



Kinematics

Tom Funkhouser
Princeton University
COS 426, Fall 2006

Computer Animation



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



Pixar



University of Illinois

Computer Animation



Pixar

Computer Animation



- Animation pipeline
 - 3D modeling
 - Motion specification
 - Motion simulation
 - Shading, lighting, & rendering
 - Postprocessing

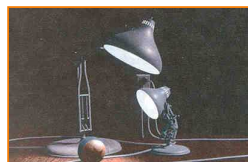


Pixar

Computer Animation



- Animation pipeline
 - 3D modeling
 - Motion specification
 - Motion simulation
 - Shading, lighting, & rendering
 - Postprocessing

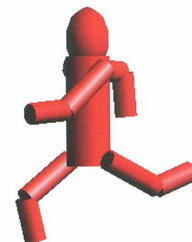


Pixar

Outline



- Articulated figures
- Keyframe animation
- Kinematics
- Dynamics
- Guidelines

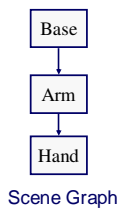


Angel Plate 1

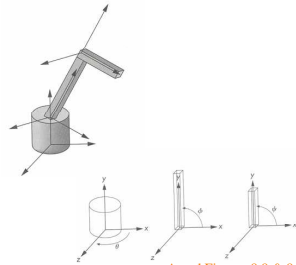
Articulated Figures



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



Scene Graph

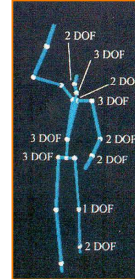
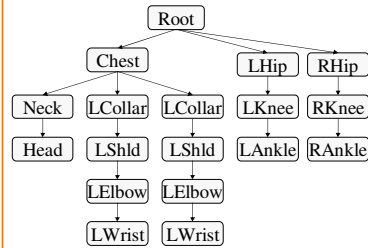


Angel Figures 8.8 & 8.9

Articulated Figures



- Well-suited for humanoid characters

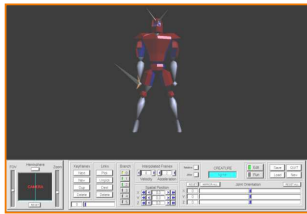


Rose et al. '96

Articulated Figures



- Joints provide handles for moving articulated figure

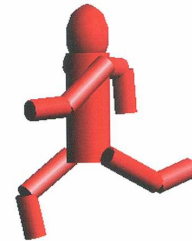


Mike Marr, COS 426, Princeton University, 1995

Outline



- Articulated figures
- Ø Keyframe animation
- Kinematics
- Dynamics
- Guidelines

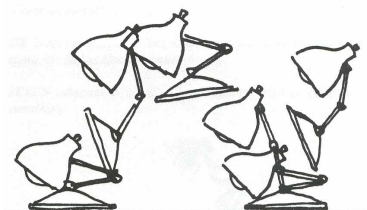


Angel Plate 1

Keyframe Animation



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

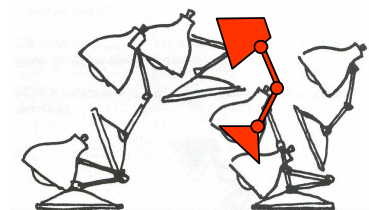


Lasseter '87

Keyframe Animation



- Interpolate variables describing keyframes to determine poses for character "in-between"

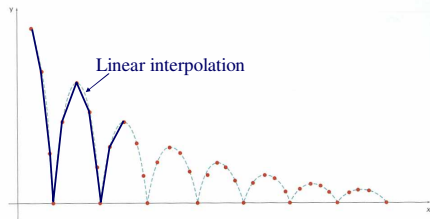


Lasseter '87

Keyframe Animation



- Inbetweening:
 - Linear interpolation - usually not enough continuity

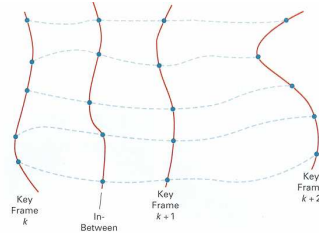


H&B Figure 16.16

Keyframe Animation



- Inbetweening:
 - Spline interpolation - maybe good enough

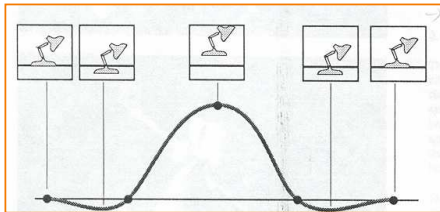


H&B Figure 16.11

Keyframe Animation



- Inbetweening:
 - Cubic spline interpolation - maybe good enough
 - » May not follow physical laws

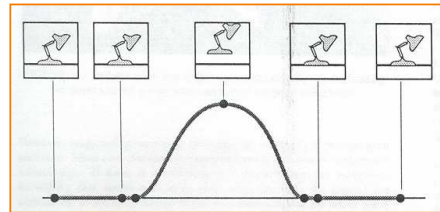


Lasseter '87

Keyframe Animation



- Inbetweening:
 - Cubic spline interpolation - maybe good enough
 - » May not follow physical laws

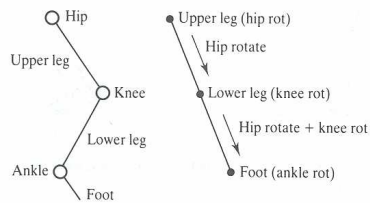


Lasseter '87

Example: Walk Cycle



- Articulated figure:

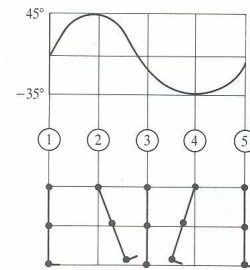


Watt & Watt

Example: Walk Cycle



- Hip joint orientation:

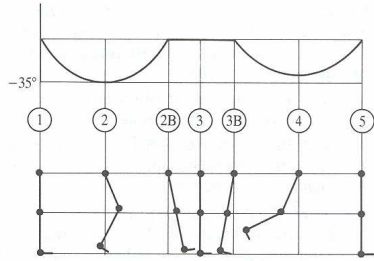


Watt & Watt

Example: Walk Cycle



- Knee joint orientation:

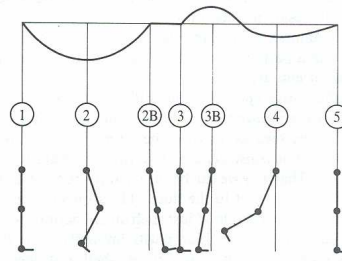


Watt & Watt

Example: Walk Cycle

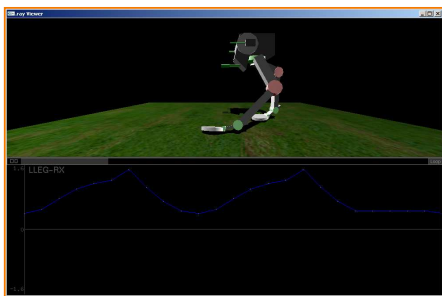


- Ankle joint orientation:



Watt & Watt

Example: Robot



Mihai Parparita, COS 426, Princeton University, 2003

Example: Ice Skating



(Mao Chen, Zaijin Guan, Zhiyan Liu, Xiaohu Qie, CS426, Fall98, Princeton University)

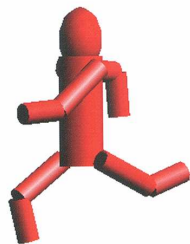
Outline



- Articulated figures
- Keyframe animation

∅ Kinematics

- Dynamics
- Guidelines



Angel Plate 1

Animating Motion

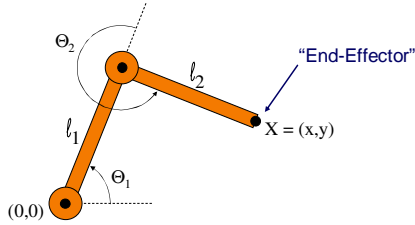


- Kinematics
 - Considers only motion
- Dynamics
 - Considers underlying forces
 - Compute motion from initial conditions and physics

Example: 2-Link Structure



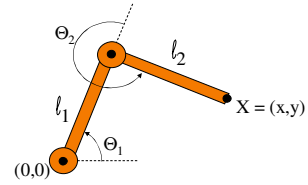
- Two links connected by rotational joints



Forward Kinematics



- Animator specifies joint angles: Θ_1 and Θ_2
- Computer finds positions of end-effector: X

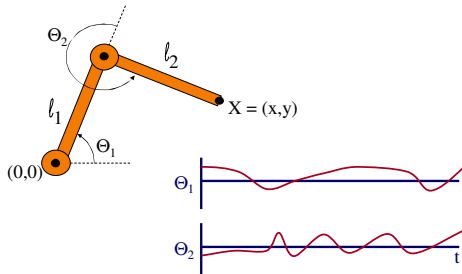


$$X = (l_1 \cos \Theta_1 + l_2 \cos(\Theta_1 + \Theta_2), l_1 \sin \Theta_1 + l_2 \sin(\Theta_1 + \Theta_2))$$

Forward Kinematics



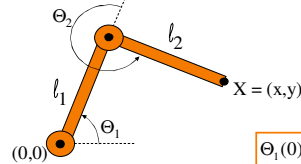
- Joint motions can be specified by spline curves



Forward Kinematics



- Joint motions can be specified by initial conditions and velocities

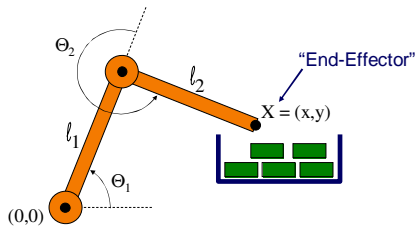


$$\begin{aligned} \Theta_1(0) &= 60^\circ & \Theta_2(0) &= 250^\circ \\ \frac{d\Theta_1}{dt} &= 1.2 & \frac{d\Theta_2}{dt} &= -0.1 \end{aligned}$$

Example: 2-Link Structure



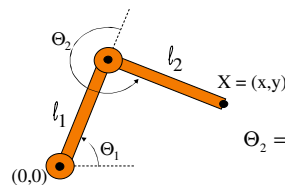
- What if animator knows position of "end-effector"



Inverse Kinematics



- Animator specifies end-effector positions: X
- Computer finds joint angles: Θ_1 and Θ_2 :



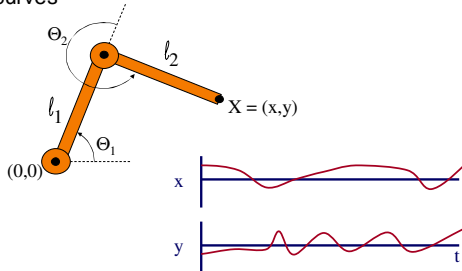
$$\Theta_2 = \cos^{-1} \left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2l_1 l_2} \right)$$

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

Inverse Kinematics



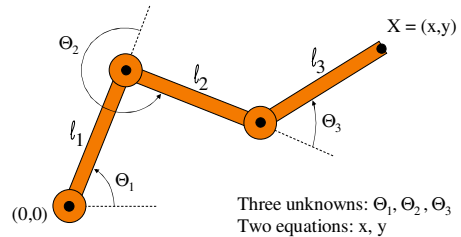
- End-effector positions can be specified by spline curves



Inverse Kinematics



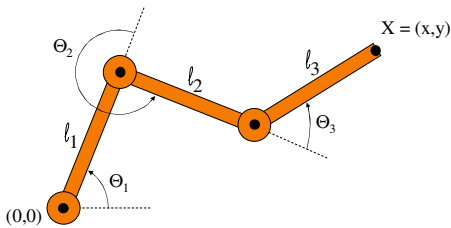
- Problem for more complex structures
 - System of equations is usually under-defined
 - Multiple solutions



Inverse Kinematics



- Solution for more complex structures:
 - Find best solution (e.g., minimize energy in motion)
 - Non-linear optimization



Example: Ball Boy



“Ballboy”

Fujito, Milliron, Ngan, & Sanoeki
Princeton University

Example: Toy Story II



Pixar

Summary of Kinematics



- Forward kinematics
 - Specify conditions (joint angles)
 - Compute positions of end-effectors
- Inverse kinematics
 - “Goal-directed” motion
 - Specify goal positions of end effectors
 - Suitable for in-betweening



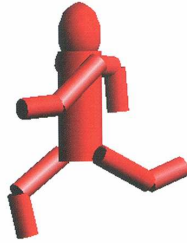
Inverse kinematics provides easier specification for many animation tasks, but it is computationally more difficult

Outline



- Articulated figures
- Keyframe animation
- Kinematics

∅ Dynamics

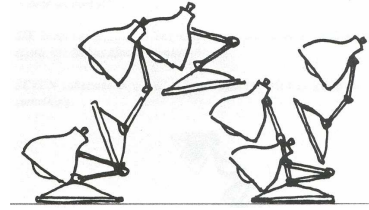


Angel Plate 1

Dynamics



- Simulation of physics insures realism of motion



Lasseter '87

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 within 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: particle with jet propulsion
 - $\mathbf{x}(t)$ is position of particle at time t
 - $\mathbf{f}(t)$ is force of jet propulsion at time t
 - Particle's equation of motion is:

$$m\mathbf{x}'' - \mathbf{f} - m\mathbf{g} = 0$$

- Suppose we want to move from a to b within t_0 to t_1 with minimum jet fuel:

$$\text{Minimize } \int_{t_0}^{t_1} |\mathbf{f}(t)|^2 dt \text{ subject to } \mathbf{x}(t_0) = \mathbf{a} \text{ and } \mathbf{x}(t_1) = \mathbf{b}$$

Witkin & Kass '88



Spacetime Constraints



- Discretize time steps:

$$\mathbf{x}'_i = \frac{\mathbf{x}_i - \mathbf{x}_{i-1}}{h}$$

$$\mathbf{x}''_i = \frac{\mathbf{x}_{i+1} - 2\mathbf{x}_i + \mathbf{x}_{i-1}}{h^2}$$

$$m \left(\mathbf{x}''_i = \frac{\mathbf{x}_{i+1} - 2\mathbf{x}_i + \mathbf{x}_{i-1}}{h^2} \right) - \mathbf{f}_i - m\mathbf{g} = 0$$

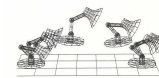
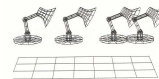
$$\text{Minimize } h \sum_i |\mathbf{f}_i|^2 \text{ subject to } \mathbf{x}_0 = \mathbf{a} \text{ and } \mathbf{x}_1 = \mathbf{b}$$

Witkin & Kass '88

Spacetime Constraints



- Solve with iterative optimization methods



Witkin & Kass '88

Spacetime Constraints

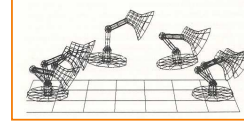


- Advantages:
 - Free animator from having to specify details of physically realistic motion with spline curves
 - Easy to vary motions due to new parameters and/or new constraints
- Challenges:
 - Specifying constraints and objective functions
 - Avoiding local minima during optimization

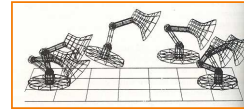
Spacetime Constraints



- Adapting motion:



Original Jump



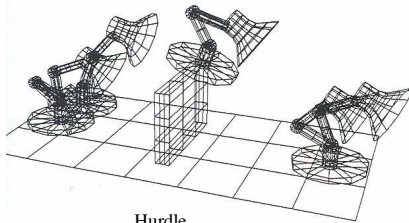
Heavier Base

Witkin & Kass '88

Spacetime Constraints



- Adapting motion:



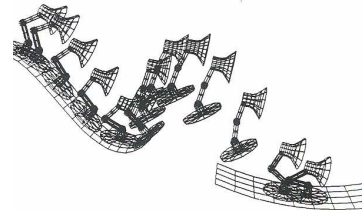
Hurdle

Witkin & Kass '88

Spacetime Constraints



- Adapting motion:



Ski Jump

Witkin & Kass '88

Spacetime Constraints



- Advantages:
 - Free animator from having to specify details of physically realistic motion with spline curves
 - Easy to vary motions due to new parameters and/or new constraints
- Challenges:
 - Specifying constraints and objective functions
 - Avoiding local minima during optimization

Example: Monsters, Inc.



Pixar

Summary



- Articulated figures
 - Hierarchies parts connected by joints
- Keyframe animation
 - Poses specified at key times
 - In-betweening to fill in the rest
- Kinematics
 - Forward kinematics
 - Inverse kinematics
- Dynamics
 - Space-time constraints
 - Also other physical simulations in previous lecture