UNIT-VI

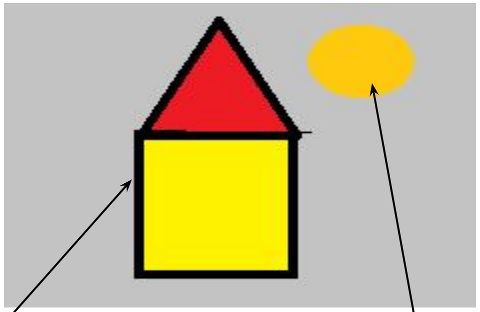
Introduction to Animation and Gaming

SEGMENT

- In practice the image on the display screen is **<u>often composed of several pictures</u>**.eg. Internal plan of living room(sofa,tv,showcase)
- Each picture is having different attribute like color, size and position.
- The image information stored in display file which is not efficient to display subpicture stucture.
- To achieve this display file divided into segments.
- **<u>Each segment corresponding to the</u>**

<u>component of overall display associate</u>

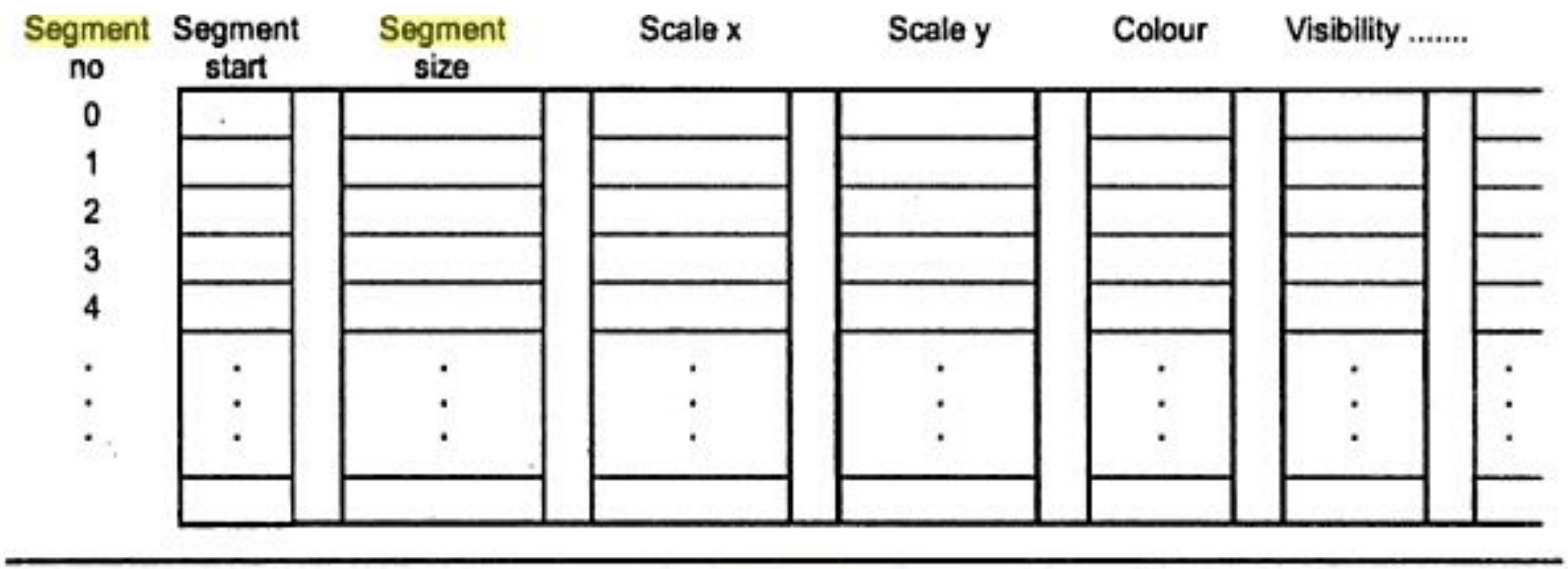
<u>a set of attributes like visibility, size.</u>



- **<u>Along with this it is associated with transformation like scaling ,rotation and translation</u>**

SEGMENT TABLE

- To identify each segment need segment name and display file location also.
- **The structure used to organize all this information related to the segment is called as segment table.**
- **It can be formed by array one store size,another store like color**



SEGMENT TABLE

- Each row of segment table represent information of one segment including name , colour,size,visibility
- **Eg.if we want fourth segment visible the no.4 segment visibility array entry will be ON.**
- Alternative approach is linked list extra field require that is **link or pointer which gives location of segment in the segment table**
- **Advantages**

1. **No limit on max no of size & dynamic**
2. **Ordering and sorting is quite easy**

Segment number	Segment start	Segment size	Scale x	Scale y	Colour	Visibility	Link
1							3
2							4
3							2
4							5
5							Null /

SEGMENT TABLE FUNCTIONS

- **FUNCTIONS ARE LIKE**

1. Create Segment
2. Close Segment
3. Rename Segment
4. Delete Segment

SEGMENT TABLE FUNCTIONS

1. Create Segment

Algorithm : Create Segment

1. Check whether any segment is open; if so display error message : "Segment is still open" and go to step 9.
2. Read the name of new segment.
3. Check whether the new segment name is valid; if not display error message : "Not a valid segment name"and go to step 9.
4. Check whether the new segment name is already existing in the same-name list; if so display error message : "Segment name already exists" and go to step 9.
5. Initialize the start of the segment at the next free storage area in the display file.
6. Initialize size of this segment equal to zero.
7. Initialize all attributes of segment to their default values.
8. Indicate that the new segment is now open.
9. Stop.

SEGMENT TABLE FUNCTIONS

2. Close Segment

1. Check whether any segment is open; if no, display error message : " No segment open" and go to step 6.
2. Change the name of currently open segment, say 0, i.e., unnamed segment.
3. Delete any unnamed segment instructions which may have been saved, i.e. initialize the unnamed segment with no instructions.
4. Initialize the start of the unnamed segment at the next free storage area in the display file.
5. Initialize size of unnamed segment equal to 0.
6. Stop.

SEGMENT TABLE FUNCTIONS

3. Delete Segment

1. Read the name of segment which is to be deleted.
2. Check whether that segment name is valid; if not, display error message "Segment not valid" and go to step 8.
3. Check whether the segment is open, if yes, display error message "Can't delete the open segment" and go to step 8.
4. Check whether the size of segment is greater than 0; if no, no processing is required, as segment contains no instructions. Therefore, go to step 8.
5. Shift the display file elements which follow the segment which is to be deleted by its size.
6. Recover the deleted space by resetting the index of the next free instruction.
7. Adjust the starting positions of the shifted segments by subtracting the size of the deleted segment from it.
8. Stop.

SEGMENT TABLE FUNCTIONS
4.Rename Segment

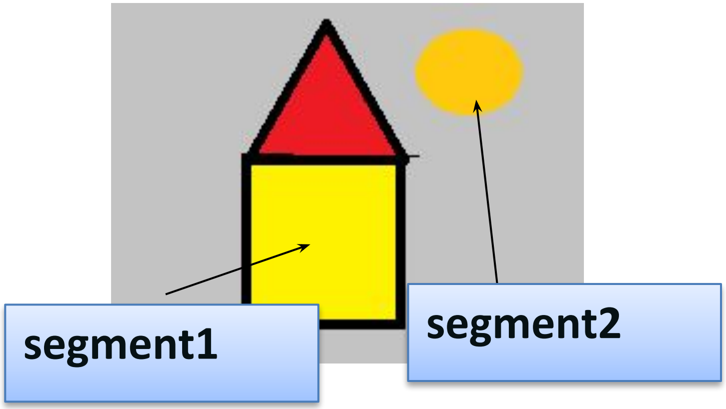
1. Check whether both old and new segment names are valid; if not, display error message "Not valid segment names" and go to step 6.
2. Check whether any of the two segments are open.
3. Check whether the new name we are going to give to the old segment is not already existing in the display file. If yes, display error message "Segment already exists" and go to step 6.
4. Copy the old segment table entry into the new position.
5. Delete the old segment.
6. Stop.

SEGMENT TABLE FUNCTIONS

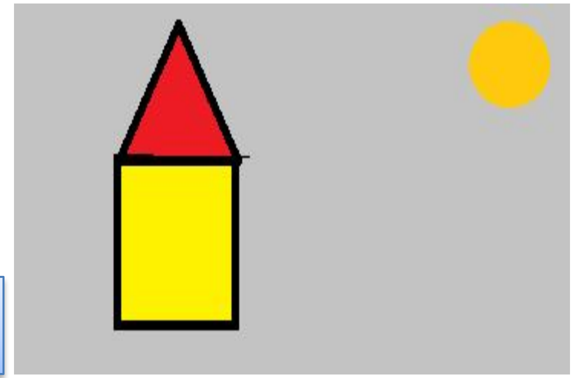
Advantages of segments

- Allow display file into sub picture structure
- **Can apply different set of attributes to different set of image**
- Selective portion of image can be displayed
- **Changes are easy and faster with segment**

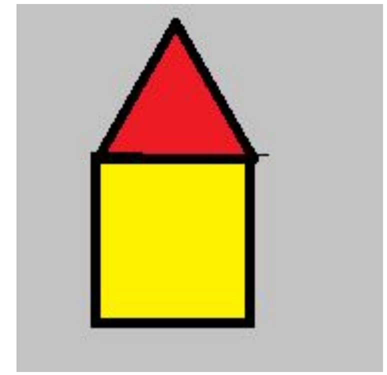
Image transformation



a) Original image



b) Segment 2 translate x



c) Segment2 visibility OFF

<u>Algorithm for fig b transformation</u>

1. Read segment name say seg_2 check valid name or not if no display error message and go to step 5
2. Read transaltion factor like tx and ty
3. Set transformation parameter translate_x[seg_2]=tx
4. Set transformation parameter translate_y[seg_2]=ty
5. Stop

Contents

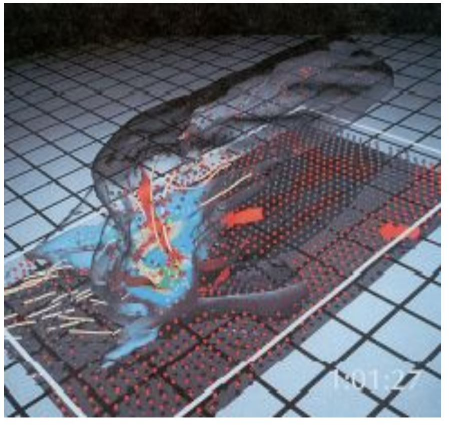
Computer Animation

What is Animation?

- Make objects change over time according to scripted actions
- Anytime sequence of visual changes in a scene.
- In animation, various transformations, along variations in object color, transparency, or surface texture are displayed.
- We can also produce animations by changing effects or other parameters and procedures associated with illumination and rendering.

What is Simulation?

- Predict how objects change over time according to physical laws

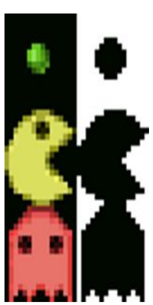


Introduction

•**Computer animation** is the process used for generating animated images (moving images) using computer graphics.

• **Animators** are artists who specialize in the creation of animation.

•From Latin **animātiō**, "the act of bringing to life"; from *animō* ("to animate" or "give life to") and -*ātiō* ("the act of").

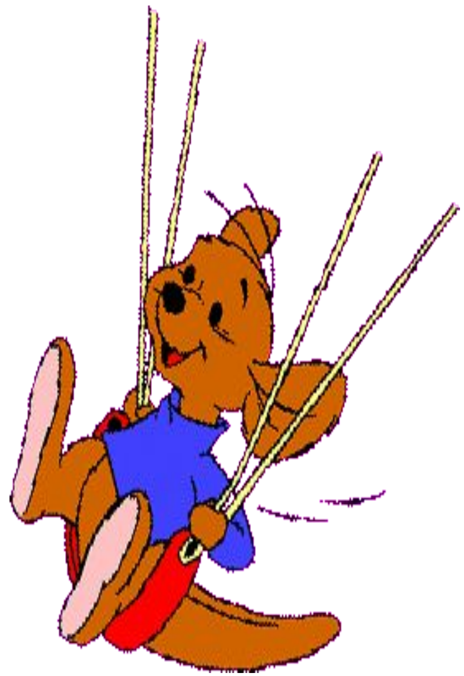


2D ANIMATION

3D ANIMATION

APPLICATIONS

Video games



cartoon

Mobile phones

Design Of Animation Sequences

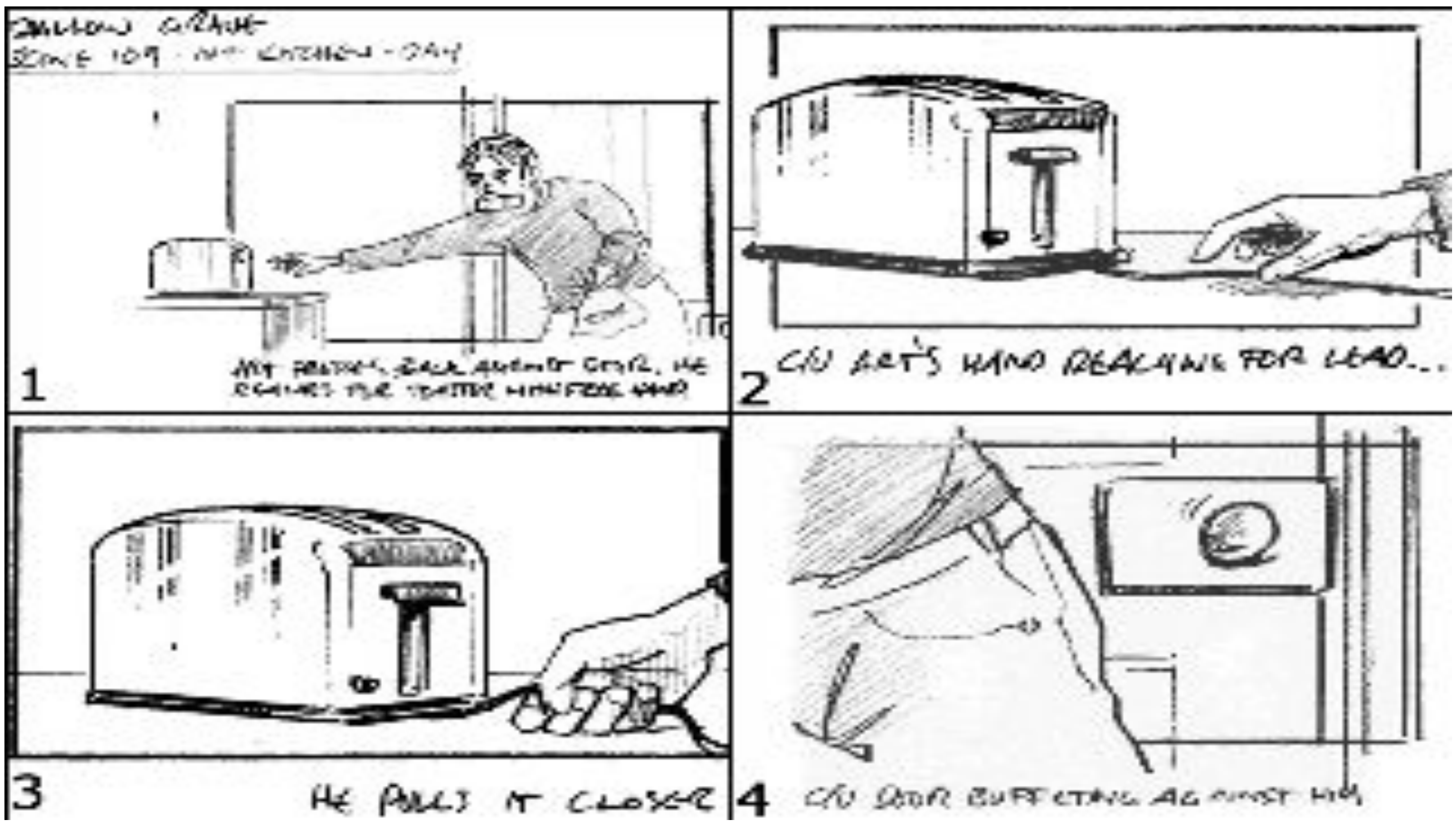
- Steps for designing animation sequences.

1. Storyboard Layout
2. Object definitions
3. Key frame specifications
4. Generation of in-between frames

Storyboard Layout

- It is the outline of the action. It defines **the motion sequence as a set basic events that are to take place**.
- Depending on the type of animation to be produced, the storyboard could consist of a rough sketches or it could be a list of the basis ideas for the motion.

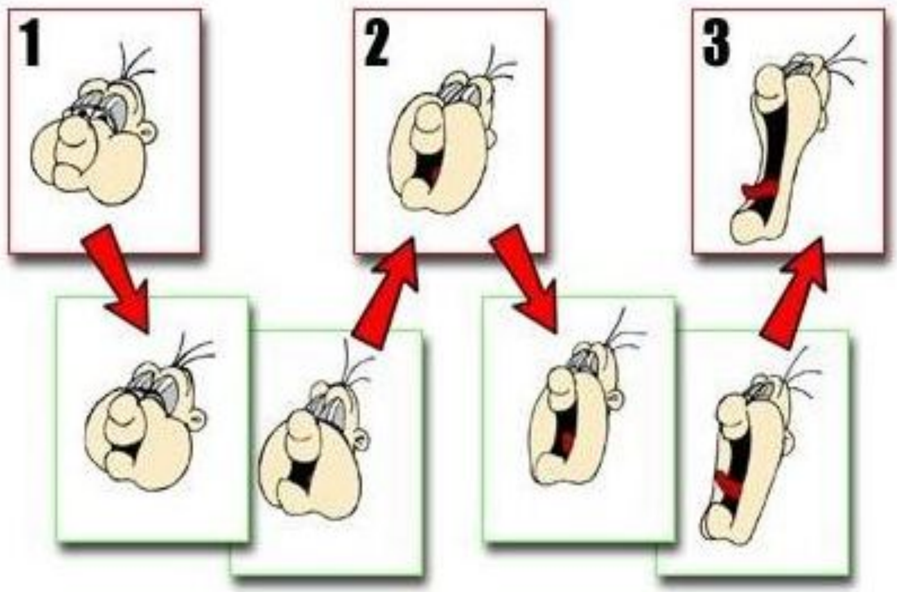
Storyboard Layout

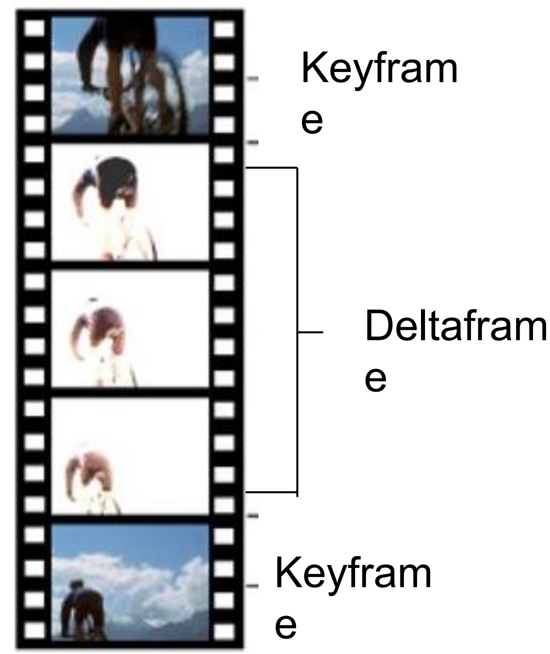


Object Definitions

- Each object participating in the action is given object definition, such as terms of basic shapes, such as polygons or splines.
- **Frame:** is one of the many single photographic images in a motion picture. Normally, 24 frames are needed for one second.
- **Key-frame:**
  - is a drawing that defines the starting and ending points of any smooth transition.
  - Within a key frame, each object is positioned according to the time for that frame.
- **In-between:**
  - Intermediate frames between the key frames.
  - The total number of in-between frames is determined by the display media that is to be use.

Frames and Key frames





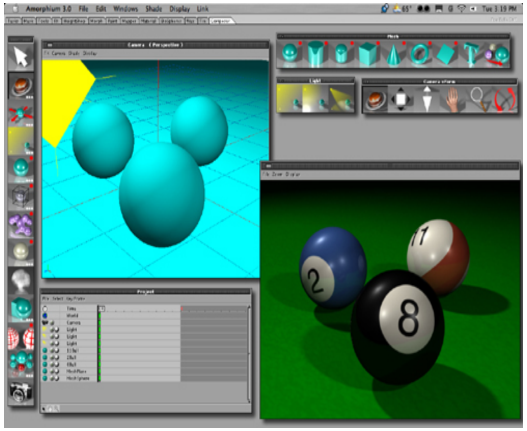
Object Definitions

GENERAL COMPUTER ANIMATION FUNCTIONS

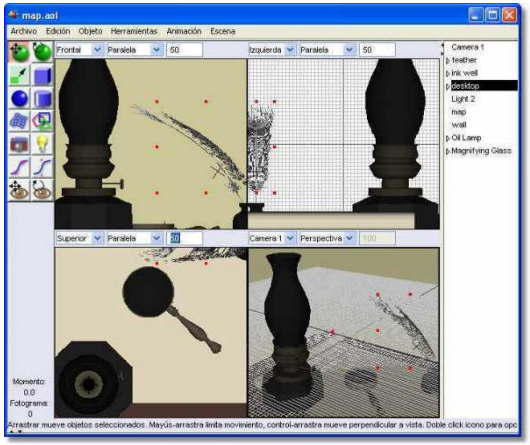
- Animation software provide basic functions to create animation and process the individual object.

FUNCTIONS

- Manipulate data object database.
- Motion generation.
- Object rendering.



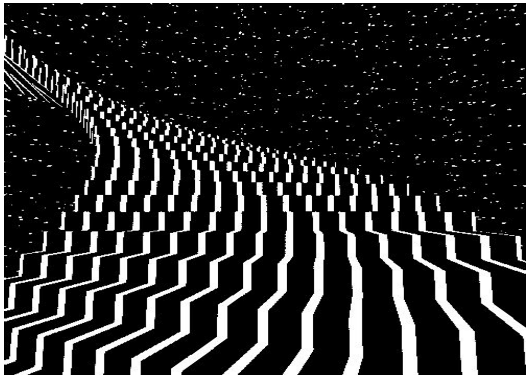
Amorphium



Art of illusion

Raster Animations

Real-time animations can be generated using raster operations.



ORGANISATION OF A VIDEO COLOUR TABLE

0 1 0 0 0 0 0 1 41H → 65

255

65 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 → Monitor

24-bit

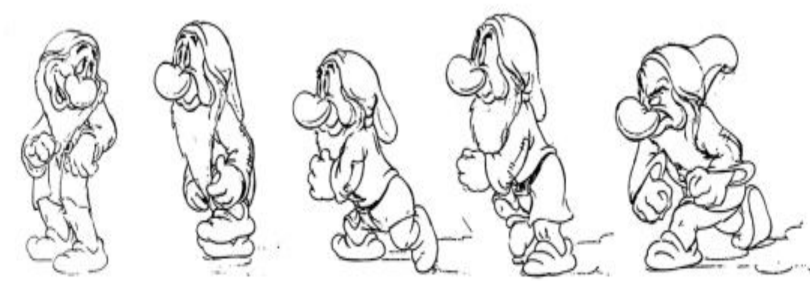
Pixel value

0 Red Green Blue

Computer Animation Languages

- **<u>GENERAL PURPOSE LANGUAGES</u>:**

  - C,C++,Pascal, or Lisp(control animation sequences).



General computer-animation functions

- Software package are developed for general animation design or performing specialized animation tasks.
- **Typical animation functions**
  - Include managing object motions, generating views of objects , producing camera-motions and the generation of in-between frames
- Some animation package provide functions for both the overall animation design and the processing of the individual objects
  - For example: Wavefront
- A set of routines is often provide for storage and managing the object database
  - Object shapes and associated parameters are stored and updated in the database.

Continued……….

- Develop routines to design and control animation sequences within a general-purpose programming language
- Specialized animation language have been developed
- Animation description
  - Scene description, includes the positioning of objects and light sources, defining the photometric parameters and setting the camera parameters
  - Action description, involves the layout of motion paths for the objects and camera
- Key-frame systems–Originally designed as a separate set of animation routines for generating the in-between from the user-specified key frames
- Parameterized systems–Allow object motion characteristics to be specified as part of the object definitions
- Scripting systems–Allow object description and animation sequences to be defined with a user-input script

SPECIALIZED ANIMATION LANGUAGES

- Key frame systems
- Parameterized systems
- Scripting systems

Key-frame Systems

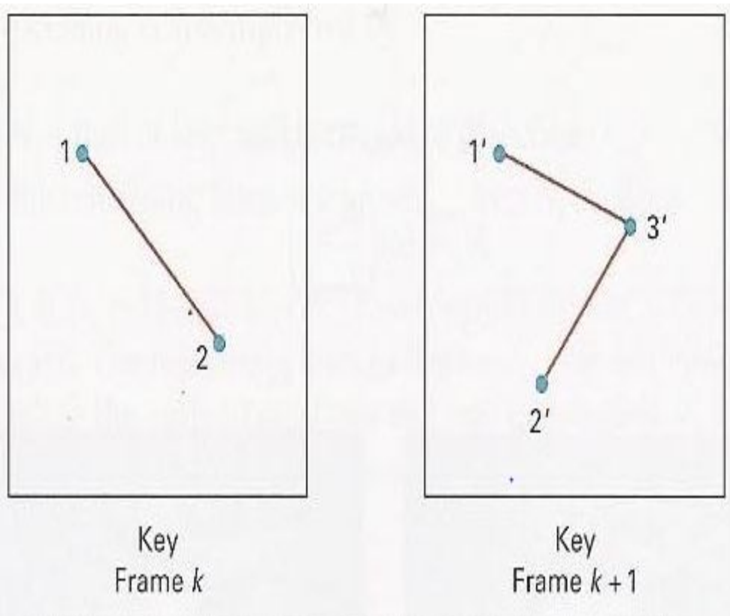
- A set of in-between can be generated from the specification of two (or more) key frames
- For complex scene, separate the frame into individual components or object called cels
  - Transparencies stacked in the order from background to foreground–Obtain complete frame by photographing transparencies
  - Obtain complete frame by photographing transparencies–Obtain the next cels for each character with the specified animations paths
- For complex object transformations
  - The shapes of objects may change overtime
  - Example: clothes, facial features and evolving shapes
- For surfaces described with polygon meshes–Changes can result in significant changes in polygon shape

- Morphing
  - Transformation of object shapes from one form to another
  - For two key frames with a different number of line segments specifying an object
- Adjust the object description in one of the frames
- To balance the number of polygon edges or the number of vertices
  - Example-Linear interpolation for transforming a triangle to a quadrilateral

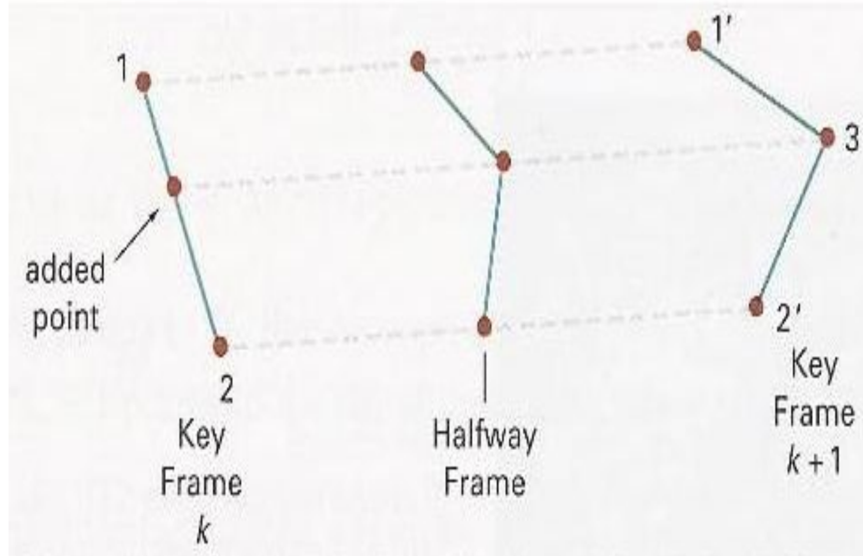
Morphing

- **Morphing** is a special effect in motion pictures and animations that changes (or morphs) one image or shape into another through a seamless transition. Traditionally such a depiction would be achieved through dissolving techniques on film.

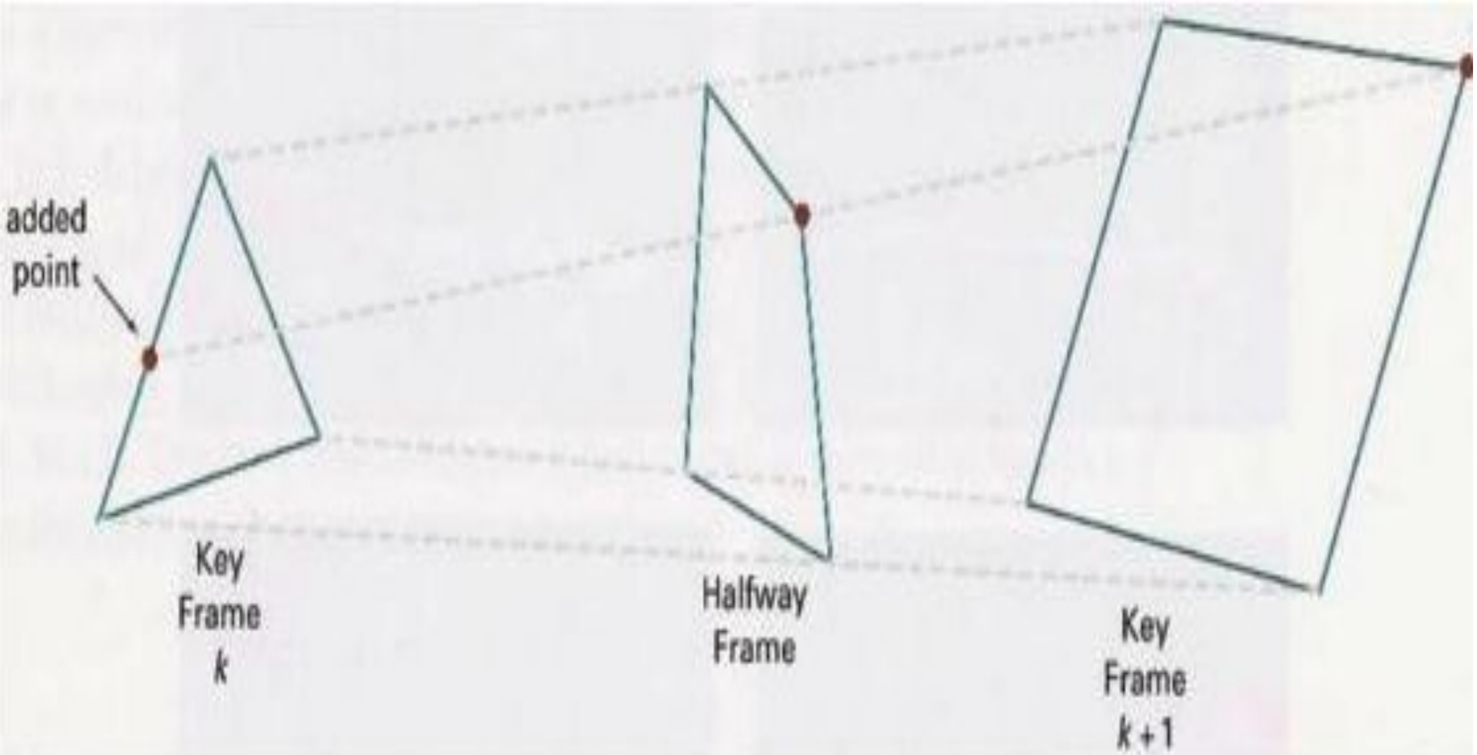
Key frame Systems



An edge with vertex positions 1 and 2 in key frame k evolves into two connected edges in key frame $k + 1$.



Linear interpolation for transforming a line segment in key frame k into two connected line segments in key frame $k + 1$.



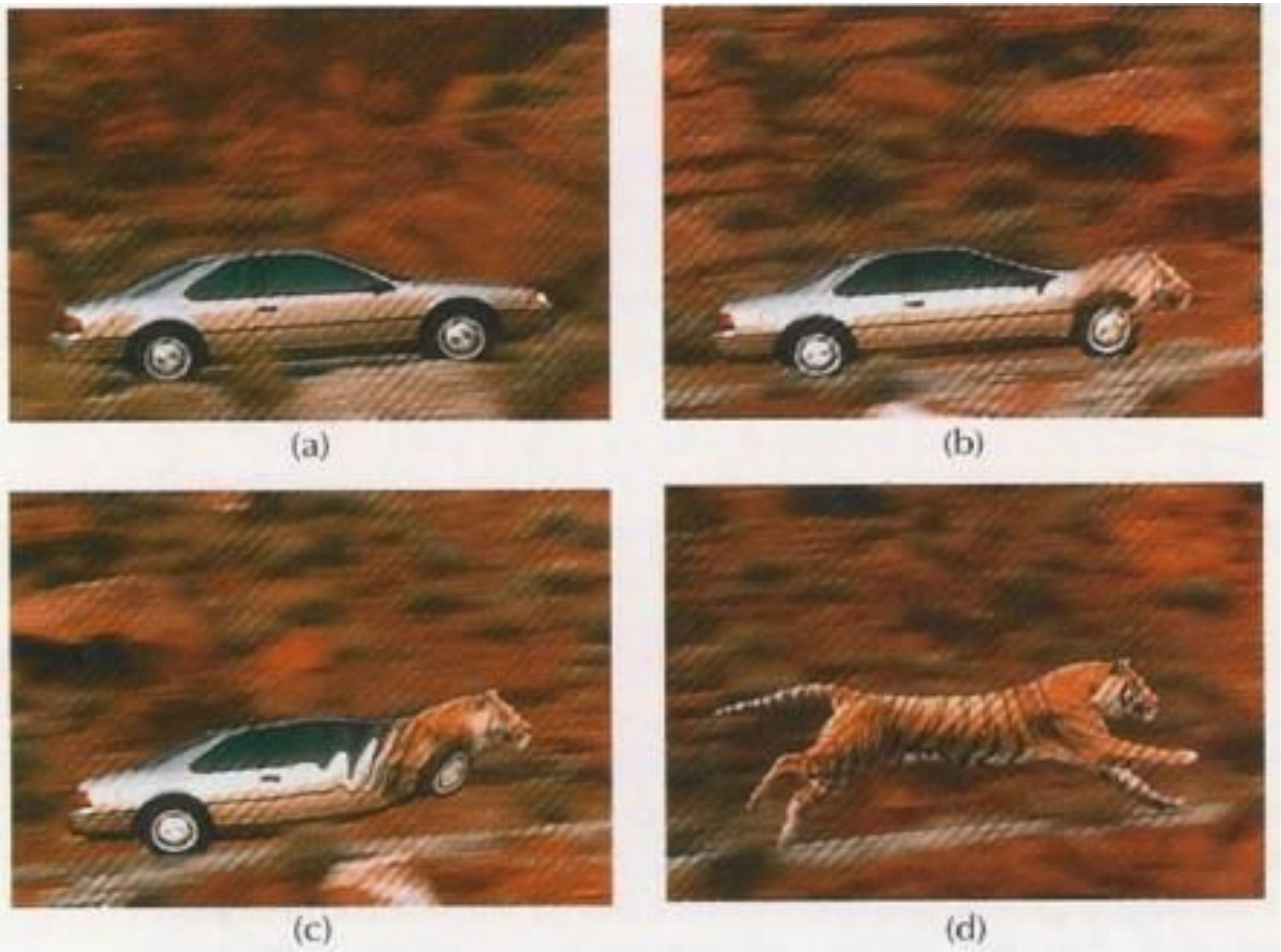
Linear interpolation for transforming a triangle into a quadrilateral.

Motion Specifications

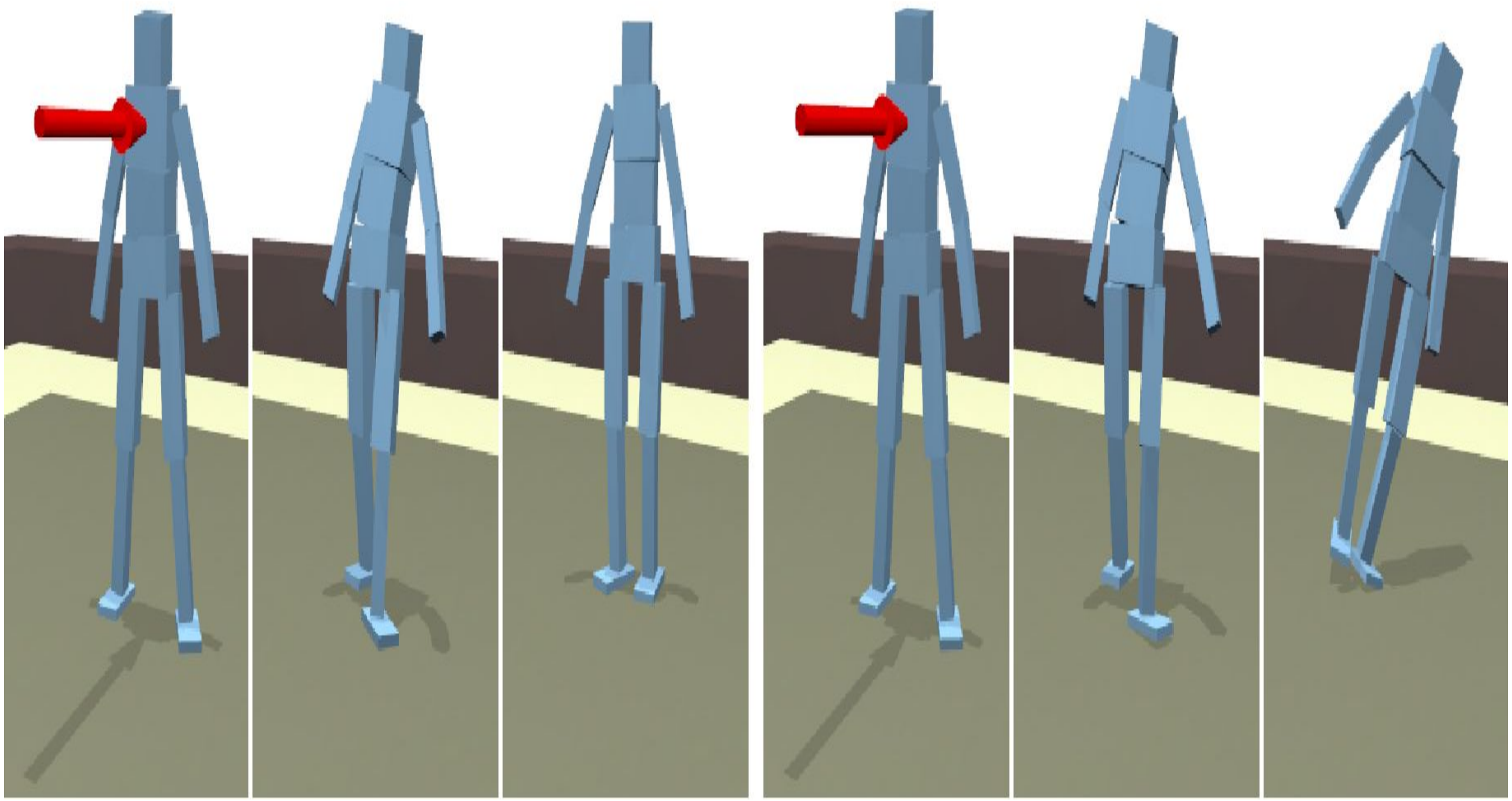
•Various ways in which motions of objects can be specified as:

- Direct Motion Specification.
- Goal-Directed Systems.
- Kinematics and Dynamics.

Direct Motion Specification



Goal Directed System



Kinematics and Dynamics

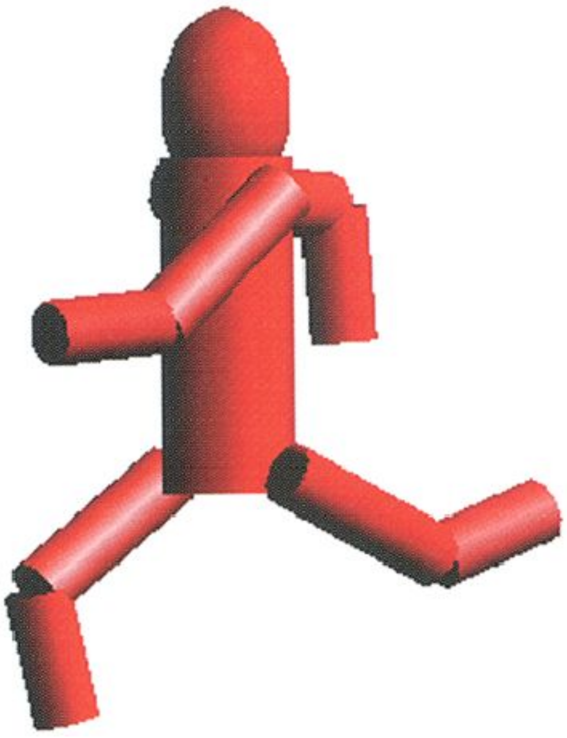
- **KINEMATICS**:
  - Motion parameters such as position , velocity and acceleration are specified without reference to the forces.
- **INVERSE KINEMATICS:**
  - Initial and final positions of objects at specified times and from that motion parameters .
- **DYNAMICS:**
  - The forces that produce the velocities and accelerations are specified(physically based modeling).
  - It uses laws such as Newton's laws of motion , Euler or Navier -stokes equations.

Outline

Principles of Animation

Keyframe Animation

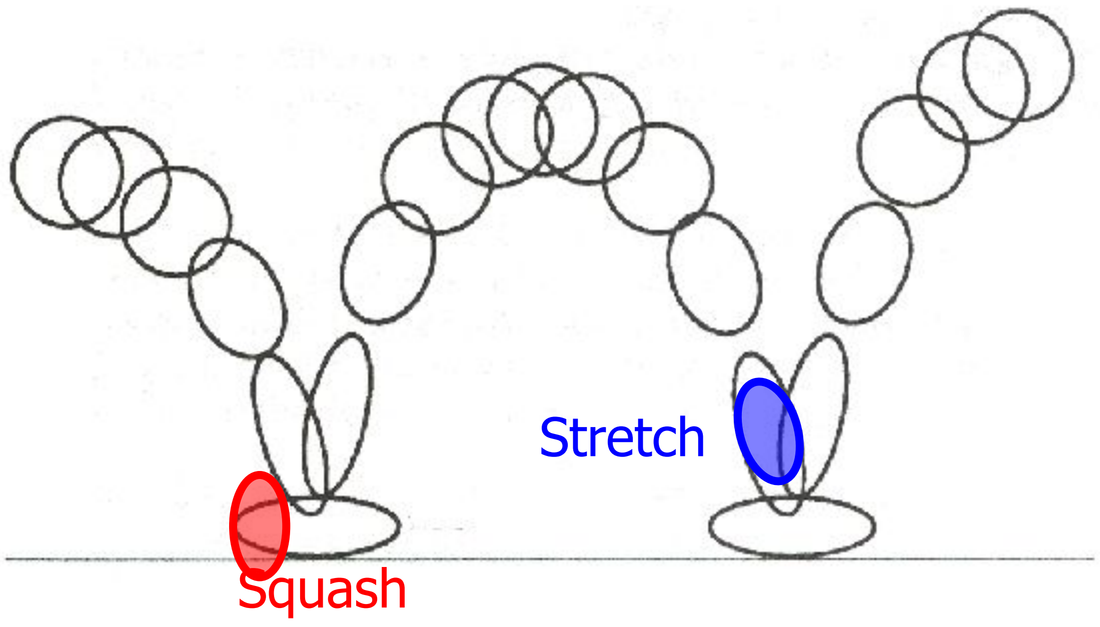
Articulated Figures



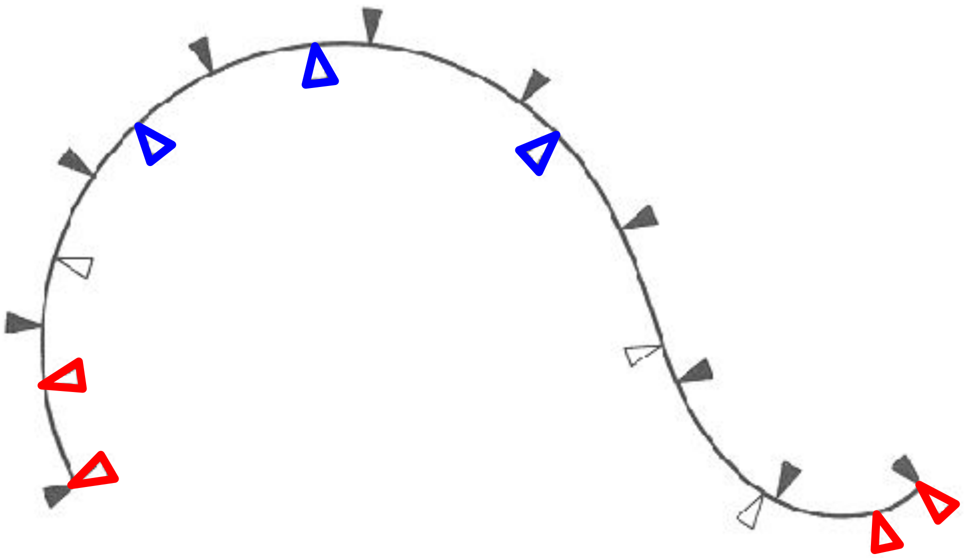
Principle of Traditional Animation

- Squash and Stretch
- Slow In and Out
- Anticipation
- Exaggeration
- Follow Through and Overlapping Action
- Timing
- Staging
- Straight Ahead Action and Pose-to-Pose Action
- Arcs
- Secondary Action
- Appeal

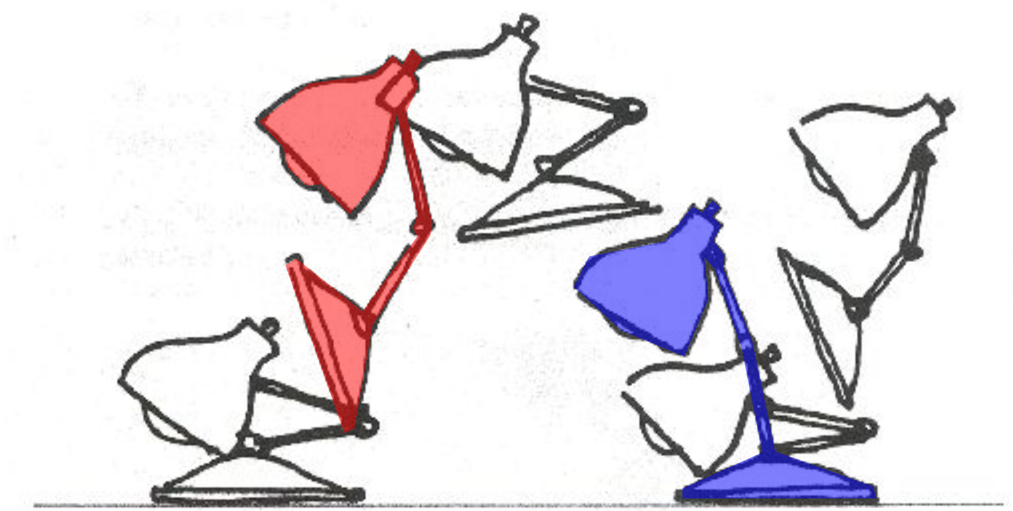
Squash and Stretch



Slow In and Out



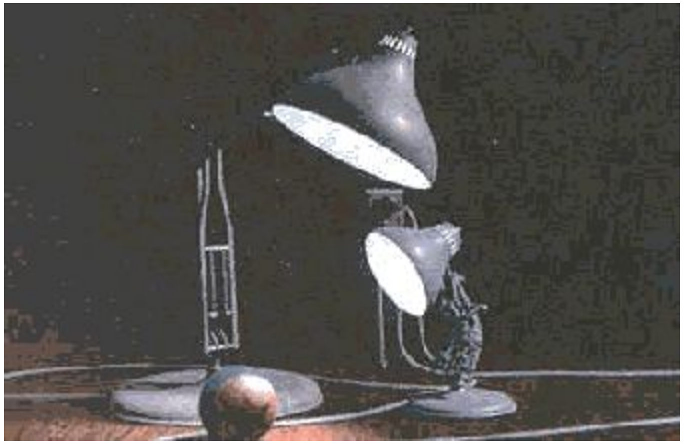
Anticipation



Computer Animation

Animation Pipeline

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

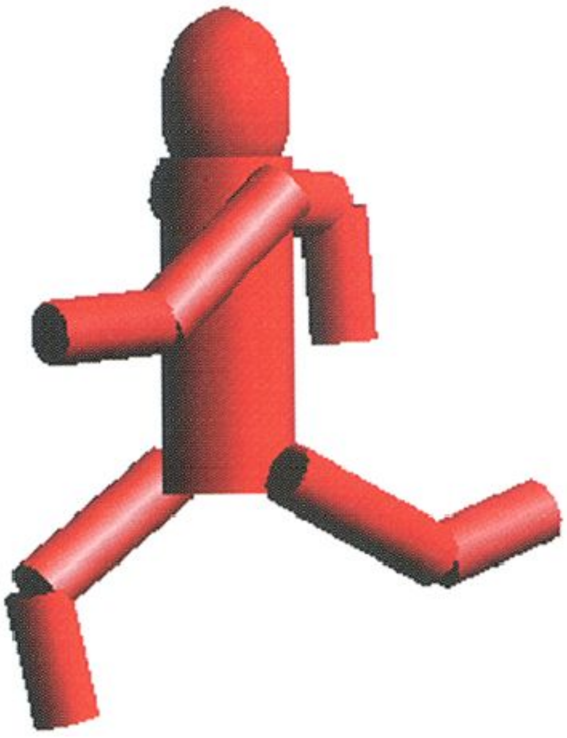


Outline

Principles of Animation

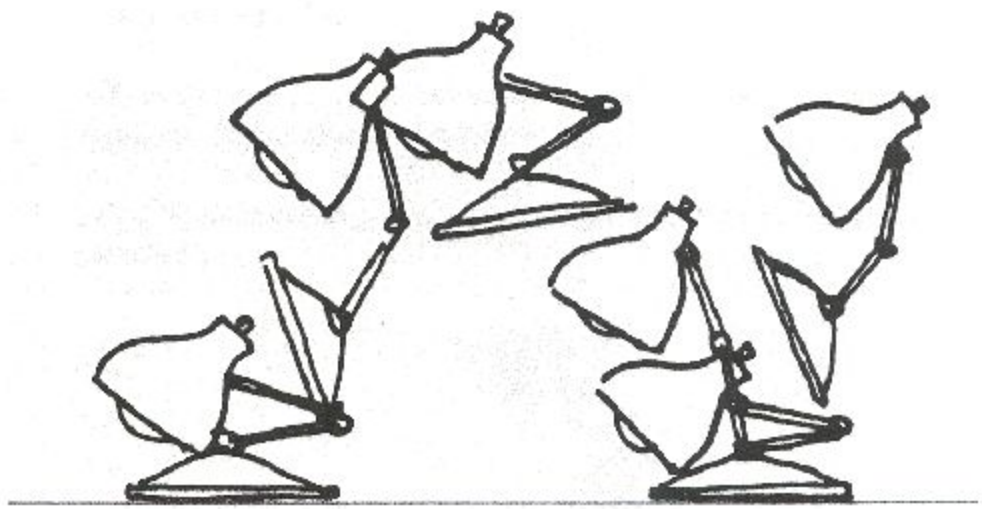
Keyframe Animation

Articulated Figures



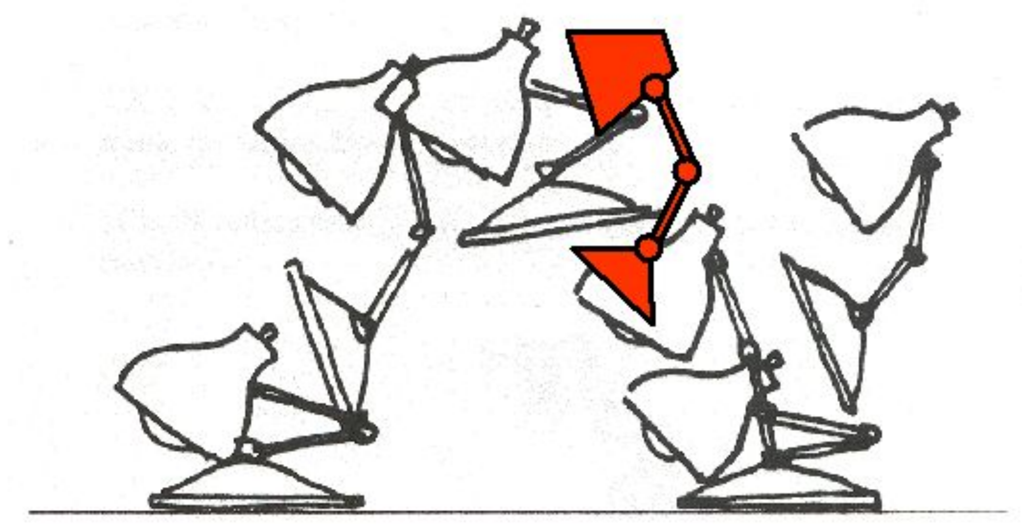
Keyframe Animation

Define Character Poses at Specific Time Steps Called "Keyframes"



Keyframe Animation

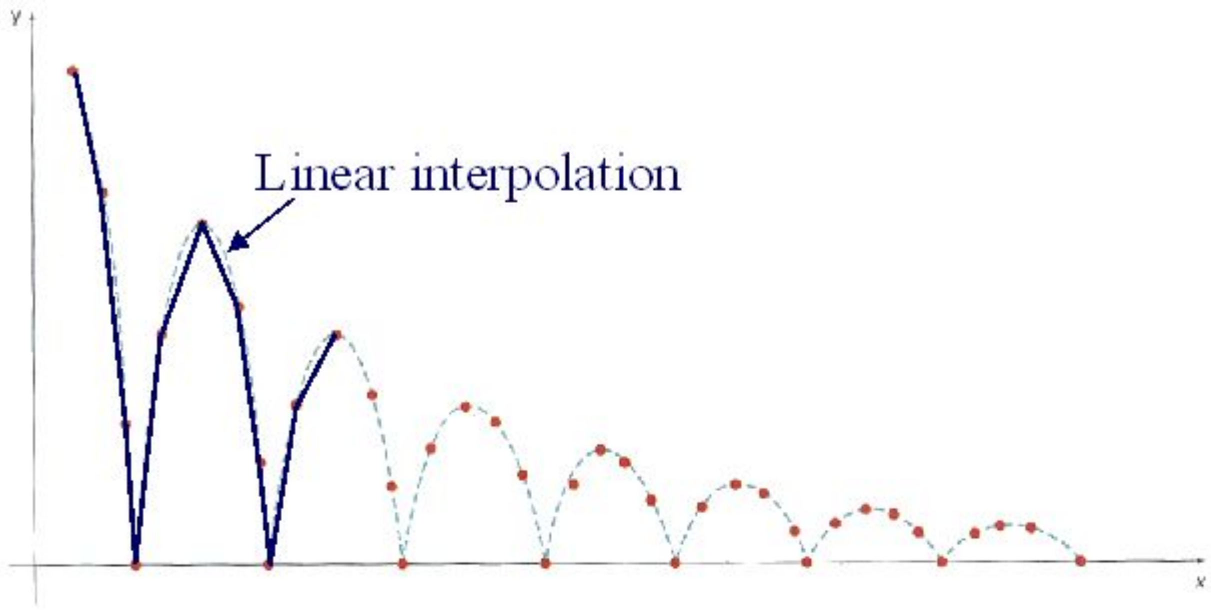
Interpolate Variables Describing Keyframes to Determine Poses for Character in between



Inbetweening

Linear Interpolation

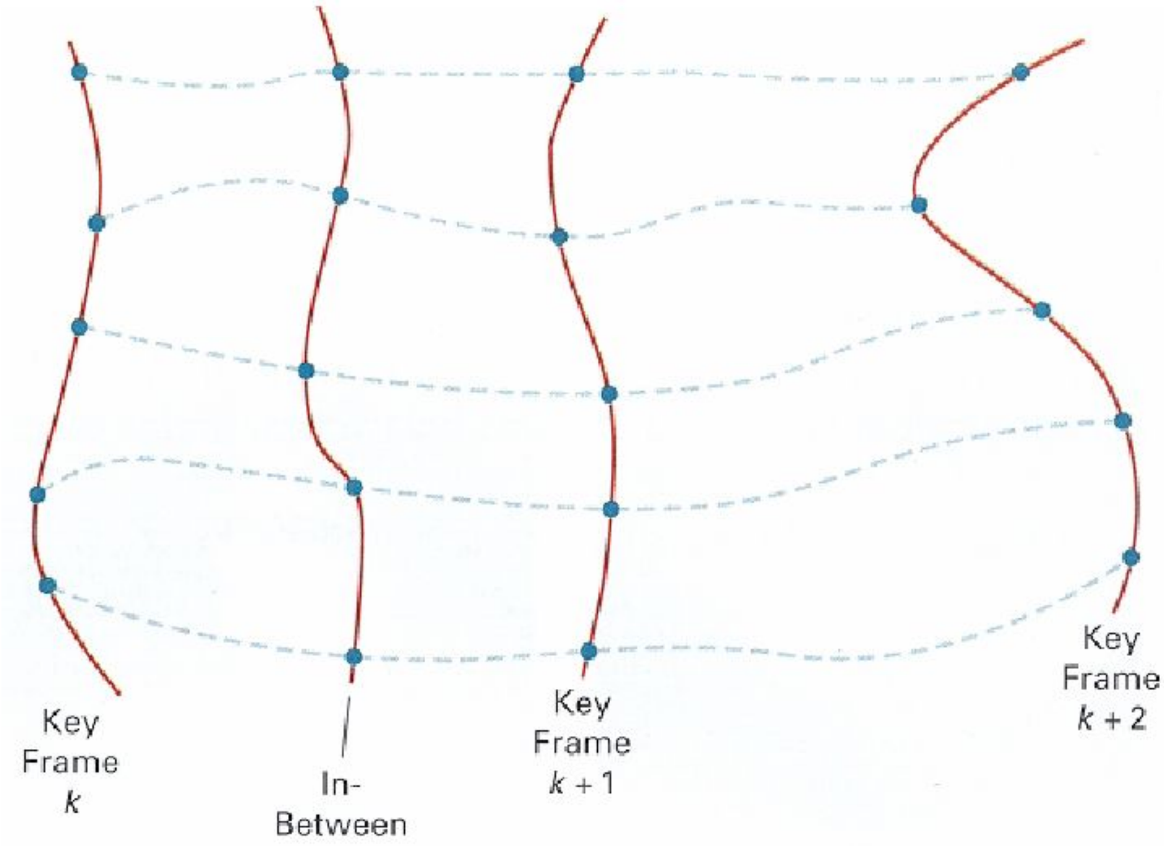
- Usually not enough continuity



Inbetweening

Spline Interpolation

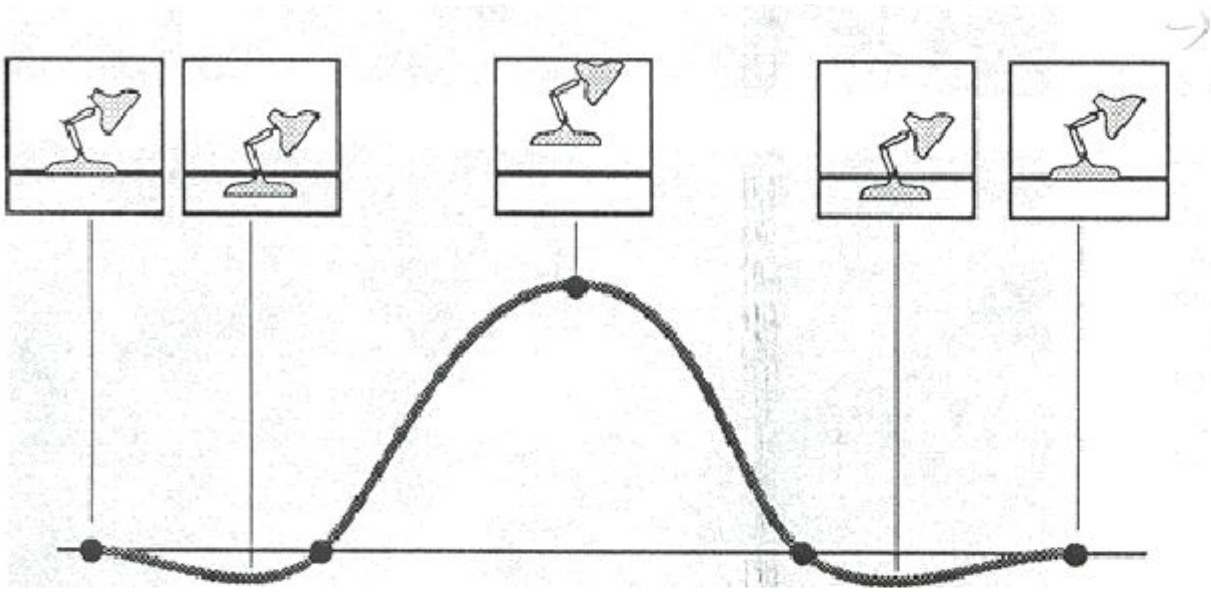
Maybe good enough



Inbetweening

Spline Interpolation

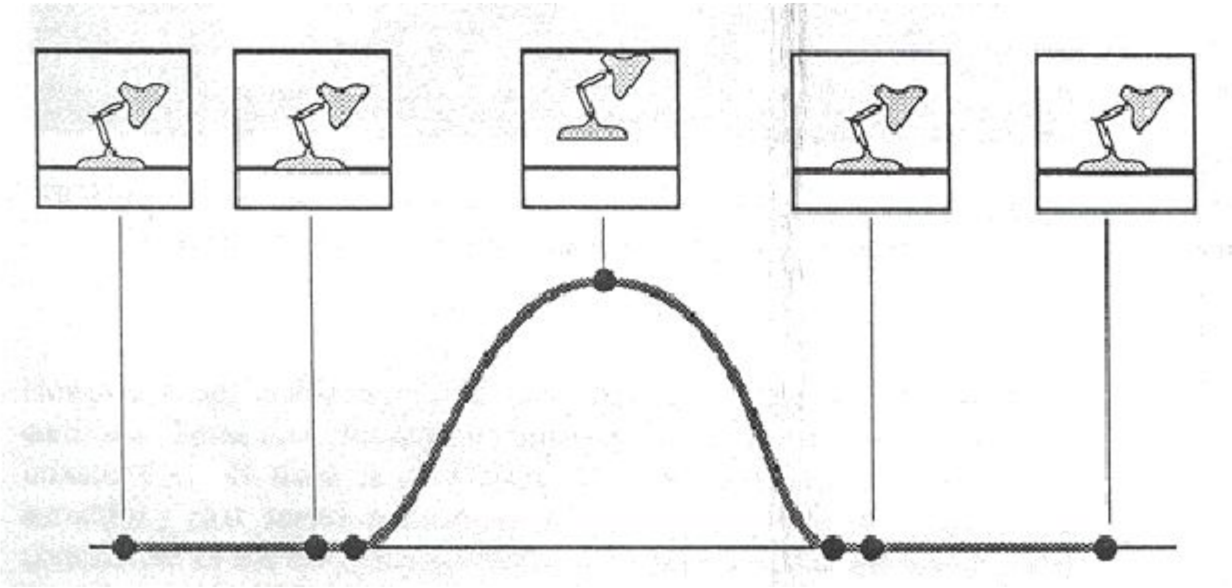
- Maybe good enough
  - May not follow physical laws



Inbetweening

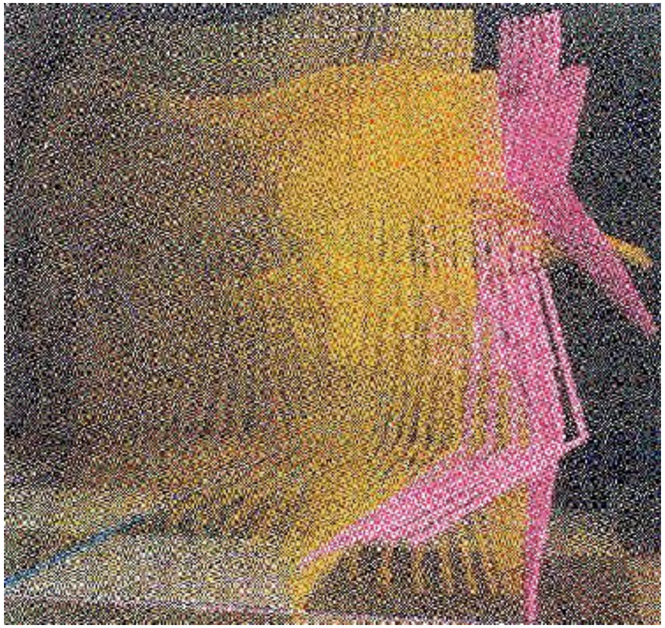
Spline Interpolation

- Maybe good enough
  - May not follow physical laws



Inbetweening

Inverse Kinematics or Dynamics



Summary

Animation Requires ...

- Modeling
- Scripting
- Inbetweening
- Lighting, shading
- Rendering
- Image processing

Overview

Kinematics

- Consider only motion
- Determined by positions, velocities, accelerations

Dynamics

- Consider underlying forces
- Compute motion from initial conditions and physics

Summary

Forward Kinematics

Specify conditions (joint angles)
Compute positions of end-effectors

Inverse Kinematics

"Goal-directed" motion
Specify goal positions of end effectors
Compute conditions required to achieve goals

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

Overview

Kinematics

Consider only motion

Determined by positions, velocities, accelerations

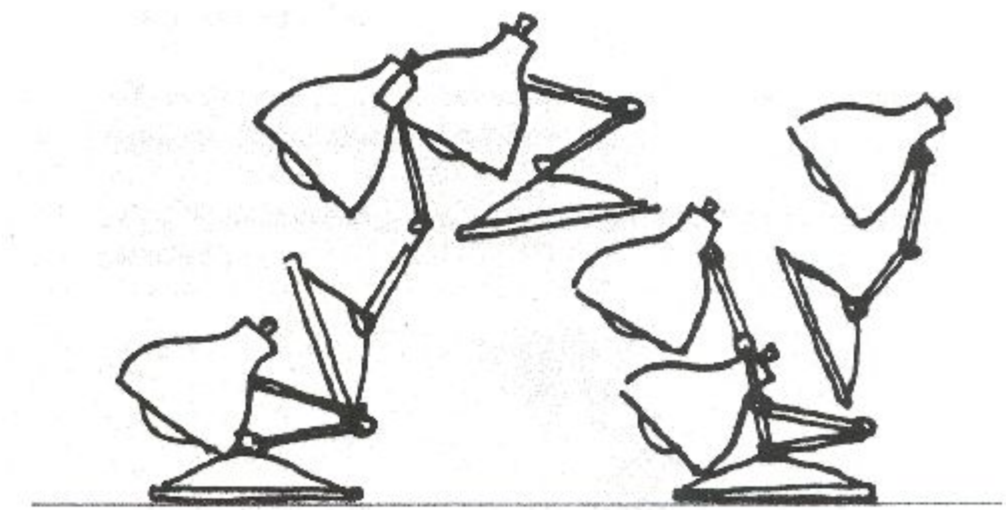
Dynamics

Consider underlying forces

Compute motion from initial conditions and physics

Dynamics

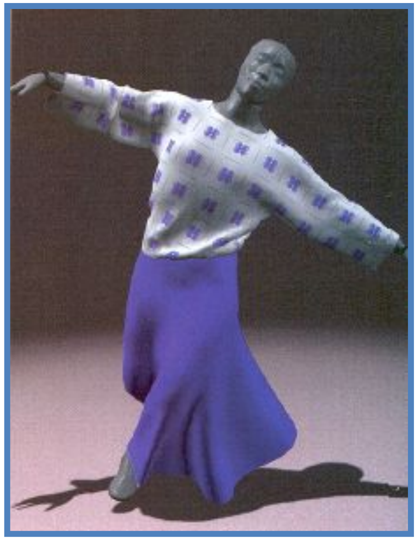
Simulation of Physics Insures Realism of Motion



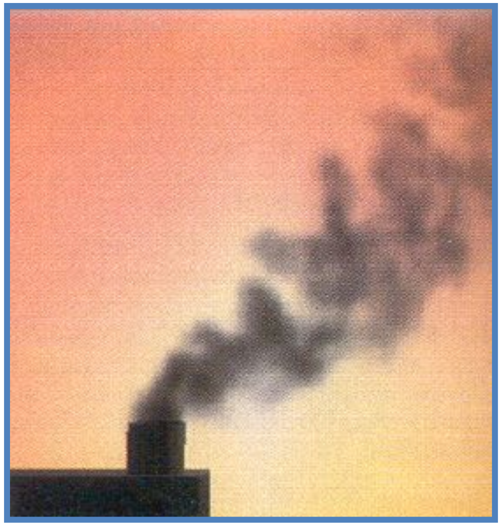
Dynamics

Other Physical Simulations

- Rigid bodies
- Soft bodies
- Cloth
- Liquids
- Gases
- etc.



Cloth



Hot Gases

Summary

Kinematics

- Forward kinematics
  - Animator specifies joints (hard)
  - Compute end-effectors (easy)
- Inverse kinematics
  - Animator specifies end-effectors (easier)
  - Solve for joints (harder)

Dynamics

- Space-time constraints
  - Animator specifies structures & constraints (easiest)
  - Solve for motion (hardest)
- Also other physical simulations