Animation in the Interface

Reading assignment: This section based on 2 papers

- Bay-Wei Chang, David Ungar, "Animation: From Cartoons to the User Interface", *Proceedings of UIST' 93*, pp.45-55.
  http://www.acm.org/pubs/articles/proceedings/uist/168642/p45-chang/p45-chang.pdf

- Scott E. Hudson, John T. Stasko, "Animation Support in a User Interface Toolkit: Flexible, Robust and Reusable Abstractions", *Proceedings of UIST '93*, pp.57-67.
  http://www.acm.org/pubs/articles/proceedings/uist/168642/p57-hudson/p57-hudson.pdf

Many things happening at once...

- 3D, VR
- Distributed UIs (ubicomp, CSCW)
- Animated Interfaces

Animation is of increasing interest

- Perceptual advantages
- Just recently had enough spare horsepower
- Now seeing this in the mainstream (Win '98)

Why animation?

- Gives a feeling of reality and liveness
  - "animation" = "bring to life"
  - make inanimate object animate

Why animation?

- Provides visual continuity (and other effects) enhancing perception
  - particularly perception of change
    - hard to follow things that just flash into & out of existence
    - real world doesn't act this way

Why Animation?

- Can also be used to direct attention
  - movement draws attention
  - strong evolutionary reasons
    - therein lies a danger
    - overuse tends to demand too much attention
      - e.g., the dreaded paper clip

Why Animation?

- Used sparingly and understandingly, animation can enhance the interface

Three principles from traditional animation

(Not mutually exclusive)

- Solidity
  - make objects appear to be solid obj
- Exaggeration
  - exaggerate certain physical actions to enhance perception
- Reinforcement
  - effects to drive home feeling of reality

Specific techniques employing these principles

- Good related paper: John Lasseter, "Principles of traditional animation applied to 3D computer animation", Proceedings of SIGGRAPH '87, pp. 35 - 44
- Solidity
  - want objects to appear solid and appear to have mass
  - Solid (filled) drawing
    - now common place

Specific techniques employing these principles

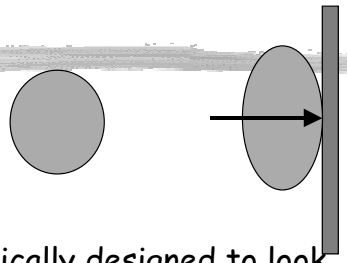
- Solidity
  - No teleportation
    - objects must come from somewhere
      - not just "pop into existence"
    - nothing in the real world does this (things with mass can't do this)

Specific techniques employing these principles

- Solidity
  - Motion blur

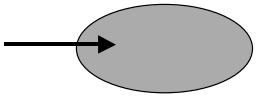
    - if objects move more than their own length (some say 1/2 length) in one frame, motion blur should be used
    - matches real world perception
    - makes movement look smoother
    - doesn't need to be realistic

Specific techniques employing these principles



- Solidity
  - Squash and stretch
    - Cartoon objects are typically designed to look "squishy"
    - When they stop, hit something, land, they tend to squash
      - like water balloon
      - compress in direction of travel

Specific techniques employing these principles



- Solidity
  - Squash and stretch
    - Also stretch when they accelerate
      - opposite direction
    - Basically an approximation of inertia + conservation of volume (area)

Specific techniques employing these principles

- Solidity
  - Squash and stretch
    - Although S&S makes things look "squishy" they contribute to solidity because they show mass
    - (This is tends to be exaggerated)

Specific techniques employing these principles

- Solidity
  - Follow through (& secondary action)
    - Objects don't just stop, they continue parts of the motion
      - e.g., clothes keep moving, body parts keep moving
    - Reinforces that object has mass via inertia
    - (also tends to be exaggerated)

Follow Through

- Notice feather lags behind character
- Also S&S here
- From: Thomas & Johnston "The Illusion of Life: Disney Animation", Hyperion, 1981

Specific techniques employing these principles

- Exaggeration
  - Cartoon animation tends to do this in a number of ways
    - paradoxically increases realism (liveness) by being less literal
  - What is really going on is tweaking the perceptual system at just the right points

Specific techniques employing these principles

- Exaggeration
  - Anticipation
    - small counter movement just prior to the main movement
    - this sets our attention on the object where the action is (or will be)
  - Squash & stretch
  - Follow through

Specific techniques employing these principles

- Reinforcement
  - Slow-in / Slow-out
    - Movement between two points starts slow, is fast in the middle, and ends slow
    - Two effects here
      - objects with mass must acceler
      - interesting parts typically @ ends
        - tweaking perception

Specific techniques employing these principles

- Reinforcement
  - Movement in arcs
    - Objects move in gently curving paths, not straight lines
    - Movements by animate objects are in arcs (due to mechanics of joints)
    - Most movements in gravity also in arcs

Recap

- Appearance of mass
  - solidity & conservation of volume
  - several ways to show inertia
- Tweak perception
  - direct attention to things that count
  - time on conceptually important parts
- Caricature of reality

Examples From Video

UIs are not Cartoons

- Active vs. passive
  - Animation can bring otherwise boring things to life, but…
- Its not a uniformly good thing
  - demands a lot of attention
  - can take time

Issues when building such UIs

- Multiple "agents"

Issues when building such UIs

- Animation

Implementation

- Integrated with I/O model
- Time-based
- Smoothness
- Interruptible
- Synchronization possible

Integration: Work Queues

- Central queue where all work is registered
  - objects with "step" functions
  - remove from queue when done

Integration: FSM

- Use FSM to model state of animation
  - Each step triggers a transition (often back to the same state)

Time-based

- Machine and load independent

Smoothness

- Time-based helps
- Smooth out motion: blur
- Speed up rate: degredation

Interruptible

- User must be in control

Synchronization

- Multiple animations can relate in complex ways

Making animation happen in a toolkit

- Paper describes model in subArctic (and predecessor)
  - high to middle level model
  - robust to timing issues
- Primary abstraction: transition
  - models movement over time
    - arbitrary space of values (eg, color)
    - screen space is most common

Transition consists of

- Reference to obj being animated
  - passage of time modeled as events
- Time interval
  - period of time animation occurs
- Trajectory
  - path taken through value space
  - timing of changes through values

Trajectory has two parts

- Curve
  - set of values we pass through
  - typically in 2D space, but could be in any space of values (e.g., font size)
- Pacing function
  - mapping from time interval (0…1) to "parameter space" of curve (0…1)
  - determines pacing along curve
    - e.g., slow-in / slow-out

Mapping from time to value

- Time normalized with respect to animation interval (0...1)
- Normalized time is transformed by pacing function (0…1)
- Paced value is then fed to curve function to get final value

To get a movement

- Create and schedule a transition
  - several predefined types (i.e., linear)
  - scheduling can be done absolute
    - start stop at the following wall clock times
  - or relative
    - D seconds from now
    - D seconds from start / end of that

System action

- Transition will deliver time as input using animatable interface
  - transition_start()
  - transition_step()
  - transition_end()
- Each delivers:
  - trajectory object, relative time & value

Transition steps

- Steps represent intervals of time, not points in time
  - deliver start and end times & values
- Typical OS can't deliver uniform time intervals
  - Number of steps (delivery rate) is not fixed in advance (animation system sends as many as it can)
  - system delivers as many as it can

Recap

- Transition
  - Object to animate
  - Time interval to work over
    - Time ➡ (0…1)
  - Trajectory to pass through
    - Pacing function (0…1) ➡ (0… 1)
    - Curve (0…1) ➡ Value