Chapter 01 - Mobile Robot Kinematics

Instructor: Andrew Vardy

Adapted from “Principles of Robot Autonomy” by Joseph Lorenzetti and Marco Pavone

January 5, 2026

Motion Planning Needs Models

Planning a trajectory and choosing control inputs both depend on understanding how the robot moves.

Motion Planning Needs Models

Planning a trajectory and choosing control inputs both depend on understanding how the robot moves.

Dynamics and kinematics together capture the physical behavior needed for actionable autonomy.

Motion Planning Needs Models

Planning a trajectory and choosing control inputs both depend on understanding how the robot moves.

Dynamics and kinematics together capture the physical behavior needed for actionable autonomy.

Without a model that links objectives to actuation, even simple point-to-point tasks stall.

Dynamics vs Kinematics

Dynamics map applied forces to changes in the robot state (e.g., $F = ma$ realized on a car).

Dynamics vs Kinematics

Dynamics map applied forces to changes in the robot state (e.g., $F = ma$ realized on a car).

Kinematics impose motion restrictions that arise from geometry rather than forces.

Dynamics vs Kinematics

Dynamics map applied forces to changes in the robot state (e.g., $F = ma$ realized on a car).

Kinematics impose motion restrictions that arise from geometry rather than forces.

Both perspectives are necessary to describe what a robot can actually do.

Modeling Questions

Dynamics are affected by mass changes, while kinematics are not—use that heuristic to classify constraints.

Modeling Questions

Dynamics are affected by mass changes, while kinematics are not—use that heuristic to classify constraints.

Decide which dynamics/kinematics actually matter for the task and which can be simplified away.

Modeling Questions

Dynamics are affected by mass changes, while kinematics are not—use that heuristic to classify constraints.

Decide which dynamics/kinematics actually matter for the task and which can be simplified away.

The right model balances complexity against accuracy for the objective at hand.

Balancing Fidelity

High-fidelity car models include engine, suspension, and tire dynamics that explain phenomena like drifting.

Balancing Fidelity

High-fidelity car models include engine, suspension, and tire dynamics that explain phenomena like drifting.

Many applications can instead enforce a kinematic no-side-slip constraint for tractability.

Balancing Fidelity

High-fidelity car models include engine, suspension, and tire dynamics that explain phenomena like drifting.

Many applications can instead enforce a kinematic no-side-slip constraint for tractability.

Picking the right abstraction speeds up planning without losing the essential behavior.

Chapter Roadmap

Express robot configurations via generalized coordinates.

Chapter Roadmap

Express robot configurations via generalized coordinates.

Encode kinematic constraints with those coordinates.

Chapter Roadmap

Express robot configurations via generalized coordinates.

Encode kinematic constraints with those coordinates.

Distinguish holonomic from nonholonomic constraints.

Chapter Roadmap

Express robot configurations via generalized coordinates.

Encode kinematic constraints with those coordinates.

Distinguish holonomic from nonholonomic constraints.

Build practical kinematic models for wheeled robots.

Generalized Coordinates

A robot's configuration can be represented in many ways; any complete set defines generalized coordinates.

Generalized Coordinates

A robot's configuration can be represented in many ways; any complete set defines generalized coordinates.

Collect the coordinates in $\boldsymbol{\xi} \in \mathbb{R}^n$ to specify the state uniquely.

Generalized Coordinates

A robot's configuration can be represented in many ways; any complete set defines generalized coordinates.

Collect the coordinates in $\boldsymbol{\xi} \in \mathbb{R}^n$ to specify the state uniquely.

Trajectories are mappings $\boldsymbol{\xi}(t) : \mathbb{R} \to \mathbb{R}^n$ that describe motion over time.

Example: Wheel Coordinates

Use (x, y, θ) to record where a wheel contacts the plane and which direction it points.

Example: Wheel Coordinates

Use (x, y, θ) to record where a wheel contacts the plane and which direction it points.

The generalized coordinate vector becomes $\boldsymbol{\xi} = [x \ y \ \theta]^\top$.

Example: Wheel Coordinates

Use (x, y, θ) to record where a wheel contacts the plane and which direction it points.

The generalized coordinate vector becomes $\boldsymbol{\xi} = [x\ y\ \theta]^\top$.

This minimal set fully captures the wheel's configuration.

Example: Wheel Coordinates

Use (x, y, θ) to record where a wheel contacts the plane and which direction it points.

The generalized coordinate vector becomes $\boldsymbol{\xi} = [x \ y \ \theta]^\top$.

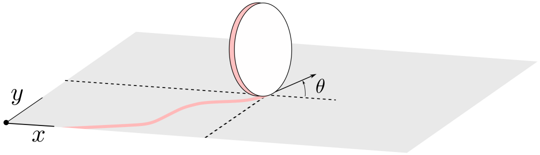
This minimal set fully captures the wheel's configuration.

Example: Wheel Coordinates

Use (x, y, θ) to record where a wheel contacts the plane and which direction it points.

The generalized coordinate vector becomes $\boldsymbol{\xi} = [x \ y \ \theta]^\top$.

This minimal set fully captures the wheel's configuration.



Kinematic Constraint Definition

Kinematic constraints relate coordinates and velocities to limit motion.

Kinematic Constraint Definition

Kinematic constraints relate coordinates and velocities to limit motion.

For k constraints, write $a_i(\xi, \dot{\xi}) = 0$ for $i = 1, \ldots, k < n$.

Kinematic Constraint Definition

Kinematic constraints relate coordinates and velocities to limit motion.

For k constraints, write $a_i(\xi, \dot{\xi}) = 0$ for $i = 1, \ldots, k < n$.

These expressions encode contacts or linkages without referencing forces.

Kinematic Constraint Definition

Kinematic constraints relate coordinates and velocities to limit motion.

For k constraints, write $a_i(\xi, \dot{\xi}) = 0$ for $i = 1, \ldots, k < n$.

These expressions encode contacts or linkages without referencing forces.

Kinematic Constraint Definition

Kinematic constraints relate coordinates and velocities to limit motion.

For k constraints, write $a_i(\xi, \dot{\xi}) = 0$ for $i = 1, \ldots, k < n$.

These expressions encode contacts or linkages without referencing forces.

$$a_i(\xi, \dot{\xi}) = 0$$

Pfaffian Constraints

Many robotic constraints are linear in $\dot{\boldsymbol{\xi}}$, yielding Pfaffian form $a_i^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = 0$.

Pfaffian Constraints

Many robotic constraints are linear in $\dot{\boldsymbol{\xi}}$, yielding Pfaffian form $a_i^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = 0$.

Stack the constraints as $A^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = \mathbf{0}$ for compact notation.

Pfaffian Constraints

Many robotic constraints are linear in $\dot{\boldsymbol{\xi}}$, yielding Pfaffian form $a_i^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = 0$.

Stack the constraints as $A^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = \mathbf{0}$ for compact notation.

Each column of $A(\boldsymbol{\xi})$ captures how a constraint depends on configuration.

Pfaffian Constraints

Many robotic constraints are linear in $\dot{\boldsymbol{\xi}}$, yielding Pfaffian form $a_i^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = 0$.

Stack the constraints as $A^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = \mathbf{0}$ for compact notation.

Each column of $A(\boldsymbol{\xi})$ captures how a constraint depends on configuration.

Pfaffian Constraints

Many robotic constraints are linear in $\dot{\boldsymbol{\xi}}$, yielding Pfaffian form $a_i^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = 0$.

Stack the constraints as $A^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = \mathbf{0}$ for compact notation.

Each column of $A(\boldsymbol{\xi})$ captures how a constraint depends on configuration.

$$A^\top(\boldsymbol{\xi})\dot{\boldsymbol{\xi}} = \mathbf{0}$$

Example: Pendulum Constraint

A rigid pendulum with Cartesian coordinates (x, y) satisfies $x^2 + y^2 = L^2$.

Example: Pendulum Constraint

A rigid pendulum with Cartesian coordinates (x, y) satisfies $x^2 + y^2 = L^2$.

Differentiating yields the Pfaffian form $2x\dot{x} + 2y\dot{y} = 0$.

Example: Pendulum Constraint

A rigid pendulum with Cartesian coordinates (x, y) satisfies $x^2 + y^2 = L^2$.

Differentiating yields the Pfaffian form $2x\dot{x} + 2y\dot{y} = 0$.

Using the single coordinate θ automatically enforces the circle constraint.

Example: Pendulum Constraint

A rigid pendulum with Cartesian coordinates (x, y) satisfies $x^2 + y^2 = L^2$.

Differentiating yields the Pfaffian form $2x\dot{x} + 2y\dot{y} = 0$.

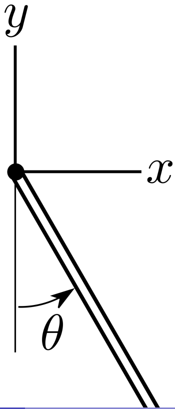
Using the single coordinate θ automatically enforces the circle constraint.

Example: Pendulum Constraint

A rigid pendulum with Cartesian coordinates (x, y) satisfies $x^2 + y^2 = L^2$.

Differentiating yields the Pfaffian form $2x\dot{x} + 2y\dot{y} = 0$.

Using the single coordinate θ automatically enforces the circle constraint.



Example: No-Slip Wheel

A rolling wheel with coordinates $[x, y, \theta]^\top$ maintains zero lateral velocity.

Example: No-Slip Wheel

A rolling wheel with coordinates $[x, y, \theta]^\top$ maintains zero lateral velocity.

Using basis vectors aligned with the wheel gives the constraint $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$.

Example: No-Slip Wheel

A rolling wheel with coordinates $[x, y, \theta]^\top$ maintains zero lateral velocity.

Using basis vectors aligned with the wheel gives the constraint $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$.

The Pfaffian form keeps the constraint linear in $\dot{x}$ and $\dot{y}$.

Example: No-Slip Wheel

A rolling wheel with coordinates $[x, y, \theta]^\top$ maintains zero lateral velocity.

Using basis vectors aligned with the wheel gives the constraint $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$.

The Pfaffian form keeps the constraint linear in $\dot{x}$ and $\dot{y}$.

Example: No-Slip Wheel

A rolling wheel with coordinates $[x, y, \theta]^\top$ maintains zero lateral velocity.

Using basis vectors aligned with the wheel gives the constraint $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$.

The Pfaffian form keeps the constraint linear in $\dot{x}$ and $\dot{y}$.

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0$$

Holonomic Constraints

Holonomic constraints depend only on configuration: $h_i(\boldsymbol{\xi}) = 0$.

Holonomic Constraints

Holonomic constraints depend only on configuration: $h_i(\boldsymbol{\xi}) = 0$.

Differentiating h_i reproduces the equivalent Pfaffian form.

Holonomic Constraints

Holonomic constraints depend only on configuration: $h_i(\boldsymbol{\xi}) = 0$.

Differentiating h_i reproduces the equivalent Pfaffian form.

Holonomic systems contain only such integrable constraints.

Holonomic Constraints

Holonomic constraints depend only on configuration: $h_i(\boldsymbol{\xi}) = 0$.

Differentiating h_i reproduces the equivalent Pfaffian form.

Holonomic systems contain only such integrable constraints.

Holonomic Constraints

Holonomic constraints depend only on configuration: $h_i(\boldsymbol{\xi}) = 0$.

Differentiating h_i reproduces the equivalent Pfaffian form.

Holonomic systems contain only such integrable constraints.

$$\frac{dh_i(\boldsymbol{\xi})}{dt} = \frac{dh_i(\boldsymbol{\xi})}{d\boldsymbol{\xi}}\dot{\boldsymbol{\xi}} = 0$$

Holonomic Access

Holonomic constraints restrict the reachable configurations to an $(n-k)$-dimensional subset.

Holonomic Access

Holonomic constraints restrict the reachable configurations to an $(n-k)$-dimensional subset.

A pendulum mass remains on a circle of radius L, reducing the space to one dimension.

Holonomic Access

Holonomic constraints restrict the reachable configurations to an $(n-k)$-dimensional subset.

A pendulum mass remains on a circle of radius L, reducing the space to one dimension.

The no-slip wheel constraint is not integrable, so it does not limit which (x, y) can be reached.

Nonholonomic Constraints

Nonholonomic constraints are Pfaffian but cannot be integrated to $h_i(\boldsymbol{\xi}) = 0$.

Nonholonomic Constraints

Nonholonomic constraints are Pfaffian but cannot be integrated to $h_i(\boldsymbol{\xi}) = 0$.

They restrict allowable motions between configurations rather than the configurations themselves.

Nonholonomic Constraints

Nonholonomic constraints are Pfaffian but cannot be integrated to $h_i(\boldsymbol{\xi}) = 0$.

They restrict allowable motions between configurations rather than the configurations themselves.

Any system subject to at least one such constraint is nonholonomic.

Reminder: Null Space

Concept Recap

Reminder: Null Space

Concept Recap

The null space of a matrix A is the set of all vectors $\mathbf{x}$ such that $A\mathbf{x} = \mathbf{0}$, where $\mathbf{0}$ denotes the zero vector.

Reminder: Null Space

Concept Recap

The null space of a matrix A is the set of all vectors $\mathbf{x}$ such that $A\mathbf{x} = \mathbf{0}$, where $\mathbf{0}$ denotes the zero vector.

Emphasizing the zero vector clarifies that every component of $A\mathbf{x}$ must vanish simultaneously.

Reminder: Null Space

Concept Recap

The null space of a matrix A is the set of all vectors $\mathbf{x}$ such that $A\mathbf{x} = \mathbf{0}$, where $\mathbf{0}$ denotes the zero vector.

Emphasizing the zero vector clarifies that every component of $A\mathbf{x}$ must vanish simultaneously.

Example: For $A = \begin{bmatrix} 1 & 2 \\ 3 & 6 \end{bmatrix}$, any scalar multiple of $\begin{bmatrix} 2 \\ -1 \end{bmatrix}$ lies in the null space because $A \begin{bmatrix} 2 \\ -1 \end{bmatrix} = \mathbf{0}$.

Instantaneous Motion Limits

Nonholonomic constraints confine $\dot{\xi}$ to the null space of $a_i^\top(\xi)$.

Instantaneous Motion Limits

Nonholonomic constraints confine $\dot{\xi}$ to the null space of $a_i^\top(\xi)$.

For the wheel, the null space is spanned by the rolling direction and pure spin.

Instantaneous Motion Limits

Nonholonomic constraints confine $\dot{\xi}$ to the null space of $a_i^\top(\xi)$.

For the wheel, the null space is spanned by the rolling direction and pure spin.

Any feasible velocity is a combination of those motions.

Building Kinematic Models

After identifying constraints, express motion as $\dot{\boldsymbol{\xi}} = G(\boldsymbol{\xi})\boldsymbol{u}$.

Building Kinematic Models

After identifying constraints, express motion as $\dot{\boldsymbol{\xi}} = G(\boldsymbol{\xi})\boldsymbol{u}$.

Columns of $G(\boldsymbol{\xi})$ span the null space of $A^\top(\boldsymbol{\xi})$, guaranteeing constraint satisfaction.

Building Kinematic Models

After identifying constraints, express motion as $\dot{\boldsymbol{\xi}} = G(\boldsymbol{\xi})\boldsymbol{u}$.

Columns of $G(\boldsymbol{\xi})$ span the null space of $A^\top(\boldsymbol{\xi})$, guaranteeing constraint satisfaction.

Inputs $\boldsymbol{u}$ parameterize allowable motions of dimension $n - k$.

Example: Wheel Kinematic Model

The wheel's constraint matrix has a one-dimensional row space, so $G(\xi)$ has two columns.

Example: Wheel Kinematic Model

The wheel's constraint matrix has a one-dimensional row space, so $G(\xi)$ has two columns.

Choose columns $[\cos\theta \;\; \sin\theta \; 0]^\top$ (rolling) and $[0 \; 0 \; 1]^\top$ (spin).

Example: Wheel Kinematic Model

The wheel's constraint matrix has a one-dimensional row space, so $G(\xi)$ has two columns.

Choose columns $[\cos\theta \;\; \sin\theta \; 0]^\top$ (rolling) and $[0\; 0\; 1]^\top$ (spin).

Inputs u_1 and u_2 become forward speed and rotational rate, respectively.

Example: Wheel Kinematic Model

The wheel's constraint matrix has a one-dimensional row space, so $G(\xi)$ has two columns.

Choose columns $[\cos\theta \ \sin\theta \ 0]^\top$ (rolling) and $[0\ 0\ 1]^\top$ (spin).

Inputs u_1 and u_2 become forward speed and rotational rate, respectively.

Example: Wheel Kinematic Model

The wheel's constraint matrix has a one-dimensional row space, so $G(\xi)$ has two columns.

Choose columns $[\cos\theta \;\; \sin\theta \;\; 0]^\top$ (rolling) and $[0\;0\;1]^\top$ (spin).

Inputs u_1 and u_2 become forward speed and rotational rate, respectively.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Why Wheeled Models

Wheeled robots are common thanks to mobility and mechanical simplicity.

Why Wheeled Models

Wheeled robots are common thanks to mobility and mechanical simplicity.

Standard templates (unicycle, differential drive, etc.) cover many platforms.

Why Wheeled Models

Wheeled robots are common thanks to mobility and mechanical simplicity.

Standard templates (unicycle, differential drive, etc.) cover many platforms.

Studying canonical cases makes it easier to adapt kinematic reasoning to new robots.

Unicycle Model

Approximates the robot as a single wheel with state $[x, y, \theta]^\top$.

Unicycle Model

Approximates the robot as a single wheel with state $[x, y, \theta]^\top$.

Shares the same no-slip constraint as the earlier wheel example.

Unicycle Model

Approximates the robot as a single wheel with state $[x, y, \theta]^\top$.

Shares the same no-slip constraint as the earlier wheel example.

Inputs (v, ω) command forward speed and body rotation.

Unicycle Model

Approximates the robot as a single wheel with state $[x, y, \theta]^\top$.

Shares the same no-slip constraint as the earlier wheel example.

Inputs (v, ω) command forward speed and body rotation.

Unicycle Model

Approximates the robot as a single wheel with state $[x, y, \theta]^\top$.

Shares the same no-slip constraint as the earlier wheel example.

Inputs (v, ω) command forward speed and body rotation.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$$

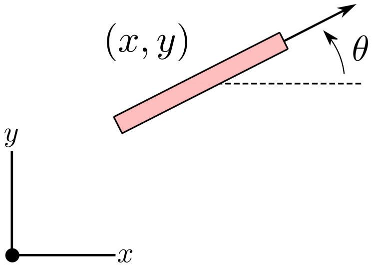
Unicycle Model

Approximates the robot as a single wheel with state $[x, y, \theta]^\top$.

Shares the same no-slip constraint as the earlier wheel example.

Inputs (v, ω) command forward speed and body rotation.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}$$



Differential Drive Geometry

Two rear wheels share an axle of width L; a passive front wheel carries no constraints.

Differential Drive Geometry

Two rear wheels share an axle of width L; a passive front wheel carries no constraints.

Positions of the left and right wheels offset the body frame by $\pm\frac{L}{2}$.

Differential Drive Geometry

Two rear wheels share an axle of width L; a passive front wheel carries no constraints.

Positions of the left and right wheels offset the body frame by $\pm\frac{L}{2}$.

Both wheels obey the same no-slip condition because they are rigidly linked.

Differential Drive Geometry

Two rear wheels share an axle of width L; a passive front wheel carries no constraints.

Positions of the left and right wheels offset the body frame by $\pm \frac{L}{2}$.

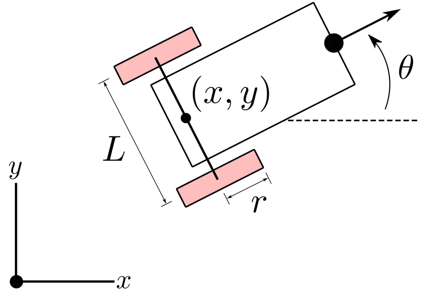
Both wheels obey the same no-slip condition because they are rigidly linked.

Differential Drive Geometry

Two rear wheels share an axle of width L; a passive front wheel carries no constraints.

Positions of the left and right wheels offset the body frame by $\pm\frac{L}{2}$.

Both wheels obey the same no-slip condition because they are rigidly linked.



Differential Drive Constraints

Wheel positions: $p_l = [x - \frac{L}{2}\sin\theta,\ y + \frac{L}{2}\cos\theta]$ and $p_r = [x + \frac{L}{2}\sin\theta,\ y - \frac{L}{2}\cos\theta]$.

Differential Drive Constraints

Wheel positions: $p_l = [x - \frac{L}{2}\sin\theta,\ y + \frac{L}{2}\cos\theta]$ and $p_r = [x + \frac{L}{2}\sin\theta,\ y - \frac{L}{2}\cos\theta]$.

Differentiating gives $\dot{p}_l$ and $\dot{p}_r$; both lead to $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$.

Differential Drive Constraints

Wheel positions: $p_l = [x - \frac{L}{2}\sin\theta,\ y + \frac{L}{2}\cos\theta]$ and $p_r = [x + \frac{L}{2}\sin\theta,\ y - \frac{L}{2}\cos\theta]$.

Differentiating gives $\dot{p}_l$ and $\dot{p}_r$; both lead to $\dot{x}\sin\theta - \dot{y}\cos\theta = 0$.

The redundant constraints confirm that a single nonholonomic condition governs the chassis.

Differential Drive Inputs

Use wheel rotation rates (ω_l, ω_r) instead of (v, ω) for more realistic actuation.

Differential Drive Inputs

Use wheel rotation rates (ω_l, ω_r) instead of (v, ω) for more realistic actuation.

Speed satisfies $v = \frac{r}{2}(\omega_l + \omega_r)$, while rotation obeys $\omega = \frac{r}{L}(\omega_r - \omega_l)$.

Differential Drive Inputs

Use wheel rotation rates (ω_l, ω_r) instead of (v, ω) for more realistic actuation.

Speed satisfies $v = \frac{r}{2}(\omega_l + \omega_r)$, while rotation obeys $\omega = \frac{r}{L}(\omega_r - \omega_l)$.

The resulting model maps wheel rates to body motion via

Differential Drive Inputs

Use wheel rotation rates (ω_l, ω_r) instead of (v, ω) for more realistic actuation.

Speed satisfies $v = \frac{r}{2}(\omega_l + \omega_r)$, while rotation obeys $\omega = \frac{r}{L}(\omega_r - \omega_l)$.

The resulting model maps wheel rates to body motion via

Differential Drive Inputs

Use wheel rotation rates (ω_l, ω_r) instead of (v, ω) for more realistic actuation.

Speed satisfies $v = \frac{r}{2}(\omega_l + \omega_r)$, while rotation obeys $\omega = \frac{r}{L}(\omega_r - \omega_l)$.

The resulting model maps wheel rates to body motion via

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}\cos\theta & \frac{r}{2}\cos\theta \\ \frac{r}{2}\sin\theta & \frac{r}{2}\sin\theta \\ \frac{r}{L} & -\frac{r}{L} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix}.$$

Why Add Dynamics

Purely kinematic models ignore actuator limits, slip, and inertia.

Why Add Dynamics

Purely kinematic models ignore actuator limits, slip, and inertia.

Inputs like velocity or wheel rate may be hard to command without considering required forces.

Why Add Dynamics

Purely kinematic models ignore actuator limits, slip, and inertia.

Inputs like velocity or wheel rate may be hard to command without considering required forces.

Some tasks therefore augment kinematics with lightweight dynamics.

Extending the Unicycle

Replace velocity inputs with accelerations by adding integrator states.

Extending the Unicycle

Replace velocity inputs with accelerations by adding integrator states.

Linear acceleration a drives $\dot{v}$, and angular acceleration α drives $\dot{\omega}$.

Extending the Unicycle

Replace velocity inputs with accelerations by adding integrator states.

Linear acceleration a drives $\dot{v}$, and angular acceleration α drives $\dot{\omega}$.

The extended model couples kinematics with simple dynamics:

Extending the Unicycle

Replace velocity inputs with accelerations by adding integrator states.

Linear acceleration a drives $\dot{v}$, and angular acceleration α drives $\dot{\omega}$.

The extended model couples kinematics with simple dynamics:

Extending the Unicycle

Replace velocity inputs with accelerations by adding integrator states.

Linear acceleration a drives $\dot{v}$, and angular acceleration α drives $\dot{\omega}$.

The extended model couples kinematics with simple dynamics:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{v} \\ \dot{\theta} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ a \\ \omega \\ \alpha \end{bmatrix}.$$

Modeling Takeaways

Choose the simplest model that captures the geometry, actuation, and performance needs of the task.

Modeling Takeaways

Choose the simplest model that captures the geometry, actuation, and performance needs of the task.

Add dynamics when necessary, but leverage kinematic structure to keep planning manageable.

Modeling Takeaways

Choose the simplest model that captures the geometry, actuation, and performance needs of the task.

Add dynamics when necessary, but leverage kinematic structure to keep planning manageable.

Revisit models as tasks evolve to ensure accuracy remains aligned with objectives.