Kinematics (Continued)

Hsiu-Chin Lin

September 10, 2020

Lecture Recordings This lecture will be recorded. By attending the live sessions, you agree to the recording, and you understand that your image, voice, and name may be disclosed to classmates

During the lecture
I will mute the audience
Ask questions through ’Chat’

Last time

Representations of Rigid Bodies
Configuration and task
Homogeneous Transformation Matrix

Last time

Representations of Rigid Bodies
Configuration and task
Homogeneous Transformation Matrix

Forward Kinematics
Given configuration position/velocity, find task position/velocity

Last time

Representations of Rigid Bodies
Configuration and task
Homogeneous Transformation Matrix

Forward Kinematics
Given configuration position/velocity, find task position/velocity

Inverse Kinematics
Given task velocity
Use inverse kinematics to find configuration velocity

Outline

Redundancy Resolution

Orientations

Outline

Redundancy Resolution

Orientations

What is Redundancy?

[0] © Youtube: Ministry of Silly Walks

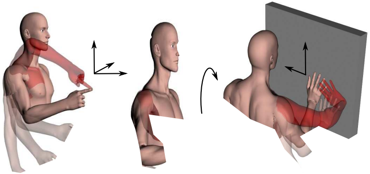
What is Redundancy?

Redundancy
More degree-of-freedom than what is required for your task
There are more than one solutions to achieve the desired task

Redundancy
More degree-of-freedom than what is required for your task
There are more than one solutions to achieve the desired task

EXAMPLE

- keep your finger tip and the same position and move your elbows



- joint-space velocity but no task-space velocity

Redundancy

Mathematically....

Redundancy

Mathematically....

- There are many solutions to $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$

Redundancy

Mathematically....

- There are many solutions to $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$
- Jacobian is rank deficient $\mathrm{r}(\mathbf{J}) \leq \mathcal{P} < \mathcal{N}$

Redundancy

Mathematically....

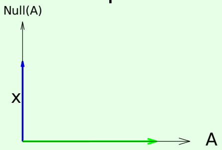
- There are many solutions to $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$
- Jacobian is rank deficient $\mathrm{r}(\mathbf{J}) \leq \mathcal{P} < \mathcal{N}$
- There exist a nullspace $\mathcal{N}(\mathbf{J}) \neq 0$

Mathematically....

- There are many solutions to $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$
- Jacobian is rank deficient $\mathrm{r}(\mathbf{J}) \leq \mathcal{P} < \mathcal{N}$
- There exist a nullspace $\mathcal{N}(\mathbf{J}) \neq 0$

Nullspace

The nullspace of a matrix **A** consists of all the vectors **x** that satisfy $\mathbf{Ax} = \mathbf{0}$

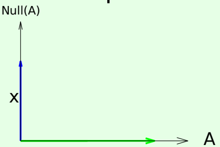


Mathematically....

- There are many solutions to $\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}}$
- Jacobian is rank deficient $\mathrm{r}(\mathbf{J}) \leq \mathcal{P} < \mathcal{N}$
- There exist a nullspace $\mathcal{N}(\mathbf{J}) \neq 0$

Nullspace

The nullspace of a matrix $\mathbf{A}$ consists of all the vectors $\mathbf{x}$ that satisfy $\mathbf{A}\mathbf{x} = \mathbf{0}$



so... our elbows generate some joint velocity $\dot{\mathbf{q}}^0$, and $\mathbf{J}\dot{\mathbf{q}}^0 = \mathbf{0}$

Inverse Kinemaitcs Controller with Redudnacy Resolution

$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{x} + \mathbf{N}\dot{\mathbf{q}}^0$ where

$\mathbf{N} = \mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}$ is the nullspace projection matrix of $\mathbf{J}$

Redundancy Resolution

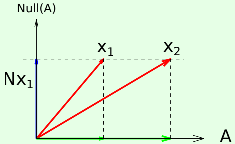
Inverse Kinemaitcs Controller with Redudnacy Resolution

$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{x} + \mathbf{N}\dot{\mathbf{q}}^{0}$ where

$\mathbf{N} = \mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}$ is the nullspace projection matrix of $\mathbf{J}$

Nullspace Projection Matrix

A nullspace projection of $\mathbf{A}$ projects a vector onto $\mathcal{N}(A)$



Inverse Kinemaitcs Controller with Redudnacy Resolution

$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{x} + \mathbf{N}\dot{\mathbf{q}}^{0}$ where

$\mathbf{N} = \mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}$ is the nullspace projection matrix of $\mathbf{J}$

- $\mathbf{N}$ projects $\dot{\mathbf{q}}^0$ onto the nullspace of the jacobian.
- $\dot{\mathbf{q}}^0$ can be **ANY** arbitrary vector.
  $\mathbf{N}\dot{\mathbf{q}}^0$ has **NO** effect on the $\dot{\mathbf{x}}$

$$\mathbf{J}\mathbf{N}\dot{\mathbf{q}}^0 = \mathbf{J}(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\dot{\mathbf{q}}^0 = (\mathbf{J} - \mathbf{J}\mathbf{J}^{\dagger}\mathbf{J})\dot{\mathbf{q}}^0 = (\mathbf{J} - \mathbf{J})\dot{\mathbf{q}}^0 = \mathbf{0}$$

Question: How do we choose $\dot{q}^0$?
Ideally, we like to move our arms in a comfortable way, or reduce the amount of energy we consume for a given task.

Redundancy

Question: How do we choose $\dot{q}^0$?
Ideally, we like to move our arms in a comfortable way, or reduce the amount of energy we consume for a given task.

Move the redundant dimension to some default positions

$$\dot{\mathbf{q}}^0 = \mathbf{q}^0 - \mathbf{q}$$

$\mathbf{q}^0$ is the home configuration

$\mathbf{q}$ is the current configuration

Matlab Example

Without Redundancy Resolution

With Redundancy Resolution

Outline

Redundancy Resolution

Orientations

Orientations

What about orientations?
Previously, we only control the positions and ignore the orientations.
How do we control orientations?

Rigid-Body Motion

Position

$$\mathbf{x} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\theta} \\ \mathbf{p} \end{bmatrix}$$

$\boldsymbol{\theta} \in \mathbb{R}^3$ orientation

$\mathbf{p} \in \mathbb{R}^3$ translation

Rigid-Body Motion

Position

$$\mathbf{x} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\theta} \\ \mathbf{p} \end{bmatrix}$$

$\boldsymbol{\theta} \in \mathbb{R}^3$ orientation

$\mathbf{p} \in \mathbb{R}^3$ translation

(Spatial) Velocity/Twist

$$\dot{\mathbf{x}} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{bmatrix}$$

$\boldsymbol{\omega} \in \mathbb{R}^3$ angular velocity

$\dot{\mathbf{p}} \in \mathbb{R}^3$ linear velocity

Rigid-Body Motion

Position

$$\mathbf{x} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\theta} \\ \mathbf{p} \end{bmatrix}$$

$\boldsymbol{\theta} \in \mathbb{R}^3$ orientation

$\mathbf{p} \in \mathbb{R}^3$ translation

(Spatial) Velocity/Twist

$$\dot{\mathbf{x}} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{bmatrix}$$

$\boldsymbol{\omega} \in \mathbb{R}^3$ angular velocity

$\dot{\mathbf{p}} \in \mathbb{R}^3$ linear velocity

Move to robot from point $\mathbf{p}$ to $\mathbf{p}^*$
Find $\dot{\mathbf{p}}^* = \mathbf{p}^* - \mathbf{p}$, and then inverse kinematics to find $\dot{\mathbf{q}}$

Rigid-Body Motion

Position

$$\mathbf{x} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\theta} \\ \mathbf{p} \end{bmatrix}$$

$\boldsymbol{\theta} \in \mathbb{R}^3$ orientation

$\mathbf{p} \in \mathbb{R}^3$ translation

(Spatial) Velocity/Twist

$$\dot{\mathbf{x}} \in \mathbb{R}^6 = \begin{bmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{bmatrix}$$

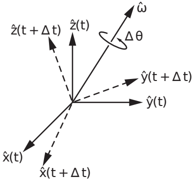
$\boldsymbol{\omega} \in \mathbb{R}^3$ angular velocity

$\dot{\mathbf{p}} \in \mathbb{R}^3$ linear velocity

Move to robot from point p to p^*
Find $\dot{\mathbf{p}}^* = \mathbf{p}^* - \mathbf{p}$, and then inverse kinematics to find $\dot{\mathbf{q}}$

Move to robot from point θ to θ^*
Find $\omega = ?$

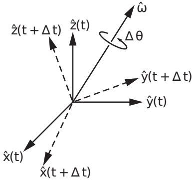
Angular Velocity



current rotation: $\mathbf{R} = [\hat{x}(t), \hat{y}(t), \hat{z}(t)]$
desired rotation: $\mathbf{R}^* = [\hat{x}(t+\triangle t), \hat{y}(t+\triangle t), \hat{z}(t+\triangle t)]$
angular velocity $\omega = ?$

[0] More details in Modern Robotics, Section 3.3

Angular Velocity



current rotation: $\mathbf{R} = [\hat{x}(t), \hat{y}(t), \hat{z}(t)]$
desired rotation: $\mathbf{R}^* = [\hat{x}(t+\triangle t), \hat{y}(t+\triangle t), \hat{z}(t+\triangle t)]$
angular velocity $\omega =?$

Skew Interpertation of a Twist

$$
\begin{aligned}
\dot{\mathsf{R}} &= \omega \times \mathsf{R} \\
&= \mathbb{S}(\omega)\mathsf{R}
\end{aligned}
$$

$$
\mathbb{S}(\omega) = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}
$$

$\mathbb{S}(\omega)$: skew-symmetric matrix representation of ω

[0] More details in Modern Robotics, Section 3.3

Angular Velocity

Desired position $\mathbf{R}^* \in \mathbb{SO}(3)$
provided by the planner

Angular Velocity

Desired position $\mathbf{R}^* \in \mathbb{SO}(3)$ provided by the planner

Durrent position $\mathbf{R} \in \mathbb{SO}(3)$ calculated by forward kinematics

Angular Velocity

Desired position $\mathbf{R}^* \in \mathbb{SO}(3)$
provided by the planner

Durrent position $\mathbf{R} \in \mathbb{SO}(3)$
calculated by forward kinematics

Approximation of Angular Velocity

$$\begin{aligned}\mathbb{S}(\omega) &= \dot{\mathbf{R}}\mathbf{R}^{-1} \\ &= \dot{\mathbf{R}}\mathbf{R}^{\top} \qquad \text{(since } \mathbf{R}^{\top} = \mathbf{R}^{-1}\text{)}\end{aligned}$$

Angular Velocity

Desired position $\mathbf{R}^* \in \mathbb{SO}(3)$
provided by the planner

Durrent position $\mathbf{R} \in \mathbb{SO}(3)$
calculated by forward kinematics

Approximation of Angular Velocity

$$
\begin{aligned}
\mathbb{S}(\omega) &= \dot{\mathbf{R}}\mathbf{R}^{-1} \\
&= \dot{\mathbf{R}}\mathbf{R}^{\top} \qquad\qquad (\text{since } \mathbf{R}^{\top} = \mathbf{R}^{-1}) \\
&\Downarrow
\end{aligned}
$$

$$
\dot{\mathbf{R}}\mathbf{R}^{\top} = (\mathbf{R}^* - \mathbf{R})\mathbf{R}^{\top} = \mathbf{R}^*\mathbf{R}^{\top} - \mathbf{I}
$$

Angular Velocity

Desired position $\mathbf{R}^* \in \mathbb{SO}(3)$ provided by the planner

Durrent position $\mathbf{R} \in \mathbb{SO}(3)$ calculated by forward kinematics

Approximation of Angular Velocity

$$\mathbb{S}(\omega) = \dot{\mathbf{R}}\mathbf{R}^{-1}$$

$$= \dot{\mathbf{R}}\mathbf{R}^{\top} \qquad \left(\text{since } \mathbf{R}^{\top} = \mathbf{R}^{-1}\right)$$

$$\Downarrow$$

$$\dot{\mathbf{R}}\mathbf{R}^{\top} = (\mathbf{R}^* - \mathbf{R})\mathbf{R}^{\top} = \mathbf{R}^*\mathbf{R}^{\top} - \mathbf{I}$$

$$\Downarrow$$

Convert skew symmetric matrix back to angles

$$\omega = \frac{1}{2}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

Matlab Example

Example (Move orientations)

$\mathbf{q} \in \mathbb{R}^7$: joints
$\mathbf{x} \in \mathbb{R}^3$: hand orientations
$\mathbf{J} \in \mathbb{R}^{3\times 7}$: Jacobian