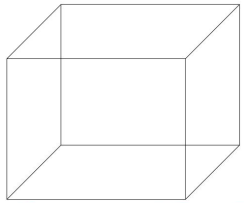
Unit 6: Solid Modeling

Prepared by Sumesh Gajmer

Solid Modeling



Wireframe Model

Solid Model

Solid is a rigid 3D object.

Solid modeling is the representation of solid object in our computer.

To make the solid object,we have to make the wireframe model of the object and convert it into 3D view.

To that 3D wired model, we add surface and it is converted into 3D view.

Solid modeling is a technique used in computer graphics to represent 3D objects with well-defined structures, allowing for operations like rendering, collision detection, and physical simulations.

Key Concepts in Solid Modeling

1. Boundary Point:

A boundary point of a solid object is a point that lies on the outer surface (boundary) of the object.

It separates the interior of the object from the exterior.

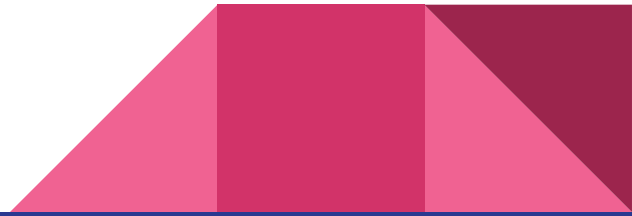
Example: The surface of a sphere contains all its boundary points.

2. Interior Point

An interior point of a solid object is a point that lies completely inside the object.

It has a small neighborhood around it that is also inside the object.

Example: Any point inside a cube but not on its faces is an interior point.

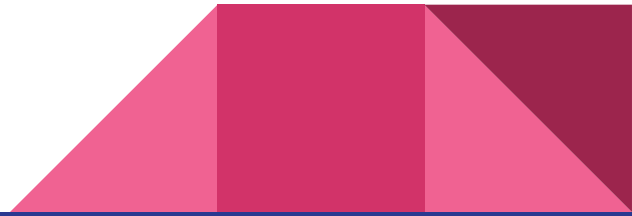


3. Closure

The closure of a solid object consists of all its interior points and boundary points.

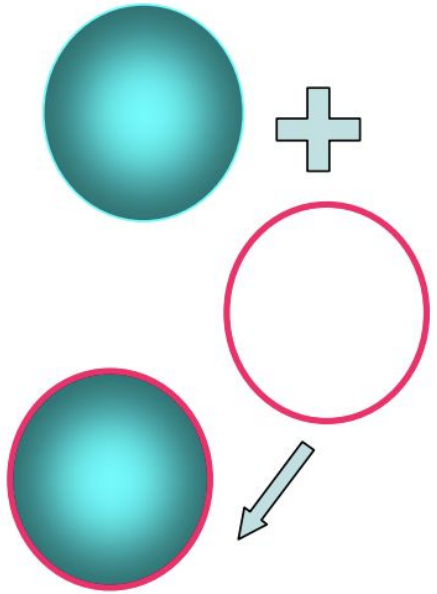
Mathematically, it is the smallest closed set containing the object.

Example: The closure of a sphere includes all points inside it and on its surface.



- Consider a unit sphere

- It's interior is $x^2 + y^2 + z^2 < 1$
- Its Boundary is $x^2 + y^2 + z^2 = 1$
- Solid sphere is
  Its closure $x^2 + y^2 + z^2 \leq 1$



Representations of Solid Objects

1. Sweep Representation:

Used to make 3D from 2D that have same kind of symmetry.

It involves moving 2D shape along a path in space.

There are two types of sweep representation.

a. Translational Sweep: In a translational sweep, a 2D shape is moved (translated) along a straight path to generate a solid object. The shape remains unchanged during translation.

Translational Sweep

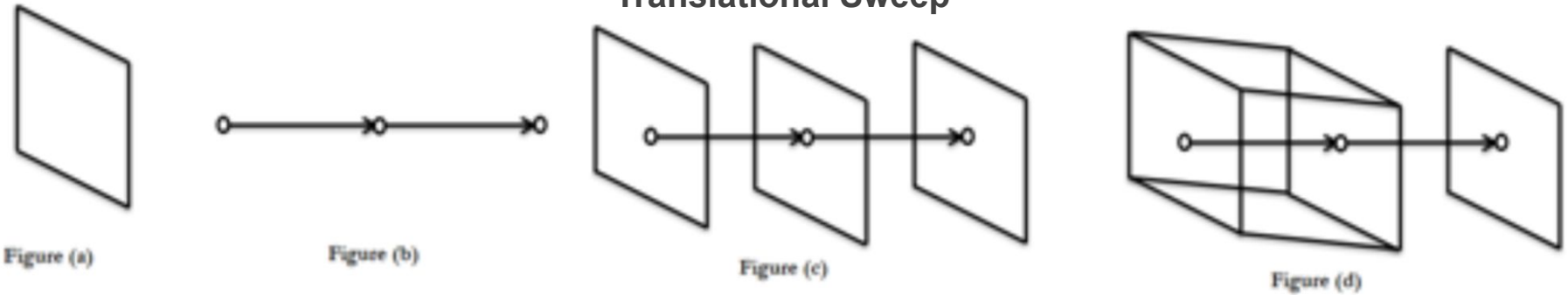
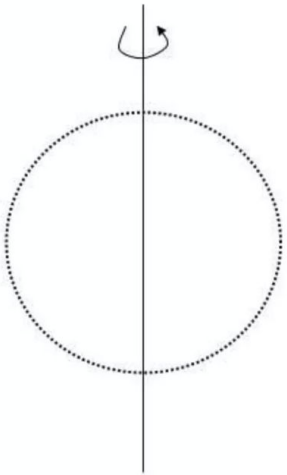


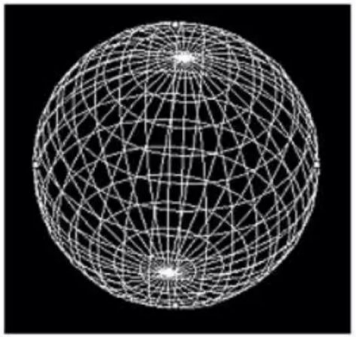
Figure (a) Figure (b) Figure (c) Figure (d)

b. Rotational sweep:

In a rotational sweep, a 2D shape is rotated around an axis to form a 3D solid.

The shape is rotated instead of moved linearly.





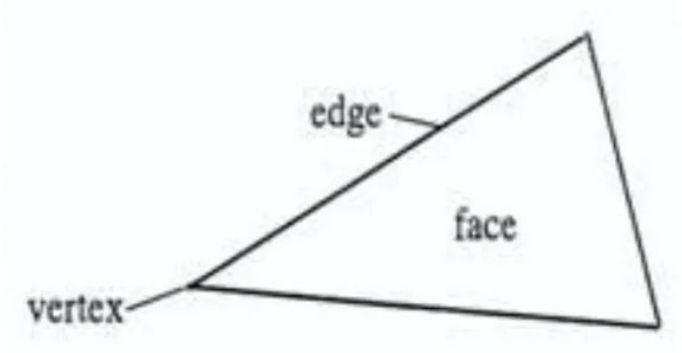
2. Boundary Representation (BRep)

Used for representing 3D object by its boundaries.

In this method, we define vertices, edges, faces and we construct solid object from that surface.

It is very appropriate to construct solid models of unusual shapes.

It requires large amount of storage.



3. Spatial Partitioning Representation.

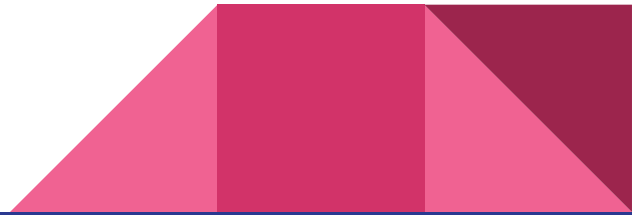
It involves dividing a space into smaller regions or cells to facilitate modeling.

It describes objects as a collection of adjoining non intersection solids.

Small solids may or may not be the same type as the original objects.

Spatial partitioning is a technique used in computer graphics, computational geometry, and game engines to efficiently manage and process 3D space.

It involves dividing space into smaller regions to optimize operations like rendering, collision detection, and ray tracing.



a. Octree Representation

Octree is 3D quad tree.

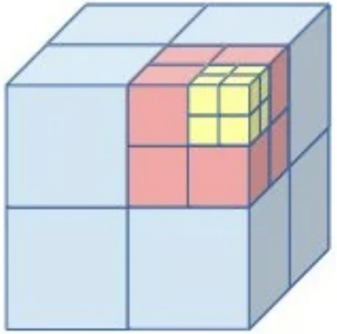
Divides 3D space into eight equal subregions.

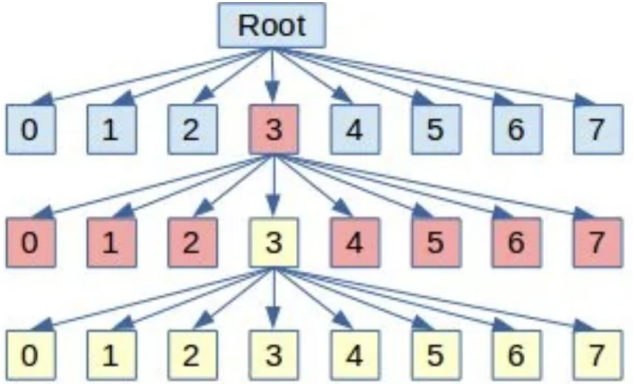
Octree recursively subdivides into octants.

Each level of the tree represents different levels of details.

Each node in the octree represents an octant of the space.

Used in collision detection algorithm.





BSP Tree

BSP is a method of dividing a 3D scene into two parts repeatedly until certain conditions are met.

A partitioning plane is chosen.

Objects in the scene are split into front and back regions.

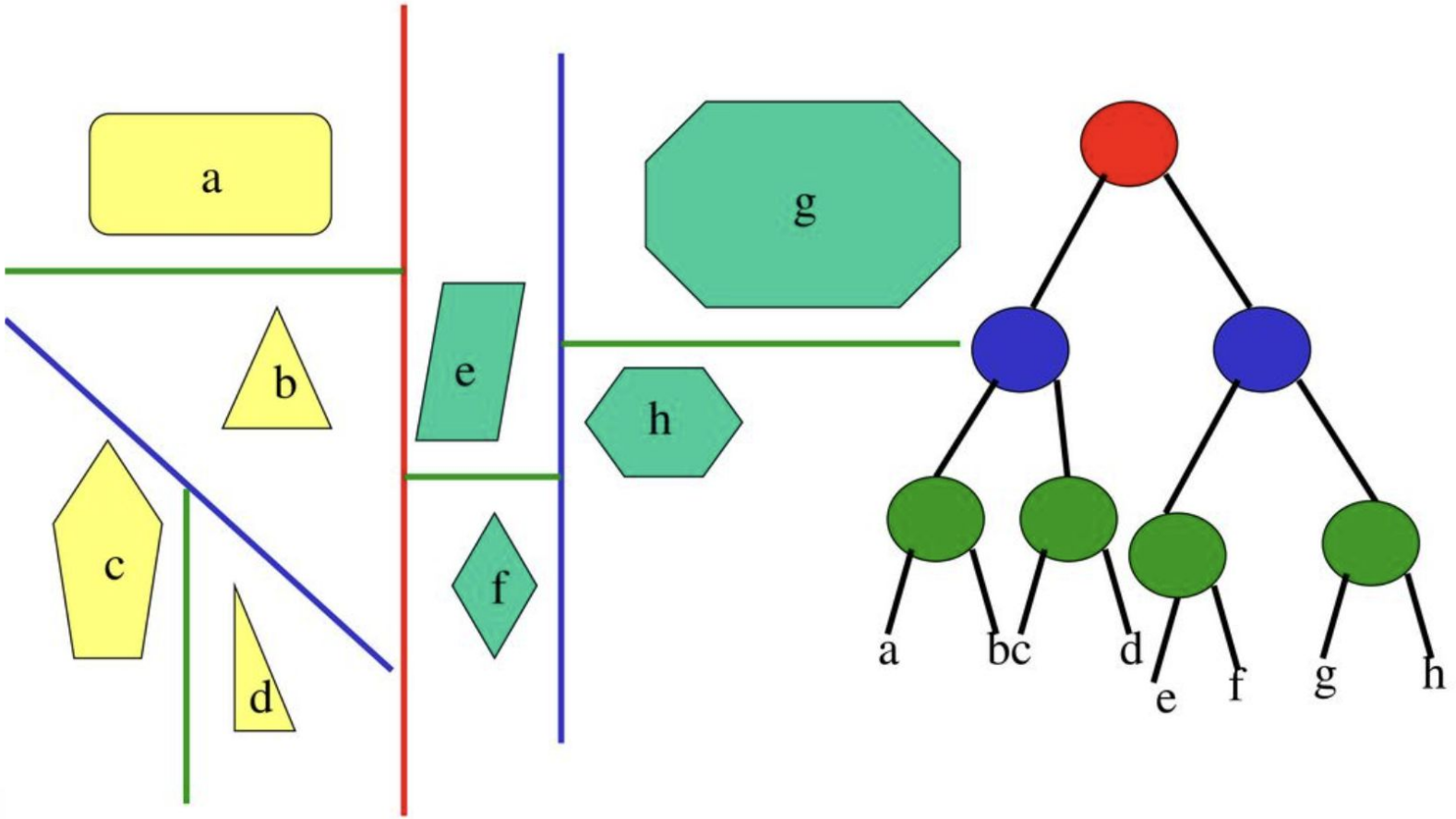
The process continues recursively until each part meets the required condition.

Efficiently organizes 3D space for rendering and collision detection.

Helps in visibility determination (deciding which objects to draw first).

Algorithm of generating a BSP Tree from a list of polygons

- Select a polygon P from the list.
- Make a node N in the BSP tree, and add P to the list of polygons at that node.
- For each other polygon in the list:
  - If that polygon is wholly in front of the plane containing P, move that polygon to the list of nodes in front of P.
  - If that polygon is wholly behind the plane containing P, move that polygon to the list of nodes behind P.
  - If that polygon is intersected by the plane containing P, split it into two polygons and move them to the respective lists of polygons behind and in front of P.
  - If that polygon lies in the plane containing P, add it to the list of polygons at node N.
- Apply this algorithm to the list of polygons in front of P.
- Apply this algorithm to the list of polygons behind P.


a
b
c
d
e
f
g
h
a
bc
d
e
f
g
h

Thank You