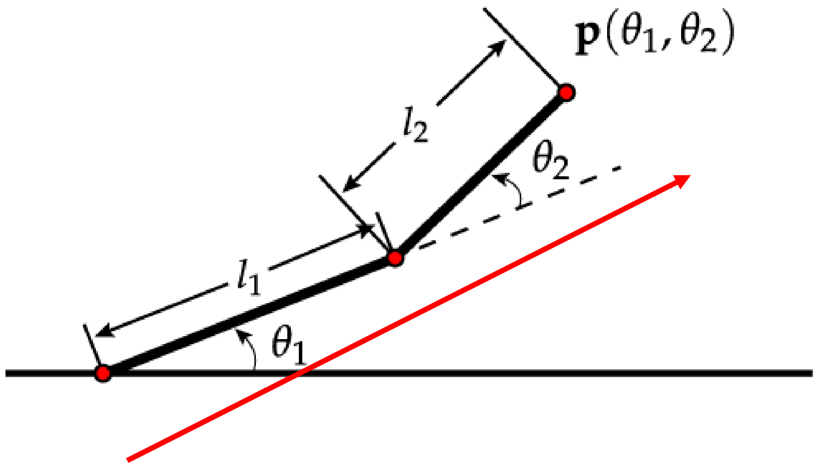
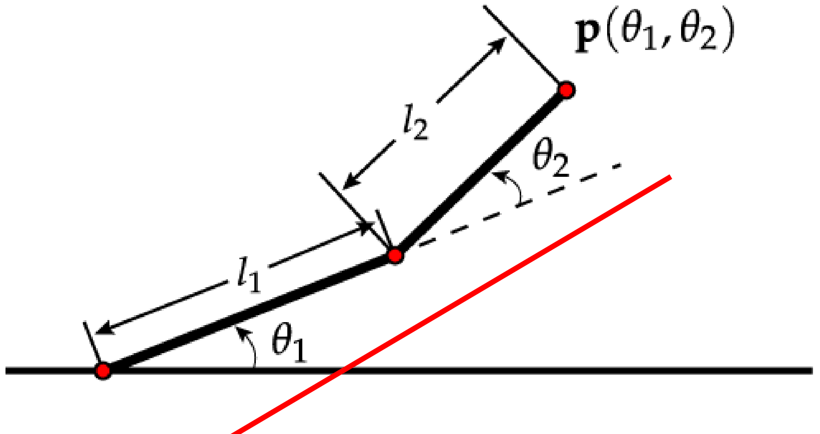
Helge Rhodin

CPSC 427
Video Game Programming

Transformations for Skeleton Kinematics





Setup

@Helge: Pressed record?

@Class: Logged into iClicker cloud?

Today

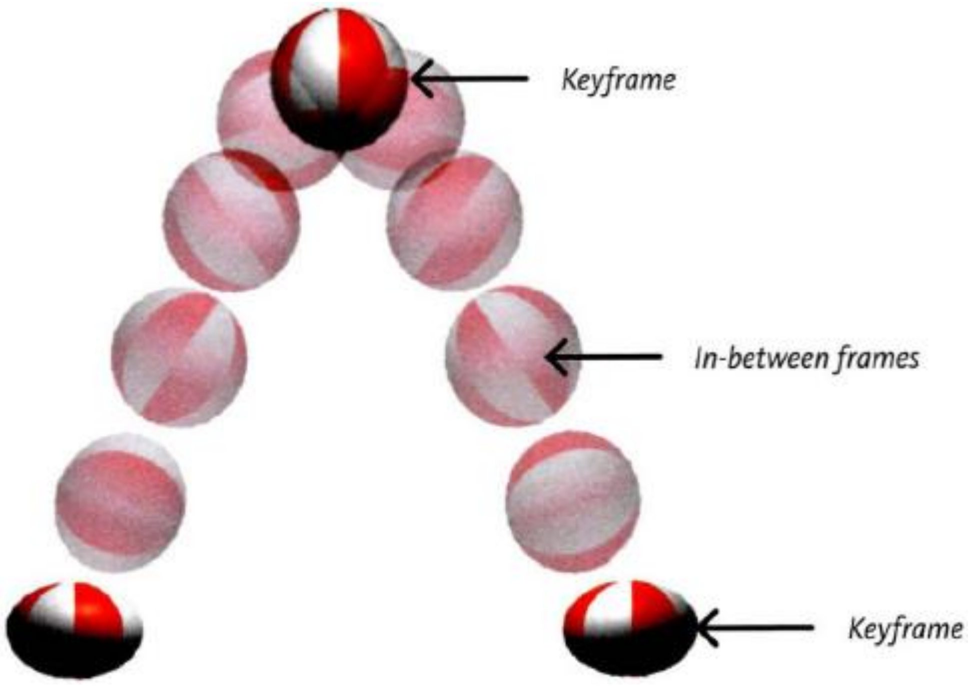
Becoming an expert on transformation

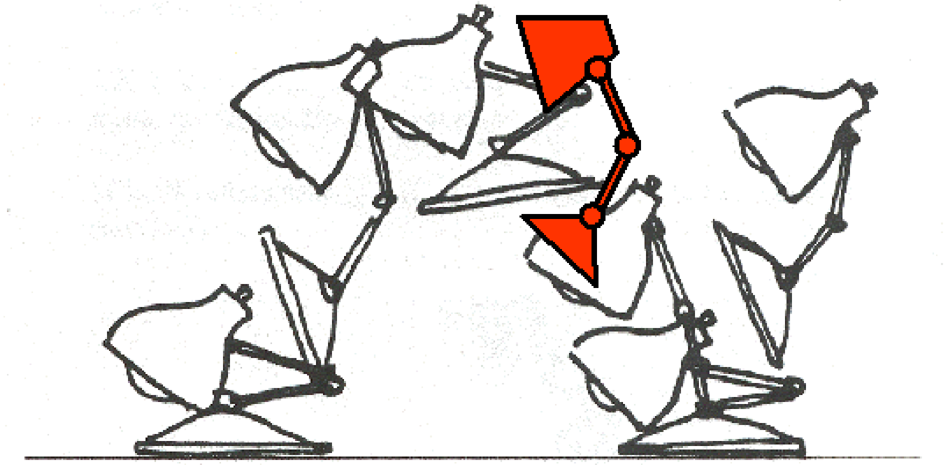
- *Mental picture*
- *Math*
- *Practical examples*

Composite transformations

- *Articulated skeleton motion*
- *Skeleton animation*

Recap: Keyframe animation





Lasseter `87

Recap: Line equation

Parametric form

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

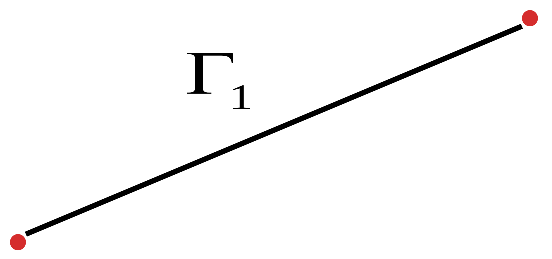
$$C(t) := \begin{pmatrix} P_y^0 \\ P_x^0 \end{pmatrix} t + \begin{pmatrix} P_y^1 \\ P_x^1 \end{pmatrix} (1-t)$$

What things can we interpolate?

Line segment

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

Γ_1



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

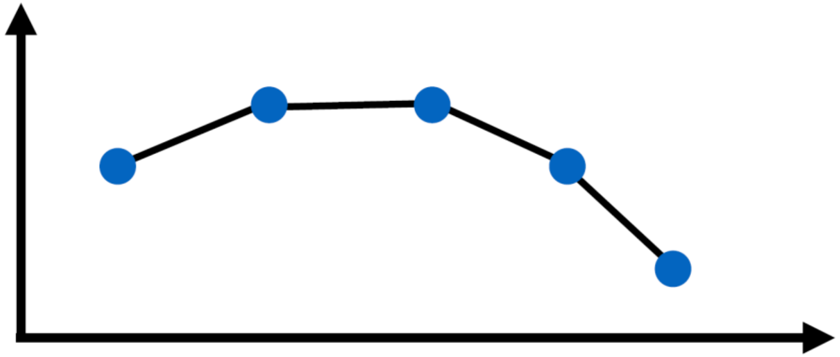
$$G_1 = \begin{cases} x^1(t) = x_0^1 + \left(x_1^1 - x_0^1\right)t \\ y^1(t) = y_0^1 + \left(y_1^1 - y_0^1\right)t \end{cases} t \in [0,1]$$

Recap: Splines

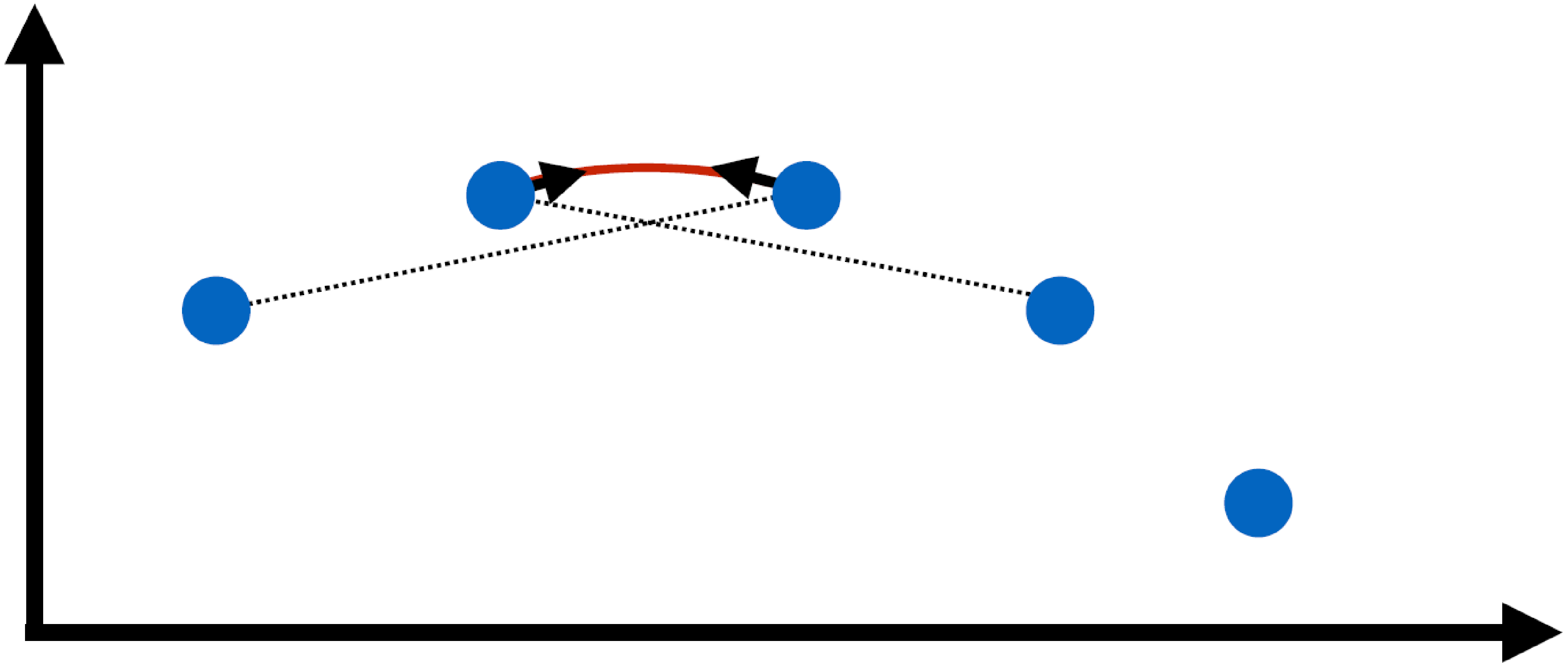
Segments of simple functions

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

E.g., linear functions



Recap: Smooth curve



Recap: 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

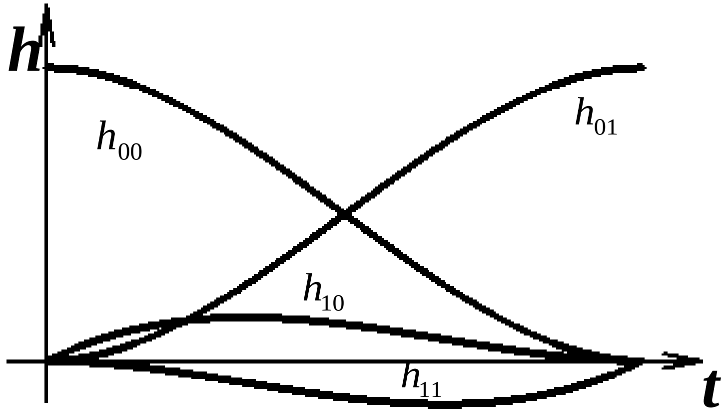
$h_{00}'(0) = h_{00}'(0) = 0$

$h_{00}(0) = 1$

Recap: Hermite Cubic Basis

Four cubic polynomials that satisfy the conditions

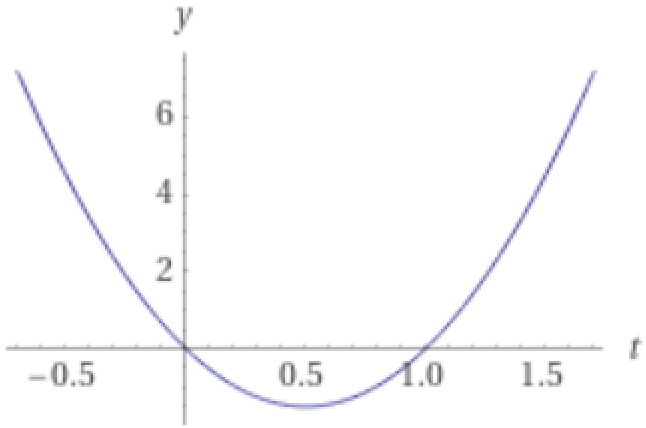
$$h_{00}(t) = t^2(2t-3)+1 \qquad h_{01}(t) = -t^2(2t-3)$$
$$h_{10}(t) = t(t-1)^2 \qquad h_{11}(t) = t^2(t-1)$$



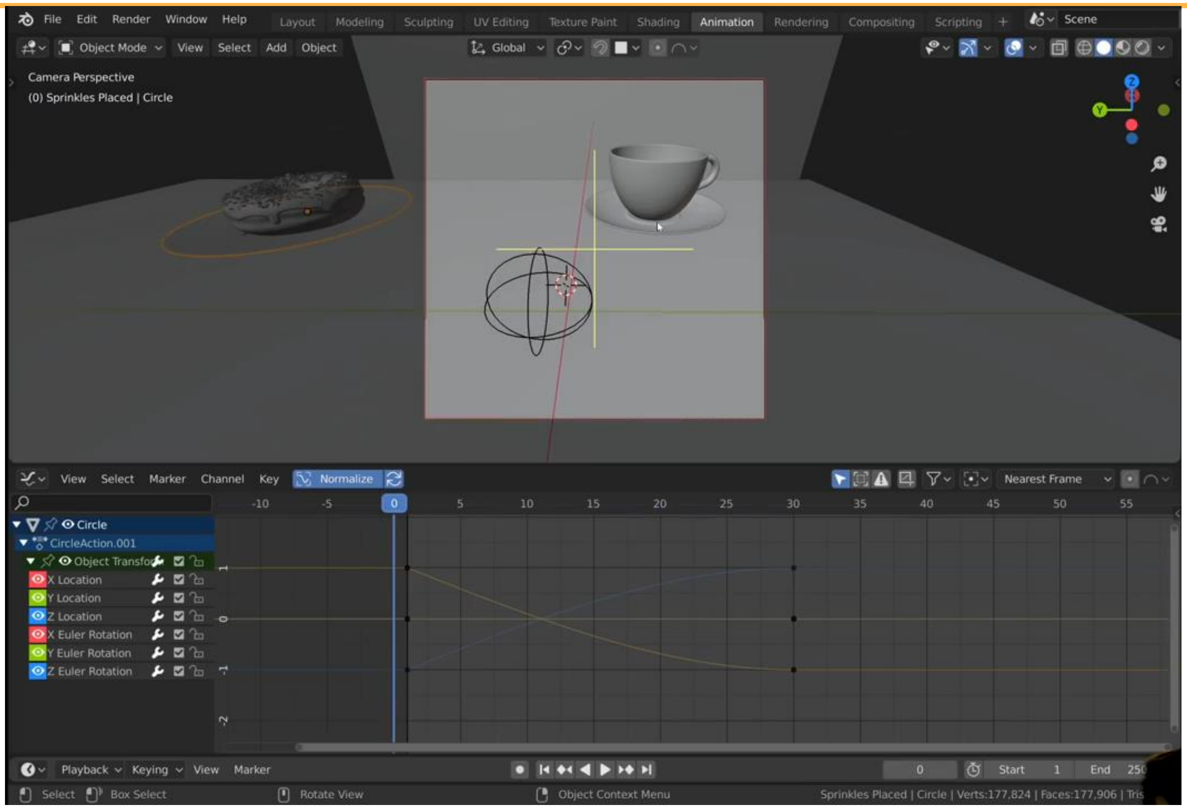
Derivative of h00

$6(-1+t)t$

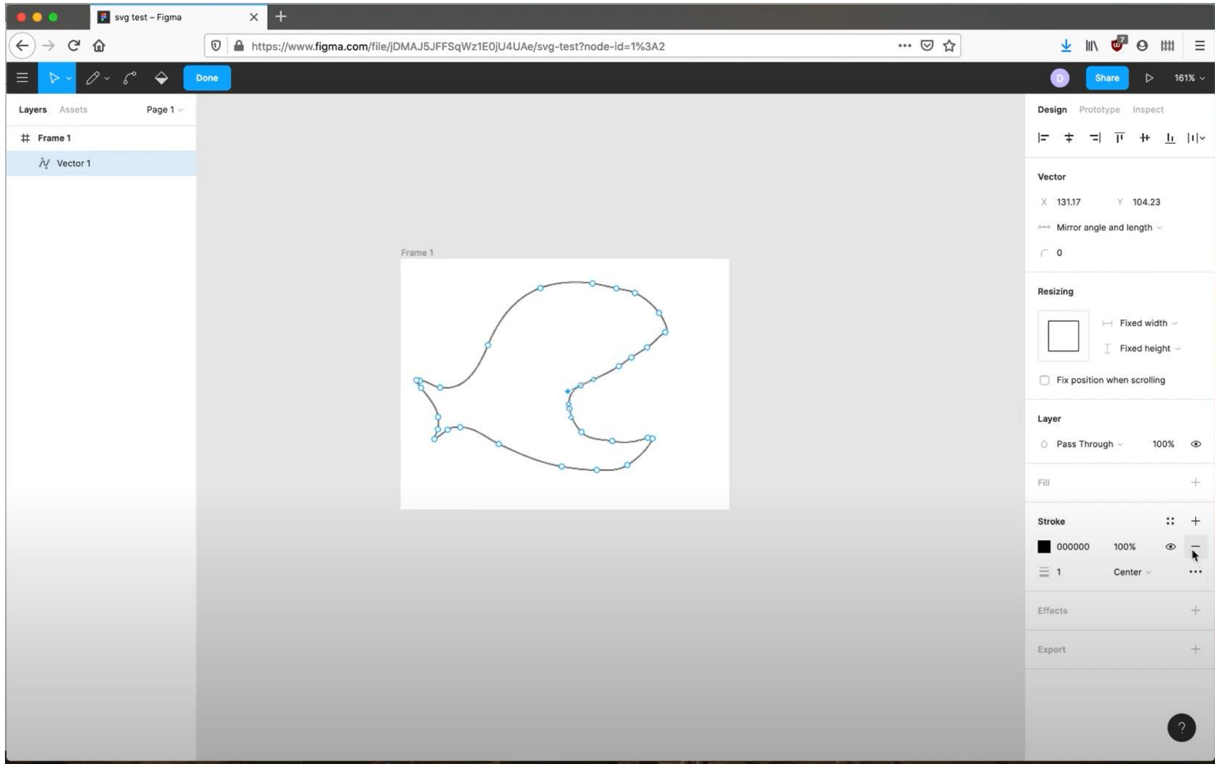
Plots:



Applications:
Keyframe animation & mesh creation



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



Dave’s Tutorial

Today

Becoming an expert on transformation

- *Mental picture*
- *Math*
- *Practical examples*

Composite transformations

- *Articulated skeleton motion*
- *Skeleton animation*

Next days

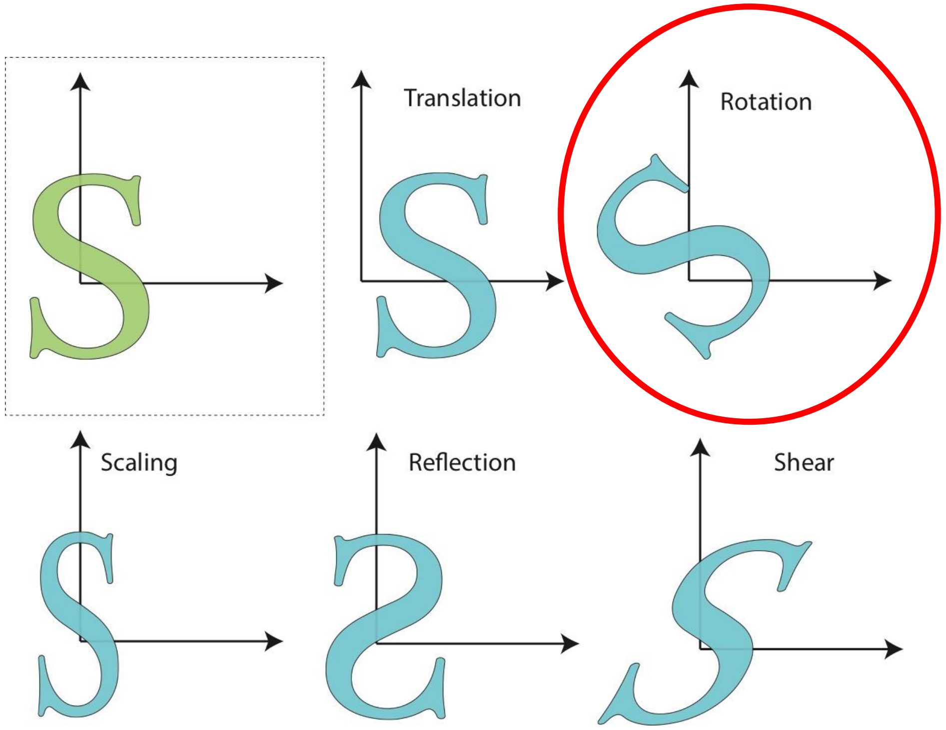
- *Guest lecture on Wednesday*

- *Face-to-face grading on Wednesday (check & sign up!)* *https://docs.google.com/spreadsheets/d/1yKsQJQ04PHF4pPBaN5rGWNO5FJUyxRpRoJM90XKsMPc/edit?usp=sharing*
- *M3 deadline on Fr.*

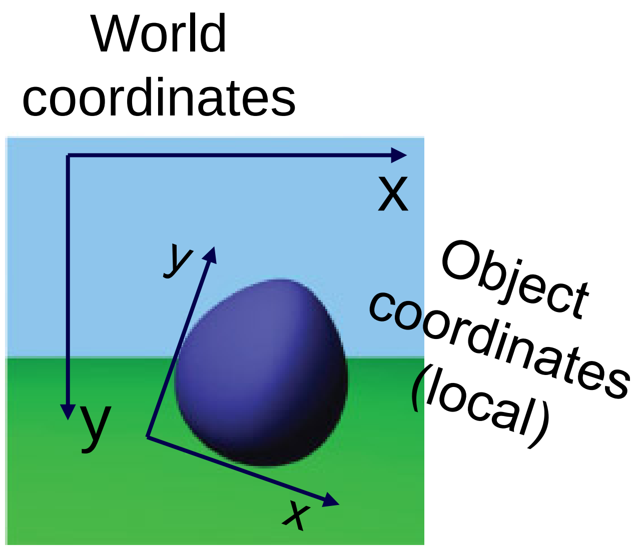
- *No more assignments! ☺*
- *Cross-play M3 on Monday (Cross-play M1 survey posted on piazza)*

Recap: Transformations

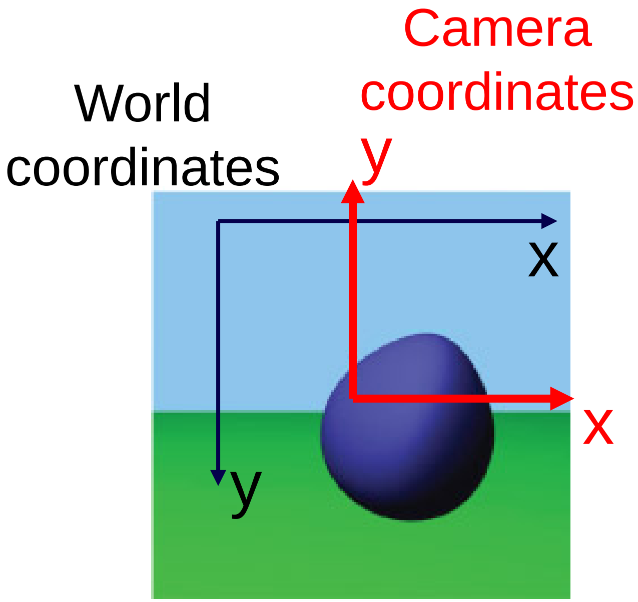
Lecture X?



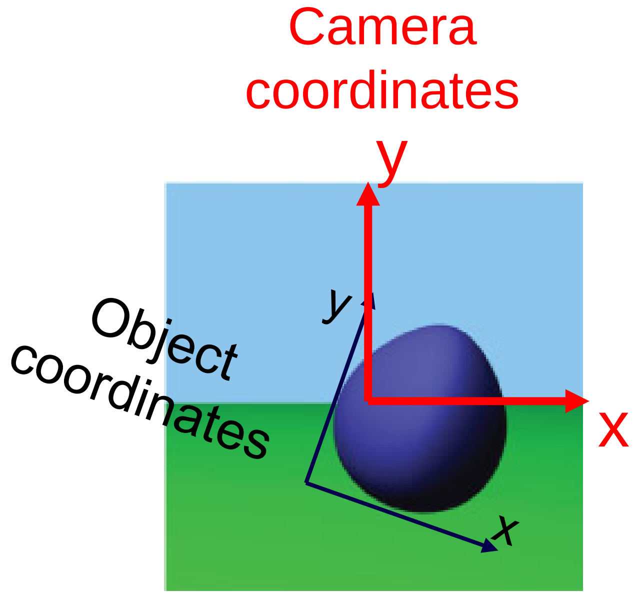
From local object to camera coordinates



object -> world



world -> camera



object -> camera

`transform`

`projection`

`projection * transform`

Recap: GLSL Vertex shader

The OpenGL Shading Language (GLSL)

- Syntax similar to the C programming language
- Build-in vector operations
  - functionality as the GLM library our assignment template uses

x and y coordinates
of a vec2, vec3 or vec4

world
-> camera

object
-> world

```
void main()
{
    // Transforming The Vertex
    vec3 out_pos = projection * transform * vec3(in_pos.xy, 1.0);
    gl_Position = vec4(out_pos.xy, in_pos.z, 1.0);
}
```

vector of 3 (vec3) and 4 (vec4) floats

float
(32 bit)

Affine transformations

- Linear transformations + translations
- Can be expressed as 2x2 matrix + 2 vector
- E.g. scale+ translation:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Modeling Transformation

Adding a third coordinate

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 2 & 0 & t_x \\ 0 & 2 & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Affine transformations are now linear

- one 3x3 matrix can express: 2D rotation, scale, shear, and translation

Forward transformations

- ***Given position, scale, angle***
- ***Compute transformation matrix***
- ***Transform the object***

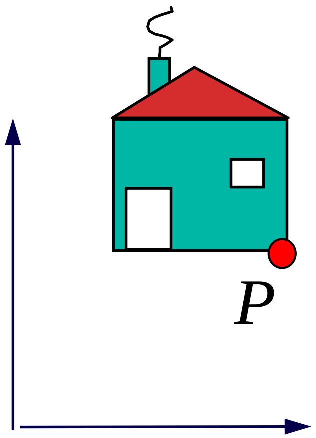
- ***Examples?***

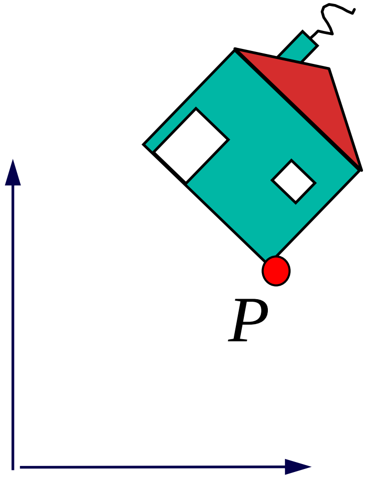
- ***Later: Inverse transformations***

TRANSFORMATION COMPOSITION

- **What operation rotates $\mathbf{XY}$ by θ around**

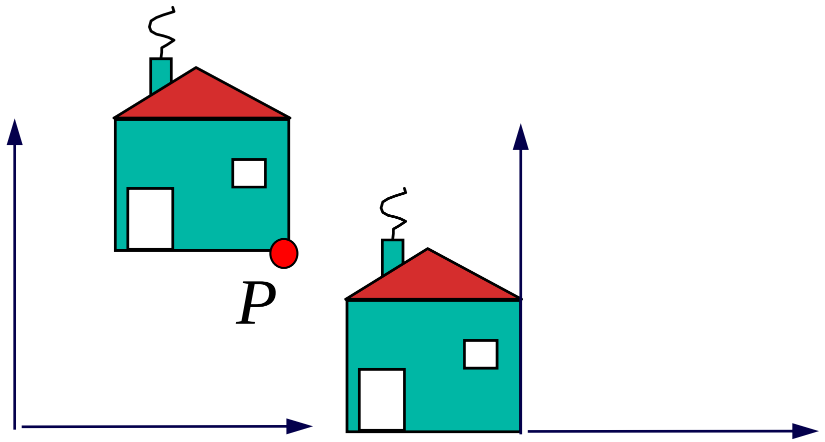
$$P = \begin{pmatrix} p_x \\ p_y \end{pmatrix}\ \textbf{?}$$

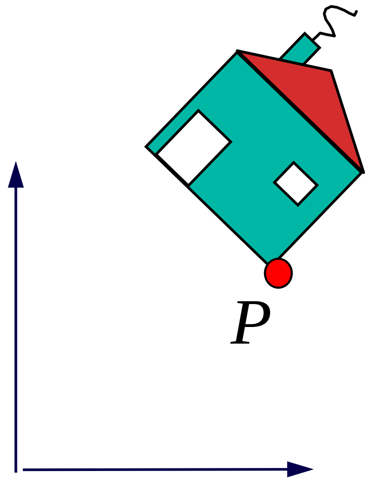




TRANSFORMATION COMPOSITION

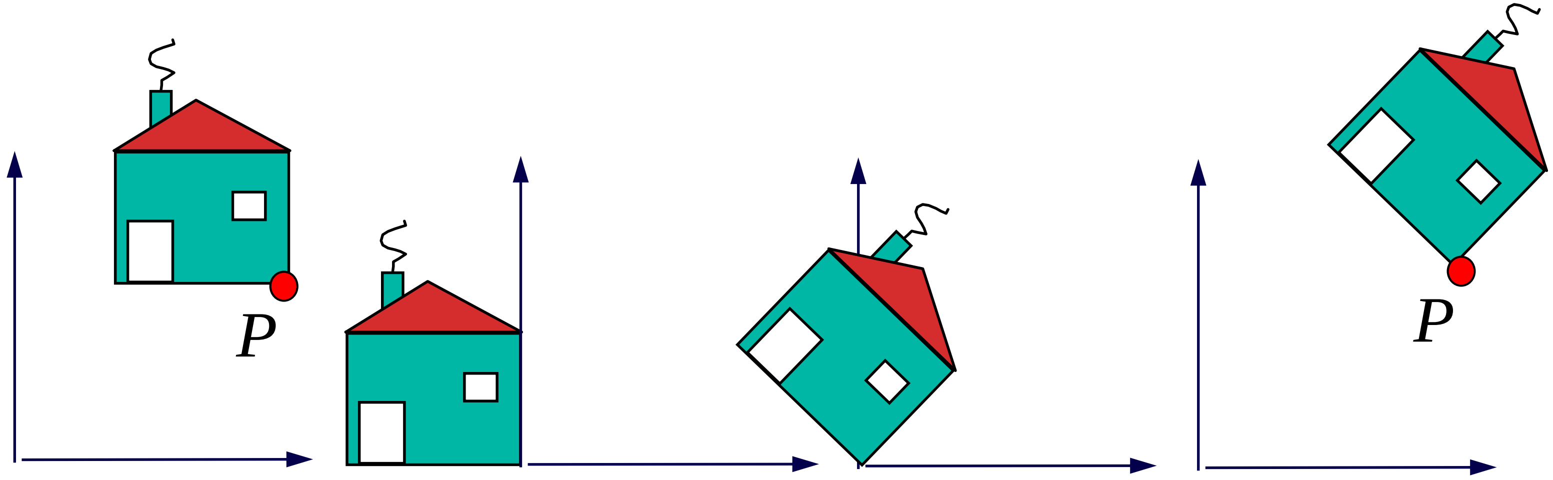
- **What operation rotates XY by θ around** $P = \begin{pmatrix} p_x \\ p_y \end{pmatrix}$
- **Answer:**
  - **Translate *P* to origin**





TRANSFORMATION COMPOSITION

- What operation rotates $\mathbf{XY}$ by $\theta < 0$ around $P = \begin{pmatrix} p_x \\ p_y \end{pmatrix}$?
- Answer:
  - Translate P to origin
  - Rotate around origin by θ
  - Translate back



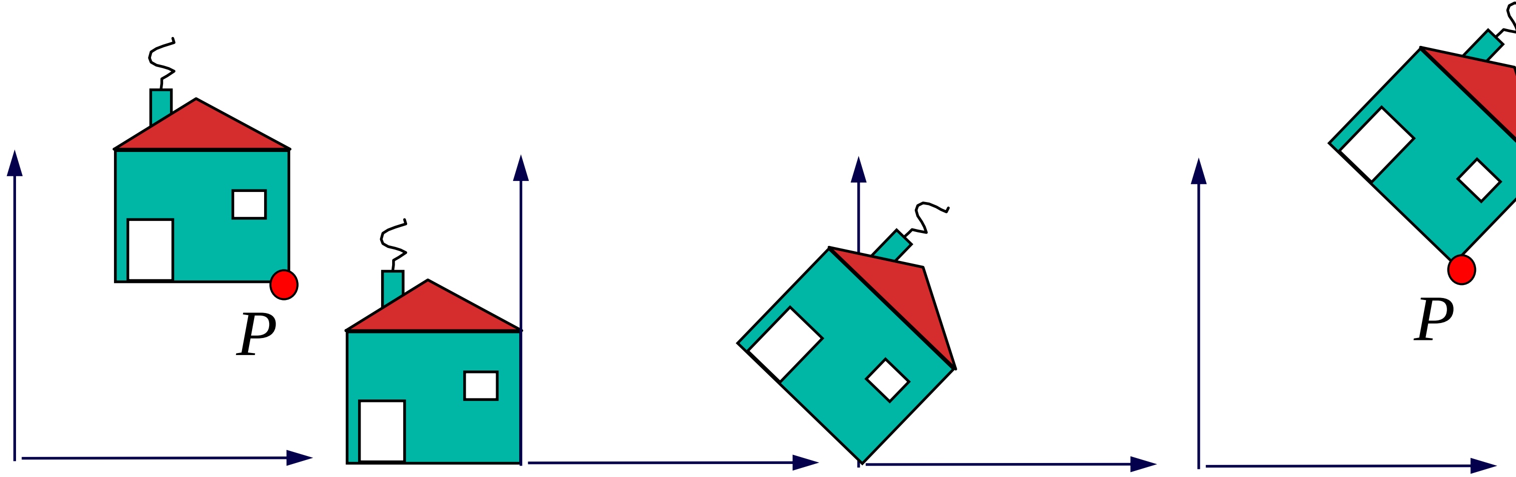
TRANSFORMATION COMPOSITION

$$T^{(p_x,p_y)} R^{\theta} T^{(-p_x,-p_y)}(V)$$

$$= \begin{bmatrix} 1 & 0 & p_x \\ 0 & 1 & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -p_x \\ 0 & 1 & -p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$

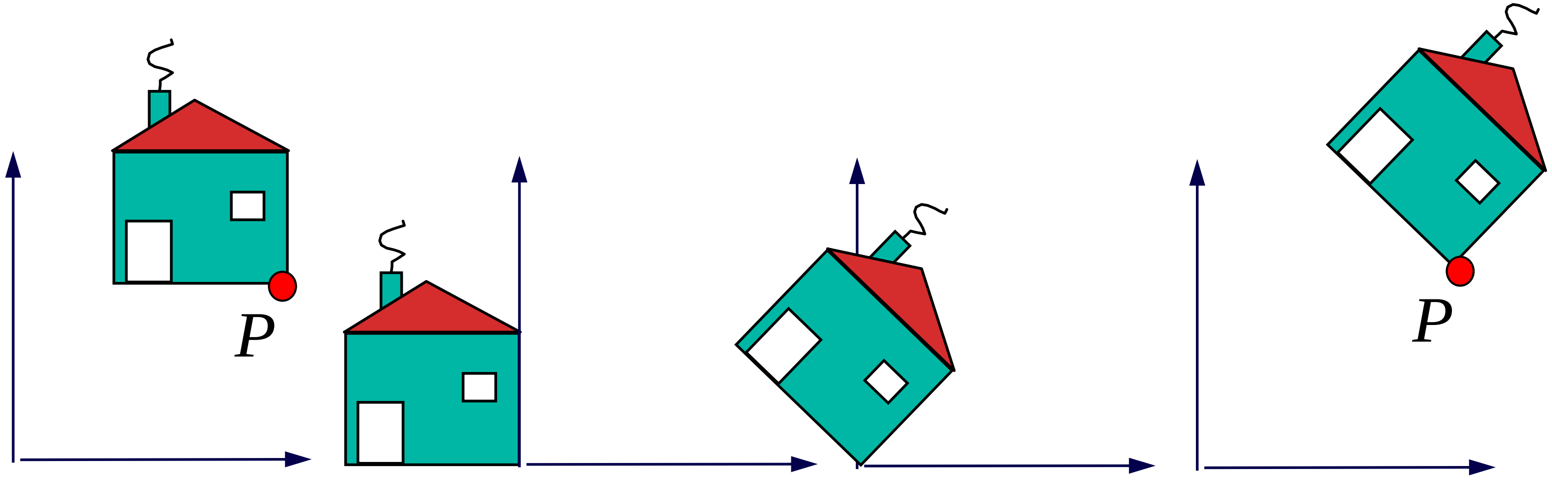
two interpretations

$$\begin{pmatrix} \cos\theta & -\sin\theta & p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ \sin\theta & \cos\theta & p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



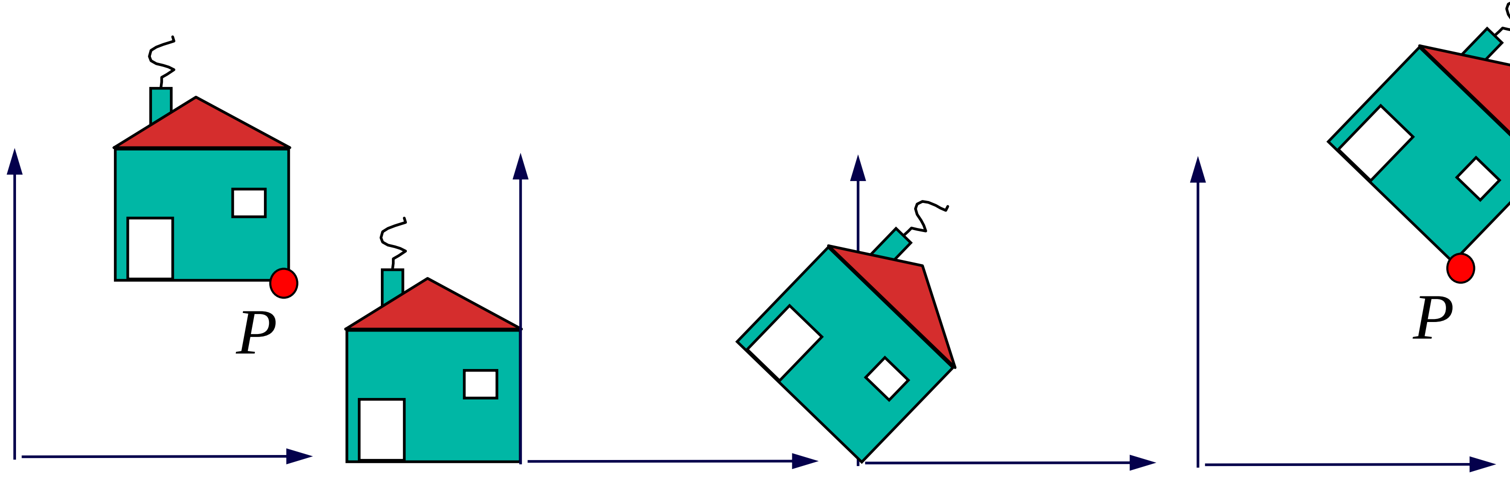
TRANSFORMING COORDINATES

$$\begin{pmatrix} \cos\theta & -\sin\theta & p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ \sin\theta & \cos\theta & p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



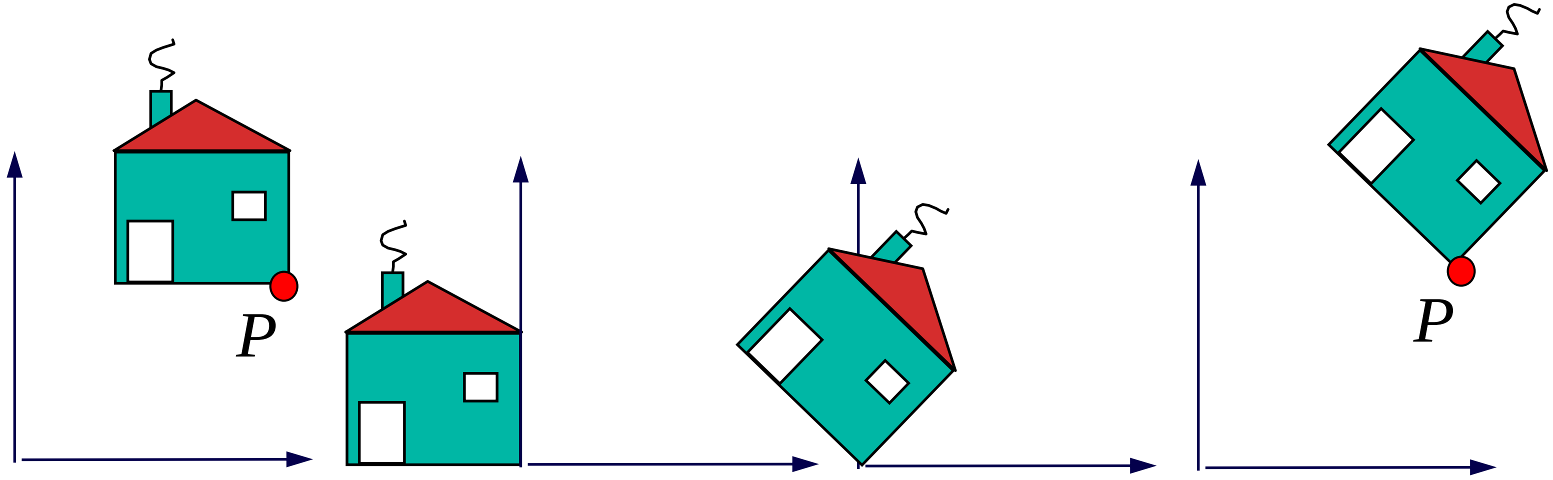
TRANSFORMING COORDINATES

$$\begin{pmatrix} \cos\theta & -\sin\theta & p_x\cdot(1-\cos\theta)+p_y\cdot\sin\theta \\ \boxed{\sin\theta \quad \cos\theta \quad p_y\cdot(1-\cos\theta)+p_x\cdot\sin\theta} \\ 0 & 0 & 1 \end{pmatrix} \boxed{\begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}}$$



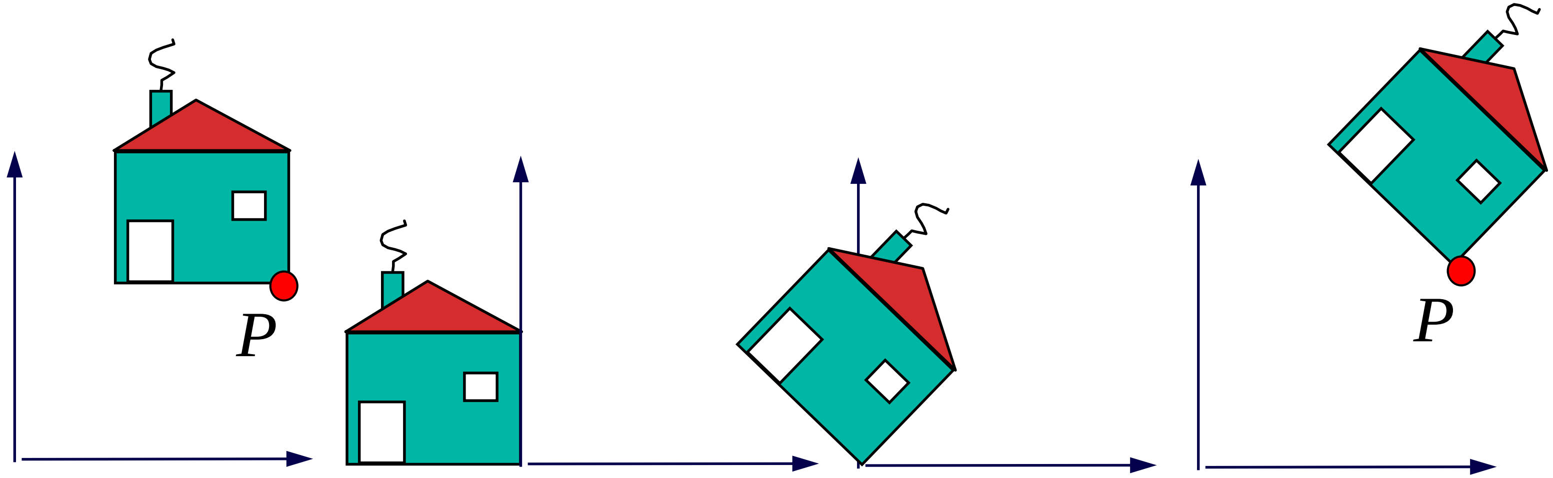
TRANSFORMING COORDINATES

$$\begin{pmatrix} \cos\theta & -\sin\theta & p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ \sin\theta & \cos\theta & p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ \boxed{0 \quad 0 \quad 1} \end{pmatrix} \boxed{\begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}}$$



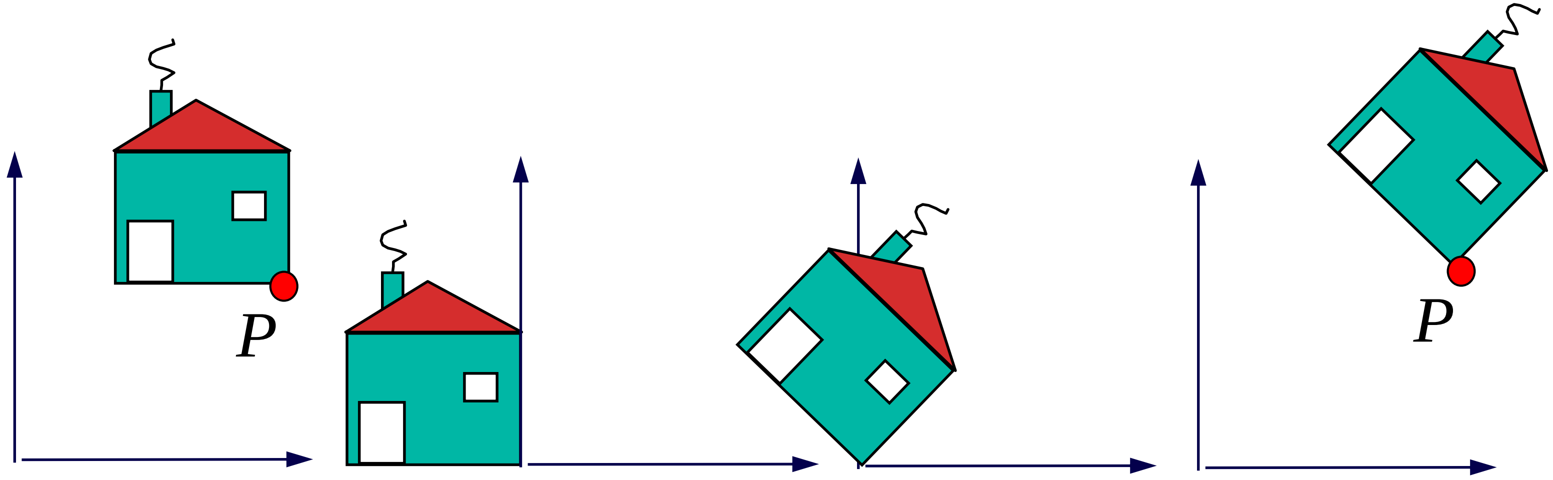
TRANSFORMING COORDINATE FRAME

$$\begin{pmatrix} \boxed{\cos\theta} & -\sin\theta & p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ \boxed{\sin\theta} & \cos\theta & p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ \boxed{0} & 0 & 1 \end{pmatrix} \begin{pmatrix} \boxed{v_x} \\ v_y \\ 1 \end{pmatrix}$$



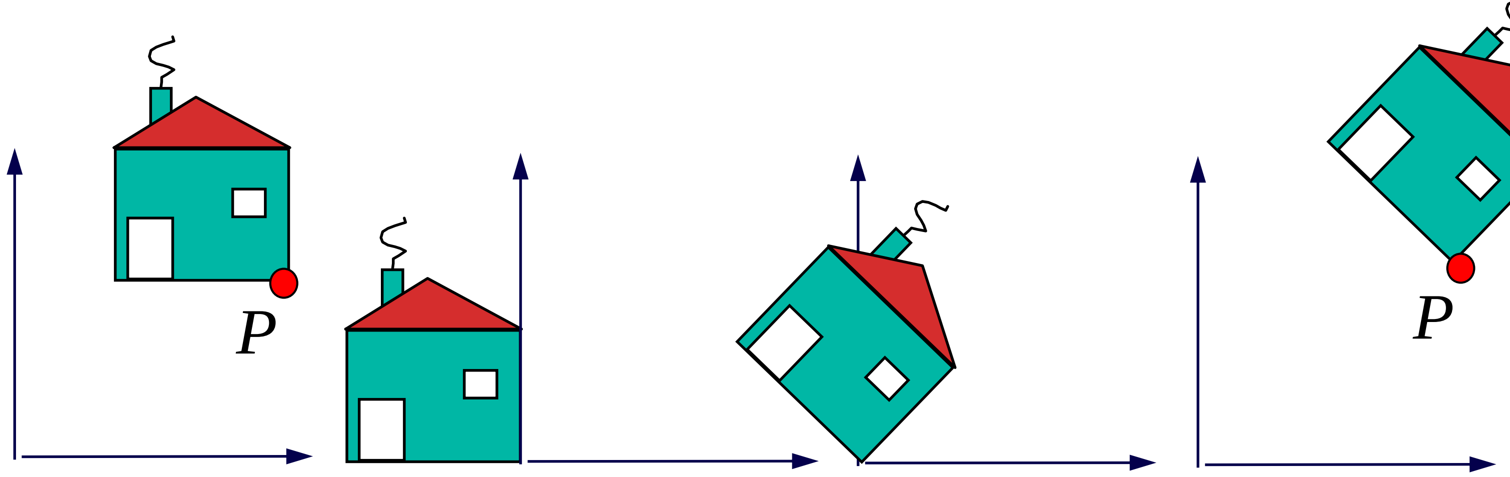
TRANSFORMING COORDINATE FRAME

$$\begin{pmatrix} \cos\theta & \boxed{\begin{matrix} -\sin\theta \\ \cos\theta \\ 0 \end{matrix}} & p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ \sin\theta & & p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ 0 & & 1 \end{pmatrix} \begin{pmatrix} v_x \\ \boxed{v_y} \\ 1 \end{pmatrix}$$



TRANSFORMING COORDINATE FRAME

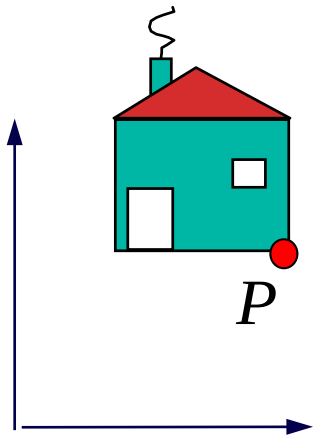
$$\begin{pmatrix} \cos\theta & -\sin\theta & \boxed{\begin{matrix} p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ 1 \end{matrix}} \\ \sin\theta & \cos\theta & \\ 0 & 0 & \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ \boxed{1} \end{pmatrix}$$

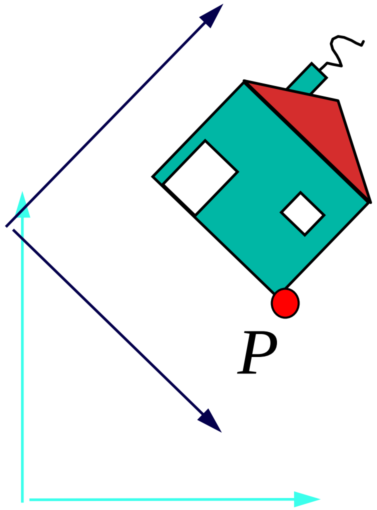


TRANSFORMING COORDINATE FRAME

Columns are new basis vectors (and new origin)!

$$\begin{pmatrix} \cos\theta & -\sin\theta & p_x \cdot (1-\cos\theta) + p_y \cdot \sin\theta \\ \sin\theta & \cos\theta & p_y \cdot (1-\cos\theta) + p_x \cdot \sin\theta \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



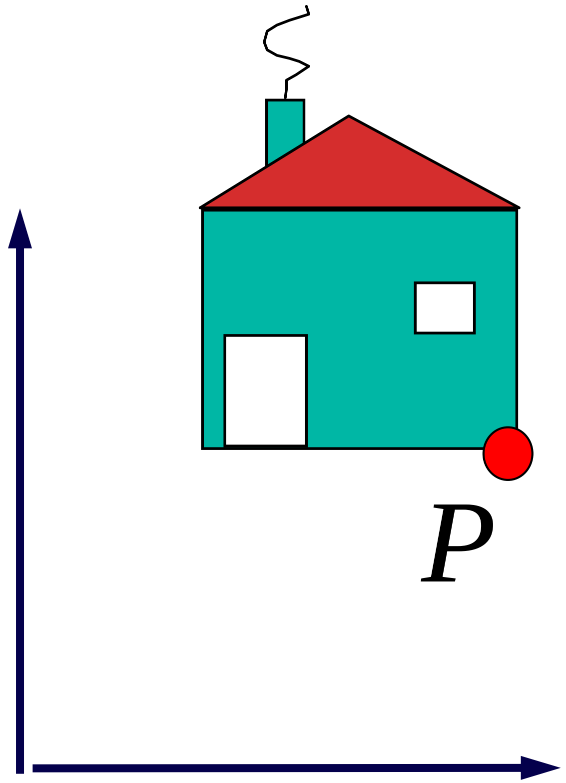


TRANSFORMING COORDINATE FRAME

$$T^{(p_x,p_y)} R^{\theta} T^{(-p_x,-p_y)} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$

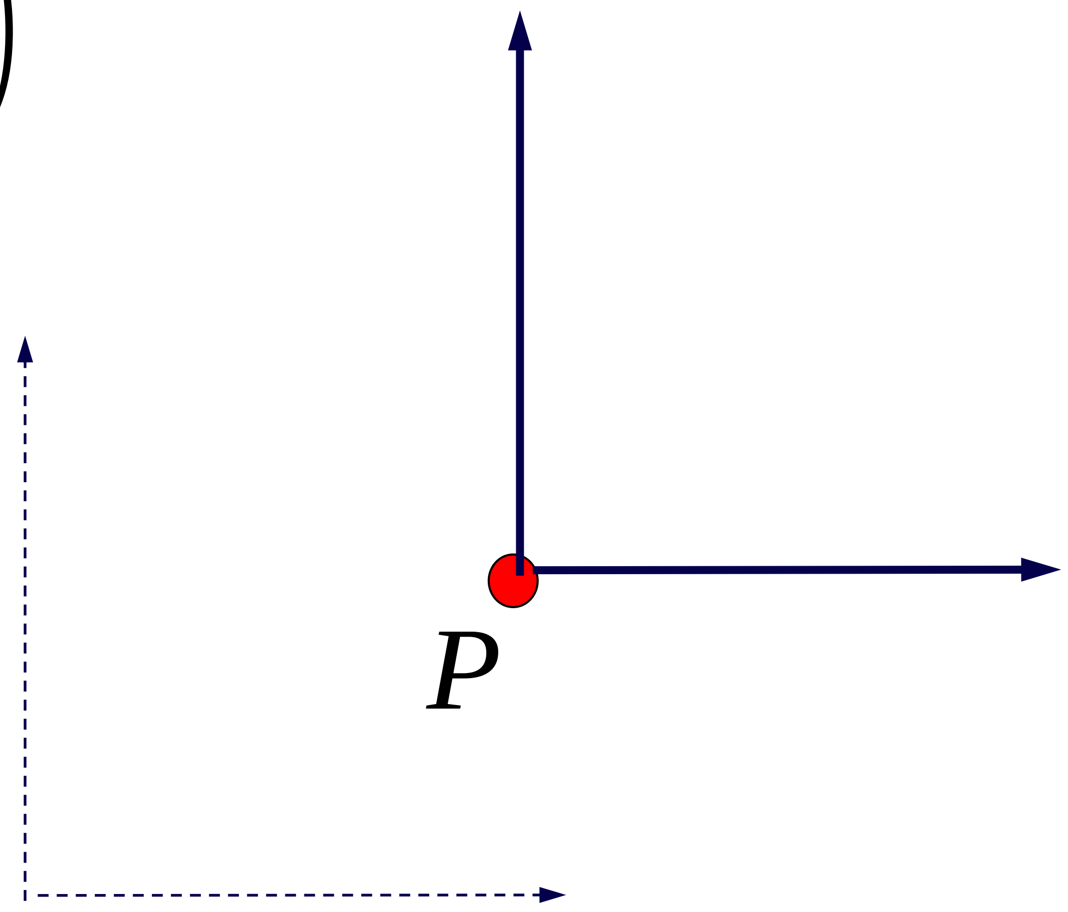
TRANSFORMING COORDINATE FRAME

$$\boldsymbol{T}^{(\boldsymbol{p}_x,\boldsymbol{p}_y)} R^{\theta} T^{(-p_x,-p_y)} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



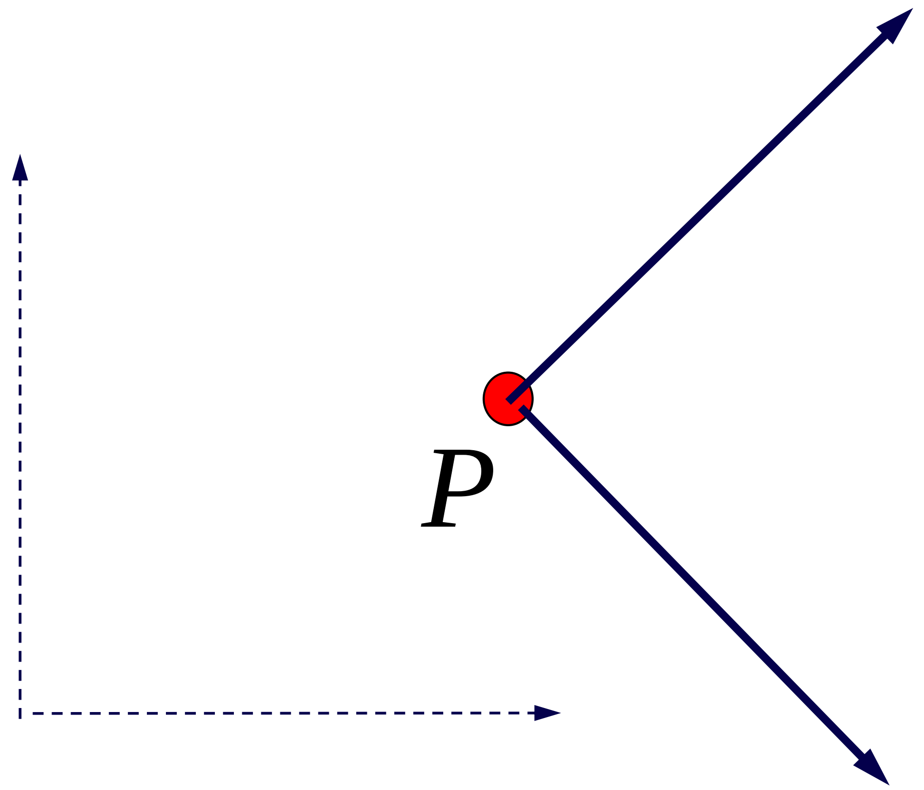
TRANSFORMING COORDINATE FRAME

$$\boldsymbol{T}^{(\boldsymbol{p}_x,\boldsymbol{p}_y)} R^{\theta} T^{(-p_x,-p_y)} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



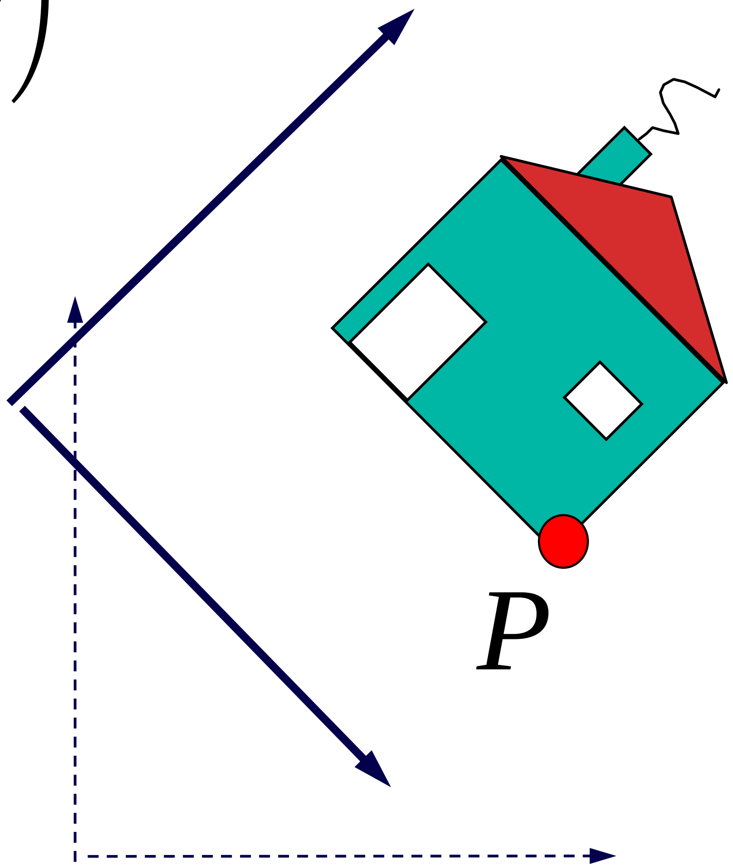
TRANSFORMING COORDINATE FRAME

$$T^{(p_x,p_y)}\boldsymbol{R}^{\boldsymbol{\theta}}T^{(-p_x,-p_y)}\begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



TRANSFORMING COORDINATE FRAME

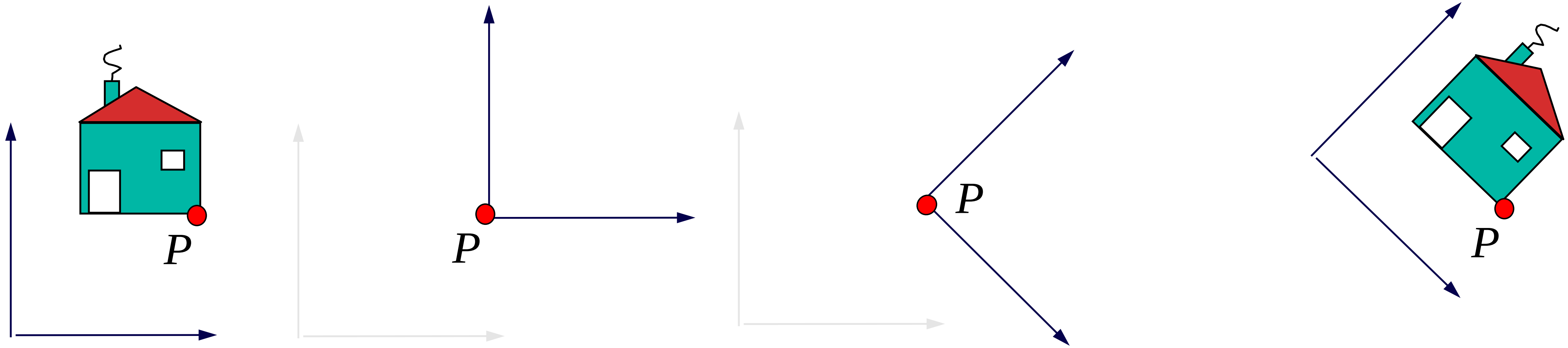
$$T^{(p_x,p_y)} R^{\theta} \boldsymbol{T}^{(-p_x,-p_y)} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



TRANSFORMING COORDINATE FRAME

World Coordinate Frame **Object Coordinate Frame**

$$\begin{pmatrix} v'_x \\ v'_y \\ 1 \end{pmatrix} = T^{(p_x,p_y)} R^{\theta} T^{(-p_x,-p_y)} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$



TWO INTERPRETATIONS OF COMPOSITE

World Coordinate Frame

Object Coordinate Frame

$$\begin{pmatrix} v'_x \\ v'_y \\ 1 \end{pmatrix} = T^{(p_x,p_y)} R^{\theta} T^{(-p_x,-p_y)} \begin{pmatrix} v_x \\ v_y \\ 1 \end{pmatrix}$$

1) read from inside-out as transformation of object

2) read from outside-in as transformation of the coordinate frame

How to go back to angles?

- *Our ECS Motion components store position and angle α*

$$\begin{bmatrix} cos(\alpha) & -sin(\alpha) & p_x \\ sin(\alpha) & cos(\alpha) & p_y \\ 0 & 0 & 1 \end{bmatrix} \xrightarrow{?} \alpha \quad \text{and} \quad \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

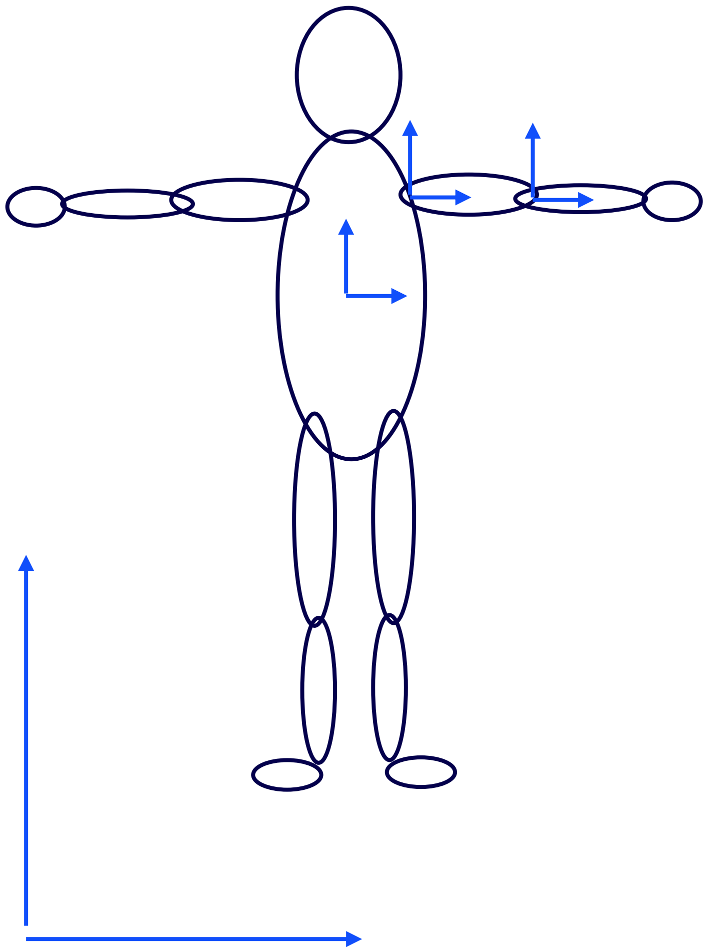
We know α = atan(y,x)

$$\alpha = \text{atan2}(sin(\alpha), cos(\alpha))$$

How to debug?

Transformation Hierarchies

Transformation Hierarchies



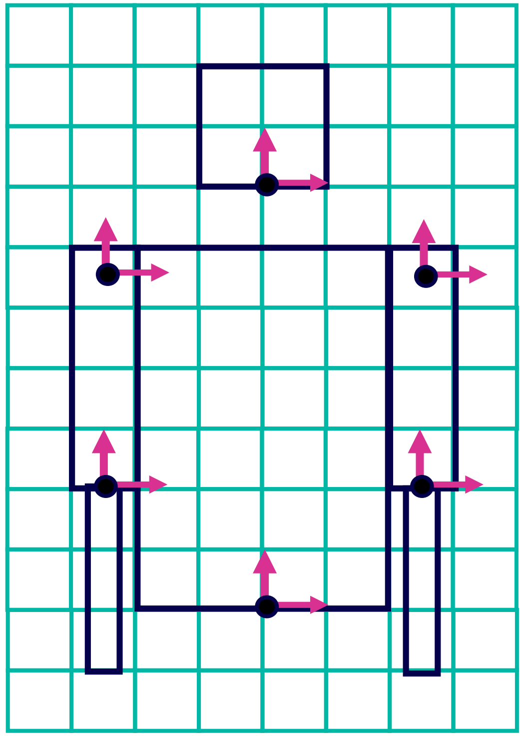
Scenes have multiple coordinate systems

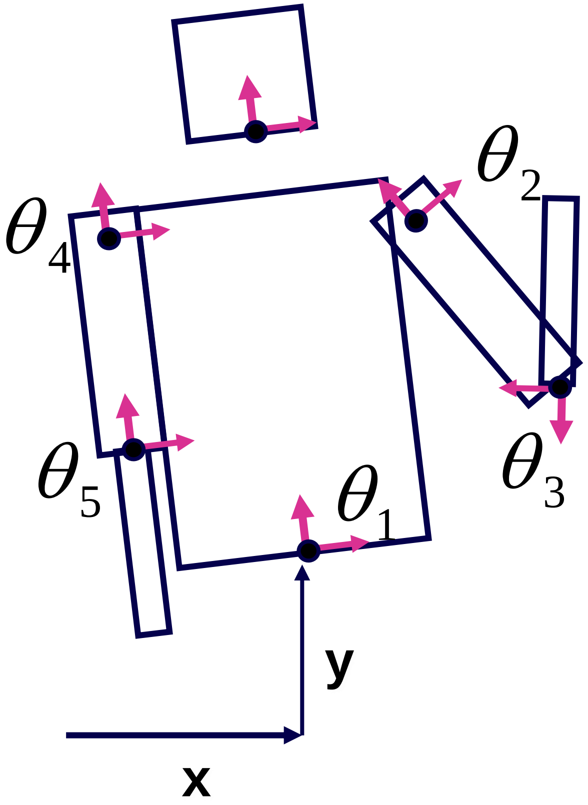
- Often strongly related
  - *Parts of the body*
  - *Object on top of each other*
    - Next to each other…

Independent definition is bug prone

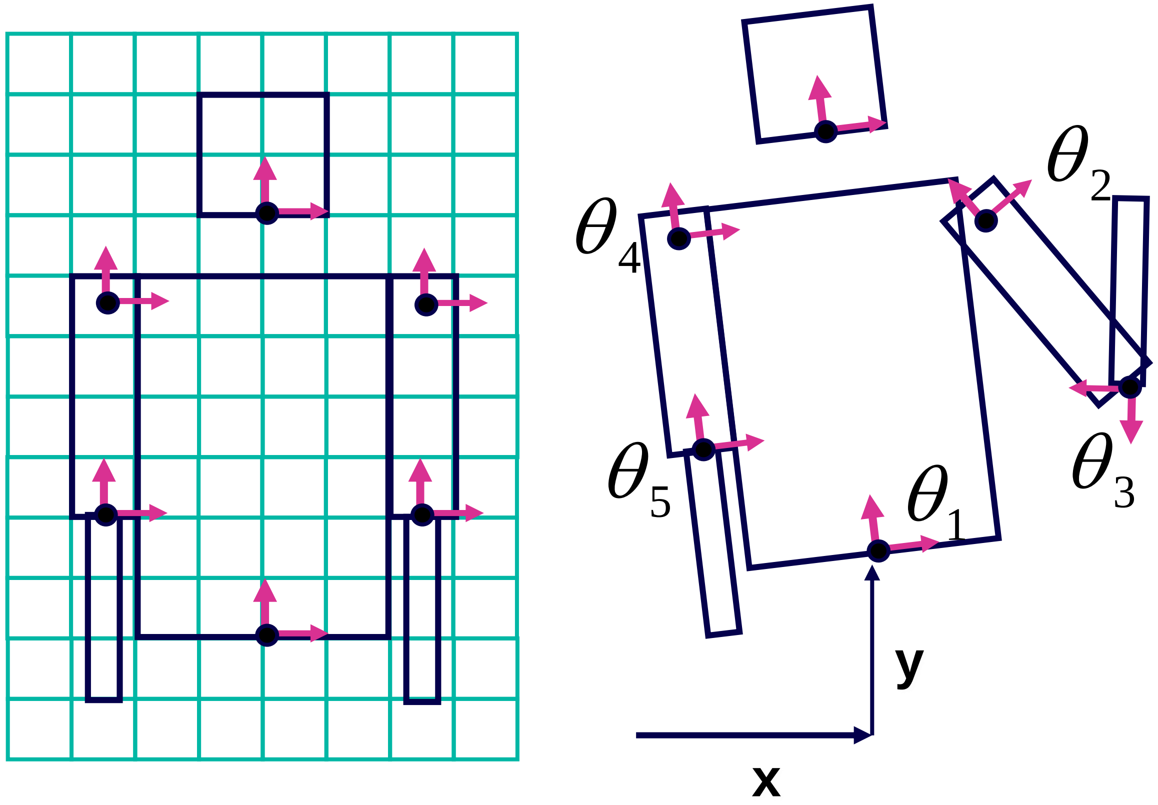
Solution: Transformation Hierarchies

Transformation Hierarchy Examples





Transformation Hierarchy Examples

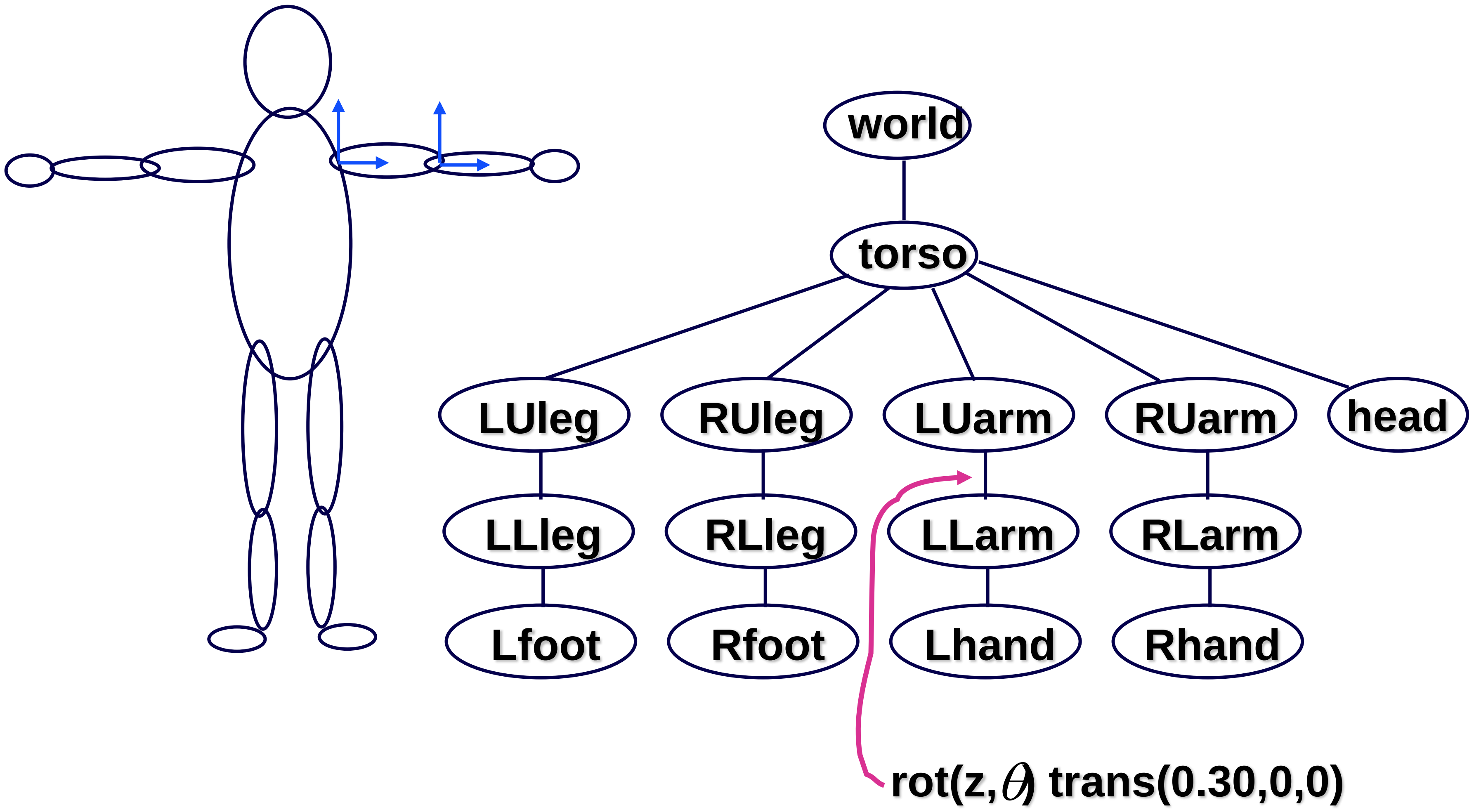


$$M_1 = Tr_{(x,y)} \cdot Rot\theta_1$$

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

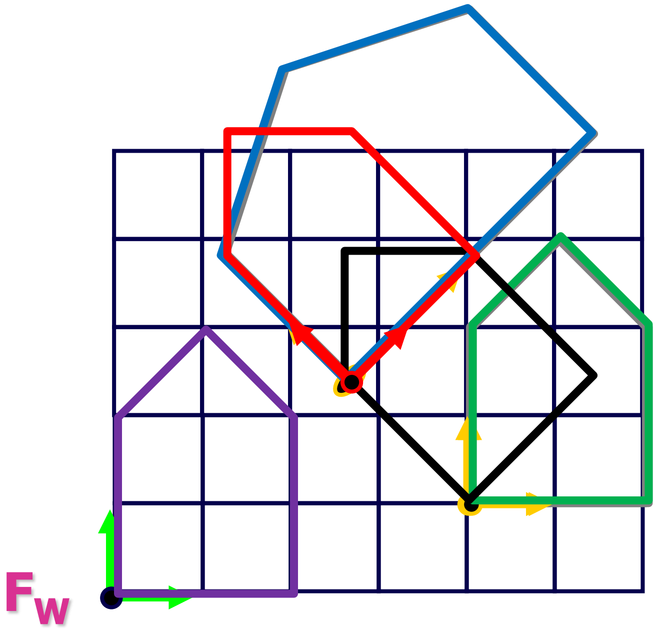
$$M_3 = M_2 \cdot Tr_{(0,-3.5)} \cdot Rot\ \theta_3$$

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?***

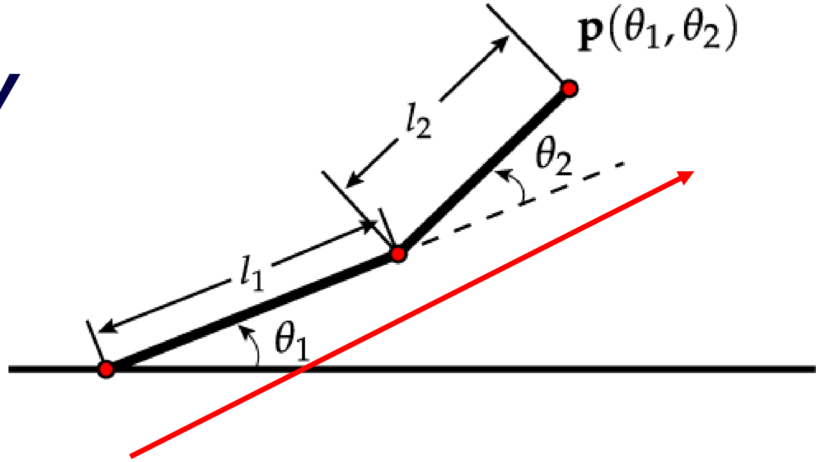
- ***What kind of a problem is this?***
  - linear/non-linear?
  - convex/non-convex?

  - How can we solve it?

Forward vs. inverse kinematics

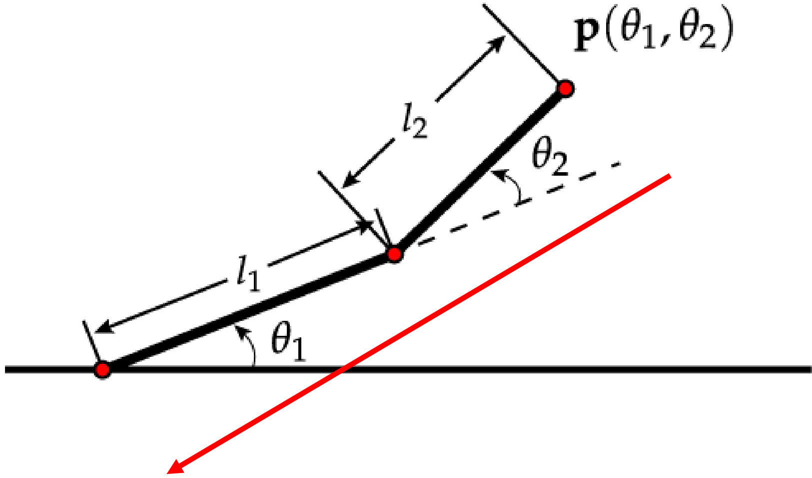
Forward kinematics

- ***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



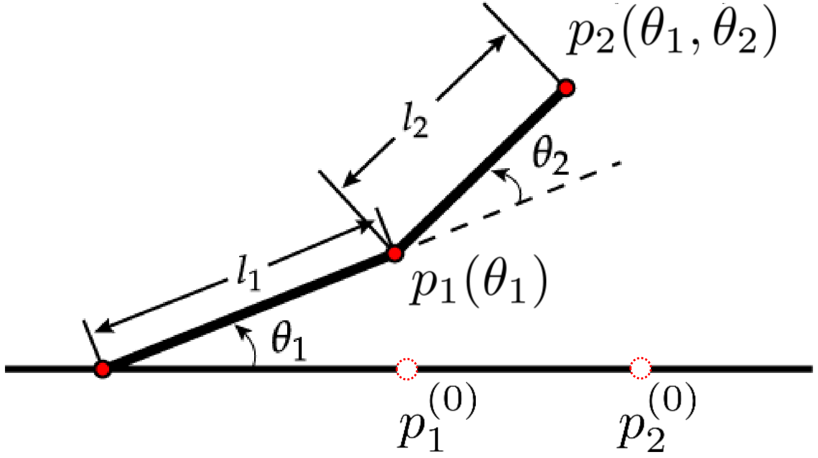
Inverse kinematics (IK)

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

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

- linear/affine given a set of rotation matrices

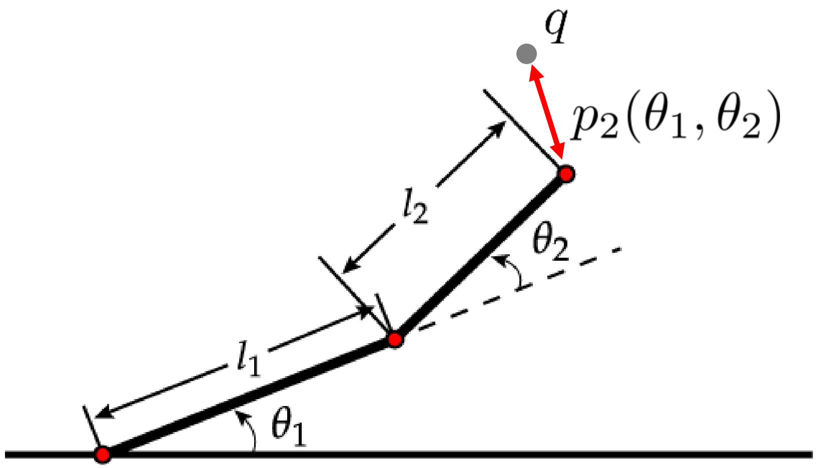
$$p_2(\theta_1, \theta_2) = M_1 M_2 (p_2^{(0)} - p_1^{(0)})$$



Inverse kinematics

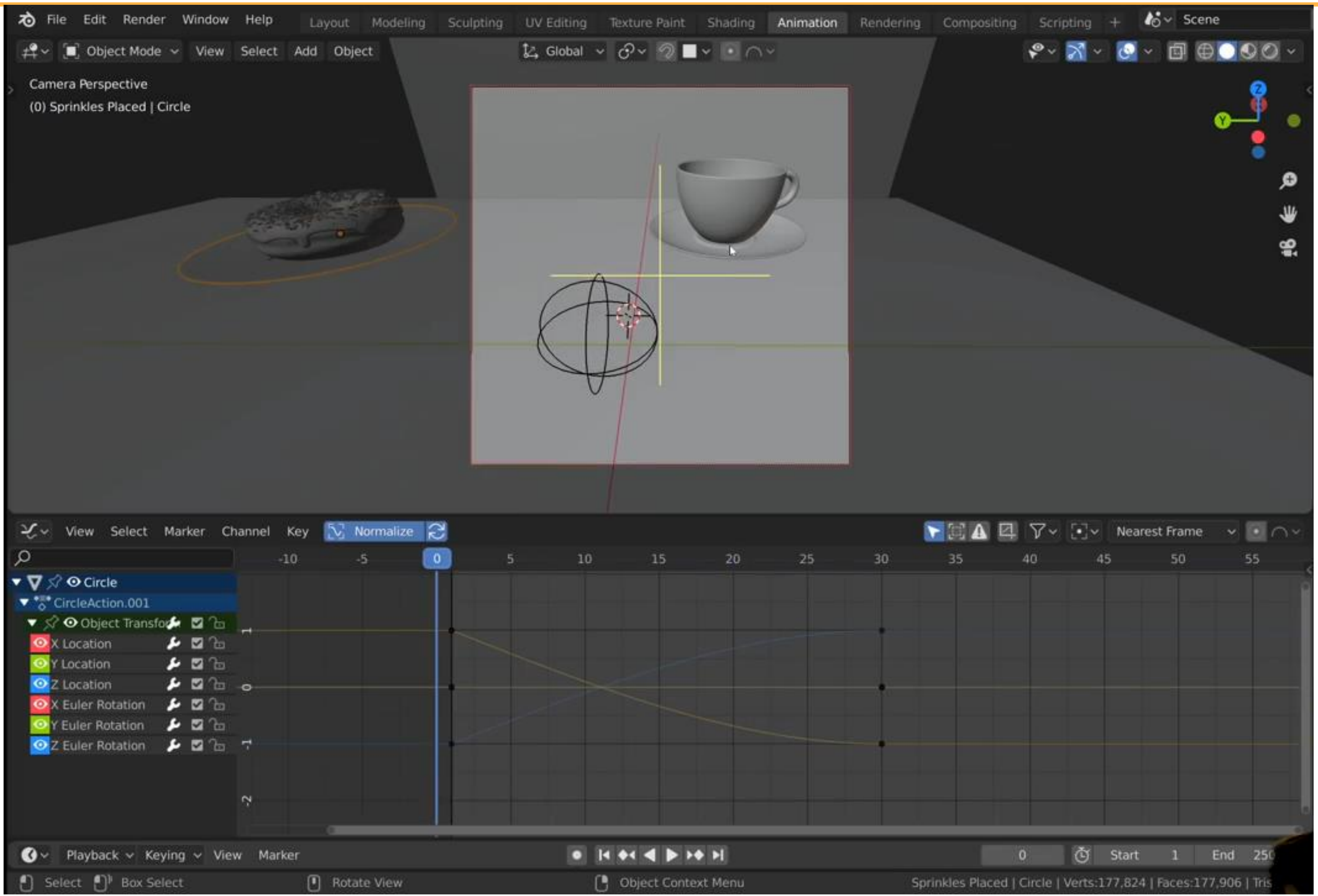
- minimize objective to reach goal location

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



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

Recap:
Keyframe animation & mesh creation



Smooth curve