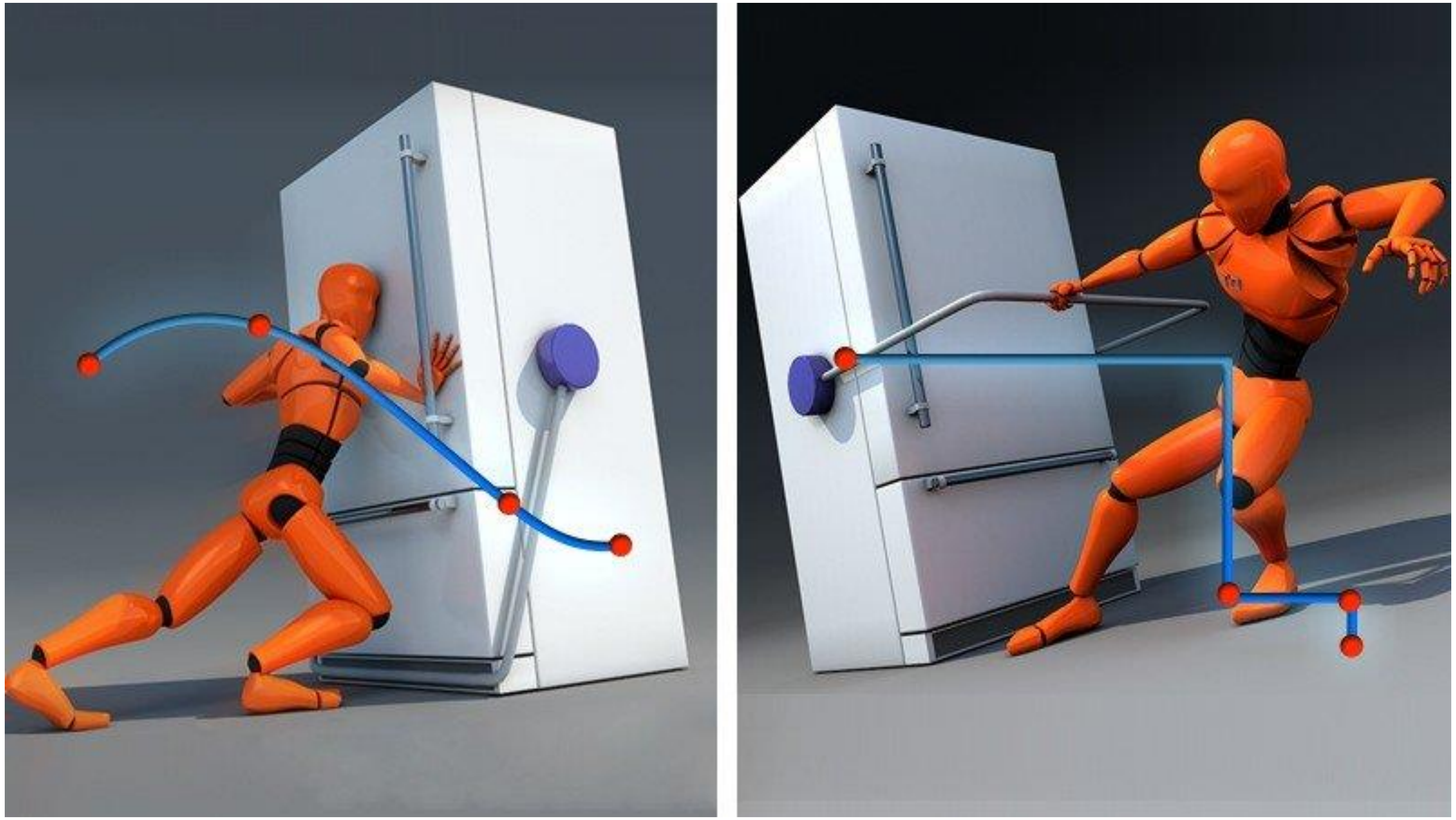
Helge Rhodin

CPSC 427
Video Game Programming

Curves and Animation



https://www.pluralsight.com/blog/film-games/stepped-vs-spline-curves-blocking-animation

Overview

1. ***Recap Physical simulation***

2. ***Impulse response***

3. ***Animation basics***

4. ***Curves***

Multiple forces?

Forces add up (and cancel):

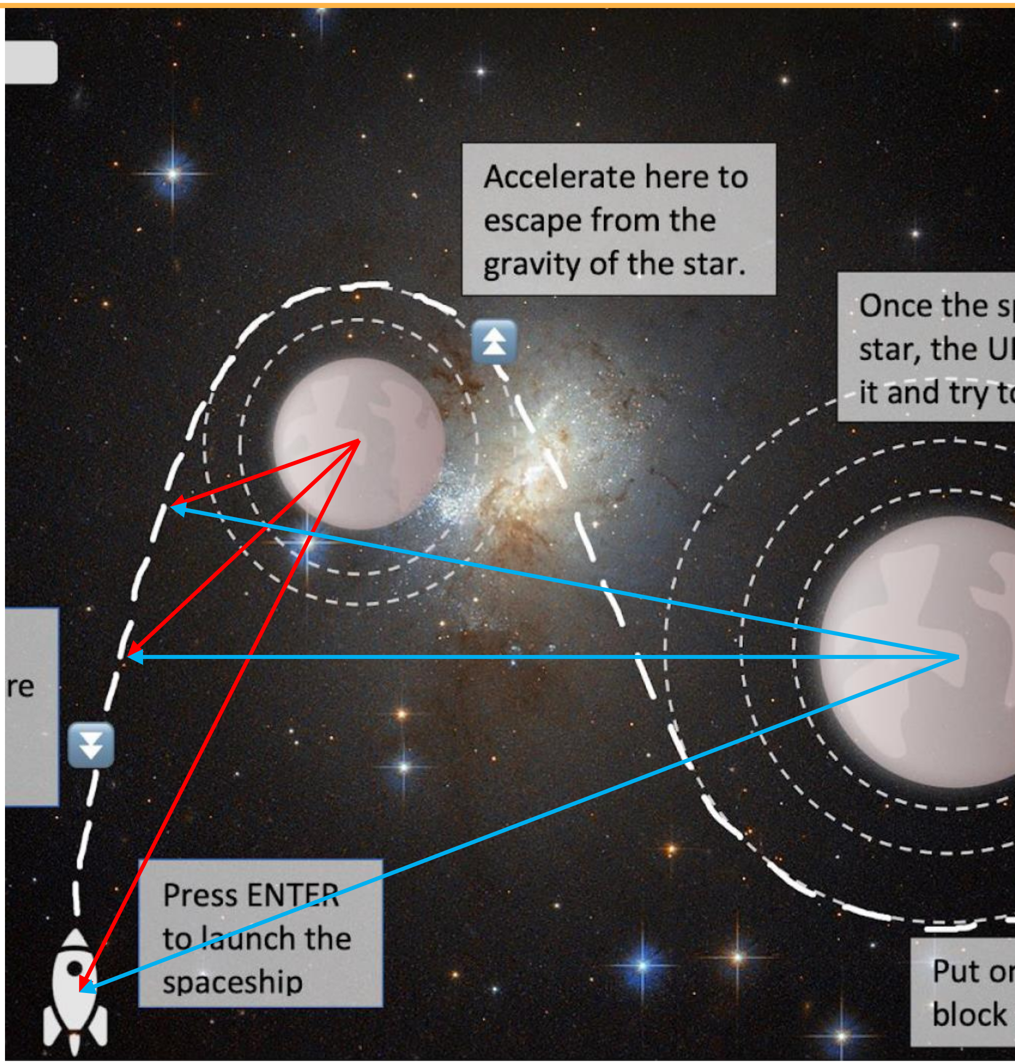
$$F = -mg_1 \begin{bmatrix} a_1 \\ b_1 \end{bmatrix} - mg_2 \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}$$

- ***This holds for all types of forces!***

- ***Notation you might see:***

$$F = \sum_i F_i = \sum F_i = \sum F$$

$$\vec{F} = F$$



Ordinary Differential Equations

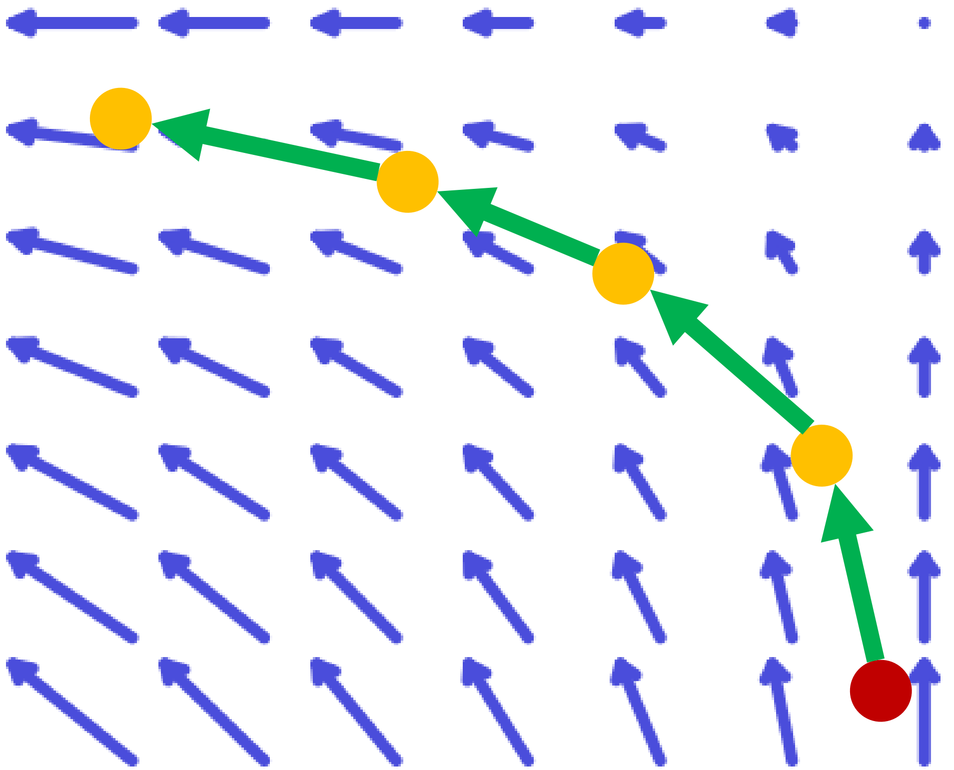
$$\frac{\partial}{\partial t}\vec{X}(t) = f(\vec{X}(t), t)$$

Given that $\vec{X}_0 = \vec{X}(t_0)$

Compute $\vec{X}(t)$ **for** $\mathrm{t} > t_0$

$$\Delta\vec{X}(t) = f(\vec{X}(t), t)\Delta t$$

- **Simulation:**
  - ***path through state-space***
  - ***driven by vector field***



ODE Numerical Integration: Explicit (Forward) Euler

$$\frac{\partial}{\partial t}\vec{X}(t) = f(\vec{X}(t), t)$$

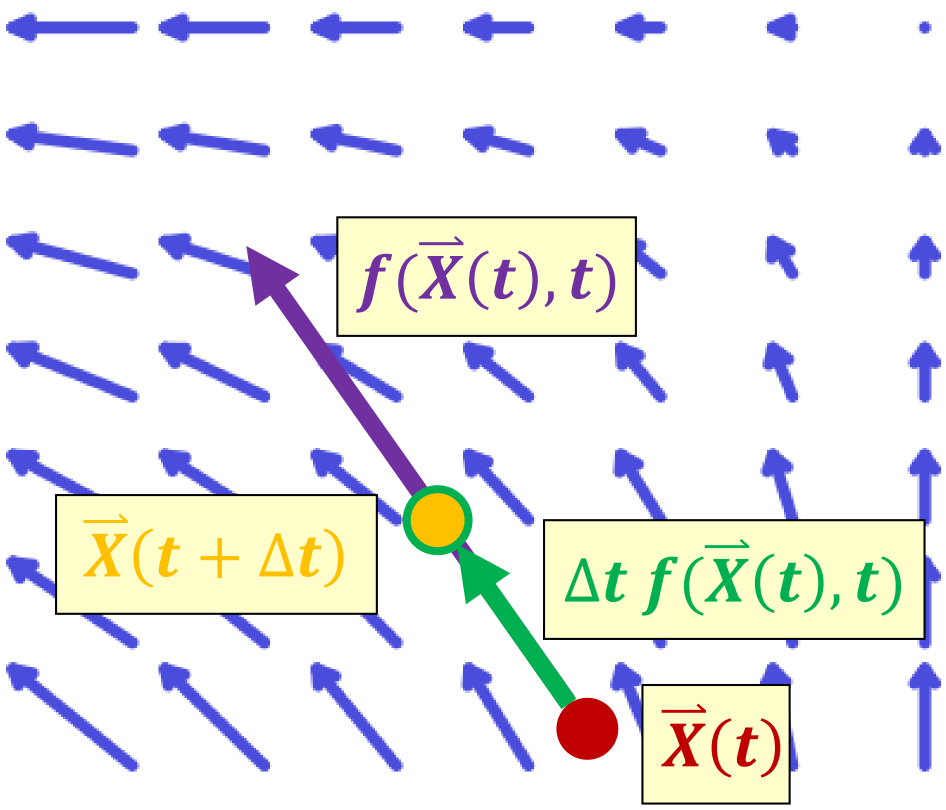
Given that $\vec{X}_0 = \vec{X}(t_0)$

Compute $\vec{X}(t)$ **for** $\mathrm{t} > \mathrm{t}_0$

$$\Delta t = \mathrm{t}_i - \mathrm{t}_{i-1}$$

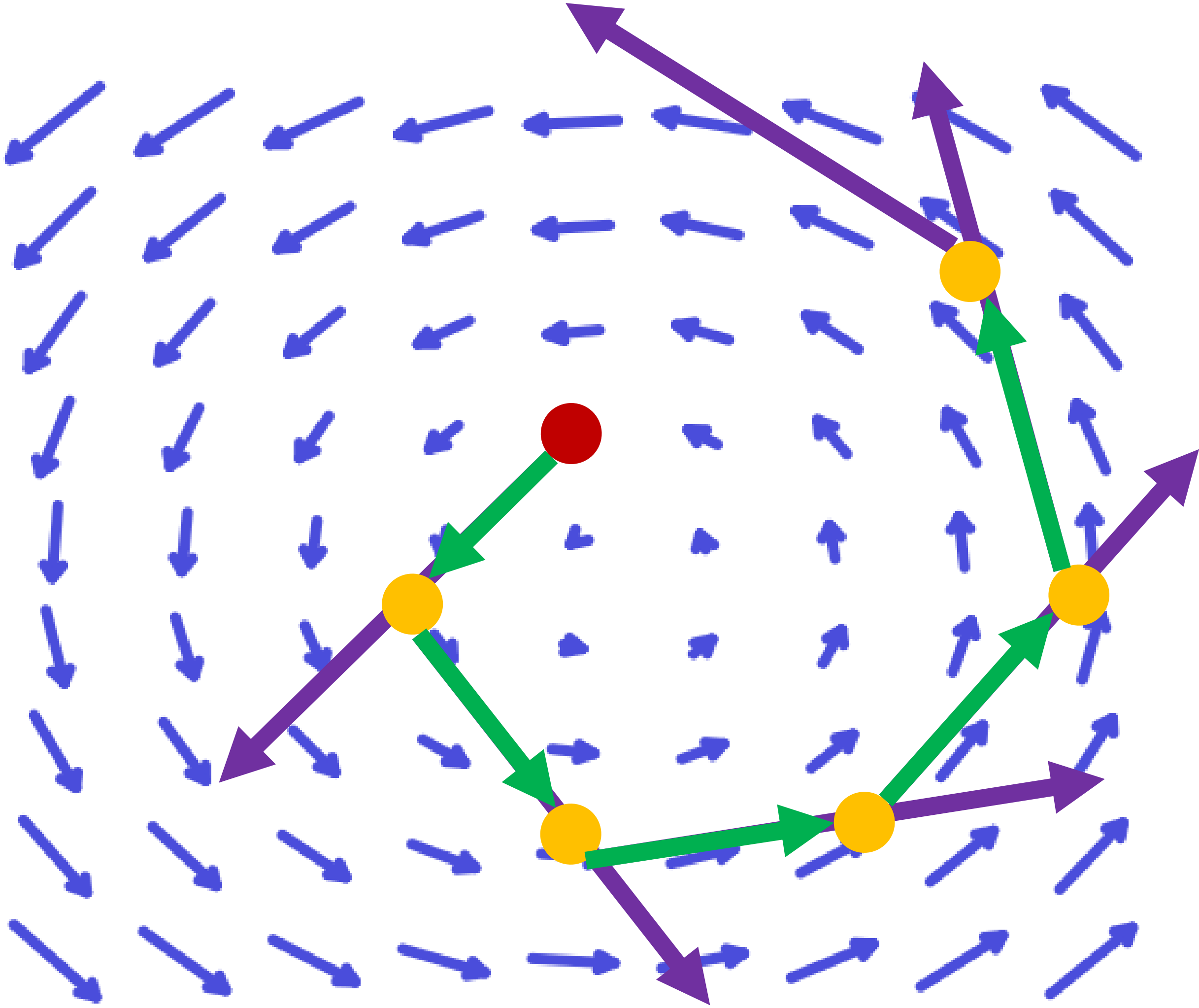
$$\Delta\vec{X}(\mathrm{t}_{i-1}) = \Delta t\, f(\vec{X}(\mathrm{t}_{i-1}), \mathrm{t}_{i-1})$$

$$\vec{X}_i = \vec{X}_{i-1} + \Delta t\, f(\vec{X}_{i-1}, \mathrm{t}_{i-1})$$



Explicit Euler Problems

- Solution **spirals** out
  - ***Even with small time steps***
  - ***Although smaller time steps are still better***

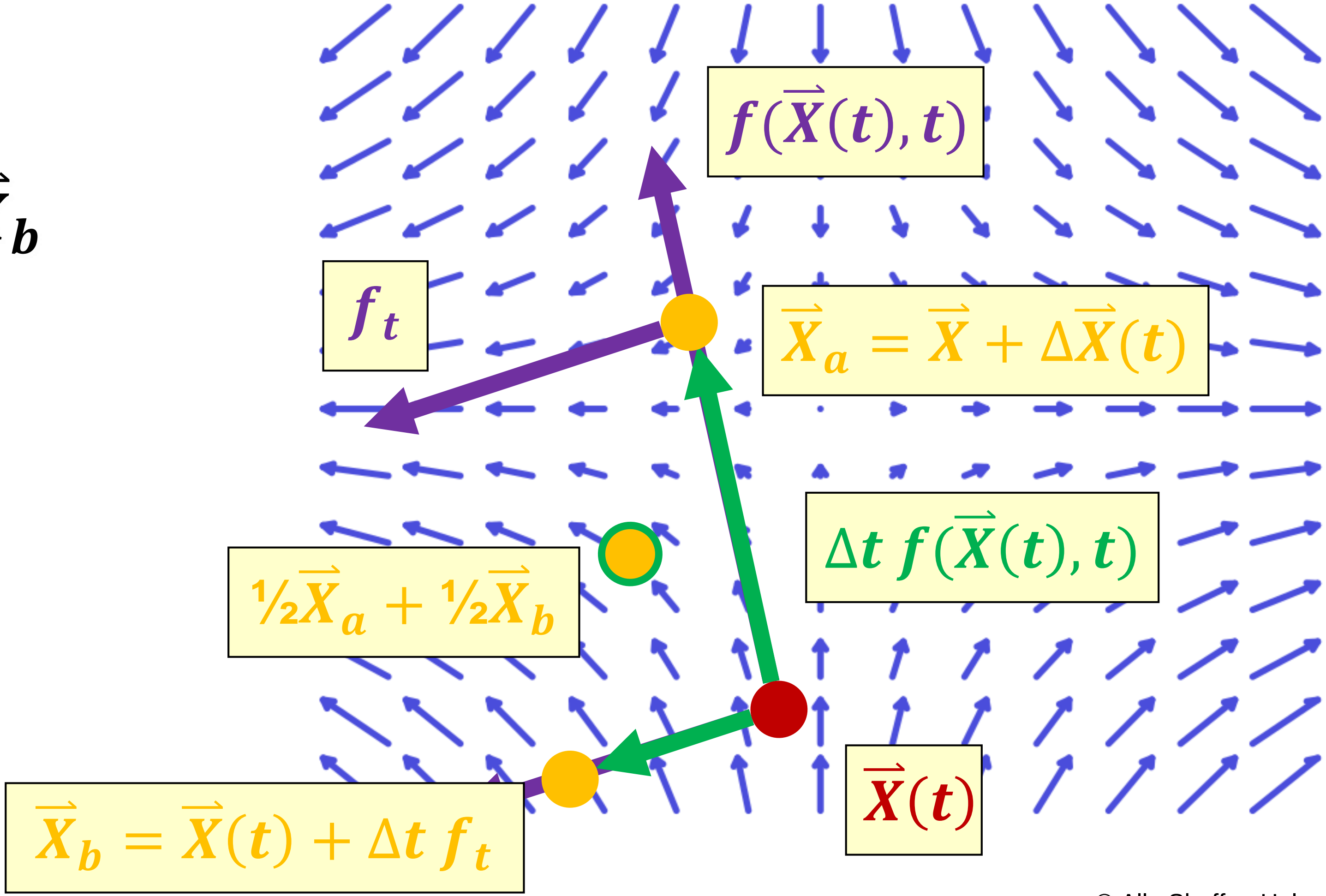


Definition: Explicit

- ***Closed-form/analytic solution***
- **no iterative solve required**

Trapezoid Method

1. full Euler step get $\vec{X}_a$
2. evaluate f_t at $\vec{X}_a$
3. full step using f_t get $\vec{X}_b$
4. average $\vec{X}_a$ and $\vec{X}_b$



Implicit (Backward) Euler:

- **Use forces at destination + derivative at the destination**

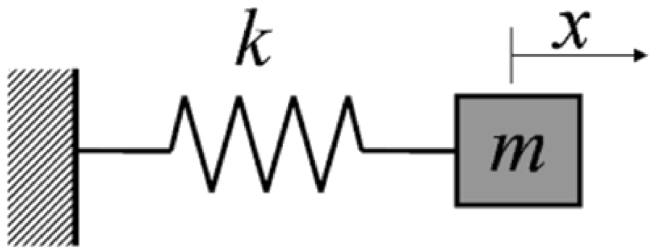
Solve system of equations

$$\frac{\partial}{\partial t}\begin{bmatrix}\vec{x}\\ \vec{v}\end{bmatrix} = \begin{bmatrix}\vec{v}\\ \Sigma\vec{F}/m\end{bmatrix}$$

$$\begin{aligned} x_{n+1} &= x_n + h\, v_{n+1} \\ v_{n+1} &= v_n + h\left(\frac{F_{n+1}}{m}\right) \end{aligned}$$

Example: Spring Force

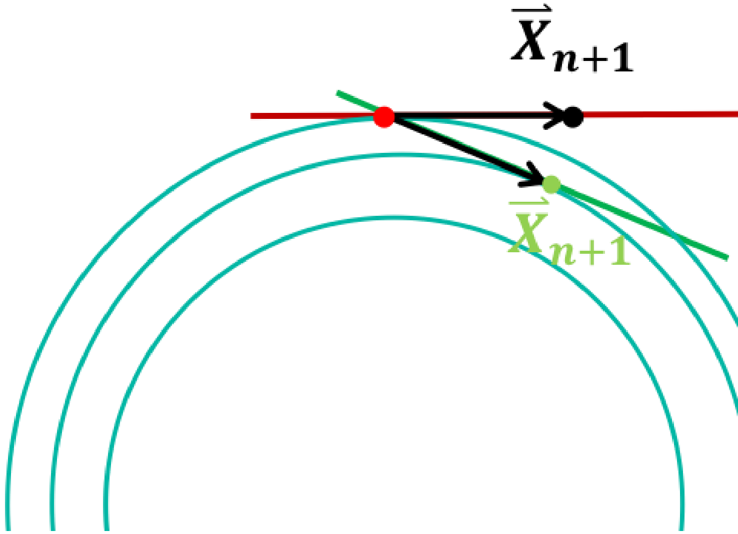
$$F = -kx$$



$$\begin{aligned} x_{n+1} &= x_n + h\, v_{n+1} \\ v_{n+1} &= v_n + h\left(\frac{-k\, x_{n+1}}{m}\right) \end{aligned}$$

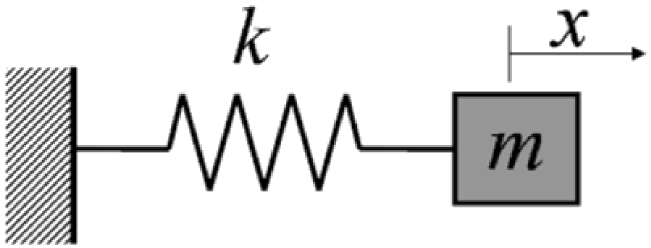
Analytic or iterative solve?

Forward vs Backward



$\vec{X}_{n+1} = \vec{X}_n + \Delta t\, f(\vec{X}_n)$

$\vec{X}_{n+1} = \vec{X}_n + \Delta t\, f(\vec{X}_{n+1})$



Forward Euler

$$x_{n+1} = x_n + h\, v_n$$
$$v_{n+1} = v_n + h\left(\frac{-k\, x_n}{m}\right)$$

Backward Euler

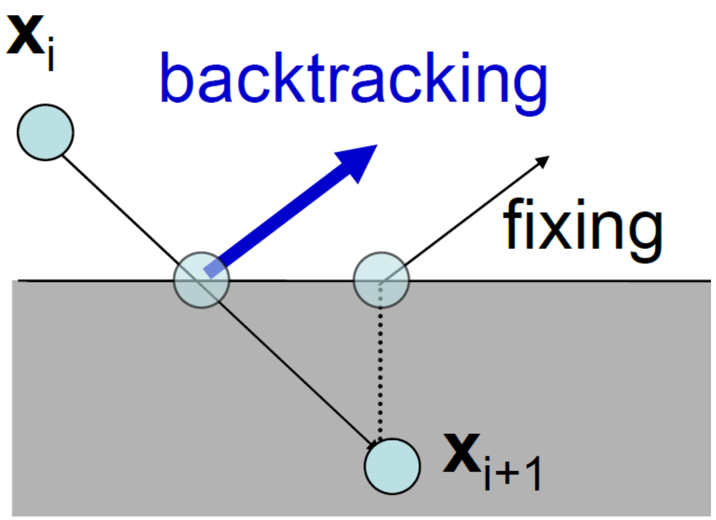
$$x_{n+1} = x_n + h\, v_{n+1}$$
$$v_{n+1} = v_n + h\left(\frac{-k\, x_{n+1}}{m}\right)$$

Could one apply the Trapezoid Method on backwards Euler?

New today: Impulse response

Collisions – Overshooting

- Usually, we detect collision when it is too late: we are already inside
- Solution: Back up
  - Compute intersection point
  - Ray-object intersection!
  - Compute response there
  - Advance for remaining fractional time step
- Other solution: Quick and dirty hack
  - Just project back to object closest point

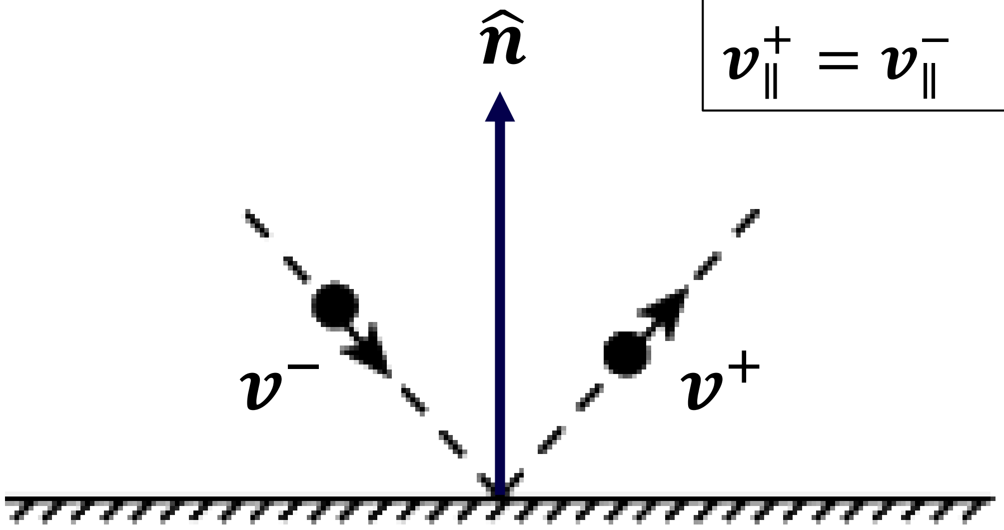


From MIT EECS 6.837 – Matusik

Particle-Plane Collisions

- ***Apply an 'impulse' $\vec{j}$ of magnitude j***
  - Inversely proportional to mass of particle
- ***In direction of normal***

Impulse in physics: Integral of F over time
In games: an instantaneous step change (not physically possible), i.e., the force applied over one time step of the simulation



$$v_\perp^+ = -\epsilon v_\perp^-$$
$$v_\parallel^+ = v_\parallel^-$$

$$j = (1+\epsilon)(v^- \cdot \hat{n})m$$

What is the effect of ϵ ?

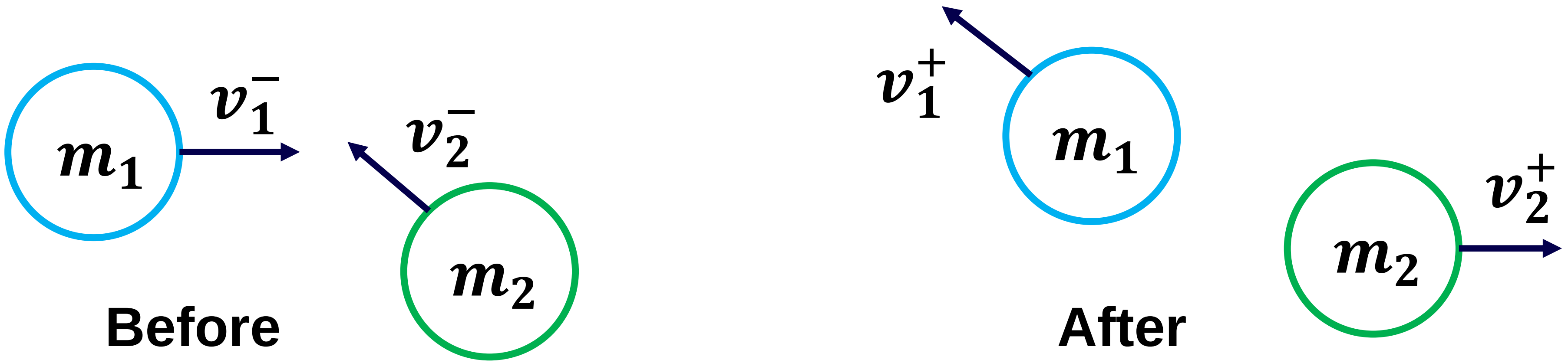
$$\vec{j} = j\,\hat{n}$$

$$v^+ = v^- + \frac{\vec{j}}{m}$$

Something missing?

Particle-Particle Collisions (radius=0)

- Particle-particle frictionless elastic impulse response



1. Momentum is preserved

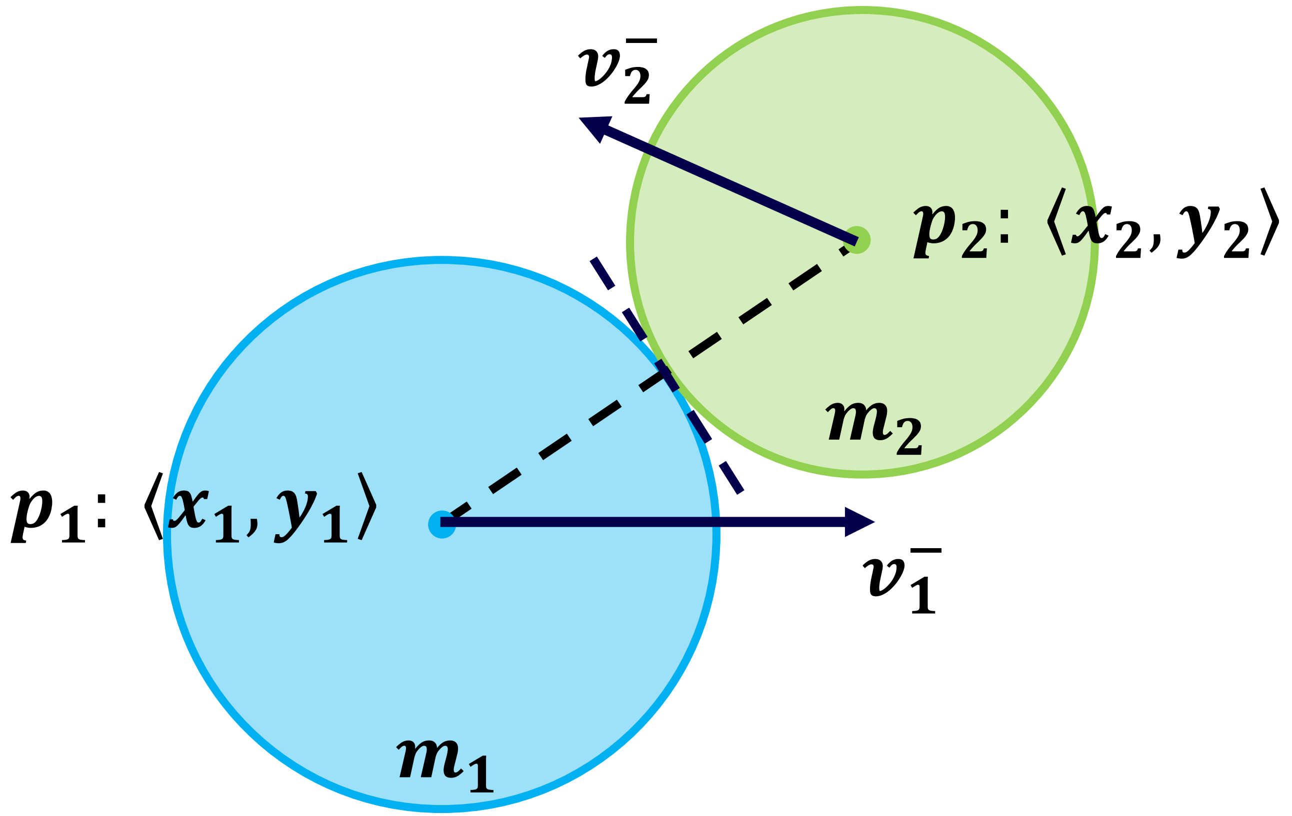
$$m_1 v_1^- + m_2 v_2^- = m_1 v_1^+ + m_2 v_2^+$$

2. Kinetic energy is preserved

$$½\, m_1 {v_1^-}^2 + ½\, m_2 {v_2^-}^2 = ½\, m_1 {v_1^+}^2 + ½\, m_2 {v_2^+}^2$$

Particle-Particle Collisions (radius >0)

- **What we know…**
  - ***Particle centers***
  - ***Initial velocities***
  - ***Particle Masses***
- **What we can calculate…**
  - ***Contact normal***
  - ***Contact tangent***

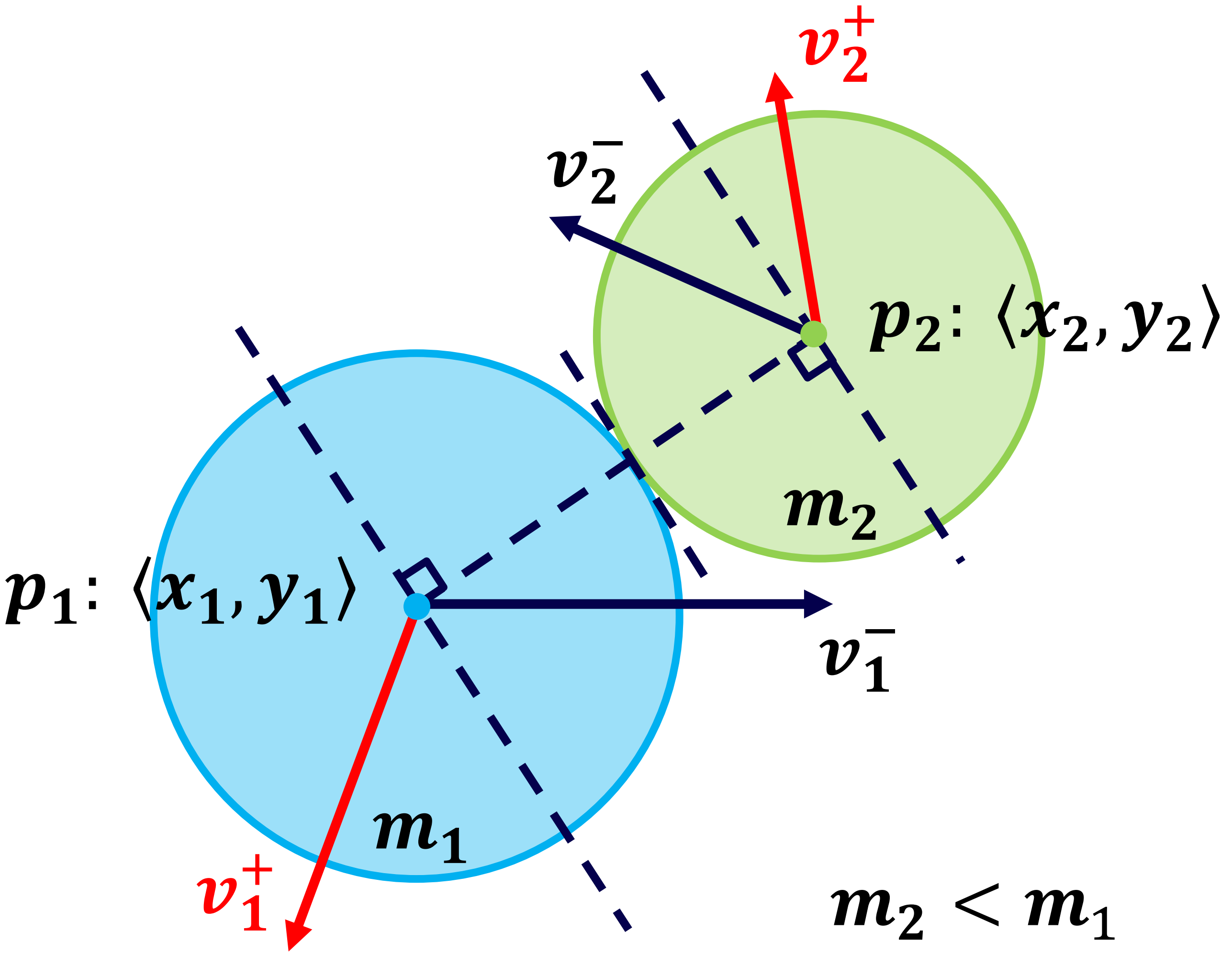


Observation: Velocity is preserved in tangential direction

Particle-Particle Collisions (radius >0)

Reduces to a 1D problem:

- Impulse **direction** along contact normal
- Impulse **magnitude** proportional to **mass of other particle**



Particle-Particle Collisions (radius >0)

https://en.wikipedia.org/wiki/Elastic_collision#Two-dimensional

- **More formally…**

$$\boldsymbol{v}_1^+ = \boldsymbol{v}_1^- - \frac{2\boldsymbol{m}_2}{\boldsymbol{m}_1 + \boldsymbol{m}_2} \frac{\langle \boldsymbol{v}_1^- - \boldsymbol{v}_2^- \rangle \cdot \langle \boldsymbol{p}_1 - \boldsymbol{p}_2 \rangle}{\|\boldsymbol{p}_1 - \boldsymbol{p}_2\|^2} \langle \boldsymbol{p}_1 - \boldsymbol{p}_2 \rangle$$

$$\boldsymbol{v}_2^+ = \boldsymbol{v}_2^- - \frac{2\boldsymbol{m}_1}{\boldsymbol{m}_1 + \boldsymbol{m}_2} \frac{\langle \boldsymbol{v}_2^- - \boldsymbol{v}_1^- \rangle \cdot \langle \boldsymbol{p}_2 - \boldsymbol{p}_1 \rangle}{\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|^2} \langle \boldsymbol{p}_2 - \boldsymbol{p}_1 \rangle$$

- This is in terms of velocity, what would the corresponding impulse be?

Self-study: change of velocity (1D)

https://en.wikipedia.org/wiki/Elastic_collision#Two-dimensional

To derive the above equations for v_1, v_2, rearrange the kinetic energy and momentum equations:

$$m_1(v_1^2 - u_1^2) = m_2(u_2^2 - v_2^2)$$

$$m_1(v_1 - u_1) = m_2(u_2 - v_2)$$

Dividing each side of the top equation by each side of the bottom equation, and using $\frac{a^2-b^2}{(a-b)} = a + b$, gives:

$$v_1 + u_1 = u_2 + v_2 \quad \Rightarrow \quad v_1 - v_2 = u_2 - u_1 .$$

That is, the relative velocity of one particle with respect to the other is reversed by the collision.

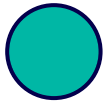
Now the above formulas follow from solving a system of linear equations for v_1, v_2, regarding m_1, m_2, u_1, u_2 as constants:

$$\begin{cases} v_1 - v_2 = u_2 - u_1 \\ m_1 v_1 + m_2 v_2 = m_1 u_1 + m_2 u_2 . \end{cases}$$

Once v_1 is determined, v_2 can be found by symmetry.

Rigid Body Dynamics (rotational motion of objects?)

- **From particles to rigid bodies…**



Particle

$$state = \begin{cases} \vec{x}\ position \\ \vec{v}\ velocity \end{cases}$$

$\mathbb{R}^4$ in 2D
$\mathbb{R}^6$ in 3D

Rigid body

$$state = \begin{cases} \vec{x}\ position \\ \vec{v}\ velocity \\ R\ rotation\ matrix\ 3x3 \\ \vec{w}\ angular\ velocity \end{cases}$$

$\mathbb{R}^{12}$ in 3D

Self-study: Rigid body collision

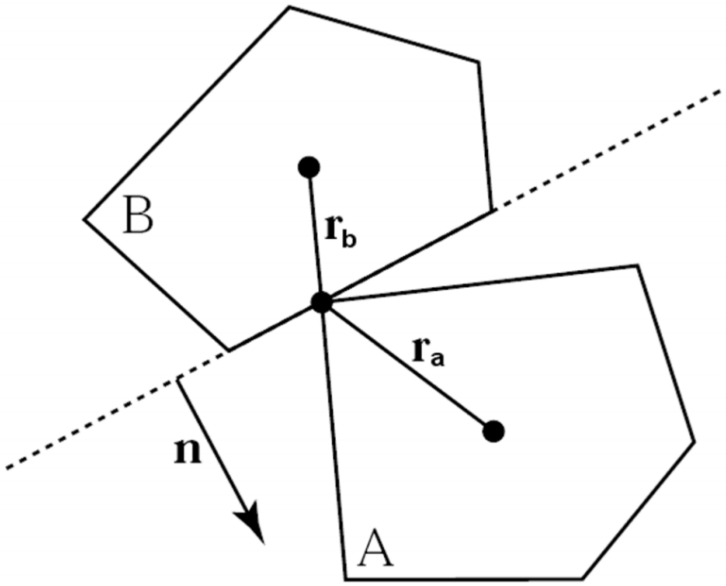
More: https://www.scss.tcd.ie/Michael.Manzke/CS7057/cs7057-1516-09-CollisionResponse-mm.pdf

$$v_{rel}^{+} = -\epsilon v_{rel}^{-}$$

$$v_{rel} = \hat{\boldsymbol{n}}(\dot{\boldsymbol{p}}_A - \dot{\boldsymbol{p}}_B)$$

$$\dot{\boldsymbol{p}}_A = \boldsymbol{v}_A + \boldsymbol{\omega}_A \times (\boldsymbol{p}_A - \boldsymbol{x}_A)$$
$$= \boldsymbol{v}_A + \boldsymbol{\omega}_A \times \boldsymbol{r}_A$$

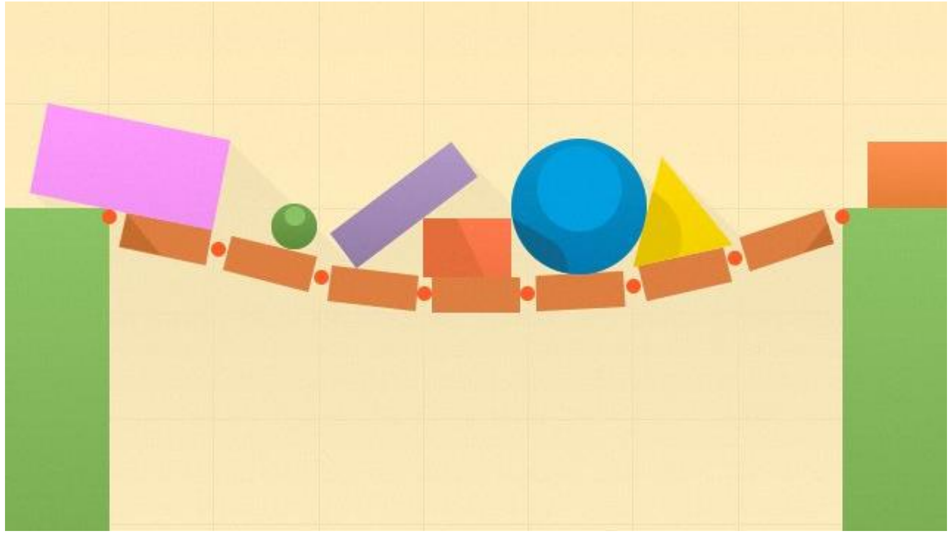
$$\dot{\boldsymbol{p}}_A = \boldsymbol{v}_B + \boldsymbol{\omega}_B \times (\boldsymbol{p}_B - \boldsymbol{x}_B)$$
$$= \boldsymbol{v}_A + \boldsymbol{\omega}_A \times \boldsymbol{r}_A$$



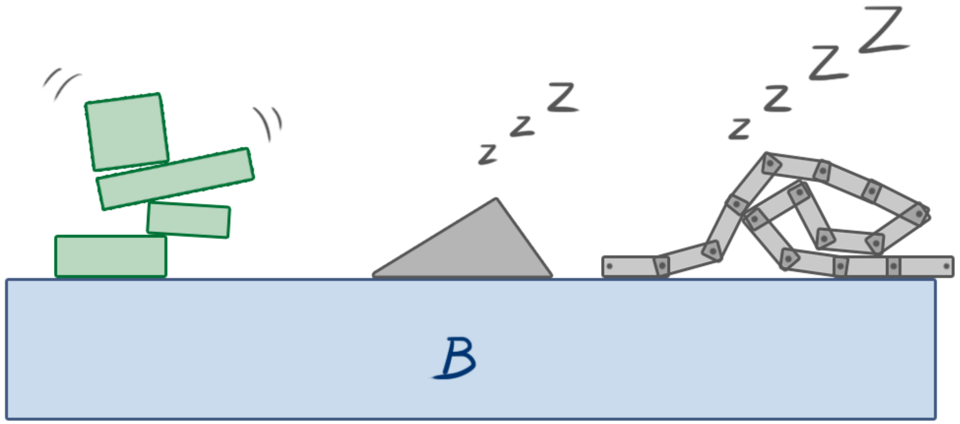
Linear velocity component

Angular component: Linear velocity of point p due to its rotation. See previous slide

Self-study: Constrained physics



By Nilson Souto
https://www.toptal.com/game/video-game-physics-part-iii-constrained-rigid-body-simulation



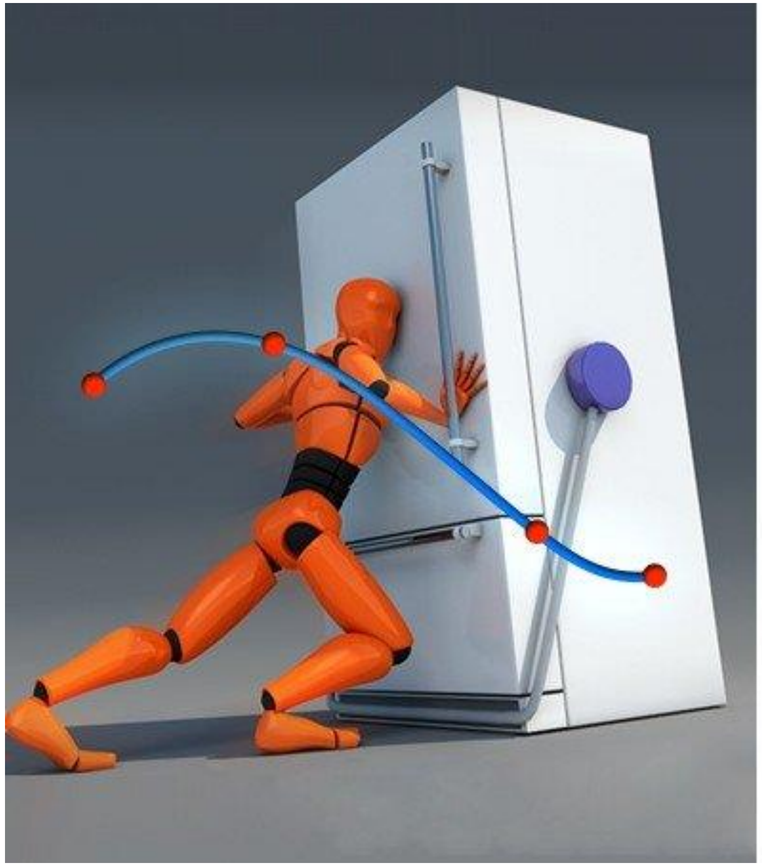
Logistics

- ***Team presentations on Tuesday (9th)***

- ***Guest lecture on Thursday (11th)***
  - *Craig Peters (EA)*
  - *Debugging and peer review*

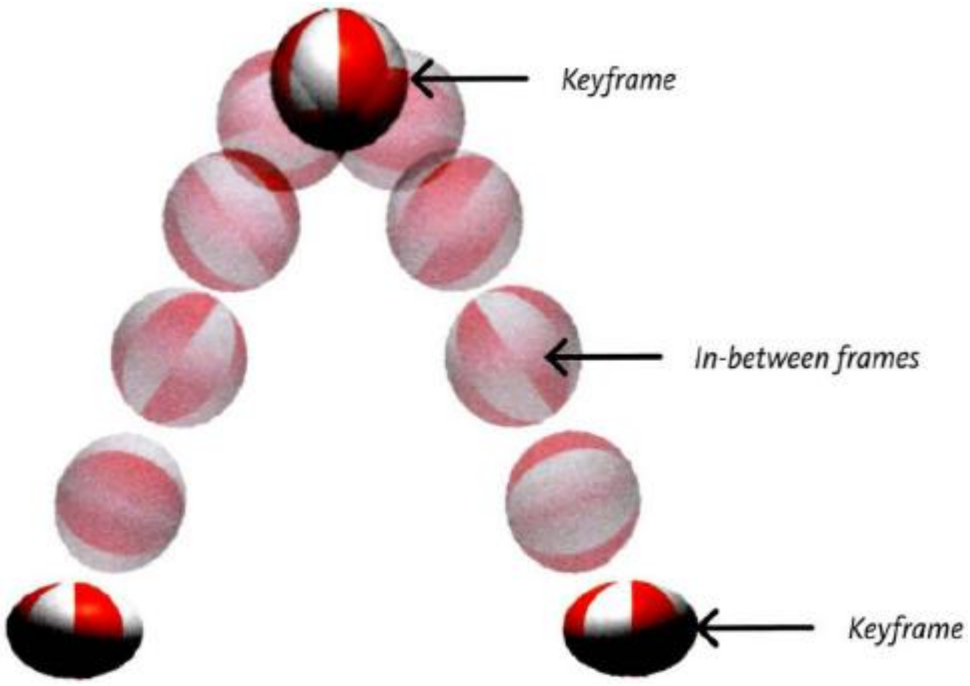
- Upcoming lectures
  - *Testing and User Studies*
  - *Composite transformations and inverse kinematics animation*

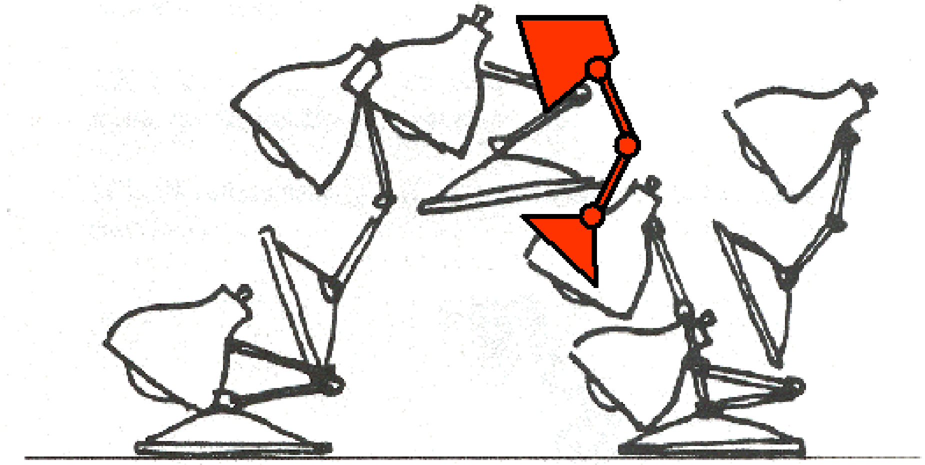
Curves and Animation




https://www.pluralsight.com/blog/film-games/stepped-vs-spline-curves-blocking-animation

Keyframe animation





Lasseter `87

Recap: Line equation

Parametric form

- 3D: x, y, and z are functions of a parameter value t

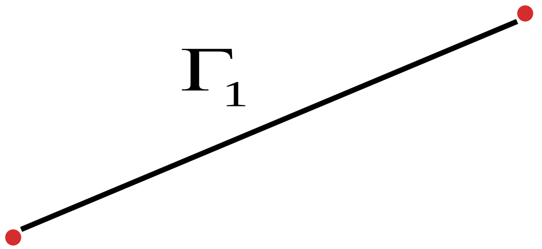
$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1 - t)$$

What things can we interpolate?

Line segment

$P_1 = \left(x_1^1, y_1^1\right)$

Γ_1



$P_0 = \left(x_0^1, y_0^1\right)$

$$G_1 = \begin{cases} x^1(t) = x_0^1 + \left(x_1^1 - x_0^1\right)t \\ y^1(t) = y_0^1 + \left(y_1^1 - y_0^1\right)t \end{cases} \quad t \in [0,1]$$

Interpolating general properties

- ***position***
- ***aspect ratio?***
- ***scale***
- ***color***
- ***What else?***

$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1-t)$$

$s^0 \qquad s^1$

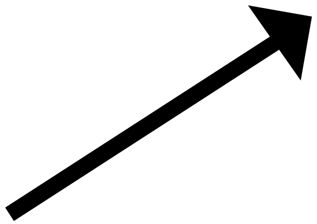
$c^0 \qquad c^1$

Barycentric coordinates / interpolation

Other Parametric Functions

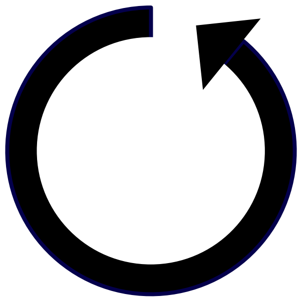
$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1-t)$$

Line segment



$$C(t) := \begin{pmatrix} \cos t \\ \sin t \end{pmatrix}$$

Circle (arc)



?

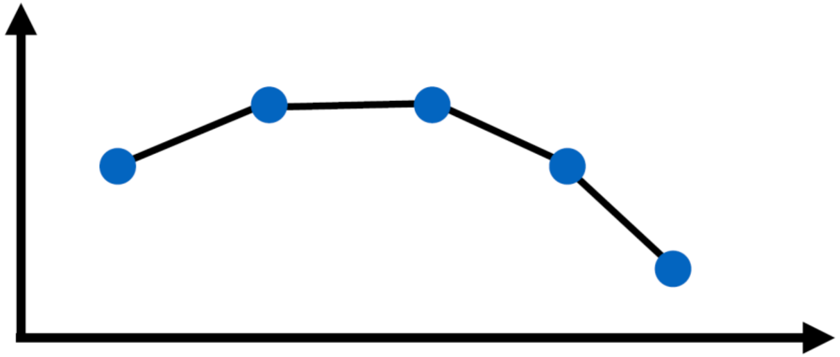
Splines

Splines

Segments of simple functions

$$f(x) = \begin{cases} f_1(x), & \text{if } x_1 < x \le x_2 \\ f_2(x), & \text{if } x_2 < x \le x_3 \\ \vdots & \vdots \\ f_n(x), & \text{if } x_n < x \le x_{n+1} \end{cases}$$

E.g., linear functions



Splines – Free Form Curves

Usually parametric

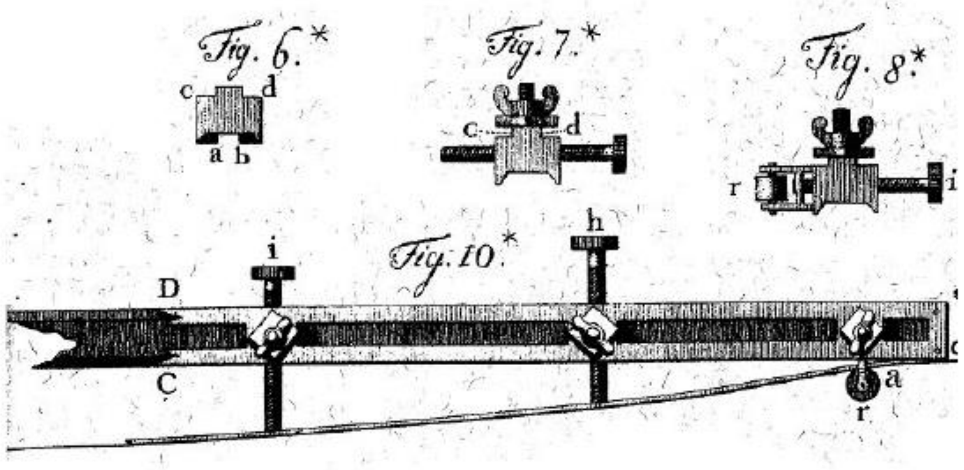
- $C(t)=[x(t),y(t)]$ or $C(t)=[x(t),y(t),z(t)]$

Description = basis functions + coefficients

$$C(t)=\sum_{i=0}^{n} P_i B_i(t)=(x(t),y(t))$$

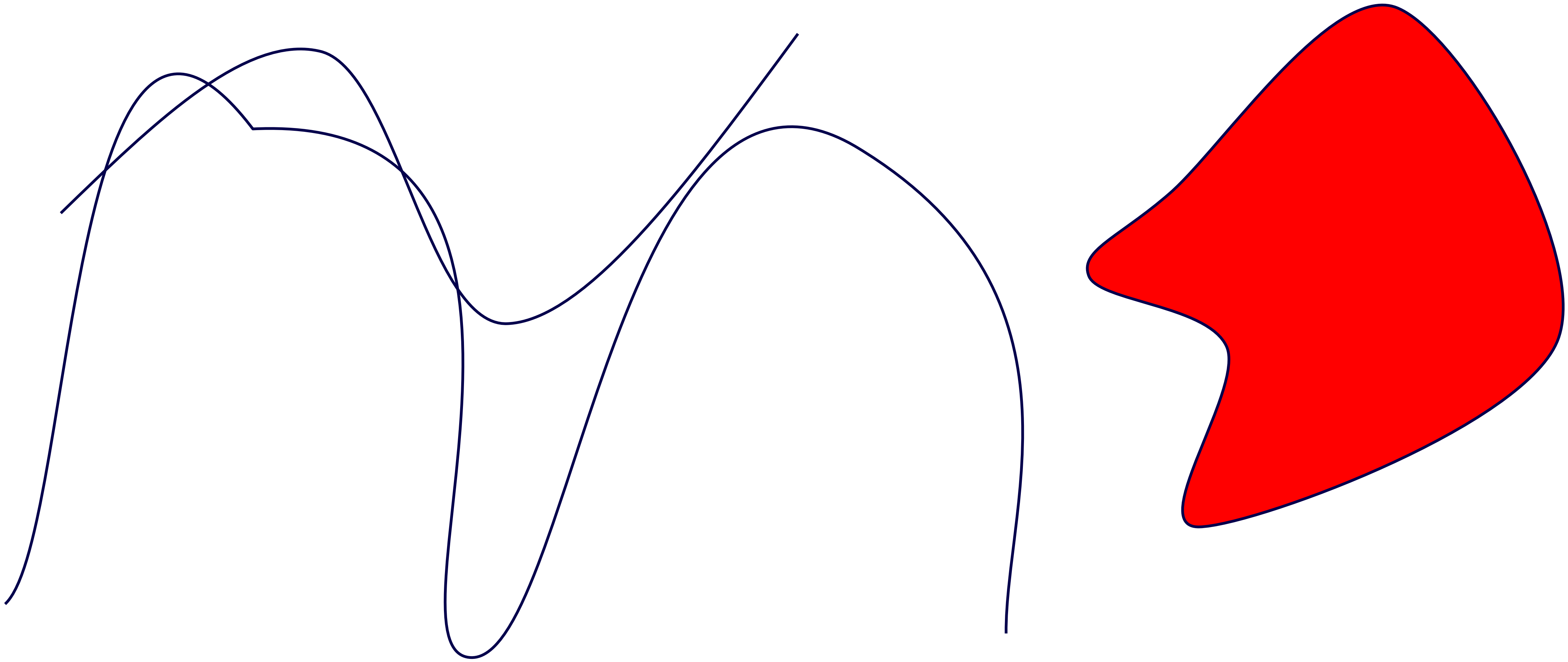
$$x(t)=\sum_{i=0}^{n} P_i^x B_i(t)$$

$$y(t)=\sum_{i=0}^{n} P_i^y B_i(t)$$

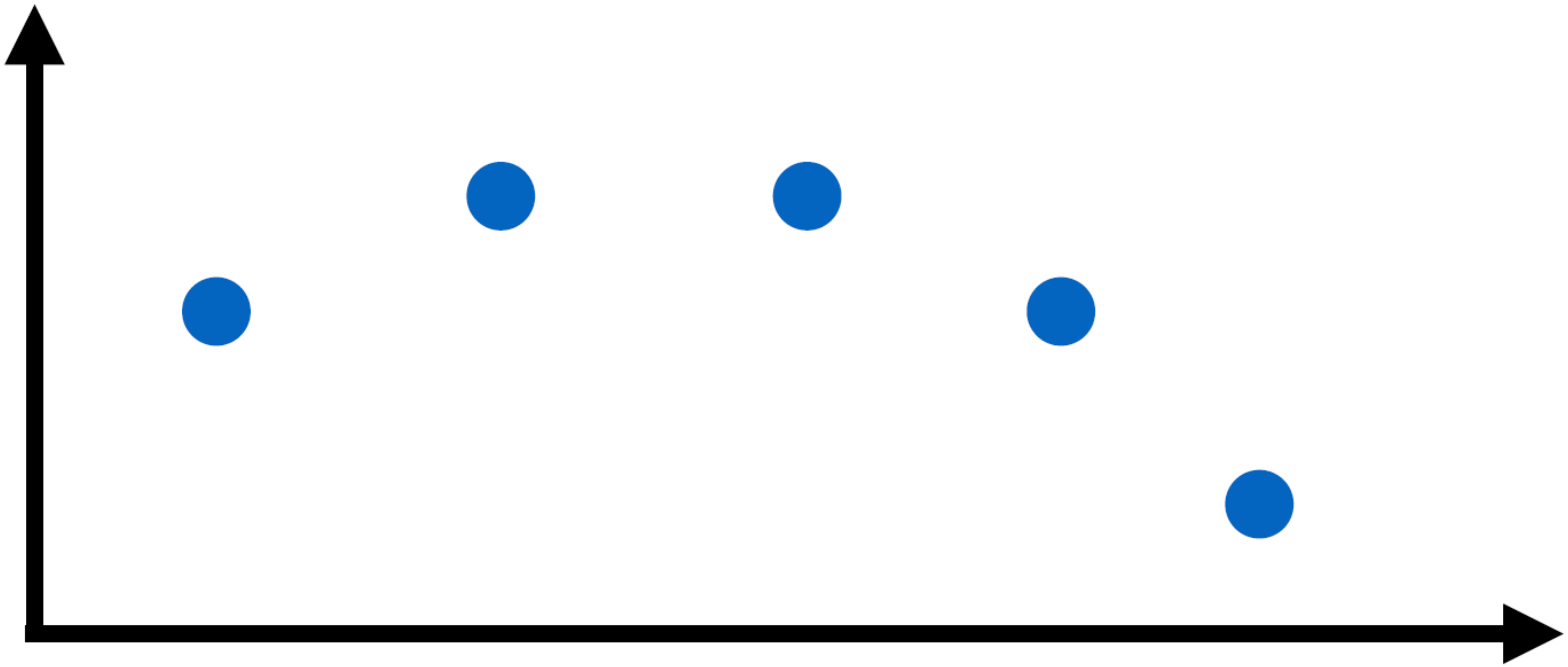


- Same basis functions for all coordinates

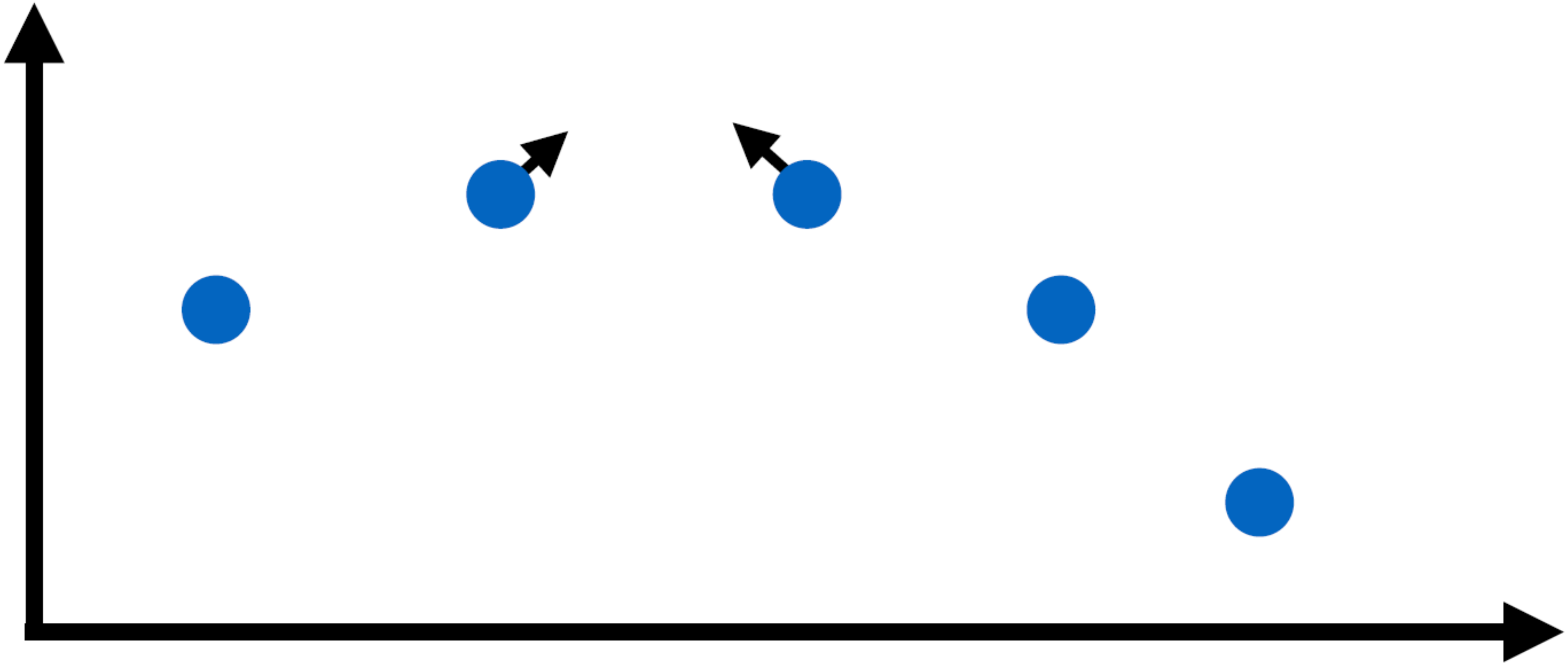
Curves



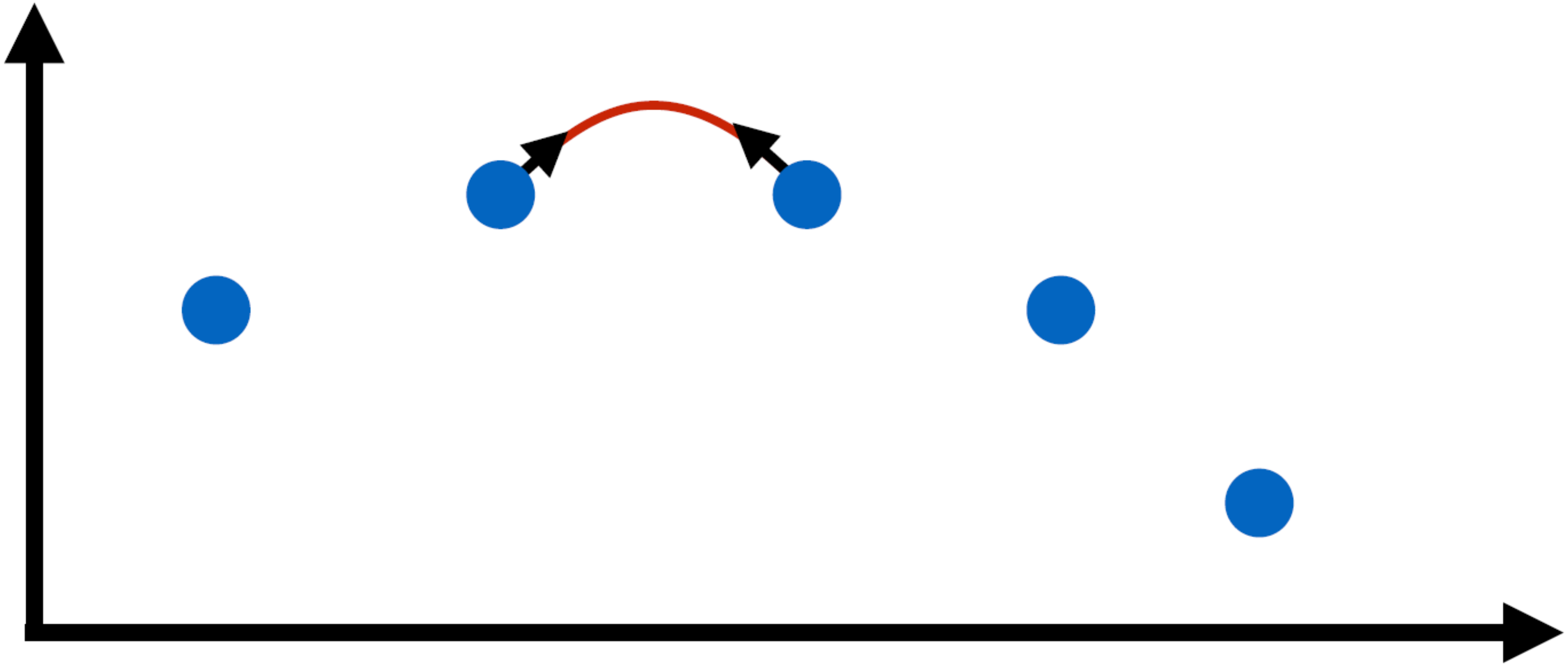
Smooth curve



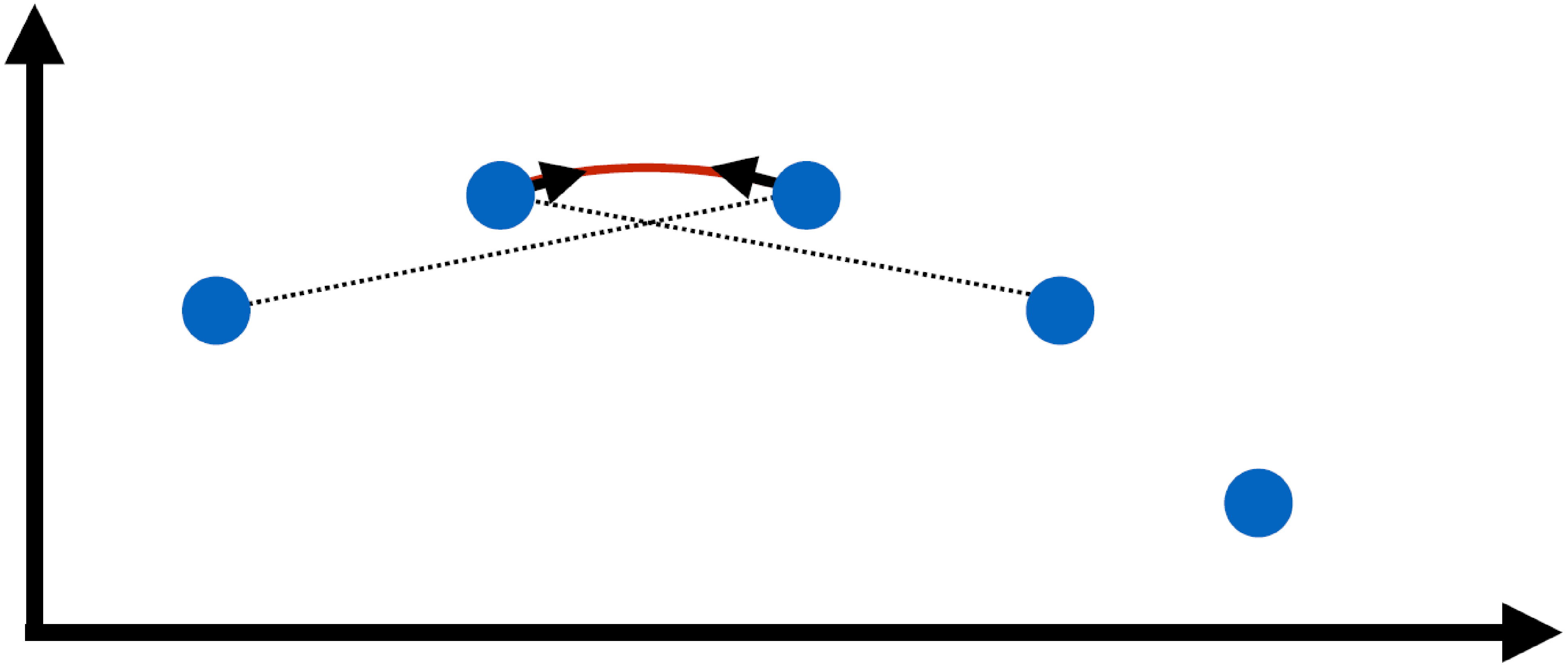
Smooth curve



Smooth curve



Smooth curve



Hermite Cubic Basis

Geometrically-oriented coefficients

- 2 positions + 2 tangents

Require $C(0)=P_0,\ C(1)=P_1,\ C'(0)=T_0,\ C'(1)=T_1$

Derivatives of C at 0 and 1

Define basis functions, one per requirement

$$C(t) = P_0 h_{00}(t) + P_1 h_{01}(t) + T_0 h_{10}(t) + T_1 h_{11}(t)$$

Hermite Basis Functions

$$C(t) = P_0 h_{00}(t) + P_1 h_{01}(t) + T_0 h_{10}(t) + T_1 h_{11}(t)$$

To enforce $C(0)=P_0,\ C(1)=P_1,\ C'(0)=T_0,\ C'(1)=T_1$ ***basis should satisfy***

$$h_{ij}(t): i, j = 0,1, t \in [0,1]$$

curve	$C(0)$	$C(1)$	$C'(0)$	$C'(1)$
$h_{00}(t)$	1	0	0	0
$h_{01}(t)$	0	1	0	0
$h_{10}(t)$	0	0	1	0
$h_{11}(t)$	0	0	0	1

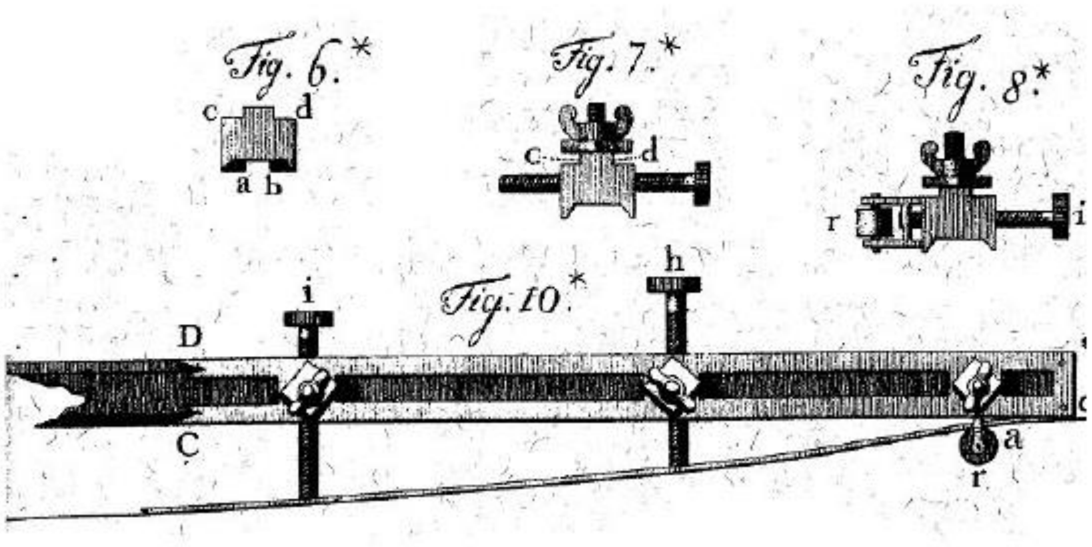
$h_{00}'(0) = h_{00}'(0) = 0$

$h_{00}(0) = 1$

Splines – Free Form Curves

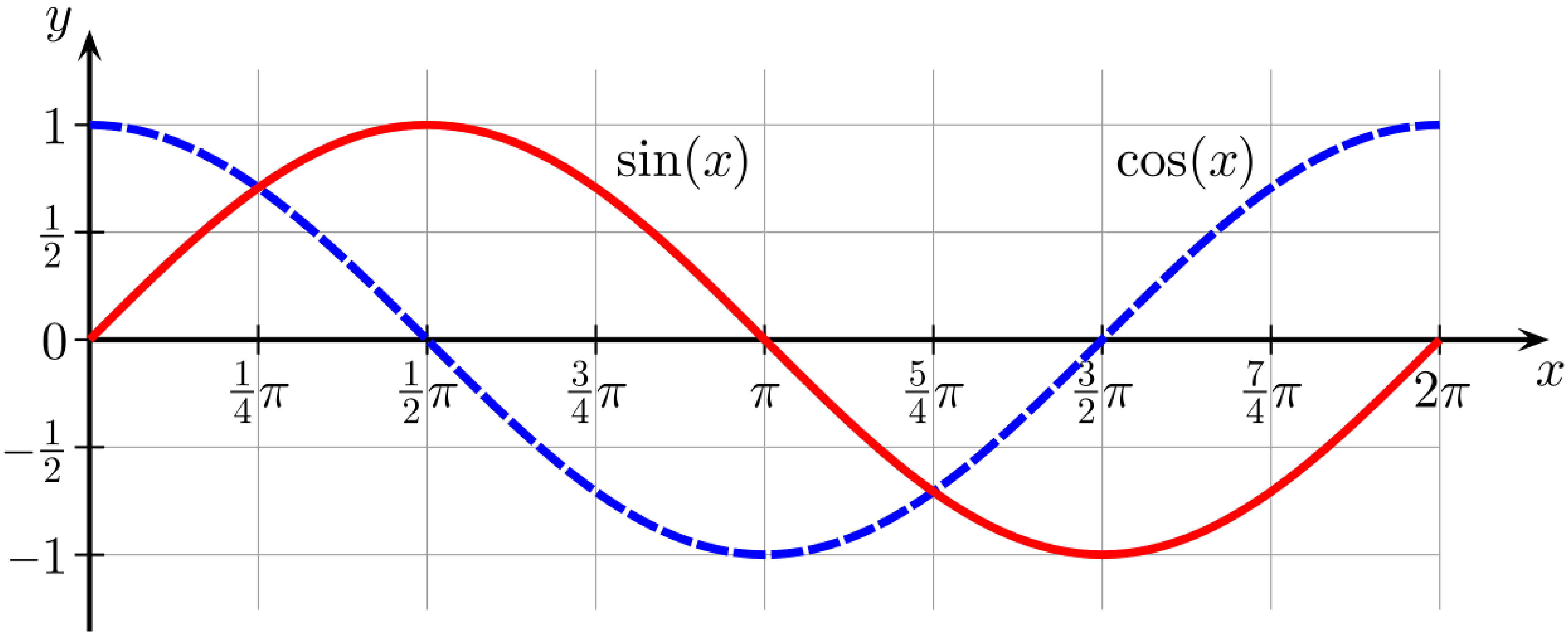
Geometric meaning of coefficients (base)

- Approximate/interpolate set of positions, derivatives, etc..



Will see one example

Possible solution?



Hermite Cubic Basis

Can satisfy with cubic polynomials as basis

$$h_{ij}(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

Obtain - solve 4 linear equations in 4 unknowns for each basis function

$$h_{ij}(t): i, j = 0,1, t \in [0,1]$$

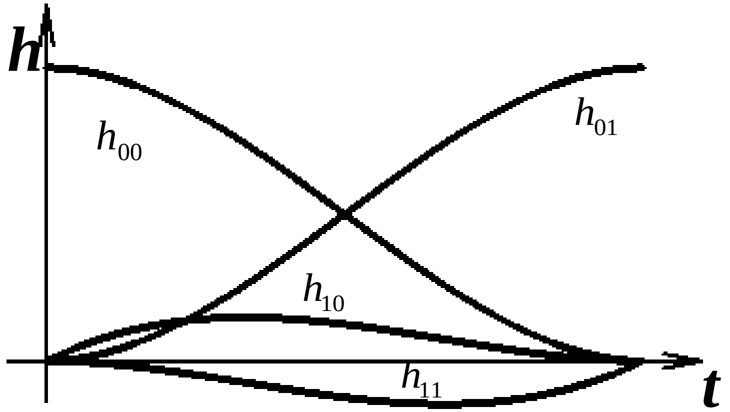
curve	$C(0)$	$C(1)$	$C'(0)$	$C'(1)$
$h_{00}(t)$	1	0	0	0
$h_{01}(t)$	0	1	0	0
$h_{10}(t)$	0	0	1	0
$h_{11}(t)$	0	0	0	1

Hermite Cubic Basis

Four cubic polynomials that satisfy the conditions

$$h_{00}(t) = t^2(2t-3)+1 \qquad h_{01}(t) = -t^2(2t-3)$$

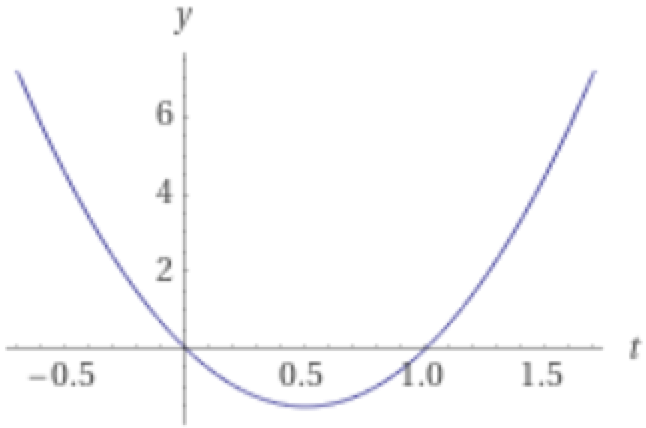
$$h_{10}(t) = t(t-1)^2 \qquad h_{11}(t) = t^2(t-1)$$



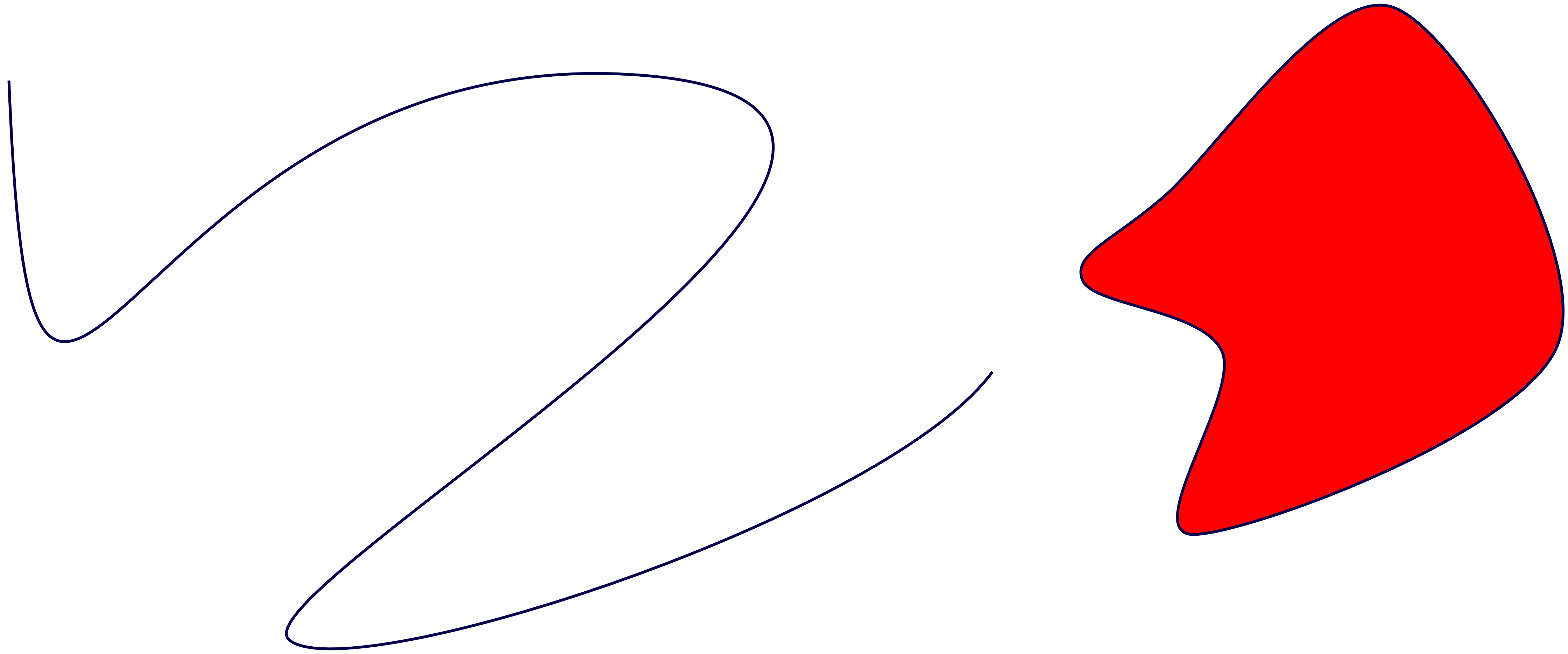
Derivative of h00

$6(-1+t)t$

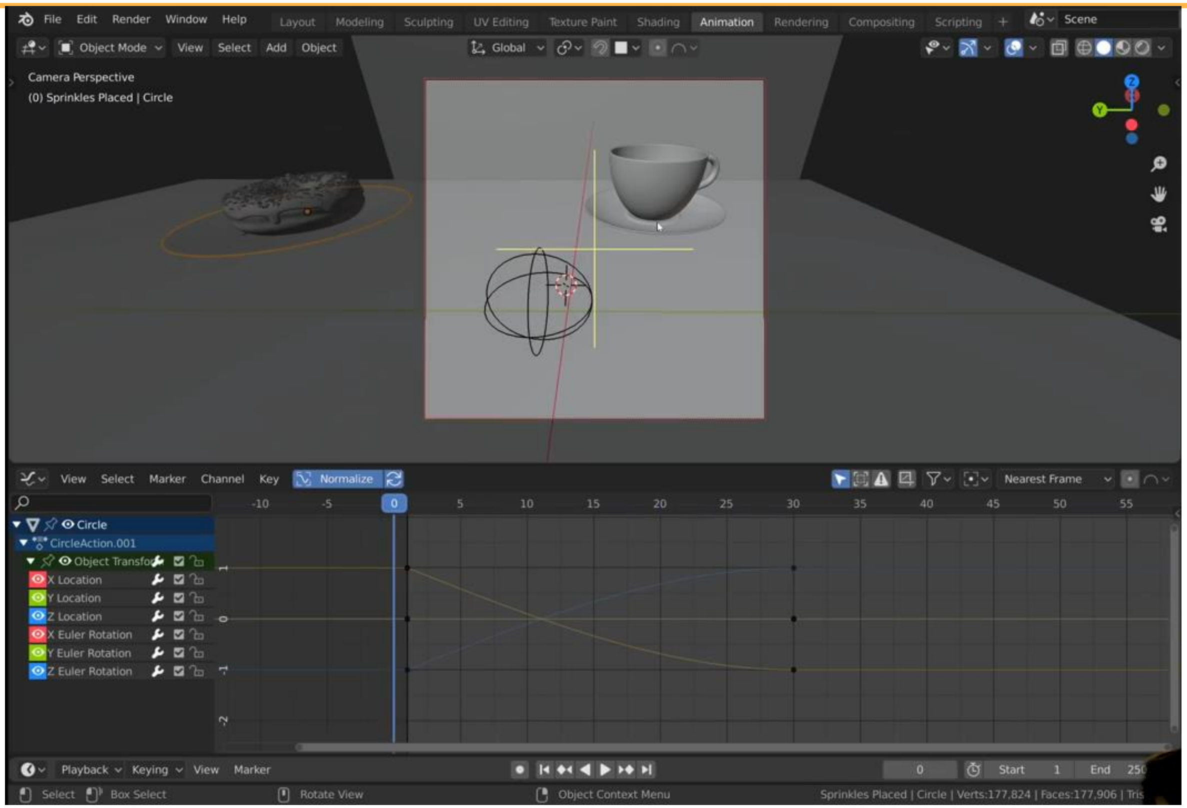
Plots:



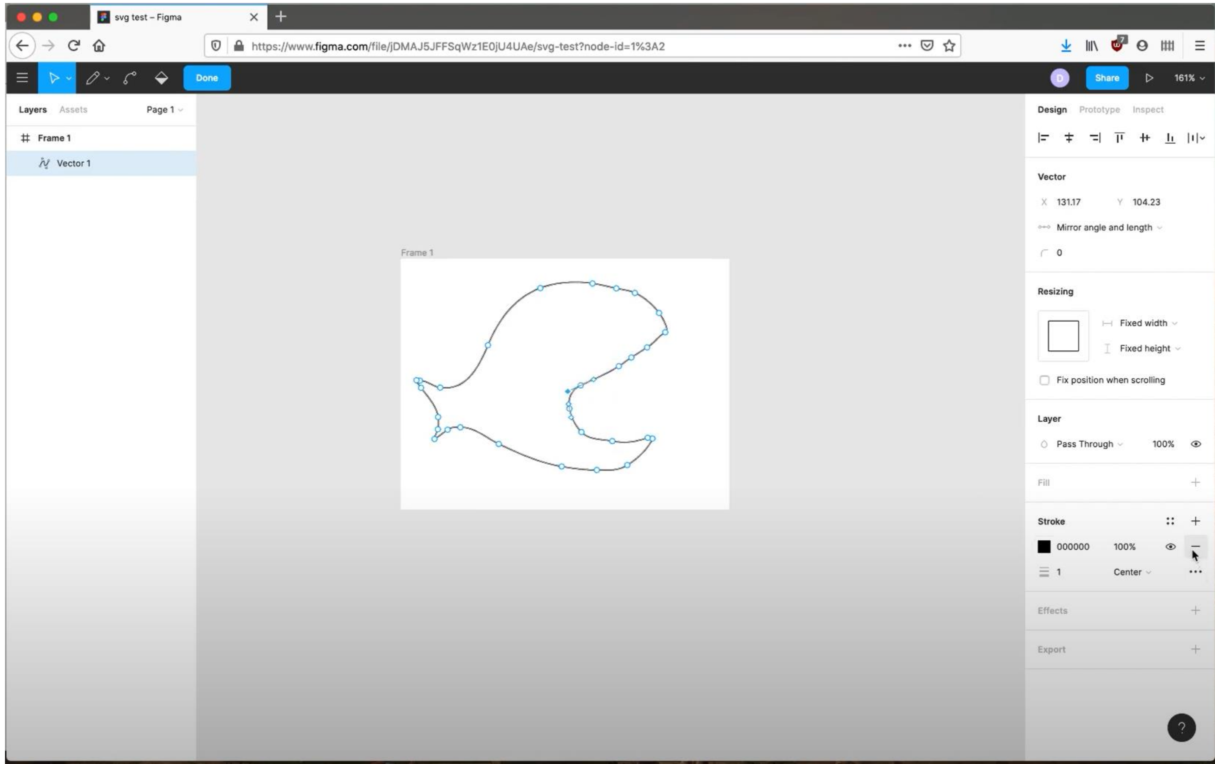
Curves



Applications:
Keyframe animation & mesh creation



https://www.youtube.com/watch?v=LLIimJxTyNw



Dave’s Tutorial