# Robot Kinematics

METR 4202: Advanced Control & Robotics  
Drs Surya Singh, Paul Pounds, and Hanna Kurniawati  
Lecture # 3  
August 6, 2012

metr4202@itee.uq.edu.au  
http://itee.uq.edu.au/~metr4202/

---

## Schedule

<table>
<thead>
<tr>
<th>Week</th>
<th>Date</th>
<th>Lecture (M: 12-1:30, 43-102)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23-Jul</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>30-Jul</td>
<td>Representing Position &amp; Orientation &amp; State (Frames, Transformation Matrices &amp; Affine Transformations)</td>
</tr>
<tr>
<td>3</td>
<td>6-Aug</td>
<td><strong>Robot Kinematics and Dynamics</strong></td>
</tr>
<tr>
<td>4</td>
<td>13-Aug</td>
<td>Robot Dynamics &amp; Control</td>
</tr>
<tr>
<td>5</td>
<td>20-Aug</td>
<td>Obstacle Avoidance &amp; Motion Planning</td>
</tr>
<tr>
<td>6</td>
<td>27-Aug</td>
<td>Sensors, Measurement and Perception</td>
</tr>
<tr>
<td>7</td>
<td>3-Sep</td>
<td>Localization and Navigation</td>
</tr>
<tr>
<td>8</td>
<td>10-Sep</td>
<td>State-space modelling &amp; Controller Design</td>
</tr>
<tr>
<td>9</td>
<td>17-Sep</td>
<td>Vision-based control</td>
</tr>
<tr>
<td></td>
<td>24-Sep</td>
<td><em>Study break</em></td>
</tr>
<tr>
<td>10</td>
<td>1-Oct</td>
<td>Uncertainty/POMDPs</td>
</tr>
<tr>
<td>11</td>
<td>8-Oct</td>
<td>Robot Machine Learning</td>
</tr>
<tr>
<td>12</td>
<td>15-Oct</td>
<td>Guest Lecture</td>
</tr>
<tr>
<td>13</td>
<td>22-Oct</td>
<td>Wrap-up &amp; Course Review</td>
</tr>
</tbody>
</table>
Recap from Last Week [1]

- A **position** vector specifies the location of a **point** in 3D (Cartesian) space

\[
P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}
\]

- **BUT** we also concerned with its orientation in 3D space. This is specified as a matrix based on each frame's **unit vectors**

\[
A\mathbf{p}_B = A\mathbf{p}_B = \begin{bmatrix} Bp_x \\ Bp_y \\ Bp_z \end{bmatrix} - \begin{bmatrix} Ap_x \\ Ap_y \\ Ap_z \end{bmatrix}
\]

Recap from last week [2]…

- The components of a rotation matrix are the unit vectors projected **onto** the unit directions of the reference frame

\[
\begin{bmatrix}
\mathbf{\hat{i}}_B \cdot \mathbf{\hat{i}}_A \\
\mathbf{\hat{j}}_B \cdot \mathbf{\hat{i}}_A \\
\mathbf{\hat{k}}_B \cdot \mathbf{\hat{i}}_A \\
\mathbf{\hat{i}}_B \cdot \mathbf{\hat{j}}_A \\
\mathbf{\hat{j}}_B \cdot \mathbf{\hat{j}}_A \\
\mathbf{\hat{k}}_B \cdot \mathbf{\hat{j}}_A \\
\mathbf{\hat{i}}_B \cdot \mathbf{\hat{k}}_A \\
\mathbf{\hat{j}}_B \cdot \mathbf{\hat{k}}_A \\
\mathbf{\hat{k}}_B \cdot \mathbf{\hat{k}}_A
\end{bmatrix}
\]
Recap from Last Week [3]

Rotation is orthonormal \( \therefore \)

- The \textbf{rows} are \{A\} \textbf{written in} \{B\}

\[
\begin{align*}
\begin{array}{c}
\frac{B}{A} \mathbf{R} = \frac{A}{B} \mathbf{R}^T = \frac{A}{B} \mathbf{R}^{-1}
\end{array}
\end{align*}
\]

- The \textbf{of a rotation matrix inverse} = the transpose

\( \mathbf{R} \cdot \mathbf{R}^T = 1 \)

- \( \therefore \) \textbf{of normality} \( \rightarrow \) the determinant = 1

\[
\text{det} \left( \mathbf{R} \right) = 1
\]

Recall from Last Week [4]…

- In many Kinematics References:
- In many Engineering Applications:

\[ \begin{array}{ccc}
X & y & z \\
\text{roll} & \text{yaw} & \text{pitch}
\end{array} \]

\[ \begin{array}{ccc}
x & y & z \\
\text{roll} & \text{yaw} & \text{pitch}
\end{array} \]

\( \rightarrow \) \textbf{Be careful:}

This name is given to other conventions too!
Coordinate Transformations [1]

- Translation Again:
  If \{B\} is translated with respect to \{A\} \textbf{without rotation}, then it is a vector sum

\[
A_P = A_P^B + B_P
\]

Coordinate Transformations [2]

- Rotation Again:
  \{B\} is rotated with respect to \{A\} then
  use rotation matrix to determine new components

\[
A_P = A_P^B R^B P
\]

- NOTE:
  - The Rotation matrix’s \textit{subscript} matches the position vector’s \textit{superscript}

\[
A_P = A_P^B R^B P
\]
  - This gives Point Positions of \{B\} ORIENTED in \{A\}
Coordinate Transformations [3]

- Composite transformation:
  \( \{B\} \) is moved with respect to \( \{A\} \):

\[
A_P = A_P^B + A_B \mathbf{R}^B P
\]

Homogenous Coordinates

\[
\hat{p} = \begin{bmatrix}
\rho p_x \\
\rho p_y \\
\rho p_z \\
\rho
\end{bmatrix}^T
\]

- \( \rho \) is a scaling value
Homogenous Transformation

\[
\begin{bmatrix}
A R_B & A p \\
\gamma & \rho
\end{bmatrix}
\]

- $\gamma$ is a projective transformation
- $\rho$ is a scaling value

General Coordinate Transformations [1]

A compact representation of the translation and rotation is known as the Homogeneous Transformation

\[
{A_B}^T = \begin{bmatrix}
{A_B} R \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

- This allows us to cast the rotation and translation of the general transform in a single matrix form

\[
\begin{bmatrix}
{A_B} p \\
1
\end{bmatrix} = {A_B}^T \begin{bmatrix}
{B_B} p \\
1
\end{bmatrix}
\]
General Coordinate Transformations [2]

Fundamental orthonormal transformations can be represented in this form too:

\[
\begin{align*}
\text{Trans}(u, v, w) &= \begin{bmatrix} 1 & 0 & 0 & u \\ 0 & 1 & 0 & v \\ 0 & 0 & 1 & w \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
\text{Rotz}(\theta) &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\theta & -s\theta & 0 \\ 0 & s\theta & c\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
\text{Roty}(\phi) &= \begin{bmatrix} c\phi & 0 & s\phi & 0 \\ 0 & 1 & 0 & 0 \\ -s\phi & 0 & c\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{align*}
\]

General Coordinate Transformations [3]

- Multiple transformations compounded as a chain

\[
\begin{align*}
\mathbf{B}_P &= \mathbf{B}_C \mathbf{T}_C \mathbf{P} \\
\mathbf{A}_P &= \mathbf{A}_B \mathbf{T}_B \mathbf{P} \\
&= \mathbf{A}_B \mathbf{T}_C \mathbf{T}_C \mathbf{P} \\
&= \mathbf{A}_C \mathbf{T}_C \mathbf{P}
\end{align*}
\]
### Projective Transformations ...

<table>
<thead>
<tr>
<th>Group</th>
<th>Matrix</th>
<th>Distortion</th>
<th>Invariant properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projective</td>
<td>$\begin{bmatrix} h_{11} &amp; h_{12} &amp; h_{13} \ h_{21} &amp; h_{22} &amp; h_{23} \ h_{31} &amp; h_{32} &amp; h_{33} \end{bmatrix}$</td>
<td>Concurrency, collinearity, order of contact: intersection (1 pt contact); tangency (2 pt contact); inflections (3 pt contact with line); tangent discontinuities and cusps. cross ratio (ratio of ratio of lengths).</td>
<td></td>
</tr>
<tr>
<td>8 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Affine</td>
<td>$\begin{bmatrix} a_{11} &amp; a_{12} &amp; t_x \ a_{21} &amp; a_{22} &amp; t_y \ 0 &amp; 0 &amp; 1 \end{bmatrix}$</td>
<td>Parallelism, ratio of areas, ratio of lengths on collinear or parallel lines (e.g. midpoints), linear combinations of vectors (e.g. centroids). The line at infinity, $I_\infty$.</td>
<td></td>
</tr>
<tr>
<td>6 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Similarity</td>
<td>$\begin{bmatrix} s_{11} &amp; s_{12} &amp; t_x \ s_{21} &amp; s_{22} &amp; t_y \ 0 &amp; 0 &amp; 1 \end{bmatrix}$</td>
<td>Ratio of lengths, angle. The circular points, I, J (see section 2.7.3).</td>
<td></td>
</tr>
<tr>
<td>4 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Euclidean</td>
<td>$\begin{bmatrix} f_{11} &amp; f_{12} &amp; t_x \ f_{21} &amp; f_{22} &amp; t_y \ 0 &amp; 0 &amp; 1 \end{bmatrix}$</td>
<td>Length, area</td>
<td></td>
</tr>
<tr>
<td>3 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

p.44, R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*

---

### Projective Transformations & Other Transformations of 3D Space

<table>
<thead>
<tr>
<th>Group</th>
<th>Matrix</th>
<th>Distortion</th>
<th>Invariant properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Projective</td>
<td>$\begin{bmatrix} A &amp; t \ v^T &amp; e \end{bmatrix}$</td>
<td>Intersection and tangency of surfaces in contact. Sign of Gaussian curvature.</td>
<td></td>
</tr>
<tr>
<td>15 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Affine</td>
<td>$\begin{bmatrix} A &amp; t \ 0^T &amp; 1 \end{bmatrix}$</td>
<td>Parallelism of planes, volume ratios, centroids. The plane at infinity, $\pi_w$, (see section 3.5).</td>
<td></td>
</tr>
<tr>
<td>12 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Similarity</td>
<td>$\begin{bmatrix} s &amp; t \ 0^T &amp; 1 \end{bmatrix}$</td>
<td>The absolute conic, $n_w$, (see section 3.6).</td>
<td></td>
</tr>
<tr>
<td>7 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Euclidean</td>
<td>$\begin{bmatrix} R &amp; t \ 0^T &amp; 1 \end{bmatrix}$</td>
<td>Volume.</td>
<td></td>
</tr>
<tr>
<td>6 dof</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

p.78, R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*
Generalizing
Special Orthogonal & Special Euclidean Lie Algebras

• SO(n): Rotations

\[ SO(n) = \{ R \in \mathbb{R}^{n \times n} : R^T \mathbf{R} = \mathbf{I}, \det R = +1 \} . \]

\[ \exp(\mathbf{\omega}) = I + \mathbf{\omega} + \frac{\mathbf{\omega}^2}{2!} + \frac{\mathbf{\omega}^3}{3!} + \ldots \]

• SE(n): Transformations of EUCLIDEAN space

\[ SE(n) := \mathbb{R}^n \times SO(n) . \]

\[ SE(3) = \{ (p, R) : p \in \mathbb{R}^3, R \in SO(3) \} = \mathbb{R}^3 \times SO(3) . \]

Screw Displacements

• Comes from the notion that all motion can be viewed as a rotation (Rodrigues formula)

• Define a vector along the axis of motion (screw vector)
  – Rotation (screw angle)
  – Translation (pitch)
  – Summations \[ \rightarrow \] via the screw triangle!
Denavit Hartenberg [DH] Notation

• J. Denavit and R. S. Hartenberg first proposed the use of homogeneous transforms for articulated mechanisms
  (But B. Roth, introduced it to robotics)

• A kinematics “short-cut” that reduced the number of parameters by adding a structure to frame selection

• For two frames positioned in space, the first can be moved into coincidence with the second by a sequence of 4 operations:
  – rotate around the $x_{i-1}$ axis by an angle $\alpha_i$
  – translate along the $x_{i-1}$ axis by a distance $a_i$
  – translate along the new $z$ axis by a distance $d_i$
  – rotate around the new $z$ axis by an angle $\theta_i$
Denavit-Hartenberg Convention

- **link length** $a_i$, the offset distance between the $z_{i-1}$ and $z_i$ axes along the $x_i$ axis;
- **link twist** $\alpha_i$, the angle from the $z_{i-1}$ axis to the $z_i$ axis about the $x_i$ axis;
- **link offset** $d_i$, the distance from the origin of frame $i-1$ to the $x_i$ axis along the $z_{i-1}$ axis;
- **joint angle** $\theta_i$, the angle between the $x_{i-1}$ and $x_i$ axes about the $z_{i-1}$ axis.

**DH: Where to place frame?**

1. Align an axis along principal motion
   - 1. **Rotary (R)**: align rotation axis along the $z$ axis
   - 2. **Prismatic (P)**: align slider travel along $x$ axis

2. Orient so as to position $x$ axis towards next frame

3. $\theta_{(\text{rot } z)} \rightarrow d_{(\text{trans } z)} \rightarrow a_{(\text{trans } x)} \rightarrow \alpha_{(\text{rot } x)}$
Denavit-Hartenberg $\rightarrow$ Rotation Matrix

- Each transformation is a product of 4 “basic” transformations (instead of 6)

\[ i-1A_i = R_{z, \theta_i}T_{x, d_i}T_{x,a_i}R_{x, \alpha_i} \]

\[
\begin{bmatrix}
    c\theta_i & -s\theta_i & 0 & 0 \\
    s\theta_i & c\theta_i & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    1 & 0 & 0 & a_i \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & d_i \\
    0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & c\alpha_i & -s\alpha_i & 0 \\
    0 & s\alpha_i & c\alpha_i & 0 \\
    0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    c\theta_i & -s\theta_i & c\alpha_i & a_i c\theta_i \\
    s\theta_i & c\theta_i & s\alpha_i & a_i s\theta_i \\
    0 & s\alpha_i & c\alpha_i & d_i \\
    0 & 0 & 0 & 1
\end{bmatrix}
\]

DH Example [1]: RRR Link Manipulator

1. Assign the frames at the joints …
2. Fill DH Table …

<table>
<thead>
<tr>
<th>Link</th>
<th>$a_i$</th>
<th>$\alpha_i$</th>
<th>$d_i$</th>
<th>$\theta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$l_1$</td>
<td>0</td>
<td>0</td>
<td>$\theta_1$</td>
</tr>
<tr>
<td>2</td>
<td>$l_2$</td>
<td>0</td>
<td>0</td>
<td>$\theta_2$</td>
</tr>
<tr>
<td>3</td>
<td>$l_3$</td>
<td>0</td>
<td>0</td>
<td>$\theta_3$</td>
</tr>
</tbody>
</table>

\[ ^iA = ^{i-1}A_i ^{i-2}A_i ^{i-3}A_i \]

\[ ^0T_1 = ^0A_1 ^0A_2 ^0A_3 \]
DH Example [2]: RRP Link Manipulator

1. Assign the frames at the joints …
2. Fill DH Table …

<table>
<thead>
<tr>
<th>Link</th>
<th>(a_i)</th>
<th>(\alpha_i)</th>
<th>(d_i)</th>
<th>(\theta_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(L_1)</td>
<td>0</td>
<td>0</td>
<td>(\theta_1)</td>
</tr>
<tr>
<td>2</td>
<td>(L_2)</td>
<td>0</td>
<td>0</td>
<td>(\theta_2)</td>
</tr>
<tr>
<td>3</td>
<td>(L_3)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

\[ ^nA = \begin{bmatrix} c_{\alpha_1} & -s_{\alpha_1} & 0 & L_c \alpha_1 \\ s_{\alpha_1} & c_{\alpha_1} & 0 & L_s \alpha_1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad ^{n}A_i = \begin{bmatrix} c_{\alpha_i} & -s_{\alpha_i} & 0 & L_c \alpha_i \\ s_{\alpha_i} & c_{\alpha_i} & 0 & L_s \alpha_i \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad ^{n}A_2 = \begin{bmatrix} 1 & 0 & 0 & L_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

\[ ^0T_i = ^nA_1^1A_2^2A_i \]

\[ ^0T_i = \begin{bmatrix} c_{\alpha_i} & -s_{\alpha_i} & 0 & L_c \alpha_i + (L_2 + L_3)c_{\alpha_i} \\ s_{\alpha_i} & c_{\alpha_i} & 0 & L_s \alpha_i + (L_2 + L_3)s_{\alpha_i} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

DH Example [3]: Puma 560

- “Simple” 6R robot exercise for the reader …

<table>
<thead>
<tr>
<th>Link</th>
<th>(a_i)</th>
<th>(\alpha_i)</th>
<th>(d_i)</th>
<th>(\theta_i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>(\theta_1)</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>(-\pi/2)</td>
<td>0</td>
<td>(\theta_2)</td>
</tr>
<tr>
<td>3</td>
<td>(L_2)</td>
<td>0</td>
<td>(D_3)</td>
<td>(\theta_3)</td>
</tr>
<tr>
<td>4</td>
<td>(L_3)</td>
<td>(-\pi/2)</td>
<td>(D_4)</td>
<td>(\theta_4)</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>(\pi/2)</td>
<td>0</td>
<td>(\theta_5)</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>(-\pi/2)</td>
<td>0</td>
<td>(\theta_6)</td>
</tr>
</tbody>
</table>
DH Example [3]: Puma 560 [2]

\[
0_{A_1} = \begin{bmatrix}
1 & -s_1 & 0 & 0 \\
 s_1 & 1 & 0 & 0 \\
 0 & 0 & 1 & 0 \\
 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
1_{A_2} = \begin{bmatrix}
 c_2 & -s_2 & 0 & 0 \\
0 & 0 & 1 & d_2 \\
-s_2 & -c_2 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
2_{A_3} = \begin{bmatrix}
 c_3 & -s_3 & 0 & L_2 \\
 s_3 & c_3 & 0 & 0 \\
 0 & 0 & 1 & d_3 \\
 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
3_{A_4} = \begin{bmatrix}
 c_4 & -s_4 & 0 & L_3 \\
0 & 0 & 1 & d_4 \\
-s_4 & c_4 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
4_{A_5} = \begin{bmatrix}
 c_4 & -s_5 & 0 & L_3 \\
0 & 0 & 1 & d_4 \\
-s_5 & c_5 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
5_{A_6} = \begin{bmatrix}
 c_6 & -s_6 & 0 & L_3 \\
0 & 0 & -1 & 0 \\
-s_6 & c_6 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[0_{T_6} = 0_{A_1}1_{A_2}2_{A_3}3_{A_4}4_{A_5}5_{A_6}\]

Demonstration: Matlab & Solidworks
Modified DH

- Made “popular” by Craig’s *Intro. to Robotics* book
- Link coordinates attached to the near by joint

\[
\begin{align*}
&\alpha \ (\text{trans } x-1) \Rightarrow \alpha \ (\text{rot } x-1) \Rightarrow \theta \ (\text{rot } z) \Rightarrow d \ (\text{trans } z)
\end{align*}
\]

Modified DH [2]

- Gives a similar result (but it’s not commutative)

\[
\Rightarrow i^{-1} A_i = R_x (\alpha_{i-1}) T_x (a_{i-1}) R_z (\theta_i) T_x (d_i)
\]

- Refactoring Standard $\Rightarrow$ to Modified

\[
\begin{align*}
&\{ R_z (\theta_1) T_z (d_1) \} \cdot \{ R_z (\theta_2) T_z (d_2) \} \cdot \{ R_z (\theta_3) T_z (d_3) \} \\
&\text{DH}_1 \quad \text{DH}_2 \quad \text{End Effector} \\
= \{ R_z (\theta_1) T_z (d_1) \} \cdot \{ T_x (a_1) R_x (\alpha_1) R_z (\theta_2) T_z (d_2) \} \cdot \{ T_x (a_2) R_x (\alpha_2) R_z (\theta_3) T_z (d_3) \} \\
&\text{Base} \quad \text{MDH}_1 \quad \text{MDH}_2
\end{align*}
\]
Forward Kinematics [1]

- Forward kinematics is the process of chaining homogeneous transforms together. For example to:
  - Find the articulations of a mechanism, or
  - the fixed transformation between two frames which is known in terms of linear and rotary parameters.
- Calculates the final position from the machine (joint variables)

- Unique for an open kinematic chain (serial arm)
- “Complicated” (multiple solutions, etc.) for a closed kinematic chain (parallel arm)
Forward Kinematics [2]

- Can think of this as “spaces”:
  - Operation space \((x, y, z, \alpha, \beta, \gamma)\):
    The robot’s position & orientation
  - Joint space \((\theta_1 \ldots \theta_n)\):
    A state-space vector of joint variables

\[
\vec{x} = \begin{bmatrix} \vec{p} \\ \vec{\Theta} \end{bmatrix}
\]

\[
\vec{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}
\]

Forward Kinematics [3]

- Consider a planar RRR manipulator
- Given the joint angles and link lengths, we can determine the end effector pose:

\[
x = L_1 \cos \theta_1 + L_2 \cos (\theta_1 + \theta_2) + \ldots + L_3 \cos (\theta_1 + \theta_2 + \theta_3)
\]

\[
y = L_1 \sin \theta_1 + L_2 \sin (\theta_1 + \theta_2) + \ldots + L_3 \sin (\theta_1 + \theta_2 + \theta_3)
\]

- This isn’t too difficult to determine for a simple, planar manipulator. BUT …
Forward kinematics [4]:

The PUMA 560!

- What about a more complicated mechanism?

Inverse Kinematics

- Forward: angles $\rightarrow$ position
  \[ \mathbf{x} = f(\theta) \]

- Inverse: position $\rightarrow$ angles
  \[ \theta = f^{-1}(\mathbf{x}) \]

Analytic Approach

Numerical Approaches:
- Jacobian:
  \[ J = \frac{\delta \mathbf{x}}{\delta \mathbf{q}} \rightarrow \delta \mathbf{q} \approx J^{-1} \delta \mathbf{x} \]
- $J^T$ Approximation:
  \[ \tau = J^T \cdot \mathbf{F} \rightarrow \Delta \mathbf{q} \approx J^T \Delta \mathbf{x} \]
  - Slotine & Sheridan method
  - Cyclical Coordinate Descent
Inverse Kinematics

- Inverse Kinematics is the problem of finding the joint parameters given only the values of the homogeneous transforms which model the mechanism (i.e., the pose of the end effector).

- Solves the problem of where to drive the joints in order to get the hand of an arm or the foot of a leg in the right place.

- In general, this involves the solution of a set of simultaneous, non-linear equations.

- Hard for serial mechanisms, easy for parallel.

Multiple Solutions

- There will often be multiple solutions for a particular inverse kinematic analysis.

- Consider the three link manipulator shown. Given a particular end effector pose, two solutions are possible.

- The choice of solution is a function of proximity to the current pose, limits on the joint angles and possible obstructions in the workspace.
Inverse kinematics

- What about a more complicated mechanism?

Solution Methods

- Unlike with systems of linear equations, there are no general algorithms that may be employed to solve a set of nonlinear equation
- **Closed-form** and **numerical** methods exist
- We will concentrate on analytical, closed-form methods
- These can be characterized by two methods of obtaining a solution: **algebraic** and **geometric**
Inverse Kinematics: Geometrical Approach

- We can also consider the geometric relationships defined by the arm.

Inverse Kinematics: Algebraic Approach

- We have a series of equations which define this system.
- Recall, from Forward Kinematics:

\[
0T_3 = \begin{bmatrix}
c_{\theta_{123}} & -s_{\theta_{123}} & 0 & L_1c_{\theta_1} + L_2c_{\theta_{12}} + L_3c_{\theta_{123}} \\
s_{\theta_{123}} & c_{\theta_{123}} & 0 & L_1s_{\theta_1} + L_2s_{\theta_{12}} + L_3s_{\theta_{123}} \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

- The end-effector pose is given by:

\[
0T_3 = \begin{bmatrix}
c_{\phi} & -s_{\phi} & 0 & x \\
s_{\phi} & c_{\phi} & 0 & y \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

- Equating terms gives us a set of algebraic relationships.
No Solution - Singularity

- Singular positions:

- An understanding of the workspace of the manipulator is important
- There will be poses that are not achievable
- There will be poses where there is a loss of control

- Singularities also occur when the manipulator loses a DOF
  - This typically happens when joints are aligned
  - $\text{det}[\text{Jacobian}]=0$
### Mobile Platforms

- The preceding kinematic relationships are also important in mobile applications.

- When we have sensors mounted on a platform, we need the ability to translate from the sensor frame into some world frame in which the vehicle is operating.

- Should we just treat this as a P(*) mechanism?

### Mobile Platforms [2]

- We typically assign a frame to the base of the vehicle.
- Additional frames are assigned to the sensors.
- We will develop these techniques in coming lectures.
Summary

• Many ways to view a rotation
  – Rotation matrix
  – Euler angles
  – Quaternions
  – Direction Cosines
  – Screw Vectors

• Homogenous transformations
  – Based on homogeneous coordinates

Cool Robotics Share

Light Field Video Stabilization

ICCV 2009
Brandon M. Smith¹, Li Zhang¹, Hailin Jin², Aseem Agarwala²

¹ UW-Madison Graphics and Vision Group
² Adobe Systems Incorporated

supplemental video with narration