

Computergrafik SS 2010

Oliver Vornberger

Kapitel 24:
Animation

Animation

Key Frame Animation

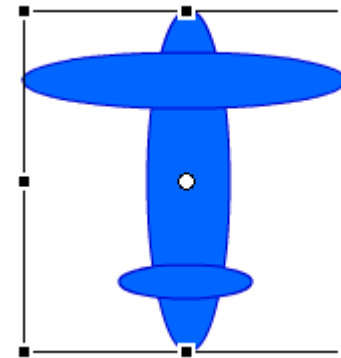
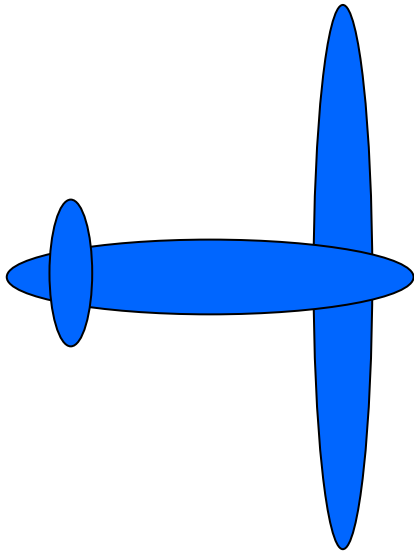
Forward Kinematics

Inverse Kinematics

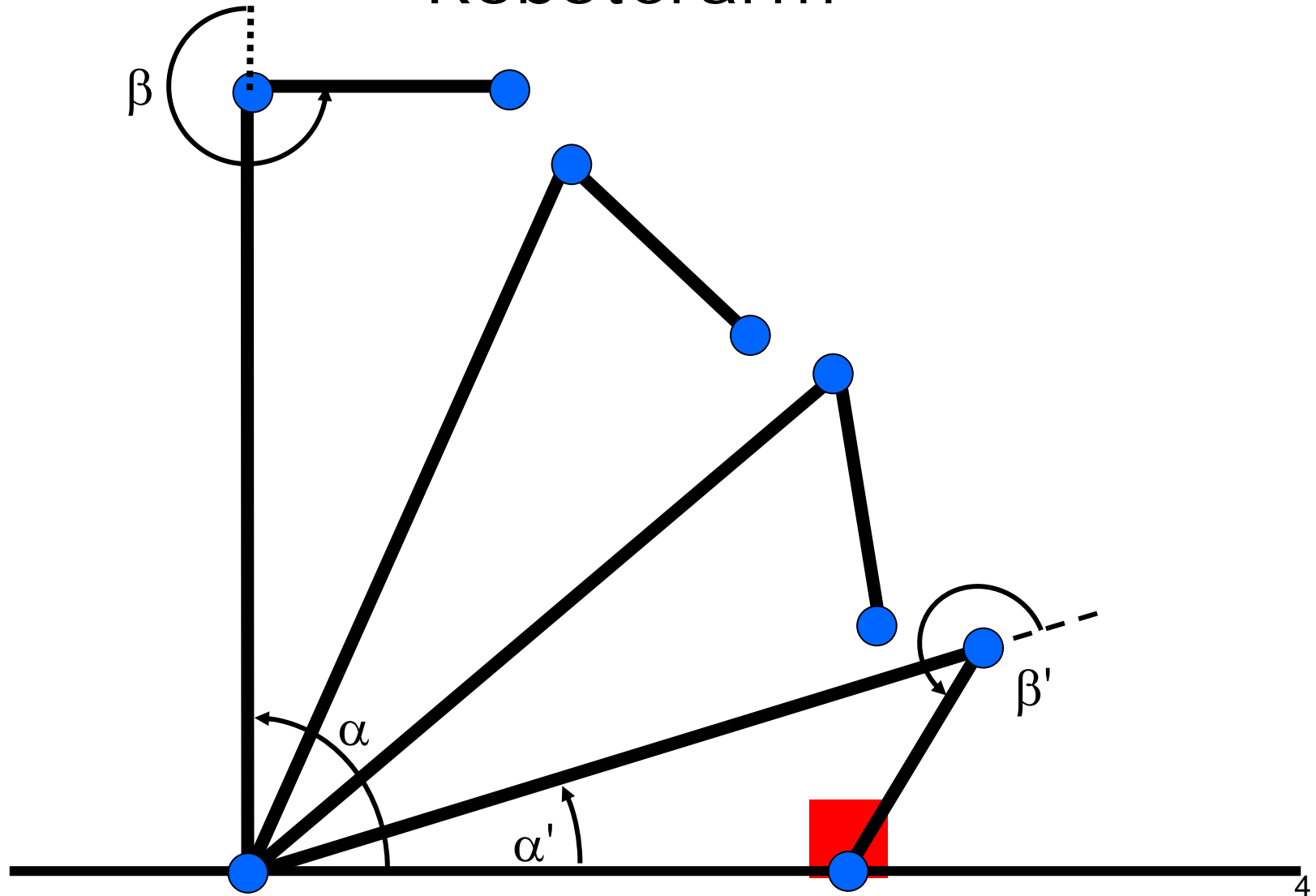
Particle Systems

Verhaltensanimation

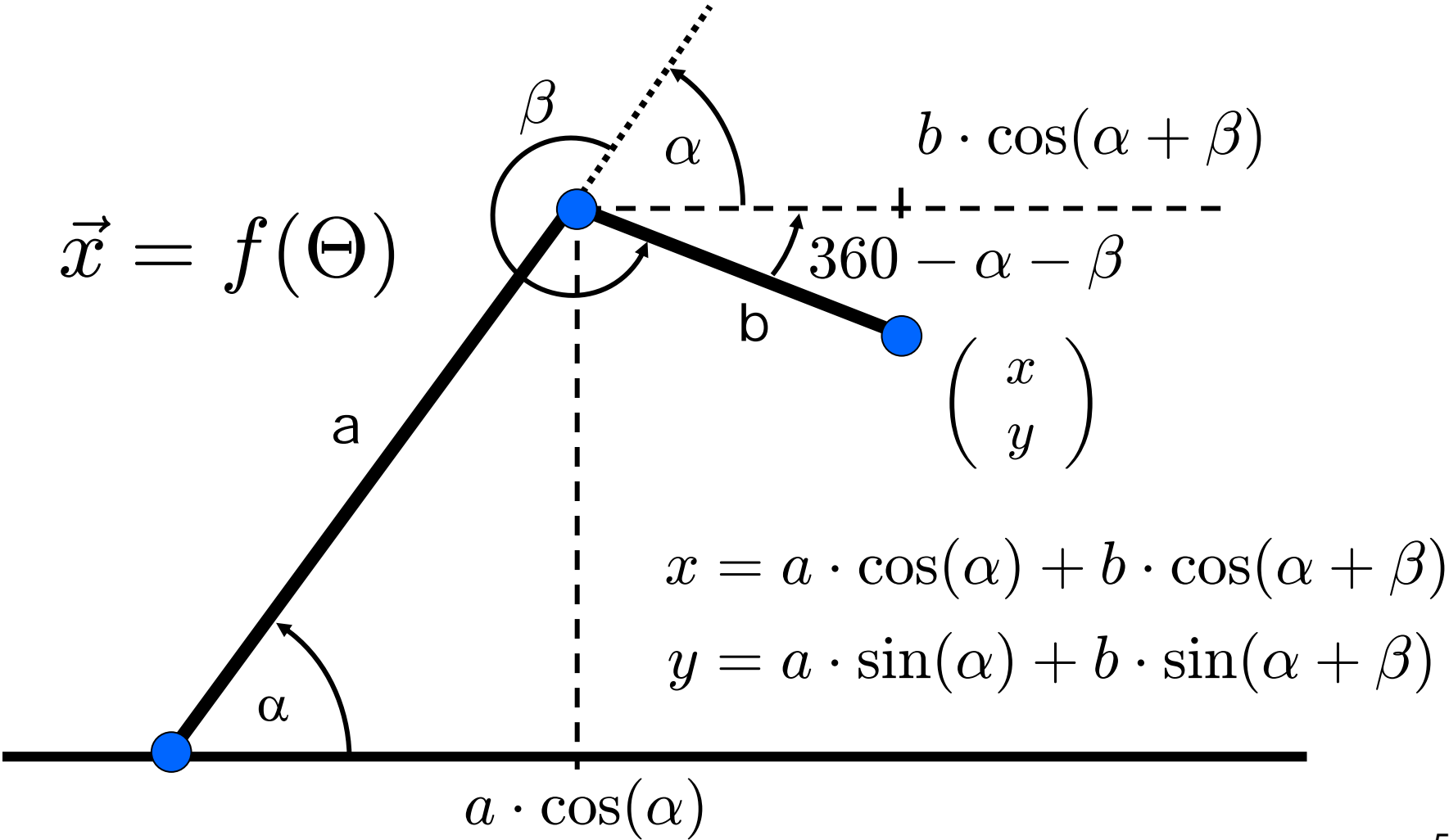
Key frame Animation



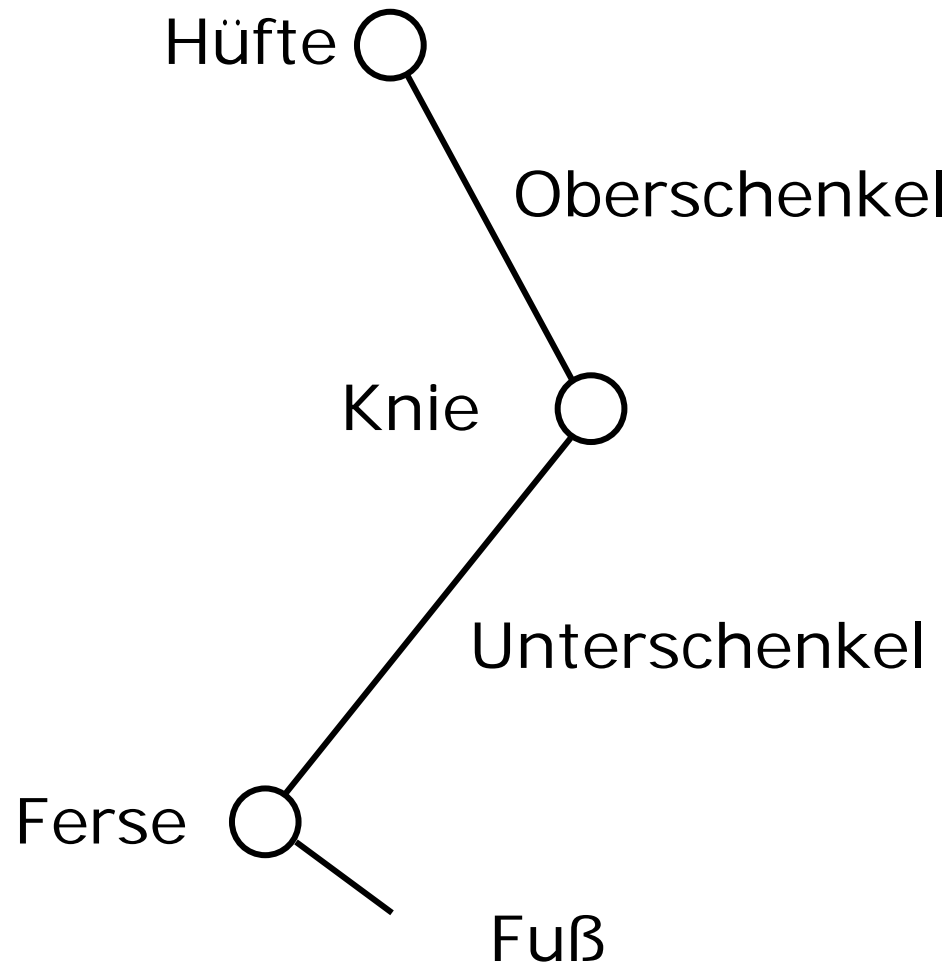
Roboterarm



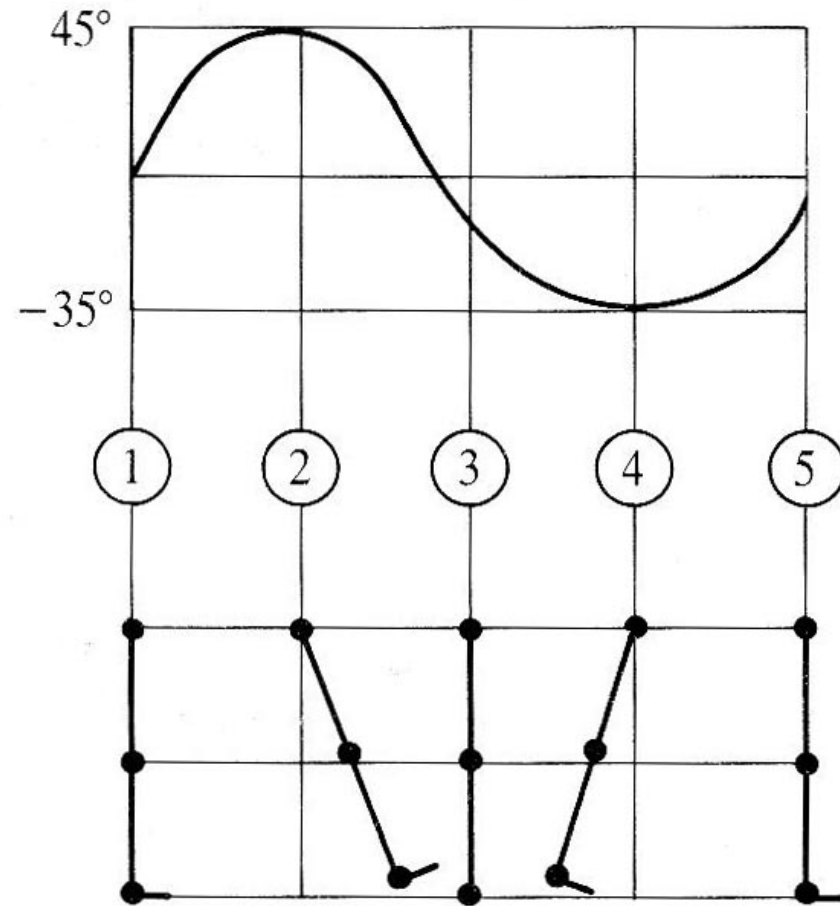
Forward Kinematics



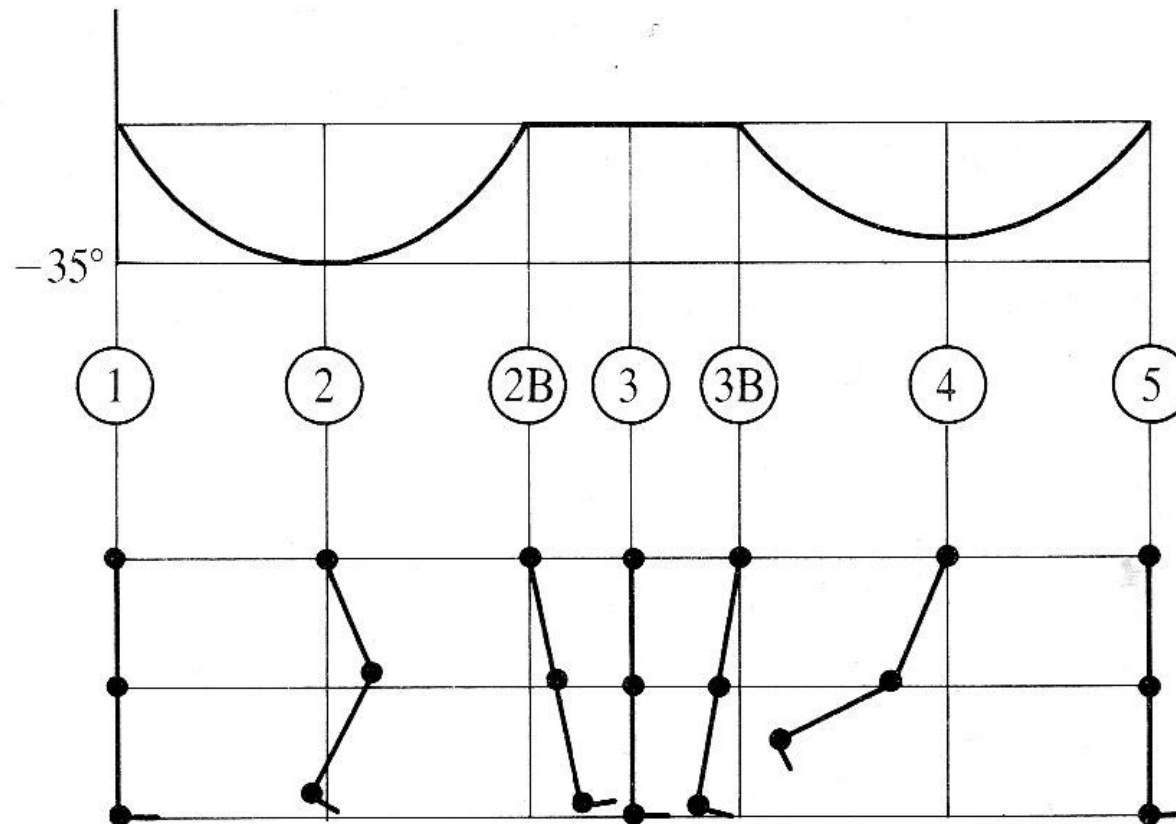
Skript für Forward Kinematics



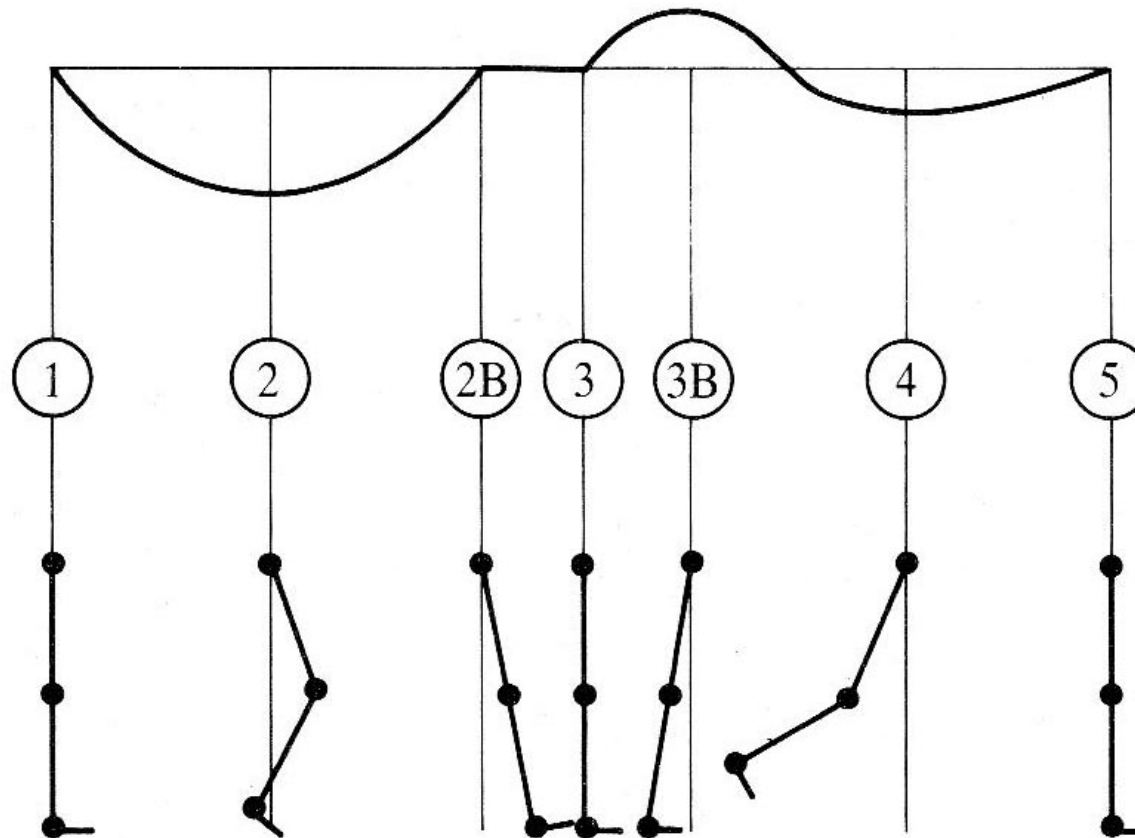
Rotation in der Hüfte



Rotation im Knie



Rotation in der Ferse

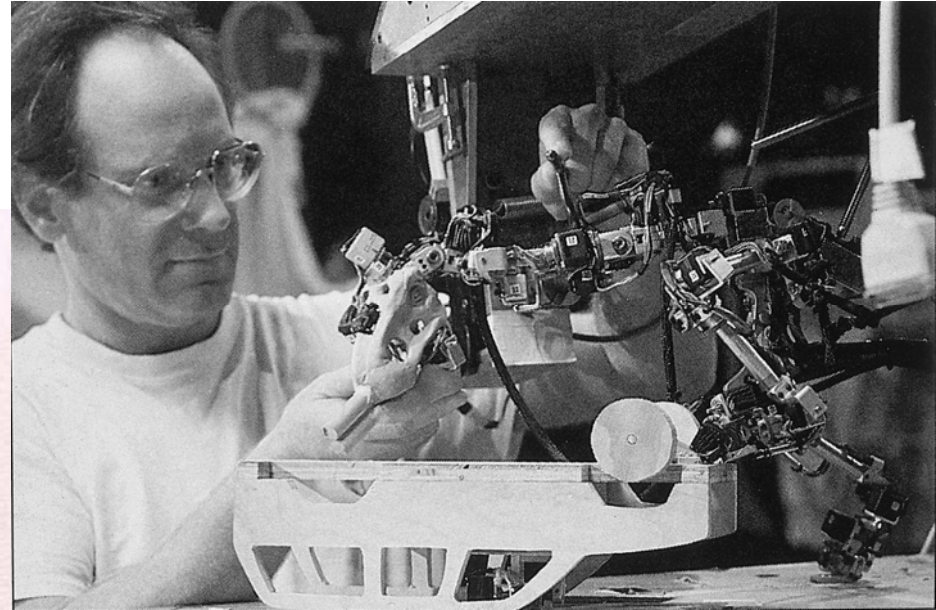
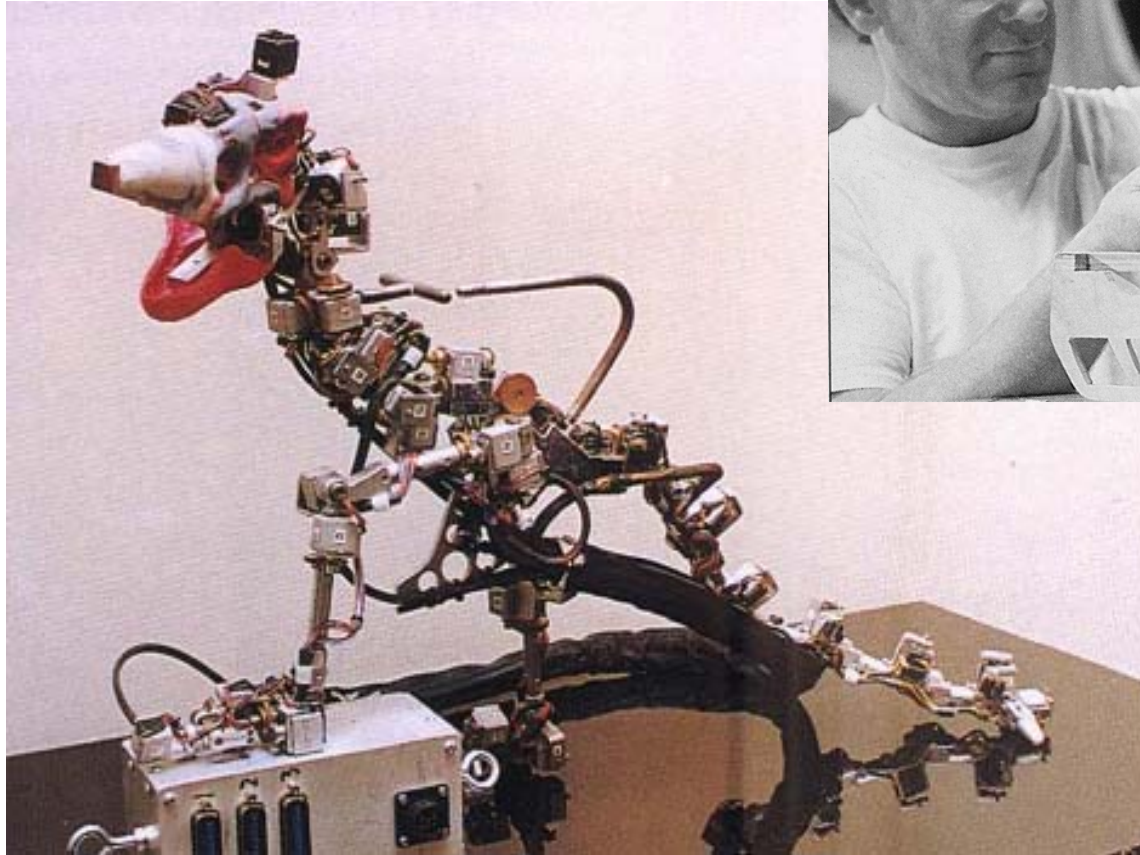


Jurassic Park



geplant als Stop Motion Film

Jurassic Park



R. Magid: "After Jurassic Park",
American Cinematographer,
December 1993.

Abgedruckt in: Alan Watt "3D-
Computergrafik", S. 549,
Pearson Studium, ein Imprint
von Pearson Education
Deutschland GmbH, 2001.

realisiert mit Dinosaur Input Device (DID)

Inverse Kinematics

$$\Theta = f^{-1}(\vec{x})$$

$$d^2 = a^2 + b^2 - 2ab \cos(\gamma)$$

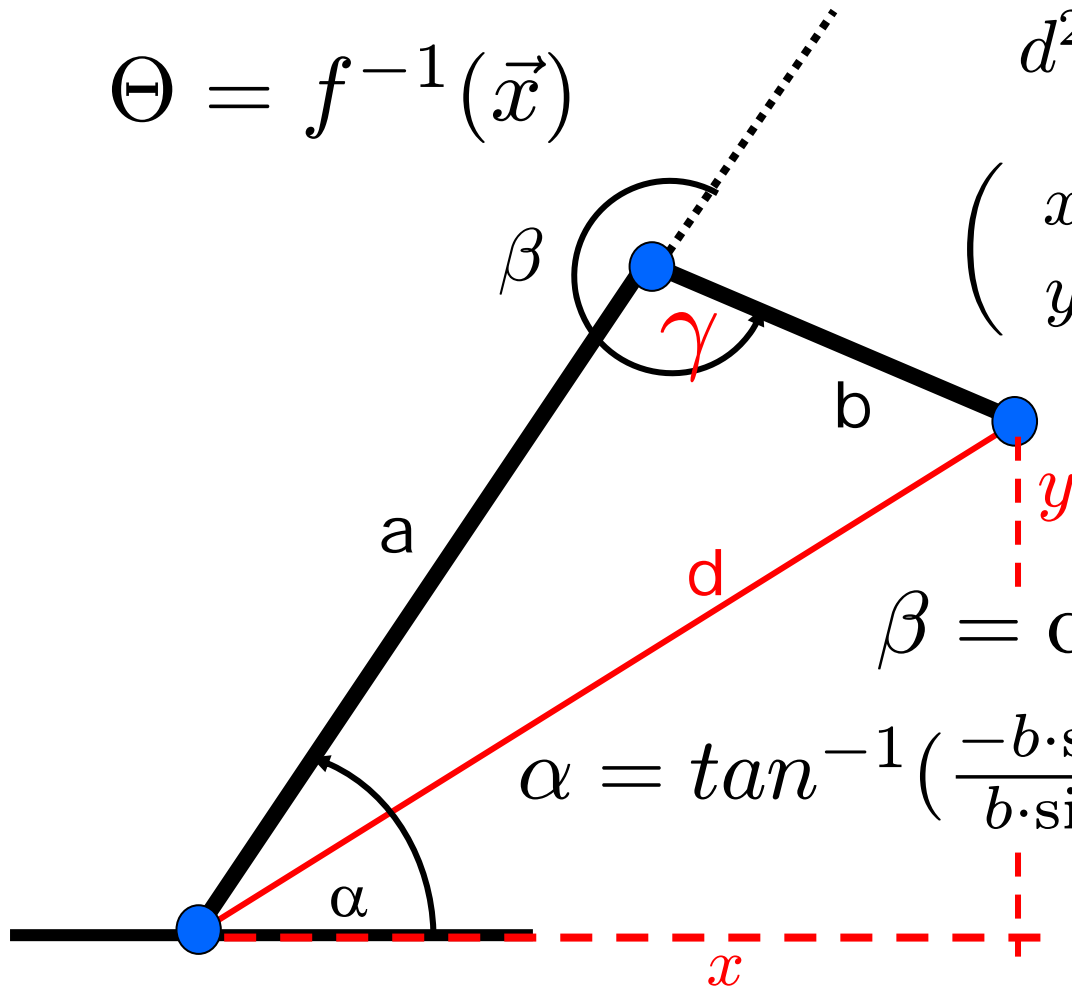
$$d^2 = x^2 + y^2$$

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

$$\cos(\gamma) = \frac{a^2 + b^2 - x^2 - y^2}{2ab}$$

$$\beta = \cos^{-1}\left(\frac{x^2 + y^2 - a^2 - b^2}{2 \cdot a \cdot b}\right)$$

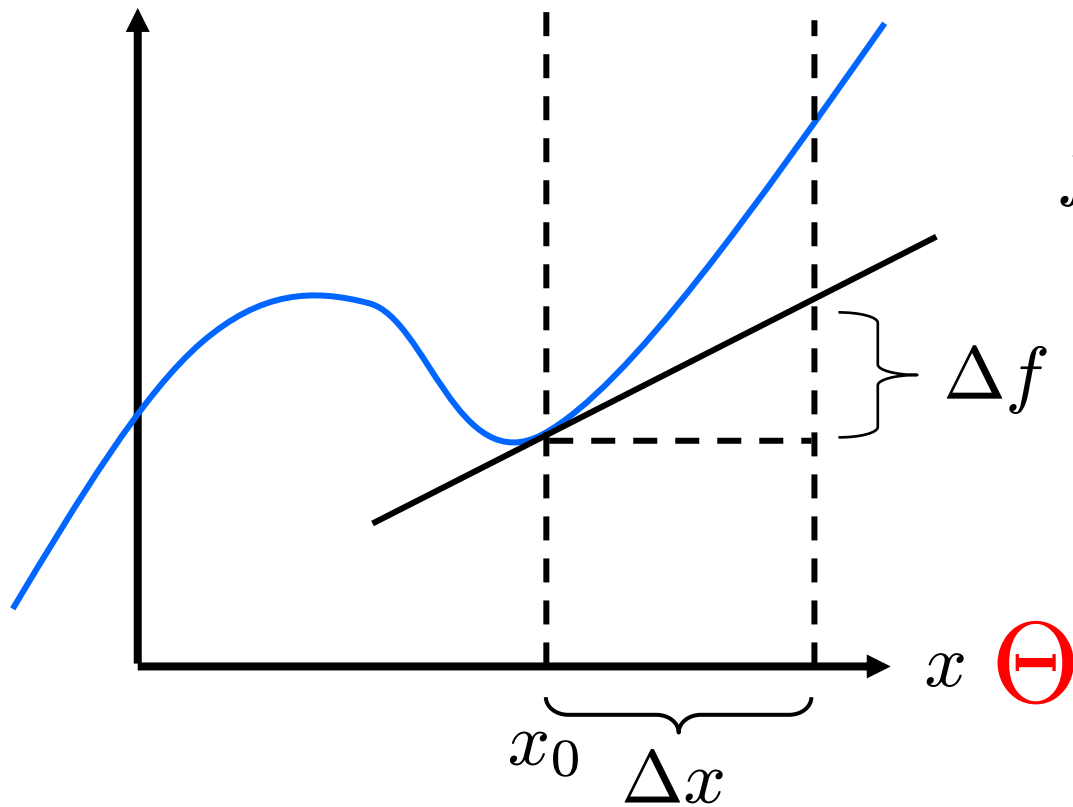
$$\alpha = \tan^{-1}\left(\frac{-b \cdot \sin(\beta) \cdot x + (a + b \cdot \cos(\beta)) \cdot y}{b \cdot \sin(\beta) \cdot y + (a + b \cdot \cos(\beta)) \cdot x}\right)$$



Differenzierbarkeit

$$\vec{x} = f(\Theta)$$

$$y = f(x)$$



$$f'(x_0) \approx \frac{\Delta f}{\Delta x}$$

$$f'(x_0) \cdot \Delta x \approx \Delta f$$

$$\Delta x \approx \frac{\Delta f}{f'(x_0)}$$

Jakobi-Matrix

Die Jakobi-Matrix einer differenzierbaren Abbildung

$$f : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

ist die $m \times n$ Matrix aller partiellen Ableitungen

$$J_f = \frac{\partial f}{\partial x} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

Abhängigkeit zwischen dx und $d\Theta$

Problem: $\Theta = f^{-1}(\vec{x})$

Aber: kleine Änderungen im Winkel verursachen kleine Änderungen in der Position

$$J(\Theta) = \frac{\partial \vec{x}}{\partial \Theta}$$

$$J(\Theta) \partial \Theta = \partial \vec{x}$$

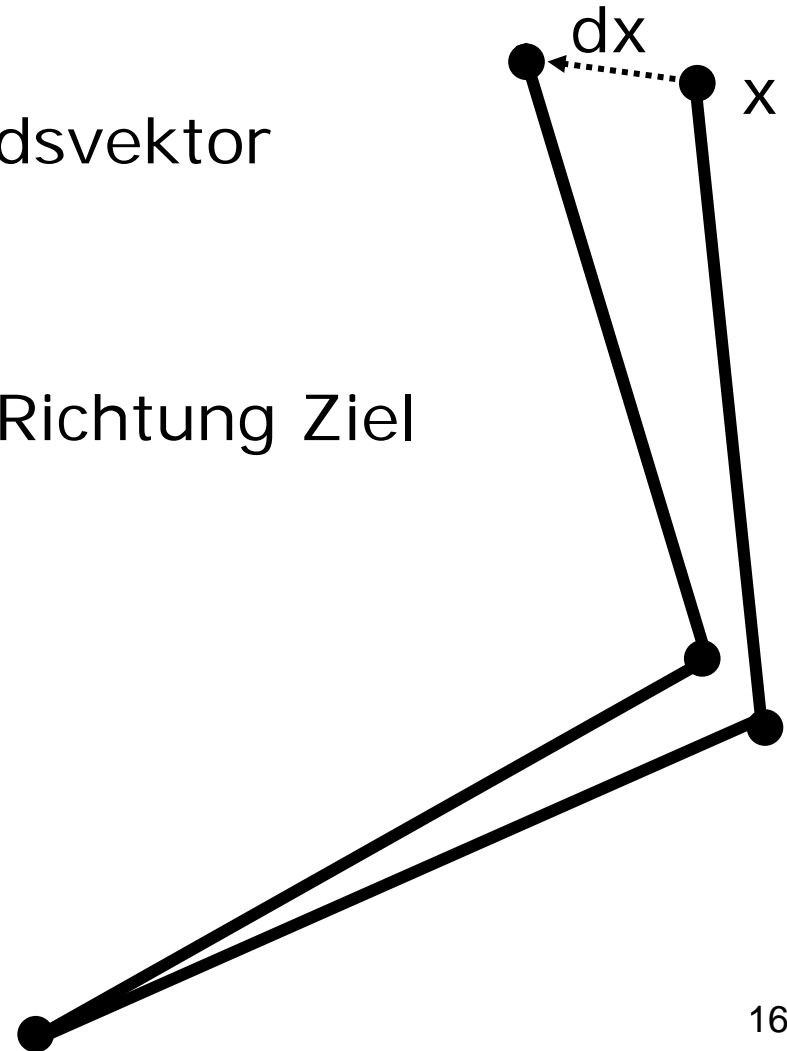
$$\partial \Theta = J^{-1}(\Theta)(\partial \vec{x})$$

Iterationsverfahren

Sei x die aktuelle Position

Sei Θ der aktuelle Zustandsvektor

```
while (!fertig) {  
    dx := kleine Bewegung Richtung Ziel  
     $J(\Theta) = dx/d\Theta$   
    berechne Inverse von J  
     $d\Theta := J^{-1}(\Theta)(dx)$   
     $x := f(\Theta + d\Theta)$   
}
```



Particle Systems

Geeignet für

- Sand
- Funken
- Wasser
- Schnee
- Feuer
- ...

Simulation physikalischer Gesetze
keine Interaktion untereinander

Pioniere

- William Reeves [1983]:
"Particle Systems - A Technique for Modeling a Class of Fuzzy Objects"
ACM Transaction on Graphics
LucasFilm, Pixar (Luxor Jr.)
- Karl Sims [1990]:
"Particle Animation and Rendering using Data Parallel Computation"
ACM Computer Graphics
Connection Machine CM-2, 65.536 Prozessoren

Partikeleigenschaften

- Position
- Geschwindigkeit
- Bewegungsrichtung
- Lebenszeit
- Größe
- Farbe
- Transparenz
- Gestalt

Phasen

- Generierung neuer Partikel
- Zuordnung von Attributen
- Entfernen von Partikeln
- Transformation von Partikeln
- Rendern des neuen Frames

Physik

Position P , Geschwindigkeit V , Beschleunigung A
Masse m_0 im Zentrum O , Flächennormale N
Dämpfungsparameter d , Grenzggeschwindigkeit v_0

$$A = g \cdot m_0 \cdot \frac{O-P}{|O-P|^3}$$

$$F = \frac{g \cdot m_0 \cdot m_1}{r^2}$$

$$V' := V + A \cdot \Delta t$$

$$A = \frac{F}{m}$$

$$P' := P + \frac{V+V'}{2} \cdot \Delta t$$

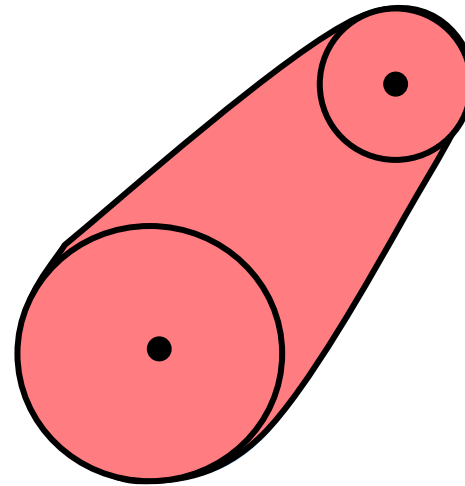
$$V' := V - 2 \cdot (V \cdot N) \cdot N$$

$$V' = V \cdot \max(1 - d\Delta t, \min(1.0, \frac{v_0}{|V|}))$$

Particle Rendering

Für Head und Tail:

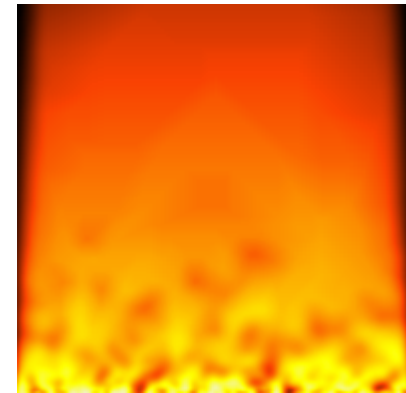
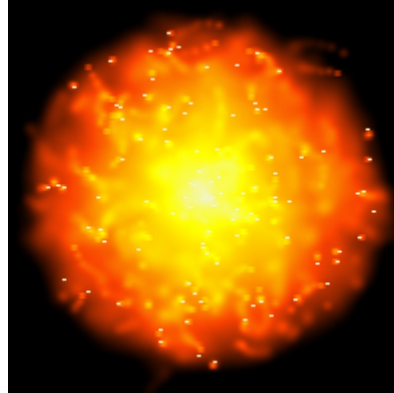
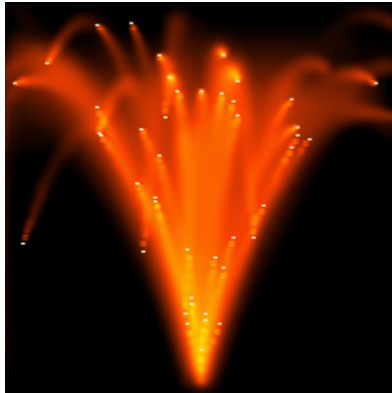
- Position
- Radius
- Farbverlauf
- Transparenz



Motion Blur berücksichtigen !

Particle Systems Demos

[~cg/2008/skript/Applets/Particle/sims.html](http://cg/2008/skript/Applets/Particle/sims.html)



<http://www.jhlab.com/java/particles2.html>

[~cg/2008/skript/Applets/Particle/jhlab.html](http://cg/2008/skript/Applets/Particle/jhlab.html)

[~cg/2008/skript/Applets/Particle/kung.html](http://cg/2008/skript/Applets/Particle/kung.html)

Pflanzen



white.sand © Alvy Ray Smith, Lucasfilm

Verhaltensanimation

Simple Vehicle Model:

- Masse
- Position
- Fahrtrichtung
- Geschwindigkeit
- Beschleunigung

Vehikel interagiert

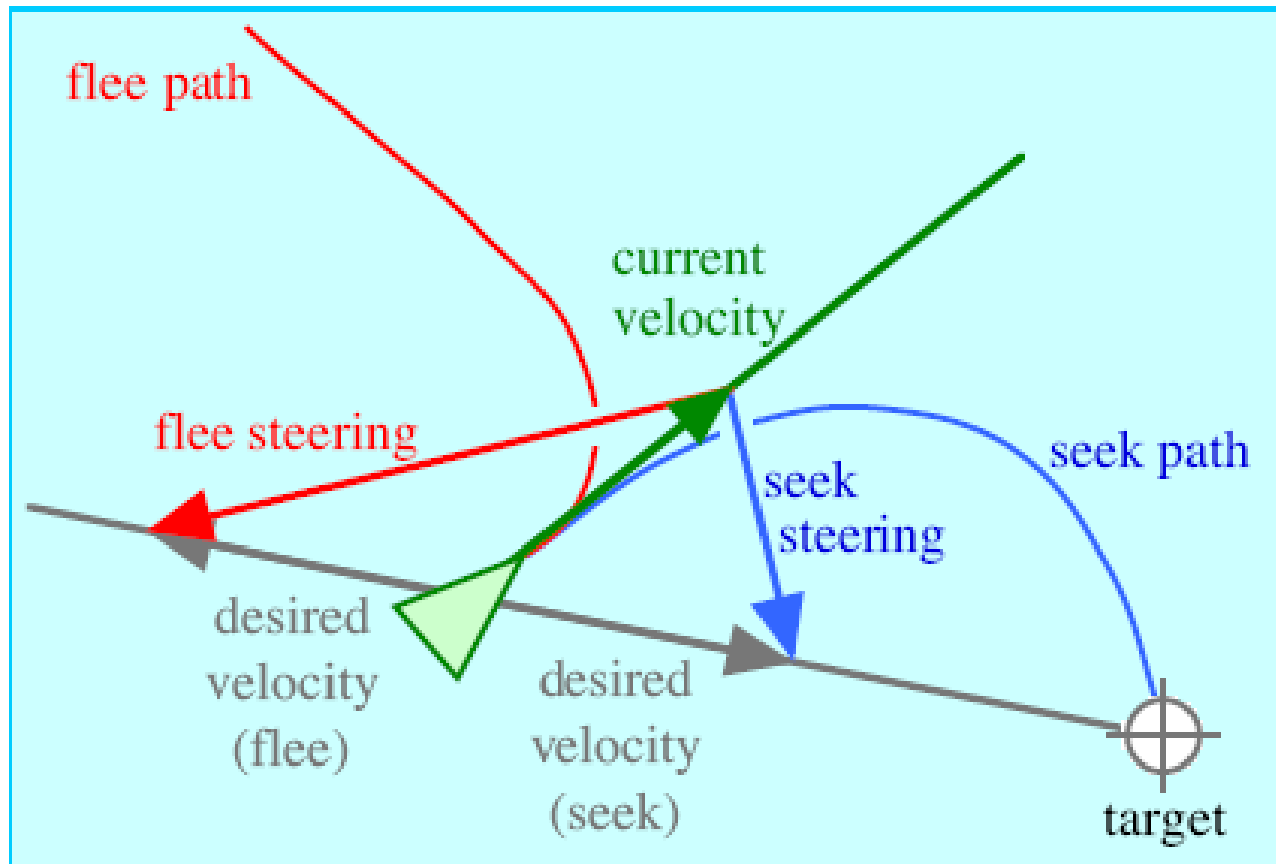
- mit Umwelt
- mit anderen Vehikeln

Pionier

- Craig W. Reynolds [1999]:
Steering Behaviors for autonomous Characters
[Sony Computer Entertainment]

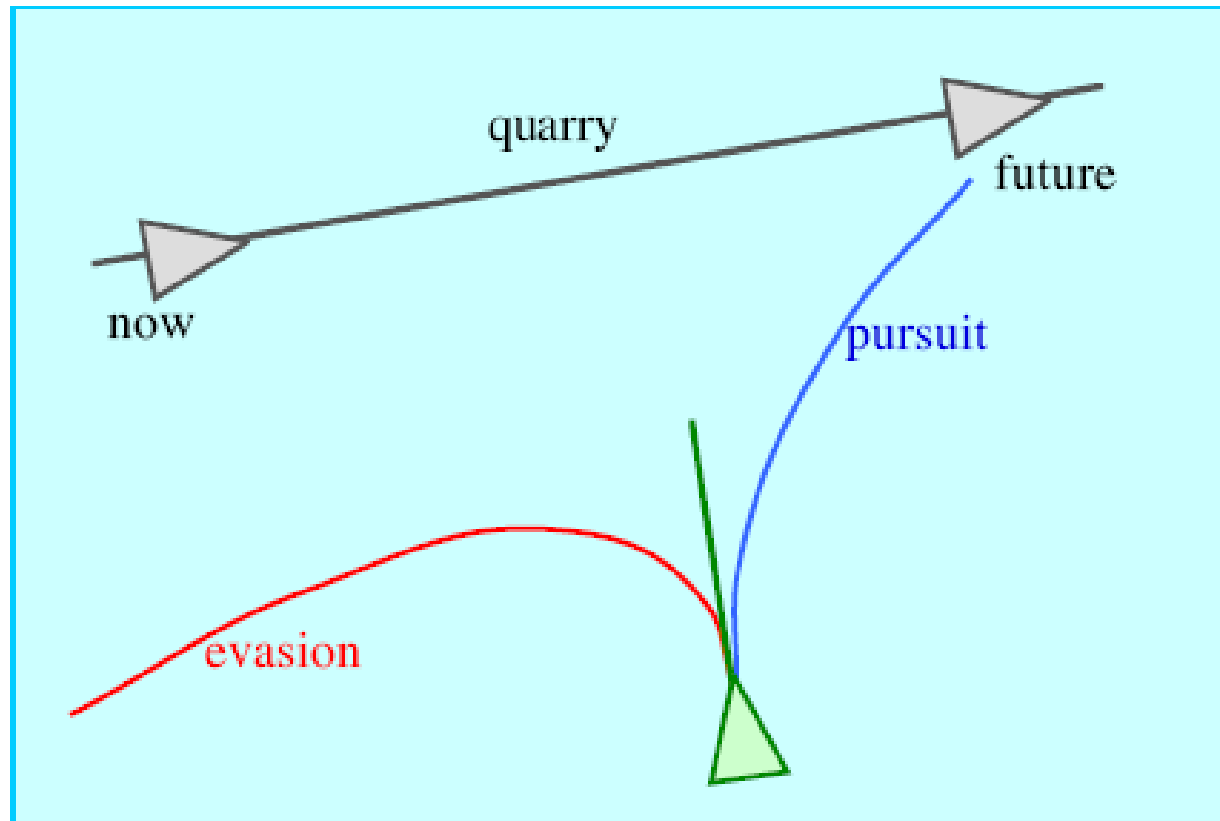
<http://www.red3d.com/cwr/steer>

Seek & Flee



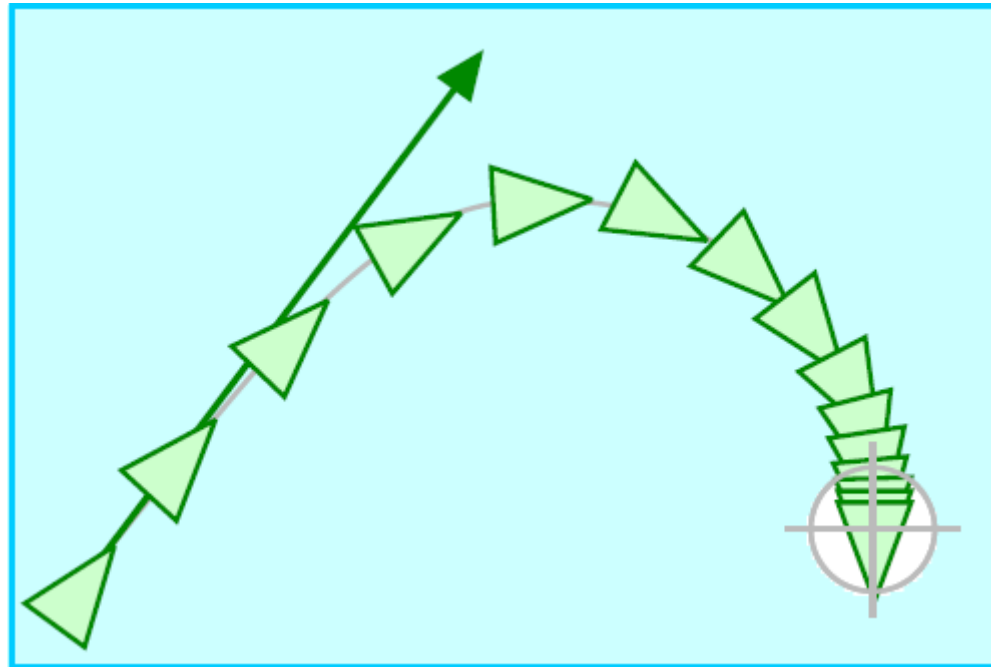
<http://www.red3d.com/cwr/steer/SeekFlee.html>

Pursuit & Evasion



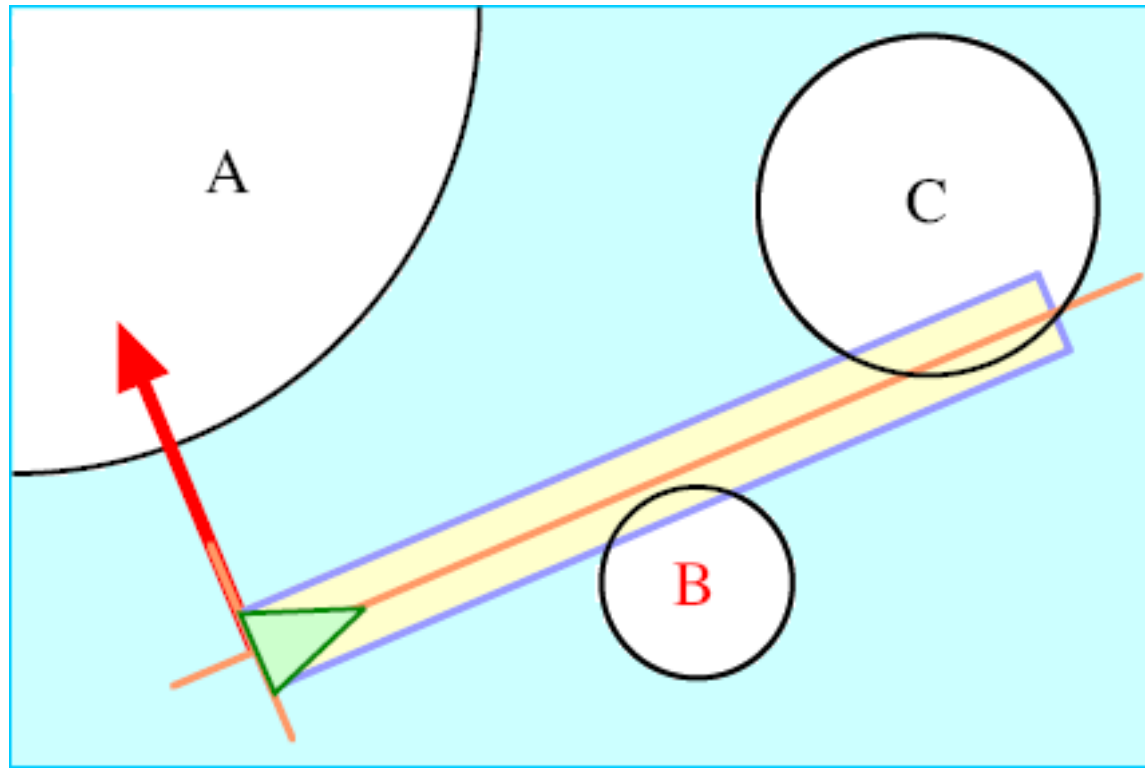
<http://www.red3d.com/cwr/steer/PursueEvade.html>

Arrival



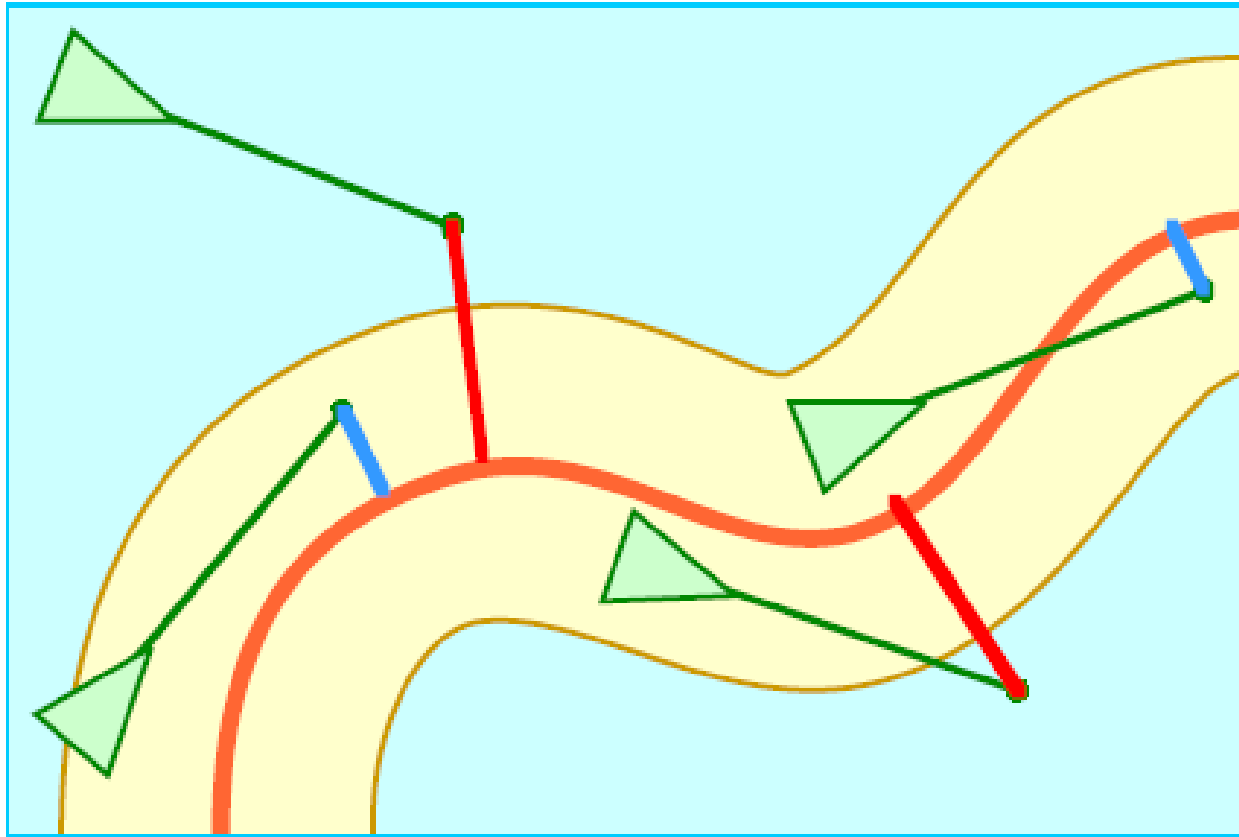
<http://www.red3d.com/cwr/steer/Arrival.html>

Obstacle Avoidance



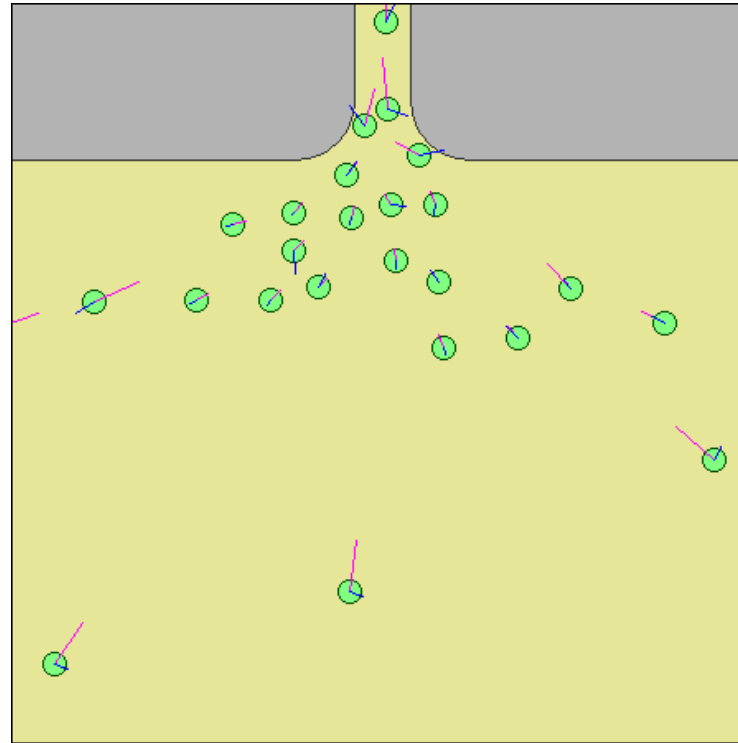
<http://www.red3d.com/cwr/steer/Obstacle.html>

Path Following



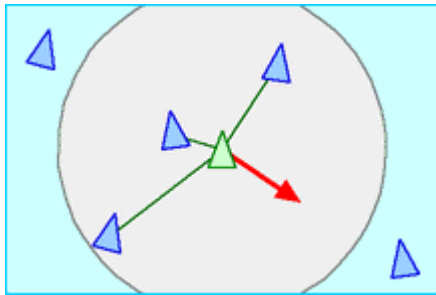
<http://www.red3d.com/cwr/steer/PathFollow.html>

Queuing Behaviour at door

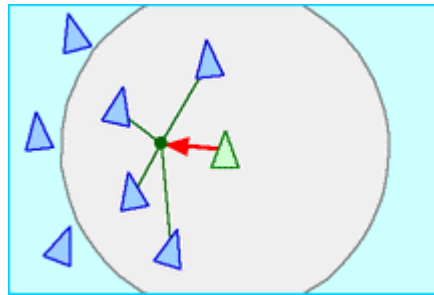


<http://www.red3d.com/cwr/steer/Doorway.html>

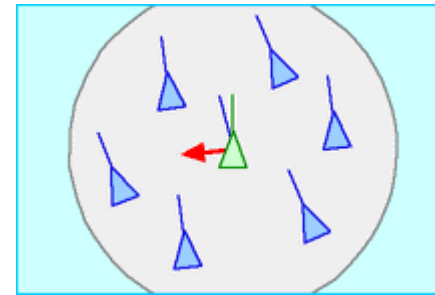
Schwarmverhalten



Separation

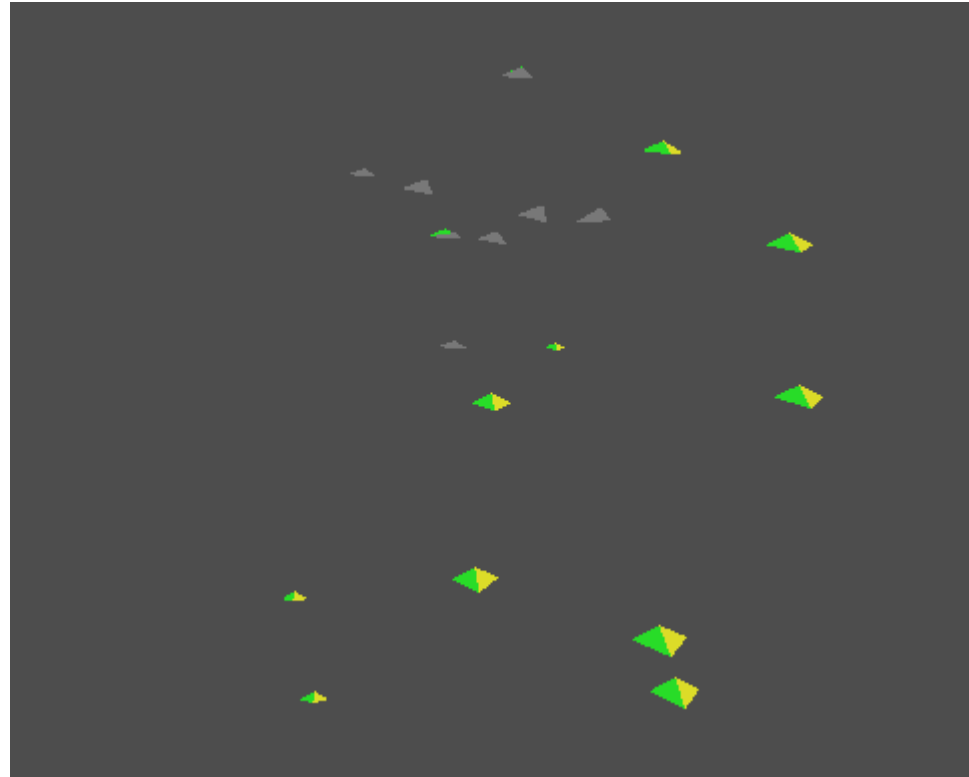


Kohäsion



Ausrichtung

Java Boids



<http://www.red3d.com/cwr/boids/applet/>

<http://www.bekkoame.ne.jp/~ishmnn/java/boid3.html>

Scanline Production GmbH, München



~cg/2010/skript/Applets/Particle/scanline.htm