Do It Yourself

DIY CAD
Computer Aided Design

Mathias Plans
Plan

- Motivation
- Boundary Representation
- Geometry Kernel
Why am I doing this?

Thesis!
Why am I doing this?

Thesis!

Grammar designer

CAD

Grammar Interpreter

Real-Time Application

User
Why am I doing this?

Thesis!

Grammar designer

- CAD
- Grammar Interpreter
- Real-Time Application

User
We want meshes in our application

We want meshes in our application, but they are inefficient for CADs.

https://structures.aero/software/femap/cad-import/
Solids are purely parametric
Cell decomposition

- Represent a solid as unit-sized elements or as a tree
Cell decomposition

- Represent a solid as unit-sized elements or as a tree
- Start with a single cell that encompasses the whole scene
Cell decomposition

- Represent a solid as unit-sized elements or as a tree
- Start with a single cell that encompasses the whole scene
- If a cell is fully inside or outside the solid, do nothing
- If a cell is half in solid, half outside solid, split into sub-cells
Cell decomposition

- Represent a solid as unit-sized elements or as a tree
- Start with a single cell that encompasses the whole scene
- If a cell is fully inside or outside the solid, do nothing
- If a cell is half in solid, half outside solid, split into sub-cells
- Continue until desired resolution is achieved
EMPTY cell

MIXED cell

FULL cell

https://shengchen-liu.github.io/robotics_gitbook/path_planning/approximate_cell_decomposition.html
Cell decomposition

A path is searched in the diagram between $q_{\text{init}}$ and $q_{\text{goal}}$. The free path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$. A path between $q_{\text{init}}$ and $q_{\text{goal}}$ is found by the method, if any, of three subpaths from $q_{\text{init}}$ to $q_{\text{goal}}$.
General sweeping

- Extrudes 2D shapes along general curves
- The class of representable shapes is narrow
General sweeping
Constructive solid geometry

- Set of primitive solids
- Can take an intersection of two solids
- Can take a union of two solids
- Can take difference of two solids

https://en.wikipedia.org/wiki/Constructive_solid_geometry
Boundary representation

- Represents solids by their skins
- Solid is a collection of surfaces that encapsulate it
More about BREP
Geometry

- Parametric objects
- Splines
- Curves
- Surfaces.

\[ \begin{align*}
  x(t) &= \cos(t) \\
  y(t) &= \sin(t) \\
  x(t) &= 3 \cos(t) \\
  y(t) &= 2 \sin(t) \\
  x(t) &= 2 \cos(x) + 5 \cos\left(\frac{2}{3}x\right) \\
  y(t) &= 2 \sin(x) + 5 \sin\left(\frac{2}{3}x\right)
\end{align*} \]
Topology

Defines how primitives are connected

Topology vs Geometry

Same geometry, different mesh topology

Same mesh topology, different geometry
Domain specific information

- Such as info about constraints
- Will not be touched in this seminar
Definitions
VERTEX: A vertex is a node of the data structure, lying at a point in space.
EDGE: An edge is a segment of a curve, running between two vertices.
LOOP: An ordered set of edges that form a cycle.
FACE: Faces are portions of surfaces. Faces are bounded by loops.
SHELL: Each closed set of faces in the object forms a Shell.
Many more

- Groups
- Wireframes
- Instances
Winged-edge link data structure

- Designed for quick modifications of topology
Winged-edge link data structure

- Designed for quick modifications of topology
- Start or end vertex (or both)

Winged-edge link data structure

- Designed for quick modifications of topology
- Start or end vertex (or both)
- Each vertex has connections to all the connected edges

Winged-edge link data structure

- Designed for quick modifications of topology
- Start or end vertex (or both)
- Each vertex has connections to all the connected edges
- Edges have connections to faces

Walkthrough

How a cube is created.
Alternatives

Figure 3.7: Loop-edge link connections

Figure 3.8: Vertex-edge link connections

Why?

- Solid is represented as a graph
Why?

- Solid is represented as a graph
- Traversing is easy
Why?

- Solid is represented as a graph
- Traversing is easy
- Localized modifications are simple to execute
Why?

- Solid is represented as a graph
- Traversing is easy
- Localized modifications are simple to execute
- This graph is *Eulerian*
Eulerian objects
Eulerian objects

- $V$ - vertices, $E$ - edges, $F$ - faces, $H$ - inner - or hole-loops, $G$ - genus, $B$ - multiplicity.
Eulerian objects

- $V$ - vertices, $E$ - edges, $F$ - faces, $H$ - inner - or hole-loops, $G$ - genus, $B$ - multiplicity.
- Genus – number of holes
- Multiplicity – number of shells
- Inner-loop – a loop inside another loop
- Hole-loop – inner-loop with extrusion
Eulerian objects

- $V$ - vertices, $E$ - edges, $F$ - faces, $H$ - inner - or hole-loops, $G$ - genus, $B$ - multiplicity.
- There cannot be negative number of elements.
Eulerian objects

- $V$ - vertices, $E$ - edges, $F$ - faces, $H$ - inner - or hole-loops, $G$ - genus, $B$ - multiplicity.
- There cannot be negative number of elements.
- If $V, E, F, H$ are 0, then $G$ and $B$ are also 0.
Eulerian objects

- $V$ - vertices, $E$ - edges, $F$ - faces, $H$ - inner - or hole-loops, $G$ - genus, $B$ - multiplicity.
- There cannot be negative number of elements.
- If $V, E, F, H$ are 0, then $G$ and $B$ are also 0.
- There is at least one vertex and at least one face.
Eulerian objects

- $V$ - vertices, $E$ - edges, $F$ - faces, $H$ - inner - or hole-loops, $G$ - genus, $B$ - multiplicity.
- There cannot be negative number of elements.
- If $V$, $E$, $F$, $H$ are 0, then $G$ and $B$ are also 0.
- There is at least one vertex and at least one face.
- The Euler–Poincare formula: $V - E + F - H = 2(B - G)$
Valid objects

- Can be represented as vectors (V, E, F, H, G, B)
Valid objects

- Can be represented as vectors \((V, E, F, H, G, B)\)
- Transforming one valid object to another produces a valid vector
Valid objects

- Can be represented as vectors \((V, E, F, H, G, B)\)
- Transforming one valid object to another produces a valid vector
- This transformation is called “Eulerian operator”
Valid objects

- Can be represented as vectors (V, E, F, H, G, B)
- Transforming one valid object to another produces a valid vector
- This transformation is called “Eulerian operator”
- There are 99 possible Eulerian operators
Basis of Eulerian operators

- Not all Eulerian operators have to be used
Basis of Eulerian operators

- Not all Eulerian operators have to be used
- The vectors have 6 coordinates
Basis of Eulerian operators

- Not all Eulerian operators have to be used
- The vectors have 6 coordinates
- It turns out that the Eulerian operators only have five dimensions
(1, 1, 0, 0, 0, 0) - MEV, Make an Edge and a Vertex
(0, 1, 1, 0, 0, 0) - MEF, Make a Face and an Edge
(1, 0, 1, 0, 0, 1) - MBFV, Make a Body (new shell), Face and Vertex
(0, 0, 0, 0, 1, 1) - MGB, Increase the Genus and Make a Body (shell)
(0, 1, 0,-1, 0, 0) - MEKH, Make and Edge and Kill a Hole

(8,12, 6, 0, 0, 1)
\[(7, 5, 1, 0, 0, 0)\]
Composition of Euler operators

- The order of operations is not defined
- Different orders are in the book
Abstraction

- The Eulerian operators are not directly used by the users
Abstraction

- The Eulerian operators are not directly used by the users
- Multiple operators are combined together to create a more complex operation
Abstraction

- The Eulerian operators are not directly used by the users
- Multiple operators are combined together to create a more complex operation
- These operations are the regular CAD functions
Abstraction

- The Eulerian operators are not directly used by the users
- Multiple operators are combined together to create a more complex operation
- These operations are the regular CAD functions, such as
  - Boolean operations
  - Sweeping
  - Chamfer
  - Blending
  - Drafting
  - Shelling
  - Tweaking
Geometry Kernel

- This seems like a lot to implement
- Luckily there are existing projects
- Geometry Kernel is the core of a CAD application
Geometry Kernel

- Different CADs use different kernels
- … which introduces problems when migrating projects
- Internals are quite different
OCCT

- Open Cascade Technology
- Uses BREP
- Is open source under GNU LGPL
- Is written in C++
- Supports all major platforms
  - Linux
  - Web
OCCT

Overview of functionality in their documentation.
The End
• https://blog.grabcad.com/blog/2013/05/14/kernels-why-cad-systems-dont-play-well-with-others/
• https://dev.opencascade.org/doc/overview/html/