

# A Conceptual Model for Computer Animation

William Joel, Director  
Center for Graphics Research  
Western Connecticut State University  
Danbury, CT USA



## Outline



- Traditional Animation
- Scenes vs. Frames
- Model
- System Design

## Traditional Animation



- How was it done?
- How does it differ from computer animation?

## Scenes vs. Frames



- Common animation software most often uses a frame-based model
- But traditional animation, and modern animation production, focus on scenes or “shots”
- Therefore, a scene-based model would be more appropriate

## Model



- Elements
- Environment
- States
- Frames
- Scenes
- Animations

## Elements



- $e_i$
- An “object” in an animation
- Can be represented by a finite set of state variables

## Environment



- $E = \{ e_i \mid 1 \leq i \leq n \}$
- Set of all identified elements
- Generally non-empty, but could be an empty set,  $\emptyset$

## States



- For each  $e \in E$  we associate a non-empty set  $S_e$ , possibly infinite

## Frames



- Let  $\{S_e\}_{e \in E}$  be the family of state sets
- Frame  $f$  on an environment  $E$  is defined as  $f \in \prod_{e \in E} S_e$
- and is written  $f = \langle s_e \mid e \in E \rangle$
- or as a function,  $f : E \rightarrow \prod_{e \in E} S_e$  where  $f(e) \in S_e$
- $F_E$  is the set of all frames for  $E$

## Scenes



- A scene for an environment  $E$  consists of an ascending sequence of reals  $I$  and a function  $c$  that maps  $I$  to  $F_E$
- $\langle c, I \rangle$
- When frames are of interest  $c(I) = (f_1, \dots, f_m)$ , where  $I = (i_1, \dots, i_m)$

## Animations



- A sequence of scenes, written  $a = (c_1, \dots, c_p)$

## Object Subdivisions



- Subscene
- Subanimation
- Semienvironment
- Semiframe
- Semiscene
- Coscene

## Subscene



- Given a scene  $\langle c, l \rangle$  based on environment  $E$ , a subscene of this scene is the pair  $\langle c, l_{x,y} \rangle$  where  $l_{x,y}$  is a contiguous subsequence of indices from  $l$ ,  $(i_x, \dots, i_y)$ , and  $x$  and  $y$  are frame numbers
- $\langle c, l_{x,y} \rangle$  is also based on  $E$

## Subanimation



- A subanimation  $a_{x,y}$  of an animation  $a = (c_1, \dots, c_p)$ , is either empty or is a subsequence  $a_{x,y} = (c_x, \dots, c_y)$  where  $1 \leq x \leq y \leq p$

## Semienvironment



- A semienvironment  $E'$  of environment  $E$  is a non-empty subset  $E' \subseteq E$

## Semiframe



- A semiframe  $f'$  of frame  $f$  is a restriction of  $f$  to a subset of its environment  $f' = f \upharpoonright E'$

## Semiscene



- A semiscene of  $\langle c, l \rangle$  is a scene  $\langle c', l \rangle$  based on semienvironment  $E' \subseteq E$ , such that  $c'(i_k) = c(i_k) \upharpoonright E'$  for each  $i_k \in l$

## Coscene



- A coscene that can be computed independently of the remainder of its parent scene

## Functions



- Restriction
- Composition
- Access
- Iteration
- Re-indexing
- Interpolation
- Concatenation

## Restriction



- Represented by the function  $restrict()$
- For example, a restricted function is written  $f|_{E'}$

## Composition



- Given frames  $f'$  and  $f''$ , corresponding to environments  $E'$  and  $E''$  respectively
- When  $E' \cap E'' = \emptyset$ ,  $f$  is a frame defined on  $E' \cup E''$  such that  $f = f' \cup f''$

## Composition



- When  $E' \cap E'' \neq \emptyset$ , an additional step is required to resolve state conflicts
- This step is represented by a function,  $resolve()$ , such that  $f = compose(f', f'', resolve)$
- This is graphically represented by

$$f = \begin{bmatrix} f' \\ f'' \end{bmatrix}$$

## Access



- **Environment**
  - $E(f) = Domain(f)$
  - $E(\langle c, l \rangle) = E(c(i))$
- **Scene function**
  - $\mathcal{L}(\langle c, l \rangle) = c$
- **Index sequence**
  - $\mathcal{I}(\langle c, l \rangle) = l$
- **Repeat a frame**
  - $\langle c', l \rangle = copy(c(j), l)$

## Iteration



- To iterate a frame function  $o$  on two or more scenes that share a common index sequence
  - $\langle c, l \rangle = iterate(o, s_1, \dots, s_k)$
- where
  - $s_j = \langle c_j, l \rangle$

## Re-indexing



- Given  $\langle c, l \rangle$ , and  $l'$ , where  $|l|=|l'|=n$   
 $\langle c', l' \rangle = \text{reindex}(\langle c, l \rangle, l')$
- and  
 $c'(i') = c(i)$ ,  $1 \leq j \leq n$

## Interpolation



- Given  $\langle c, l \rangle$ , and  $l'$ , where  $|l'| \neq |l|$   
 $\langle c', l' \rangle = \text{inter}(\langle c, l \rangle, l', Fn)$
- where  
 $c'(i') = Fn(\langle c, l \rangle, l')$
- It is assumed that
- $c(i) = c'(j)$ , when  $i=j$

## Composition



- Given  
 $\text{compose}.(f_1, f_2) = \text{compose}(f_1, f_2, \text{resolve})$
- then  
 $\langle c_3, l_3 \rangle = \text{iterate}(\text{compose}, \langle c_1, l_1 \rangle, \langle c_2, l_2 \rangle)$

## Concatenation



- Given  $\langle c_1, l_1 \rangle$  and  $\langle c_2, l_2 \rangle$ , where  $i < j$  for all  
 $i \in l_1$  and  $j \in l_2$
- $\langle c, l \rangle = \langle c_1, l_1 \rangle \bullet \langle c_2, l_2 \rangle$   
 $= \langle c_1 \cup c_2, l_1 \bullet l_2 \rangle$

## Functions (continued)



- Locate
- Take
- Find
- Fade-in, Fade-out
- Dissolve

## Locate



- $c(i) = \text{locate}(\langle c, l \rangle, \ell, \text{order})$
- $\ell(f)$  takes as input a frame and determines whether a condition is true

## Take



- $\langle c, l, x, y \rangle = \text{take}(\langle c, l \rangle, a, b)$
- Where  $a < i_x < i_y < b$

## Find



- $c(i) = \text{find}(\langle c, l \rangle, i)$

## Fade-in, Fade-out



- $\langle c', l \rangle = \text{fadein}(\langle c, l \rangle, x, y, \text{rule})$
- $\langle c', l \rangle = \text{fadeout}(\langle c, l \rangle, x, y, \text{rule})$
- Where the rule is expressed as  $\text{rule}(\text{pixel}, t)$

## Dissolve



- $\text{cross}(\langle c', l \rangle, \langle c'', l \rangle, x, y, \text{rule}) =$   
[  $\langle c', l \rangle = \text{fadeout}(\langle c', l \rangle, x, y, \text{rule})$   
 $\langle c', l \rangle = \text{fadein}(\langle c'', l \rangle, x, y, \text{rule})$  ]

## Generating Functions



- User defined
- Built on existing primitives
- Re-usable
- $\mathcal{B}(\hat{C}, P)$
- Where  $\hat{C}$  is a set of input scenes, and  $P$  is a set of additional parameters

## Behaviors



- Generating functions
- Restricted to specific elements
- $b_e(\hat{C}, P) = \mathcal{B}(c, P) \mid \{e\}$