Data Structures for Modelling

Refs:
Jessica Crouch's ODU course

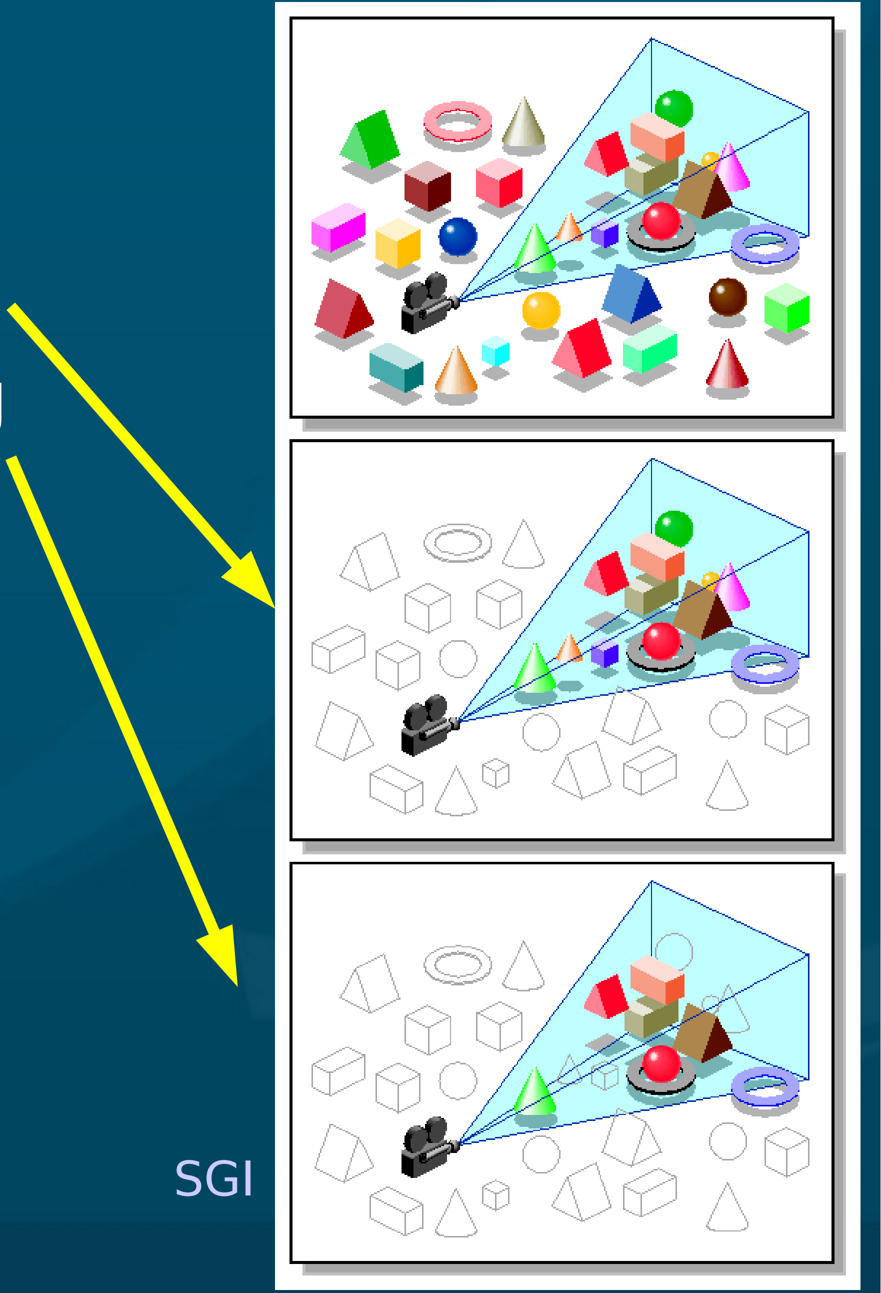
2 April 2009
CMPT370
Dr. Sean Ho
Trinity Western University

Spatial data structures

- Storing the geometry in a smarter way
- Space-subdivision:
  - Grids
  - Octrees
  - k-d trees and BSP trees
- Object-centred:
  - Bounding volumes
  - Scene graphs
- OpenSceneGraph

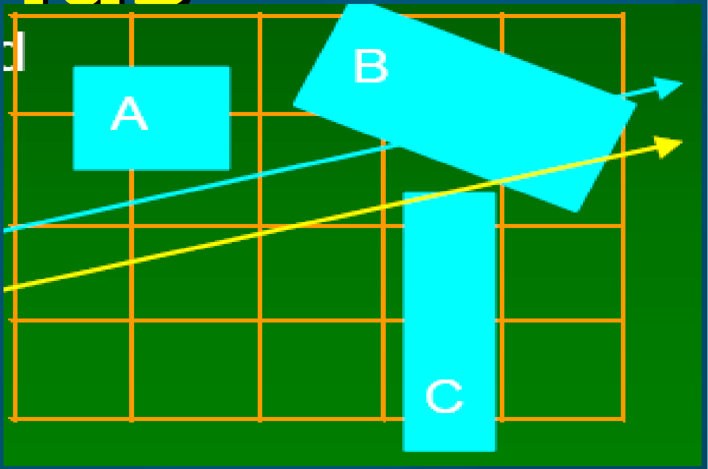
Why use spatial data structs?

- Geometry culling for speed
  - View frustum culling
  - Hidden-surface culling
  - Culling small details
- Collision detection
  - Robotics
  - Virtual world / gaming
  - Chemical / drug simulation
- Ray tracing
- Parallel rendering



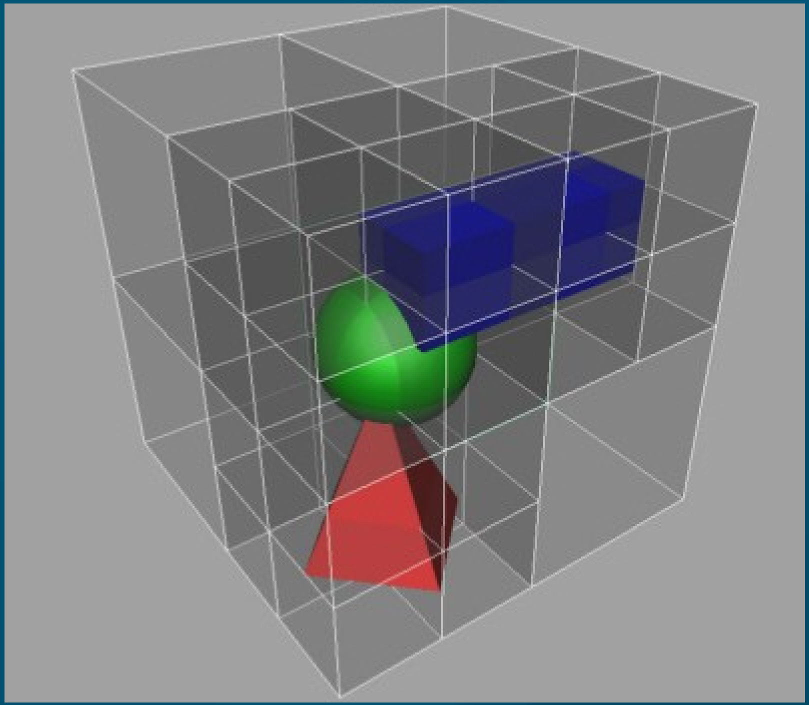
SGI

Spatial subdivision: grids

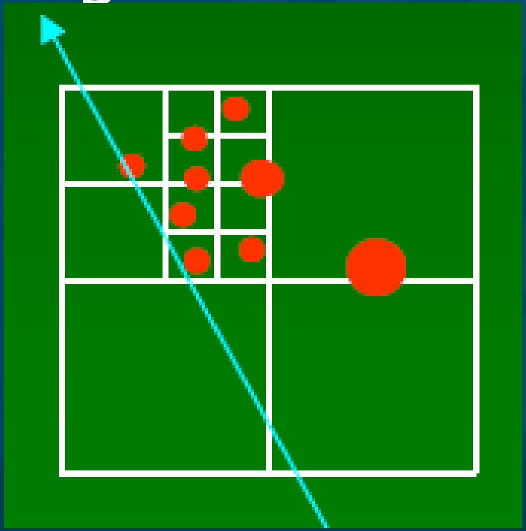


- Partition space (view frustum)
- Grids: 3D array of cells that tile the space: voxels
- Each voxel keeps list of all intersecting surfaces
- For each voxel intersected by the ray:
  - Test for intersection with each surface in voxel
- Best if objects are uniformly spread in space
  - Voxels too big => too many surfaces per cell
  - Voxels too small => wasted empty cells
- Try non-uniform cell spacing

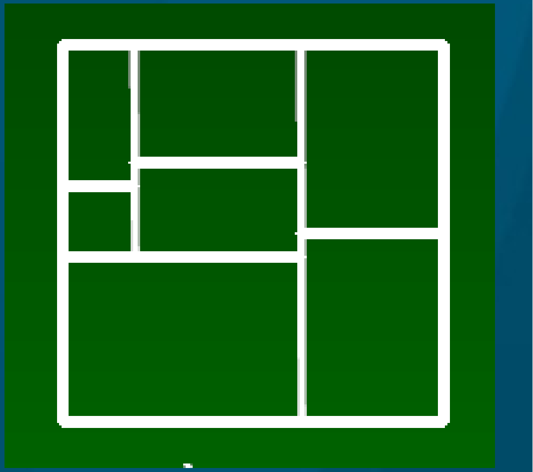
Octrees



- In 1D: binary tree
- In 2D: quadtree
- Each cell (node of tree) is a cube
- Recursively split into 8 equal sub-cubes
  - Adaptive subdivision: stop dividing based on number of surfaces in the cell
- Ray intersection: traverses tree
  - Tradeoff: tough to step to next cell along ray



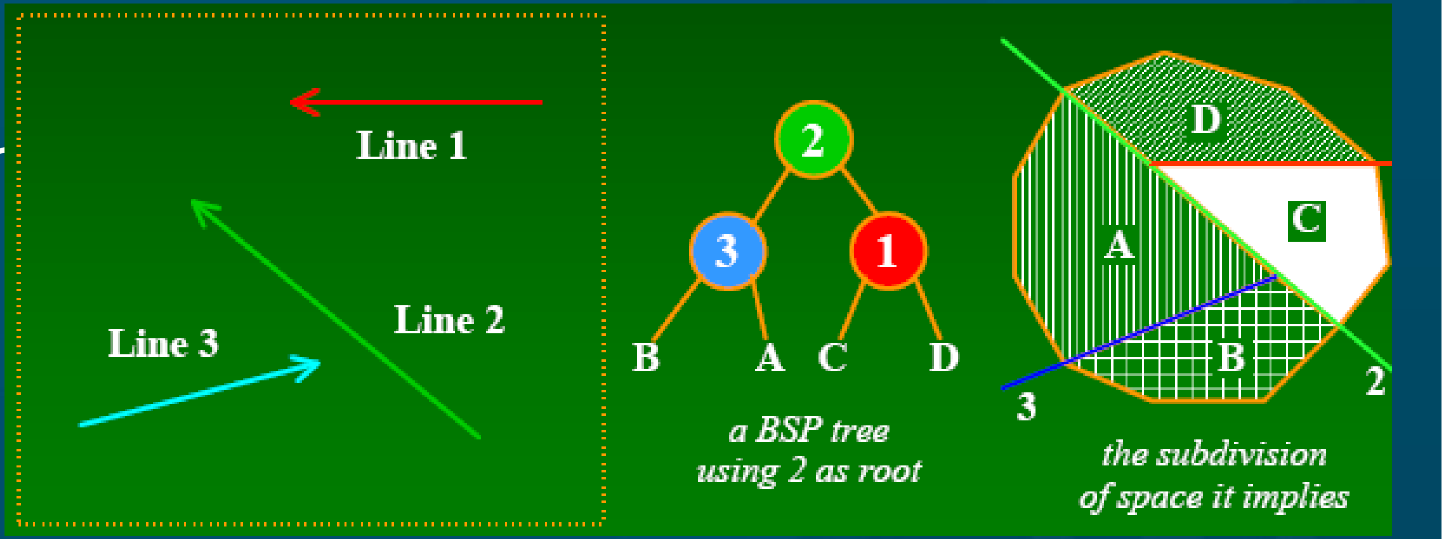
k-d trees and BSP trees



- Relaxing the rules on octrees
- k-d (k-dimensional) trees:
  - Split each cell one dimension at a time at arbitrary point within cell
- BSP (binary space partitioning) trees:
  - Split with plane of any orientation
    - In k-dims, split with hyperplane of dimension k-1
  - Used for hidden-surface removal
    - Painter's algorithm: planes oriented relative to camera

Building balanced trees

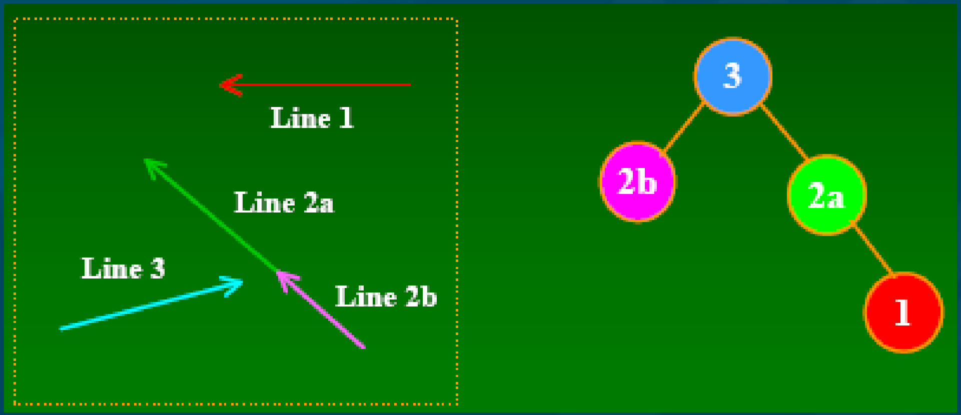
- Use the objects to guide choice of splitting plane
- Example with simple line segments



a BSP tree using 2 as root

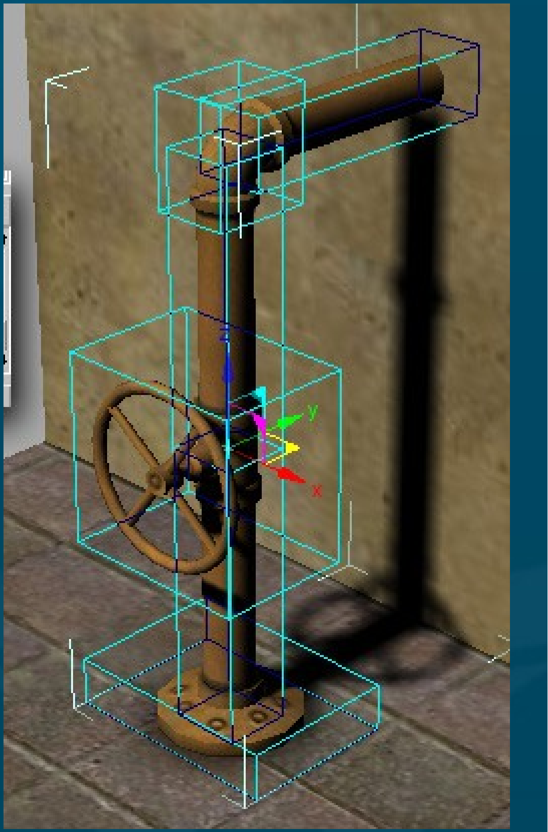
the subdivision of space it implies

- Using Line3 as root requires splitting Line2
- Splitting gives more surfaces but often a more balanced tree



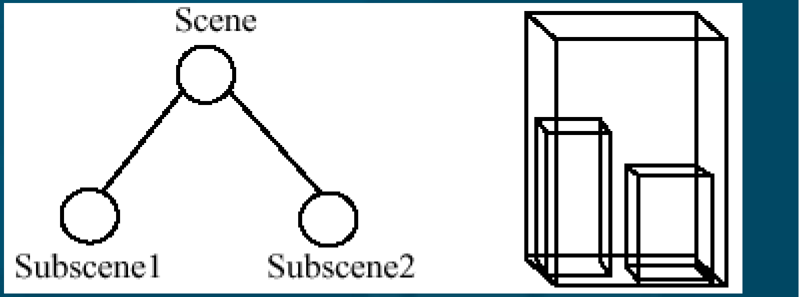
Object bounding volumes

- Object-centred data structure
- Wrap complex objects in simple ones
  - Level of detail
- If ray does not intersect bounding volume, it won't intersect the object
- Common types:
  - Axis-aligned boxes
  - Oriented boxes
  - Spheres
  - Convex hulls



Bounding volume hierarchies

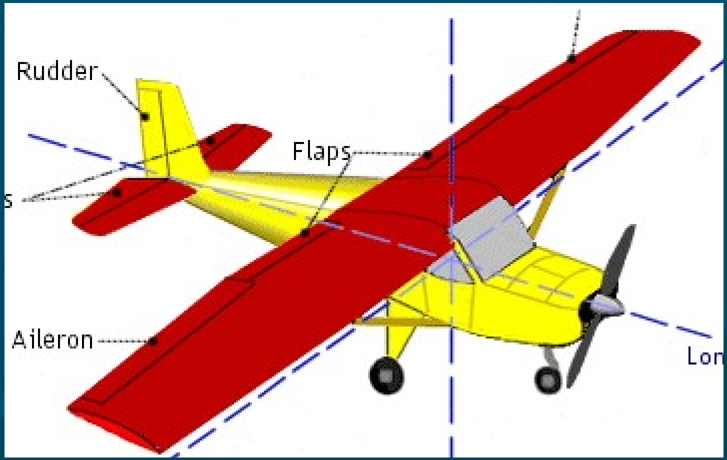
- Straightforward bounding volumes still store objects in a flat list: O(n) intersection tests
- Use a tree structure: boxes within boxes
- Recursively test for intersections:
  - If ray misses large box, don't need to descend tree
  - If ray hits large box, recurse into smaller boxes

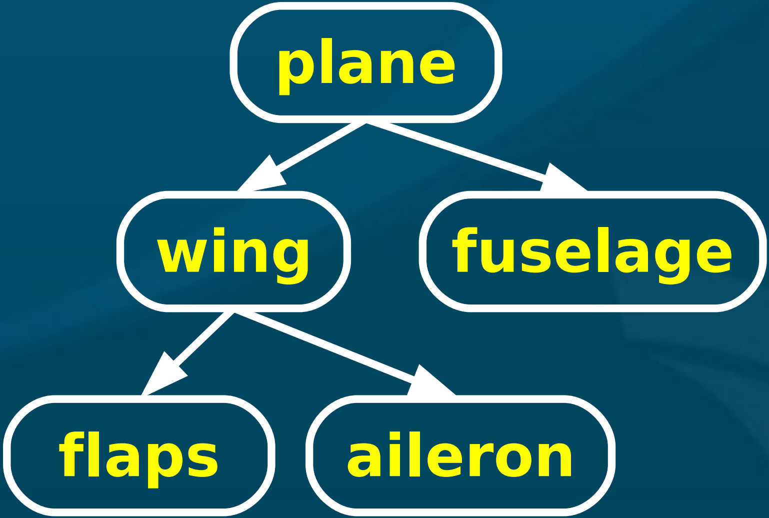


- Challenges:
  - Constructing full balanced tree

Scene graph

- Hierarchical grouping of objects
  - crowd → person → torso → arm → elbow
  - Directed acyclic graph; usually a tree
- Geometry is in leaf nodes
- Nodes may contain matrix transforms



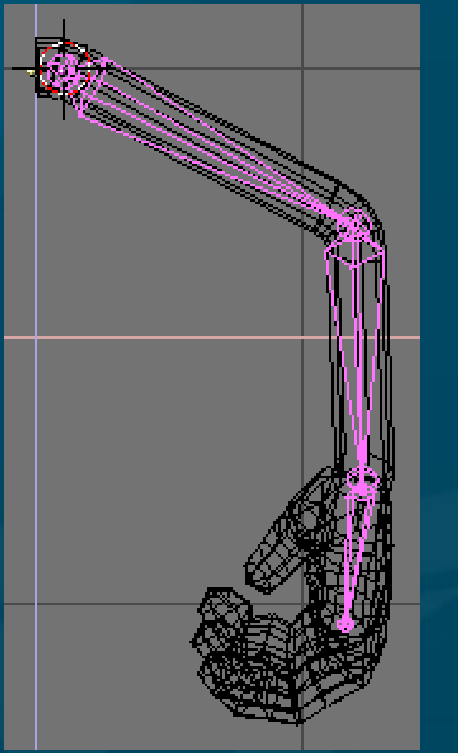

plane

wing

fuselage

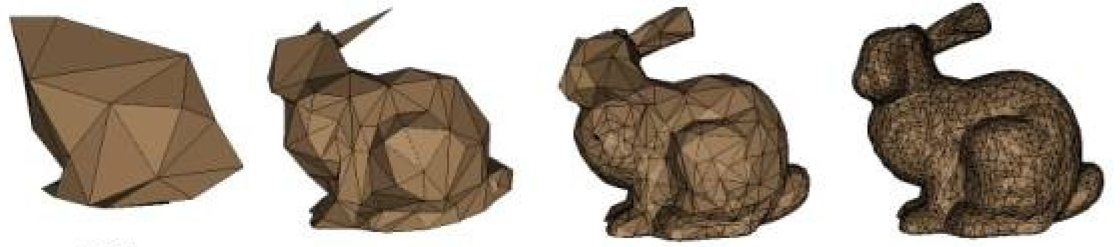
flaps

aileron



Controlling detail

- Consider projected size of geometry on screen
- Detail culling:
  - Don't render tiny triangles
- Levels of detail (LoD):
  - Like mip-maps but for geometry
  - Generate several versions of geometry
  - Choose LoD based on projected size



(a) 72 faces (d) 297 faces (c) 1474 faces (d) Original model : 6123 faces

<Figure 1> : Each leve of detail represents high quality approximation of original model

CVC @U Texas

OpenSceneGraph library

- www.openscenegraph.org

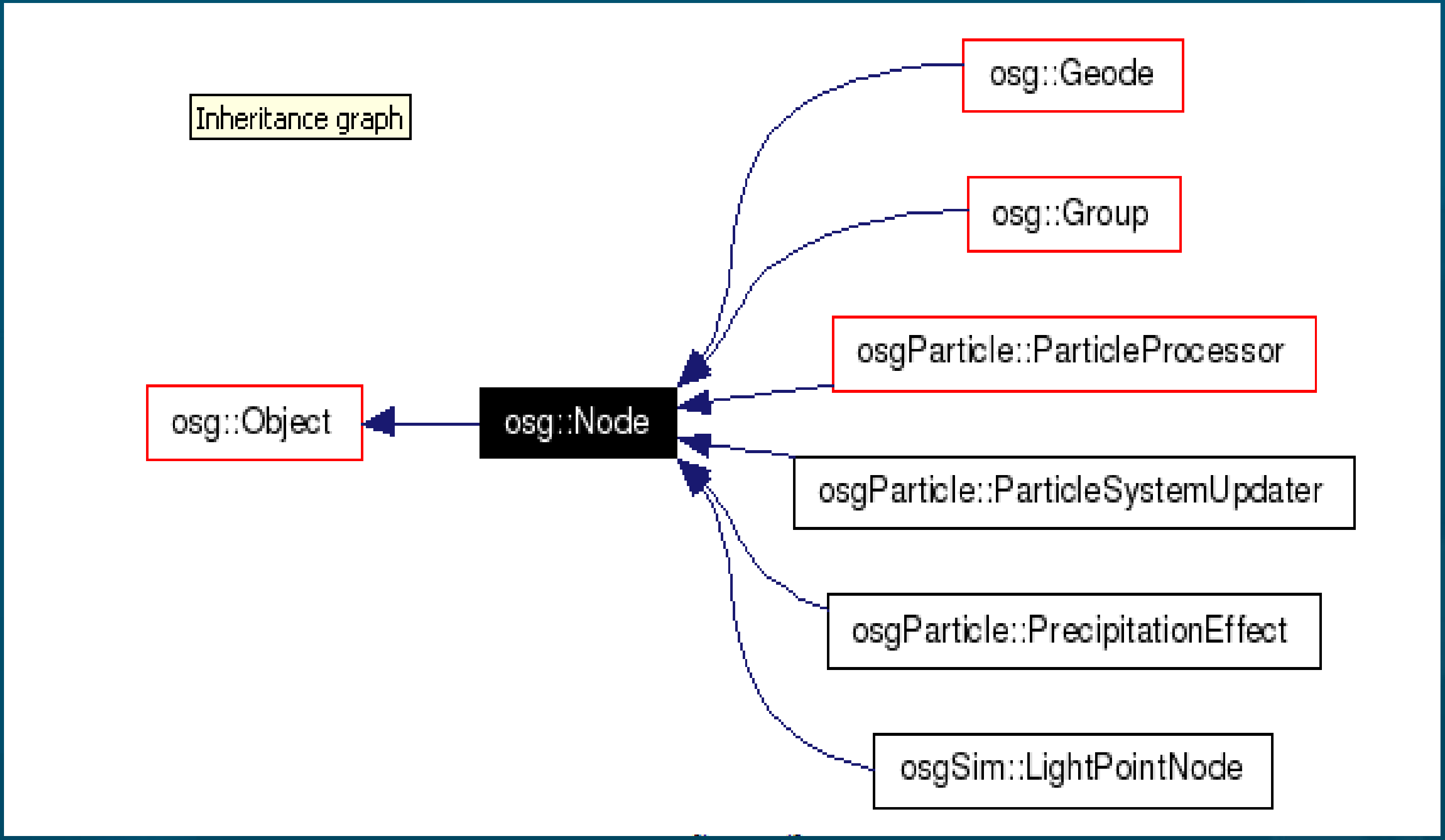
- Layer above OpenGL
- Heavy use of OO, templates, design patterns
  - C++, plus bindings in Java, Python, etc.
- Still developing, but growing popularity in graphics community
- Supports: view frustum cull, occlusion cull, detail cull, LoD, much more
- Alternatives: OpenSG (parallelized), VTK (visualization, 2D/3D image processing), OpenInventor (old), OpenRM, Fahrenheit (old)

The OSG distribution

- Core OSG: main libraries, node types
- NodeKits: additional node classes
- Plugins: for reading/writing various file types
  - 3D geometry: 3DS, AC3D, Alias LightWave, OpenFlight, TerraPage, VRML, etc.
  - Images: jpg, gif, png, tiff, etc.
- Interoperability libraries: for interfacing with other libraries, languages
- Examples

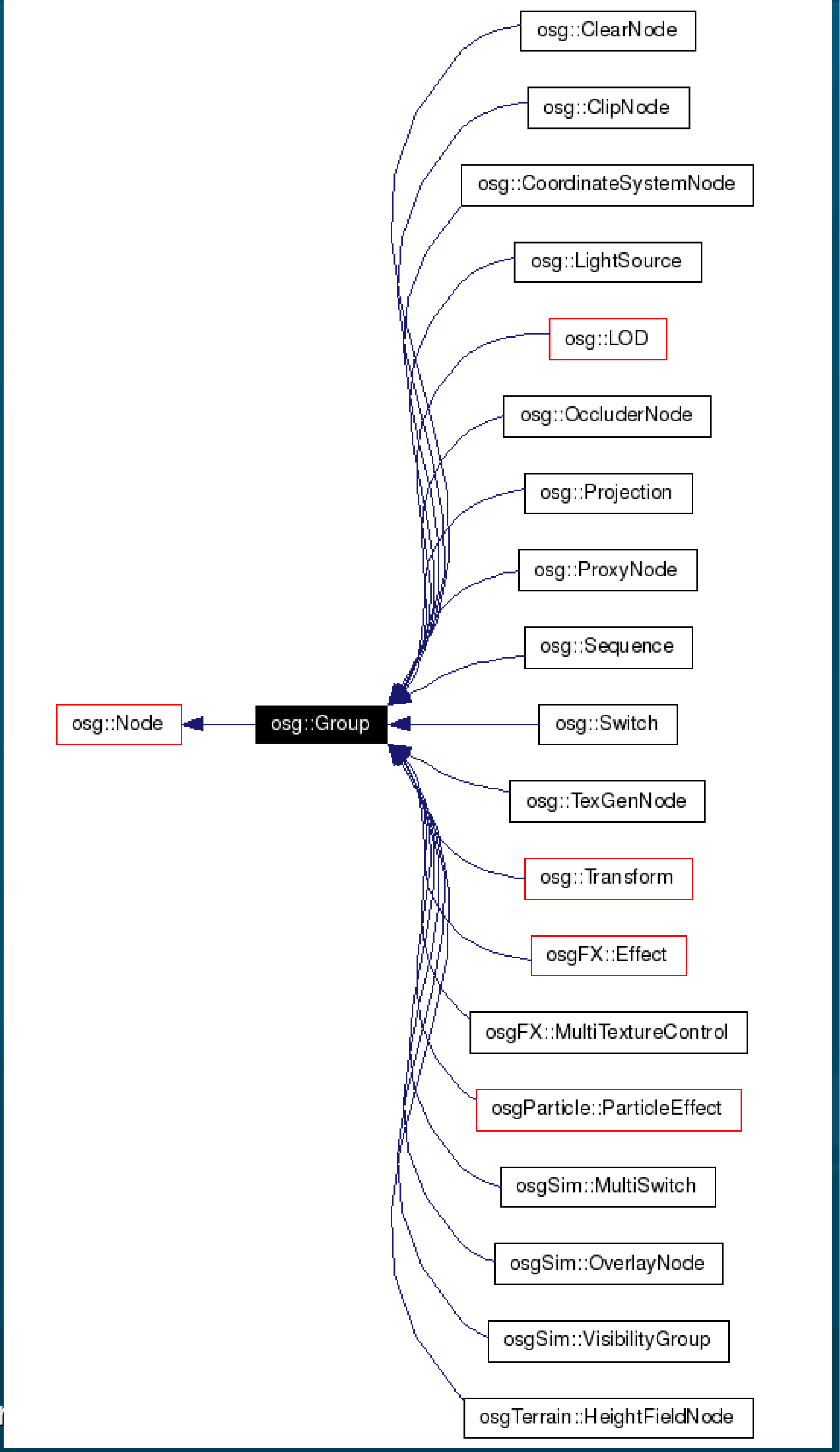
OSG class hierarchy: Node

- Everything subclasses from OSG::Node


Inheritance graph
osg::Object
osg::Node
osg::Geode
osg::Group
osgParticle::ParticleProcessor
osgParticle::ParticleSystemUpdater
osgParticle::PrecipitationEffect
osgSim::LightPointNode

OSG::Group

- Generic node that may contain children



osg::Node

osg::Group

osg::ClearNode

osg::ClipNode

osg::CoordinateSystemNode

osg::LightSource

osg::LOD

osg::OccluderNode

osg::Projection

osg::ProxyNode

osg::Sequence

osg::Switch

osg::TexGenNode

osg::Transform

osgFX::Effect

osgFX::MultiTextureControl

osgParticle::ParticleEffect

osgSim::MultiSwitch

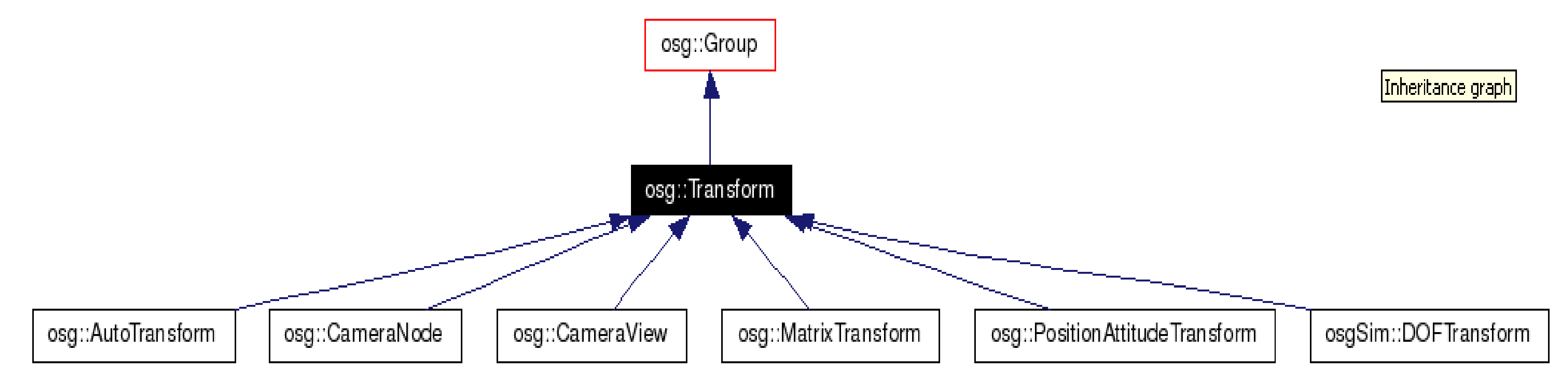
osgSim::OverlayNode

osgSim::VisibilityGroup

osgTerrain::HeightFieldNode

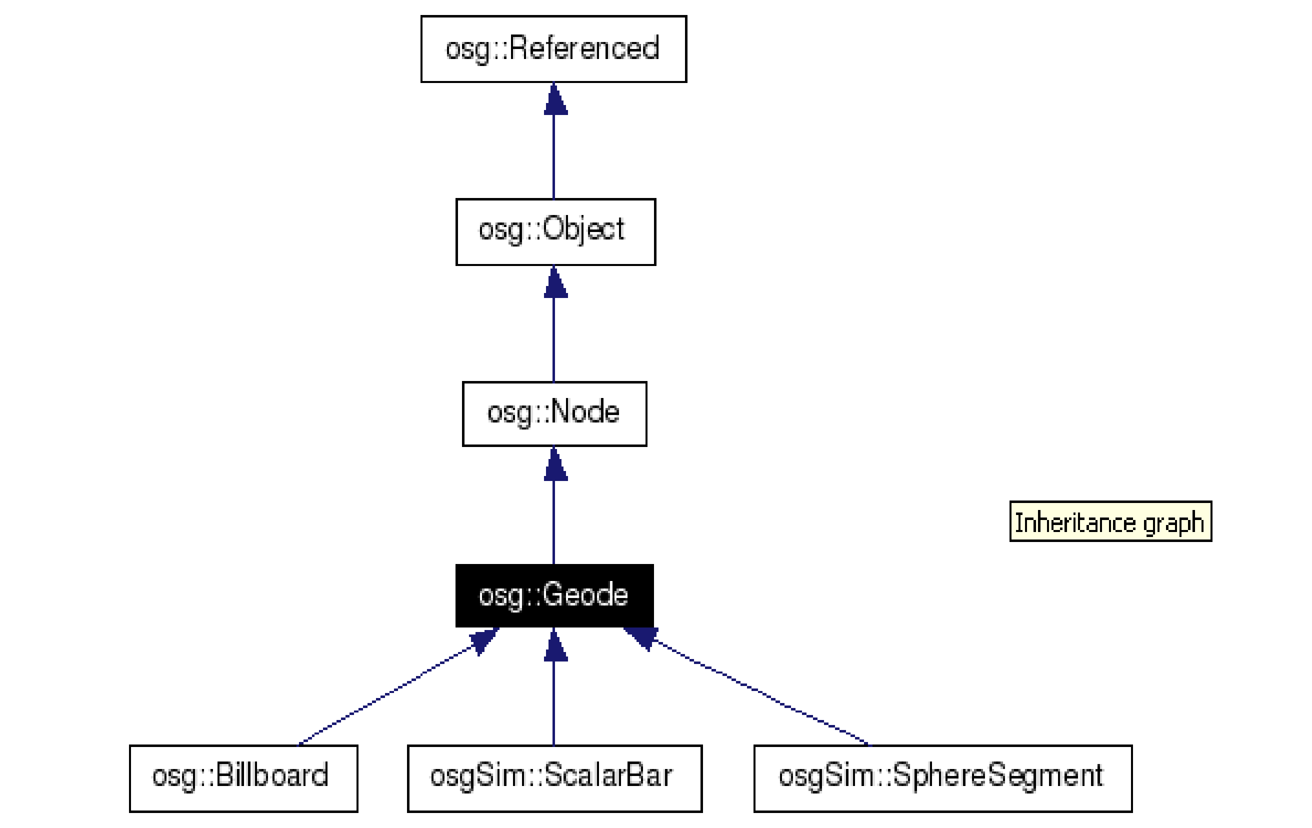
OSG::Transform

- Transforms all children by a 4x4 matrix
  - Position objects
  - Move camera / trackball
  - Animation

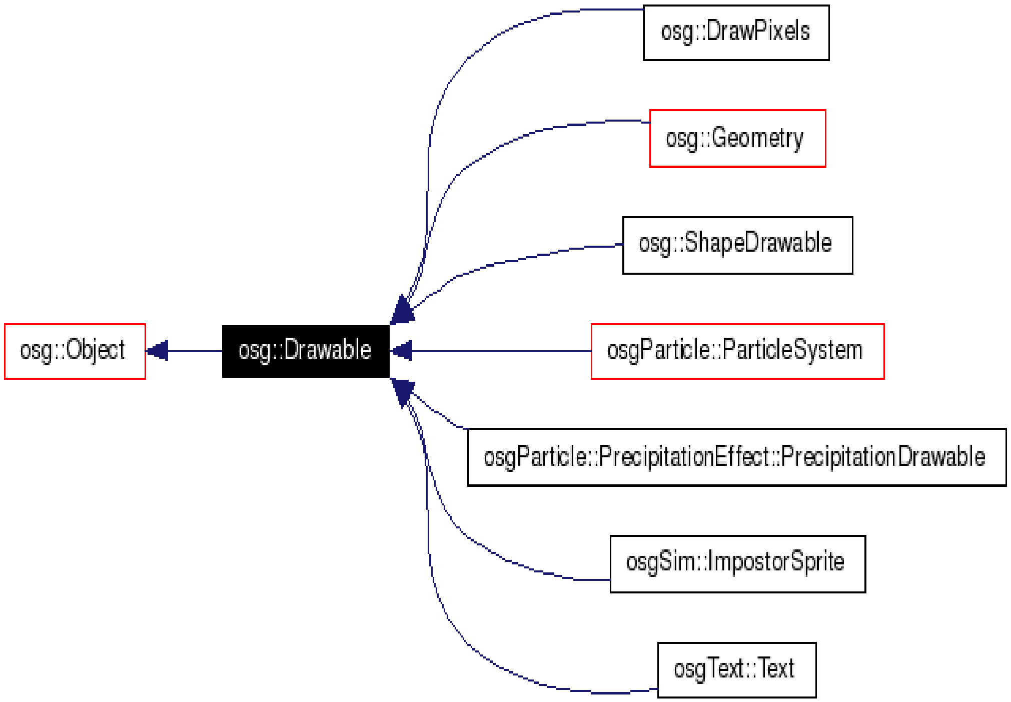


OSG::Geode

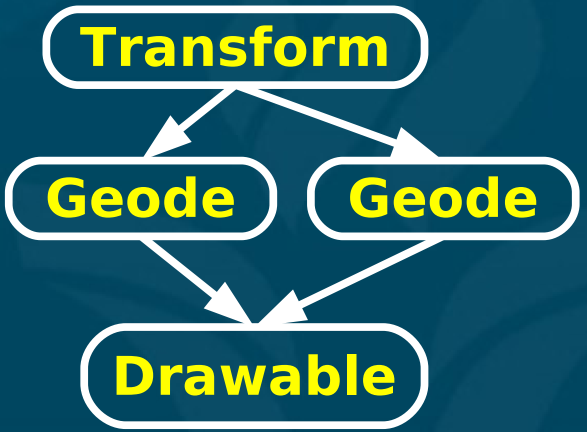
- Contains the actual geometry: leaf node
- Groups together OSG::Drawable objects



OSG::Drawable

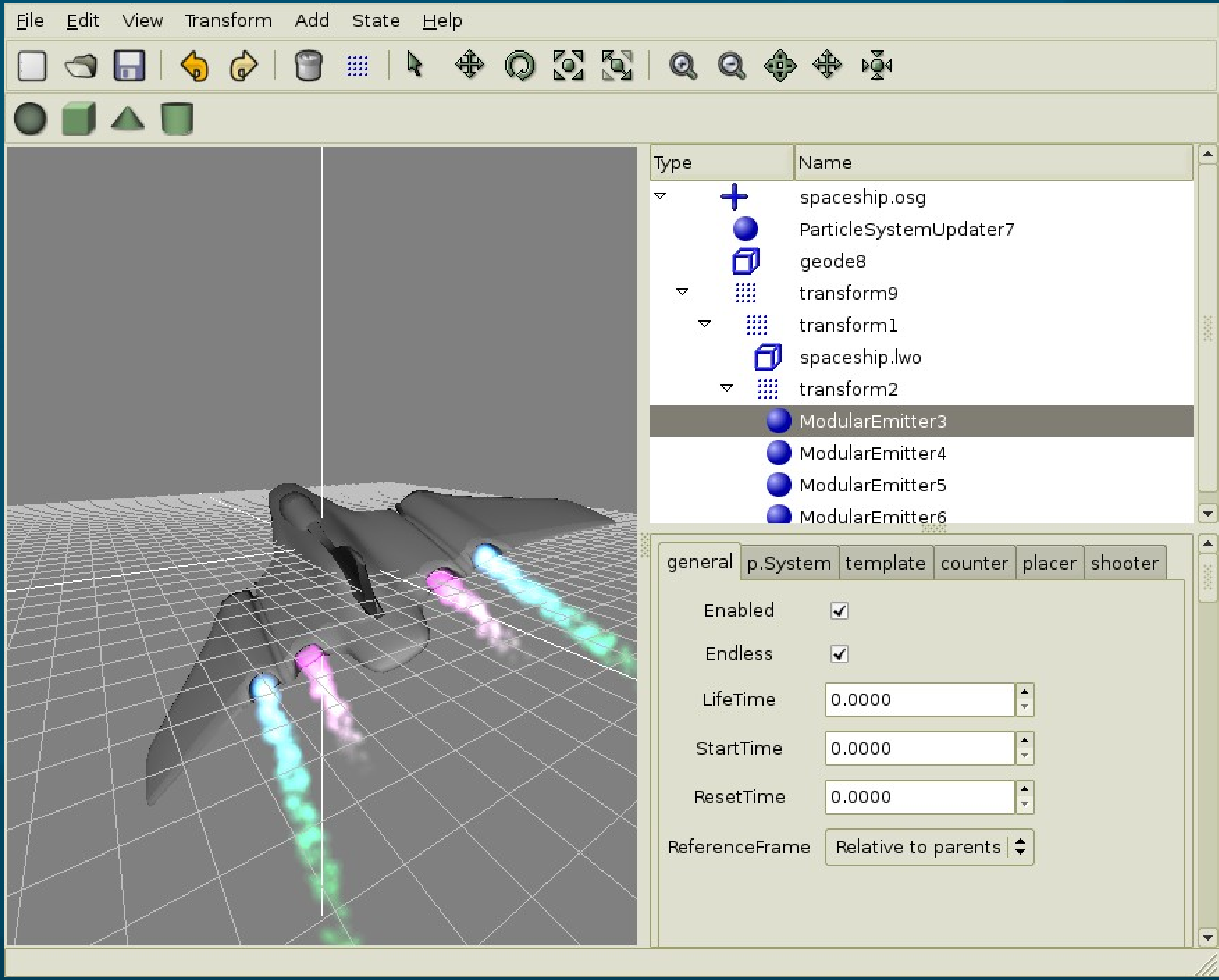


- Anything that is renderable
- Not a Node, but attached to a Geode
- OSG::Drawable is abstract (pure virtual) supercl
- Drawables may be shared with several Geodes
  - Same geometry used several ways
  - Scene graph not a tree, but directed acyclic graph



OSGEdit

- Import models from other programs
- Arrange and organize complex scenes
- Use hierarchical transforms
- osgedit.sf.net



spaceship.osg
ParticleSystemUpdater7
geode8
transform9
transform1
spaceship.lwo
transform2
ModularEmitter3
ModularEmitter4
ModularEmitter5
ModularEmitter6

Enabled ☑
Endless ☑
LifeTime 0.0000
StartTime 0.0000
ResetTime 0.0000
ReferenceFrame Relative to parents