CMT107 Visual Computing

III.2 Scene Representation

Xianfang Sun

School of Computer Science & Informatics
Cardiff University

Overview

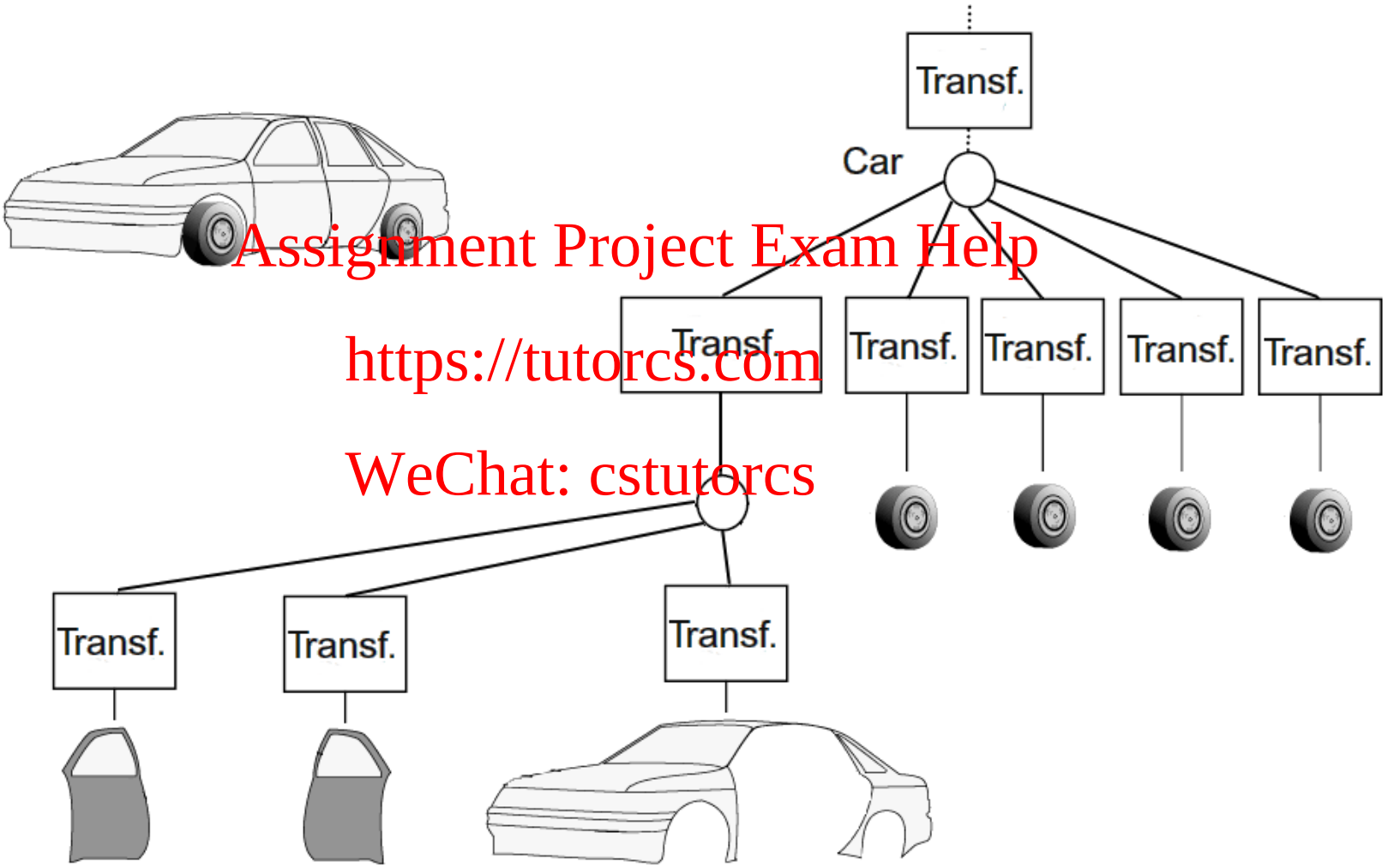
- Hierarchical modelling
  - Scene graphs
  - Constructing scene graphs
- Spatial data structures
  - Uniform grids
  - Octrees
  - kD-trees
  - BSP-trees
- Multi-resolution models

Hierarchical Modelling

- A scene is the complete description of the environment
  - A view is a particular part of the scene visible from the camera position
  - A scene consists of many models, only some are visible
- A scene can be represented by a hierarchical structure
  - A node represents some part of the scene
  - Top node is the whole scene
  - Leaf nodes are the actual geometric models
- Objects specified in *local coordinates*
  - Add transformation to hierarchy to specify location in scene

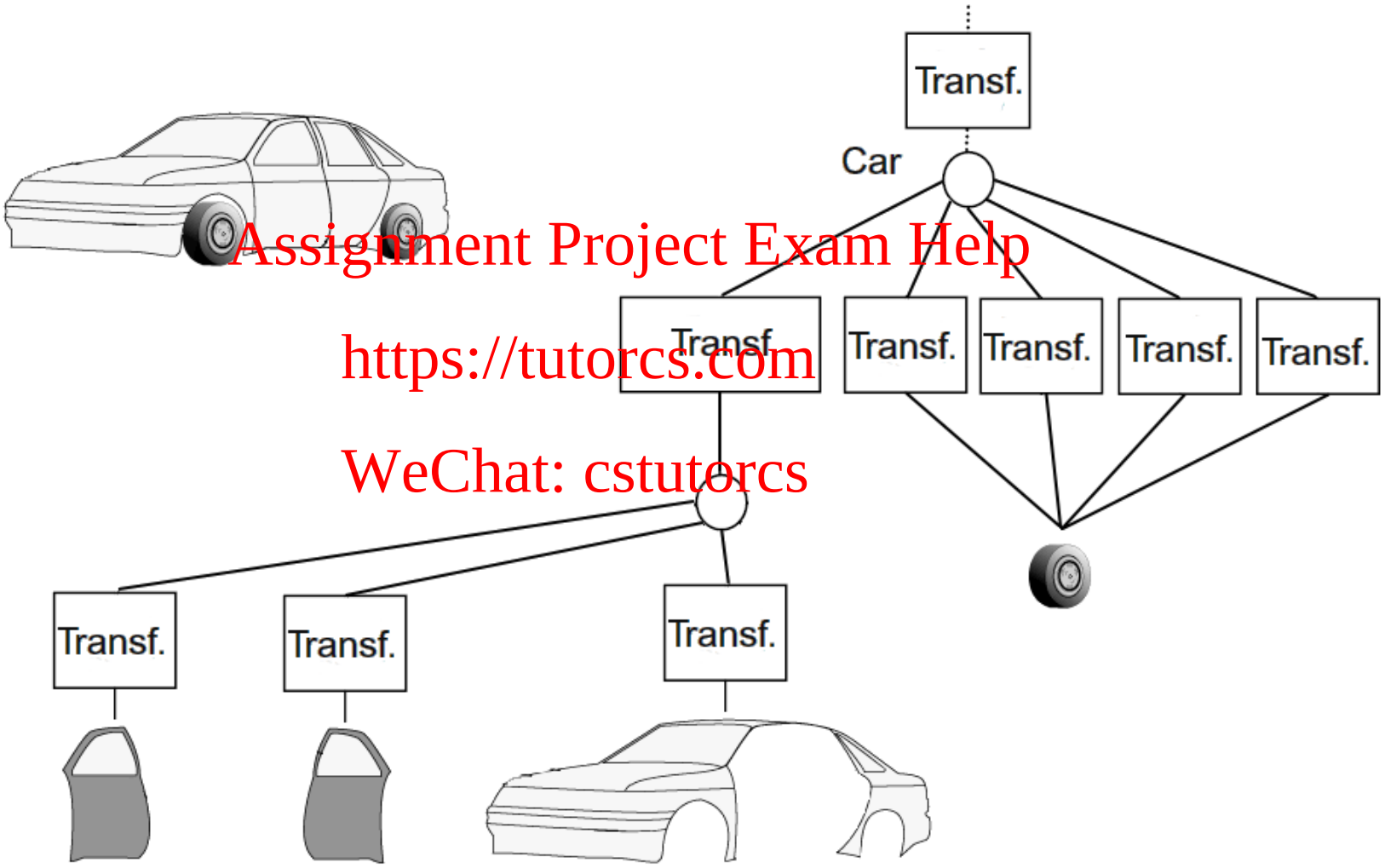
Scene Tree Example

- Scene tree for a simple car



Scene Graph Example

- Scene graph: combine congruent objects



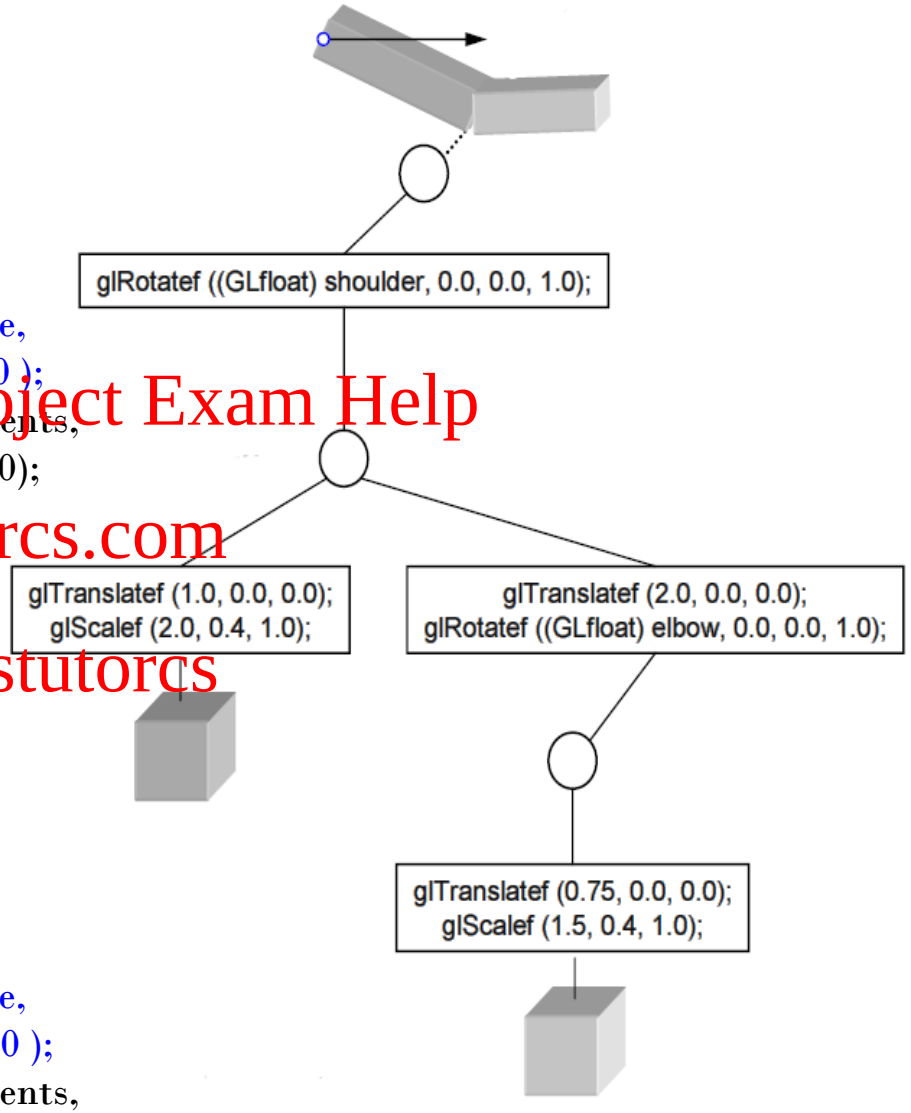
Scene Graphs

- Scene Graphs are in general *acyclic directed graphs*
  - Explicitly represented by graph data structure
  - Or implicitly by sequence of instructions / function calls
- Attributes and inheritance
  - Graph may contain material, transformation, . . . nodes representing object attributes
  - Attributes are usually inherited by all sub-nodes
- Also suitable for animations:
  - Make transformations dependent on parameter, e.g. time, motion control parameters, . . .

Robot Arm—OpenGL Implementation

```
T.initialize();
T.scale(0.5f, 0.5f, 0.5f);
T.scale(2f,0.4f,1f);
T.translate(1,0,0);
T.rotateA(-50f, -0.2f, 0f, 1f);
gl.glUniformMatrix4fv( ModelView, 1, true,
                       T.getTransformv(), 0 );
gl.glUniformMatrix4fv( NormalTransform, 1, true,
                       T.getInvTransformTv(), 0 );
gl.glDrawElements(GL_TRIANGLES, numElements,
                  GL_UNSIGNED_INT, 0);

T.initialize();
T.scale(0.5f, 0.5f, 0.5f);
T.scale(1.5f,0.4f,1);
T.translate(0.75f,0,0);
T.rotateZ(50);
T.translate(2.00f, 0.0f, 0);
T.rotateA(-50f, -0.2f, 0f, 1f);
gl.glUniformMatrix4fv( ModelView, 1, true,
                       T.getTransformv(), 0 );
gl.glUniformMatrix4fv( NormalTransform, 1, true,
                       T.getInvTransformTv(), 0 );
gl.glDrawElements(GL_TRIANGLES, numElements,
                  GL_UNSIGNED_INT, 0);
```



Hierarchy Construction

- Problem: find *optimal hierarchy* for scene graph
  - Choose bounding volumes
    spheres, boxes, convex hulls, . . .
  - Construct hierarchy of objects based on some heuristic
    (depends on application)
- Consider solutions for special cases
  - Spatial closeness of models
  - Standard spatial data structures

Spatial Data Structures

- Represent *spatial relations* in scene graph
  - Which models are visible from a camera position?
  - Which models can be accessed from a certain position?
  - With which models did a model collide?
  - . . .
- The more information about the spatial relations between models is known, the faster the scene can be processed
  - *Partition* space and place objects within subregions
  - Create *hierarchy* of spatially close models
  - Helps algorithms to determine relevant models quickly

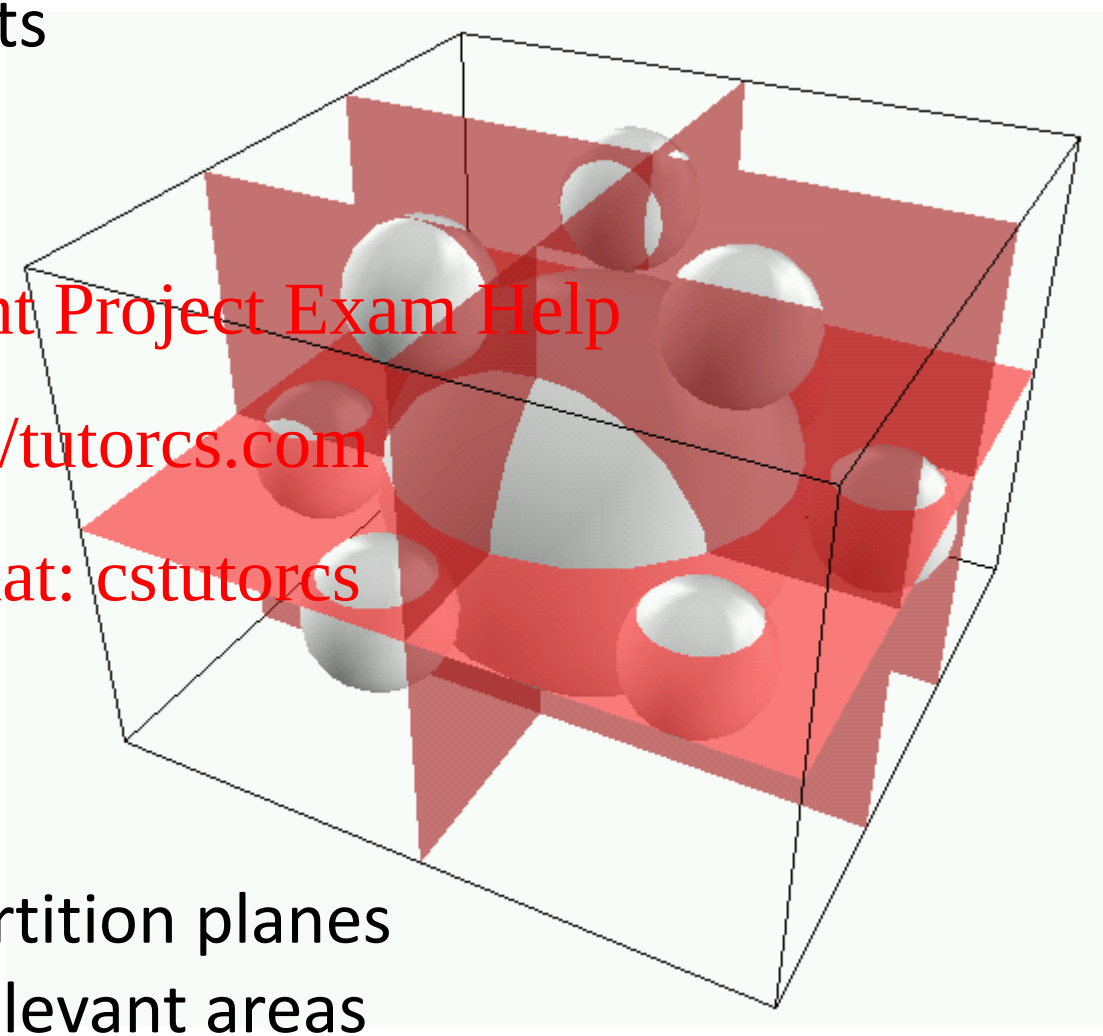
Example 3D Scene

- 3D scene example

  (ray-tracing)

Uniform Grids

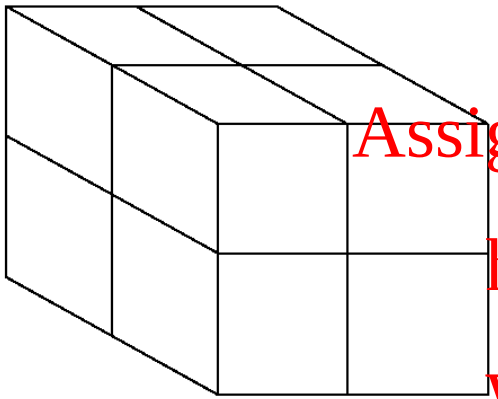
- Partition space *uniformly* using a 3D grid
  - 3D array of model lists

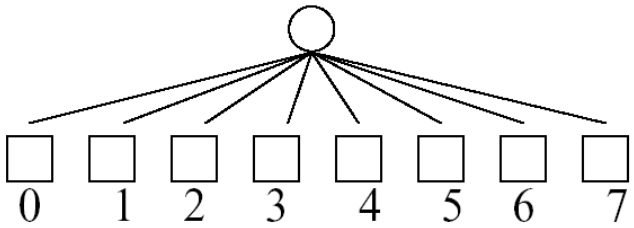


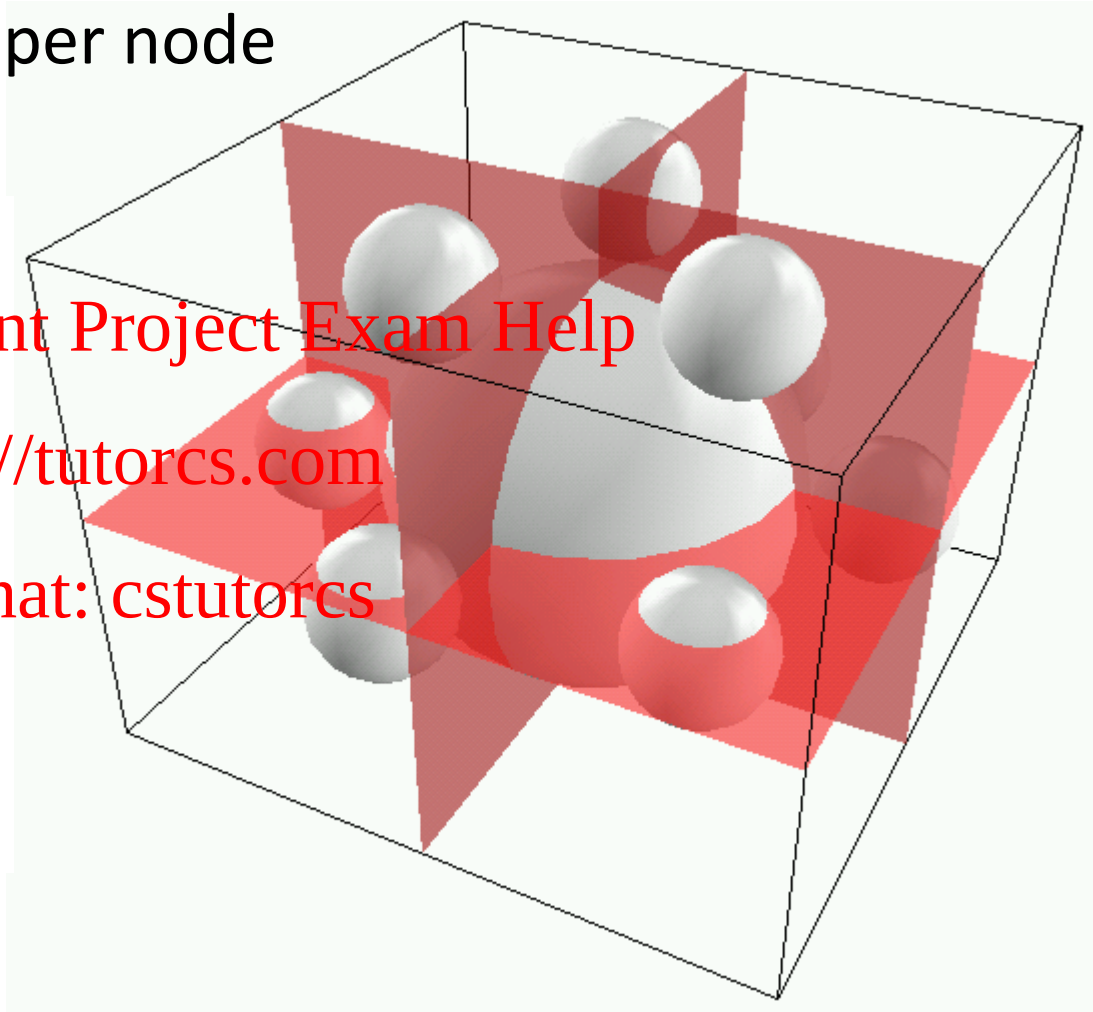
  - Cut models along partition planes
  - Or add them to all relevant areas

Octrees

- Partition space using a 3D *hierarchical grid*
  - Tree with 8 children per node

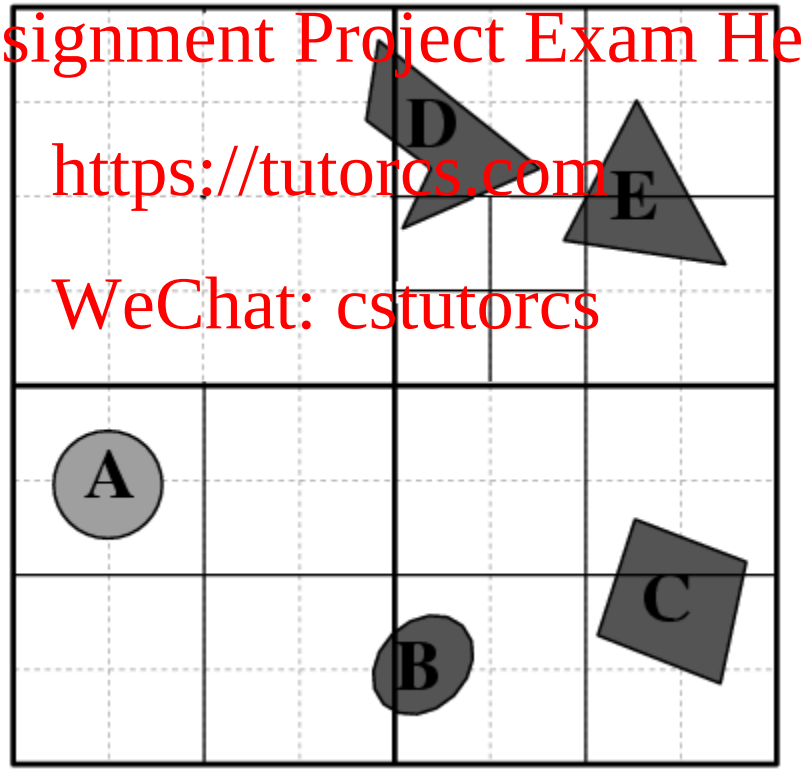






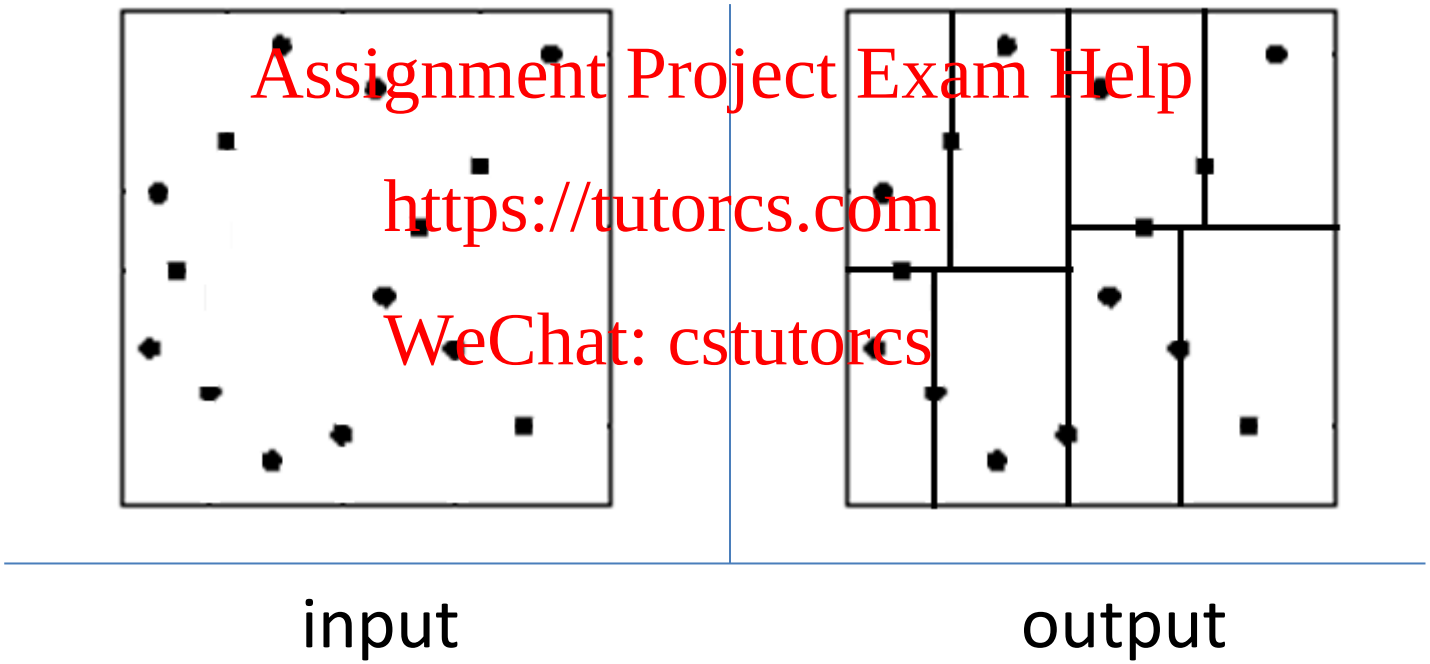
Octrees for Scene Graph Hierarchy

- Octree construction (Quadtree in 2D)
  - Generate octree for models until no cell contains more than one model
  - Group models/nodes in the same cell at the same level



kD-Trees

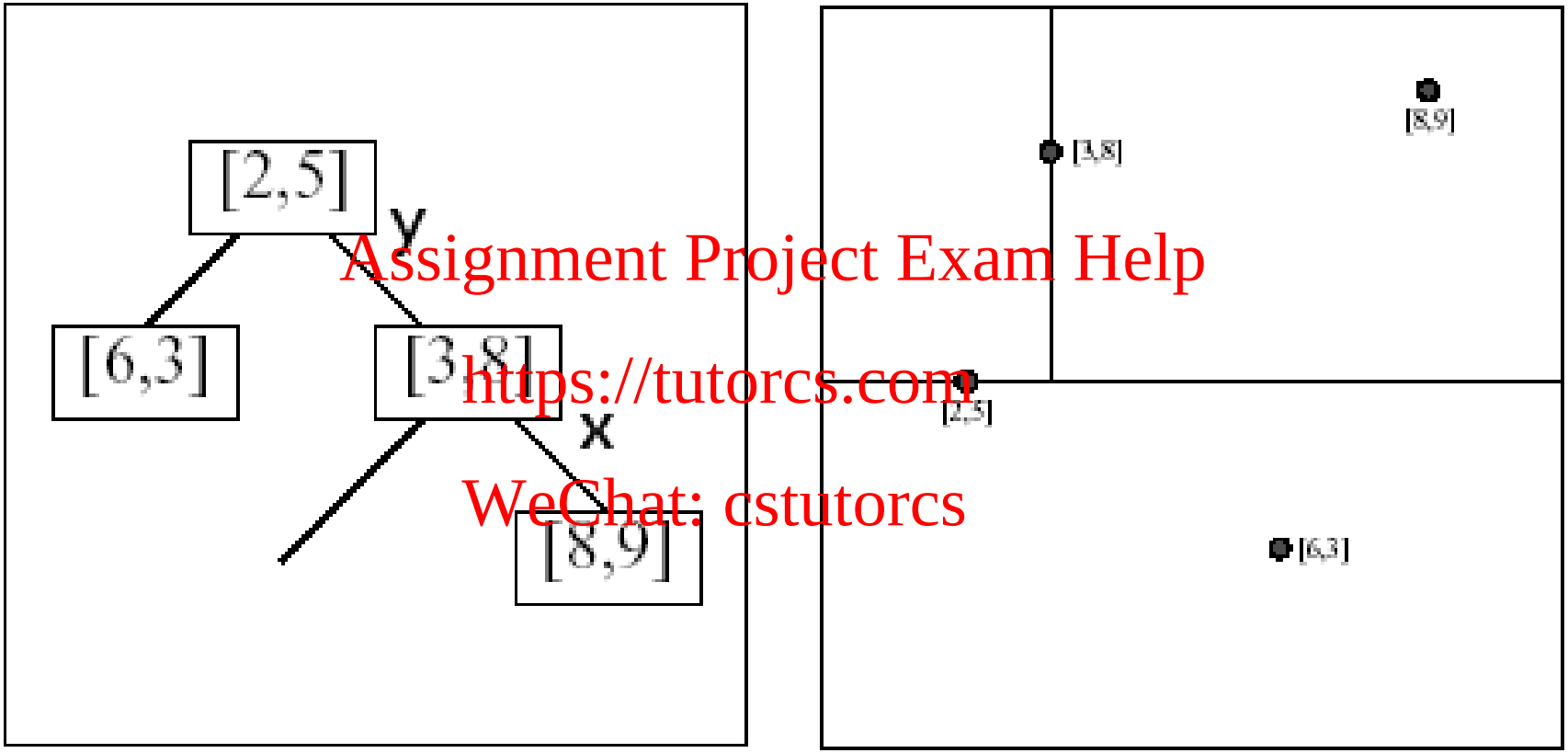
- Input: n points in k dimensions
- Output: tree that *partitions space at axis-aligned planes*
  - Each point is contained in its own box-shaped region



kD-Trees

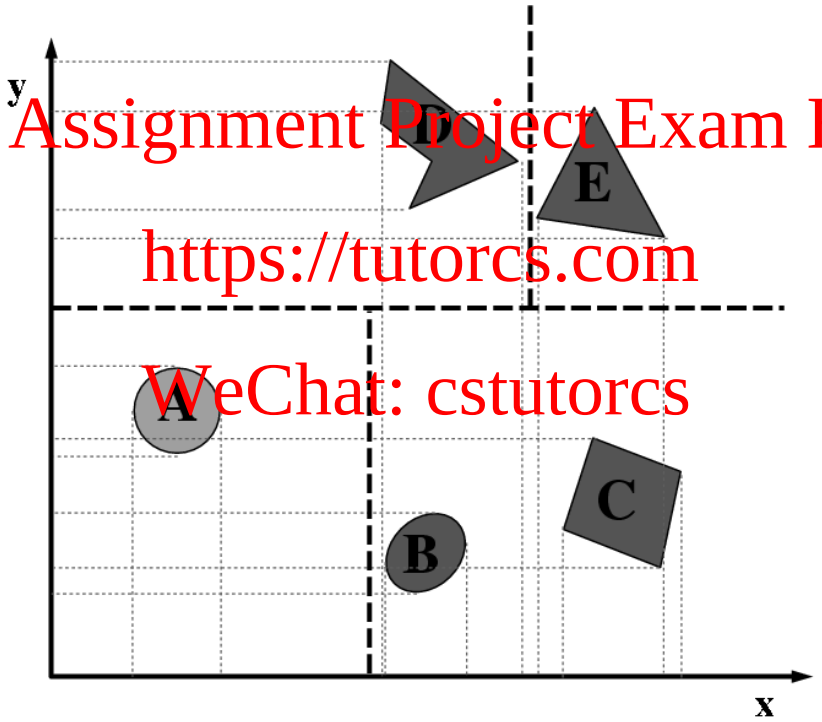
- Generalisation of *binary search trees*
  - At each node find a point which *separates* remaining points into two (approximately) equal sized sets
- In k dimensions, repeat per level:
  - *Choose* one dimension
  - *Sort* points in 1D
  - *Split* points at median
- Choice of dimension:
  - Regular, e.g. x, y, z, x, y, . . .
  - Dimension where distance between points is maximal
  - Some other clever strategy. . .

kD-Trees



kD-Tree Generalisation

- kD-Trees can be generalised to handle models
  - Median cut in x, then y, . . .
  - Search for best gaps for a small set of plane orientations

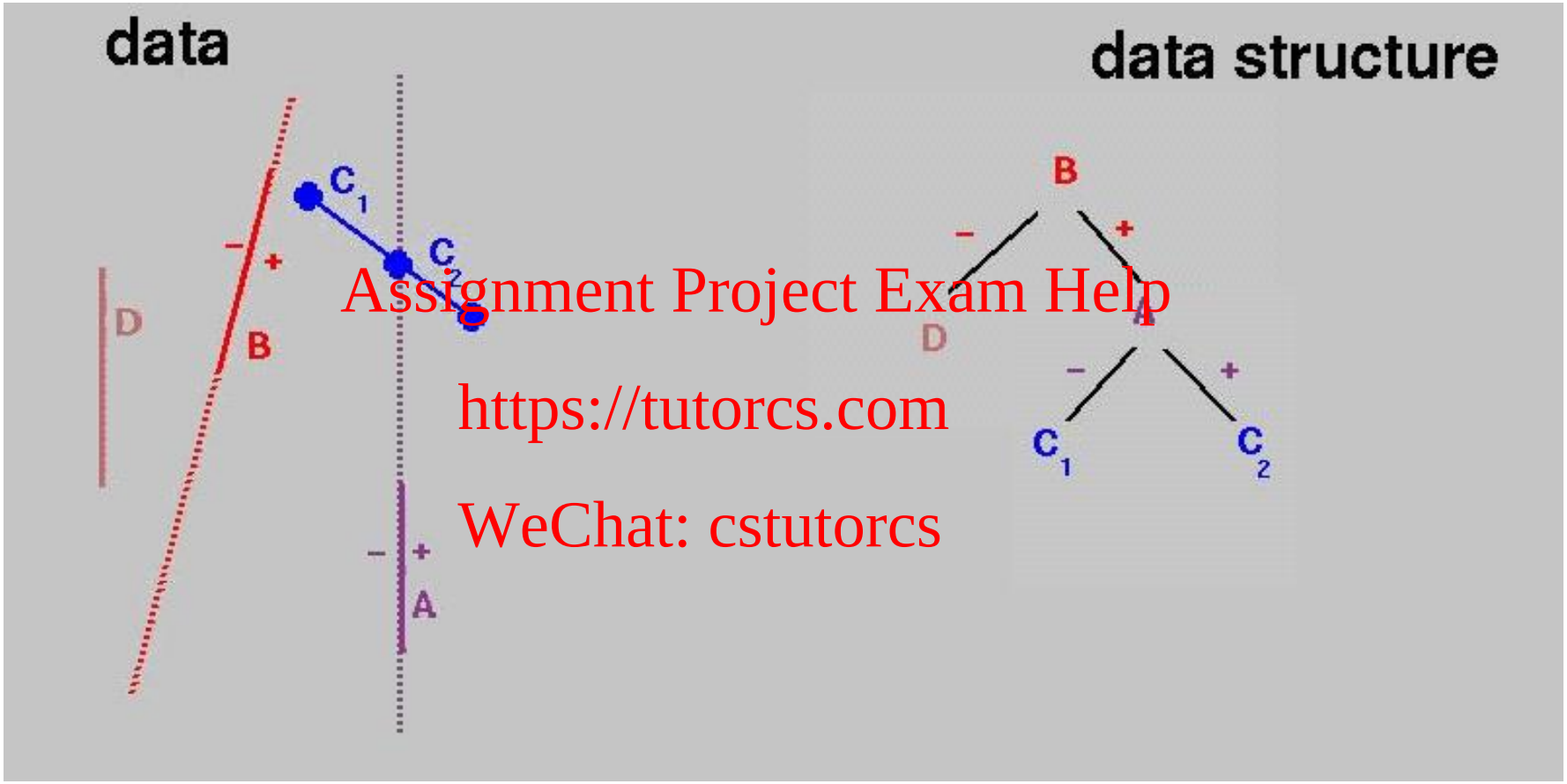


- kD-Tree gives hierarchy for scene graph

BSP-Trees

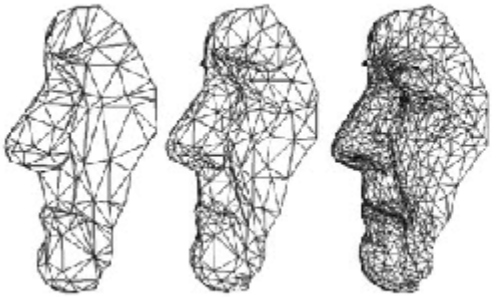
- Use a *binary space partitioning* (BSP) tree to order models
- Identify planes to *partition objects* into those in front of and those behind these planes hierarchically
  - For polygons we can choose one of them to define a *partitioning plane*
  - Polygons intersecting the plane are cut in two
  - Recursively continue splitting the polygon sets
- Particularly useful when view point changes, but objects remain at same position (partitioning does not change)
- kD-tree is a special case of BSP-tree

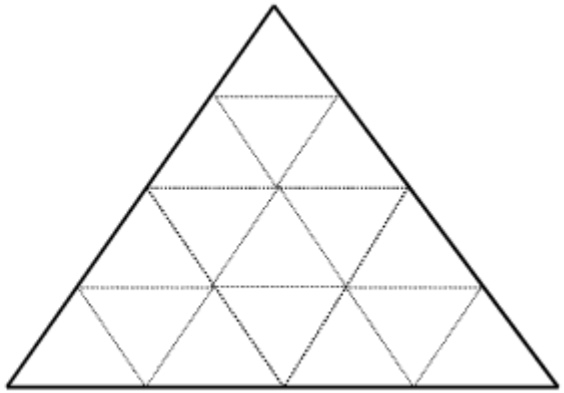
BSP Tree Example



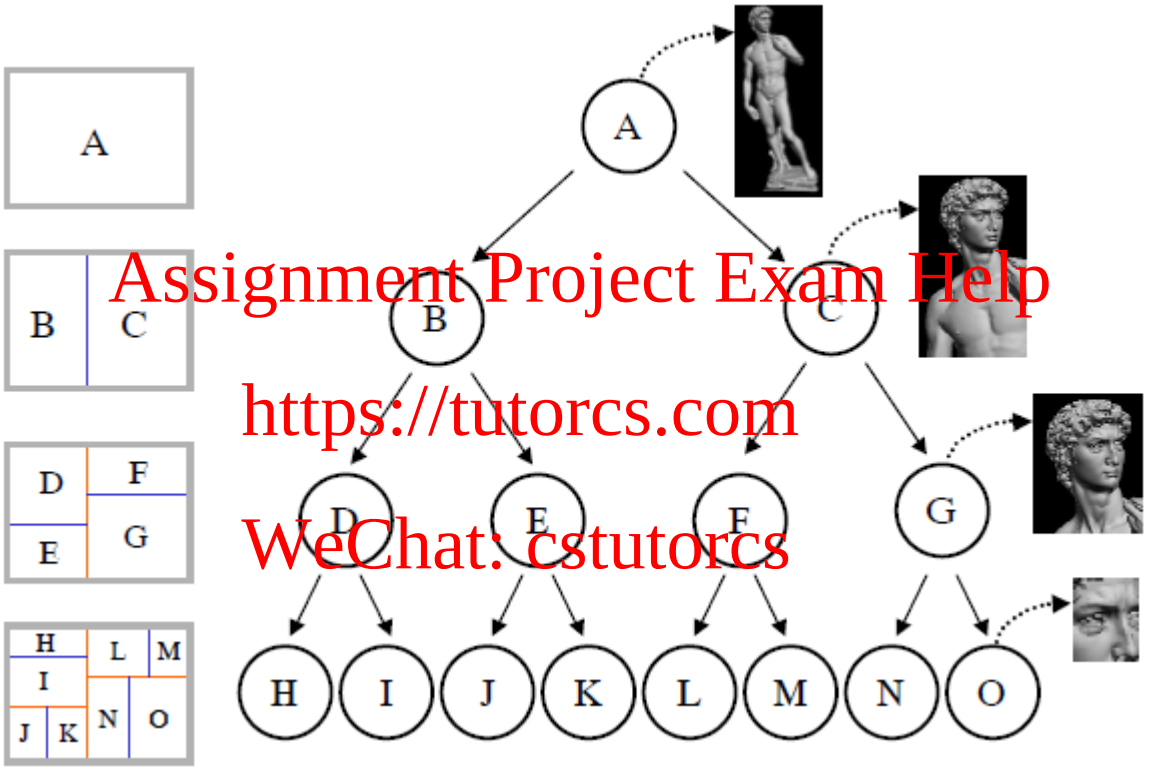
Multi-resolution models

- Hierarchical representation also suitable for simple *multi-resolution models*
  - Represent model at different levels of detail (LOD) for efficient rendering and processing

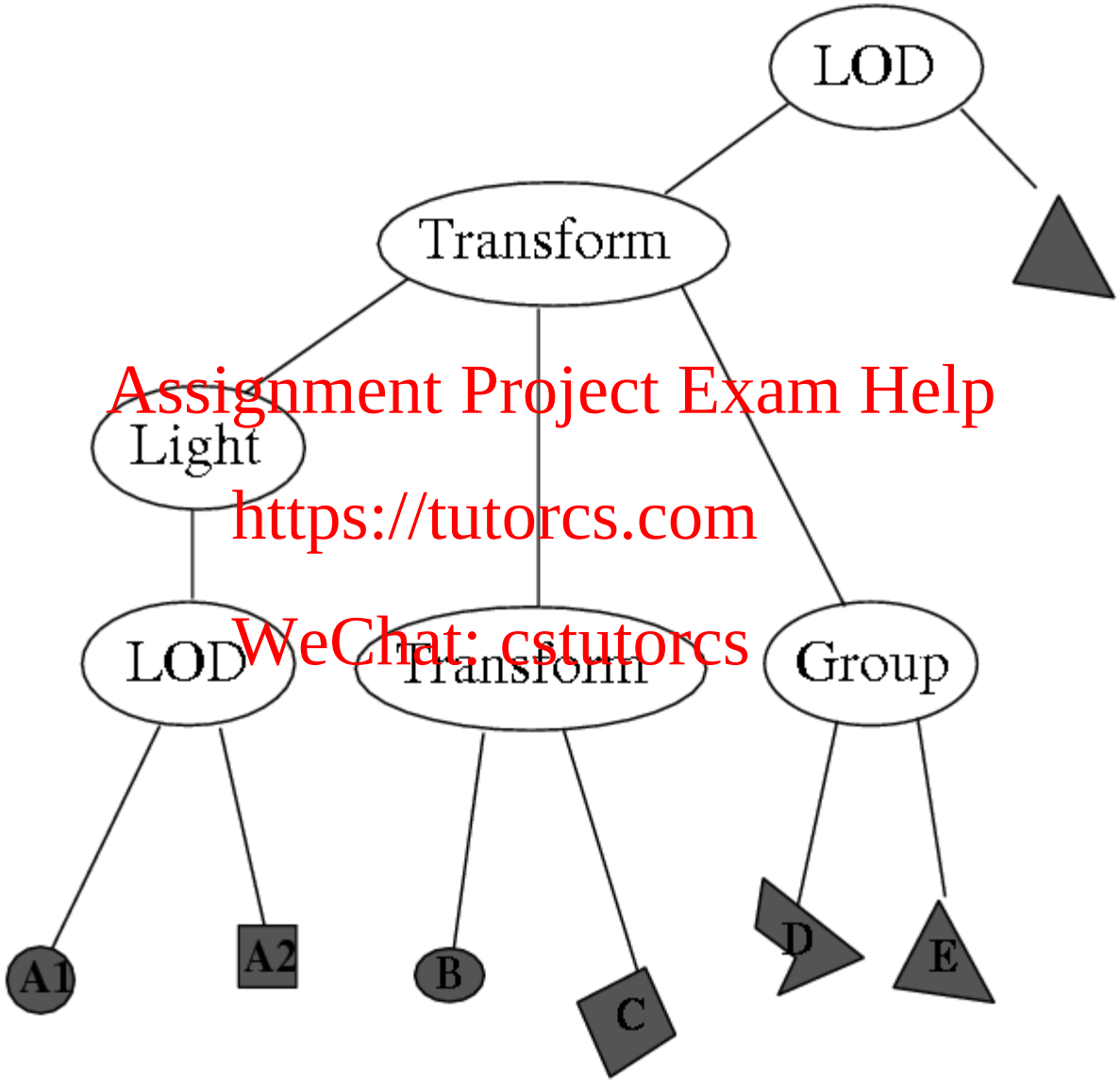




Multi-resolution Scene Graph



Multi-resolution Scene Graph



Scene Graph Issues

- *Minimise transformations*
  - Each transformation is expensive during rendering, etc.
  - Need automatic algorithms to reduce transformation nodes
- *Minimise attribute changes* (materials, etc.)
  - Each state change is expensive during rendering
- Many more scene graph optimisation problems. . .

Summary

- What is a scene graph / tree?
- Explain the principles of the following spatial data structures:
  - Uniform grid
  - Octree
  - kD-tree
  - BSP-tree
  - Given a set of objects, how are these data structures constructed?
  - How can these data structures be used to improve scene graph performance?