Fundamentals of Computer Graphics and Image Processing

Animations, Dynamics (08)

doc. RNDr. Martin Madaras, PhD.
martin.madaras@fmph.uniba.sk

Outline

- Principles of animation
- Keyframe animation
- Articulated figures
- Kinematics
- Dynamics

How the lectures should look like #1

- Ask questions, please!!!
- Be communicative
- More active you are, the better for you!

Manual animation

- Stop-motion animation
- e.g. Coraline, Wallace & Gromit, etc.

Computer Animation

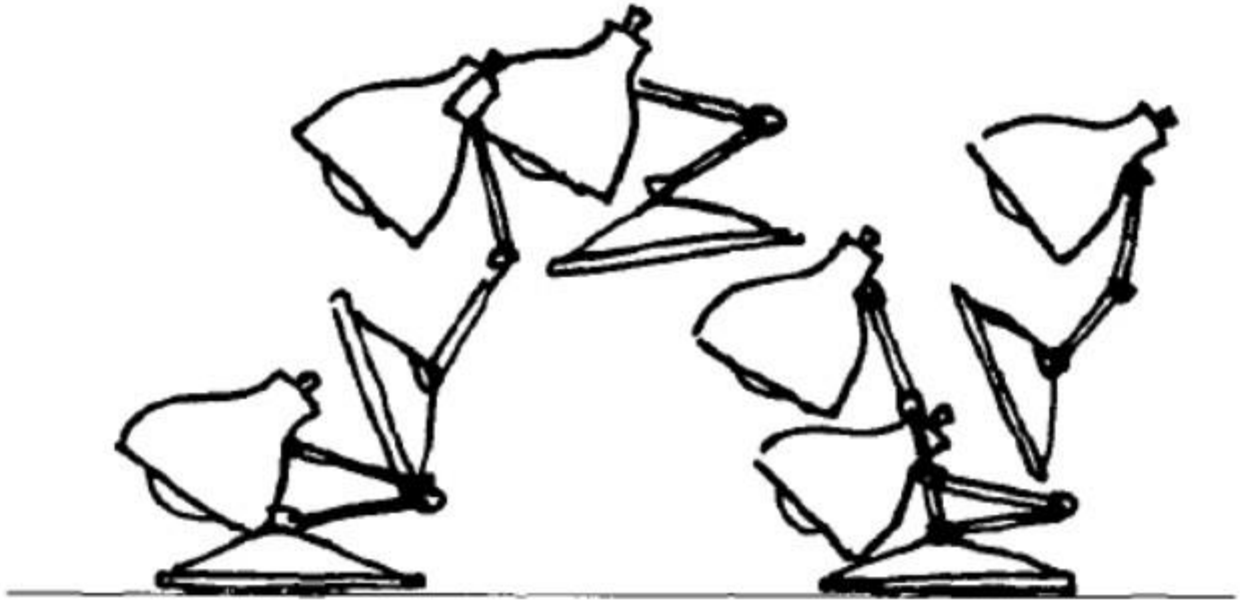
- What is animation?
  - Make objects change over time according to scripted actions
- What is simulation?
  - Predict how object change over time according to physical laws

Computer Animation

- Animation pipeline
  - 3D Modeling
  - Articulation
  - Motion specification
  - Motion simulation
  - Shading
  - Lighting
  - Rendering
  - Postprocessing

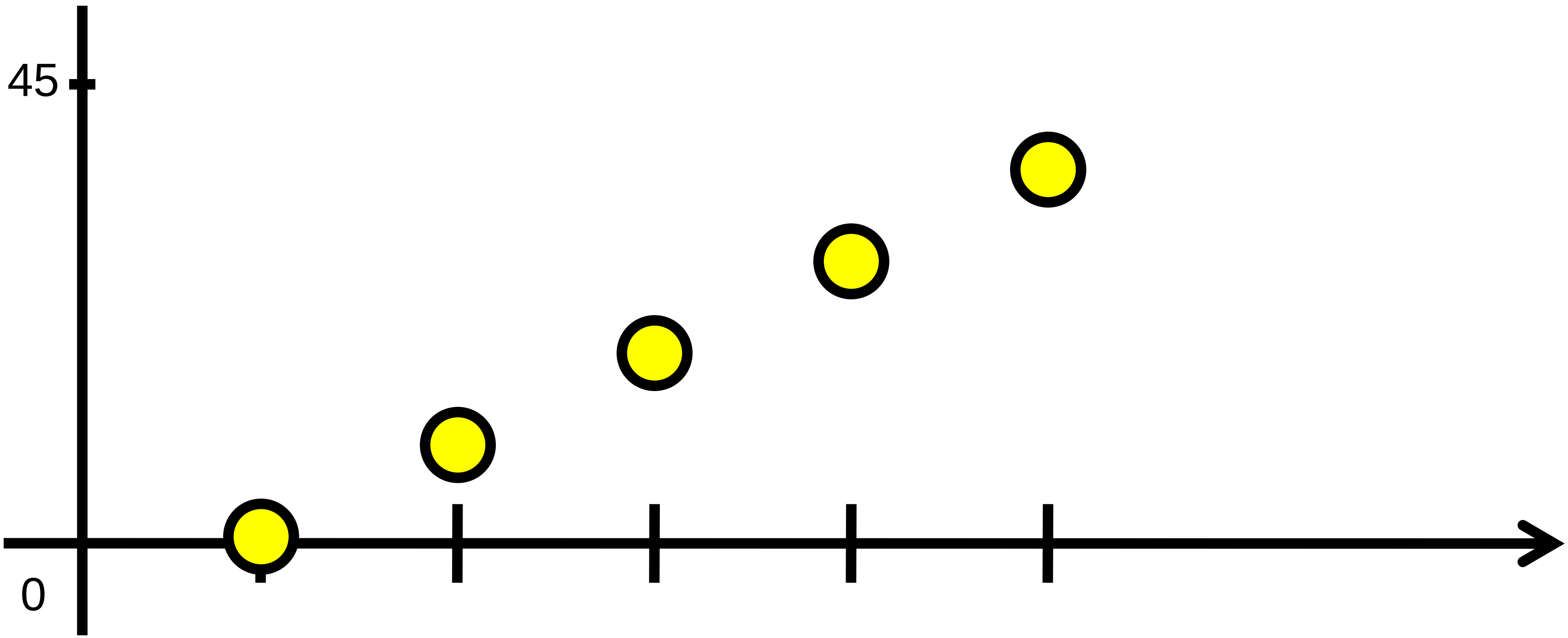
Keyframe Animation

- Define character poses at specific time steps called "keyframes"



Inbetweening ("tweening")

- Computing missing values based on existing surrounding values



Inbetweening ("tweening")

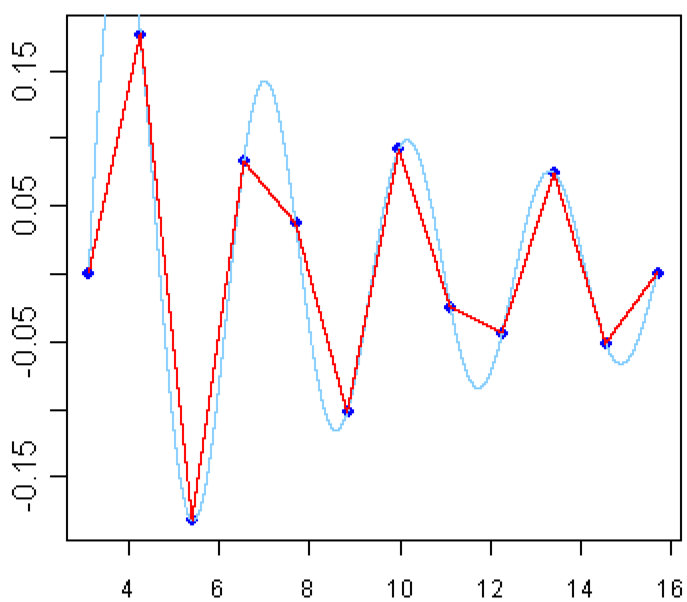
- Linear (constant)

- Ease-in, ease-out

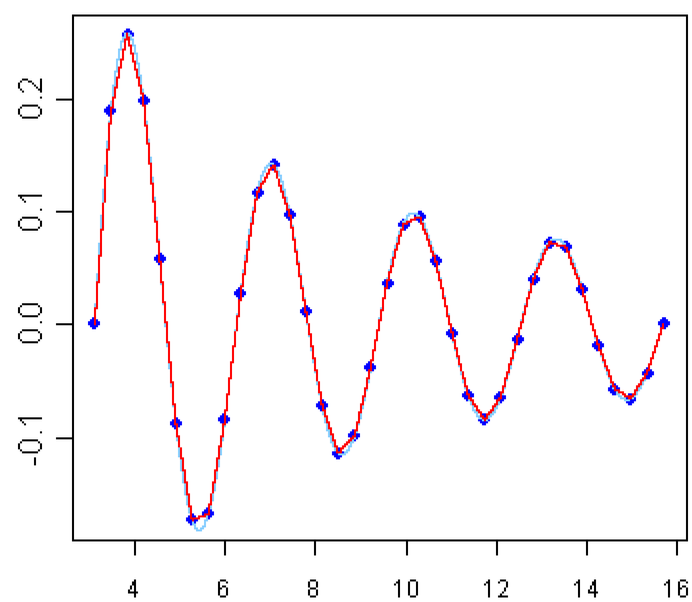
Keyframe Animation

- Inbetweening:
  - Linear interpolation – usually not enough continuity

12 Points



36 Points



Spline Continuity

- How to ensure curves are "smooth"
- Generally, we have three levels of continuity
- C0 - The curves meet
- C0 & C1 - The tangents are shared
- C0 & C1 & C2 - The "speed" is the same

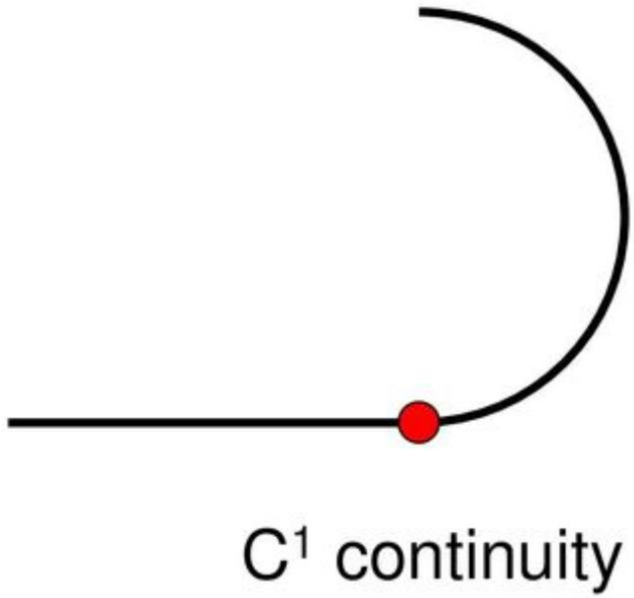
C0 Continuity

- Zero order parametric continuity

C^0 continuity

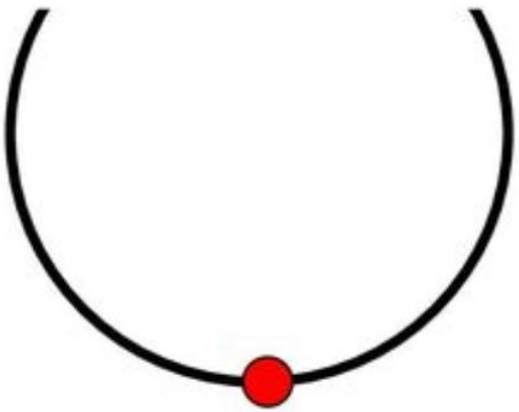
C0 & C1 Continuity

- First order parametric continuity



C0 & C1 & C2 Continuity

- Second order parametric continuity

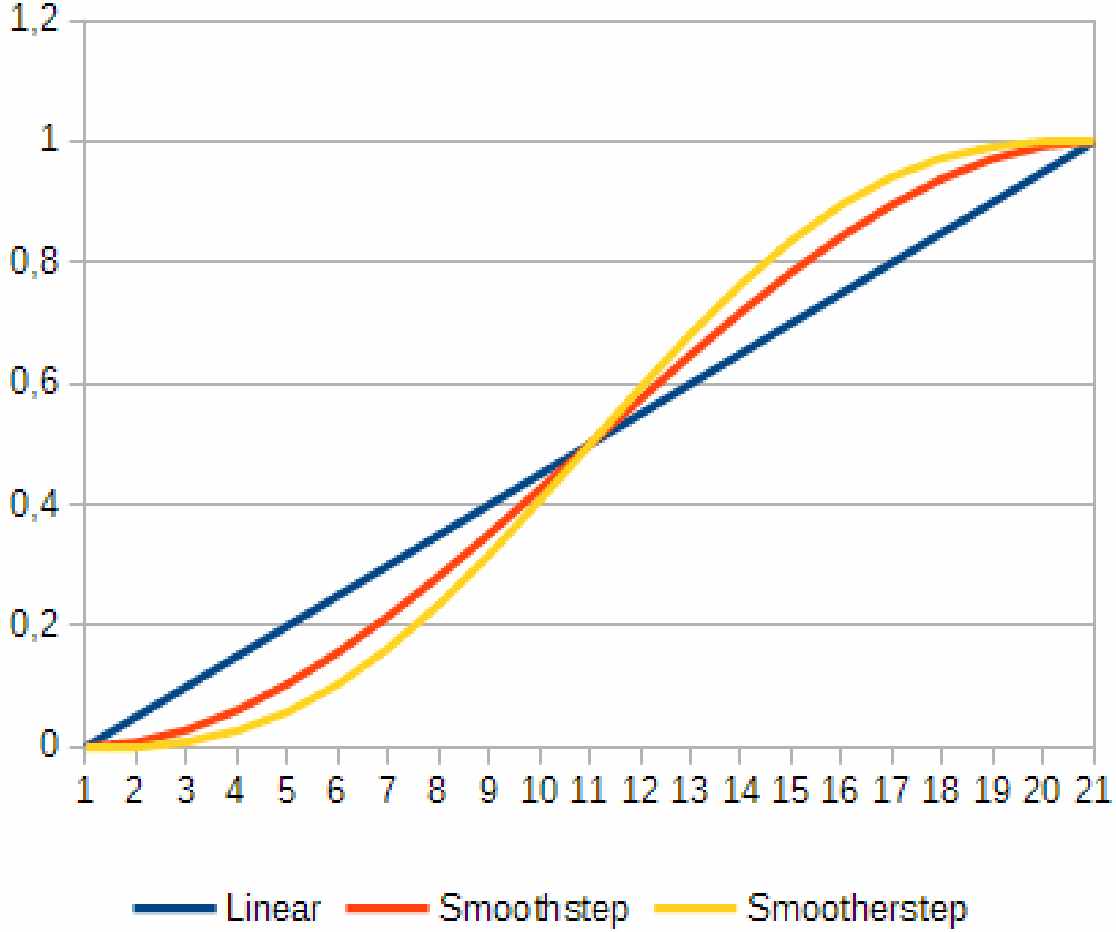


C^2 continuity

Implications for animation

- Linear interpolation is only C0
  - Movement changes instantly at keyframes
  - Very unnatural looking
- We need at least C0 & C1 continuity
  - Hermite interpolation
  - Spline interpolation
  - “Smoothstep” function

Smoothstep



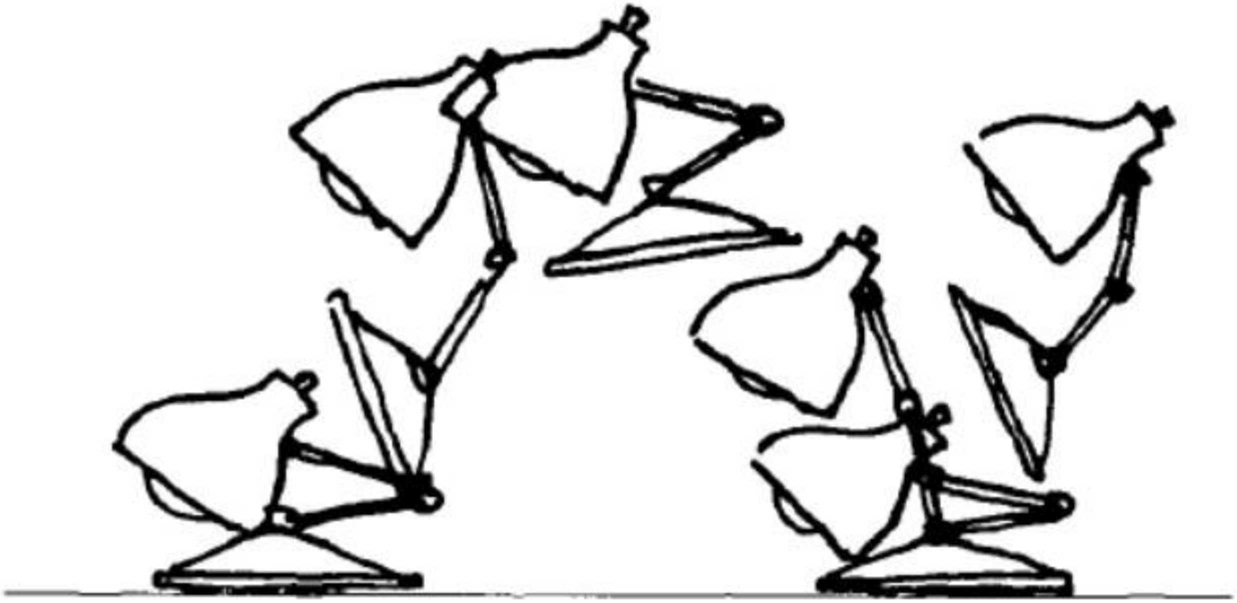
1,2
1
0,8
0,6
0,4
0,2
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
Linear
Smoothstep
Smootherstep

Smoothstep

```
float smootherstep(float edge0, float edge1, float x)
{
  // Scale, and clamp x to 0..1 range
  x = clamp((x - edge0)/(edge1 - edge0), 0.0, 1.0);
  // Evaluate polynomial
  return x*x*x*(x*(x*6 - 15) + 10);
}
```

Keyframe Animation

- Inbetweening:
  - Spline interpolation – may be visually good enough
  - May not follow physical laws

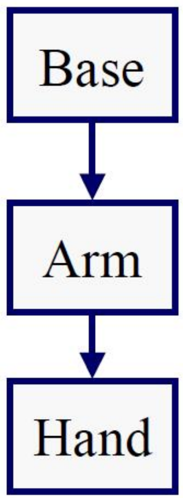


Keyframe Animation

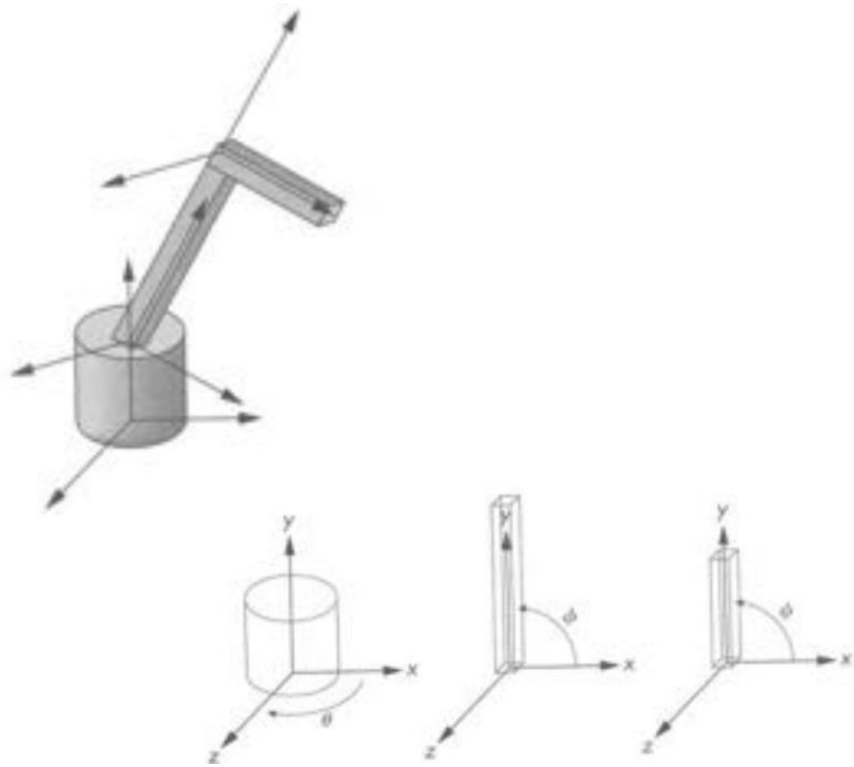
- Inbetweening:
  - Inverse kinematics or dynamics

Articulated Figures

- Character poses described by set of rigid bodies connected by "joints"

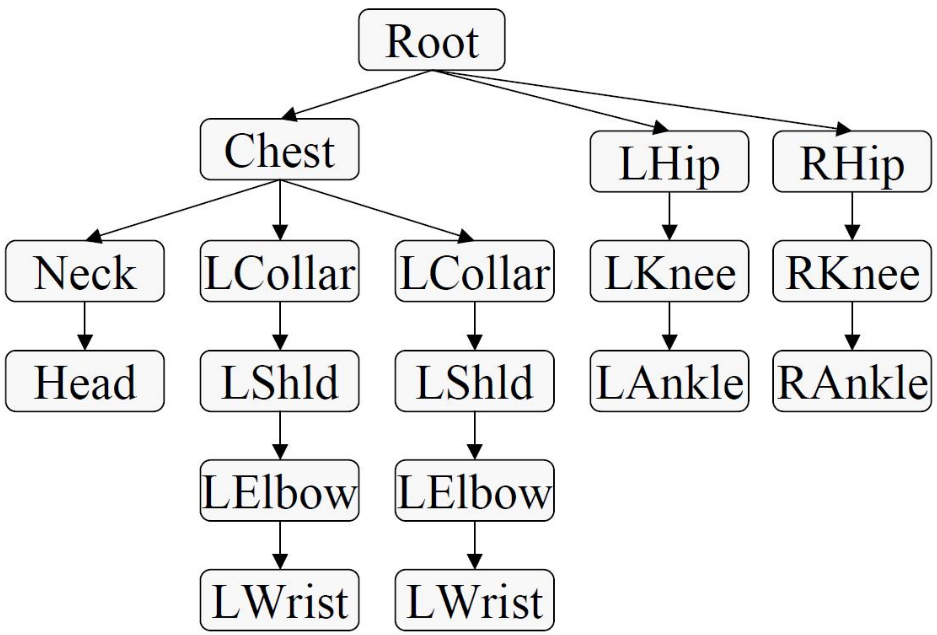


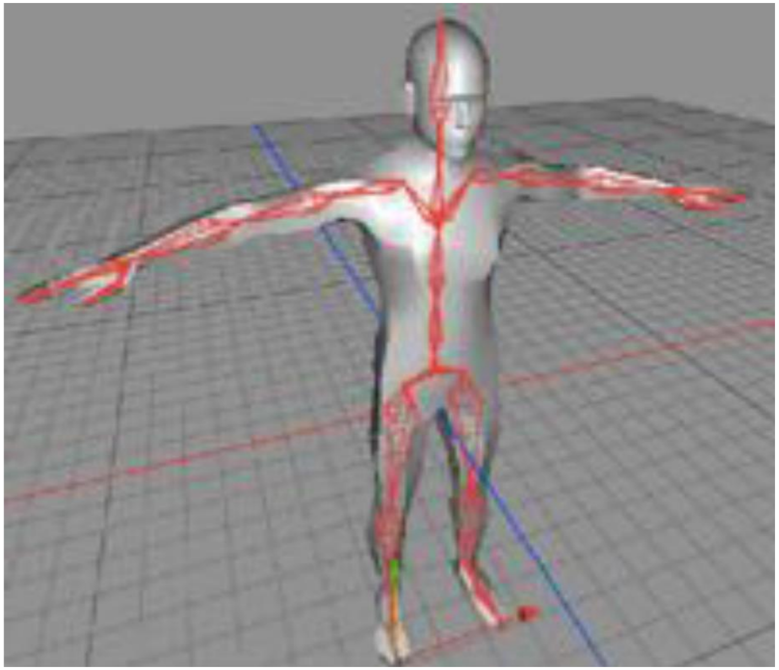
Scene Graph



Articulated Figures

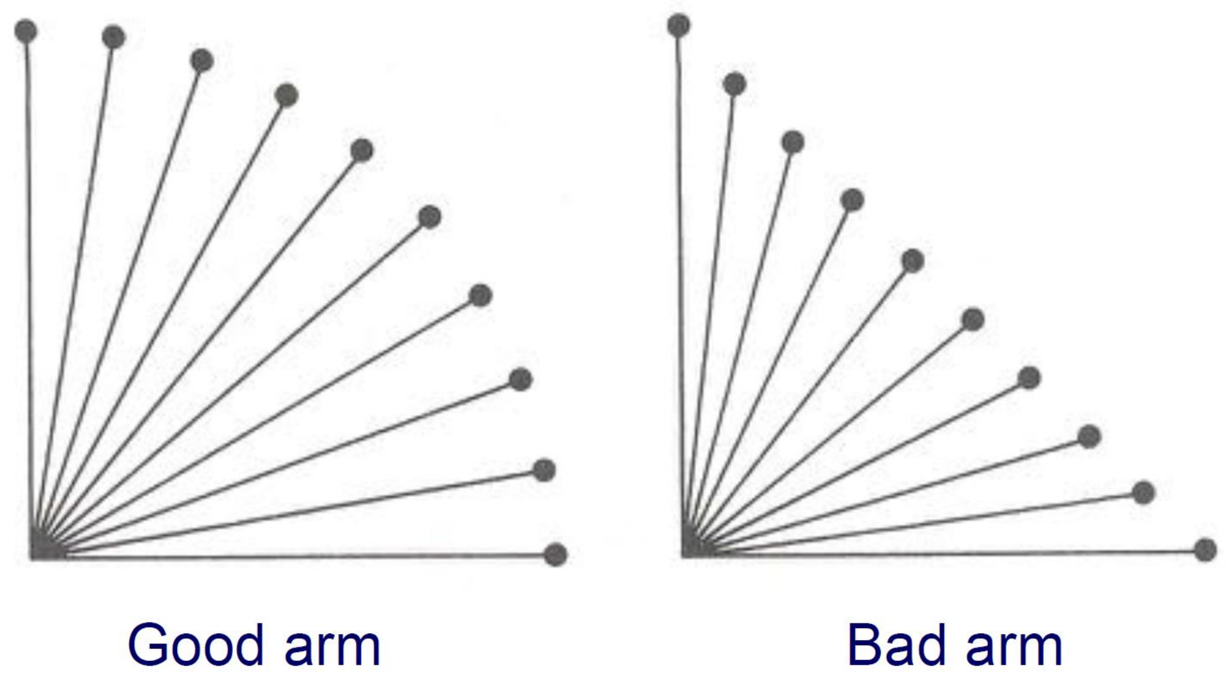
- Well suited for humanoid characters





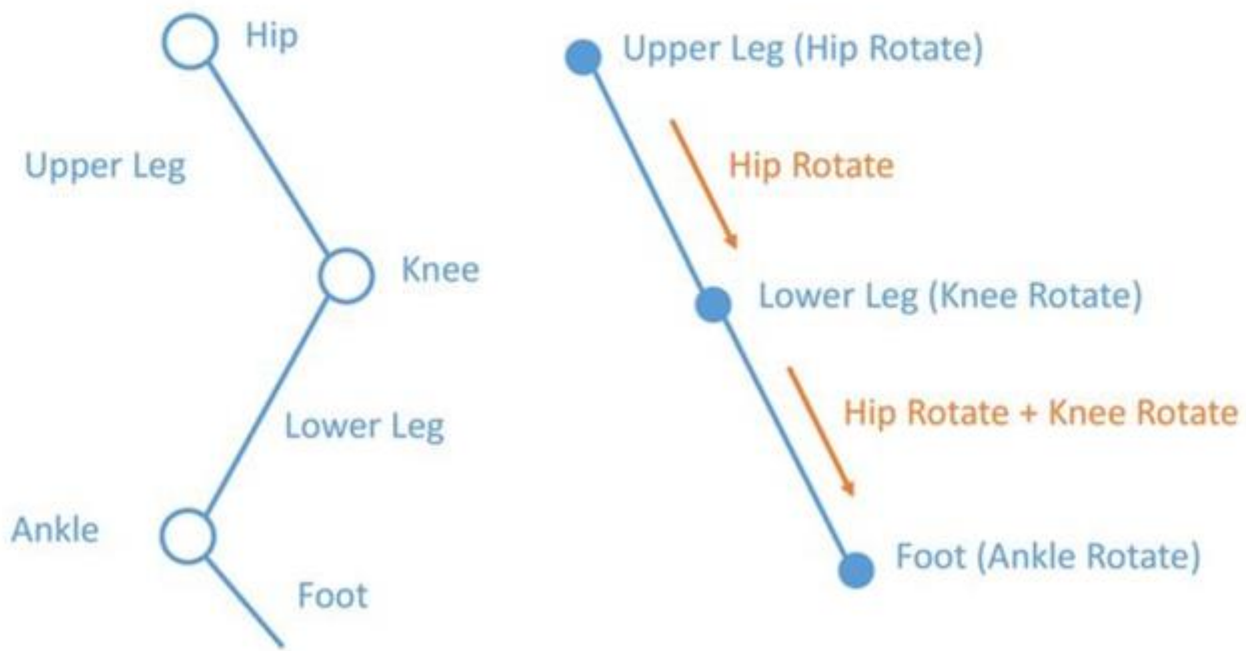
Keyframe Animation

- Inbetweening:
  - Compute angles between keyframes



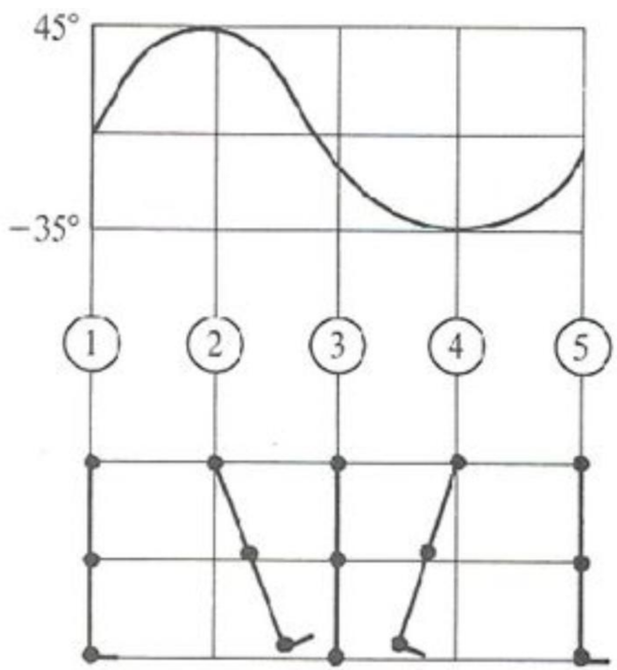
Example: Walk Cycle

- Inbetweening:
  - Compute angles between keyframes



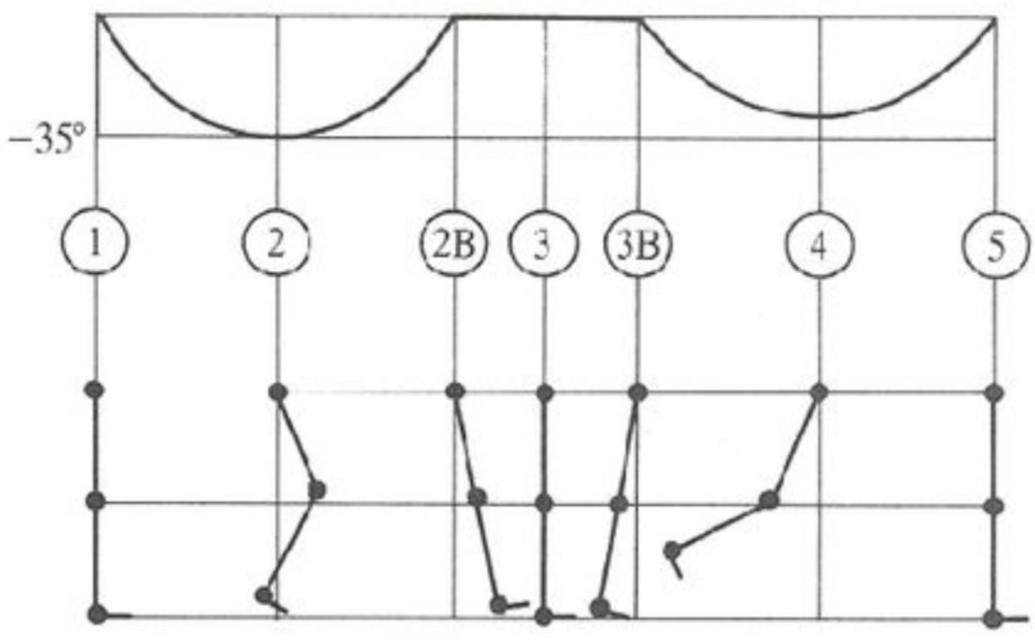
Example: Walk Cycle

- Hip joint orientation



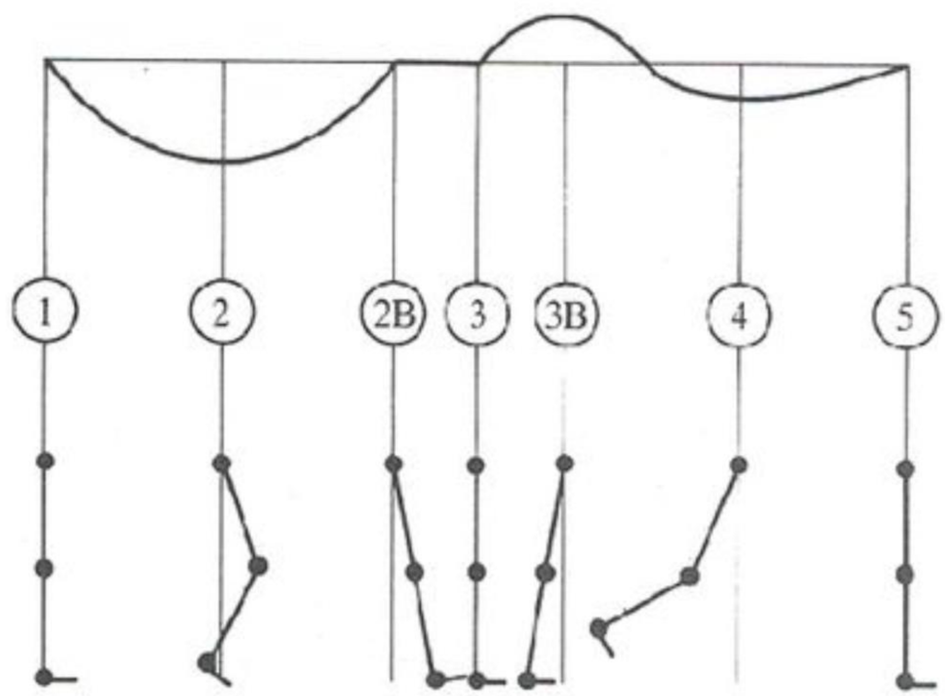
Example: Walk Cycle

- Knee joint orientation



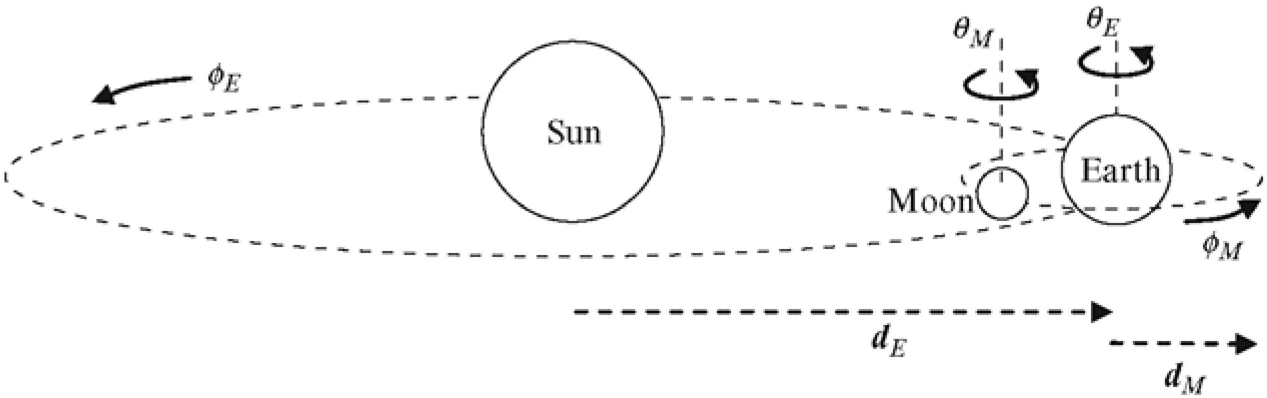
Example: Walk Cycle

- Ankle joint orientation



Animation Hierarchies

- Animate objects in relation to their parent
  - Sun matrix is Ms
  - Earth matrix is MsMe
  - Moon matrix is MsMeMm

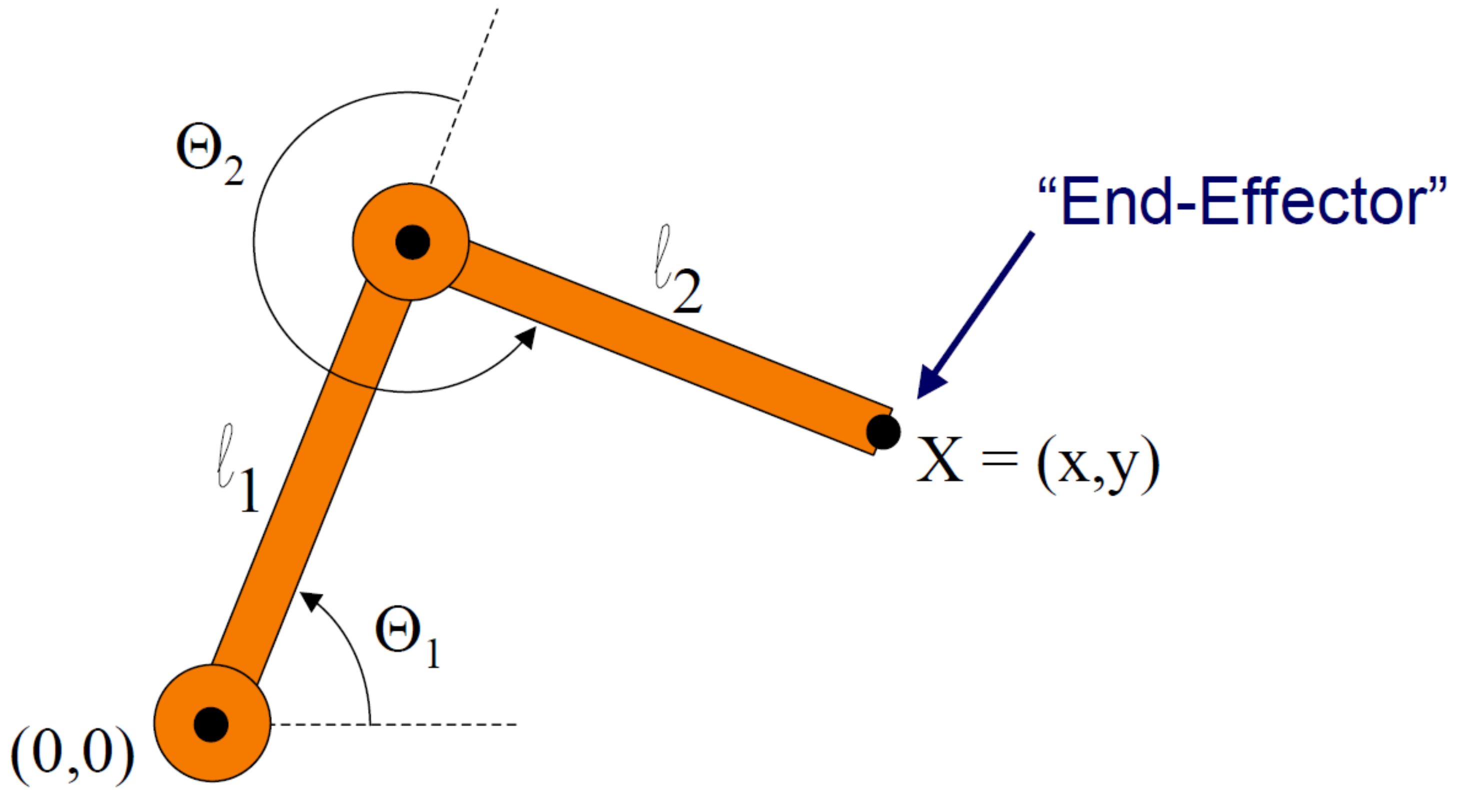


Kinematics and Dynamics

- Kinematics
  - Considers only motion
  - Determined by positions, velocities, accelerations
- Dynamics
  - Considers underlaying forces
  - Capture motion from initial positions and physics

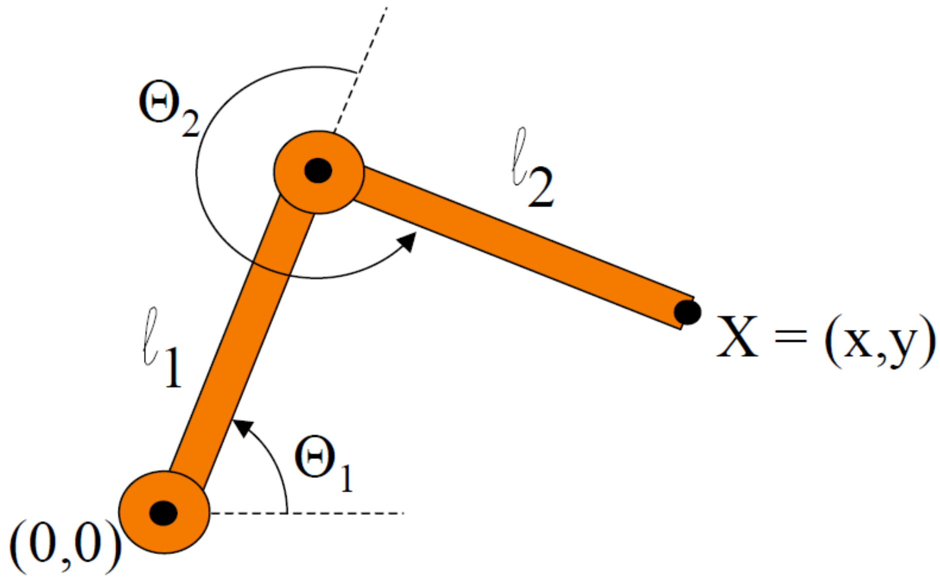
Example: 2-Link Structure

- Two links connected by rotational joints



Forward Kinematics

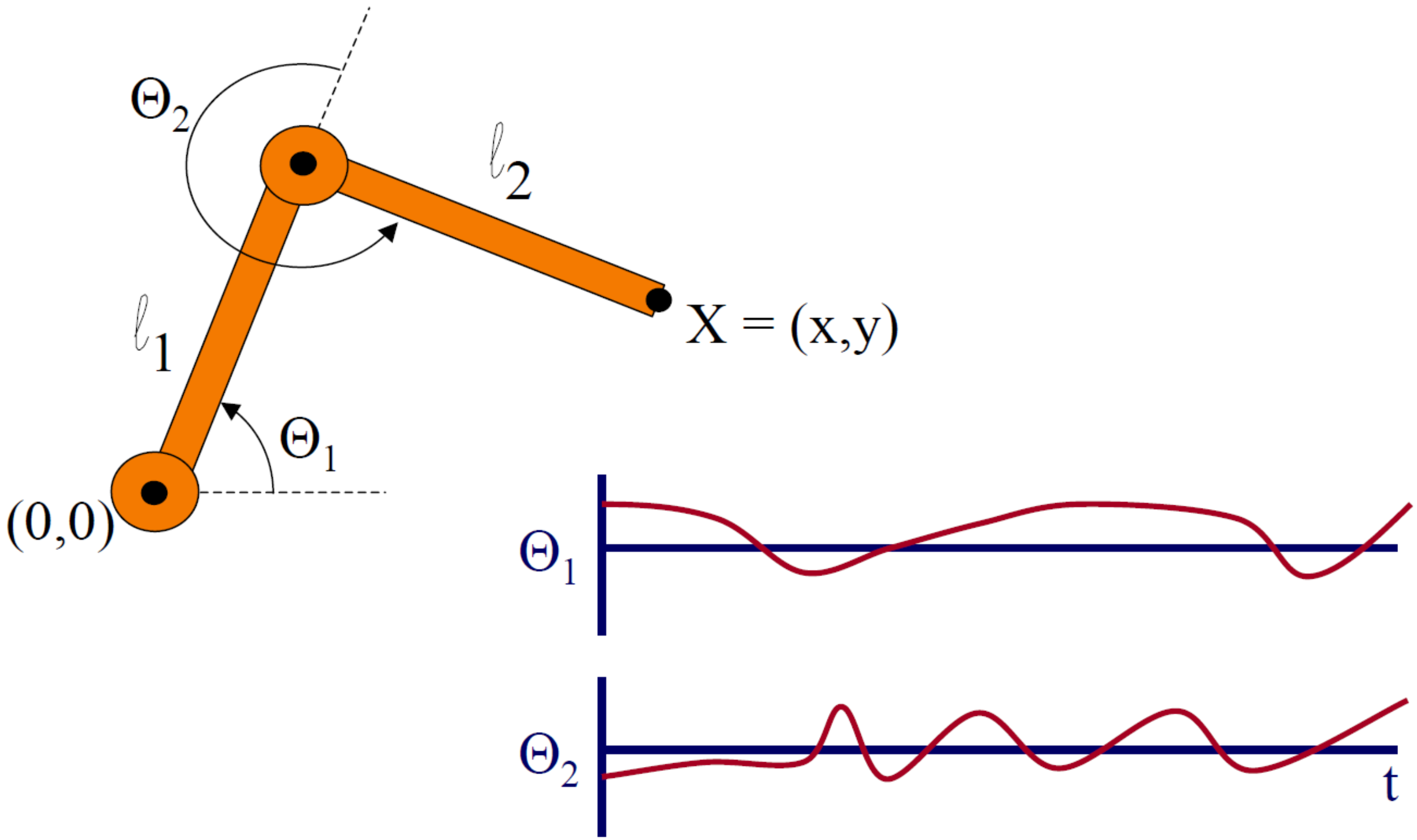
- Animators specifies angles Θ1 and Θ2
- Computer finds position of end effector: X



$$X = (l_1 \cos\Theta_1 + l_2 \cos(\Theta_1 + \Theta_2), l_1 \sin\Theta_1 + l_2 \sin(\Theta_1 + \Theta_2))$$

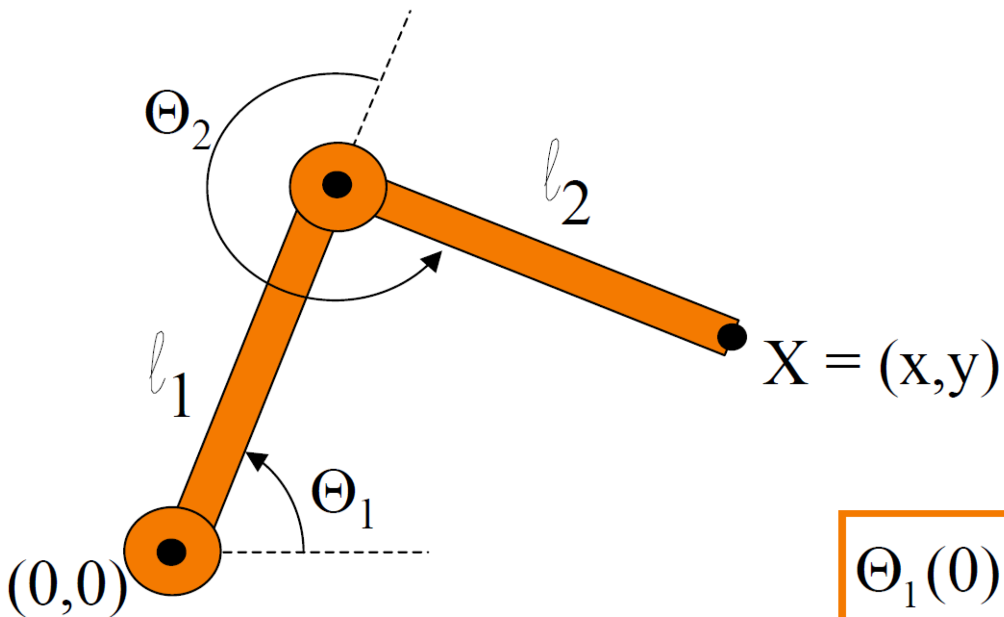
Forward Kinematics

- Joint motion can be specified by spline curves



Forward Kinematics

- Joint motions can be specified by initial conditions

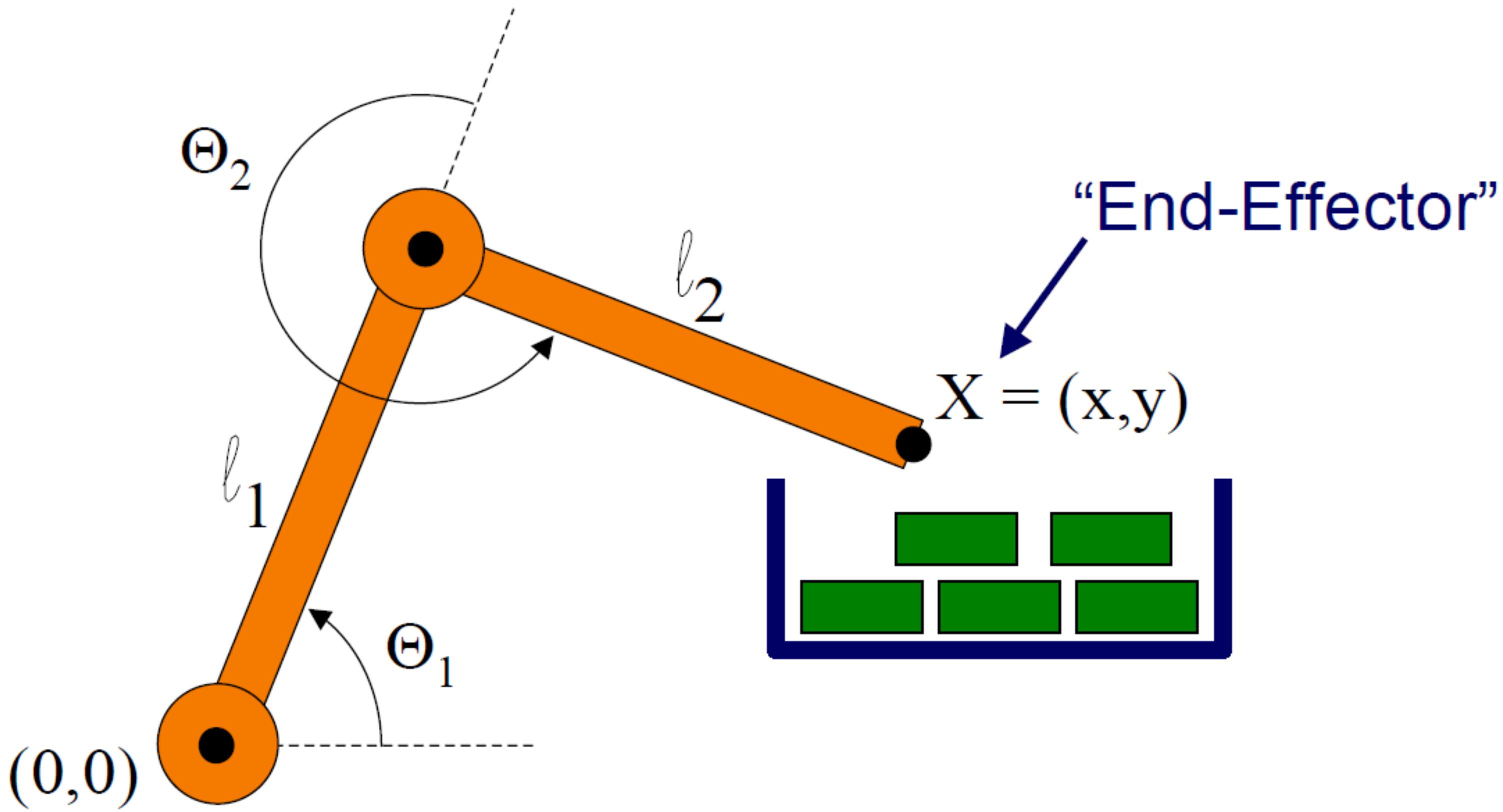


$$\Theta_1(0) = 60^\circ \qquad \Theta_2(0) = 250^\circ$$

$$\frac{d\Theta_1}{dt} = 1.2 \qquad \frac{d\Theta_2}{dt} = -0.1$$

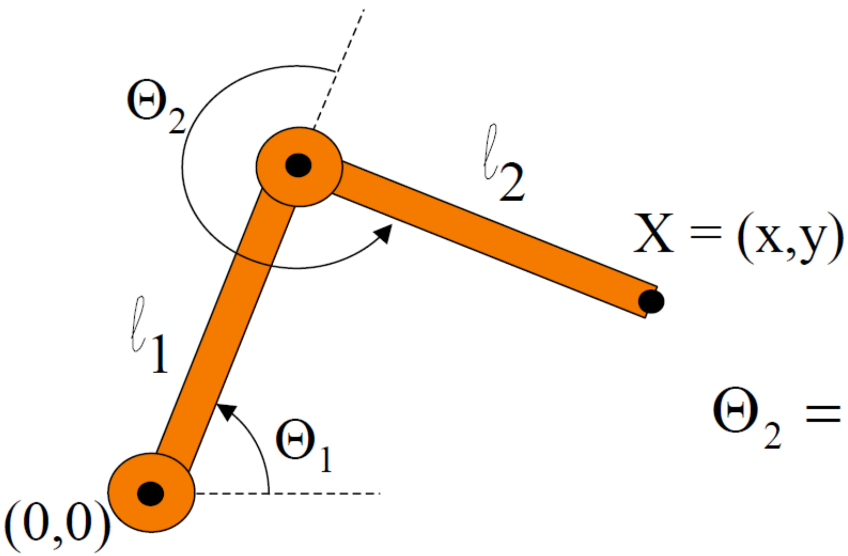
Example: 2-Link Structure

- What if animator knows position of "end effector"



Inverse Kinematics

- Animator specifies end effector positions: X
- Computer finds joint angles Θ1 and Θ2

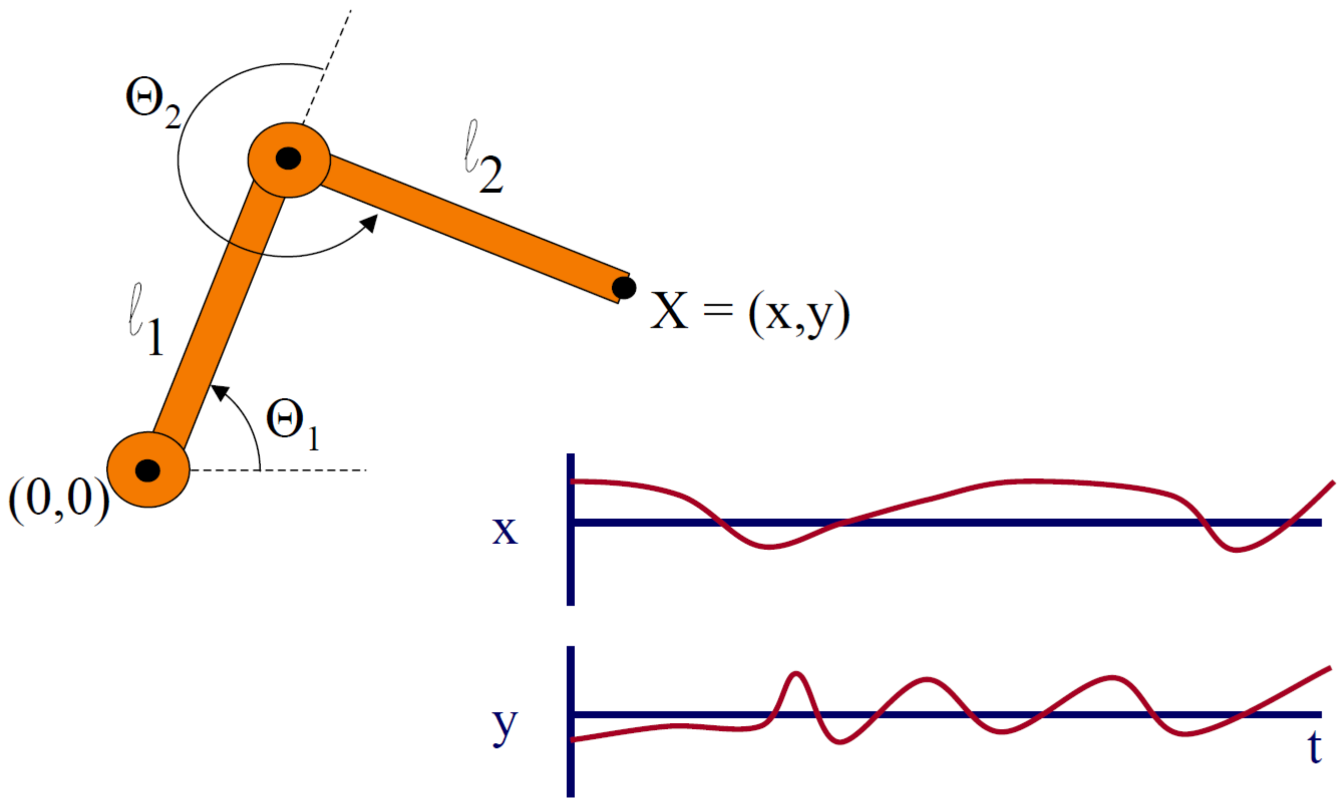


$$\Theta_2 = \cos^{-1}\left(\frac{x^2 + x^2 - l_1{}^2 - l_2{}^2}{2l_1 l_2}\right)$$

$$\Theta_1 = \frac{-(l_2 \sin(\Theta_2)x + (l_1 + l_2 \cos(\Theta_2))y}{(l_2 \sin(\Theta_2))y + (l_1 + l_2 \cos(\Theta_2))x}$$

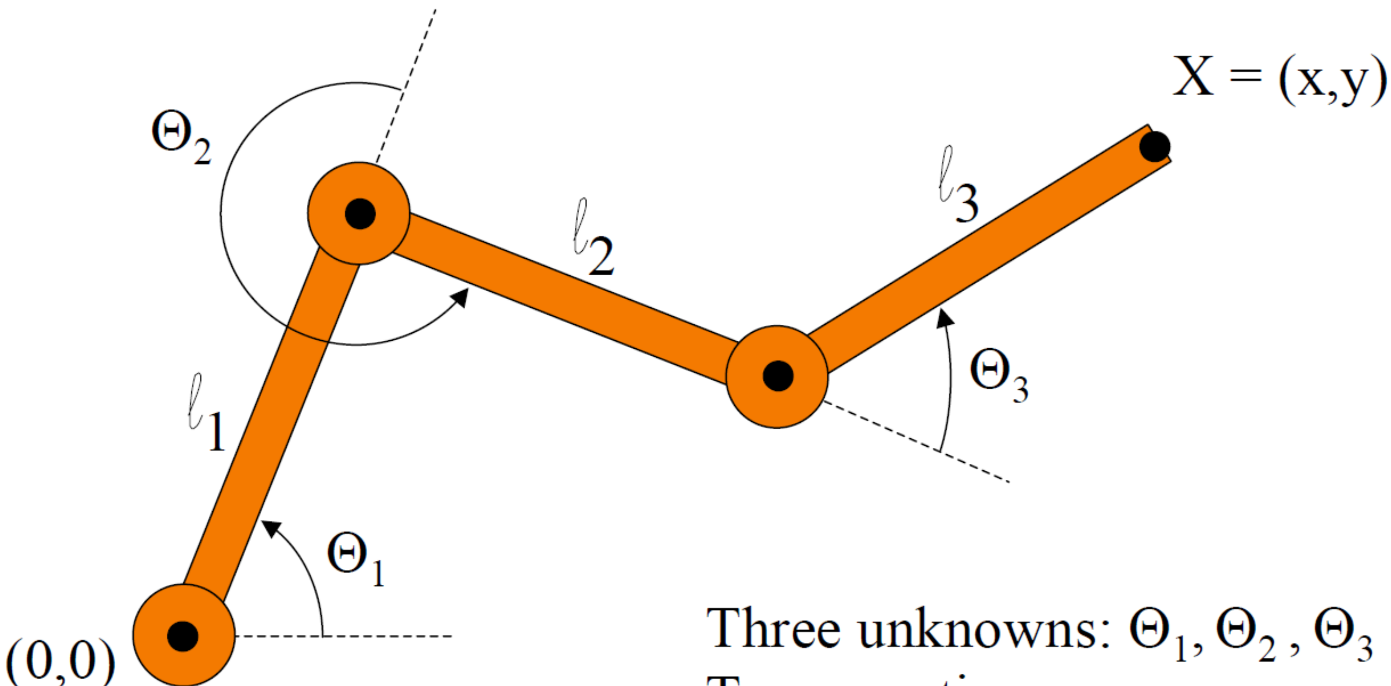
Inverse Kinematics

- End-effector positions can be specified by splines



Inverse Kinematics

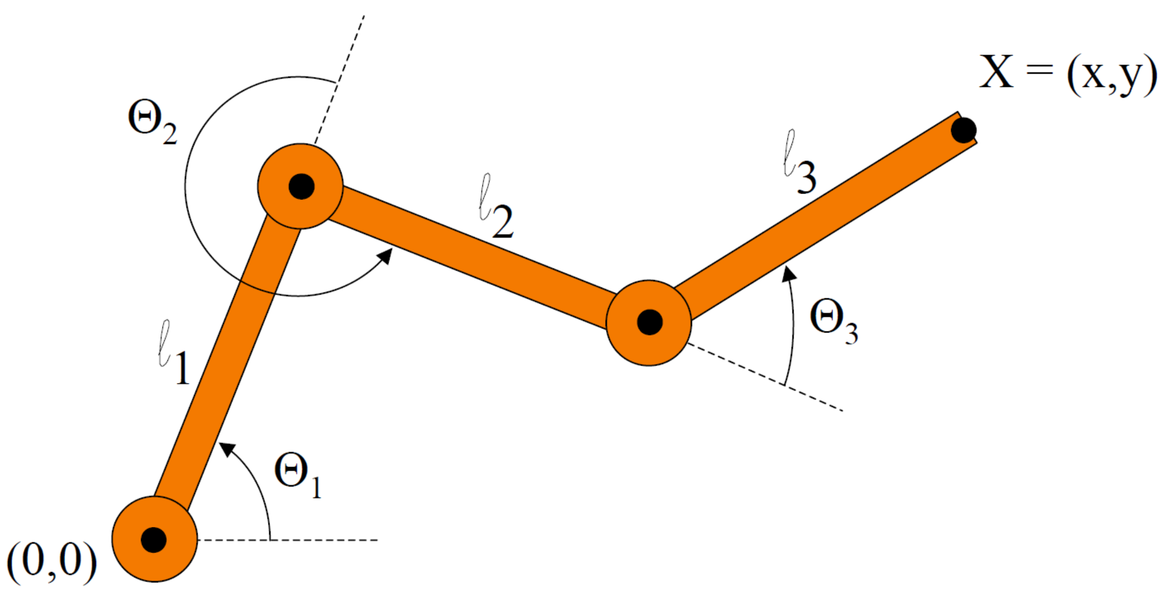
- Problem with more complex structures
  - System with equation is usually under-defined
  - Multiple solutions



Three unknowns: $\Theta_1, \Theta_2, \Theta_3$
Two equations: x, y

Inverse Kinematics

- Solution for more complex structures
  - Find best solution (eg. minimize energy in motion)
  - Non-linear optimization



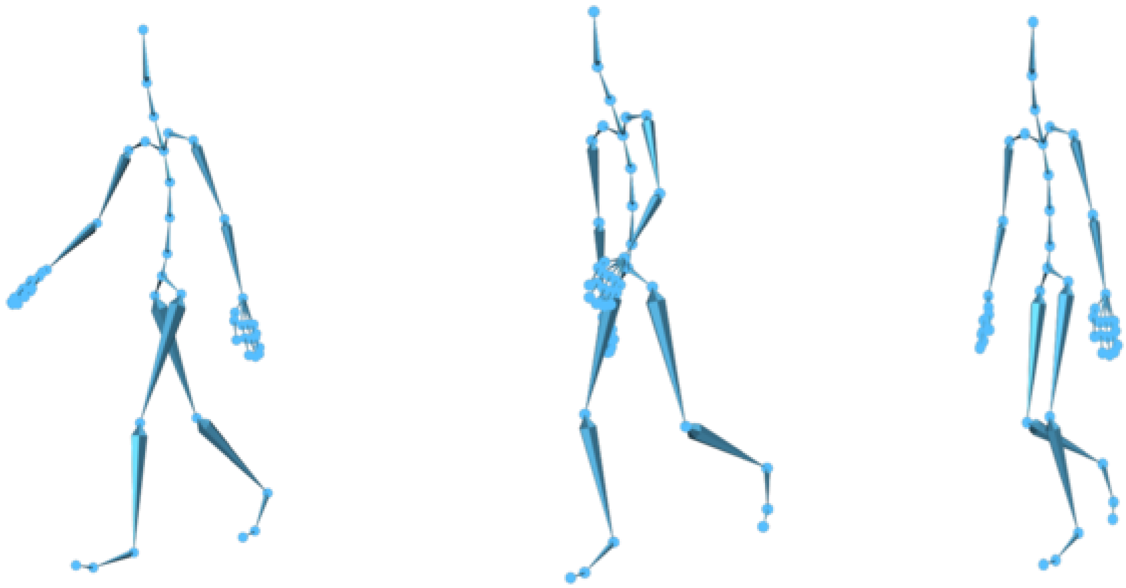
Inverse Kinematics

- Forward Kinematics
  - Specify conditions (joint angles)
  - Compute conditions of end effectors
- Inverse Kinematics
  - “Goal-directed” motion
  - Specify goal positions of end effectors
  - Compute conditions required to achieve goal

Character Animation
(Linear Blend Skinning)
(Skeletal Animation)

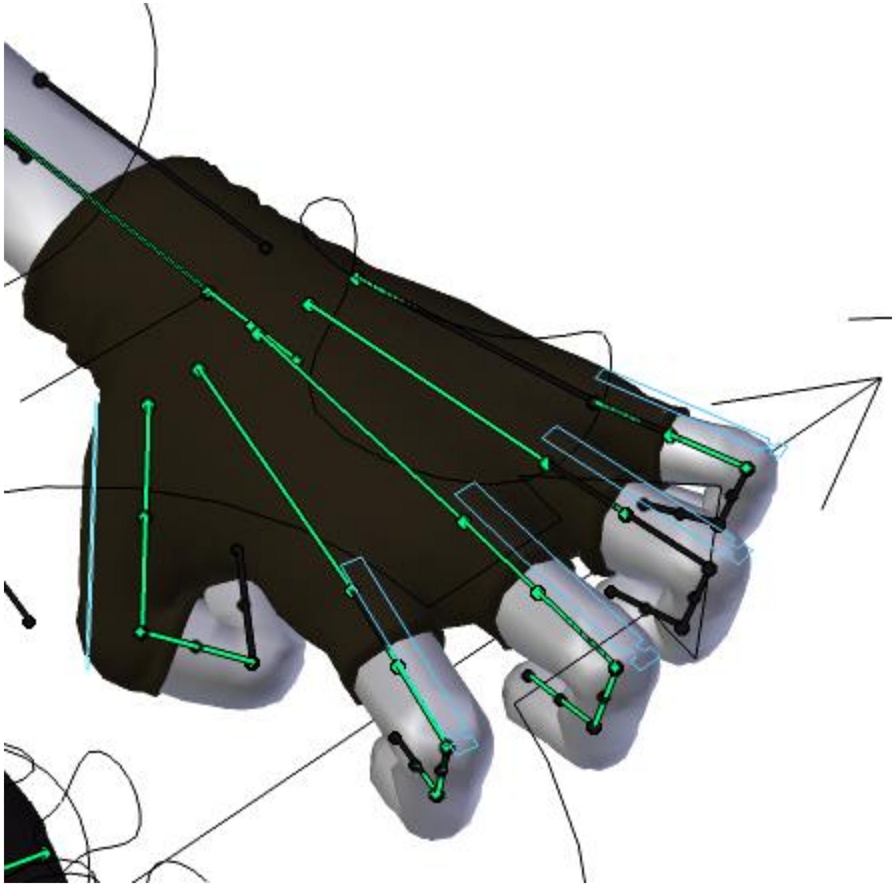
Skeletal Animation

- Hierarchical graph structure called Skeleton
  - Nodes and edges (bones)



Skeletal Animation

- Graph structure can be disconnected in space



Real-time skeletal skinning

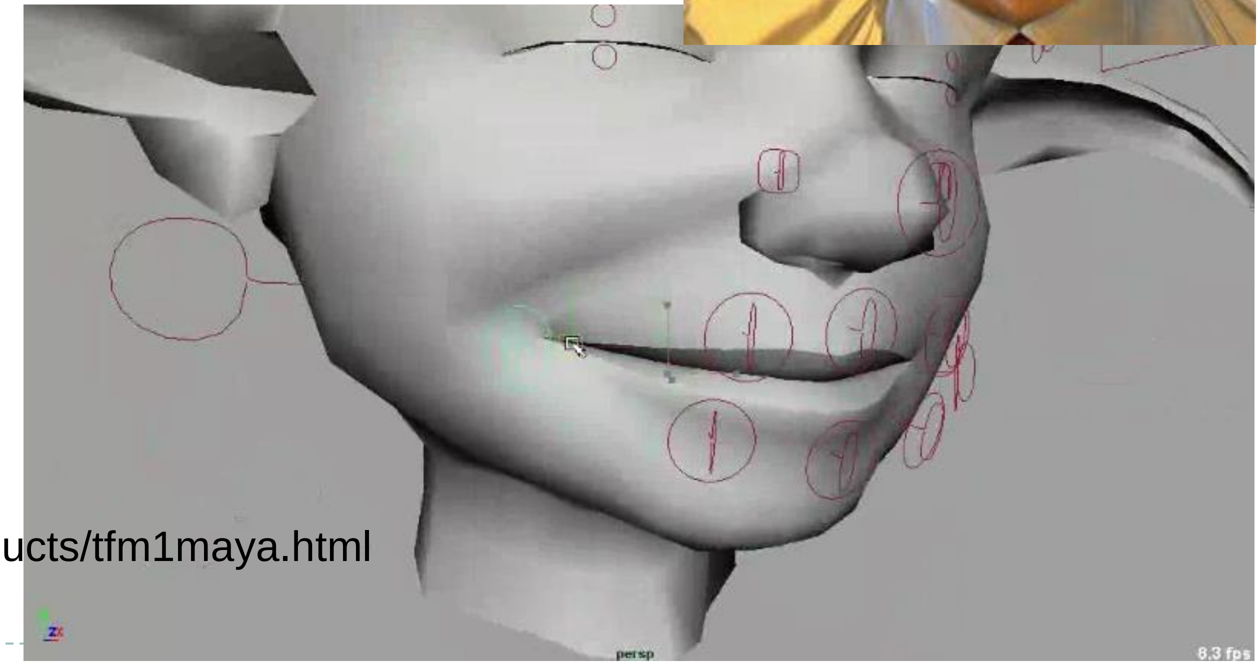
Real-time Skeletal Skinning with Optimized Centers of Rotation

Binh Huy Le
Jessica K. Hodgins

https://www.youtube.com/watch?v=DflfcQiC2oA

Facial animation

- Facial expressions
- Lips to speech synchronization
- Controllers
- Skinning
- Morphing



http://www.anzovin.com/products/tfm1maya.html

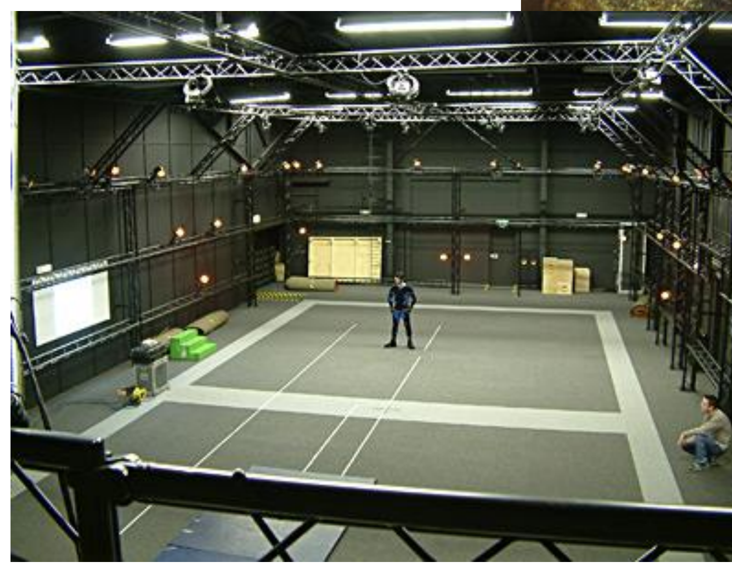
Reusable animation

- One skeleton – different models

http://www.studiopendulum.com/alterego/

Motion capture

- Markers on actor's body
- Optical / magnetic sensors
- 3D reconstruction of markers' position
- Motion mapping
  to virtual character



Outline

- Principles of animation
- Keyframe animation
- Articulated figures
- Animation Hierarchies
- Kinematics
- **Dynamics**

Dynamics

- Dynamics
  - Considers underlaying forces
  - Capture motion from initial positions and physics
  - Simulation of Physics insures realism of motion

Procedural animation

- Programmed rules for changing parameters of the animated objects
- E.g. according to music, physics, psychology

Physically based animation

- Rigid bodies
  - No geometry deformation
  - Collision response

- Soft bodies
  - Allow for deformation
  - Energy damping

Animation construction

- Set body properties
  - Mass, elasticity, friction, …
- Set physical rules
  - Gravity, collisions, wind, …
- Set initial state
  - Position, velocity, direction, …
- Set constraints

- Run simulation / animation

Spacetime constraints

- Animator specifies constraints:
  - What the character's physical structure is
    - e.g. articulated figure
  - What the character has to do
    - e.g. jump from here to there in time *t*
  - What other physical structures are present
    - e.g. floor to push off and land
  - How the motion should be performed
    - e.g. minimize energy

Spacetime constraints

- Computer finds the "best" physical motion satisfying constraints
- Example: Simulate objects using 2nd Newtons law
- F = ma
- Ordinary differential equation (ODE)
- Numerically solved using Euler's method
- Use discrete time steps

Euler Integration

- Euler Integration

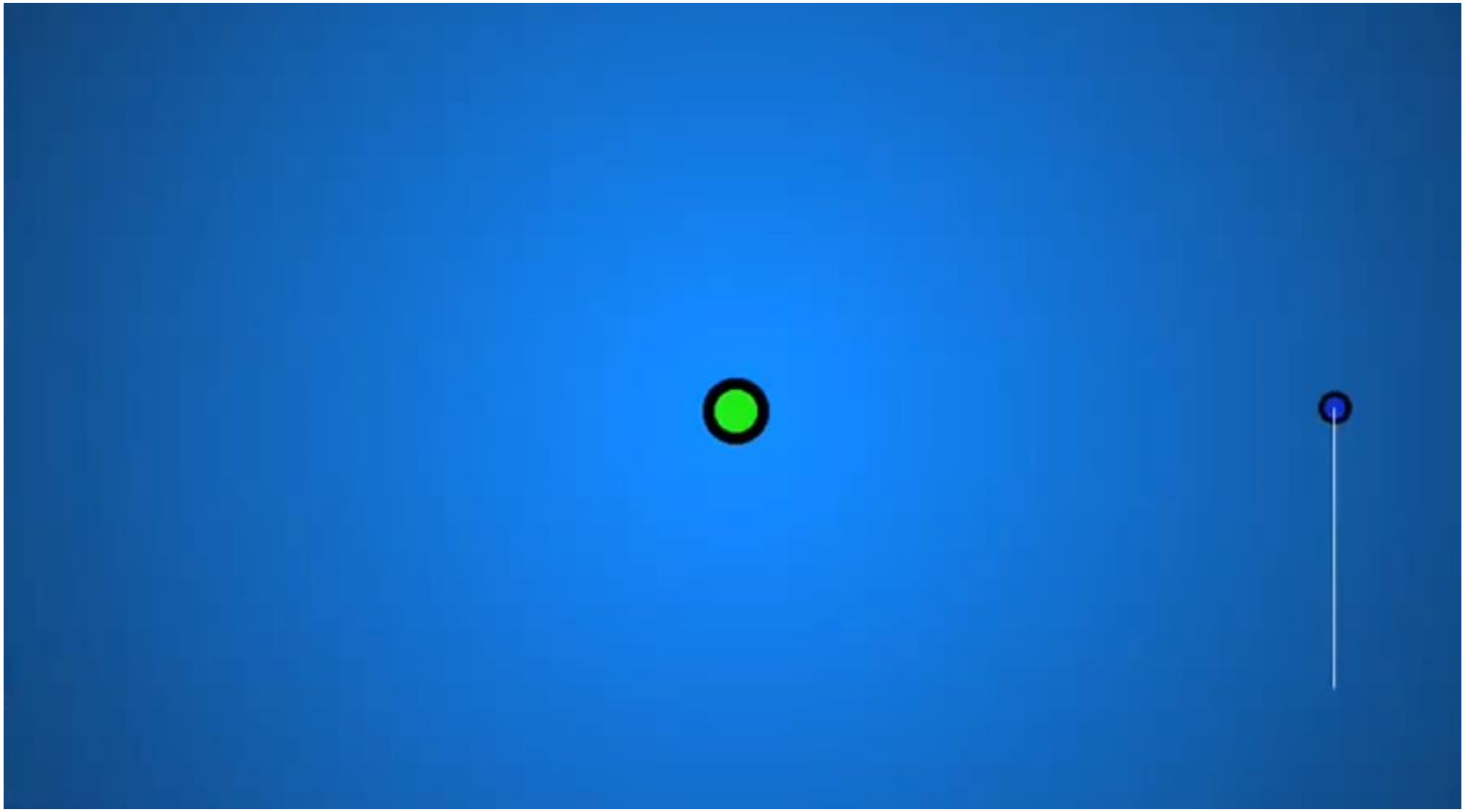
```
Object::Update(float dt){
  /* Constant acceleration: gravity */
  a = vec3(0, 0, -9.81);
  /* New, velocity */
  v = v + a * dt;
  /* New, position */
  p = p + v * dt;
}
```

Euler Integration

- External forces can influence motion

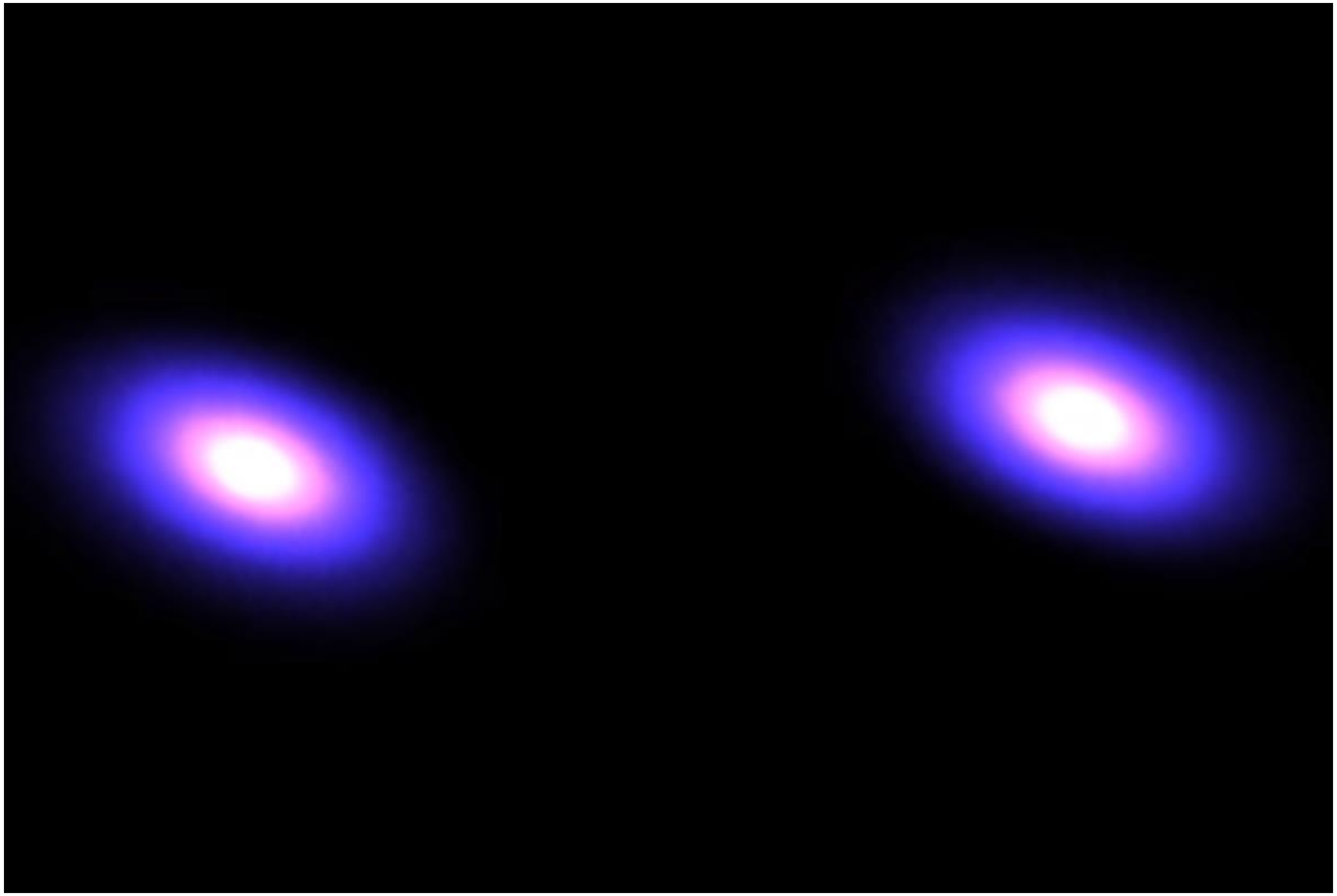
```
Object::Update(float dt)
{
  /* Use mass and external forces */
  float m = this->Mass();
  F = sumExternalForces(this);
  /* Compute acceleration */
  a = F/m;
  /* New, velocity */
  v = v + a * dt;
  /* New, position */
  p = p + v * dt;
}
```

Gravity simulation



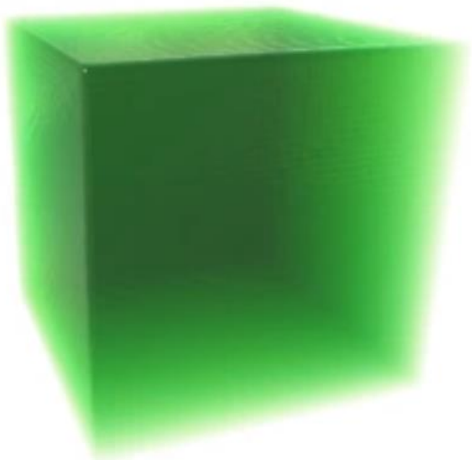
https://www.youtube.com/watch?v=ztwkXq4Hj7Q

N-body simulation



https://www.youtube.com/watch?v=ua7YlN4eL_w

Fluid simulation



https://www.youtube.com/watch?v=r17UOMZJbGs

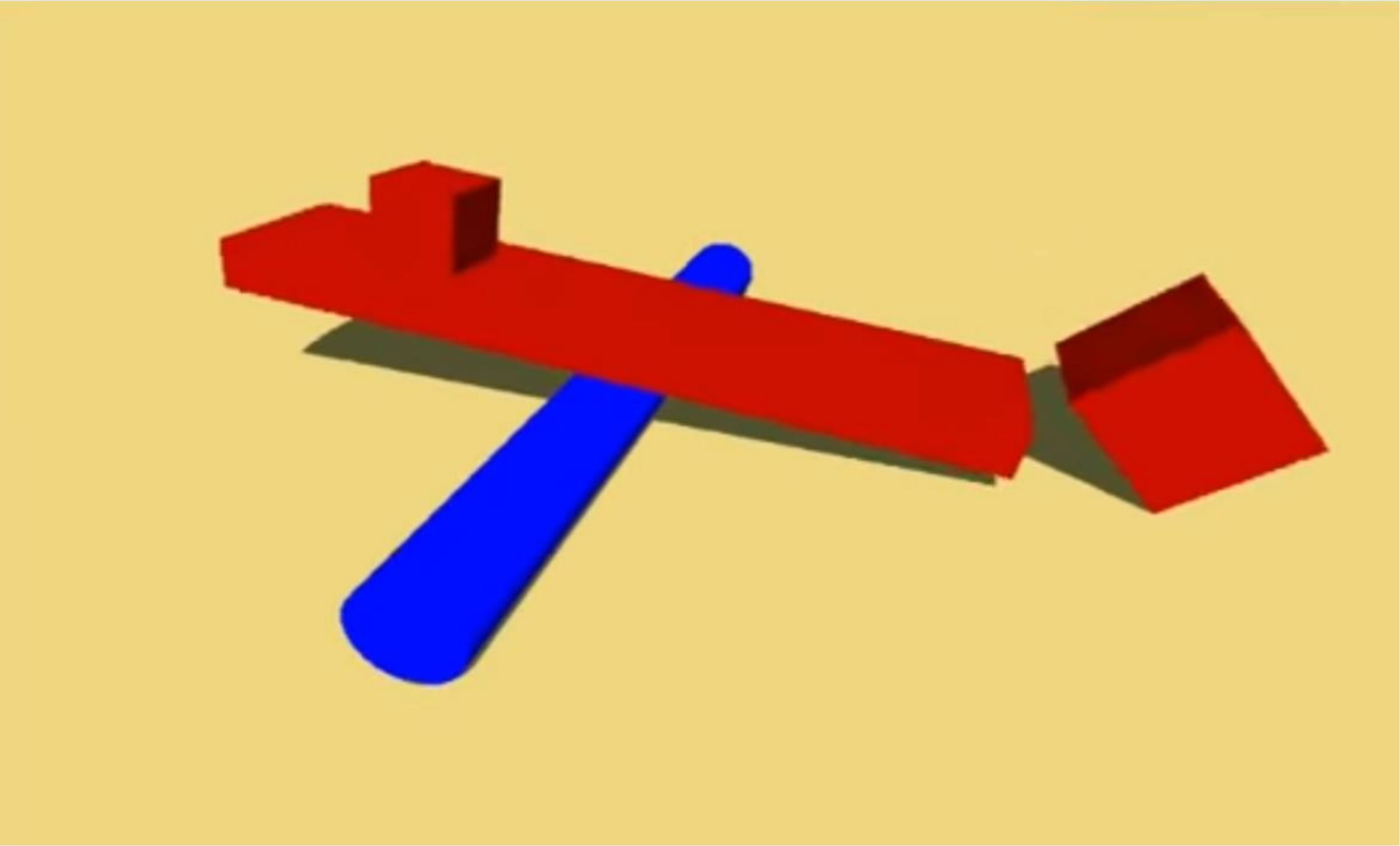
Smoke and Dust

https://www.youtube.com/watch?v=RuZQpWo9Qhs

Rigid-Body Dynamics

- Assume objects are rigid
- Preserve and calculate angular momentum
- Calculate collisions based on geometry
- Define center of mass

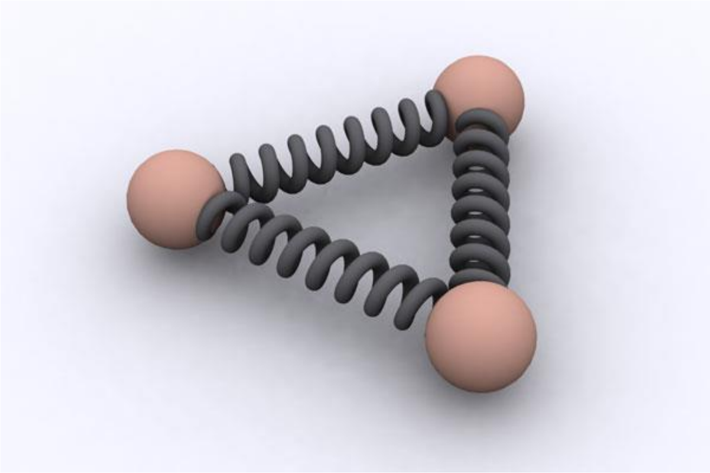
Rigid-Body Dynamics



https://www.youtube.com/watch?v=dvXBstJah5s

Soft-Body Dynamics

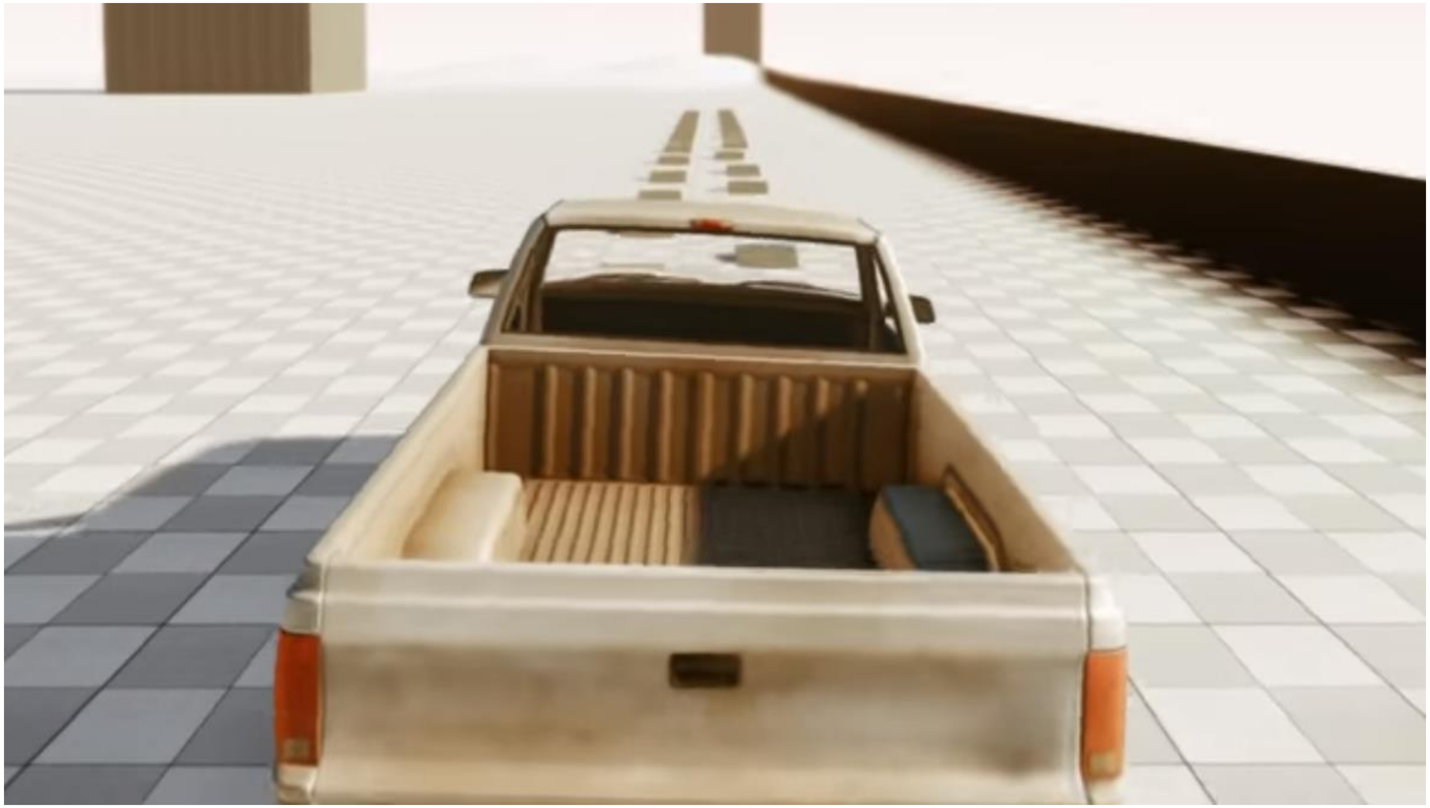
- Model objects using linked particles
- Usually using spring/mass models
- Polygon edges can represent springs
- Vertices are simulated using particles with mass



Cloth simulation

https://www.youtube.com/watch?v=M2XuQSZ-8h4

Soft-Body simulation



https://www.youtube.com/watch?v=KppTmsNFneg

AI controlled

https://www.youtube.com/watch?v=IQEx56O73b8

Summary

- Advantages
  - Animation is emergent
  - No need to specify animation details
  - Animations can very between runs
- Challenges
  - Accuracy and stability of simulation
  - Specification of constraints

Next Lecture

Raycasting

Acknowledgements

- Thanks to all the people, whose work is shown here and whose slides were used as a material for creation of these slides:

Matej Novotný, GSVM lectures at FMFI UK

Peter Drahoš, PPGSO lectures at FIIT STU

Output of all the publications and great team work

Very best data from 3D cameras

Questions ?!

Skeletex
RESEARCH

www.skeletex.xyz

madaras@skeletex.xyz

martin.madaras@fmph.uniba.sk