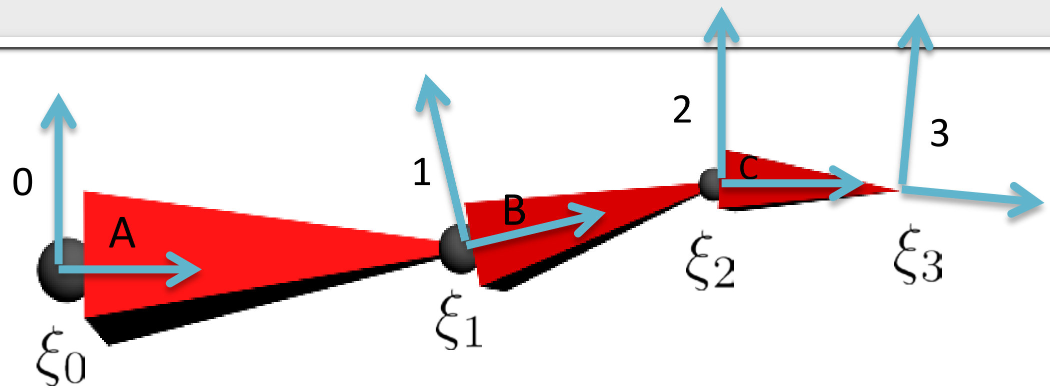
Mathematical Foundations of Computer Graphics and Vision

Inverse Kinematics II and Motion Capture

Luca Ballan

Comparison



Fake exponential map

$$(\omega, t) \rightarrow \begin{bmatrix} e^{\widehat{\omega}} & t \\ 0 & 1 \end{bmatrix}$$

- t is equal to the length of the bone
- computing the derivative on the angle is easy

Real exponential map

$$(\omega, v) \rightarrow \begin{bmatrix} e^{\widehat{\omega}} & \frac{1}{\|\omega\|}(I - e^{\widehat{\omega}})(\omega \times v) + \frac{\omega\omega^T}{\|\omega\|}v \\ 0 & 1 \end{bmatrix}$$

- the meaning of v is not intuitive

However

- this incorporates the real concept of geodesic
- interpolation/averaging has to be done in this space

Special Euclidean group SE(3)

$$\xi = (\omega, v) \xrightarrow{\exp} \begin{bmatrix} e^{\widehat{\omega}} & \frac{1}{\|\omega\|}(I - e^{\widehat{\omega}})(\omega \times v) + \frac{\omega\omega^T}{\|\omega\|} v \\ 0 & 1 \end{bmatrix} = e^{\widehat{\xi}} \in SE(3)$$

Twist

Angle/axis representation of the rotation $\widehat{\omega} \in so(3)$

- **Screw motion**: rotation along an axis + a translation along the same axis.

- **Proposition:** Any rigid transformation in SE(3) can be expressed as a rotation about an axis combined with a translation parallel to that axis.

Special Euclidean group SE(3)

$$\xi = (\omega, v)$$

Twist

hat

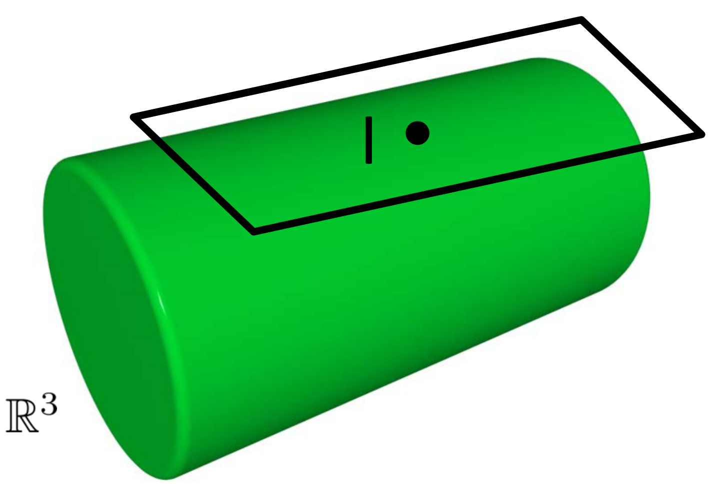
$$\widehat{\xi} = \begin{bmatrix} \widehat{\omega} & v \\ 0 & 0 \end{bmatrix} \in se(3)$$

Tangent space of SE(3)
at the identity

exp

$$\begin{bmatrix} e^{\widehat{\omega}} & \frac{1}{\|\omega\|}(I - e^{\widehat{\omega}})(\omega \times v) + \frac{\omega\omega^T}{\|\omega\|} v \\ 0 & 1 \end{bmatrix} = e^{\widehat{\xi}} \in SE(3)$$

$$\exp(M) = \sum_{k=0}^{\infty} \frac{1}{k!} M^k$$



$se(3)$

$SE(3)$
$= SO(3) \times \mathbb{R}^3$

Properties

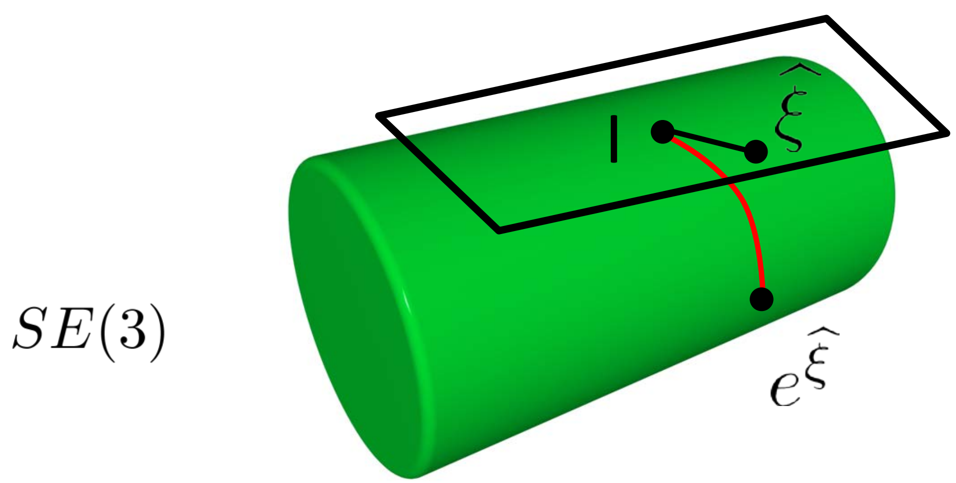
$e^{\widehat{0}} = I$	**Identity**	
$e^{-X} = (e^{X})^{-1}$	**Inverse**	(basic property of exp map)
$e^{X+Y} \neq e^{X}e^{Y}$	**in general not “Linear”**	**(like in so(3))**
$\partial e^{X} = \partial X e^{X} = e^{X} \partial X$	**Derivative**	

Metric on SE(3)

$$d_R(M_1, M_2) = \frac{1}{\sqrt{2}} \| \log(M1^{-1} M2) \|_F$$

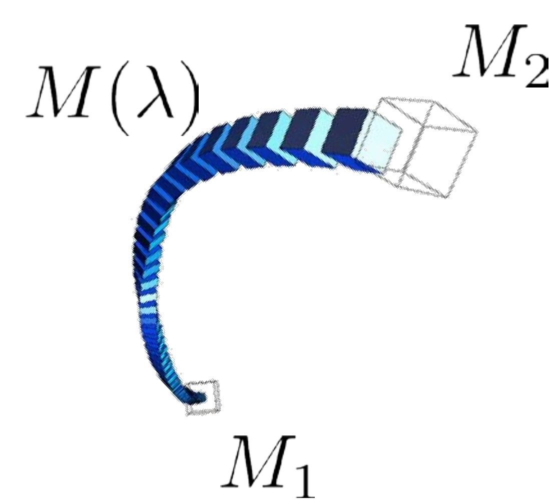
Riemannian/Geodesic/Angle metric
(= the length of the geodesic
connecting M_1 and M_2)

- Interpolation and averaging can be performed in the space of rigid transformations also



$$M(\lambda) = M_1 e^{\lambda \log(M_1^{-1} M_2)}$$

SLERP
(spherical linear
interpolation)

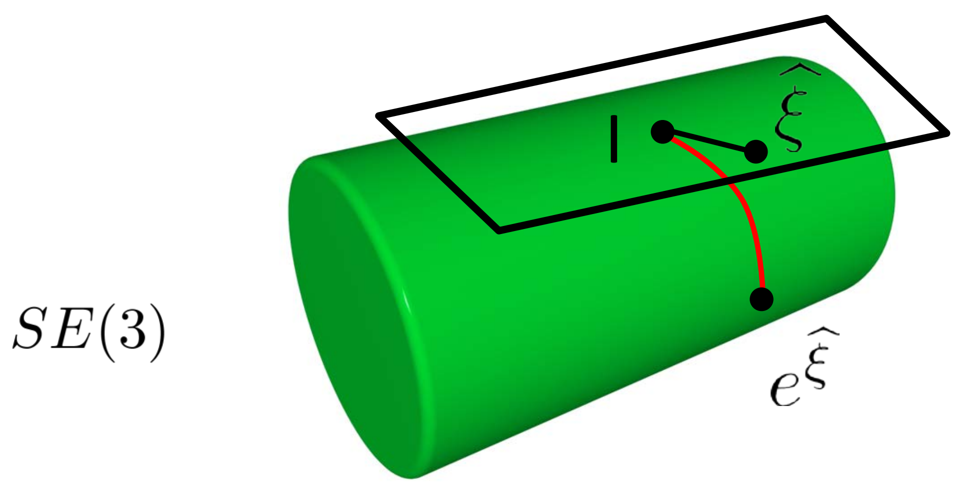


Metric on SE(3)

$$d_R(M_1, M_2) = \frac{1}{\sqrt{2}} \| \log(M1^{-1}M2) \|_F$$

Riemannian/Geodesic/Angle metric
(= the length of the geodesic
connecting M_1 and M_2)

- Interpolation and averaging can be performed in the space of rigid transformations also



$$M(\lambda) = M_1 e^{\lambda \log(M_1^{-1} M_2)}$$

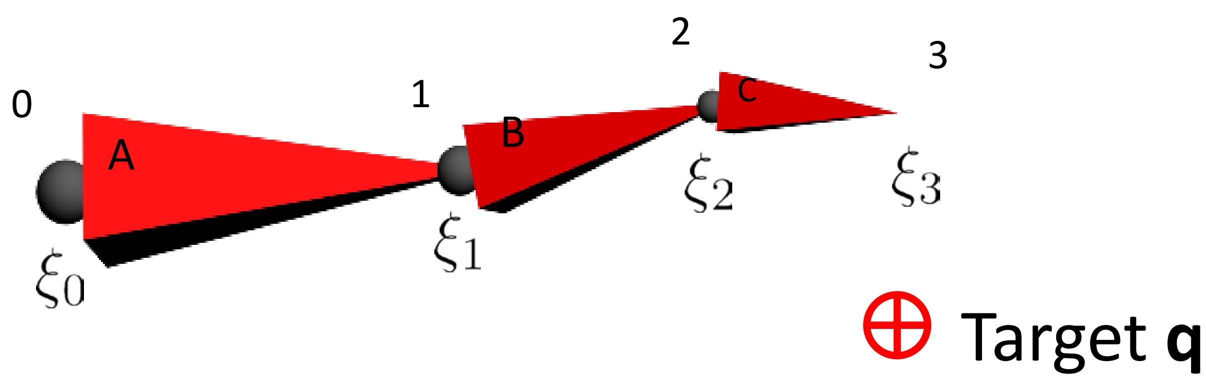
SLERP
(spherical linear
interpolation)

$$\underset{M \in SE(3)}{\operatorname{argmin}} \sum_{i=1}^{n} d_R(M, M_i)^2$$

Fréchet mean

A note on Interpolation

Inverse Kinematics



Newton's method

- let $\overline{x}$ be the current estimate for the solution
- compute the Taylor expansion of $p(x)$around $\overline{x}$

$$p(x + \Delta x) = p(\overline{x}) + Jp(\overline{x})\Delta x + \ldots$$

$$\arg\min \|\overbrace{p(\overline{x}) + Jp(\overline{x})\Delta x} - q\|$$

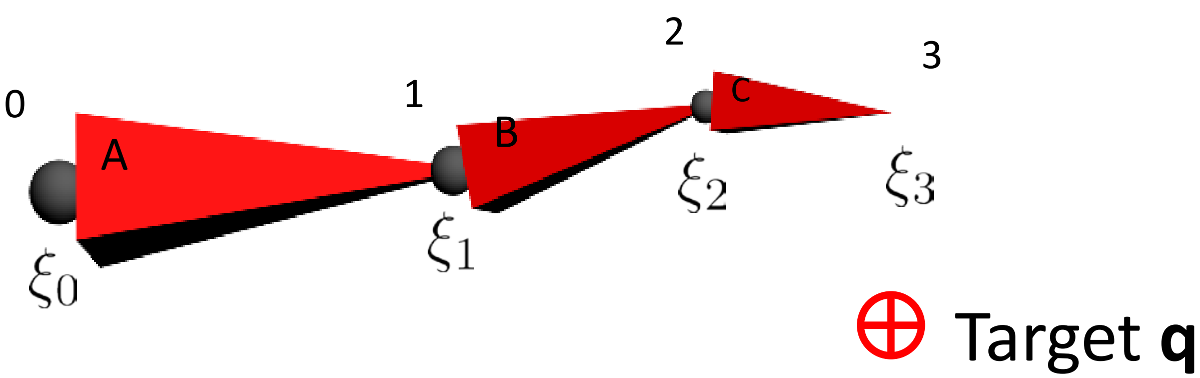
$$\Updownarrow$$

$$p(\overline{x}) + Jp(\overline{x})\Delta x - q = 0$$

$$\Updownarrow$$

$$\Delta x = Jp(\overline{x})^{\dagger}(q - p(\overline{x}))$$

Inverse Kinematics



Newton's method

- let $\overline{x}$ be the current estimate for the solution
- compute the Taylor expansion of $p(x)$around $\overline{x}$

$$p(x + \Delta x) = p(\overline{x}) + Jp(\overline{x})\Delta x + \dots$$

$$\arg\min \|\overbrace{p(\overline{x}) + Jp(\overline{x})\Delta x} - q\|$$

$$\Updownarrow$$

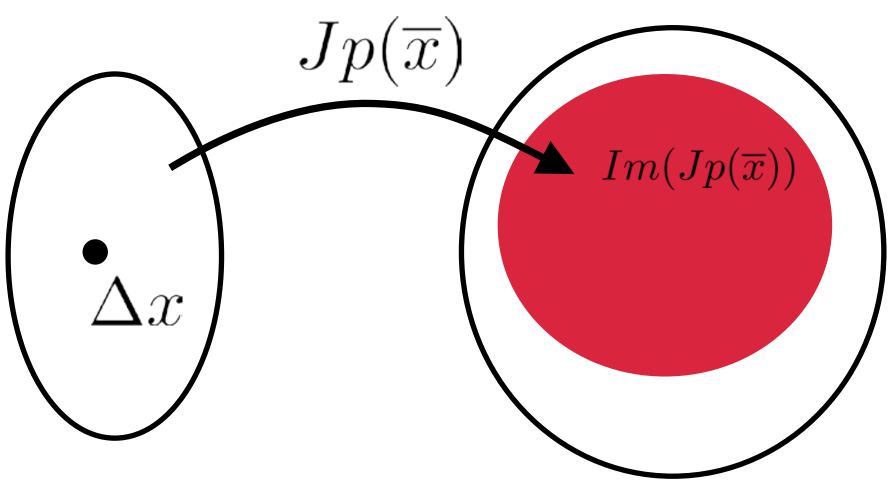
$$p(\overline{x}) + Jp(\overline{x})\Delta x - q = 0$$

$$\Updownarrow$$

Find Δx such that $Jp(\overline{x})\Delta x = (q - p(\overline{x}))$

Inverses

Find Δx such that $Jp(\overline{x})\Delta x = (q - p(\overline{x}))$



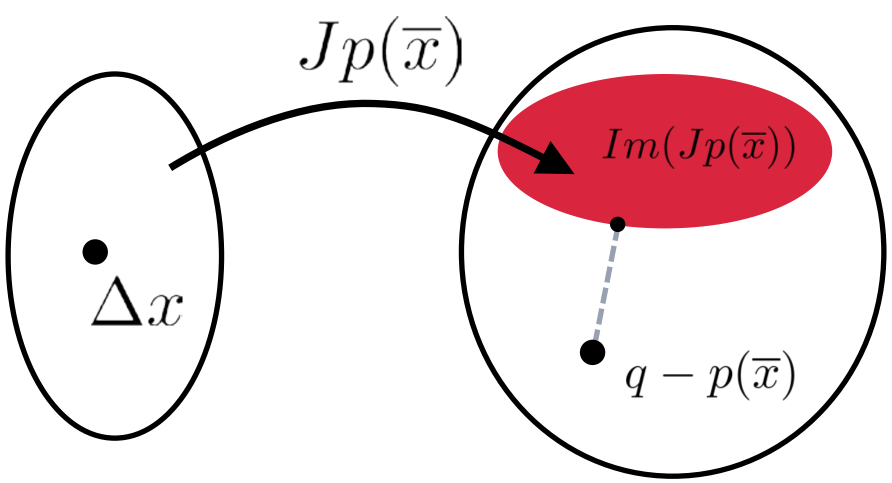
$Jp(\overline{x})$ is not injective (in general)
(multiple inverse exists for each point of the image)

If it is injective, then the left-Inverse exists

$$\Delta x = Jp(\overline{x})^{-1}_{left}(q - p(\overline{x}))$$

Inverses

Find Δx such that $Jp(\overline{x})\Delta x = (q - p(\overline{x}))$



$Im(Jp(\overline{x}))$ might not contain the solution

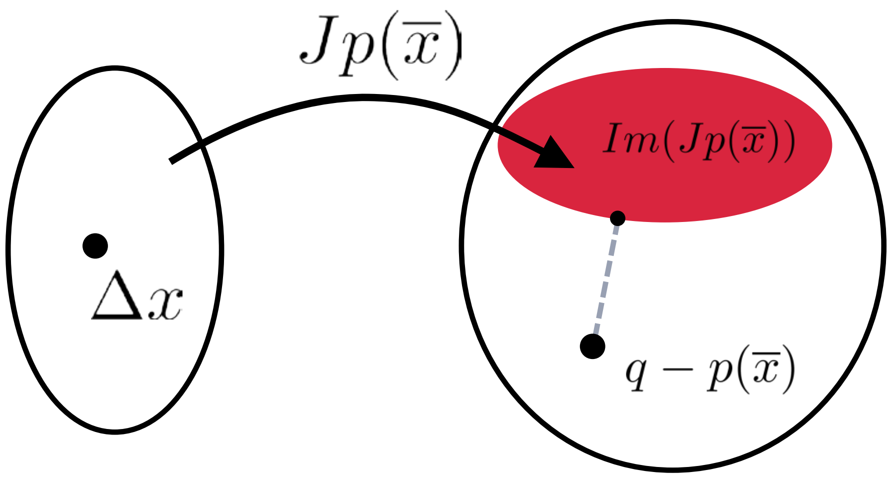
Better to find the Δx such that

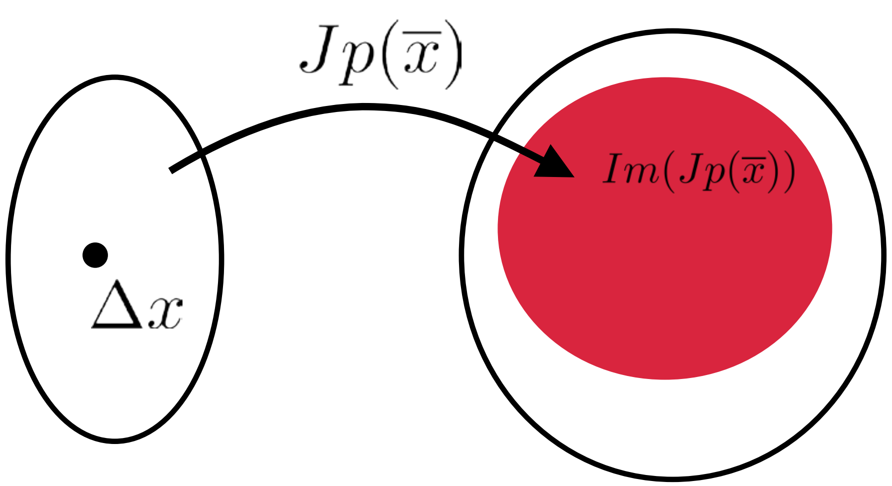
$$\|Jp(\overline{x})\Delta x - (q - p(\overline{x}))\|$$

is minimized

Inverses

Find Δx such that $\|Jp(\overline{x})\Delta x - (q - p(\overline{x}))\|$ is minimized,





To force the unicity of the solution

we force Δx to be the one with minimum norm

$$\Delta x = Jp(\overline{x})^{\dagger}(q - p(\overline{x}))$$

Singularities

- Pseudo inverse works well in many cases but not near a singularity (singular values close to 0)

- Make the constraint on the norm of Δx soft, not hard as before

$$\arg\min \|Jp(\overline{x})\Delta x - q + p(\overline{x})\|^2 + \lambda^2 \|\Delta x\|^2$$

- the solution of this is the same as solving for

$$(Jp(\overline{x})^T Jp(\overline{x}) + \lambda I)\Delta x = Jp(\overline{x})^T (q - p(\overline{x}))$$

  - this is always non singular if **the damping factor** λ is correctly chosen

- The Newton's method with this damping step is known as the **Damped Least Square method**, or as the **Levenberg-Marquardt algorithm**.

Heuristic approaches

- **Cyclic Coordinate Descent** is an **alternating optimization approach** where only one coordinate at a time is optimized.

$$\arg\min \|p(\xi_0, \xi_1, \xi_2, \xi_3) - q\|$$

- Let $(\xi_0^t, \xi_1^t, \xi_2^t, \xi_3^t)$ be the estimate of the solution at iteration t

$$\xi_0^{t+1} = \arg\min_x \|p(\mathbf{x}, \xi_1^t, \xi_2^t, \xi_3^t) - q\|$$
$$\xi_1^{t+1} = \arg\min_x \|p(\xi_0^{t+1}, \mathbf{x}, \xi_2^t, \xi_3^t) - q\|$$
$$\xi_2^{t+1} = \arg\min_x \|p(\xi_0^{t+1}, \xi_1^{t+1}, \mathbf{x}, \xi_3^t) - q\|$$
$$\xi_3^{t+1} = \arg\min_x \|p(\xi_0^{t+1}, \xi_2^{t+1}, \xi_1^{t+1}, \mathbf{x}) - q\|$$

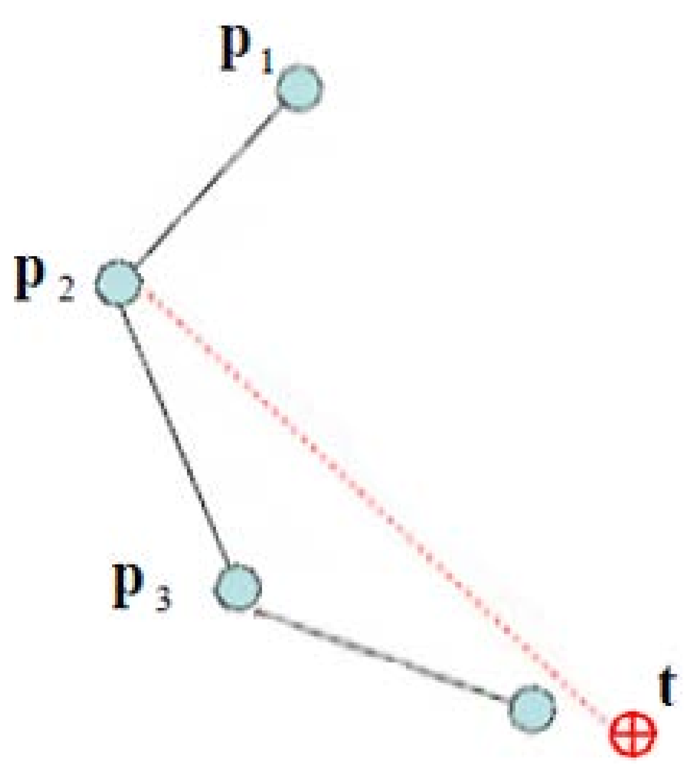
Heuristic approaches

- **Cyclic Coordinate Descent** is an **alternating optimization approach** where only one coordinate at a time is optimized.

$$\arg\min \|p(\xi_0, \xi_1, \xi_2, \xi_3) - q\|$$

- Let $(\xi_0^t, \xi_1^t, \xi_2^t, \xi_3^t)$ be the estimate of the solution at iteration t

$$\xi_0^{t+1} = \arg\min_x \|p(\mathbf{x}, \xi_1^t, \xi_2^t, \xi_3^t) - q\|$$
$$\xi_1^{t+1} = \arg\min_x \|p(\xi_0^{t+1}, \mathbf{x}, \xi_2^t, \xi_3^t) - q\|$$
$$\xi_2^{t+1} = \arg\min_x \|p(\xi_0^{t+1}, \xi_1^{t+1}, \mathbf{x}, \xi_3^t) - q\|$$
$$\xi_3^{t+1} = \arg\min_x \|p(\xi_0^{t+1}, \xi_2^{t+1}, \xi_1^{t+1}, \mathbf{x}) - q\|$$



Heuristic approaches

- **Cyclic Coordinate Descent** is an **alternating optimization approach** where only one coordinate at a time is optimized.

$$\arg\min \|p(\xi_0, \xi_1, \xi_2, \xi_3) - q\|$$

- Let $(\xi_0^t, \xi_1^t, \xi_2^t, \xi_3^t)$ be the estimate of the solution at iteration t

$$\xi_0^{t+1} = \arg\min_x \|p(\mathbf{x}, \xi_1^t, \xi_2^t, \xi_3^t) - q\|$$

solving for a single twist is a very easy problem

$$e^{\widehat{\xi_0}} e^{\widehat{\xi_1}} e^{\widehat{\xi_2}} e^{\widehat{\xi_3}} p = q$$

Heuristic approaches

$$e^{\widehat{\xi_0}} e^{\widehat{\xi_1}} e^{\widehat{\xi_2}} e^{\widehat{\xi_3}} p = q$$

$$\left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right) e^{\widehat{\xi_2}} \left(e^{\widehat{\xi_3}} p\right) = q$$

$$\begin{aligned}
e^{\widehat{\xi_2}} \tilde{p} &= \left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right)^{-1} q \\
e^{\widehat{\omega_2}} \tilde{p} + T_2 &= \left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right)^{-1} q \\
e^{\widehat{\omega_2}} \tilde{p} &= \left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right)^{-1} q - T_2 \\
e^{\widehat{\omega_2}} \tilde{p} &= \tilde{q}
\end{aligned}$$

*

Heuristic approaches

$$e^{\widehat{\xi_0}} e^{\widehat{\xi_1}} e^{\widehat{\xi_2}} e^{\widehat{\xi_3}} p = q$$

$$\left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right) e^{\widehat{\xi_2}} \left(e^{\widehat{\xi_3}} p\right) = q$$

$$\begin{aligned}
e^{\widehat{\xi_2}} \tilde{p} &= \left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right)^{-1} q \\
e^{\widehat{\omega_2}} \tilde{p} + T_2 &= \left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right)^{-1} q \\
e^{\widehat{\omega_2}} \tilde{p} &= \left(e^{\widehat{\xi_0}} e^{\widehat{\xi_1}}\right)^{-1} q - T_2 \\
e^{\widehat{\omega_2}} \tilde{p} &= \tilde{q}
\end{aligned}$$

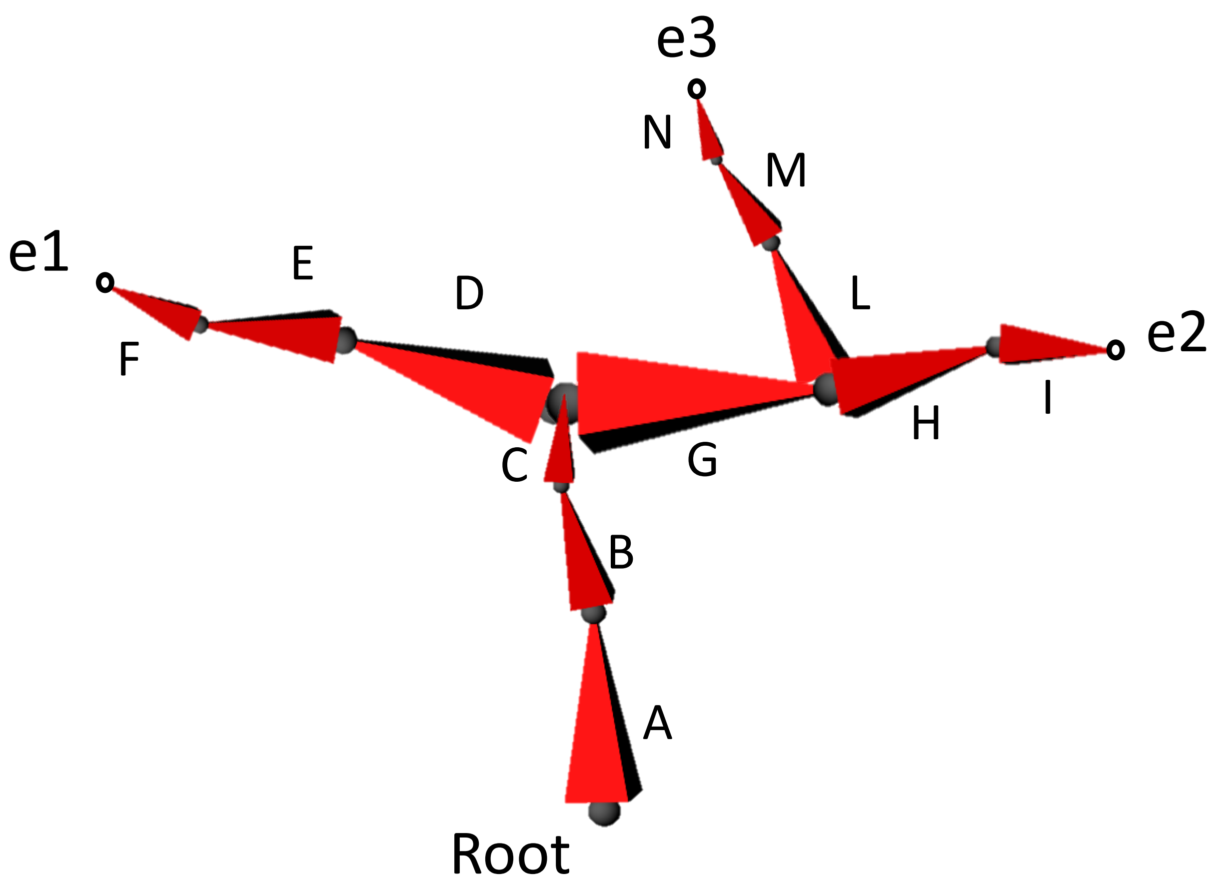
$$\left\{ \begin{aligned}
\tilde{p}\tilde{q}^T &= U \Sigma V^T \\
R &= V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^T
\end{aligned} \right.$$

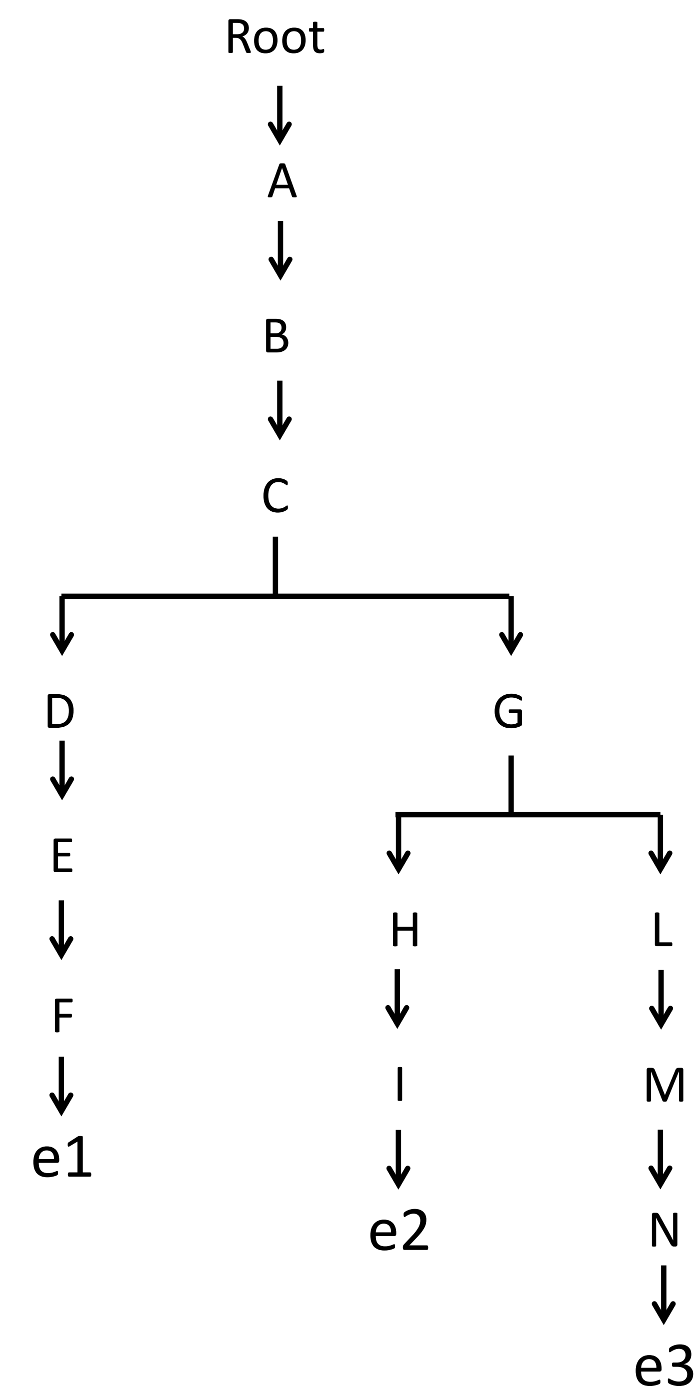
Content

- Inverse Kinematics
- **Kinematic Trees/Graphs**
- Pose Estimation/Motion Capture

Kinematic Tree

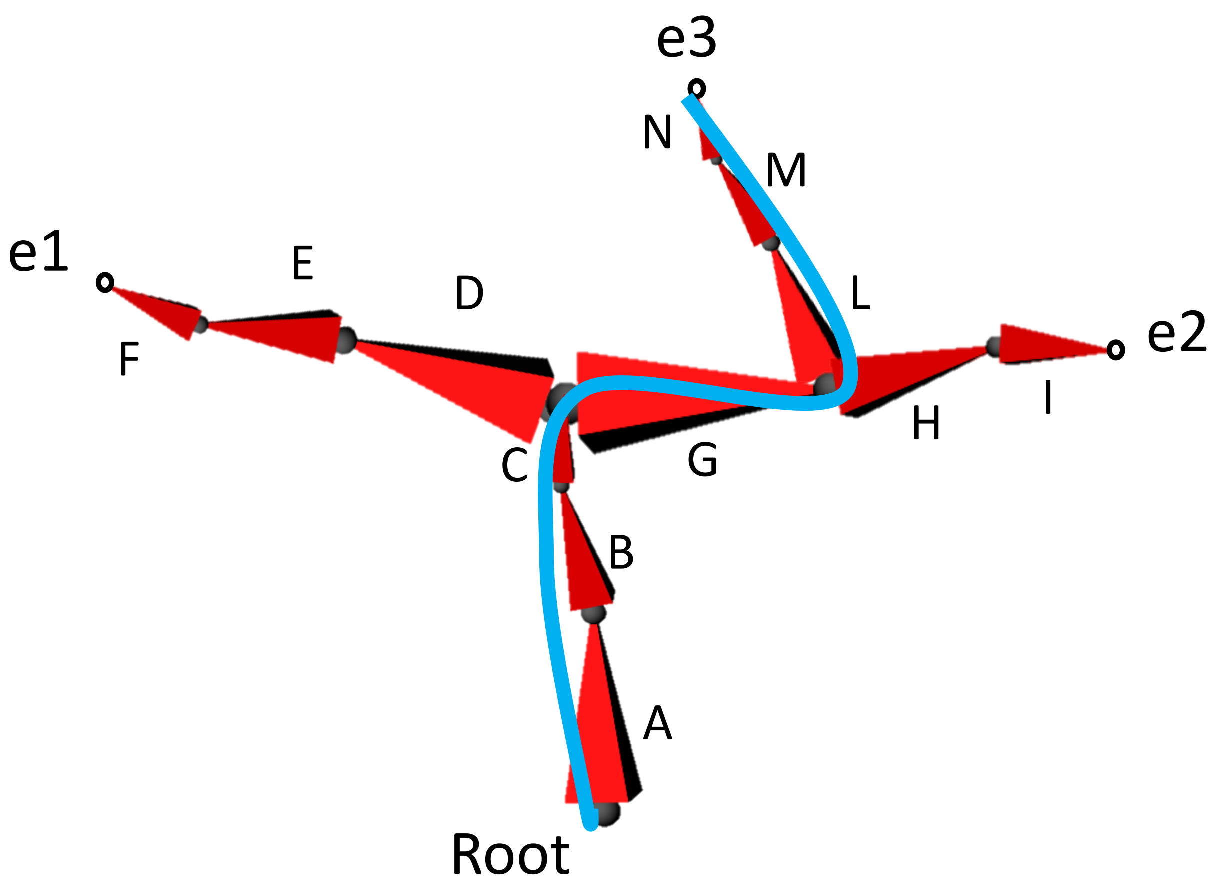
- a **kinematic tree** is a tree of rigid transformations

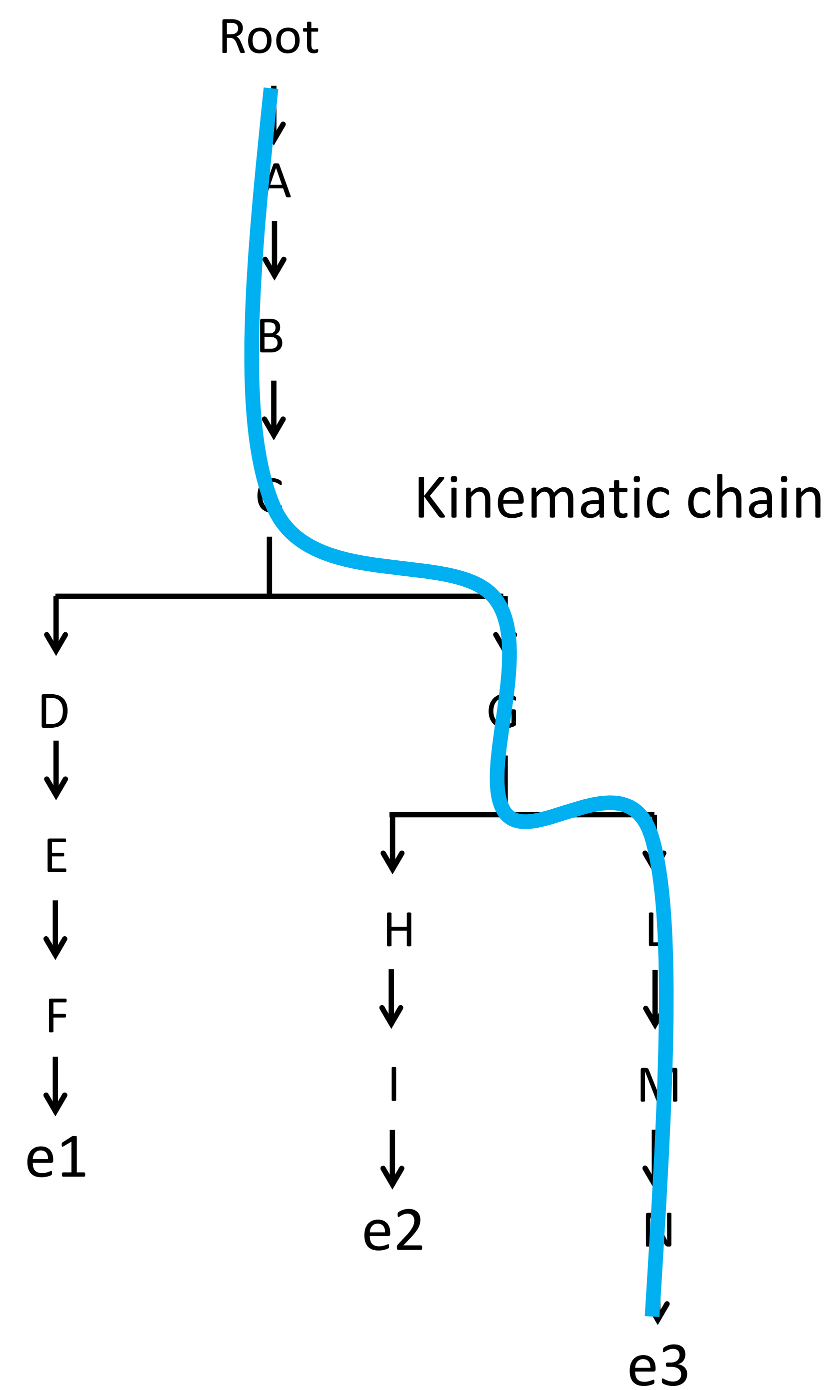




Kinematic Tree

- a **kinematic tree** is a tree of rigid transformations

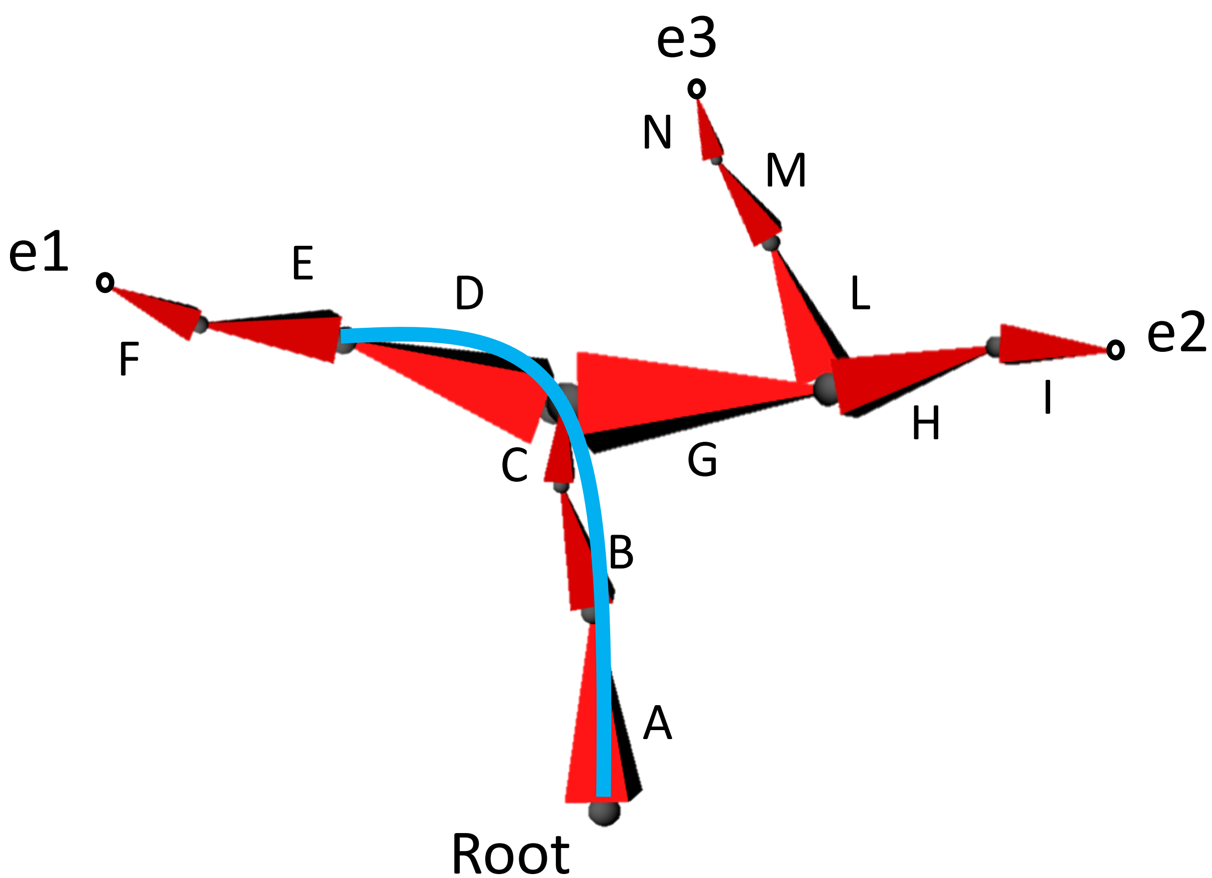


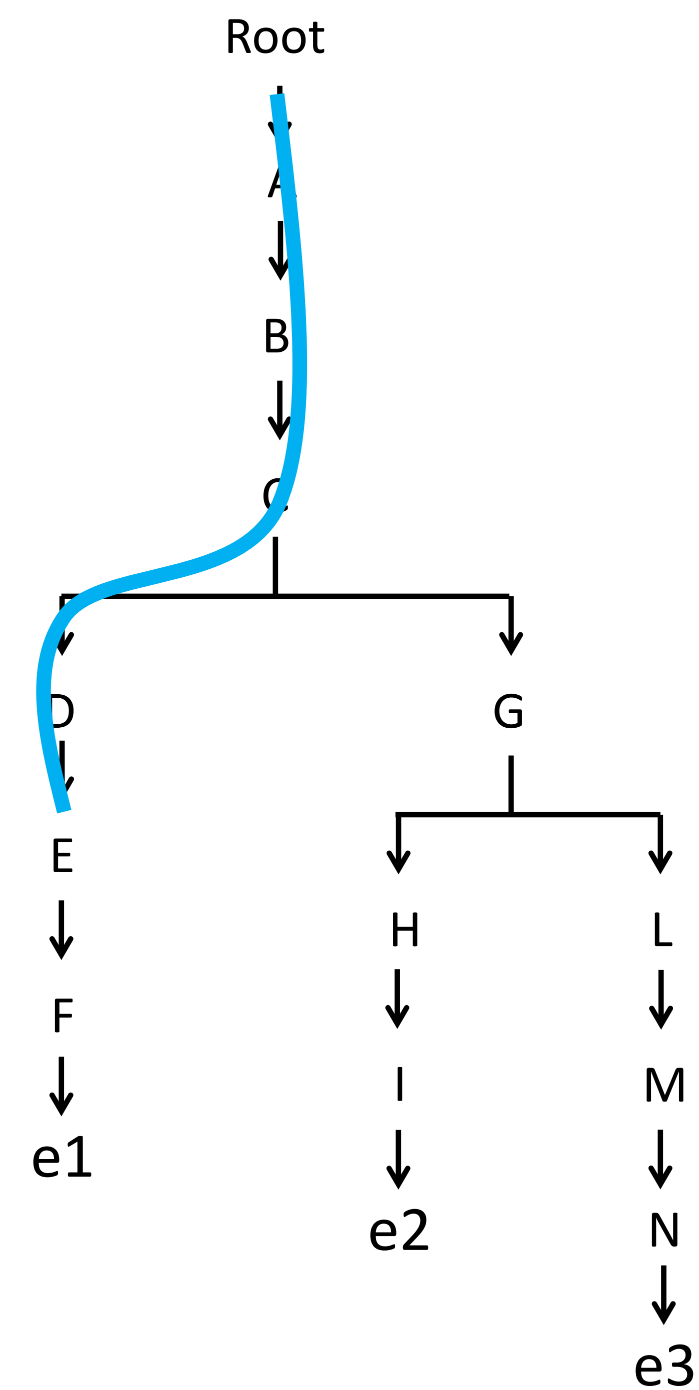


- each path in a **kinematic tree** from the root to <u>any</u> other node is a **kinematic chain**

Kinematic Tree

- a **kinematic tree** is a tree of rigid transformations

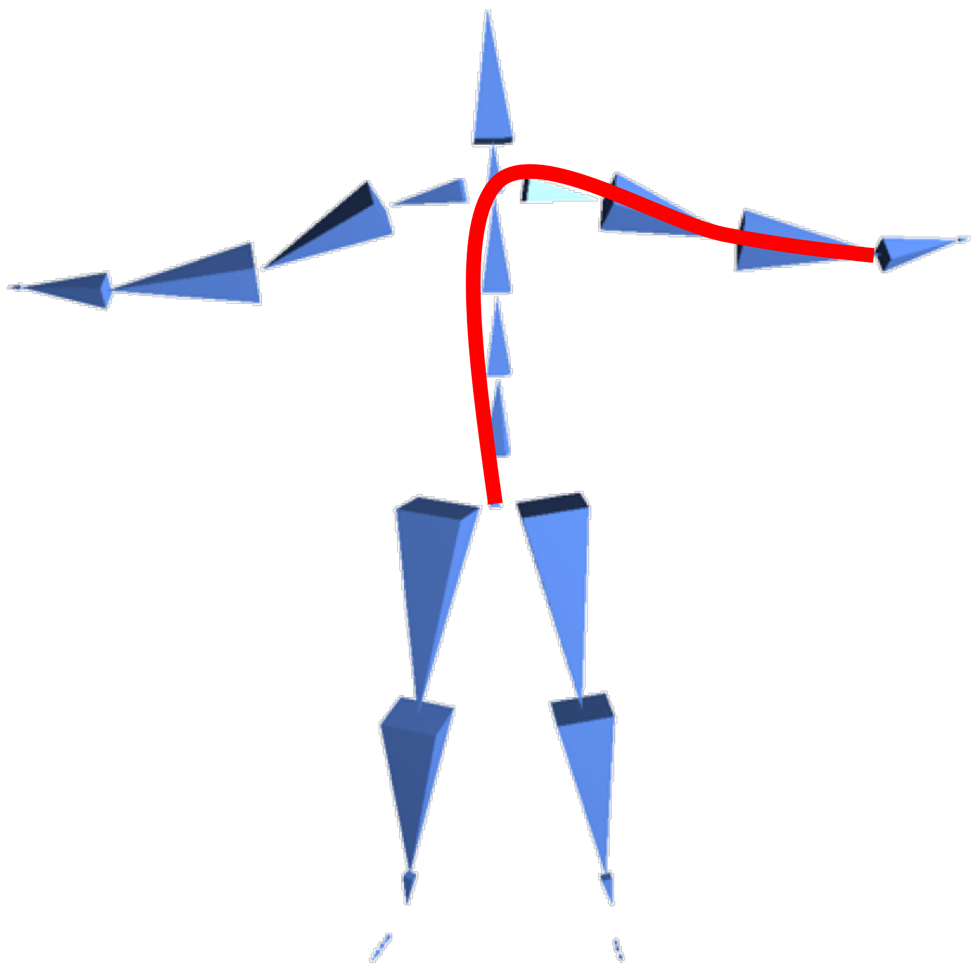


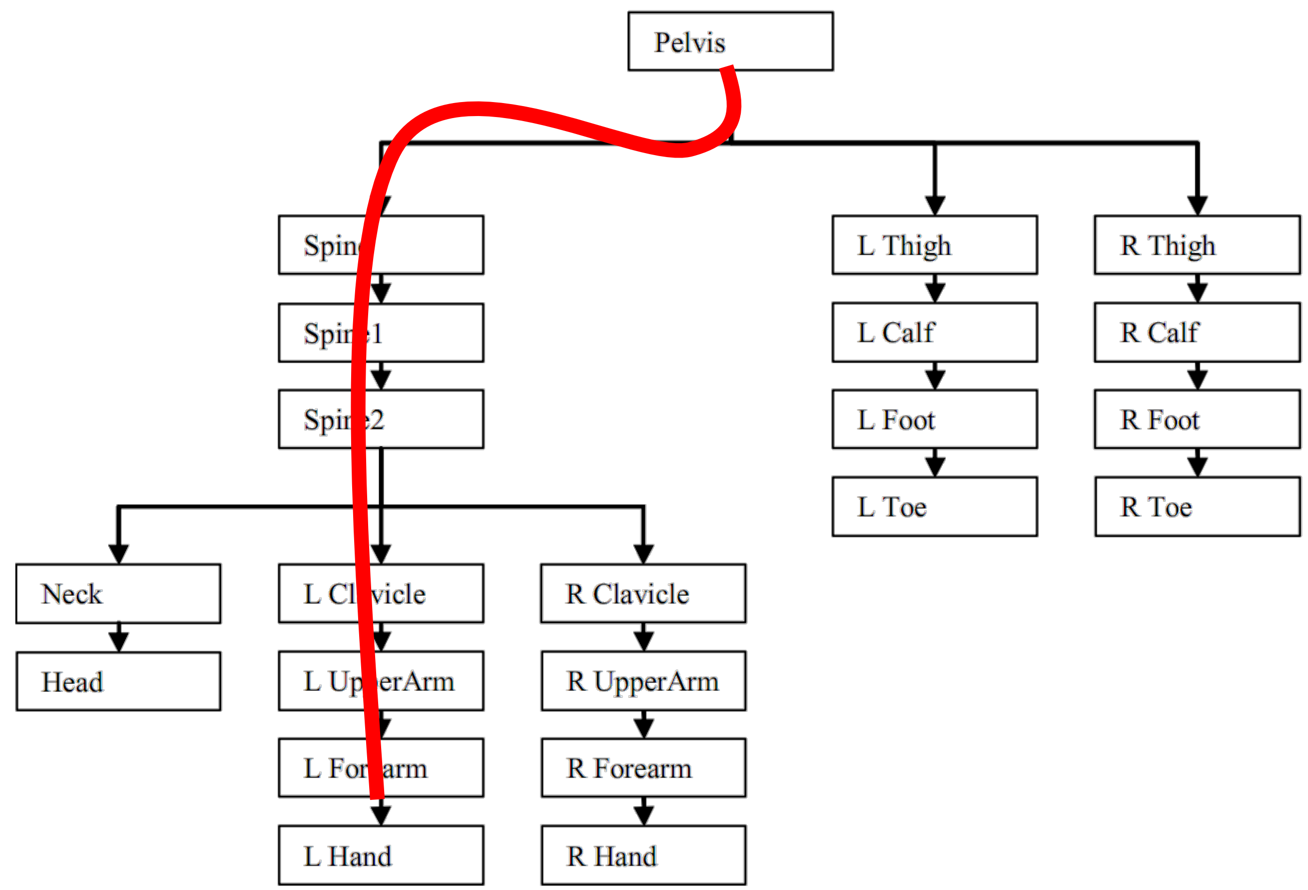


- each path in a **kinematic tree** from the root to <u>any</u> other node is a **kinematic chain**

Kinematic Tree: Human Skeleton

- a **kinematic tree** is a tree of rigid transformations



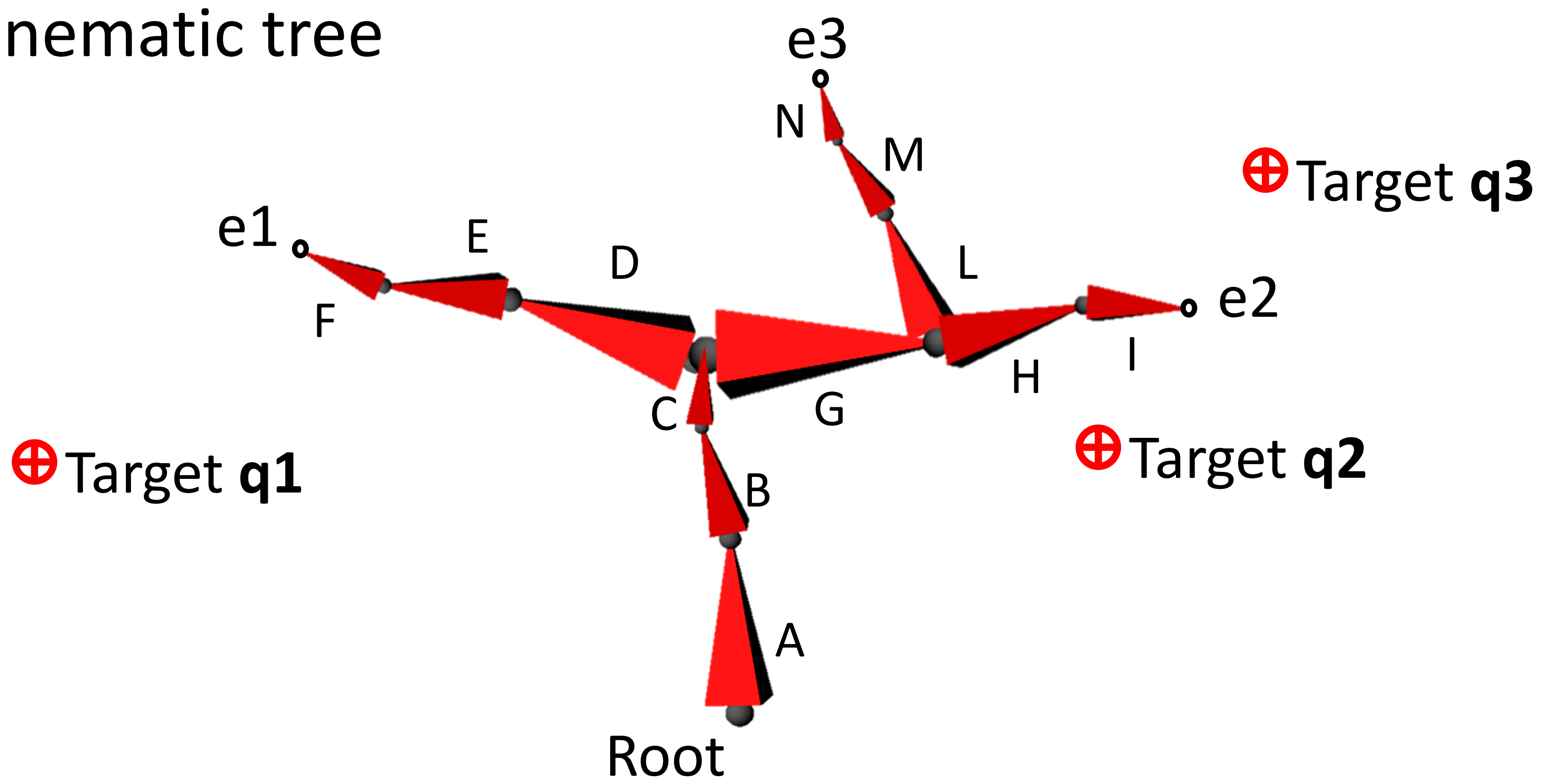


- each path in a **kinematic tree** from the root to <u>any</u> other node is a **kinematic chain**

- The pelvis is typically denoted as the root of the tree (but this is only a convention)

Inverse Kinematics for Trees

- Given a kinematic tree

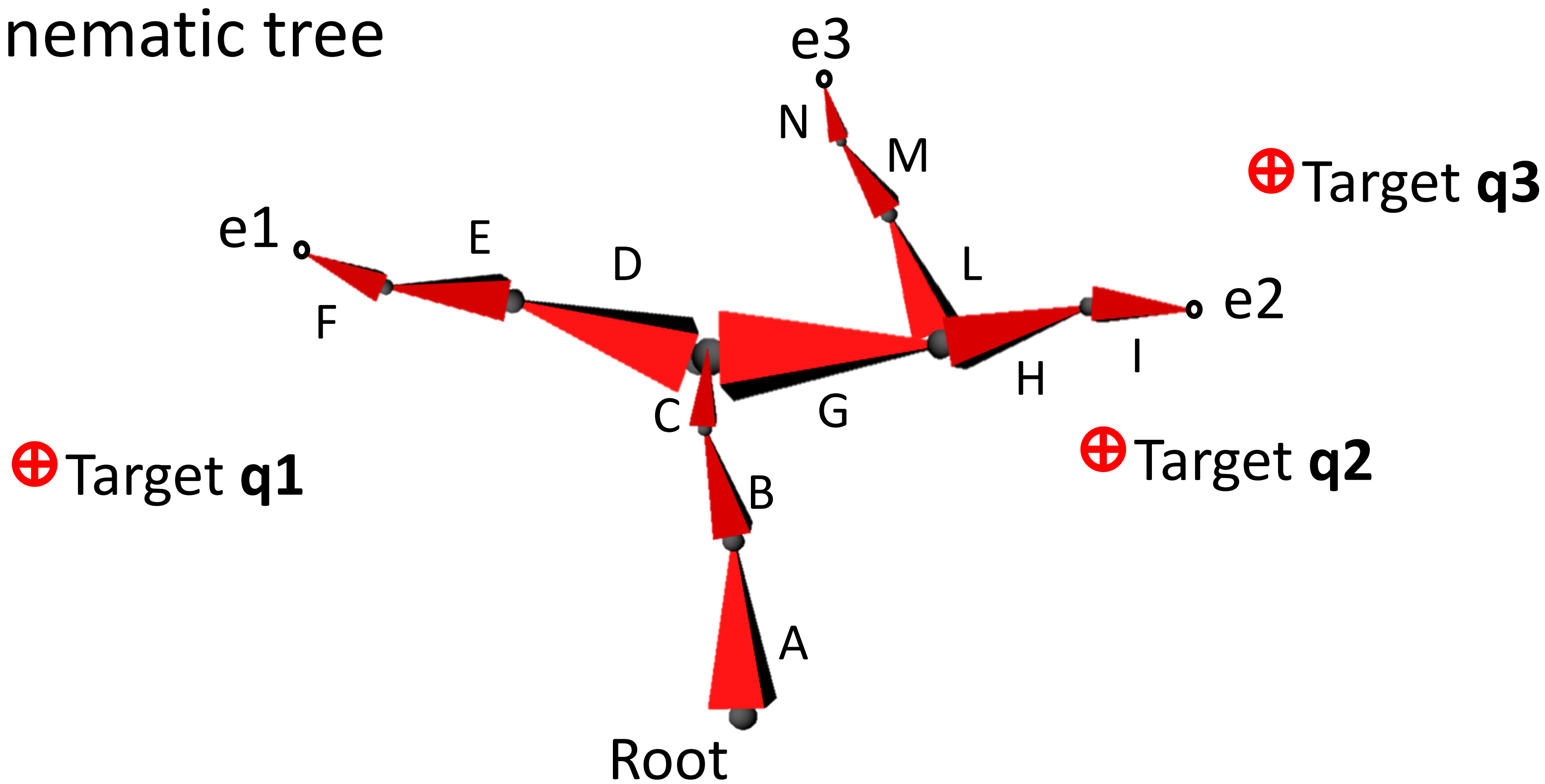


- An **Inverse Kinematics Problem** on a tree consists in finding the configuration of the kinematic chain for which the distance between the end effectors and some pre-defined targets points are minimized

- it is an **Inverse Kinematics Problem with Multiple Targets**

Inverse Kinematics for Trees

- Given a kinematic tree



$$\arg\min_x \begin{cases} \|p_1(x) - q_1\| \\ \|p_2(x) - q_2\| \\ \|p_3(x) - q_3\| \end{cases}$$

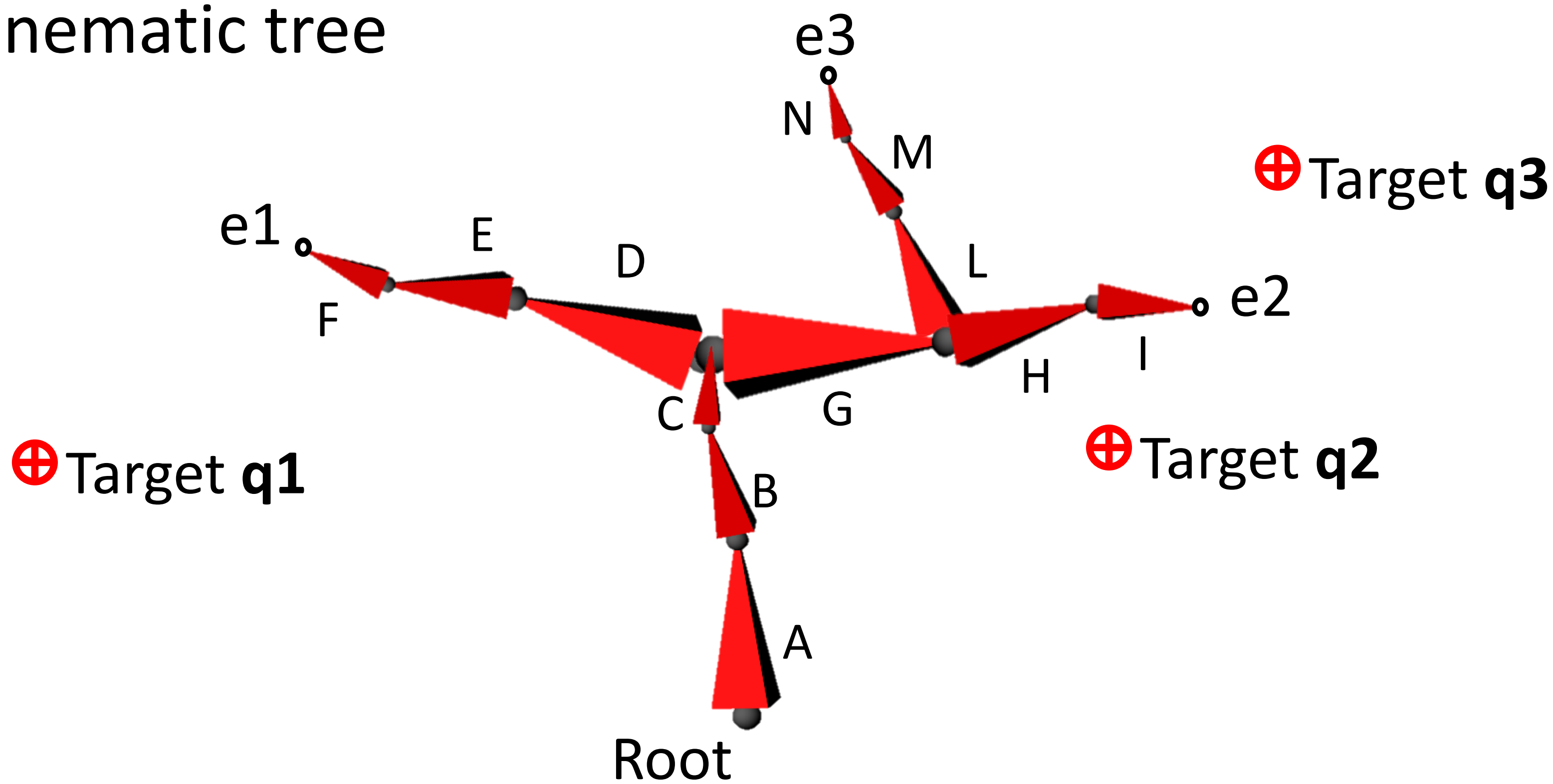
Find **x** such that it simultaneously minimizes all the distances with the targets

Multi-objective optimization

It is difficult to find x satisfying all the minima

Inverse Kinematics for Trees

- Given a kinematic tree



equivalent to

$$\|p(x) - q\|_2^2$$

ℓ_2 norm

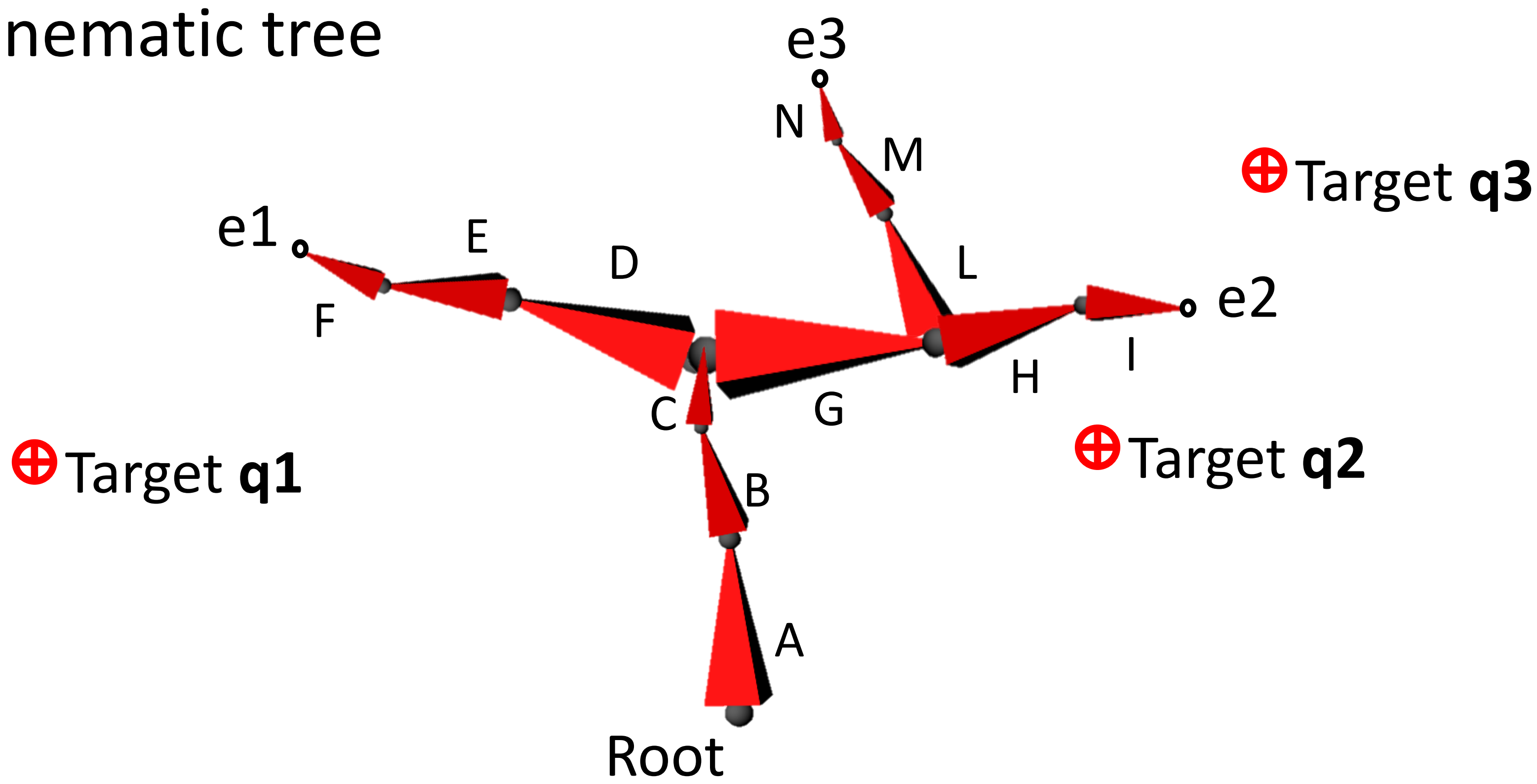
$$\arg\min_x \begin{cases} \|p_1(x) - q_1\| \\ \|p_2(x) - q_2\| \\ \|p_3(x) - q_3\| \end{cases} \Rightarrow \arg\min_x \sum_i \|p_i(x) - q_i\|^2$$

Find the x which minimizes the sum of the squared errors

i.e. the mean solution

Inverse Kinematics for Trees

- Given a kinematic tree



$$\arg\min_x \begin{cases} \|p_1(x) - q_1\| \\ \|p_2(x) - q_2\| \\ \|p_3(x) - q_3\| \end{cases} \Rightarrow \arg\min_x \sum_i \|p_i(x) - q_i\|$$

equivalent to

$$\|p(x) - q\|_1$$

ℓ_1 norm

Find the x which minimizes the sum of the errors

i.e. the median solution

Inverse Kinematics for Trees: Solution

- The ℓ_2 norm case

$$\arg\min_x \sum_i \|p_i(x) - q_i\|^2$$

- is still a **non-linear least square optimization problem**
  - Newton's method or Levenberg-Marquardt

$$p(x + \Delta x) = p(\overline{x}) + Jp(\overline{x})\Delta x + \ldots$$

$$\arg\min \|\overbrace{p(\overline{x}) + Jp(\overline{x})\Delta x} - q\|$$

$$\Updownarrow$$

$$p(\overline{x}) + Jp(\overline{x})\Delta x - q = 0$$

$$\Updownarrow$$

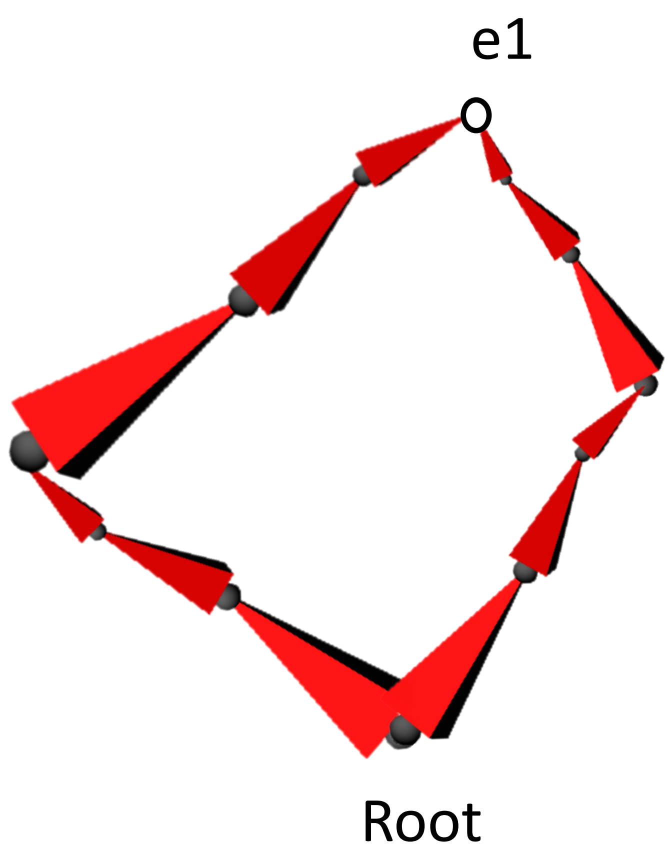
$$\Delta x = Jp(\overline{x})^{\dagger}(q - p(\overline{x}))$$

$$p(x) = \begin{bmatrix} p_1(x) \\ \vdots \\ p_n(x) \end{bmatrix}$$

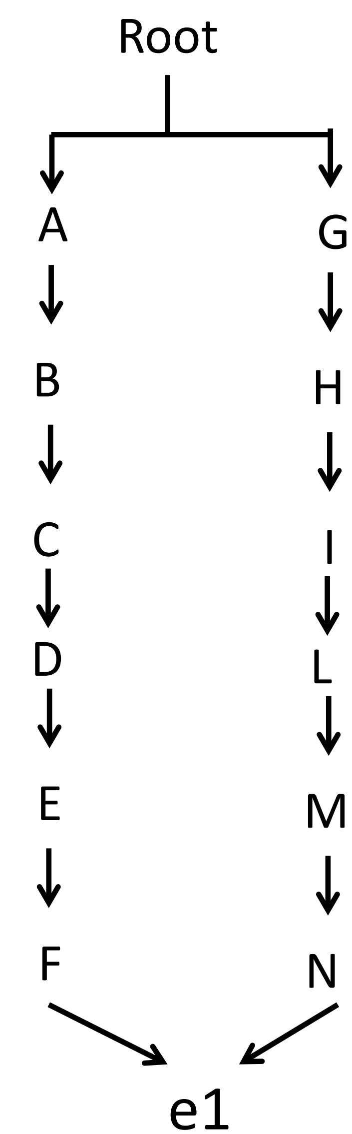
Kinematic Graph

- a **kinematic graph** is a graph of rigid transformations

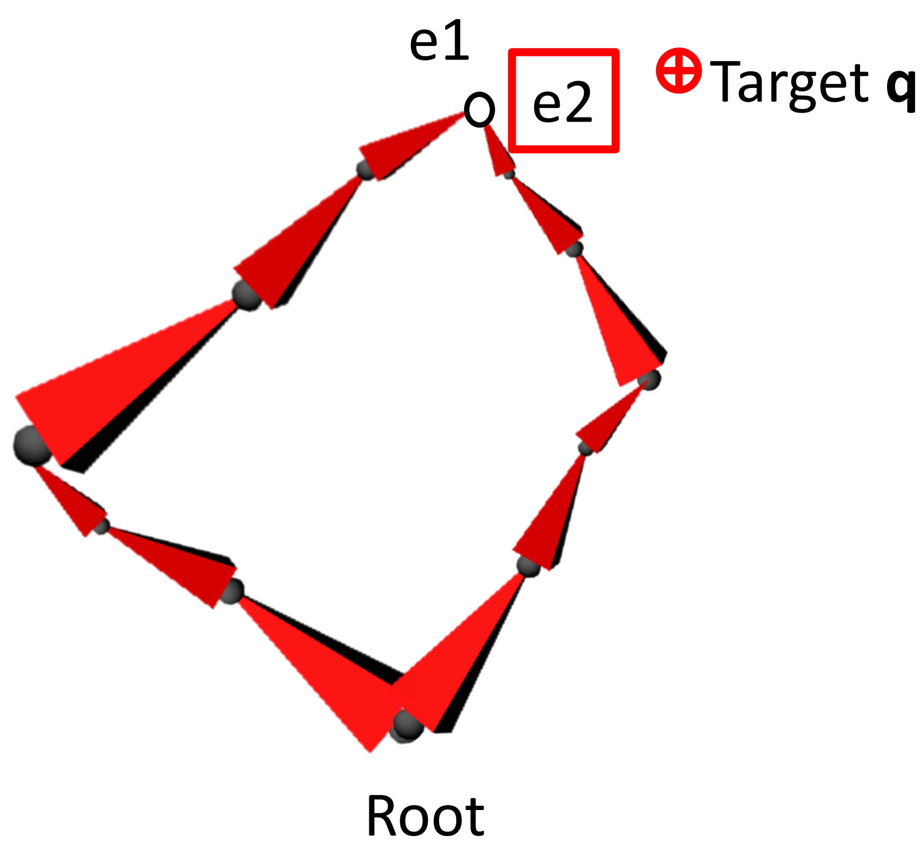
(it contains cycles)

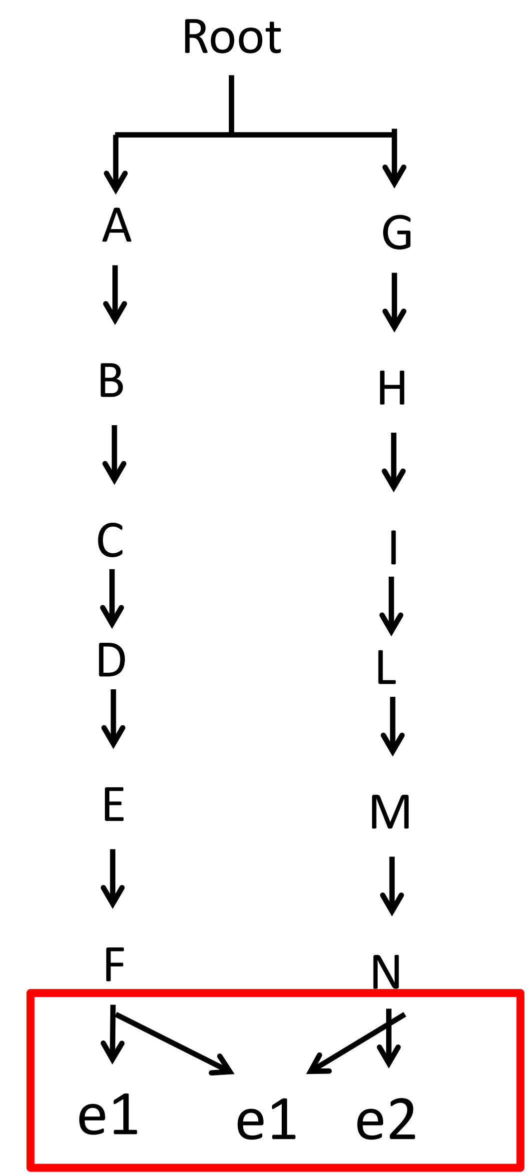


(parallel manipulators)



Inverse Kinematics for Graphs





$$\arg\min \|p_1(x) - q\|$$

$$\boxed{\text{subject} \quad p_1(x) = p_2(x)}$$

Constrained optimization

- Lagrange Multipliers
- or simply

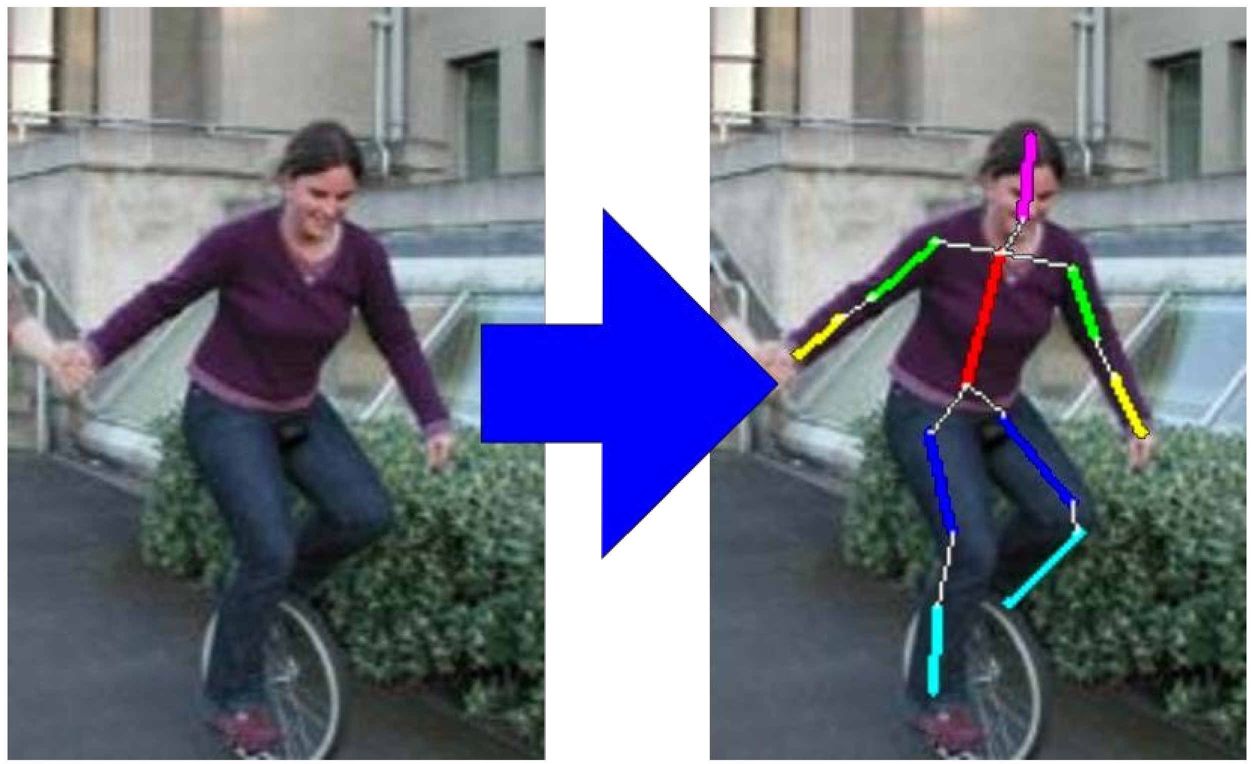
$$\arg\min \|p_1(x) - q\| + \|p_1(x) - p_2(x)\|$$

Content

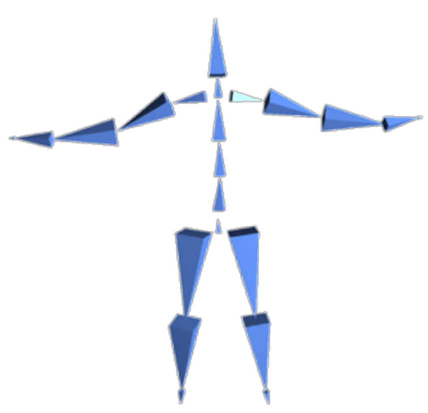
- Inverse Kinematics
- Kinematic Trees/Graphs
- **Pose Estimation/Motion Capture**

Pose Estimation

- Given an image depicting an articulated object, estimate the pose of that object



[Eichner 10]



determine all the DOF of that pose

Pose Estimation

- **Marker-based** pose estimation

- **Marker-less** pose estimation

[Faceshift]

- Pose estimation for multiple frames is called **motion capture**

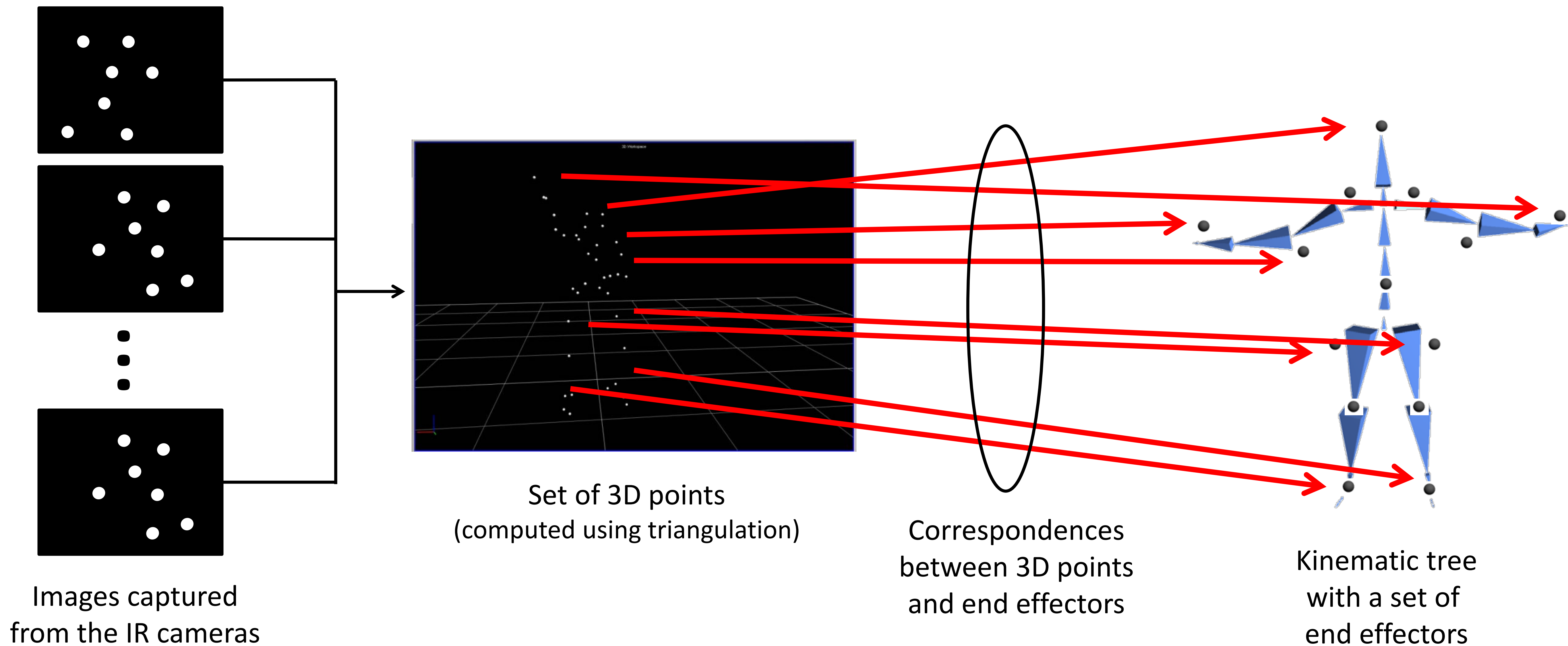
Marker-based Pose Estimation

Vicon system

Infrared cameras with IR illuminator

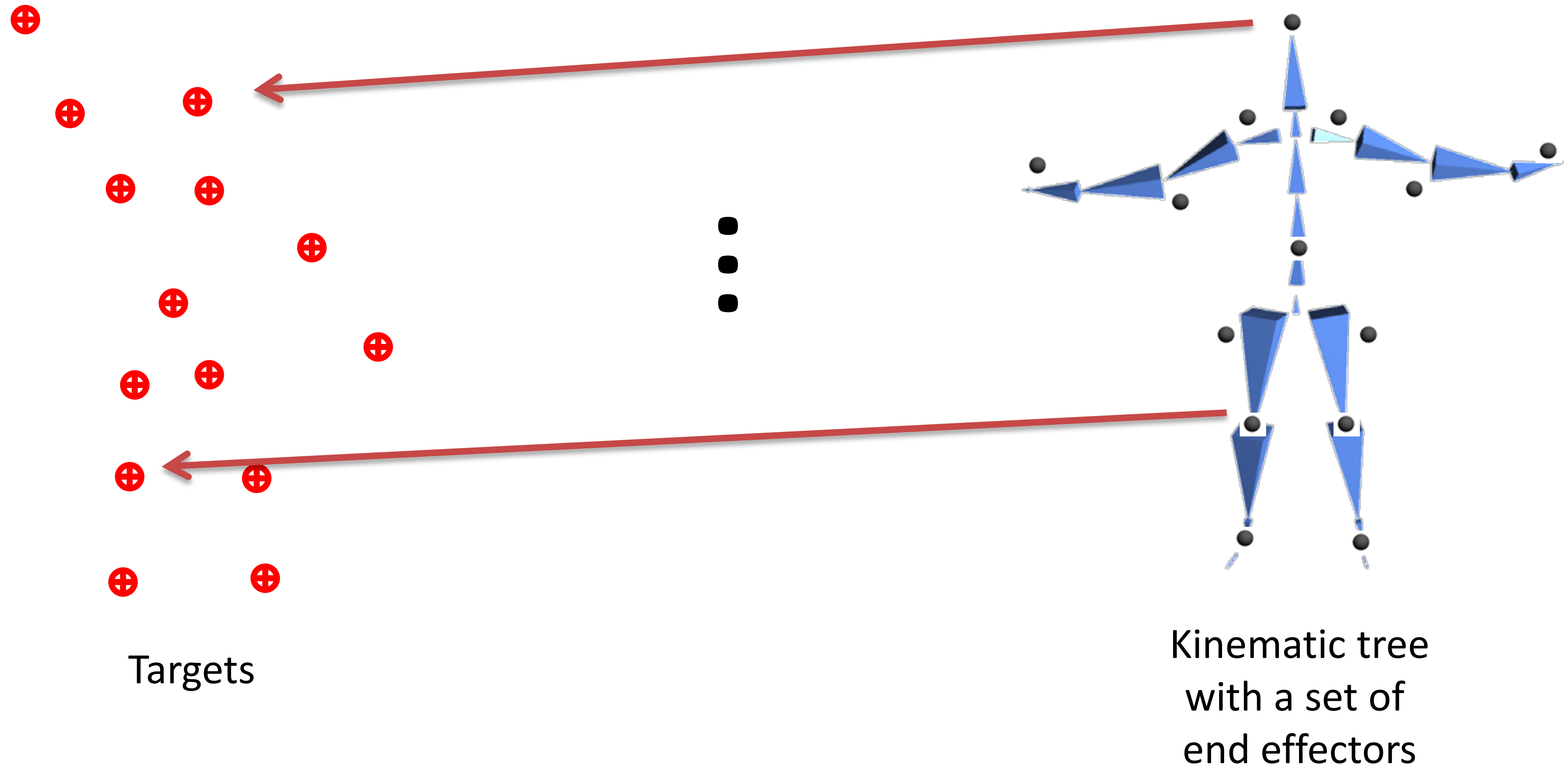
IR reflective markers

Marker-based Pose Estimation



- Triangulate 2D points -> 3D points
- Compute correspondences between 3D points and end effectors

Marker-based Pose Estimation



$$\arg\min_x \begin{cases} \|p_1(x) - q_1\| \\ \vdots \\ \|p_n(x) - q_n\| \end{cases}$$

Multi-objective optimization

$$\arg\min_x \sum_{i=1}^{n} \|p_i(x) - q_i\|$$

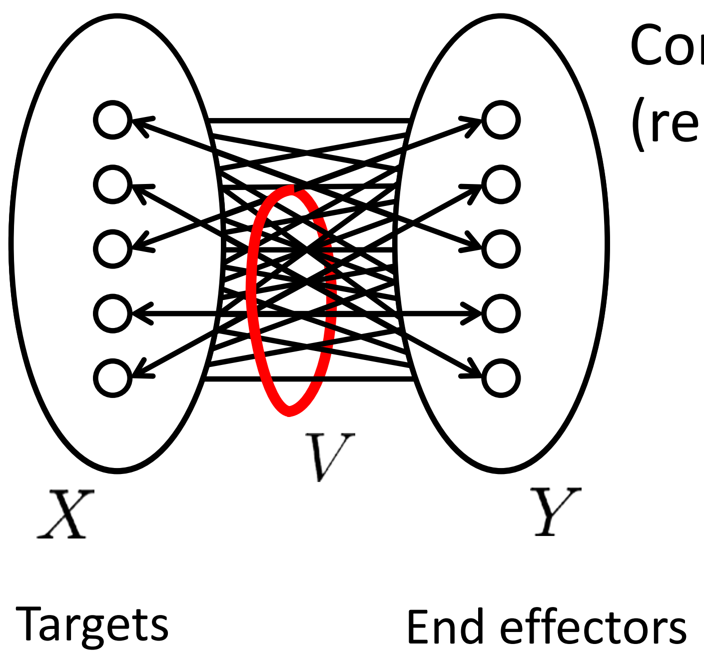
Correspondences

- If correspondences between 3D points and targets are not known (or the heuristic used to compute them is not accurate), these need to be estimated together with the pose

- One of the possible solution to this problem is the ICP registration (ICP=Iterative Closest Points) [Besl & McKay 92]

- But how can we formulate mathematically this problem?

Correspondences



Complete bi-partite graph (with weights w_{ij})
(representing all the possible matching)

$$G = (X \cup Y, E)$$

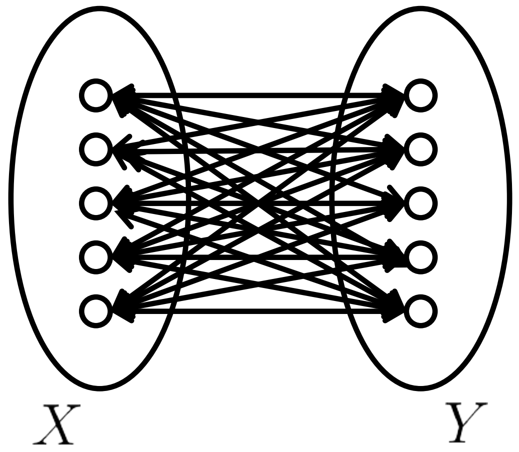
- All the edges on this graph are possible matching candidates
- We need to find the actual matches $V \subset E$ (unknown to the problem)

Correspondences

$e_{ij} = 1 \Leftrightarrow (i,j) \in V$ Target i in X is matched with end effector j in Y

$\boxed{w_{ij} \quad \text{cost of matching target i in X to end effector j in Y}}$

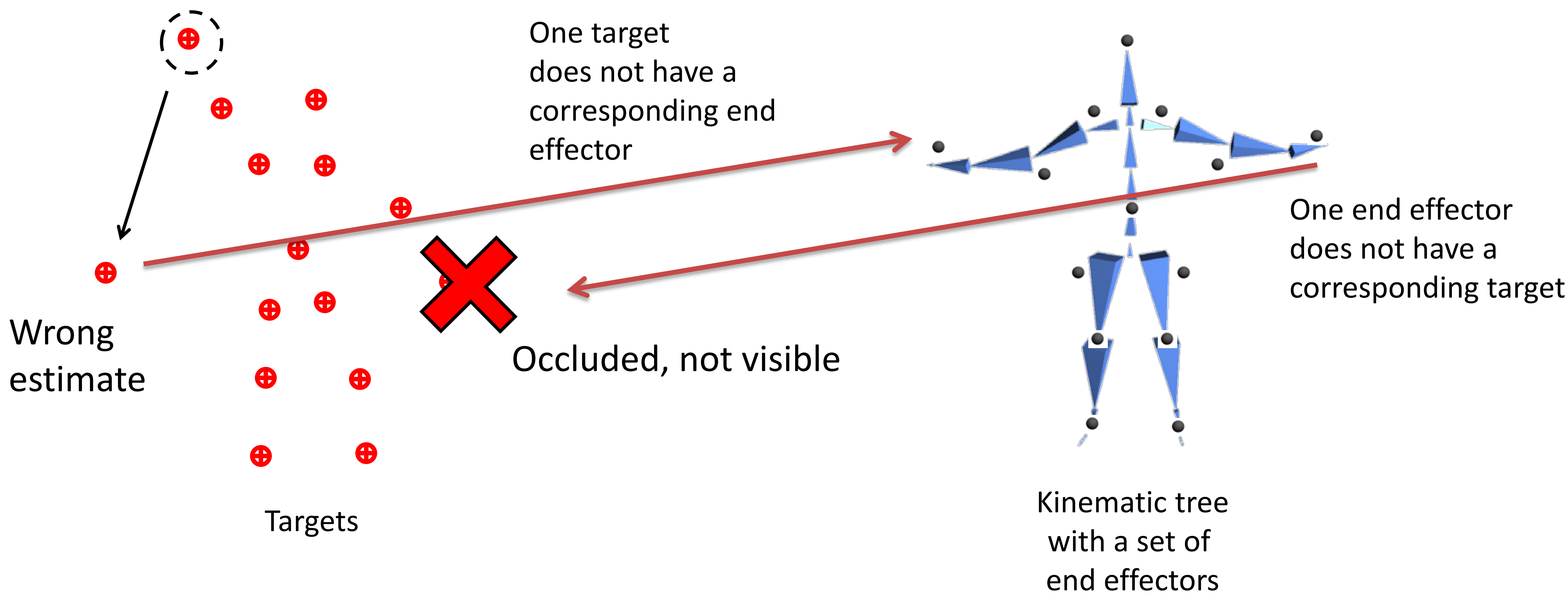
$$\begin{cases} \min\limits_{x, e_{ij}} \quad \sum_{ij} e_{ij} \|p_i(x) - q_j\| \quad \boxed{+\lambda \sum_{ij} w_{ij} e_{ij}} \\ \text{subject to} \quad \sum_i e_{ij} = 1 \qquad \forall i \\ \qquad\qquad\quad\; \sum_j e_{ij} = 1 \qquad \forall j \\ \qquad\qquad\quad\; e_{ij} \in \{0, 1\} \qquad \forall i, j \end{cases}$$



- Binary Integer problem + Continuous problem
- Alternating optimization approach
  - Integer Programming (Hungarian algorithm) + Gradient Descent

Correspondences

- It might happen that some targets are missing or that some targets are wrong

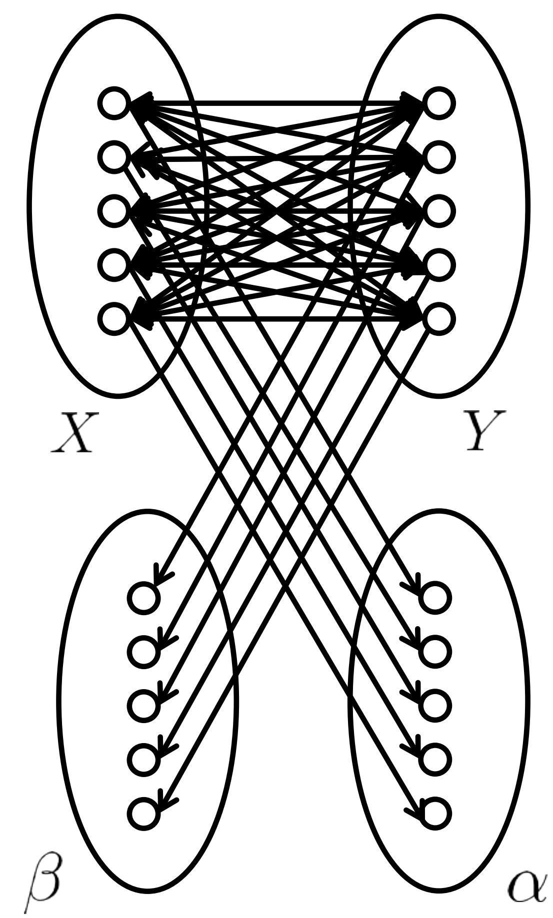


Correspondences

$$
\left\{
\begin{array}{lll}
\min\limits_{x, e_{ij}} & \sum_{ij} e_{ij} \|p_i(x) - q_j\| \quad + \lambda \sum_{ij} w_{ij} e_{ij} + \lambda_\alpha \sum_i \alpha_i + \lambda_\beta \sum_j \beta_j & \\
\text{subject to} & \sum_i e_{ij} + \alpha_i = 1 & \forall i \\
 & \sum_j e_{ij} + \beta_j = 1 & \forall j \\
 & e_{ij}, \alpha_i, \beta_j \in \{0, 1\} & \forall i, j
\end{array}
\right.
$$

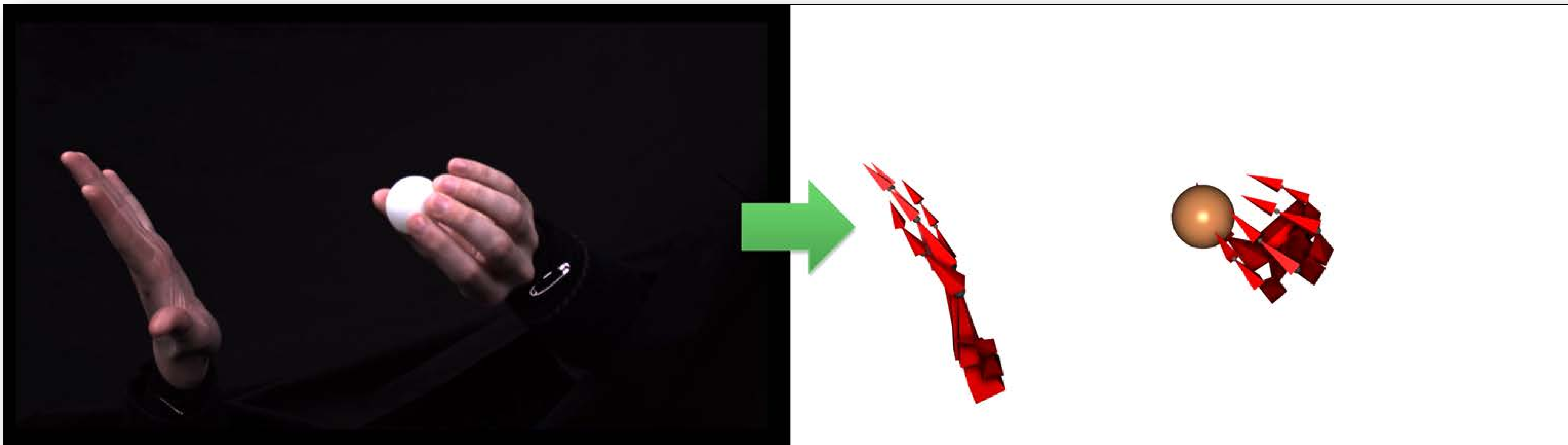
additional unknowns

virtual nodes



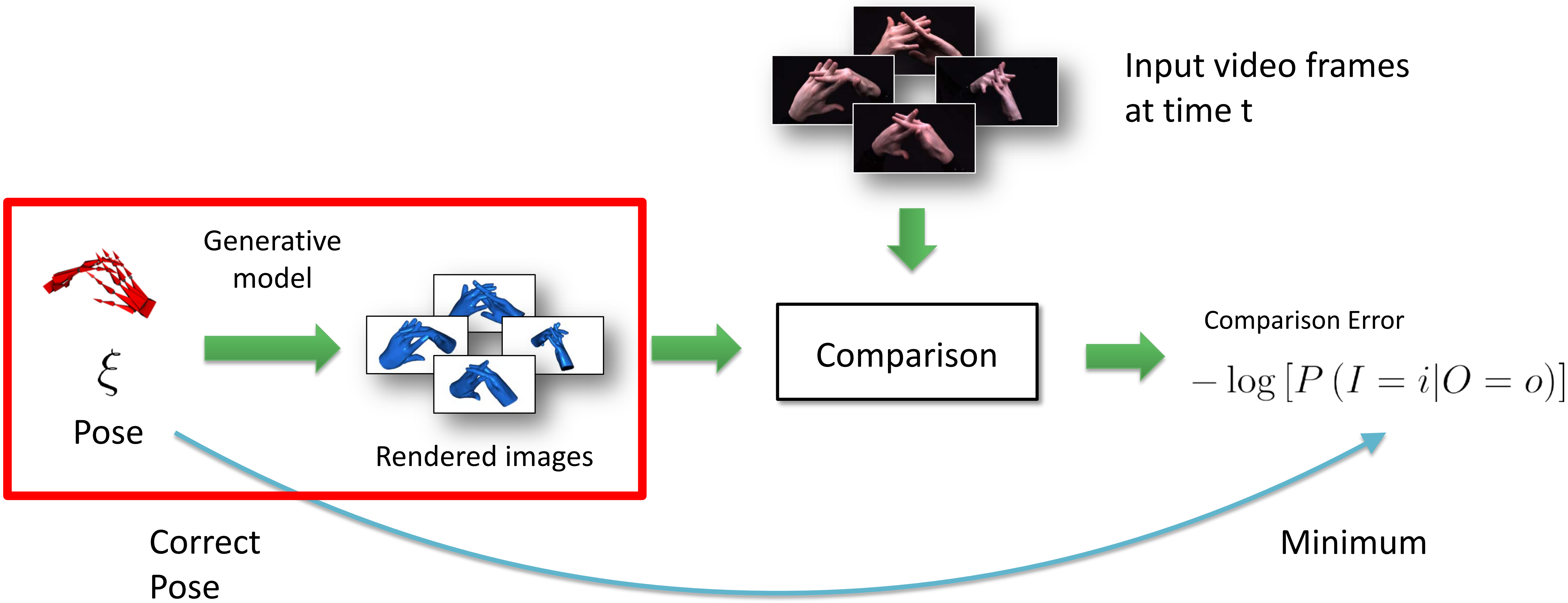
Marker-Less Pose Estimation

Input: Output:



Scene Motion
(angles and positions)

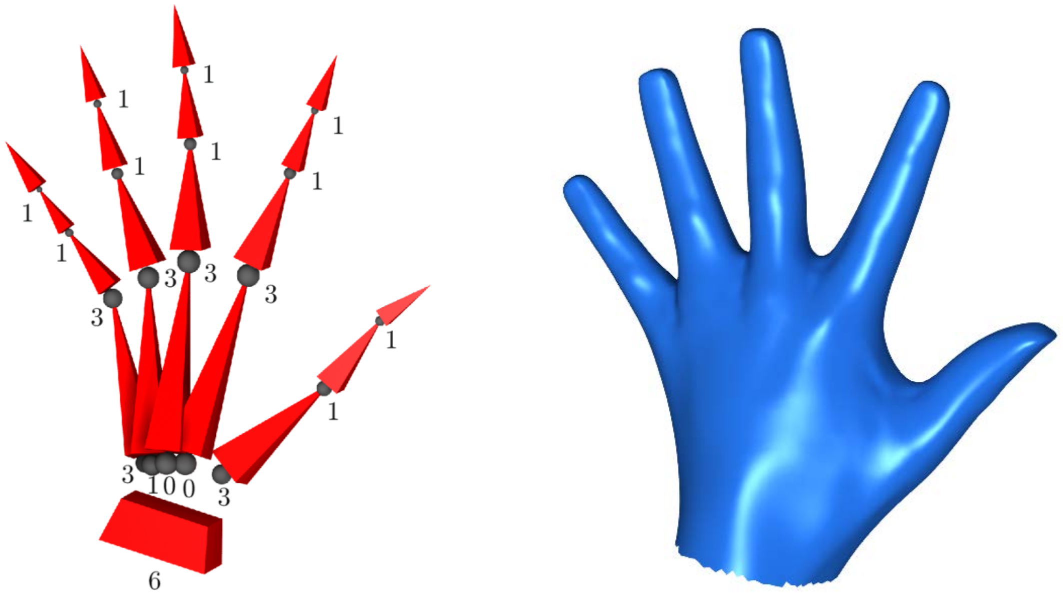
Marker-Less Pose Estimation



$$\pi(i) = \arg\max_{o} P\left(I=i|O=o\right)$$

Maximum likelihood

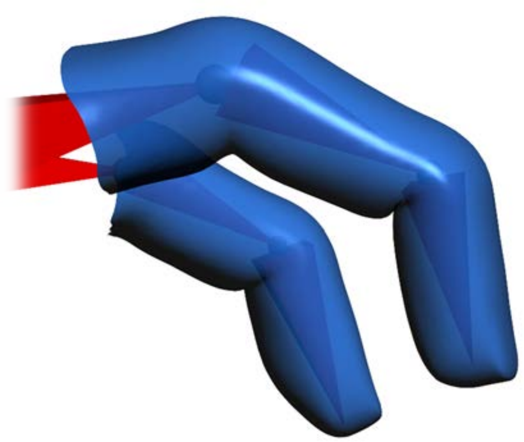
Generative Model



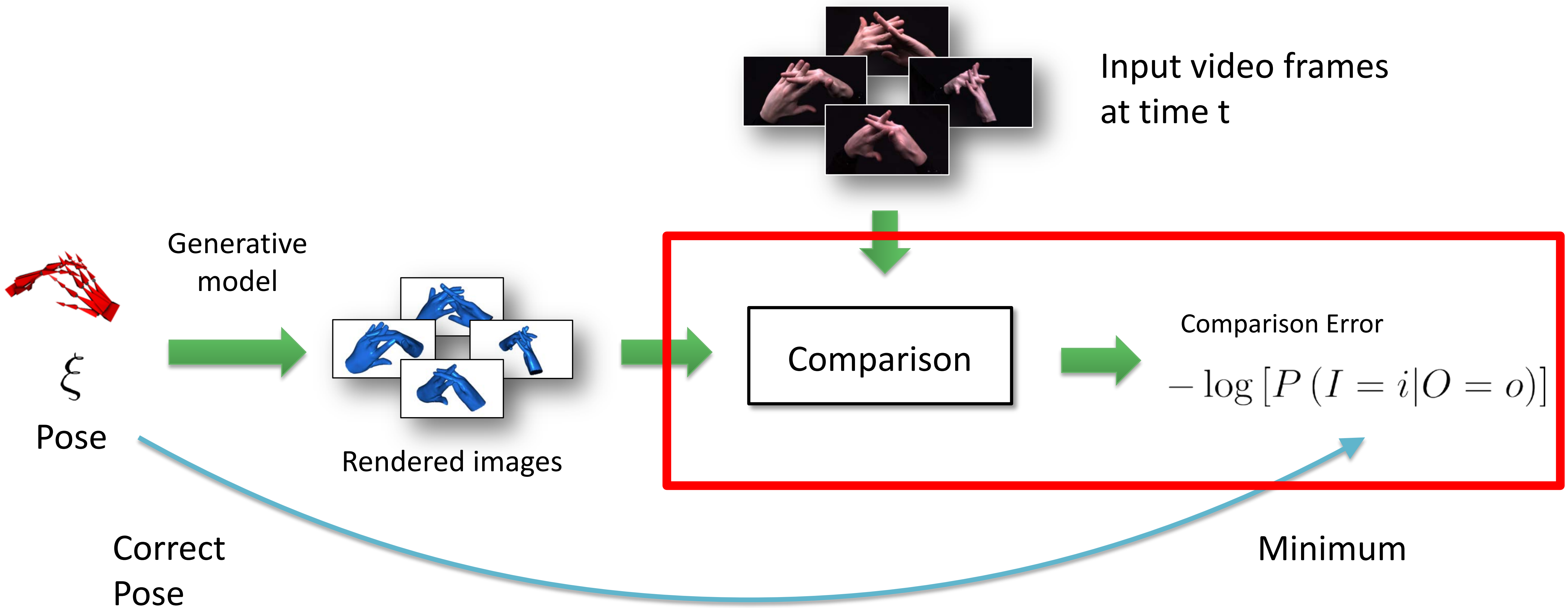
Kinematic tree + 3D model
At a rigging pose

$$v_k(\boldsymbol{\xi}) = \sum_{j=1}^{m} \alpha_{k,j} T_j(\boldsymbol{\xi}) T_j(0)^{-1} v_k(0)$$

the motion of a vertex → the linear combination of all the motions that the vertex would undergo if rigidly attached to every bone, one at a time



Marker-Less Pose Estimation

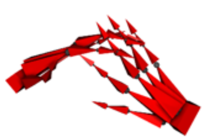


$$\pi(i) = \arg\max_{o} P\left(I=i|O=o\right)$$

Maximum likelihood

Comparison

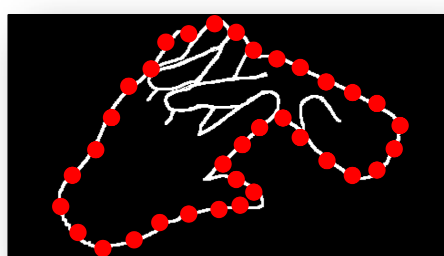
Input video frame
at time t



ξ

Pose

Generative
model



$$\sum_k d(v_k(\boldsymbol{\xi}), S)$$

$d(\cdot, S)$ = is a distance map in 3D between a point and S

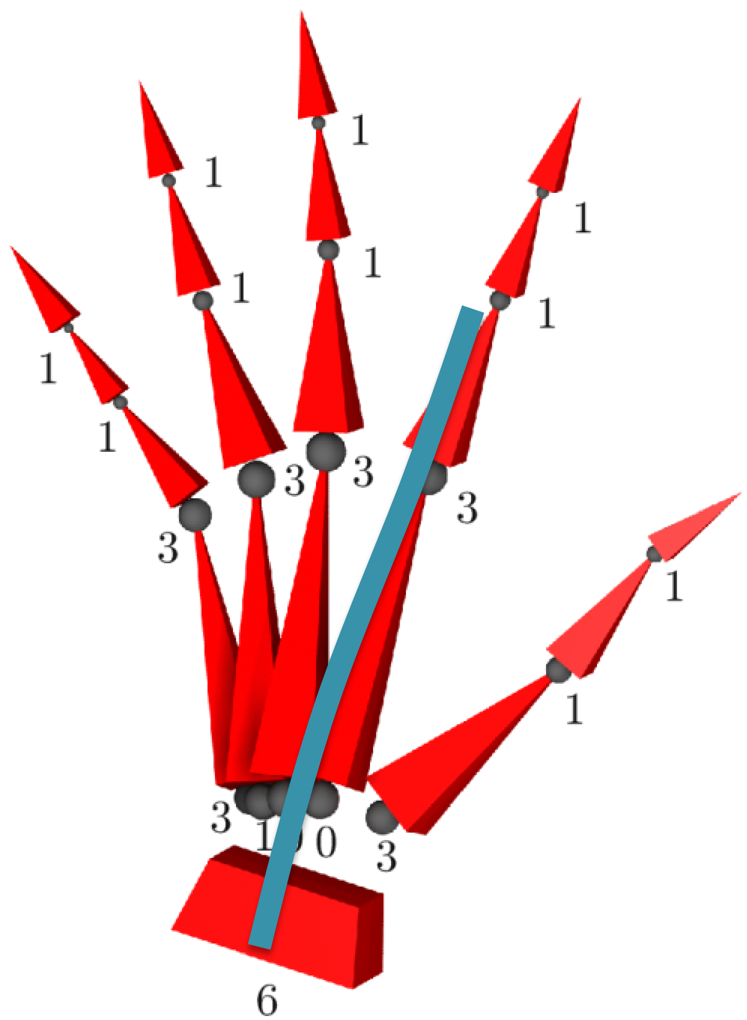
$$v_k(\boldsymbol{\xi}) = \sum_{j=1}^{m} \alpha_{k,j} T_j(\boldsymbol{\xi}) T_j(0)^{-1} v_k(0)$$

Solution

$$L(\boldsymbol{\xi}) = \sum_k d(v_k(\boldsymbol{\xi}), S)$$

$d(\cdot, S)$ = is a distance map in 3D

$$\frac{\partial L}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}) = \sum_k \nabla d(v_k(\boldsymbol{\xi}), S) \cdot \frac{\partial v_k}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi})$$



$$v_k(\boldsymbol{\xi}) = \sum_{j=1}^{m} \alpha_{k,j} T_j(\boldsymbol{\xi}) T_j(0)^{-1} v_k(0)$$

$$\frac{\partial v_k}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}) = \sum_{j=1}^{m} \alpha_{k,j} \frac{\partial T_j}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}) T_j(0)^{-1} v_k(0)$$

$$T_j(\boldsymbol{\xi}) = e^{\xi_{c_1}} \ldots e^{\xi_{c_q}}$$

$$\frac{\partial T_j}{\partial \boldsymbol{\xi}}(\boldsymbol{\xi}) = \cdots$$

Final Examination

- Homework: 10% each
- Oral Exam: 60%
    - (60%-40%) about **40min** per person (20 Luca, 20 J.C.)
    - (100%) about **1h** per person (30 Luca, 30 J.C.)

- each person has to answer different questions/exercises spanning the **entire program**

- ETH will schedule the exam sometime in August

- For the ones
    - who didn't do the homework
    - who did not do **all** the homework
    - who is not satisfied with the homework grades and want to improve it

he/she can do the 100% exam to recover the missing points

(inform in advance)