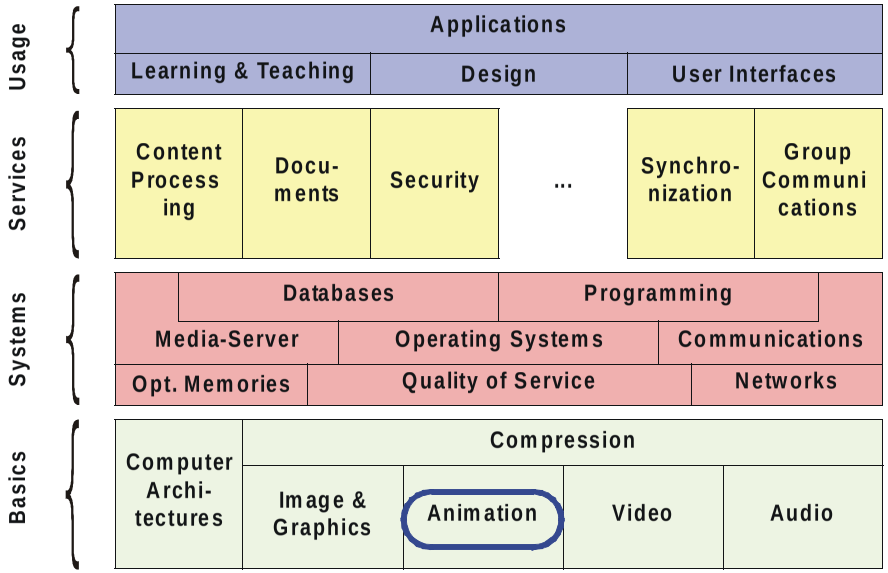
2.5 Animations



Contents

1. Terminology

- **to animate = "to get things alive"**
  i.e., to make them change
- **visual effects: varying**
  - position
  - shape
  - color
  - transparency
  - structure
  - pattern
  - ...
- **caused by:**
  - activity of objects themselves (e.g. translation, rotation, growth, ...)
  - varying environmental conditions (e.g. illumination)
  - activity of the viewer (e.g. „walking" through an artifical world)
- **related to**
  - producing, storing, transmitting, displaying animations
    with computer support
- **i.e.**
  - many similarities / overlaps / combinations with conventional animation

2. Generation of (Computer) Animation (1)

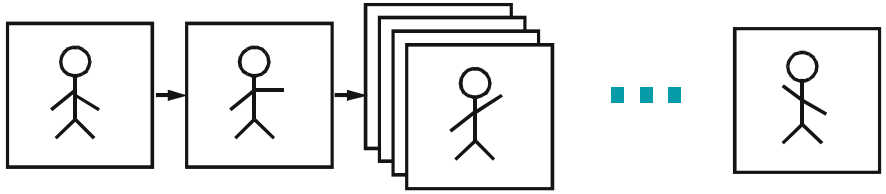
1. **to describe primitives**
   - by means of computer-generated images
   - digitalization of photos or drawings
   - generation of „body models" by scanning characteristic points

2. **to combine them (picture composition) to produce single independent frames**

3. **to describe the dynamics of the scene**
   - depending on the characteristics of the objects (constant changes, or even „alive" and changing?)
   - translations, rotations, growth, zoom ...

Generation of (Computer) Animation (2)

4. **to change and to combine the primitives according to the dynamics**
   - inter-frames of moving pictures could be interpolated, e.g. by means of Linear Interpolation (Lerping) or
   - (more realistically): to use splines to describe a movement



3. Specification and Control of an Animation (1)

Specification: 3 main categories of notation:

- **linear lists, describing**
  - start and end frame, during which a certain change is hapening
  - events / changes to be triggered during that time
  - e.g.:
  - 42, 53, B, ROTATE „PALME“, 1, 30
  - between frame 42 and 53 rotate object „PALME“ 30 degrees around axis 1

- **by means of a (Higher Level) Programming Language**
  - values of variables describe change of certain parameters
  - control flow / equations describe the dynamics
  - e.g.:
  - Language ASAS (LISP extension) with support for graphic primitives (vectors, colors, groups, views ...)

Specification and Control of an Animation (2)

- **Special Languages to describe Graphics**
  - allow for interactive description in a „visual way“
  - e.g.:
    GENESYS, DIAL, S-Dynamics System

Control of Animations (1)

- **explicit / open**
  - simplest way with explicit description of dynamics for each object

- **procedural**
  - objects interact by forwarding information
  - to use knowledge about their characteristics
  - dependencies (are 2 objects at the same place at the same time?) may be tested
  - behaviour of active participants (in „actor-based“ systems) may vary due to the activities of others

- **according to varying conditions**
  - basic idea: systems are more or less coupled
  - models of dynamics of real objects and their material characteristics as basis for motion according to changing conditions

Control of Animations (2)

- **by analyseing real motions**
  - e.g. **Rotoscoping**: a real person takes the role during the production, its body will later replaced by the the animation (e.g. by partial recolouring)
  - use sensors / indicators to get a model of specific points of the actor

- **kinematics and dynamics**
  - objects and their movement described by kinematics and dynamics of characteristic points („mass points“)

4. Displaying Animations

basic knowledge

- frame rate, etc.
- already known from lecture „Video“

often support by means of special hardware usage:

- **„Sprites“:**
  - hardware support for the animation of small objects

- **Double Buffering:**
  - write a frame to a buffer that is currently not read by the display adapter
  - switch buffers with frame change frequency
  - allows for slower access to the video memory
    while still having a dynamic impression without hard transitions

5. Transmitting Animations

„pixel representation“

- as series of single images
- well suited for almost any content
- less computional effort at the receiver side
- high data rate, (encoding techniques to reduce it)

„symbolic representation“

- as specifica of graphic objects (e.g. sphere with center, radius and color)
- description of dynamics (e.g. translation or rotation speed of any object)
- depends on finding an equivalent model
- high computional effort at the receiver side
- lower data rate
- well suited for individual interactive access by many users
  - (imagine a „world“ to be served by a WWW server, that everybody can visit on his own)
- ideas have been pushed by Java and other means of actively executing code at the receiver side

6. Storing/Transmitting/ Accessing Animations

MPEG

- see “Compression”

QuickTime

- see “Programming”

AVI

- pseudo standard for animations, integrates a number of dedicated codecs

Animated Gifs

- a sequence of pictures in one file

Server Side Pushes

- so picture gets reloaded every x seconds

Java

- see “Programming”

VRML

7. Virtual Reality Modeling Language (VRML)

Standard for description of 3-dimensional interactive worlds

- export and exchange format for all major modelling systems
- e.g., CAD systems for describing single objects

History

- development started in Mai 1994
- to be used in the WWW
- versions:
  - VRML 1.0
  - VRML 2.0
  - VRML 97
  - (outcome of VRML 2.0 ISO/IEC standardization with few minor extensions)

„Worlds“ are described in

- ASCII Files
- (File extension .wrl, or .wrz for compressed represantation)
- combining primitives and describing their dynamics and interactions
- MIME type: model/vrml or x-world/x-vrml (outdated)

VRML 1.0 vs. VRML 2.0

VRML 1.0

- standard objects (cube, sphere, cone, cylinder, text)
- arbitary objects (surfaces, linesets, pointsets)
- ability to
  - fly trough, walk trough, to examine scenes
- lights, cameras (viewpoints)
- textures on objects
- clickable links
- define and reuse of objects

VRML 2.0 (all VRML 1.0 features)

- animated objects
- switches, sensors
- scripts (Java or JavaScript) for describing behaviour
- interpolators (color, position, orientation, ...), extrusions
- background colors and textures
- sound (.wav and MIDI)
- animated textures, event routing
- additional efficient mechanism for defining and reusing objects

Using VRML (1)

see VRML Repository at the WWW

- **http://www.sdsc.edu/vrml/**

tools

- **VRML viewers**
  - standalone or as plugins for WWW browsers
  - e.g. CosmoPlayer (Win) or VRWeb (many Unix dialects, Linux)
- **„World builders“ for editing**

Using VRML (2)

```
#VRML V1.0 ascii

Separator {

    Material {
      ambientColor  1 0 0
      diffuseColor 1 0 0
    }
    Cube {
      width   1
      height  1
      depth   1
    }

    Translation { translation 2 0 0 }

    Material {
      ambientColor    0 1 0
      diffuseColor    0 1 0
    }
    Sphere {
      radius  1
    }
```

Using VRML (3)

```
    Translation { translation 2 0 0 }

    Material {
      ambientColor    0 0 1
      diffuseColor    0 0 1
    }
    Cone {
      parts   ALL
      bottomRadius    1
      height  2
    }

    Translation { translation 3 0 0 }

    Cylinder {
      parts   ALL
      radius  1
      height  2
    }
}
```


Using VRML - an example (4)

Forms this scene:

- **wireframe**
- **texture**
- **from a different viewpoint**