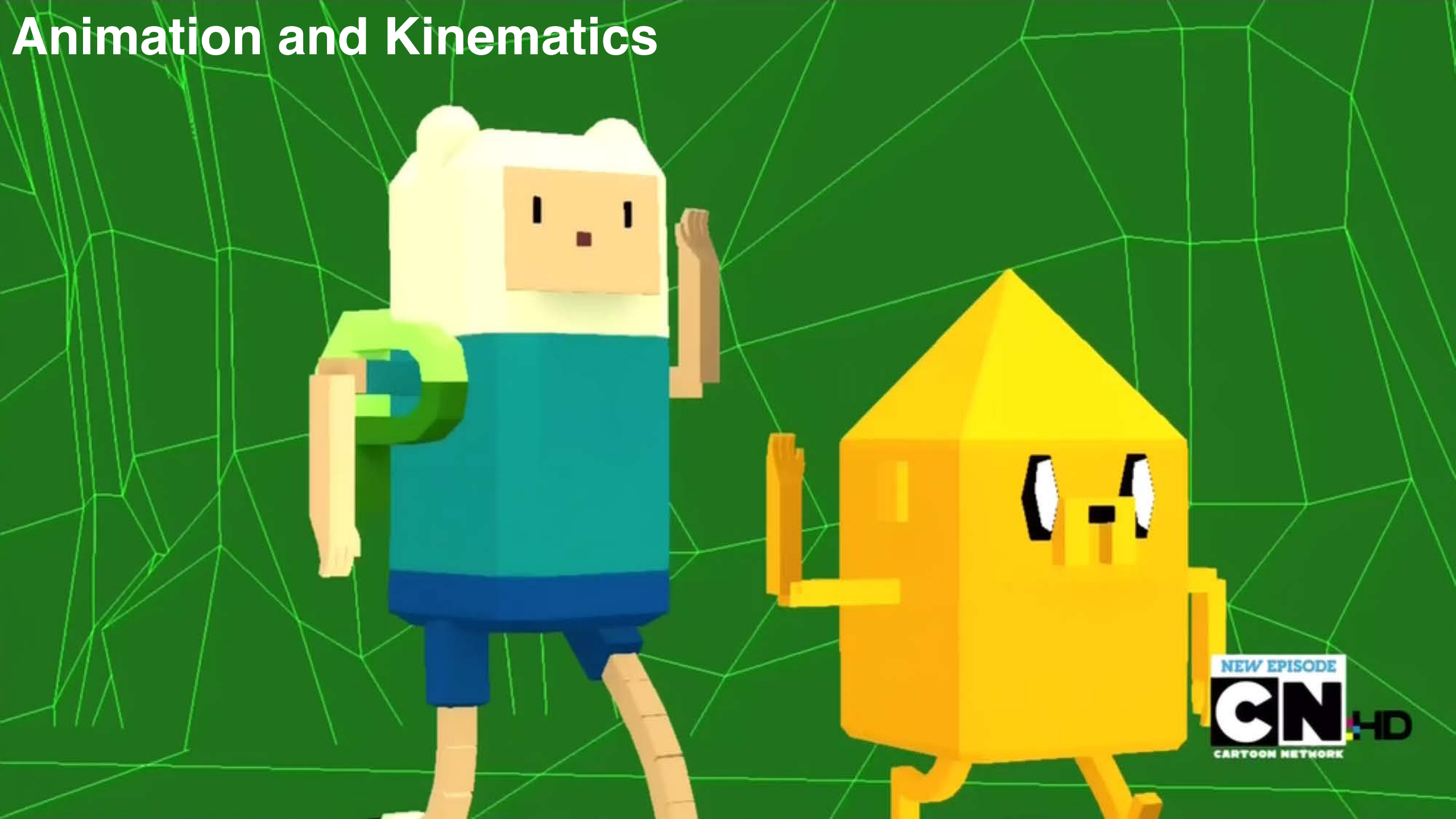
CSC418/2504 Computer Graphics

Some Slides/Images adapted from Marschner and Shirley

Animation and Kinematics



Announcements

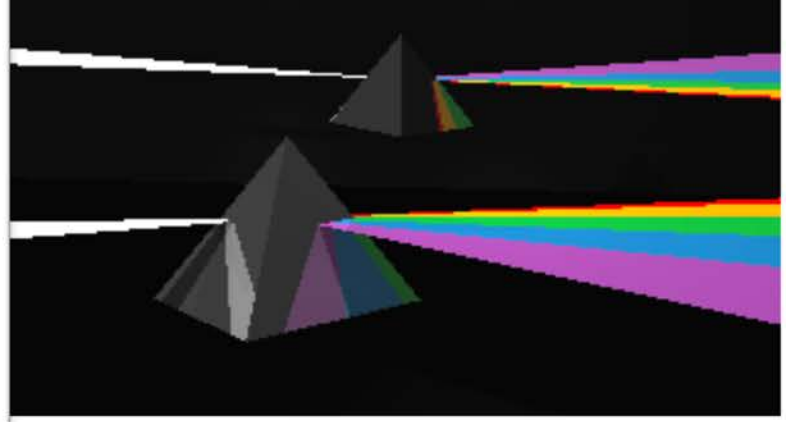
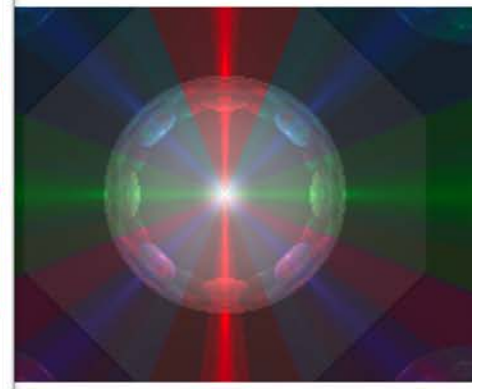
Midterm grades are out (Avg 73%)

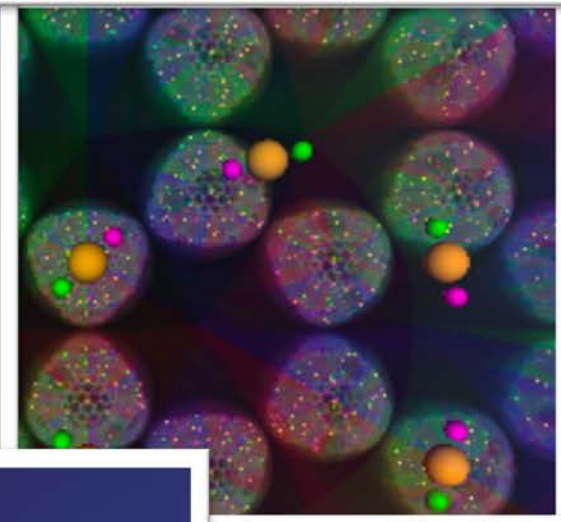
TAs will go over solutions in the tutorial

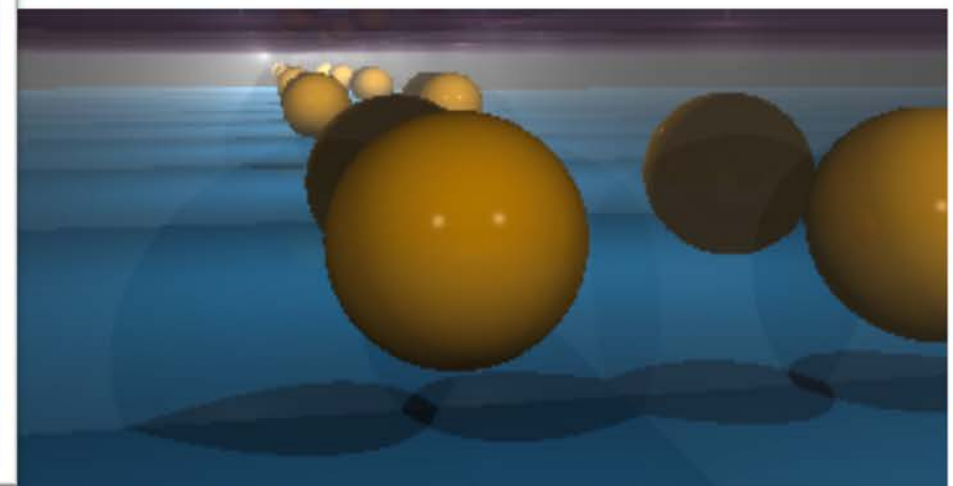
If you want to see your midterm come to office hours

Handle remark requests through MarkUs

Assignment 7 out today, due March 13th



Any Questions ?

The Toronto ACM SIGGRAPH Chapter is once again bringing home the SIGGRAPH 2019 Computer Animation Festival for a screening with the full movie theater experience at The Royal on March 3rd. This year only, admission will be free and open to all!

The Computer Animation Festival consists of some of the best animated shorts and VFX breakdown reels created by students and professionals from around the world, and is selected by an international jury of experts. Since 1999, the Computer Animation Festival has been recognized as a qualifying festival by the Academy of Motion Picture Arts and Sciences, and several works originally presented have received an Academy Award for Best Animated Short Film.

Playing host for the night will be Mark Elendt, a local VFX veteran, Academy Award winner and longtime SIGGRAPH volunteer.

This is a chance for you to see some of the best computer animation from the past year, so come join us on Tuesday March 3rd at 7pm at the Royal theater (608 College St., Toronto, ON, Canada) and enjoy the show!

Get your tickets at https://www.eventbrite.ca/e/siggraph-2019-computer-animation-festival-travelling-show-tickets-95114715675

Animation and Kinematics

Animation in Computer Graphics

Skinning for Mesh Deformation

Forward Kinematics

Keyframe Animation

Inverse Kinematics

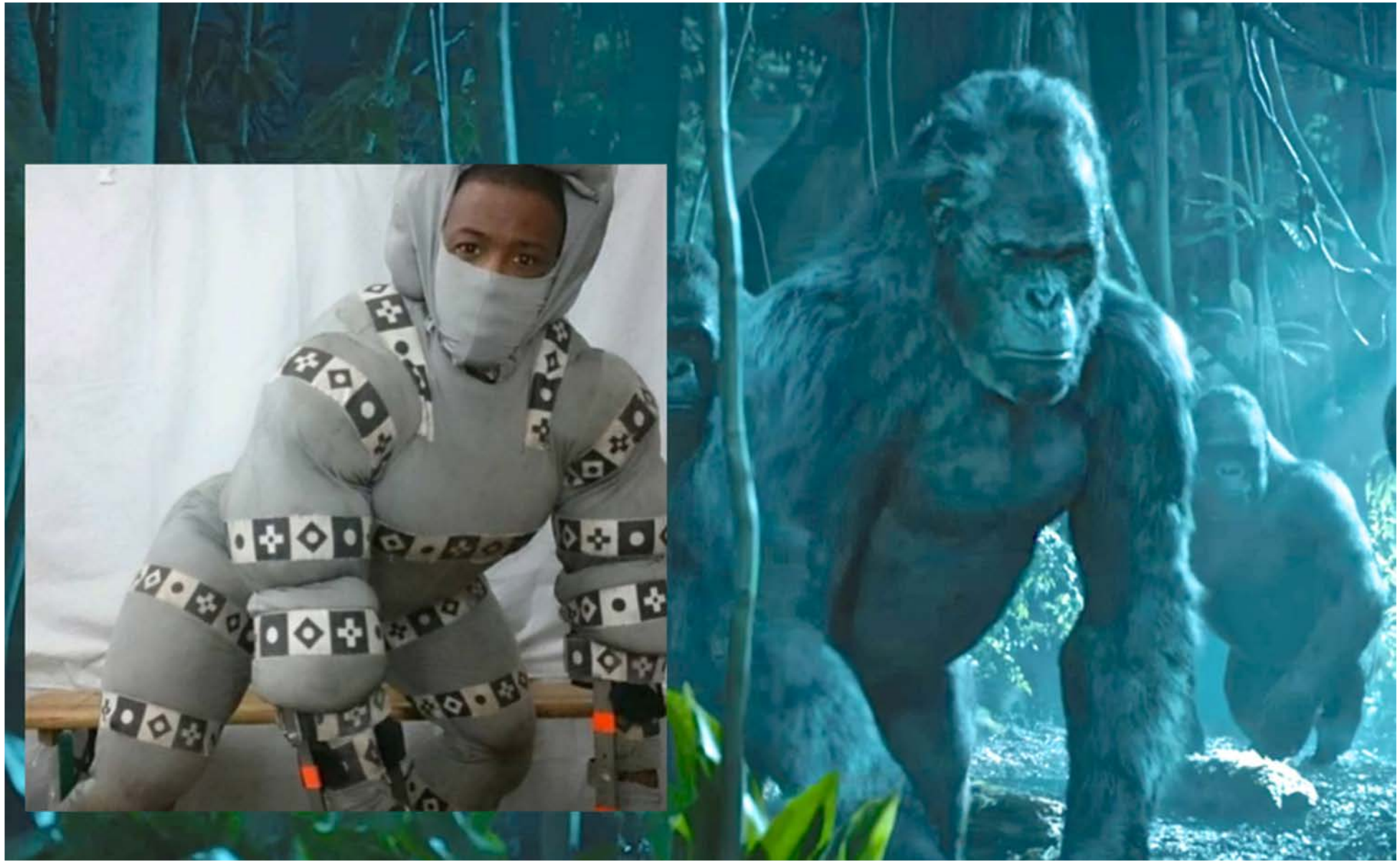
“Core” Areas of Computer Graphics

Modeling

Rendering

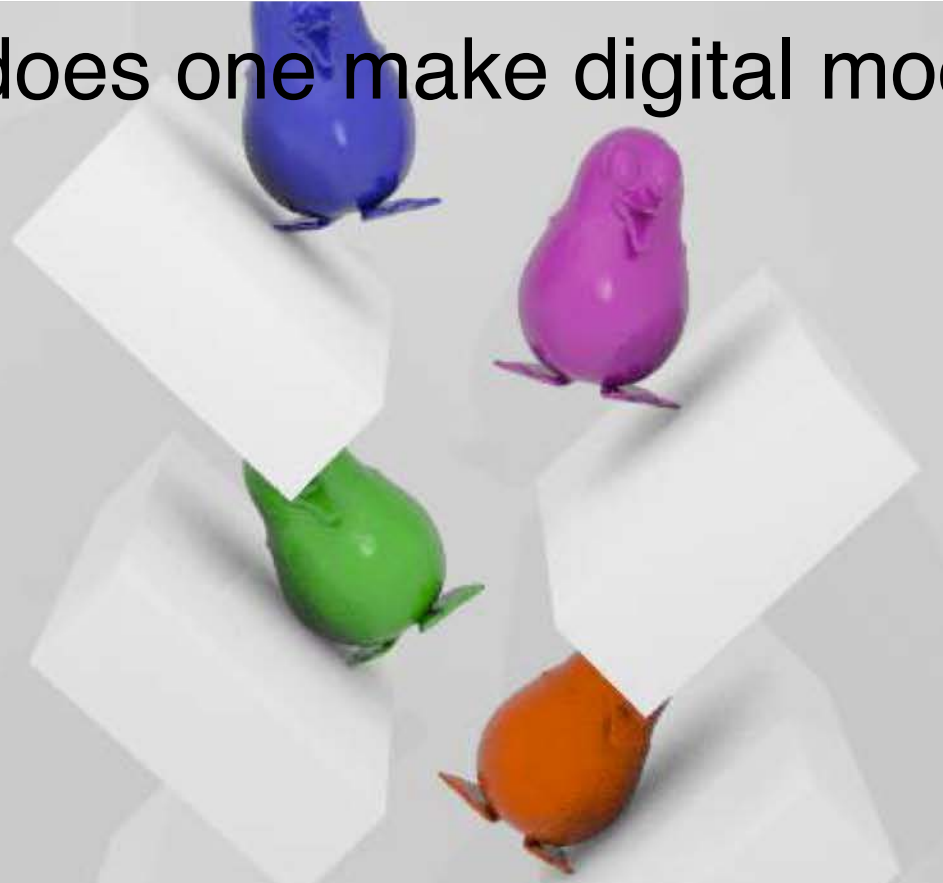
Animation

Animation How does one make digital models move?



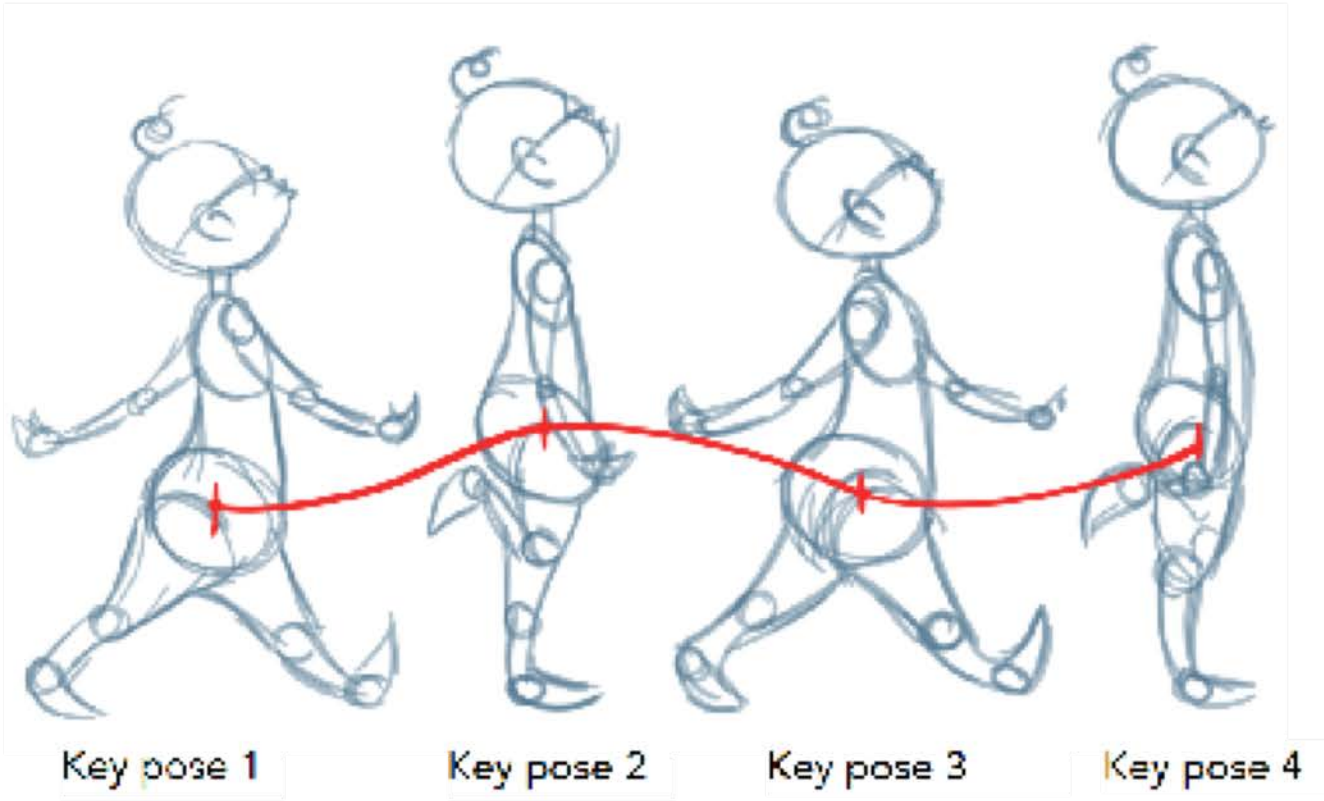
Motion Capture

Animation How does one make digital models move?



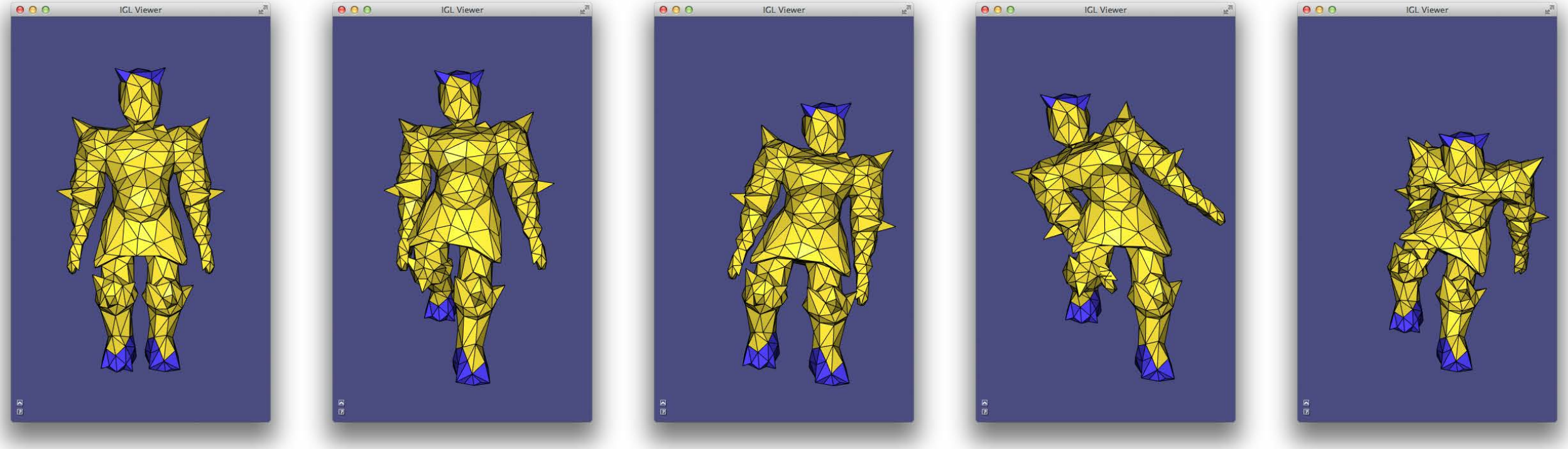
Physics Simulation

Animation How does one make digital models move?



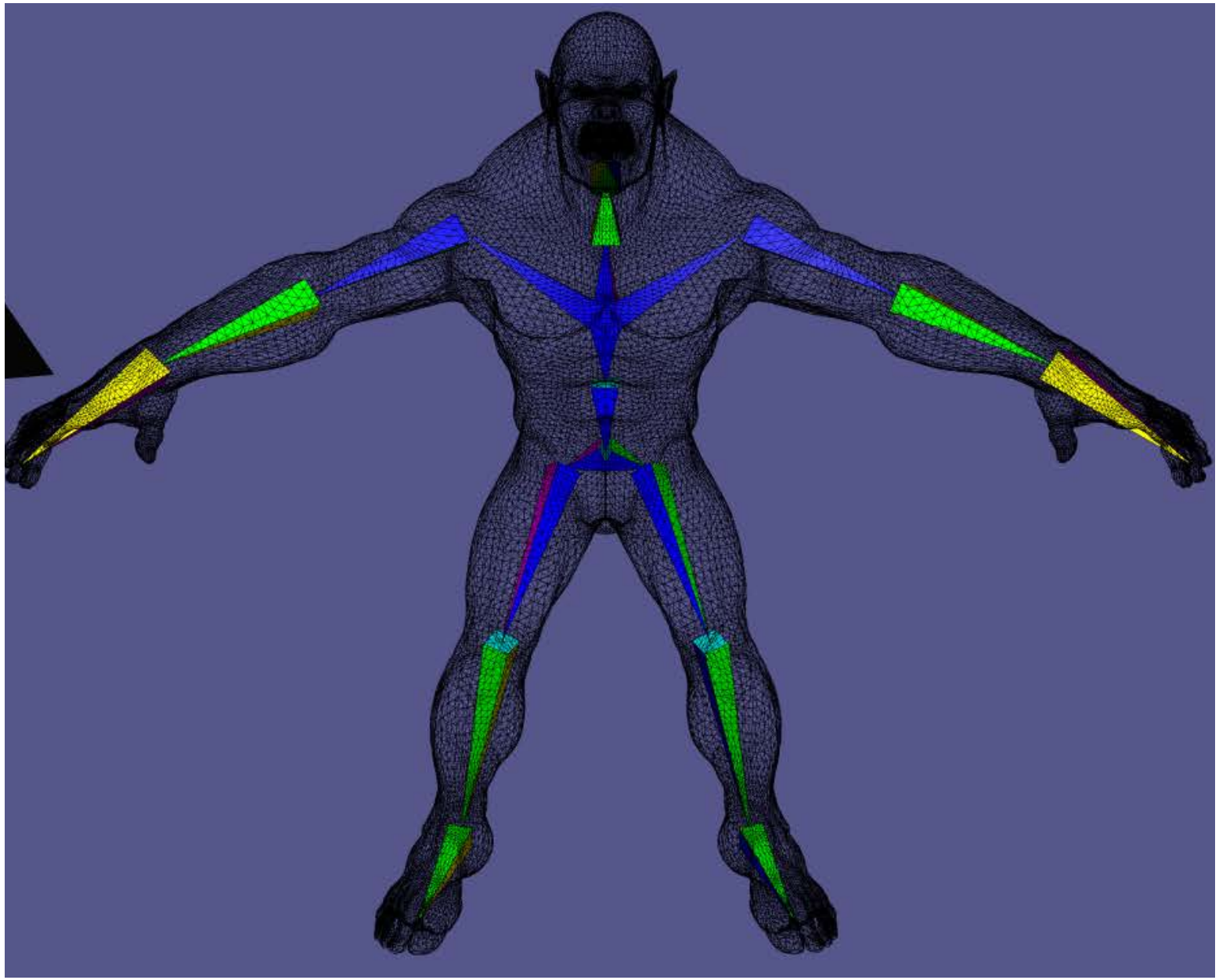
"Draw" Important Poses, interpolate in-between

How Can We Specify an Animation

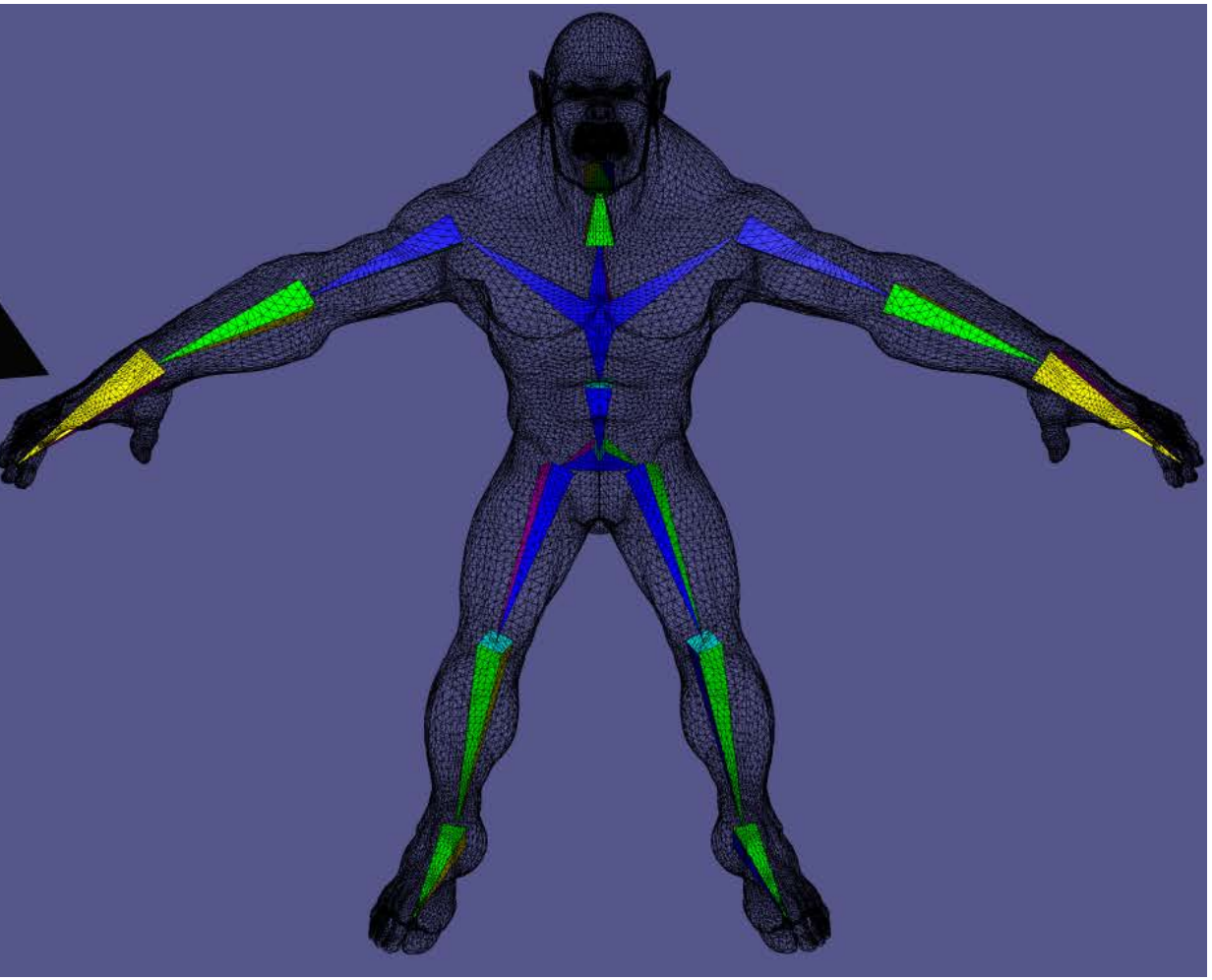


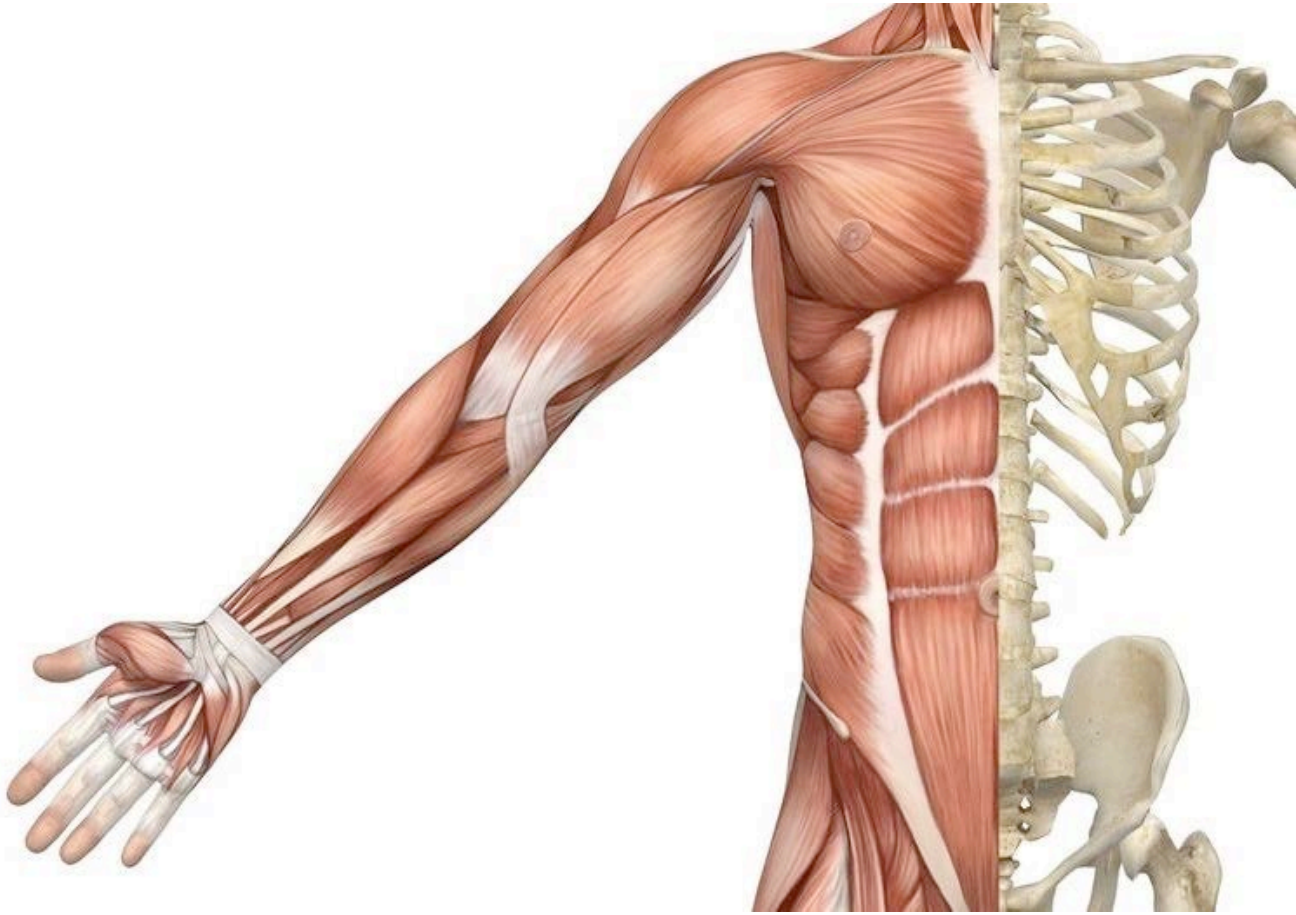
Per-Vertex?

Skinning

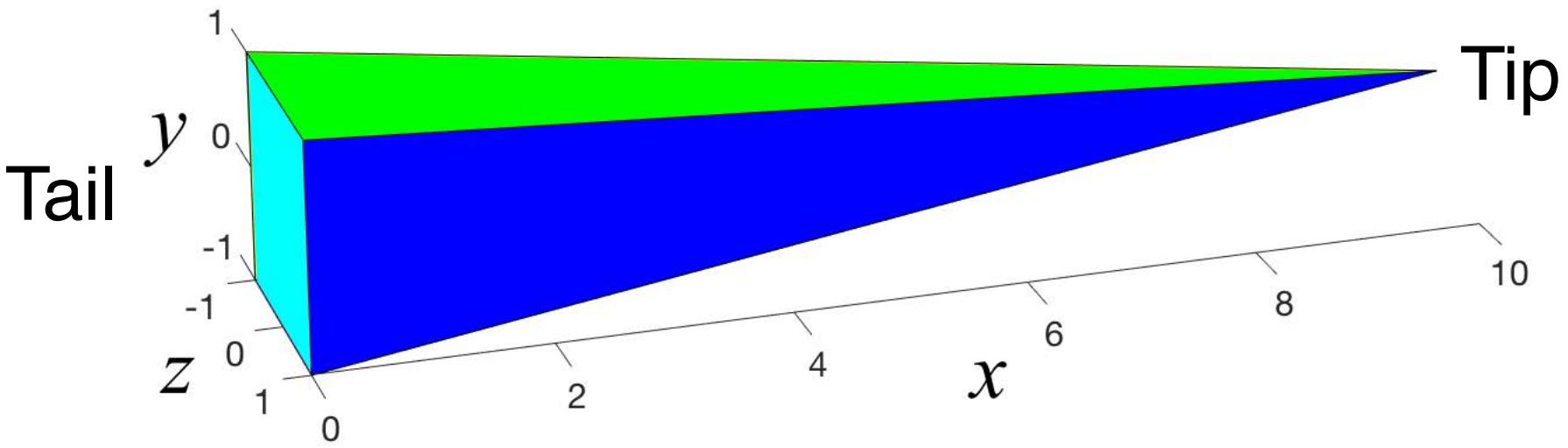


Bones

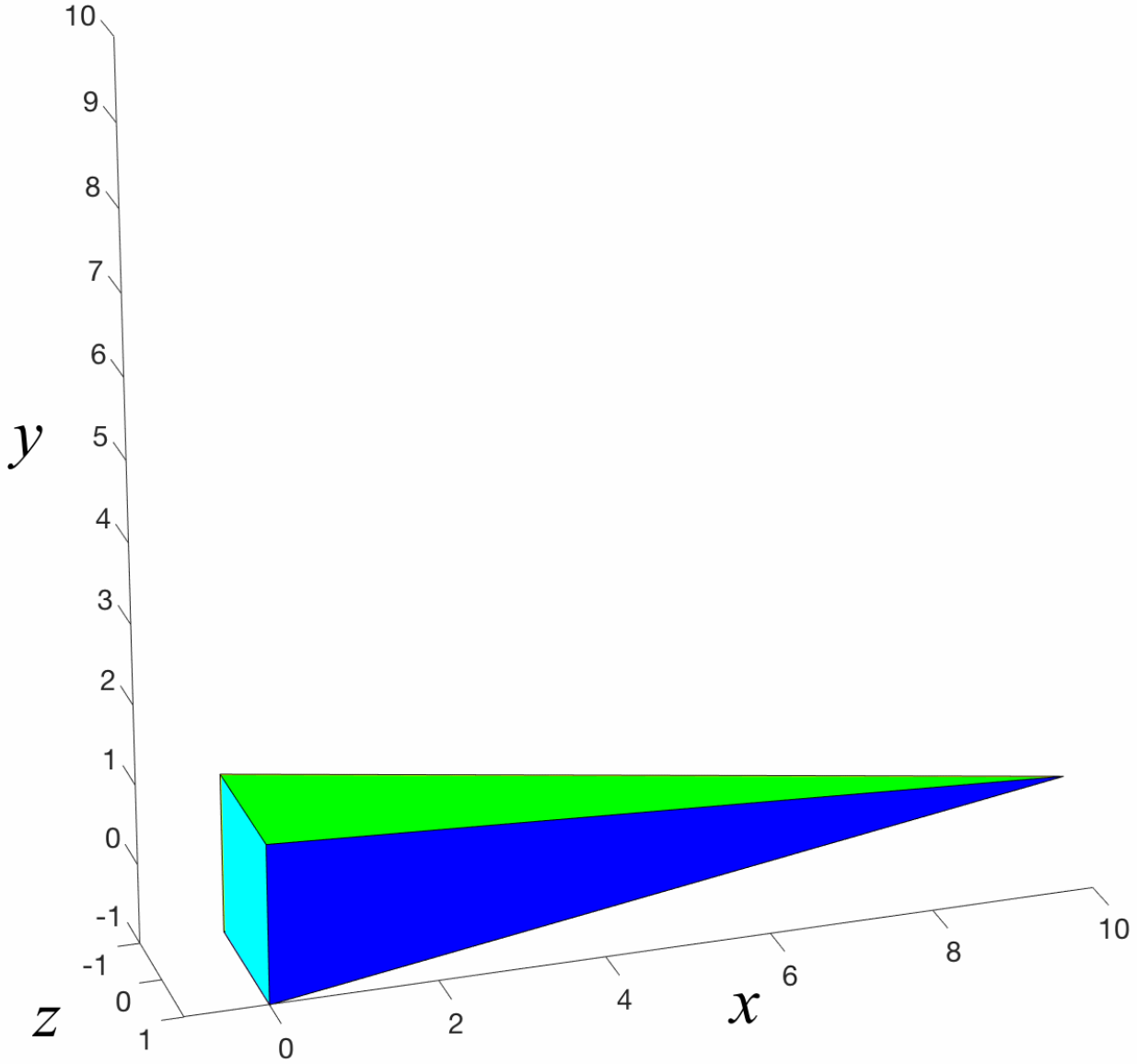




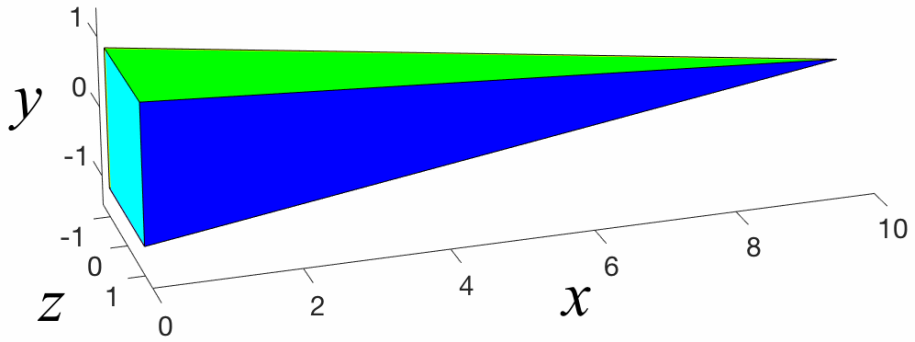
Bone of length $\ell = 10$



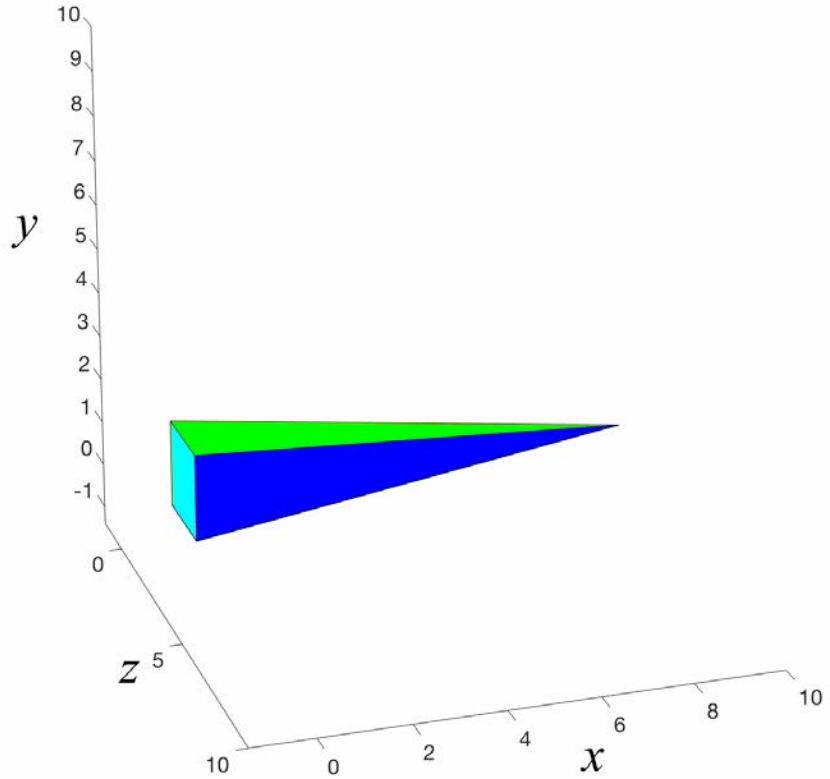
Bending around z axis: $\theta_2 = 0°$



Twisting around x axis: $\theta_1 = 0°$



Twist-bend-twist: $(\theta_1, \theta_2, \theta_3) = (0°, 0°, 0°)$



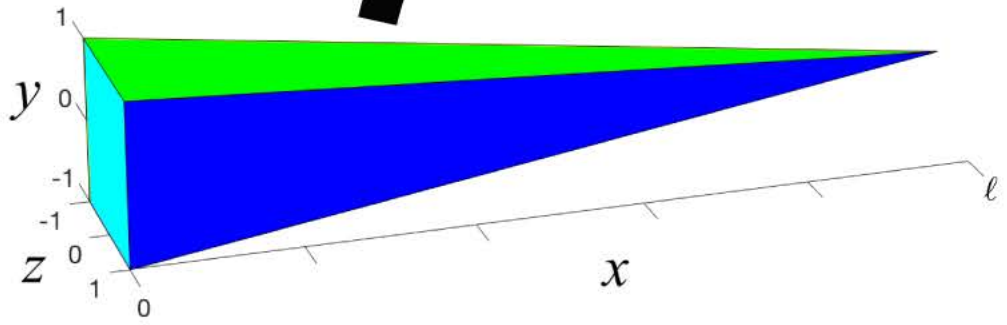
Bones in the Rest Pose

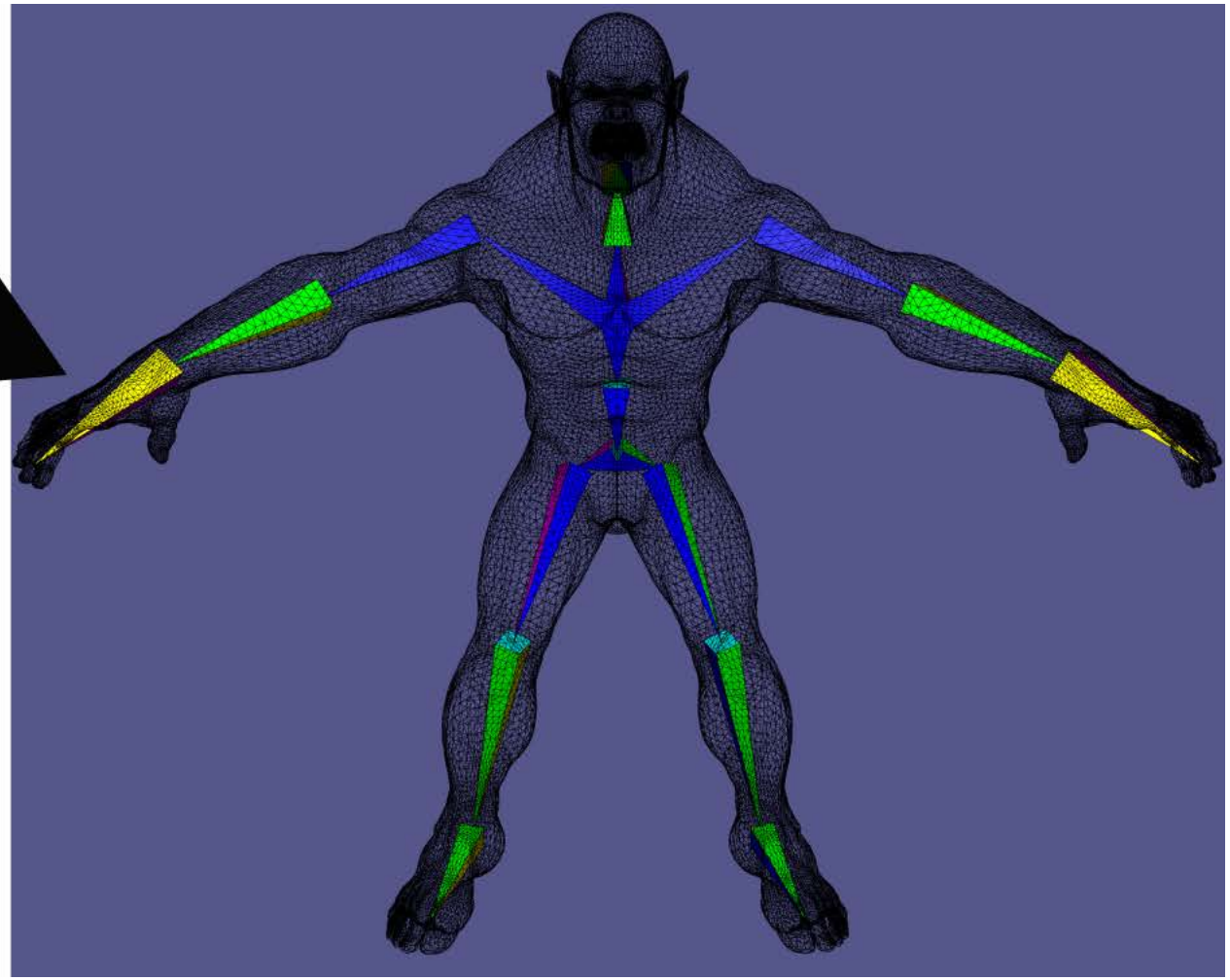
T = Transformation

R = Rotation

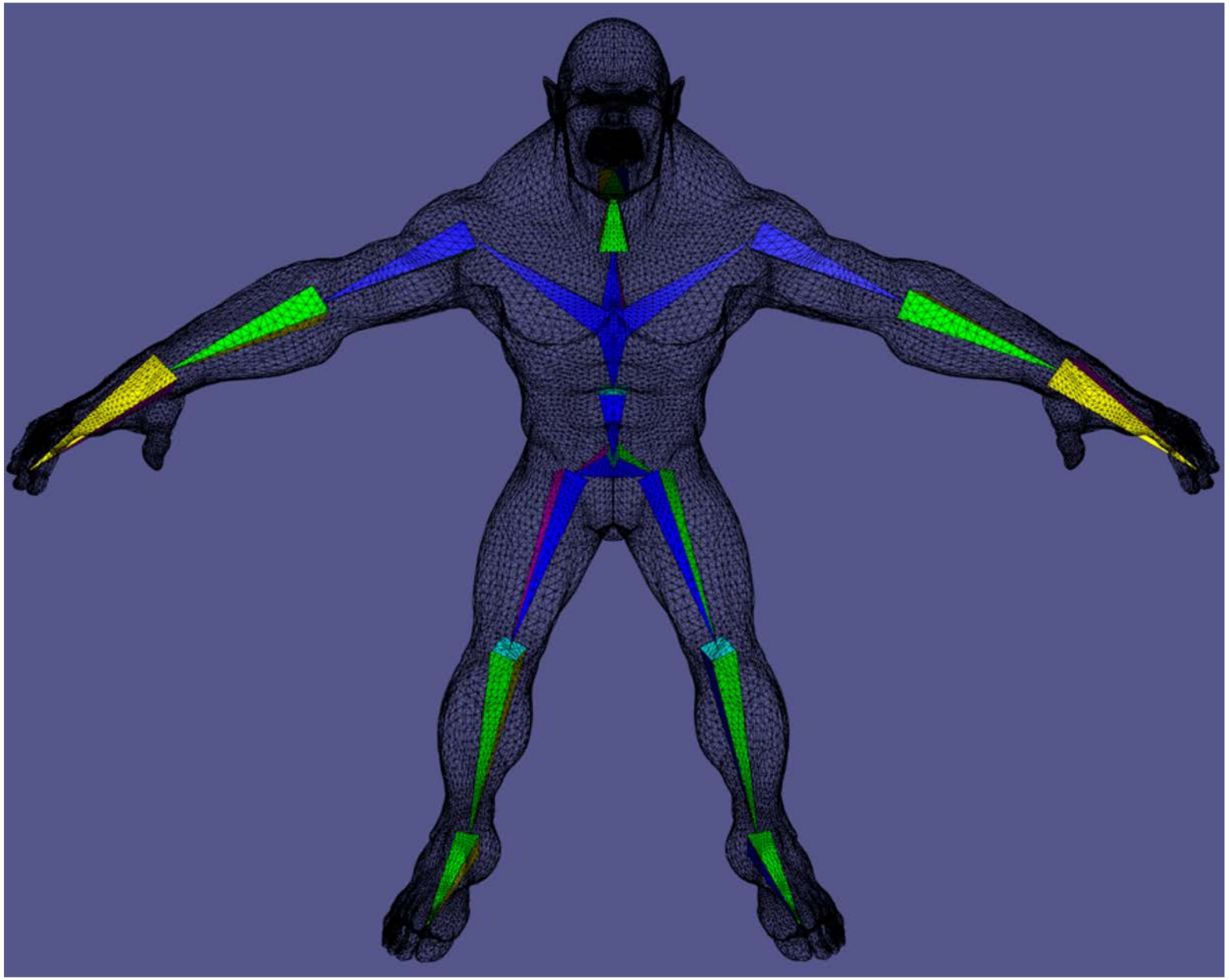
t = translation

$$\widehat{\mathbf{T}} = (\widehat{\mathbf{R}} \quad \hat{\mathbf{t}}) \in \mathbb{R}^{3\times4}$$





Posing a Bone

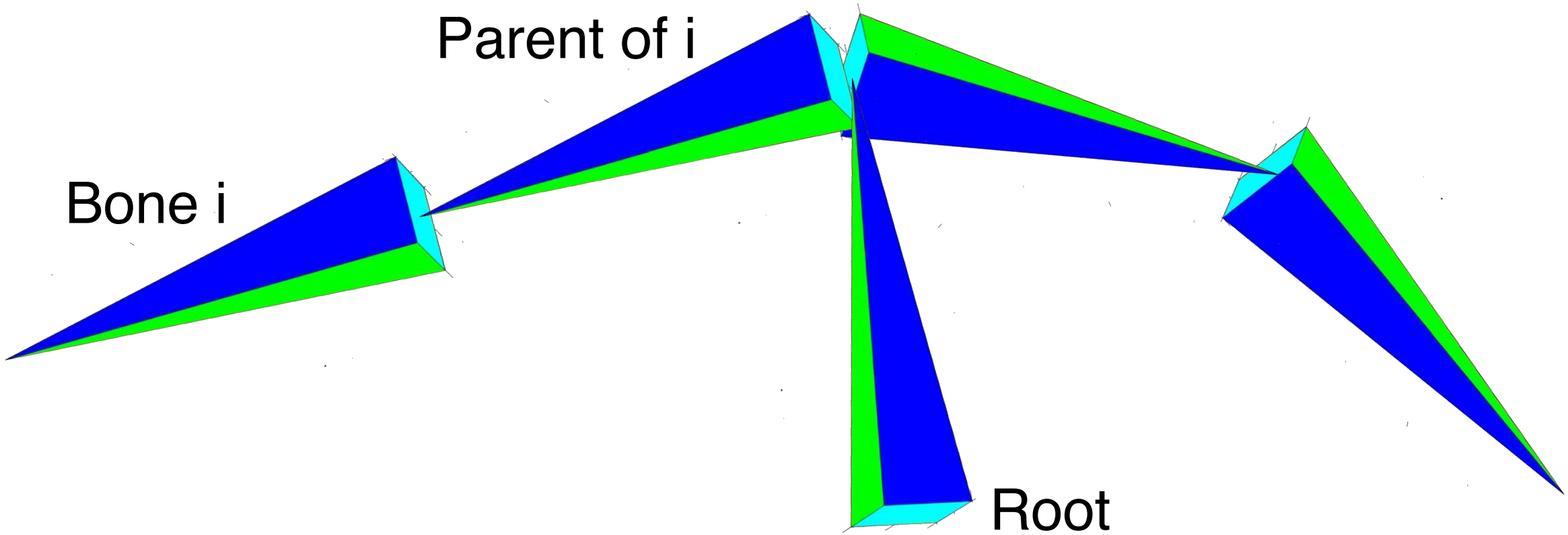


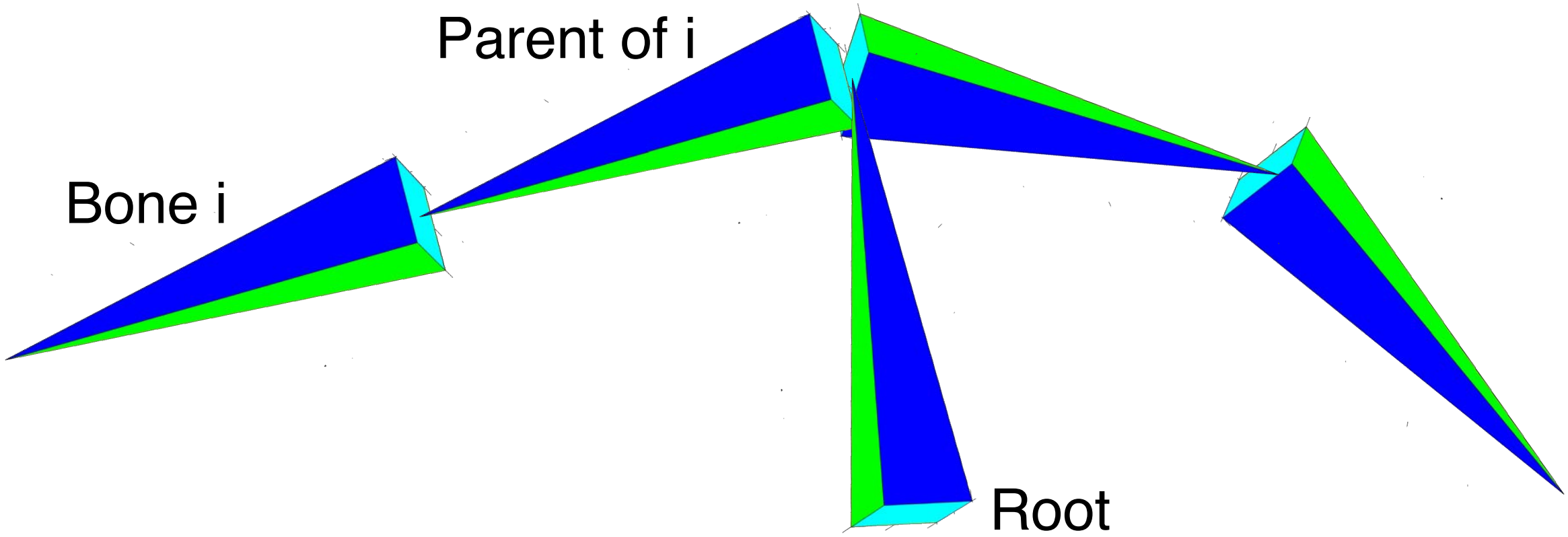
Forward Kinematics

Kinematics – study of motion without consideration of what causes that motion

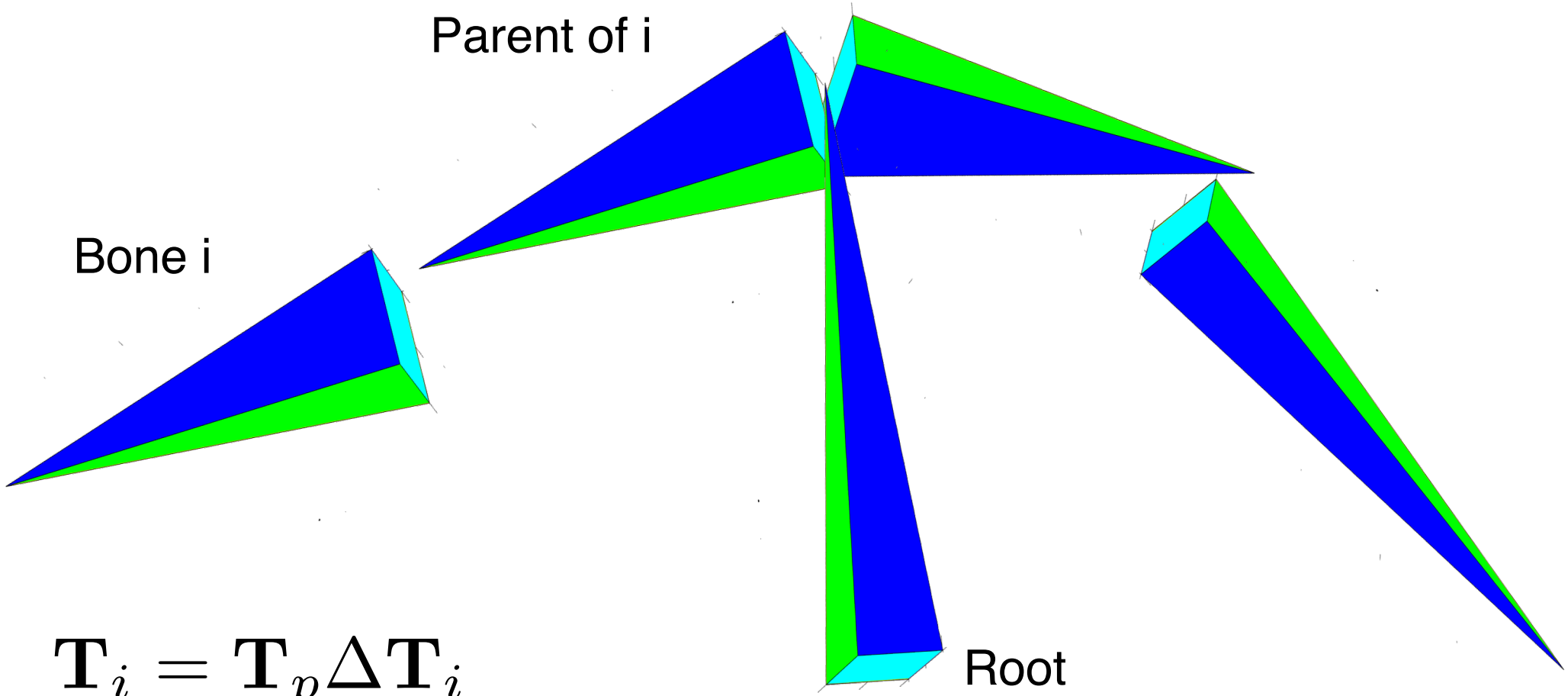
Forward Kinematics – Generate motion by setting all the bone positions by hand

To do this we need a more rigourous understanding of how our bone motions are represented


Parent of i
Bone i
Root



We will express bone transformations incrementally from there parent

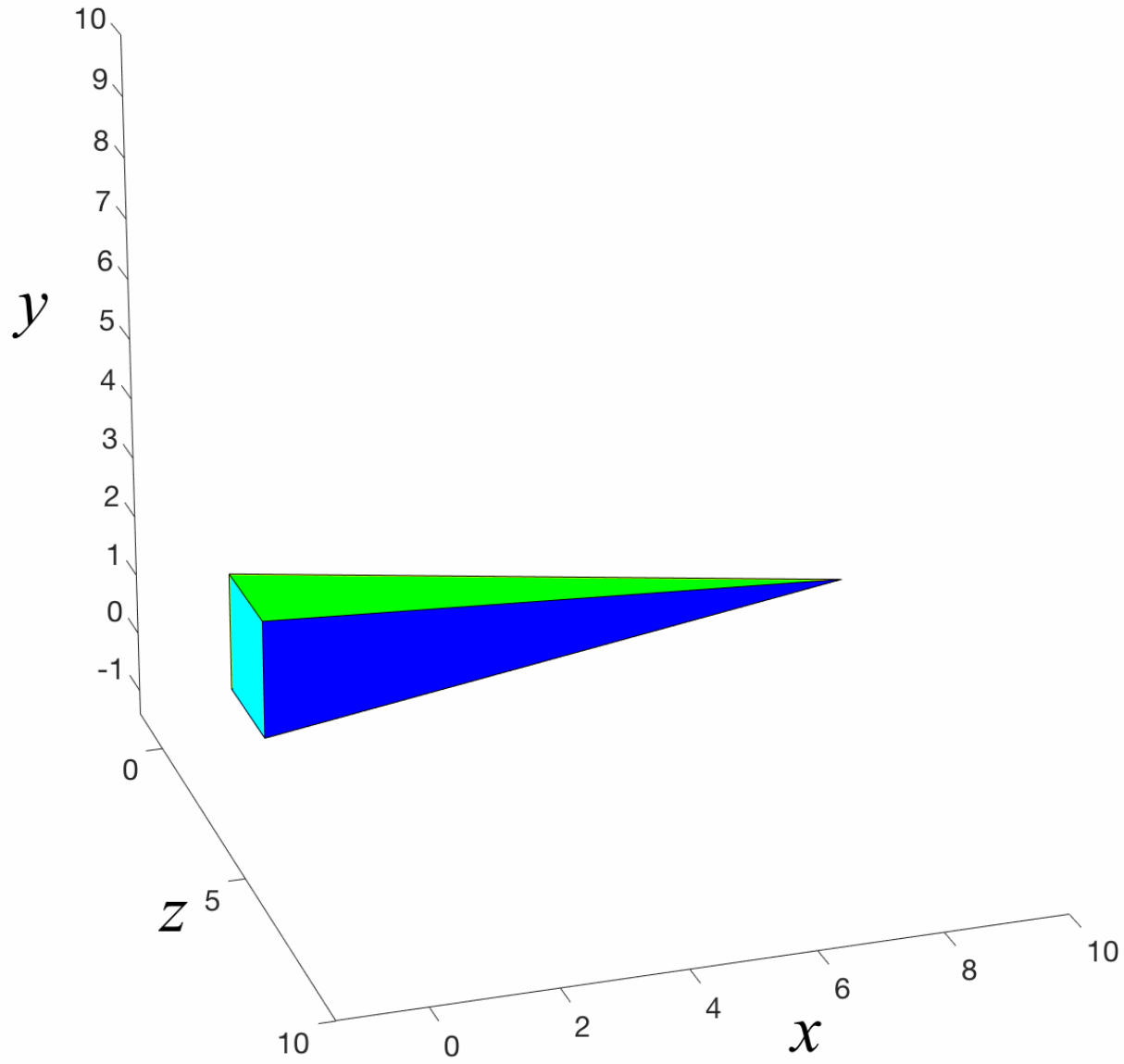


$$\mathbf{T}_i = \mathbf{T}_p \Delta \mathbf{T}_i$$

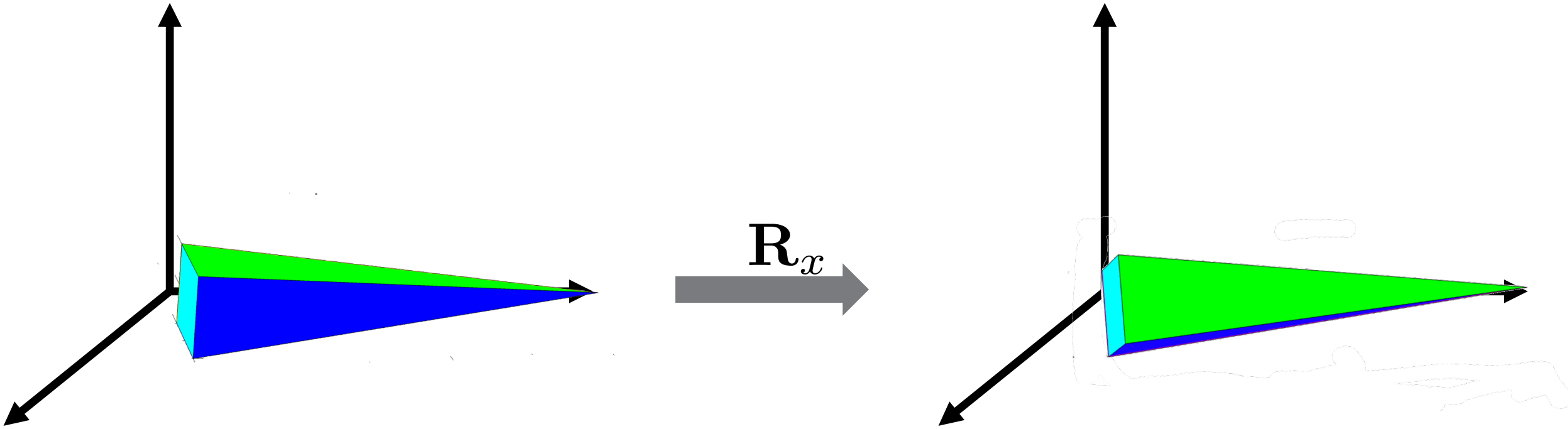
$$\Delta \mathbf{T}_i = \begin{pmatrix} \Delta \mathbf{R}_i & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix}$$

Euler Angles

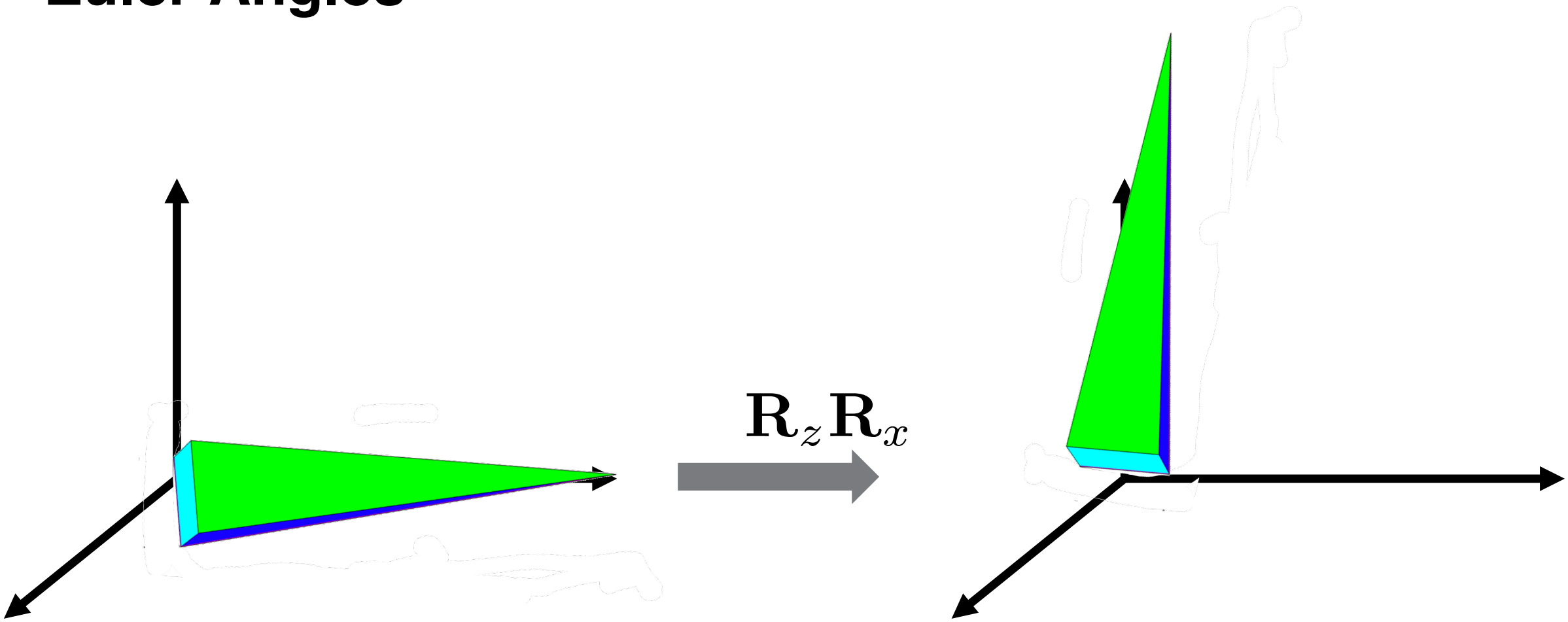
Twist-bend-twist: $(\theta_1, \theta_2, \theta_3) = (0^\circ, 0^\circ, 0^\circ)$



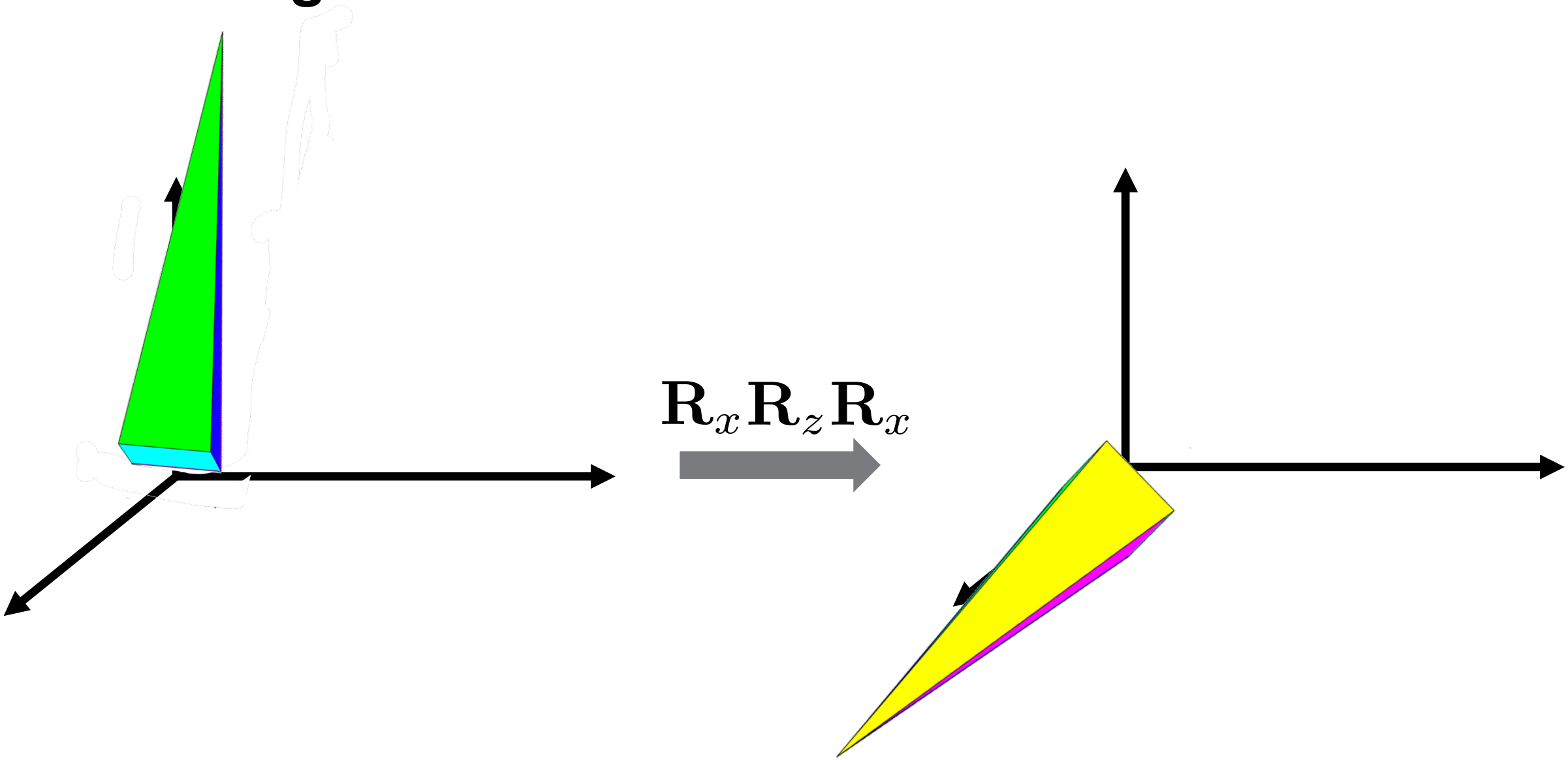
Euler Angles



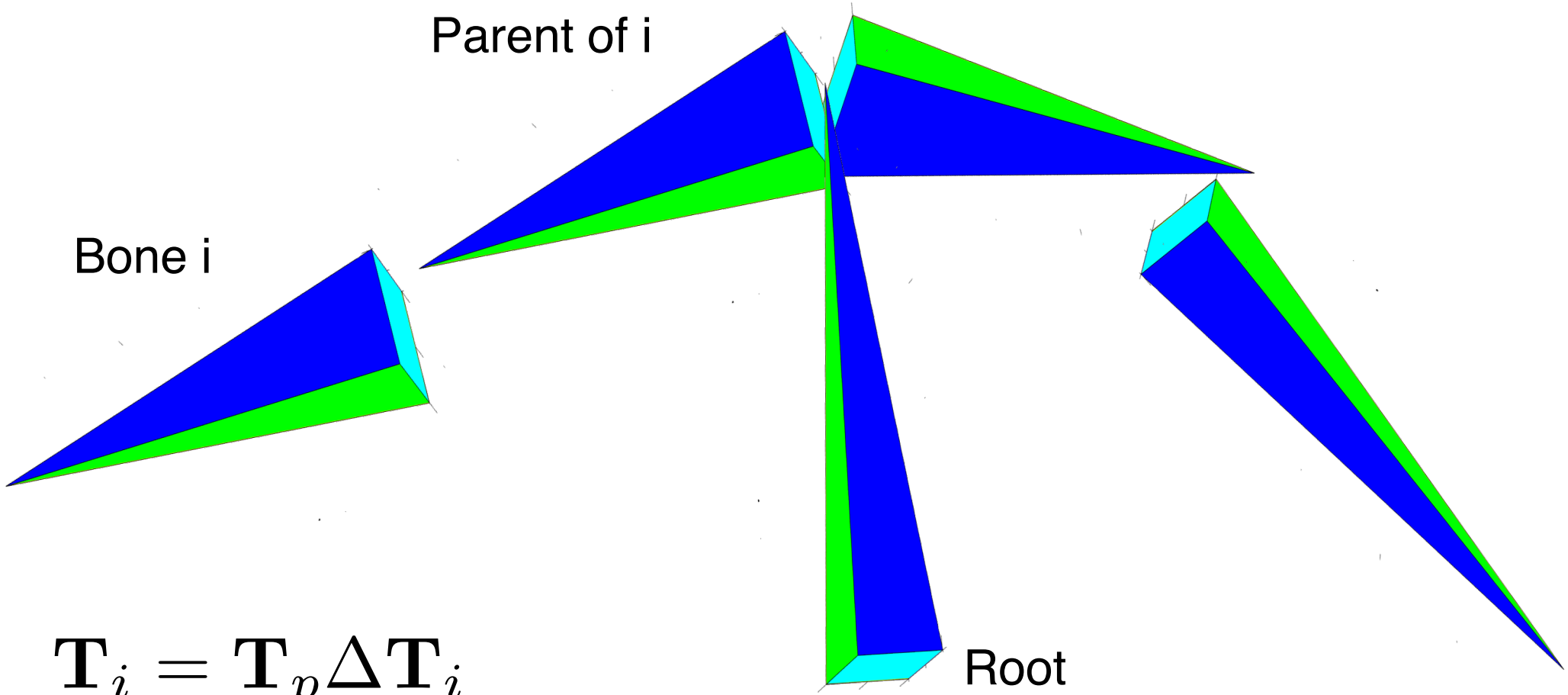
Euler Angles



Euler Angles



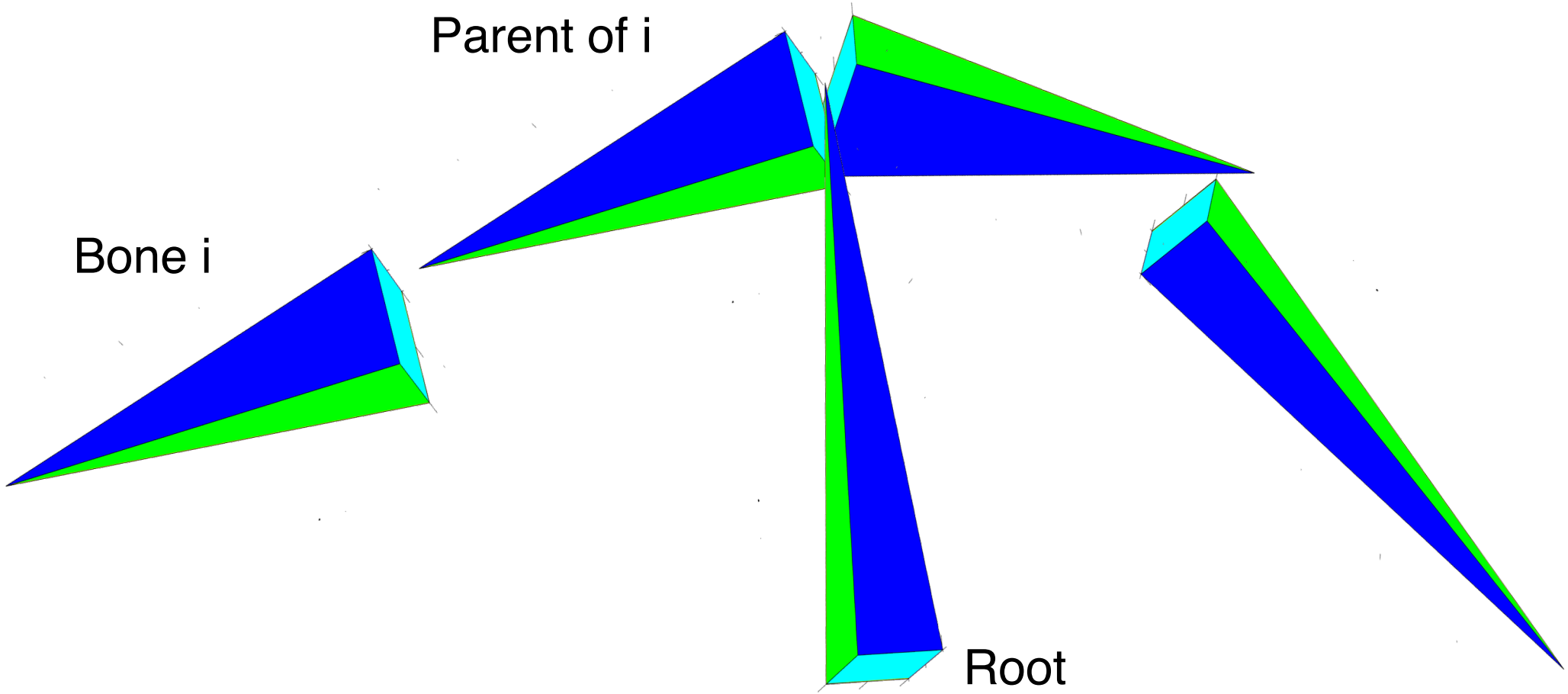
All expressed in the bone space



$$\mathbf{T}_i = \mathbf{T}_p \Delta \mathbf{T}_i$$

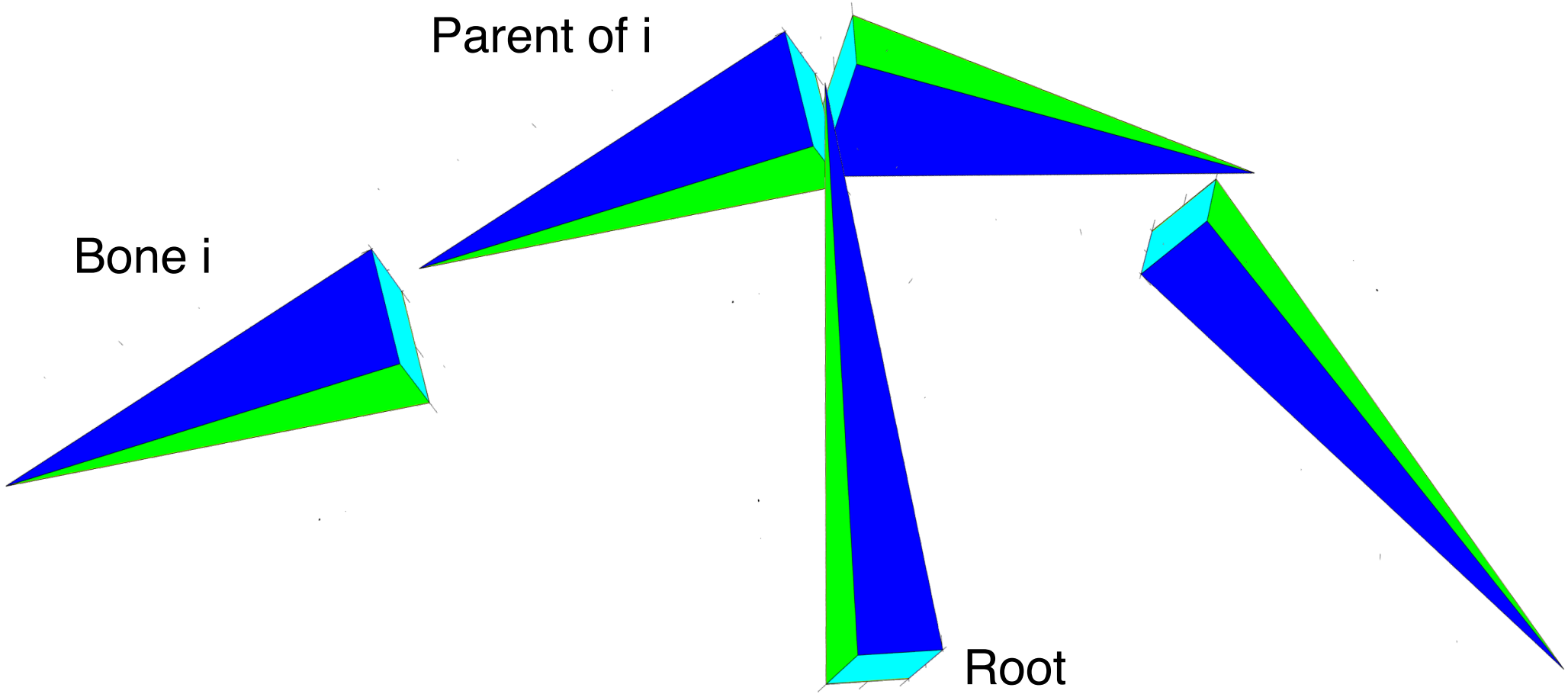
$$\Delta \mathbf{T}_i = \begin{pmatrix} \Delta \mathbf{R}_i & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix}$$

In the “rest” space not the bone space



$$\Delta \mathbf{T}_i = ? \begin{pmatrix} \mathbf{R}_x(\theta_{i3}) & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_z(\theta_{i2}) & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_x(\theta_{i1}) & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix} ?$$

In bone space, how do I get into/out of rest space



$$\Delta \mathbf{T}_i = \begin{pmatrix} \hat{\mathbf{T}}_i \\ 0\ 0\ 0\ 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_x\left(\theta_{i3}\right) & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_z\left(\theta_{i2}\right) & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{R}_x\left(\theta_{i1}\right) & \mathbf{0} \\ 0\ 0\ 0 & 1 \end{pmatrix} \begin{pmatrix} \hat{\mathbf{T}}_i \\ 0\ 0\ 0\ 1 \end{pmatrix}^{-1}$$

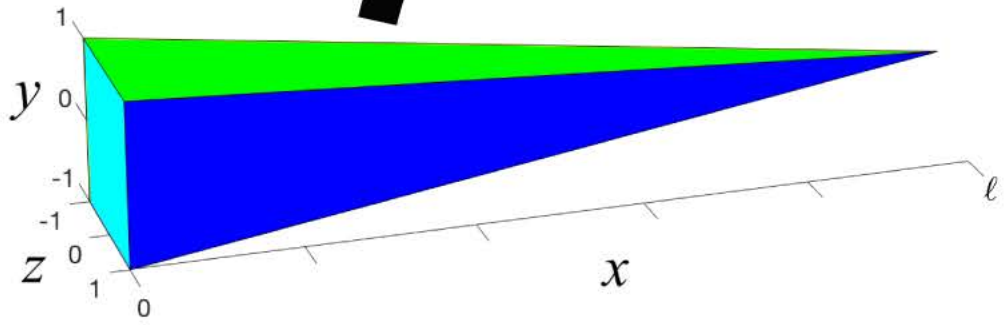
Connecting Bones to Vertices

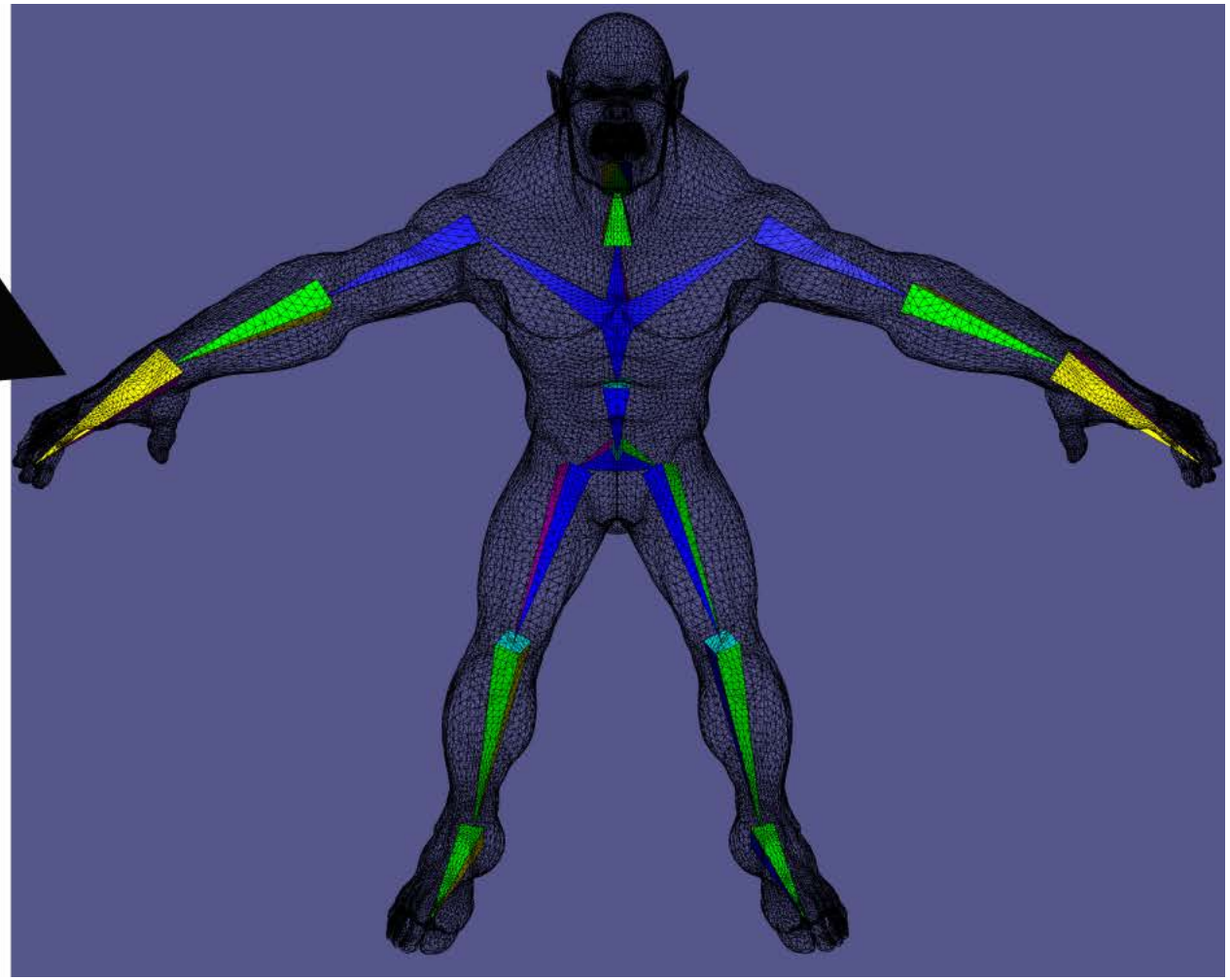
T = Transformation

R = Rotation

t = translation

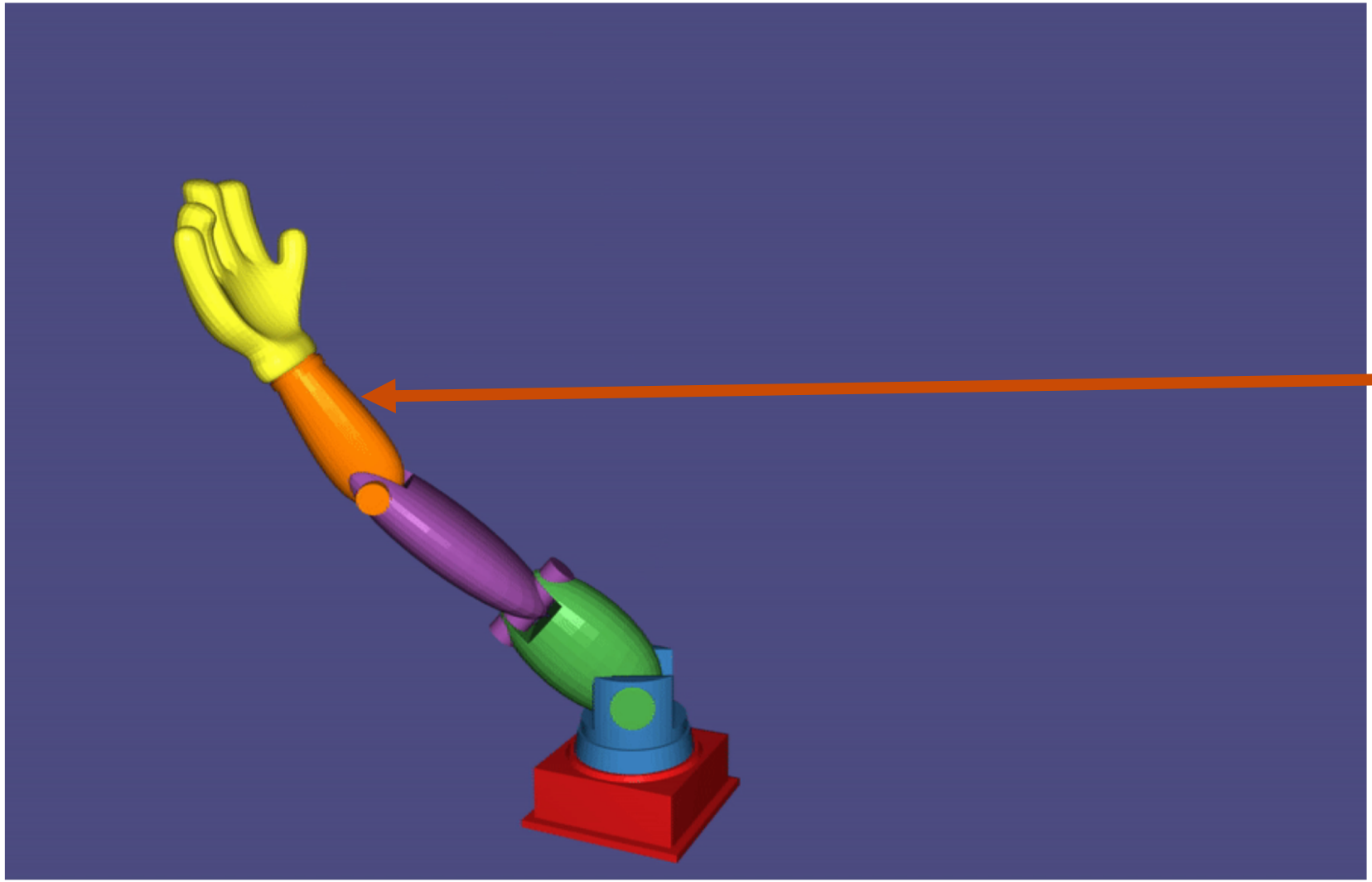
$$\widehat{\mathbf{T}} = (\widehat{\mathbf{R}} \quad \hat{\mathbf{t}}) \in \mathbb{R}^{3\times4}$$





Rigid “Skinning”

Idea: Attach each vertex to a single bone



$$\mathbf{v}_j = \mathbf{T}_i \hat{\mathbf{v}}_j$$

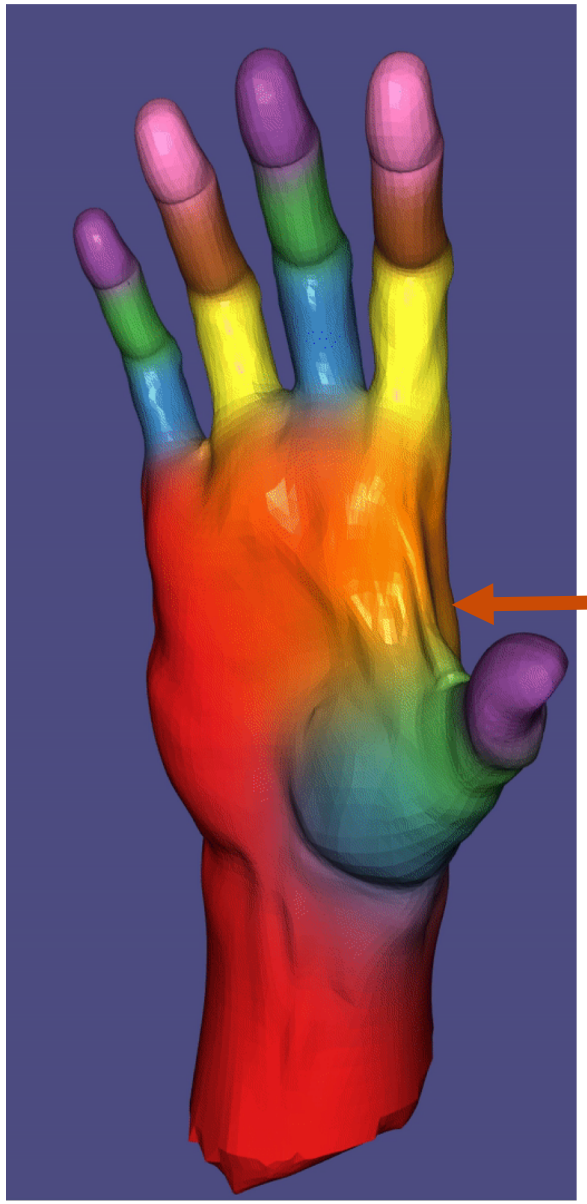
Deformable Skinning

Rigid Skinning is fine for mechanical things, but for smoother deformations we need to try something else

Rather than attach each vertex to a single bone, we attach each vertex to multiple bones and blend their transformations

If this blending is linear in the transformations, we call it linear blend skinning.

Linear Blend Skinning



$$\mathbf{v}_j = \sum_{i=1}^{nBones} w_{ij} \mathbf{T}_i \hat{\mathbf{v}}_j$$

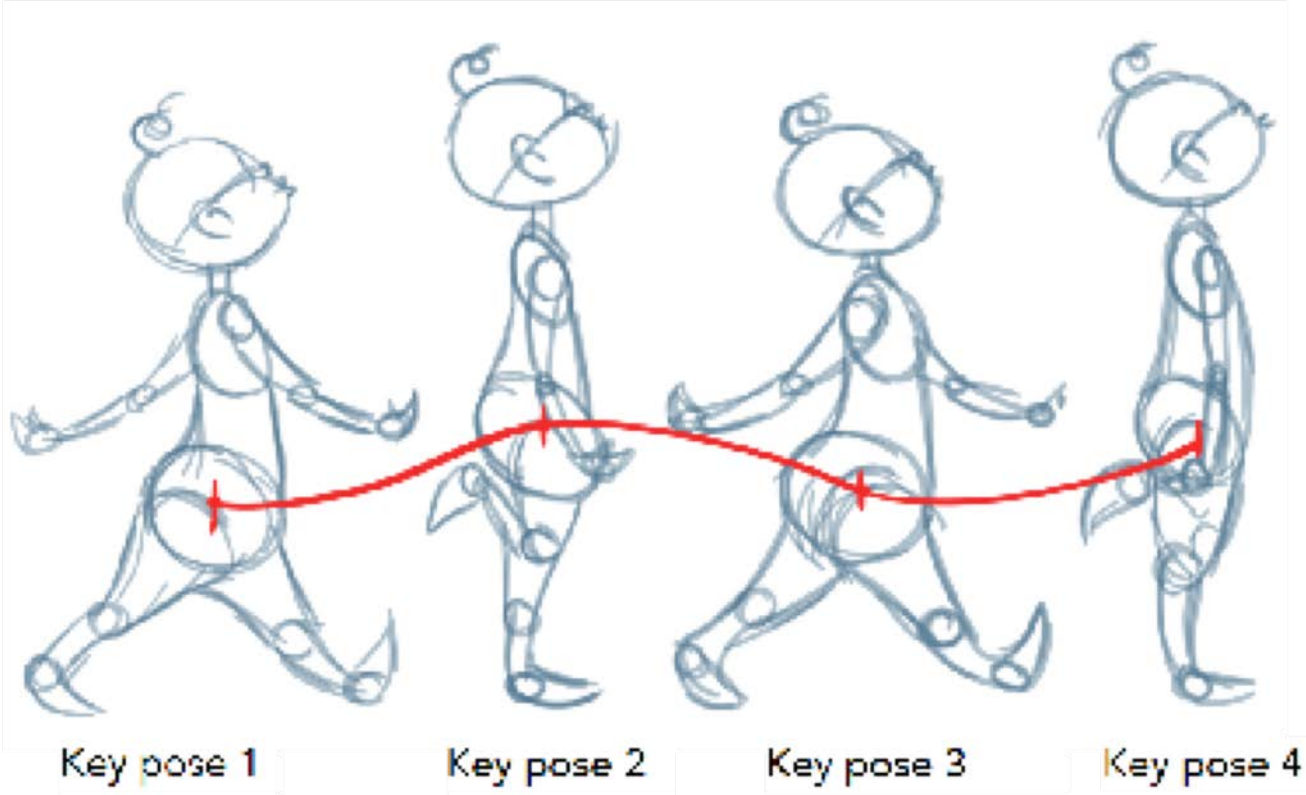
weight

Rigid vs Linear Blend Skinning

Animation via Keyframing

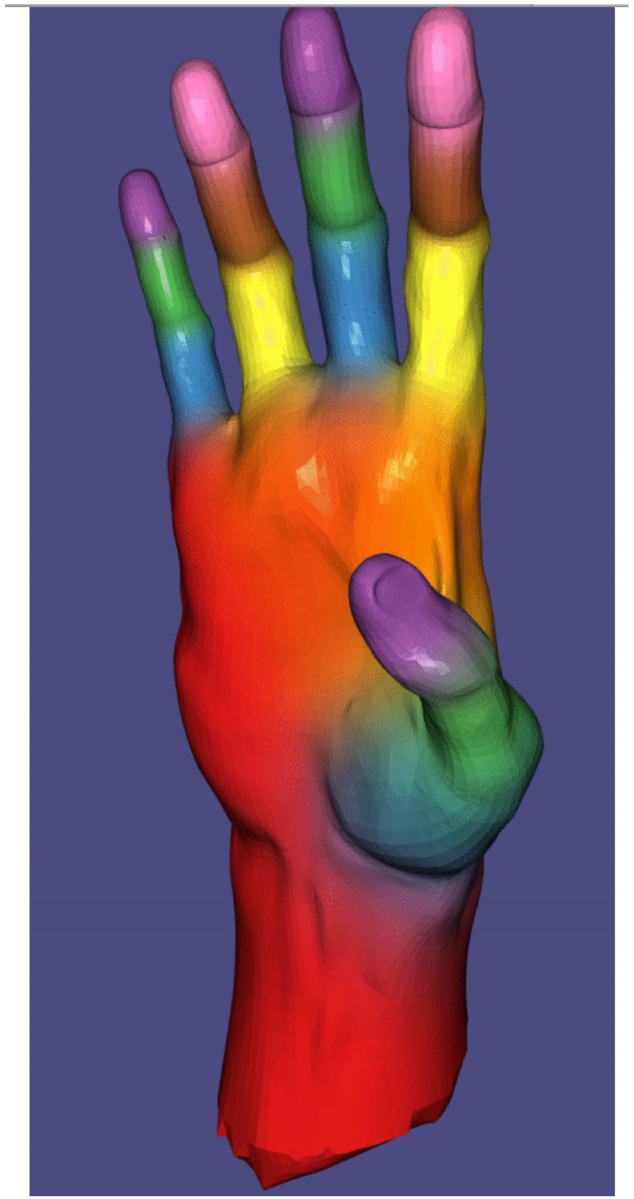
We can pose objects now

How do we generate an animation ?



Specifying Keyframes

Time = 0



Time = 10s

Poses are generated by specifying rotations of bones

Each pose can be represented as

$$\theta = \begin{pmatrix} \theta_{11} \\ \theta_{11} \\ \theta_{11} \\ \vdots \\ \theta_{n1} \\ \theta_{n2} \\ \theta_{n3} \end{pmatrix}$$

Specifying Keyframes

Time = 0

??????????

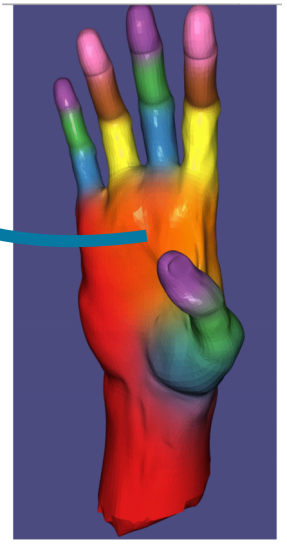
Time = 10s

Specifying Keyframes

$\theta = \mathbf{c}(t)$

Time = 0

$\theta = \theta_0$



Time = 10s

$\theta = \theta_1$

Pose Space

Interpolating Keyframes

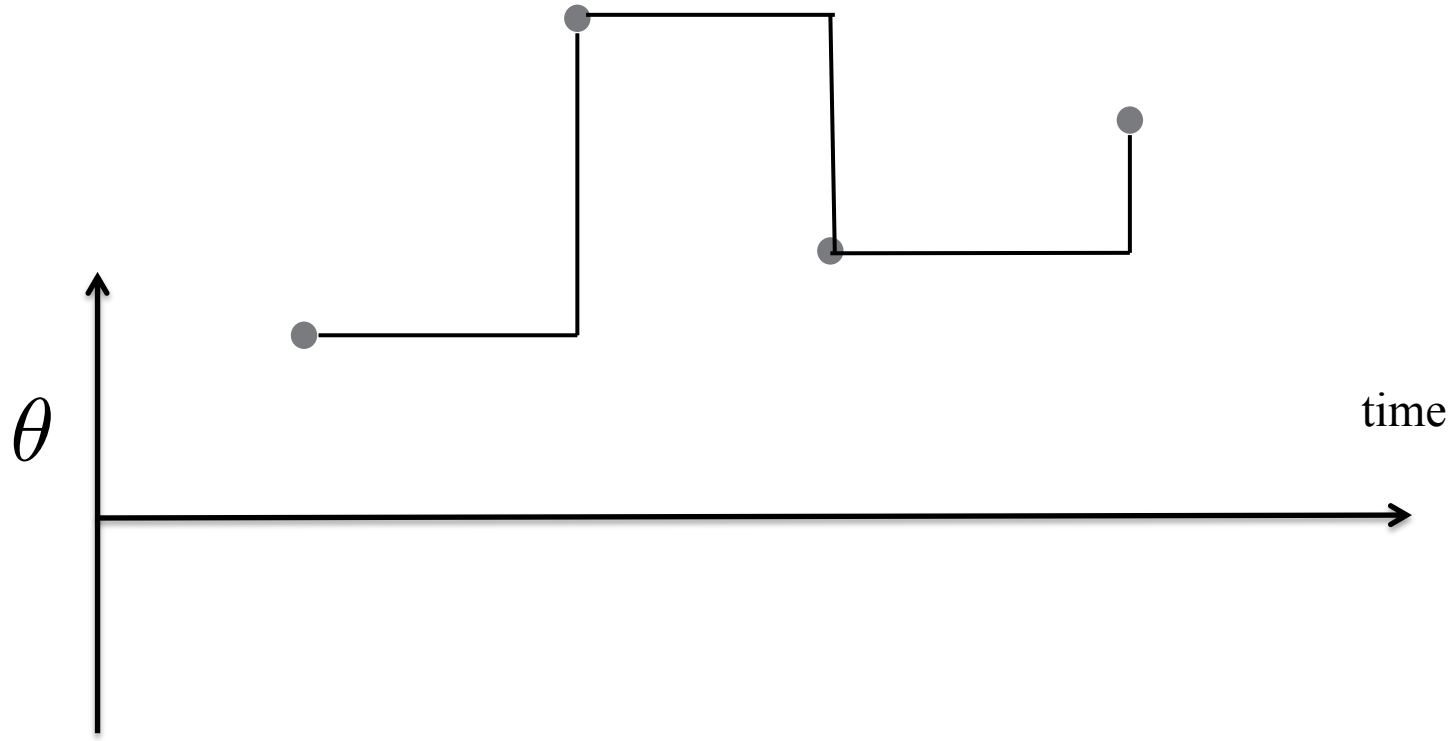
$\theta = \mathbf{c}(t)$ is a curve in the pose space

How could we construct such a curve from keyframes ?

Interpolating Keyframes

$\theta = \mathbf{c}(t)$ is a curve in the pose space

How could we construct such a curve from keyframes ?

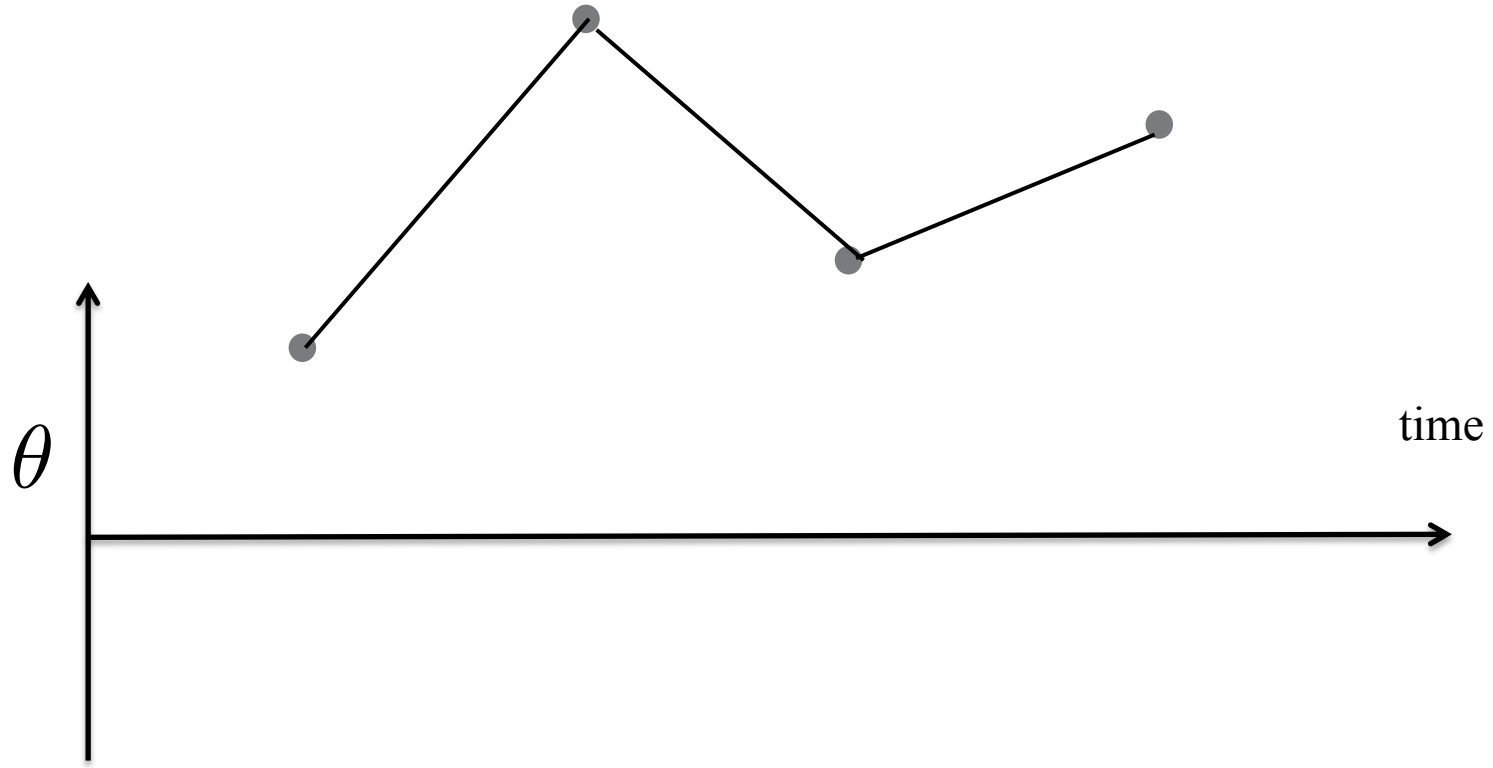


Hold

Interpolating Keyframes

$\theta = \mathbf{c}(t)$ is a curve in the pose space

How could we construct such a curve from keyframes ?

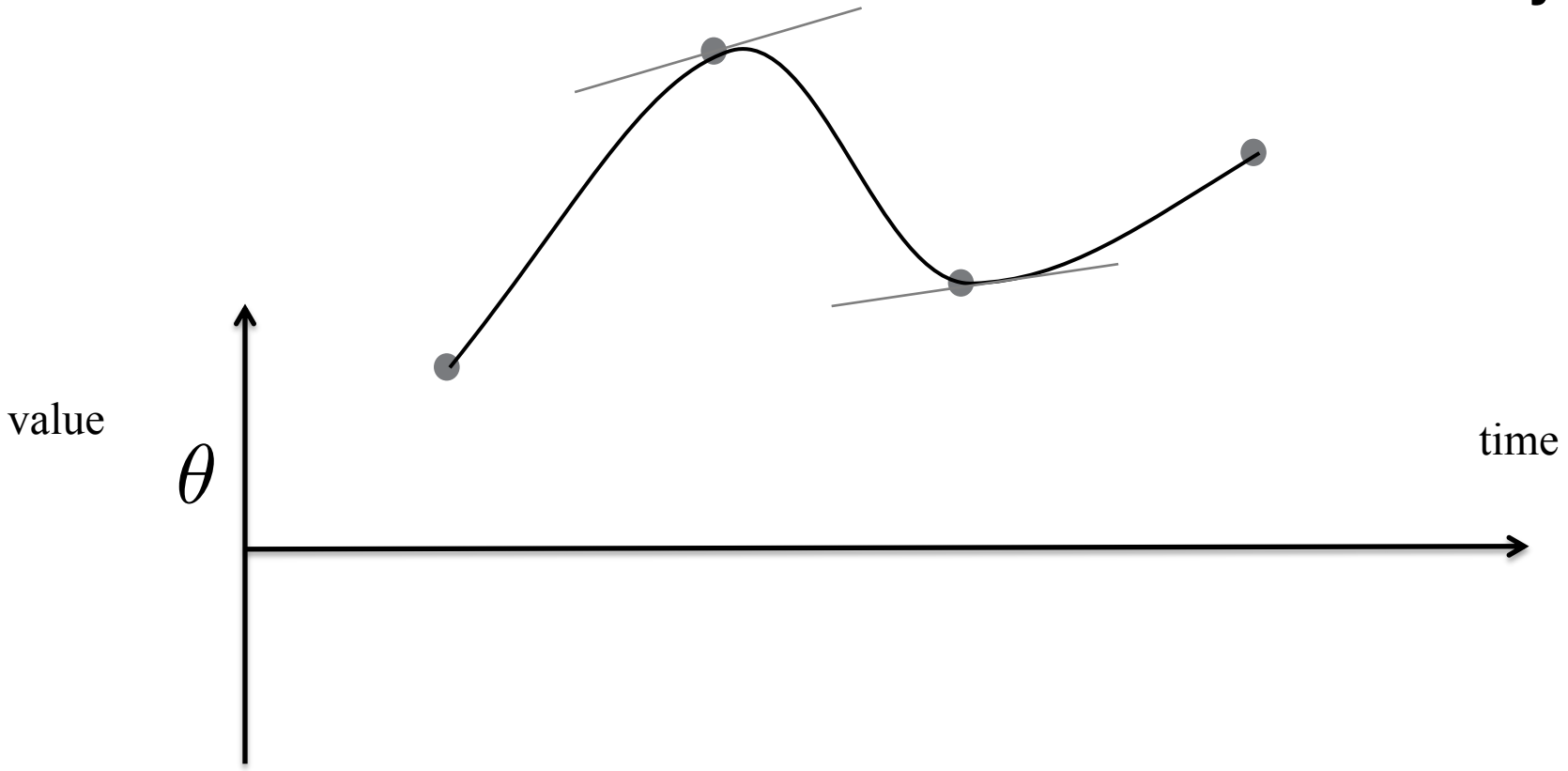


Linear

Interpolating Keyframes

$\theta = \mathbf{c}(t)$ is a curve in the pose space

How could we construct such a curve from keyframes ?



Spline

Catmull-Rom Spline

A **cubic** curve created by specifying the end points and the tangents of the curve.

$$c(t) = at^3 + bt^2 + ct + d$$

Catmull-Rom Spline

A **cubic** curve created by specifying the end points and the tangents of the curve.

$$c(0) = d$$

$$c(1) = a + b + c + d$$

$$\frac{dc}{dt}(1) = 3a + 2b + 1c$$

$$\frac{dc}{dt}(0) = 1c$$

Catmull-Rom Spline

$$\begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 3 & 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{a}^T \\ \mathbf{b}^T \\ \mathbf{c}^T \\ \mathbf{d}^T \end{pmatrix} = \begin{pmatrix} \theta_0^T \\ \theta_1^T \\ \mathbf{m}_0^T \\ \mathbf{m}_1^T \end{pmatrix}$$

Catmull-Rom Spline

After solving and rearranging we end up with

$$\mathbf{c}(t) = (2t^3 - 3t^2 + 1)\theta_0 + (t^3 - 2t^2 + t)\mathbf{m}_0 + (-2t^3 + 3t^2)\theta_1 + (t^3 - t^2)\mathbf{m}_1$$

Remember this is for t = 0 to t = 1.

For arbitrary intervals substitute $t = (t' - t_0)/(t_1 - t_0)$

Catmull-Rom Spline

After solving and rearranging we end up with

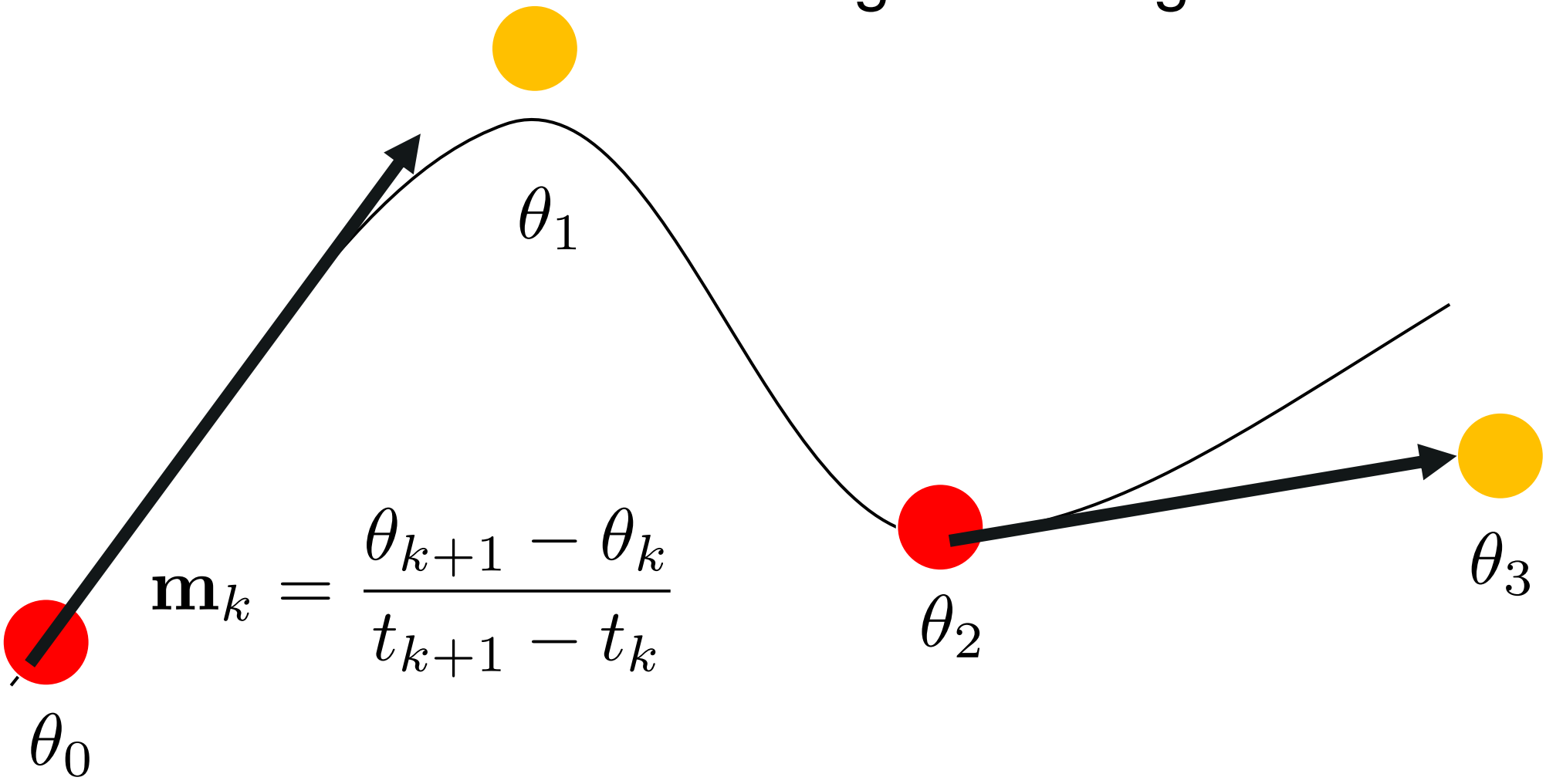
$$\mathbf{c}(t) = (2t^3 - 3t^2 + 1)\theta_0 + (t^3 - 2t^2 + t)\mathbf{m}_0 + (-2t^3 + 3t^2)\theta_1 + (t^3 - t^2)\mathbf{m}_1$$

This is a general cubic spline (cubic in t).

Catmull-Rom choses the tangents using “Finite Differences”

Catmull-Rom Spline

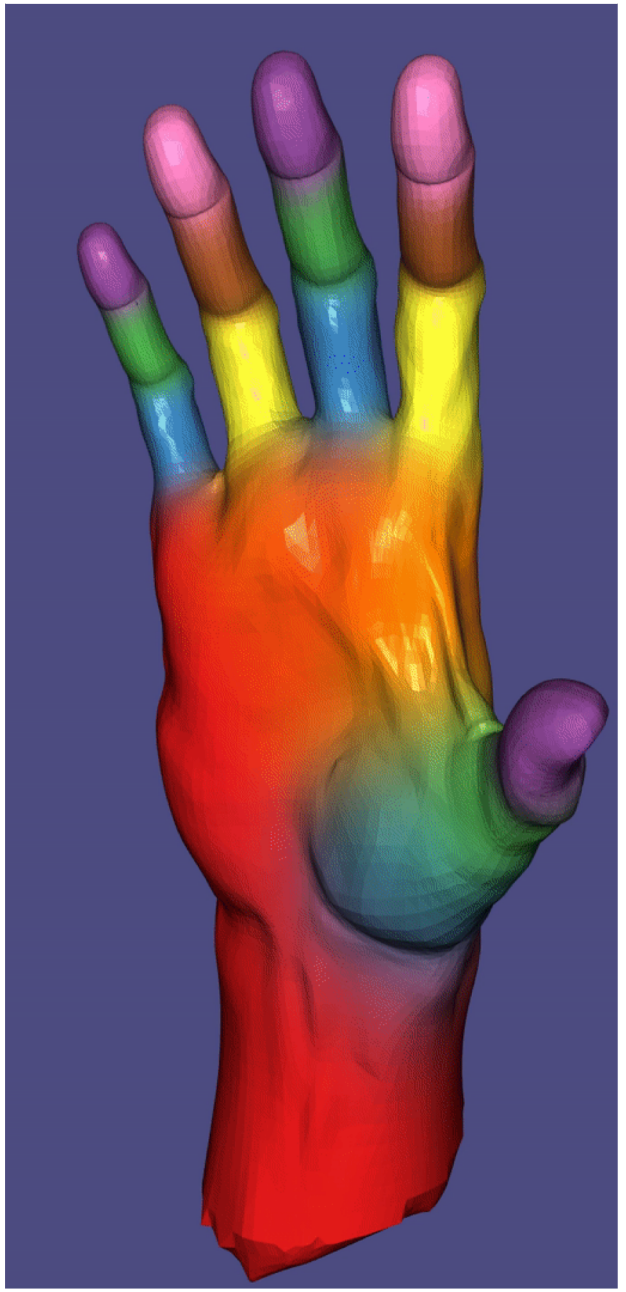
Catmull-Rom chooses the tangents using “Finite Differences”



Catmull-Rom Animation

For each time, t, create a new θ(t) using your spline

Deform your bones using θ and draw



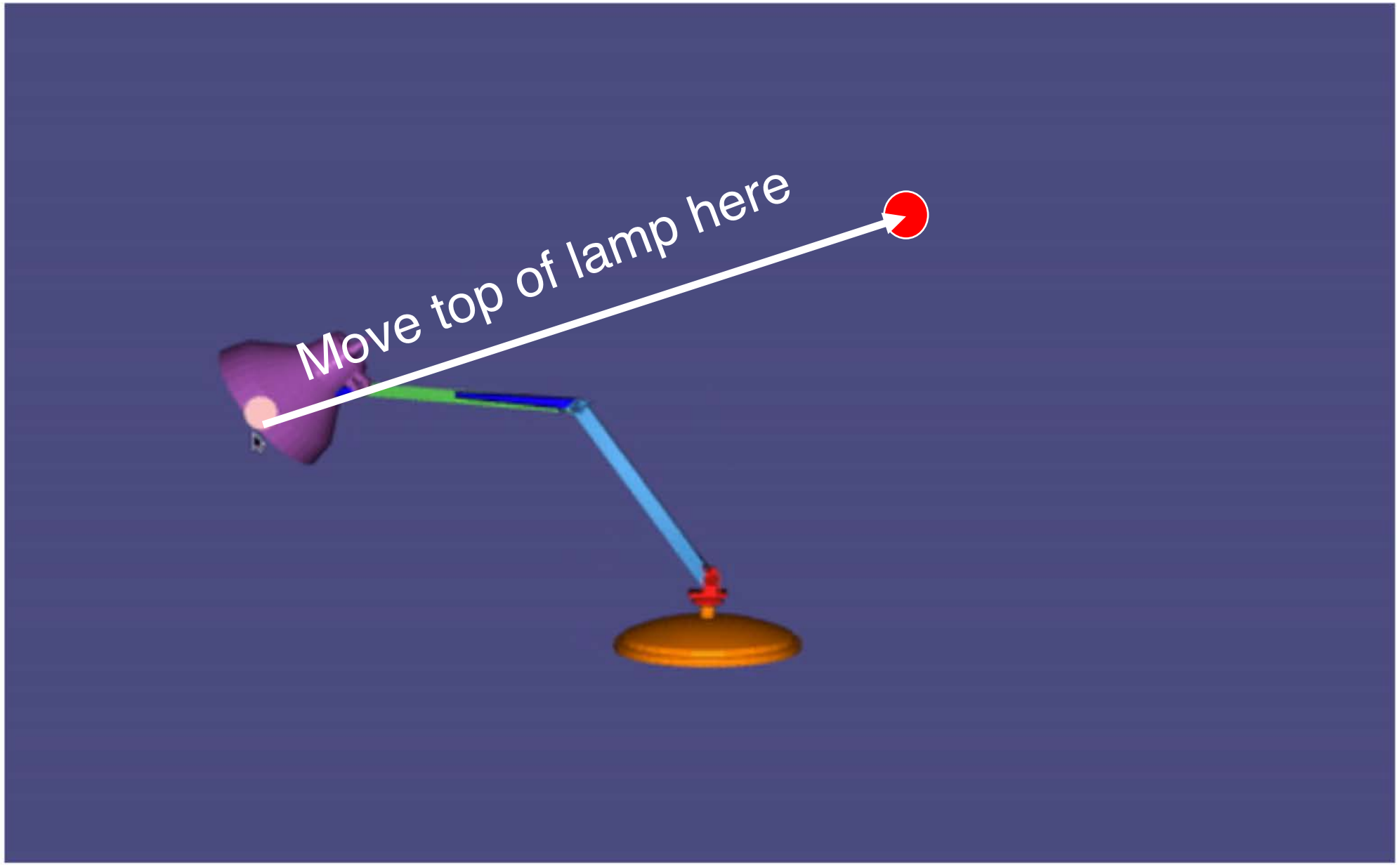
Inverse Kinematics

One last thing …

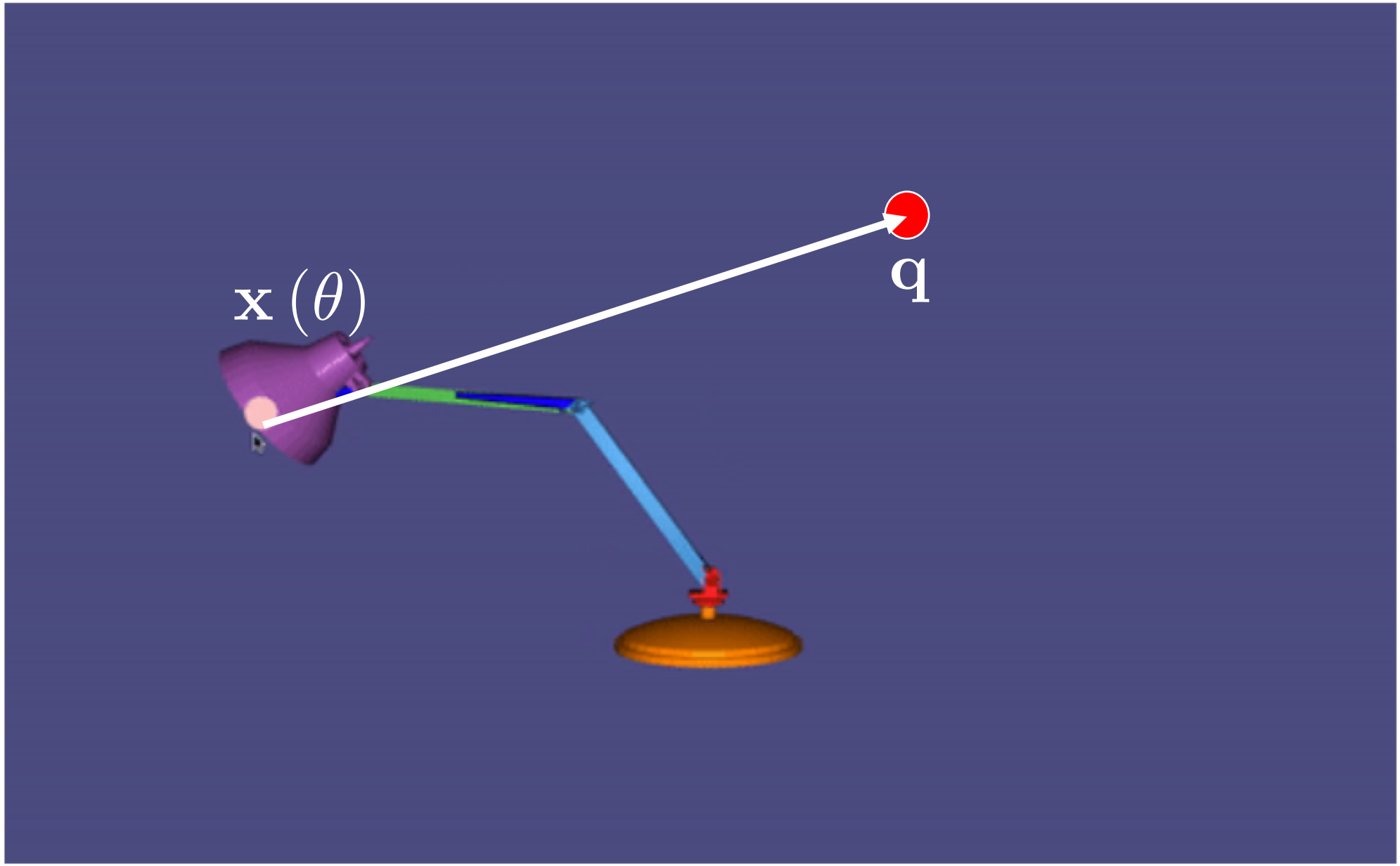
Posing all those bones can be tedious, wouldn't it be great if you could just specify a few bones and the rest would be automatically computed ?

That's what inverse kinematics does

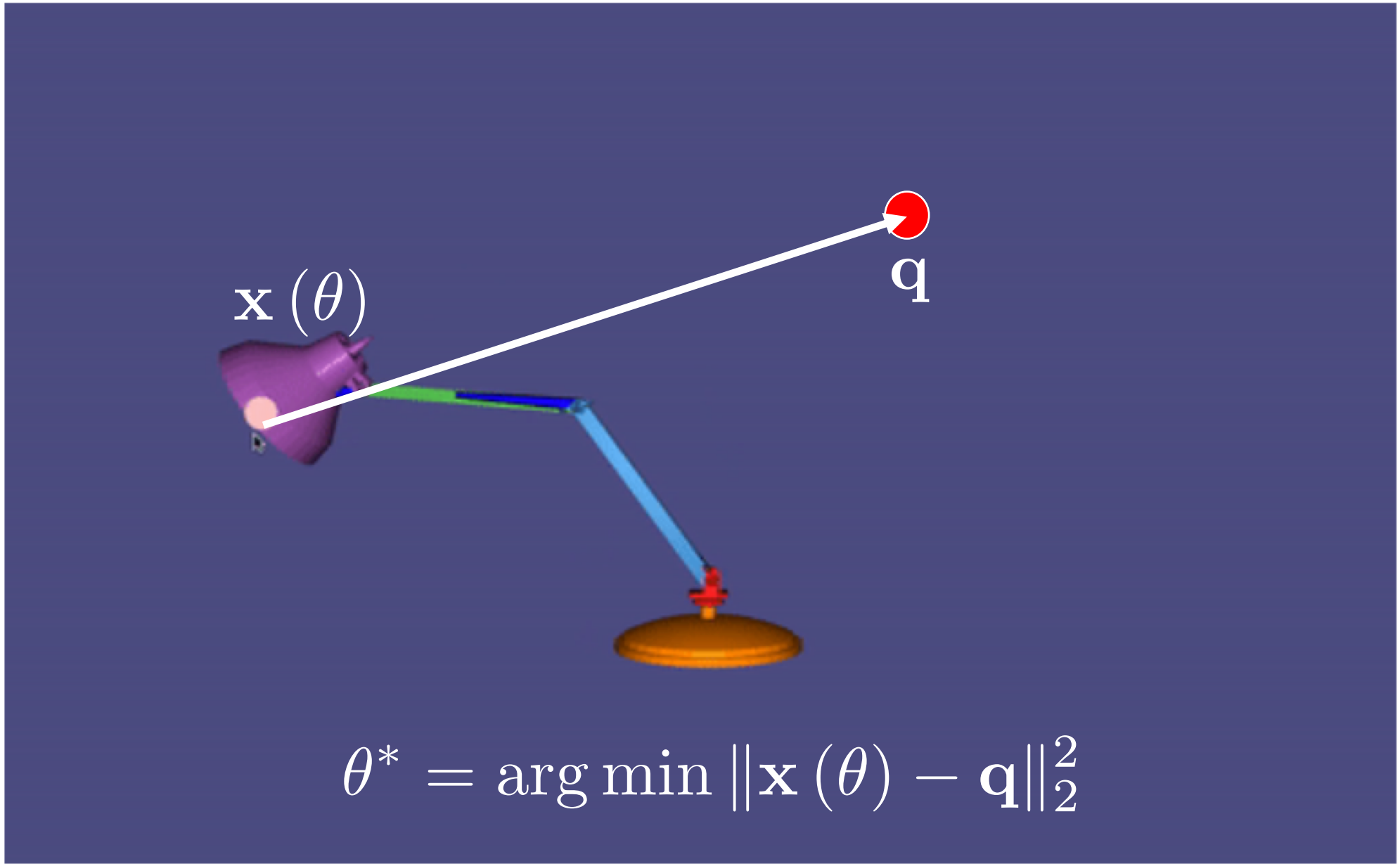
Inverse Kinematics



Inverse Kinematics



Inverse Kinematics



Gradient Descent

Recall that the gradient of a function is given by

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial \mathbf{x}_1} \quad \frac{\partial f}{\partial \mathbf{x}_2} \quad \cdots \quad \frac{\partial f}{\partial \mathbf{x}_n} \right)$$

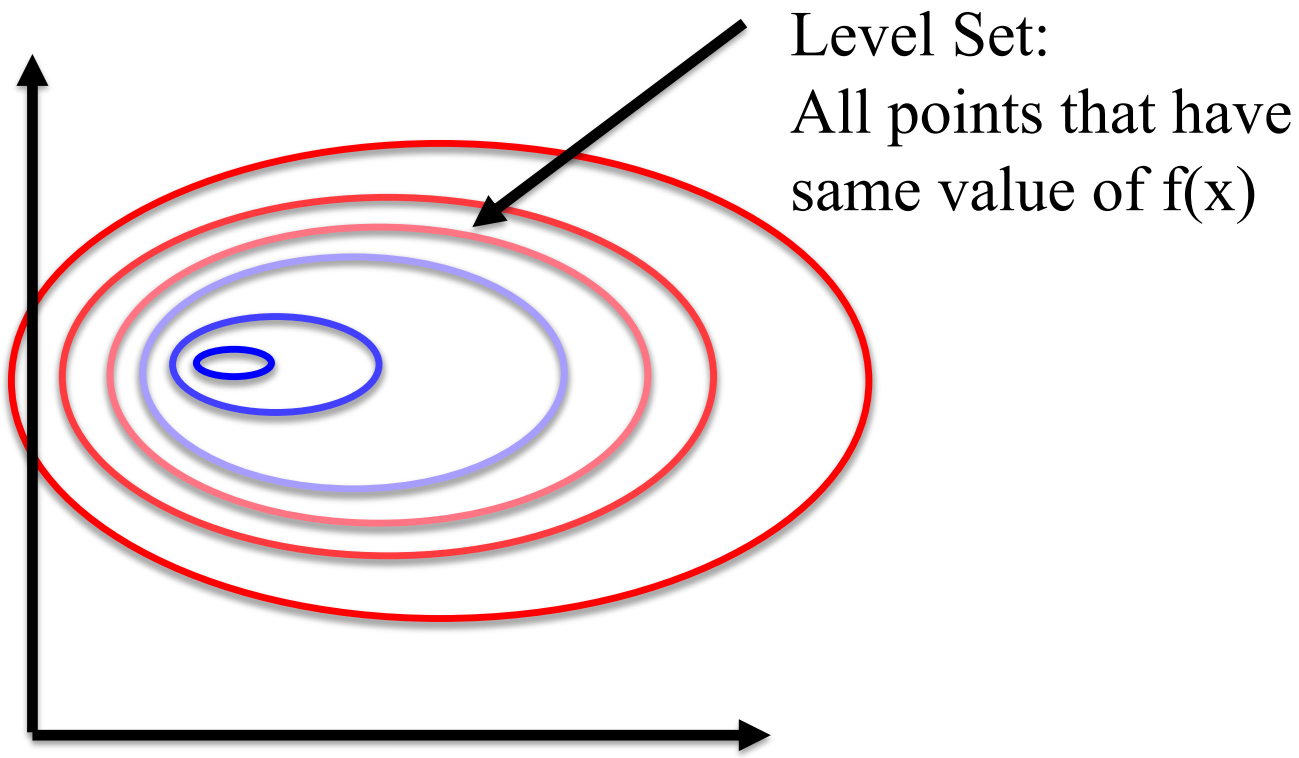
Gradient Descent

Recall that the gradient of a function is given by

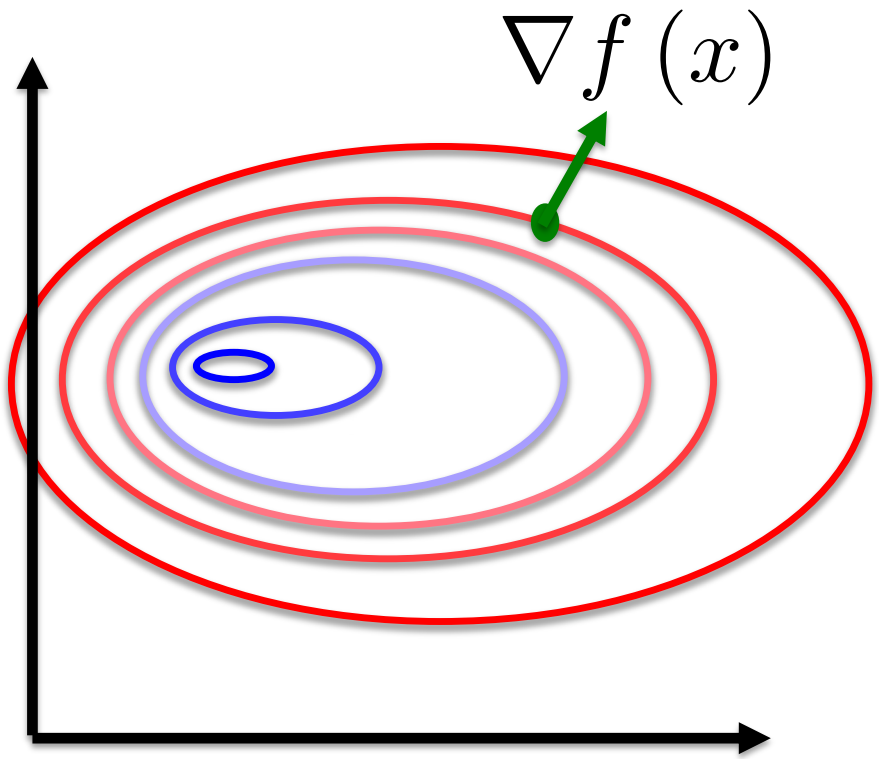
Points in direction of maximum ascent

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial \mathbf{x}_1} \quad \frac{\partial f}{\partial \mathbf{x}_2} \quad \cdots \quad \frac{\partial f}{\partial \mathbf{x}_n} \right)$$

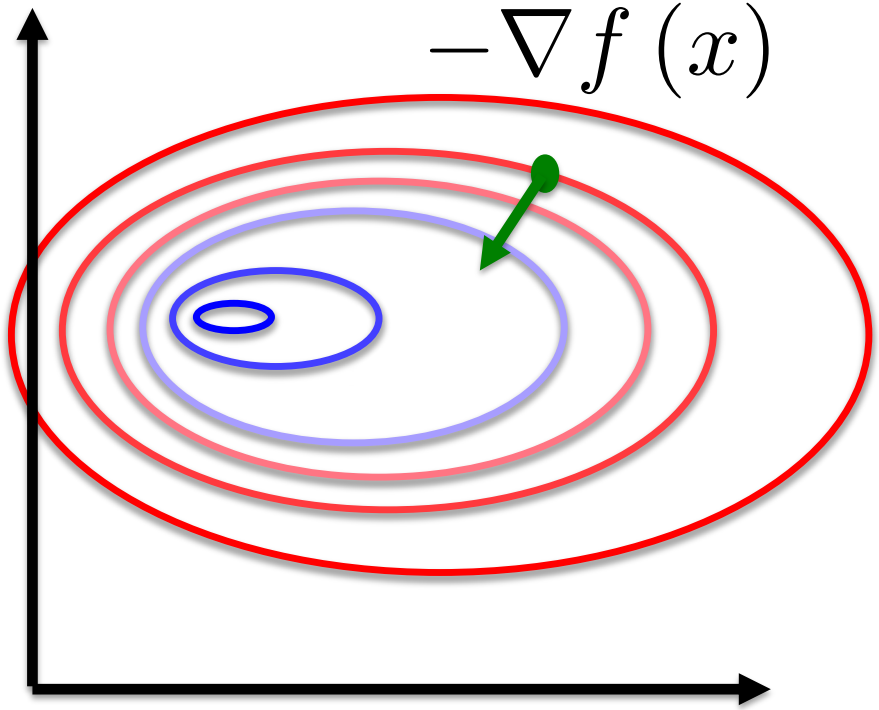
An Aside: Level Sets



Gradient Descent



Gradient Descent

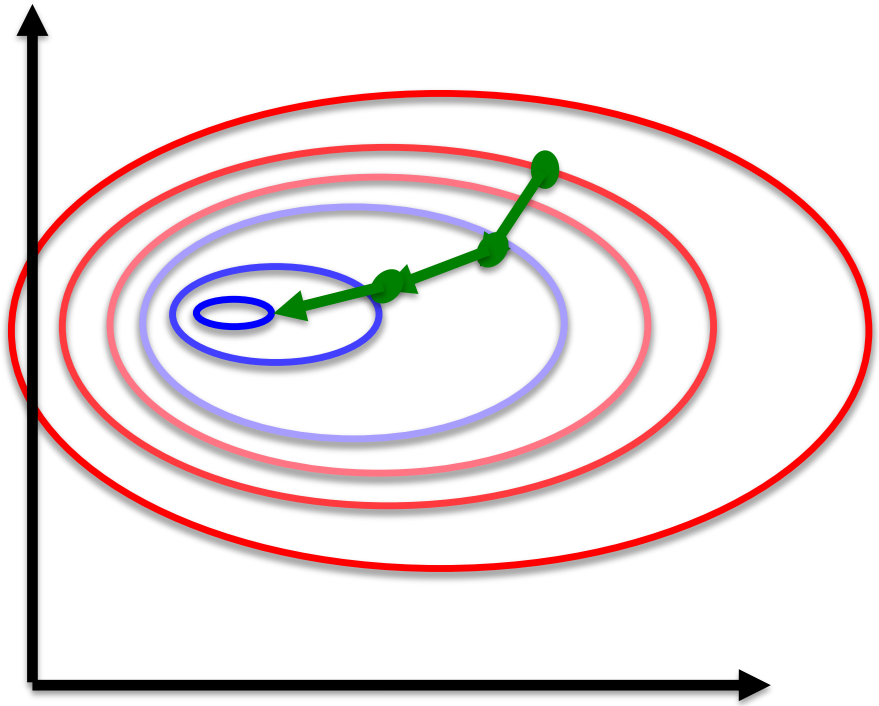


Simple Gradient Descent Algorithm

While not at an optimal point

- Compute the gradient at current point (x)
- Move to new point $x = x - h\nabla f(x)$

Gradient Descent

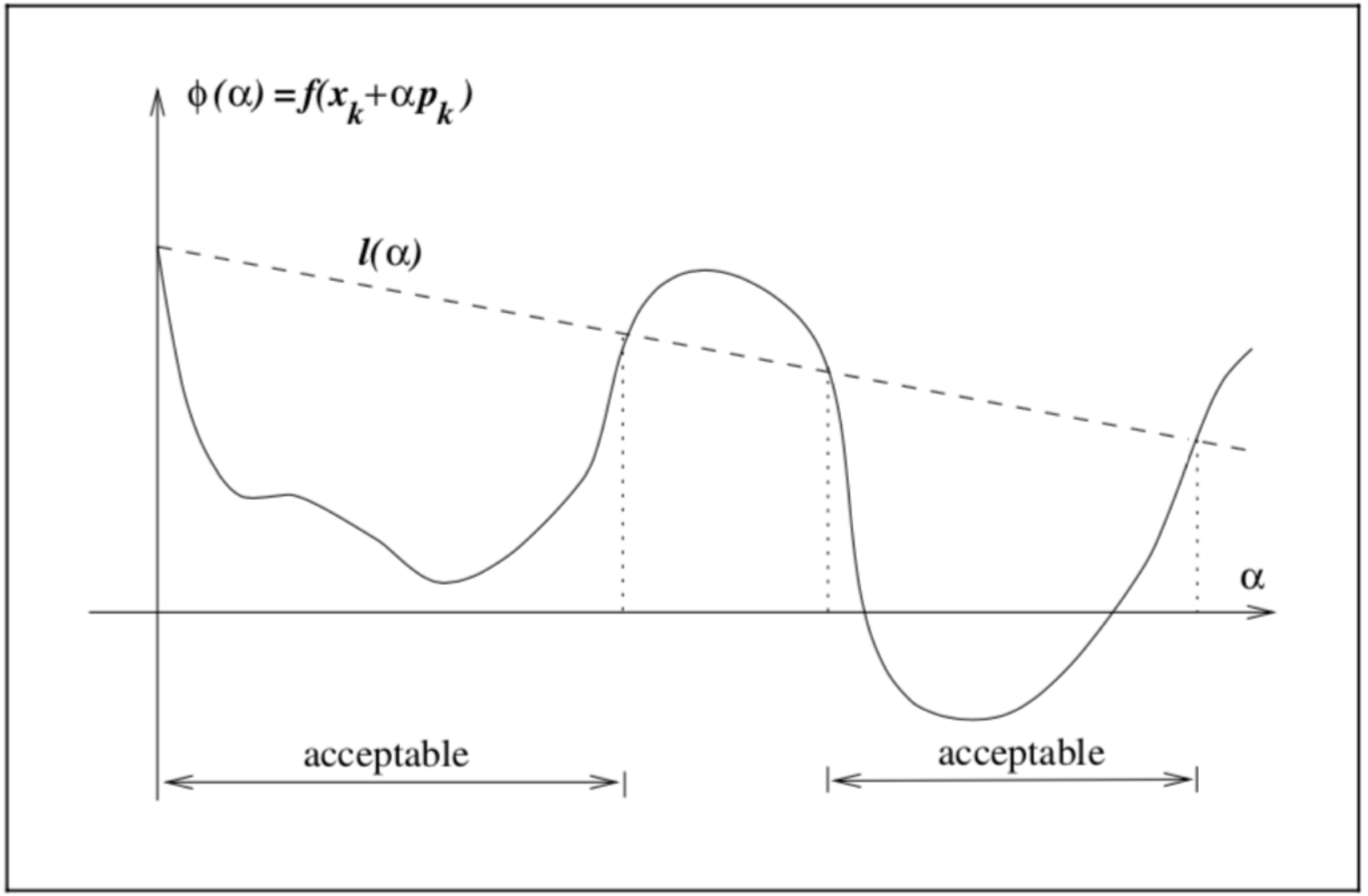


Simple Gradient Descent Algorithm

While not at an optimal point

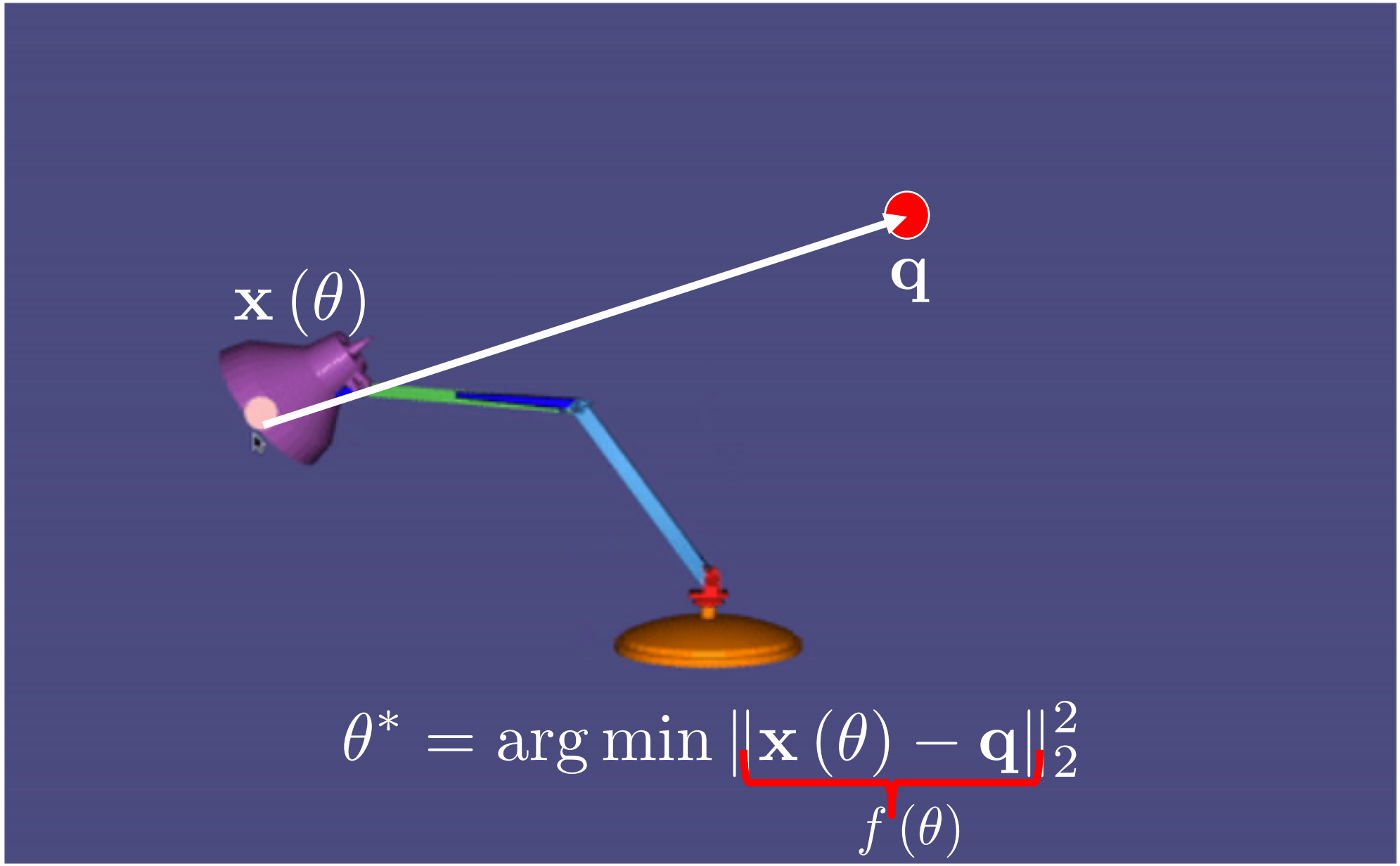
- Compute the gradient at current point (x)
- Move to new point $x = x - h\nabla f(x)$

When Good Optimizations Go Bad

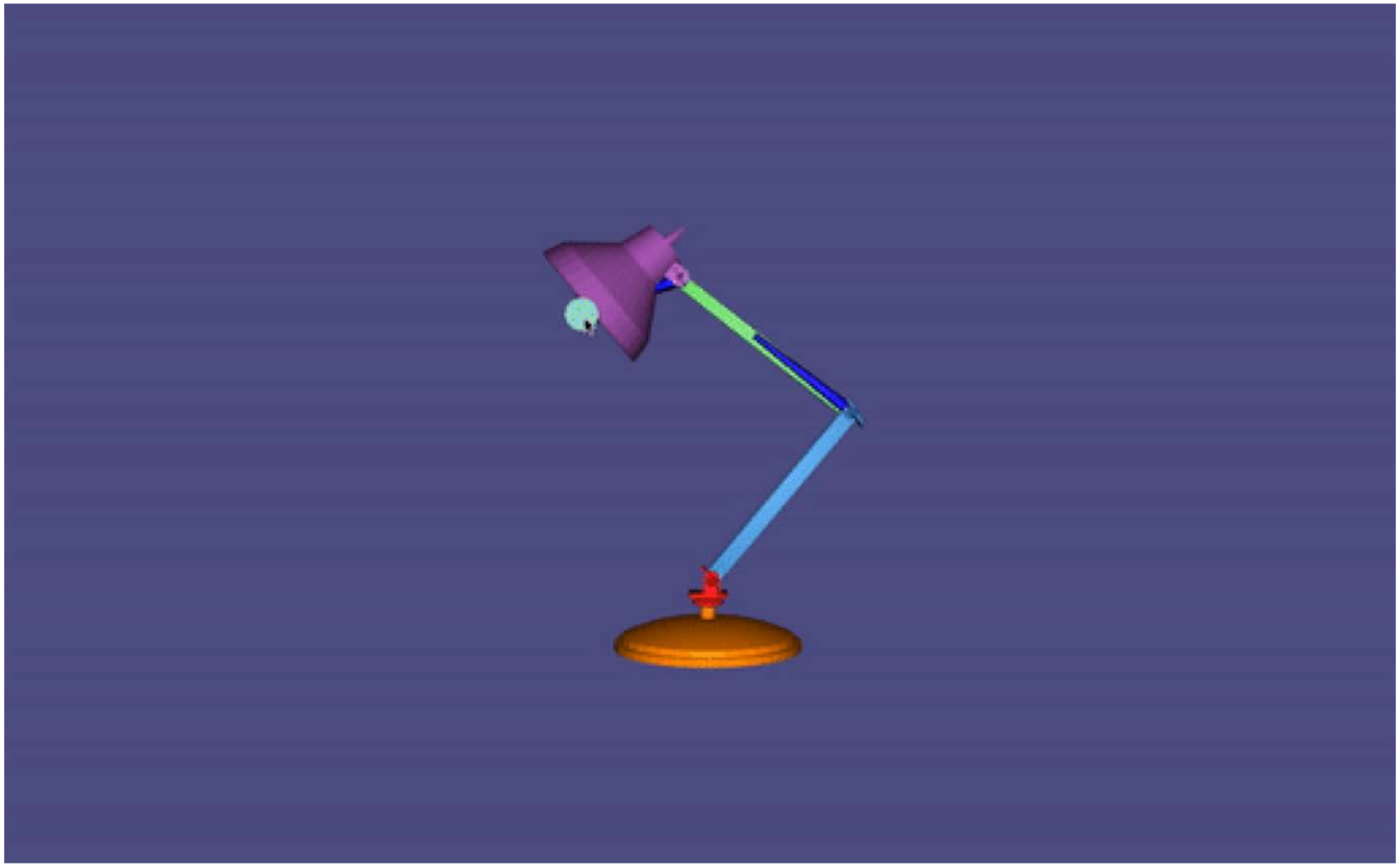


Numerical Optimization – Nocedal and Wright, pg. 33

Inverse Kinematics



Inverse Kinematics



Done for Today

Office hours: Right now! BA5268