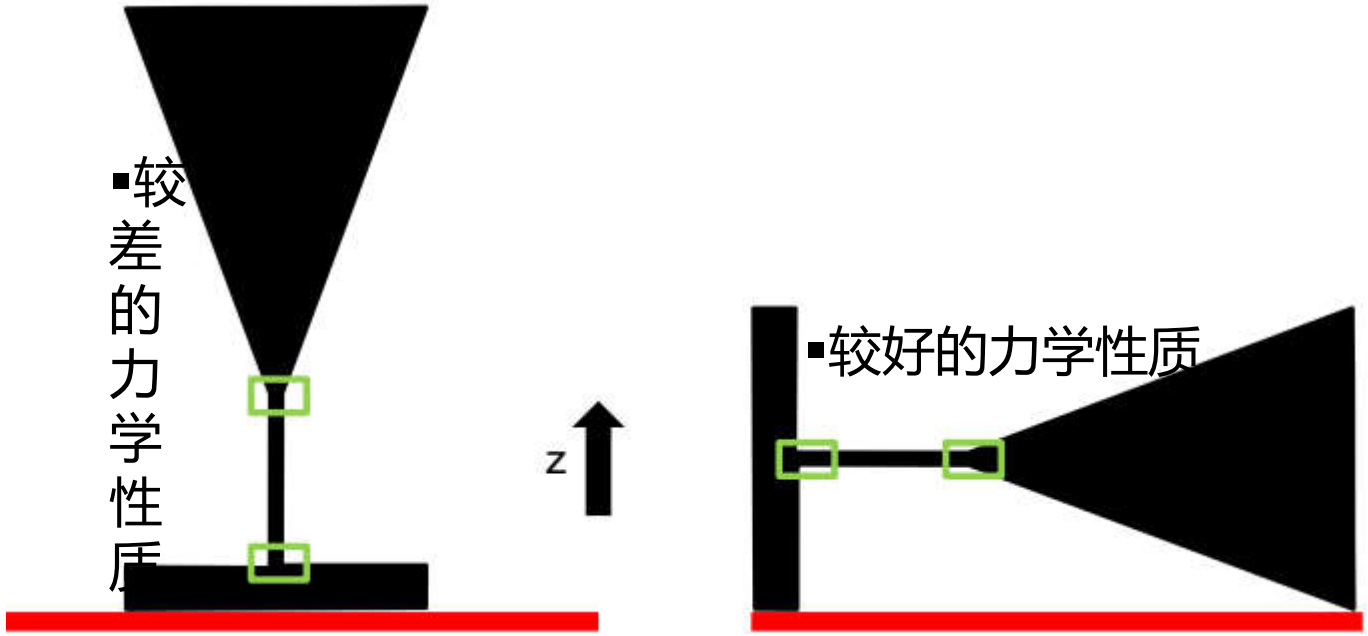
打印方向

- 力学特性
  - 在没考虑拓扑优化的情况下：
    - * 在同层次表现出各向同性，但在不同层次是不一致的

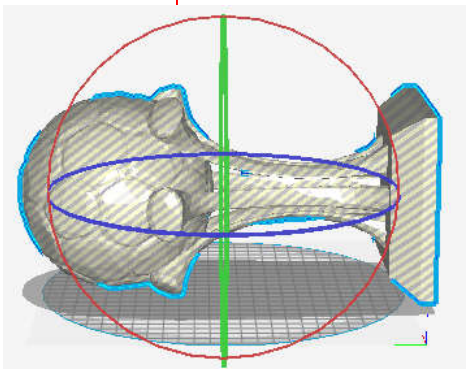


打印方向

- 力学特性
  - 一般情况下：
    - * 在同层强度/刚度较好，在层间较差



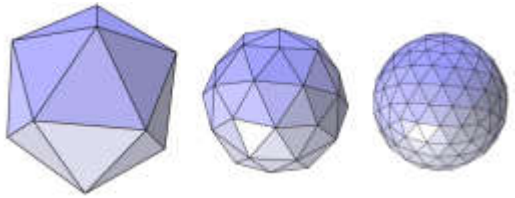
打印方向



- 确定打印方向的算法
  - 手工
    - eg.在Cura等系统中由用户手动指定
  - 半自动方法
    - 用户将工件放置到工作空间
    - 系统计算打印时间、支撑面积、支撑体积和力学特性
  - 全自动方法
    - 建立目标函数(打印时间、支撑面积、支撑体积和力学特性)，优化求解

打印方向确定

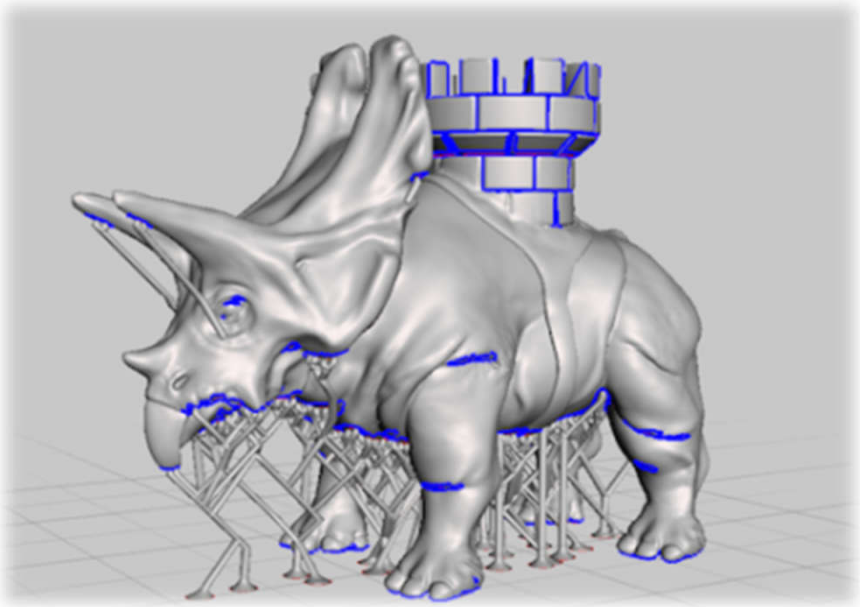
- 简单算法：穷举搜索



  - 测试一组设定的方向
    - Eg. 将工件进行20面体细分，以此这些方向进行搜索
  - 针对每个方向根据目标函数计算
    - 打印时间
    - 支撑体积
    - 支撑面积
    - 力学强度
  - 选取是目标函数最小的方向
    - 可选单目标函数
    - 也可选多目标函数，但确定各个目标的权值需考虑实际意义

3D打印流程

- 输入模型
- 打印方向
- 支撑生成
- 切片
- 路径规划
- 面向工艺的G-Code生成



打印支撑

- 不需要支撑
  - SLS，DMLS，LOM…
- 需要支撑
  - FDM，SLA，各类光固化方法
- 不同的目标
  - 避免树脂硬化时的翘曲
  - 简化支撑
  - 保证稳定性
  - 避免过度收缩
  - 支撑悬空或倾斜的结构

打印支撑设计

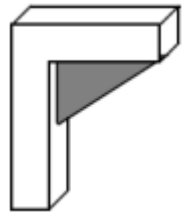
- 基于一些观察得到的规则
  - 例如,对FDM和SLA具有不同的规则
- 基于制造方法

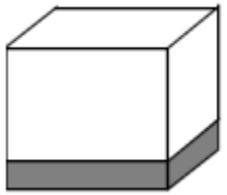
▪SLA

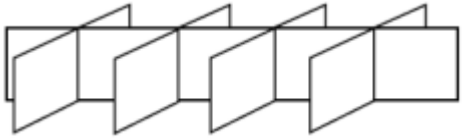
▪FDM（45°支撑角）

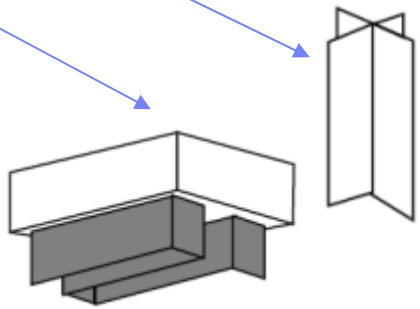
打印支撑类型

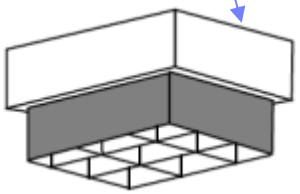
- 节点支撑
- 基础支撑
- 网状支撑
- 列状支撑
- Z字形支撑
- 瓶口形支撑





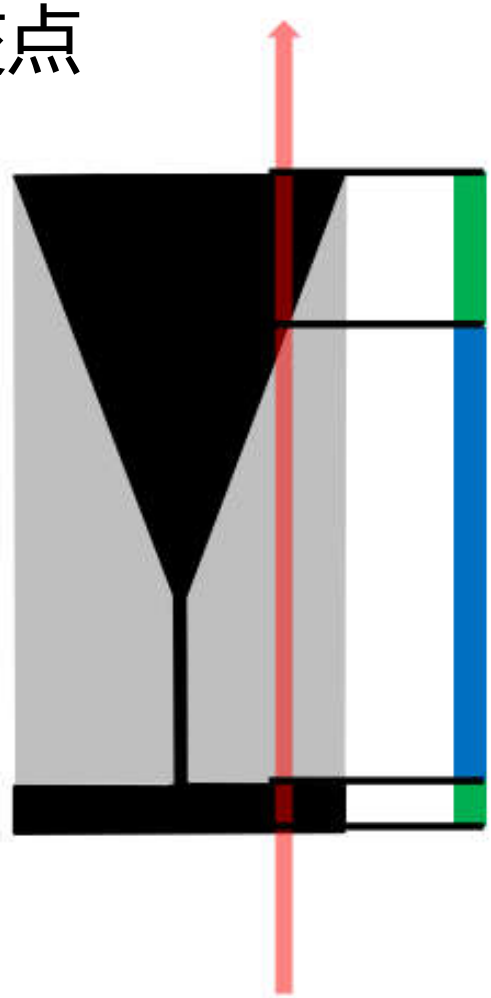






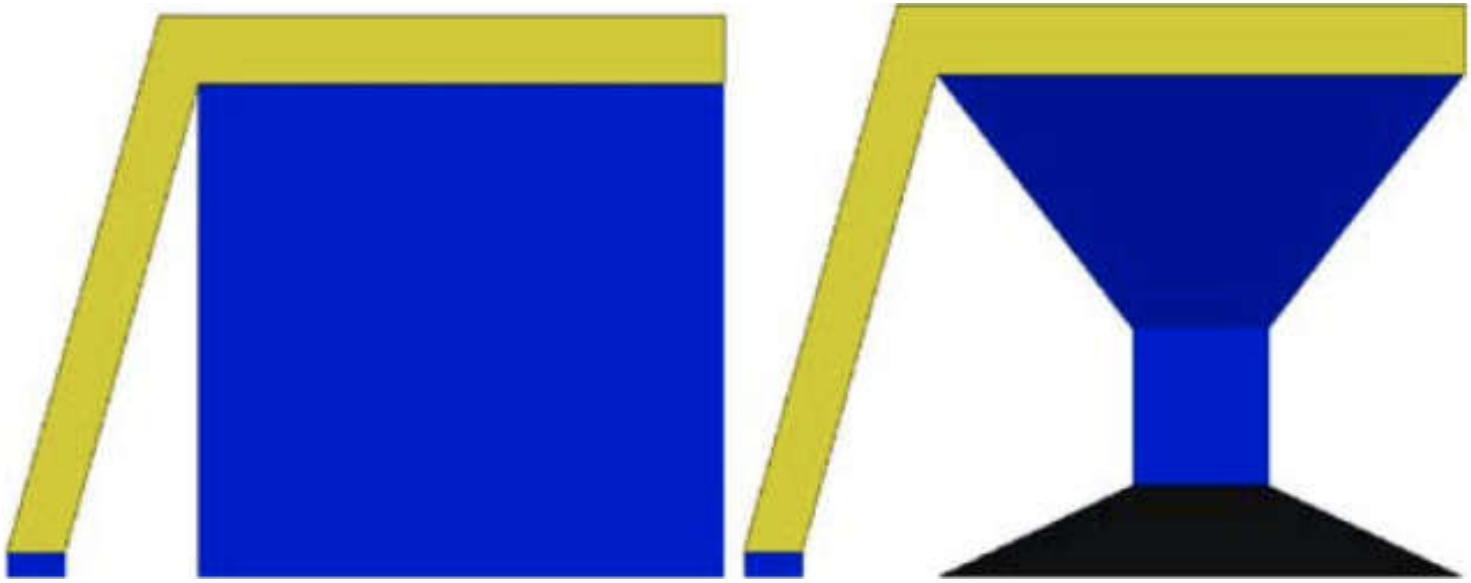
简单的支撑生成算法

- 在z轴使用光线追踪法计算所有交点
- 对交点排序，以确定模型内和外
- 在所有出模型的节点和它下一个节点间添加支撑



复杂的支撑生成算法

- 最小化支撑体积（节省材料）

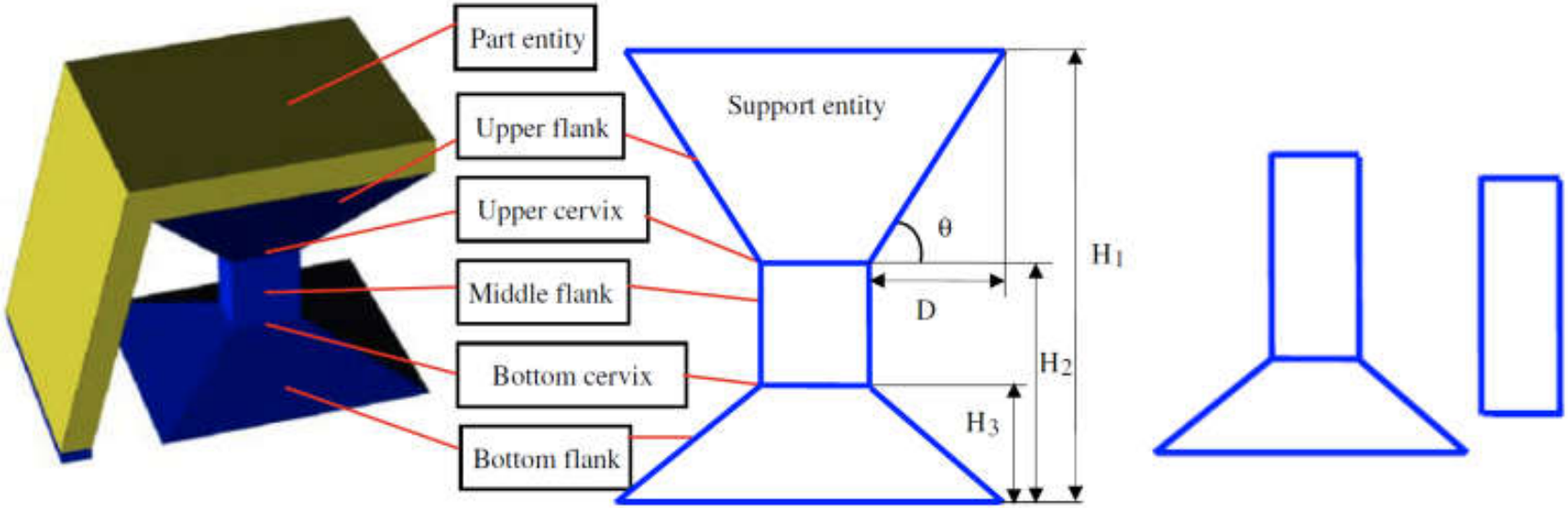


▪直接支撑

▪倾斜的支撑

复杂的支撑生成算法

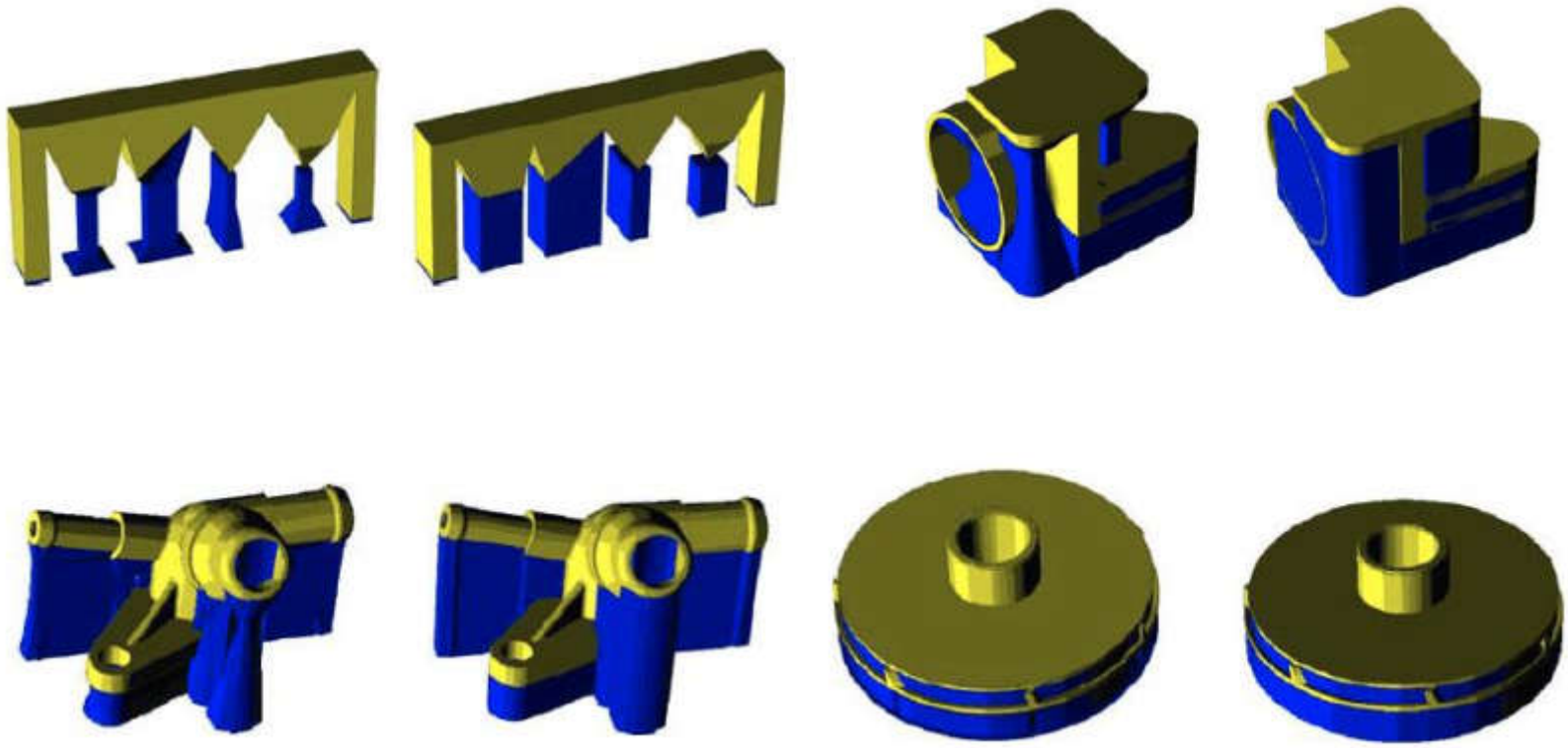
- 最小化支撑体积（节省材料）



- 预定义结构
- 参数优化

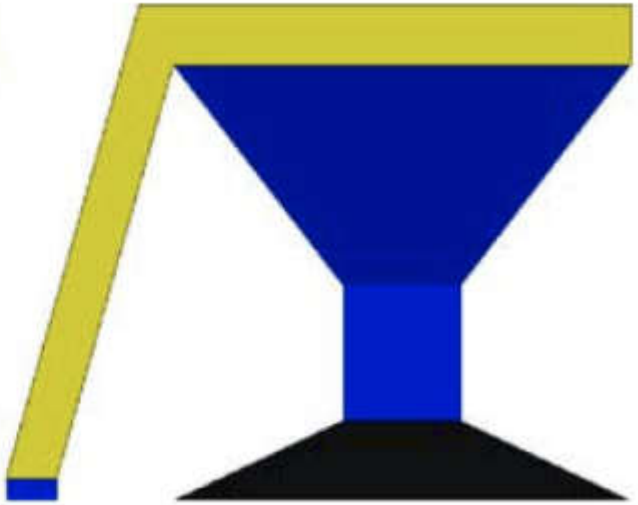
复杂的支撑生成实例

- 优化与未优化



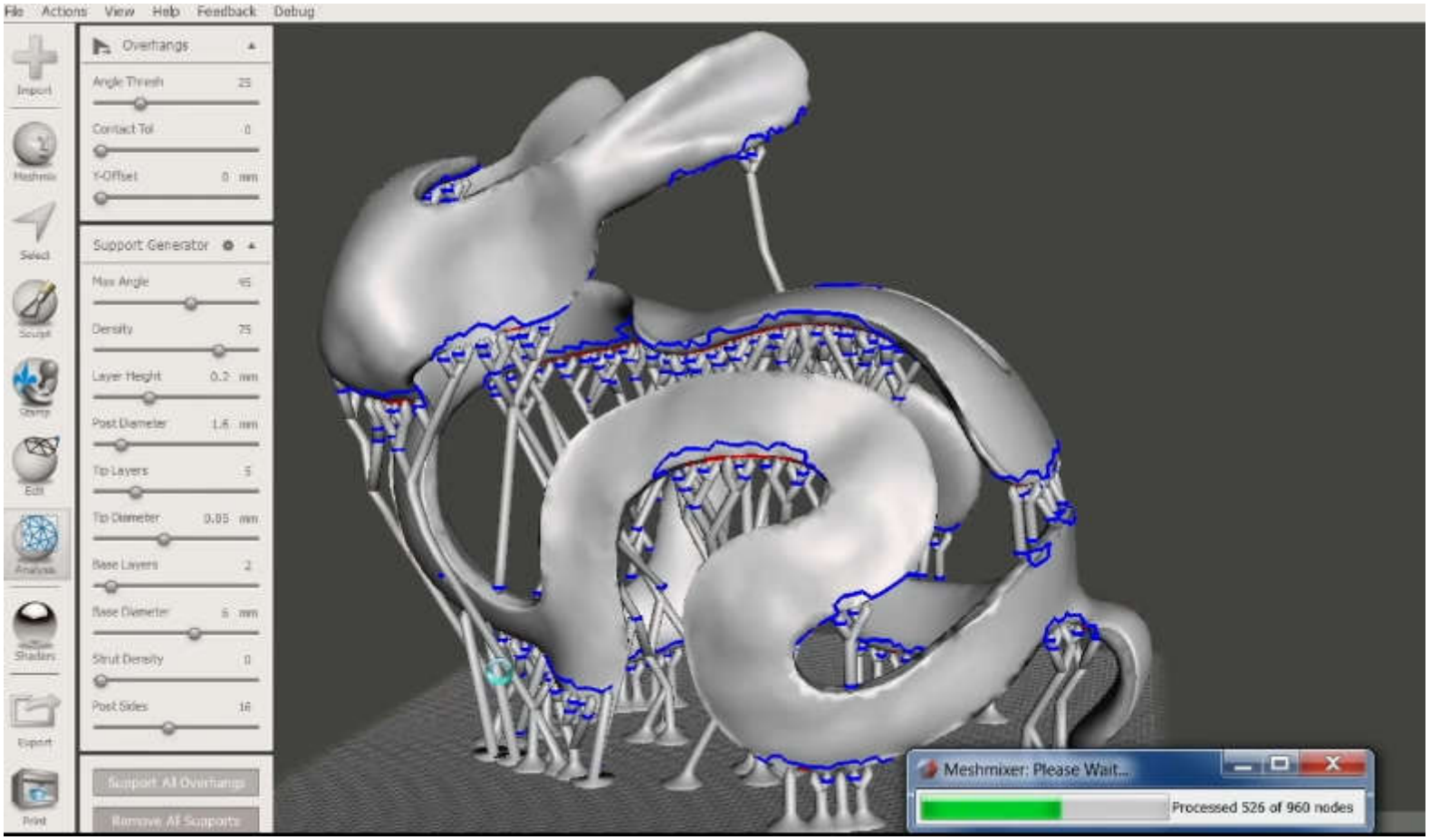
复杂的支撑生成实例

- 制造实例



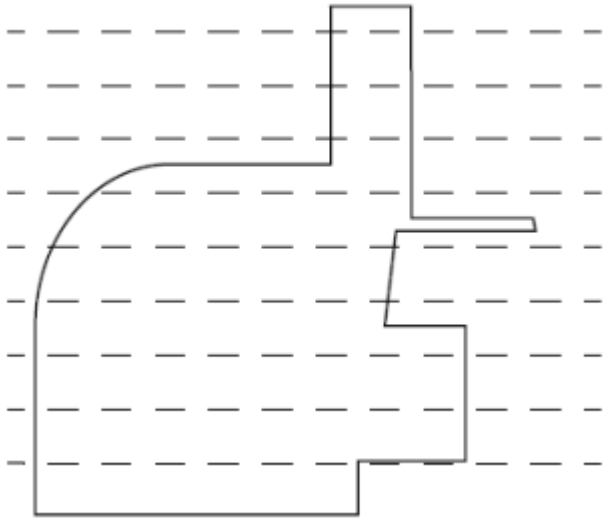
复杂的支撑生成实例

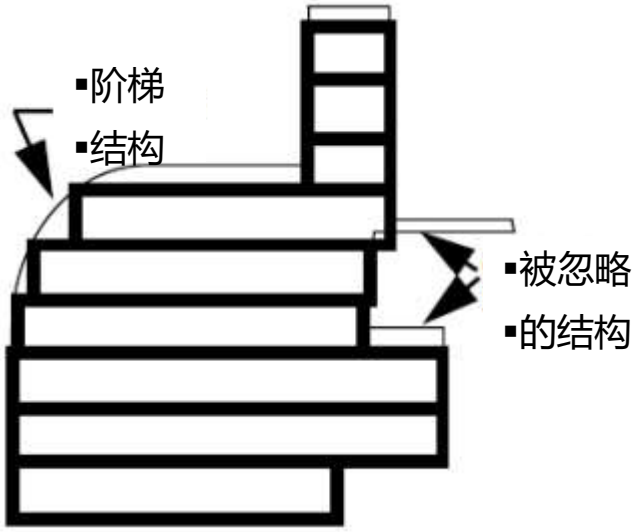
- Autodesk Meshmixer



3D打印流程

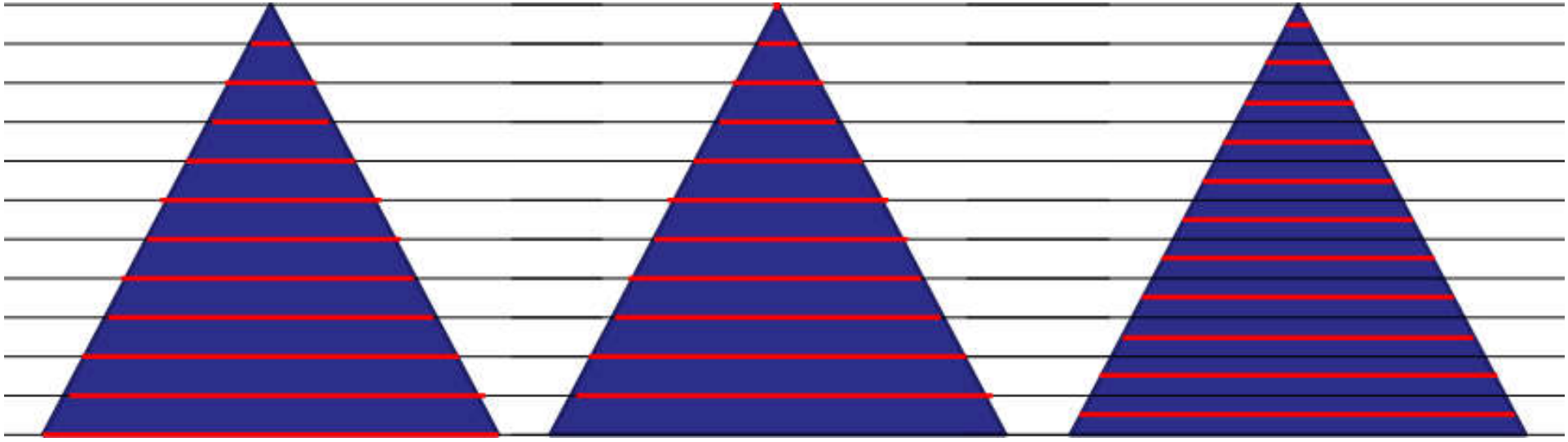
- 输入模型
- 打印方向
- 支撑生成
- 切片
- 路径规划
- 面向工艺的G-Code生成





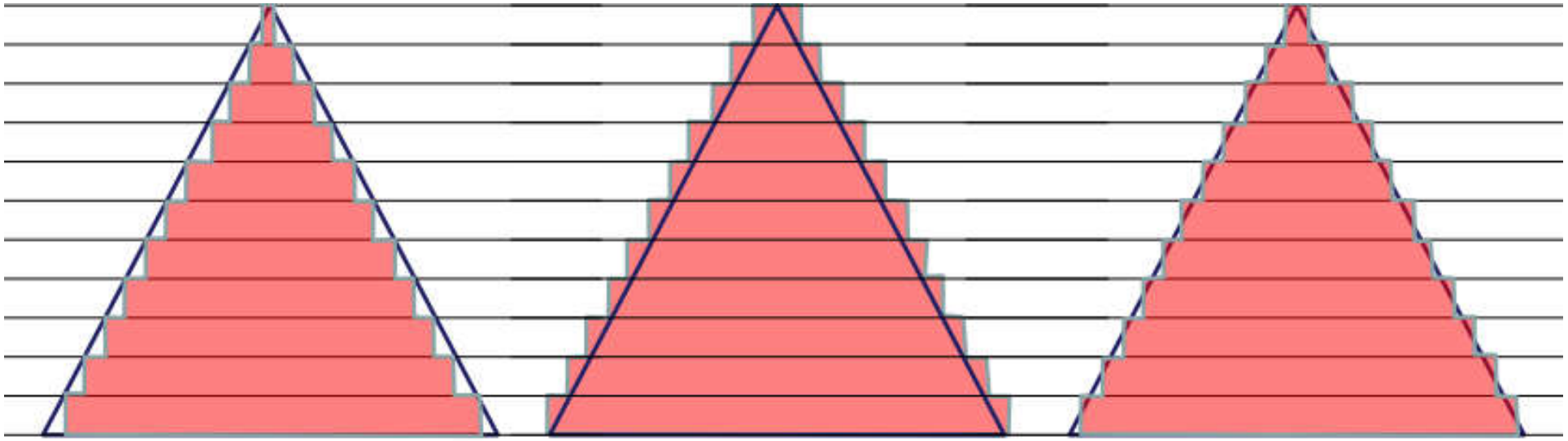
切片中的问题

- 给定每个层次的厚度，截面选在什么地方？
  - 底面
  - 中分面
  - 顶面



切片中的问题

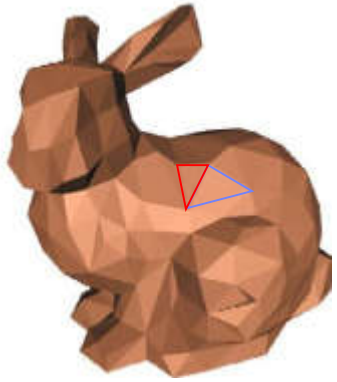
- 边界的定义
  - 在模型内部?
  - 在模型外部?
  - 在中间?



STL切片：体素法

- STL文件中并没有给出连续的结构

- 体素化三角网格
  - 针对每个体素判断内外
  - 提取每层的轮廓



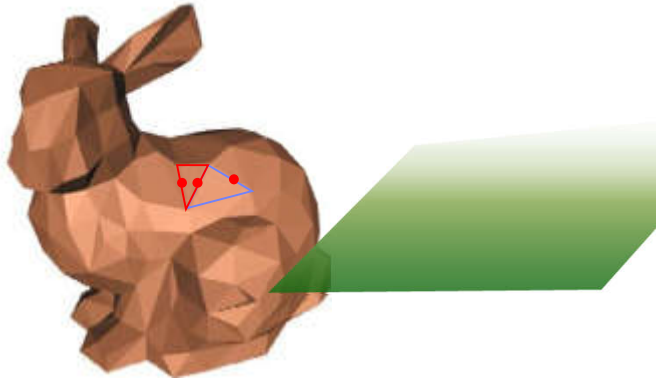
```
solid Wheel
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 2.913194e+002 7.026579e+001
         vertex 7.095000e+001 2.914028e+002 7.636772e+001
         vertex 7.095000e+001 3.106206e+002 8.149973e+001
      endloop
   endfacet
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 3.106206e+002 8.149973e+001
         vertex 7.095000e+001 2.914028e+002 7.636772e+001
         vertex 7.095000e+001 2.882984e+002 1.048139e+002
      endloop
   endfacet
   facet normal -1.000000e+000 0.000000e+000 0.000000e+000
      outer loop
         vertex 7.095000e+001 3.106206e+002 8.149973e+001
         vertex 7.095000e+001 2.882984e+002 1.048139e+002
```

STL切片

- STL文件中并没有给出连续的结构

算法：

- For each z plan
  - for each triangle
    - > 判断是否与z相交
      - > 如果相交，存储交线
  - 连接交线，形成轮廓



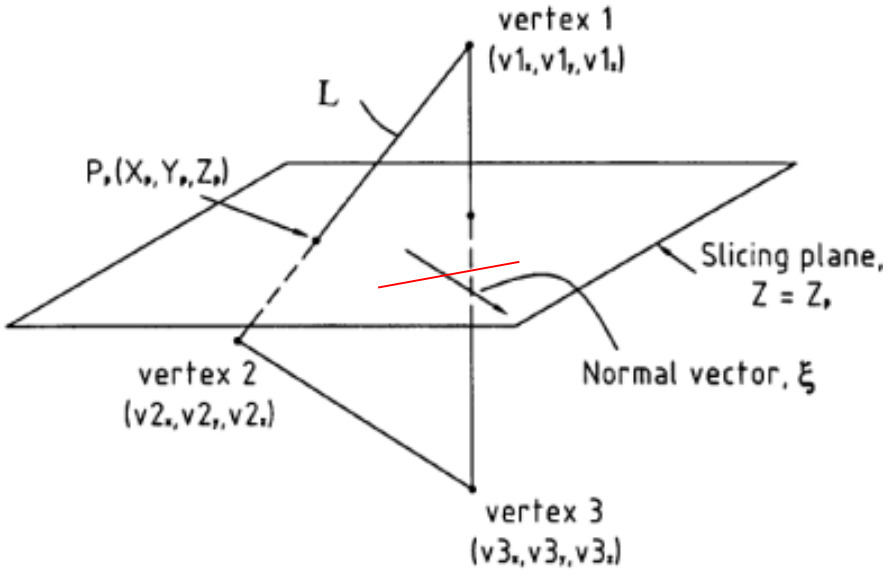
```
solid Wheel
  facet normal -1.000000e+000 0.000000e+000 0.000000e+000
    outer loop
      vertex 7.095000e+001 2.913194e+002 7.026579e+001
      vertex 7.095000e+001 2.914028e+002 7.636772e+001
      vertex 7.095000e+001 3.106206e+002 8.149973e+001
    endloop
  endfacet
  facet normal -1.000000e+000 0.000000e+000 0.000000e+000
    outer loop
      vertex 7.095000e+001 3.106206e+002 8.149973e+001
      vertex 7.095000e+001 2.914028e+002 7.636772e+001
      vertex 7.095000e+001 2.882984e+002 1.048139e+002
    endloop
  endfacet
  facet normal -1.000000e+000 0.000000e+000 0.000000e+000
    outer loop
      vertex 7.095000e+001 3.106206e+002 8.149973e+001
      vertex 7.095000e+001 2.882984e+002 1.048139e+002
```

STL切片

- STL文件中并没有给出连续的结构

算法:

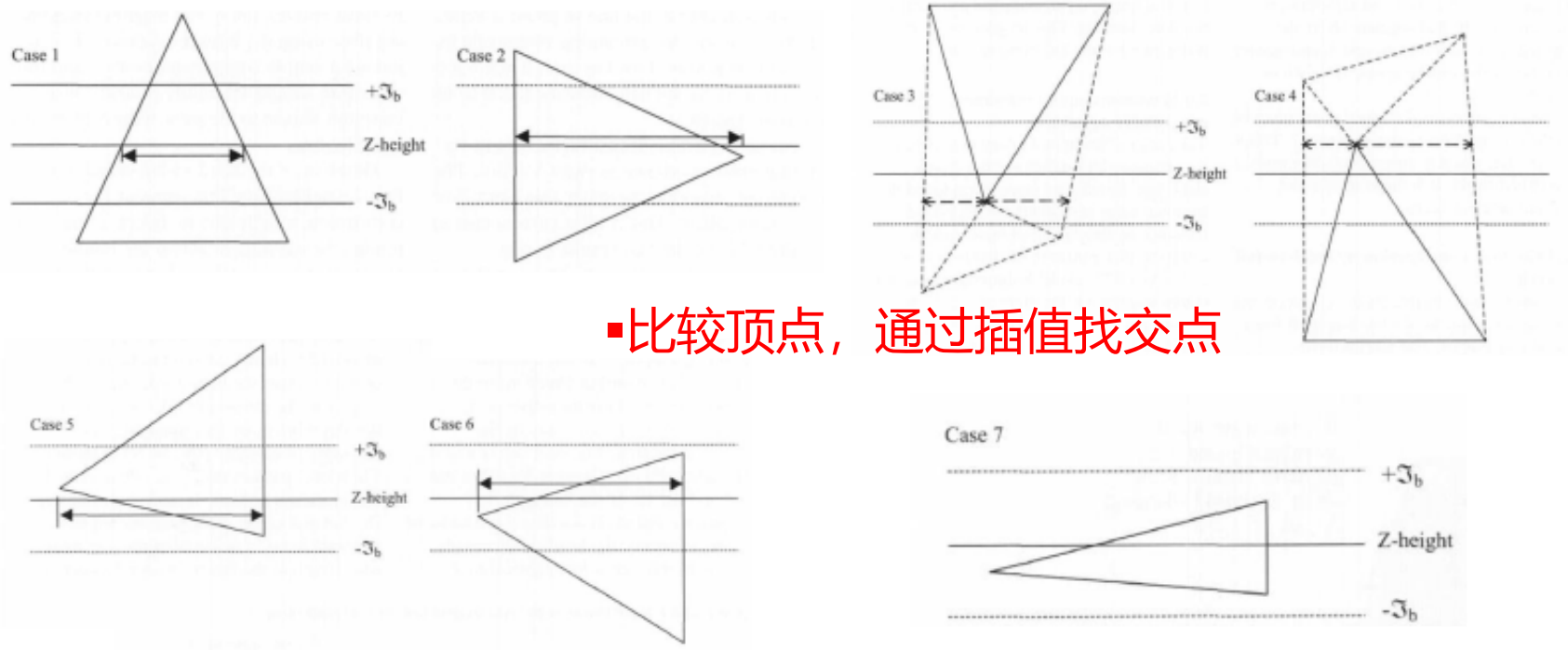
- For each z plan
  - for each triangle
    - > 判断是否与z相交
      - > 如果相交，存储交线
  - 连接交线，形成轮廓



From Choi and Kwok, 2002

STL切片

- 数值计算的问题
  - 为每一个三角形计算z平面交点代价太高
  - 有一些技巧，eg. 比较顶点



- 比较顶点，通过插值找交点

From Choi and Kwok, 2002

STL切片

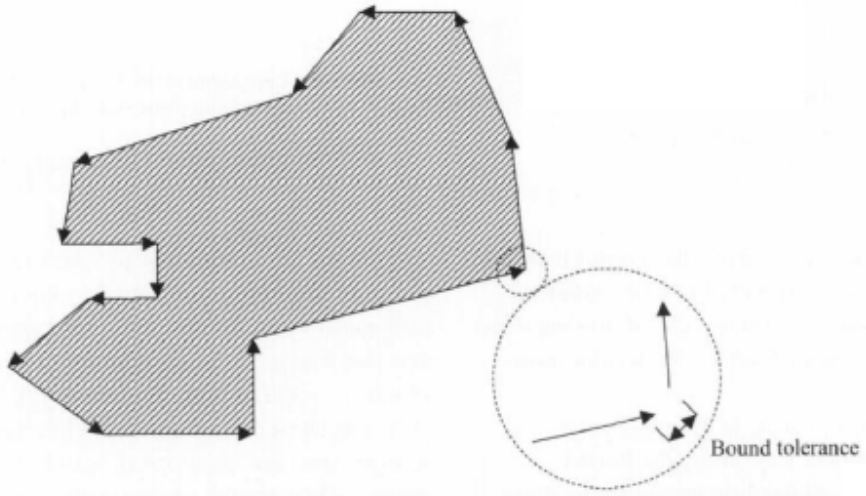
- 算法:
  - For each z plan
  - * for each triangle
  - > 判断是否与z相交
  - > 如果相交，存储交线
  - * 连接交线，形成轮廓

▪轮廓连接算法

▪问题:

1.产生虚假的线段

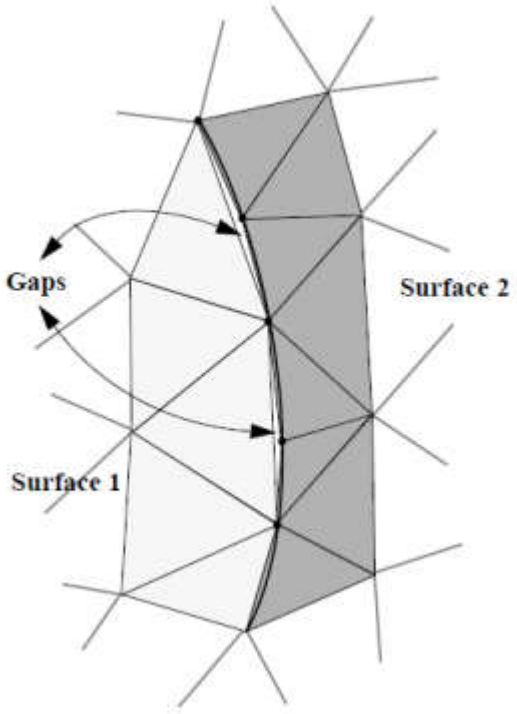
2.有些线段缺失



From Choi and Kwok, 2002

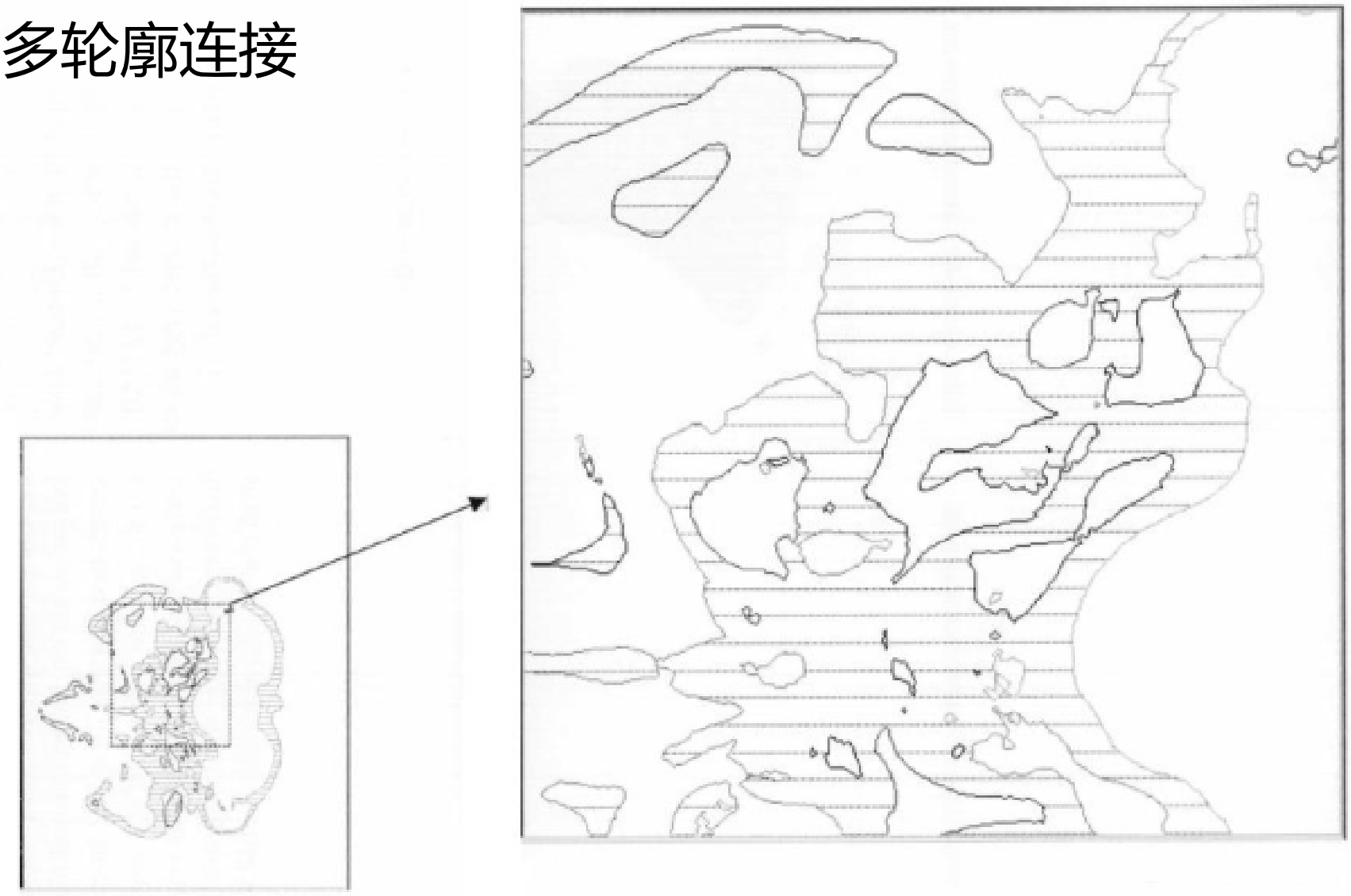
STL切片

- 三角网格本身的质量问题：
  - 裂缝



STL切片结果

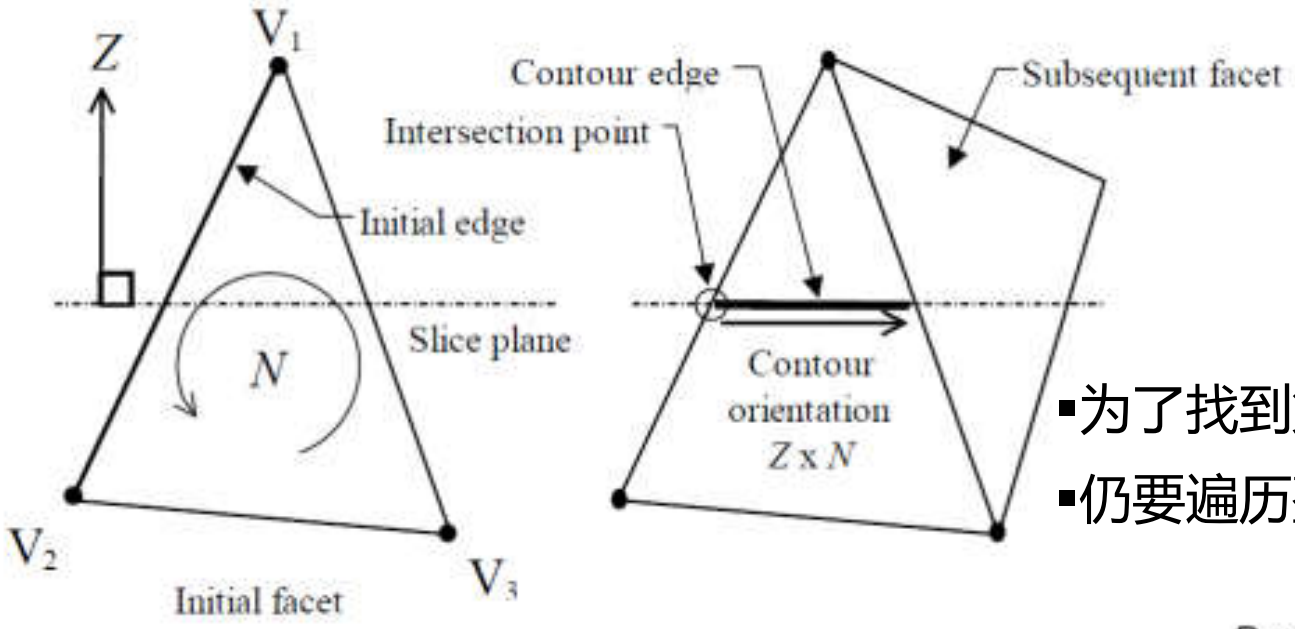
- 多轮廓连接



From Choi and Kwok. 2002

更有效的切片

- 利用拓扑信息:
  - 预先保存相邻三角面的信息
  - 搜索第一个相交的三角面
  - 再到当前三角面邻域去搜索其它相交三角面

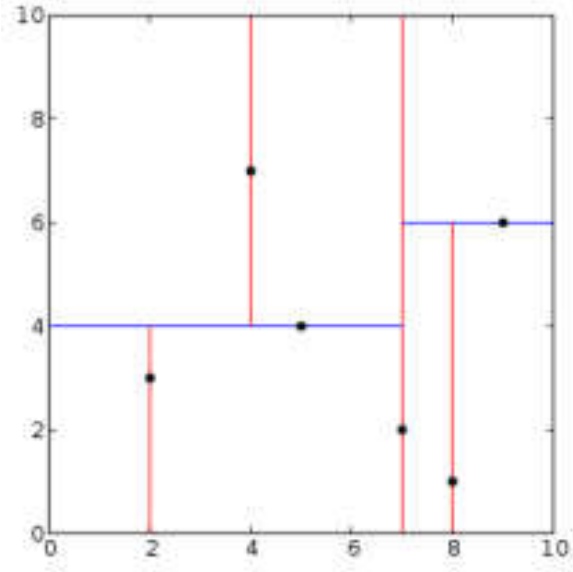


- 为了找到第一个相交三角
- 仍要遍历整个模型

Rock and Wozny [Rock91b]

更有效的切片

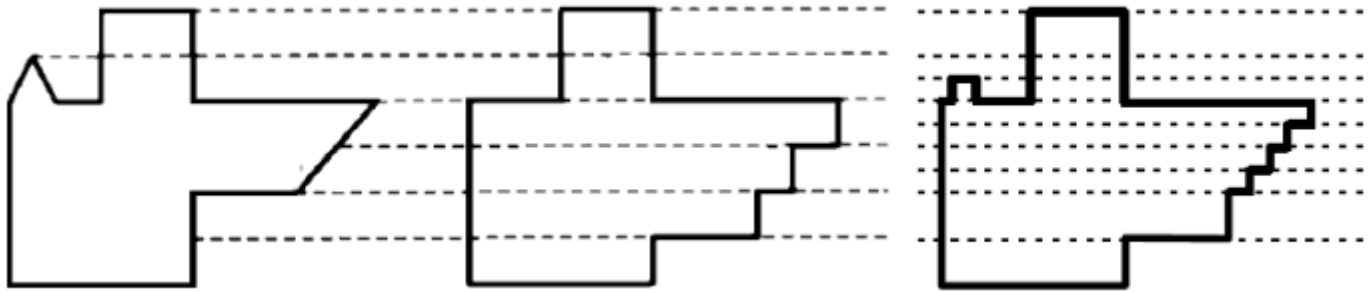
- 大规模三角网格模型的切片仍然困难
- 当前算法有很多可以改进的方向：
  - 更有效的out-of-core方法
  - 用计算机图形学方法更有效的组织数据结构
  - Z-排序
  - GPU加速



- 针对特定工艺或许有更好的思路

自适应切片

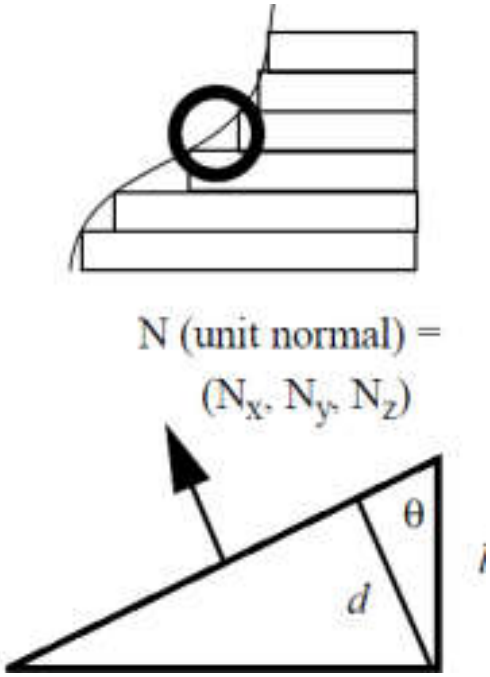
- 根据z方向的特征确定切片
- 很少使用



▪输入模型　　▪一致厚度　　▪自适应厚度

自适应切片

- 根据法向确定切片厚度
- 横向变化越平缓，层厚度越小

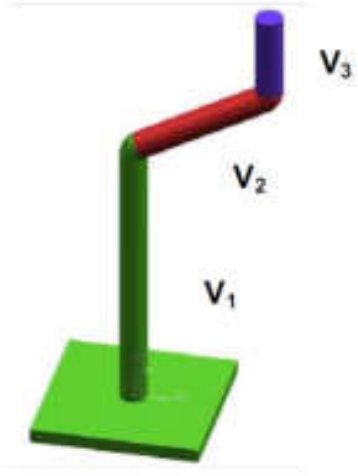


$$l = \frac{d}{\sin\theta} = \frac{d}{N_z}$$

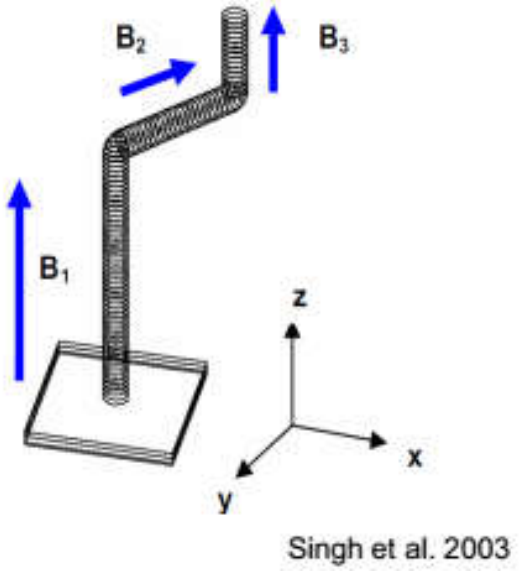
多轴切片

- 根据不同z方向分割模型
- 分别制造后装配

▪输入模型



▪分区切片



Singh et al. 2003

▪装配

3D打印流程

- 输入模型
- 打印方向
- 支撑生成
- 切片
- 路径规划
- 面向工艺的G-Code生成

路径规划

- 两种类型
  - 整层材料同时填充，如LOM，DLP
  - 整层材料逐渐填充，如FDM，SLA
  - 影响制造时间、精度、刚度、强度和变形

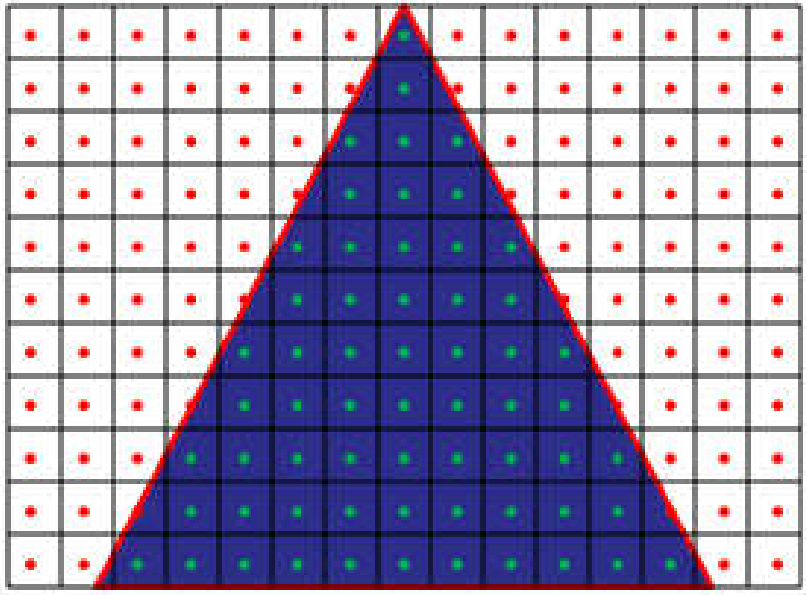
- 路径

路径规划

- 制造时间
  - 重定位耗时
  - 路径方向变化时加减速耗时
- 表面精度
  - 纤维尺寸
  - 倾斜角度
- 变形
  - 高热膨胀系数的材料
  - 在一层材料上打印另一层后，上层收缩，导致工件变形
- 刚度和强度
  - 取决于填充的体积
  - 材料性能
  - 填充遍历时间

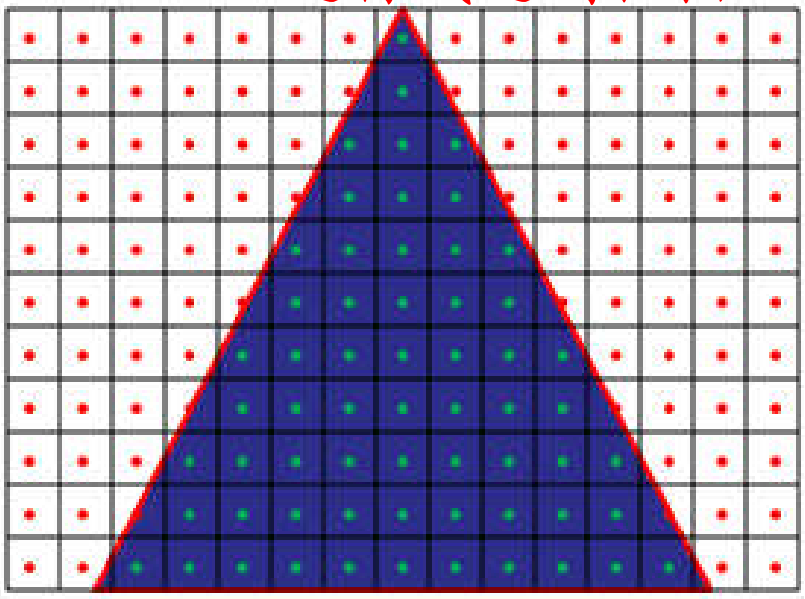
1. 简单路径规划

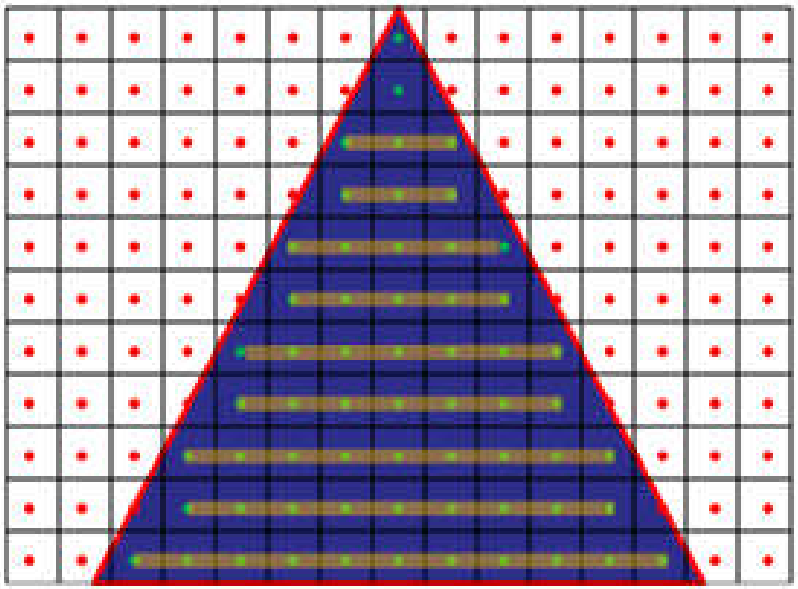
- 体素化输入模型
- 测试体素属于模型内部或外部
- 适合DLP，热相变和喷射打印



简单路径规划

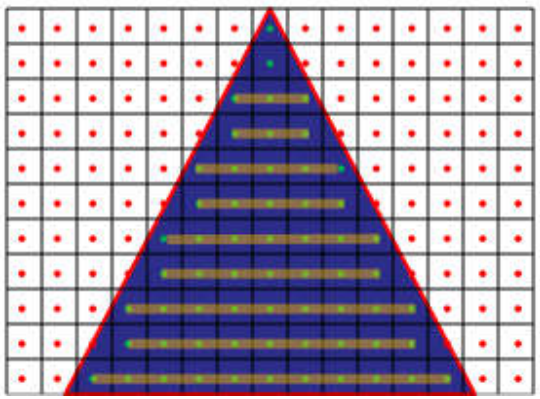
- 体素化输入模型
- 测试体素属于模型内部或外部
- 每层的行和列可作为填充路径
  - 遇到边界切换状态
  - 逐行或逐列扫描

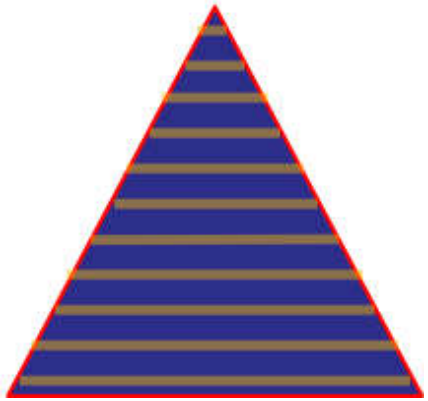




1. 简单路径规划

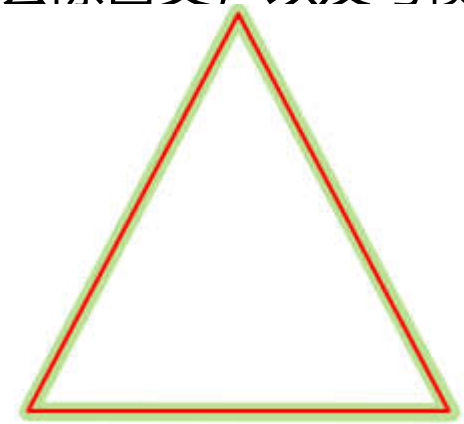
- 体素化输入模型
- 测试体素属于模型内部或外部
- 每层的行和列可作为初始填充路径
  - 遇到边界切换状态
  - 逐行或逐列扫描
- 计算路径与边界轮廓的交线，生成最终工具路径



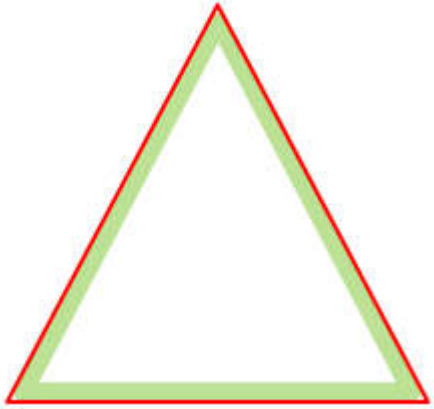


2. 边界追踪

- 为增加外表面质量
- 将外部轮廓向内偏移一个纤维半径的距离
  - 针对多边形轮廓
    - 平移顶点
    - 去除自交，以及与模型相交的部分



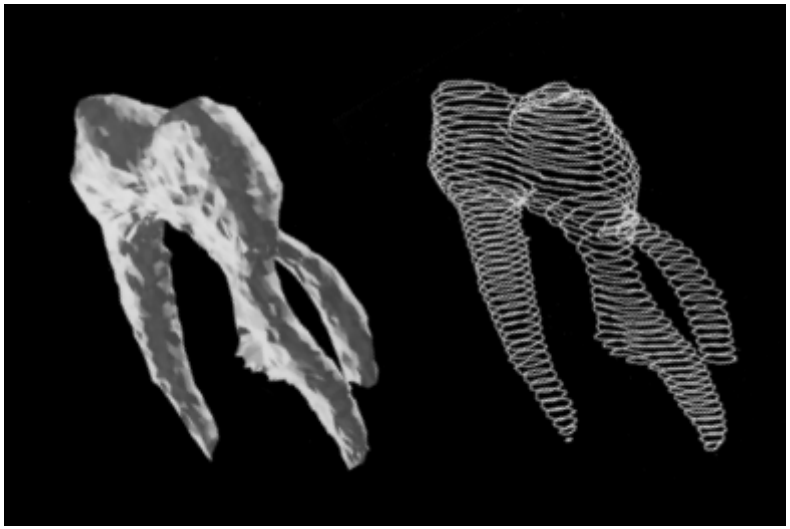
▪没有平移



▪向内平移

2. 边界追踪

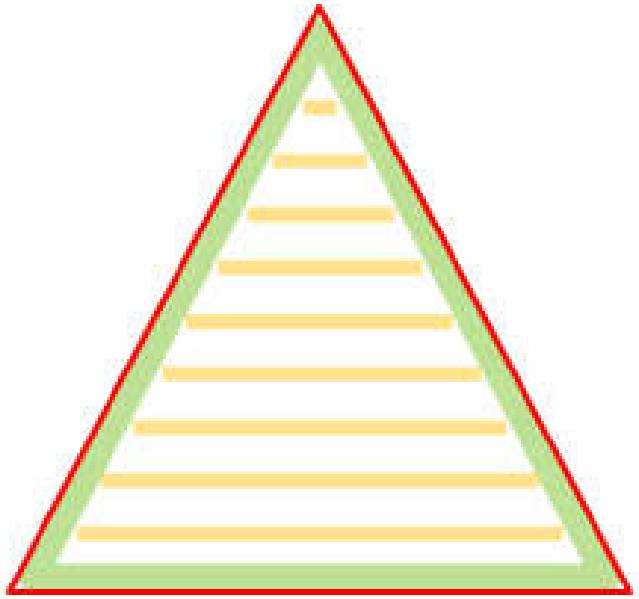
- 允许打印中空的结构
  - 包含倾斜的表面
  - 包含部分悬空结构
- 减少喷头的空行程
- 减少支撑



2. 边界追踪

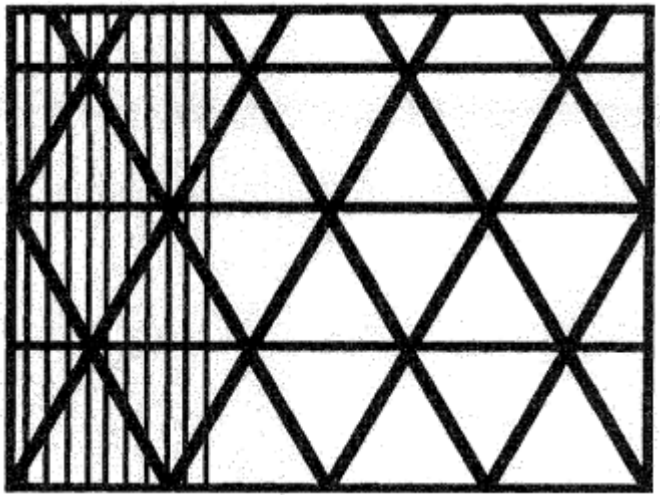
- 与内部填充相结合
- 大部分FDM路径规划工具选择的方案

  - 生成一定厚度的边界
  - 在边界内扫描



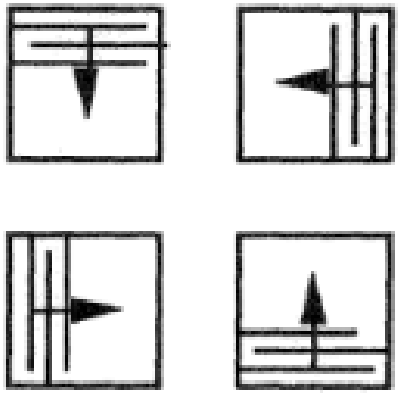
3. 更复杂的填充模式

- 三角填充
- 1993年开始应用于SLA
  - 内部层次用等边三角形填充
  - 外部边界全部填充

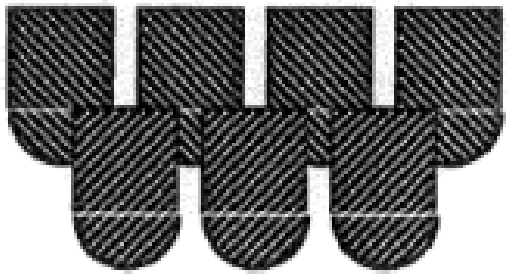


3. 更复杂的填充模式

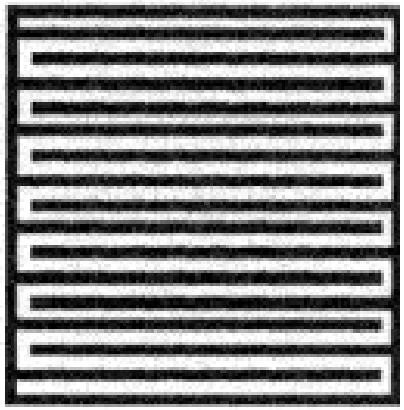
- 星形编织
  - 3D System提出
  - 扫描路径与前一个层次垂直
  - 交替切换纤维方向
  - 在外轮廓内扫描



▪变换的扫描路径



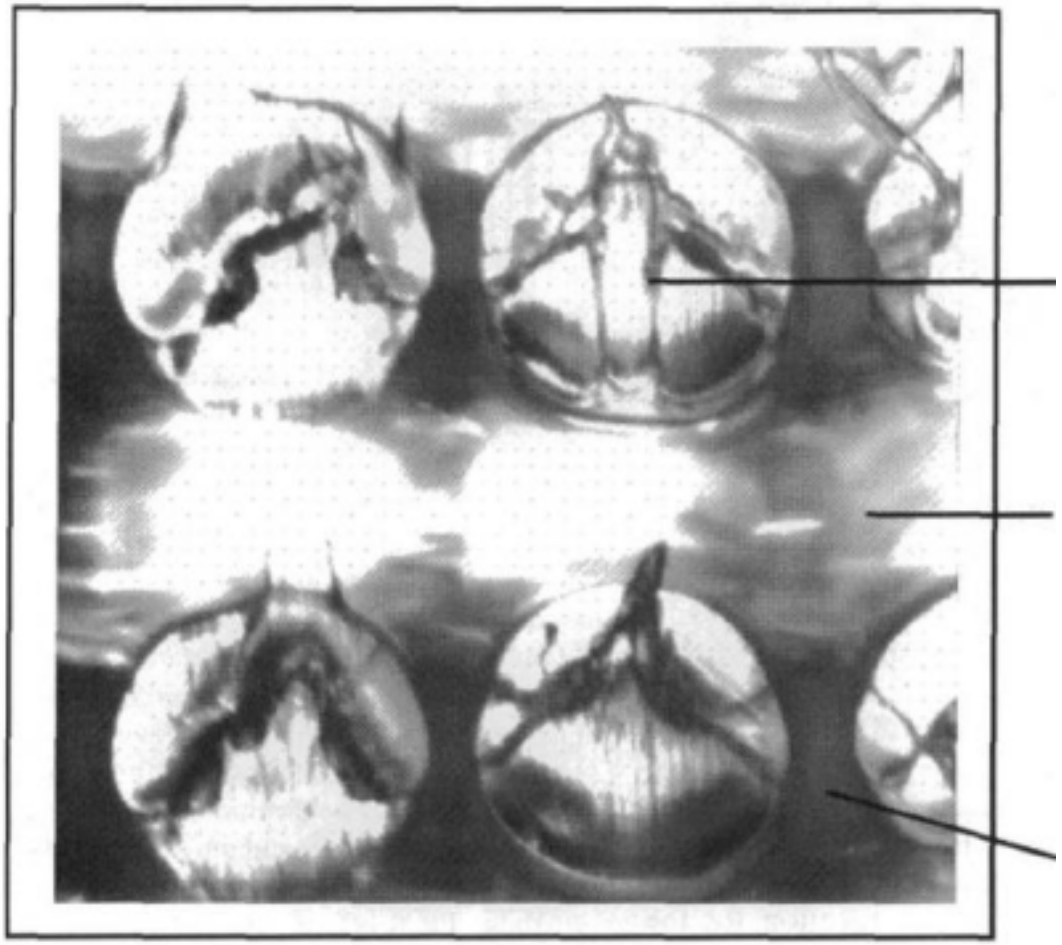
▪交错填充，可带偏移



▪单层填充模式

4. 模型填充效果

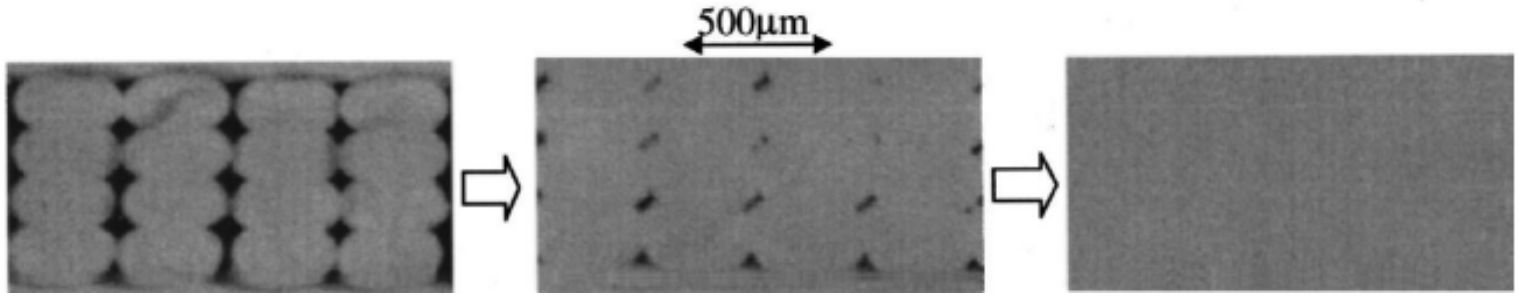
- 交错模式



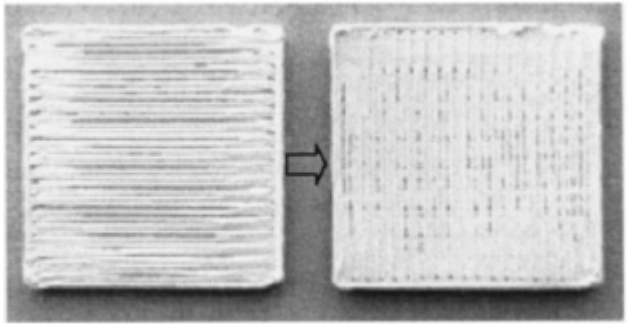
- 0°走向的纤维
- 90°走向的纤维
- 空腔

4. 模型填充效果

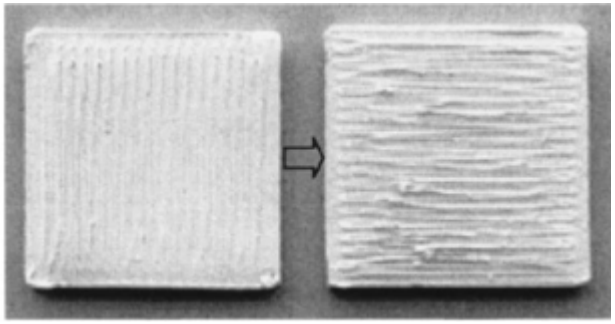
- 交错模式



▪无偏移堆叠　▪-25um偏移　▪-75um偏移



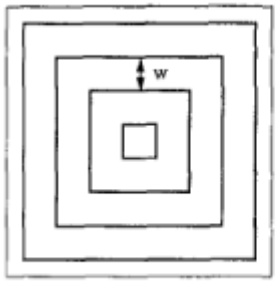
▪欠填充



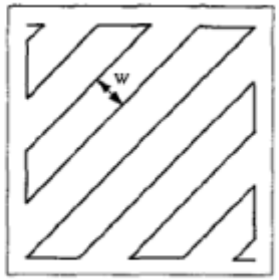
▪过度填充

5. 其它填充模式

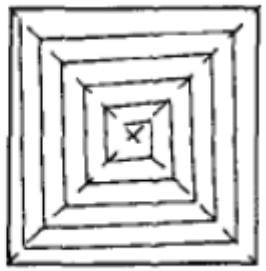
- 线填充



contour



raster


spiral

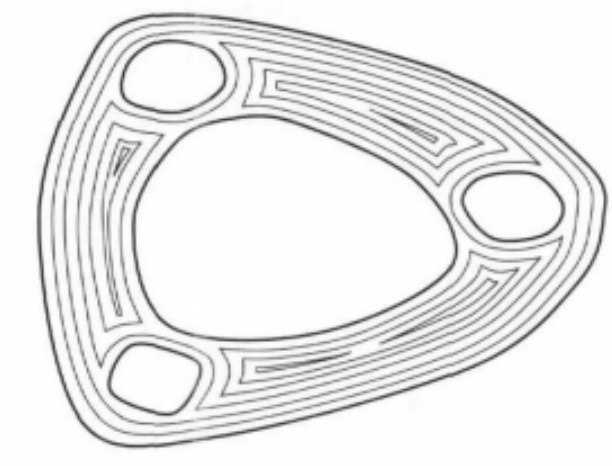
- 

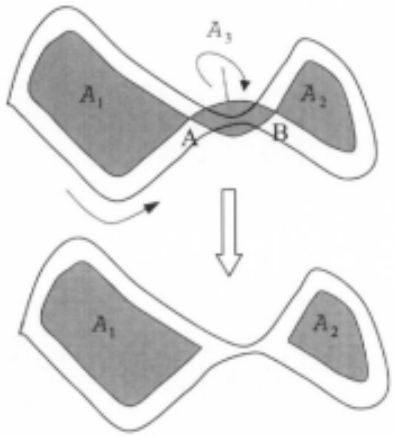
▪蜂巢模式

▪陀螺模式

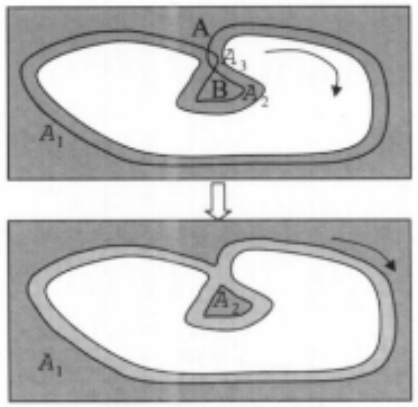
5. 其它填充模式

- 等高线模式





inward offset

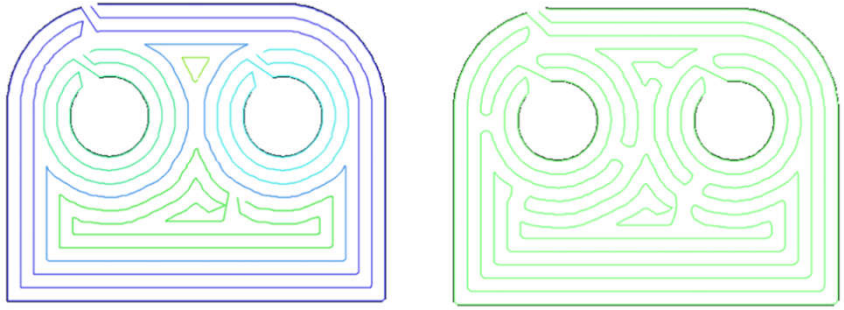


outward offset

- 注意连接顺序，优化空行程

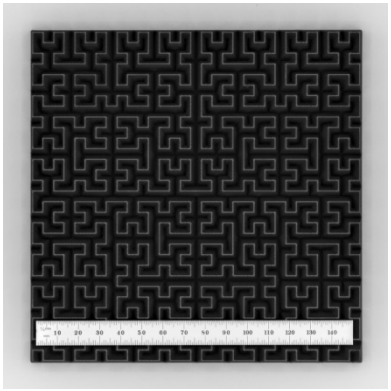
5. 其它填充模式

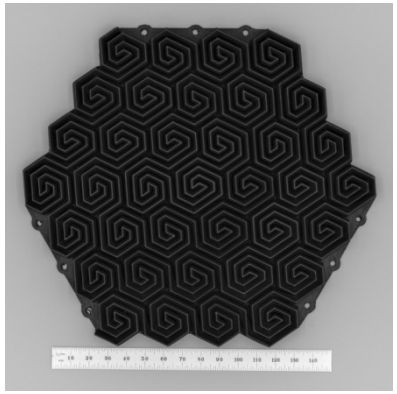
- 连续螺线



(a) (b)

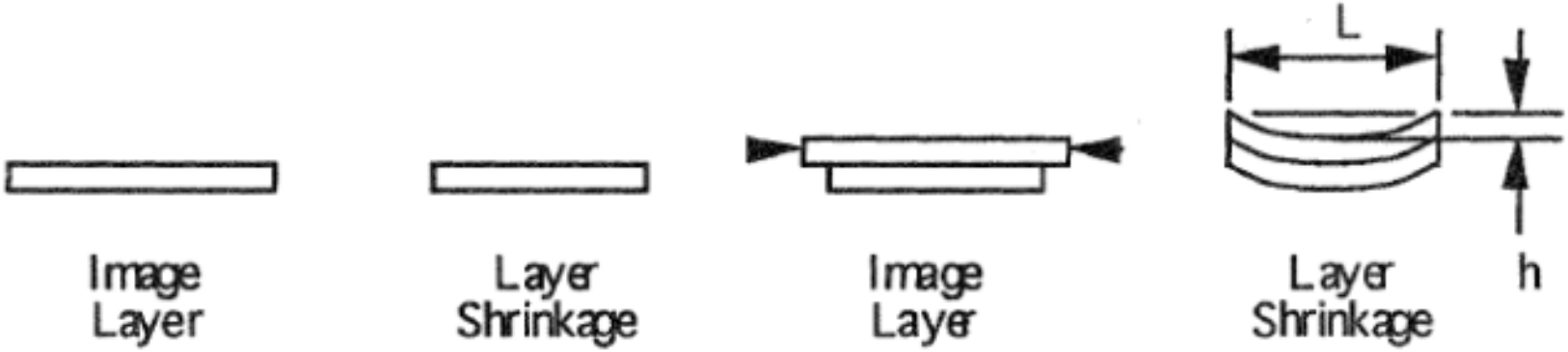
- Hilbert 曲线

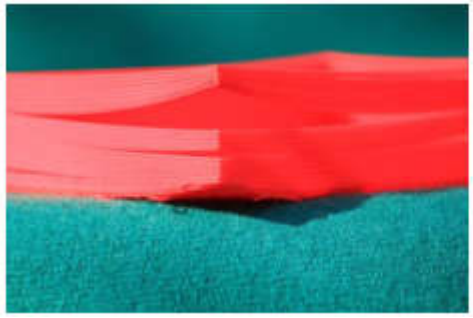




6. 材料收缩与变形

- 材料冷却和固化时发生收缩
- 优化填充模式可以减少这种现象

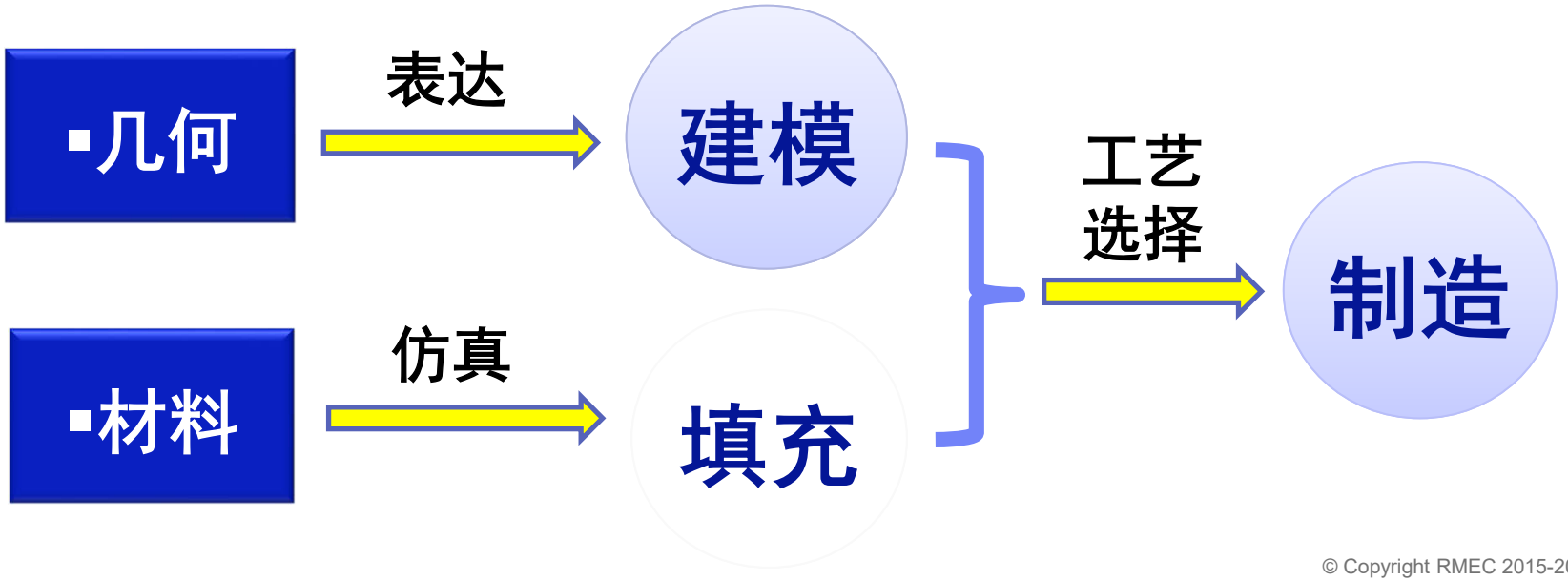




更复杂的处理-计算制造

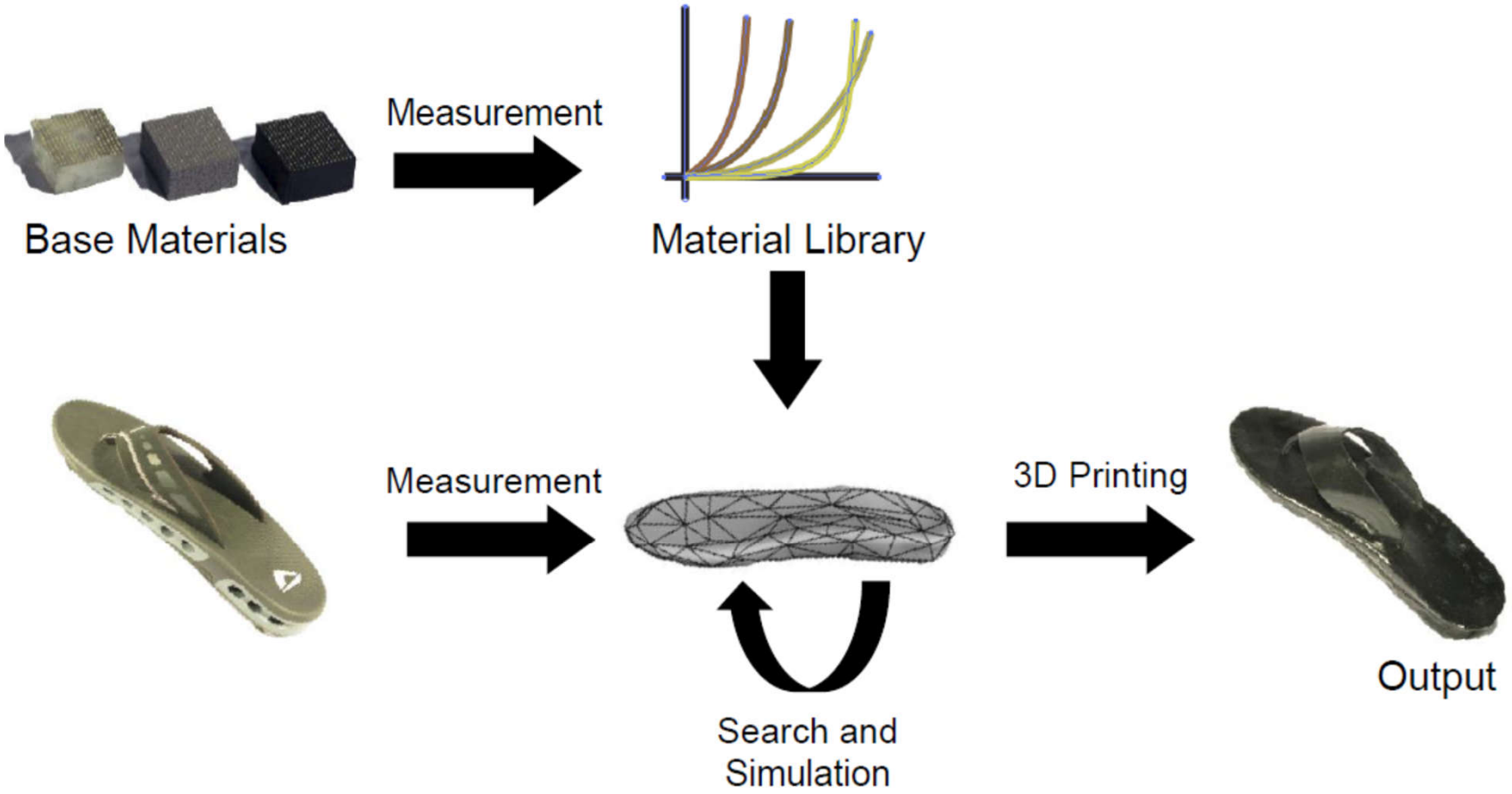
用计算的方法，生成数字化的功能结构，并根据工艺生成物理世界的材料分布。

综合考虑功能需求和材料成形过程，进行设计、制造的流程。



计算制造实例

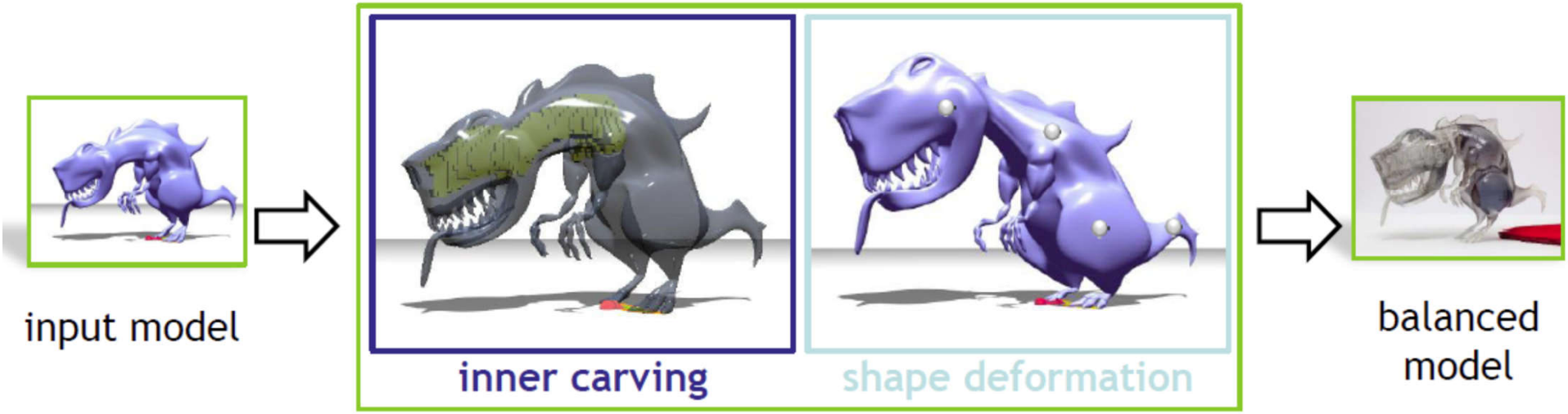
- 多材料打印问题



- [Bickel et al. 2010

计算制造实例

- 平衡问题



- [Prevost et al. 20

计算制造流程

- 水转印

总结

- 数字几何是现代计算机辅助建模的基础
- 数字几何包含数据结构的表达和相应的计算方法
- 数字几何是3D打印技术的基础
- 数字几何技术可以用于解决3D打印设计和制造中很多问题

问题

1. 在UG、ProE和Auto CAD系统中，模型的描述各采用什么形式的图形表达方式？

3. 3D打印中，STL格式表示的零件定义为一个什么样的网格？这种表达有什么缺点？

作业2

根据本节课所设计的3D打印流程，使用C++/python建立一个切片引擎。要求：

1. 以stl文件为输入，可以手工确定打印方向。

2. 支持层高控制

3. 类似SLC格式的轮廓数据，用折现表示各层轮廓，可自定义格式。

```
$$HEADERSTART
// This is a example for the use of the Layer Format //
$$ASCII
$$UNITS/1              // all coordinates are given in mm  //
// $$UNITS/0.01     all coordinates are given in units 0.01 mm //
$$DATE/070493                       // 7. April 1993 //
$$LAYERS/100                        //  100 layers //
$$HEADEREND

$$GEOMETRYSTART          // start of GEOMETRY-section//
$$LAYER/5.5              // Layer at height z = 5.5 mm//

$$POLYLINE/0,0,5,1.00,2.02,3.30,3.42,5.23,5.01,1.57,5.6,1.00,2.02
$$HATCHES/0,2,10.2,10.4,12.34,12.5,8.8,9.3,15.7,13.2
$$POLYLINE/0,1,10,1.2,4.01,.....
..
..
$$LAYER/5.6
$$POLYLINE/0,0,200,10.23,12.34,.....
.....
..
..
$$LAYER/15.5
$$POLYLINE/0,0,200,13.23,12.34,.....
.....
..
..
$$GEOMETRYEND
```