Animation & Simulation

He Wang (王鹤)

Interpolating Values

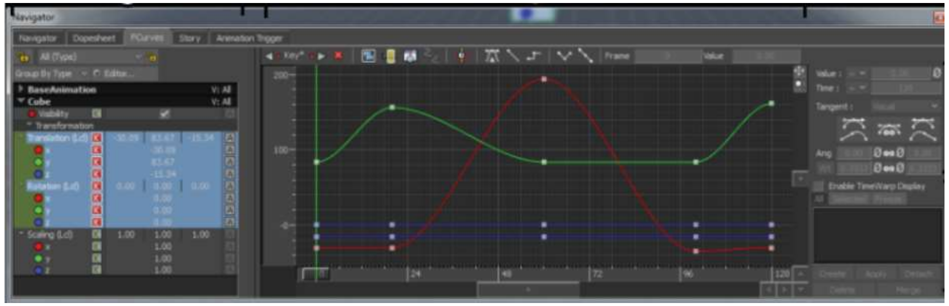
- Basic concepts
  - Explicit
  - Implicit
  - Parametric
  - Polynomials
  - Other functions: trigonometrics, logs, called *transcendental* here
  - Continuity, nth-order continuous derivatives
  - A function formed by curve segments has *piece-wise* properties
  - Distinction between parametric continuity and geometric continuity

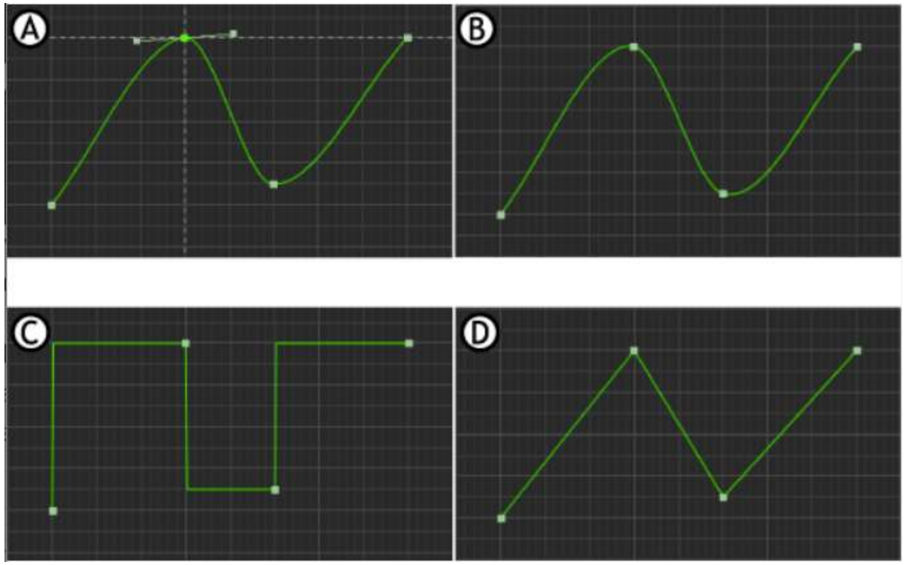
Interpolating Values

- Interpolation
  - Hermite interpolation
  - Catmull-Rom Spline
  - Four-point form (fit a cubic curve to four points)
  - Blended parabolas
  - Bezier interpolation/approximation
  - Tension, continuity and bias control
  - B-splines

Interpolating Values

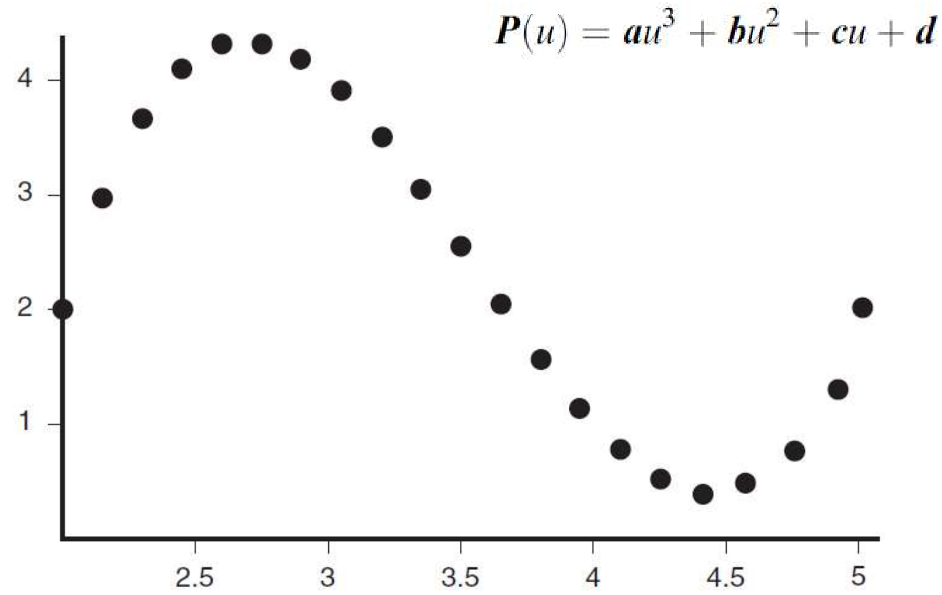
- Interpolation
  - MotionBuilder





Interpolating Values

- Controlling the motion
  - Given a motion function, control higher-order information (speed, acc.)
  - Need to control them by arc length (instead of raw coordinates)

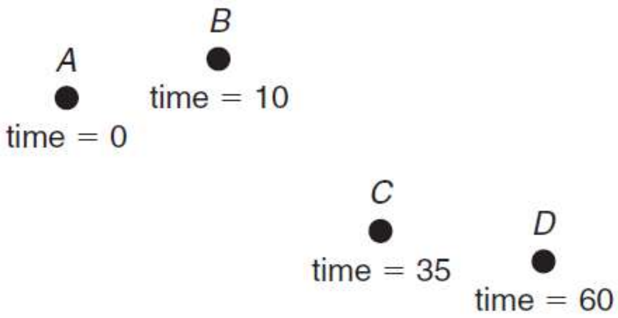


Equally increment on parameters
of a cubic curve.

Not evenly distributed on the curve!

Interpolating Values

- Controlling the motion
    - Computing arc length (Analytic)



Assume $\boldsymbol{P}(u) = \boldsymbol{a}u^3 + \boldsymbol{b}u^2 + \boldsymbol{c}u + \boldsymbol{d}$

$$\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}$$

Interpolating Values

- Controlling the motion
  - Computing arc length (Analytic)
    - Only a space curve
    - Distance-time function
      - distance=arc length

Assume $\boldsymbol{P}(u) = \boldsymbol{a}u^3 + \boldsymbol{b}u^2 + \boldsymbol{c}u + \boldsymbol{d}$

$$
\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}
$$

Interpolating Values

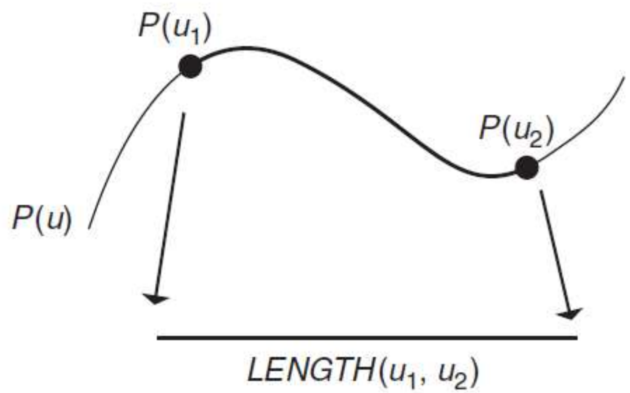
- Controlling the motion
  - Computing arc length (Analytic)

    Say arc length is s, s = S(u) where S is the function of variable u

    Equally, u = U(s)

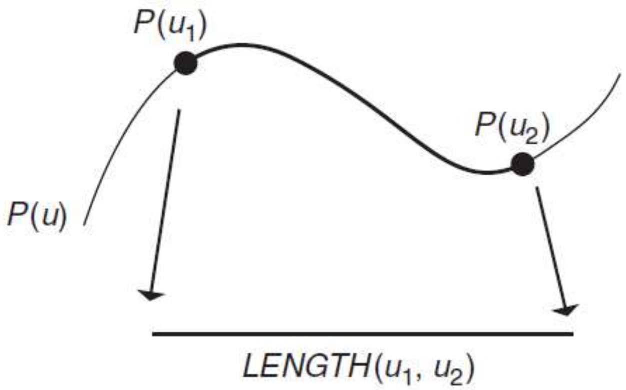
$$
\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}
$$

1. Given parameters u_a and u_b, find $LENGTH(u_a, u_b)$.
2. Given an arc length s and a parameter value u_a, find u_b such that $LENGTH(u_a, u_b) = \mathbf{s}$. This is equivalent to finding the solution to the equation $s - LENGTH(u_a, u_b) = 0$.



Interpolating Values

- Controlling the motion
  - Computing arc length (Analytic)



$$\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}$$

$$s = \int_{u_a}^{u_b} |dP/du|\,du$$

$$d\boldsymbol{P}/du = ((dx(u)/(du)), (dy(u)/(du)), (dz(u)/(du)))$$

$$|d\boldsymbol{P}/du| = \sqrt{(dx(u)/du)^2 + (dy(u)/du)^2 + (dz(u)/du)^2}$$

Interpolating Values

- Controlling the motion
  - Computing arc length (Analytic)

$$
\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}
$$

$$
s = \int_{u_a}^{u_b} |dP/du| du
$$

$$
d\boldsymbol{P}/du = ((dx(u)/(du)), (dy(u)/(du)), (dz(u)/(du)))
$$

$$
|d\boldsymbol{P}/du| = \sqrt{(dx(u)/du)^2 + (dy(u)/du)^2 + (dz(u)/du)^2}
$$

$$
\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}
$$

$$
dx(u)/du = 3a_x u2 + 2b_x u + c_x
$$

$$
Au^4 + Bu^3 + Cu^2 + Du + E
$$

Interpolating Values

- Controlling the motion
  - Computing arc length (Analytic)

$$
\begin{aligned}
\boldsymbol{P}(u) &= (x(u),\ y(u),\ z(u)) \\
x(u) &= a_x u^3 + b_x u^2 + c_x u + d_x \\
y(u) &= a_y u^3 + b_y u^2 + c_y u + d_y \\
z(u) &= a_z u^3 + b_z u^2 + c_z u + d_z
\end{aligned}
$$

$$dx(u)/du = 3a_x u2 + 2b_x u + c_x$$

$$Au^4 + Bu^3 + Cu^2 + Du + E$$

A 2D case

$$
\begin{aligned}
A &= 9(a_x^2 + a_y^2) \\
B &= 12(a_x b_x + a_y b_y) \\
C &= 6(a_x c_x + a_y c_y) + 4(b_x^2 + b_y^2) \\
D &= 4(b_x c_x + b_y c_y) \\
E &= c_x^2 + c_y^2
\end{aligned}
$$

Interpolating Values

- Controlling the motion
  - Computing arc length (Analytic)
  - Computing arc length (forward differencing)

    Not all curves can be analytically represented and solved

Interpolating Values

- Controlling the motion
  - Computing arc length (forward differencing)

    $\boldsymbol{P}(u)$, u = 0.00, 0.05, 0.10, 0.15…

$G[0] = 0.0$
$G[1] =$ the distance between $\boldsymbol{P}(0.00)$ and $\boldsymbol{P}(0.05)$
$G[2] = G[1]$ plus the distance between $\boldsymbol{P}(0.05)$ and $\boldsymbol{P}(0.10)$
$G[3] = G[2]$ plus the distance between $\boldsymbol{P}(0.10)$ and $\boldsymbol{P}(0.15)$
. . .
$G[20] = G[19]$ plus the distance between $\boldsymbol{P}(0.95)$ and $\boldsymbol{P}(1.00)$

Interpolating Values

- Controlling the motion
  - Computing arc length (forward differencing)

Table 3.1 Parameter, Arc Length Pairs

Index	Parametric Value (V)	Arc Length (G)
0	0.00	0.000
1	0.05	0.080
2	0.10	0.150
3	0.15	0.230
4	0.20	0.320
5	0.25	0.400
6	0.30	0.500
7	0.35	0.600
8	0.40	0.720
9	0.45	0.800
10	0.50	0.860
11	0.55	0.900
12	0.60	0.920
13	0.65	0.932

Interpolating Values

- Controlling the motion
  - Computing arc length (forward differencing)

    P(u), u = 0.00, 0.05, 0.10, 0.15…

    find the closest in the table, v = 0.73, d = 0.05

$$i = \left\lfloor \frac{v}{d} + 0.5 \right\rfloor = \left\lfloor \frac{0.73}{0.05} + 0.5 \right\rfloor = 15$$

Table 3.1 Parameter, Arc Length Pairs

Index	Parametric Value (*V*)	Arc Length (*G*)
0	0.00	0.000
1	0.05	0.080
2	0.10	0.150
3	0.15	0.230
4	0.20	0.320
5	0.25	0.400
6	0.30	0.500
7	0.35	0.600
8	0.40	0.720
9	0.45	0.800
10	0.50	0.860
11	0.55	0.900
12	0.60	0.920
13	0.65	0.932
14	0.70	0.944
15	0.75	0.959
16	0.80	0.972
17	0.85	0.984
18	0.90	0.994
19	0.95	0.998

Interpolating Values

- Controlling the motion
  - Computing arc length (forward differencing)

    P(*u*), *u* = 0.00, 0.05, 0.10, 0.15…

    interpolate between two entries

$$i = \left\lfloor \frac{v}{d} \right\rfloor = \left\lfloor \frac{0.73}{0.05} \right\rfloor = 14$$

$$i = \left\lfloor \frac{v}{d} + 0.5 \right\rfloor = \left\lfloor \frac{0.73}{0.05} + 0.5 \right\rfloor = 15$$

$$\begin{aligned} s &= G[i] + \frac{v - V[i]}{V[i+1] - V[i]}(G[i+1] - G[i]) \\ &= 0.944 + \frac{0.73 - 0.70}{0.75 - 0.70}(0.959 - 0.944) \\ &= 0.953 \end{aligned}$$

Table 3.1 Parameter, Arc Length Pairs

Index	Parametric Value (*V*)	Arc Length (*G*)
0	0.00	0.000
1	0.05	0.080
2	0.10	0.150
3	0.15	0.230
4	0.20	0.320
5	0.25	0.400
6	0.30	0.500
7	0.35	0.600
8	0.40	0.720
9	0.45	0.800
10	0.50	0.860
11	0.55	0.900
12	0.60	0.920
13	0.65	0.932
14	0.70	0.944
15	0.75	0.959
16	0.80	0.972
17	0.85	0.984
18	0.90	0.994
19	0.95	0.998

Interpolating Values

- Controlling the motion
  - Computing arc length (forward differencing)

~~1. Given parameters u_a and u_b, find $LENGTH(u_a, u_b)$.~~

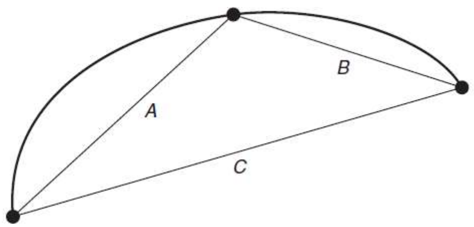
2. Given an arc length s and a parameter value u_a, find u_b such that $LENGTH(u_a, u_b) = \mathbf{s}$. This is equivalent to finding the solution to the equation $s - LENGTH(u_a, u_b) = 0$.

- Search table for s or closest to s, then work out u_b
  - Binary search to find the closest s' to s.
  - Interpolate two *u*s and add a difference to u_a
  - Pros: Easy to implement, quick
  - Cons: errors from both values and parameters
  - Mitigation: subsampling curves, higher-order interpolation

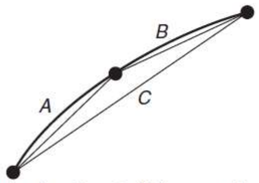
Interpolating Values

- Controlling the motion
  - Computing arc length (*Adaptive* forward differencing)
    - Same as before
    - Test if error is above some threshold, if so, split it into two

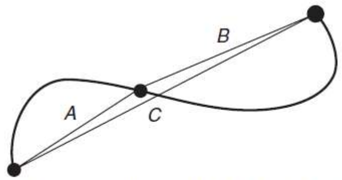
$$\big|\,||P(0.0) - P(1.0)|| - \big(||P(0.0) - P(0.5)|| + ||P(0.5) - P(1.0)||\big)\big| < \varepsilon$$



Lengths $A + B - C$ above error tolerance



Lengths $A + B - C$ within error tolerance



Lengths $A + B - C$ erroneously report that the error is within tolerance

Interpolating Values

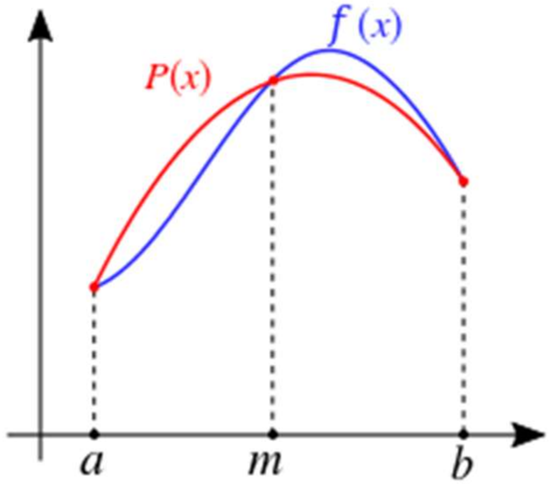
- Controlling the motion
  - Computing arc length (*Numerically*)
    - Numerical integration!! (remember integration for animation?)
    - On an unknown arbitrary curve
    - Simpson’s and trapezoidal integration, Gaussian quadrature

Interpolating Values

- Controlling the motion
  - Computing arc length (*Numerically*)
    - Simpson's integration
      - Point a, b and the middle point m (the simplest version)

$$P(x) = f(a)\frac{(x-m)(x-b)}{(a-m)(a-b)} + f(m)\frac{(x-a)(x-b)}{(m-a)(m-b)} + f(b)\frac{(x-a)(x-m)}{(b-a)(b-m)}$$

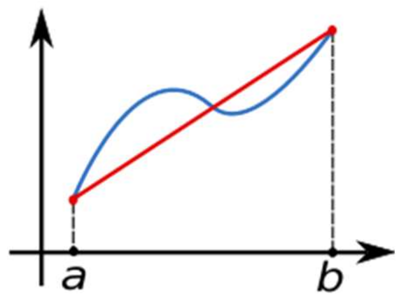
$$\int_a^b P(x)\,dx = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right]$$



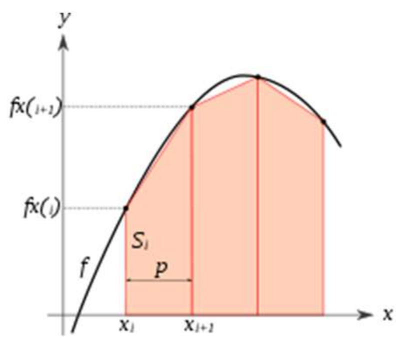
Interpolating Values

- Controlling the motion
  - Computing arc length (*Numerically*)
    - Trapezoidal integration

$$\int_a^b f(x)\,dx \approx (b-a)\left[\frac{f(a)+f(b)}{2}\right]$$



$$\int_a^b f(x)\,dx \approx \sum_{k=1}^{N} \frac{f(x_{k-1})+f(x_k)}{2}\Delta x_k$$



Interpolating Values

- Controlling the motion
  - Computing arc length (*Numerically*)
    - Gaussian quadrature

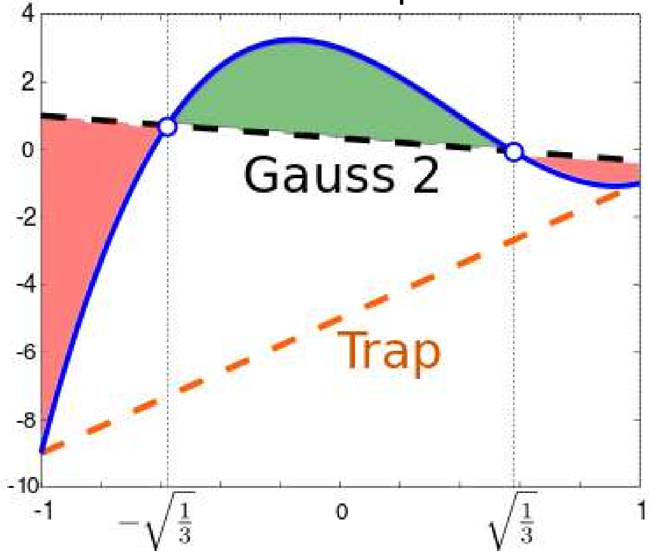
$$\int_{-1}^{1} f(x)\,dx = \int_{-1}^{1} \omega(x) g(x)\,dx \approx \sum_{i=1}^{n} w_i' g(x_i').$$

where $g(x)$ is approximately polynomial

$\omega(x) = 1/\sqrt{1-x^2}$ (Chebyshev–Gauss) $\omega(x) = e^{-x^2}$ (Gauss–Hermite)

Interval	$\omega(x)$	Orthogonal polynomials
[−1, 1]	1	Legendre polynomials
(−1, 1)	$(1-x)^\alpha (1+x)^\beta, \quad \alpha, \beta > -1$	Jacobi polynomials
(−1, 1)	$\frac{1}{\sqrt{1-x^2}}$	Chebyshev polynomials (first kind)
[−1, 1]	$\sqrt{1-x^2}$	Chebyshev polynomials (second kind)
[0, ∞)	e^{-x}	Laguerre polynomials
[0, ∞)	$x^\alpha e^{-x}, \quad \alpha > -1$	Generalized Laguerre polynomials
(−∞, ∞)	e^{-x^2}	Hermite polynomials

Blue: ground truth
Orange: Trapezoidal
Black: Gaussian quadrature



$y(x) = 7x^3 - 8x^2 - 3x + 3,$

Interpolating Values

- Controlling the motion
  - Computing arc length (*Numerically*)
    - Adaptive Gaussian Integration
      - Test errors, divide it into halves

Interpolating Values

- Controlling the motion
    - Computing arc length (*Numerically*)
        - Finding a point, given a distance along a curve

$$s - LENGTH(u_1, U(p_{n-1}))$$

Solve for $s - LENGTH(u_a, u) = 0$

- Newton-Rapson

$$p_n = p_{n-1} - f(p_{n-1})/f'(p_{n-1})$$

- Problems: p_n is not on the curve or $d\mathbf{P}/du$ = 0 (binary division can help)

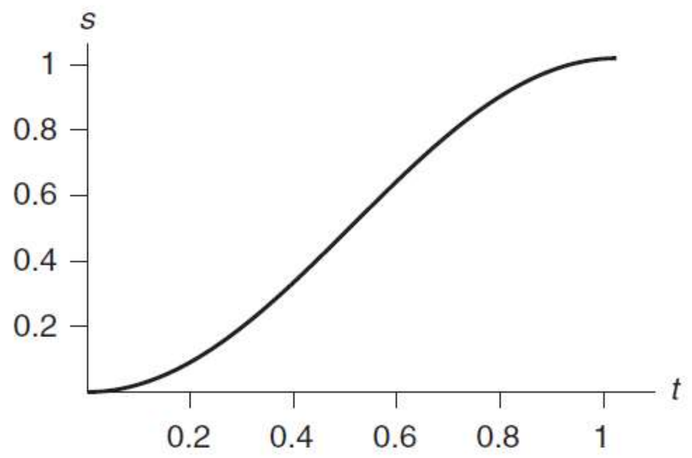
When finding u such that $LENGTH(0, u) = s$ find the values s_i , s_{i+1} such that $s_i < s < s_{i+1}$

- No need for Gaussian quadrature anymore

Interpolating Values

- Controlling the motion
  - Speed control
    - A function of arc length and time
    - Time is usually normalised [0, 1]

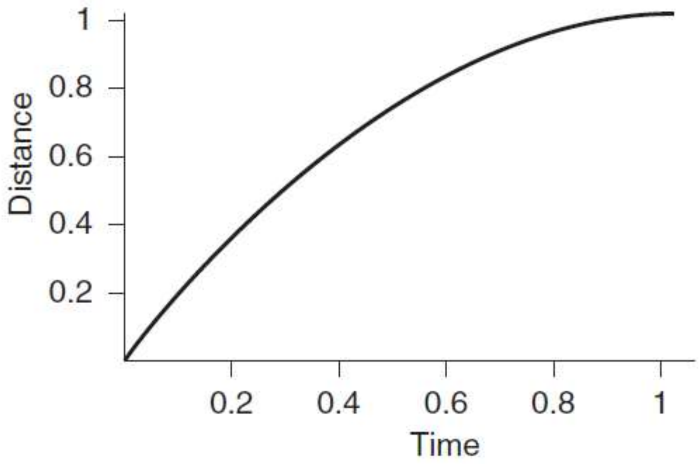
$S(t)$ is the desired distance



1. The distance-time function should be monotonic in t, that is, the curve should be traversed without backing up along the curve.
2. The distance-time function should be continuous, that is, there should be no instantaneous jumps from one point on the curve to a nonadjacent point on the curve.

Interpolating Values

- Controlling the motion
  - Speed control
    - Ease-in/Ease-out

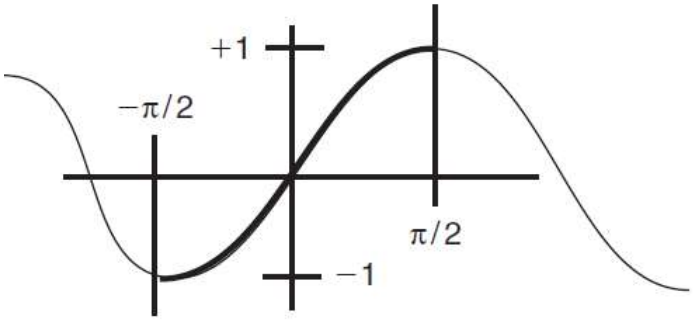


distance-time function $(2 - t)^* t$.

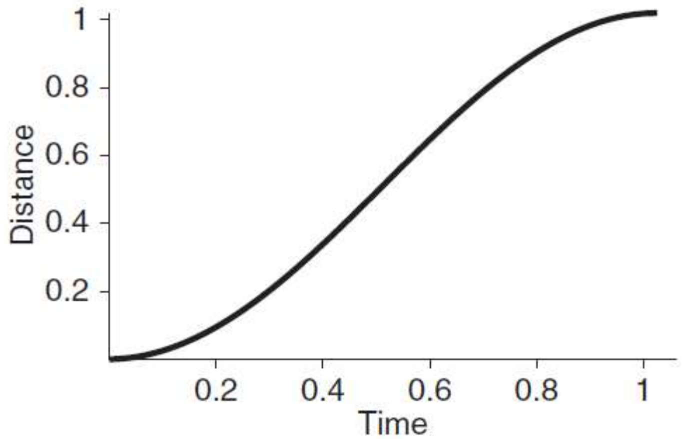
Interpolating Values

- Controlling the motion
  - Speed control
    - Sinusoidals for acceleration and deceleration

$$s = ease(t) = \frac{\sin\left(t_{\pi} - \frac{\pi}{2}\right) + 1}{2}$$



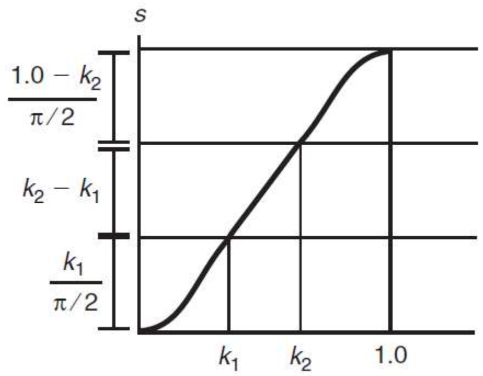
Sine curve segment to use as ease-in/ease-out control



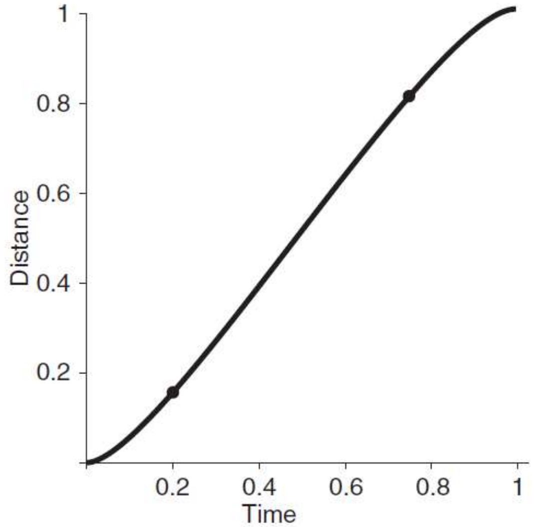
Sine curve segment mapped to useful values

Interpolating Values

- Controlling the motion
  - Speed control
    - Sinusoidals for acceleration and deceleration



Ease-in/ease-out curve as it is initially pieced together



Curve segments scaled into useful values with points marking segment junctions

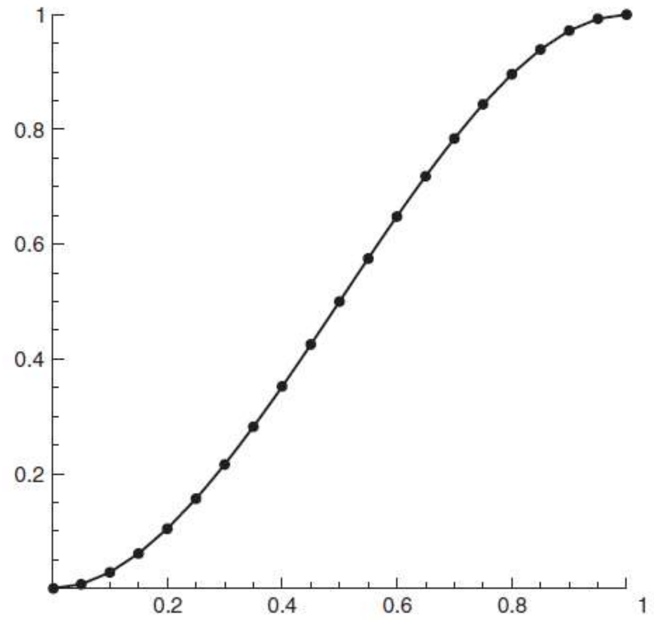
$$
ease(t)\begin{cases} = \left(k_1 \dfrac{2}{\pi}\left(\sin\left(\dfrac{t}{k_1}\dfrac{\pi}{2} - \dfrac{\pi}{2}\right) + 1\right)\right)/f & t \le k_1 \\ = \left(\dfrac{k_1}{\pi/2} + t - k_1\right)/f & k_1 \le t \le k_2 \\ = \left(\dfrac{k_1}{\pi/2} + k_2 - k_1 + \left((1-k_2)\dfrac{2}{\pi}\right)\sin\left(\left(\dfrac{t-k_2}{1-k_2}\right)\dfrac{\pi}{2}\right)\right)/f & k_2 \le t \end{cases}
$$

where $f = k_1(2/\pi + k_2 - k_1 + (1-k_2)(2/\pi))$

Interpolating Values

- Controlling the motion
  - Speed control
    - Single cubic polynomial

$$s = -2t^3 + 3t^2$$

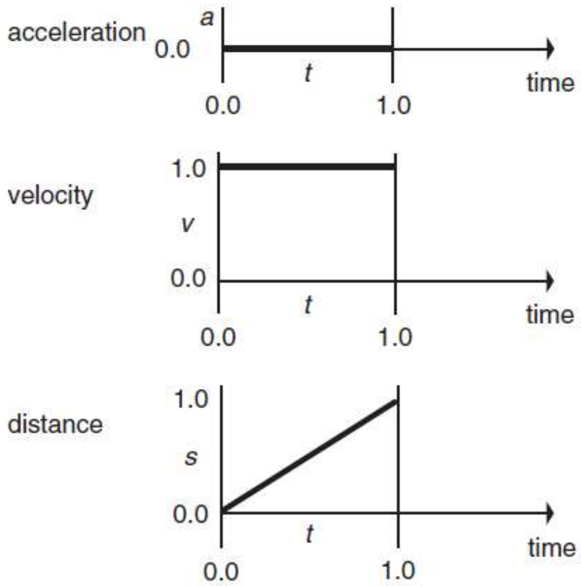


Interpolating Values

- Controlling the motion
  - Speed control
    - Constant acceleration: parabolic ease-in/ease-out

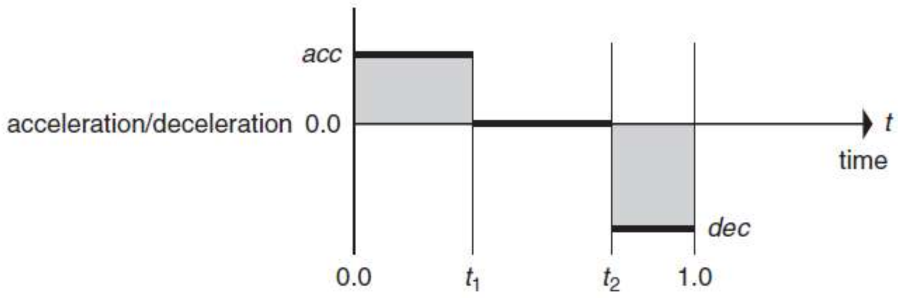
$$\text{distance} = \mathit{speed} \cdot \mathit{time}$$

$$v_0 = \frac{\mathrm{d}_{\mathrm{total}}}{\mathrm{t}_{\mathrm{total}}}$$



Interpolating Values

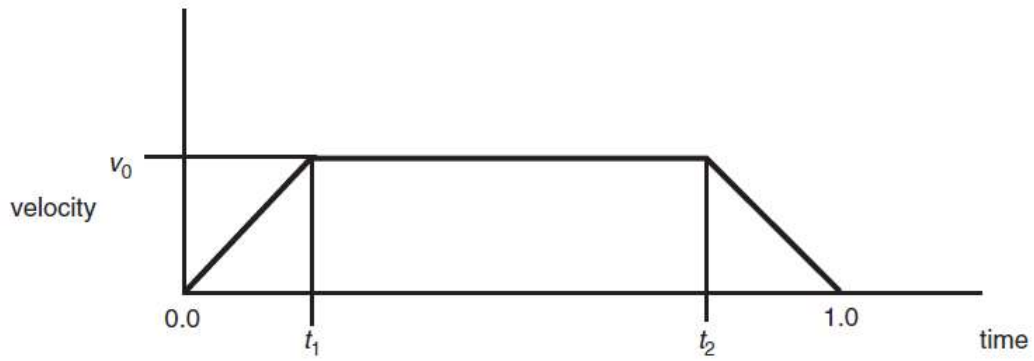
- Controlling the motion
  - Speed control
    - Constant acceleration: parabolic ease-in/ease-out



$a = acc \quad 0 < t < t_1$

$a = 0.0 \quad t_1 < t < t_2$

$a = dec \quad t_2 < t < 1.0$



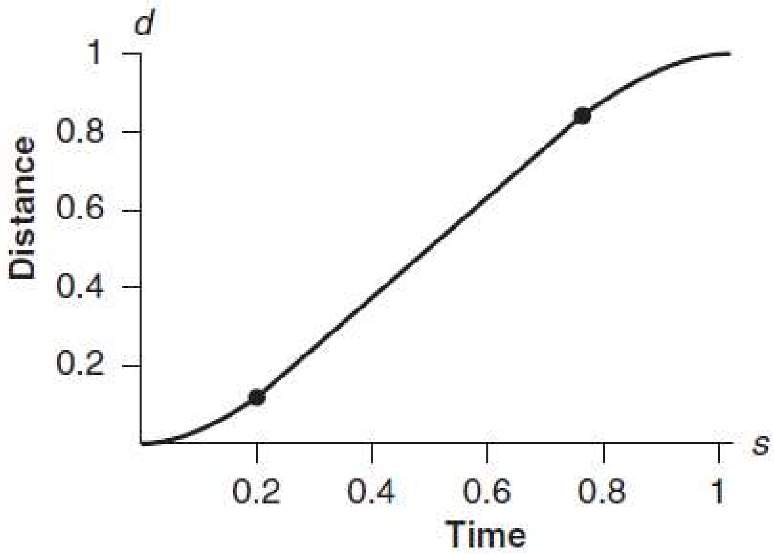
$$v = v_0 \cdot \frac{t}{t_1} \qquad 0.0 < t < t_1$$

$$v = v_0 \qquad t_1 < t < t_2$$

$$v = v_0 \cdot \left(1.0 - \frac{t - t_2}{1 - t_2}\right) \qquad t_2 < t < 1.0$$

Interpolating Values

- Controlling the motion
  - General distance-time functions



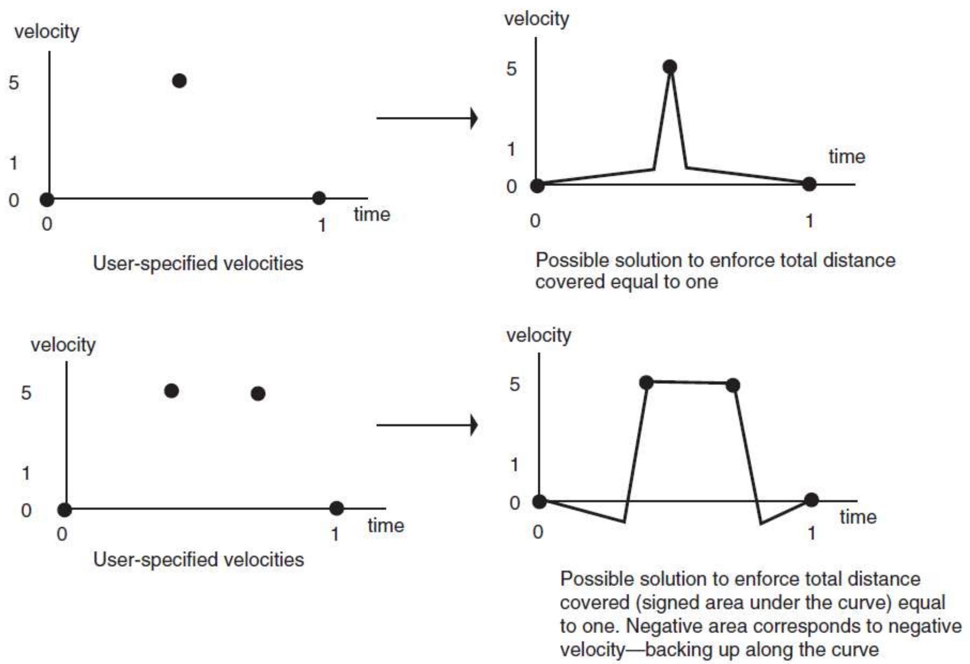
$$d = v_0 \frac{t^2}{2t_1} \qquad 0.0 < t < t_1$$

$$d = v_0 \frac{t_1}{2} + v_0 (t - t_1) \qquad t_1 < t < t_2$$

$$d = v_0 \frac{t_1}{2} + v_0 (t_2 - t_1) + \left[v_0 - \frac{\left(v_0 \frac{t - t_2}{1 - t_2} \right)}{2} \right] (t - t_2) \qquad t_2 < t < 1.0$$

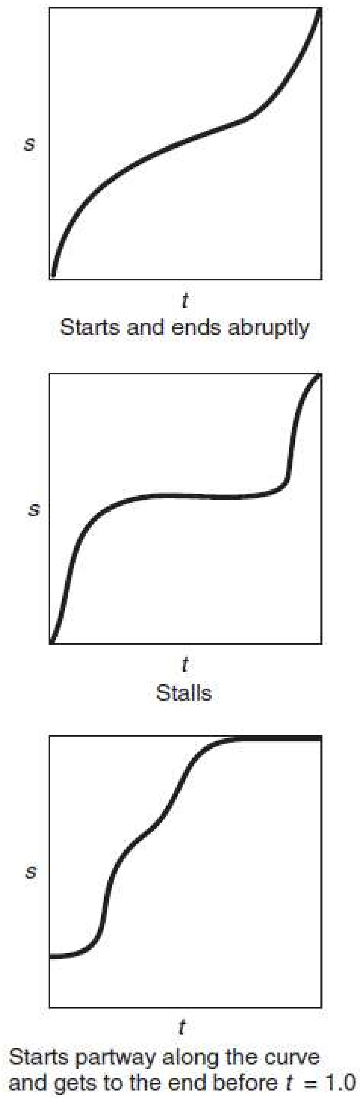
Interpolating Values

- Controlling the motion
  - General distance-time functions (arbitrary velocity)->interpolation of vel



Interpolating Values

- Controlling the motion
  - General distance-time functions
    (arbitrary distance-time)



Interpolating Values

- Controlling the motion
  - General distance-time functions

    (both distance and speed)