Computer Animation:

What is animation?
- Modifying scene parameters as a function of time

Why animate?
- Provides more information
- Interaction heightens immersion
- Fun
**Principles of Animation**

1. **Squash and Stretch** -- rigidity & mass by distortion
2. **Timing** -- weight & size & even personality by spacing action
3. **Anticipation** -- preparation
4. **Staging** -- unmistakably clear presentation of ideas
5. **Follow Through and Overlapping Action** -- relationship to the next action
6. **Straight Ahead Action and Pose-To-Pose Action** -- 2 contrasting approaches
7. **Slow In and Out** -- subtlety of timing and movement.
8. **Arcs** -- path of natural movement
9. **Exaggeration** -- accentuating the essence
10. **Secondary Action** -- action resulting from another
11. **Appeal** -- make the audience enjoy watching.

[Lasseter’87]

**Overview**

1) Scripting

2) Keyframing

3) Kinematics

4) Motion Processing

5) Higher Level Animation

6) Dynamics and Simulation
**Example of Scripting**

Specifying the parameters at every frame

```plaintext
define spinningCube()
    rotAngle = pi*frameNumber / 50

define carScript()
    carTranslation = 10*(frameNumber / 100)
    wheelRotation = pi*frameNumber / 5
```

**Keyframing**

Specify only the important frames, interpolate the frames in-between

What and how to interpolate is important
**Kinematics**

The study or specification of motion, independent of the underlying physics that created the motion.

Articulated Figure:
A figure made up of a series of links (bones) connected at joints.

---

**Forward Kinematics**

Given the character’s state, calculate its pose

\[ X = f(\theta) \]

\[ X = \begin{bmatrix} l_1 \cos \theta_1 + l_2 \cos(\theta_1 + \theta_2) \\ l_1 \sin \theta_1 + l_2 \sin(\theta_1 + \theta_2) \end{bmatrix} \]
**Inverse Kinematics**

Given the character’s pose, calculate its state

\[ \theta = f^{-1}(X) \]

**Motion Processing**

- **Motion Capture**
- **Motion Editing**
Behavioral Animation

Animating by describing an actor's behavior

An actor’s behavior defines how the actor interacts with other actors and the environment

TRex()
  if(player is close)
    eatPlayer()
  else if(can see player)
    chasePlayer()
  else
    wander()

Useful for crowd animations
Dynamics

Using “physics” to define the animation

Model choice is important

\[ \begin{align*}
\dot{X} &= V \\
\dot{V} &= \frac{F}{m} \\
\dot{\theta} &= \omega \\
\dot{L} &= T
\end{align*} \]

Can use “augmented” laws of physics

Dynamics — Particle Systems

Particle Systems [Reeves83]

Represent “fuzzy” objects (such as fire, smoke) as a collection of particles

Particels contain local state
- Position
- Velocity
- Age
- Lifespan
- Rendering properties
Dynamics — Rigid Bodies

Rigid Bodies
- Integration
- Collisions
- Constraints

Dynamics — Deformable Objects

Deformable Objects
- FFD
- Spring systems
- Finite Elements
DYNAMICS — CLOTH

Cloth Simulation
- Stable Integration
- Adaptivity
- Material Properties

DYNAMICS — FLUIDS

Fluid Simulation
- Navier Stokes, plus *lots* of topology changes

[Foster & Fedkiw ’01]
Real Time Animation

Zelda

Offline Animation — Anything Goes

Final Fantasy  Pixar movies
Be Careful: Zombie Line

[Entis’07]

Animation quality

Model Fidelity