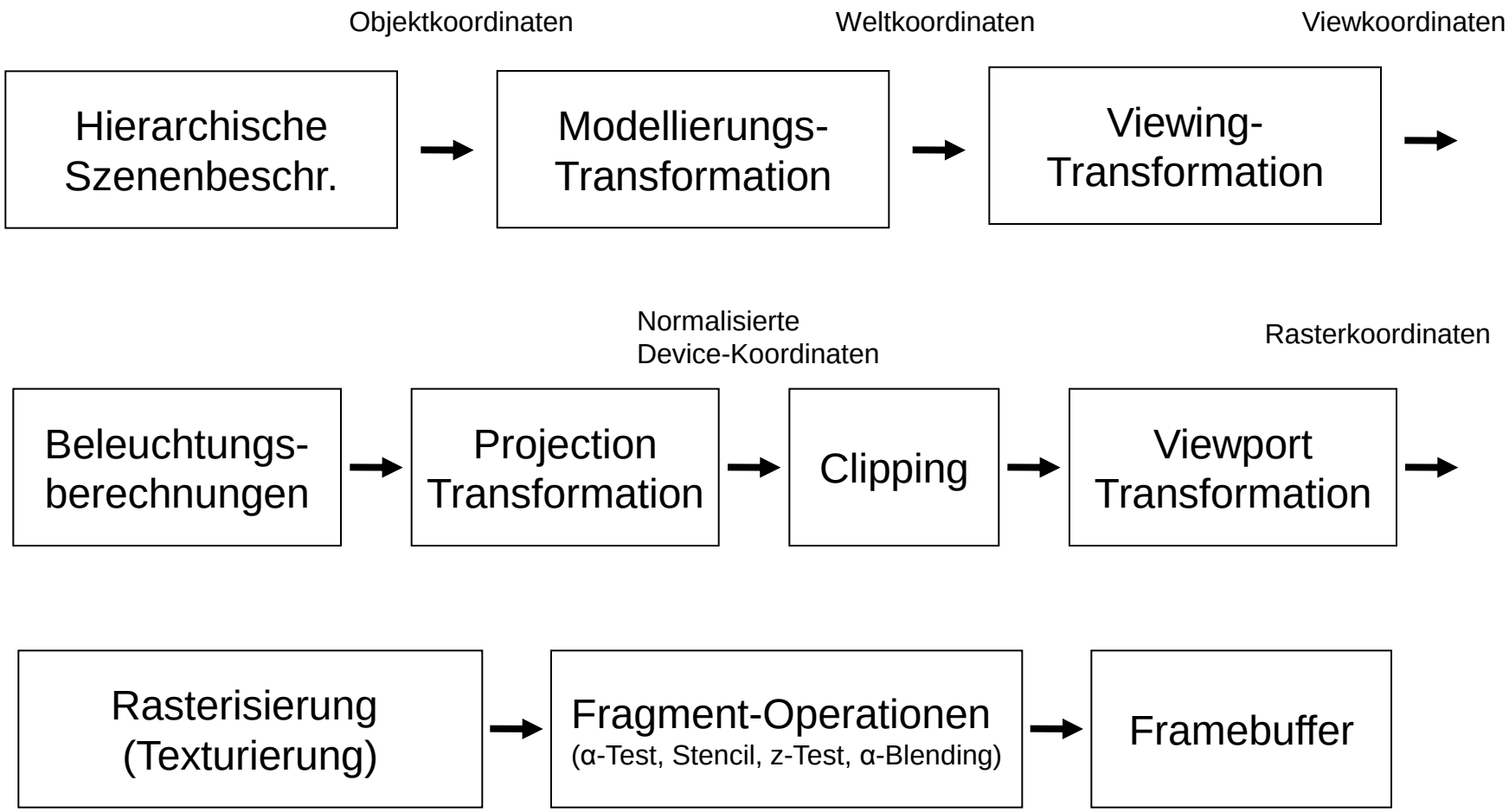
Graphik-Praktikum 2013

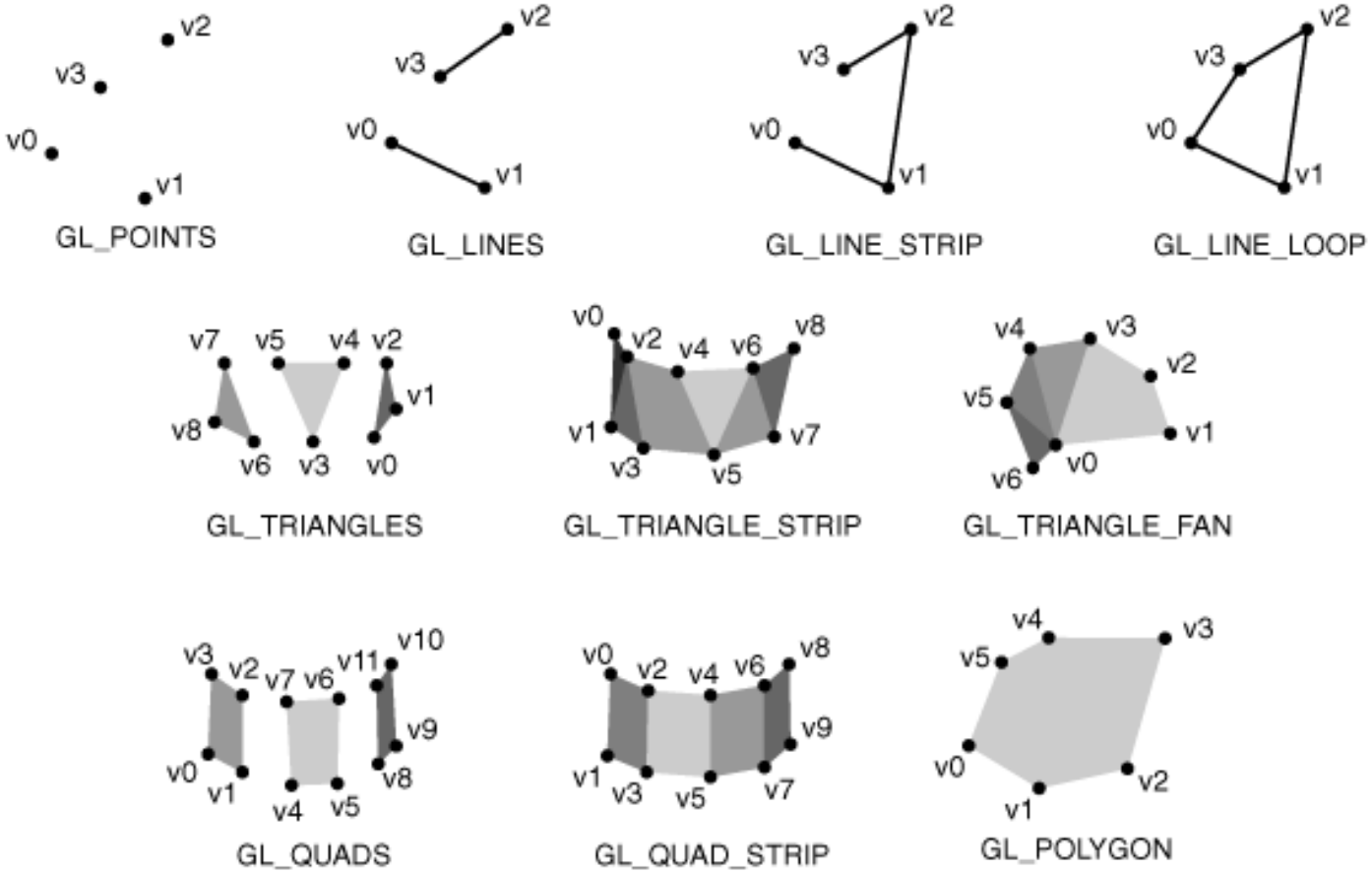
Aufgabe 1

University of Siegen • Institute for Vision and Graphics
Computer Graphics and Multimedia Systems Group

Pipeline Overview



Primitive Typen



Geometrie Definition

```
glColor4f(r, g, b, a)

glBegin(GL_TRIANGLES);
    glVertex3f(x1, y1, z1);
    glVertex3f(x2, y2, z2);
    glVertex3f(x3, y3, z3);
glEnd();
```

2 - (x, y)
3 - (x, y, z)
4 - (x, y, z, w)

b	byte
ub	unsigned byte
s	short
us	unsigned short
i	int
ui	unsigned int
f	float
d	double
v	Vektor / Array

Transformationen

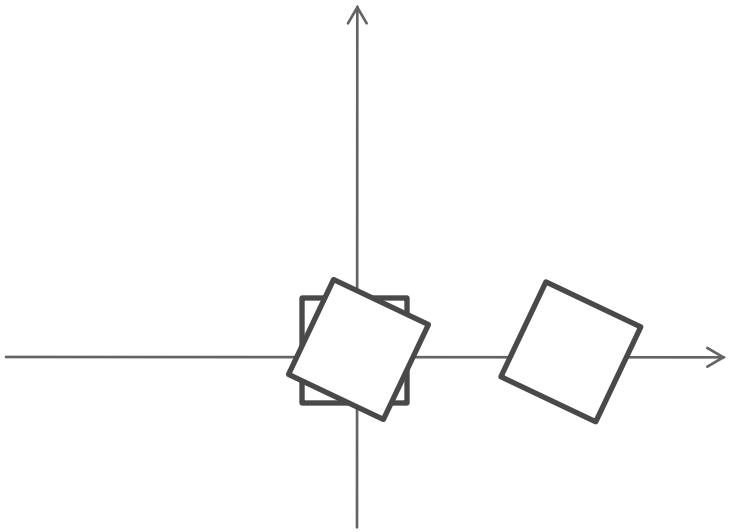
- Transformationen
  - glTranslate, glRotate, glScale
  - **vor** Primitiv-Definition

- glMatrixMode(*mode*)
  - GL_MODELVIEW – Objekt & Kamera-Transformation
  - GL_PROJECTION – Projektions-Transformation
  - GL_TEXTURE – Textur-Transformation

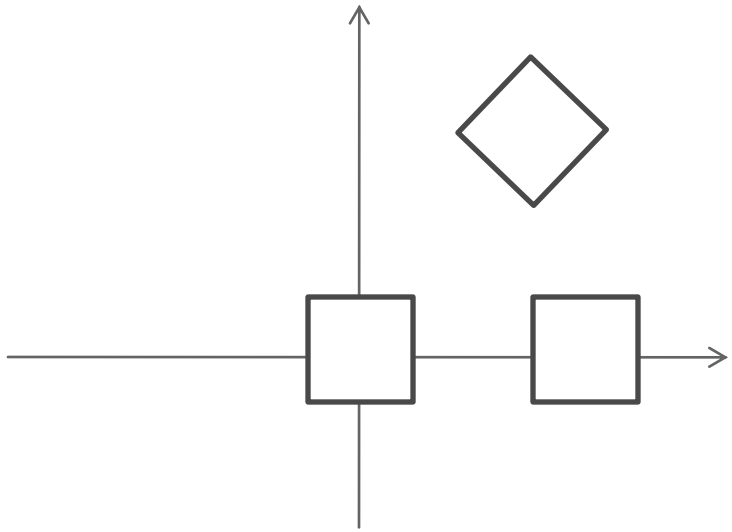
- glPush() / glPop()
  - Speichern / Laden der aktuellen Matrix
  - Realisierung hierarchischer Transformationen (lokales Koordinatensystem)

Transformationen – Reihenfolge

glTranslate()
glRotate()



glRotate()
glTranslate()



- glEnable(**GL_LIGHTING**)
- glEnable(**GL_LIGHT0**)

- glLight() – Lichtquellen-Eigenschaften z.B. Öffnungswinkel
- glLightModel() – allg. Beleuchtungsparameter

- glMaterial() / **glColorMaterial** – Materialeigenschaften
- glNormal3f(n_1, n_2, n_3)

Texturing – Ablauf

- glEnable(GL_TEXTURE_**X**D) - **X** $\in$ [1, 2, 3]
- glGenTextures(GLsizei *n*, GLuint* *textureIDs*)
  IDs für die Texturen reservieren
- glActiveTexture(GL_TEXTURE0)
  Texture-Unit setzen
- glBindTexture(GL_TEXTURE_**X**D, texture*ID*)
  Binden der Textur
- glTexParameter, glTexImage
  Texturparameter setzen und Textur laden
- *Geometrie mit Texturkoordinaten zeichnen*
- glDeleteTextures(GLsizei *n*, GLuint* *textureIDs*)
  IDs für die Texturen wieder freigeben

Texturing – Erstellung, Parameter

glTexImage**X**D(	
target: Glenum,	z.B. **GL_TEXTURE_2D**
level: Glint,	MipMap-Level (meist **0**)
internalformat: Glint,	z.B. **GL_RGBA**
width, *height*: Glsizei,	$2^n + 2$ x *border*
border: Glint,	**0** o. **1**
format, type: Glenum	z.B. **GL_RGB, GL_UNSIGNED_BYTE**
const *pixels*: Glvoid*	**const void*** pointer
)	
glTexParameterf(	
target: Glenum,	z.B. **GL_TEXTURE_2D, …**
pname: Glenum,	z.B. **GL_TEXTURE_MIN_FILTER**
params: Glfloat	GL_NEAREST o. GL_LINEAR
)	

Texturing – Koordinatenerstellung

- glTexCoord2f (s, t), glMultiTexCoord2f (GL_TEXTURE**X**, s, t)
  - s = 0..1
  - von unten links zeilenweise nach oben rechts
- glMatrixMode(**GL_TEXTURE**) - Transformation von Texturkoordinaten

- glEnable(**GL_TEXTURE_GEN_X**) - **X** $\in$ [**S**, **T**]
- glTexGen(

  coord, z.B. **GL_S, GL_T**

  pname, **GL_TEXTURE_GEN_MODE**

  params z.B. **GL_OBJECT_LINEAR, GL_EYE_LINEAR**

)
- **OBJECT_LINEAR:** Textur wird im **Objektraum** projiziert (bewegt sich mit Objekt mit)
- **EYE_LINEAR:** Textur wird im **Kameraraum** projiziert (bewegt sich mit Kamera mit)

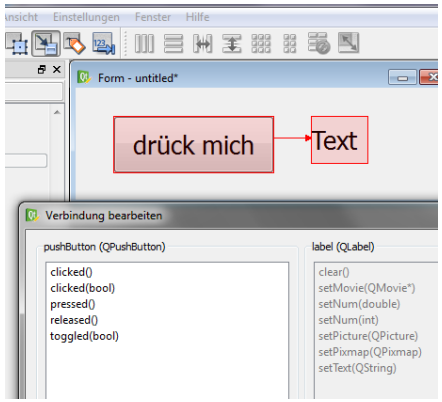
Texturing – Laden von Texturdaten mit Qt

```
QImage img(filename), tex;

if (img.isNull()) {
    error handling
    return;
}

tex = QGLWidget::convertToGLFormat(img); // ARGB to RGBA
glTexImage2D(… tex.width(), tex.height() … tex.bits());
```

Signal-Slot-Prinzip



- Steuerung der Kommunikation zwischen Objekten über Ereignisse
- direkt im Qt-Designer (eingeschränkt)
- mithilfe der Codezeile

```
connect( Sendername, SIGNAL(Funktionsname(Typenliste)),
         Empfängername, SLOT(Funktionsname(Typenliste)));
```

- Typenliste muss übereinstimmen
- SLOT wird nach „{*sichtbarkeit*} slots:“ als normale Funktion in der Empfängerklasse definiert

```
public slots:
    void doSomething(int value)));
```