UNIVERSITY OF MINNESOTA
Driven to Discover®

Animation and Inverse Kinematics

CSCI 4611: Programming Interactive Computer Graphics and Games

Evan Suma Rosenberg | CSCI 4611 | Fall 2022

Applications of Animation

- Movies and television
- Video games
- Simulation and training
- Medicine
- Robotics
- Scientific visualization
- Education
- ... and plenty more

Major Topics in Computer Animation

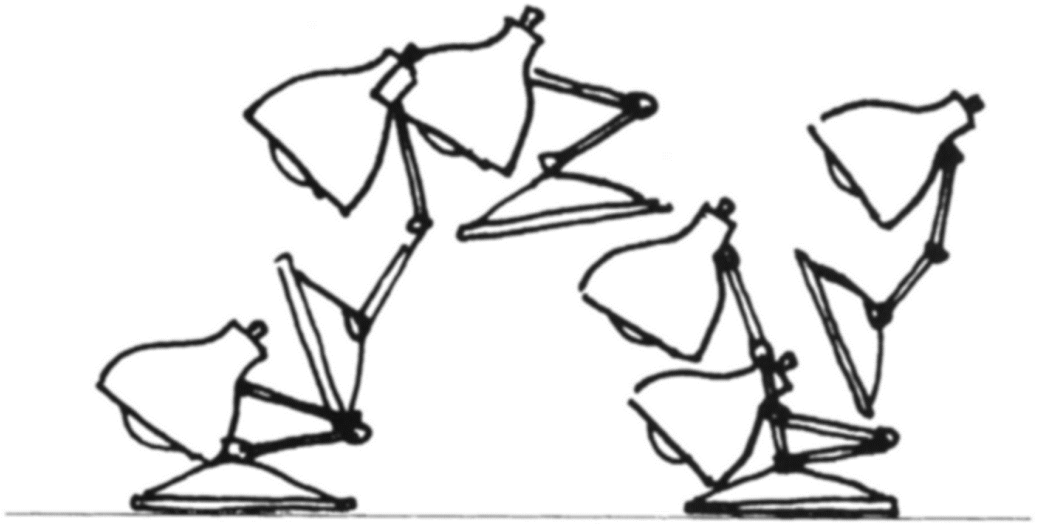
- **Physics (dynamics, simulation, mechanics)**
  - Particles
  - Rigid bodies (collisions, contact, stacking, rolling, sliding)
  - Articulated bodies (hinges, constraints)
  - Deformable bodies (elasticity, plasticity, fracture, cloth)
  - Fluid dynamics (fluid flow, combustion, phase changes)
  - Character dynamics (body motion, skin and muscle, hair, clothing)
- **Character animation**
  - Rigging
  - Motion playback (keyframing, blending, sequencing)
  - Motion synthesis (locomotion, inverse kinematics, retargeting, procedural animation)
  - Motion capture
- **Artificial intelligence**
  - Behavioral animation
  - Background characters (flocks, herds, crowds)
  - Video games (non-player characters)

Key Framing

Key framing is the approach used in traditional cel (celluloid) animation.

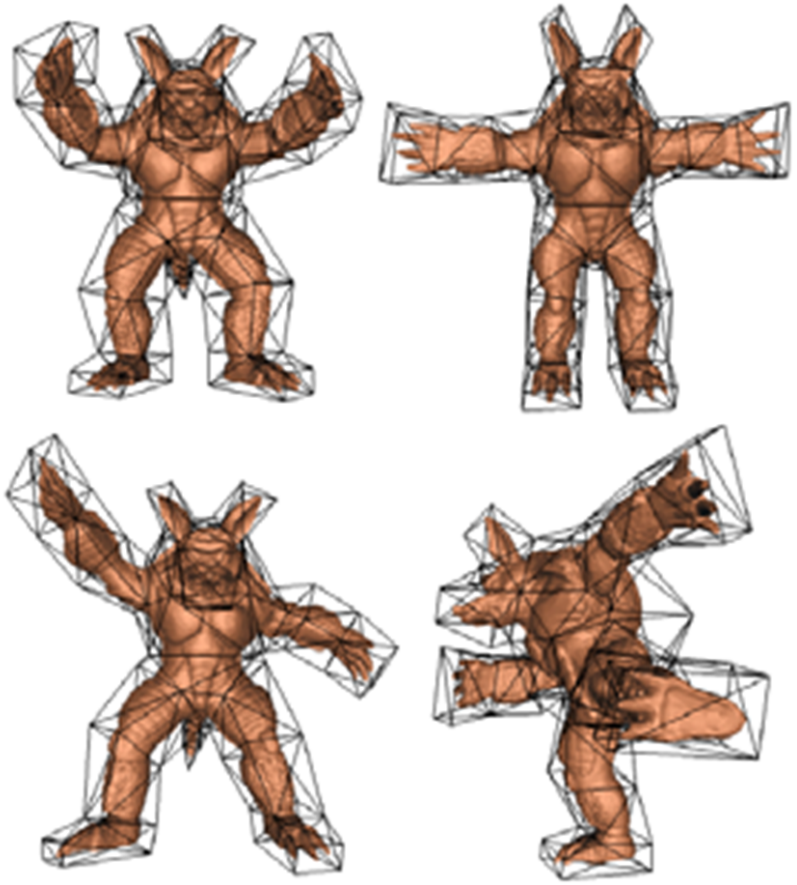
The animator starts by specifying the positions and orientations at various key points in time.

In-betweeners generate the images in-between the key frames. This was once a job for apprentice animators. These days it is largely done by computer.

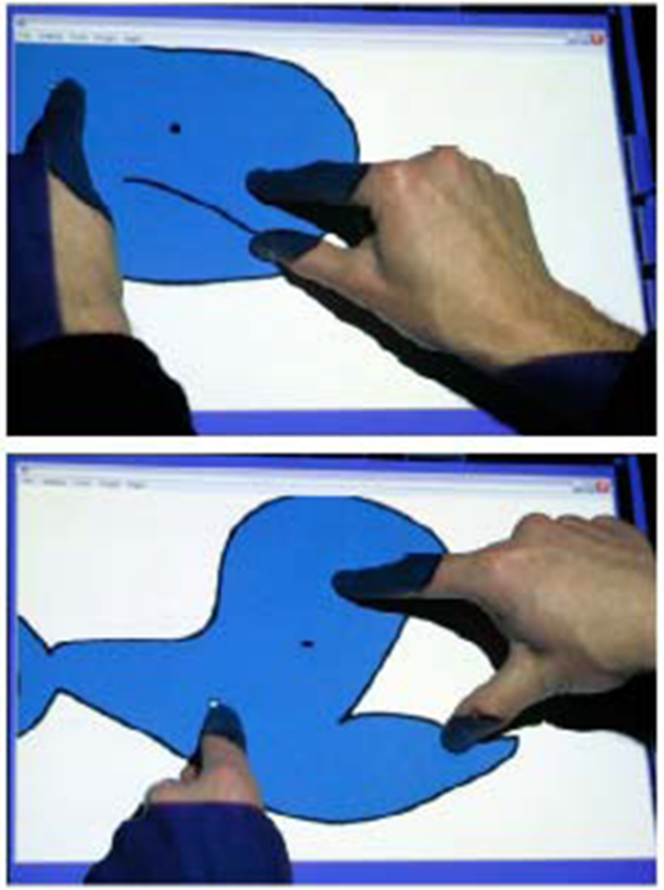


The illustrations in this section are taken from a classical paper on animation by John Lasseter from Pixar. "Principles of Traditional Animation Applied to 3D Computer Graphics," SIGGRAPH '87, pp. 35-44.

Specifying Key Frames

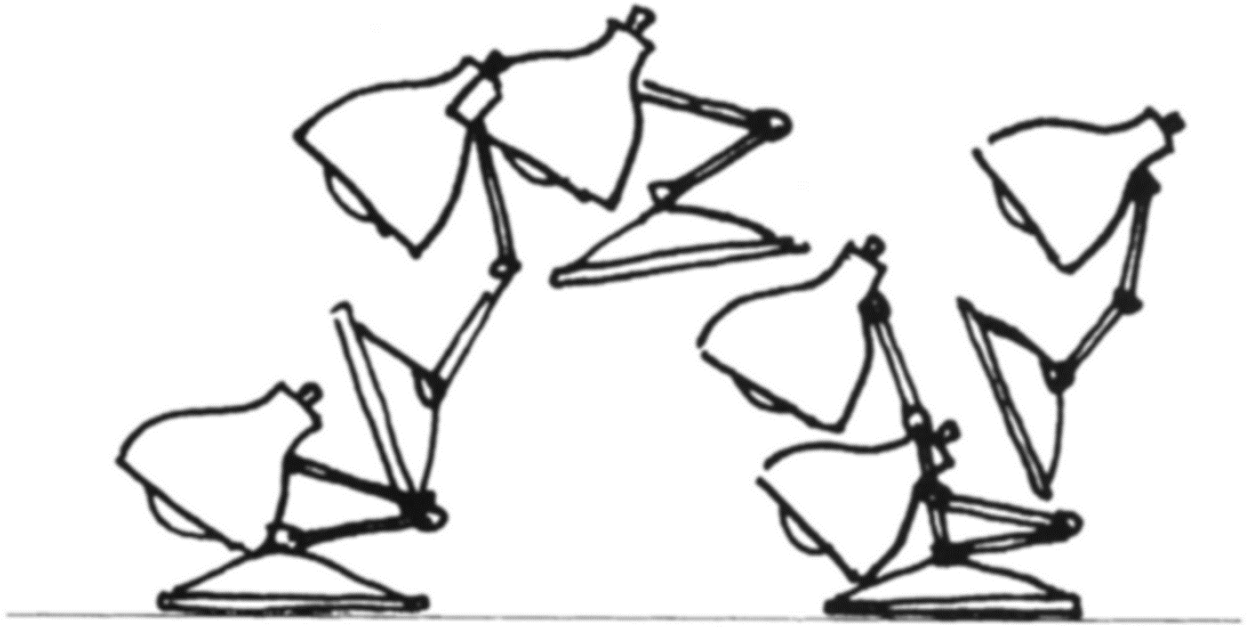


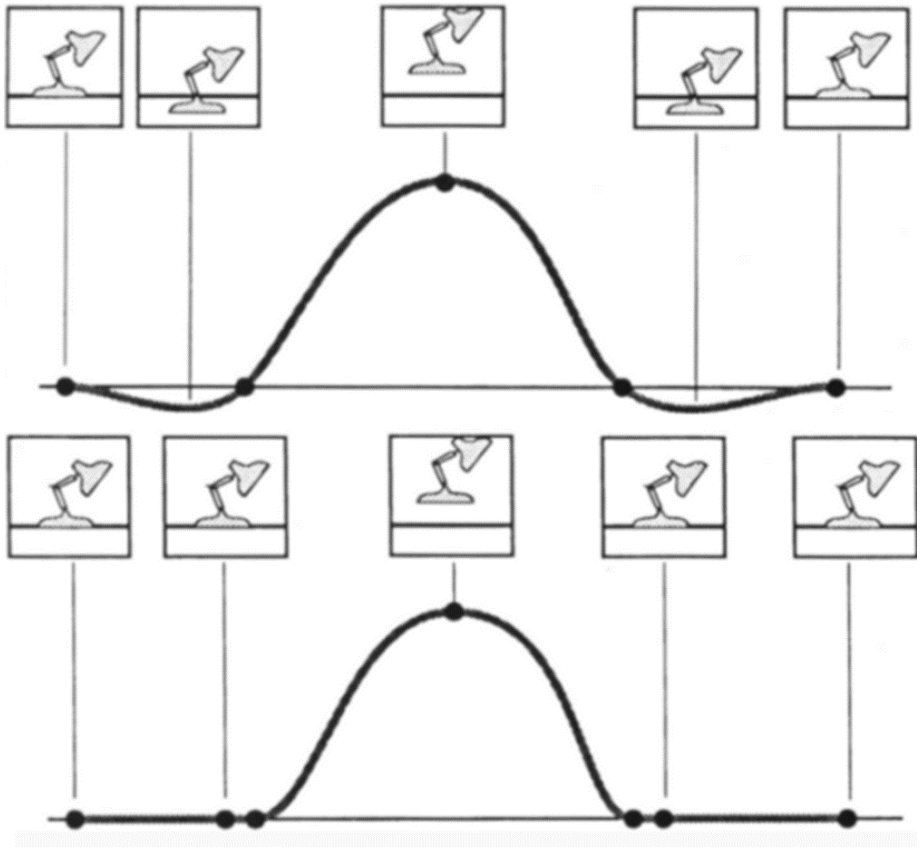
Mean-value coordinates for closed triangular meshes, Ju et. al. SIGGRAPH 2005



As-Rigid-As-Possible Shape Manipulation, Igarashi et. al., SIGGRAPH 2005

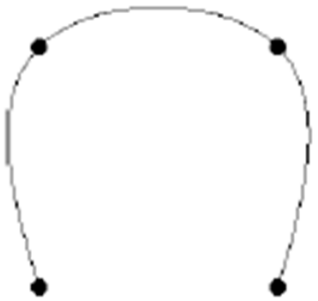
Interpolating Key Frames



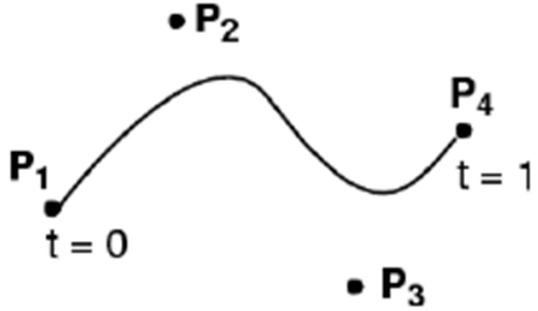


Splines are used to interpolate the positions of objects between key frames.

Splines

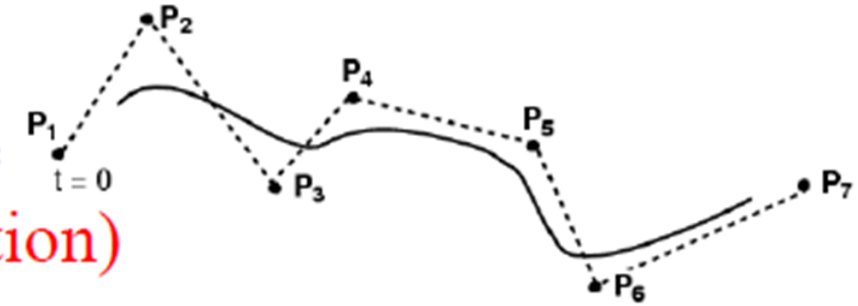


Interpolation



Bézier (approximation)

BSpline
(approximation)



Splines are smooth curves defined by control points.

Moving the control points changes the curve.

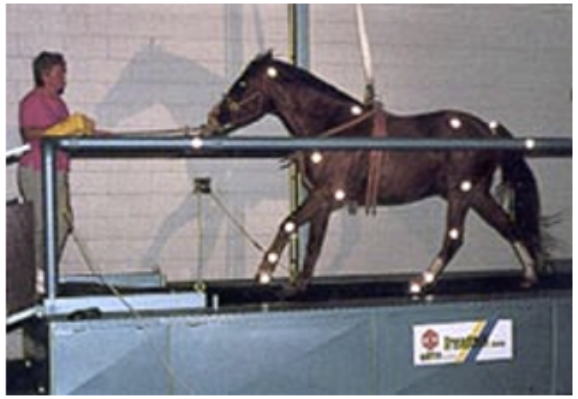

YouTube

Skeletons and Skinning

- **Skeleton**: a collection of bones and joints
- **Skinning**: associating each vertex on the shape to an combination of bone/joint locations.
- Real-time deformation by applying **skinning weights** to the skeleton in an arbitrary pose.


YouTube

Motion Capture



Motion capture is a common technique for animating computer generated characters. These techniques rely on computer vision and tracking technologies to acquire motion that is **retargeted** to the animated character.

Using Mocap Data

- **We often collect a database of motion capture clips.**

  This raises a number of interesting questions.

- **When animating a character, how do we smoothly transition from one clip to the next (blending)?**

- **How do we get our character to do something similar, but not exactly what we collected via motion capture (editing)?**

- **How do we map motion onto characters of different sizes and shapes in a way that looks plausible (retargeting)?**

- **Data is often noisy and requires manual cleaning.**


YouTube

Real-Time Animation

The previous examples worked well for videos that are pre-rendered, such as movies.

What if the animated character needs to move and respond realistically in real-time?

Video games use **procedural animation** is used to automatically generate movements "on the fly."

This allows for a more diverse series of actions than could otherwise be created using predefined animations.


HOW
TRICO
YouTube
WAS
ANIMATED

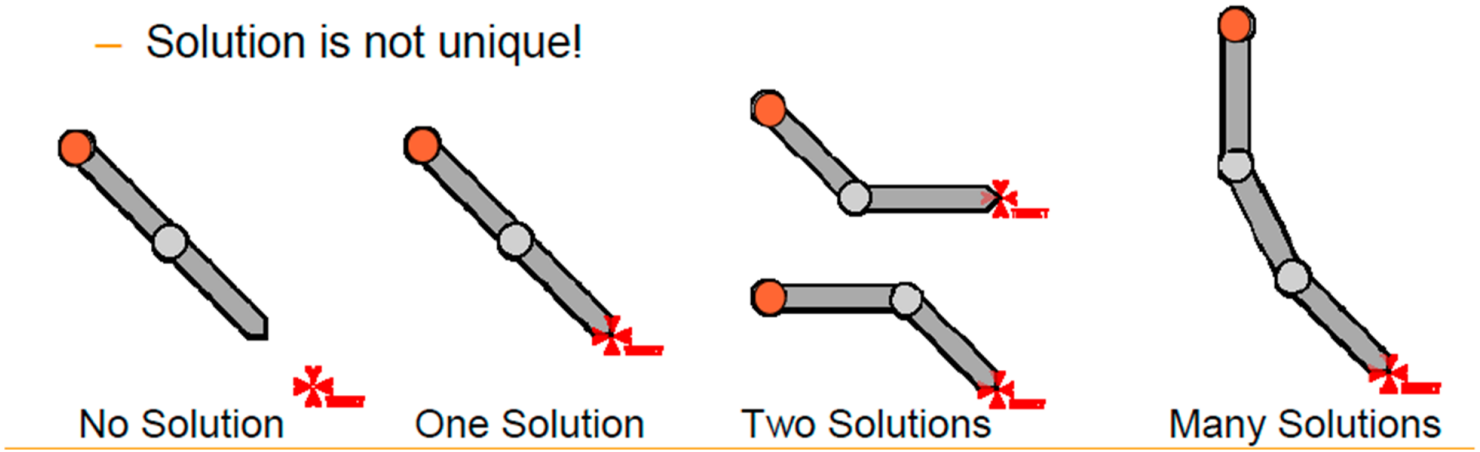
Kinematics

Forward kinematics (FK)

- Calculating the locations of every bone based on the joint angles
- This is what we implemented in the previous lecture

Inverse kinematics (IK)

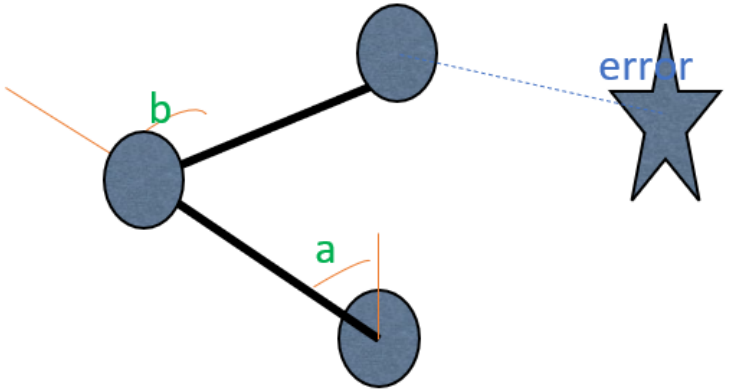
- Given locations of an end effector, we compute the joint angles required to reach it
- Important in video games, robotics, and model design

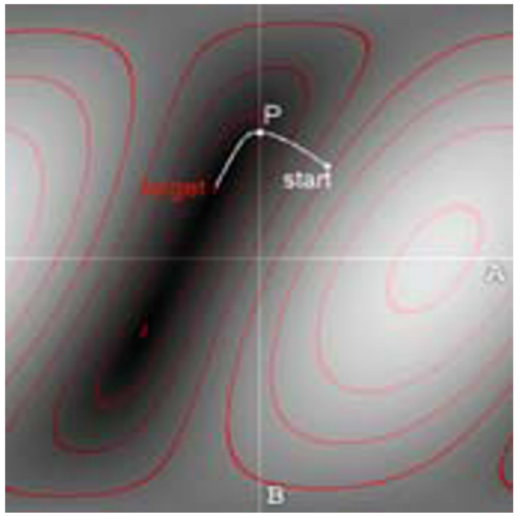


Solving for Inverse Kinematics

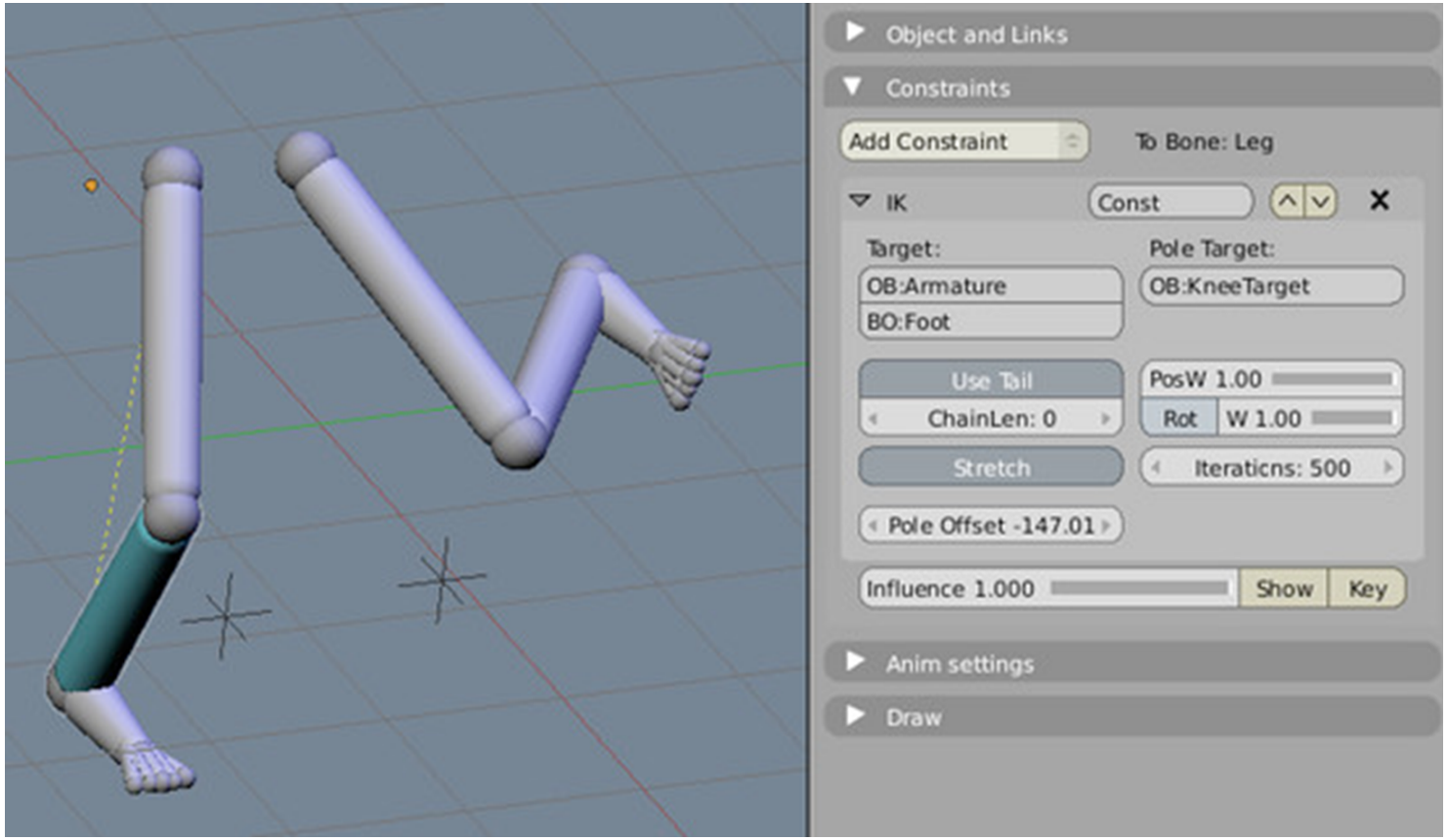
- Solving inverse kinematics requires finding correct angles (a,b) to reach the target.
- This reduces to a **configuration space** problem.
- Simple solution: starting from current configuration, search for the minimum by following the gradient (going downhill)
- More complex algorithms include CCD (Cyclic Coordinate Descent) and FABRIK (Forward and Backward Reaching Inverse Kinematics)

  To learn more, take CSCI 5611: Animation and Planning in Games!





Solving for Inverse Kinematics



IK solvers can produce poses that are implausible for character animation.

We can prevent this by imposing **constraints** when solving for joint angles. This is very important in robotics!

Let's extend our program from last week with IK!