Jacobian: Velocity propagation 4/4
Jacobian Matrix - Derivation Methods

Explicit Method
- Differentiation the Forward Kinematics Eqs. (Method 1)

Iterative Methods
Recursive Equations
- Velocity Propagation – Base to EE (Method 2)
- Force/Torque Propagation – EE to Base (Method 3)

Jacobian Matrix
Jacobian Iterative Method - Velocity propagation (Method No.2)
Explicit Method

Differentiation the Forward Kinematics Eqs. (Method 1)

Iterative Methods
Recursive Equations

Velocity Propagation – Base to EE (Method 2)

Force/Torque Propagation – EE to Base (Method 3)

Jacobian Matrix
Jacobian: Velocity propagation

- The recursive expressions for the adjacent joint linear and angular velocities describe a relationship between the joint angle rates ($\dot{\theta}$) and the transnational and rotational velocities of the end effector ($\dot{X}$):

$$i+1 \omega_{i+1} = i+1 R^i \omega_i + \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_{i+1} \end{bmatrix}$$

$$i+1 v_{i+1} = i+1 R(i \omega \times i P_{i+1} + i v_i) + \begin{bmatrix} 0 \\ 0 \\ \dot{d}_{i+1} \end{bmatrix}$$
Jacobian: Velocity propagation

- Therefore the recursive expressions for the adjacent joint linear and angular velocities can be used to determine the Jacobian in the end effector frame.

\[ \overset{N}{\dot{X}} = \overset{N}{J}(\theta)\dot{\theta} \]

- This equation can be expanded to:

\[ \overset{N}{\dot{X}} = \begin{bmatrix} \overset{N}{\dot{x}} \\ \overset{N}{\dot{y}} \\ \overset{N}{\dot{z}} \\ \overset{N}{\Omega}_x \\ \overset{N}{\Omega}_y \\ \overset{N}{\Omega}_z \end{bmatrix} = \begin{bmatrix} \overset{N}{v}_N \\ \overset{N}{\omega}_N \end{bmatrix} = \overset{N}{J}(\theta) \begin{bmatrix} \overset{N}{\dot{\theta}_1} \\ \overset{N}{\dot{\theta}_2} \\ \ldots \\ \overset{N}{\dot{\theta}_N} \end{bmatrix} \]
• The linear angular velocities of the end effector (N=4)

\[
\begin{bmatrix}
L2s3\dot{\theta}_2 \\
(L2c3 + L3)\dot{\theta}_2 + L3\dot{\theta}_3 \\
(-L1 - L2c2 - L3c23)\dot{\theta}_1
\end{bmatrix}
\]

\[
\begin{bmatrix}
s23\dot{\theta}_1 \\
c23\dot{\theta}_1 \\
\dot{\theta}_2 + \dot{\theta}_3
\end{bmatrix}
\]
Jacobian - 3R - Example

- Re-arranged to previous two terms gives an expression that encapsulates

\[ 4 \dot{X} = 4 J(\theta) \dot{\theta} \]

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z} \\
\dot{\Omega}_x \\
\dot{\Omega}_y \\
\dot{\Omega}_z \\
\end{bmatrix}
= \begin{bmatrix} 4 v_4 \\ 4 \omega_4 \end{bmatrix}
= \begin{bmatrix}
L2s3 \dot{\theta}_2 \\
(L2c3 + L3) \dot{\theta}_2 + L3 \dot{\theta}_3 \\
(-L1 - L2c2 - L3c23) \dot{\theta}_1 \\
s23 \dot{\theta}_1 \\
c23 \dot{\theta}_1 \\
\dot{\theta}_2 + \dot{\theta}_3 \\
\end{bmatrix}
\]

- We can now factor out the joint velocities vector \( \dot{\theta} = [\dot{\theta}_1 \dot{\theta}_2 \dot{\theta}_3]^T \) from the above vector to formulate the Jacobian matrix \( 4 J(\theta) \)
The equations for $\mathbf{v}_N^N$ and $\mathbf{\omega}_N^N$ are always a linear combination of the joint velocities, so they can always be used to find the $6 \times N$ Jacobian matrix ($^N J(\theta)$) for any robot manipulator.

Note that the Jacobian matrix is expressed in frame $\{4\}$.
Using the velocity propagation method we expressed the relationship between the velocity of the robot end effector measured relative to the robot base frame \{0\} and expressed in the end effector frame \{N\}.

\[
N \dot{X} = N J(\theta) \dot{\theta}
\]

Occasionally, it may be desirable to express (represent) the end effector velocities in another frame (e.g. frame \{0\}, in which case we will need a method to provide the transformation.

\[
0 \dot{X} = 0 J(\theta) \dot{\theta}
\]
There are two methods to change the references frame (frame of representation) of the Jacobian Matrix

- **Method 1**: Transforming the linear and angular velocities to the new frame prior to formulating the Jacobian matrix.

- **Method 2**: Transforming the Jacobian matrix from its existing frame to the new frame after it was formulated.
Jacobian: Frame of Representation – Method 1

- Consider the velocities in a different frame \{B\}

\[ \dot{B} \dot{X} = \begin{bmatrix} B v_N \\ B \omega_N \end{bmatrix} = B J(\theta) \dot{\theta} \]

- We may use the rotation matrix to find the velocities in frame \{A\}:

\[ \dot{A} \dot{X} = \begin{bmatrix} A v_N \\ A \omega_N \end{bmatrix} = A \dot{R}^B \begin{bmatrix} B v_N \\ B \omega_N \end{bmatrix} \]
Example: Analyzing a 6 DOF manipulator while utilizing velocity propagation method results in expressing the end effector (frame 6) linear and angular velocities.

\[ \dot{\mathbf{X}}^6 = \begin{bmatrix} \mathbf{v}_6^6 \\ \mathbf{\omega}_6^6 \end{bmatrix} \]

Using the forward kinematics formulation, the rotation matrix from frame 0 to frame 6 can be defined as

\[ \mathbf{T}^{0}_{6} = \mathbf{T}_1^0 \mathbf{T}_2^1 \mathbf{T}_3^2 \mathbf{T}_4^3 \mathbf{T}_5^4 \mathbf{T}_6^5 = \begin{bmatrix} \mathbf{R}_6^0 & \mathbf{0}_{6 \times 6} \\ \mathbf{0}_{6 \times 6} & \mathbf{1} \end{bmatrix} \]

The linear and angular velocities can then be expressed in frame 0 prior to extracting the Jacobian in frame 0

\[ \dot{\mathbf{X}}^0 = \begin{bmatrix} \dot{\mathbf{v}}_6^0 \\ \dot{\mathbf{\omega}}_6^0 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_6^0 \dot{\mathbf{v}}_6^0 \\ \mathbf{R}_6^0 \dot{\mathbf{\omega}}_6^0 \end{bmatrix} = \mathbf{J} \dot{\mathbf{\theta}} \]
It is possible to define a Jacobian transformation matrix $^A_B R_J$ that can transform the Jacobian from frame A to frame B

$$^A X = ^A J (\theta) \dot{\theta} = ^A B R_J ^B J (\theta) \dot{\theta}$$

The Jacobian rotation matrix $^A_B R_J$ is given by

$$^A_B R_J = \begin{bmatrix}
0 & 0 & 0 & [^A_B R] \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}$$
or equivalently,

\[ A J(\theta) = \begin{bmatrix} [A_R] & [0 0 0] \\ 0 0 0 & [B_R] \end{bmatrix} B J(\theta) \]
Jacobian: Frame of Representation - 3R Example

- The rotation matrix \( ^0_4 R \) can be calculated based on the direct kinematics given by

\[
0 \theta = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
\end{bmatrix}
\]

\[
J(\theta) = \begin{bmatrix}
clc23 & -c1s23 & s1 \\
slc23 & -s1s23 & -c1 \\
s23 & c23 & 0 \\
\end{bmatrix}
\]

\[
^0_4 R_J = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]

\[
^0_4 T = ^0_1 T^1_2 T^2_3 T^3_4 T = \begin{bmatrix}
^0_4 R & ^0_4 P_{ORG} \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\]
Inverse Jacobian

- **Given**
  - Tool tip path (defined mathematically)
  - Tool tip position/orientation
  - Tool tip velocity
  - Jacobian Matrix

\[ \dot{x} = J(\theta) \dot{\theta} \]

- **Problem:** Calculate the joint velocities

- **Solution:**
  - Compute the inverse Jacobian matrix
  - Use the following equation to compute the joint velocity

\[ \dot{\theta} = J(\theta)^{-1} \dot{x} \]
Inverse Jacobian

- Cases in which the Jacobian matrix $J(\theta)$ is not inevitable ($J(\theta)^{-1}$ does not exist). Non invertible matrix is called singular matrix
  
  - **Case 1** - The Jacobian matrix is not squared
    In general the 6xN Jacobian matrix may be non-square in which case the inverse is not defined
  
  - **Case 2** - The determinant ($\det(J(\theta))$) is equal to zero
Inverse Jacobian - Reduced Jacobian

- **Problem**
  - When the number of joints (N) is less than 6, the manipulator does not have the necessary degrees of freedom to achieve independent control of all six velocities components.

\[
\dot{X} = \begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z} \\
\Omega x \\
\Omega y \\
\Omega z
\end{bmatrix}
\]

- **Solution**
  - We can reduce the number of rows in the original Jacobian to describe a reduced Cartesian vector. For example, the full Cartesian velocity vector is given by
Jacobian: Reduced Jacobian - 3R Example

- Matrix Reduction - Option 1

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z} \\
\Omega_x \\
\Omega_y \\
\Omega_z
\end{bmatrix}
= \begin{bmatrix}
0 & L2s3 & 0 \\
0 & L2c3 + L3 & L3 \\
- \frac{L1}{2} - L2c2 - L3c23 & 0 & 0 \\
s23 & 0 & 0 \\
c23 & 0 & 0 \\
0 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2 \\
\dot{\theta}_3
\end{bmatrix}
\]

- Column of zeroes
- The determinant is equal to zero
- Only two out of the three variables can be independently specified

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Matrix Reduction - Option 2

- Two columns of zeroes
- The determinate is equal to zero
- Only one out of the three variables can be independently specified
• Matrix Reduction - Option 3

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z} \\
\dot{\Omega}_x \\
\dot{\Omega}_y \\
\dot{\Omega}_z
\end{bmatrix}
= \begin{bmatrix}
0 & L2s3 & 0 \\
0 & L2c3 + L3 & L3 \\
-L1 - L2c2 - L3c23 & 0 & 0 \\
s23 & 0 & 0 \\
c23 & 0 & 0 \\
0 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{\theta}_1 \\
\dot{\theta}_2 \\
\dot{\theta}_3
\end{bmatrix}
\]

• The resulting reduced Jacobian will be square (the number of independent rows in the Jacobian are equal to the number of unknown variables) and can be inverted unless in a singular configuration.
To avoid singular configurations, the determinant of the Jacobian is often computed symbolically to find the set of joint values for which singularities will occur. Singularities often occur under two situations:

1. Workspace Boundary: the manipulator is fully extended or folded back upon itself.

2. Workspace Interior: generally caused by two or more axes intersecting.
Jacobian: Singular Configuration - 3R Example

- If we want to use the inverse Jacobian to compute the joint angular velocities we need to first find out at what points the inverse exists.

\[ \dot{\theta} = J(\theta)^{-1} \dot{x} \]

- Considering the 3R robot

\[
^4J_r(\theta) = \begin{bmatrix}
0 & L2s3 & 0 \\
0 & L2c3 + L3 & L3 \\
- L1 - L2c2 - L3c23 & 0 & 0
\end{bmatrix}
\]

- The determinate of the Jacobian is defined as follows

\[
\left| ^4J_r(\theta) \right| = -(L1 + L2c2 + L3c23)(L2s3)L3
\]
The reduced Jacbian matrix is singular when it determinate is equal to zero

\[ \left| ^4 J_r(\theta) \right| = -(L_1 + L_2c_2 + L_3c_{23})(L_2s_3)L_3 \]

- The singular condition occur when either of the following are true

\[ -(L_1 + L_2c_2 + L_3c_{23})(L_2s_3)L_3 = 0 \]

- The singular condition occur when either of the following are true

\[ \begin{cases} s_3 = 0 \\ L_1 - L_2c_2 - L_3c_{23} = 0 \end{cases} \]
Jacobian: Singular Configuration - 3R Example

- Case 1: \( s_3 = 0 \)

\[
s_3 = 0 \Rightarrow \begin{cases} 
\theta_3 = 0^\circ \\
\theta_3 = 180^\circ 
\end{cases}
\]

\[
^4J_r(\theta) = \begin{bmatrix}
0 & L2s3 & 0 \\
0 & L2c3 + L3 & L3 \\
- L1 - L2c2 - L3c23 & 0 & 0 
\end{bmatrix}
\]

- The first row of the Jacobian is zero
- The 3R robot is losing one DOF.
- The robot can no longer move along the X-axis of frame \{4\}
Jacobian: Singular Configuration - 3R Example

Case 2: \(-L1 - L2c2 - L3c23 = 0\)
\[
L1 = -L2c2 - L3c23
\]

Occur only if \(L2 + L3 \geq L1\)

\[
^4J_r(\theta) = \begin{bmatrix}
0 & L2s3 & 0 \\
0 & L2c3 + L3 & L3 \\
-L1 - L2c2 - L3c23 & 0 & 0
\end{bmatrix}
\]

- The third row of the Jacobian is zero
- The origin of frame \{4\} intersects the Z-axis of frame \{1\}
- The 3R robot is losing one DOF.
- The robot can no longer move along the Z-axis of frame \{4\}
Joint Velocity Near Singular Position - 3R Example

- Robot: 3R robot
- Task: Visual inspection

\[ \dot{\theta} = J(\theta)^{-1}\dot{x} \]

Control

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Joint Velocity Near Singular Position - 3R Example

- **Singularity (Case 2)** - The origin of frame \( \{4\} \) intersects the Z-axis of frame \( \{1\} \)

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\dot{z}
\end{bmatrix} =
\begin{bmatrix}
0 & L2s3 & 0 \\
0 & L2c3 + L3 & L3 \\
-L1 - L2c2 - L3c23 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\dot{\theta_1} \\
\dot{\theta_2} \\
\dot{\theta_3}
\end{bmatrix}
\]

- Solve for \( \dot{\theta_1} \) in terms of \( \dot{z} \) we find

\[
\dot{\theta_1} = \frac{\dot{z}}{-L1 - L2c2 - L3c23}
\]

\[
-L1 - L2c2 - L3c23 = 0
\]

\( \dot{\theta_1} \to \infty \)
Joint Velocity Near Singular Position - 3R Example

- Singularity - \[ \det(J(\theta)) = 0 \]

- Problems:
  - **Motor Constrains** - The robot is physically limited from moving in unusual high joint velocities by motor power constrains. Therefore, the robot will be unable to track the required joint velocity trajectory exactly resulting in some perturbation to the commanded Cartesian velocity trajectory.
  - **Gears and Shafts** - The derivative of the angular velocity is the angular acceleration. The high acceleration of the joint resulting form approaching too close to a singularity may cause damage to the gear/shafts.
  - **DOF** - At a singular configuration (specific point in space) the manipulator loses one or more DOF.

- Consequences – Certain tasks can not be performed at a singular configuration
Designing Well Conditioned Workspace

- Difficulty in operating at
  - Workspace Boundaries
  - Neighborhood of singular point inside the workspace

- The further the manipulator is away from singularities the better it moves uniformly and apply forces in all directions

- **Manipulability Measure** - How far / close the manipulator is from singularity
  - Range \( 0 \leq w < \infty \)
  - For Non redundant manipulators \( w = \det(J(\theta)) \)
  - For redundant manipulators \( w = \sqrt{\det(J(\theta)J^T(\theta))} \)
  - A good manipulator design has large area of characterized by high value of the manipulability \( (w) \)
Jacobian Explicit Method -
Differentiation the Forward Kinematics Eqs. (Method No. 1)
Jacobian Matrix - Derivation Methods

Explicit Method

Differentiation the Forward Kinematics Eqs. (Method 1)

Iterative Methods
Recursive Equations

Velocity Propagation – Base to EE (Method 2)

Force/Torque Propagation – EE to Base (Method 3)

Jacobian Matrix
Jacobian – Explicit Form – Overview

\[
\begin{bmatrix}
\dot{v}_n^x \\
\dot{v}_n^y \\
\dot{v}_n^z \\
\dot{w}_n^x \\
\dot{w}_n^y \\
\dot{w}_n^z
\end{bmatrix}
\]

\[= J_v \dot{\Theta}
\]

\[
\begin{bmatrix}
\dot{\Theta}_1 \\
\dot{\Theta}_2 \\
\dot{\Theta}_3 \\
\dot{\Theta}_4 \\
\dot{\Theta}_5 \\
\dot{\Theta}_6
\end{bmatrix}
\]

\[
F \text{ frame } 0
\]

\[
J = \begin{bmatrix}
J_v & J_w
\end{bmatrix}
\]

\[6 \times 6\]
Jacobian – Explicit Form

\[
\begin{bmatrix}
J_v & J_w
\end{bmatrix}
\]

Prismatic Joint

Revolute Joint

\[
\begin{bmatrix}
\frac{\partial \mathbf{p}_6}{\partial \mathbf{a}_i} \\
\mathbf{Z}
\end{bmatrix}
\]

\[
\mathbf{Z} \times (\mathbf{p}_6 - \mathbf{p}_5)
\]

\[
\begin{bmatrix}
\mathbf{R} \\
\mathbf{0}
\end{bmatrix}
\]

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Jacobian – Explicit Form – Angular Velocity \( J_\omega \)

\[
i_{i-1}W_{i+1} = i_iR i_iW_i + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}i_i + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}i_i = 0 \text{ prismatic}\]

\[
i = 0 \quad 1W_1 = 1R 0W_0 + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}1 + 1W_1 \]

\[
i = 1 \quad 2W_2 = 2R 1W_1 + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}2 = 2R \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}2 + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}2 \]

\[
i = 2 \quad 3W_3 = 3R 2W_2 + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}3 = 3R \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}3 + \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}3 \]

\[
\begin{bmatrix} 0 \\ 1R \\ 2R \\ \cdots \end{bmatrix}W_n = 1R \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}1 + 2R \begin{bmatrix} 0 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_4 \end{bmatrix}\frac{d}{dt}2 + \cdots \]

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Jacobian – Explicit Form – Angular Velocity \( J_\omega \)

\[
\dot{\omega}_n = \dot{\omega}_R^1 R^1 R \begin{bmatrix} \dot{\theta}_1 \\ 0 \end{bmatrix} f_1 + \dot{\omega}_R^2 R^2 R \begin{bmatrix} \dot{\theta}_2 \\ 0 \end{bmatrix} f_2 + \cdots + \dot{\omega}_R^n R^n R \begin{bmatrix} 0 \\ \dot{\theta}_n \end{bmatrix} f_n
\]

\[
\dot{\omega}_n = \dot{\omega}_R^1 \begin{bmatrix} \dot{\theta}_1 \\ 0 \end{bmatrix} f_1 + \dot{\omega}_R^2 \begin{bmatrix} \dot{\theta}_2 \\ 0 \end{bmatrix} f_2 + \cdots + \dot{\omega}_R^n \begin{bmatrix} 0 \\ \dot{\theta}_n \end{bmatrix} f_n
\]

\[
\dot{\omega}_n = \sum_{i=1}^{n} \dot{\omega}_R^i \begin{bmatrix} 0 \\ \dot{\theta}_i \end{bmatrix} f_i = \sum_{i=1}^{n} \dot{\omega}_R^i \begin{bmatrix} 0 \\ \dot{\theta}_i \end{bmatrix} f_i
\]

\[
\dot{\omega}_R = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}
\]
Jacobian – Explicit Form – Angular Velocity $J_\omega$
Jacobian – Explicit Form – Linear Velocity

\[
\dot{P}_n = \sum \frac{\partial \dot{P}_n}{\partial q_i} \cdot \dot{q}_i
\]

\[
J_{vi} = \frac{\partial \dot{P}_n}{\partial q_i}
\]
Jacobian – Explicit Form – Linear Velocity
Case 1- Prismatic Joint

\[ \dot{P}_n = \dot{d}_i \cdot R \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \cdot \dot{d}_i \cdot \hat{z} \]

\[ J_{vi} = \hat{z} \]

\[ \dot{R} \]

\[ \dot{\hat{z}} \]

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Jacobian – Explicit Form – Linear Velocity
Case 2 – Revolute Joint

\[ J_v = z_i \times (p_n - p_i) \]

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Jacobian Iterative Method -
Force/Torque Propagation (Method No. 3)
Jacobian Matrix - Derivation Methods

Explicit Method

Differentiation the Forward Kinematics Eqs. (Method 1)

Iterative Methods
Recursive Equations

Velocity Propagation – Base to EE (Method 2)

Force/Torque Propagation – EE to Base (Method 3)

Jacobian Matrix
Statics - Forces & Torques

Problem

Given: Typically the robot’s end effector is applying forces and torques on an object in the environment or carrying an object (gravitational load).

Compute: The joint torques which must be acting to keep the system in static equilibrium.

Solution

Jacobian - Mapping from the joint force/torques - $\tau$ to forces/torque in the Cartesian space applied on the end effector) - $f$.

$$\tau = J^T f$$

Free Body Diagram - The chain like nature of a manipulator leads to decompose the chain into individual links and calculate how forces and moments propagate from one link to the next.
Step 1
Lock all the joints - Converting the manipulator (mechanism) to a structure

Step 2
Consider each link in the structure as a free body and write the force / moment equilibrium equations

(3 Eqs.)\[ \sum F = 0 \]
(3 Eqs.)\[ \sum M = 0 \]

Step 3
Solve the equations - 6 Eq. for each link.
Apply backward solution starting from the last link (end effector) and end up at the first link (base)
Static Analysis Protocol - Free Body Diagram 2/

- Special Symbols are defined for the force and torque exerted by the neighbor link

\[ f_i \] - Force exerted on link \( i \) by link \( i-1 \)

\[ n_i \] - Torque exerted on link \( i \) by link \( i-1 \)

Reference coordinate system \( \{B\} \)

Exerted on link A by link A-1

- For easy solution superscript index (A) should the same as the subscript (B)
Static Analysis Protocol - Free Body Diagram 3/

- For serial manipulator in static equilibrium (joints locked), the sum the forces and torques acting on link \( i \) in the link frame \( \{i\} \) are equal to zero.

\[
\sum i F = 0 \quad \Rightarrow \quad \sum i F = f_i - f_{i+1} = 0
\]

\[
\sum i M = 0 \quad \Rightarrow \quad \sum i M = n_i - n_{i+1} - P_{i+1} \times f_{i+1} = 0
\]
**Procedural Note**: The solution starts at the end effector and ends at the base.

Re-writing these equations in order such that the known forces (or torques) are on the right-hand side and the unknown forces (or torques) are on the left, we find:

\[
\begin{align*}
^i f_i &= ^{i+1} f_i \\
^i n_i &= ^i n_{i+1} + ^{i+1} P_i \times ^i f_{i+1} = ^i n_{i+1} + ^{i+1} P_i \times ^i f_i
\end{align*}
\]
• Changing the reference frame such that each force (and torque) is expressed upon their link’s frame, we find the static force (and torque) propagation from link \( i+1 \) to link \( i \)

\[
if_i = if_{i+1} = i+1R \ i+1f_{i+1}
\]

\[
in_i = in_{i+1} + iP_{i+1} \times if_{i+1} = i+1n_{i+1} + iP_{i+1} \times if_i
\]

• These equations provide the static force (and torque) propagation from link to link. They allow us to start with the force and torque applied at the end effector, and calculate the force and torque at each joint all the way back to the robot base frame.
• **Question:** What torques are needed at the joints in order to balance the reaction moments acting on the link (Revolute Joint).
• **Question:** What forces are needed at the joints in order to balance the reaction forces acting on the link (**Prismatic Joint**).
• **Answer:** All the components of the force and moment vectors are resisted by the structure of mechanism itself, except for the torque about the joint axis (revolute joint) or the force along the joint (prismatic joint).

• Therefore, to find the joint the torque or force required to maintain the static equilibrium, the dot product of the joint axis vector with the moment vector or force vector acting on the link is computed

\[
\tau_i = n_i^T \hat{z}_i = \begin{bmatrix} i_{n_x} & i_{n_y} & i_{n_z} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\]

**Revolute Joint**

\[
f_i = f_i^T \hat{z}_i = \begin{bmatrix} i_{f_x} & i_{f_y} & i_{f_z} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\]

**Prismatic Joint**
Example - 2R Robot - Static Analysis

Problem

Given:
- 2R Robot
- A Force vector $^3f_3$ is applied by the end effector
- A torque vector $^3n_3 = 0$

Compute:

The required joint torque as a function of the robot configuration and the applied force
Example - 2R Robot - Static Analysis
Example - 2R Robot - Static Analysis

Solution

• Lock the revolute joints

• Apply the static equilibrium equations starting from the end effector and going toward the base

\[ i \ f_i = i+1 R \ i+1 \ f_{i+1} \]

\[ i \ n_i = i+1 R \ i+1 n_{i+1} + i P_{i+1} \times i f_i \]
Example - 2R Robot - Static Analysis

- For $i=2$

$$2^2 f_2 = 2^3 R \ 3^3 f_3$$

$$2^2 f_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix}$$

$$2^2 n_2 = 2^3 R \ 3^3 n_3 + 2^2 P_3 \times 2^2 f_2$$

$$2^2 n_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \times \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix} = \begin{bmatrix} 0 \\ l_2 \\ 0 \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix} \begin{bmatrix} l_2 f_y \end{bmatrix}$$
Example - 2R Robot - Static Analysis

• For i=1

\[ 1f_1 = 2R \begin{bmatrix} c_2 & -s_2 & 0 \\ s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x \\ f_y \\ 0 \end{bmatrix} = \begin{bmatrix} c_2f_x - s_2f_y \\ s_2f_x - c_2f_y \\ 0 \end{bmatrix} \]

\[ 1n_1 = 2R 2n_2 + 1P_2 \times 1f_1 \]

\[ 1n_1 = \begin{bmatrix} c_2 & -s_2 & 0 \\ s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ l_2f_y \end{bmatrix} + 0 \times \begin{bmatrix} 0 \\ 0 \\ l_2f_y \end{bmatrix} = \begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ l_2f_y \end{bmatrix} = \begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix} + \begin{bmatrix} c_2f_x - s_2f_y \\ s_2f_x + c_2f_y \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ l_2f_y \end{bmatrix} + \begin{bmatrix} c_2f_x - s_2f_y \\ s_2f_x + c_2f_y \\ 0 \end{bmatrix} \]
Example - 2R Robot - Static Analysis

\[
\begin{align*}
^1n_1 &= \begin{bmatrix} 0 \\ 0 \\ l_1 s_2 f_x + l_1 c_2 f_y + l_2 f_y \end{bmatrix} \\
^2n_2 &= \begin{bmatrix} 0 \\ 0 \\ l_2 f_y \end{bmatrix}
\end{align*}
\]

\[
\tau_i = \begin{bmatrix} i_n^T \hat{r}_i \\ 1 \end{bmatrix} = \begin{bmatrix} i_n_{ix} \\ i_n_{iy} \\ i_n_{iz} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\]

\[
\tau_1 = l_1 s_2 f_x + l_1 c_2 f_y + l_2 f_y \\
\tau_2 = l_2 f_y
\]
Example - 2R Robot - Static Analysis

- Re-writing the equations in a matrix form

\[
\tau = \begin{bmatrix}
  l_1 s_2 & l_1 c_2 + l_2 \\
  0 & 0
\end{bmatrix}
\begin{bmatrix}
  f_x \\
  f_y
\end{bmatrix}
= [N J]^T f
\]
Jacobian Propagation to the Tip of the Tool
Jacobian Propagation to the Tip of the Tool

Goal

\[
T_{\text{path}} = \begin{pmatrix}
0 & T^1 & T^2 & T^3 & T^4 & T^5 & T^6 \\
1 & & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
\end{pmatrix}
\]

The position of the tool does not change as a function of time in frame 6.

Multiply both sides by \((T^6)^{-1} = (T^6)^T = T^6^T\)

\[
T_{\text{path}}(T^6)^{-1} = \begin{pmatrix}
0 & T^1 & T^2 & T^3 & T^4 & T^5 & T^6 \\
1 & & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
\end{pmatrix}
\]

\[
T_{\text{path}}^T = \begin{pmatrix}
0 & T^1 & T^2 & T^3 & T^4 & T^5 & T^6 \\
1 & & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
& & & & & & \\
\end{pmatrix}
\]

→ solve the inverse kin (IK)
Jacobian Propagation to the Tip of the Tool

FORCE / TORQUE

VELOCITIES (LINEAR & ANGULAR)

WRIST

F/T SENSOR

GRIPER

TOOL

WRIST

GRASPER

UCLA
Jacobian Propagation to the Tip of the Tool

**Position**

**Vector Form**

\[ AP_Q = AP_{BQP} + ABP_Q \]

**Matrix Form**

\[ AP_Q = \begin{bmatrix} ABP_Q \end{bmatrix} \]

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Jacobian Propagation to the Tip of the Tool

**VELOCITY - FOR TWO FRAMES THAT ARE RIGIDLY CONNECTED**

**VECTOR FORM**

\[
\dot{\mathbf{w}}_{i+1} = \dot{\mathbf{r}}_{i} \mathbf{w}_{i} + \dot{\mathbf{z}}_{i+1}
\]

\[
\dot{\mathbf{u}}_{i+1} = \dot{\mathbf{r}}_{i} \left( \mathbf{v}_{i} + \mathbf{w}_{i} \times \mathbf{p}_{i+1} \right)
\]

\[
\begin{cases}
\dot{\mathbf{w}}_{i+1} \\
\dot{\mathbf{u}}_{i+1}
\end{cases} \Rightarrow \begin{cases}
\mathbf{A} \mathbf{w}_{A} \\
\mathbf{A} \mathbf{w}_{A}
\end{cases}
\]

\[
\begin{cases}
\mathbf{v}_{B} \\
\mathbf{w}_{B}
\end{cases} = \begin{bmatrix}
\mathbf{A} \mathbf{r} \\
\mathbf{A} \mathbf{r}
\end{bmatrix}
\]

**MATRIX FORM**

\[
\begin{bmatrix}
\mathbf{B} \\
\mathbf{A}
\end{bmatrix} \begin{bmatrix}
\mathbf{A} \mathbf{V}_{A} \\
\mathbf{A} \mathbf{V}_{A}
\end{bmatrix}
\]

Instructor: Jacob Rosen
Advanced Robotic - MAE 263B - Department of Mechanical & Aerospace Engineering - UCLA
Jacobian Propagation to the Tip of the Tool

\[
\begin{align*}
\text{FORCE/TORQUE} \\
\mathbf{f}_i &= \mathbf{R} \mathbf{f}_{i+1} \\
\mathbf{R} &= \mathbf{N}_{i+1} + \mathbf{P}_{i+1} \times \mathbf{f}_{i+1} \\
\mathbf{f} &\rightarrow \mathbf{A} \\
\mathbf{f}_{i+1} &\rightarrow \mathbf{B}
\end{align*}
\]
Jacobian Propagation to the Tip of the Tool

- The end effector holds a tool
- Located at the point where the end effector attached to the manipulator is a force sensing wrist. This device can measure the forces and torque applied to it.
Jacobian Propagation to the Tip of the Tool

\[
\begin{align*}
T &= sT + \frac{wT}{sT} \\
\text{Multiply both sides by } (sT)^{-1} &\rightarrow \frac{wT}{sT} \\
\frac{wT}{sT} &= \frac{sT}{wT} \frac{wT}{sT} \\
\text{Inverting } \frac{1}{sT} &\rightarrow \frac{1}{sT} \\
\text{The forces & torques applied on the tool tip based on the measurement at the sensor can be calculated as}
\end{align*}
\]
Jacobian Propagation to the Tip of the Tool

\[
\begin{align*}
T_f &= T_{\text{tip}} \cdot T_{\text{f}} \\
T_{sf} &= \begin{bmatrix}
T_{SR} \\
T_{P_{\text{tool}}} \times T_{\text{SR}} \\
T_{SR} \\
\end{bmatrix}
\end{align*}
\]
Jacobian Methods – Reference Frame - Summary
## Jacobian Methods of Derivation & the Corresponding Reference Frame – Summary

<table>
<thead>
<tr>
<th>Method</th>
<th>Jacobian Matrix Reference Frame</th>
<th>Transformation to Base Frame (Frame 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit (Diff. the Forward Kinematic Eq.)</td>
<td>$^0 J^N_N$</td>
<td>None</td>
</tr>
<tr>
<td>Iterative Velocity Eq.</td>
<td>$^N J^N_N$</td>
<td>Transform Method 1: $^0 v_N = ^N R^N v_N$ [0 \omega_N = ^N R^N \omega_N]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform Method 2: $^0 J_N(\theta) = \begin{bmatrix} \begin{bmatrix}^N R &amp; 0 \ 0 &amp; ^N R \end{bmatrix}^T \end{bmatrix}^N J_N(\theta)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transform $^0 J_N(\theta) = \begin{bmatrix} \begin{bmatrix}^N R &amp; 0 \ 0 &amp; ^N R \end{bmatrix}^T \end{bmatrix}^N J_N(\theta)$</td>
</tr>
</tbody>
</table>
Jacobian – High Level Overview
The Jacobian – Duality

\[ \dot{X} = J \dot{\theta} \]

is a linear mapping of the joint space velocities \( \dot{\theta} \) which is a \( n \)-dimensional vector space \( \dot{\theta} \in \mathbb{R}^n \) to the end effector velocities \( \dot{X} \) which is a \( m \)-dimensional vector space \( \dot{X} \in \mathbb{R}^m \).
The subset of all the end effector velocities \( \dot{X} \) resulting from the mapping \( \dot{X} = J \dot{\theta} \) represents all the possible end effector velocities that can be generated by the \( n \) joints given the arm configuration.
If the rank of the Jacobian matrix $J$ is at full of row rank (square matrix) the joint space $\dot{\theta}$ covers the entire end effector vector $\dot{X}$, otherwise there is at least one direction in which the end effector can not be moved $\dot{X} = J\dot{\theta} = 0$.
The subset $N(J)$ is the null space of the linear mapping. Any element in this subspace is mapped into a zero vector in $\mathbb{R}^m$ such that $\dot{\mathbf{x}} = J\dot{\mathbf{\theta}} = 0$. Therefore, any joint velocity vector $\dot{\mathbf{\theta}}$ that belongs to the null space does not produce any velocity at the end effector.
If the Jacobian of a manipulator is full rank the dimension of the null space \( \dim(N(J)) \) is the same as the redundant degrees of freedom \( n-m \). For example the human arm has 7 DOF whereas the hand may have 6 linear and angular velocities therefore the null dimension is one \( n-m=7-1=1 \).
If the Jacobian of a manipulator is full rank (i.e. $n>m$ full row rank where the rows are linearly independent) the dimension of the null space $\dim(N(J))$ is the same as the redundant degrees of freedom ($n-m$). For example the human arm has 7 DOF whereas the hand may have 6 linear and angular velocities therefore the null dimension is one ($n-m=7-1=1$).
If the Jacobian of a manipulator is full rank (i.e. for redundant manipulator n>m full row rank where the rows are linearly independent) the dimension of the null space \( \dim(N(J)) \) is the same as the redundant degrees of freedom (n-m). For example the human arm has 7 DOF whereas the end effector (hand) may have 6 linear and angular velocities therefore the null dimension is one (n-m=7-1=1)
When the Jacobian matrix degenerates (i.e. not full rank e.g. due to singularity) the dimension of the range space $\dim(R(J))$ decreases at the same time as the dimension of the null space increases $\dim(N(J))$ by the same amount. The sum of the two is always equal to $n$:

$$\dim(R(J)) + \dim(N(J)) = n$$
If the null space is not empty set, the instantaneous kinematic equation has an infinite number of solutions that cause the same end effector velocities (recall the 3 axis end effector)
**Jacobian – Duality**

Unlike the mapping of the instantaneous kinematics, the mapping of the static external forces is from the \( m \)-th vector space \( f_{\text{ext}} \in \mathbb{R}^m \) associated with the end effector coordinates to the \( n \)-th dimensional vector space \( \tau \in \mathbb{R}^n \) associated with the torques at the joint space. Therefore, the joint torque are always determined uniquely from any end effector point force.
The null space $N(J^T)$ represents the set of all end point forces that do not require any torques at the joints to bear the corresponding load (e.g. 2R fully stretched or collapsed elbow).
Jacobian – Duality

When the Jacobian matrix is degenerated or the arm is in a singular configuration external endpoint force is borne entirely by the structure and not by the joint torque.