Computergrafik SS 2010

Oliver Vornberger

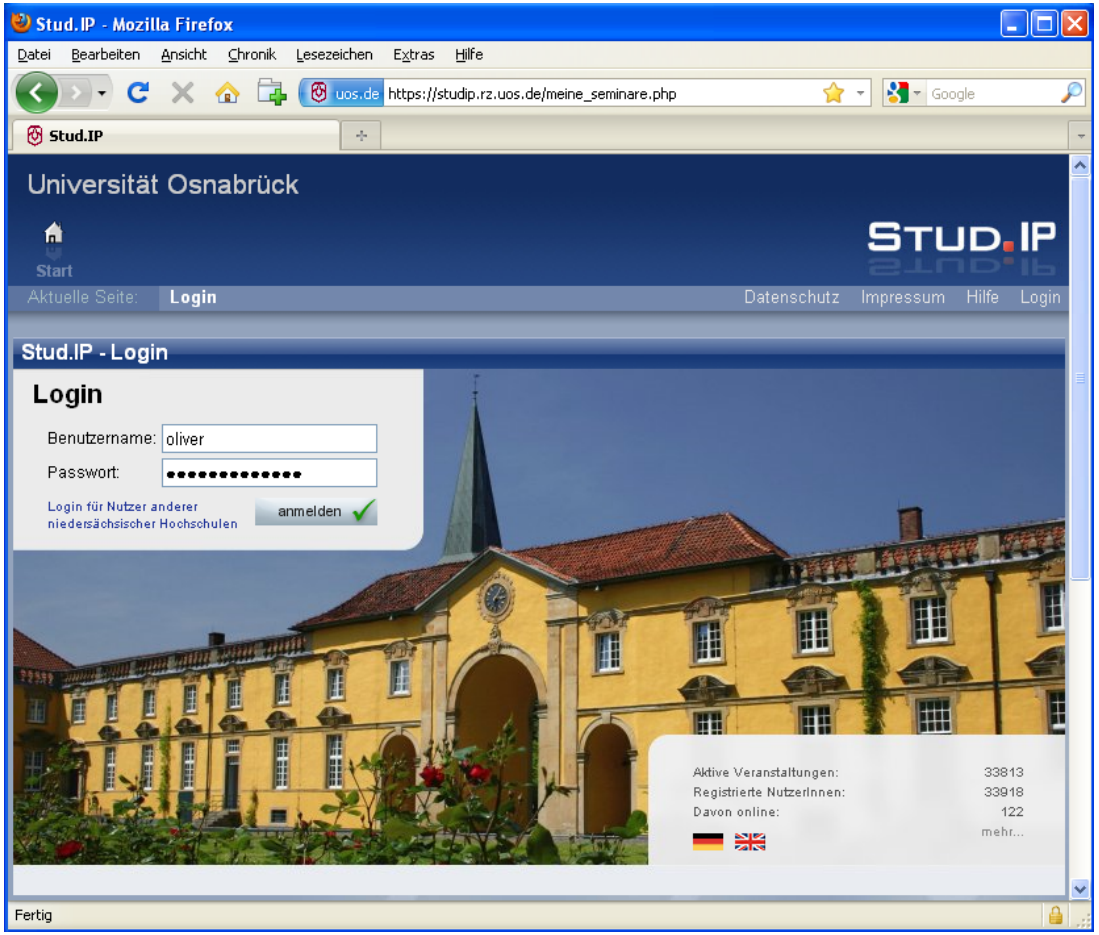
Institut für Informatik
Universität Osnabrück

Vorlesung vom 06.04.2010

Organisation

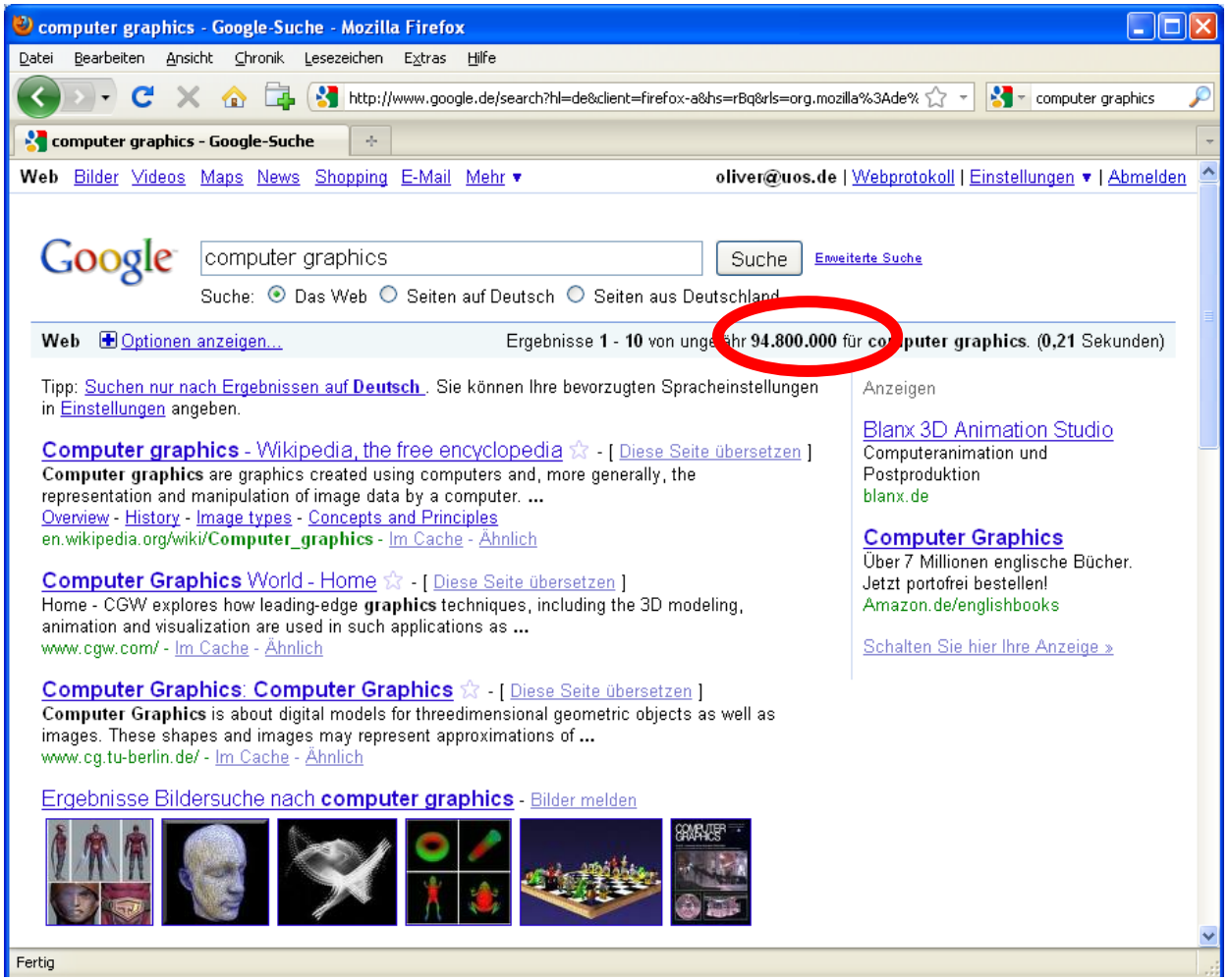
- Vorlesung: Mo+Di, 10:15 Uhr, 31/449a
- Übung: Do, 14:15 + 16:00 Uhr, 31/449a
- Übungsblatt am Dienstag
- Übung am Donnerstag
- Testate am Montag, Dienstag, Mittwoch
- Klausur am 05.07.10

stud.ip



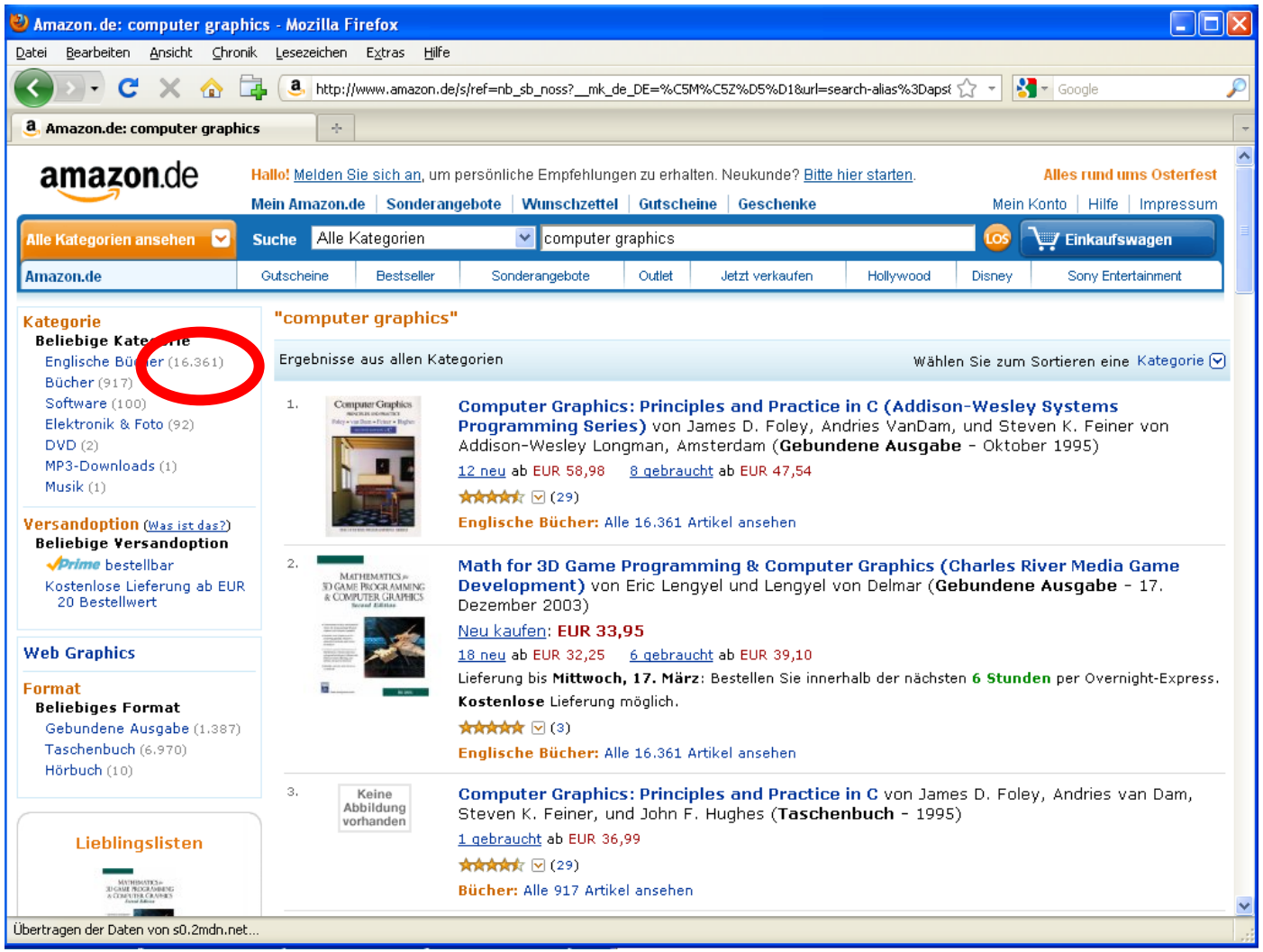
https://studip.rz.uos.de

Google



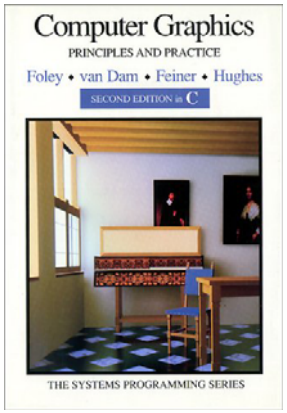
http://www.google.de/search?q=computer+graphics

Amazon



http://www.amazon.de

Literatur

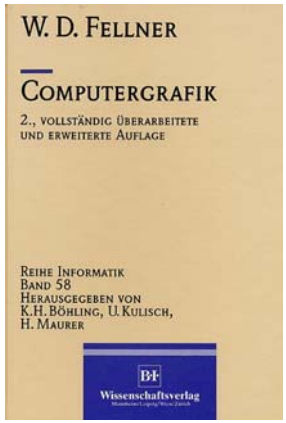


James Foley et al:

Computer Graphics Principles and Practice

2nd Edition

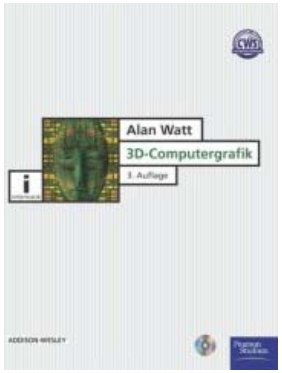
Addison Wesley 1995



Dieter Fellner:

Computergrafik

BI 1994

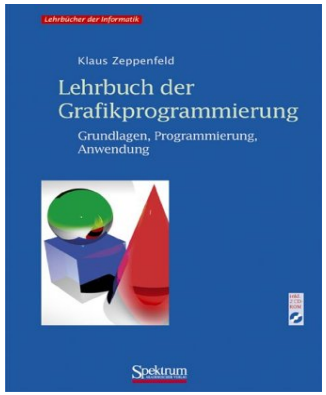


Alan Watt:

3D-Computergrafik

3. Auflage

Addison Wesley 2002



Klaus Zeppenfeld:

Lehrbuch der Grafik-programmierung

Spektrum 2004

Begleitmaterial

- Skript, gedruckt, 7.50 €
- Skript in HTML
- Skript in PDF
- Folien in PDF
- Videomitschnitt im virtPresenter-Format
- Videopodcast im mp4-Format
- Audiomitschnitt im mp3-Format

http://www-lehre.inf.uos.de/~cg/2010

Classroomquiz

Motivation

- Bild sagt mehr als 1000 Worte
- Auge erfasst 40.000.000 Bit/sec
- Lesegeschwindigkeit
  = 10 Worte à 5 Zeichen/sec
  = 10 · 5 · 8 = 400 Bit/sec
- $\Rightarrow$ Faktor 100.000

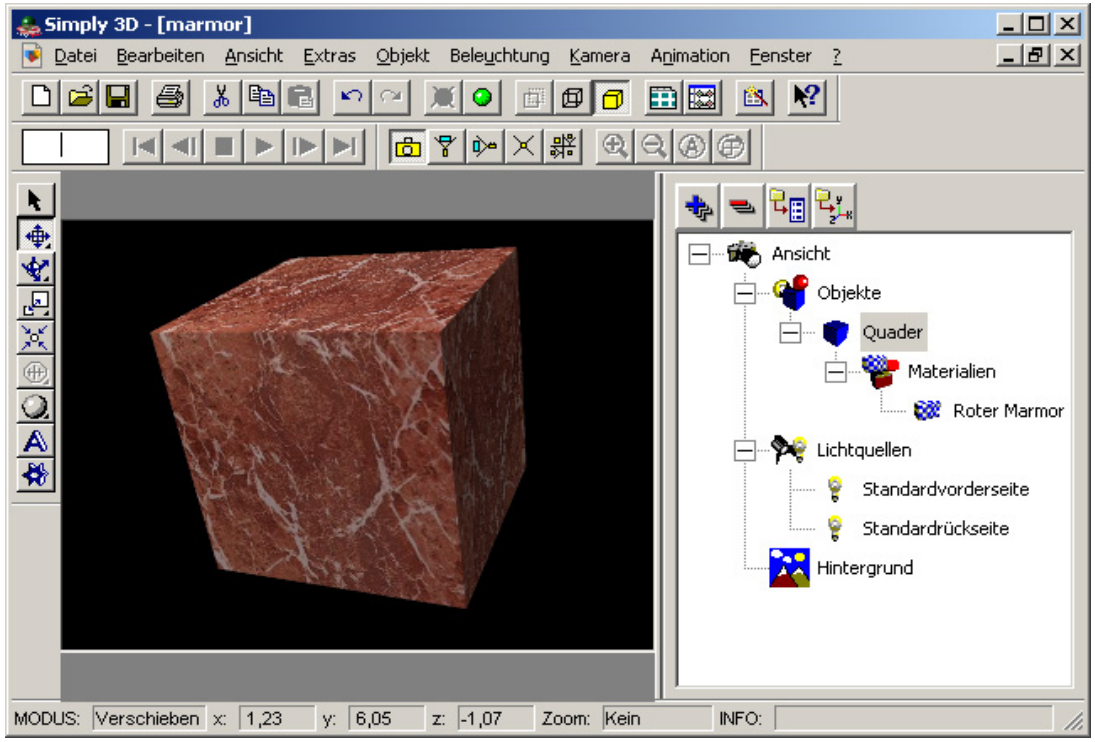
Grafische Datenverarbeitung

- Bildverarbeitung
  - Licht, Radar, Röntgen, Ultraschall, …
  - Vereinfachung, Verbesserung
- Mustererkennung
  - Analyse von Rasterdaten
  - Optical Character Recognition (OCR)
- Generative Computergrafik
  - Eingabe der Repräsentation
  - Ausgabe der Darstellung

Anwendungen

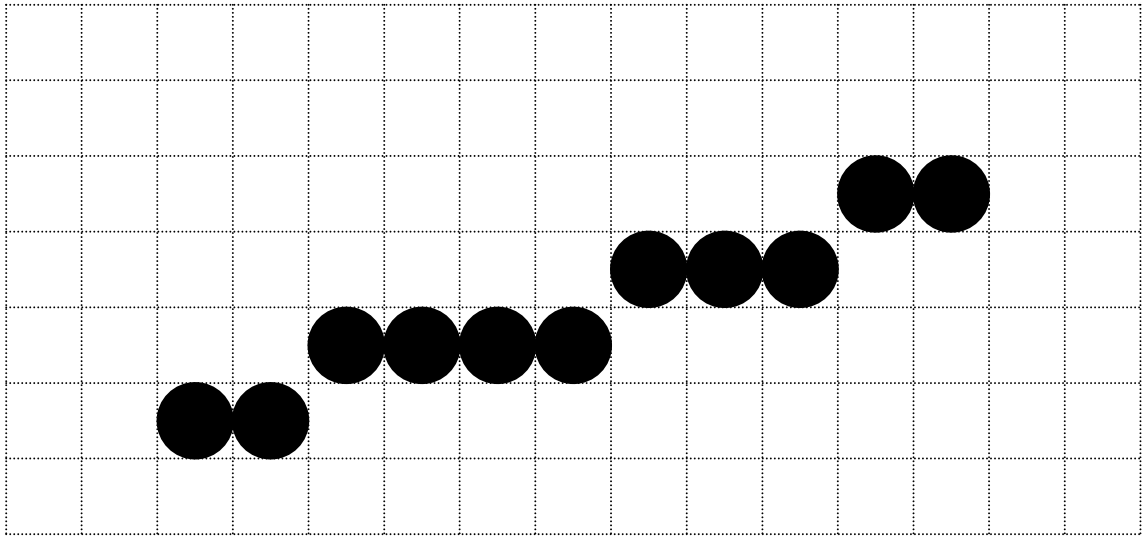
- Business-Grafik
- Grafische Benutzeroberflächen
- Kartografie
- CAD (Haus, Auto,...)
- Visualisierung (Molekül, Strömung, Scan)
- Simulation (Fahrzeug, Flugzeug,...)
- Virtual Reality (Computerspiele,...)

Modellieren, Projizieren, Rendern

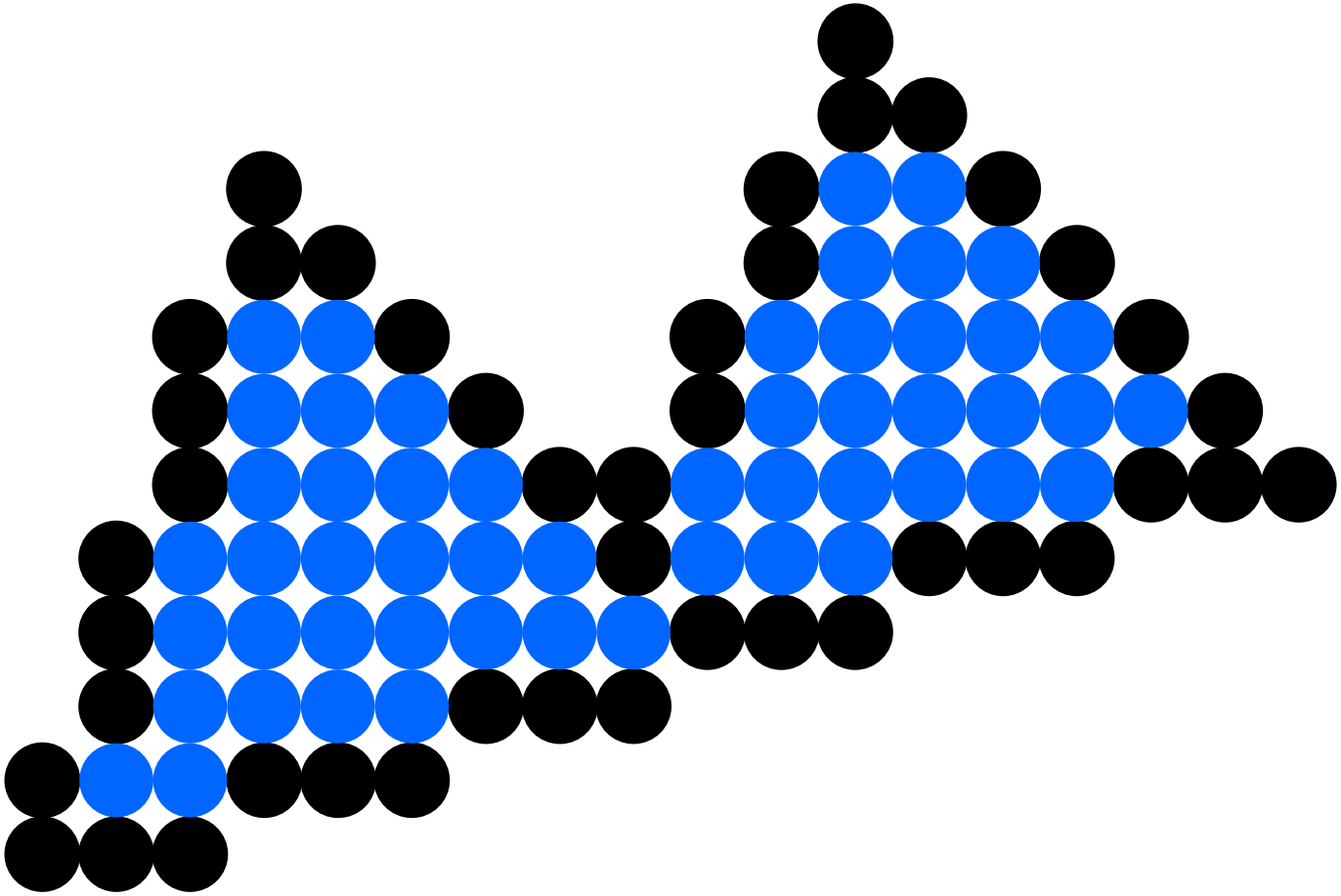


Micrografx Simply 3D

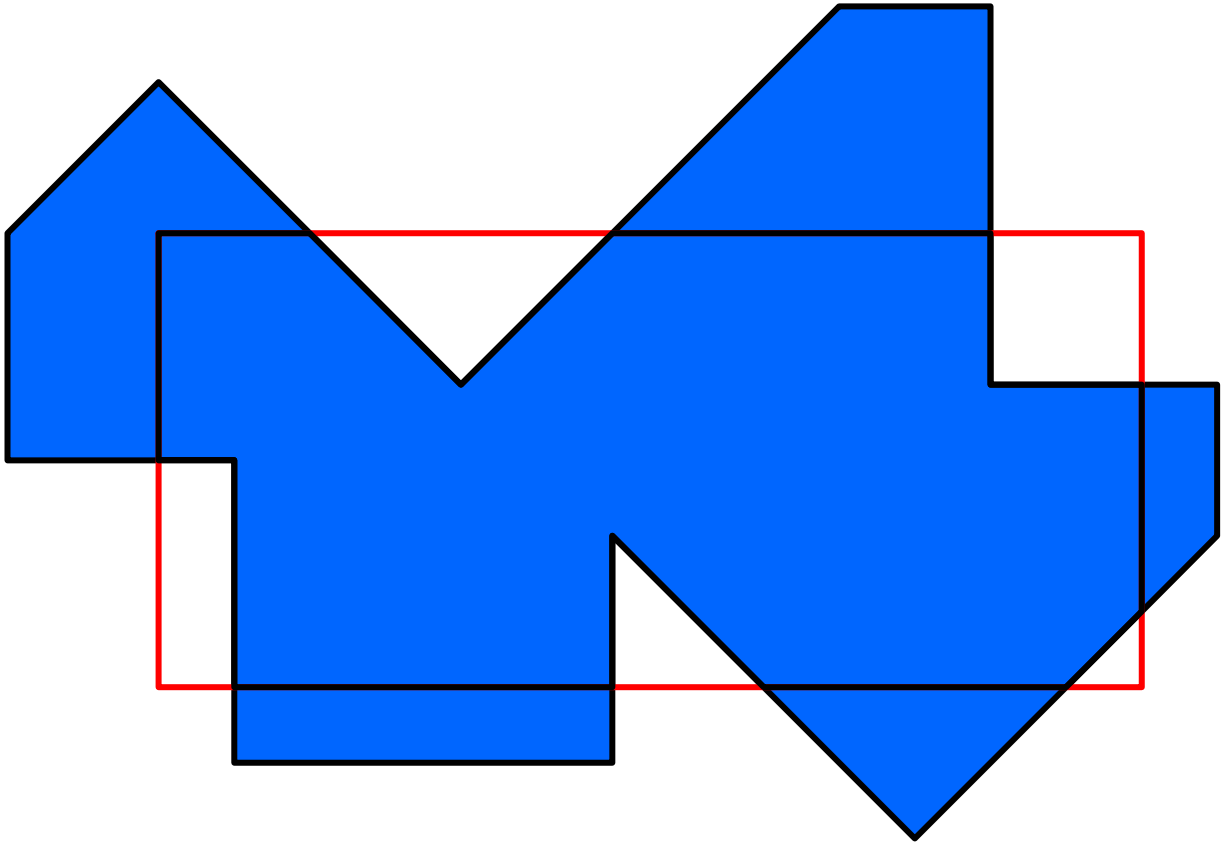
2D-Grundlagen



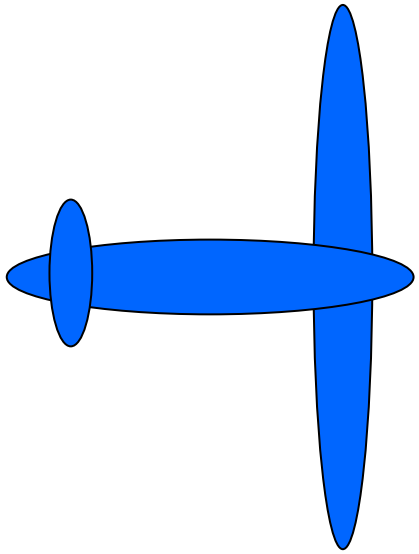
2D-Füllen



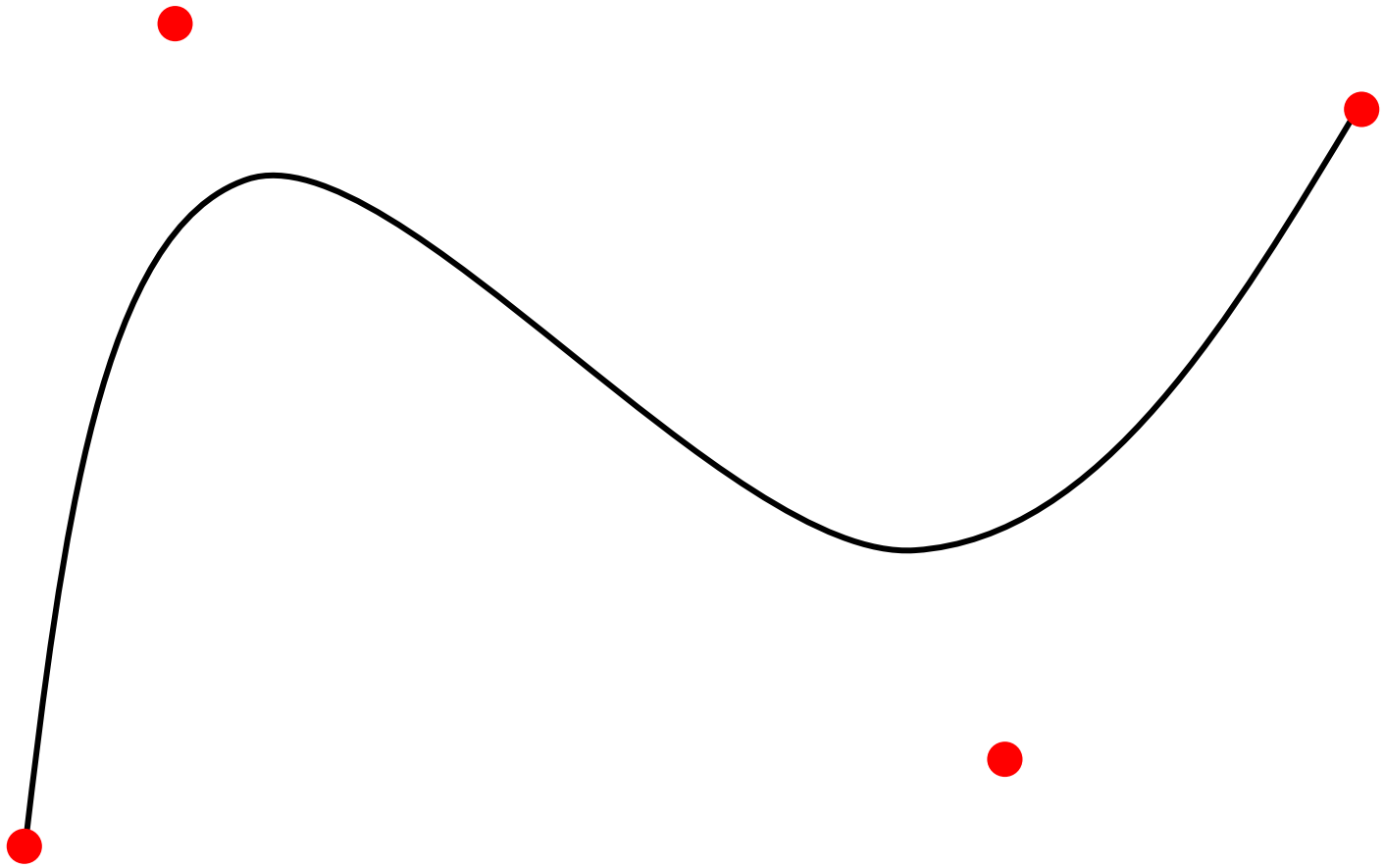
2D-Clipping



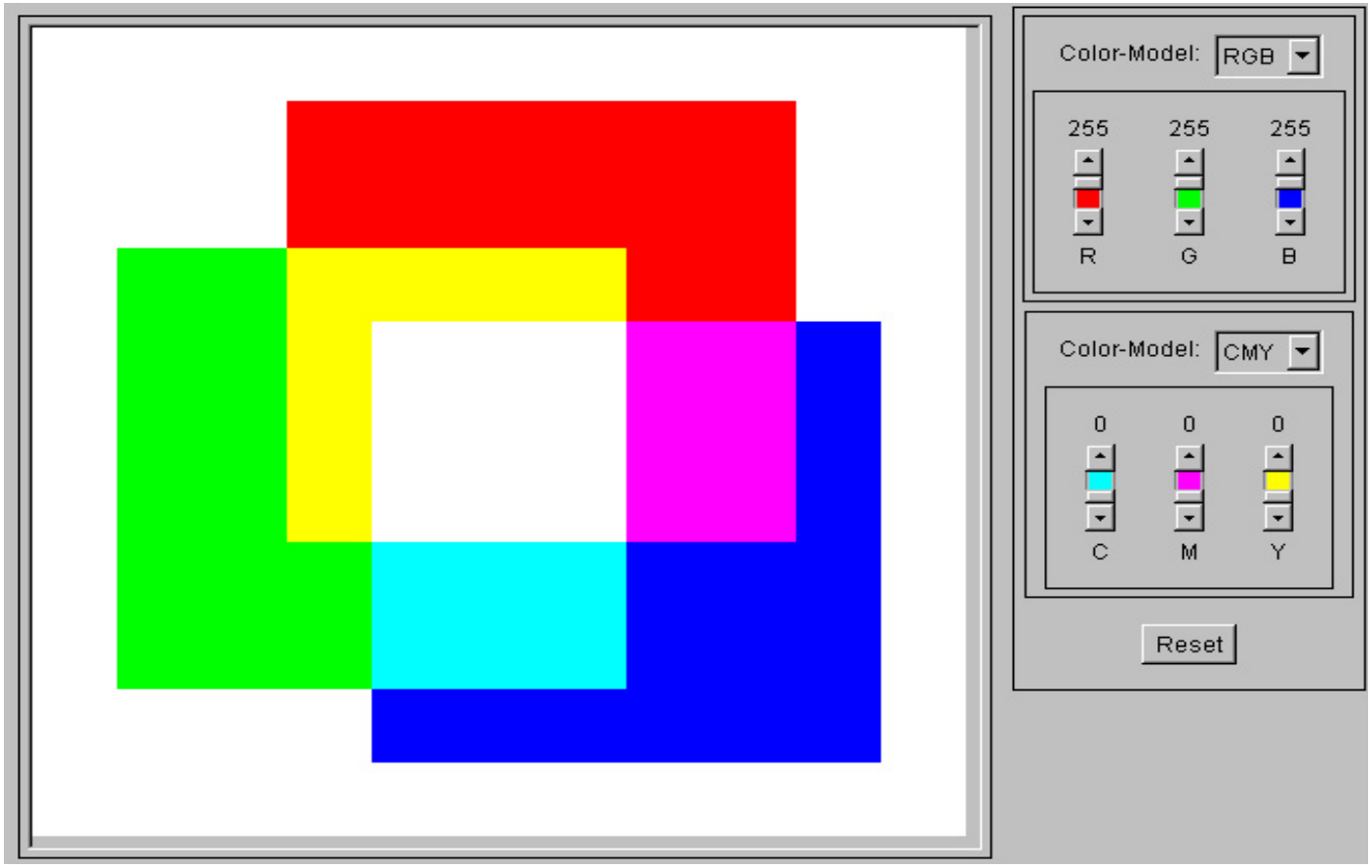
Transformation



Kurven

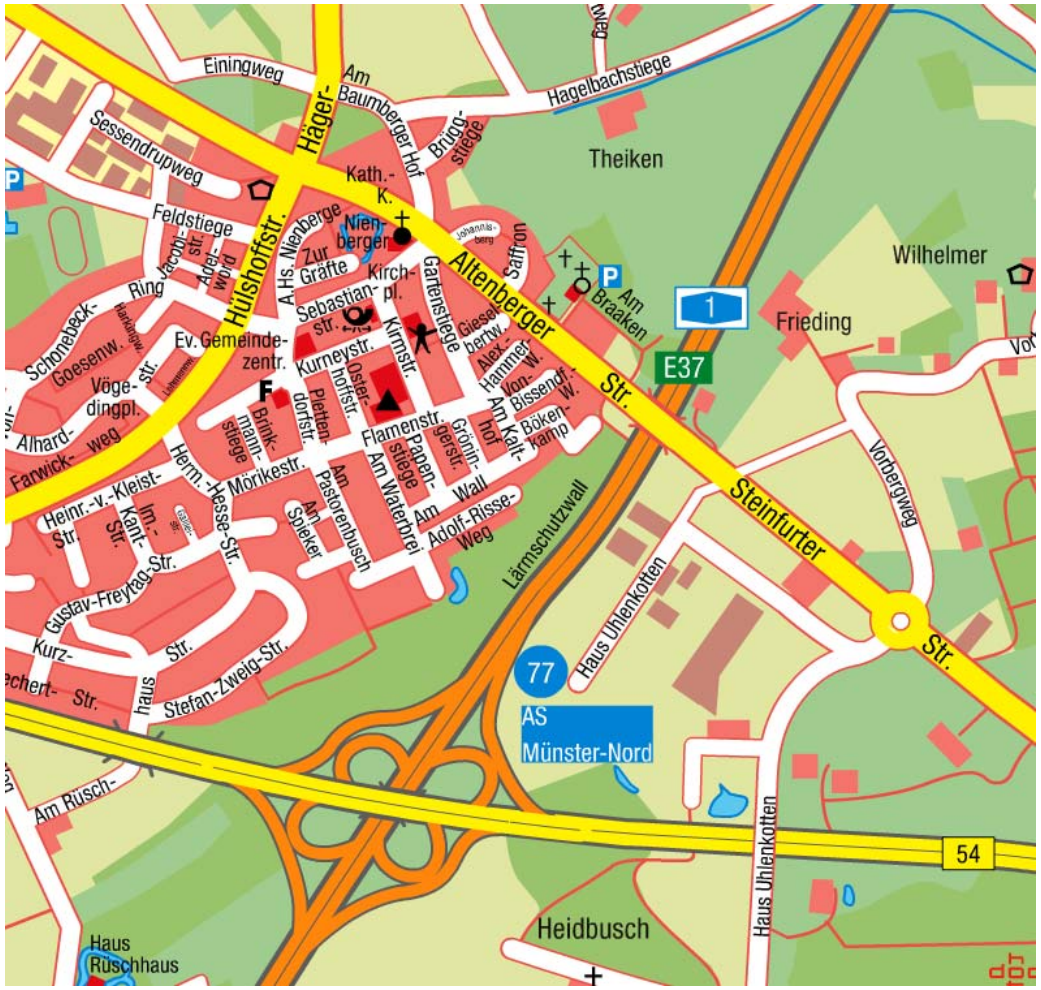


Farbe

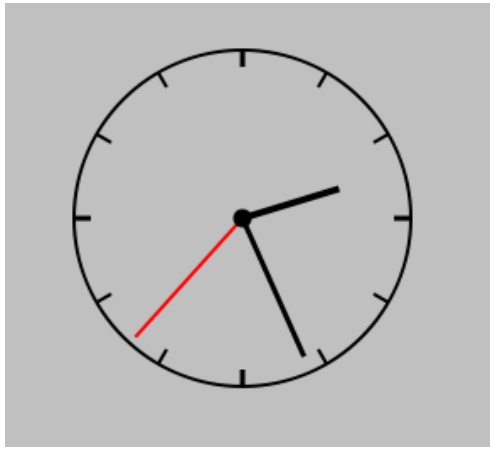


Pixeldateiformate

Macromedia Flash



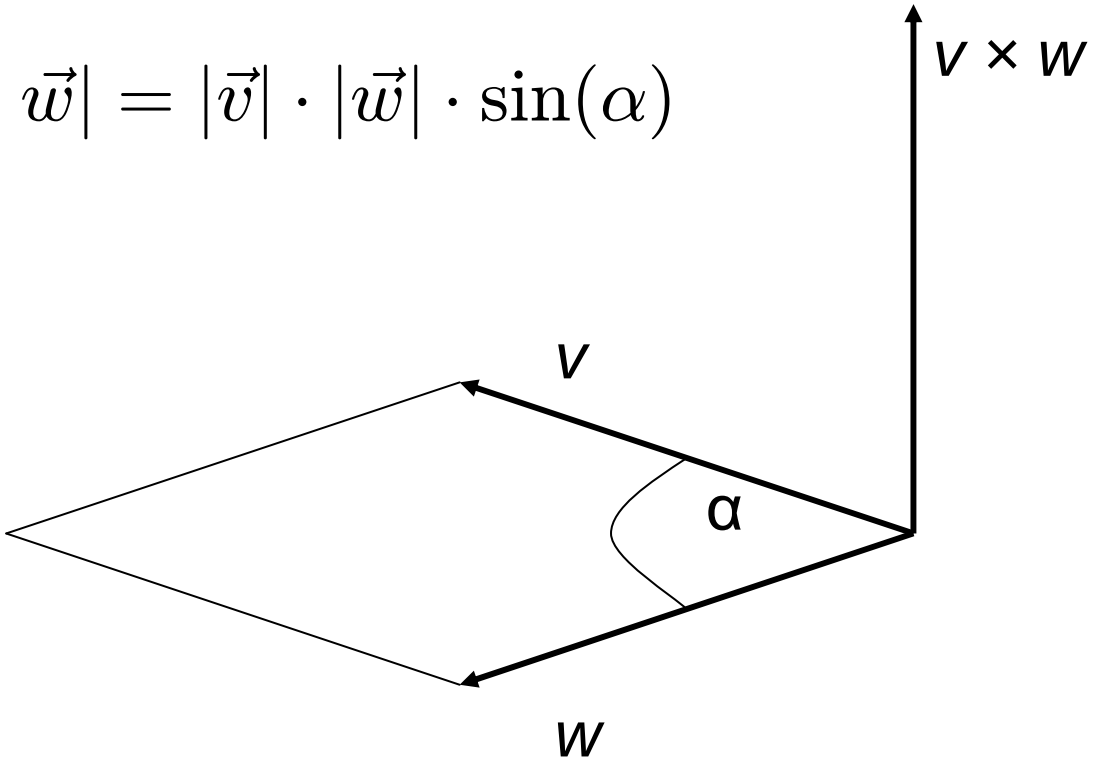
SVG



Fraktale

3D-Grundlagen

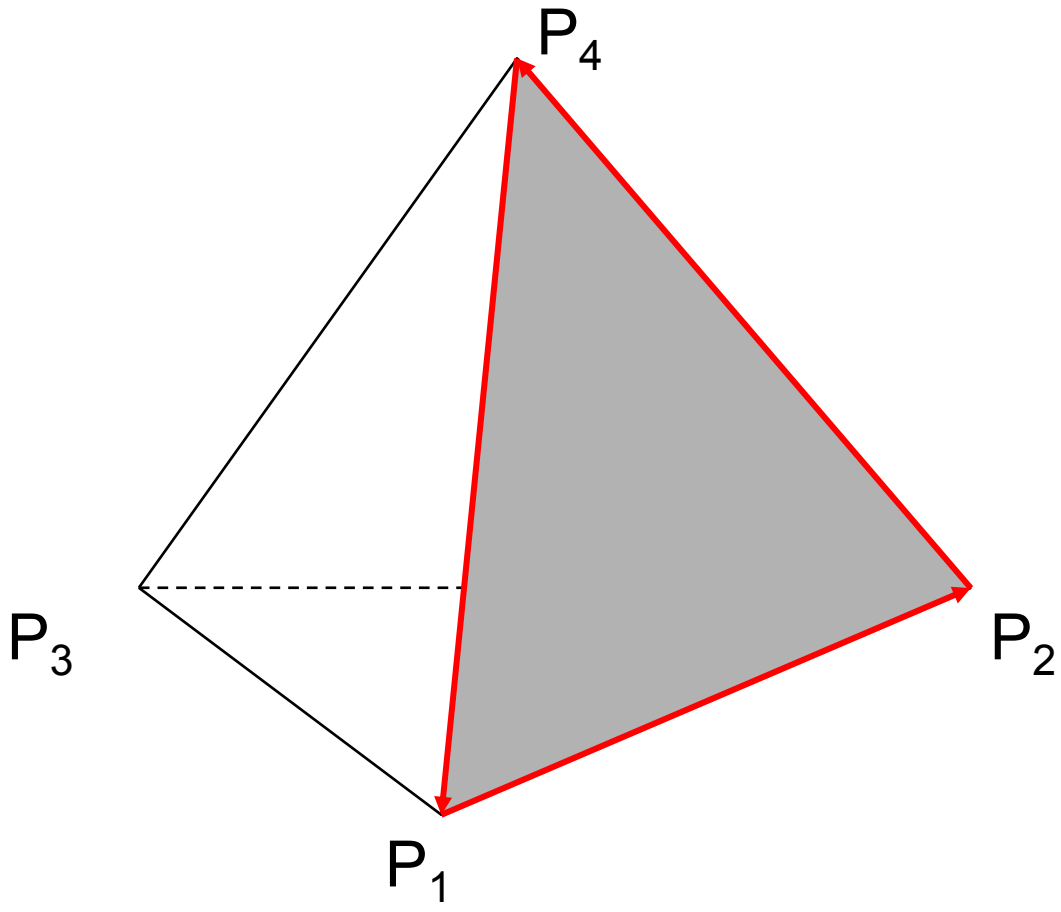
$$|\vec{v} \times \vec{w}| = |\vec{v}| \cdot |\vec{w}| \cdot \sin(\alpha)$$



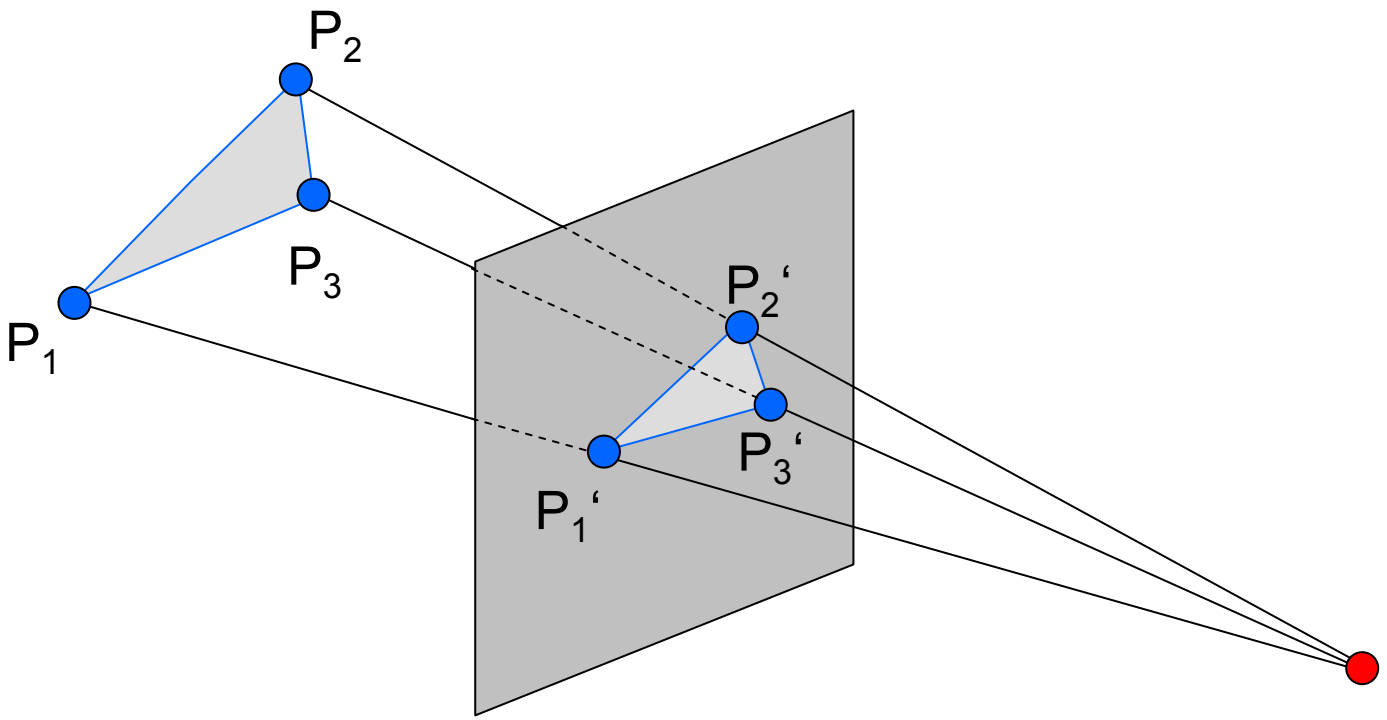
3D-Transformationen

$$R_y(\delta) = \begin{pmatrix} \cos(\delta) & 0 & \sin(\delta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\delta) & 0 & \cos(\delta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

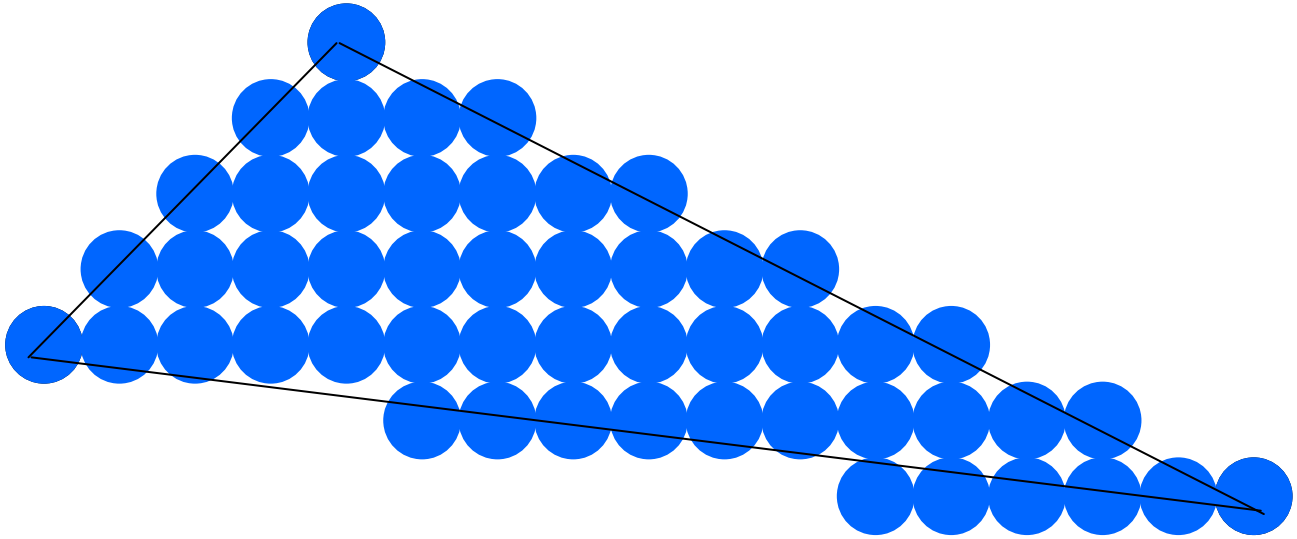
3D-Repräsentation



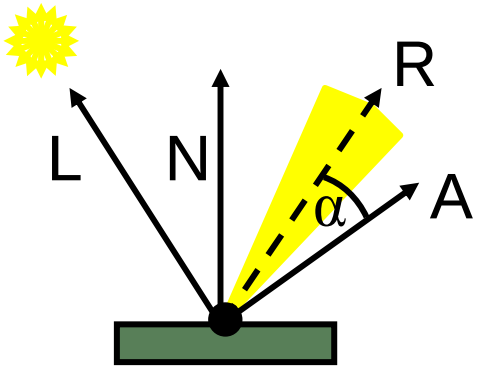
Projektion

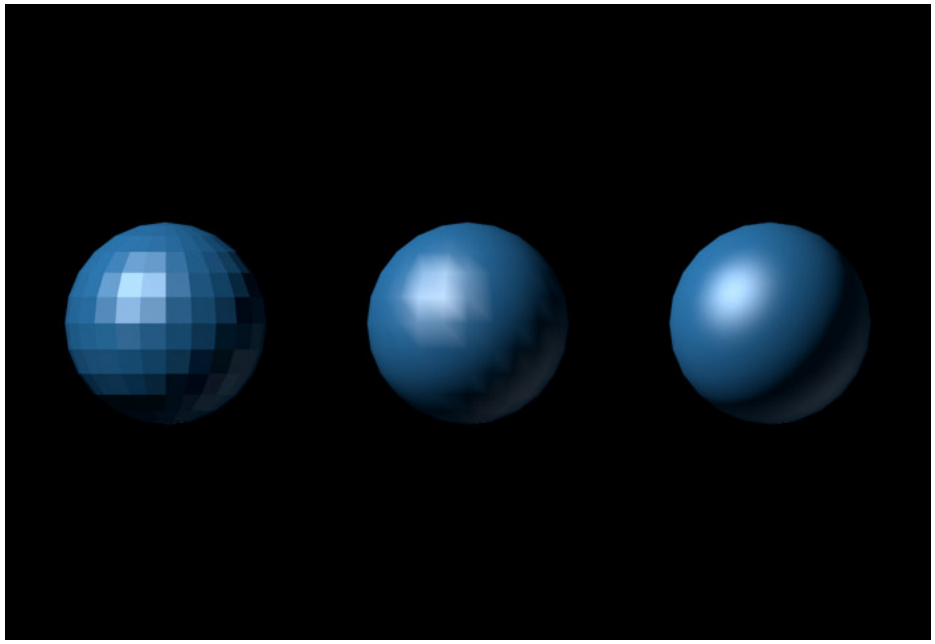


Rendern

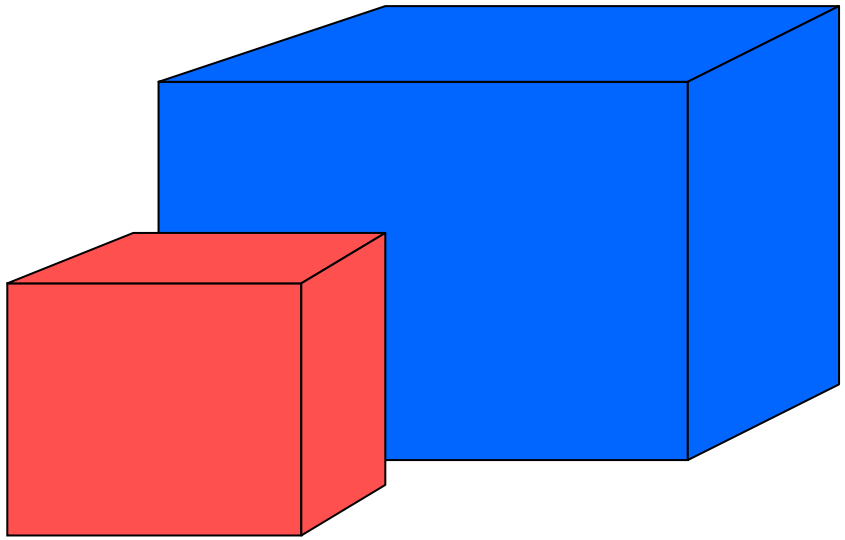


Beleuchtung

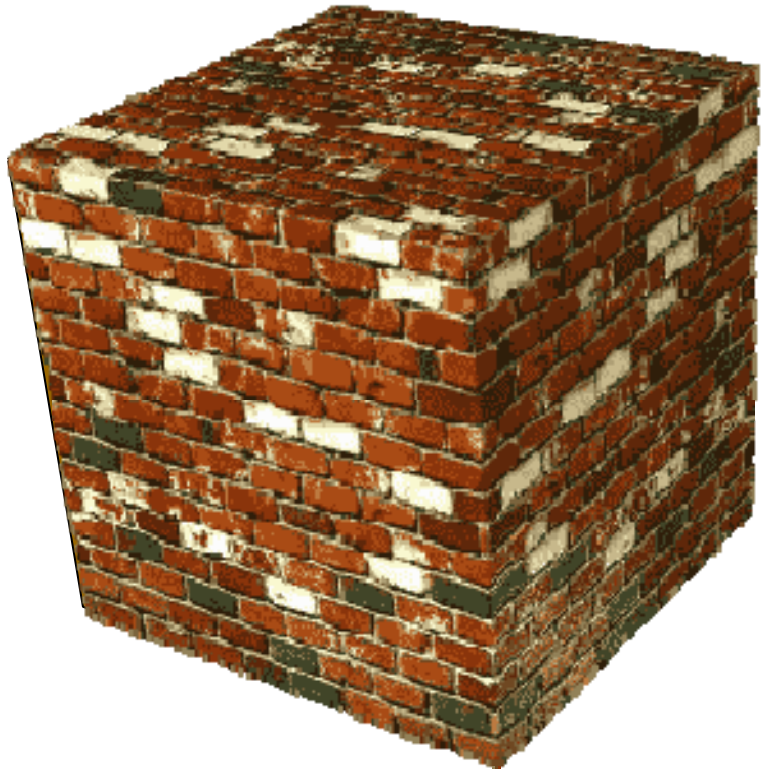




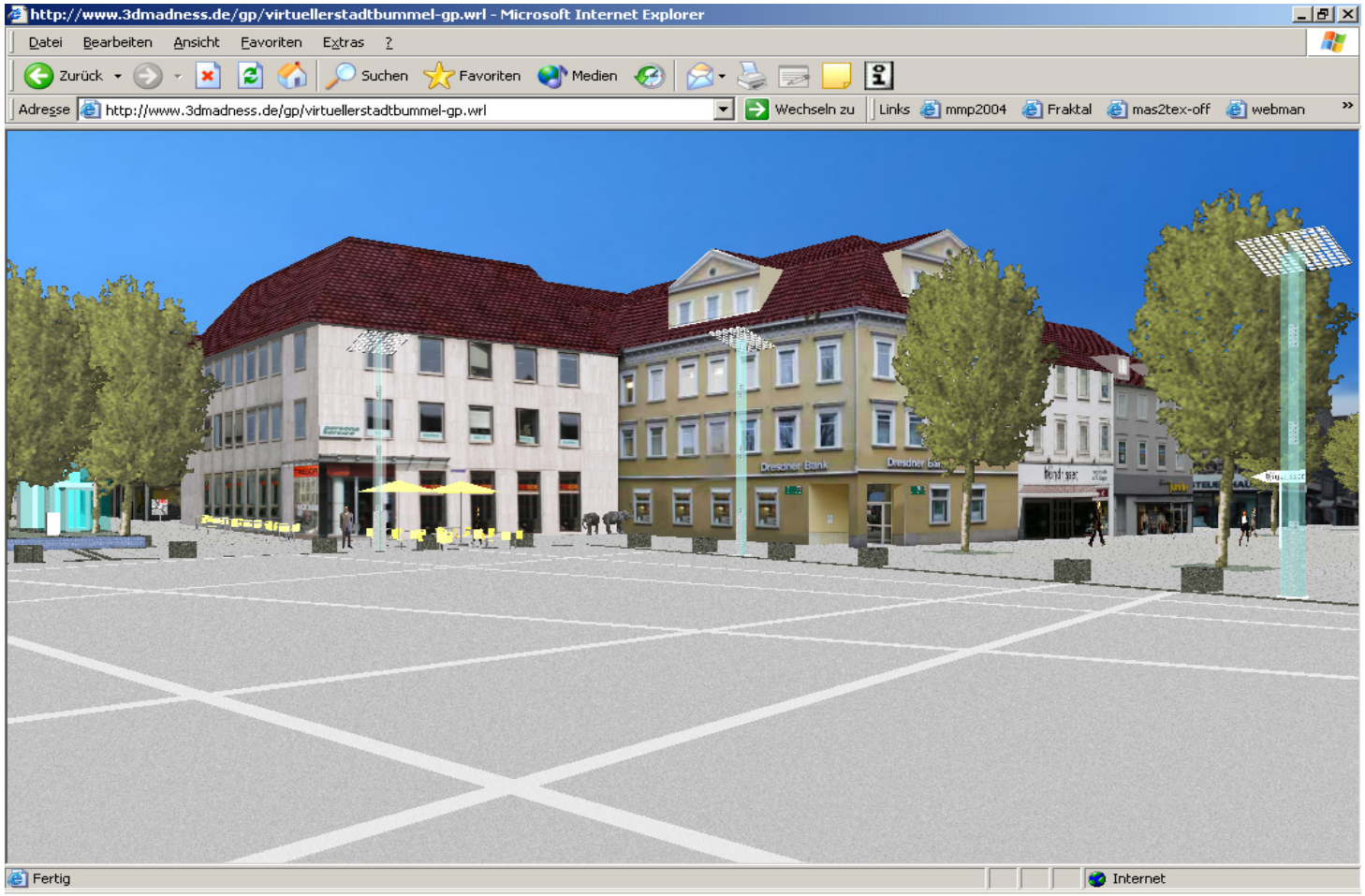
Culling



Texturing

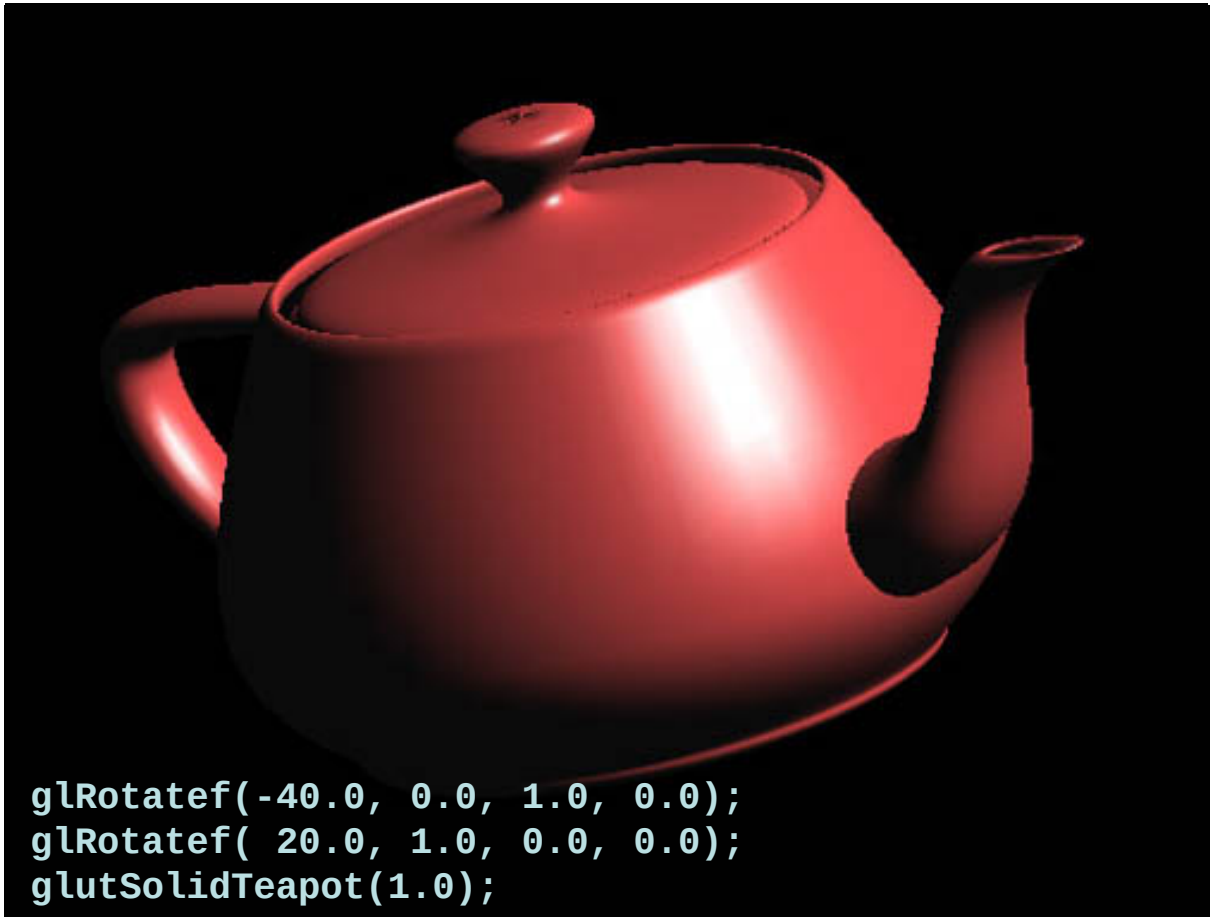


VRML



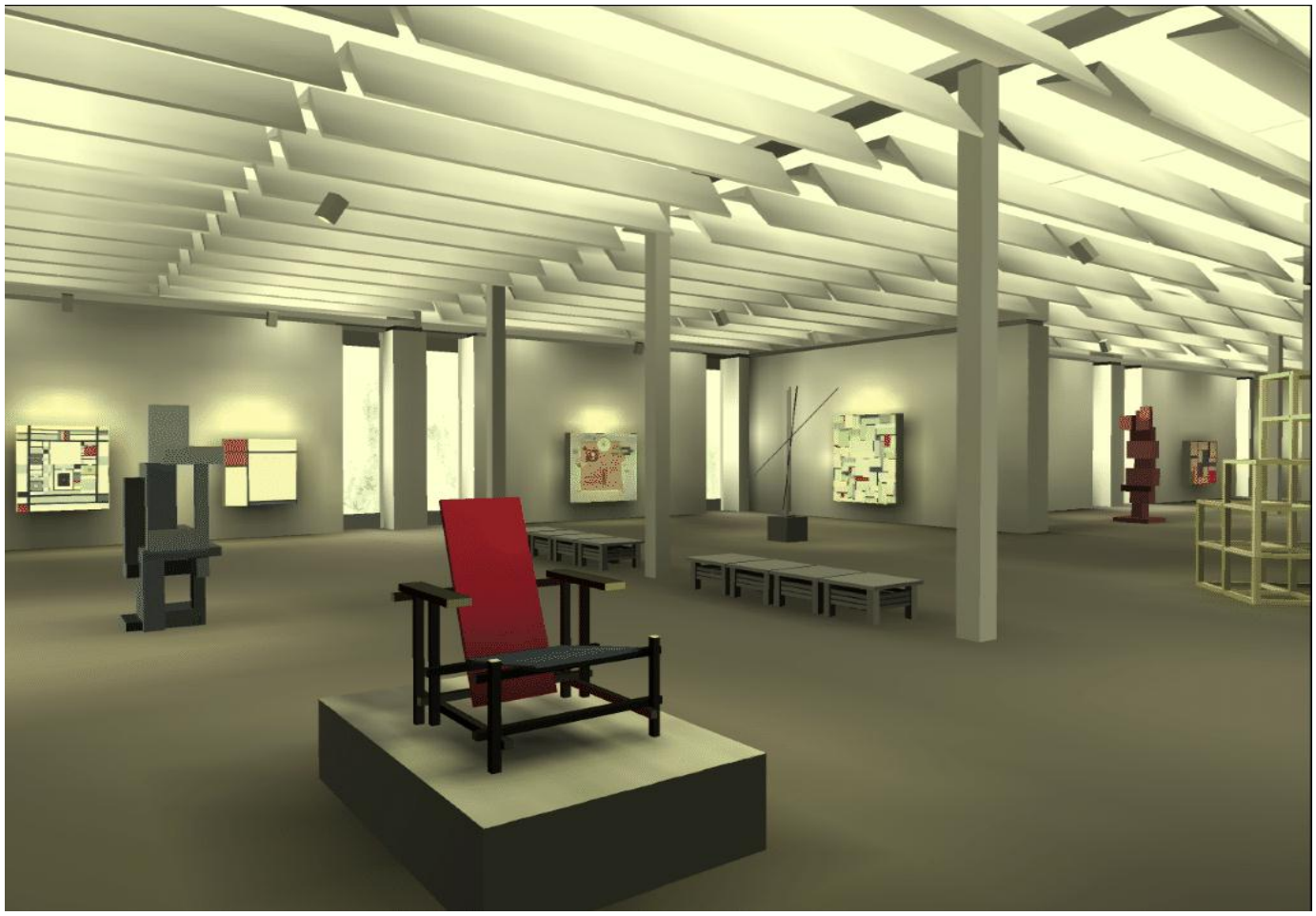
http://www-lehre.inf.uos.de/gp/

OpenGl

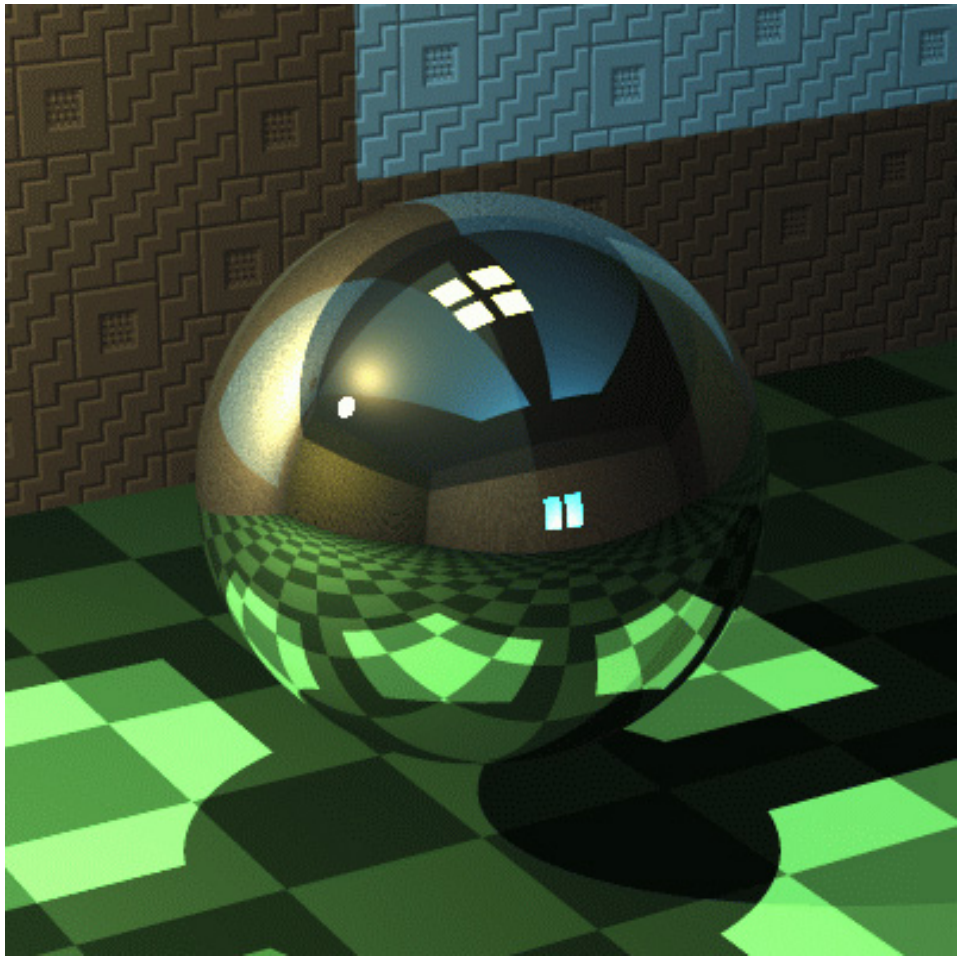


```
glRotatef(-40.0, 0.0, 1.0, 0.0);
glRotatef( 20.0, 1.0, 0.0, 0.0);
glutSolidTeapot(1.0);
```

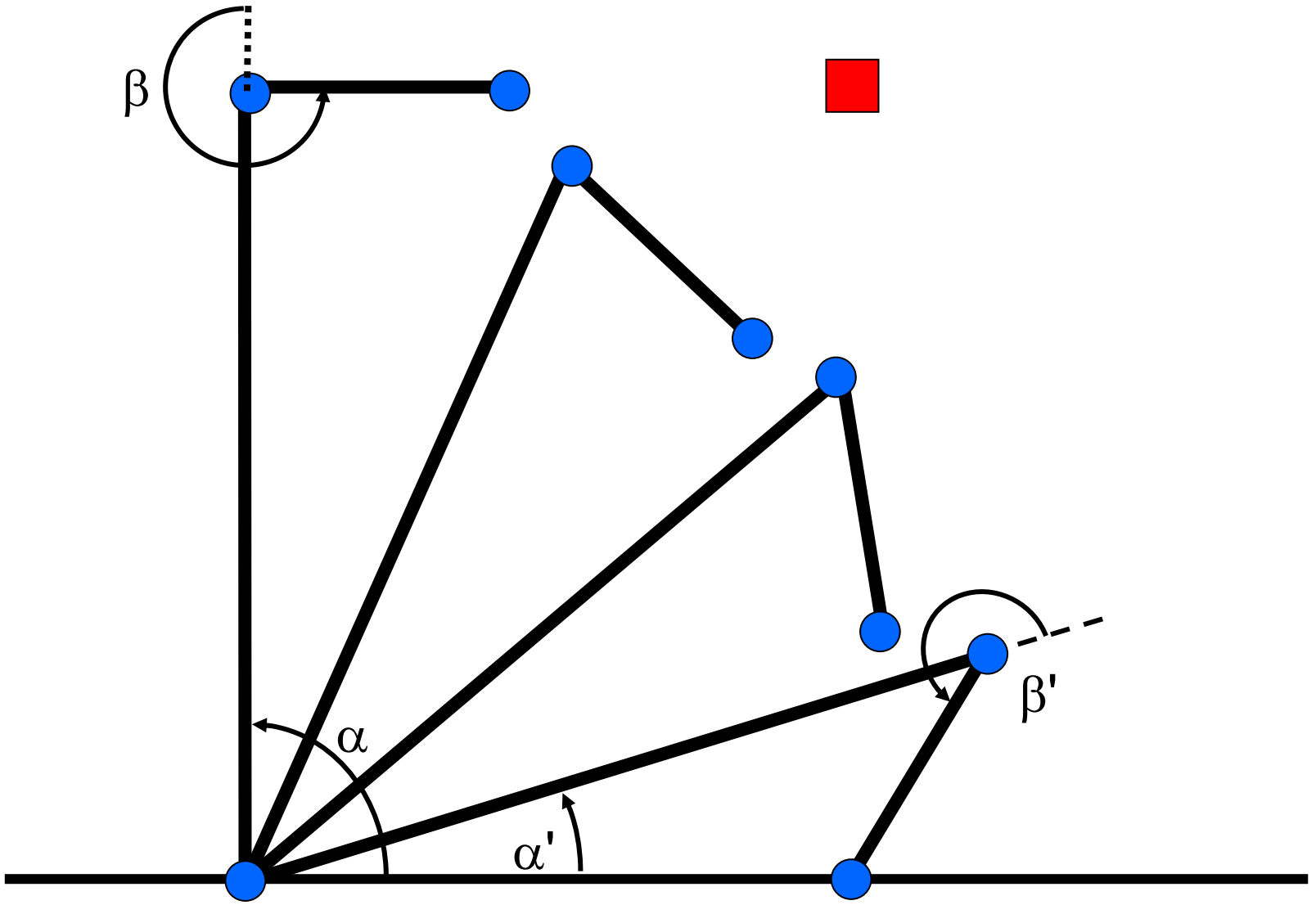
Radiosity



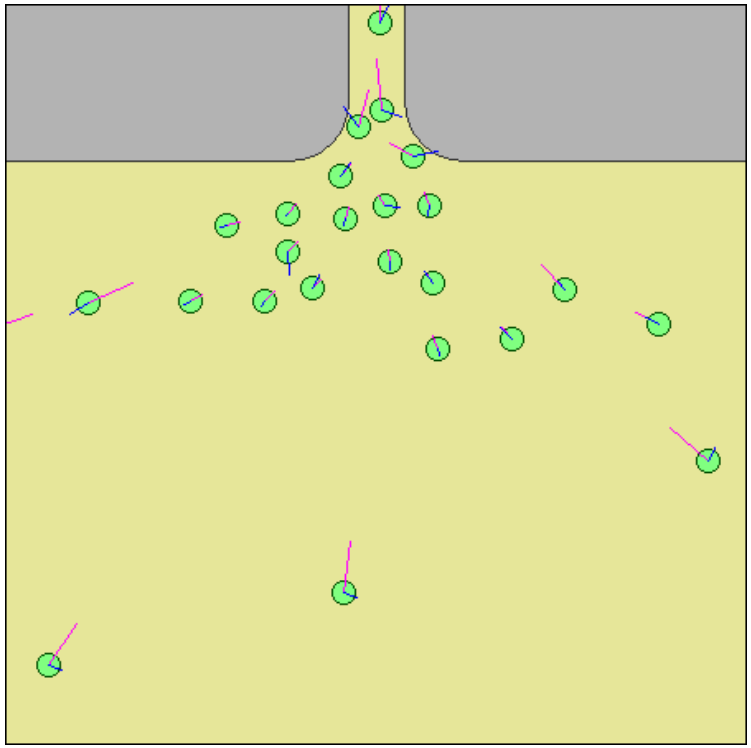
Ray Tracing



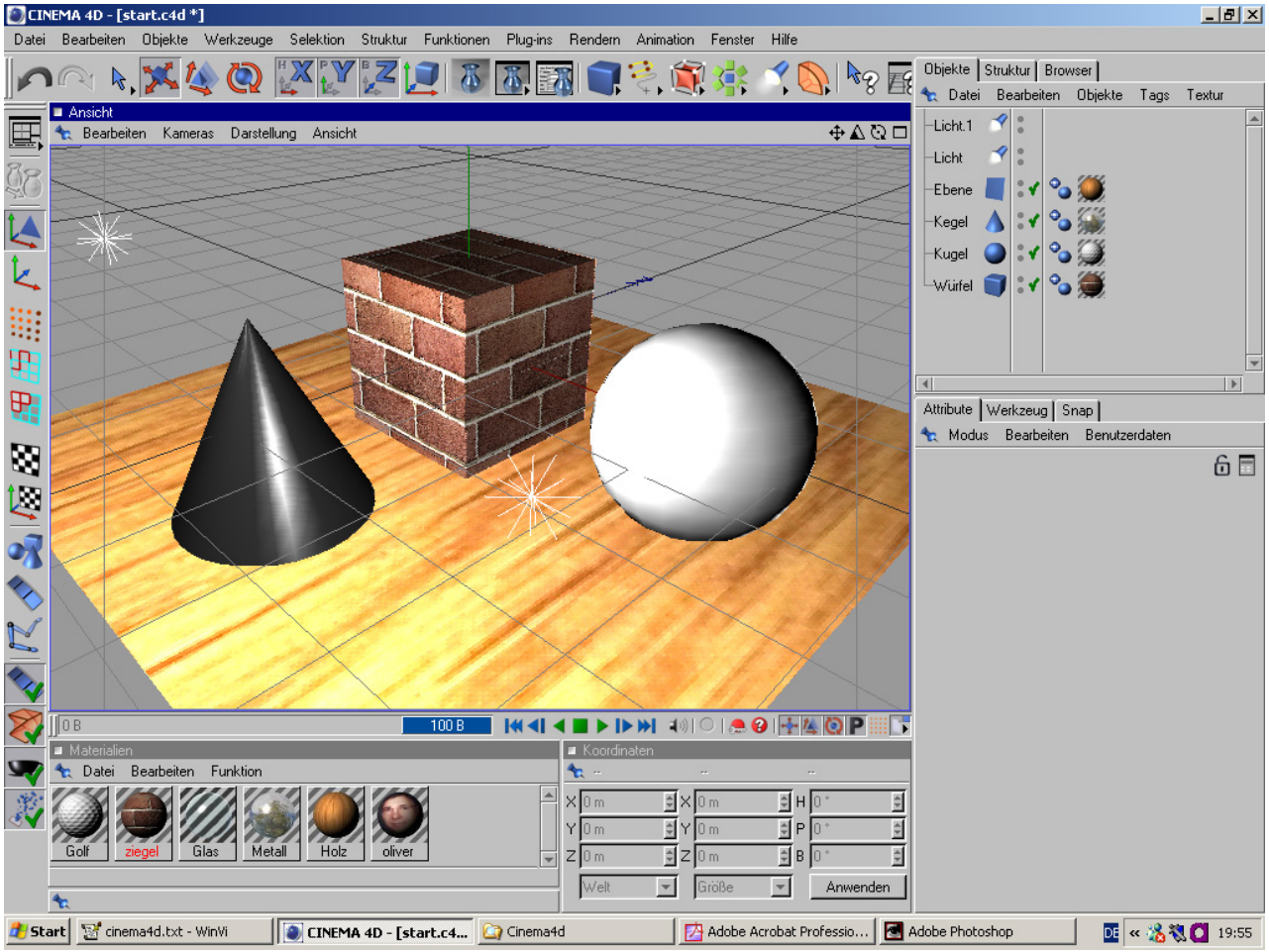
Inverse Kinematik



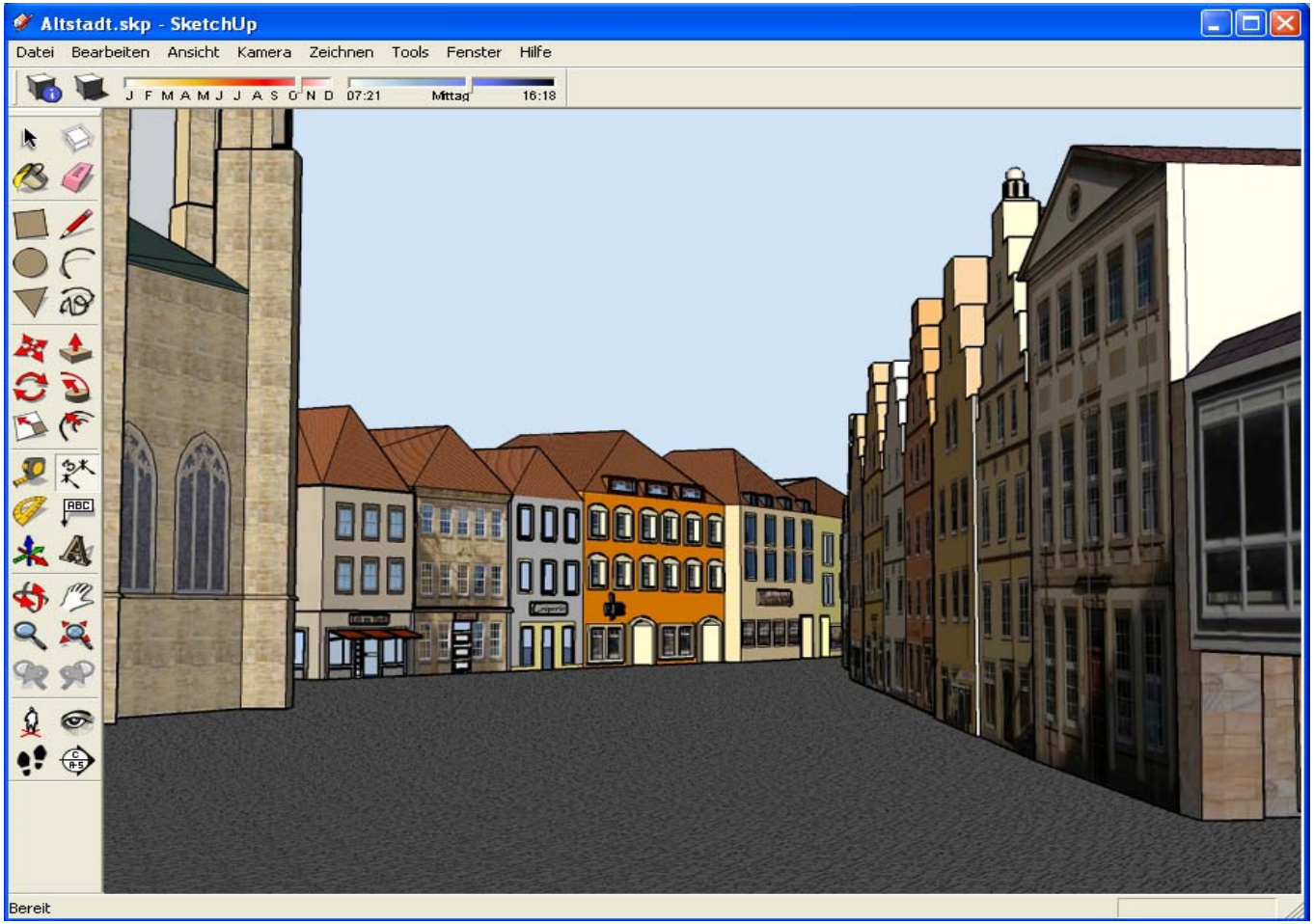
Partikelsysteme



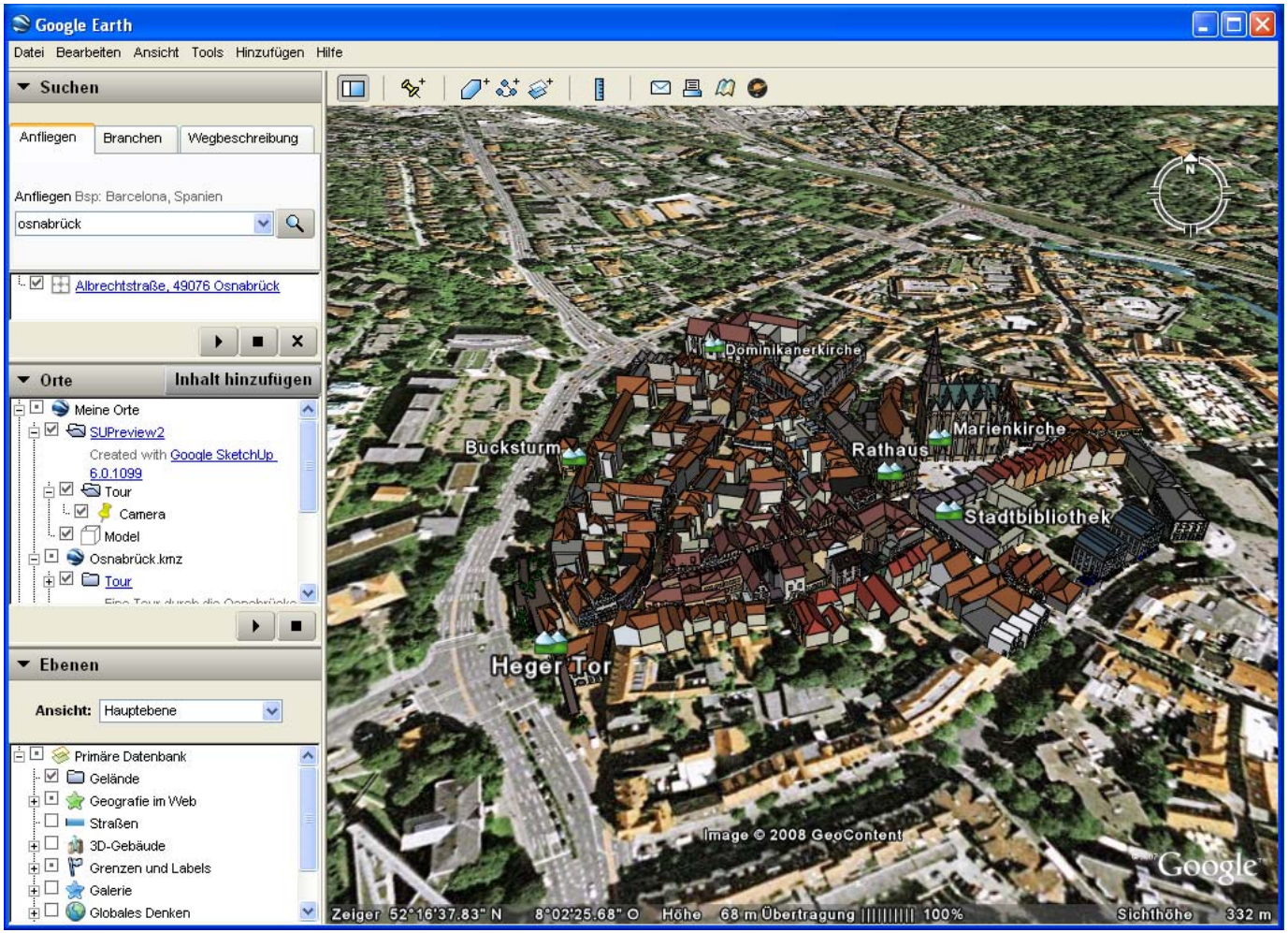
Maxon Cinema4D



Google SketchUp

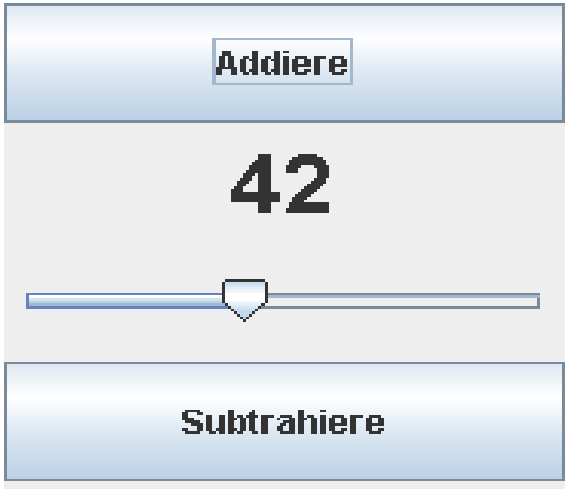


Google Earth



Kapitel 2:

Grafische Benutzeroberflächen



GUI-Programmierung

- Windowmanager
- AWT (Abstract Window Toolkit)
  - reicht Kommandos weiter an Betriebssystem
  - plattformabhängig
- Swing
  - pure Java
  - einheitliches Look & Feel

GUI-Komponenten

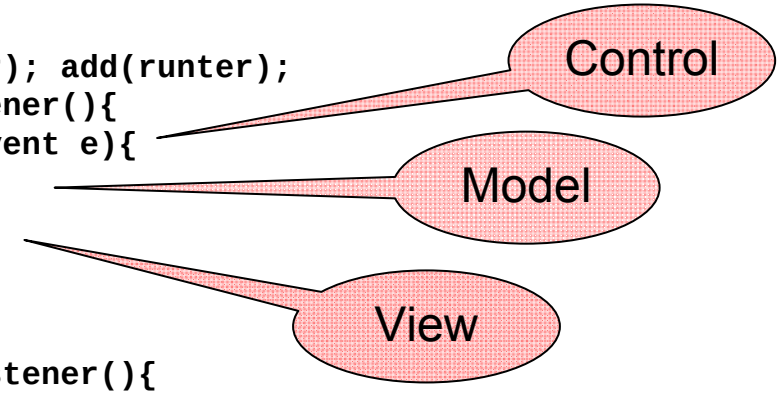
- JFrame
- GridLayout
- JButton
- JLabel
- JSlider
- ActionListener
- actionPerformed

RaufRunterApplikation.java

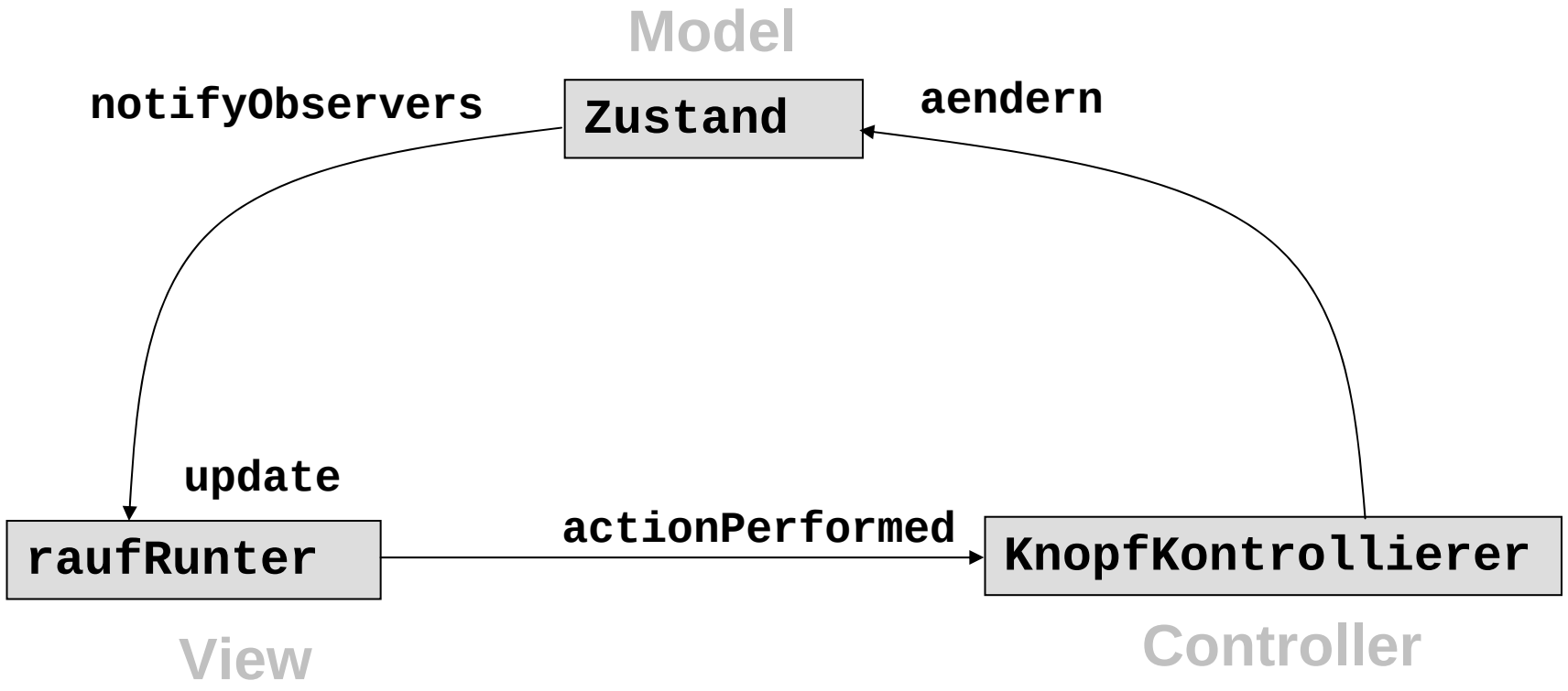
```
import java.awt.*; import java.awt.event.*; import javax.swing.*;

public class RaufRunterApplikation extends JFrame {
  private int     zaehler  = 42;
  private JButton rauf     = new JButton("Addiere");
  private JLabel  ergebnis = new JLabel("42 ",JLabel.CENTER);
  private JSlider schieber = new JSlider(0, 100, zaehler);
  private JButton runter   = new JButton("Subtrahiere");

  public RaufRunterApplikation() {
    setLayout(new GridLayout(0,1));
    add(rauf); add(ergebnis); add(schieber); add(runter);
    rauf.addActionListener(new ActionListener(){
      public void actionPerformed(ActionEvent e){
        zaehler++;
        ergebnis.setText(zaehler + " ");
        schieber.setValue(zaehler);
      }
    });
    runter.addActionListener(new ActionListener(){
      public void actionPerformed(ActionEvent e){
        zaehler--;
        ergebnis.setText(zaehler + " ");
        schieber.setValue(zaehler);
      }
    });
    pack(); setVisible(true);
  }
  public static void main (String [] args) {
    new RaufRunterApplikation();
  }
}
```



Model-View-Controller



Zustand.java

```
import java.util.Observer;
import java.util.Observable;

public class Zustand extends Observable{

  private int zaehler;

  public Zustand(int zaehler){
    this.zaehler=zaehler;
  }

  int get(){return zaehler;}

  void aendern(int delta){
    zaehler = zaehler + delta;
    setChanged();
    notifyObservers();
  }
}
```

KnopfKontrollierer.java

```
import java.awt.*;
import java.awt.event.*;

public class KnopfKontrollierer implements ActionListener {

  private Zustand z;
  private int delta;

  public KnopfKontrollierer(Zustand z, int delta) {
    this.z       = z;
    this.delta   = delta;
  }

  public void actionPerformed(ActionEvent e) {
    z.aendern(delta);
  }
}
```

RaufRunter.java, Teil 1

```
import java.util.*;
import java.awt.*;
import javax.swing.*;

public class RaufRunter extends JPanel implements Observer {

  private JButton rauf;
  private JButton runter;
  private Zustand z;
  private JLabel  ergebnis;
  private JSlider schieber;
  private Font    font;


 public void update(Observable z, Object dummy){
   ergebnis.setText(((Zustand)z).get() + " ");
   schieber.setValue(((Zustand)z).get());
}
```

RaufRunter.java, Teil 2

```
public RaufRunter() {
    setLayout(new GridLayout(0,1));
    rauf   = new JButton("Addiere");
    runter = new JButton("Subtrahiere");
    schieber = new JSlider(0,100,42);
    ergebnis = new JLabel("42",JLabel.CENTER);
    font = new Font("SansSerif",Font.BOLD,30);
    ergebnis.setFont(font);
    add(rauf);
    add(ergebnis);
    add(schieber);
    add(runter);
    z = new Zustand(42);
    z.addObserver(this);
    KnopfKontrollierer raufK;
    raufK = new KnopfKontrollierer(z,+1);
    rauf.addActionListener(raufK);
    KnopfKontrollierer runterK;
    runterK = new KnopfKontrollierer(z,-1);
    runter.addActionListener(runterK);
  }
}
```

RaufRunterApp.java

```
import java.awt.BorderLayout;
import javax.swing.JFrame;

public class RaufRunterApp {

  public static void main(String args[]) {
    JFrame rahmen = new JFrame("RaufRunter-Applikation");
    rahmen.add(new RaufRunter(),BorderLayout.CENTER);
    rahmen.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    rahmen.pack();
    rahmen.setVisible(true);
  }

}
```

RaufRunterApplet.java

```
import java.awt.BorderLayout;
import javax.swing.JApplet;

public class RaufRunterApplet extends JApplet {

  public void init() {
    add(new RaufRunter(), BorderLayout.CENTER);
  }
}
```

RaufRunter.html

```
<HTML>
  <HEAD>
    <TITLE>RaufRunter-Applet</TITLE>
  </HEAD>
  <BODY>
    <CENTER>
    <H1>RaufRunter-Applet</H1>
    <APPLET
       width   = 200
       height  = 150
       code    = "RaufRunterApplet.class"
       archive = "raufRunter.jar">
    </APPLET>
    </CENTER>
  </BODY>
</HTML>
```

Dateisystem