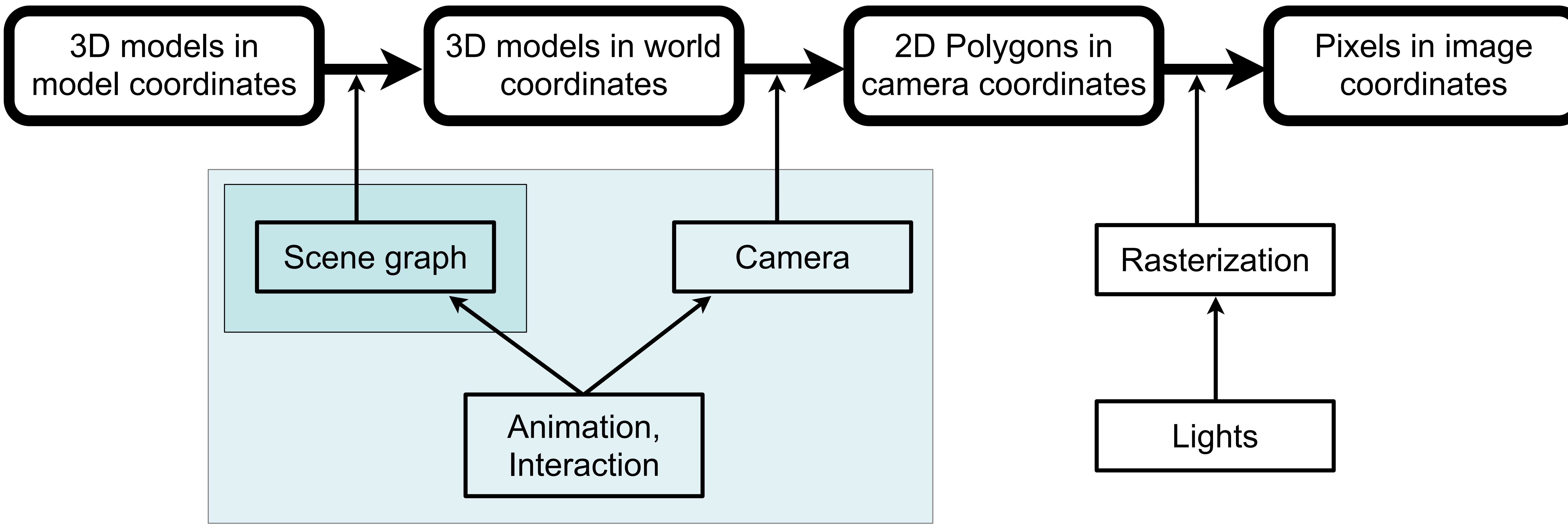
Computer Graphics 1

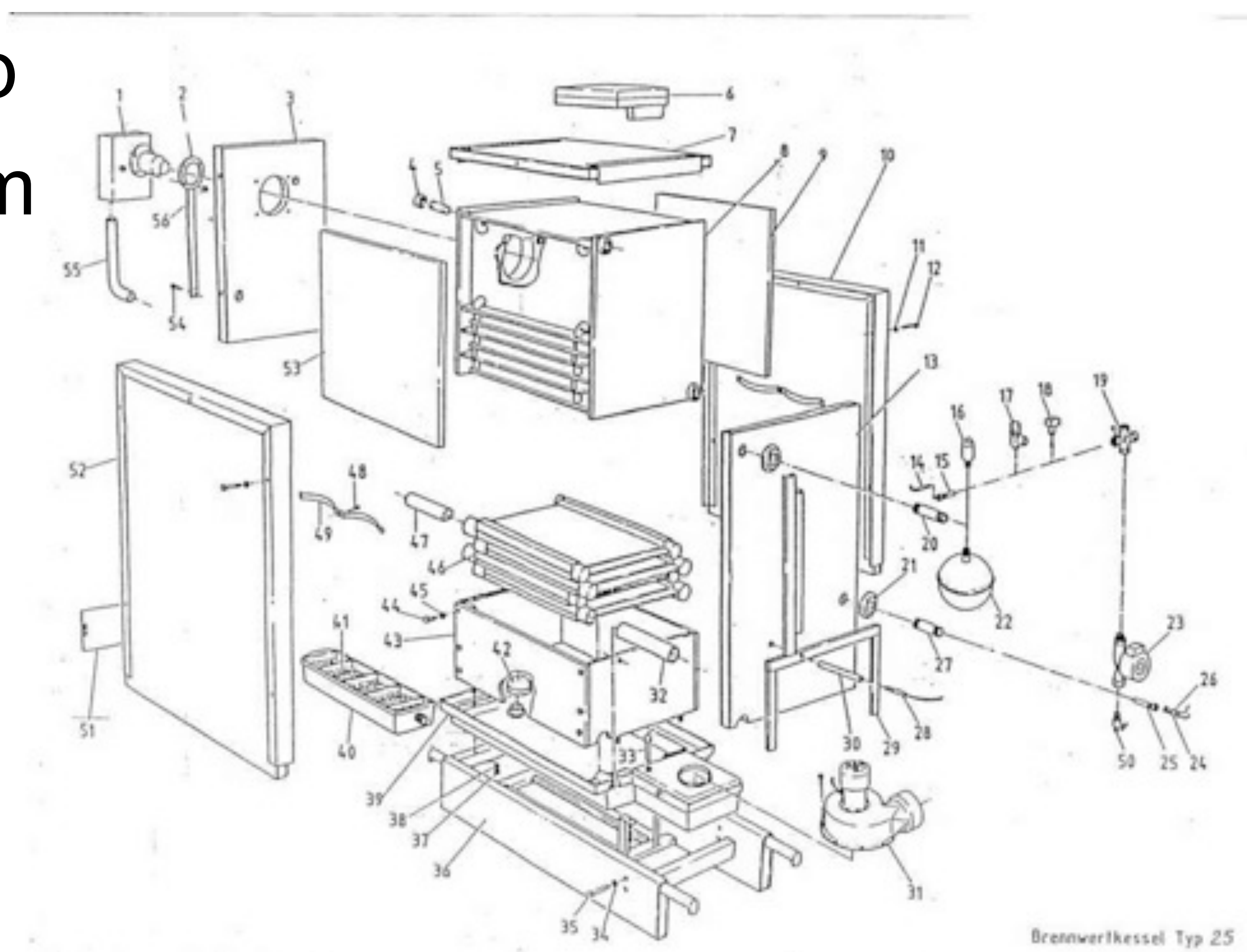
Chapter 4 (May 20th, 2010, 2-5pm):
The scene graph

The 3D rendering pipeline (our version for this class)



Why a scene graph?

- Naive approach: for each object in the scene, set its transformation by a single matrix (i.e., a tree 1 level deep and N nodes wide)
  - advantage: very fast for rendering
  - disadvantage: if several objects move, all their transforms change
- Observation: Things in the world are made from parts
- Approach: define an object hierarchy along the part-of relation
  - transform all parts only relative to the whole group
  - transform group as a whole with another transform
  - parts can be groups again

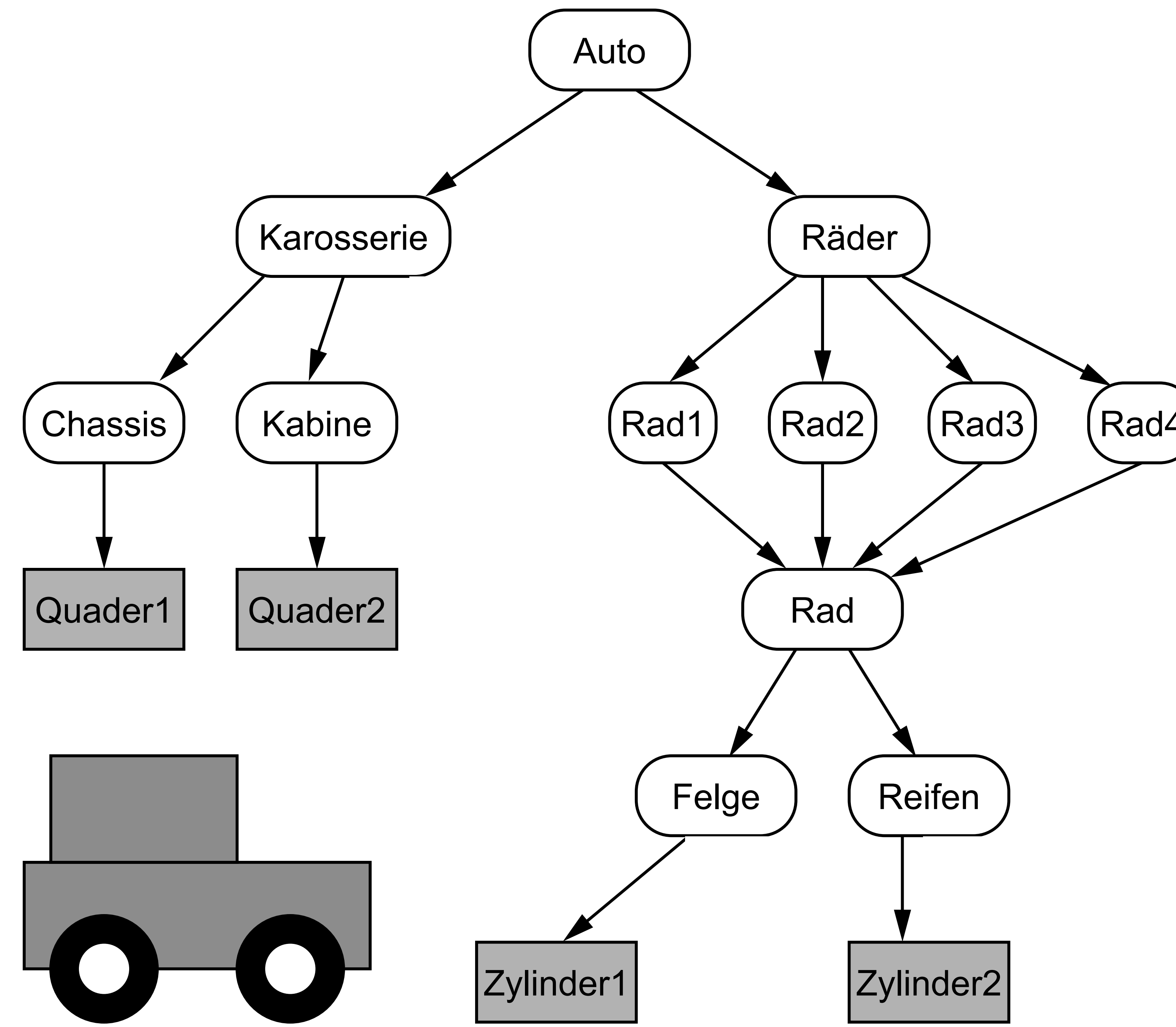


http://www.bosy-online.de/Veritherm/Explosionszeichnung.jpg

Geometry in the scene graph

- Leafs are basic 3D objects
- Non-leaf nodes (groups) contain a transformation
  - can have one or several children
  - transformation is given by a hom. Matrix
- Root is the entire world

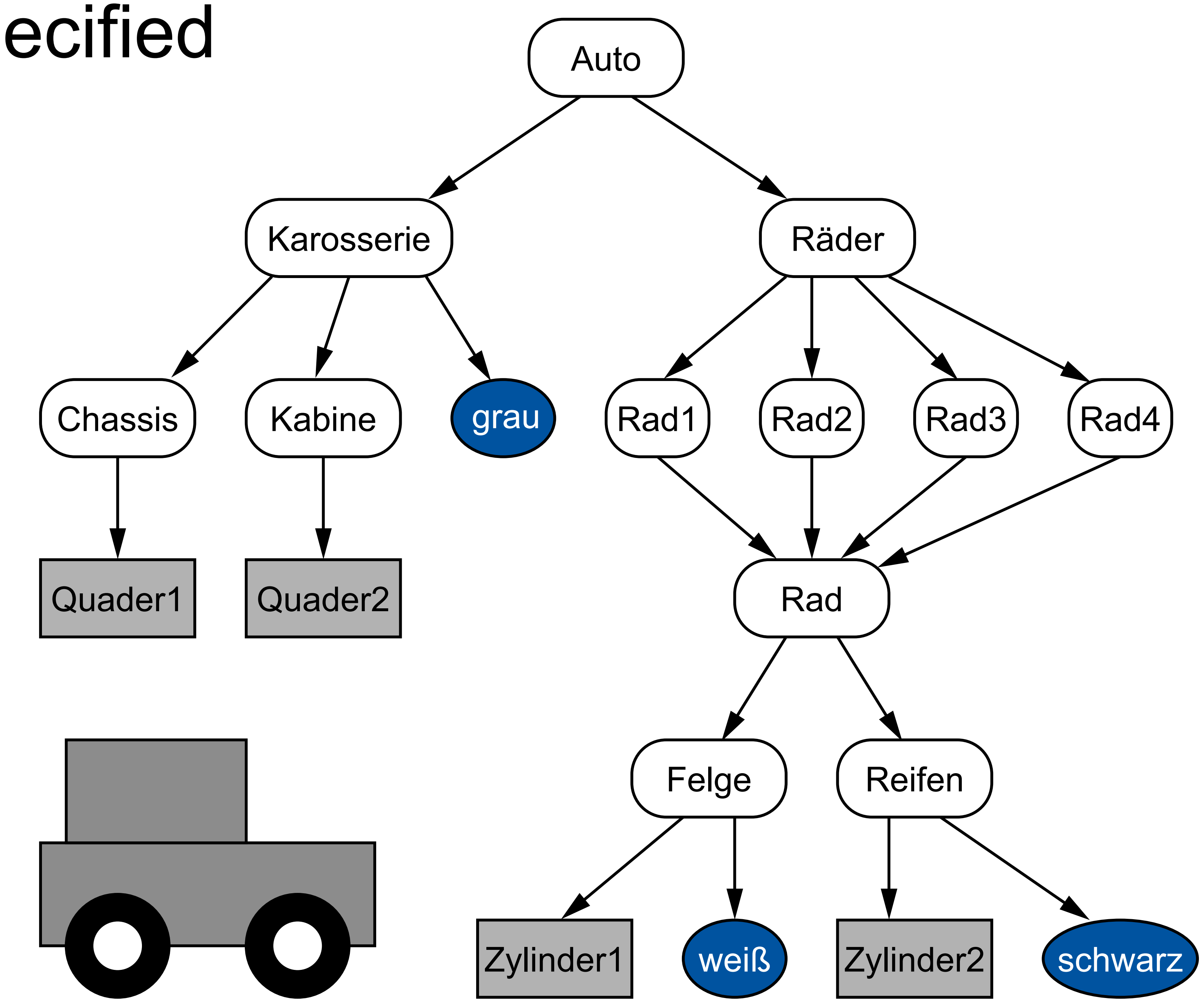
- Nodes can be the child of several groups
  - not a tree, but a directed acyclic graph (DAG)
  - effective reuse of geometry



Appearance in the scene graph

- Scene graph also contains appearances
  - can be reused similarly to geometry

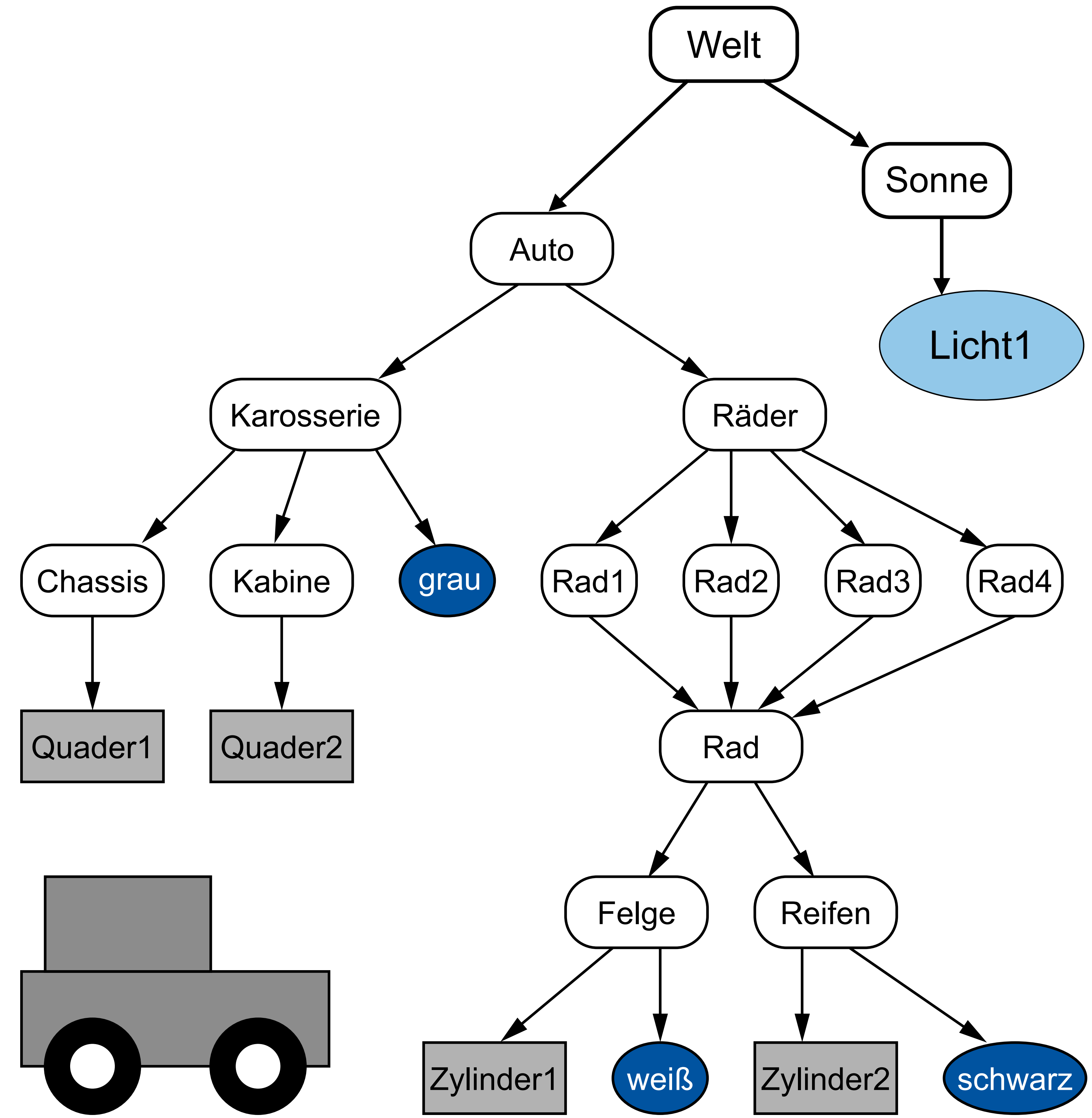
- Appearance can be only partially specified
  - unspecified values are inherited



Lights in the scene graph

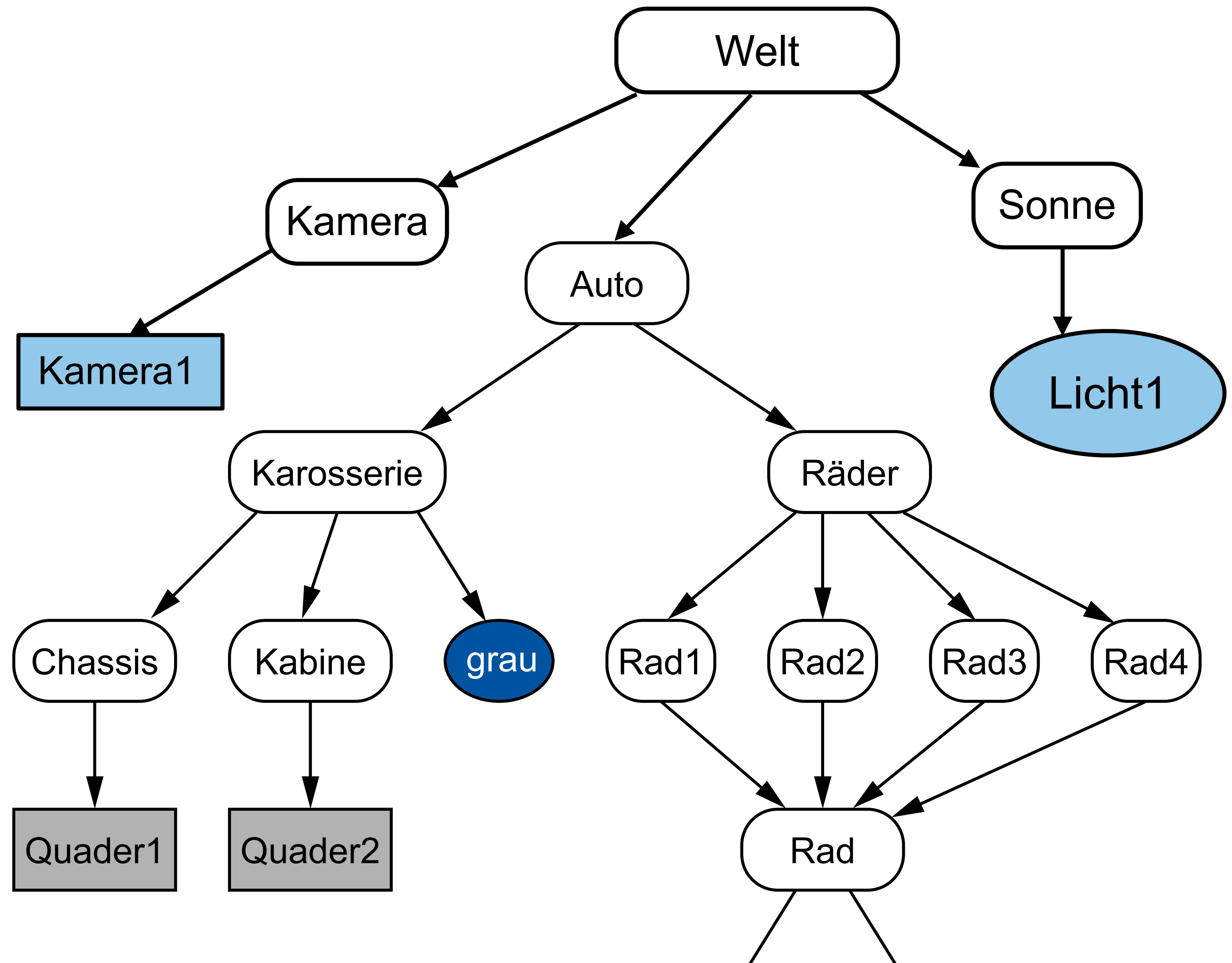
- Light sources also need a position and/or direction
  - Just include them into the scene graph
  - Can be animated just like geometry

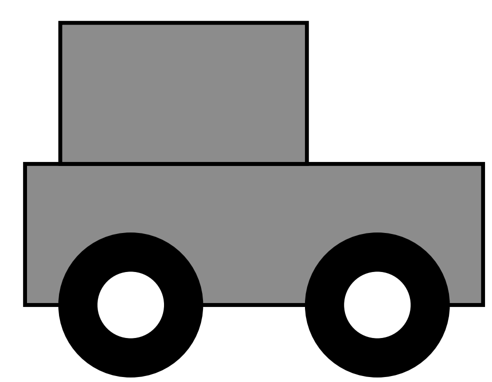
- Lights can be in local coordinarte systems of geometry groups
  - move with them
  - example: lights on a car



The camera in the scene graph

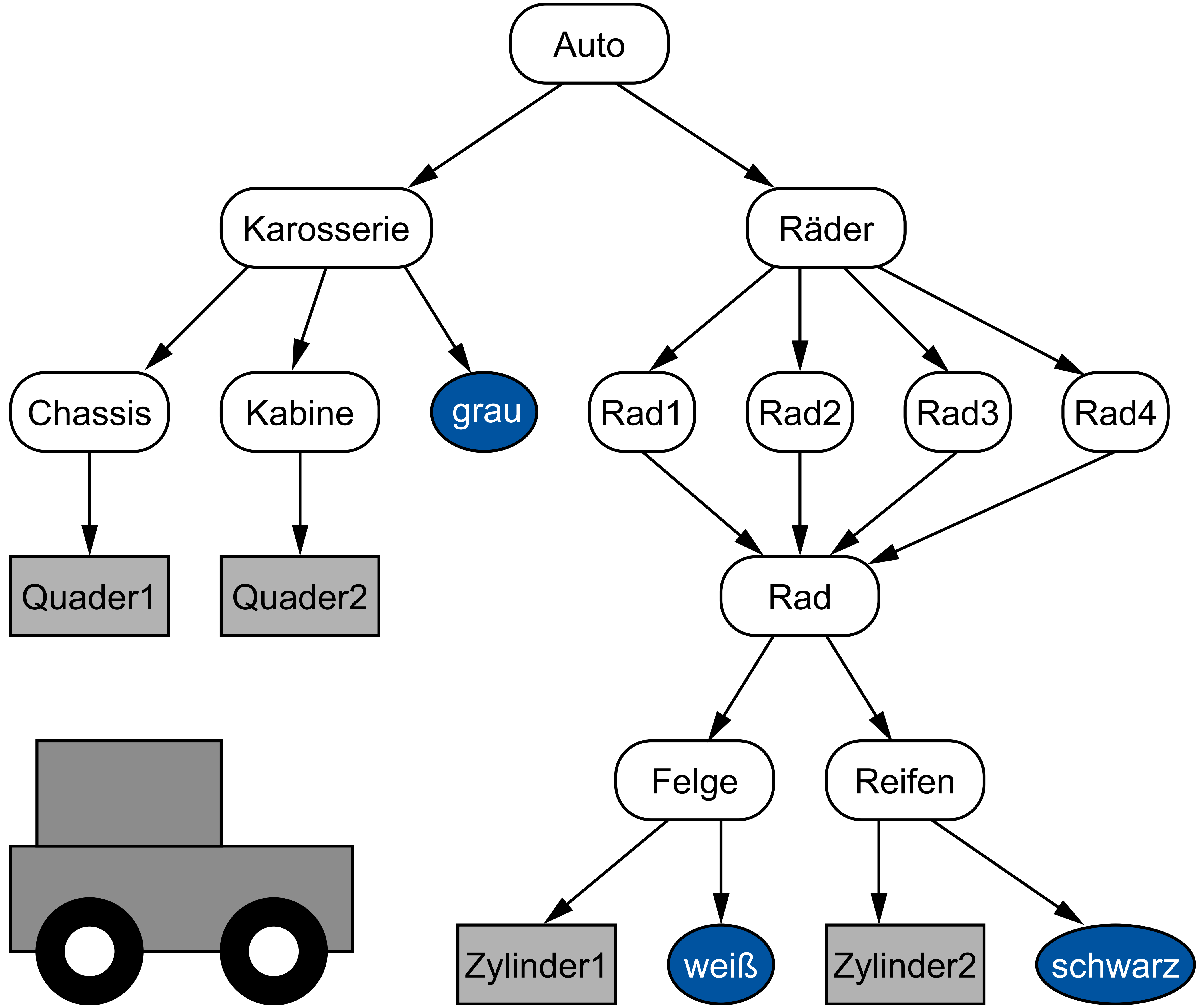
- Camera also needs a position and direction
  - Just include it into the scene graph
  - Can be animated just like geometry
- Camera can be in local coordinate systems of geometry groups
  - move with them
  - example: driver‘s view from a car





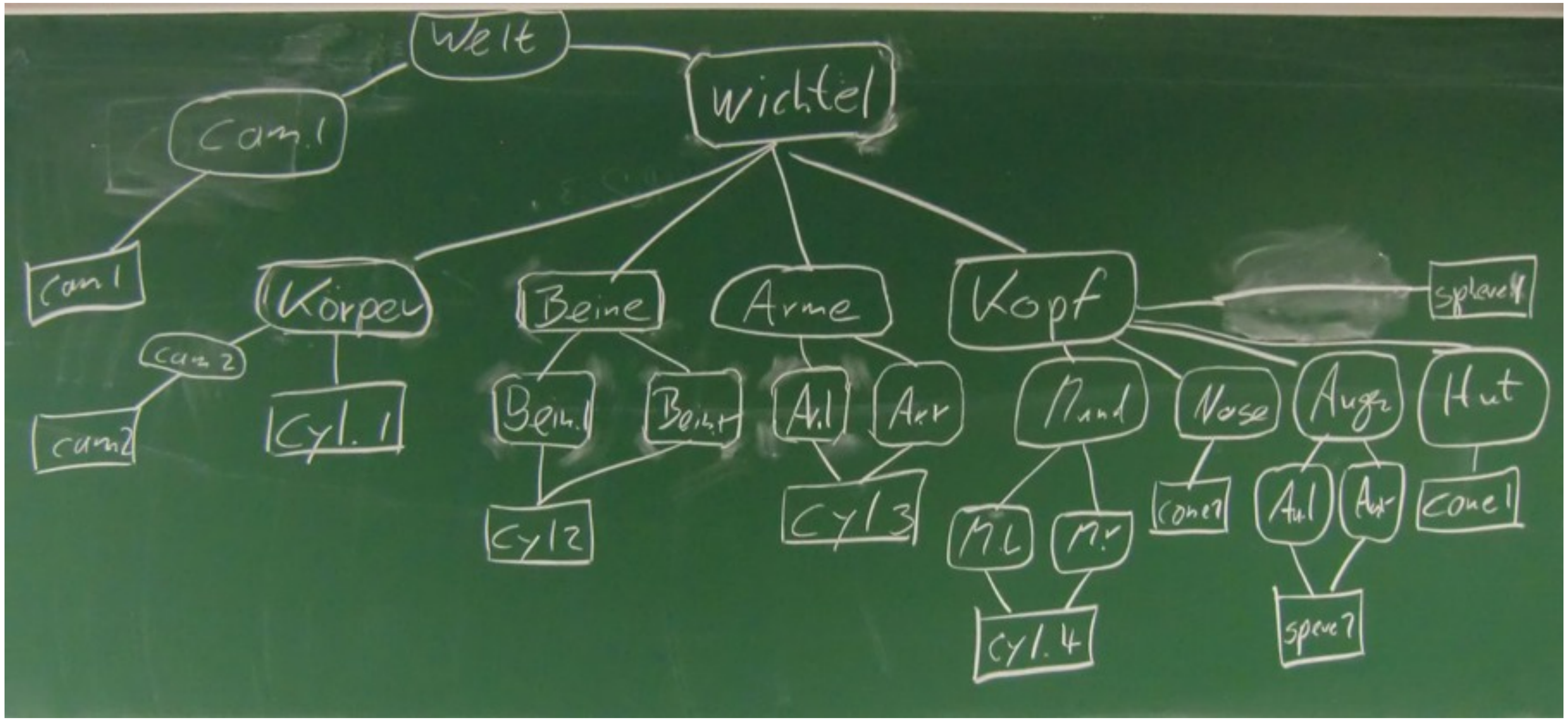
Scene graph traversal for rendering

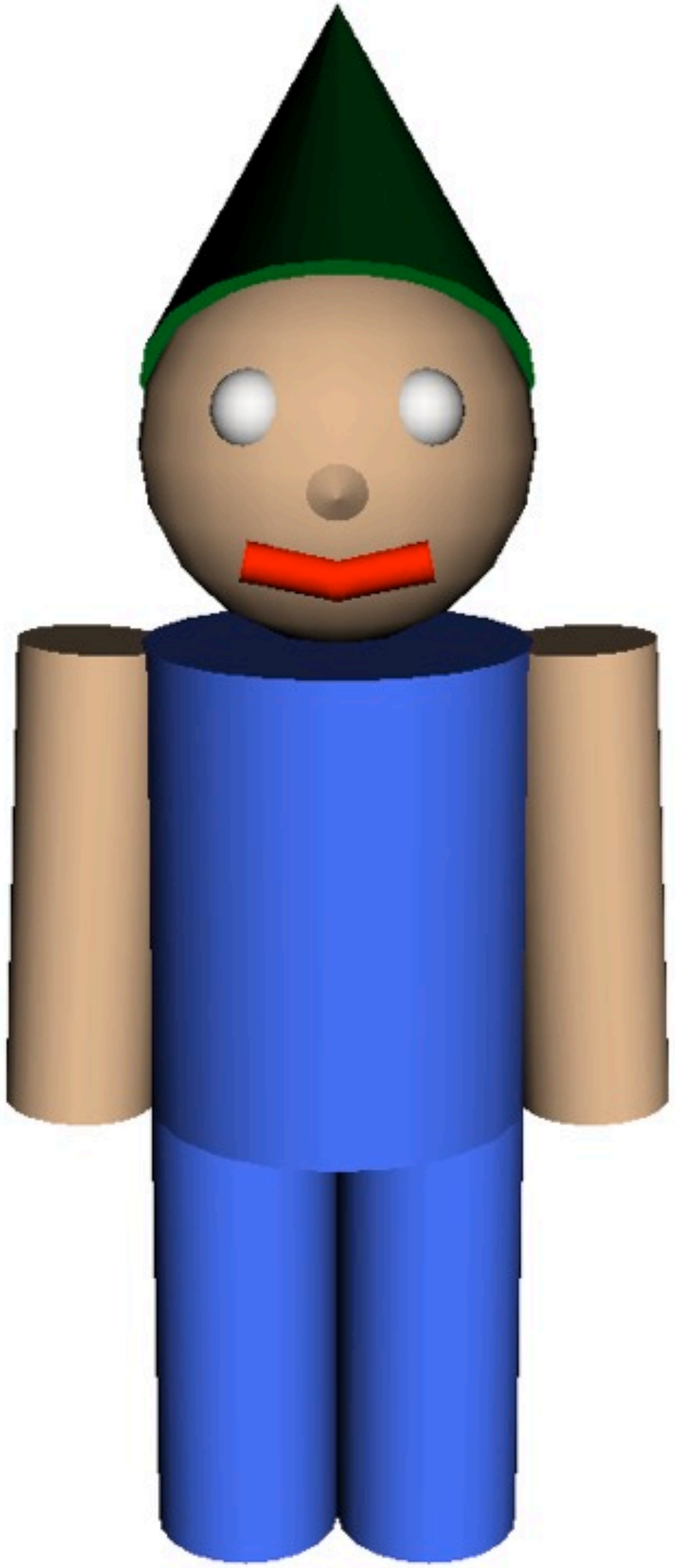
- set T_{act} to T_{Auto}
- save state
- set T_{act} to T_{act} x $T_{Karosserie}$
- save state
- set T_{act} to T_{act} x $T_{Chassis}$
- render Quader1
- restore state
- set T_{act} to T_{act} x T_{Kabine}
- render Quader2
- restore state
- restore state
- set T_{act} to T_{act} x $T_{Räder}$
- ...



Example of a scene graph

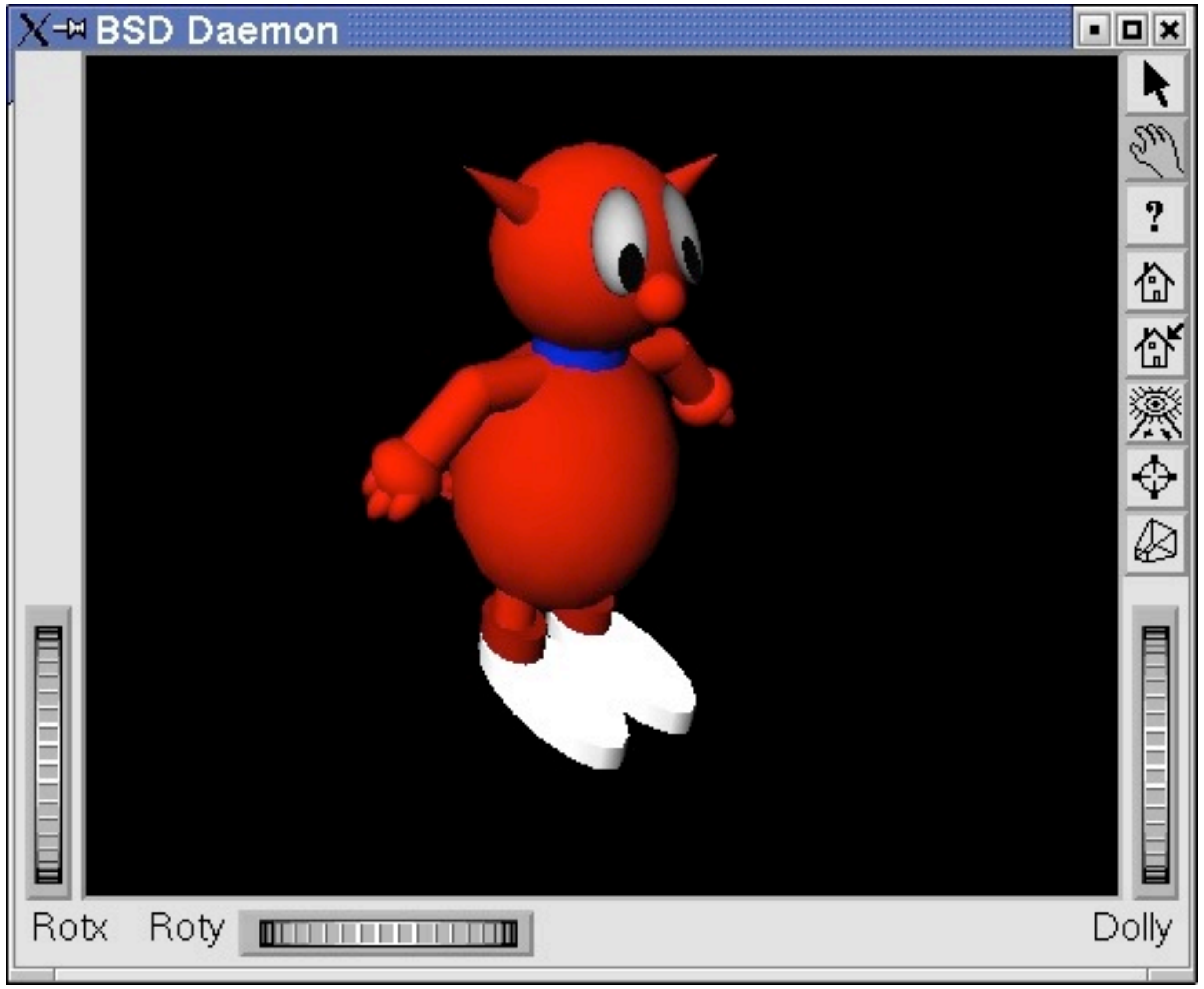
- Graph to be drawn together in the lecture
- VRML world linked from the class page





Scene graph libraries

- VRML/X3D
  - as seen in the examples
  - nice, because text format
- OpenInventor
  - based on C++ and OpenGL
  - used to be a commercial library
  - originally Silicon Graphics, 1988
  - now supported by VSG3d.com
- Java3D
  - Uses OpenGL for rendering
  - provides 3D data structures in Java
  - not supported anymore



http://www.shlomifish.org/open-source/bits-and-bobs/open-inventor-bsd-daemon/