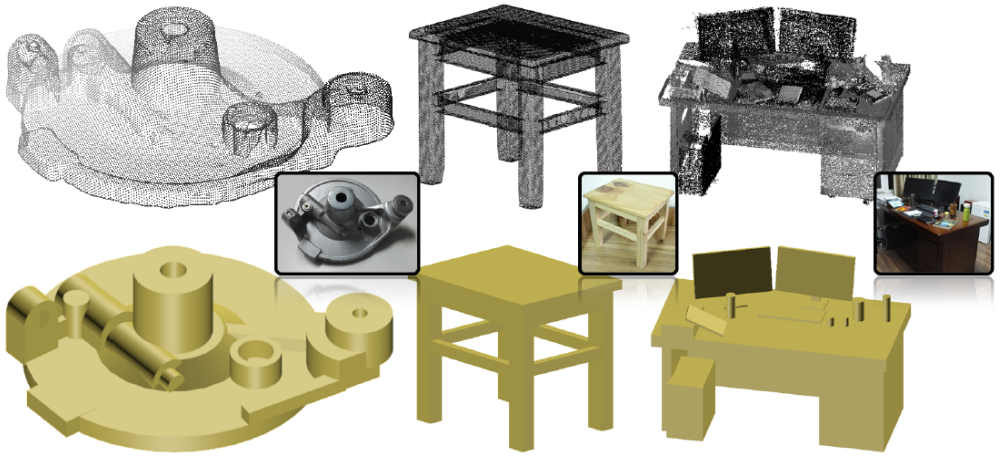
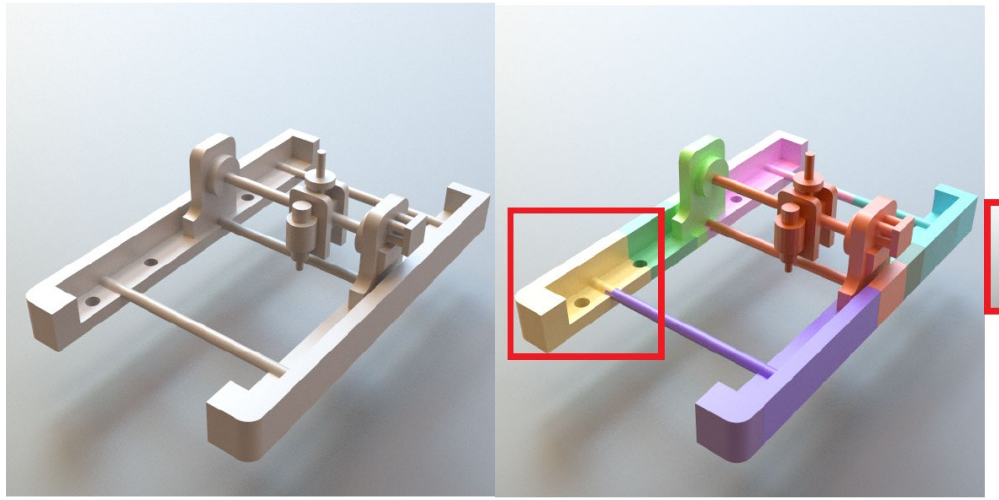


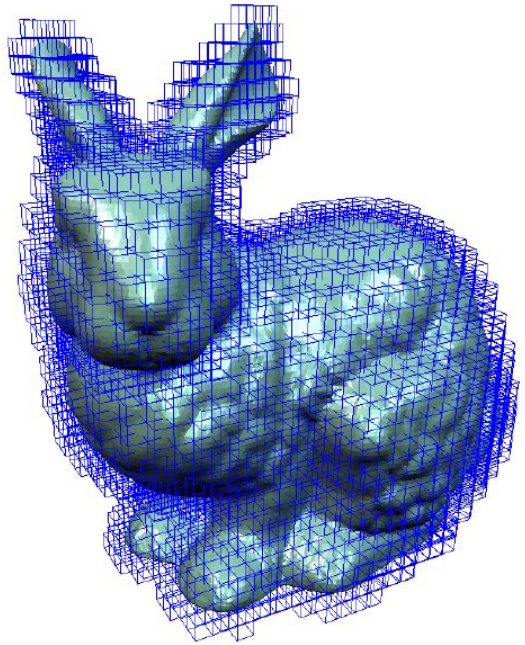


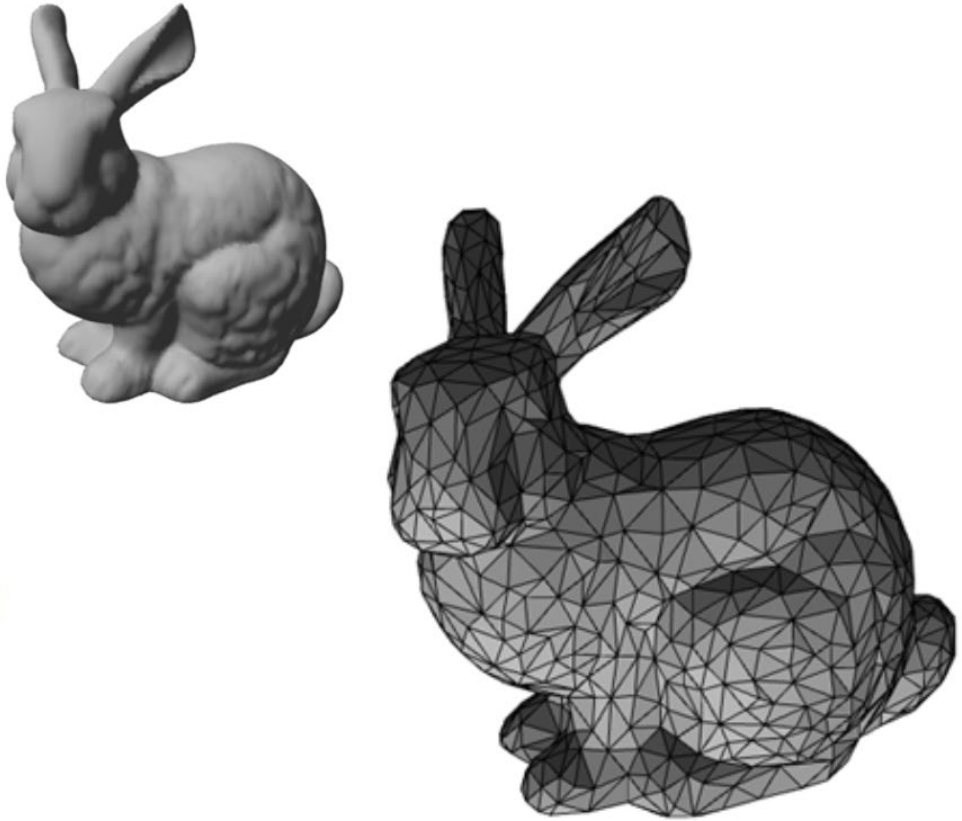
Inverse Constructive Solid Geometry (CSG) Modeling

Tutorial 4, COMP0119

Surface Representations

- Explicit vs. Implicit
  - Triangle meshes
  - Signed distance fields

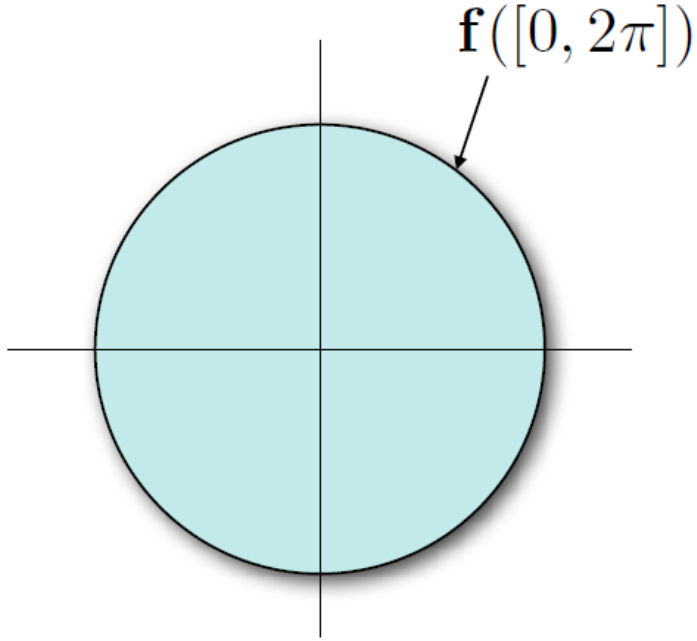




Surface Representations

- Explicit
  - Range of parameterization function

$$\mathbf{f}(x) = (r\cos(x), r\sin(x))^T$$



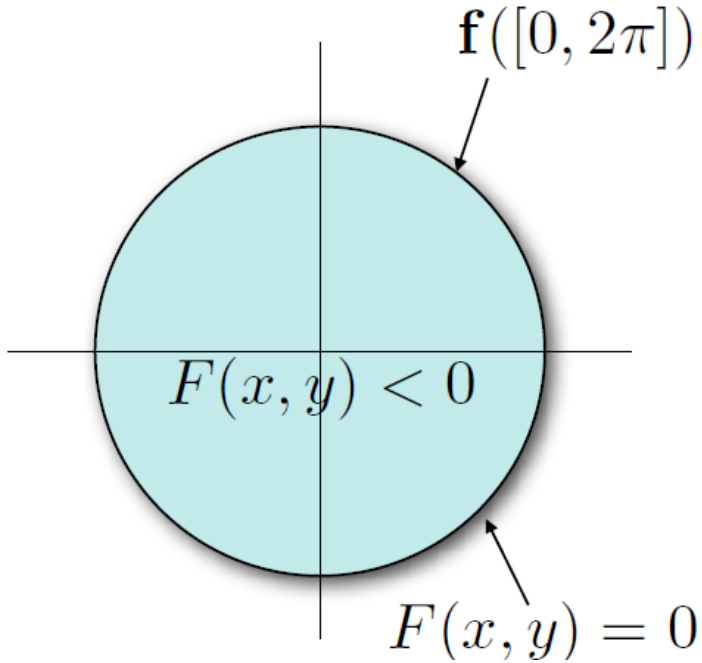
Surface Representations

- Explicit
  - Range of parameterization function

$$\mathbf{f}(x) = (r\cos(x), r\sin(x))^T$$

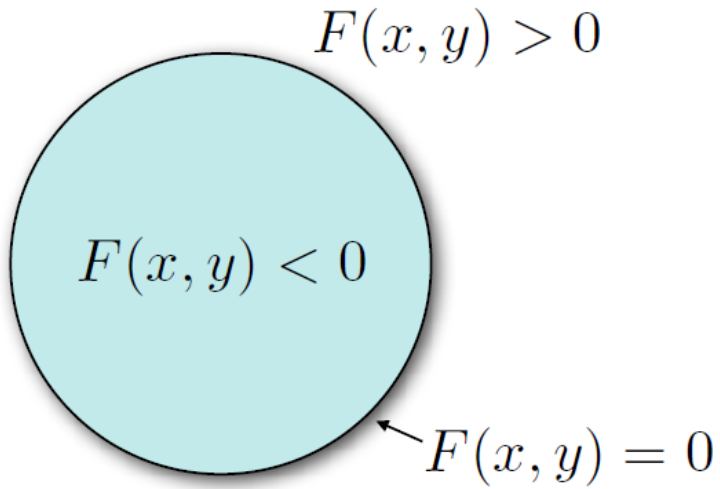
- Implicit
  - Kernel of implicit function

$$F(x,y) = \sqrt{x^2 + y^2} - r$$



Implicit Representations

- General implicit function:
  - Interior: $F(x, y, z) < 0$
  - Exterior: $F(x, y, z) > 0$
  - Surface: $F(x, y, z) = 0$



- Special case
  - Signed distance function (SDF)
  - Gradient ∇F is unit surface normal

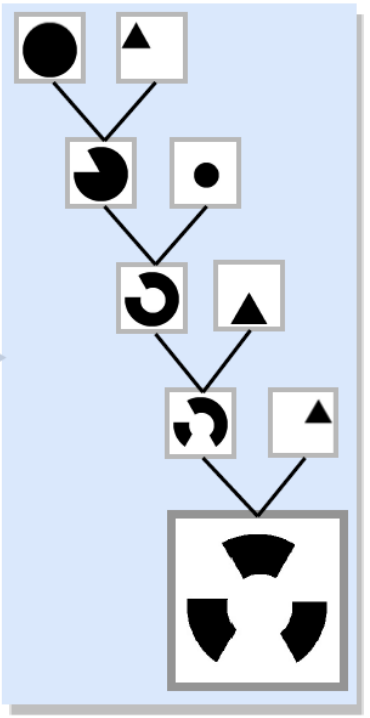
- Very suitable representation for
  - *Constructive Solid Geometry* (CSG)

Constructive Solid Geometry

- Formal Definition
  - CSG encodes geometries as trees that are constructed by recursively applying *Boolean operators* to *primitive shapes* [Requicha and Rossignac 1992].

P1 = Circle1
P2 = Triangle1
E1 = Subtract(P1, P2)
P3 = Circle2
E2 = Subtract(E1, P3)
P4 = Triangle2
E3 = Subtract(E2, P4)
P5 = Triangle3
Out = Subtract(E3, P5)

Program



Parse tree

Exchangeable

Constructive Solid Geometry

- Primitive shapes
  - sphere
  - cylinder
  - cone
  - pyramid
  - Cube
  - Box

Implicit representation

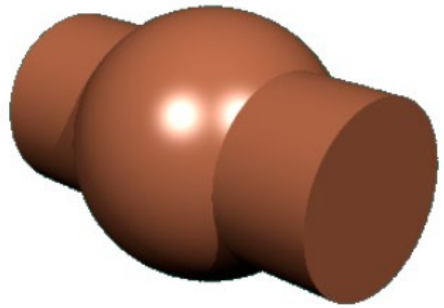
- Do not contain: points, lines, planes

Constructive Solid Geometry

Operations

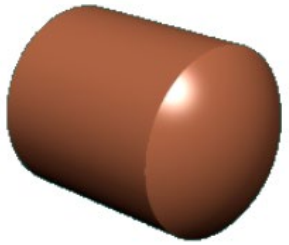
- Union

$$F_{C \cup S}(\cdot) = \min \{F_C(\cdot), F_S(\cdot)\}$$



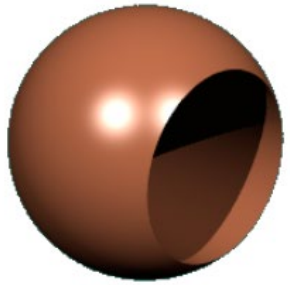
- Intersection

$$F_{C \cap S}(\cdot) = \max \{F_C(\cdot), F_S(\cdot)\}$$



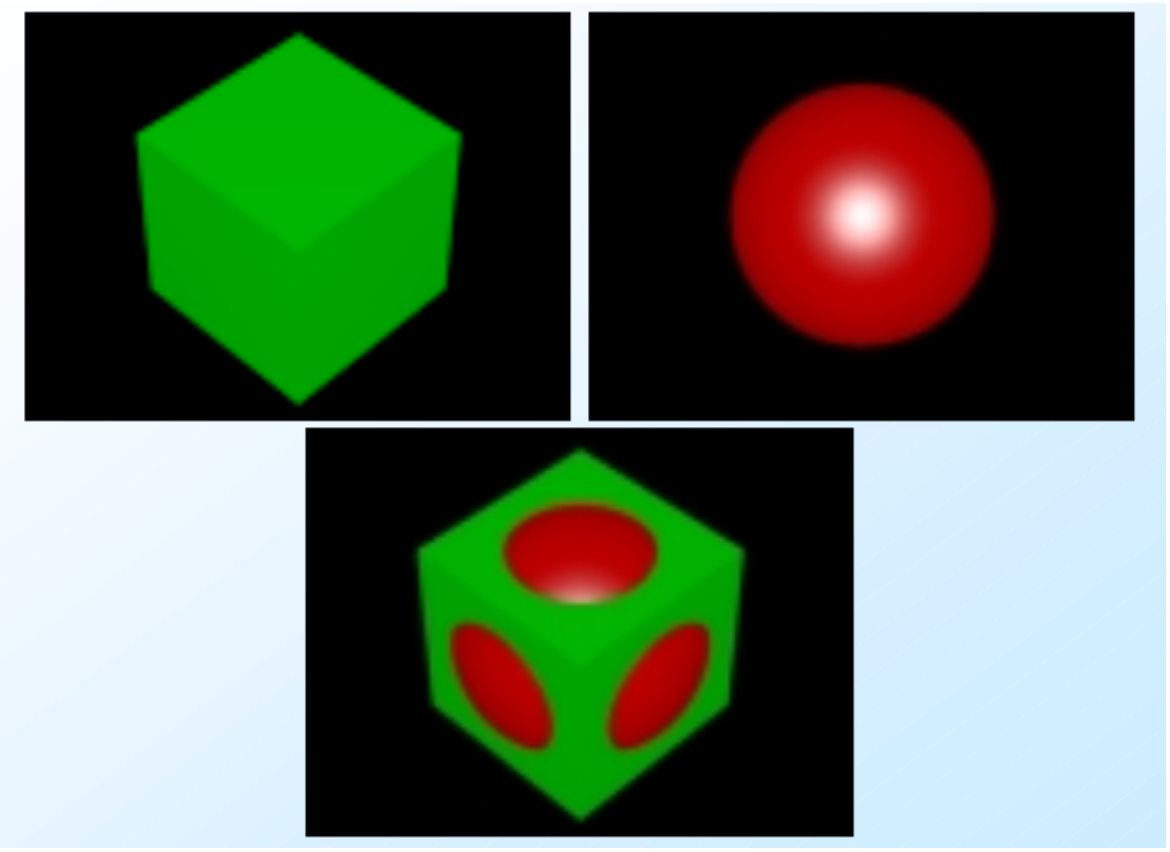
- Difference

$$F_{S \setminus C}(\cdot) = \max \{-F_C(\cdot), F_S(\cdot)\}$$



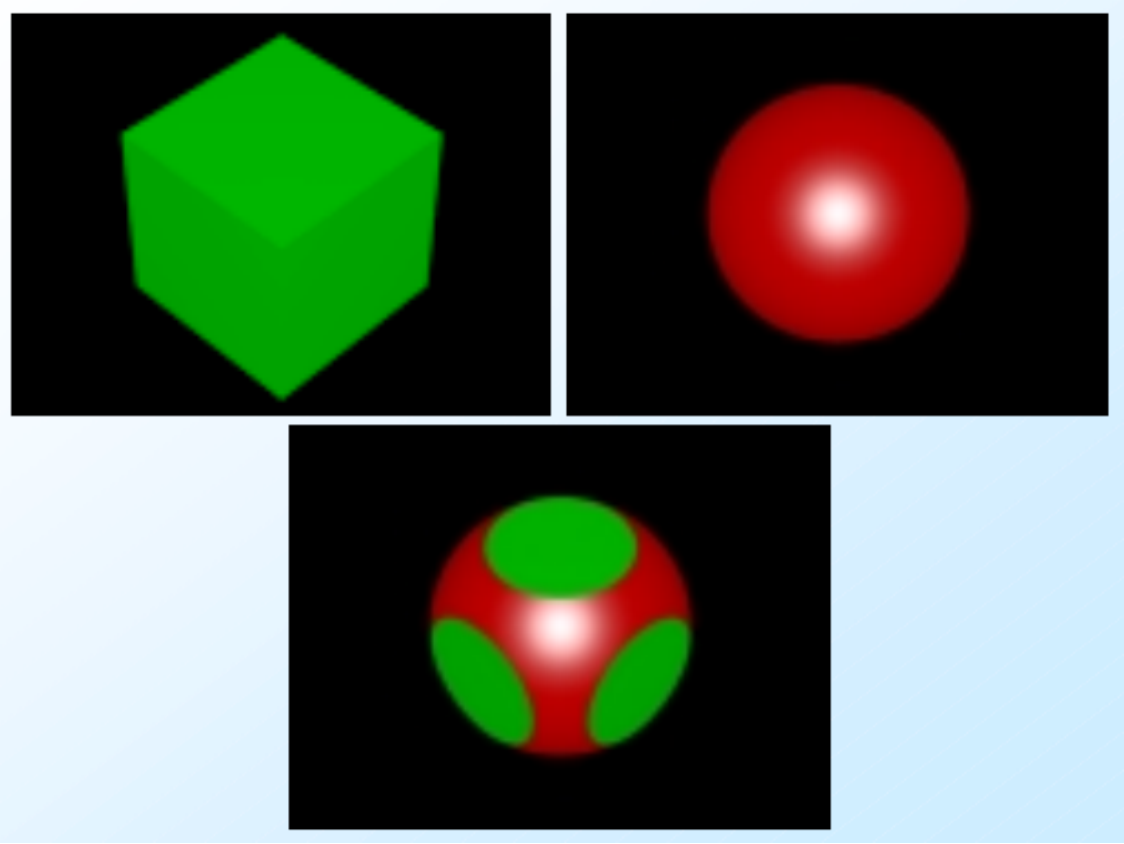
Constructive Solid Geometry

- Box
- Sphere

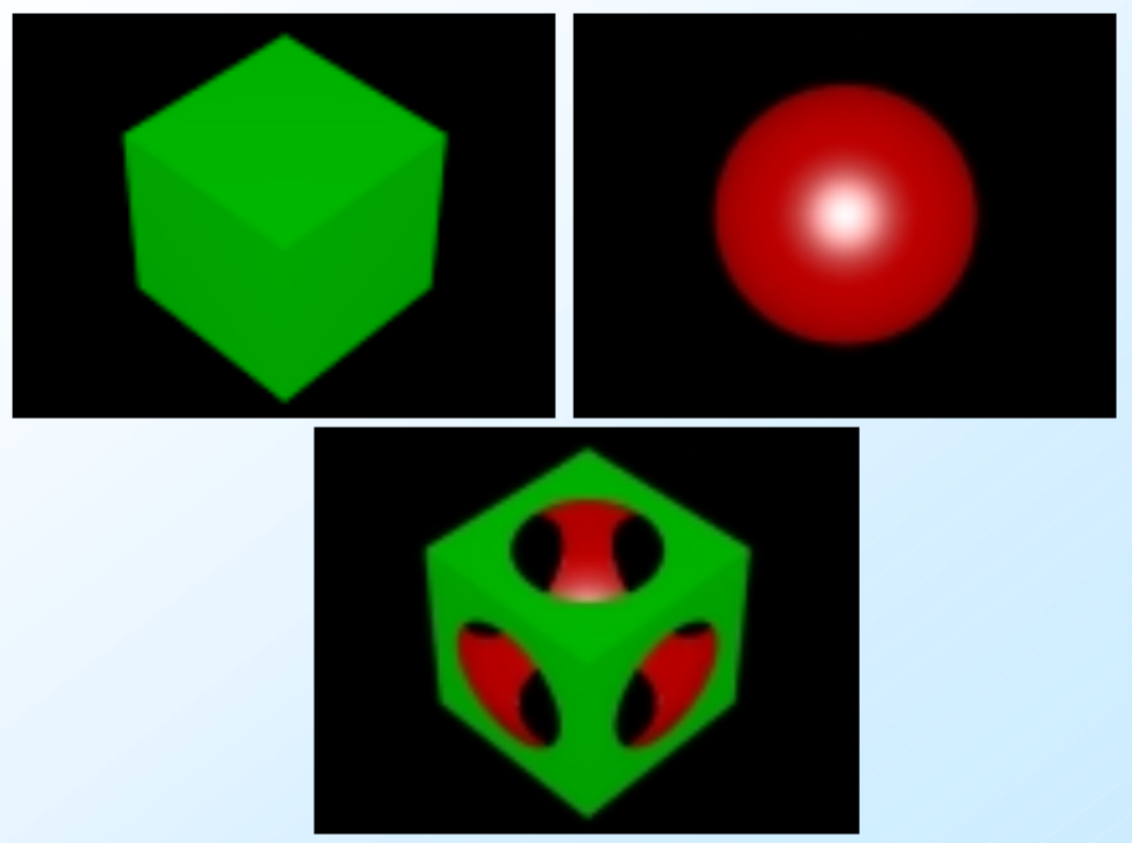


Union

Constructive Solid Geometry

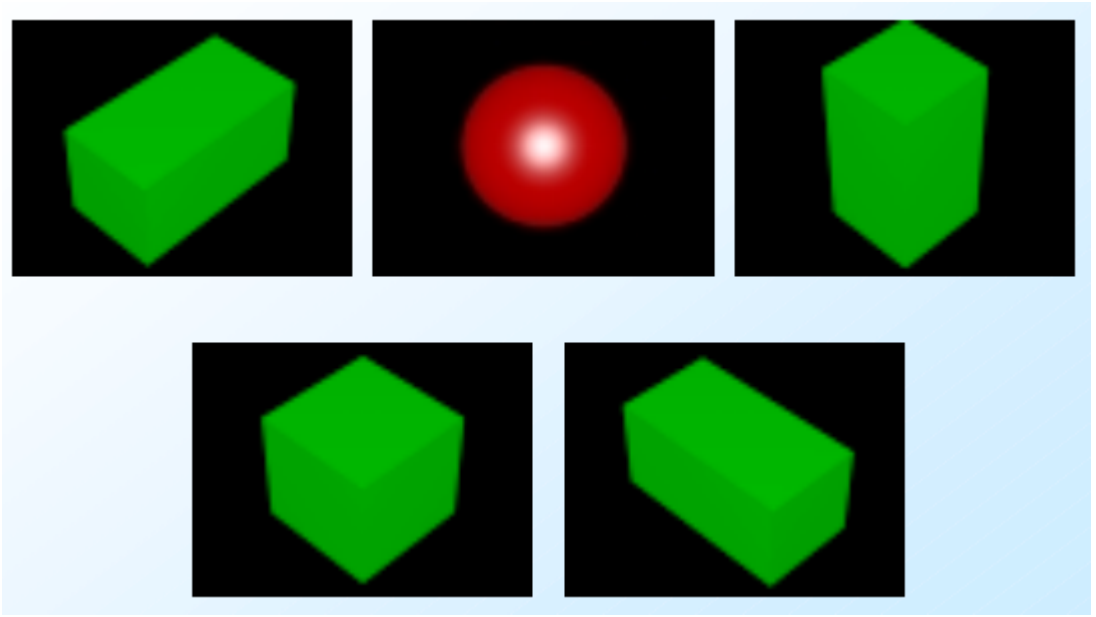


Intersection

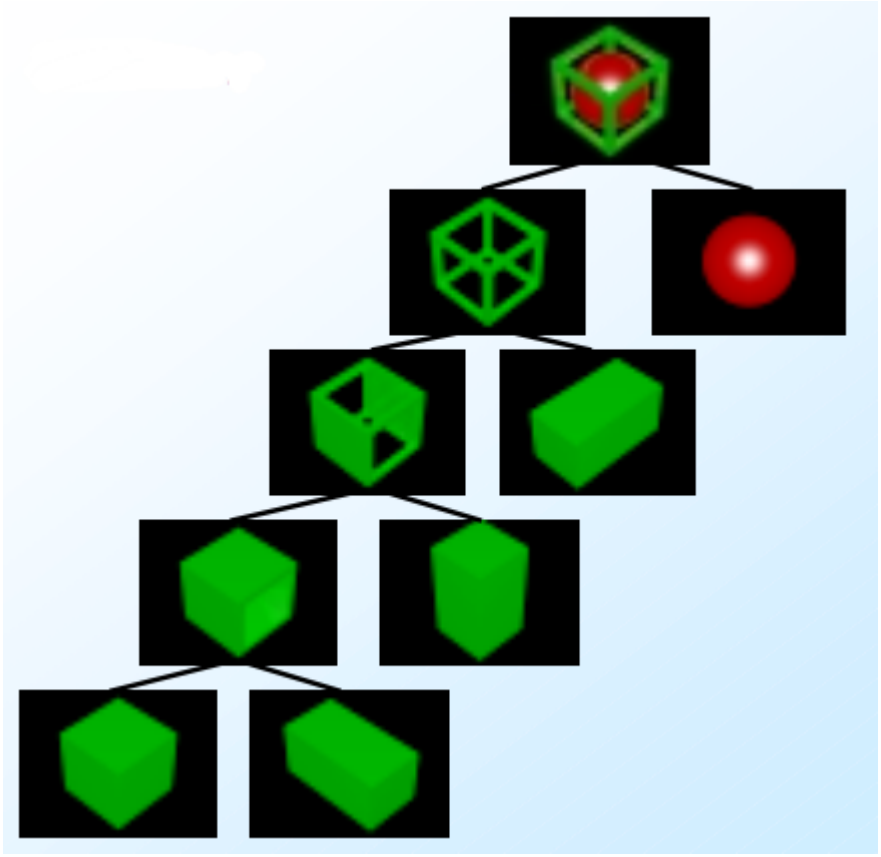


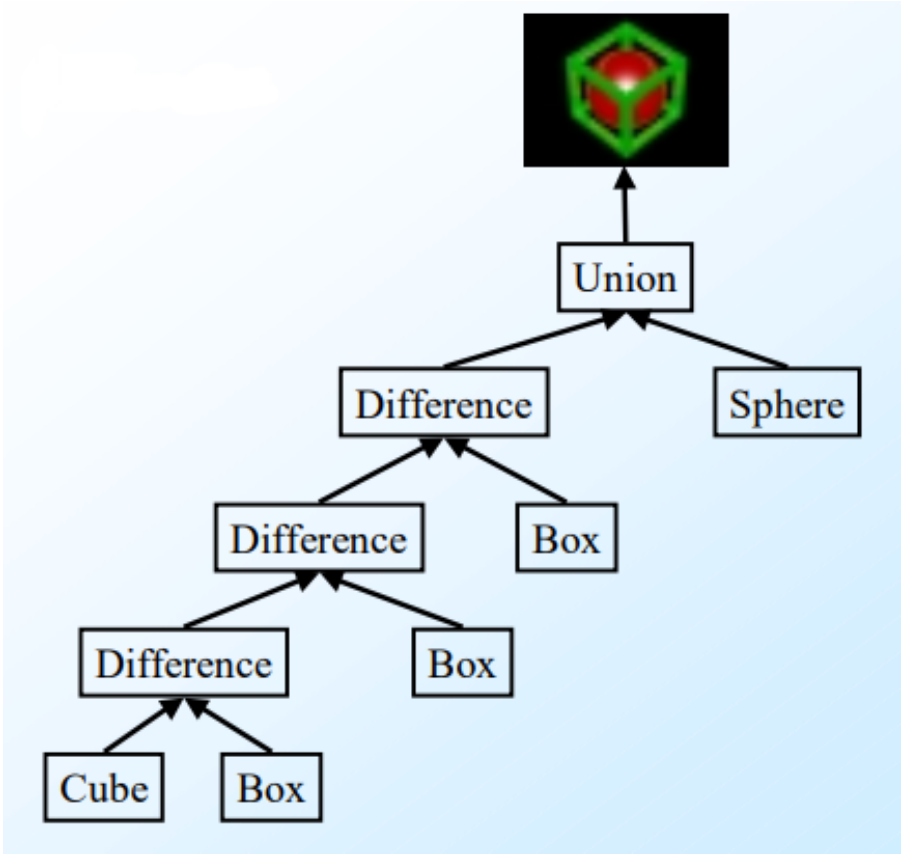
Difference

CSG Tree



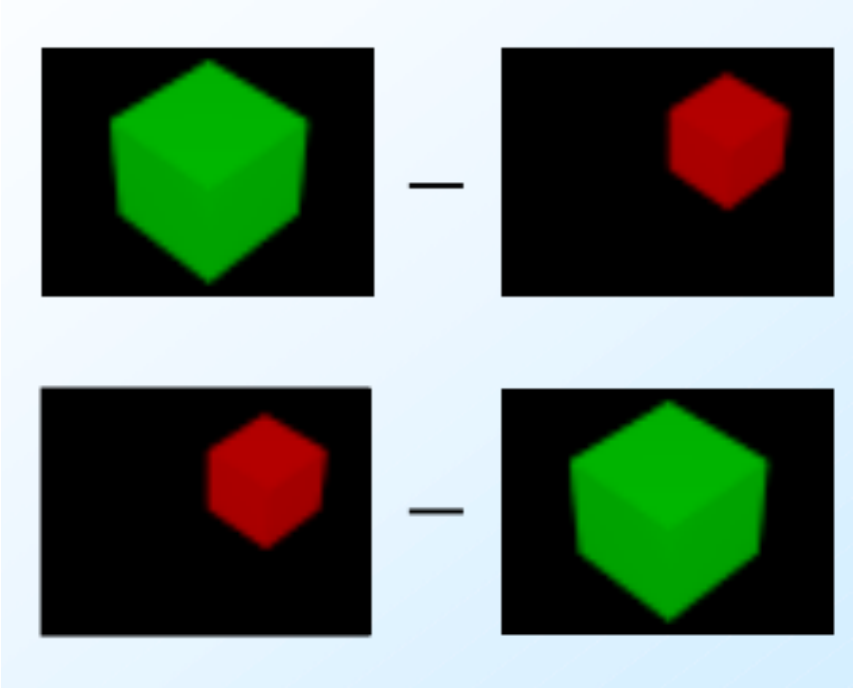
CSG Tree




Union
Difference
Sphere
Difference
Box
Difference
Box
Cube
Box

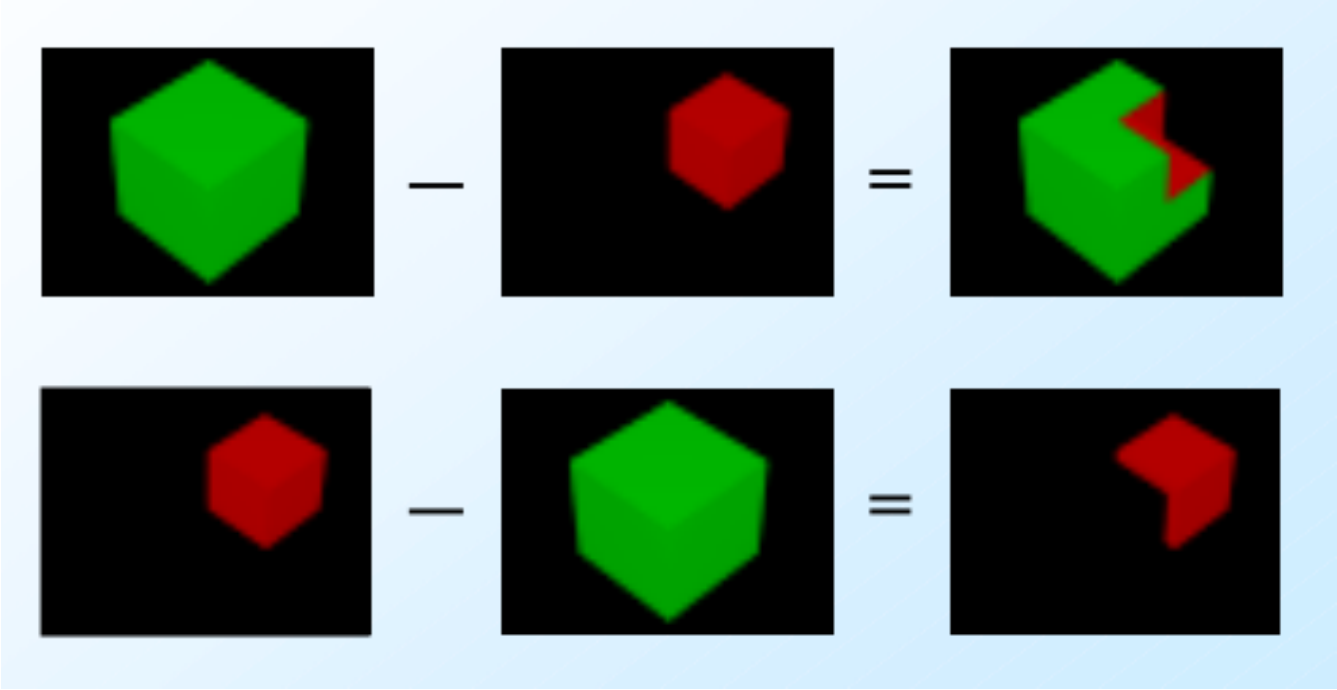
CSG Properties

- *Commutative ?*



CSG Properties

- Not all CSG operations are *commutative*:

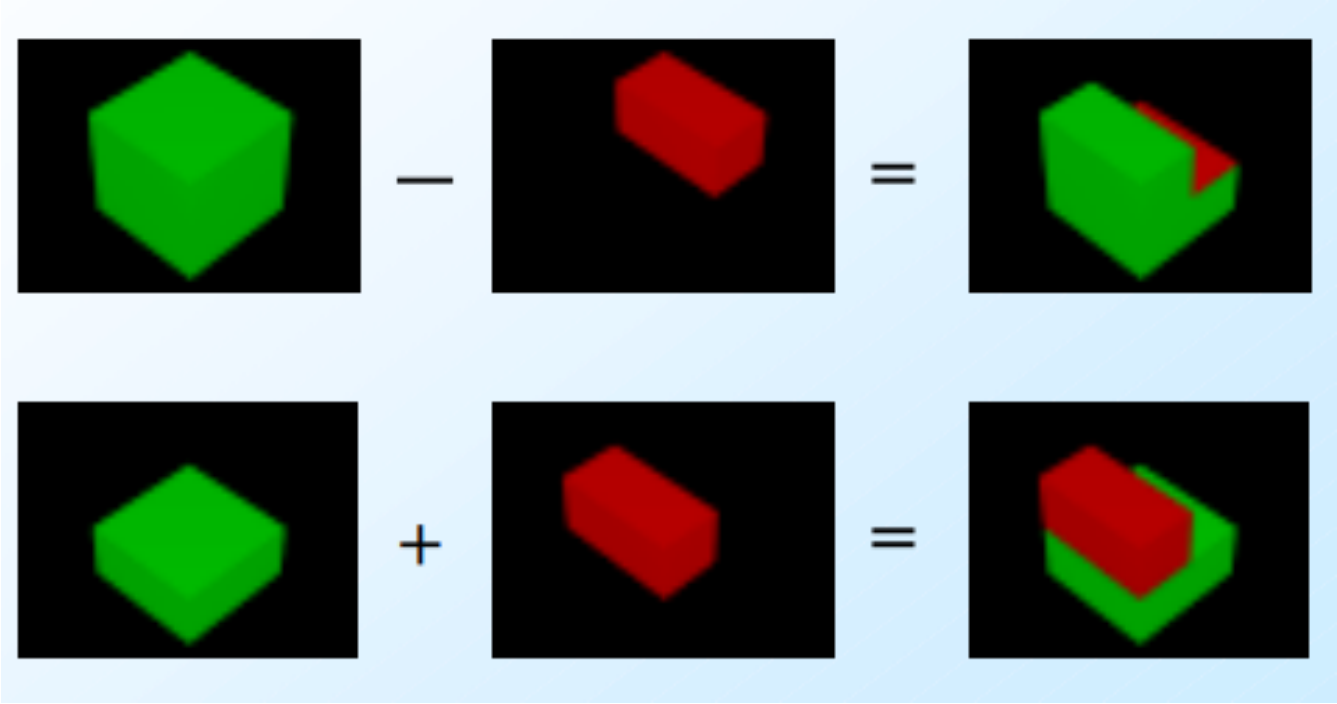


CSG Properties

- CSG operations are *unique?*

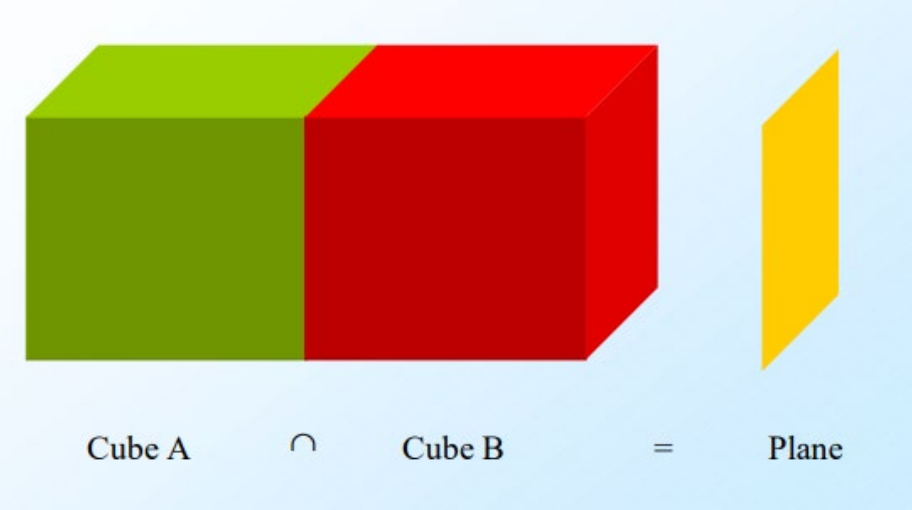
CSG Properties

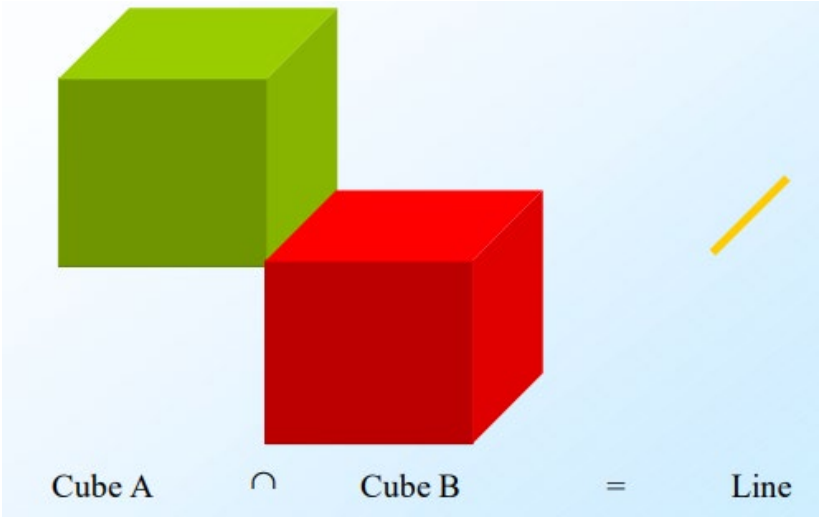
- CSG operations are *not unique*:

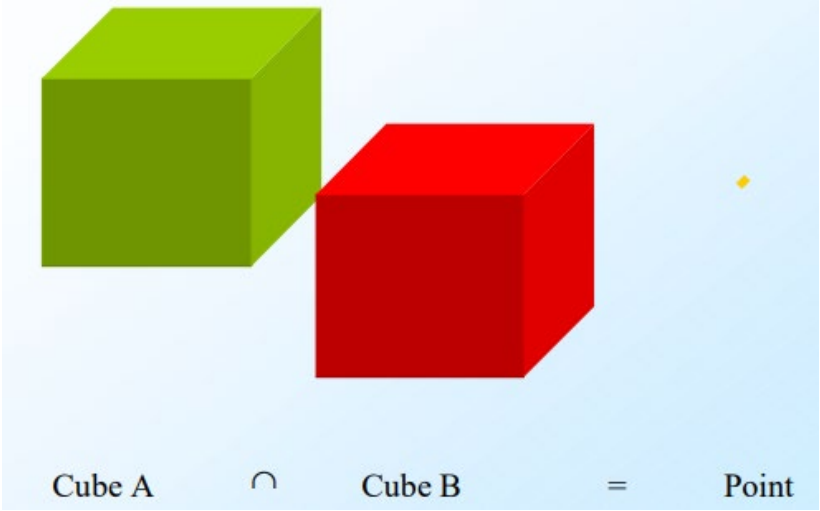


CSG Problems

- Unsupported primitives







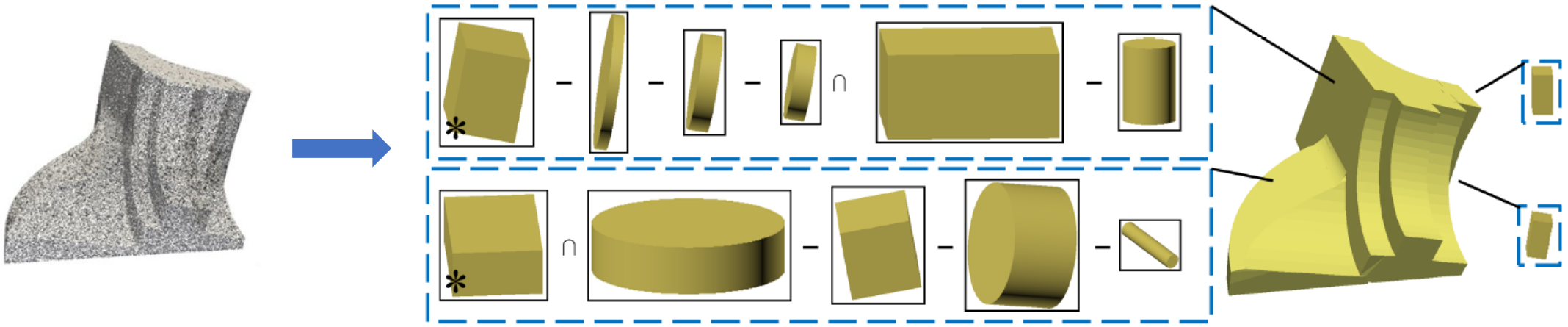
CSG

- Expressive and concise rep.
- Compact rep. to store
- Easy to edit

- Widely used in
  - computer aided design (CAD), e.g., Blender, 3Ds Max, etc.
  - Computer assisted manufacturing

Inverse CSG

- Input: 2D graphical shape or 3D surface/point cloud/voxel
- Output: resulting CSG model and the expression tree



Two Contemporary Papers

SGP 2018

Constructing 3D CSG Models from 3D Raw Point Clouds

Q. Wu[1], K. Xu[2] and J. Wang[†1]

[1] Nanjing University of Aeronautics and Astronautics, China
[2]National University of Defense Technology, China

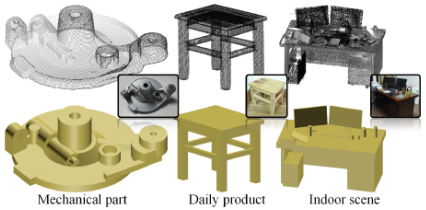
Abstract

The Constructive Solid Geometry (CSG) tree, encoding the generative process of an object by a recursive compositional structure of bounded primitives, constitutes an important structural representation of 3D objects. Therefore, automatically recovering such a compositional structure from the raw point cloud of an object represents a high-level reverse engineering problem, finding applications from structure and functionality analysis to creative redesign. We propose an effective method to construct CSG models and trees directly over raw point clouds. Specifically, a large number of hypothetical bounded primitive candidates are first extracted from raw scans, followed by a carefully designed pruning strategy. We then choose to approximate the target CSG model by the combination of a subset of these candidates with corresponding Boolean operations using a binary optimization technique, from which the corresponding CSG tree can be derived. Our method attempts to consider the minimal description length concept in the point cloud analysis setting, where the objective function is designed to minimize the construction error and complexity simultaneously. We demonstrate the effectiveness and robustness of our method with extensive experiments on real scan data with various complexities and styles.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Shape Modeling—Procedural modeling

1. Introduction

Constructing 3D models from raw point clouds is one of the most fundamental tasks in computer aided design and computer graphics. The ultimate goals encompass three aspects: reproduction, quality control, and redesign and modification applications [LMM04]. Typically, different goals require different degrees of recovering. For the third aspect, some high-level design information embedded in the data points, such as the geometric structures and the logic of construction, is desired, which is the focus of our work. More precisely, to support a higher level of interaction with constructions,



CVPR 2018

CSGNet: Neural Shape Parser for Constructive Solid Geometry

Gopal Sharma Rishabh Goyal Difan Liu Evangelos Kalogerakis Subhransu Maji
University of Massachusetts, Amherst
{gopalsharma,risgoyal,dliu,kalo,smaji}@cs.umass.edu

Abstract

We present a neural architecture that takes as input a 2D or 3D shape and outputs a program that generates the shape. The instructions in our program are based on constructive solid geometry principles, i.e., a set of boolean operations on shape primitives defined recursively. Bottom-up techniques for this shape parsing task rely on primitive detection and are inherently slow since the search space over possible primitive combinations is large. In contrast, our model uses a recurrent neural network that parses the input shape in a top-down manner, which is significantly faster and yields a compact and easy-to-interpret sequence of modeling instructions. Our model is also more effective as a shape detector compared to existing state-of-the-art detection techniques. We finally demonstrate that our network can be trained on novel datasets without ground-truth program annotations through policy gradient techniques.

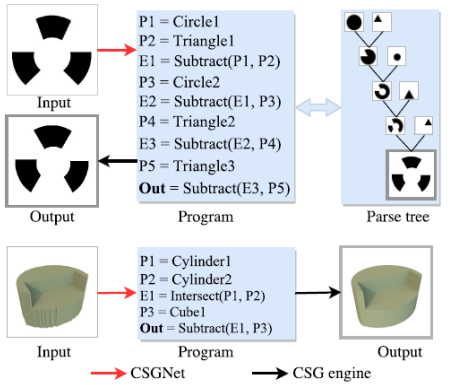
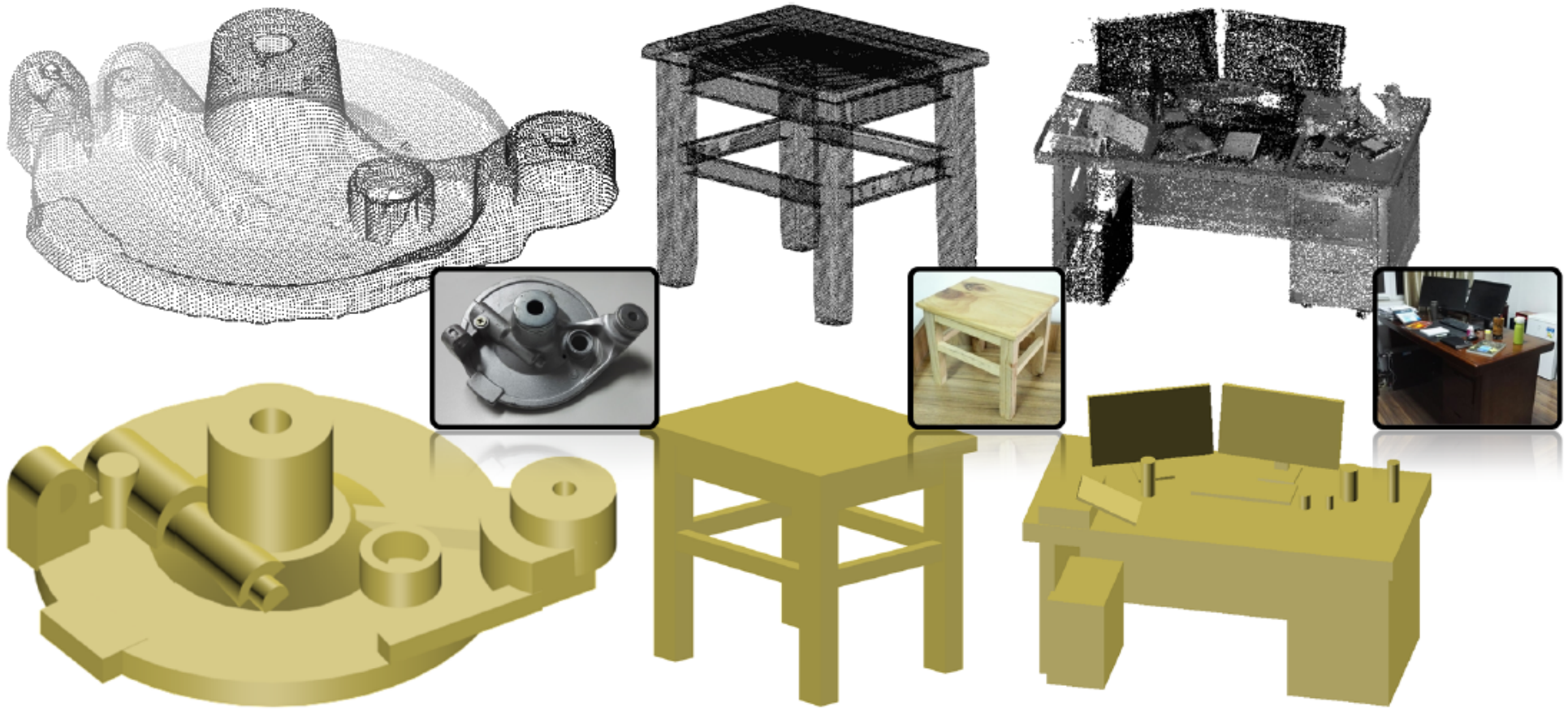


Figure 1. **Our shape parser produces a compact program that generates an input 2D or 3D shape**. On top is an input image of 2D shape, its program and the underlying parse tree where primitives are combined with boolean operations. On the bottom is an input voxelized 3D shape, the induced program, and the resulting shape from its execution.

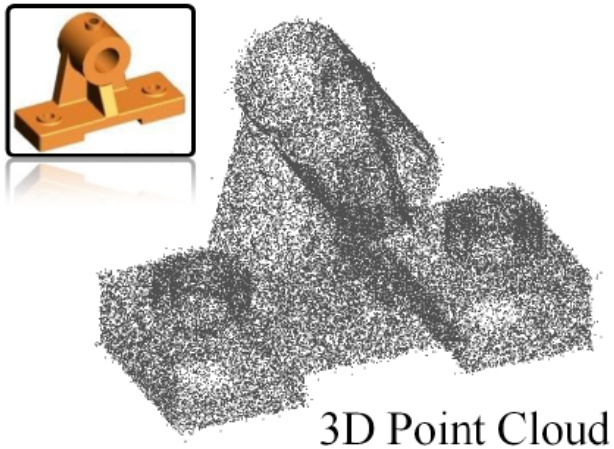


Constructing 3D CSG Models from 3D Raw Point Clouds

Wu, Qiaoyun, Kai Xu, and Jun Wang. "Constructing 3D CSG models from 3D raw point clouds." In *Computer Graphics Forum*, vol. 37, no. 5, pp. 221-232. 2018.

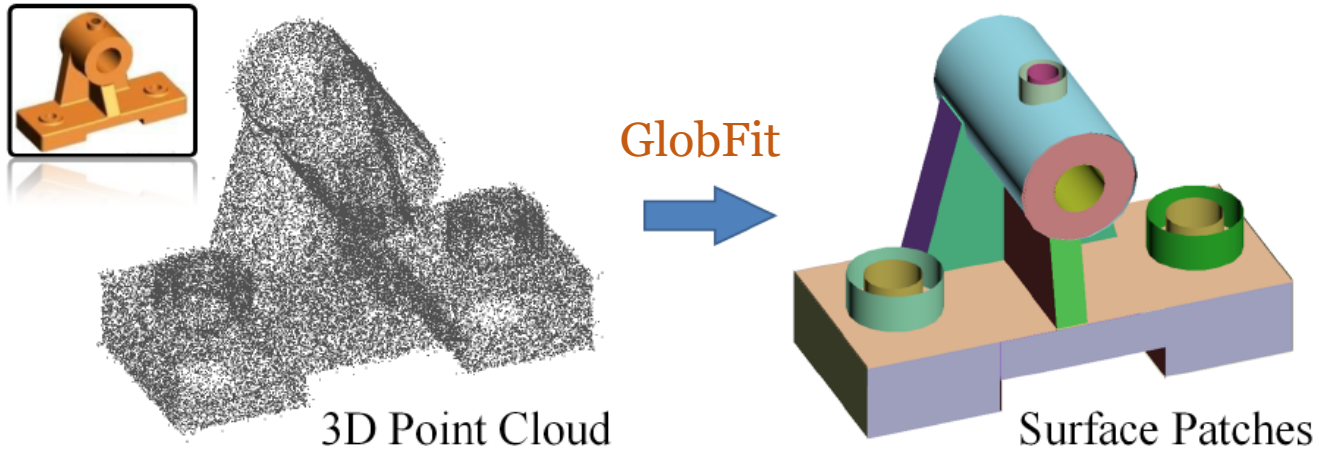
Challenges (C) and Key Ideas (K)

- C1 and K1
  - Point clouds are unstructured without well-defined topology



Challenges (C) and Key Ideas (K)

- C1 and K1
  - Point clouds are unstructured without well-defined topology
  - Construct surface patches using GlobFit



Challenges (C) and Key Ideas (K)

- C2 and K2
  - Order of primitives and Boolean operations matter

Primitives: A, B, C, D
Operators: ∩, ∪, −

$$A \cap B \cup C - D \neq B \cap D - C \cup D$$

Challenges (C) and Key Ideas (K)

- C2 and K2
  - Order of primitives and Boolean operations matter
  - Bottom-up: subtree construction, full construction
    - Divide the operators into two groups

Primitives: A, B, C, D
Operators: ∩, ∪, −

$$A \cap B \cup C - D \neq B \cap D - C \cup D$$

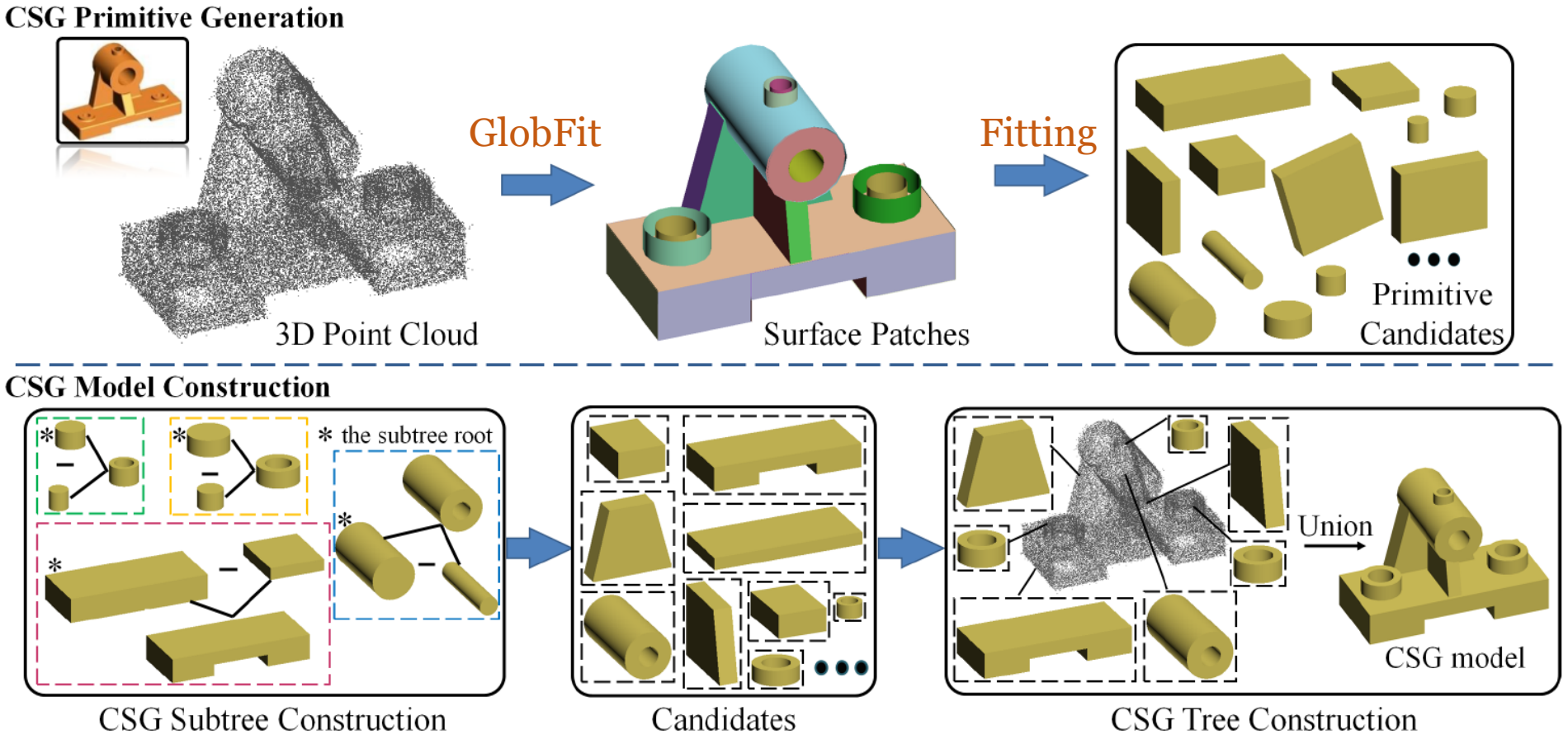
Challenges (C) and Key Ideas (K)

- C3 and K3
  - Use as less as possible primitives
  - Optimization balancing accuracy and complexity

$$A \cap B \cup C - D - E \cup F = H \cup I - J$$

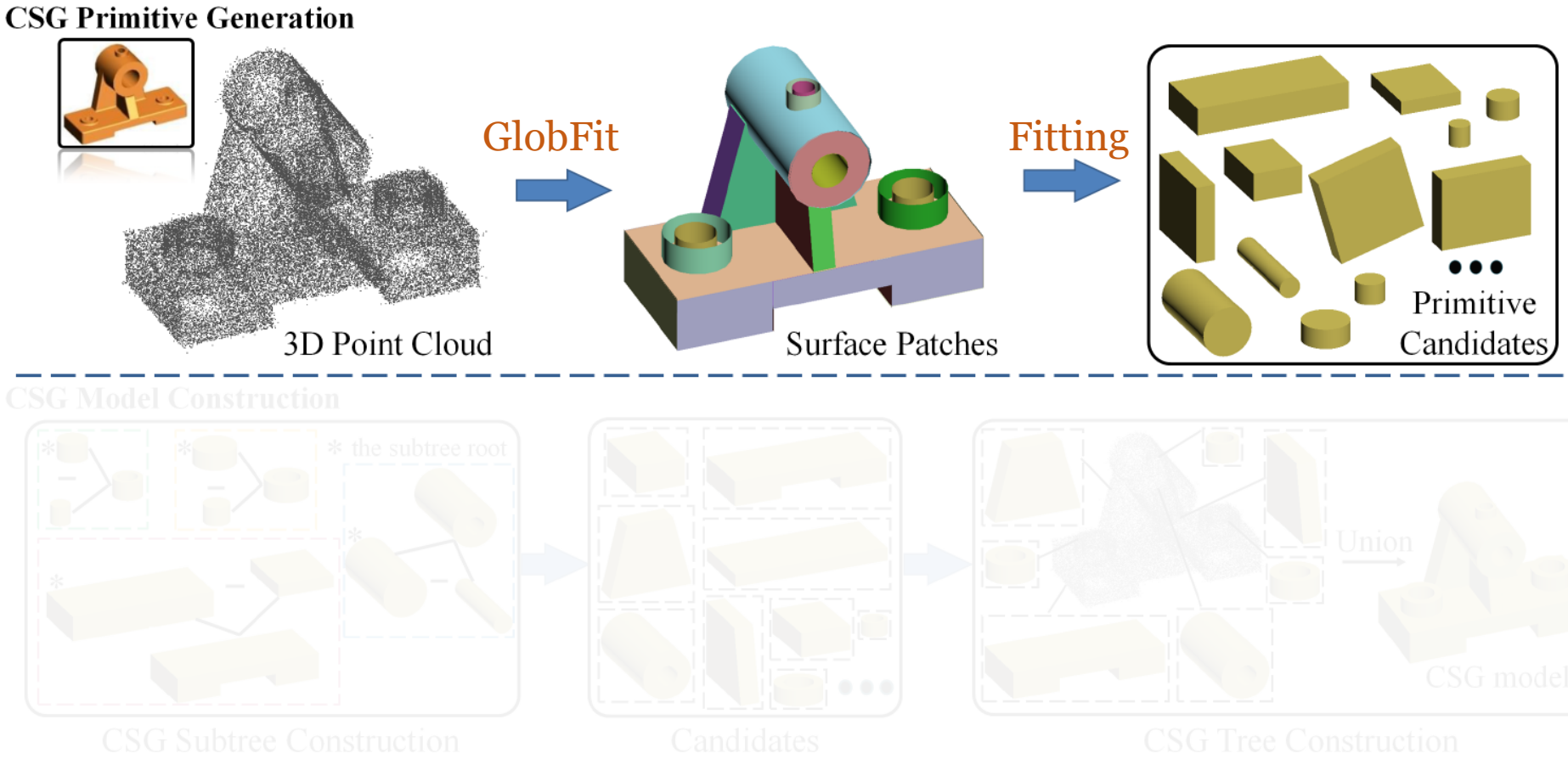
Pipeline

Two stages:
- CSG Primitive Generation
- CSG Model Construction

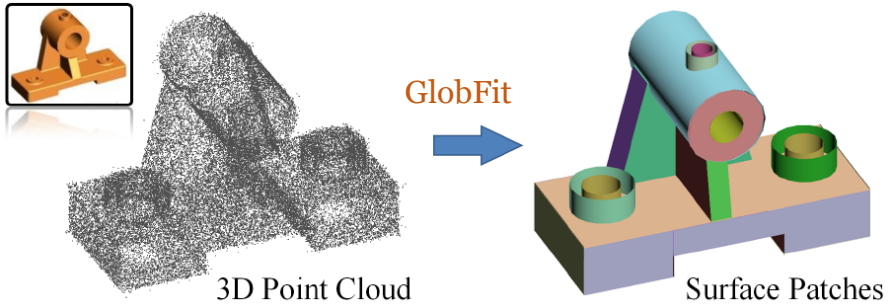


Pipeline

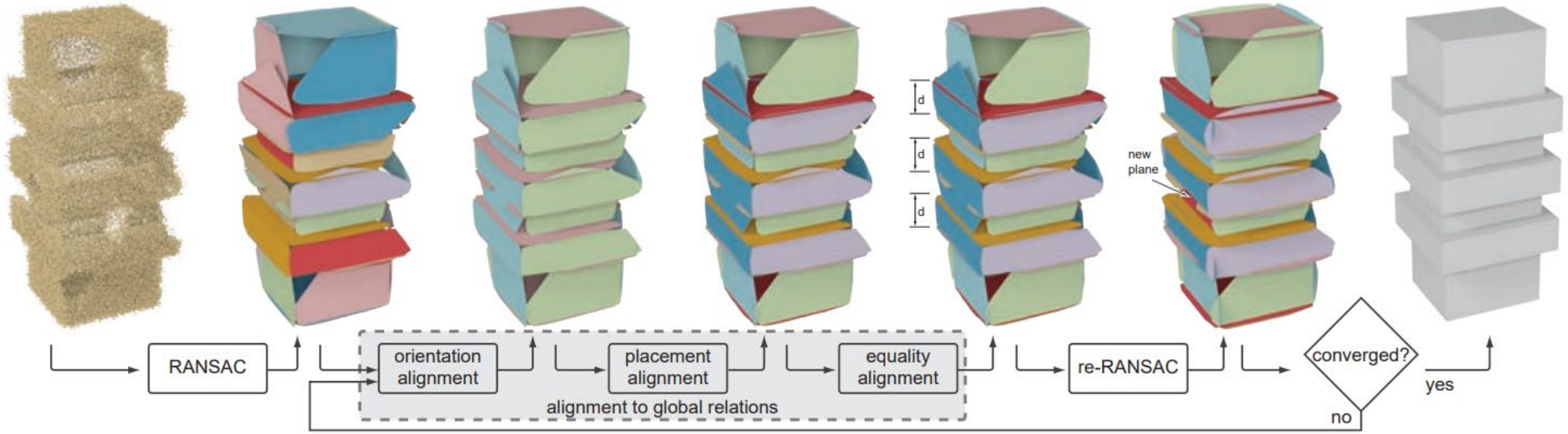
Two stages:
- CSG Primitive Generation
- CSG Model Construction



CSG Primitive Generation

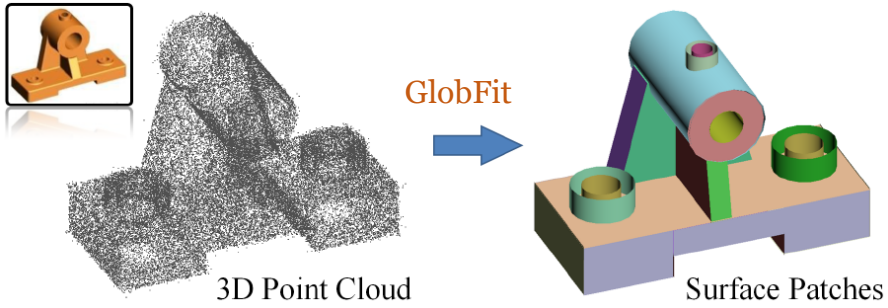


- Surface fitting – GlobFit [1]

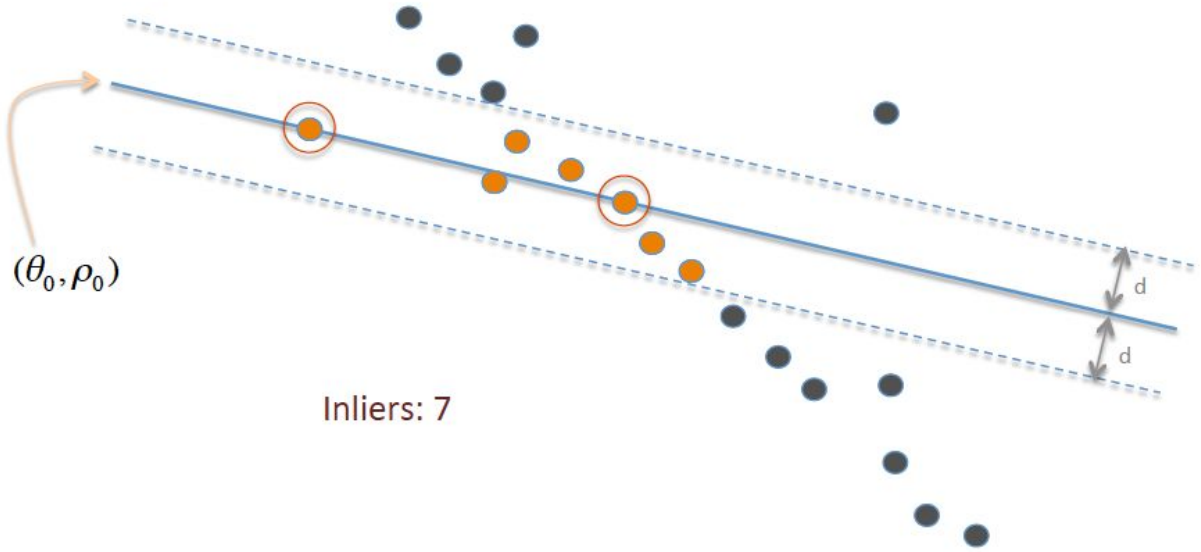


[1] Li, Yangyan, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. "Globfit: Consistently fitting primitives by discovering global relations." In *ACM SIGGRAPH 2011 papers*, pp. 1-12. 2011.

CSG Primitive Generation



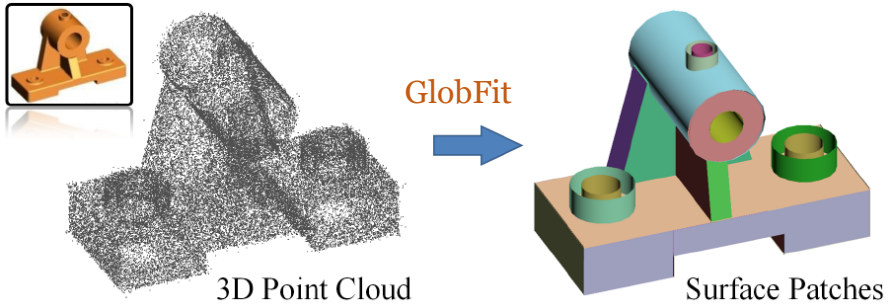
- RANSAC (Random Sample Consensus)
  - Outlier detection



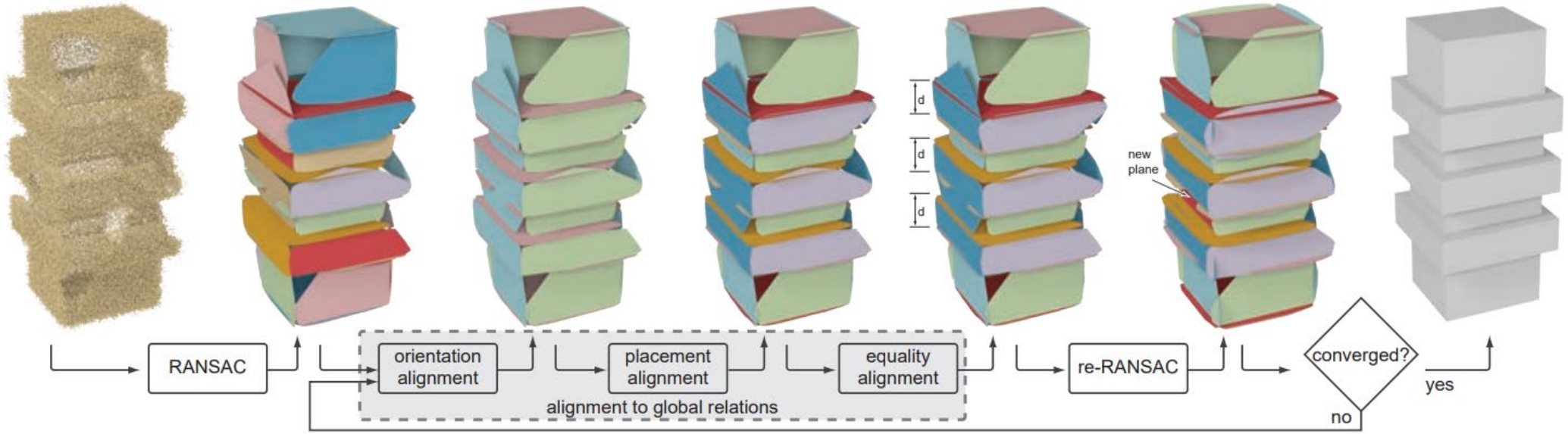
Input: points
- Inlier points
- Outlier points

Output: a line

CSG Primitive Generation



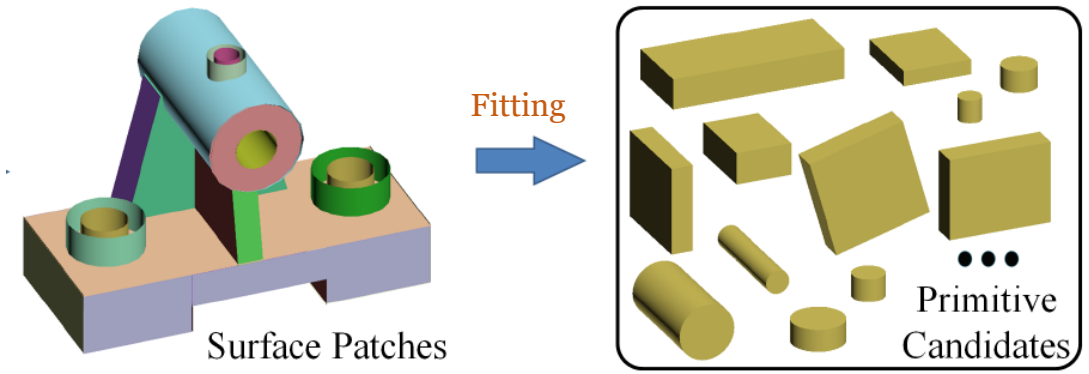
- Surface fitting – GlobFit [1]



[1] Li, Yangyan, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. "Globfit: Consistently fitting primitives by discovering global relations." In *ACM SIGGRAPH 2011 papers*, pp. 1-12. 2011.

CSG Primitive Generation

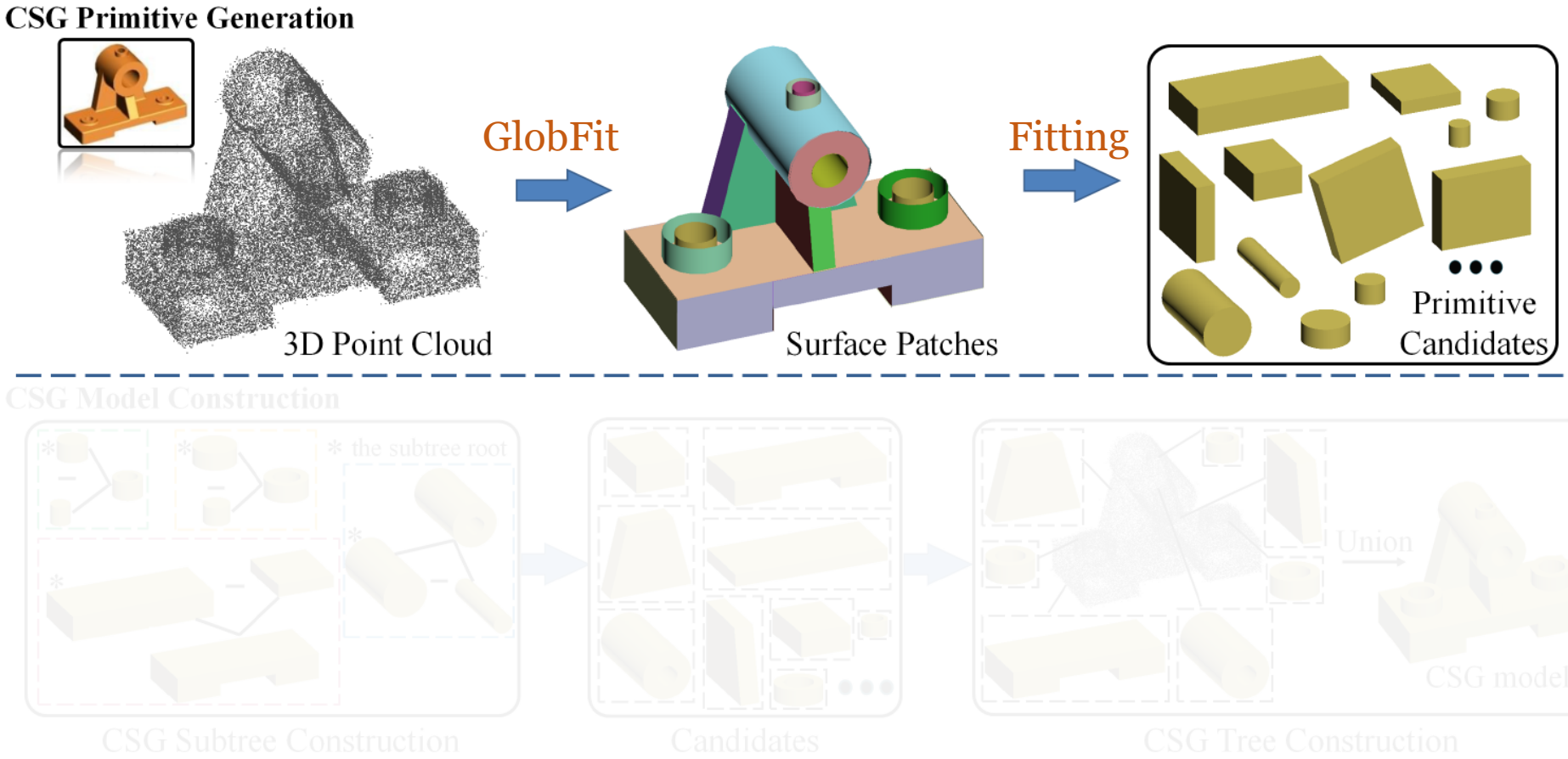
- Primitive Fitting



Primitives:
Cuboids,
Cylinders
Cones
Spheres
Tori

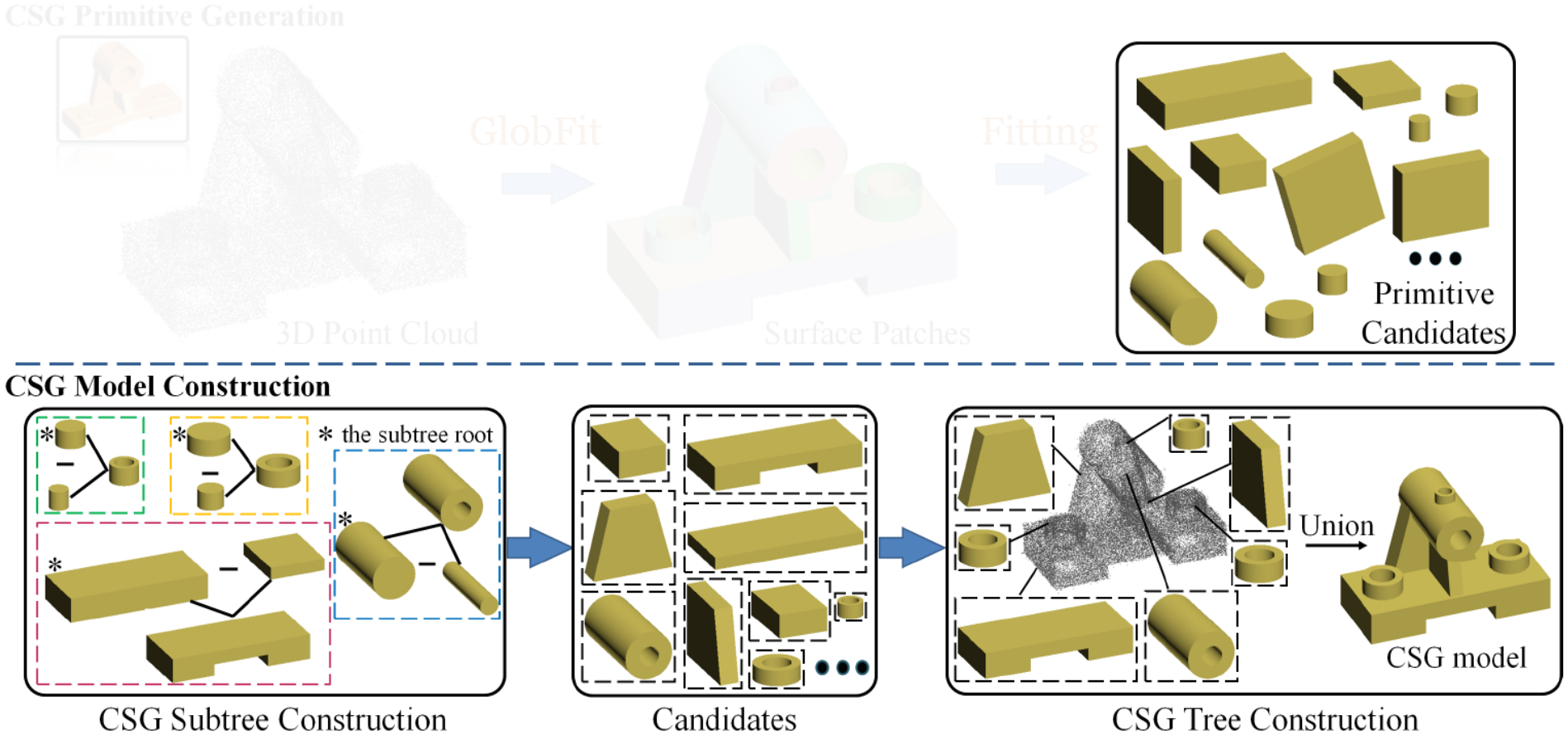
Pipeline

Two stages:
- CSG Primitive Generation
- CSG Model Construction



Pipeline

Two stages:
- CSG Primitive Generation
- CSG Model Construction



CSG Model Construction

Observations

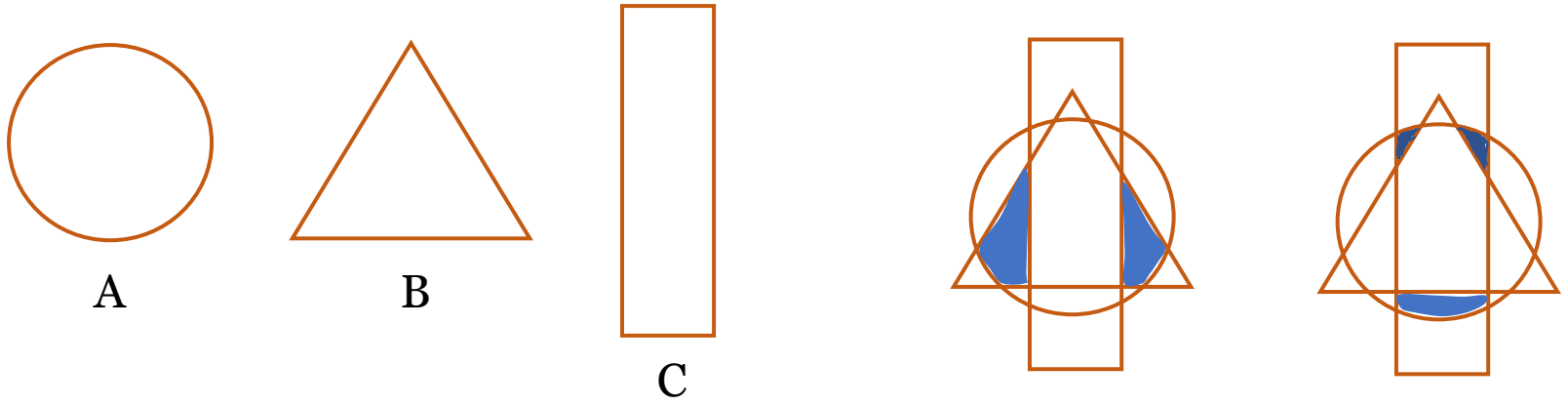
- Subtree construction
  - Union operation
    - Commutative and associative
    - The order does not matter!

    - $A \cup B \cup C \cup D = B \cup D \cup A \cup C = A \cup D \cup B \cup C = \ldots$

CSG Model Construction

Observations

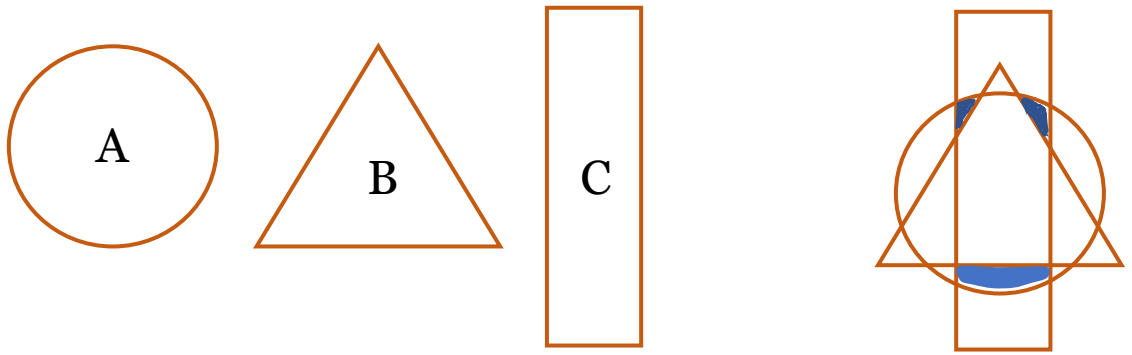
- Subtree construction
  - Union operation
    - Commutative and associative
    - The order does not matter!
  - Intersection and difference
    - The order matters, e.g., $A \cap B - C \neq C - B \cap A$



CSG Model Construction

Observations

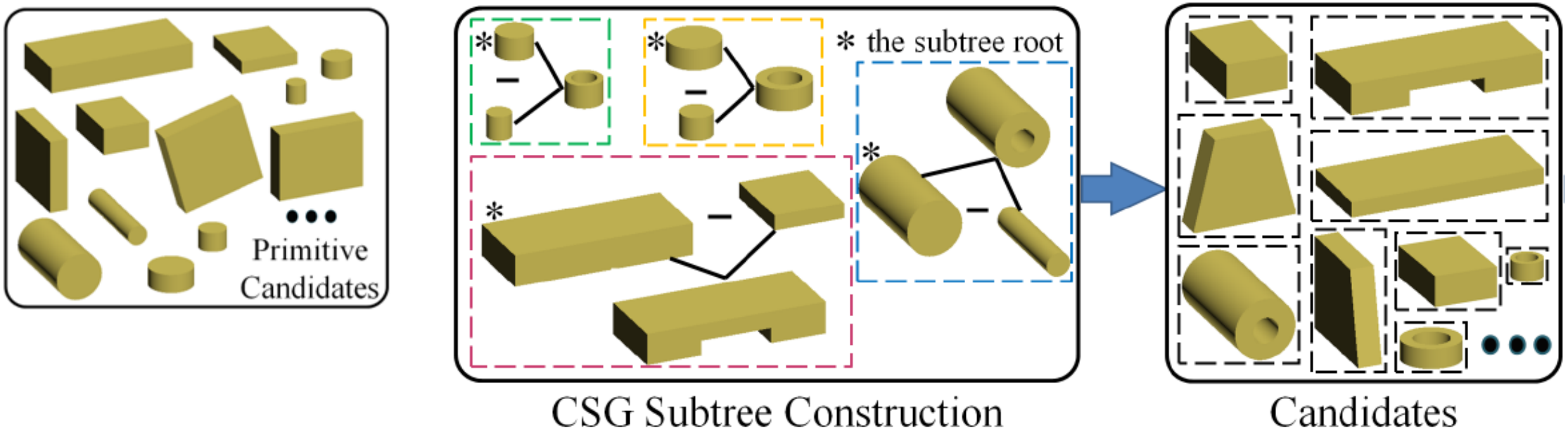
- Subtree construction
  - Union operation
    - Commutative and associative
    - The order does not matter!
  - Intersection and difference
    - The order matters, e.g., $A \cap B - C \neq C - B \cap A$
    - But, once the start operand is determined, the order dose not matter!
    - $A - B \cap C = A \cap C - B$



CSG Model Construction

- Subtree construction

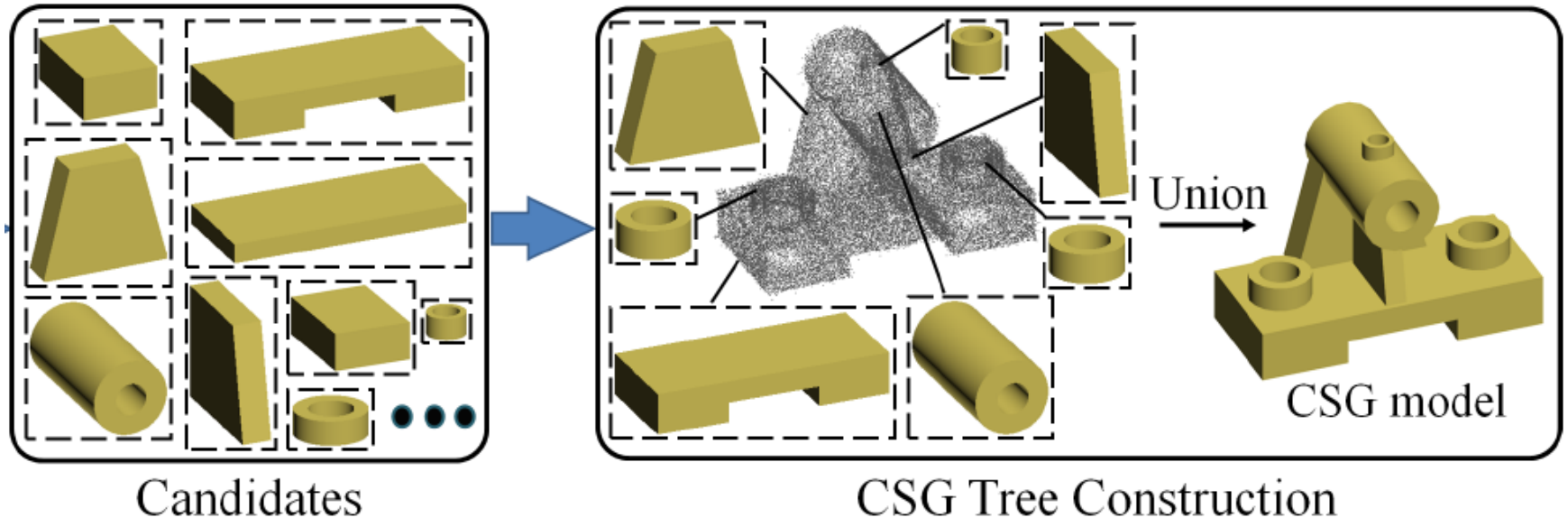
Intersection, difference, start operand



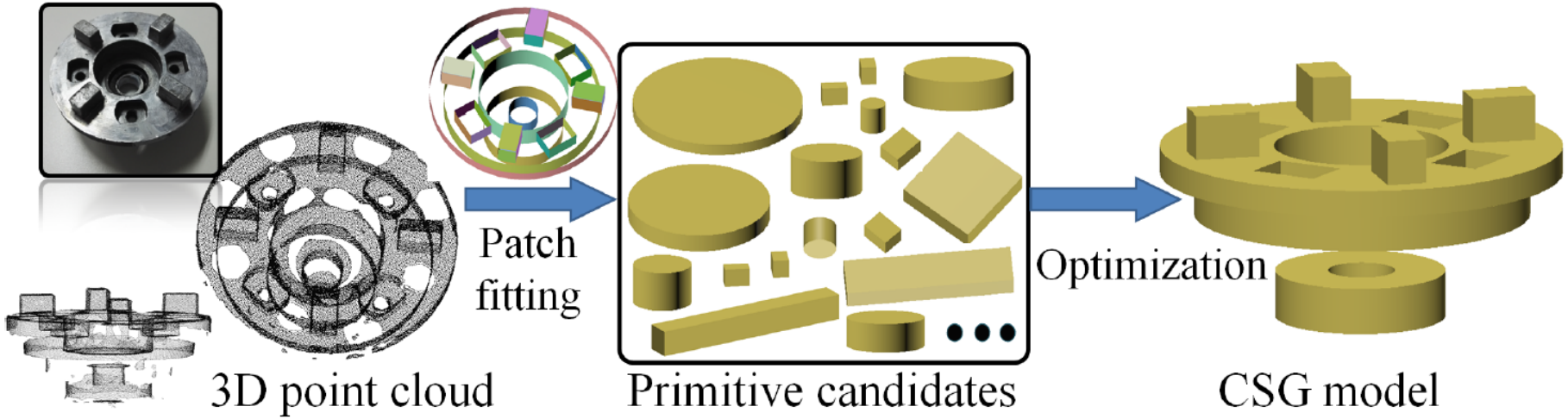
CSG Model Construction

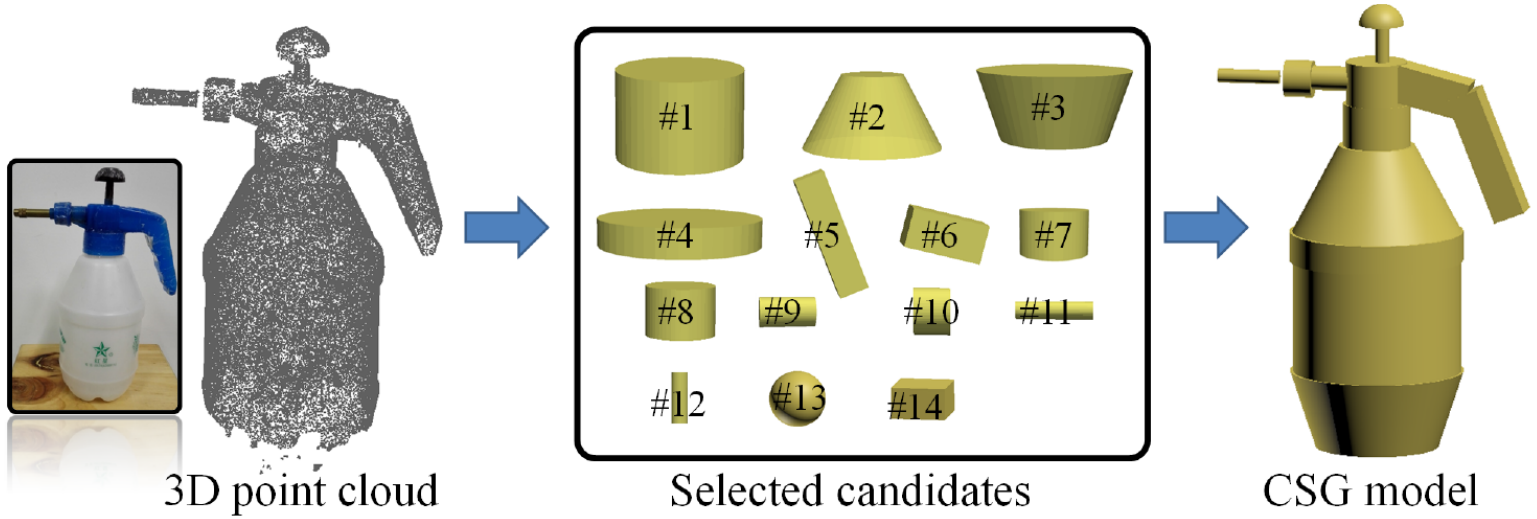
- CSG Tree Construction

Union

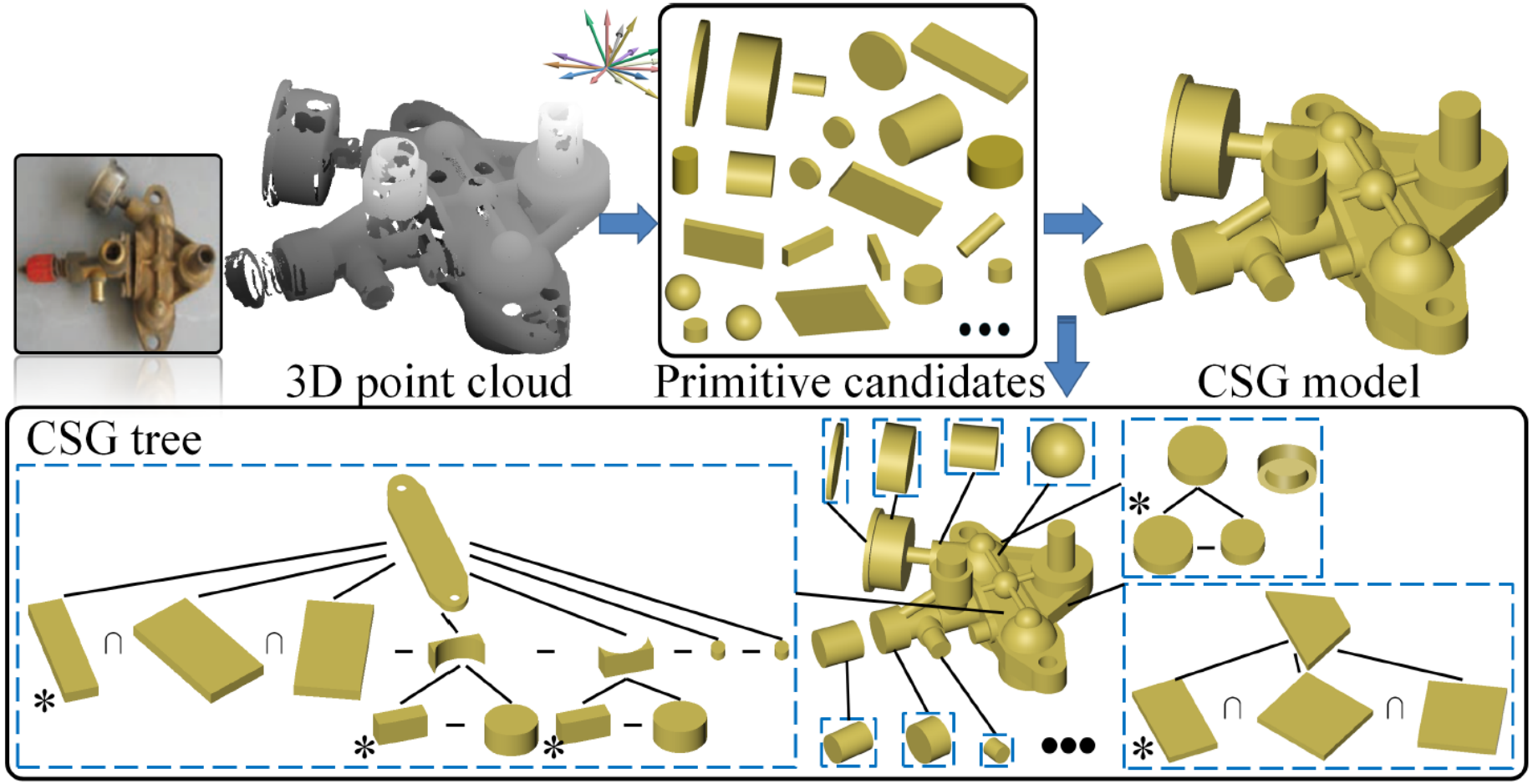


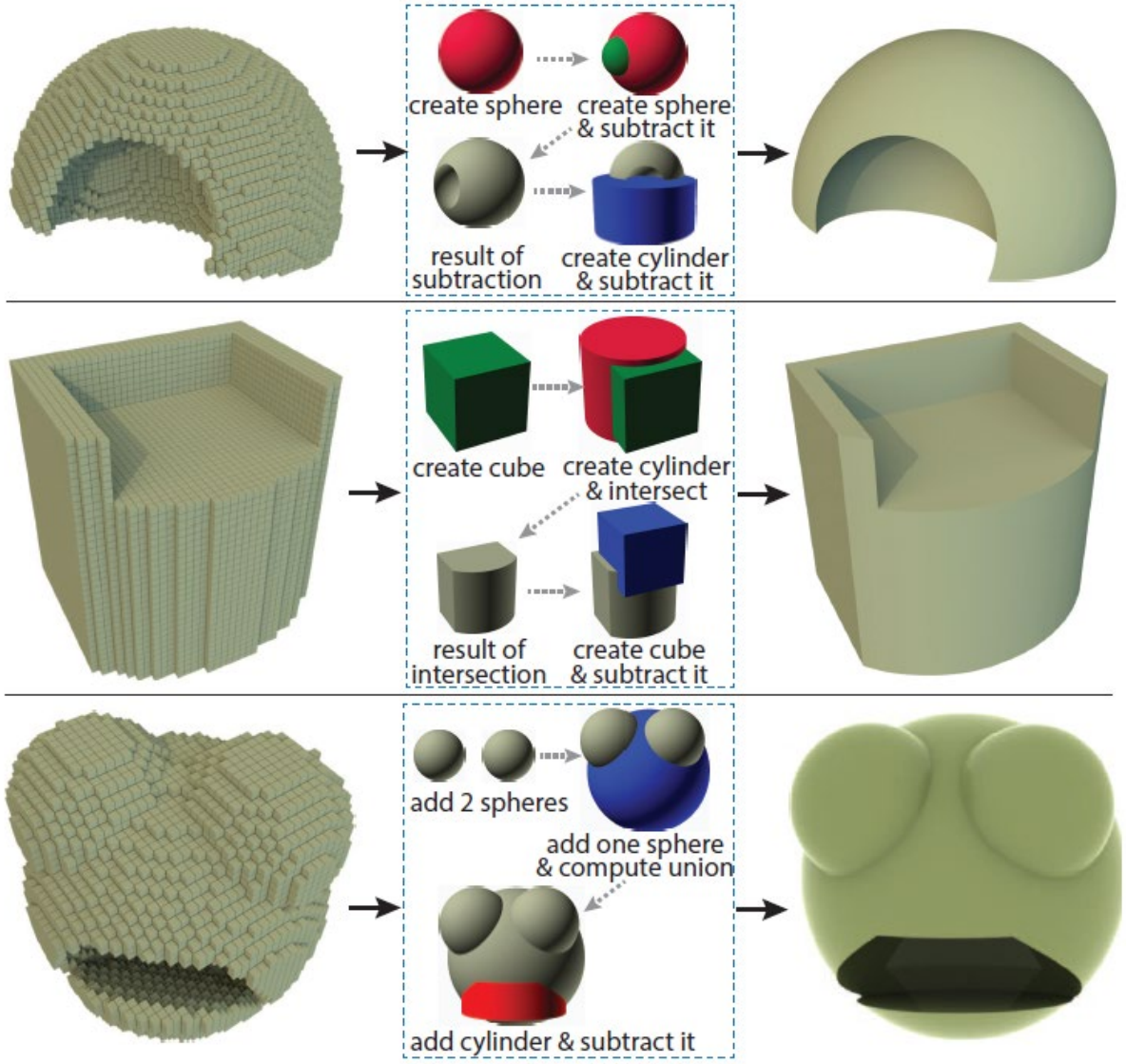
Results


Patch
fitting
Optimization
3D point cloud
Primitive candidates
CSG model


#1
#2
#3
#4
#5
#6
#7
#8
#9
#10
#11
#12
#13
#14
3D point cloud
Selected candidates
CSG model

Results



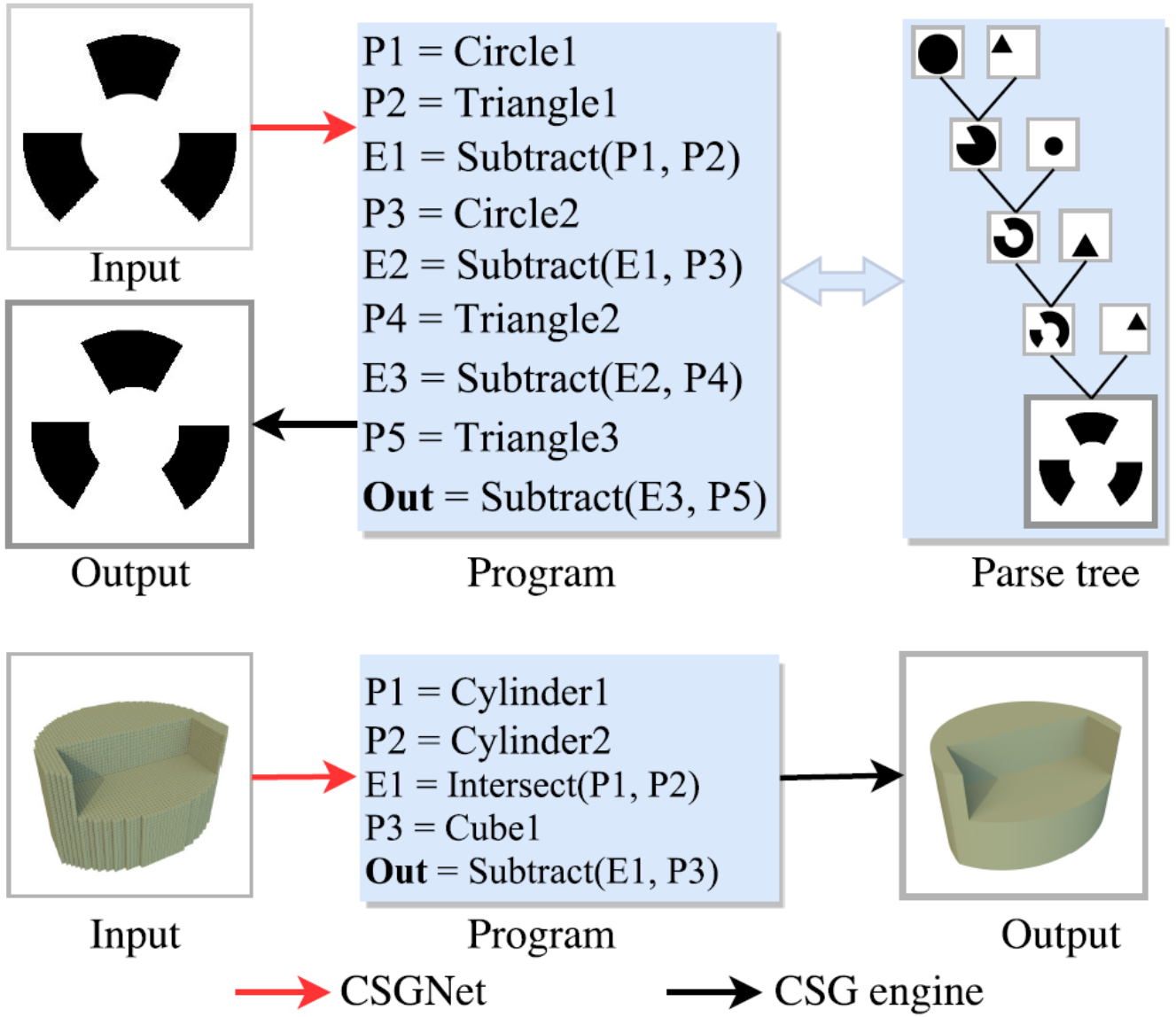


CSGNet: Neural Shape Parser For Constructive Solid Geometry

Sharma, Gopal, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhransu Maji. "Csgnet: Neural shape parser for constructive solid geometry." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5515-5523. 2018.

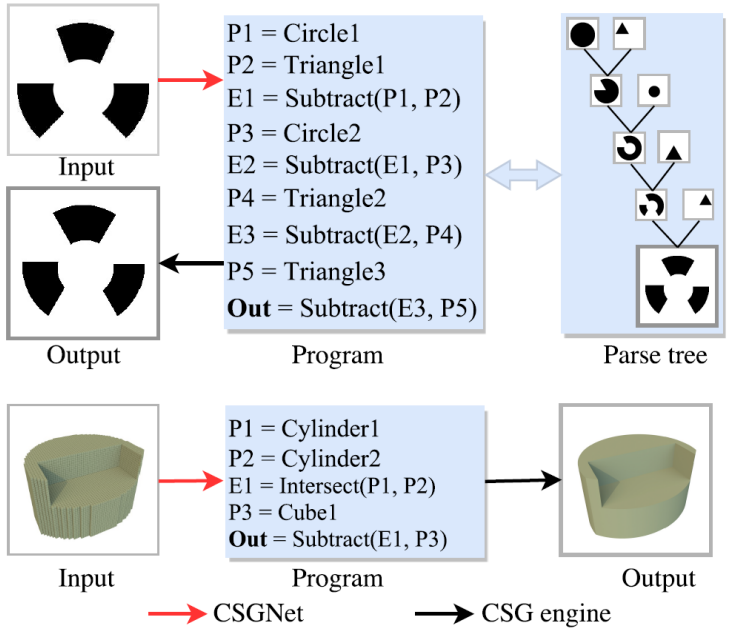
Goal

- Input:
  - 2D image
  - 3D voxel
- Output:
  - CSG program
  - Reconstructed shape

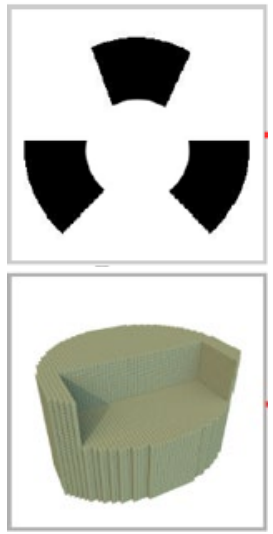


Challenges

- The number of primitives and operations is not the same for all shapes
- The order of the operations matters
- Learn an efficient parser to generate a compact program



Ideas – Deep learning solution



Input

Encoder

Decoder

Synthetic dataset

P1 = Circle1
P2 = Triangle1
E1 = Subtract(P1, P2)
P3 = Circle2
E2 = Subtract(E1, P3)
P4 = Triangle2
E3 = Subtract(E2, P4)
P5 = Triangle3
Out = Subtract(E3, P5)

P1 = Cylinder1
P2 = Cylinder2
E1 = Intersect(P1, P2)
P3 = Cube1
Out = Subtract(E1, P3)

Prediction

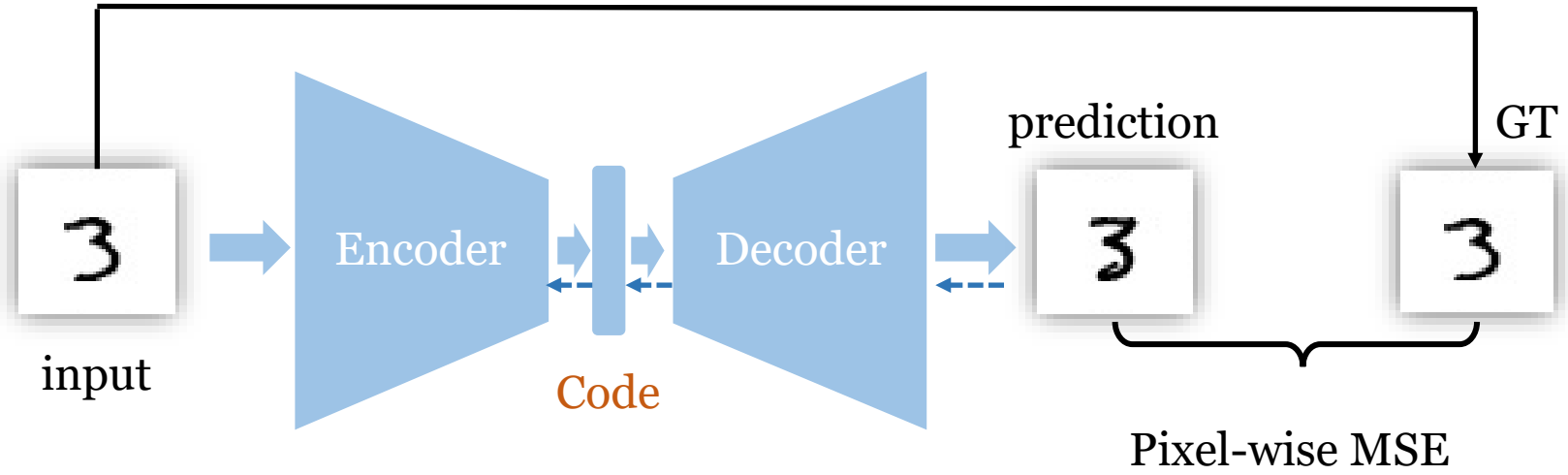
Deep Learning Knowledge

- Task: handwritten digit reconstruction
- Dataset: MNIST
- Network structure: Auto-encoder
- Loss: pixel-wise L2 loss (MSE)

Resources: https://www.youtube.com/watch?v=UNmqTiOnRfg&ab_channel=LuisSerrano
https://www.youtube.com/watch?v=WCUNPb-5EYI&ab_channel=BrandonRohrer

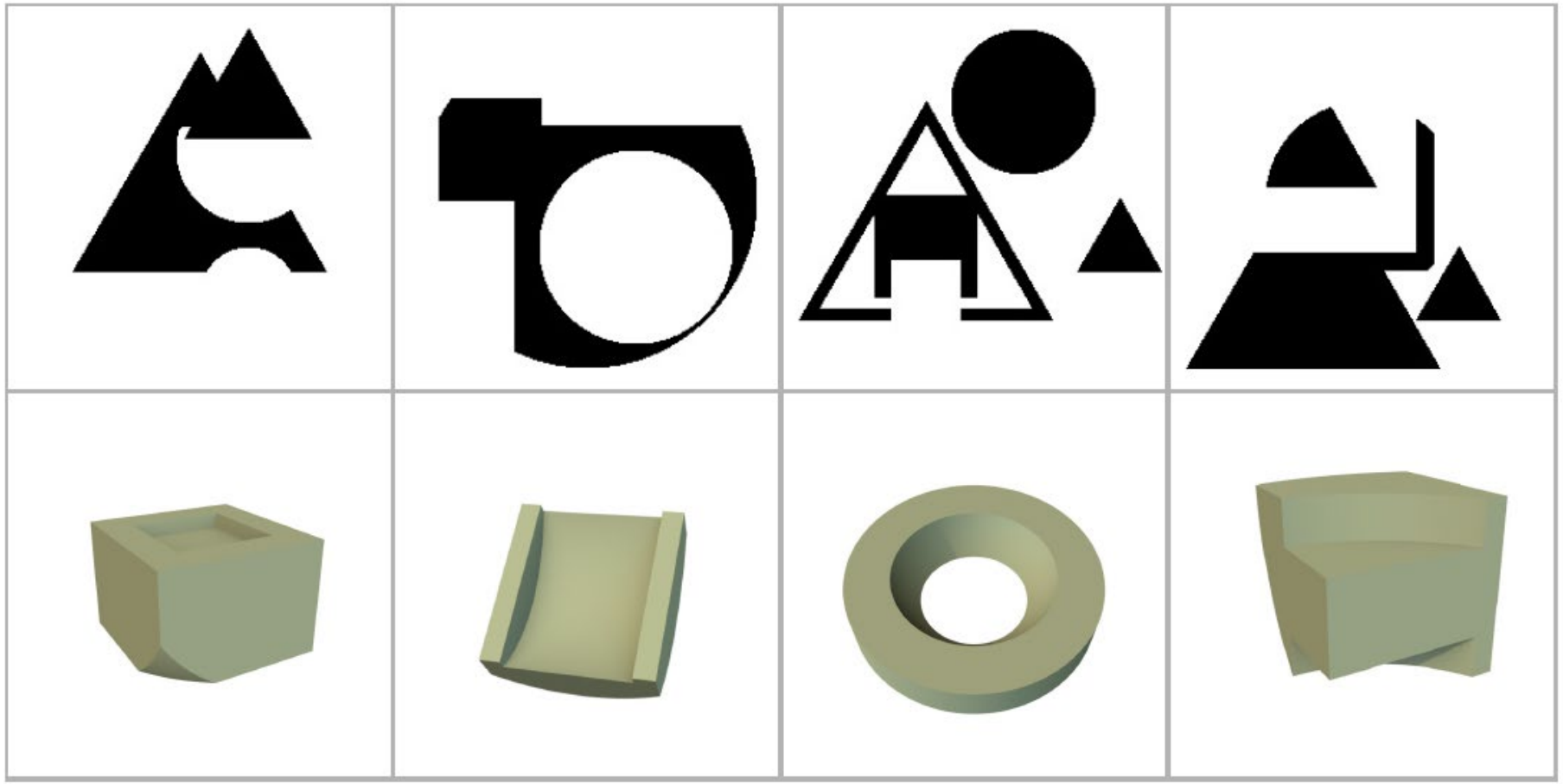
Deep Learning Knowledge

Supervised learning

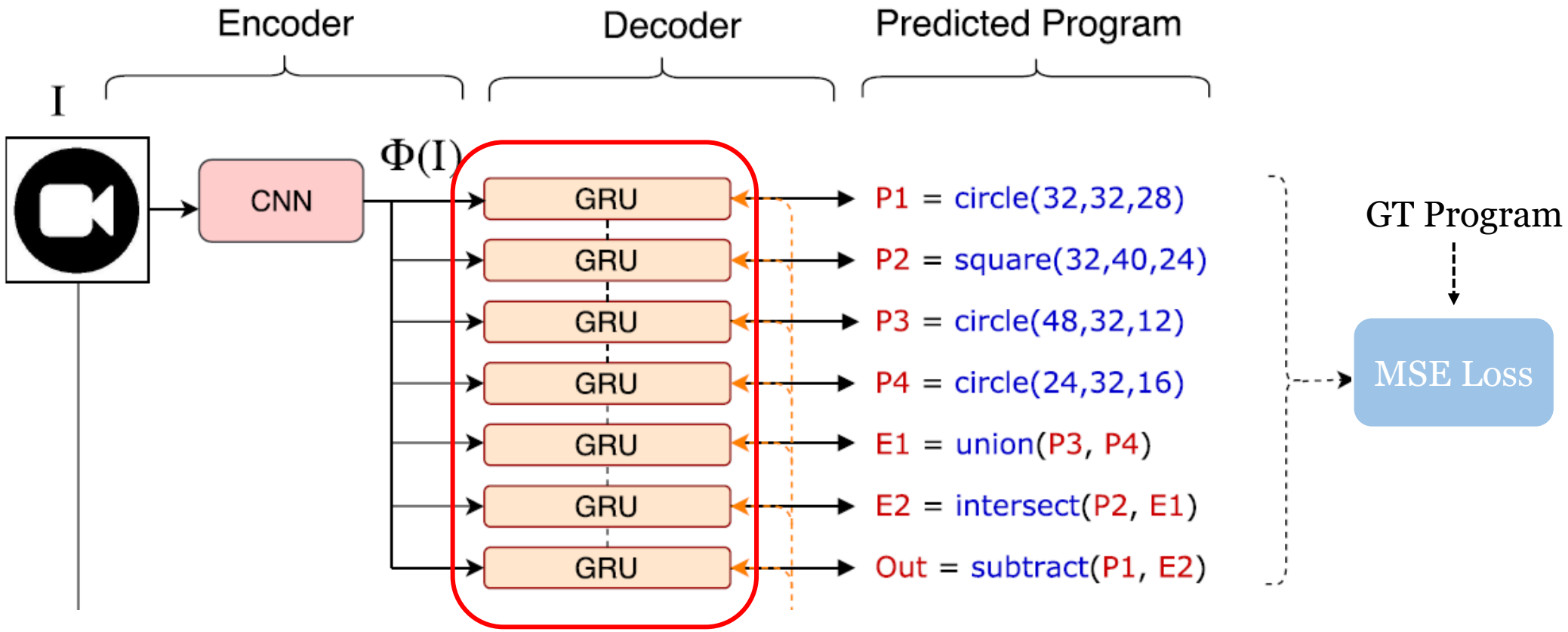


Resources: http://cs231n.stanford.edu/2019/

Synthetic Dataset

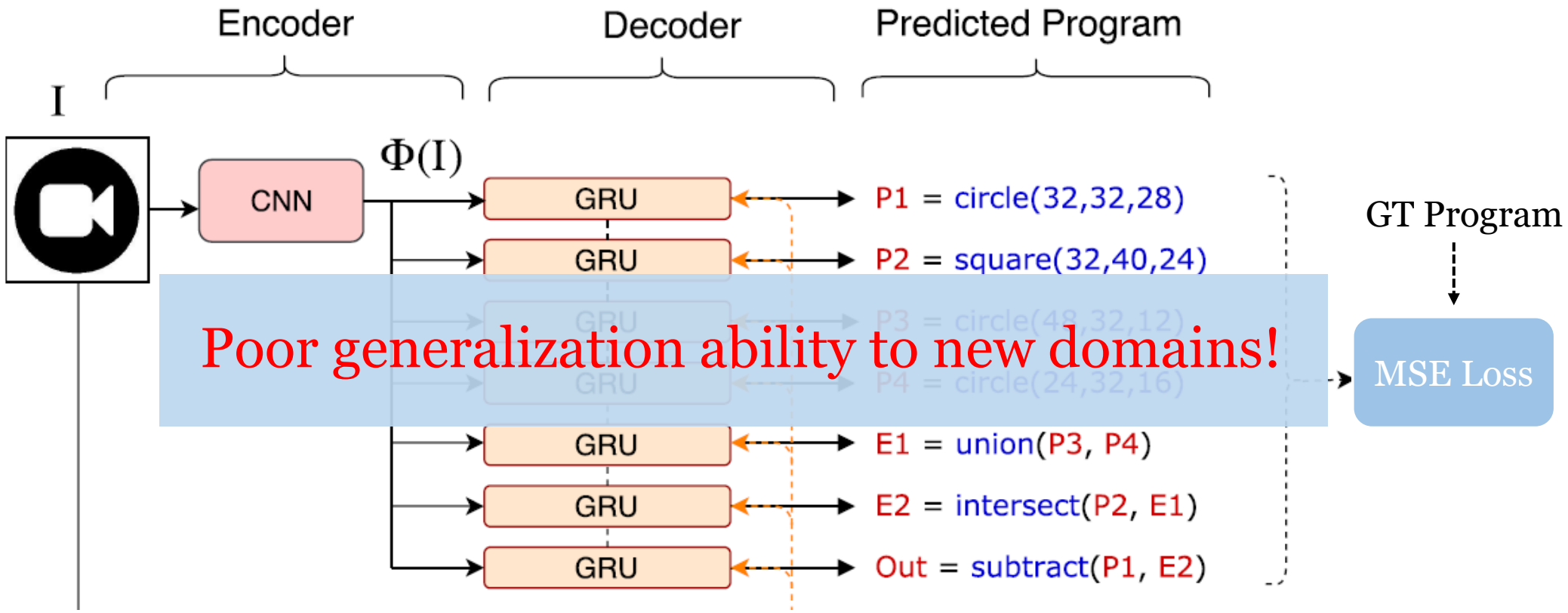


Supervised Learning

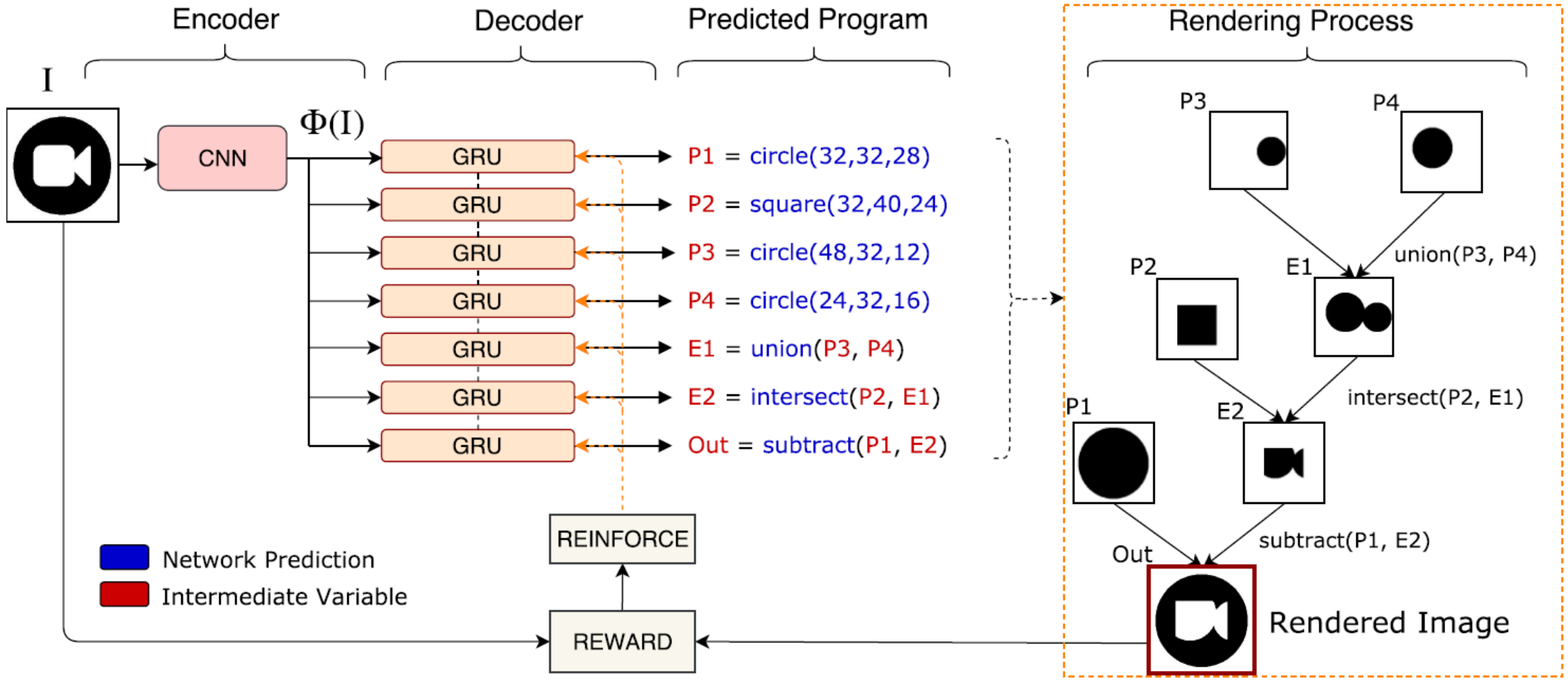


GRU: Gated Recurrent Units, automatically decide how many outputs

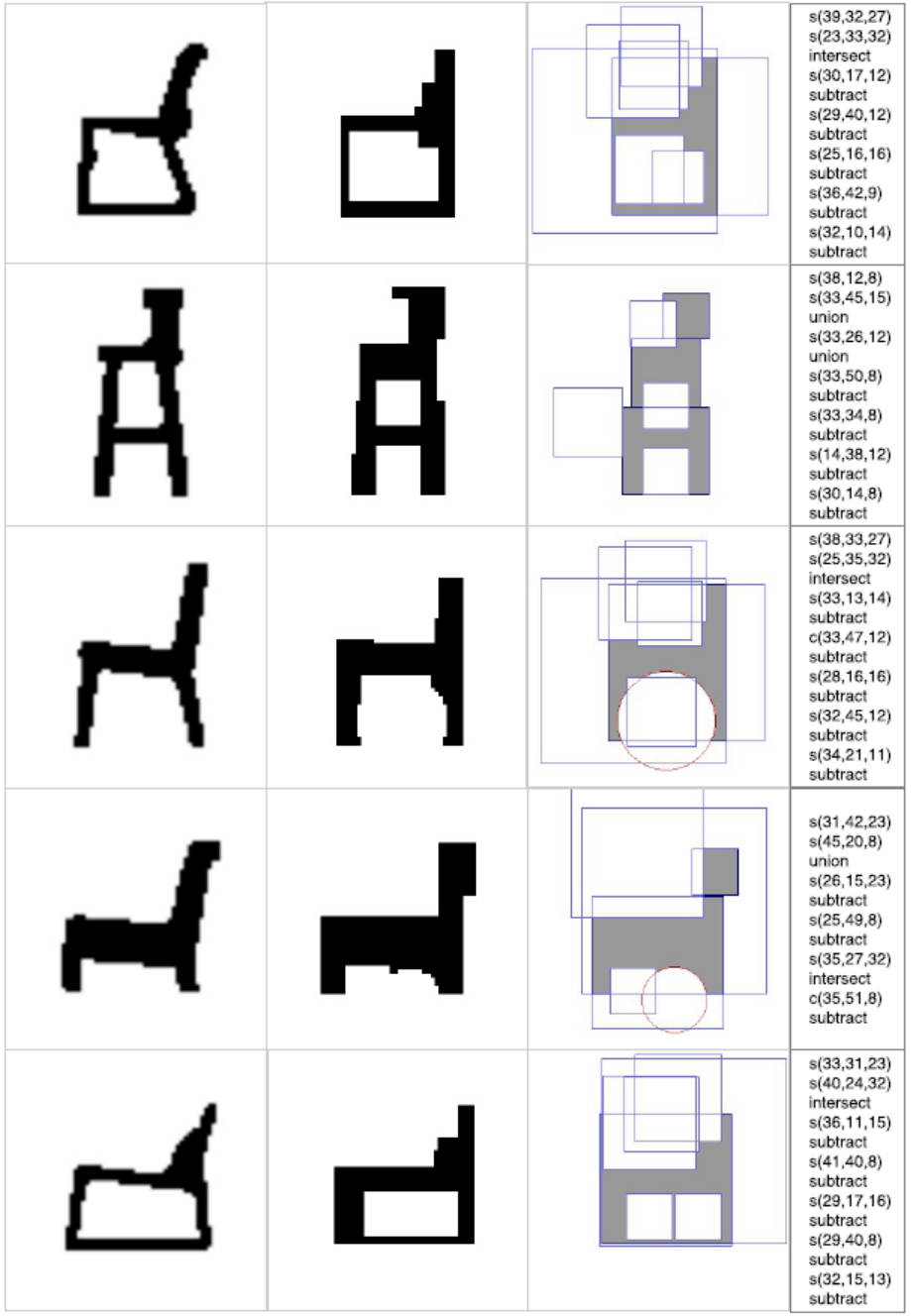
Supervised Learning

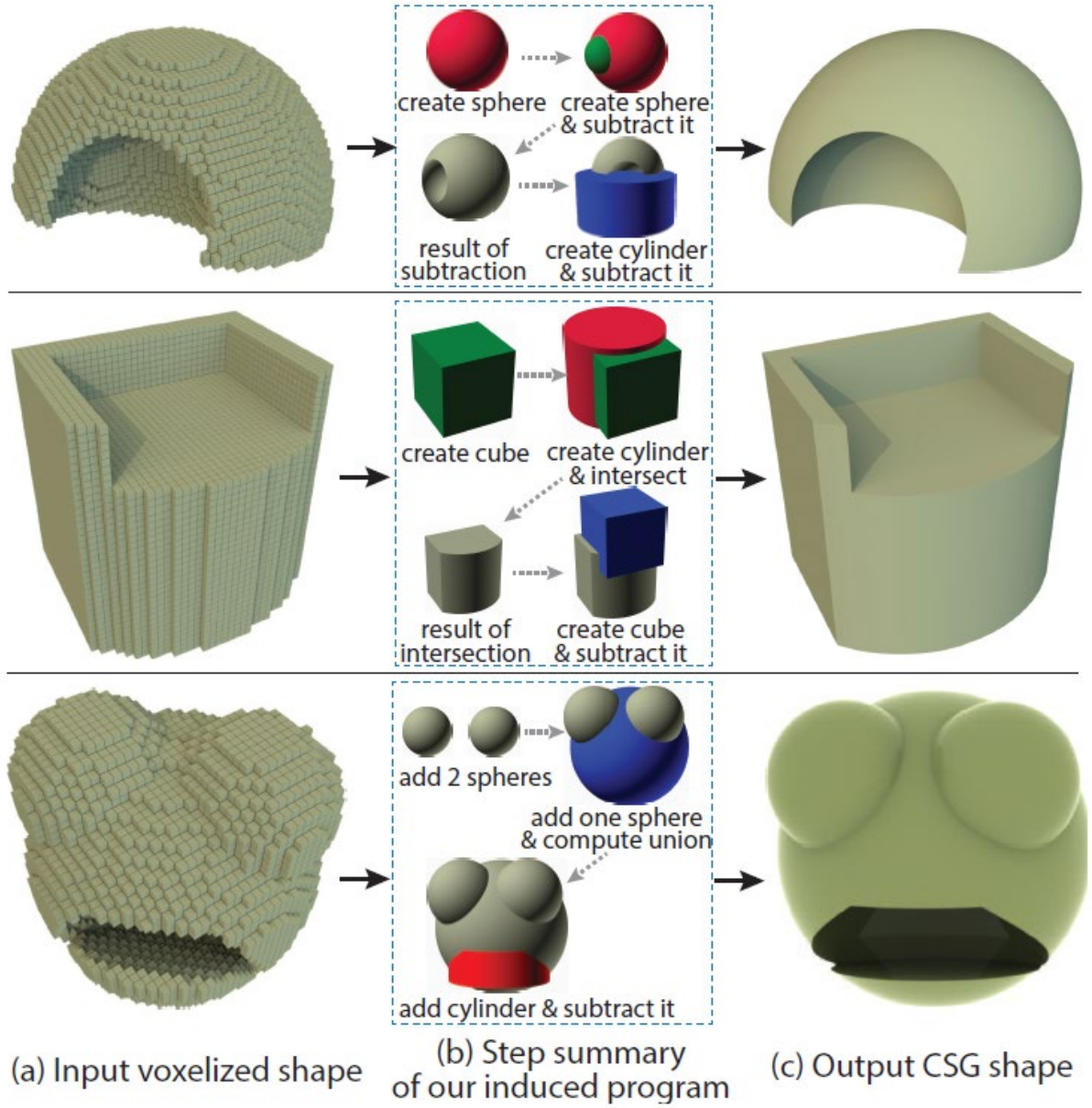


Reinforcement Learning



Results

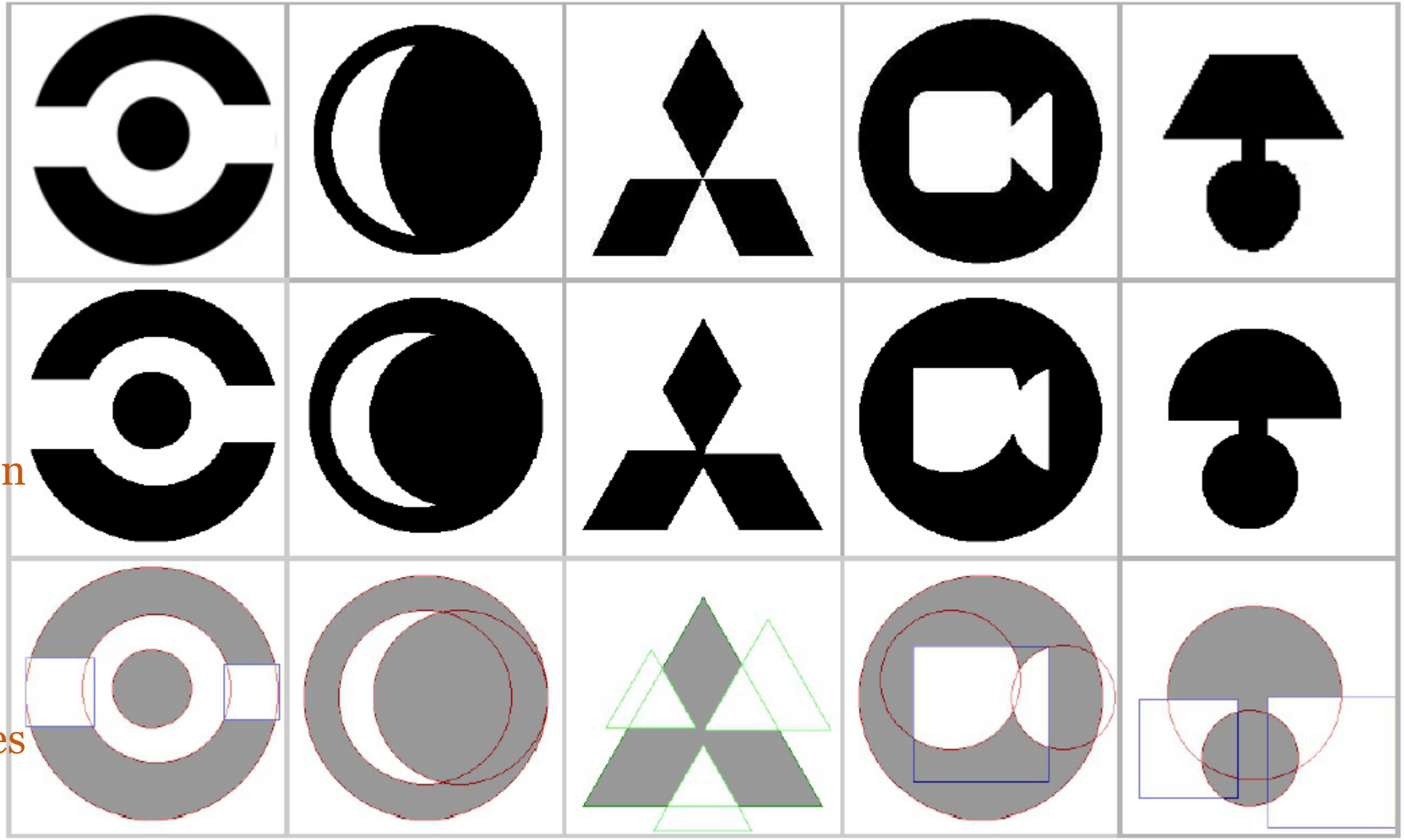

s(39,32,27)
s(23,33,32)
intersect
s(30,17,12)
subtract
s(29,40,12)
subtract
s(25,16,16)
subtract
s(36,42,9)
subtract
s(32,10,14)
subtract
s(38,12,8)
s(33,45,15)
union
s(33,26,12)
union
s(33,50,8)
subtract
s(33,34,8)
subtract
s(14,38,12)
subtract
s(30,14,8)
subtract
s(38,33,27)
s(25,35,32)
intersect
s(33,13,14)
subtract
c(33,47,12)
subtract
s(28,16,16)
subtract
s(32,45,12)
subtract
s(34,21,11)
subtract
s(31,42,23)
s(45,20,8)
union
s(26,15,23)
subtract
s(25,49,8)
subtract
s(35,27,32)
intersect
c(35,51,8)
subtract
s(33,31,23)
s(40,24,32)
intersect
s(36,11,15)
subtract
s(41,40,8)
subtract
s(29,17,16)
subtract
s(29,40,8)
subtract
s(32,15,13)
subtract
a) Input
b) Output
c) Primitives
d) Program


create sphere
create sphere & subtract it
result of subtraction
create cylinder & subtract it
create cube
create cylinder & intersect
result of intersection
create cube & subtract it
add 2 spheres
add one sphere & compute union
add cylinder & subtract it
(a) Input voxelized shape
(b) Step summary of our induced program
(c) Output CSG shape

a)
Target

b)
Prediction

c)
Primitives



Conclusion

- GSG representation
  - Expressive and concise rep.
  - Compact rep. to store
  - Easy to edit
  - Well accepted by modern CAD system

- CSG and inverse CSG modeling
  - Still long way to go …