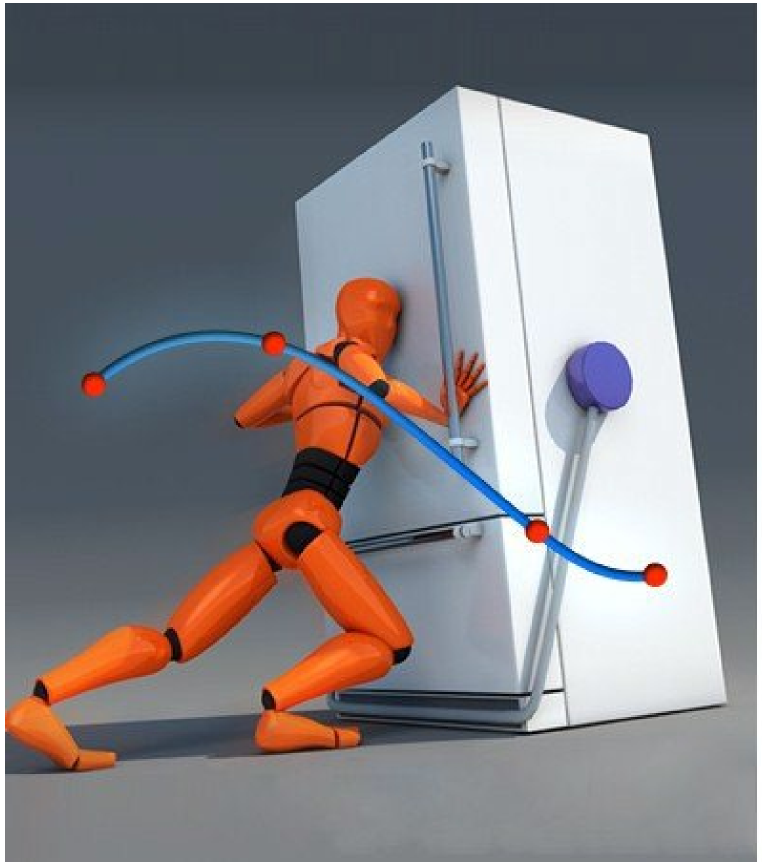
Helge Rhodin

CPSC 427
Video Game Programming

Curves and Animation



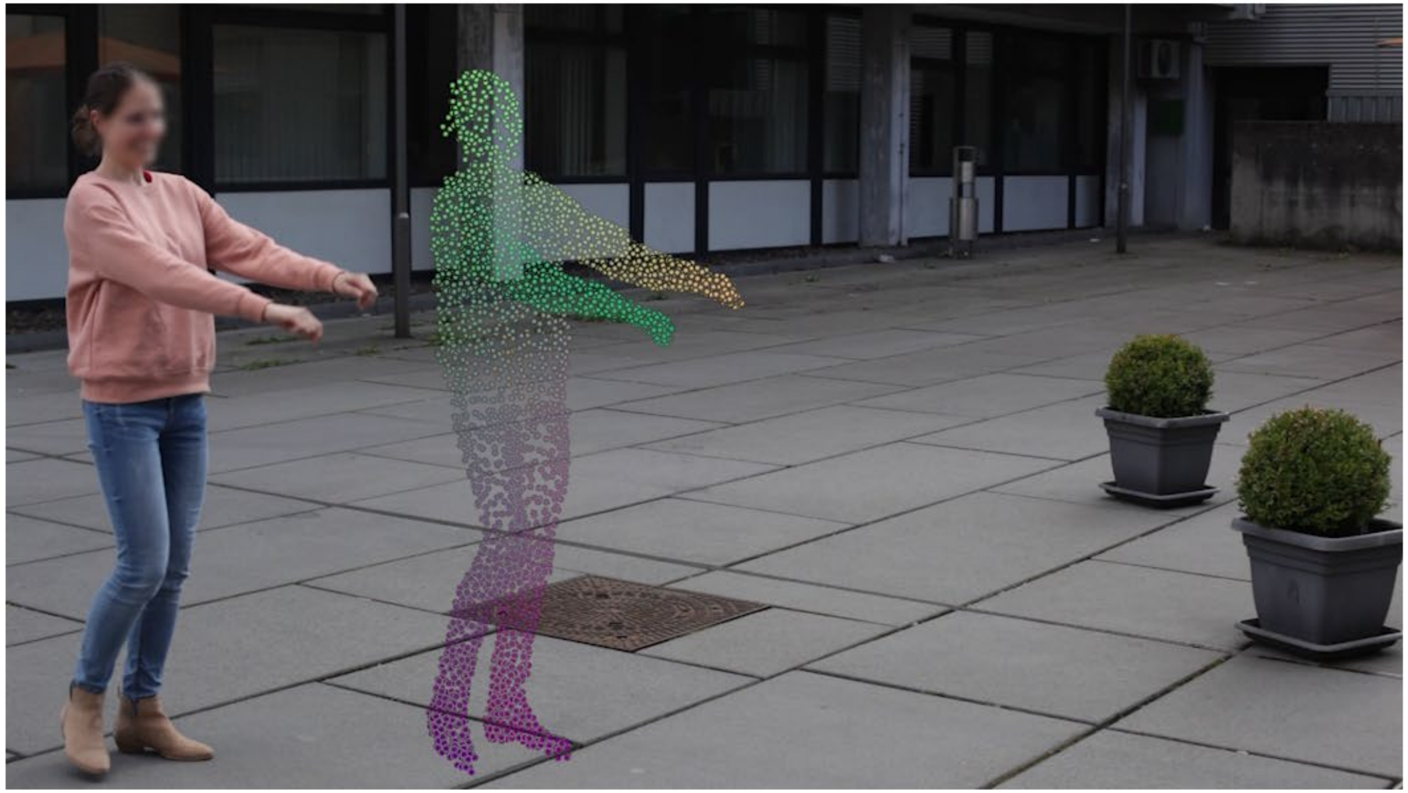

https://www.pluralsight.com/blog/film-games/stepped-vs-spline-curves-blocking-animation

Overview

1. ***Animation basics***
2. ***Curves***
3. ***Skeleton animation***

Is all this math useful?

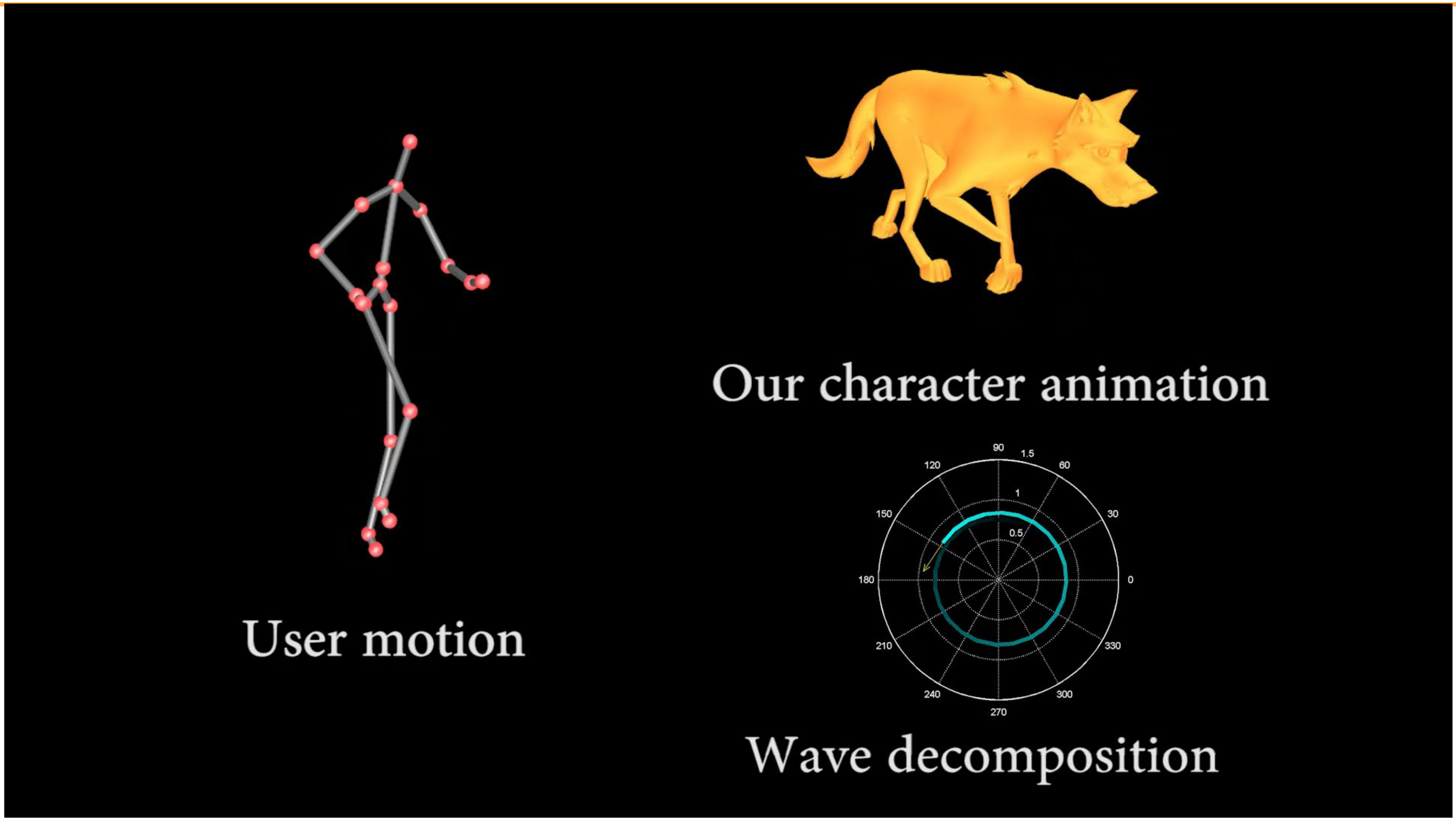
NPC: Neural Point Characters from Video [ICCV 2023]



Side-by-side overlaying of animated surface points and ground truth video frames
Setting: unseen, animated poses; fixed camera view

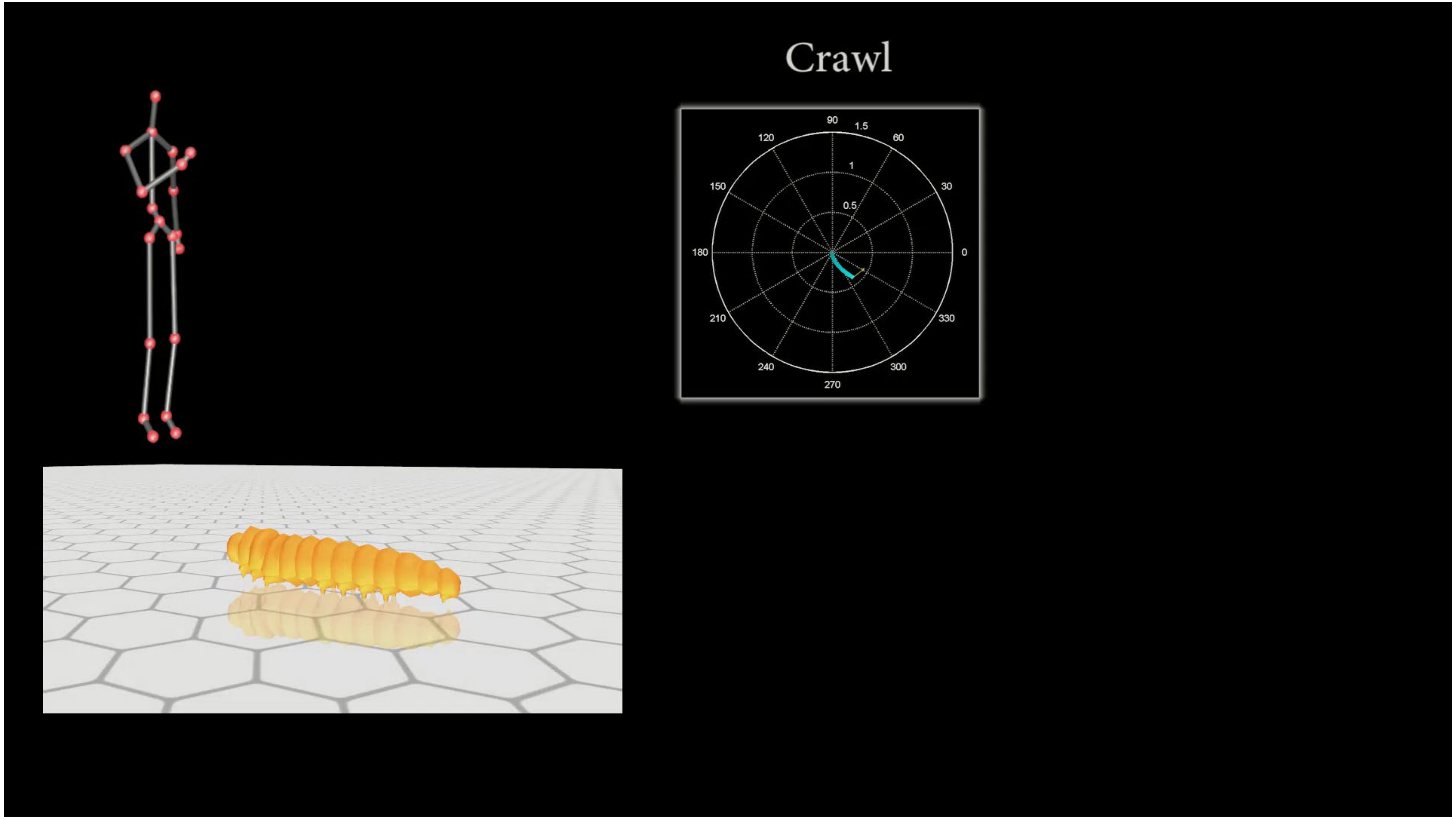
My animation background?

SIGGRAPH
Asia 2015



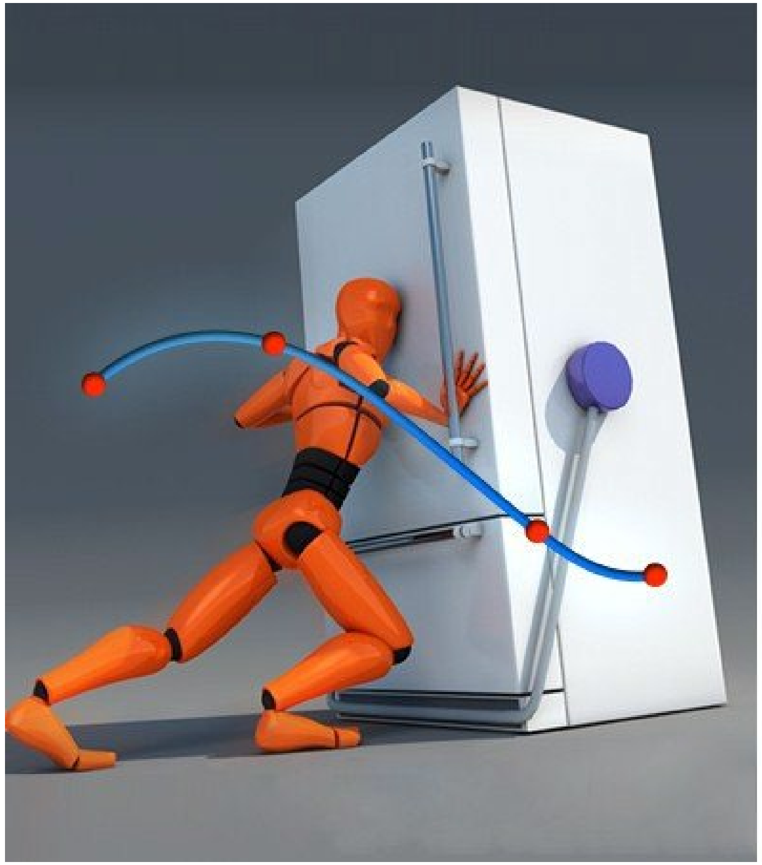
Generalizing Wave Gestures from Sparse Examples for Real-time Character Control

My animation background?



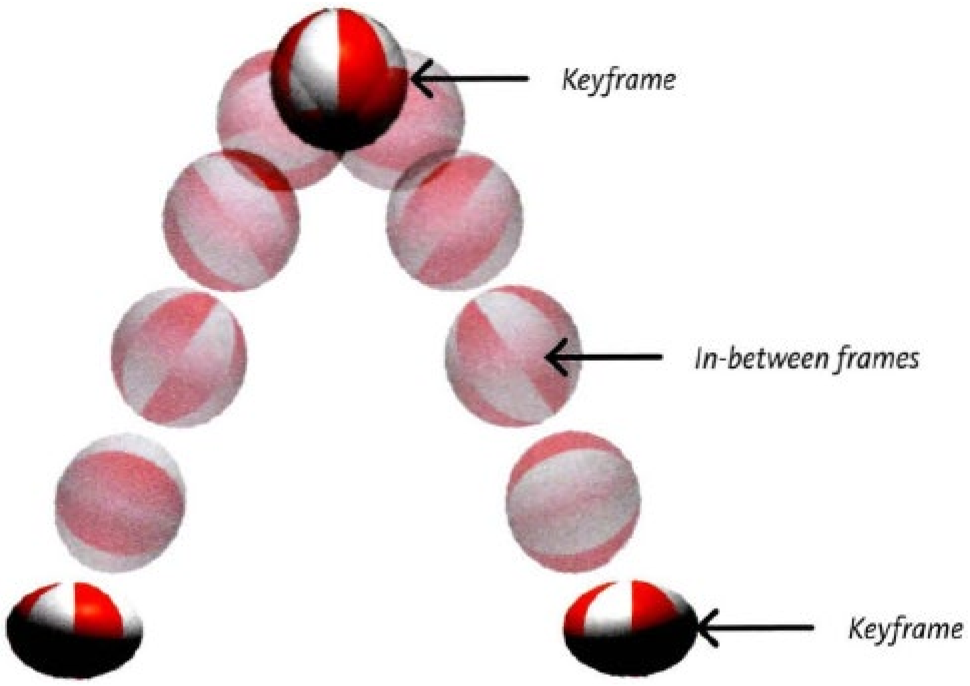
CPSC 427
Video Game Programming

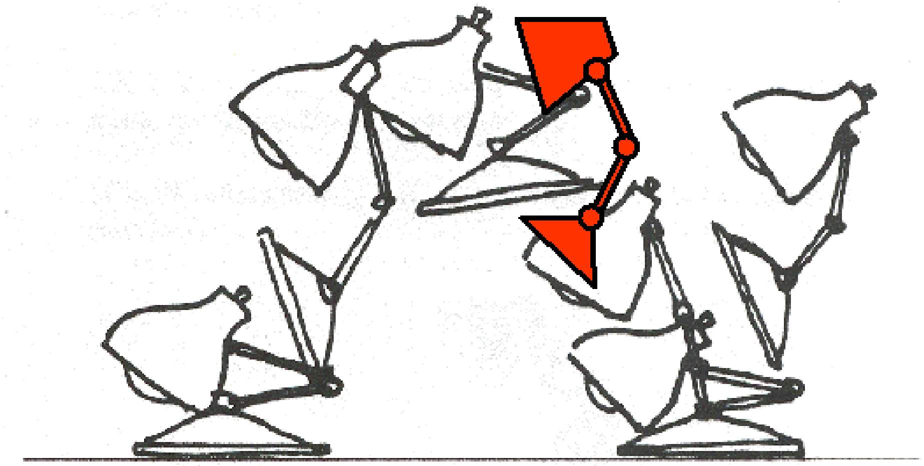
Curves and Animation



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

Keyframe animation





Lasseter `87

Recap: Line equation

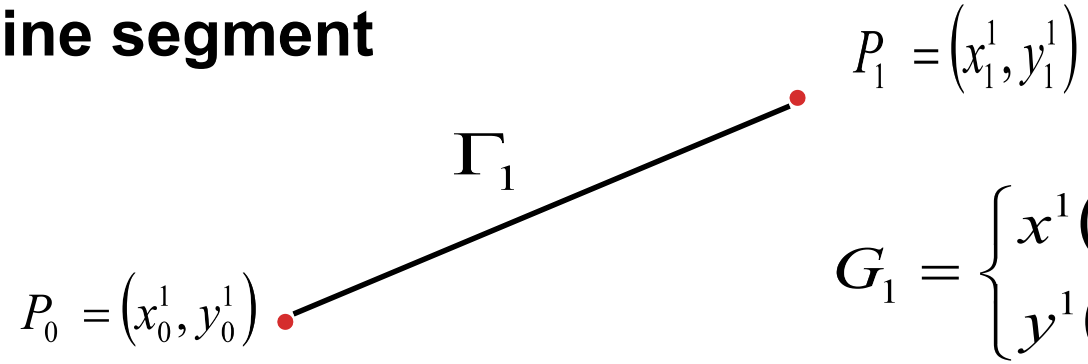
Parametric form

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

$$C(t) := \begin{pmatrix} P^0_y \\ P^0_x \end{pmatrix} t + \begin{pmatrix} P^1_y \\ P^1_x \end{pmatrix} (1-t)$$

What things can we interpolate?

Line segment



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

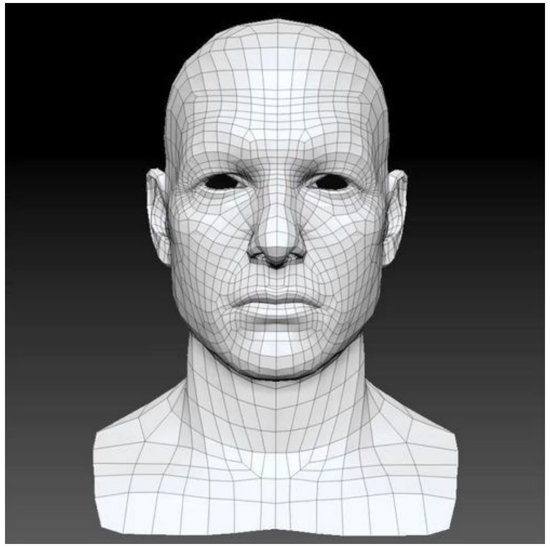
Interpolating general properties

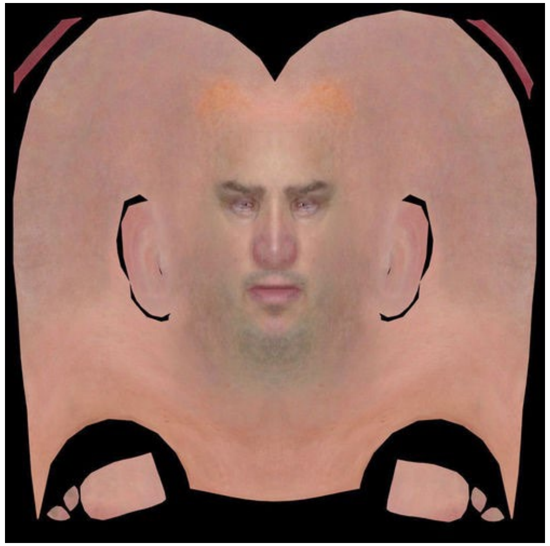
- *position*
- *aspect ratio?*
- *scale*
- *color*
- *What else?*

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

$$s^0 \qquad s^1$$

$$c^0 \qquad c^1$$





Barycentric coordinates / interpolation

Other Parametric Functions

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

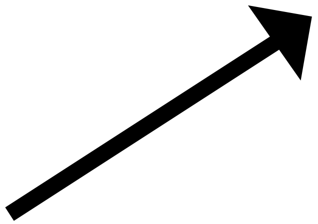
$$C(t) := \begin{pmatrix} \cos t \\ \sin t \end{pmatrix}$$

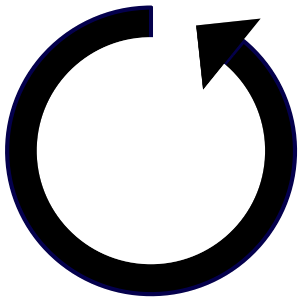
Line segment

Circle (arc)

?

Splines



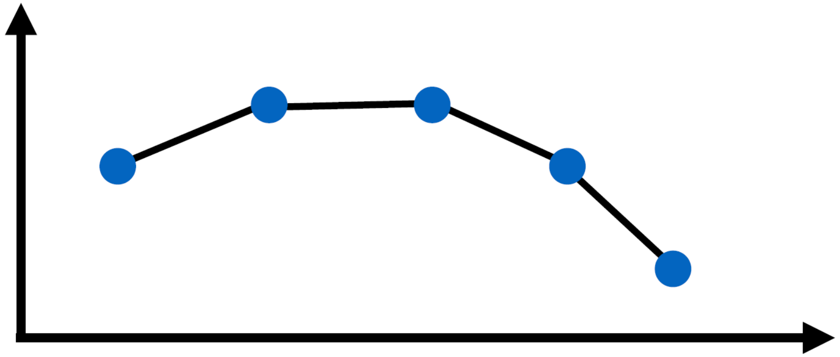


Splines

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

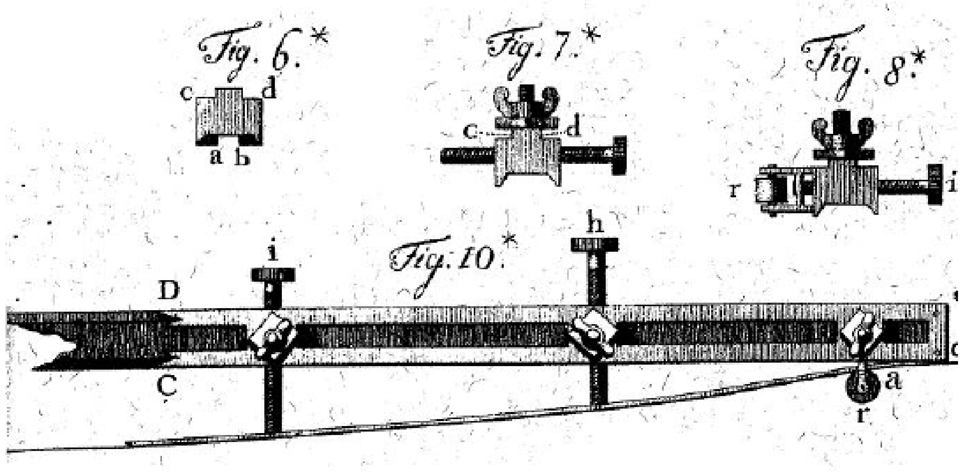
- $C(t)=[x(t),y(t)]$ or $C(t)=[x(t),y(t),z(t)]$

Description = basis functions + coefficients

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

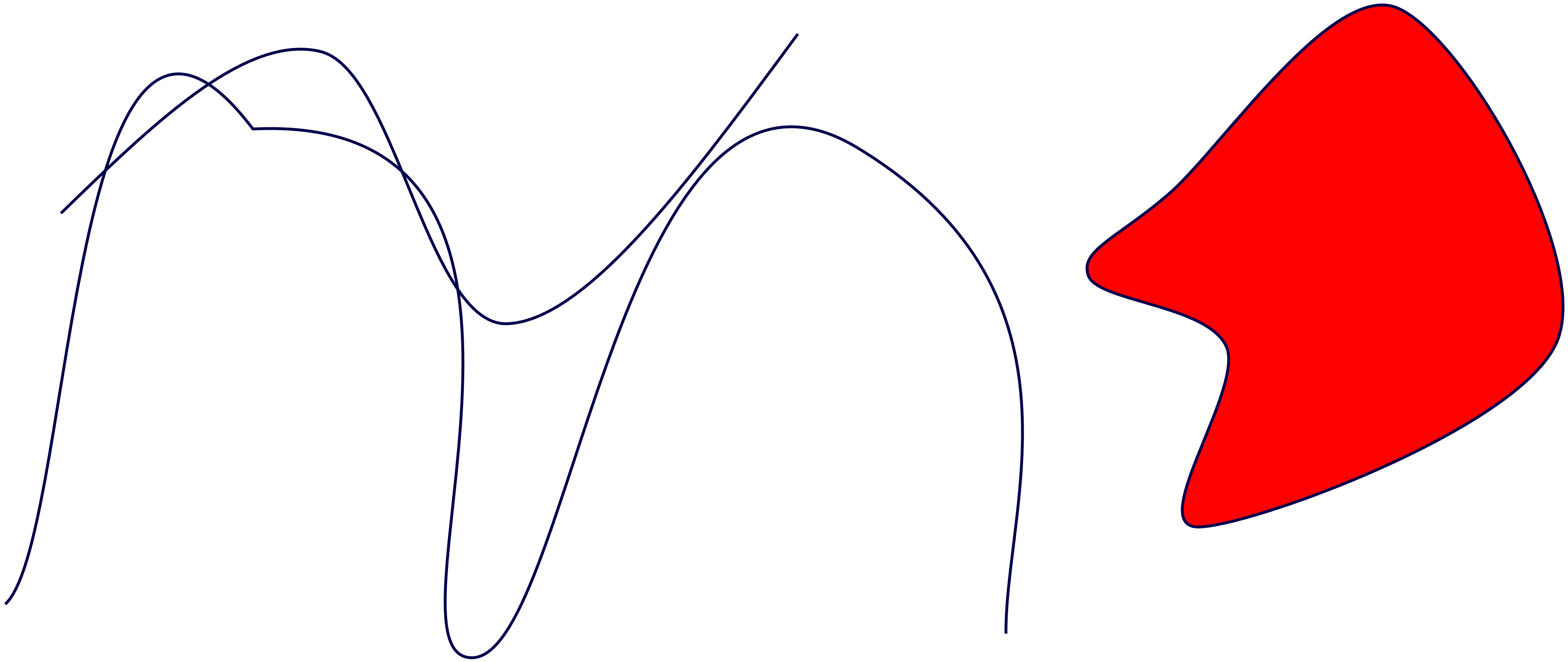
$$x(t)=\sum_{i=0}^{n}P_i^xB_i(t)$$

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

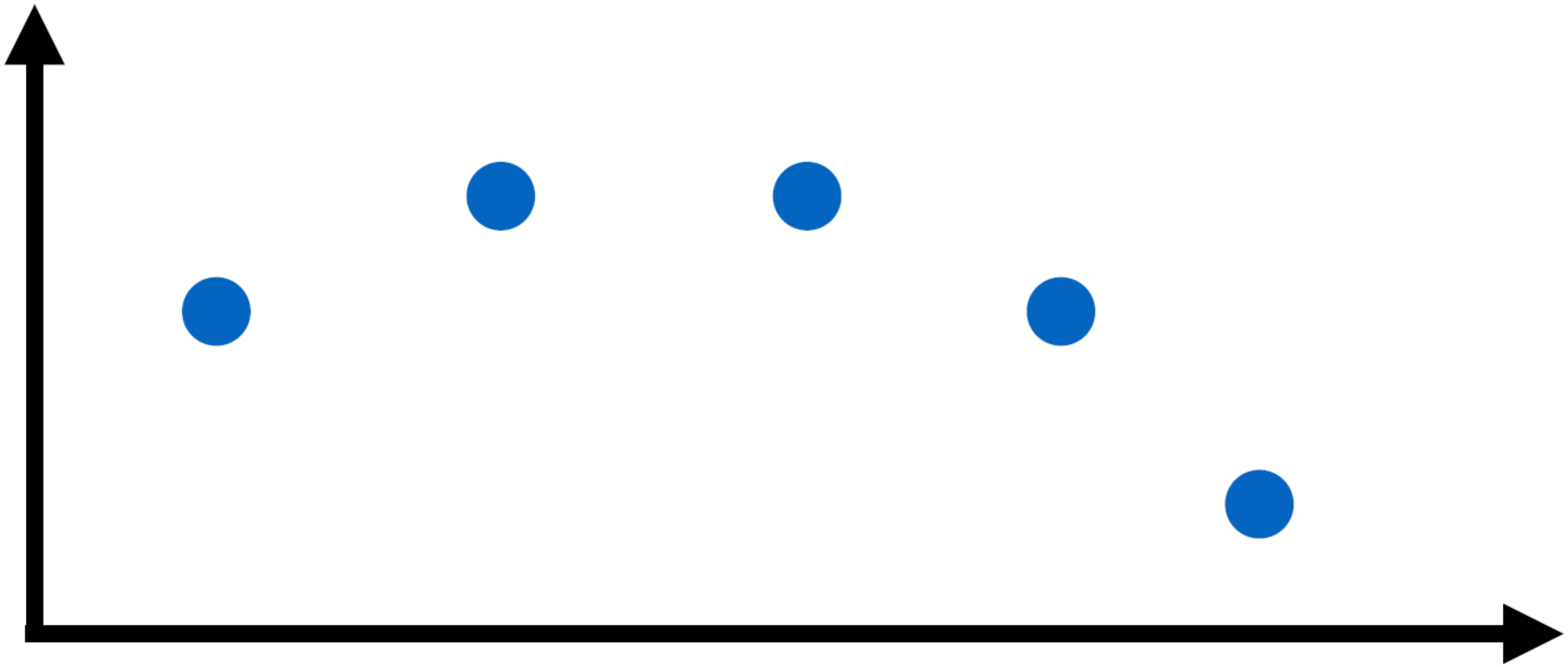


- Same basis functions for all coordinates

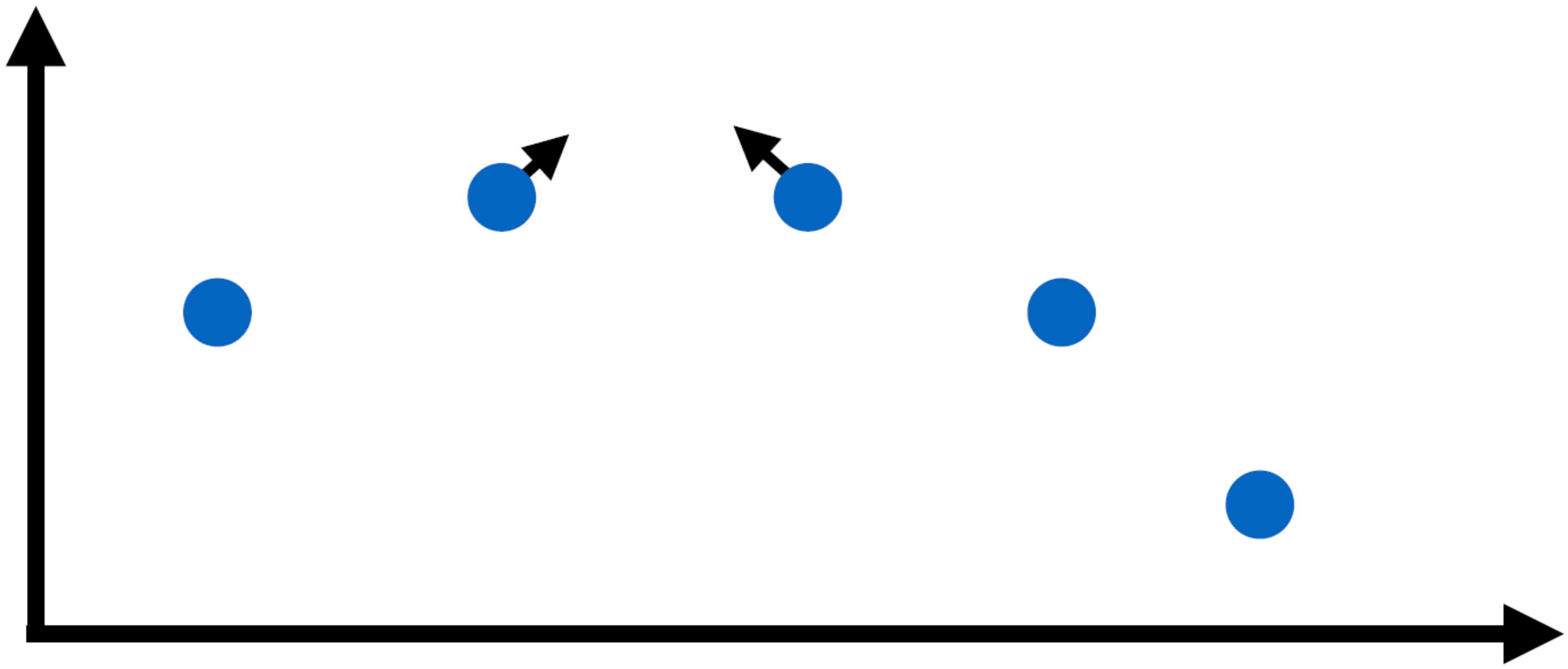
Curves



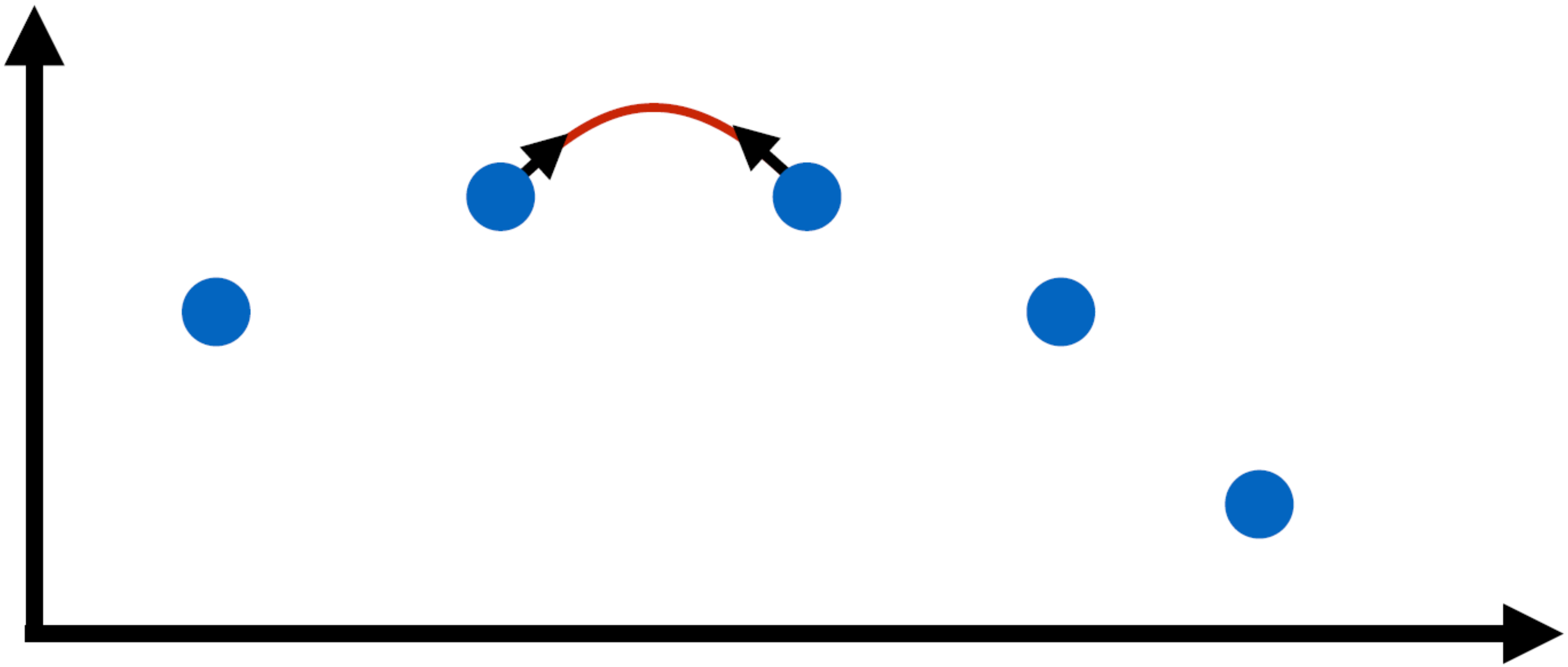
Smooth curve



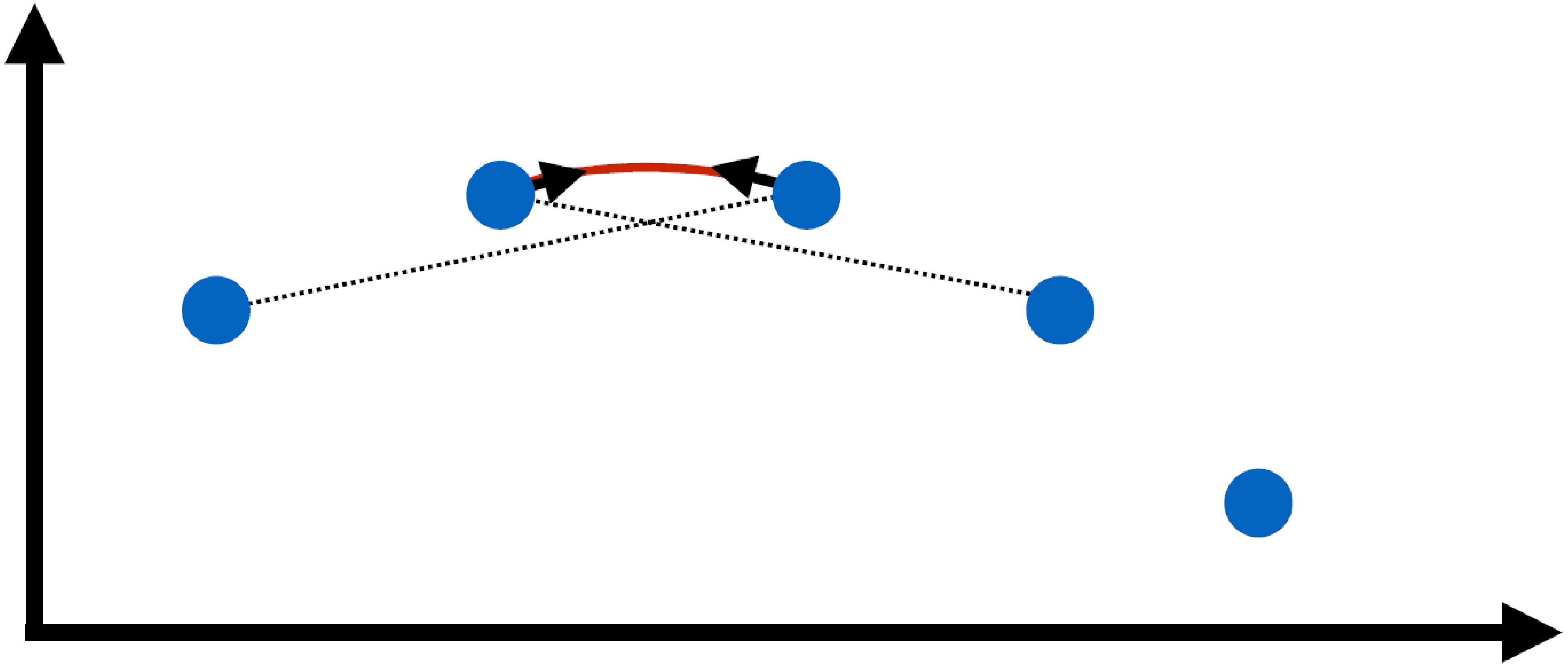
Smooth curve



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

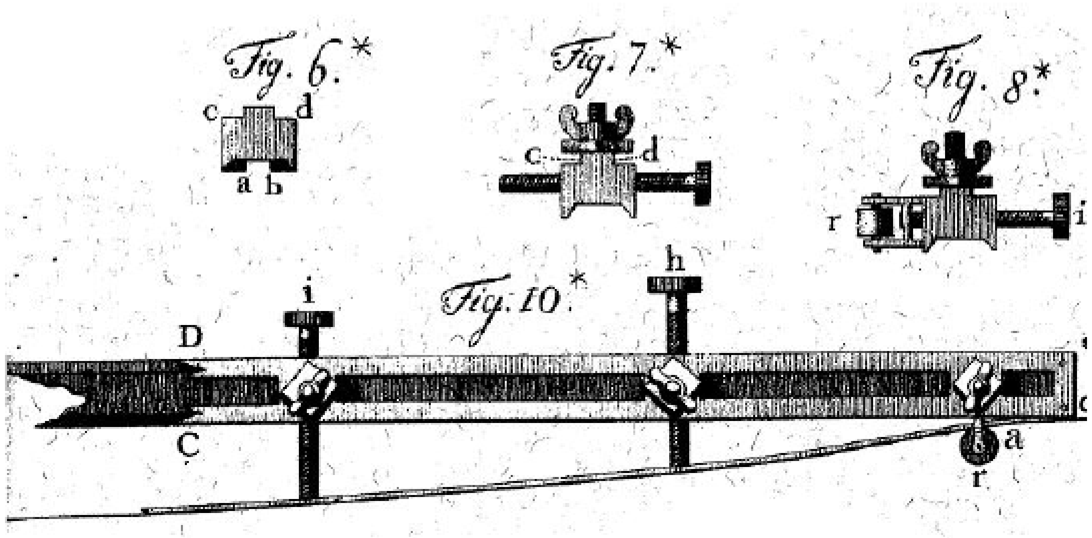
$\boldsymbol{h_{00}'(0) = h_{00}'(0) = 0}$

$\boldsymbol{h_{00}(0) = 1}$

Splines – Free Form Curves

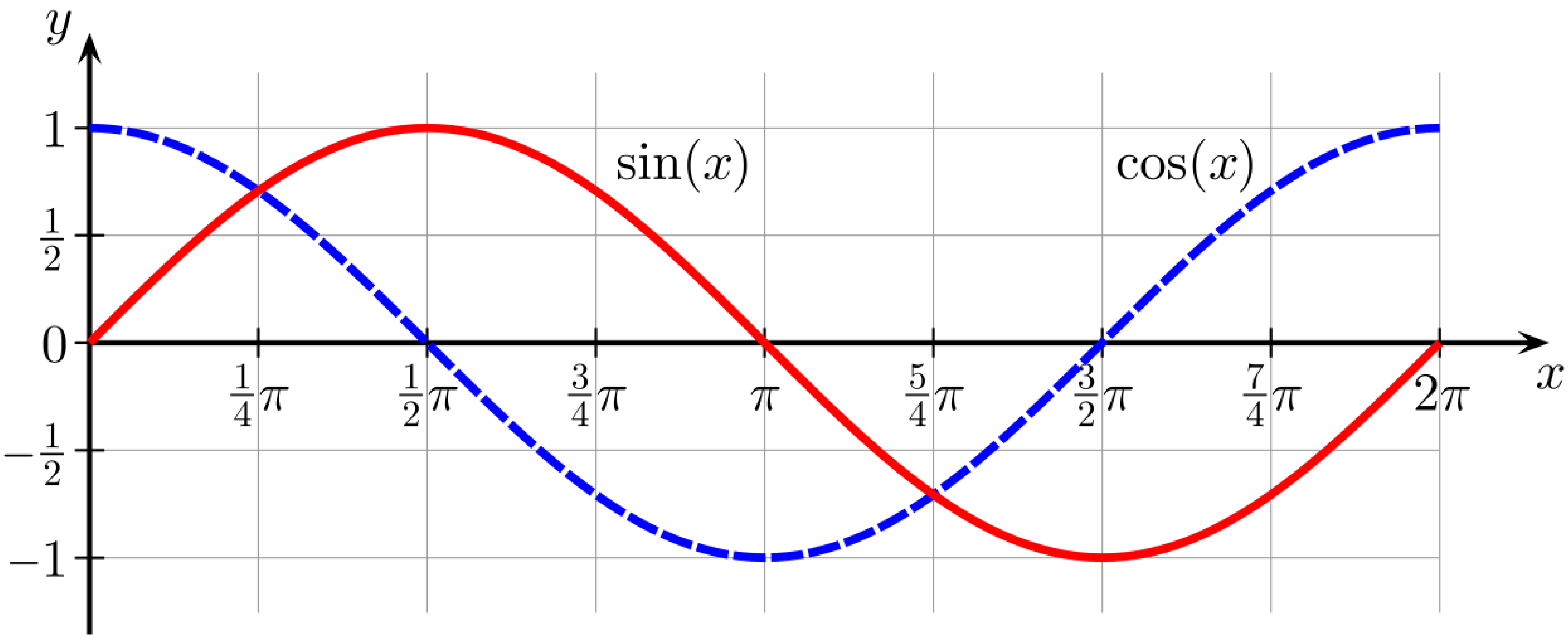
Geometric meaning of coefficients (base)

- 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

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

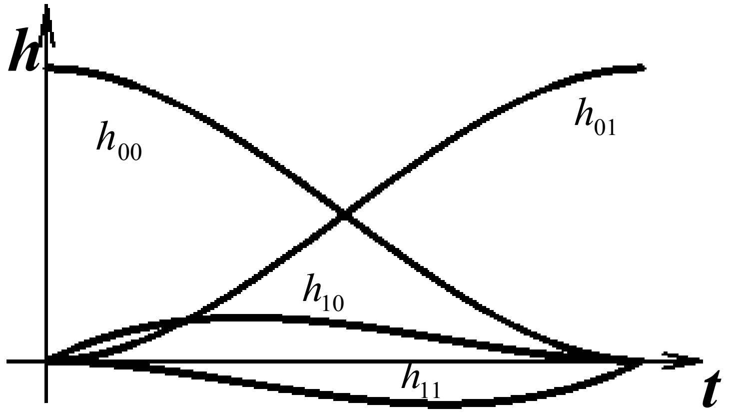
curve	$C(0)$	$C(1)$	$C'(0)$	$C'(1)$
$h_{00}(t)$	1	0	0	0
$h_{01}(t)$	0	1	0	0
$h_{10}(t)$	0	0	1	0
$h_{11}(t)$	0	0	0	1

Hermite Cubic Basis

Four cubic polynomials that satisfy the conditions

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

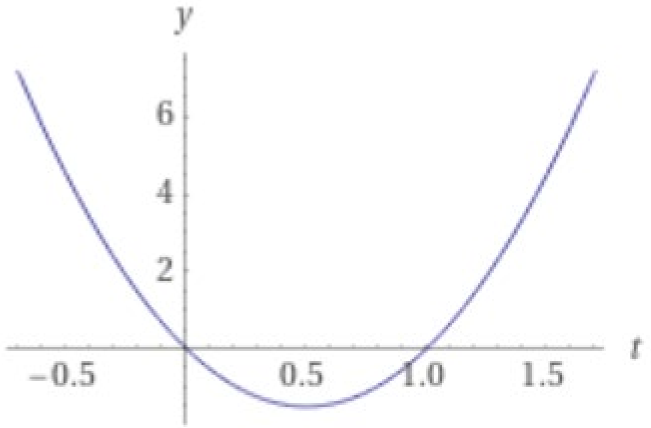
$$h_{10}(t) = t(t-1)^2 \qquad h_{11}(t) = t^2(t-1)$$



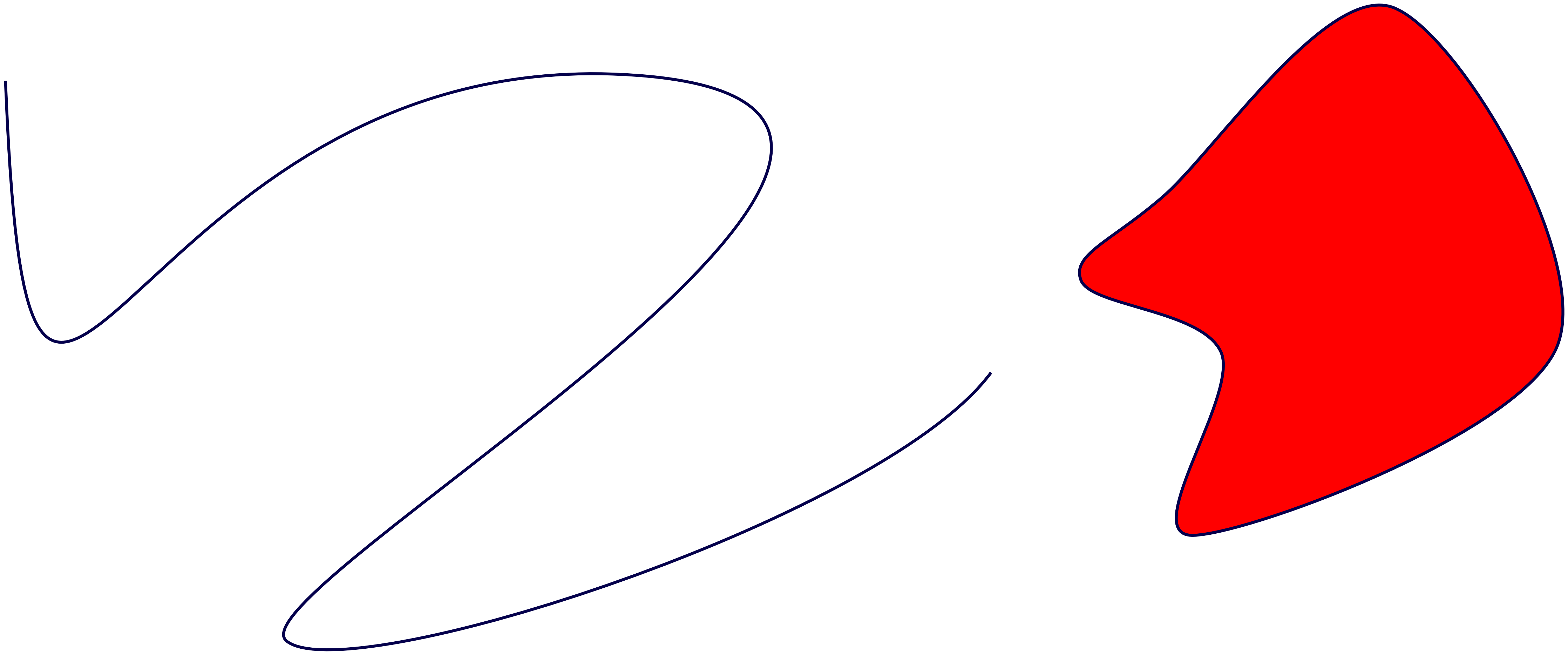
Derivative of h00

$6(-1+t)t$

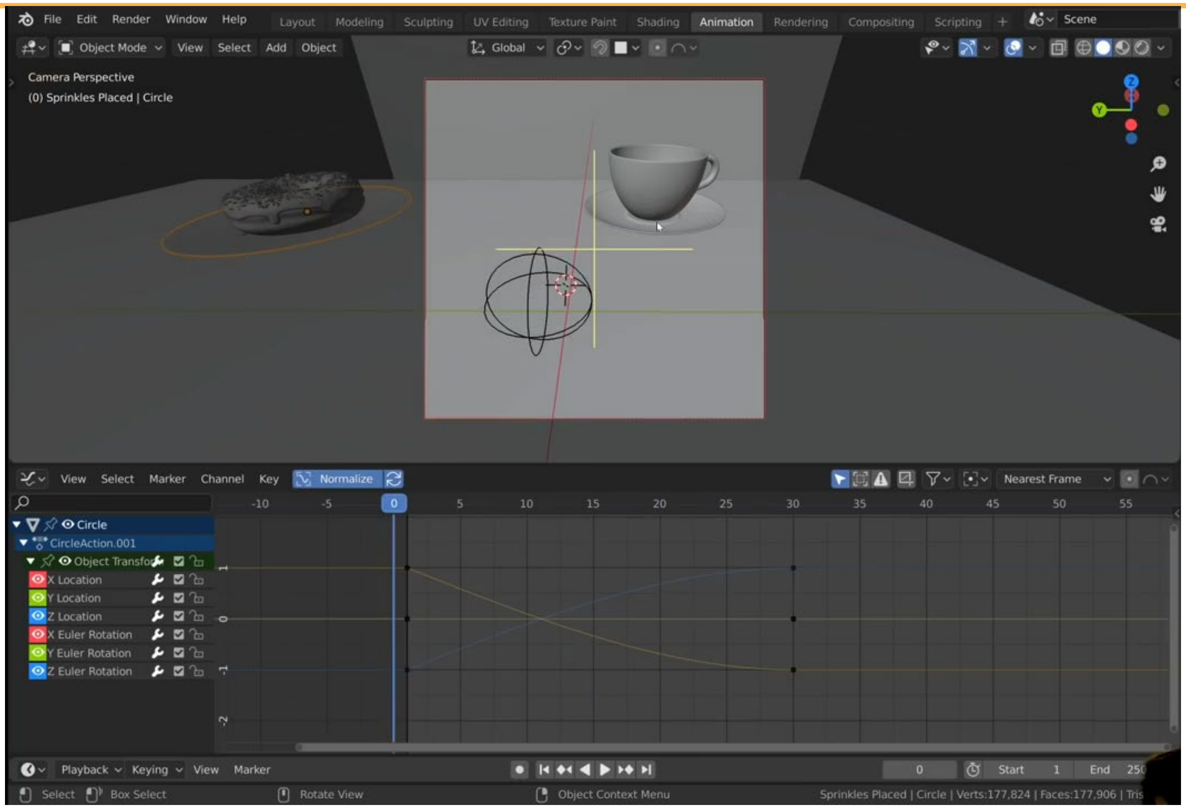
Plots:



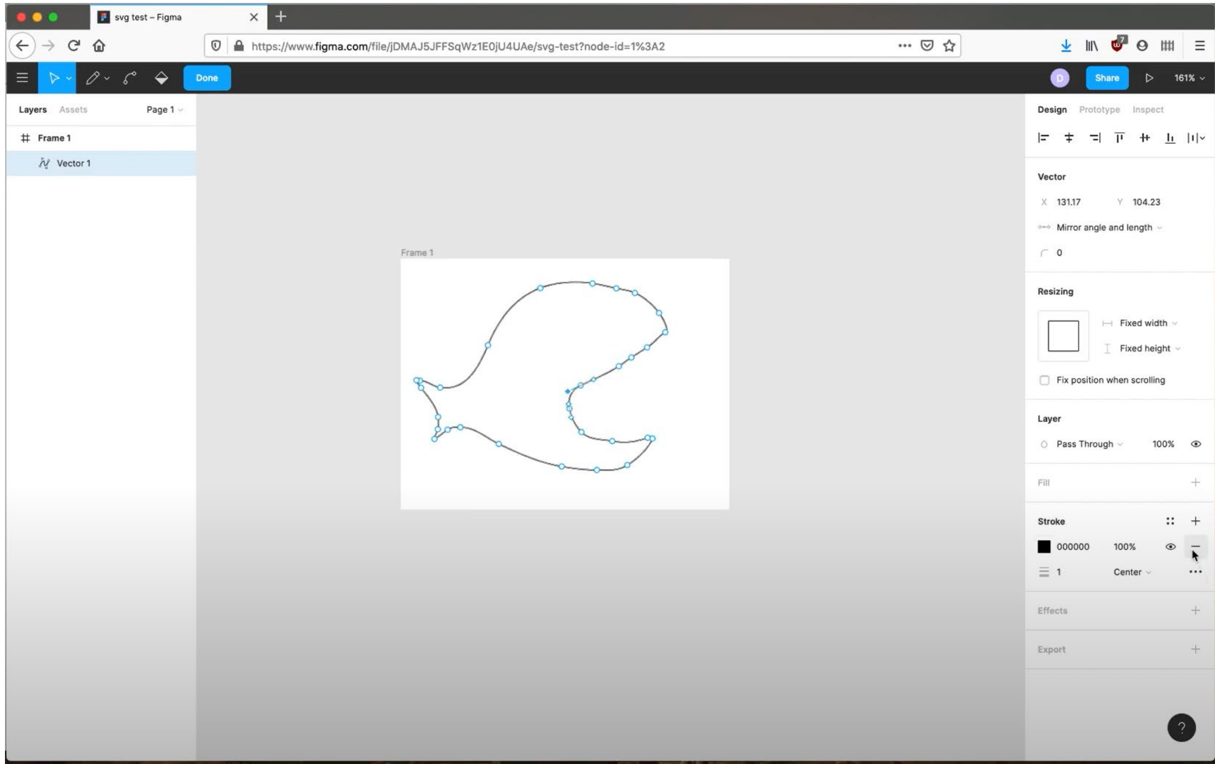
Curves



Applications:
Keyframe animation & mesh creation



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



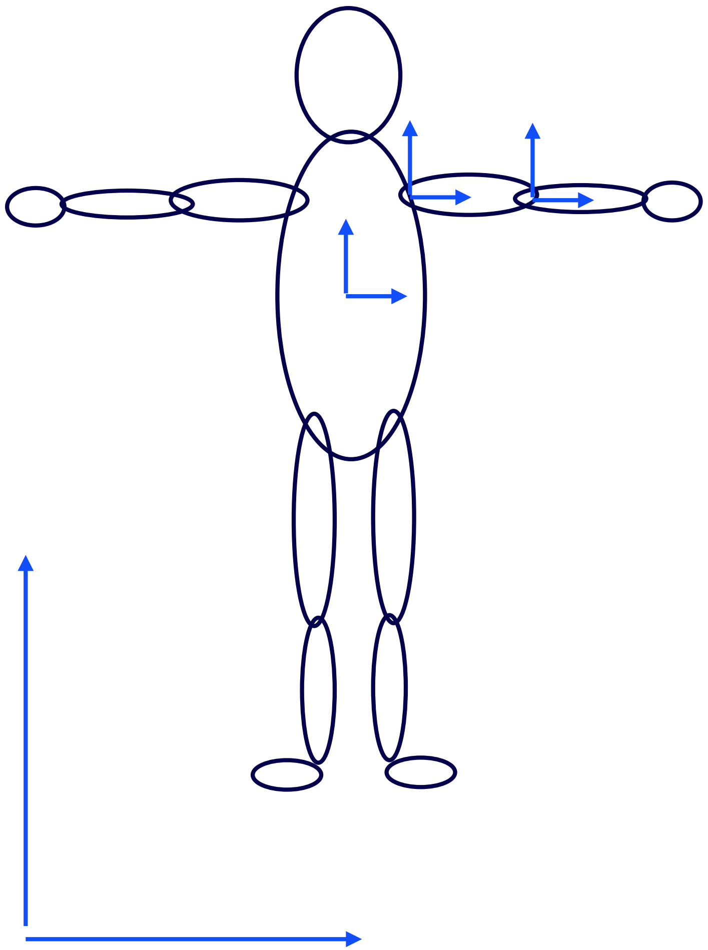
Dave’s Tutorial

Cross play moved, to have more space!

9	Mon	30-Oct	Lecture: Curves and splines	Milestone 2	
	Wed	1-Nov	Lecture: Team-report M2 Tutorial: Face2Face grading M2		
10	Mon	6-Nov	Lecture: Simple AI Lecture: Two-player AI	Assignment 2	
	Wed	8-Nov	Cross-play M2 Tutorial: Face2Face grading A2	Peer review A2 (due on Thursday)	
11	Mon	13-Nov	Break		
	Wed	15-Nov	Break		
12	Mon	20-Nov	Lecture: Balancing games Guest lecture by Tink Leguiader? (Behaviour Interactive)	Milestone 3	
	Wed	22-Nov	Lecture: Cross-play M3 Tutorial: Face2Face grading M3		
13	Mon	27-Nov	Guest lecture by Yggy King (Blackbird Interactive) on "ECS in practice" Team-report M3		
	Wed	29-Nov	Guest lecture by Cloé Veilleux (Relic Entertainment) "Cutting corners in AI"		
14	Mon	4-Dec	Lecture: The history and future of game technology Guest Lecture by Alex Denford and Cate Mackenzie (SkyBox) "Empowering Creators"	M4 submission	
	Wed	6-Dec	Lecture: Team-report M3 Tutorial: Face2Face grading M4		
Exam slot TBD			Cross-play M4 Awards		

Transformation Hierarchies

Transformation Hierarchies



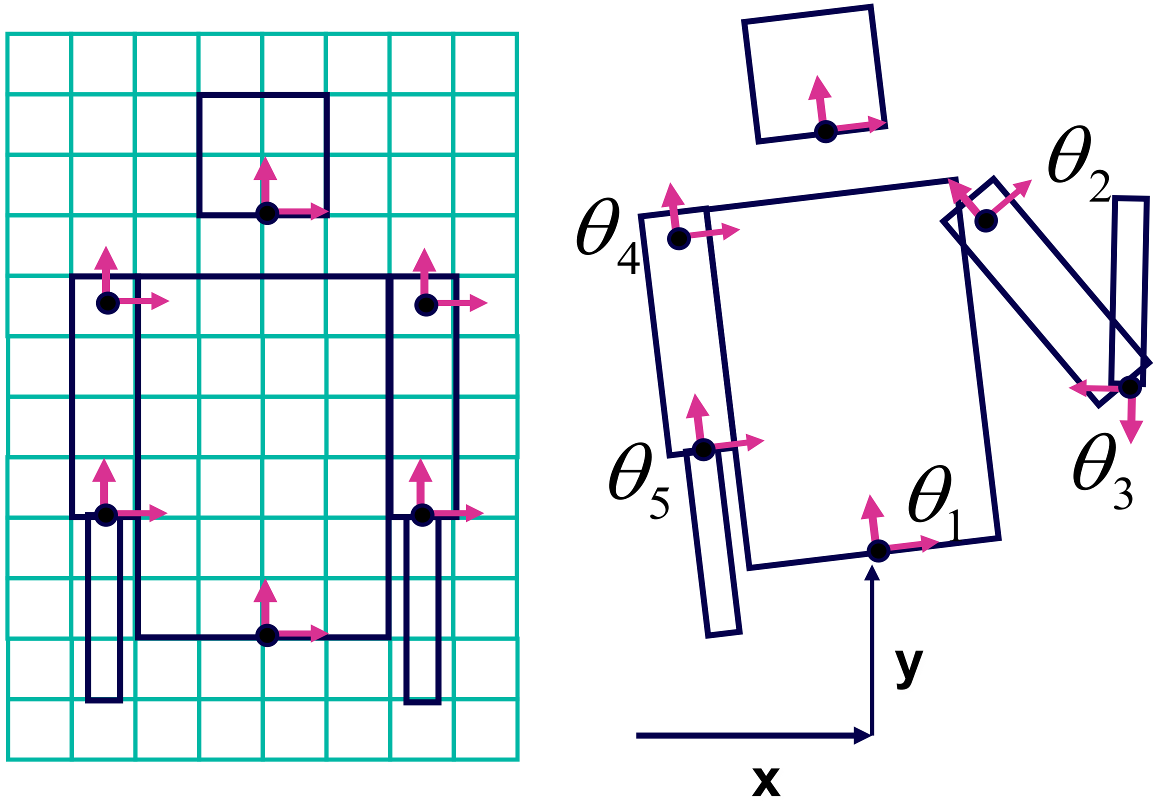
Scenes have multiple coordinate systems

- Often strongly related
  - *Parts of the body*
  - *Object on top of each other*
    - Next to each other…

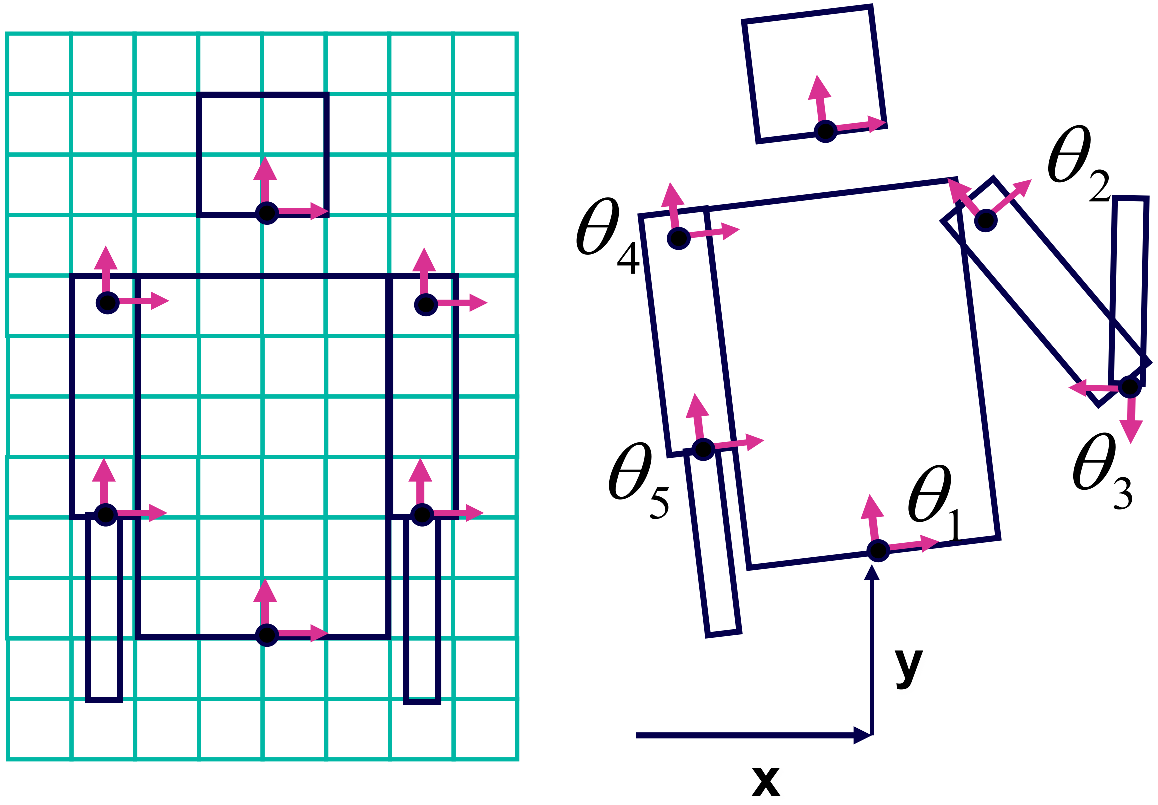
Independent definition is bug prone

Solution: Transformation Hierarchies

Transformation Hierarchy Examples



Transformation Hierarchy Examples

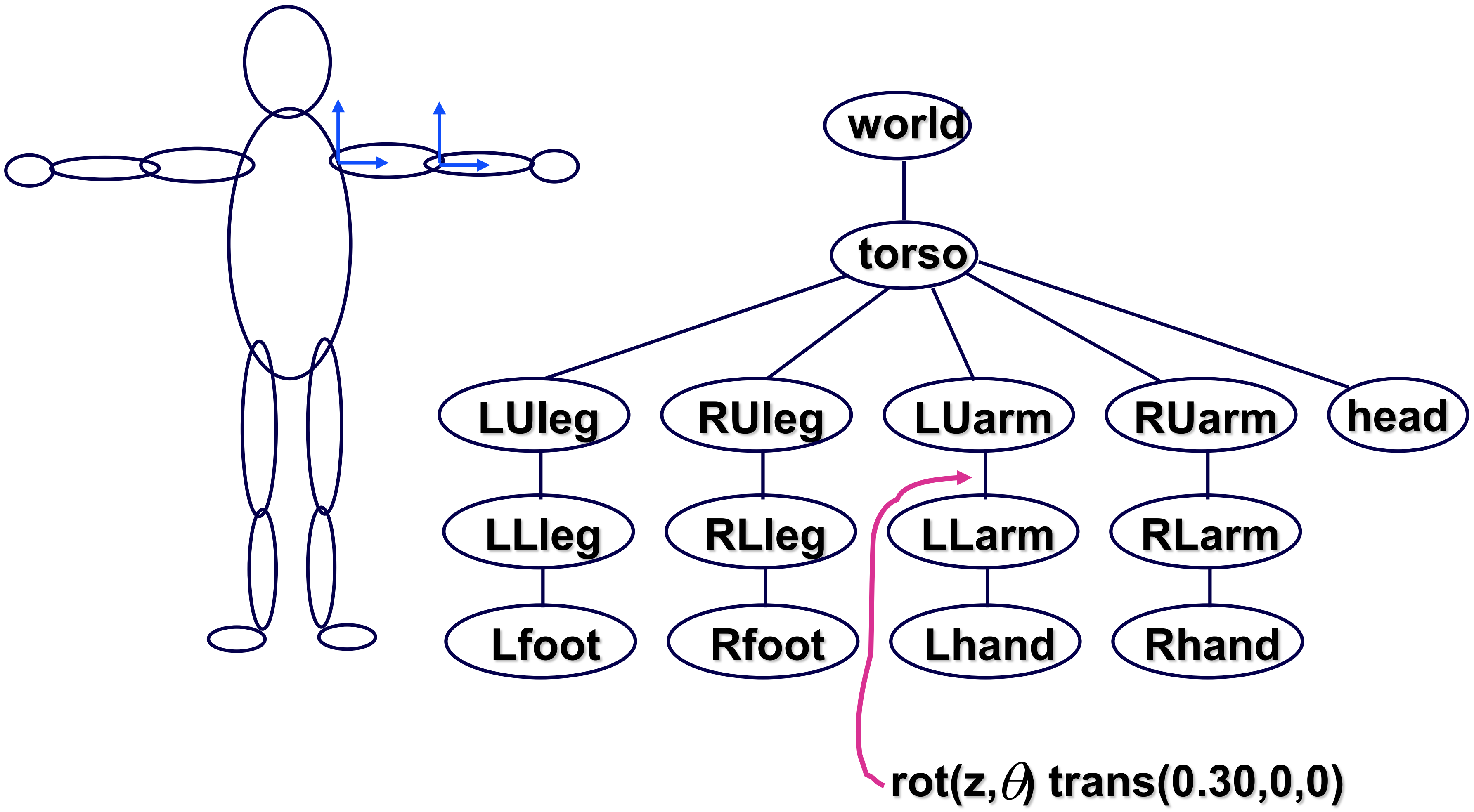


$$M_1 = Tr_{(x,y)} \cdot Rot\theta_1$$

$$M_2 = M_1 \cdot Tr_{(2.5,5.5)} \cdot Rot\ \theta_2$$

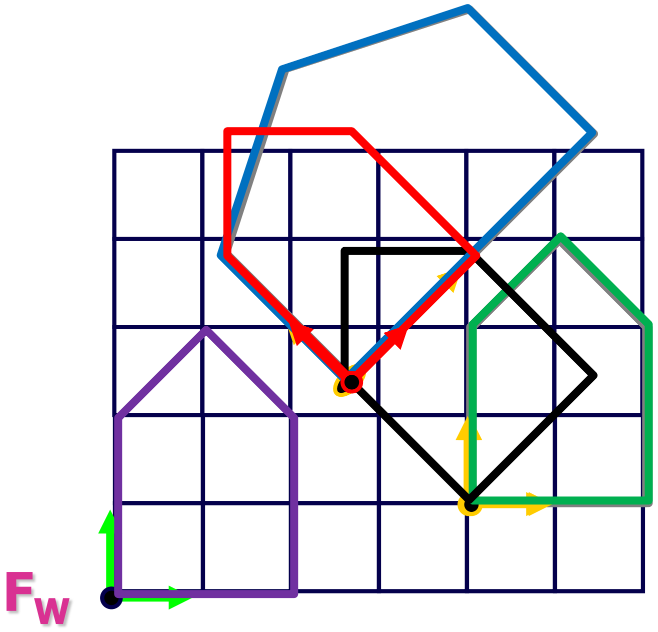
$$M_3 = M_2 \cdot Tr_{(0,-3.5)} \cdot Rot\ \theta_3$$

Transformation Hierarchies



Transformation Hierarchy Quiz

```
M.setIdentity();
M = M*Translation(4,1,0);
M = M*Rotation(pi/4,0,0,1);
House.matrix = M;
```



Which color house will we draw?

A. Red
B. Blue
C. Green
D. Orange
E. Purple

Hierarchical Modeling

Advantages

- Define object once, instantiate multiple copies
- Transformation parameters often good control knobs
- Maintain structural constraints if well-designed

Limitations

- Expressivity: not always the best controls
- Can’t do closed kinematic chains
  - *E.g., how to keep a hand on the hip?*

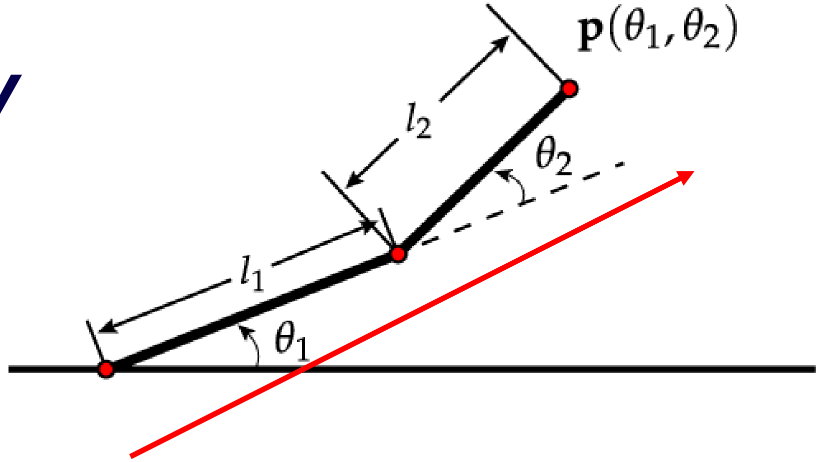
Inverse Kinematics

- ***How to reach goal position?***
- ***Chain of transformation to reach a certain point?***
- ***What kind of a problem is this?***
  - linear/non-linear?
  - convex/non-convex?
  - How can we solve it?

Forward vs. inverse kinematics

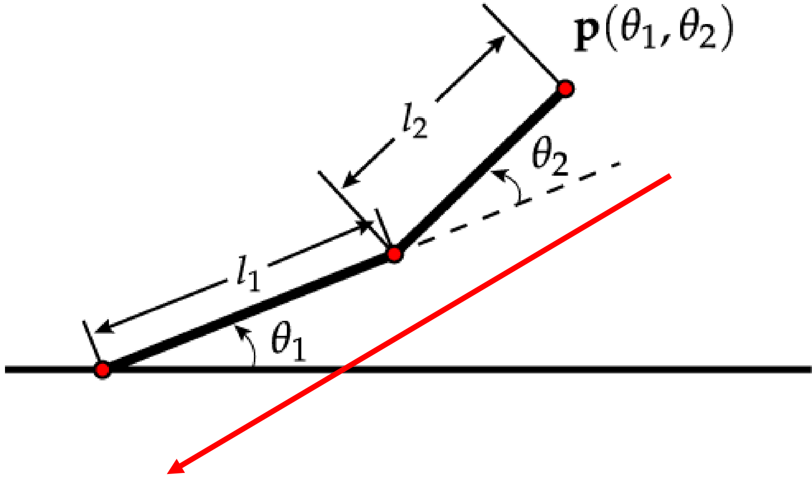
Forward kinematics

- ***given joint axis, angle, and skeleton hierarchy***
- ***compute joint locations***
  - *start at the end-effector (e.g. arm)*
    - rotate all parent joints (up the hierarchy) by θ
  - *iteratively continue from child to parent*



Inverse kinematics

- given skeleton hierarchy and goal location
- optimize joint angles (e.g. gradient descent)
- minimize distance between end effector (computed by forward kinematics) and goal locations



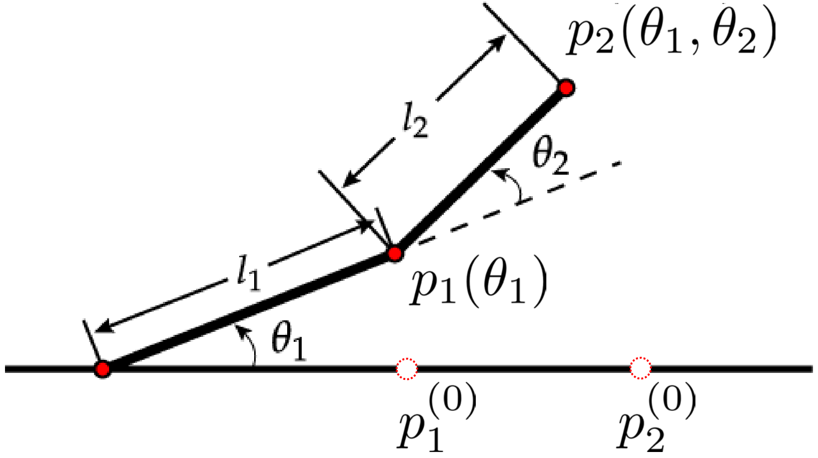
Inverse kinematics (IK)

- ***non-linear in the angle (due to cos and sin)***

$$M_1 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \end{bmatrix} \quad M_2 = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & -l_1 \\ \sin\theta_2 & \cos\theta_2 & 0 \end{bmatrix}$$

- linear/affine given a set of rotation matrices

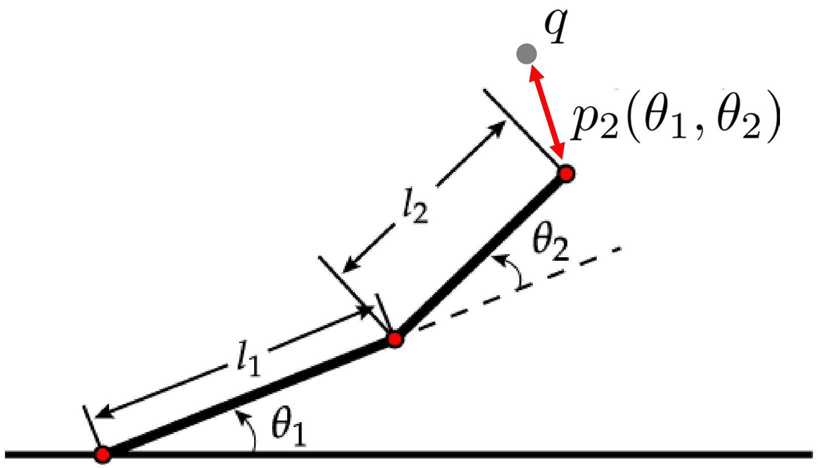
$$p_2(\theta_1, \theta_2) = M_1 M_2 (p_2^{(0)} - p_1^{(0)})$$



Inverse kinematics

- minimize objective to reach goal location

$$O(\theta_1, \theta_2) = \|q - p_2(\theta_1, \theta_2)\|$$



Example IK framework:
https://rgl.s3.eu-central-1.amazonaws.com/media/pages/hw4/CS328_-_Homework_4_3.ipynb