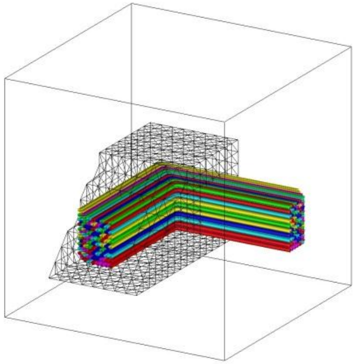
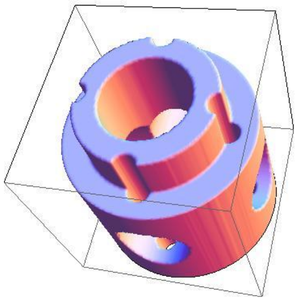
Degree-Driven Algorithm Design for Computing Volumes of CSG Models

David L. Millman Jack Snoeyink

Department of Computer Science – UNC-Chapel Hill





June 19, 2012

Motivation and Background

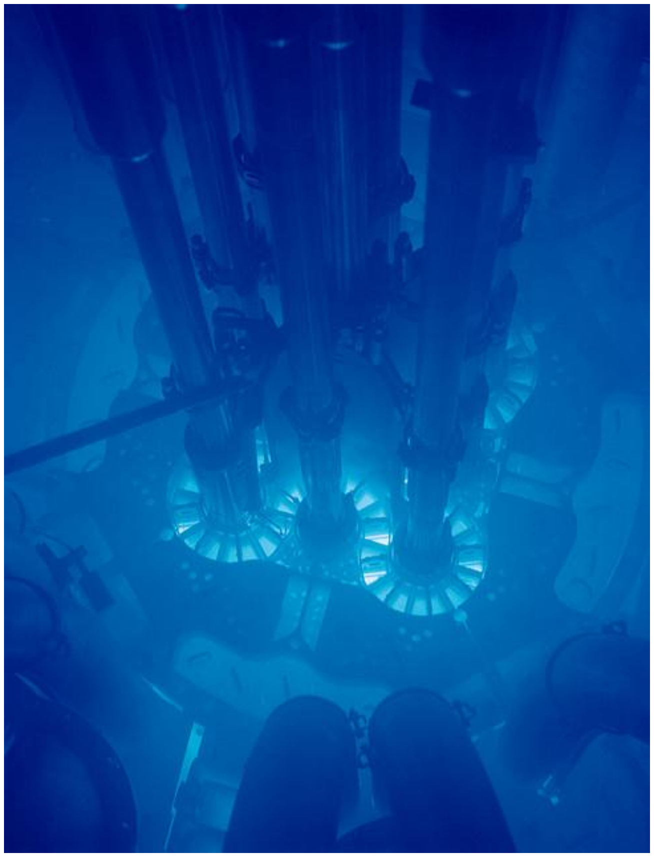


Image from Idaho National Lab, Flickr

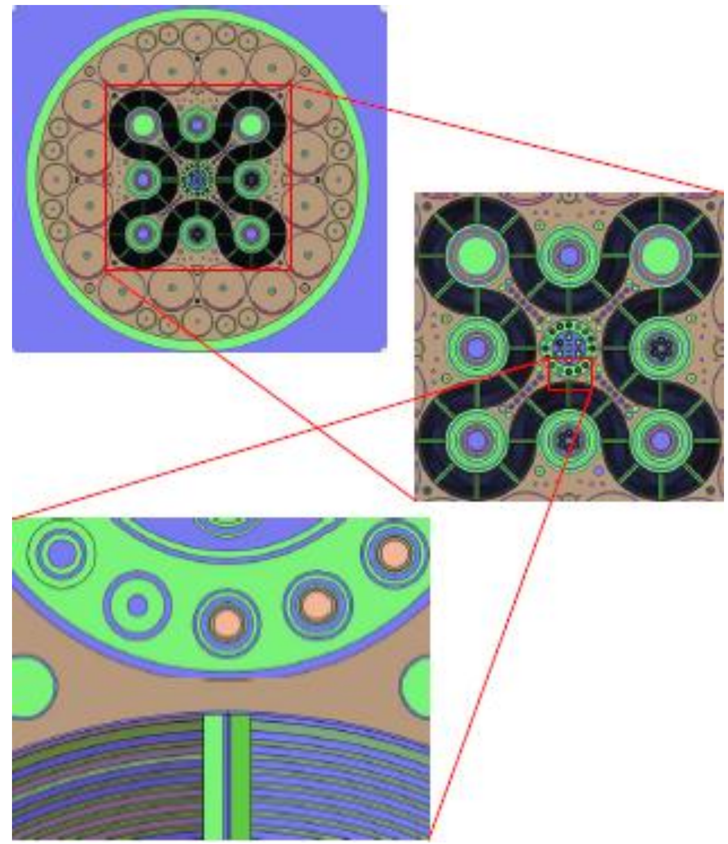
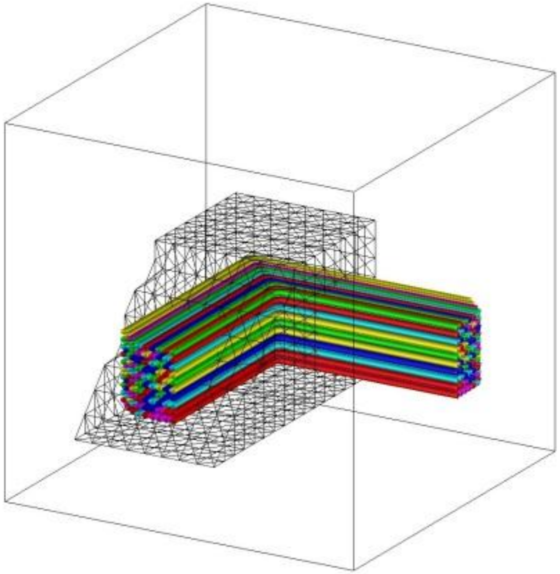


Image from: T.M. Sutton, et. al., *The MC21 Monte Carlo Transport Code*, Proceedings of (M&C + SNA 2007)

Volume Calculation Framework Overview



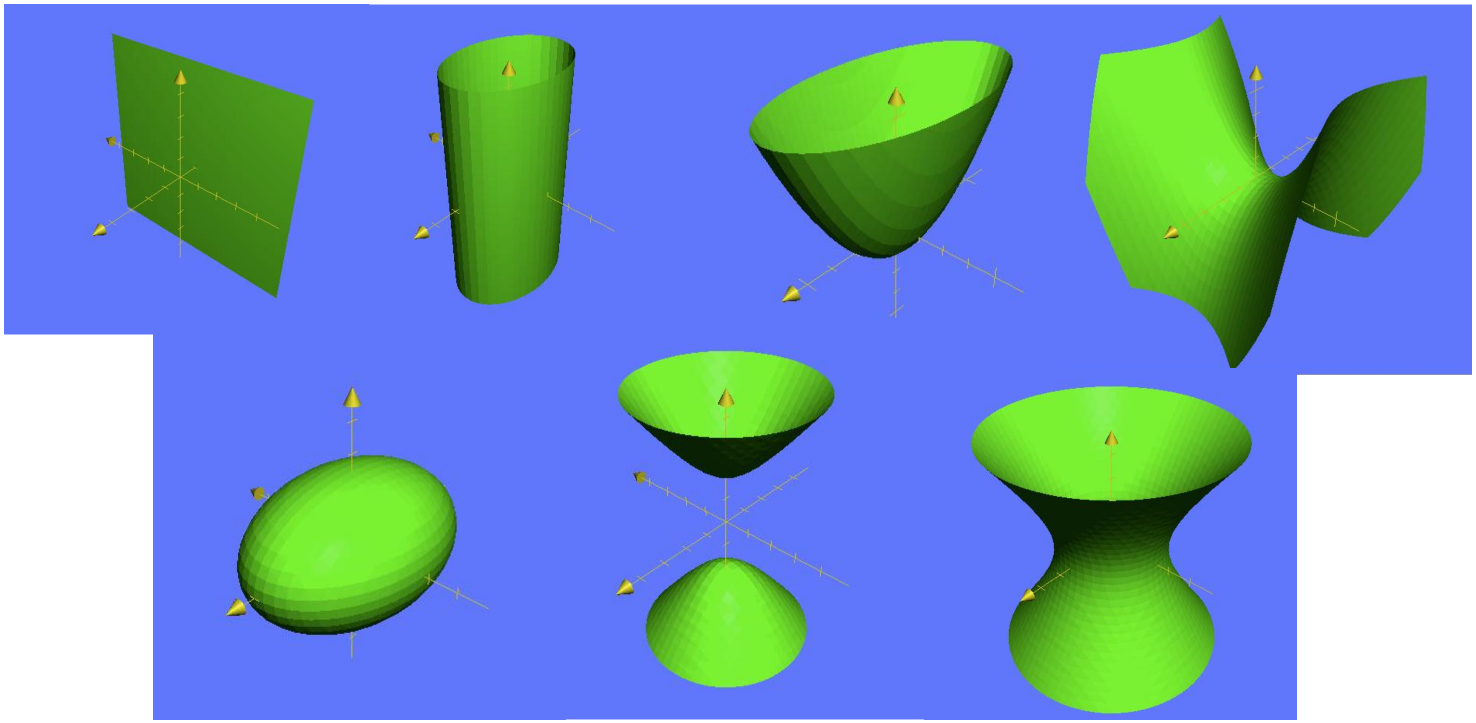
Basic idea: *Divide-and-conquer*.
Use an octree to decompose space into boxes,
determining the surfaces affecting each box,
stopping when the box is small enough
or surfaces are simple enough
that we can approximate volume accurately.

Our contribution: Framework that computes each component's volume in multi-comp. CSG models.

Based on a minimal, extensible set of predicates that handles any model & is very efficient on common cases.

Algorithm	Error	Time (sec)
Old	**<1e-5**	**790.28**
New	**<1e-6**	**1.41**

Primitives:
Signed Quadratic Surfaces

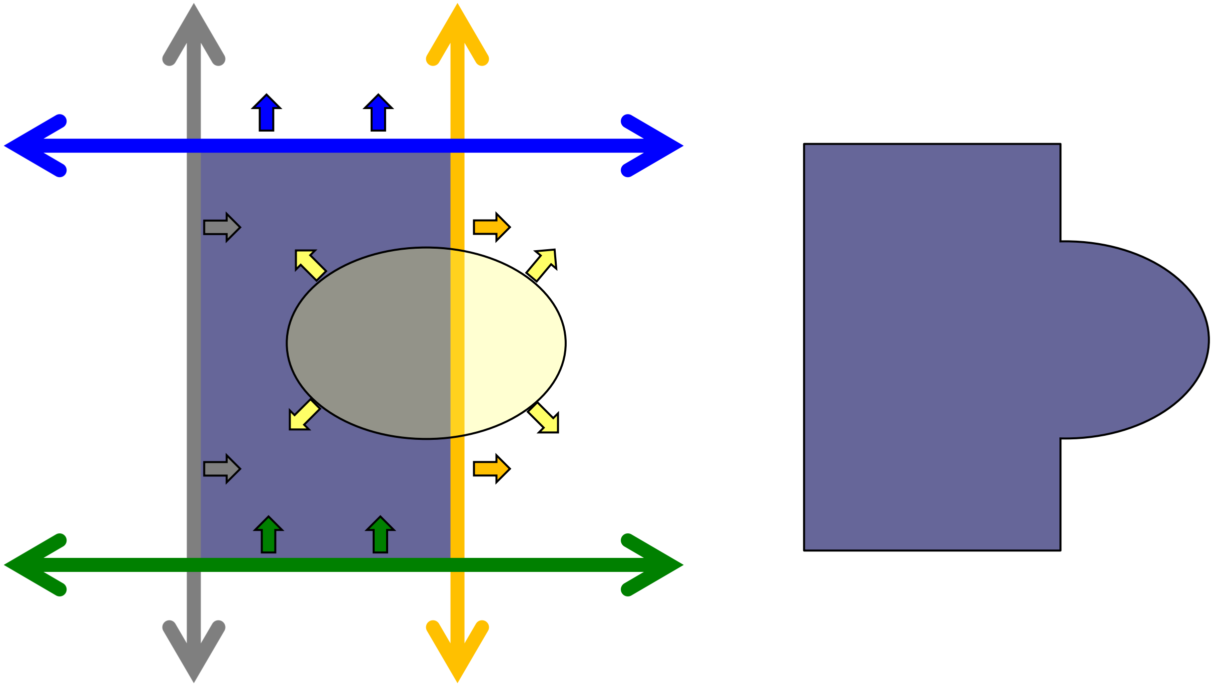


$$f(x,y,z) < a_1x^2 + a_2y^2 + a_3z^2$$
$$+a_4xy + a_5xz + a_6yz$$
$$+a_7x + a_8y + a_9z + a_{10}$$

Model Representation
Basic Component: Boolean Formula

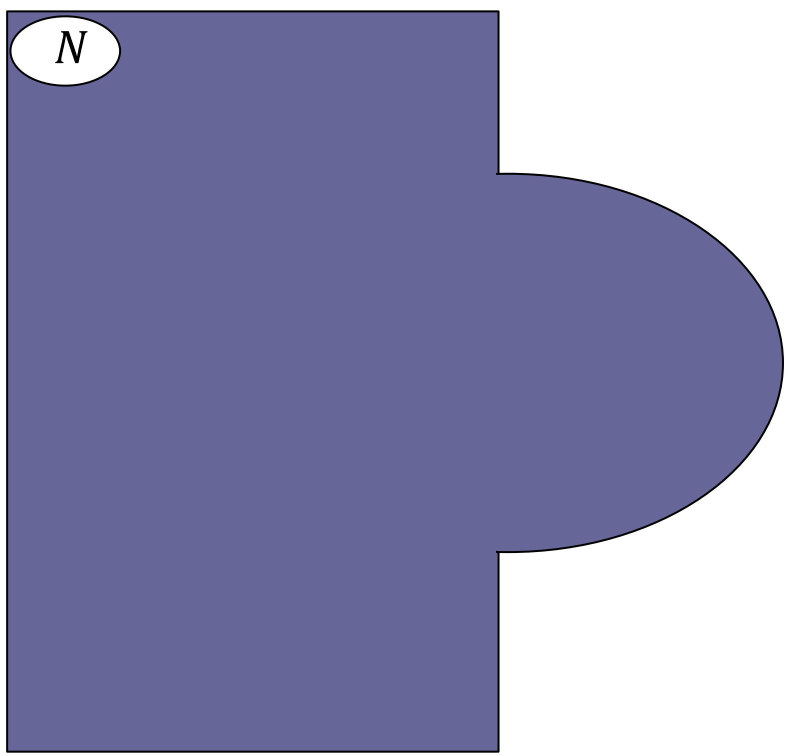
A *basic component* defined by intersections and unions of signed surfaces

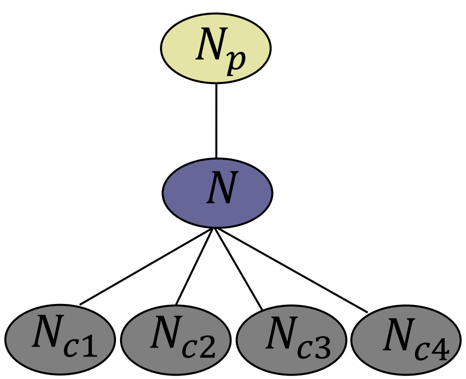
$$\left(-S_{blue} \cap S_{grey} \cap S_{green} \cap -S_{orange}\right) \cup -S_{yellow}$$



Model Representation
Component Hierarchy: Boolean Formulae

Basic comp: $B(N)$, $\cup$ and $\cap$ of signed surfs.

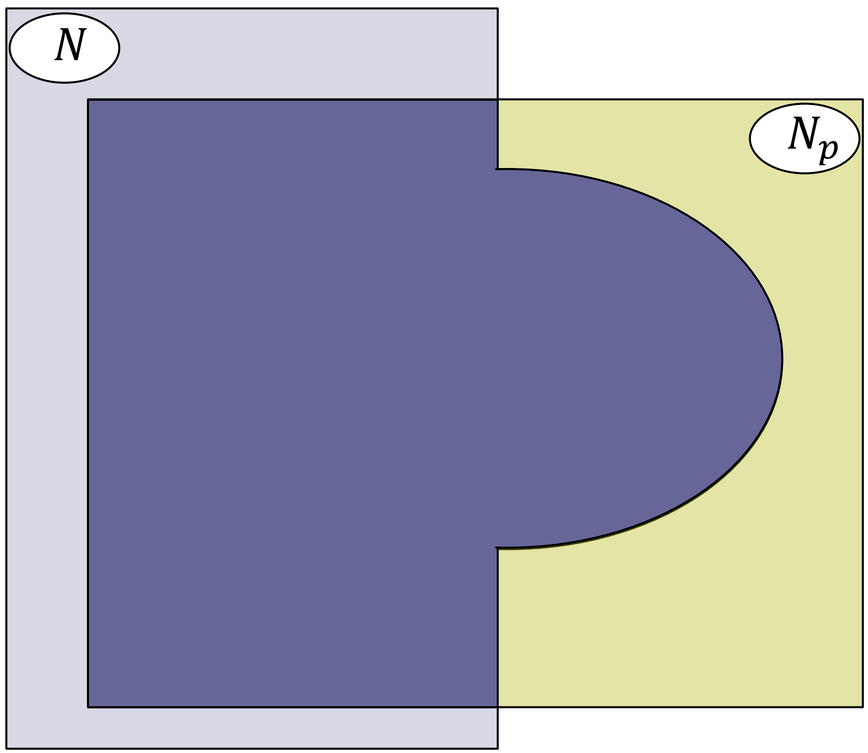


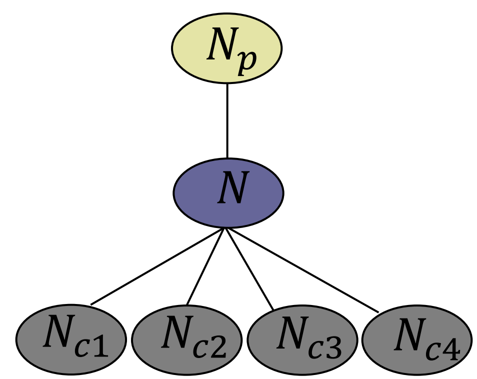


Model Representation
Component Hierarchy: Boolean Formulae

Basic comp: $B(N)$, $\cup$ and $\cap$ of signed surfs.

Restricted comp: $R(N) = B(N) \cap R(N_p)$



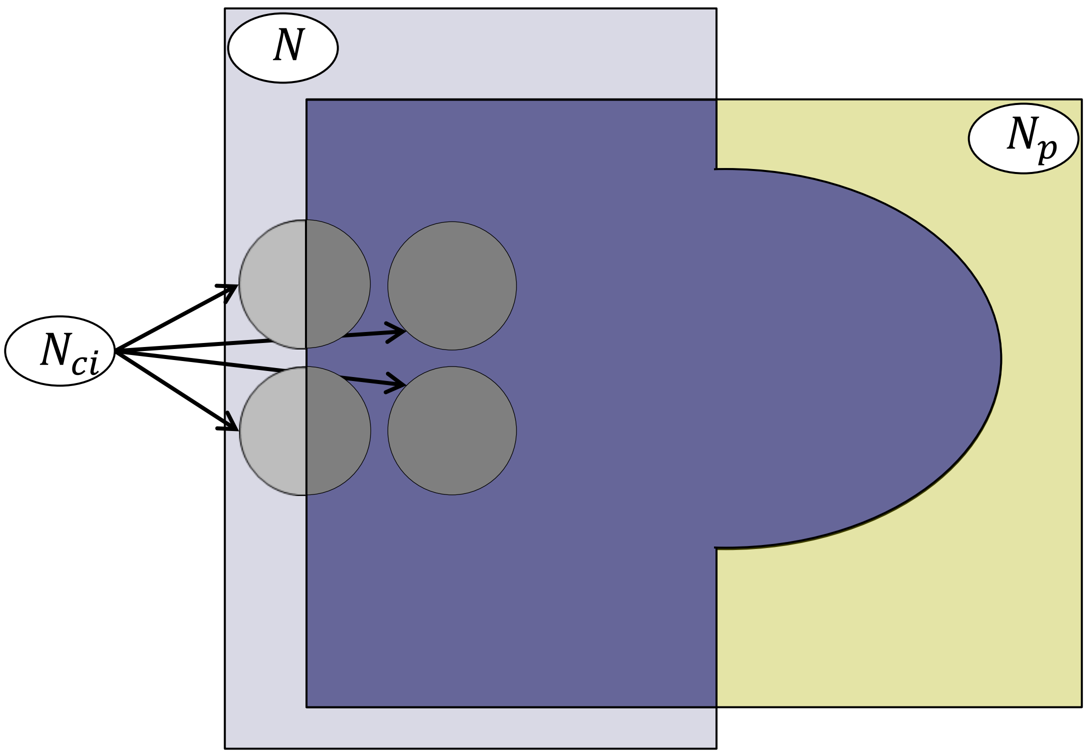


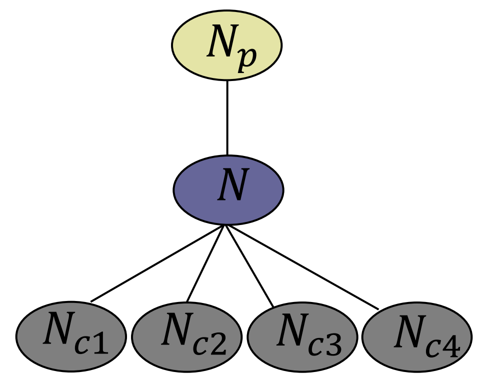
Model Representation
Component Hierarchy: Boolean Formulae

Basic comp: $B(N)$, $\cup$ and $\cap$ of signed surfs.

Restricted comp: $R(N) = B(N) \cap R(N_p)$

Hierarchical comp: $H(N) = R(N) \setminus \sum_i R(N_{ci})$

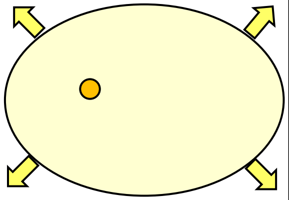




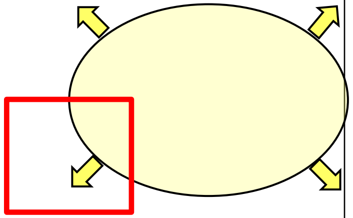
Operations on Primitives

Operations on signed surface S with point or box:

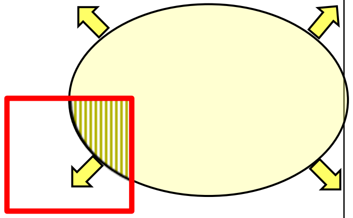
- *Point inside* – return if query point is inside S.



- *Box classification* – return if the points of an axis-aligned box are inside, outside or both with respect to S.

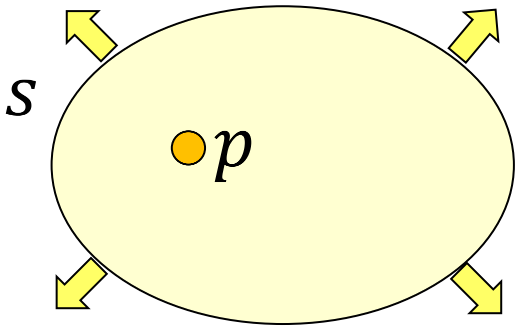


- *Integrator* – return the intersection volume of the interior of S with an axis-aligned box.



Analyzing Precision [LPT99]

Point inside – return if query point is inside S.

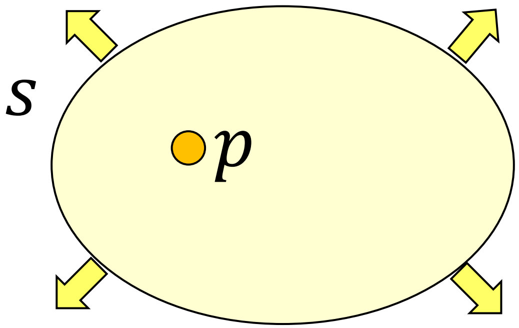


$\text{pointInside}(s, p)$

$$p = (p_1, p_2, p_3)$$
$$s = (s_1, s_2, \ldots, s_{10})$$
$$p_i, s_i \in \{-U, \ldots, U\}$$

Analyzing Precision [LPT99]

Point inside – return if query point is inside S.



$$p = (p_1, p_2, p_3)$$

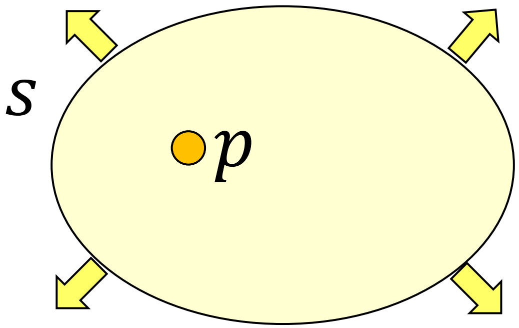
$$s = (s_1, s_2, \dots, s_{10})$$

$$p_i, s_i \in \{-U, \dots, U\}$$

$$\text{pointInside}(s, p) = \text{sign}(\, s_1 p_1^2 + s_2 p_2^2 + s_3 p_3^2 \\ + s_4 p_1 p_2 + s_5 p_1 p_3 + s_6 p_2 p_3 \\ + s_7 p_1 + s_8 p_2 + s_9 p_3 + s_{10})$$

Analyzing Precision [LPT99]

Point inside – return if query point is inside S.

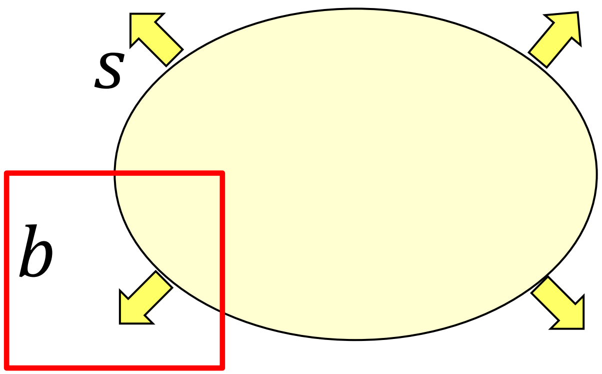


$$p = (p_1, p_2, p_3)$$
$$s = (s_1, s_2, \dots, s_{10})$$
$$p_i, s_i \in \{-U, \dots, U\}$$

$$\text{pointInside}(s, p) = \text{sign}(\, s_1 p_1^2 + s_2 p_2^2 + s_3 p_3^2$$
$$+ s_4 p_1 p_2 + s_5 p_1 p_3 + s_6 p_2 p_3$$
$$+ s_7 p_1 + s_8 p_2 + s_9 p_3 + s_{10})$$
$$= \text{sign}\,(\,3\,)$$

Box classification test

Box classification – return if the points of an axis-aligned box are inside, outside or both with respect to S.



$$b = (b_1, p_2, \dots, p_6)$$
$$s = (a_1, a_2, \dots, a_{10})$$
$$b_i, s_i \in \{-U, \dots, U\}$$

classify(s, b)

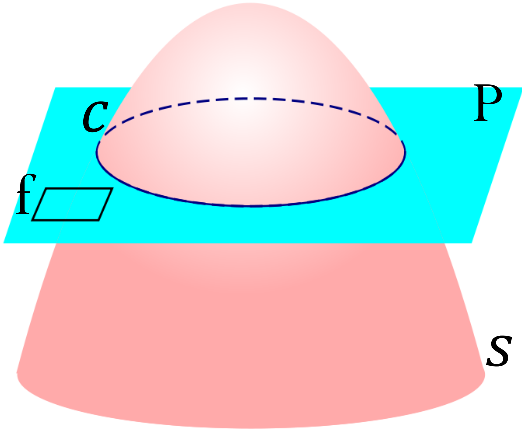
(1) check if any vertices of *b* are on different sides of *s*. -- *Degree 3*

(2) check if any edge of *b* intersect *s*. -- *Degree 4*

(3) check if any face of *b* intersects *s*. -- *Degree 5*

Face test

Test if a face f intersects s.

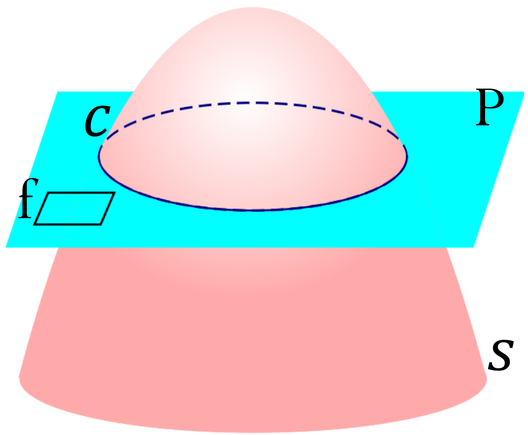


Let c be the intersection curve of the plane P containing the face and s.

$$c(x,y) = \begin{pmatrix} x & y & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \\ 2 & 2 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Face test

Test if a face f intersects s.



Let c be the intersection curve of the plane P containing the face and s.

$$c(x,y) = \begin{pmatrix} x & y & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \\ 2 & 2 & 3 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

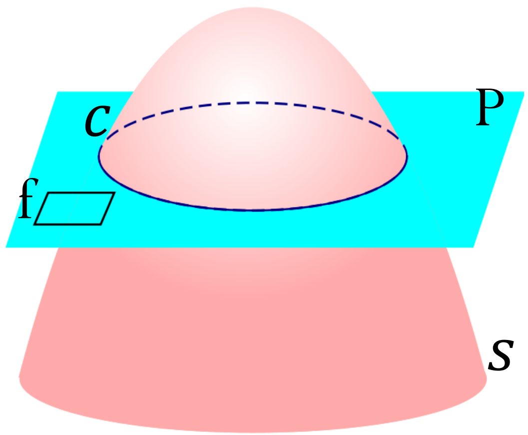
To determine if s intersects the face test properties of the matrix.

Test if c is an ellipse: $\operatorname{sign}\left(\begin{vmatrix} 1 & 1 \\ 1 & 1 \end{vmatrix} \right) = 2$

Test if c is real or img: $\operatorname{sign}\left(\begin{vmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \\ 2 & 2 & 3 \end{vmatrix} \right) = 5$

Face test

Test if a face f intersects s.



Let c be the intersection curve of the plane P containing the face and s.

$$c(x,y) = \begin{pmatrix} x & y & 1 \end{pmatrix} \begin{pmatrix} \textcircled{1} & \textcircled{1} & \textcircled{2} \\ \textcircled{1} & \textcircled{1} & \textcircled{2} \\ \textcircled{2} & \textcircled{2} & \textcircled{3} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

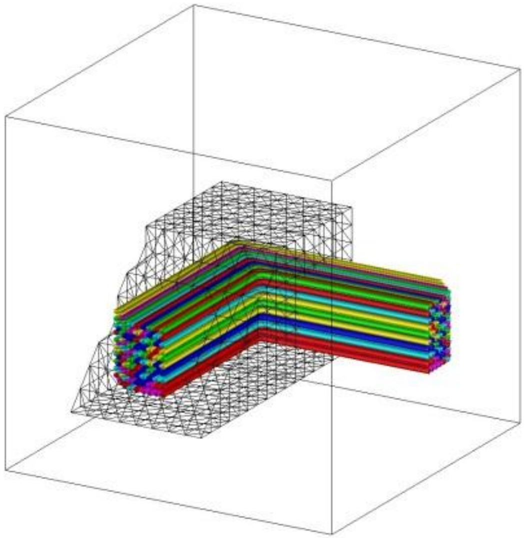
To determine if s intersects the face test properties of the matrix.

Test if c is an ellipse: $\operatorname{sign}\left(\begin{vmatrix} \textcircled{1} & \textcircled{1} \\ \textcircled{1} & \textcircled{1} \end{vmatrix} \right) = \textcircled{2}$

Test if c is real or img: $\operatorname{sign}\left(\begin{vmatrix} \textcircled{1} & \textcircled{1} & \textcircled{2} \\ \textcircled{1} & \textcircled{1} & \textcircled{2} \\ \textcircled{2} & \textcircled{2} & \textcircled{3} \end{vmatrix} \right) = \textcircled{5}$

Can we reduce the box test to degree 4?

Experiment: Accuracy and Time



Algorithm	Requested Accuracy	Error	Time (sec)
Old	**1e-4**	**<1e-5**	**790.28**
New	**1e-4**	**<1e-6**	**1.41**

Conclusion

Current challenges:

- Lower degree box classification
- Tighter error bounds
- Translating other problems from reactor physics into the language of computational geometry.

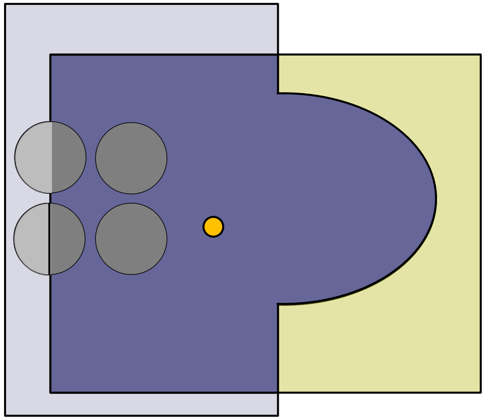
Contact:

David L. Millman

dave@cs.unc.edu

http://cs.unc.edu/~dave

Model Operations
Component Hierarchy: Boolean Formulae



Operations for a comp. hierarchy:

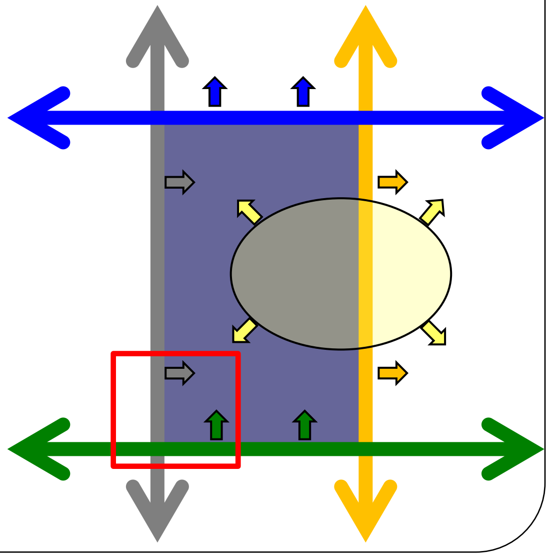
- *Point location* – return the hierarchical component containing a point.

- *Formula restricted to a box* – given an axis aligned box b, a Boolean formula F and the classification for all surfs of F for b, replace all surfs of F in which b is completely inside or outside with True or False respectively.

Model Operations
Component Hierarchy: Boolean Formulae

- *Formula restricted to a box* – given an axis aligned box b, a Boolean formula F and the classification for all surfs of F for b, replace all surfs of F in which b is completely inside or outside with True or False respectively.

$$(-S_{blue} \cap S_{grey} \cap S_{green} \cap -S_{orange}) \cup -S_{yellow}$$

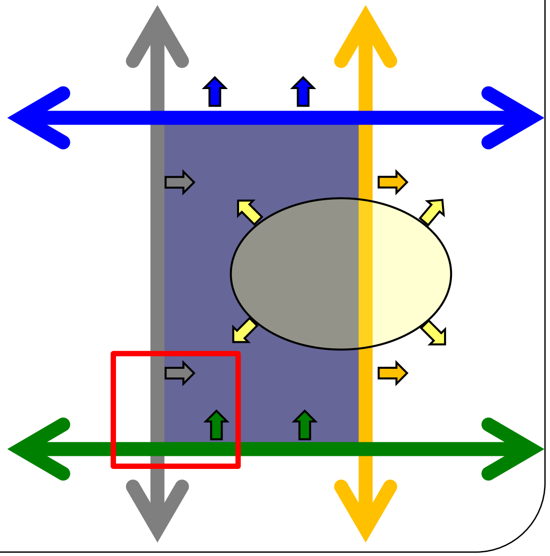


Model Operations
Component Hierarchy: Boolean Formulae

- *Formula restricted to a box* – given an axis aligned box b, a Boolean formula F and the classification for all surfs of F for b, replace all surfs of F in which b is completely inside or outside with True or False respectively.

$$(-S_{blue} \cap S_{grey} \cap S_{green} \cap -S_{orange}) \cup -S_{yellow}$$

$$(\quad T \quad \cap S_{grey} \cap S_{green} \cap \quad T \quad) \cup \quad F$$



Model Operations

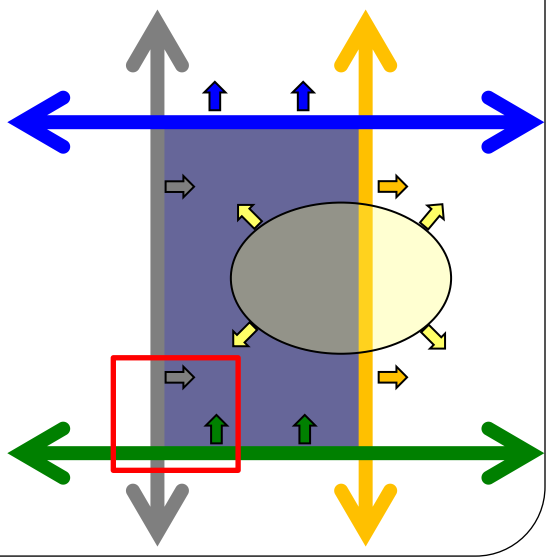
Component Hierarchy: Boolean Formulae

- *Formula restricted to a box* – given an axis aligned box b, a Boolean formula F and the classification for all surfs of F for b, replace all surfs of F in which b is completely inside or outside with True or False respectively.

$$\left(-S_{blue} \cap S_{grey} \cap S_{green} \cap -S_{orange}\right) \cup -S_{yellow}$$

$$\left(\quad T \quad \cap S_{grey} \cap S_{green} \cap \quad T \quad \right) \cup \quad F$$

$$\left(S_{grey} \cap S_{green}\right)$$



Surface-in-Box Integrators

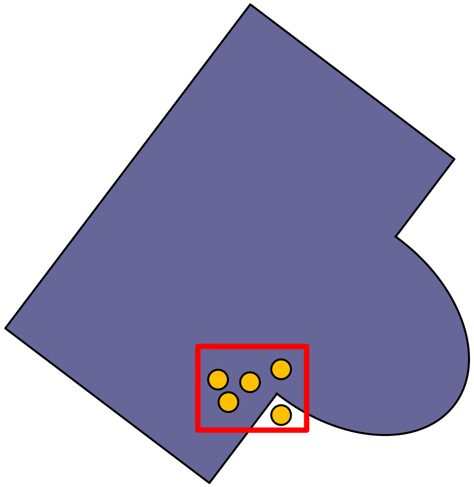
Given a component hierarchy, axis-aligned box b,
and target error ε and confidence δ,
an *integrator* either computes volumes of each
hierarchical comp's intersection with B to within ε and δ,
or flags B as "needs subdivision."

Basic integrators:

- *Monte Carlo Integrator (MC)*
- *Box Integrator (Box)*

Advanced integrators:

- *Pair of Planes Integrator (2Plane)*
- *Bundle of Cylinders Integrator (BunCyl)*



Surface-in-Box Integrators

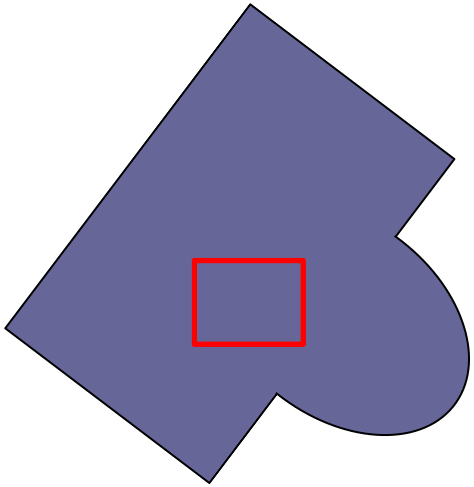
Given a component hierarchy, axis-aligned box b,
and target error ε and confidence δ,
an *integrator* either computes volumes of each
hierarchical comp's intersection with B to within ε and δ,
or flags B as "needs subdivision."

Basic integrators:

- *Monte Carlo Integrator (MC)*
- *Box Integrator (Box)*

Advanced integrators:

- *Pair of Planes Integrator (2Plane)*
- *Bundle of Cylinders Integrator (BunCyl)*



Surface-in-Box Integrators

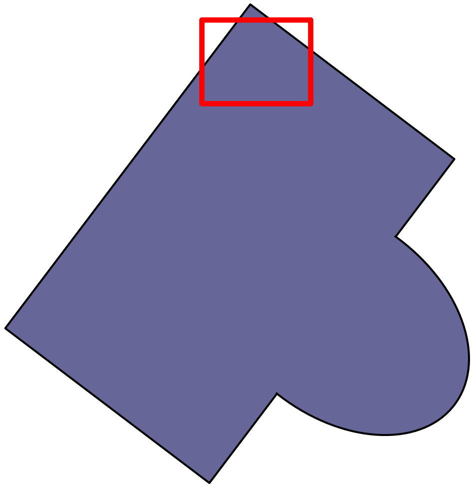
Given a component hierarchy, axis-aligned box b,
and target error ε and confidence δ,
an *integrator* either computes volumes of each
hierarchical comp's intersection with B to within ε and δ,
or flags B as "needs subdivision."

Basic integrators:

- *Monte Carlo Integrator (MC)*
- *Box Integrator (Box)*

Advanced integrators:

- *Pair of Planes Integrator (2Plane)*
- *Bundle of Cylinders Integrator (BunCyl)*



Surface-in-Box Integrators

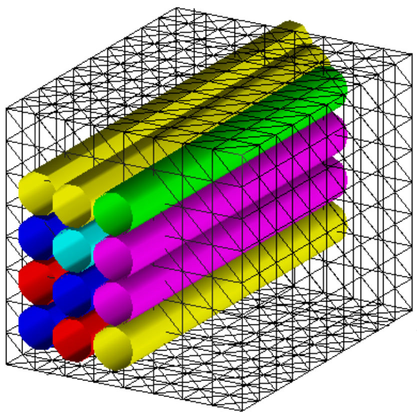
Given a component hierarchy, axis-aligned box b,
and target error ε and confidence δ,
an *integrator* either computes volumes of each
hierarchical comp's intersection with B to within ε and δ,
or flags B as "needs subdivision."

Basic integrators:

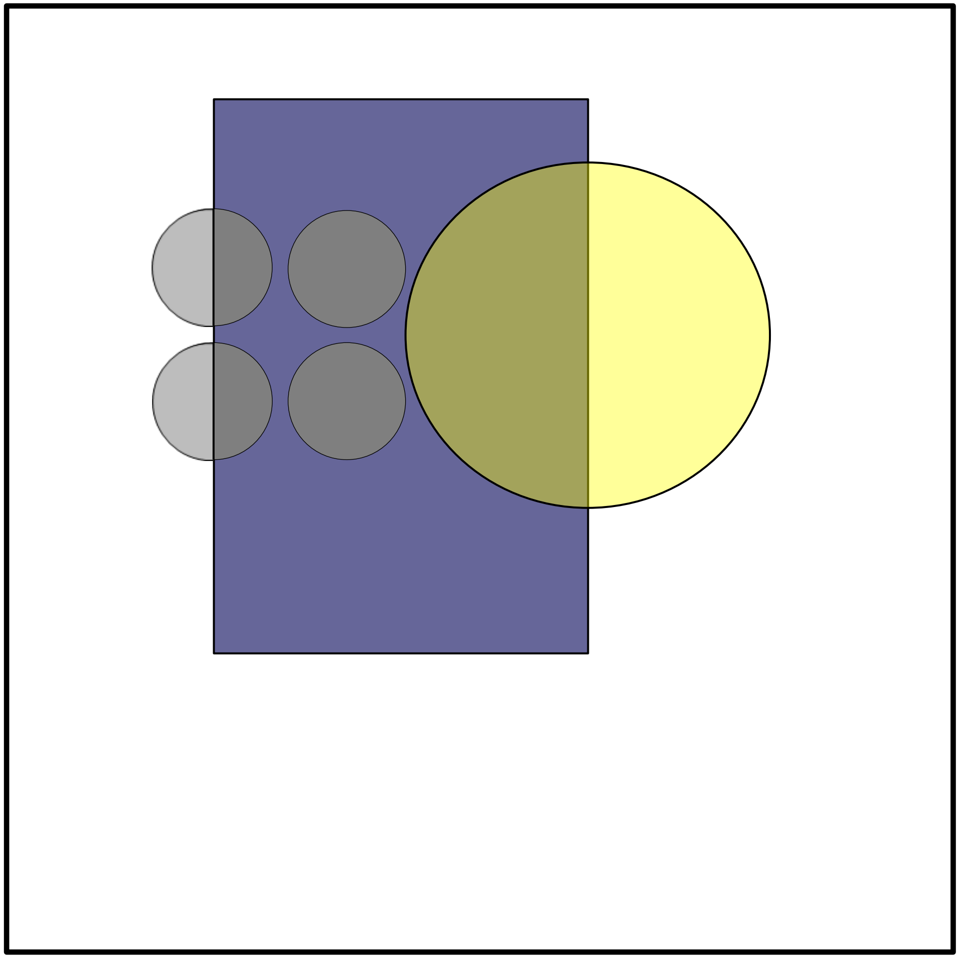
- *Monte Carlo Integrator (MC)*
- *Box Integrator (Box)*

Advanced integrators:

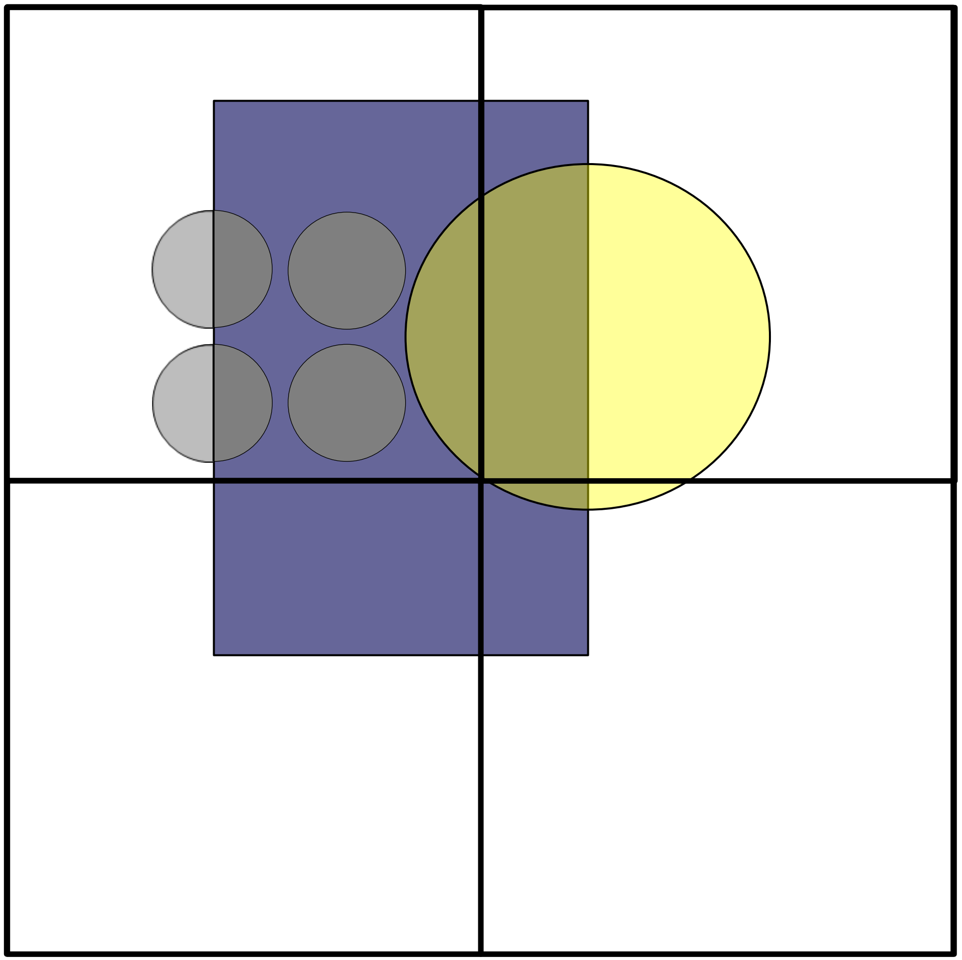
- *Pair of Planes Integrator (2Plane)*
- *Bundle of Cylinders Integrator (BunCyl)*



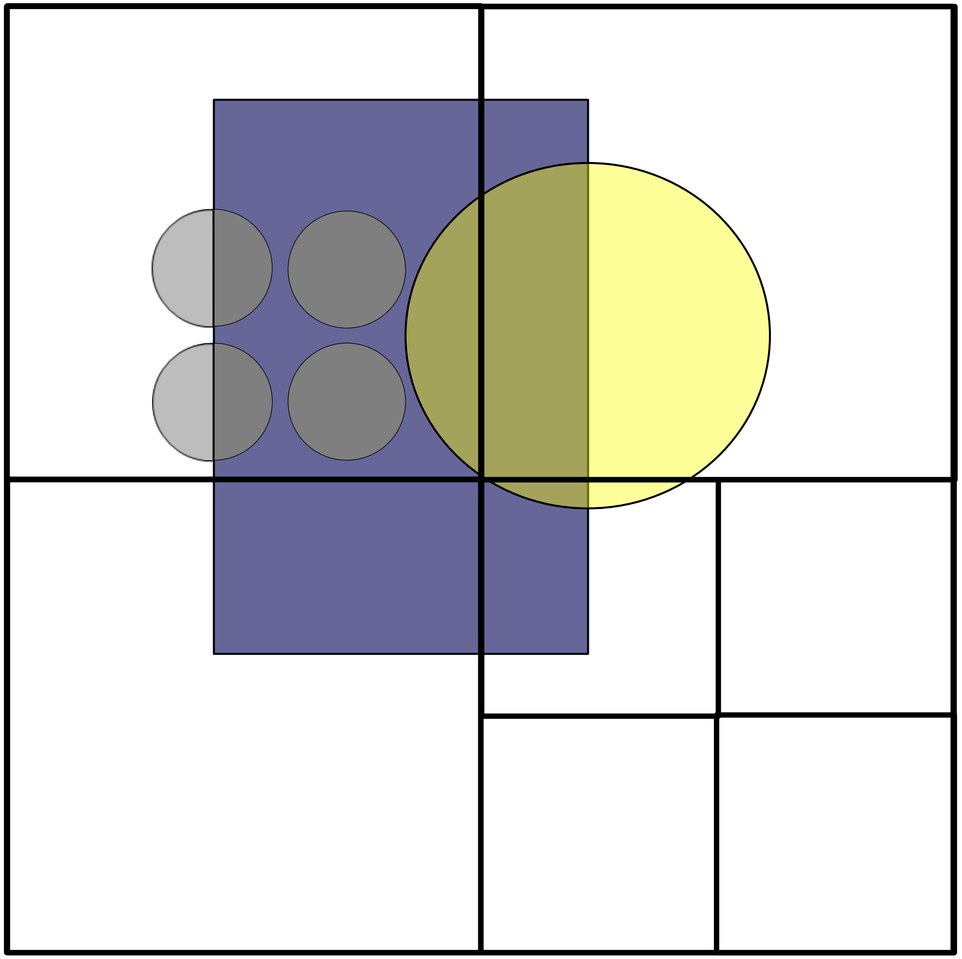
Algorithm Animation



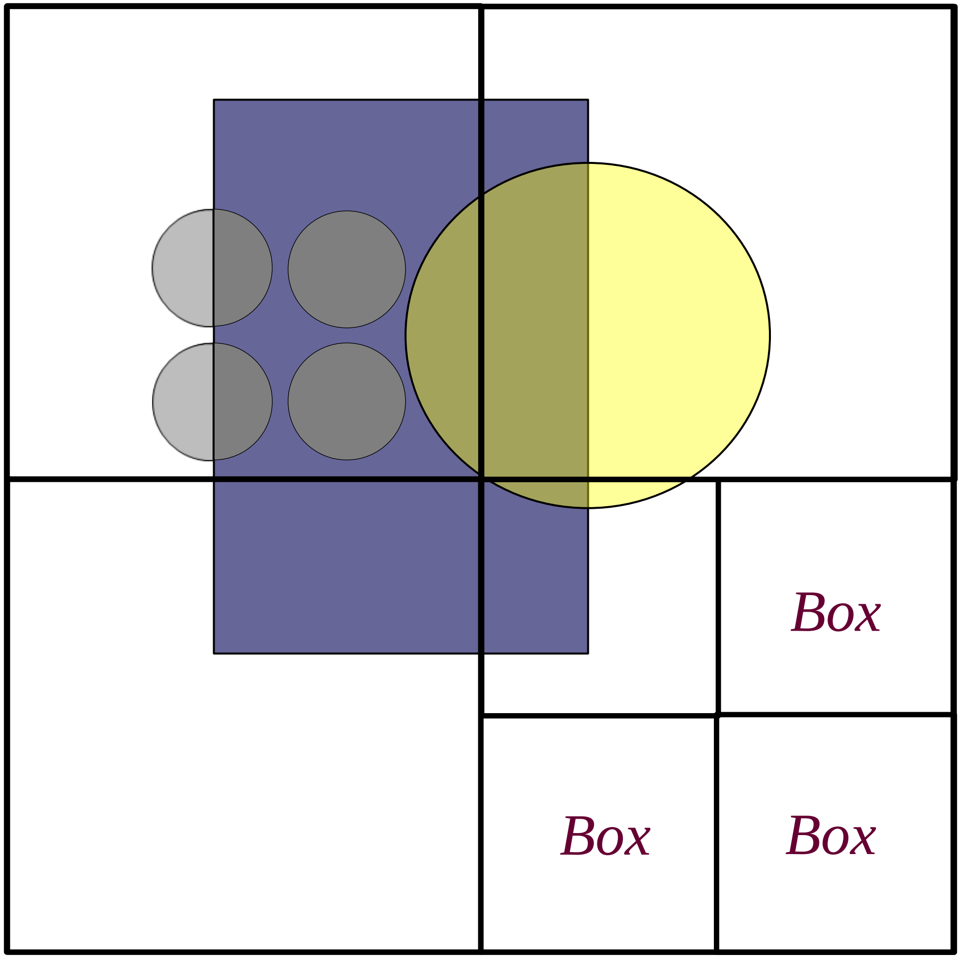
Algorithm Animation



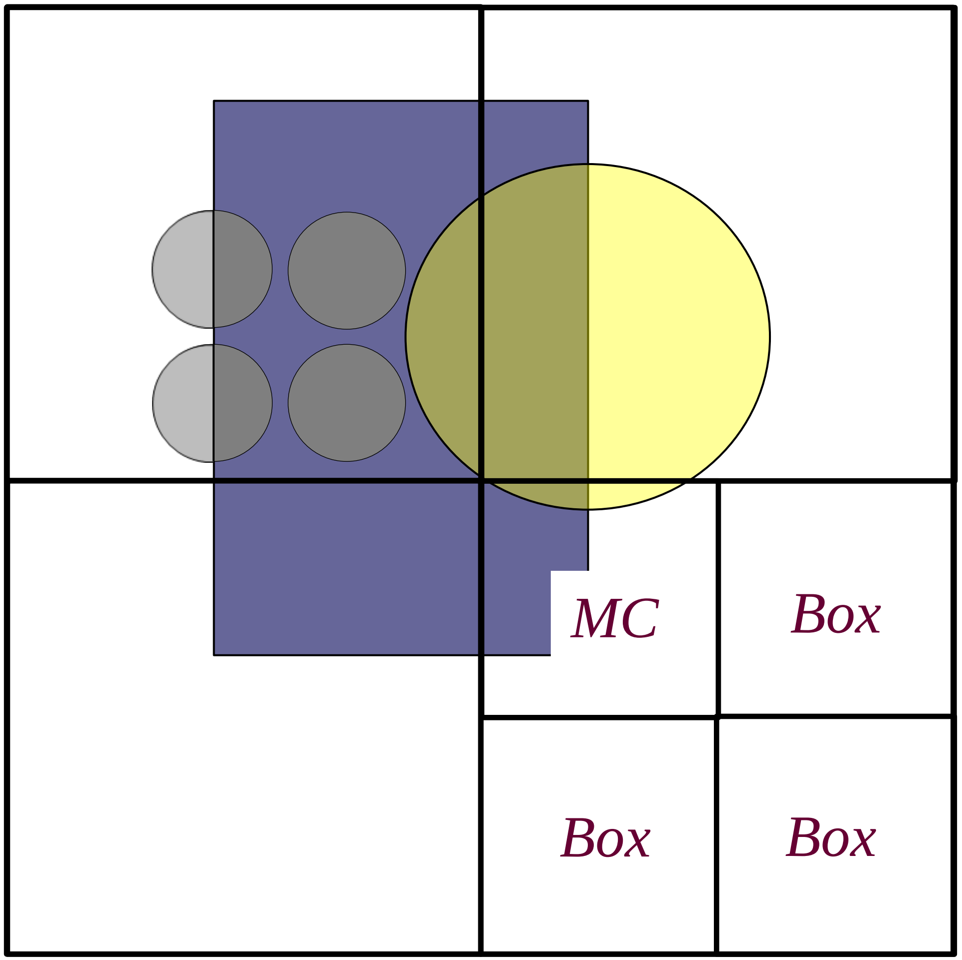
Algorithm Animation



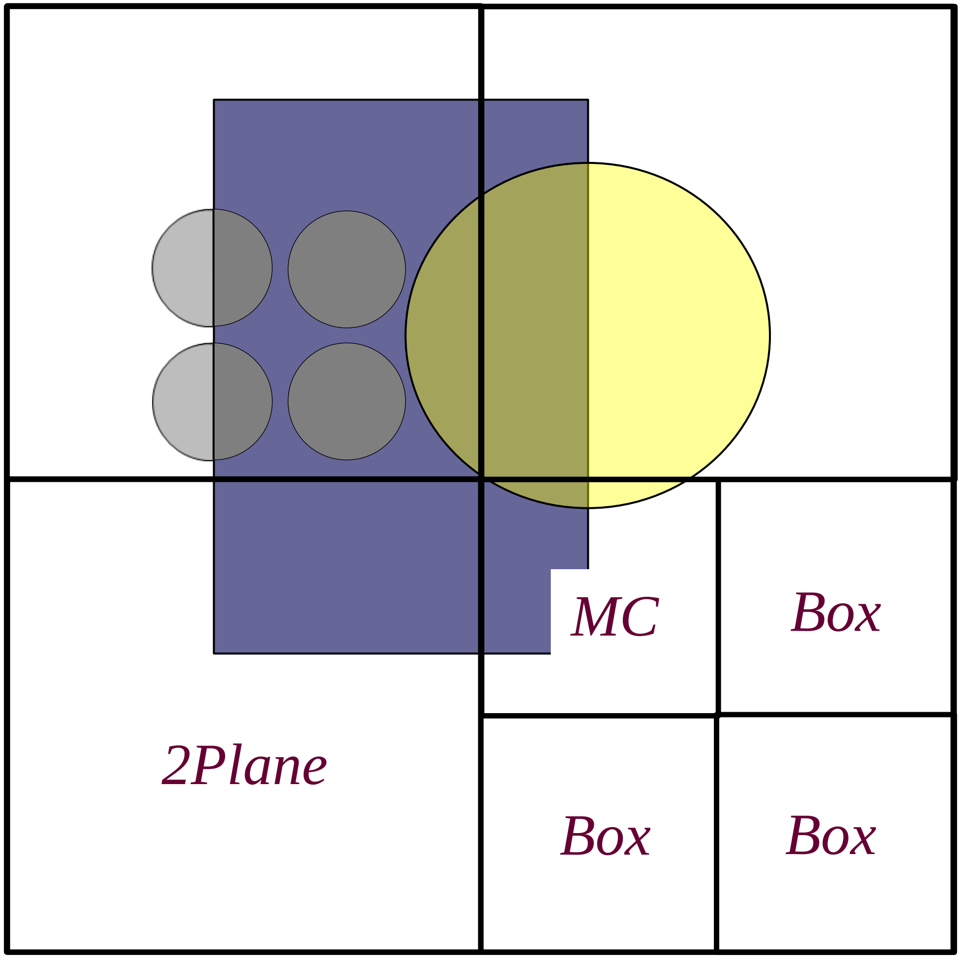
Algorithm Animation



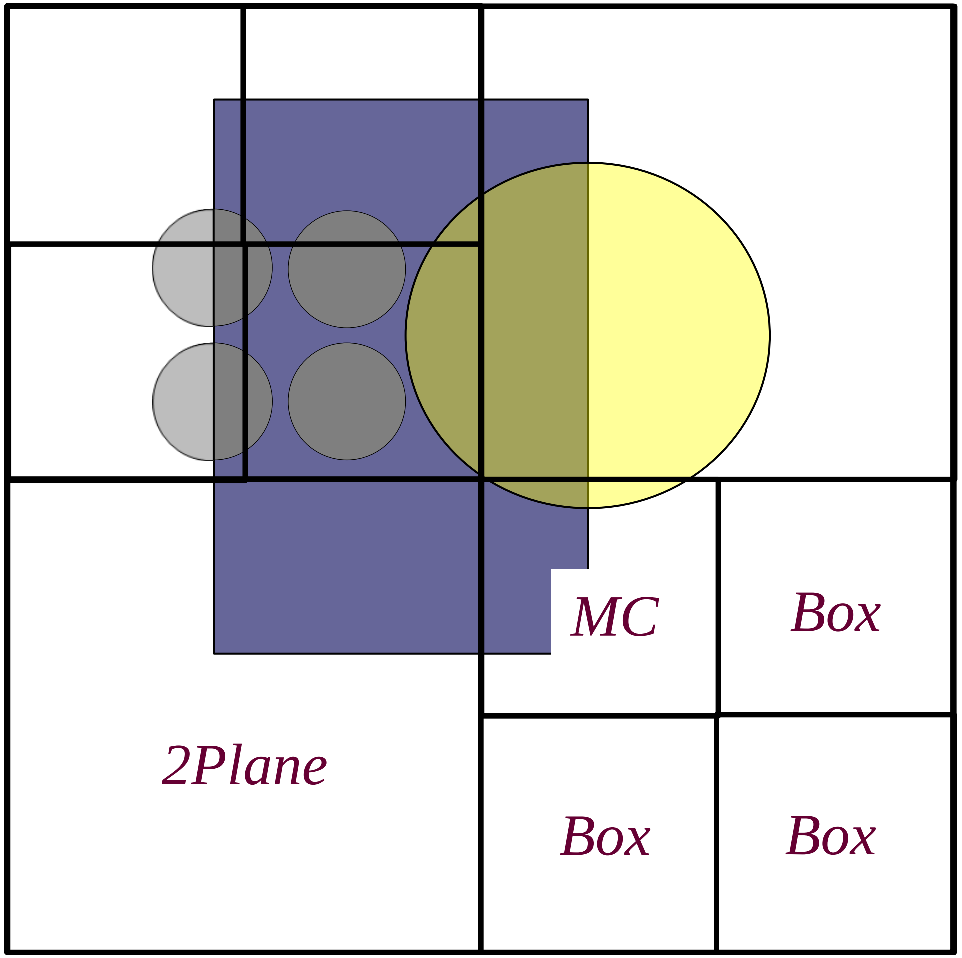
Algorithm Animation



Algorithm Animation



Algorithm Animation



Algorithm Animation