Creative Computing II

Christophe Rhodes
c.rhodes@gold.ac.uk

Autumn 2010, Wednesdays:
10:00–12:00: RHB307 & 14:00–16:00: WB316
Winter 2011, TBC

Animation

- *anima*, Latin: soul or mind;
- cognates: Greek *anemos*: wind; Sanskrit *aniti* breathes.
- "to bring to life"; "making things move"
- component of computer graphics:
  - realtime constraints for interactive media (*e.g.* games);
  - offline (non-realtime) rendering for film.

Animation

Two distinct approaches to computer animation:

- Physical modelling
  - Describe situation in (simplified) physical terms;
  - Simulate solution to equations of motion:
  - Use simulation to place animated objects appropriately.

Animation

Two distinct approaches to computer animation:

- Physical modelling
  - Describe situation in (simplified) physical terms;
  - Simulate solution to equations of motion:
  - Use simulation to place animated objects appropriately.
- Keyframing
  - Specify positions of animated objects at particular times;
  - Interpolate between specified position to achieve desired smoothness.

Given equation of motion:

$$\ddot{\mathbf{x}}_i = \frac{\mathrm{d}^2\mathbf{x}_i}{\mathrm{d}t^2} = f_i(\mathbf{x})$$

Solution is:

$$\begin{aligned}
\mathbf{x}_i(t+\delta t) &= \mathbf{x}_i(t) + \dot{\mathbf{x}}_{\mathbf{i}}(t)\delta t; \\
\dot{\mathbf{x}}_i(t+\delta t) &= \dot{\mathbf{x}}_i(t) + f_i(\mathbf{x})\delta t.
\end{aligned}$$

Given equation of motion:

$$\ddot{\mathbf{x}}_i = \frac{\mathrm{d}^2\mathbf{x}_i}{\mathrm{d}t^2} = f_i(\mathbf{x})$$

Solution is:

$$\begin{aligned}\mathbf{x}_i(t+\delta t) &= \mathbf{x}_i(t) + \dot{\mathbf{x}}_\mathbf{i}(t)\delta t;\\ \dot{\mathbf{x}}_i(t+\delta t) &= \dot{\mathbf{x}}_i(t) + f_i(\mathbf{x})\delta t.\end{aligned}$$

So given starting positions $\mathbf{x}(t_0)$ and velocities $\dot{\mathbf{x}}(t_0)$, we can evolve the physical model forwards (and backwards) in time.

Observations:

- $f_i(\mathbf{x})$ may be difficult to derive;
- even once derived, $f_i(\mathbf{x})$ may be difficult to compute;
- what if we want to specify initial and final positions?

Observations:

- $f_i(\mathbf{x})$ may be difficult to derive;
- even once derived, $f_i(\mathbf{x})$ may be difficult to compute;
- what if we want to specify initial and final positions?
- difficult to handle *agency* in this framework;
  - what is the brain's equation of motion?
- suitable for:
  - visualisation of simulations (particle systems, physics, "artificial life");
  - animation of very simple situations.

Animation

Keyframing

- the frame where an object's position is specified is a 'key frame' for that object;
- $\mathbf{x}_i(t_k) = \mathbf{x}_i^{(k)}$
- how to decide suitable times and positions?
  - rôle of the *lead animator*;
  - data-driven approach (motion capture).

Animation

Keyframing

- the frame where an object's position is specified is a 'key frame' for that object;
- $\mathbf{x}_i(t_k) = \mathbf{x}_i^{(k)}$
- how to decide suitable times and positions?
  - rôle of the *lead animator*;
  - data-driven approach (motion capture).
- interpolation:
  - traditionally: frames in between filled in by *assistant animators*;
  - computationally: generate a suitable path for object between key frames;
  - 'in-betweening' or 'tweening'.

Animation

Flip Books

- Fundamental method of animation;
- Existed in Victorian era;
- Present a sequence of pictures in rapid succession:
  - smooth *motion* illusion for small differences between pictures.
- Animation on modern displays (TV, cinema, computers) is essentially a flip book.

Animation

Cel Animation

- Similar to flip books;
- Each frame is an image, drawn individually;

Animation

Cel Animation

- Similar to flip books;
- Each frame is an image, drawn individually;
- Layering:
  - Distinguish between objects and background:
  - Use a single non-animated image for the background;
  - Use transparent layers for animated objects;
  - Camera motion can still pan over the background.

Animation

Cel Animation

- Similar to flip books;
- Each frame is an image, drawn individually;
- Layering:
  - Distinguish between objects and background:
  - Use a single non-animated image for the background;
  - Use transparent layers for animated objects;
  - Camera motion can still pan over the background.
- "Shooting on twos":
  - For slow motion, only generate half as many new object positions;
  - For fast motion, need all frames different for smooth motion illusion.

Animation

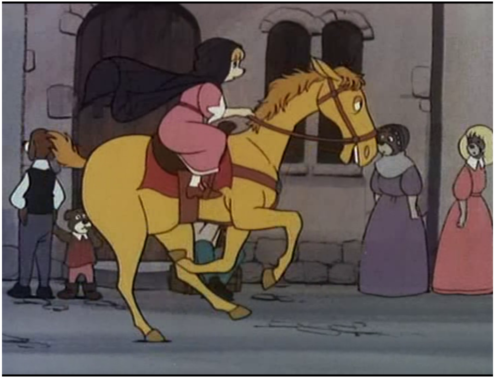
Cel Animation

- Similar to flip books;
- Each frame is an image, drawn individually;
- Layering:
  - Distinguish between objects and background:
  - Use a single non-animated image for the background;
  - Use transparent layers for animated objects;
  - Camera motion can still pan over the background.
- "Shooting on twos":
  - For slow motion, only generate half as many new object positions;
  - For fast motion, need all frames different for smooth motion illusion.
- Animation Loops:
  - Suitable for repetitive motion (*e.g.* walking);
  - Easily-detected, can be distracting.

Animation

Cel Animation

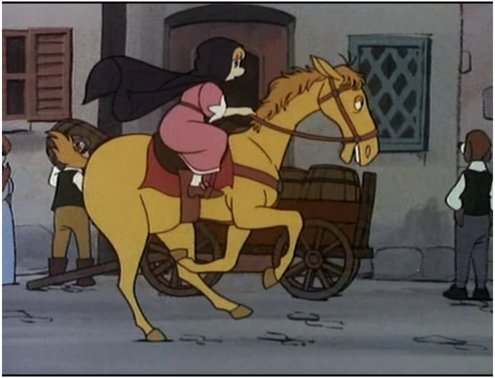
Animation Loop:



Animation

Cel Animation

Animation Loop:



Animation

Stop-motion Animation

Basic workflow:

- Create models of scene and characters;
- Pose, take a photo;
- Move slightly, take a photo;
- Lather, rinse, repeat.

Observations:

- Huge effort in initial creation;
- Each individual frame is not much extra work;
- Very time-consuming.

Animation

Stop-motion Animation

Modern computer animation is quite similar to traditional stop-motion animation:

- Generate a computational (virtual) model of scene and characters;

Modern computer animation is quite similar to traditional stop-motion animation:

- Generate a computational (virtual) model of scene and characters;
- Animate by moving the characters frame-by-frame:
  - character has multiple control points (joints on a ‘skeleton’);
  - move control points quasi-independently;
  - ‘rigging’ a character.

Animation

Stop-motion Animation

Modern computer animation is quite similar to traditional stop-motion animation:

- Generate a computational (virtual) model of scene and characters;
- Animate by moving the characters frame-by-frame:
  - character has multiple control points (joints on a ‘skeleton’);
  - move control points quasi-independently;
  - ‘rigging’ a character.
- Generating successive frames is easy.

Animation

Keyframing

In computer animation context, keyframing is a labour-saving device:

- animator specifies positions of control points at suitable times;
- *computer* performs in-betweening.

Animation

Keyframing

In computer animation context, keyframing is a labour-saving device:

- animator specifies positions of control points at suitable times;
- *computer* performs in-betweening.
- Note:
  - key frame is not an image
  - cf. key frames in cel animation
- in-betweening by interpolation
  - linear interpolation;
  - splines (Bézier curves, Hermite polynomials).

Animation

Interpolation

Linear interpolation between $(\mathbf{x}_0, t_0)$ and $(\mathbf{x}_1, t_1)$:

- at time t_0, position is $\mathbf{x}_0$;
- at time t_1, position is $\mathbf{x}_1$;
- in between those times, position is
  - on the line joining $\mathbf{x}_0$ and $\mathbf{x}_1$;
  - appropriate for motion at a constant velocity;
  - (so position is a linear function of time).

Animation

Interpolation

Linear interpolation between $(\mathbf{x}_0, t_0)$ and $(\mathbf{x}_1, t_1)$:

- at time t_0, position is $\mathbf{x}_0$;
- at time t_1, position is $\mathbf{x}_1$;
- in between those times, position is
  - on the line joining $\mathbf{x}_0$ and $\mathbf{x}_1$;
  - appropriate for motion at a constant velocity;
  - (so position is a linear function of time).

So for $t_0 \leq t \leq t_1$

$$\mathbf{x}(t) = \mathbf{x}_0 + \frac{\mathbf{x}_1 - \mathbf{x}_0}{t_1 - t_0}(t - t_0)$$

Animation

Interpolation

Linear interpolation between $(\mathbf{x}_0, t_0)$ and $(\mathbf{x}_1, t_1)$:

- at time t_0, position is $\mathbf{x}_0$;
- at time t_1, position is $\mathbf{x}_1$;
- in between those times, position is
  - on the line joining $\mathbf{x}_0$ and $\mathbf{x}_1$;
  - appropriate for motion at a constant velocity;
  - (so position is a linear function of time).

So for $t_0 \leq t \leq t_1$

$$\mathbf{x}(t) = \mathbf{x}_0 + \frac{\mathbf{x}_1 - \mathbf{x}_0}{t_1 - t_0}(t - t_0)$$

Observations:

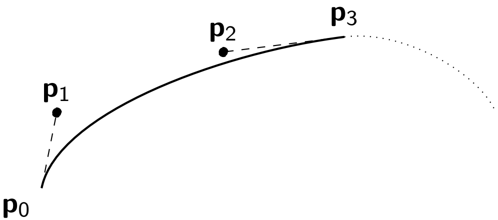
- Motion during interpolation is smooth;
- Jerkiness at change between interpolated segments.

Animation

Interpolation

Cubic Bézier curves: for each segment, specify

- two curve end points $\mathbf{p}_0$, $\mathbf{p}_3$;
- two control points $\mathbf{p}_1$, $\mathbf{p}_2$;
- for $0 \leq t \leq 1$:
  $\mathbf{x}(t) = (1-t)^3\mathbf{p}_0 + 3t(1-t)^2\mathbf{p}_1 + 3t^2(1-t)\mathbf{p}_2 + t^3\mathbf{p}_3$.

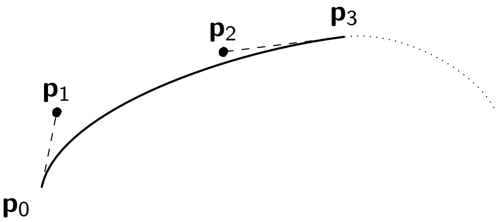


Animation

Interpolation

Cubic Bézier curves: for each segment, specify

- two curve end points $\mathbf{p}_0$, $\mathbf{p}_3$;
- two control points $\mathbf{p}_1$, $\mathbf{p}_2$;
- for $0 \leq t \leq 1$:
  $\mathbf{x}(t) = (1-t)^3\mathbf{p}_0 + 3t(1-t)^2\mathbf{p}_1 + 3t^2(1-t)\mathbf{p}_2 + t^3\mathbf{p}_3$.



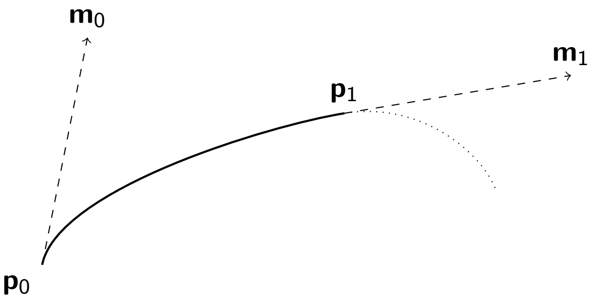
- smooth curve;
- does not go through control points.

Animation

Interpolation

Cubic Hermite curves: for each segment, specify

- two curve end points $\mathbf{p}_0$, $\mathbf{p}_1$;
- tangents at those end points $\mathbf{m}_0$, $\mathbf{m}_1$;
- for $0 \leq t \leq 1$: $\mathbf{x}(t) = (2t^3 - 3t^2 + 1)\mathbf{p}_0 + (t^3 - 2t^2 + t)\mathbf{m}_0 + (-2t^3 + 3t^2)\mathbf{p}_1 + (t^3 - t^2)\mathbf{m}_1$

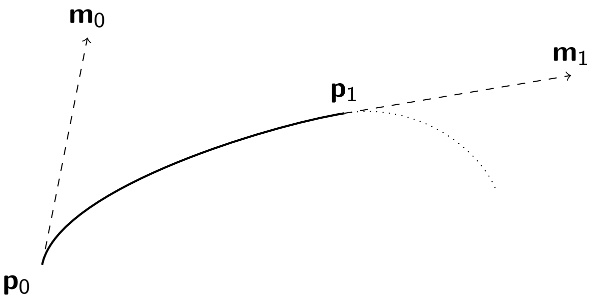


Animation

Interpolation

Cubic Hermite curves: for each segment, specify

- two curve end points $\mathbf{p}_0$, $\mathbf{p}_1$;
- tangents at those end points $\mathbf{m}_0$, $\mathbf{m}_1$;
- for $0 \leq t \leq 1$: $\mathbf{x}(t) = (2t^3-3t^2+1)\mathbf{p}_0+(t^3-2t^2+t)\mathbf{m}_0+(-2t^3+3t^2)\mathbf{p}_1+(t^3-t^2)\mathbf{m}_1$



- smooth curve;
- choose tangents automatically.

Catmull-Rom splines:

- Hermite curves;
- Tangents chosen as half the distance between surrounding keyframes:
- $\mathbf{m}_k = \frac{\mathbf{p}_{k+1} - \mathbf{p}_{k-1}}{2}$
- Smooth motion over multiple segments;
- Zero-tangents at first and last keyframe give ‘slow-in slow-out’ behaviour.