# Introducing
# Basic Boolean Set Operations
# for GroIMP

Florian Schöler, Volker Steinhage
Department of Computer Science III,
University of Bonn, Germany

Tutorial and Workshop „Modelling with GroIMP and XL“
Georg-August Universität Göttingen, February 27th - 29th, 2012

universitätbonn

# Introduction

- Modeling / representation of complex objects
  - Visualization
  - Compute physical properties
  - Computer games
- Voxel representation, polygonal meshes, …
- Closed surface, inside/outside

# Constructive Solid Geometry

- Basic building blocks: geometric primitives (sphere, cylinder, …)
- Objects modeled as combination of primitives
- Boolean Set Operations: union, intersection, difference
- Represented as rooted binary tree
  - Leaves: primitives
  - Nodes: results of single operations
  - Root: final object
- Well suited for use in GroIMP
  - Embed tree in graph
  - Many primitives available

# CSG Example

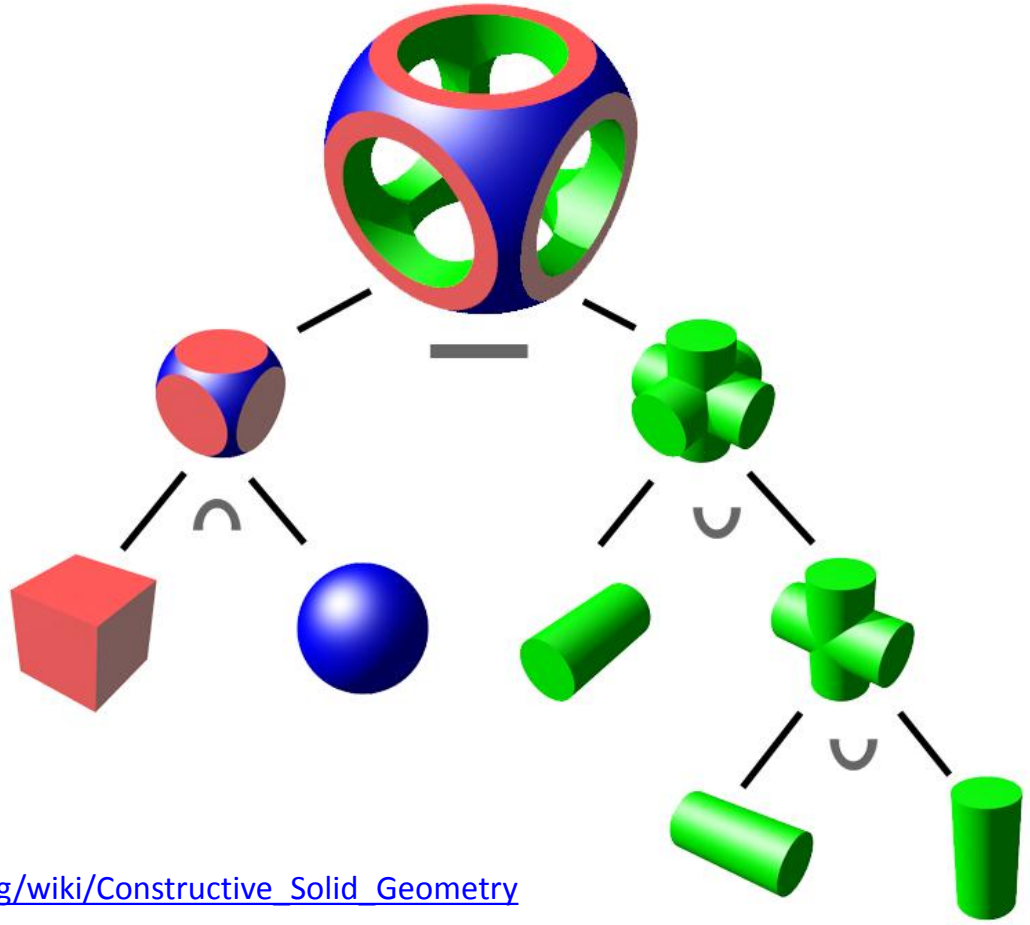

Image from
http://de.wikipedia.org/wiki/Constructive_Solid_Geometry
03.02.2012

# The Plugin

- UnBBoolean library by Danilo Balby (http://unbboolean.sourceforge.net, last page view 09.01.2012)
- Input/Output: Sets of triangles
- Introduce three classes:
  - BSO: To be inserted into the graph
  - Transformer: Computes triangle sets from GroIMP primitives
  - Sollid: Stores surface triangulations of objects and provides operations

# The Plugin

- Invoke BSO::apply() to generate result
  - Recursively traverse GroIMP-graph
  - If primitive p found, include result of Transformer::getSolidFor(p) in operation
  - If BSO found, compute its result, then also include in operation
- Transformer
  - Resolution of triangle nets can be adjusted
- Solid
  - List of vertices
  - List of indices (into list of vertices)
  - Methods for operations

# Examples

- Live Demo!

# Summary

- Prototypic implementation of Boolean Set Operations in GroIMP
- All operations and primitives supported
- Arbitrary combinations
- Further applications of results (e.g., volume)
- UnBBoolean inefficient and can run into infinite loops

# Future Work

- More efficient implementation
- More robust implementation
- Allow different colors for different parts