Realtime Computer Graphics on GPUs

What did not fit elsewhere

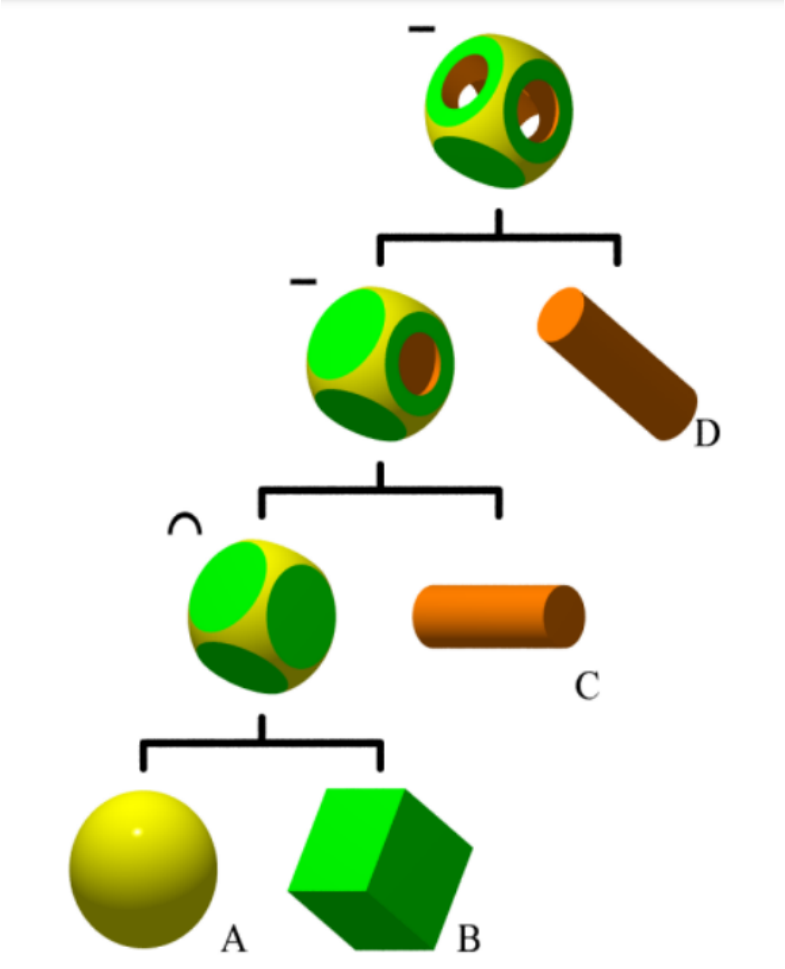
Jan Kolomazník

Department of Software and Computer Science Education
Faculty of Mathematics and Physics
Charles University in Prague

Constructive Solid Geometry

What is CSG?

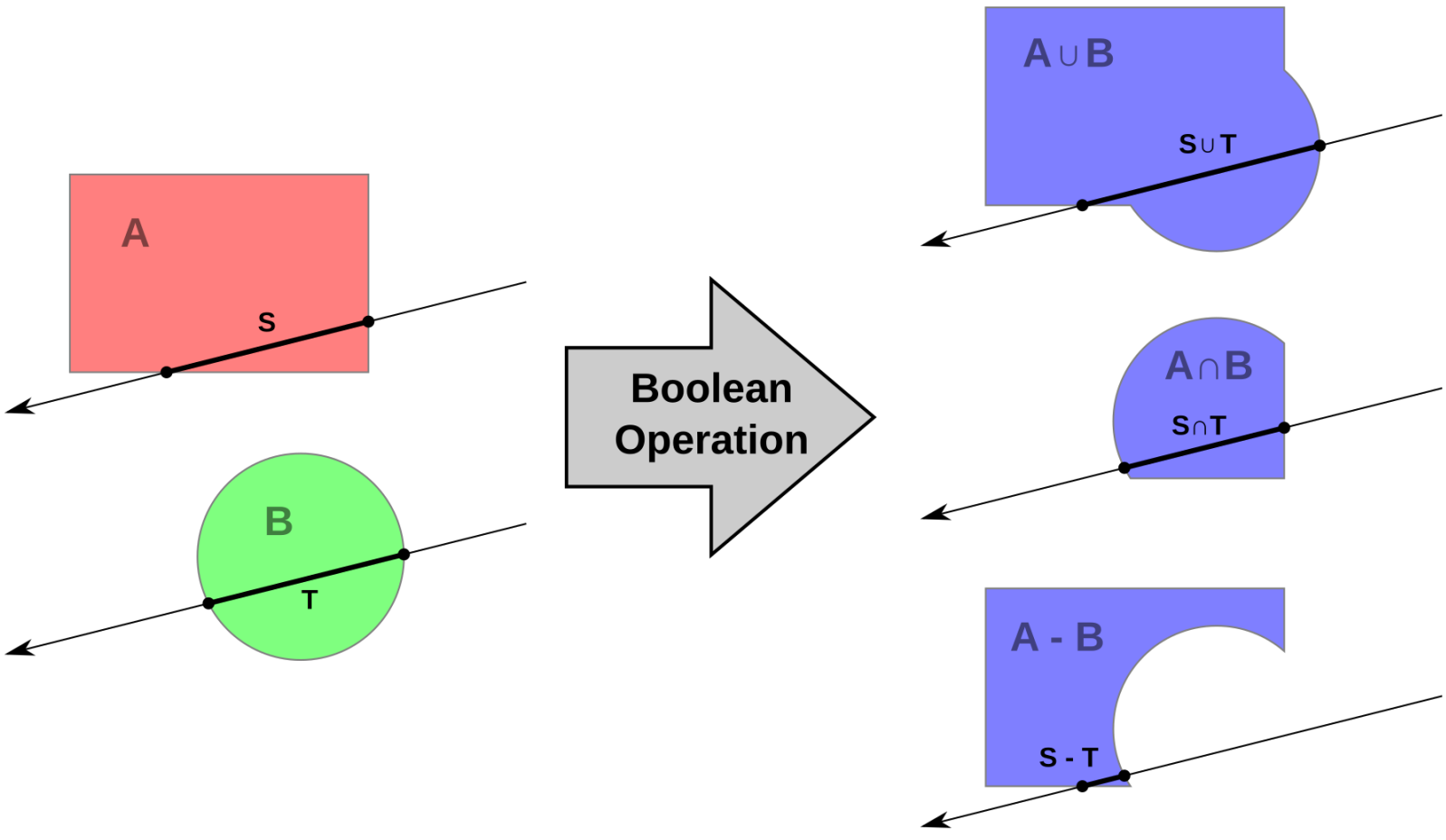
- Used by CAD applications
- Set operations on geometry primitives
  - Union
  - Intersection
  - Subtraction



How to render?

- Conversion to tri-mesh
- Raytracing
- Rasterization pipeline

RAYTRACING



Rasterization Pipeline

- Elementary solids converted to polyhedra
- set operations evaluated on the GPU
  - union is trivial (default depth-buffer based rendering)
  - intersection and subtraction: use of stencil buffer, considering front vs. back facesunion is trivial (default depth-buffer based rendering)

GOLDFEATHER

- 1989: Goldfeather et al.
- normalization of a CSG tree – decomposition to union of ”products” (intersections and differences)
- implementation uses several depth-buffers and a stencil buffer (needs to copy depth-buffers)

SEQUENCED CONVEX SUBTRACTION

- 2000:Stewart et al. – Sequenced Convex Subtraction (SCS)
- does not need depth-buffer copying, complex depth-tests
- all elementary solids have to be convex

SCS Phases

1. preprocessing (CSG normalization, sorting of subtraction sequences /front-to-back/)
2. depth-buffer processing (for every product + merge)
3. final rendering to frame-buffer

INTERSECTION OF N SOLIDS

- init: depth = near; stencil = 0;
- passing through front faces of individual solids

```
if ( front > depth ) depth = front;
```

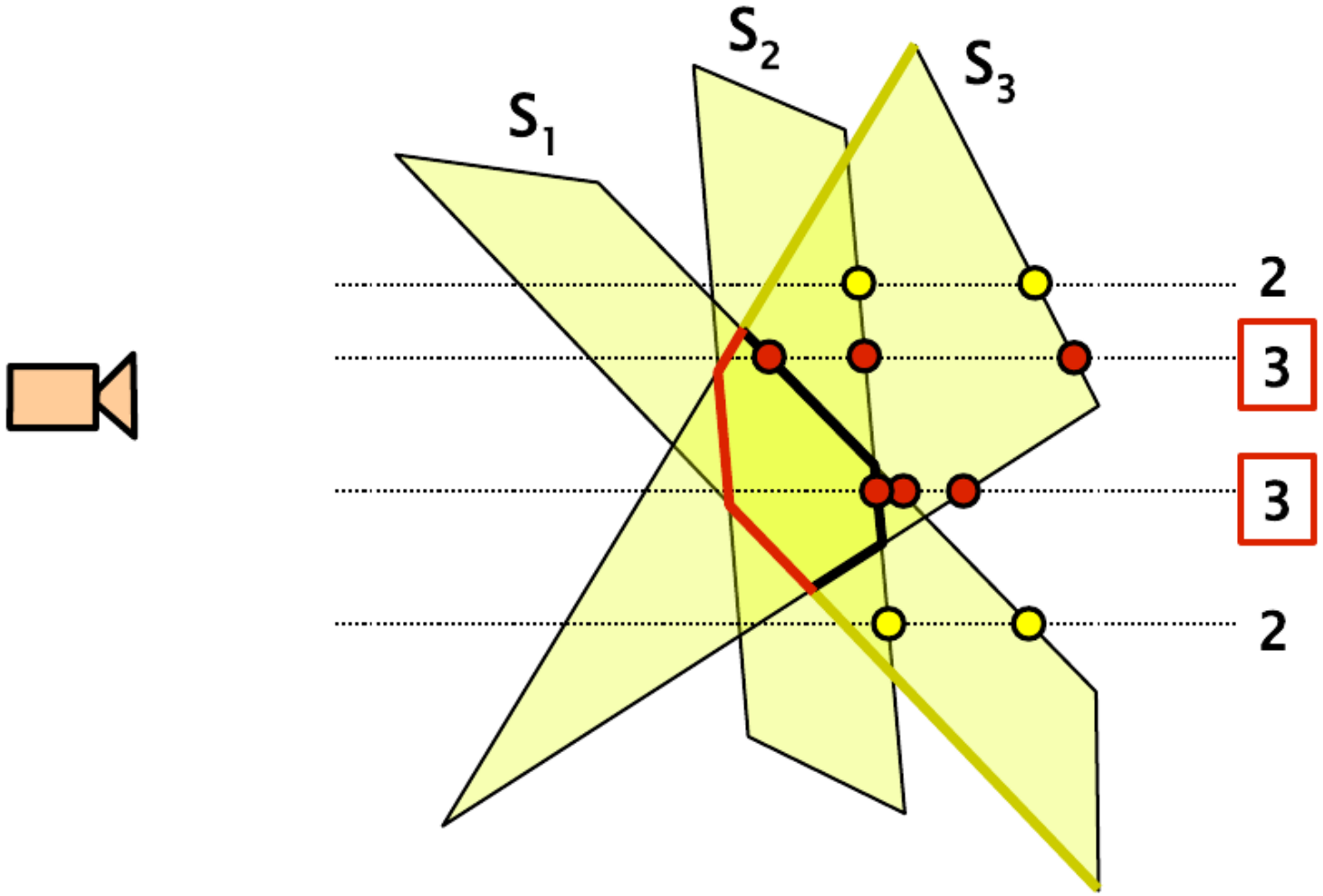
- passing through back faces (occlusions)

```
if ( back > depth ) stencil++;
```

- removing pixels with occlusion number ¡ n

```
if ( stencil < n )
{
stencil = 0;
depth = far;
}
```

INTERSECTION – EXAMPLE



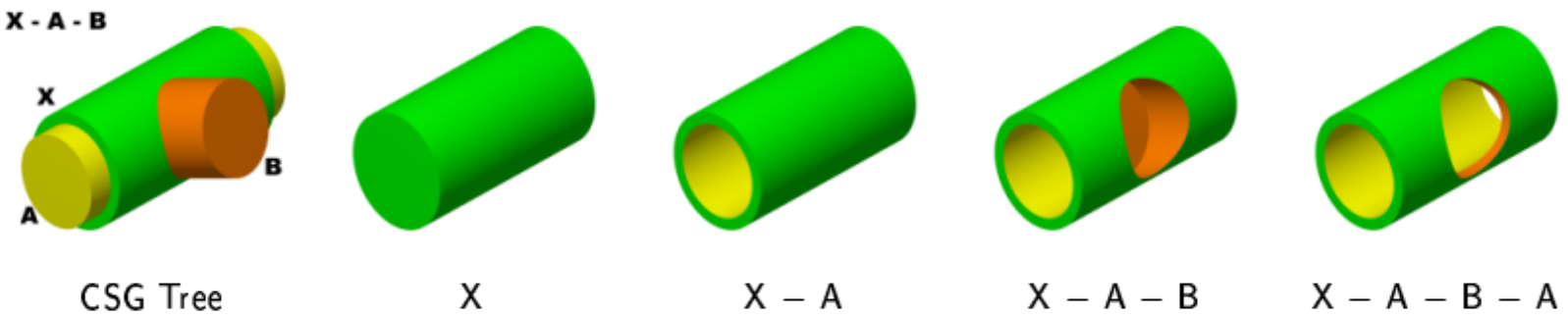
SUBTRACTING SEQUENCES

- determining correct subtracting sequence;
  - front-to-end subtraction
  - universal subtractin sequence – contains all occlusion permutations
- subtracting from the front: passing all subtr. solids

```
if ( front < depth ) stencil = 1;
else stencil = 0;
```

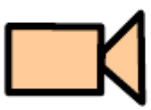
- every back-face is processed immediately as well

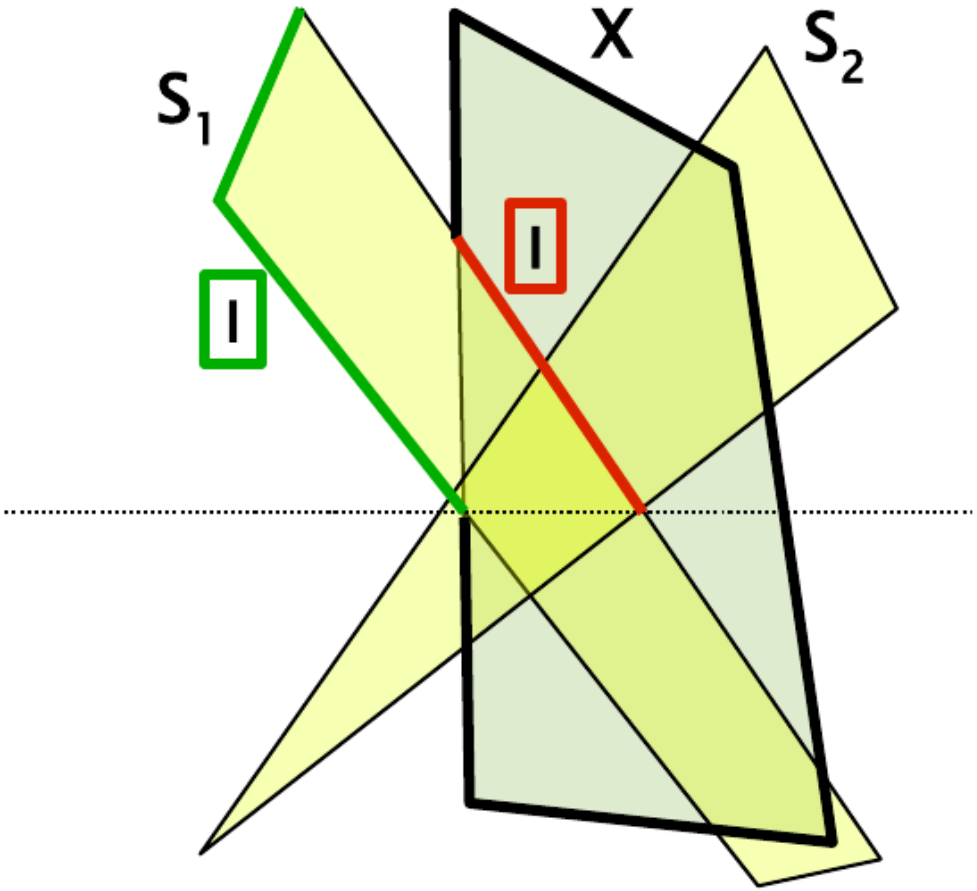
```
if ( back > depth && stencil == 1 )
depth = back;
```



SUBTRACTION – STEP I

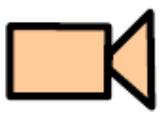
$X - S_1 - S_2 - S_1$

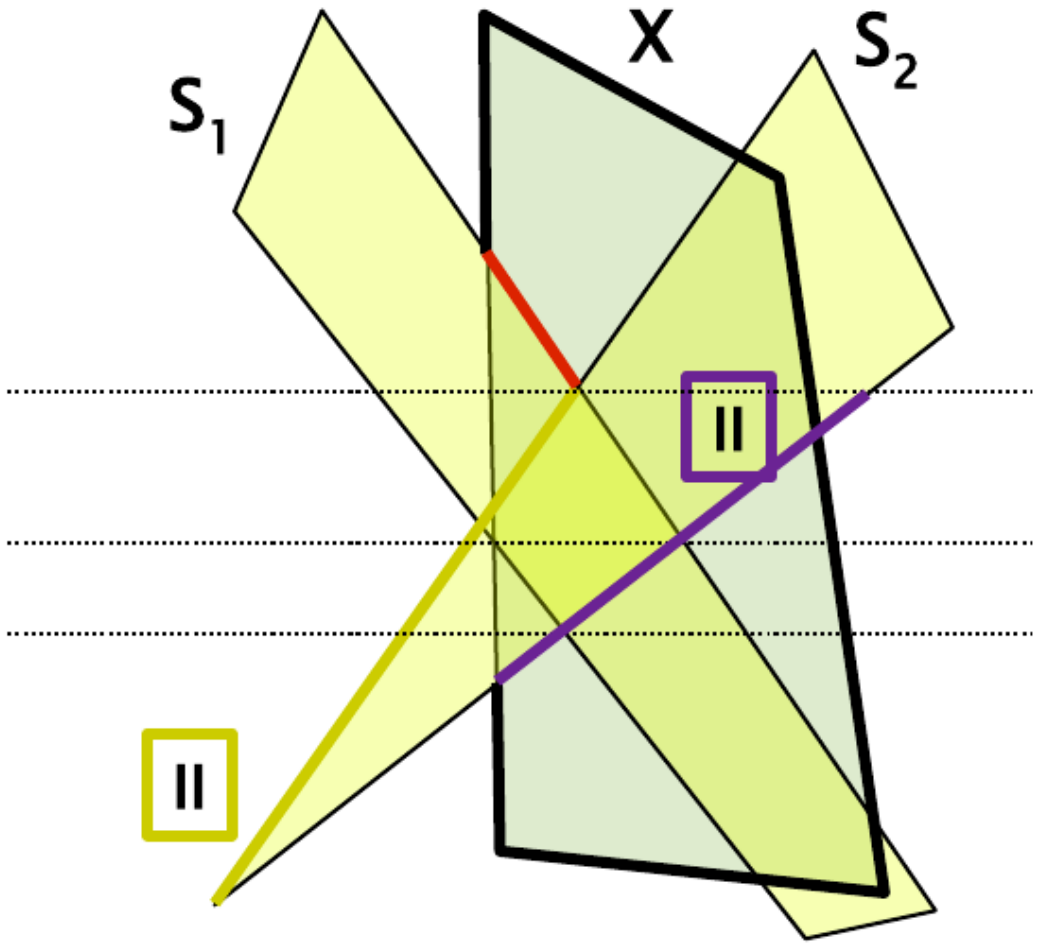




SUBTRACTION – STEP II

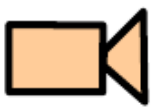
$X - S_1 - S_2 - S_1$

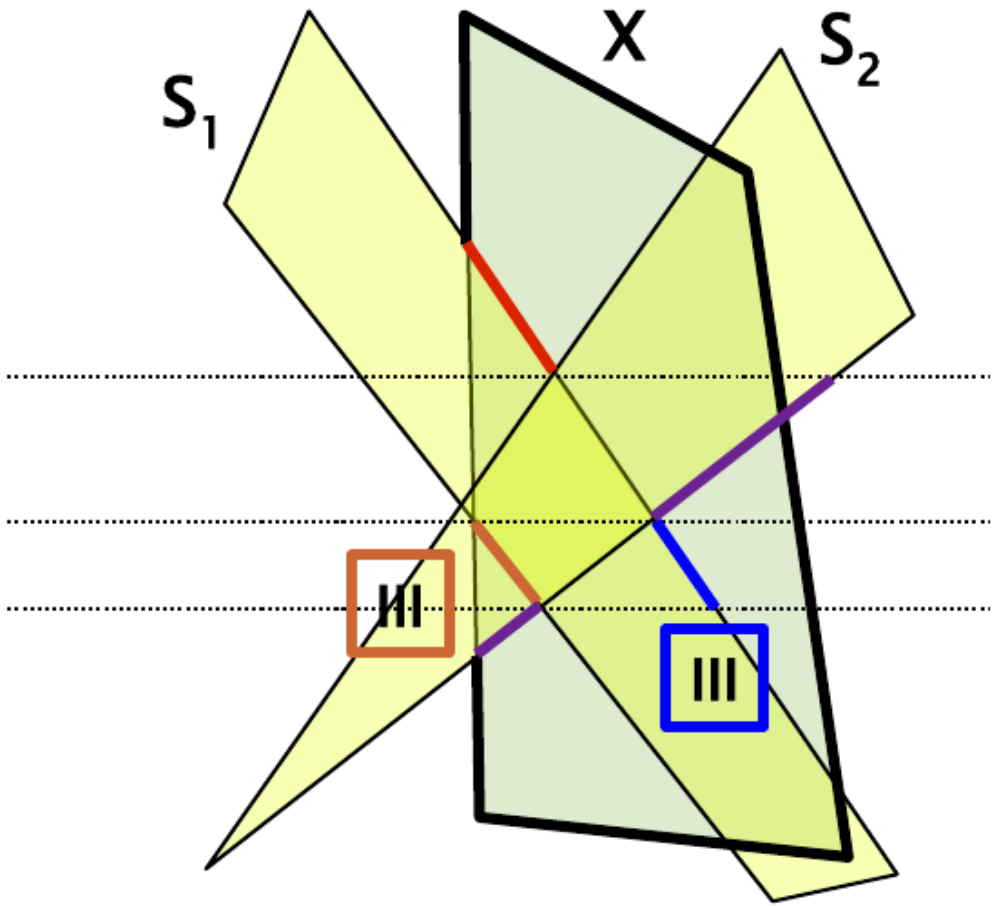




SUBTRACTION – STEP III

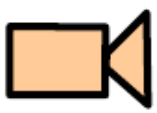
$X - S_1 - S_2 - S_1$

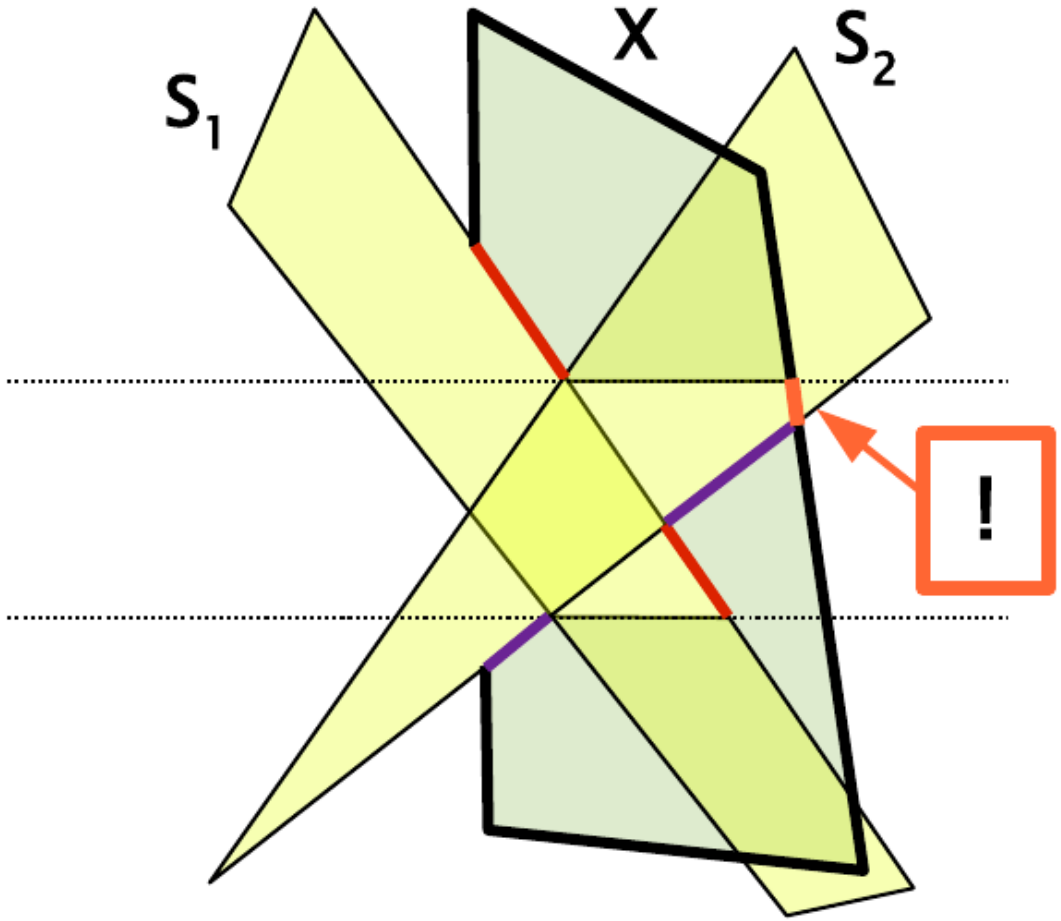




SUBTRACTION – RESULT

$X - S_1 - S_2 - S_1$





COMPLETELY SUBTRACETED PARTS

- removing parts of common intersection, which were eliminated completely
- passing through all intersection solids (back faces only – looking for empty results)
- elimination of completely subtracted parts

Merging products and final rendering

- product result = its depth buffer
- merging results of one product (i.e. union operation)
- intersections render front faces
- subtractions render back faces