Representations of geometric objects:

There are a variety of options, each supports some queries well, others poorly, not aware of anything that does everything well.

Considerations:

- Should the model be a solid?
- Do you care about boundaries (e.g., curvature, normals, contact, cutting, etc.)
- Will volumes?
  (e.g., being a solid, path planning, some kinds of collision)
- Will you intersect, union, etc. objects?
- At what resolution do you wish to rep.?
  Will that change?
- Will there be interactive modelling?
- Are you going to manufacture?
- Do you care to model physical properties?
  (stiffness, etc.)
Voxels

- Insert a grid in space, label if occupied else zero

+ Powerful, flexible; volume easy to estimate; ignores detail below some scale

- Ignores detail below some scale; can be wasteful of space; can give poor reports of some objects.

+ Fast intersection, union, models are solids, relatively easy to render; ray carving algorithms allow some easy modelling of free space

Octahedrons

+ More space-efficient than voxels

- Savings can be hard to get.
CSG (constructive solid geometry)

- Select primitives, usually simple
  (often:)
  always
  - sphere
  - infinite cylinder
  - half-space
  sometimes
  - cylinder w/ given x-sector
    (extruded curve)
  - volume created by spinning curve around axis

- Now object is logical combination of
  primitives
  - operators
    - rotate, translate, scale
    - \( \bigcup, \cap, \setminus \) cartesian
      subtraction

- Notice: these operators need to be
  stabilized.
\[(\text{translate}(A) \cap B) \cup \text{translate}(C)\]

In the worst case, this could be a sphere w/a whisker!

So define

- \(\partial\) - takes boundary of set
- \(i\) - takes interior
- \(c\) - takes closure.

Then:

\[A \cap^* B = c\left(i(B) \cap i(A)\right)\]

gets rid of this problem

\[A \cup^* B = c\left(i(A) \cup i(B)\right)\]

etc.
CSG has some advantages

- wo/ regularized operators objects are solids
- model contains a coherent, slightly semantic record of how it was made.
- Good choices of primitives lead to
  - easy alg's for
    - ray tracing
    - union
    - intersection
    - volume computation

But
- v hard to tell if two models are the same.
- v hard to get to CSG from any other rep.
  
Raytracing CSG

- assume primitives admit exact intersections / rays (rootfinder).
- Now a ray to prims, reason about first intersection pt.
eg. \( \text{op} (A \otimes, B) \)

\[ \begin{array}{c}
\text{Ray} \\
\text{local pt} \\
\text{in } A \\
\text{in } B \\
\text{out } A \\
\text{out } B.
\end{array} \]

\[ \text{first } \cap \text{ with } A \cup B. \]

\[ \text{first } \cap \text{ with } A \cup B. \]

\[ \text{notice} \]
\[ \text{regularizing operator can be achieved by allowing pts to "cancel" } \]
\[ A \otimes B \]

\[ \text{eg. } A \otimes B \]

\[ \begin{array}{c}
\text{Ray} \\
\text{in } A \\
\text{Bin, out on top of one another} \\
\text{Bin by } e \text{ towards Bin} \\
\text{Bin by } e \text{ towards Bin} \\
\rightarrow \text{ No } \cup
\end{array} \]

\[ \begin{array}{c}
\text{OR} \\
\text{Bin, out enters + leaves } A \cup B \\
\text{at same location} \\
\text{ cancel, ray doesn't enter}
\end{array} \]
Scan to voxels/octree

- easy, if primitives are convex
- keep track of fact there are 3 types of box
  - all verts in = inside
  - otherwise = boundary
  - all verts out = outside
- I know no alg to go from boxes to CSG better than
  \[ \bigcup (\text{all boxes}) \]
  (which gives bad CSG cash)

Big issue for vision

- Many ways to represent the same thing as CSG
  + (maybe) nice story about increasing detail
- For $\phi(x)$ a continuous fn, $\phi(x) > 0$ gives a solid.
- Generally hard to work with in this generality.

Algebraic

- $\phi$ polynomial.
- Can use this as CSG primitive.

BUT

- Raytracing is tricky.
- Exact roots of polynomial.
- Use Sturm sequence + deflation.

BUT

- Telling whether a voxel lies in or out is tricky to handle.
  Simplest cases, EVEN 2D.
Sketch of Sturm sequences.

- I wish to know if a polynomial \( p(x) \) of one var, \( x \), has a root in the interval \([x_0, x_1]\).

- If \( p(x) \) is linear, easy - test signs at either end of interval.

Quadratic case:

- \( + - \) → guarantees 1 root
- \( + + \) - no guarantees!
- \( - - \) - no roots

Notice, if you see \( +, + \), to have 2 roots, the derivative must have a zero.

So \( + + \) poly means \( + + \) deriv means \( \boxed{\text{no roots}} \).
But $p + x + - \frac{1}{x} + \frac{1}{x}$ doesn't guarantee roots.

Example:

Now consider $p = ax^2 + bx + c$.

- $\text{rem}(p, p_x) = b^2 - 4ac = \delta$

\[ \begin{array}{c|c|c}
+ & + & \leftarrow p \\
- & + & \leftarrow p_x \\
+ & + & \leftarrow \text{rem} \\
\end{array} \quad \rightarrow \quad 2 \text{ roots}
\]

\[ \begin{array}{c|c|c}
+ & + & \leftarrow p \\
- & + & \leftarrow p_x \\
+ & + & - \leftarrow \text{rem} \\
\end{array} \quad \rightarrow \quad 0 \text{ roots}
\]

\[ \begin{array}{c|c|c}
+ & + & \leftarrow p \\
- & + & \leftarrow p_x \\
- & + & \leftarrow \text{rem} \\
\end{array} \quad \rightarrow \quad 1 \text{ root.}
\]
general construction

\[
p_1 = p \\
p_2 = p \\
p_n = -\text{rem}(p_{n-1}, p_{n-2}) \\
\vdots
\]

all the way to a constant

\[
\# \text{ of sign changes at one end} = \# \text{ roots}
\]

- \# at other

NOTICE THE PROBLEM:

coefficients of \( p_n \) are high degree in coeff of \( p \),
\( \phi(x, y) = 0 \), \( \phi \) a cubic

Example, in 2D

- fairly obvious algorithm may suggest itself
  - check
    1. all verts in/out
    2. no \( \lor \) \( \land \) faces.

- this is OK, not great in 2D.

- serious probs in 3D

**REQUIRED**

test that, in some block, a polynomial has no zeros
This test, a multivariate Sturm sequence exists, but is of no practical use in any case of interest.
Shape from contour

- Intriguing, classic, vision qu.
- I supply an object contour in various forms:
  - outline silhouette only
  - silhouette + internal boundaries
  - + creases?
- Produce a reasonable estimate of shape

- Hard, pretty obviously
- People do this quite well, though not clear how one measures this precisely
Some strategies from literature

- Assume view is orthographic (universal)
- Assume surface is SOR, and axis is parallel to
  is at known angle to image plane.
- Assume SHGC, x-section observed, axis as above

doesn't happen often - very special cases.

- Build a height map
- Issue: reconstruction is not height map at contour
- Issue: what is height of contour
Now, notice at outline points, normal is \((0, c, 1)\) \((x, \beta, 0)\)

But for \((x, y, z(x,y))\), normal is

\[
\left( -\frac{z_x}{\sqrt{1 + z_x^2 + z_y^2}}, -\frac{z_y}{\sqrt{1 + z_x^2 + z_y^2}}, 1 \right)
\]

\((i.e.\ at\ contour,\ z_x, z_y \to 0)\)

So height map doesn't work.
Strategy 1

"ignore" this issue, argue that we do not WANT this constraint to be met.

Eg: for reconstruction, may view recon at new angle - what texture, geometry should we use?

[Diagram showing view direction changing]

Can't see - OK

What goes here?
Instead:

recon w/ crease at boundary

crease:

use symmetry

This can be achieved by:

- assume CG is plane (wrong, but)

- now est normal along outline ($x, \beta, 0$)

- Now use Poisson smoothing to get $x, \beta$ in interior
\[ \nabla^2 \alpha = 0 \quad \text{in } \mathbb{R} \]

\text{st. } \alpha = \text{meas} \text{ on } \partial \mathbb{R}

\text{etc.}

- now assume \( \frac{1}{\sqrt{1+p^2+q^2}} \)

- get unit normal field by

  \[ \alpha, \beta \text{ from above} \]

  \[ \mathbf{U.N.} = \left[ \alpha, \beta, \sqrt{1-\alpha^2-\beta^2} \right] \]

  \text{OK, by Poisson Smoothing}

  \[ = (n_0, n_1, n_2) \]

- Mod U.N. = \[
\begin{cases} 
(n_0, n_1, n_2) & \text{if } n_2 > 0 \\
\left( \frac{n_0}{n_0 + n_1 + n_2}, \frac{n_1}{n_0 + n_1 + n_2}, \frac{n_2}{n_0 + n_1 + n_2} \right) & \text{otherwise}
\end{cases}
\]

  \[ = \left( n_0^*, n_1^*, n_2^* \right) \]
Then solve by integration.

$$\min_h \left( \left( h_x + \frac{n_0^*}{n_2^*} \right)^2 + \left( h_y + \frac{n_1^*}{n_2^*} \right)^2 \right)$$

s.t. \quad h = 0 \quad on \ boundary

Variants are likely possible, but this is quite good.
Alternative (Prasad + Zisserman)

Choose curves in \( s,t \) to csp to outlines (e.g., \( t = \text{const} \)); csp between curves

Choose cost fn. e.g.

Minimize linearized bending energy

\[
\int \left( (x_{ss})^2 + 4(x_{st})^2 + (x_{tt})^2 \right) \, ds \, dt
\]

Then we get linear constraints from outline
\[ x(s,t) = (x_0(s,t), x_1(s,t), x_2(s,t)) \]

then

\[ x_0(s, \text{const}) = \text{outline} \, x(s) \]
\[ x_1(s, \text{const}) = \text{outline} \, y(s) \]

on outline points

\[ n(s) \cdot x(s, \text{const}) = 0 \]

\[ n(s) \cdot x_2(s, \text{const}) = 0 \]

i. Grid up \((s,t)\); consider

\[ x_0(s,t), x_1(s,t), x_2(s,t) \] on grid

Becomes quadratic problem in node values w/ linear constraints

\[ \rightarrow \text{easy} \]
Super Quadrics

\[
\left(\frac{x}{A}\right)^r + \left(\frac{y}{B}\right)^s + \left(\frac{z}{C}\right)^t = 1
\]

where \( r, s, t \) are not necessarily integer

**ADV:** blobby objects, can make sharp corners

**DISADV:** most ops slower than for algebraic (flat fractional power).

**Meta-Balls:**

- Define some reference objects (pts, lines, etc).

  \( \cdot \)

- Define a distance cost fn. \( f(r) \)

  \[ f(r) = (1 - r^2)^2 \]

- Volume = \( \sum_{i} f(\|x - o\|) < \text{threshold} \).
ADV:
- rich families of quite complex shapes
- seems like an obvious application to shape from contour (cf. Adelson's putfball below)

DISADV:
- mechanics, rendering, etc., fairly complex, comparable to low degree algebraic case

PUFFBALLS
- see paper.