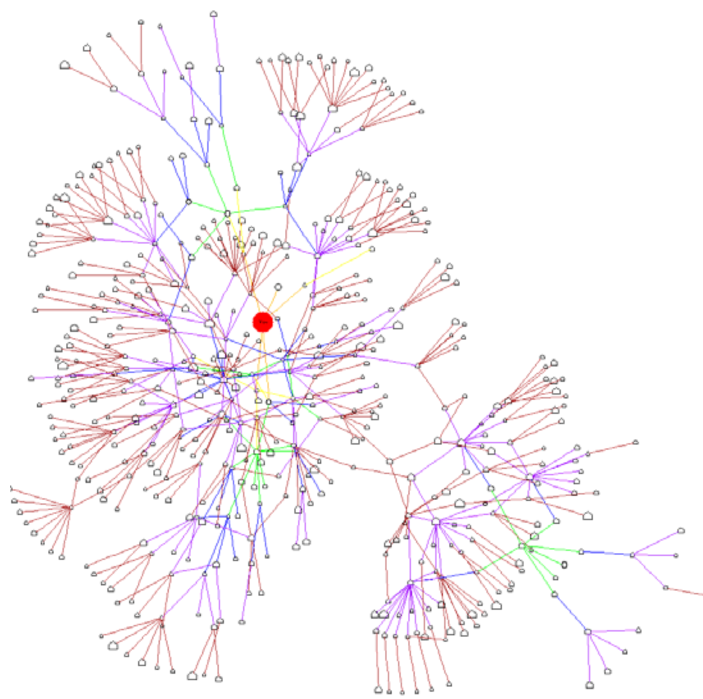
Prof. Dr. Andreas Butz

Dipl.-Medieninf. Hendrik Richter
Dipl.-Medieninf. Raphael Wimmer

5

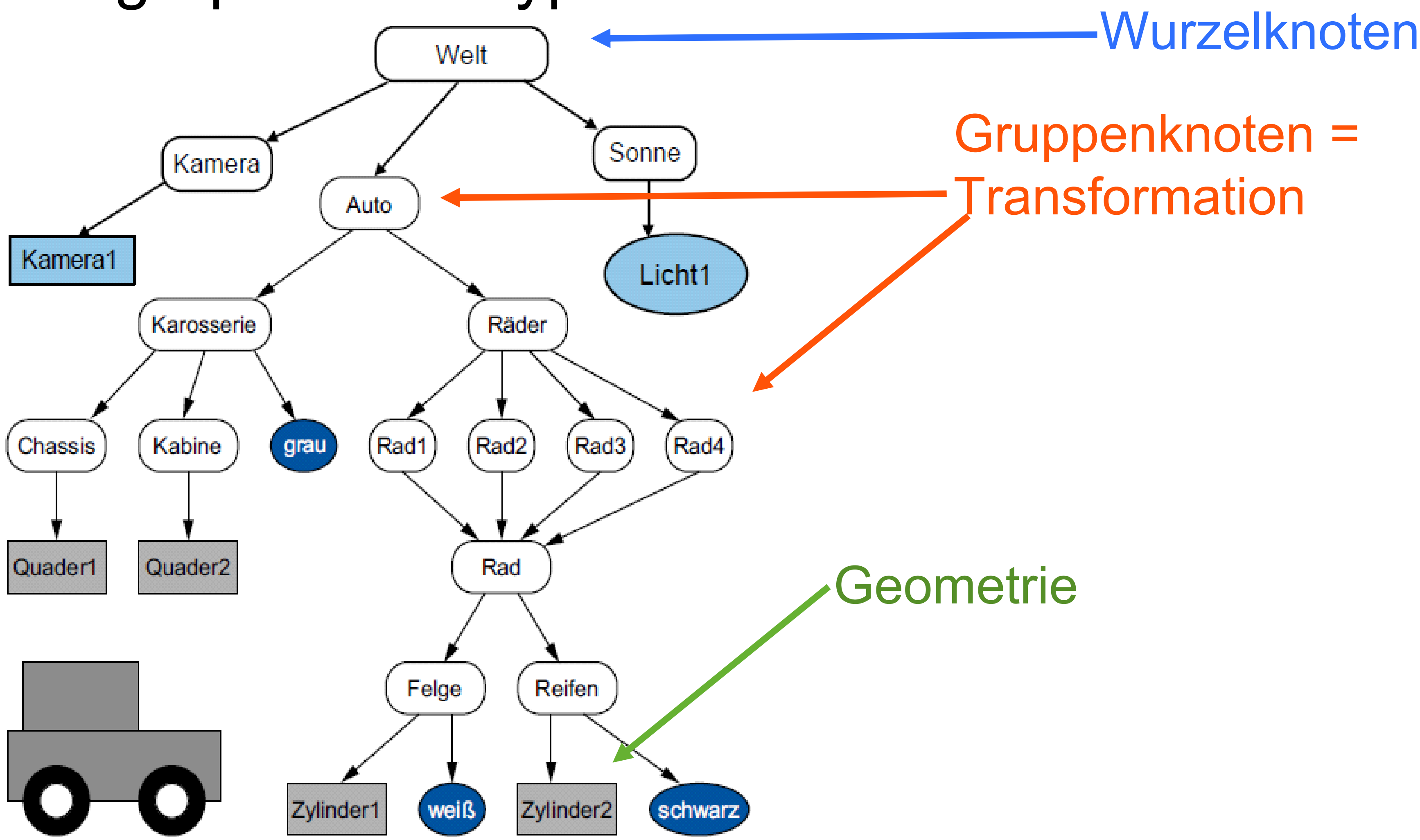
Computergrafik 1 Übung

Szenegraphen



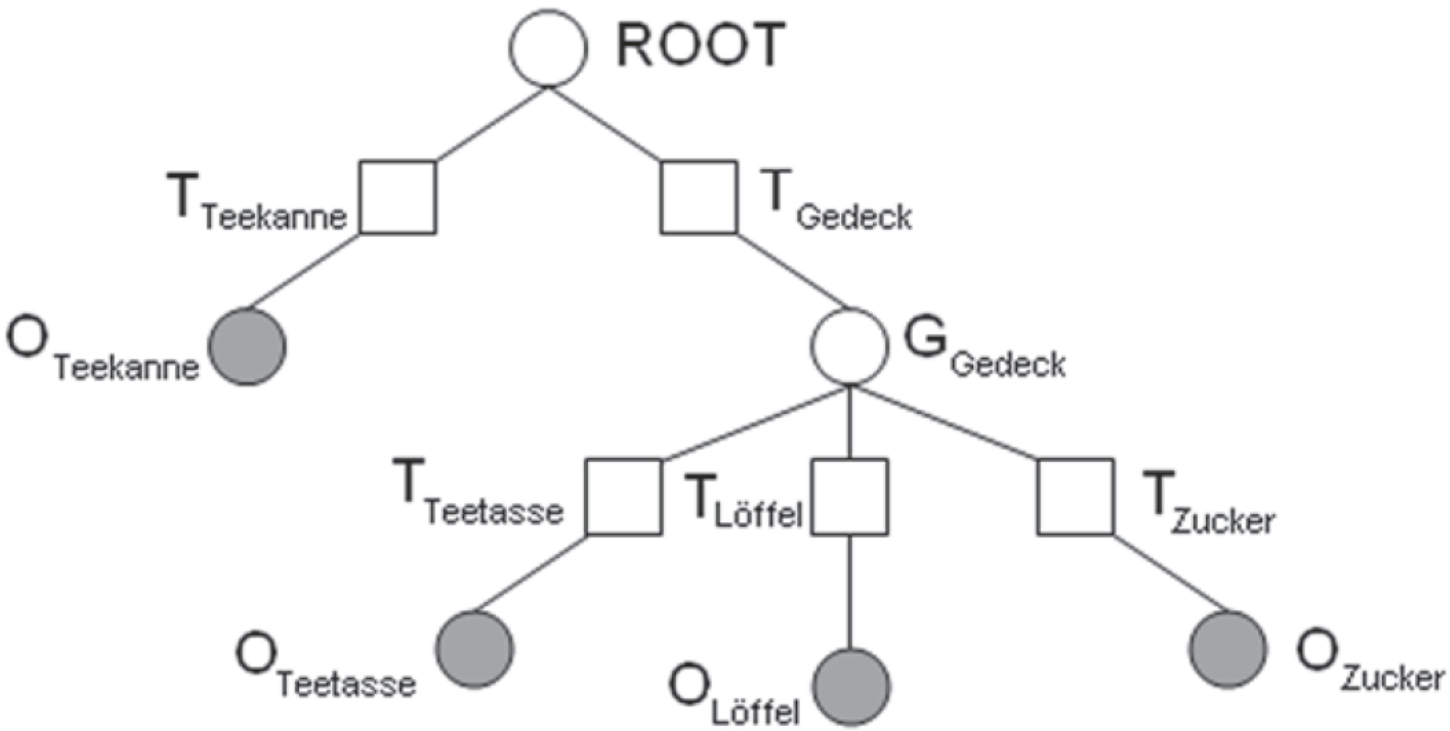
Szenegraph

Szenegraph - Nodetypen



Szenegraph - Nodetypen Beispiel 2

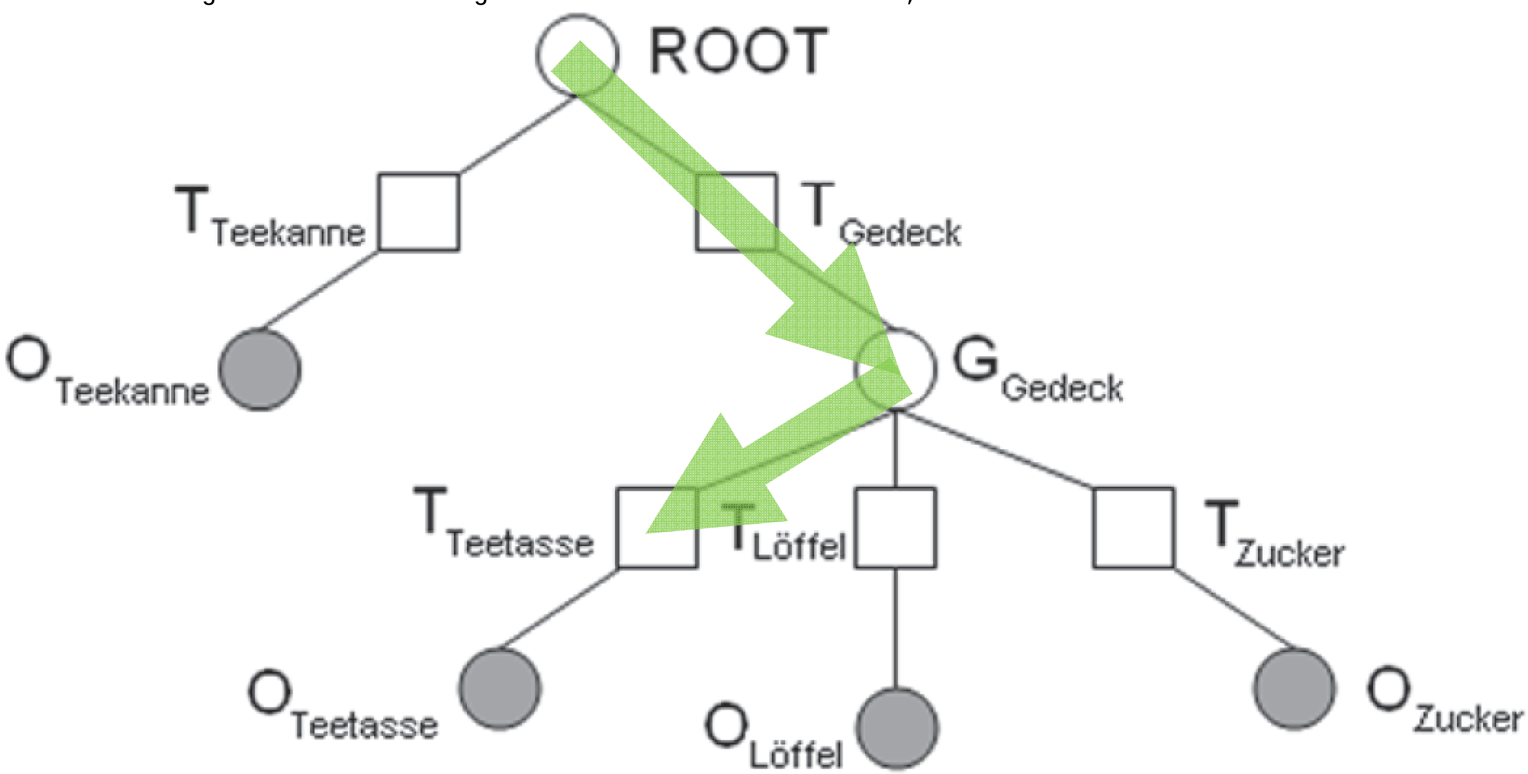
Gruppenknoten: Zusammenfassung mehrerer Kinder zur Hierarchisierung von Objekten
Transformationsknoten: geometrische Transformationen, z.B. Rotationen, Translationen
Strukturknoten: geometrische Objekte, z.B. ein Tisch oder Stühle



Szenegraph - Rendern

Rendern = Durchlauf in Tiefensuche, Abarbeiten von Kindern von Gruppenknoten in fester Ordnung

1. Bestimme den Knoten, an dem die Suche beginnen soll
2. Expandiere den Knoten und speichere alle Nachfolger in einem Stack
3. Rufe rekursiv für jeden der Knoten in dem Stack DFS (depth first search oder Tiefensuche) auf
4. Falls der Stack leer sein sollte, tue nichts
5. Falls das gesuchte Element gefunden worden sein sollte, brich die Suche ab und liefere ein Ergebnis



Implementierung (Beispiel)

Basisklasse **Node**, von der alle Nodetypen erben

Ein Node hat einen **Parent** und mehrere **Children** (jeder Node-Typ darf Kinder kriegen)

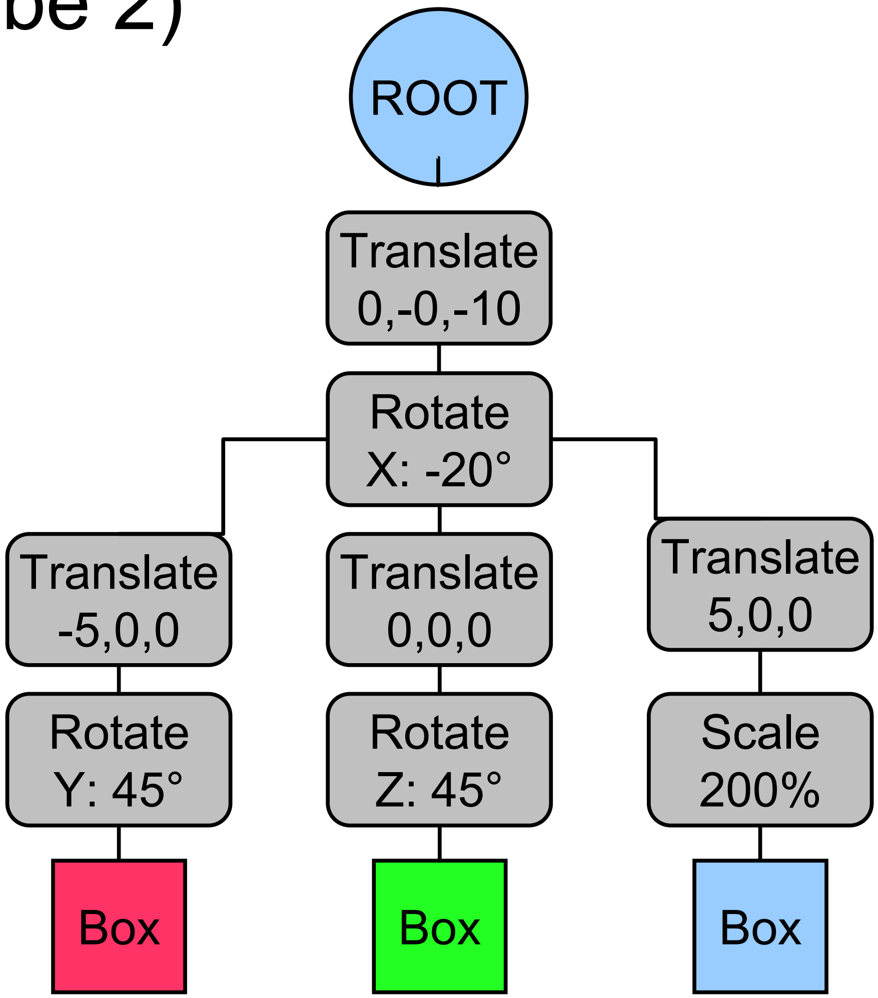
Es gibt einen **Root-Node** (generischer Typ)

Node kümmert sich darum, sich und seine Kinder zu zeichnen

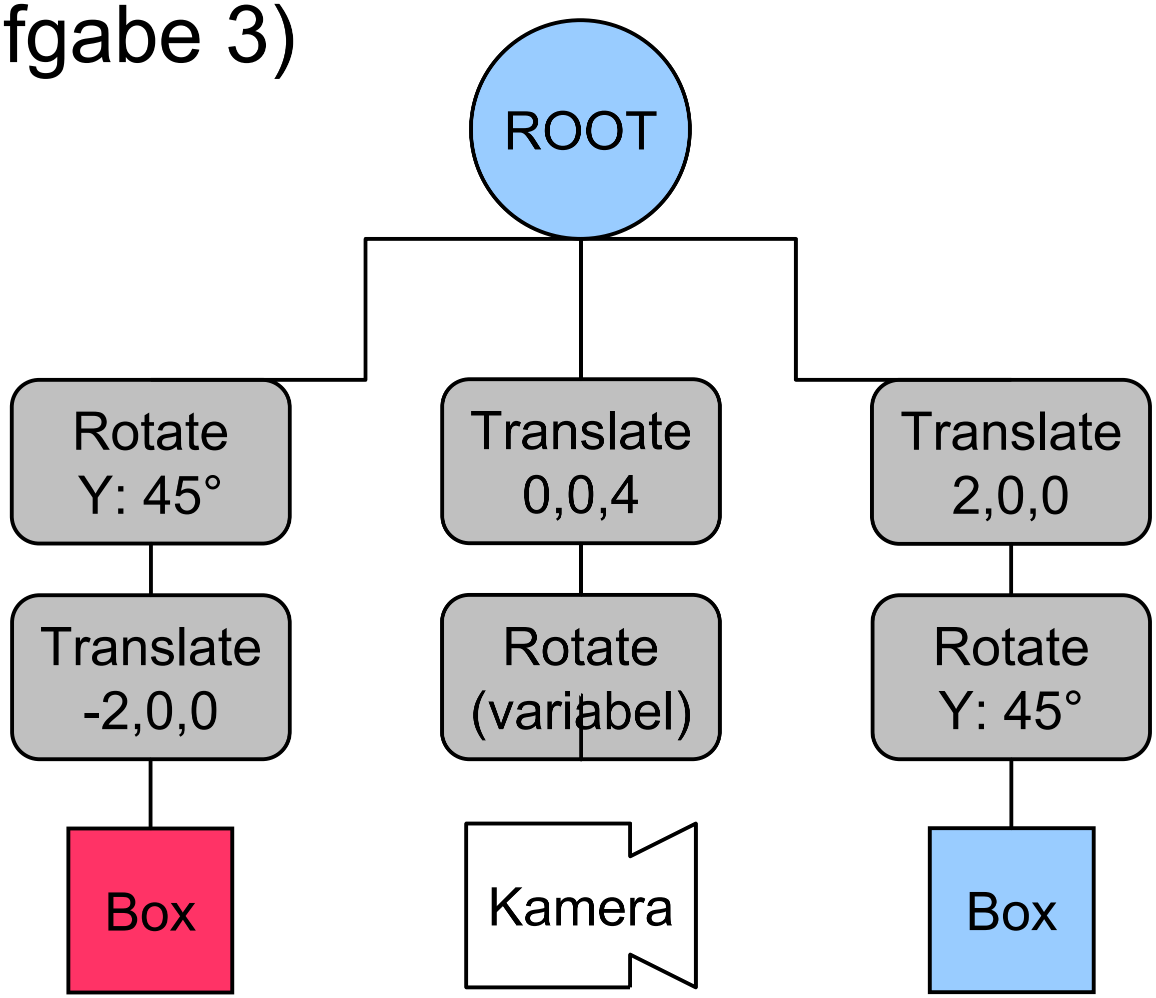
Node-Typen

- TransformNode
  - RotateNode
  - TranslateNode
  - …
- GeometryNode
  - BoxNode
  - MeshNode
  - …
- LightNode
  - OmniLightNode
  - AmbientLightNode
  - …
- CameraNode

Beispiel (Aufgabe 2)



Beispiel (Aufgabe 3)



Transformationen invertieren → Linksinverse

Matrizenmultiplikation ist nicht kommutativ!

Wenn gilt **a * v = e**
dann heisst **v** linksinvertierbar mit dem linksinversen Element **a**.

(auf Deutsch: eine Transformation ist invertierbar, wenn es eine Transformation gibt, die diese wieder aufhebt)

Transformationen invertieren → Linksinverse

Matrizenmultiplikation ist nicht kommutativ!

Wenn gilt **$a * v = e$**
dann heisst **v** linksinvertierbar mit dem linksinversen Element **a**.

(auf Deutsch: eine Transformation ist invertierbar, wenn es eine Transformation gibt, die diese wieder aufhebt)

Die gute Nachricht:
Transformationsmatrizen lassen sich einfach invertieren

Translationsmatrix invertieren

$$T = \begin{pmatrix} 0 & t \\ 0 & 1 \end{pmatrix} \Rightarrow T^{-1} = \begin{pmatrix} 0 & -t \\ 0 & 1 \end{pmatrix}$$

Rotationsmatrix invertieren

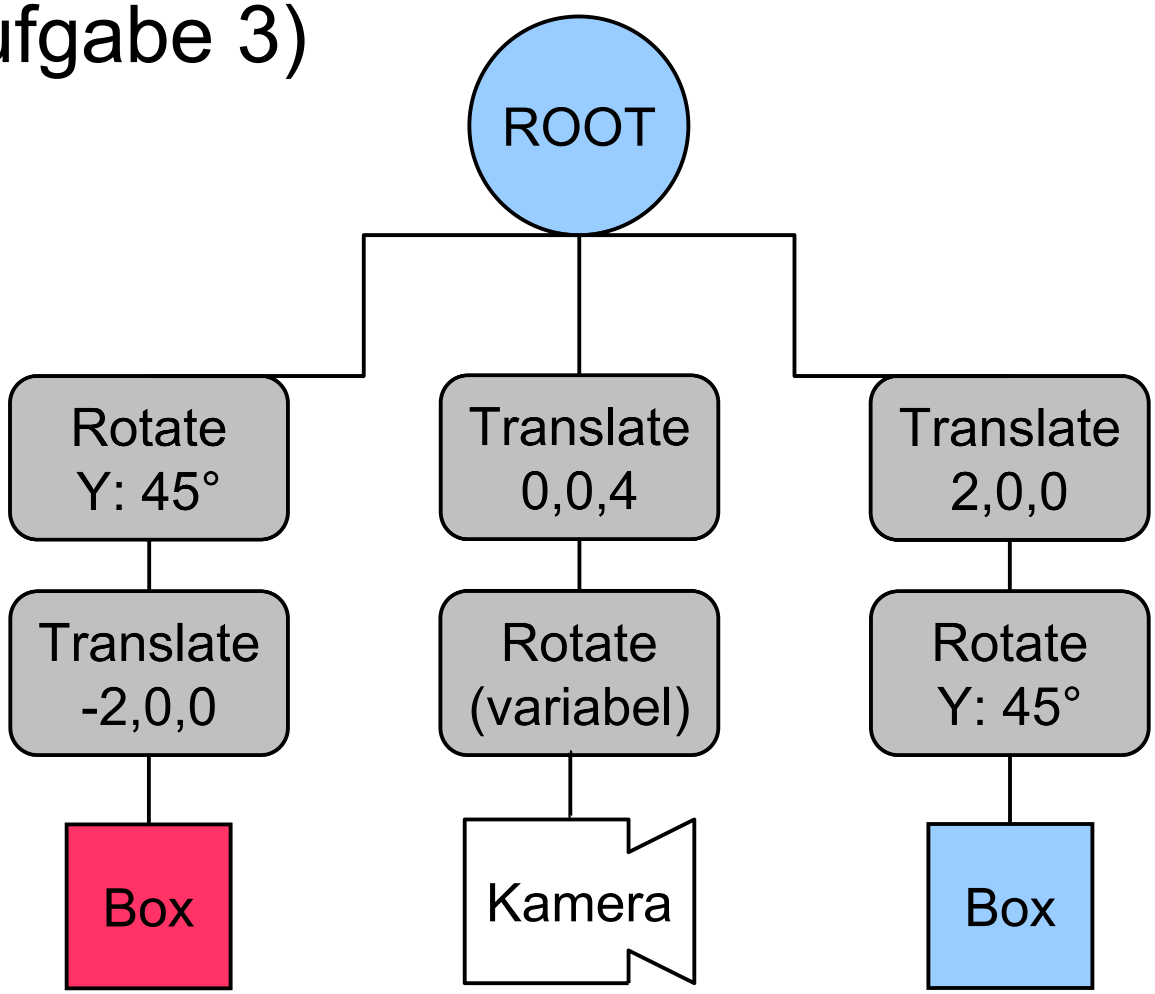
$$T = \begin{pmatrix} R^* & 0 \\ 0 & 1 \end{pmatrix} \Rightarrow T^{-1} = \begin{pmatrix} R^{*-1} & 0 \\ 0 & 1 \end{pmatrix}$$

$$\mathrm{R}^{*-1} = \mathrm{R}^{*T}$$

Skalierungsmatrix invertieren

$$T = \begin{pmatrix} S* & 0 \\ 0 & 1 \end{pmatrix} \Rightarrow T^{-1} = \begin{pmatrix} \frac{1}{S*} & 0 \\ 0 & 1 \end{pmatrix}$$

Beispiel (Aufgabe 3)



Rekursives Zeichnen des Baumes

```
void Node::apply(){
    applySelf();
    for (unsigned int i = 0; i < children.size(); i++){
        glPushMatrix();
        children[i]->apply();
        glPopMatrix();
    }
}
void BoxNode::applySelf(){
    glColor3d(red, green, blue);
    GLdouble vertices[] = ...;
    GLubyte indices[] = ...;
    // draw cube
}
void TransformNode::applySelf(){
    glMultMatrixd(matrix);
}
```

Vielen Dank!