CEng 732 Computer Animation

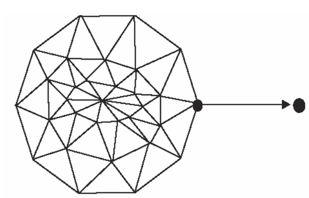
Spring 2006-2007
Week 4
Shape Deformation
Animating Articulated Structures:
Forward Kinematics/Inverse Kinematics

This week

- Shape Deformation
  - FFD: Free Form Deformation
- Hierarchical Modeling of Articulated Objects
- Forward Kinematics
- Local Coordinate Frames
  - Denavit-Hartenberg Notation
- Inverse Kinematics

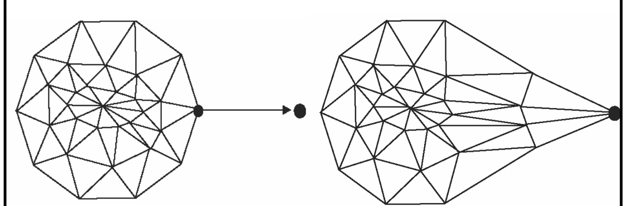
Warping an Object

- Displace one vertex of an object
  - And as a consequence make neighbor vertices move with the displaced vertex



Warping an Object

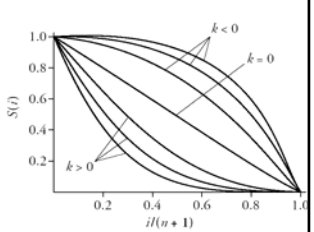
- Displace one vertex of an object
  - And as a consequence make neighbor vertices move with the displaced vertex



Warping an Object

- Use an attenuation function to determine the amount of displacement for the other vertices:

$$S(i) = 1.0 - \left(\frac{i}{n+1}\right)^{k+1} \qquad k \geq 0$$
$$= \left(1.0 - \left(\frac{i}{n+1}\right)\right)^{-k+1} \qquad k < 0$$



2D Grid Deformation

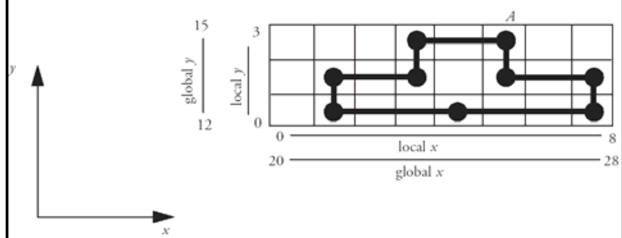
- Initially construct a 2D grid around the object as a local coordinate system aligned with the global axes
  - Global to local transformation can be done by simple translate and scale
- Then distort the grid by moving the vertices of the grid.
  - This will distort the local coordinate system and hence the vertices of the object will be relocated in the global coordinate system

2D Grid Deformation

- The location of the object vertex is found using bilinear interpolation

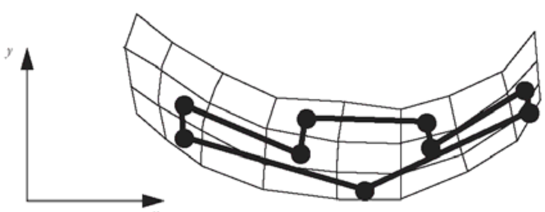
2D Grid Deformation

- Initial grid



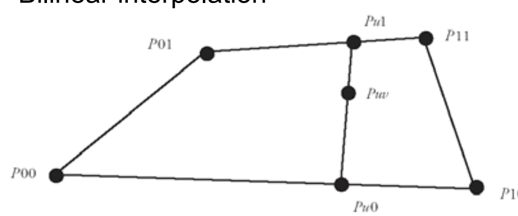
2D Grid Deformation

- Deformed grid



2D Grid Deformation

- Bilinear interpolation



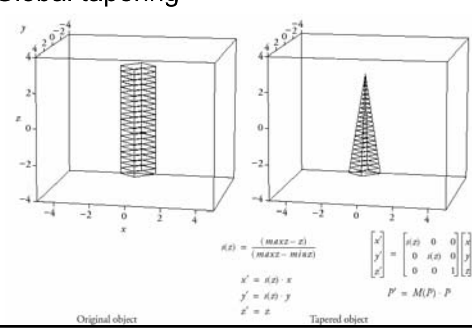
$$Pu0 = (1-u)\cdot P00 + u\cdot P10$$
$$Pu1 = (1-u)\cdot P01 + u\cdot P11$$
$$Puv = (1-v)\cdot Pu0 + v\cdot Pu1$$
$$= (1-u)\cdot(1-v)\cdot P00 + (1-u)\cdot v\cdot P01 + u\cdot(1-v)\cdot P10 + u\cdot v\cdot P11$$

Global Deformations

- Apply a 3x3 transformation matrix to all the vertices of an object

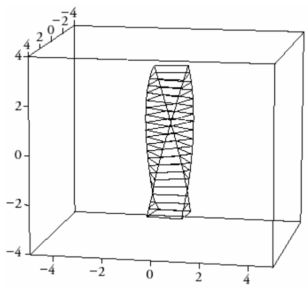
Global Deformations

- Global tapering



Global Deformations

- Twist about an axis



k = twist factor

$$x' = x \cdot \cos(k \cdot z) - y \cdot \sin(k \cdot z)$$
$$y' = x \cdot \sin(k \cdot z) + y \cdot \cos(k \cdot z)$$
$$z' = z$$

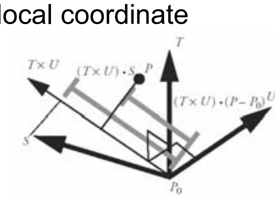
Free-Form Deformation

- 3D extension of the 2D grid deformation technique
- Usually cubic interpolation is used instead of linear interpolation
- A local coordinate system is defined by S,T,U vectors (and an origin)
  - S,T,U are not necessarily orthogonal
- S,T,U axes are uniformly divided into a grid to facilitate manipulation of the coordinate system

Free-Form Deformation

- Locating a point in the local coordinate system

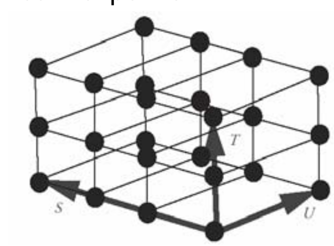
$$s = (T \times U) \bullet (P - P0) / ((T \times U) \bullet S)$$
$$t = (U \times S) \bullet (P - P0) / ((U \times S) \bullet T)$$
$$u = (S \times T) \bullet (P - P0) / ((S \times T) \bullet U)$$



$$P = P0 + s \cdot S + t \cdot T + u \cdot U$$

Free-Form Deformation

- Grid of control points



$$P_{ijk} = P_0 + \frac{i}{l} \cdot S + \frac{j}{m} \cdot T + \frac{k}{n} \cdot U$$

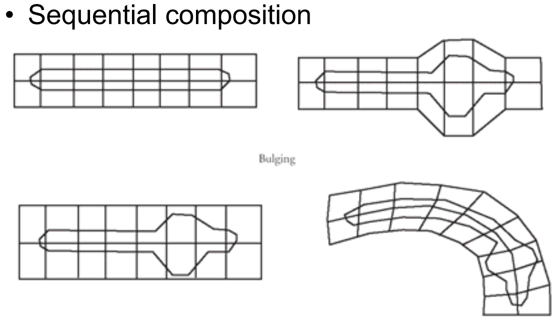
Free-Form Deformation

- Bezier interpolation is used two find the deformed global coordinate of the object vertex, given the global coordinates of the grid control points

$$P(s, t, u) = \sum_{i=0}^{l} \binom{l}{i} (1-s)^{l-i} s^i \cdot \left(\sum_{j=0}^{m} \binom{m}{j} (1-t)^{m-j} t^j \cdot \left(\sum_{k=0}^{n} \binom{n}{k} (1-u)^{n-k} u^k P_{ijk} \right) \right)$$

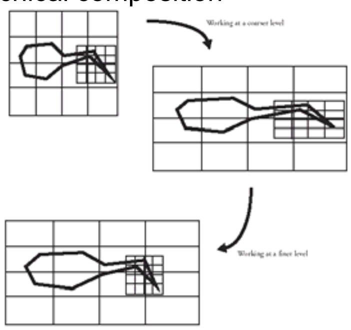
FFD composition

- Sequential composition



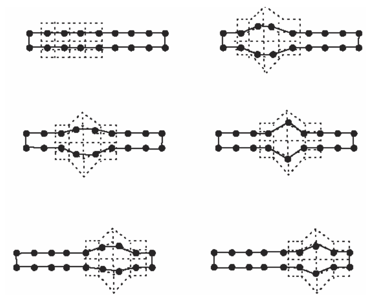
FFD composition

- Hierarchical composition



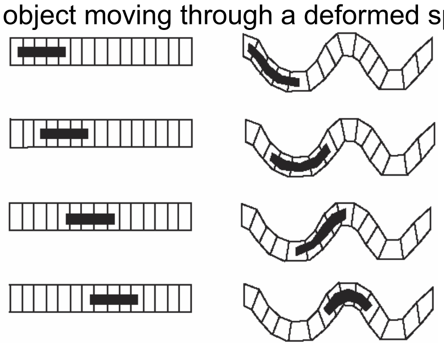
Animation using FFDs

- Moving a distorted grid over an object



Animation using FFDs

- An object moving through a deformed space



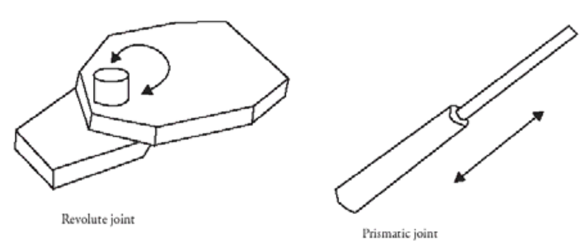
Hierarchical Kinematic Modeling

- Kinematics:
  - Studying the movement of objects without considering the forces involved in producing the movement
- Dynamics
  - Studying the underlying forces that produce the movement
- Hierarchical modeling
  - Organizing objects in a treelike structure and specifying movement parameters between their components

Some definitions

- Articulated objects
  - Hierarchical objects connected end to end to form multibody jointed chains
  - Manipulators: a sequence of objects connected in a chain by joints. Example: robot arm
- The rigid objects between joints are called *links*. The last *link* in a series of links is called the *end effector* (e.g. the hand of a robot arm)
- The local coordinate system associated with each joint is referred to as the *frame*.

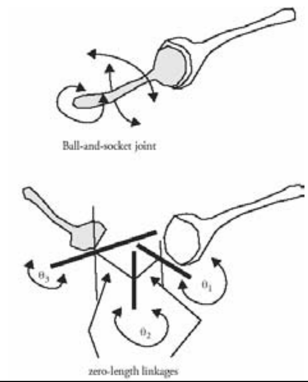
Types of joints



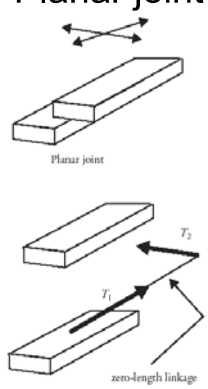
Simple vs. Complex Joints

- Joints that allow motion in one directions have one degree of freedom
- Complex joints have more degrees of freedom and they can be represented as a series simple joints connected to each other by zero length links.
  - Examples:
    - Ball-and-socket joint (3 DOF)
    - Planar joint (2 DOF)

Ball-and-socket joint



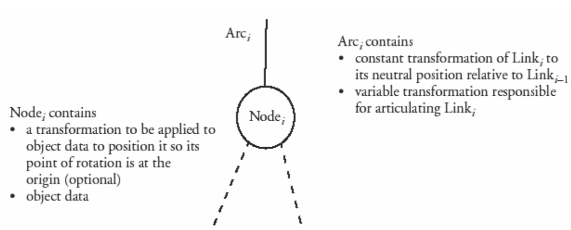
Planar joint



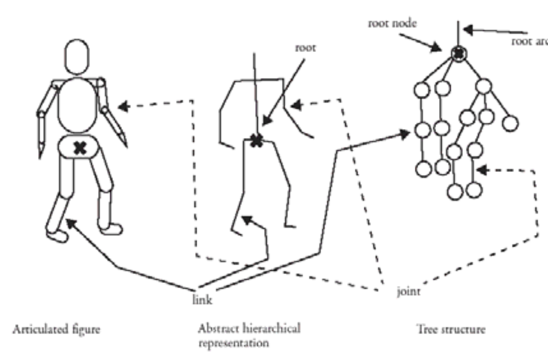
Hierarchical Models

- Represented as trees
  - Nodes connected by arcs
- The highest node of the tree is called the root node which corresponds to the root object whose position is known in the global coordinate system
- The position of an intermediate node in the tree can be found by position of the root node and the transformations on the path from root to that node
- Nodes represent object parts (i.e., links)
- Arcs represent joints

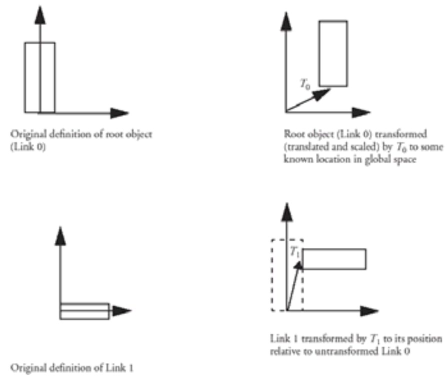
Information stored in nodes and arcs



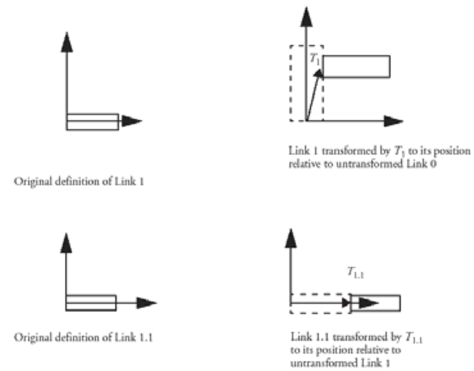
A hierarchical model



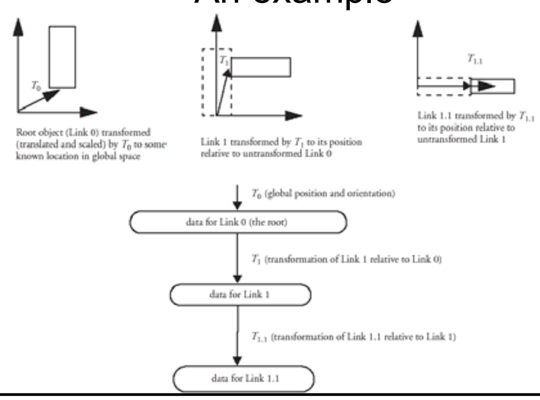
An example



An example



An example



Positions of vertices

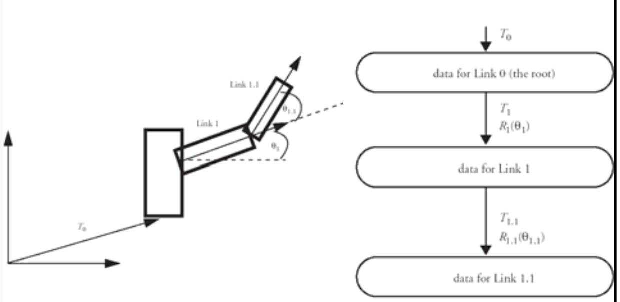
- Are found by traversing the tree from top to bottom and concatenating the transformations at the joints

$$V_0' = T_0 \cdot V_0$$

$$V_1' = T_0 \cdot T_1 \cdot V_1$$

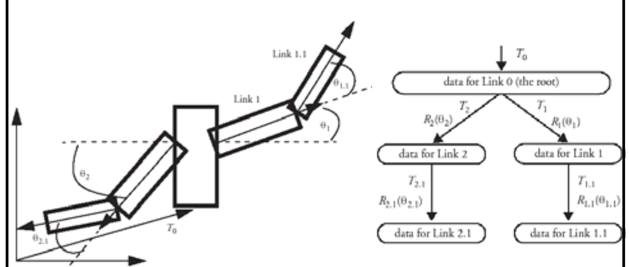
$$V_{1.1}' = T_0 \cdot T_1 \cdot T_{1.1} \cdot V_{1.1}$$

Rotations at the joints



$$V_{1.1}' = T_0 \cdot T_1 \cdot R_1(\theta_1) \cdot T_{1.1} \cdot R_{1.1}(\theta_{1.1}) \cdot V_{1.1}$$

Two appendages



Forward Kinematics

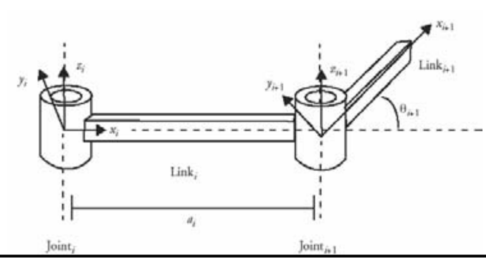
- Finding the location (and orientation) of the end effector(s) by applying all the joint transformations sequentially
  - All the intermediate joint angles are given by the user
- Depth-first tree traversal of the tree representations and a stack to store intermediate composition of transformation matrices is used
  - OpenGL's pushMatrix/popMatrix functions can be used easily to accomplish this

Local Coordinate Frames

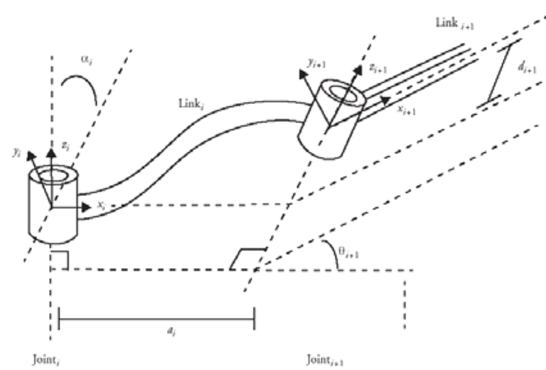
- Denavit-Hartenberg Notation from robotics
- A local coordinate frame around a joint is represented by four variables:
  - Link offset
  - Joint angle
  - Link length
  - Link twist

Simple case

- When two successive joints and the axis of rotation are co-planar
  - Link offset and link twist is zero



General case



Denavit-Hartenberg Parameters

Table 4.1 Denavit-Hartenberg Joint Parameters for Joint i

Name	Symbol	Description
Link offset	d_i	distance from x_{i-1} to x_i along z_i
Joint angle	θ_i	angle between x_{i-1} and x_i about z_i
Link length	a_i	distance from z_i to z_{i+1} along x_i
Link twist	α_i	angle between z_i and z_{i+1} about x_i

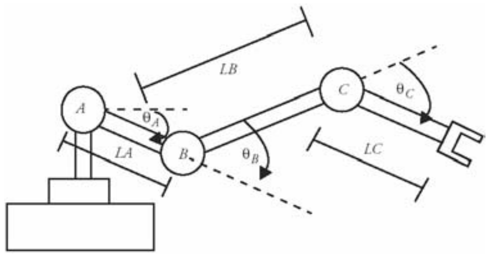
Relating two successive frames

Table 4.2 Parameters That Relate the ith Frame and the $i+1$ Frame

Name	Symbol	Description	Screw Transformation
Link offset	d_{i+1}	distance from x_i to x_{i+1} along z_{i+1}	relative to z_{i+1}
Joint angle	θ_{i+1}	angle between x_i and x_{i+1} about z_{i+1}	relative to z_{i+1}
Link length	a_i	distance from z_i to z_{i+1} along x_i	relative to x_i
Link twist	α_i	angle between z_i and z_{i+1} about x_i	relative to x_i

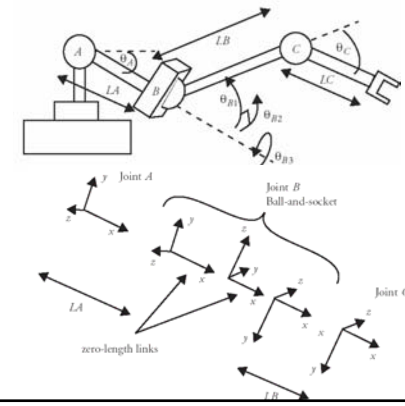
$$V_i = T_X(a_i) R_X(\alpha_i) T_Z(d_{i+1}) R_Z(\theta_{i+1}) V_{i+1}$$

A simple example

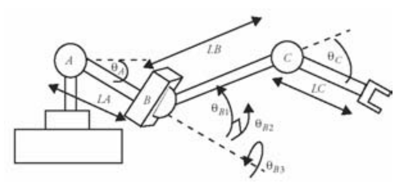


Joint/Parameter	Link Displacement	Joint Angle	Link Length	Link Twist
A	0	θ_A	0	0
B	0	θ_B	LA	0
C	0	θ_C	LB	0

A Ball-and-socket joint



Joint Parameters

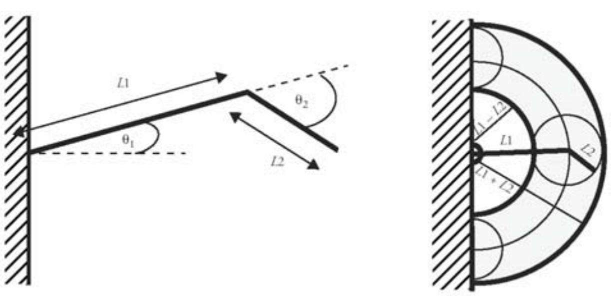


Joint/Parameter	Link Displacement	Joint Angle	Link Length	Link Twist
A	0	θ_A	0	0
$B1$	0	θ_{B1}	LA	90
$B2$	0	$90 + \theta_{B2}$	0	90
$B3$	0	θ_{B3}	0	0
C	0	θ_C	LB	0

Inverse Kinematics

- Find the intermediate joint angles given the position and orientation of the end effector
  - Some constraints may also be given
    - E.g., joint angles in a range
- There may be no solutions
  - Overconstrained
- There may be multiple solutions
  - Underconstrained

Analytic computation for simple cases

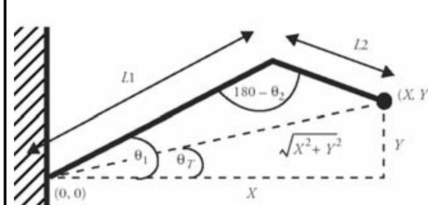


Configuration

Reachable workspace

Given (x,y) coordinate of the end point of the end effector, compute θ_1 and θ_2

Solution



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

$$\theta_T = \operatorname{acos}\left(\frac{X}{\sqrt{X^2+Y^2}}\right)$$

$$\cos(\theta_1 - \theta_T) = \frac{L1^2 + X^2 + Y^2 - L2^2}{2 \cdot L1 \cdot \sqrt{X^2+Y^2}}$$

$$\cos(180 - \theta_2) = \frac{L1^2 + L2^2 - (X^2+Y^2)}{2 \cdot L1 \cdot L2}$$

$$\theta_1 = \operatorname{acos}\left(\frac{L1^2 + X^2 + Y^2 - L2^2}{2 \cdot L1 \cdot \sqrt{X^2+Y^2}}\right) + \theta_T$$

$$\theta_2 = \operatorname{acos}\left(\left(\frac{L1^2 + L2^2 - (X^2+Y^2)}{2 \cdot L1 \cdot L2}\right)\right)$$

The Jacobian

- In many complex joints however, such analytic solutions are not possible.
- Therefore we use the Jacobian matrix to find the correct joint angle increments that will lead us to the final end effector configuration
- The Jacobian matrix is a matrix of partial derivatives
  - Each entry shows how much the change in an input parameter effects an output parameter

Example Jacobian

$$x_1 = r \sin\phi \cos\theta$$
$$x_2 = r \sin\phi \sin\theta$$
$$x_3 = r \cos\phi$$

$$J_F(r,\phi,\theta) = \begin{bmatrix} \frac{\partial x_1}{\partial r} & \frac{\partial x_1}{\partial \phi} & \frac{\partial x_1}{\partial \theta} \\ \frac{\partial x_2}{\partial r} & \frac{\partial x_2}{\partial \phi} & \frac{\partial x_2}{\partial \theta} \\ \frac{\partial x_3}{\partial r} & \frac{\partial x_3}{\partial \phi} & \frac{\partial x_3}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \sin\phi\cos\theta & r\cos\phi\cos\theta & -r\sin\phi\sin\theta \\ \sin\phi\sin\theta & r\cos\phi\sin\theta & r\sin\phi\cos\theta \\ \cos\phi & -r\sin\phi & 0 \end{bmatrix}.$$

Example Jacobian

$$y_1 = x_1$$
$$y_2 = 5x_3$$
$$y_3 = 4x_2^2 - 2x_3$$
$$y_4 = x_3 \sin(x_1)$$

$$J_F(x_1,x_2,x_3) = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \frac{\partial y_1}{\partial x_2} & \frac{\partial y_1}{\partial x_3} \\ \frac{\partial y_2}{\partial x_1} & \frac{\partial y_2}{\partial x_2} & \frac{\partial y_2}{\partial x_3} \\ \frac{\partial y_3}{\partial x_1} & \frac{\partial y_3}{\partial x_2} & \frac{\partial y_3}{\partial x_3} \\ \frac{\partial y_4}{\partial x_1} & \frac{\partial y_4}{\partial x_2} & \frac{\partial y_4}{\partial x_3} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 5 \\ 0 & 8x_2 & -2 \\ x_3\cos(x_1) & 0 & \sin(x_1) \end{bmatrix}.$$

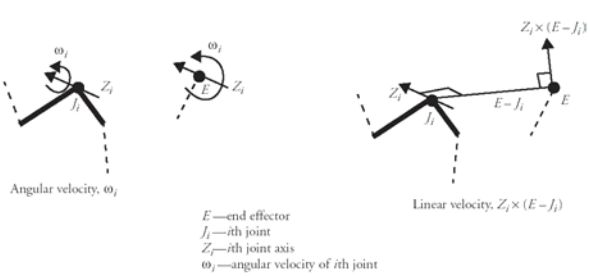
Using the Jacobian

$$V = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^T$$
$$\dot{\theta} = [\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \ldots, \dot{\theta}_n]^T$$

$$V = J(\theta)\dot{\theta}$$
$$J^{-1}V = \dot{\theta}$$

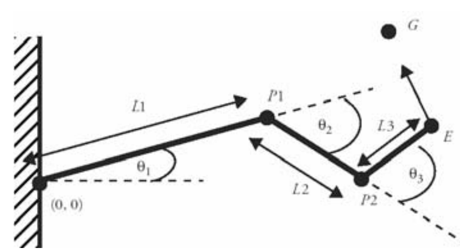
$$J = \begin{bmatrix} \frac{\partial v_x}{\partial \theta_1} & \frac{\partial v_x}{\partial \theta_2} & \ldots & \frac{\partial v_x}{\partial \theta_n} \\ \frac{\partial v_y}{\partial \theta_1} & \frac{\partial v_y}{\partial \theta_2} & \ldots & \frac{\partial v_y}{\partial \theta_n} \\ \ldots & \ldots & \ldots & \ldots \\ \frac{\partial \omega_z}{\partial \theta_1} & \frac{\partial \omega_z}{\partial \theta_2} & \ldots & \frac{\partial \omega_z}{\partial \theta_n} \end{bmatrix}$$

Computing the Jacobian

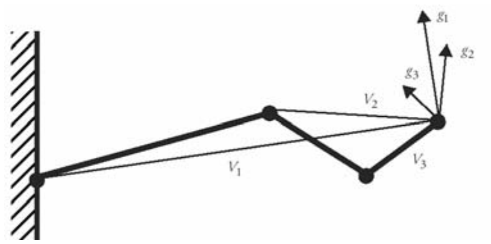


A simple example

- Assume we want to find the change in the rotation angles to get the end effector to G



Effect of changing θs



The equation

$$\begin{bmatrix} (G-E)_x \\ (G-E)_y \\ (G-E)_z \end{bmatrix} = \begin{bmatrix} ((0,0,1)\times E)_x & (0,0,1)\times(E-P_1)_x & (0,0,1)\times(E-P_2)_x \\ ((0,0,1)\times E)_y & (0,0,1)\times(E-P_1)_y & (0,0,1)\times(E-P_2)_y \\ ((0,0,1)\times E)_z & (0,0,1)\times(E-P_1)_z & (0,0,1)\times(E-P_2)_z \end{bmatrix} \cdot \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix}$$

Solving for θs

$$V = J\dot{\theta}$$

$$J^{-1}V = \dot{\theta}$$

If J is not a square matrix

- Use the pseudo-inverse to compute the joint angles

$$V = J\dot{\theta}$$

$$J^T V = J^T J\dot{\theta}$$

$$(J^T J)^{-1} J^T V = (J^T J)^{-1} J^T J\dot{\theta}$$

$$J^+ V = \dot{\theta}$$

Inverse Kinematics Videos

- Video 1
- Video 2