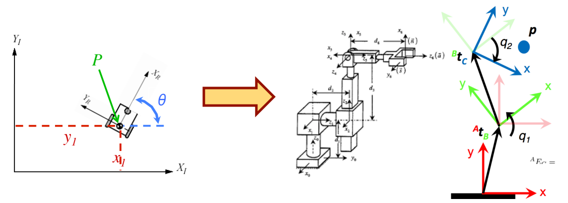
Inverse Kinematics



Many slides adapted (with thanks!) from:
Siegwart, Nourbakhsh and Scaramuzza, Autonomous Mobile Robots
Renata Melamud, An Introduction to Robot Kinematics, CMU
Rick Parent, Computer Animation, Ohio State
Steve Rotenberg, Computer Animation, UCSD
Angela Sodemann, www.youtube.com/watch?v=IVjFhNv2N8o, ASU

Bookkeeping

2

- Project status
  - Have you looked carefully at the turnins?
- Milestone 4
- Homework 3
- Scheduling
- A note about DH kinematics:
  - The last few slides from last time covered an example in some detail; please do make sure it makes sense.

Inverse Kinematics

3

- So far we've mostly been working forward.
- **Goal:** Compute vector of joint DoFs that puts end effector in some desired goal state
  - Inverse of previous problem
- Instead of function from world space to robot space.

$$\mathbf{e} = f(\boldsymbol{\Phi}) \leftrightarrow \boldsymbol{\Phi} = f^{-1}(\mathbf{e})$$

(Reminder: Φ = parameters, e = end effector configuration)

Inverse Kinematics

4

- IK is challenging!
  - $f()$ is (usually) relatively easy to evaluate
  - $f^{-1}()$ usually isn't
  - Perfect solution means solving a *nonlinear system of equations.*
- Sometimes not actually possible!
- Many different approaches to solving IK problems
  - Which is usually a sign there's no great solution…

Inverse Kinematics

5

- Underconstrained
  - Fewer constraints than DoFs
  - Many solutions
- Overconstrained
  - Too many constraints
  - No solution
- Unreachable workspace
  - Volume the end effector can reach ≠ goal
- Dextrous workspace
  - Volume end effector can reach in any orientation ≠ goal

Analytical vs. Numerical

6

- One major way to classify IK-solving approaches: **analytical** vs **numerical** methods
- Analytical
  - Find an exact solution by directly inverting the forward kinematics equations.
  - Works only for relatively simple chains.
- Numerical
  - Use approximation and iteration to converge on a solution.
  - More expensive, more general purpose.
- We will look at one technique: Jacobians

Analytical vs. Numerical 2

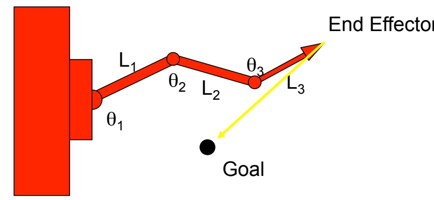
7

1. Set **goal configuration** of **end effector**
2. Calculate interior **joint angles**

- Compute the vector of joint DOFs that will cause the end effector to reach some desired goal state
- Analytic approach
  - Directly calculate joint angles in configuration that satisfies goal
- Numeric approach
  - At each time slice, determine joint movements that take you in direction of goal position (and orientation)

Inverse Kinematics

8



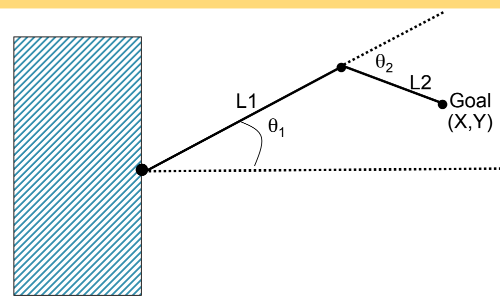
Analytic IK Solving

9

- Given arm configuration (L1, L2, …)
- Given desired goal position (and orientation) of end effector: [x,y,z, ψ1,ψ2, ψ3]
- Analytically compute goal configuration (θ1,θ2, …)
- Interpolate pose vector from initial to goal

Analytic Inverse Kinematics

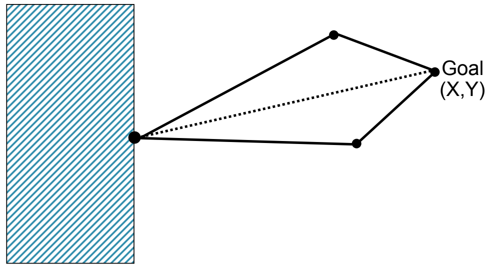
10



Analytic Inverse Kinematics

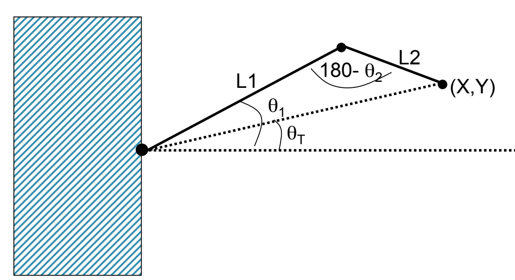
11

Multiple solutions



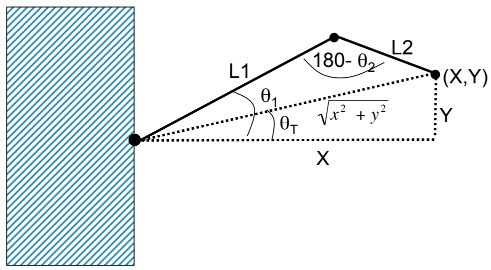
Analytic Inverse Kinematics

12



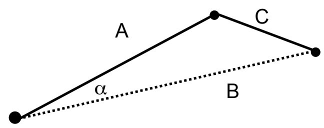
Analytic Inverse Kinematics

13



Law of Cosines

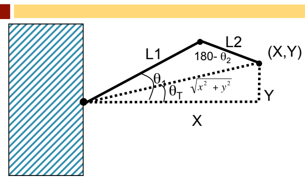
14



$$\cos(\alpha) = \frac{A^2 + B^2 - C^2}{2AB}$$

Analytic Inverse Kinematics

15



$$\cos(\theta_T) = \frac{X}{\sqrt{X^2 + Y^2}}$$

$$\theta_T = \cos^{-1}\left(\frac{X}{\sqrt{X^2 + Y^2}}\right)$$

$$\cos(\theta_1 - \theta_T) = \frac{L_1^2 + X^2 + Y^2 - L_2^2}{2L_1\sqrt{X^2 + Y^2}}$$

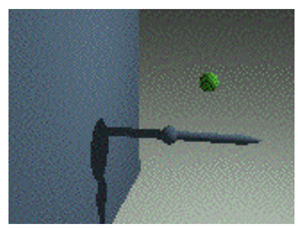
$$\theta_1 = \cos^{-1}\left(\frac{L_1^2 + X^2 + Y^2 - L_2^2}{2L_1\sqrt{X^2 + Y^2}}\right) + \theta_T$$

$$\cos(180 - \theta_2) = \frac{L_1^2 + L_2^2 - (X^2 + Y^2)}{2L_1L_2}$$

$$\theta_2 = 180 - \cos^{-1}\left(\frac{L_1^2 + L_2^2 - (X^2 + Y^2)}{2L_1L_2}\right)$$

Analytic Inverse Kinematics

16



Iterative Inverse Kinematics

17

- When linkage is too complex for analytic methods:
- At each time step, take end effector towards goal position and orientation (somehow)
  - A form of greedy search
  - Does greedy search always find a solution?
- Need to recompute at each time step
- Big question: what is "somehow"?
- These are *approximate methods.*

More Specifically…

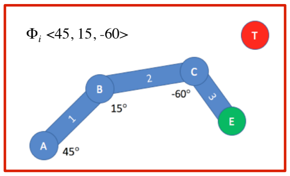
18

- For each timestep:
  1. Choose some movement in some DOF
     - By heuristic
     - By immediate velocity
     - Other…
  2. Perform *forward* kinematics – where's robot?
     - Don't have to move to do this!
  3. Consider undoing that move
  4. Loop

Jacobian IK

- Φ_i is the initial pose vector for all joints
  - This is all we start off knowing!

1. Find the final joint configurations: Φ_f
2. Compute the change in rotations: $\Delta\Phi$
3. Compute the Jacobian: J



https://medium.com/unity3danimation/overview-of-jacobian-ik-a33939639ab2

Finding Joint Moves from J

- We iteratively change $\Delta\Phi$ by small amounts
  - Size of Δ is controlled by a parameter
- $V = J * \Delta\Phi$
  - V = difference between end effector and target
  - (For now, J is magic)
- $\Delta\Phi = J^T * V$
- But now need to computer to Jacobian matrix, J

https://medium.com/unity3danimation/overview-of-jacobian-ik-a33939639ab2

The Jacobian Matrix

- Each term in the Jacobian shows how a change in one joint angle changes the end effector.
  - Example: the first term gives end effector change along the X-axis, if Joint A's angle is changed by Δ.

$$J = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_A} & \frac{\partial p_x}{\partial \theta_B} & \frac{\partial p_x}{\partial \theta_C} \\ \frac{\partial p_y}{\partial \theta_A} & \frac{\partial p_y}{\partial \theta_B} & \frac{\partial p_y}{\partial \theta_C} \\ \frac{\partial p_z}{\partial \theta_A} & \frac{\partial p_z}{\partial \theta_B} & \frac{\partial p_z}{\partial \theta_C} \end{bmatrix}$$

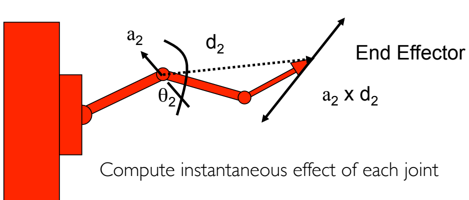
https://medium.com/unity3danimation/overview-of-jacobian-ik-a33939639ab2

Finding J

- Analytical solution
  - 3-DOF example

$$J = \begin{bmatrix} (R_A \times (E-A))_X & (R_B \times (E-B))_X & (R_C \times (E-C))_X \\ (R_A \times (E-A))_Y & (R_B \times (E-B))_Y & (R_C \times (E-C))_Y \\ (R_A \times (E-A))_Z & (R_B \times (E-B))_Z & (R_C \times (E-C))_Z \end{bmatrix}$$

Inverse Jacobian Method

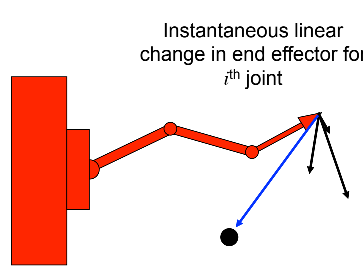


Compute instantaneous effect of each joint

Linear approximation to curvilinear motion

Find linear combination to take end effector **towards** goal position

Inverse Jacobian Method



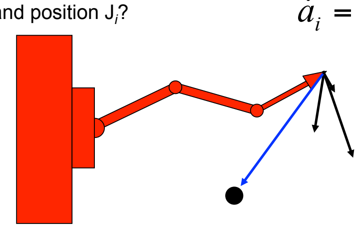
Inverse Jacobian Method

25

What is the change in orientation of end effector induced by joint *i* that has axis of rotation a_i and position J_i?

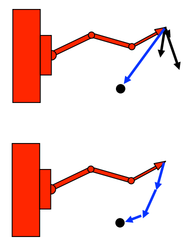
Angular velocity

$$\dot{a}_i = \omega_i$$



Inverse Jacobian Method

26



Solution only valid for an instantaneous step

Angular affect is really curved, not straight line

Once a step is taken, need to recompute solution

Inverse Jacobian: Math

27

Set up equations

y_i: state variable

x_i : system parameter

f_i : relate system parameters to state variable

$$y_1 = f_1(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_2 = f_2(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_3 = f_3(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_4 = f_4(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_5 = f_5(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_6 = f_6(x_1, x_2, x_3, x_4, x_5, x_6)$$

Inverse Jacobian: Math

28

$$y_1 = f_1(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_2 = f_2(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_3 = f_3(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_4 = f_4(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_5 = f_5(x_1, x_2, x_3, x_4, x_5, x_6)$$
$$y_6 = f_6(x_1, x_2, x_3, x_4, x_5, x_6)$$

Matrix Form $\rightarrow$ $\boxed{Y = F(X)}$

Inverse Jacobian: Math

29

$$y_i = f_i(x_1, x_2, x_3, x_4, x_5, x_6)$$

Use chain rule to differentiate equations to relate changes in system parameters to changes in state variables

$$dy_i = \frac{\partial f_i}{\partial x_1} dx_1 + \frac{\partial f_i}{\partial x_2} dx_2 + \frac{\partial f_i}{\partial x_3} dx_3 + \frac{\partial f_i}{\partial x_4} dx_4 + \frac{\partial f_i}{\partial x_5} dx_5 + \frac{\partial f_i}{\partial x_6} dx_6$$

Inverse Jacobian: Math

30

$$\partial y_i = \frac{\partial f_i}{\partial x_1} dx_1 + \frac{\partial f_i}{\partial x_2} dx_2 + \frac{\partial f_i}{\partial x_3} dx_3 + \frac{\partial f_i}{\partial x_4} dx_4 + \frac{\partial f_i}{\partial x_5} dx_5 + \frac{\partial f_i}{\partial x_6} dx_6$$

Matrix Form

$$Y = F(X)$$

$$\boxed{dY = \frac{\partial F}{\partial X} dX}$$

Inverse Jacobian: Math

31

$$dY = \frac{\partial F}{\partial X} dX$$

Change in position (and orientation) of end effector

Change in joint angles

Linear approximation that relates change in joint angle to change in end effector position (and orientation)

Inverse Jacobian: Math

32

- We've now derived the form we saw initially.

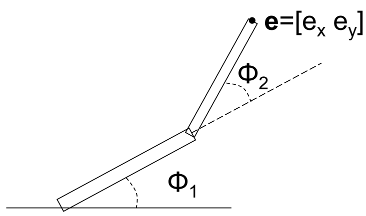
$$dY = \frac{\partial F}{\partial X} dX$$

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \frac{\partial p_x}{\partial \theta_1} & \frac{\partial p_x}{\partial \theta_2} & \cdots & \frac{\partial p_x}{\partial \theta_n} \\ \frac{\partial p_y}{\partial \theta_1} & \frac{\partial p_y}{\partial \theta_2} & \cdots & \frac{\partial p_y}{\partial \theta_n} \\ \frac{\partial p_z}{\partial \theta_1} & \cdots & \cdots & \cdots \\ \frac{\partial a_x}{\partial \theta_1} & \cdots & \cdots & \cdots \\ \frac{\partial a_y}{\partial \theta_1} & \cdots & \cdots & \cdots \\ \frac{\partial a_z}{\partial \theta_1} & \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \cdots \\ \dot{\theta}_n \end{bmatrix}$$

Jacobians

33

- Let's say we have a simple 2D robot arm with two 1-DOF rotational joints:



Jacobians

34

- The Jacobian matrix J(**e**,**Φ**) shows how each component of **e** varies wrt. each joint angle

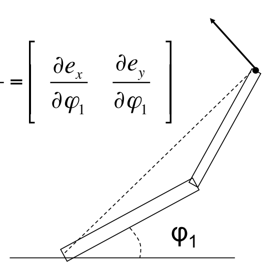
$$J(\mathbf{e}, \mathbf{\Phi}) = \begin{bmatrix} \frac{\partial e_x}{\partial \phi_1} & \frac{\partial e_x}{\partial \phi_2} \\ \frac{\partial e_y}{\partial \phi_1} & \frac{\partial e_y}{\partial \phi_2} \end{bmatrix}$$

Jacobians

35

- Consider what would happen if we increased ϕ_1 by a small amount. What would happen to **e** ?

$$\frac{\partial \mathbf{e}}{\partial \varphi_1} = \begin{bmatrix} \frac{\partial e_x}{\partial \varphi_1} & \frac{\partial e_y}{\partial \varphi_1} \end{bmatrix}$$

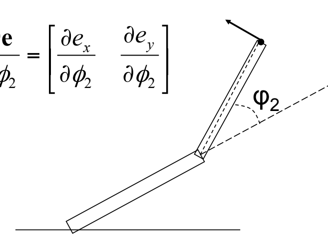


Jacobians

36

- What if we increased ϕ_2 by a small amount?

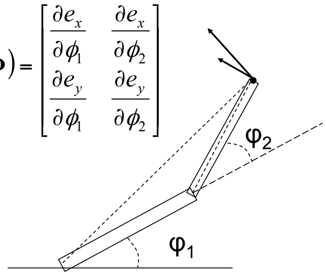
$$\frac{\partial \mathbf{e}}{\partial \phi_2} = \begin{bmatrix} \frac{\partial e_x}{\partial \phi_2} & \frac{\partial e_y}{\partial \phi_2} \end{bmatrix}$$



Jacobian for a 2D Robot Arm

37

$$J(\mathbf{e},\mathbf{\Phi}) = \begin{bmatrix} \frac{\partial e_x}{\partial \phi_1} & \frac{\partial e_x}{\partial \phi_2} \\ \frac{\partial e_y}{\partial \phi_1} & \frac{\partial e_y}{\partial \phi_2} \end{bmatrix}$$



Other Numeric IK

38

- Jacobian transpose
- Alternate Jacobian – use goal position
- HAL – human arm linkage
- Damped Least Squares
- CCD