Introduction to Solid Modeling

Hongxin Zhang and Jieqing Feng

2007-01-15

State Key Lab of CAD&CG
Zhejiang University

Contents

Solid Representations

- Solid Model: geometric object with interior, such as cube, piston engine
- Solid representation: describe the geometry and characteristics completely

What is a good solid representation?

Requirements for Solid Representation

- Domain
- Unambiguity
- Uniqueness
- Accuracy
- Validness
- Closure
- Compactness and Efficiency

Requirements for Solid Representation

- Domain

  While no representation can describe all possible solids, a representation should be able to represent a useful set of geometric objects.

- Unambiguity

  When you see a representation of a solid, you will know what is being represented without any doubt. An unambiguous representation is usually referred to as a complete one.

Requirements for Solid Representation

- Uniqueness

  That is, there is only one way to represent a particular solid. If a representation is unique, then it is easy to determine if two solids are identical since one can just compare their representations.

- Accuracy

  A representation is said **accurate** if no approximation is required.

Requirements for Solid Representation

- Validness

  This means a representation should not create any invalid or impossible solids. More precisely, a representation will not represent an object that does not correspond to a solid.

- Closure

  Solids will be transformed and used with other operations such as union and intersection. "Closure" means that transforming a valid solid always yields a valid solid

Requirements for Solid Representation

- Compactness and Efficiency

  A good representation should be compact enough for saving space and allow for efficient algorithms to determine desired physical characteristics

About Solid Representations

- Designing representations for solids is a difficult job
- The requirements may be contradictory with each other
  - Compromises are often necessary
- Three classical representations
  - Wireframes
  - Boundary Representations (B-Rep)
  - Constructive Solid Geometry (CSG)

Wireframe Models

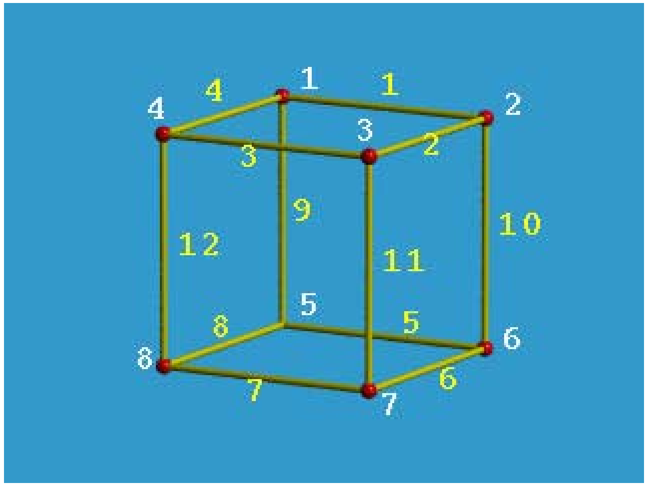
- Wireframe model consists of two tables
  - Vertex table: vertices and their coordinate values
  - Edge table: two incident vertices of edges
- A wireframe model *does not* have face information

Example of Wireframe Model

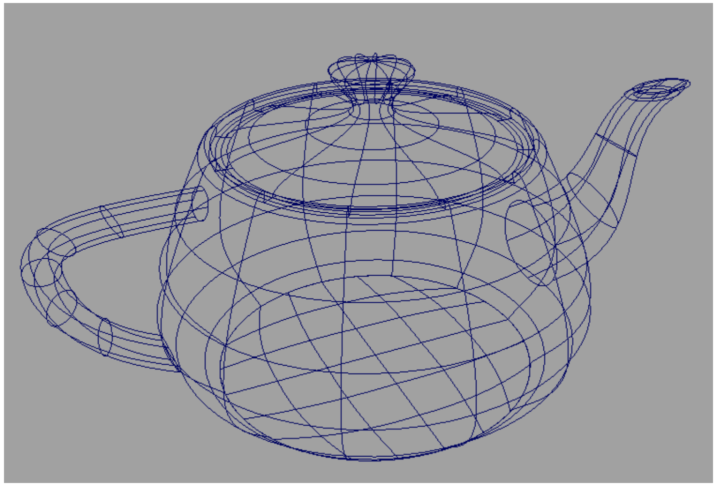
Vertex Table			
Vertex #	x	y	z
1	1	1	1
2	1	-1	1
3	-1	-1	1
4	-1	1	1
5	1	1	-1
6	1	-1	-1
7	-1	-1	-1
8	-1	1	-1

Edge Table		
Edge #	Start Vertex	End Vertex
1	1	2
2	2	3
3	3	4
4	4	1
5	5	6
6	6	7
7	7	8
8	8	5
9	1	5
10	2	6
11	3	7
12	4	8

Example of Wireframe Model



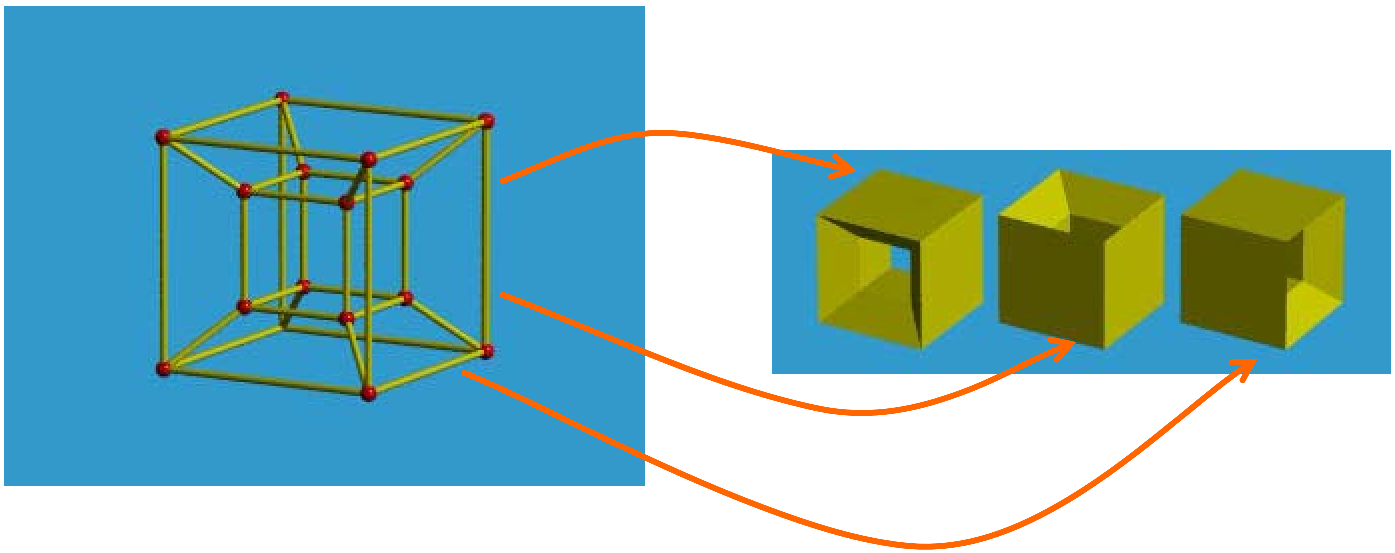
Wireframe model described by previous tables



Wireframe model with curve edges

Wireframe Models Are Ambiguous

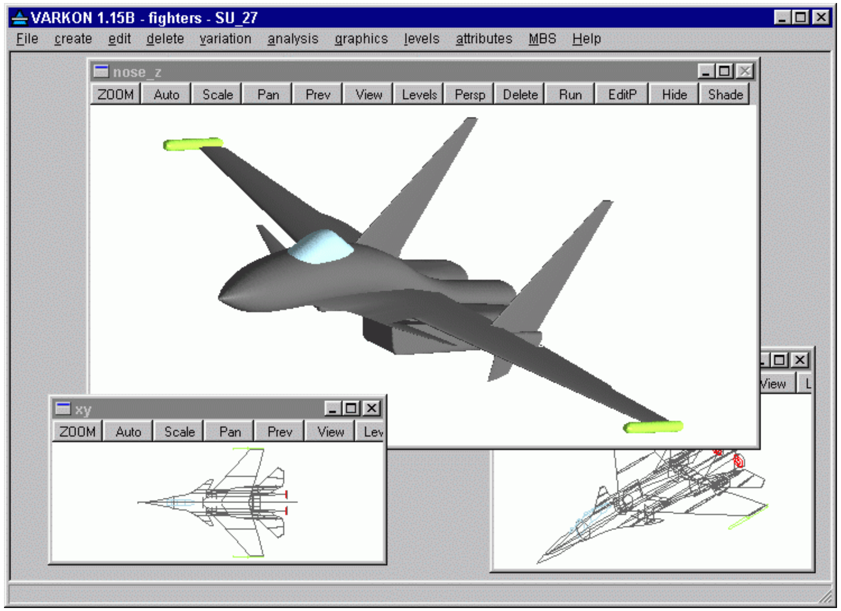
- Examples: 16 vertices and 32 edges



All interpretations are right!

Application of Wireframe Models

- Preview the complex solid models
  - Shading is time-consuming
  - provide a general feeling of the final result



Boundary Representations

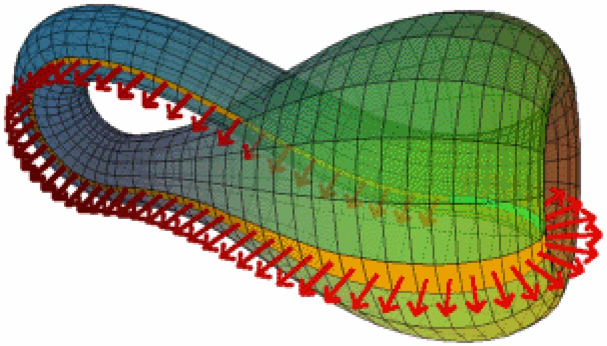
- Boundary Representation, or B-rep
  - Extension to the wireframe model by adding face information
  - A solid is bounded by its surface and has its *interior* and *exterior*

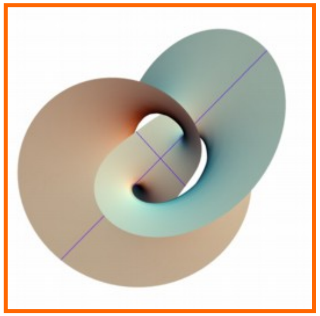
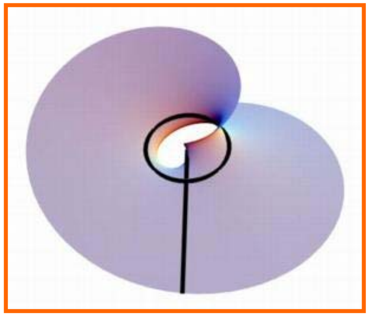
Boundary Representations

- Two types of information in B-rep
  - Topological information:
    - relationships among vertices, edges and faces
    - orientation of edges and faces
  - Geometric information:
    - equations of the edges and faces

Boundary Representations

- Orientation of face is important
  - Count Clockwise: normal points to the exterior of model
  - Faces
    - Orientable
    - Non-orientable

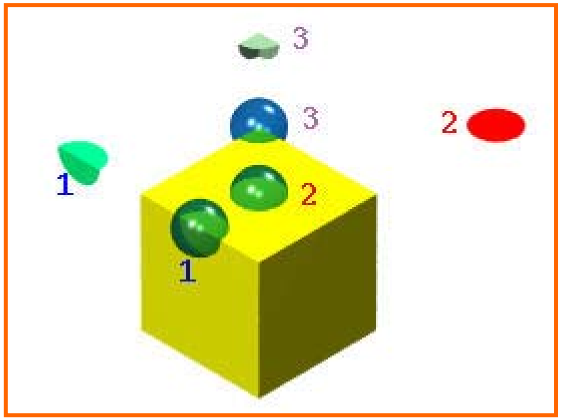


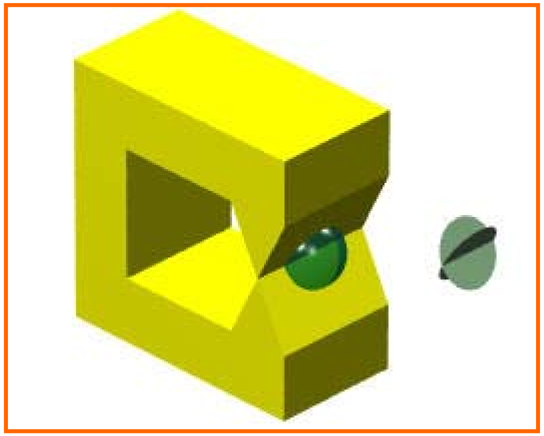
Manifolds (Review)

- Manifold Solid Modeling
  - The surface of a solid is 2-D manifold
  - 2-D manifold
    - For each point *x* on the surface, there exists an open ball with center *x* and sufficiently small radius, so that the intersection of this ball and the surface can be continuously deformed to an open disk
    - Open ball: $x^2 + y^2 + z^2 < r^2$
- Non-manifold Solid Modeling

Example of 2-D manifold



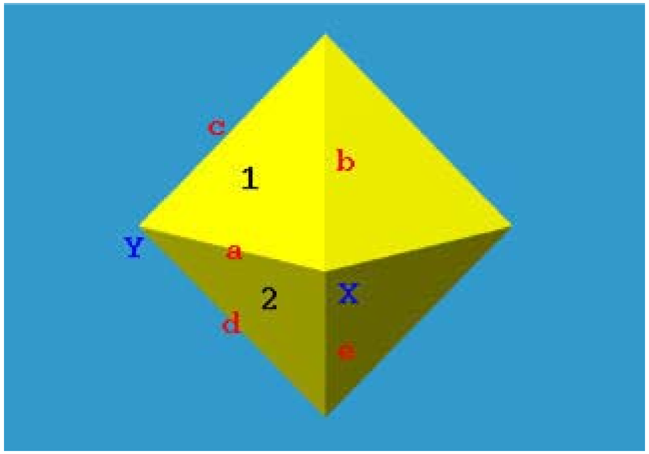
2-D manifold

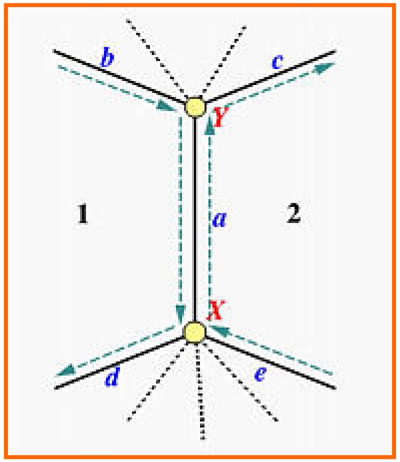


2-D non-manifold

The Winged-Edge Data Structure

- The winged-edge data structure uses *edges* to keep track all information in the solid model





The Winged-Edge Data Structure

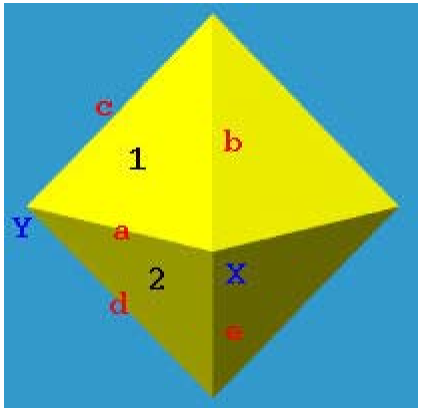
- In the following example, assuming
  - No hole in the face (can be extended later)
  - Edges and faces are line segments and polygons (extended to curves and surfaces)
  - Description
    - Vertices ➔ upper cases (A, B, C)
    - Edges ➔ lower cases (a, b, c)
    - Faces ➔ digits (1, 2, 3)

The Winged-Edge Data Structure

- Edge: a
  - Two incident vertices: X and Y
  - Two incident faces: 2(left) and 1 (right) in case a=XY

- Face: 1
  - Three ordered edges: a, c, b

- Edge: a
  - In face 1: X → Y
  - In face 2: Y → X



What information is important?

The Winged-Edge Data Structure

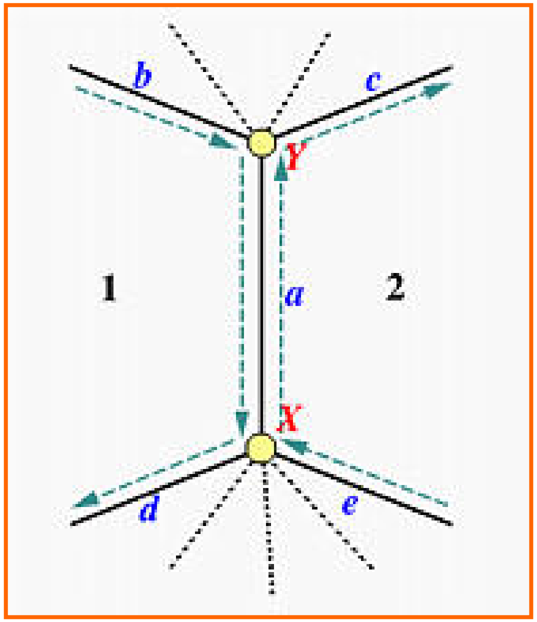
- Vertices of this edge
- Its *left* and *right* faces
- The predecessor and successor of this edge when traversing its left face, and
- The predecessor and successor of this edge when traversing its right face

Edge Table

- Edge name
- Start vertex and end vertex
- Left face and right face
- The predecessor and successor edges when traversing its left face
- the predecessor and successor edges when traversing its right face

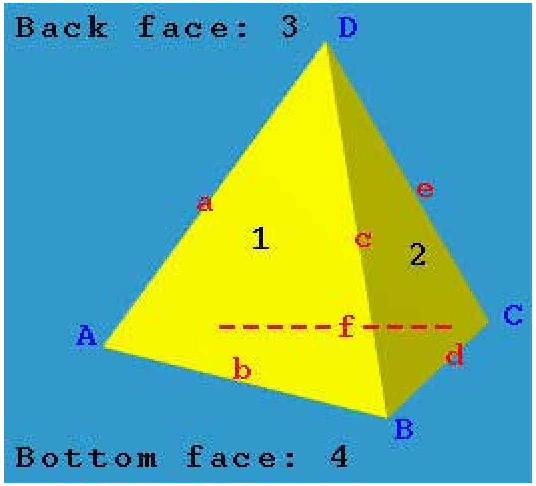
Edge Table

Edge	Vertices		Faces		Left Traverse		Right Traverse	
Name	Start	End	Left	Right	Pred	Succ	Pred	Succ
a	X	Y	1	2	b	d	e	c



Winged edge a: b, c, d, e are the wings of edge a !

Complete Edge Tables

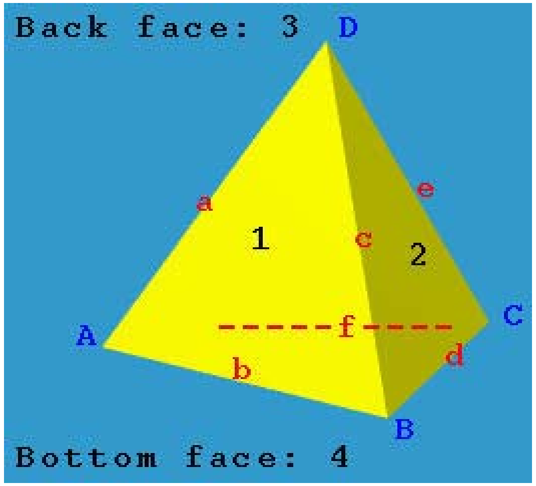


Edge	Vertices		Faces		Left Traverse		Right Traverse	
Name	Start	End	Left	Right	Pred	Succ	Pred	Succ
a	A	D	3	1	e	f	b	c
b	A	B	1	4	c	a	f	d
c	B	D	1	2	a	b	d	e
d	B	C	2	4	e	c	b	f
e	C	D	2	3	c	d	f	a
f	A	C	4	3	d	b	a	e

Other Tables

- Vertex table: an edge incidents to this vertex
- Face Table: an face contains this edge

Vertex Name	*Incident Edge*
A	a
B	b
C	d
D	e



Face Name	*Incident Edge*
1	a
2	c
3	a
4	b

These tables are not unique!

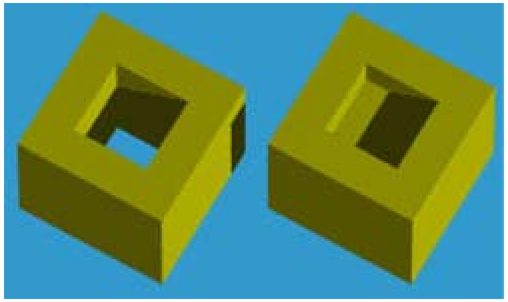
The Adjacency Relation

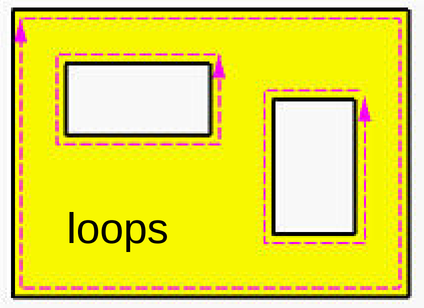
- The Adjacency Relation
  - From edge → vertex, face, edge ?
  - From face → vertex, edge, face ?
  - From vertex → edge, face, vertex ?

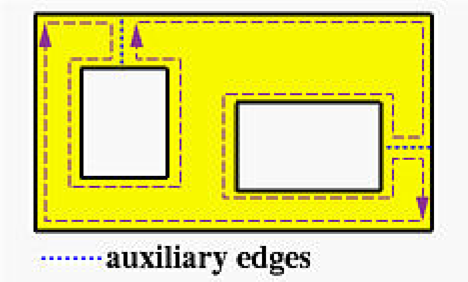
- The Winged Edge data structure can accomplish these queries efficiently!

Face with Holes

- Two solutions
  1. Introducing loops: reverse direction of face edge order
  2. Introducing auxiliary edges:
     - Identify the auxiliary edges: its left and right faces are same







The Euler-Poincaré Formula

- Euler-Poincaré Formula can be used for check the validness of a solid

- A more elaborate formula: for potholes and penetrated holes

$$V - E + F - (L - F) - 2(S - G) = 0$$

V-E+F-(L-F)-2(S-G)=0

- **V**: the number of vertices
- **E**: the number of edges
- **F**: the number of faces
- **G**: the number of penetrated holes (*genus*)
- **S**: the number of *shells*
  - A shell is bounded by a 2-manifold surface, which can have its own genus value
  - The solid itself is counted as a shell
- **L**: the number of all outer and inner loops

Examples (1)

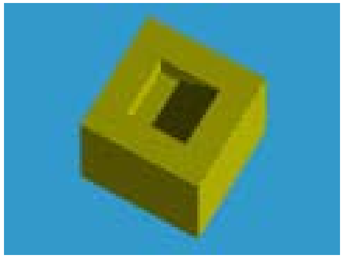
- A cube: eight vertices (**V** = 8), 12 edges (**E** = 12) and six faces (**F** = 6), no holes and one shell (**S**=1); **L** = **F** (each face has only one outer loop)

$$V-E+F-(L-F)-2(S-G)$$
$$= 8-12+6-(6-6)-2(1-0)$$
$$= 0$$

Examples (2)

- 16 vertices, 24 edges, 11 faces, no holes, 1 shell and 12 loops (11 faces + one inner loop on the top face)

$$V-E+F-(L-F)-2(S-G)$$
$$= 16-24+11-(12-11)-2(1-0)$$
$$=0$$



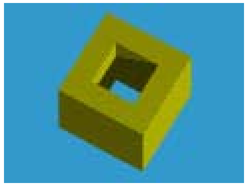
Examples (3)

- 16 vertices, 24 edges, 10 faces, 1 hole (*i.e.*, genus is 1), 1 shell and 12 loops (10 faces + 2 inner loops on top and bottom faces)

$$V-E+F-(L-F)-2(S-G)$$

$$= 16-24+10-(12-10)-2(1-1)$$

$$=0$$



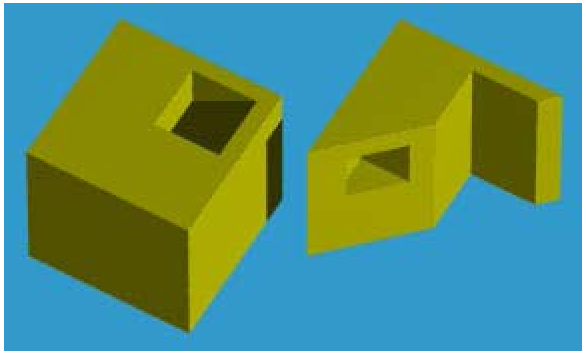
Examples (4)

The following solid has a penetrating hole and an internal cubic chamber as shown by the right cut-away figure. It has 24 vertices, 12*3 (cubes) = 36 edges, 6*3 (cubes) - 2 (top and bottom openings) = 16 faces, 1 hole (*i.e.*, genus is 1), 2 shells and 18 loops (16 faces + 2 inner loops on top and bottom faces)

V-E+F-(L-F)-2(S-G)

= 24-36+16-(18-16)-2(2-1)

=0



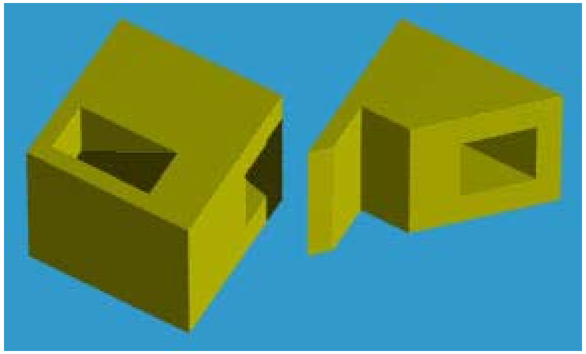
Examples (4)

The following solid has two penetrating holes and no internal chamber as shown by the right cut-away figure. It has 24 vertices, 36 edges, 14 faces, 2 hole (*i.e.*, genus is 2), 1 shells and 18 loops (14 faces + 4)

V-E+F-(L-F)-2(S-G)

=24-36+14-(18-14)-2(1-2)

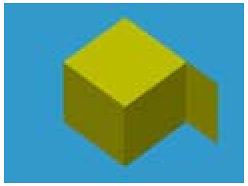
=0



The Euler-Poincaré Formula

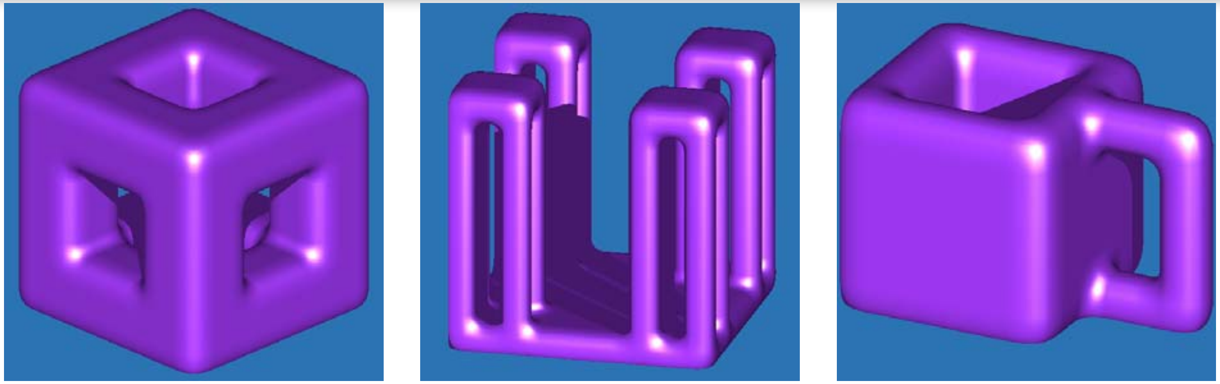
- Topological information and geometric information should be consistent
- Checking validness of solid by Euler-Poincaré formula
  - If the value of Euler-Poincaré formula is non-zero, the representation is definitely not a valid solid
  - the value of the Euler-Poincaré formula being zero does not guarantee the representation would yield a valid solid

10 vertices, 15 edges, 7 faces, 1 shell and no hole



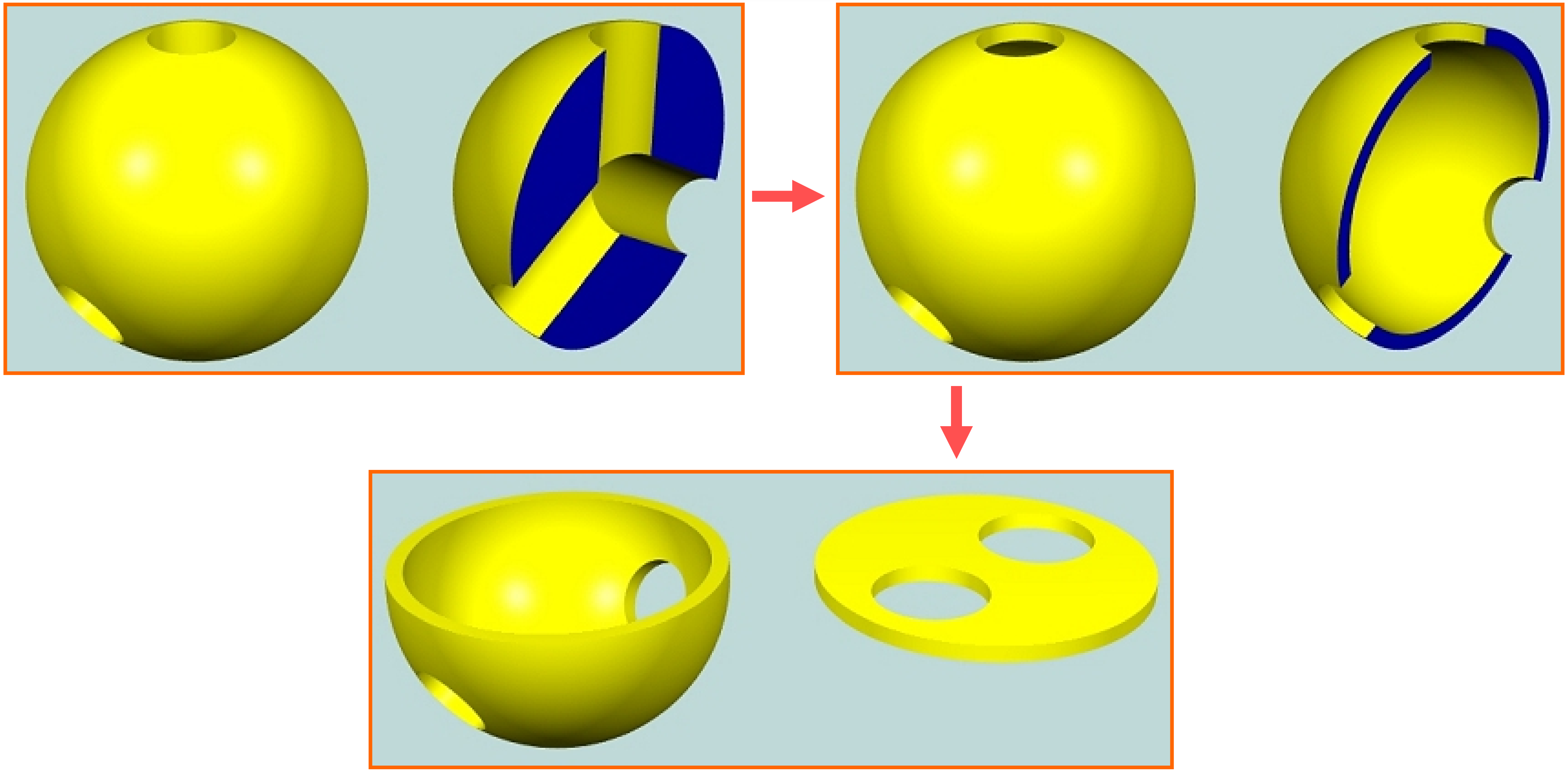
$$V-E+F-(L-F)-2(S-G) = 10-15+7-(7-7)-2(1-0)=0$$

Count Genus Correctly



- The Euler-Poincaré Formula describes the topological property amount vertices, edges, faces, loops, shells and genus
- Any topological transformation applied to the model will *not* alter this relationship

Sphere Punched by Three Tunnels



The Genus is 2

Euler Operators

- Euler Operators: modification of solid model while keeping the Euler-Poincaré formula tenable

$$V-E+F-(L-F)-2(S-G)=0$$

- There are two groups of such operators
  - the Make group: M
  - the Kill group: K

Euler Operators

- Euler operators are written as:
  - Mxyz: x,y,z are vertex, edge, face, loop, shell and genus, e.g., MEV—adding an edge and a vertex
  - Kxyz: similar
- Euler operators form a complete set of modeling primitives for manifold solids (Mantyla) ←→ Every topologically valid polyhedron can be constructed from an initial polyhedron by a finite sequence of Euler operations

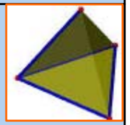
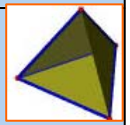
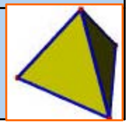
The Make Group of Euler Operators

- Adding some elements into the existing model creating a new one: $V-E+F-(L-F)-2(S-G)=0$

Operator Name	Meaning	V	E	F	L	S	G
MEV	Make an edge and a vertex	+1	+1				
MFE	Make a face and an edge		+1	+1	+1		
MSFV	Make a shell, a face and a vertex	+1		+1	+1	+1	
MSG	Make a shell and a hole					+1	+1
MEKL	Make an edge and kill a loop		+1		-1		

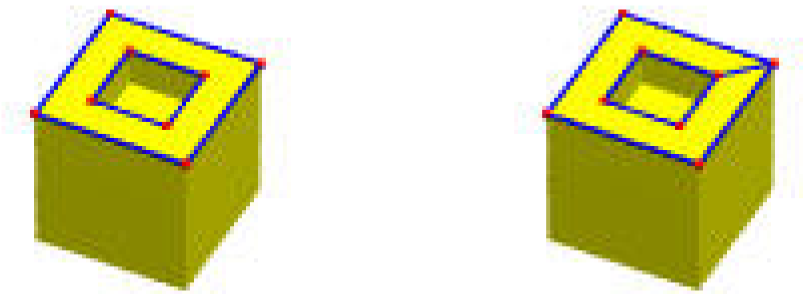
Note: adding a face produces a loop, the outer loop of that face

Example: construct a tetrahedron

Operator Name	*Meaning*	*V*	*E*	*F*	*L*	*S*	*G*	**Result**
MSFV	**Make a shell, a face and a vertex**	**+1**		**+1**	**+1**	**+1**		
MEV	**Make an edge and a vertex**	**+1**	**+1**					
MEV	**Make an edge and a vertex**	**+1**	**+1**					
MEV	**Make an edge and a vertex**	**+1**	**+1**					
MFE	**Make a face and an edge**		**+1**	**+1**		**+1**		
MFE	**Make a face and an edge**		**+1**	**+1**		**+1**		
MFE	**Make a face and an edge**		**+1**	**+1**		**+1**		

Example: MEKL

- MEKL: make an edge and kill a loop



The Kill Group of Euler Operators

- The Kill group just performs the opposite of what the Make group does

Operator Name	Meaning	V	E	F	L	S	G
KEV	Kill an edge and a vertex	-1	-1				
KFE	Kill a face and an edge		-1	-1	-1		
KSFV	Kill a shell, a face and a vertex	-1		-1	-1	-1	
KSG	Kill a shell and a hole					-1	-1
KEML	Kill an edge and make a loop		-1		+1		

Constructive Solid Geometry

- Solids representation: *Constructive Solid Geometry*, or *CSG* for short
- A CSG solid is constructed from a few *primitives* with *Boolean operators*
- CSG solid
  - Representation
  - Design methodology, Design process

CSG Primitives

- Standard CSG primitives: block (cube), triangular prism, sphere, cylinder, cone, torus
- *Instantiated* primitives via transformation: scaling, translation, rotation

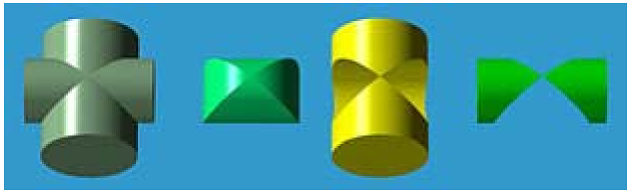
Block: center (0,0,0), size (2,2,2)

instantiated block: center(3,2,3), size(5,3,3)

translate(scale(Block, < 2.5, 1.5, 1.5 >), < 3, 2, 3 >)

Boolean Operators

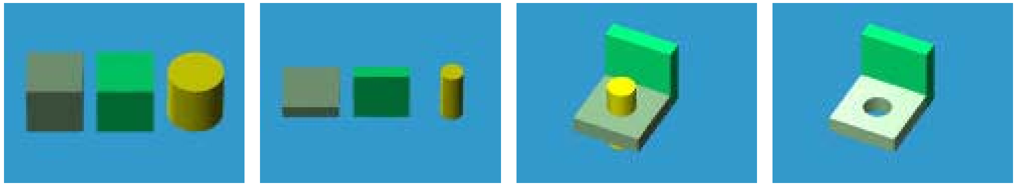
- Set operations between sets **A** and **B**
  - Union: all points from either **A** or **B**
  - Intersection: all points in both **A** and **B**
  - Difference: all points in **A** but not in **B**
- Example: **A** and **B** are two orthogonal cylinders



$\cup$ $\cap$ **A-B** **B-A**

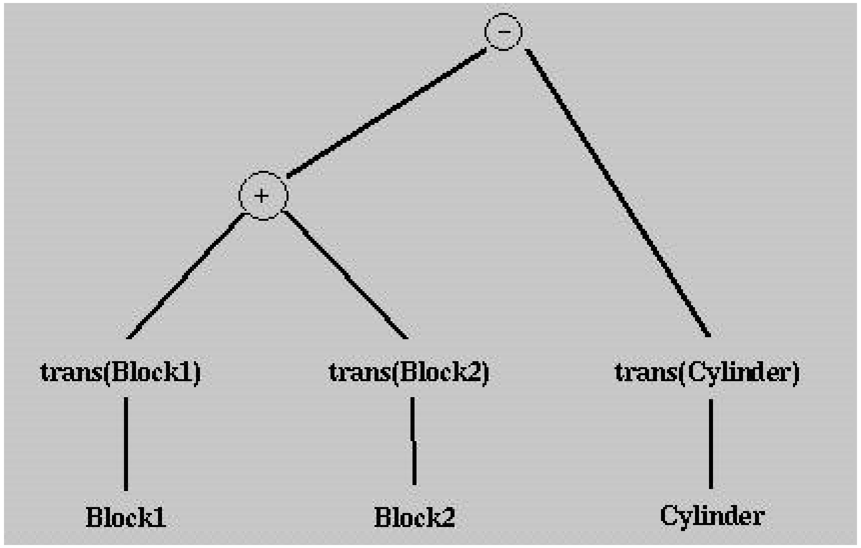
Boolean Operators

- Bracket Model Example
  - scaling blocks and cylinder
  - (scaled block) union (scaled block)
    or (block) difference (scaled block)
  - (union blocks) difference (scaled cylinder)
    or (difference blocks) difference (scaled cylinder)



CSG Expressions

- use **+**, **^** and **-** for (regularized) set union, intersection and difference



CSG Tree

CSG Expression

- CSG representations are not unique

Interior, Exterior and Closure

- A solid is a 3D object, so does its interior and exterior, its boundary is a 2D surface

- Example
  - sphere: $x^2+y^2+z^2=1$
  - Interior: $x^2+y^2+z^2<1$
  - Closure of interior: $x^2+y^2+z^2\leq 1$
  - Exterior: $x^2+y^2+z^2>1$

Formal Definitions: interior

- **int(*S*)**:
  - A point ***P*** is an *interior point* of a solid ***S*** if there exists a radius *r* such that the open ball with center ***P*** and radius *r* is contained in the solid ***S***.
  - The set of all interior points of solid ***S*** is the *interior* of ***S*** , written as **int(*S*)**

Formal Definitions: exterior

- **ext(*S*)**:
  - A point ***Q*** is an *exterior point* of a solid ***S*** if there exists a radius *r* such that the open ball with center ***Q*** and radius *r* does not intersect the solid ***S***.
  - The set of all exterior points of solid ***S*** is the *exterior* of ***S*** , written as **ext(*S*)**

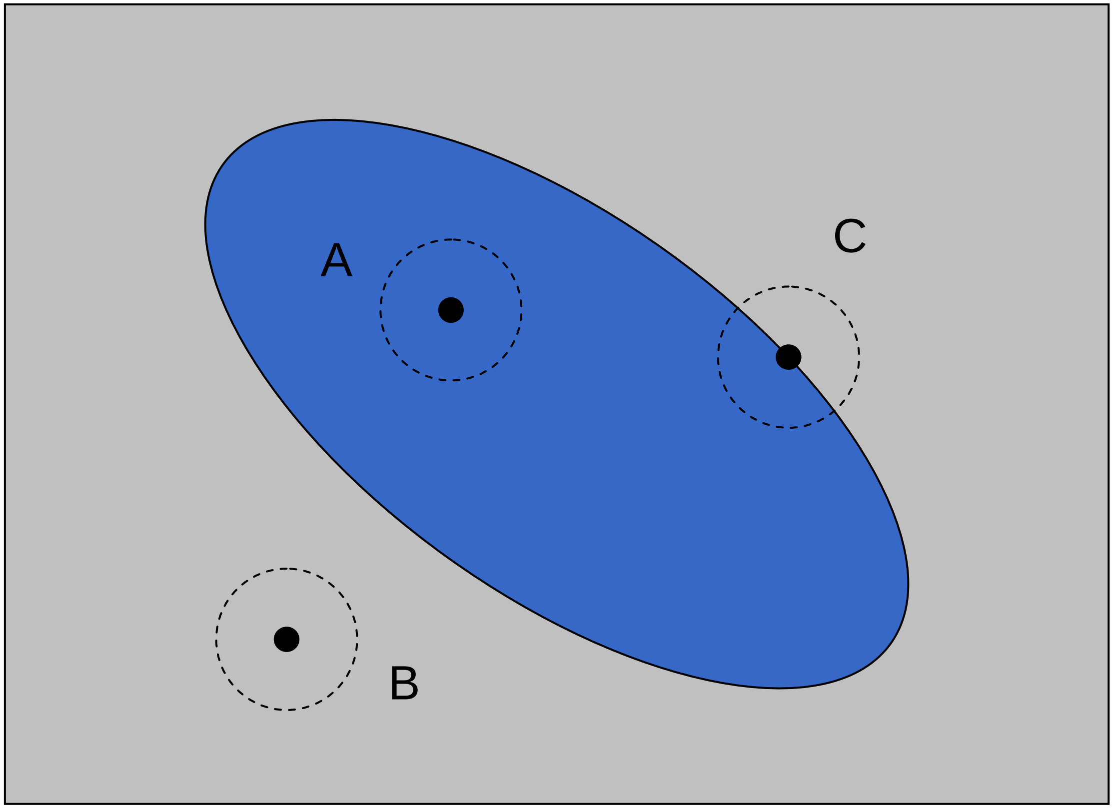
Formal Definitions: closure

- **b(S)**: Those points that are not in the interior nor in the exterior of a solid ***S*** constitutes the *boundary* of solid ***S*** , written as **b(S)**.
- **closure(S)**: The *closure* of a solid ***S*** is defined to be the union of ***S***'s interior and boundary, written as **closure(*S*)**

Formal Definitions: some notes

- The union of interior, exterior and boundary of a solid is the whole space.
- The closure of solid ***S*** contains all points that are not in the exterior of ***S***

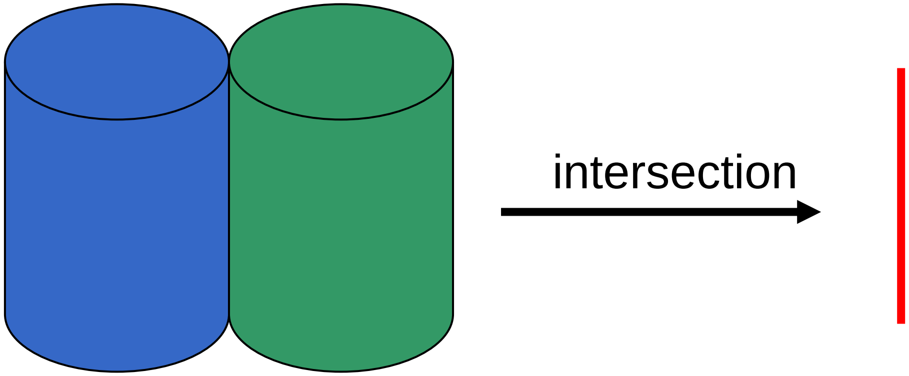
Examples



A: interior point   B: exterior point   C: boundary point

Regularized Boolean Operators

- The Boolean operation of two solids is always still solid?



Regularized Boolean Operators

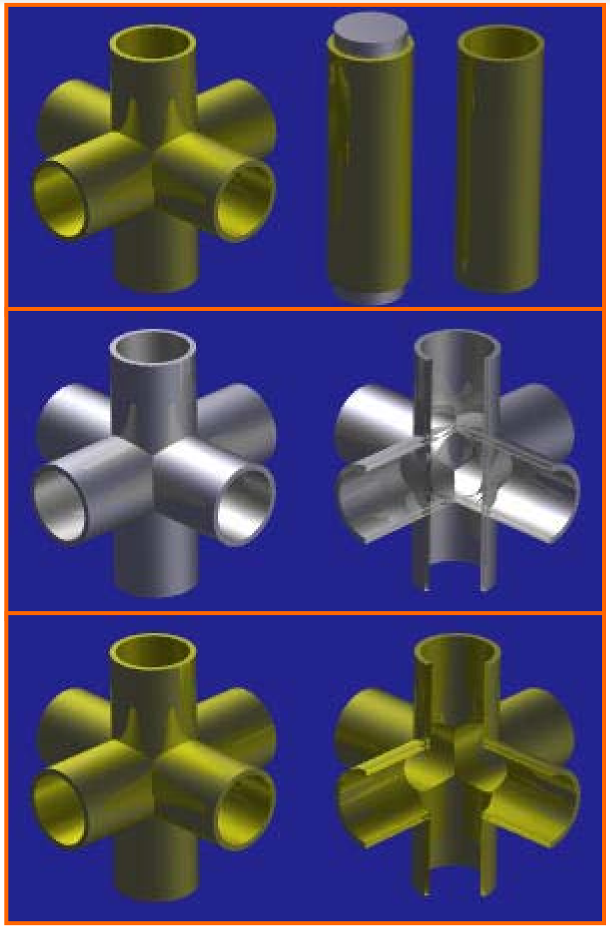
- Let **+**, **^** and **-** be regularized set union, intersection and difference

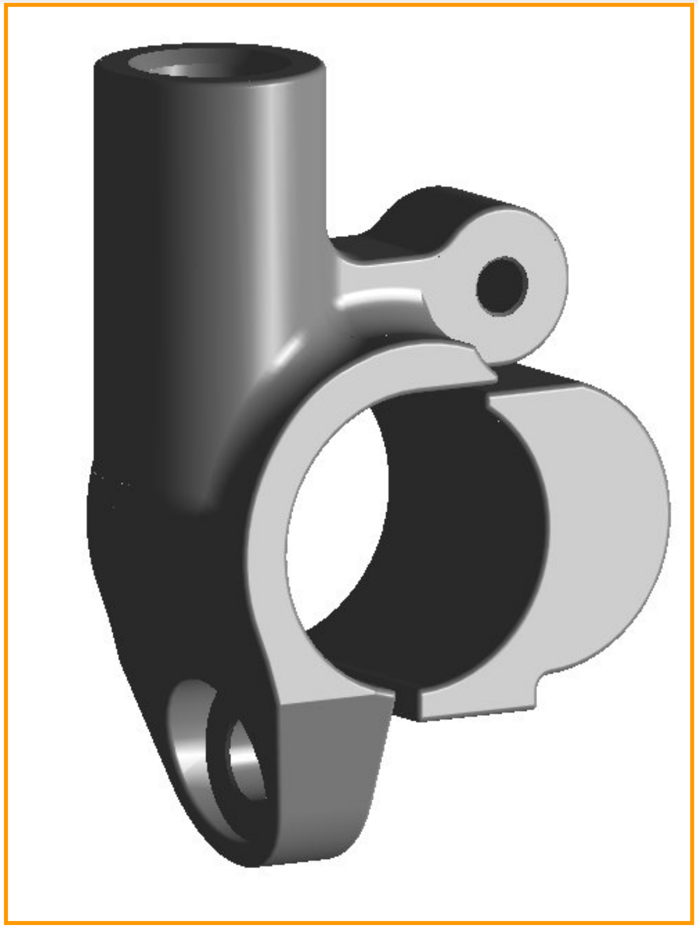
A*+*B = **closure(int(**set union of ***A*** and ***B*)**

A*^*B = **closure(int(**set intersection of ***A*** and ***B*)**

A*-*B = **closure(int(**set difference of ***A*** and ***B*)**

CSG Design Examples





Download courses

http://www.cad.zju.edu.cn/home/jqfeng/GM/GM08.zip

About project and report

- Deadline: 2007.03.01
- Compressed all files, which should include
  1) Descriptions of your work: name, student number, master or Ph.D student, grade, programming environment, report topic, etc.
  2) Source codes and report
  3) File format: GM_ChineseName_StudentNum.rar
- Send email to: zhx at cad . zju . edu . cn

- Sincerely welcome comments on GM course to

  {jqfeng, zhx} at cad . zju . edu. cn

Thanks