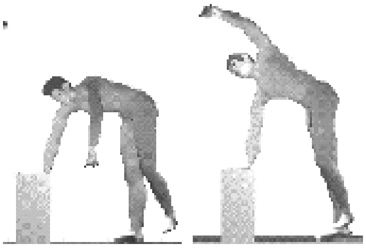
Inverse Kinematics

What is IK?

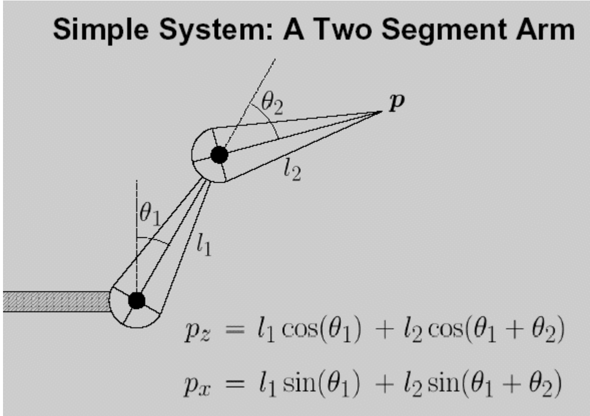
[Fanuc]



[Ronan Boulic]

A Simple Example

Two link robot



Simple System: A Two Segment Arm

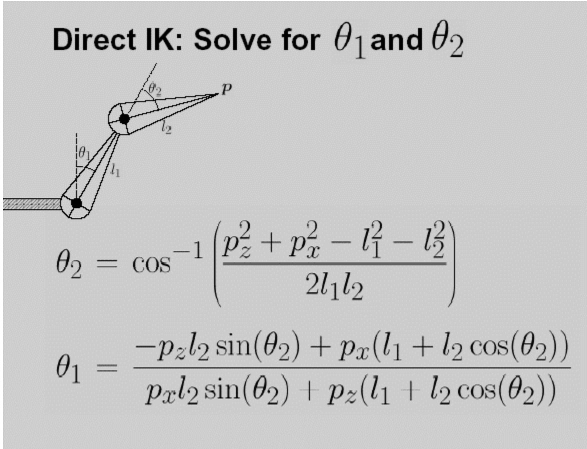
$$p_z = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2)$$

$$p_x = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2)$$

[James O'Brien]

A Simple Example

Direct IK Solution



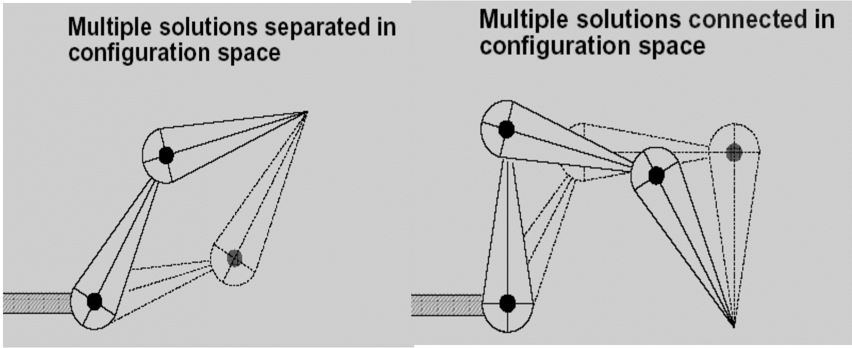
Direct IK: Solve for θ_1 and θ_2

$$\theta_2 = \cos^{-1}\left(\frac{p_z^2 + p_x^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right)$$

$$\theta_1 = \frac{-p_z l_2 \sin(\theta_2) + p_x(l_1 + l_2 \cos(\theta_2))}{p_x l_2 \sin(\theta_2) + p_z(l_1 + l_2 \cos(\theta_2))}$$

[James O'Brien]

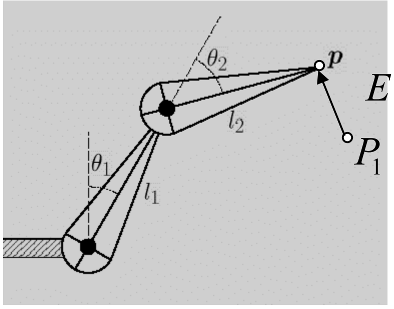
Problems



[James O'Brien]

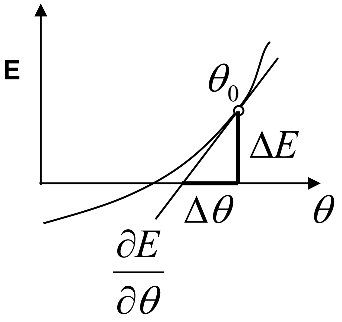
- + unreachable goals

Solving for Constraints



$$E = P - P_1$$

Newton's Method



$$\frac{\partial E}{\partial \theta} = \frac{\Delta E}{\Delta \theta}$$

$$\Delta\theta = \left(\frac{\partial E}{\partial \theta}\right)^{-1} \Delta E$$

$$\theta' = \theta - \Delta\theta$$

$$\frac{\partial E}{\partial \theta} = \frac{\partial P}{\partial \theta}$$

Jacobian

$$\Delta\theta = \left(\frac{\partial P}{\partial \theta}\right)^{-1} \Delta P \qquad \Delta\theta = J^{-1}\Delta P$$

Jacobian is given by $$J = \frac{\partial P}{\partial \theta} \approx \frac{\Delta P}{\Delta \theta}$$

$$J \cdot \Delta\theta = \Delta P$$

velocities: $$J \cdot \frac{\Delta\theta}{\Delta t} = \frac{\Delta P}{\Delta t}$$

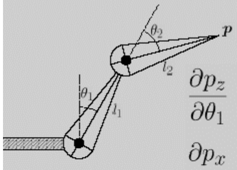
$$J \cdot \dot{q} = \dot{x}$$

Jacobian Example

$$p_z = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2)$$

$$p_x = l_1 \sin(\theta_1) + l_2 \sin(\theta_1 + \theta_2)$$

Simple System: A Two Segment Arm



$$\frac{\partial p_z}{\partial \theta_1} = -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2)$$

$$\frac{\partial p_x}{\partial \theta_1} = l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2)$$

$$\frac{\partial p_z}{\partial \theta_2} = - l_2 \sin(\theta_1 + \theta_2)$$

$$\frac{\partial p_x}{\partial \theta_2} = + l_2 \cos(\theta_1 + \theta_2)$$

[James O'Brien]

Jacobian Example

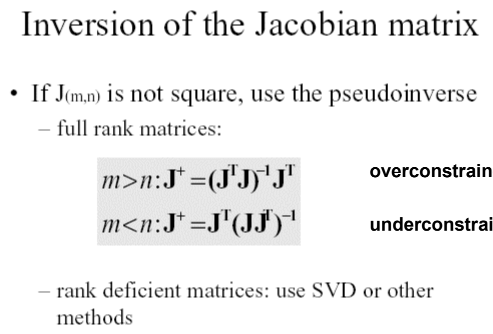
Example for two segment arm

$$J = \begin{bmatrix} \frac{\partial p_z}{\partial \theta_1} & \frac{\partial p_z}{\partial \theta_2} \\ \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} \end{bmatrix}$$

[James O'Brien]

Inverting the Jacobian

- m constraints, n DOF



Inversion of the Jacobian matrix

- If $J_{(m,n)}$ is not square, use the pseudoinverse
  - full rank matrices:

$$m>n: \mathbf{J}^+ = (\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T$$

$$m<n: \mathbf{J}^+ = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T)^{-1}$$

  - rank deficient matrices: use SVD or other methods

overconstrained, minimizes $\left\| J \cdot \dot{q} - \dot{x} \right\|$

underconstrained, minimizes $\left\| \dot{q} \right\|$

Pseudoinverse

Least-Squares Inverse

$$AA^+A = A$$

$$(AA^+)^T = AA^+$$

$$A^+AA^+ = A^+$$

$$(A^+A)^T = A^+A$$

$$\left(A^+\right)^+ = A$$

$$(A^+)^T = (A^T)^+$$

derived

but...

$$(AB)^+ \neq B^+A^+$$

Iterative IK solution

University of
British Columbia

```
repeat
  E = P - Ptarget
  dX = k*error;  // k<1
  compute J
  compute J*(J)
  compute dQ = J* dX
  Q = Q + dQ     // update joint angles
until |error| < epsilon
```

Pseudoinverse Discussion

University of
British Columbia

- instable around singularities
- can weight the joint movement

Cost function with a weighting matrix: $C(dQ) = dQ^T W dQ$

↑ nxn weighting matrix
positive definite

$$dQ = W^{-1} J^T (J W^{-1} J^T)^{-1} dX$$

- damped least squares solution helps avoid singularities

$$(A^+)^\lambda = A^T (AA^T + \lambda^2 I)^{-1}$$

minimizes $\left\|A\dot{q} - \dot{x}\right\|^2 + \lambda^2 \left\|\dot{q}\right\|^2$

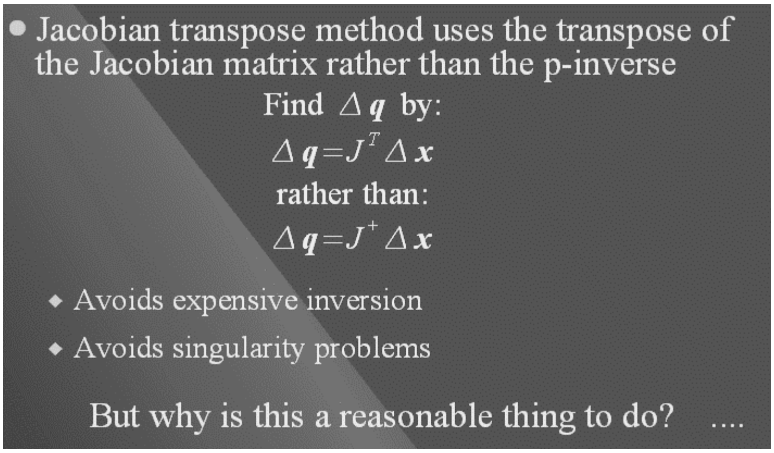
Pseudoinverse Discussion

secondary task

- obstacle avoidance
- joint limit avoidance
- singularity avoidance

$$\dot{q} = J^{+}\dot{x} + (I - J^{+}J)z$$

Jacobian Transpose Method



- Jacobian transpose method uses the transpose of the Jacobian matrix rather than the p-inverse

Find $\Delta \boldsymbol{q}$ by:

$$\Delta \boldsymbol{q} = J^{T} \Delta \boldsymbol{x}$$

rather than:

$$\Delta \boldsymbol{q} = J^{+} \Delta \boldsymbol{x}$$

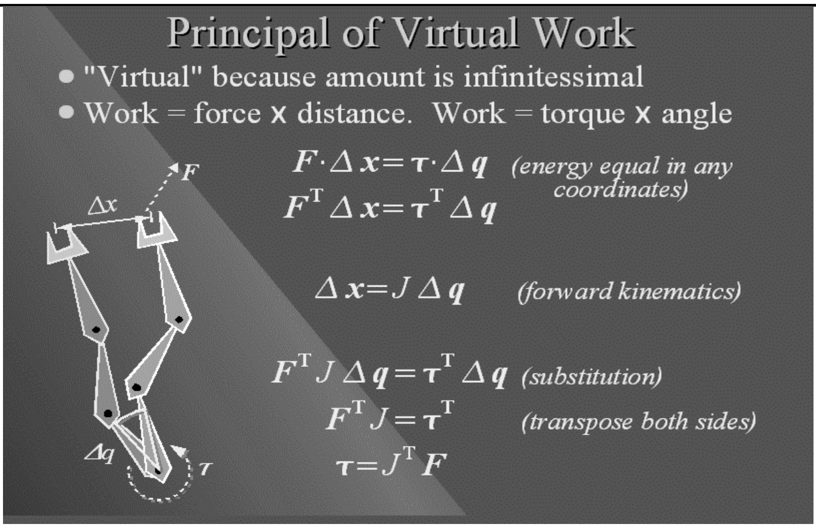
  - Avoids expensive inversion
  - Avoids singularity problems

But why is this a reasonable thing to do? ....

[Baxter, UNC]

Jacobian Transpose Method

University of
British Columbia



Principal of Virtual Work

- "Virtual" because amount is infinitessimal
- Work = force x distance. Work = torque x angle

$$\boldsymbol{F}\cdot\Delta\,\boldsymbol{x}=\boldsymbol{\tau}\cdot\Delta\,\boldsymbol{q}$$ (energy equal in any coordinates)

$$\boldsymbol{F}^{\mathrm{T}}\,\Delta\,\boldsymbol{x}=\boldsymbol{\tau}^{\mathrm{T}}\,\Delta\,\boldsymbol{q}$$

$$\Delta\,\boldsymbol{x}=J\,\Delta\,\boldsymbol{q}$$ (forward kinematics)

$$\boldsymbol{F}^{\mathrm{T}}\,J\,\Delta\,\boldsymbol{q}=\boldsymbol{\tau}^{\mathrm{T}}\,\Delta\,\boldsymbol{q}$$ (substitution)

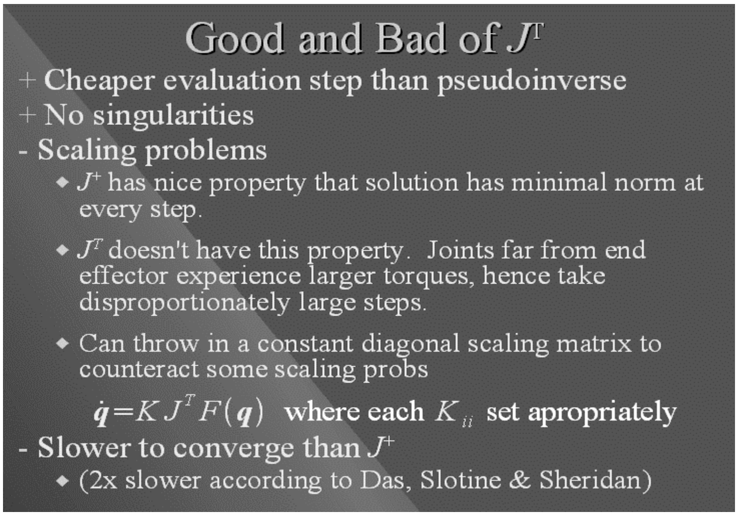
$$\boldsymbol{F}^{\mathrm{T}}\,J=\boldsymbol{\tau}^{\mathrm{T}}$$ (transpose both sides)

$$\boldsymbol{\tau}=J^{\mathrm{T}}\,\boldsymbol{F}$$

[Baxter, UNC]

Jacobian Transpose Method

University of
British Columbia



Good and Bad of J^T

+ Cheaper evaluation step than pseudoinverse

+ No singularities

- Scaling problems
  - J^+ has nice property that solution has minimal norm at every step.
  - J^T doesn't have this property. Joints far from end effector experience larger torques, hence take disproportionately large steps.
  - Can throw in a constant diagonal scaling matrix to counteract some scaling probs

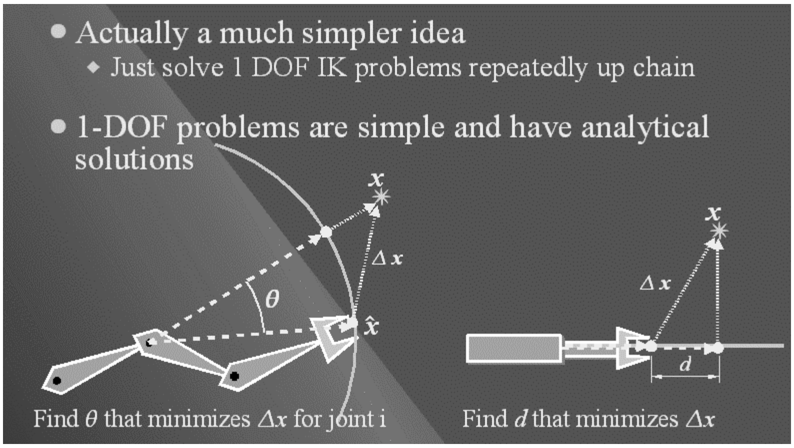
$$\dot{\boldsymbol{q}}=K\,J^{T}\,F(\boldsymbol{q})$$ where each K_{ii} set apropriately

- Slower to converge than J^+
  - (2x slower according to Das, Slotine & Sheridan)

[Baxter, UNC]

Cyclic Coordinate Descent

University of
British Columbia



- Actually a much simpler idea
  - Just solve 1 DOF IK problems repeatedly up chain
- 1-DOF problems are simple and have analytical solutions

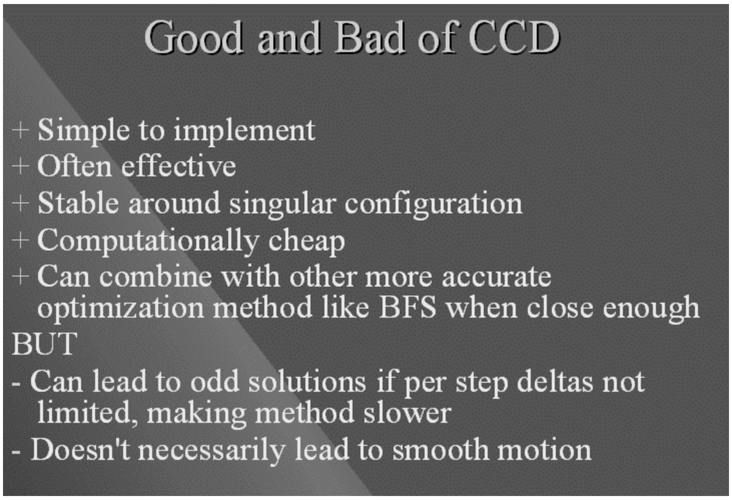
Find θ that minimizes Δx for joint i

Find d that minimizes Δx

[Baxter, UNC]

Cyclic Coordinate Descent

University of
British Columbia



Good and Bad of CCD

+ Simple to implement
+ Often effective
+ Stable around singular configuration
+ Computationally cheap
+ Can combine with other more accurate optimization method like BFS when close enough

BUT

- Can lead to odd solutions if per step deltas not limited, making method slower
- Doesn't necessarily lead to smooth motion

[Baxter, UNC]

Machine Learning

University of
British Columbia

$$f : \vec{x} \rightarrow \vec{q}$$

problem: one-to-many mapping

$$f : \vec{x}, \vec{q}_0 \rightarrow \vec{q}$$

Human Motion

University of
British Columbia

- hands typically travel in straight-line paths
- strength influences trajectory of some motions
- course project ?