Principles of Robot Autonomy I

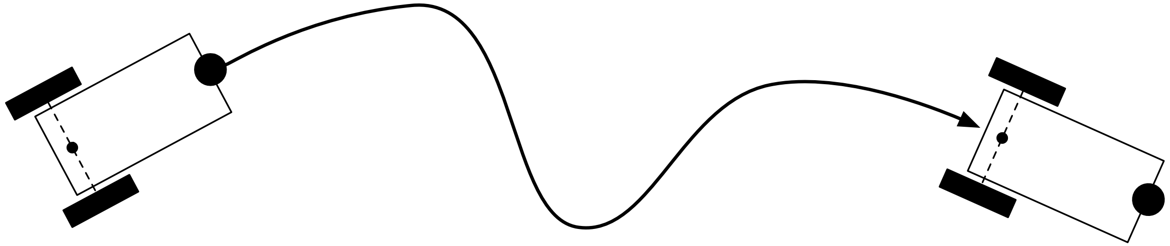
Trajectory tracking and closed-loop control

Stanford University

Autonomous Systems Lab

Motion control

- Given a nonholonomic system, how to control its motion from an initial configuration to a final, desired configuration



- Aim
  - Learn how to handle bound constraints via space-time separation
  - Learn about trajectory tracking
  - Learn about closed-loop control

- Readings
  - B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo. Robotics: modelling, planning and control. 2010. Chapter 11.

Summary of previous lecture

- A nonlinear system $\dot{\mathbf{x}} = \mathbf{a}(\mathbf{x}, \mathbf{u})$ is differentially flat if there exists a set of outputs $\mathbf{z}$ such that

$$\mathbf{x} = \beta(\mathbf{z}, \dot{\mathbf{z}}, \ldots, \mathbf{z}^{(q)})$$
$$\mathbf{u} = \gamma(\mathbf{z}, \dot{\mathbf{z}}, \ldots, \mathbf{z}^{(q)})$$

- One can then use any interpolation scheme (e.g., polynomial) to plan the trajectory of $\mathbf{z}$ in such a way as to satisfy the appropriate boundary conditions
- The evolution of the state variables $\mathbf{x}$, together with the associated control inputs $\mathbf{u}$, can then be computed algebraically from $\mathbf{z}$

Summary of previous lecture

- Constraints on the system can be transformed into the flat output space and (typically) become limits on the curvature or higher order derivative properties of the curve

- An important class of constraints is represented by bounds on some of the system variables, and in particular the inputs, for example:

$$|v(t)| \leq v_{\max} \text{ and } |\omega(t)| \leq \omega_{\max}$$

- Bound constraints can be effectively addressed via time scaling

Path and time scaling law

- The problem of planning a trajectory can be divided into two steps:
  1. computing a path, that is, a purely geometric description of the sequence of configurations achieved by the robot, and
  2. devising a time scaling law, which specifies the times when those configurations are reached
- Mathematically, a trajectory $\mathbf{x}(t)$ can be broken down into a geometric path $\mathbf{x}(s)$ and a timing law $s = s(t)$, with the parameter s varying between $s(t_0) = s_0$ and $s\left(t_f\right) = s_f$ in a monotonic fashion, i.e., with $\dot{s}(t) > 0$
- A possible choice for s is the arc length along the path (in this case, $s_0 = 0$, and $s_f = L$, the length of the path)

Enforcing bound constraints

- Such a space-time separation implies that

$$\dot{\mathbf{x}}(t) = \frac{d\mathbf{x}(t)}{dt} = \frac{d\mathbf{x}(s(t))}{ds}\,\dot{s}(t)$$

- Thus, once the geometric path is determined, the choice of a timing law $s = s(t)$ will identify a particular trajectory along this path, with a corresponding set of time-scaled inputs (Problem 1 in pset)

- Example, for unicycle model
  - $v(t) = \frac{d|\mathbf{x}(t)|}{dt} = \frac{d|\mathbf{x}(s(t))|}{ds}\dot{s}(t) = \tilde{v}(s)\dot{s}(t)$
  - $\omega(t) = \frac{d\theta(t)}{dt} = \frac{d\theta(s(t))}{ds}\dot{s}(t) = \tilde{\omega}(s)\dot{s}(t) = \tilde{\omega}(s)\frac{v(t)}{\tilde{v}(s)}$

- Simplest choice, with s being arc length: $s(t) = t\,L/T$

Trajectory tracking

- Back to two-step design strategy

Tracking control law

$$\mathbf{u}^*(t) = \mathbf{u}_d(t) + \boxed{\pi(\mathbf{x}(t), \mathbf{x}(t) - \mathbf{x}_d(t))}$$

- Reference trajectory and control history (i.e., $\mathbf{x}_d(t)$ and $\mathbf{u}_d(t)$) are computed via open-loop techniques (e.g., differential flatness)
- For reference tracking (Problem 3 in pset)
  - Geometric (e.g., pursuit) strategies
  - Linearization (either approximate or exact) + linear structure
  - Non-linear control
  - Optimization-based techniques (e.g., MPC)

Trajectory tracking for differentially flat systems

- Key fact (see, e.g,. Levine 2009): a differentially flat system can be linearized by (dynamic) feedback and coordinate change, that is it can be equivalently transformed into the system

$$\mathbf{z}^{(q+1)} = \mathbf{w}$$

- One can then design a tracking controller for the linearized system by using linear control techniques; in particular, for a given reference flat output $\mathbf{z}_d$, define the *component-wise* error

$$e_i := z_i - z_{i,d}, \text{ which implies } e_i^{(q+1)} = w_i - w_{i,d}$$

- For guaranteed convergence to zero of tracking error, one can set

$$w_i = w_{i,d} - \sum_{j=0}^{q} k_{i,j} e_i^{(j)},$$

with the gains $\{k_{i,j}\}$ chosen so as to enforce stability

Trajectory tracking for differentially flat systems

- Example: dynamically extended unicycle model

$$\begin{aligned}\dot{x}(t) &= V\cos(\theta(t))\\ \dot{y}(t) &= V\sin(\theta(t))\\ \dot{V}(t) &= a(t)\\ \dot{\theta}(t) &= \omega(t)\end{aligned}$$

- The system is differentially flat with flat outputs (x, y), in particular

$$\begin{bmatrix}\ddot{x}(t)\\ \ddot{y}(t)\end{bmatrix} = \underbrace{\begin{bmatrix}\cos(\theta) & -V\sin(\theta)\\ \sin(\theta) & V\cos(\theta)\end{bmatrix}}_{:=J}\begin{bmatrix}a\\ \omega\end{bmatrix} := \begin{bmatrix}w_1\\ w_2\end{bmatrix}$$

Trajectory tracking for differentially flat systems

- Then one can use the following virtual control law for trajectory tracking:

$$w_1 = \ddot{x}_d + k_{px}(x_d - x) + k_{dx}(\dot{x}_d - \dot{x})$$
$$w_2 = \ddot{y}_d + k_{py}(y_d - y) + k_{dy}(\dot{y}_d - \dot{y})$$

where $k_{px}, k_{dx}, k_{py}, k_{dy} > 0$ are control gains

- Such a law guarantees exponential convergence to zero of the Cartesian tracking error

Closed-loop control

- General closed-loop control: we want to find

$$\mathbf{u}^*(t) = \pi(\mathbf{x}(t), t)$$

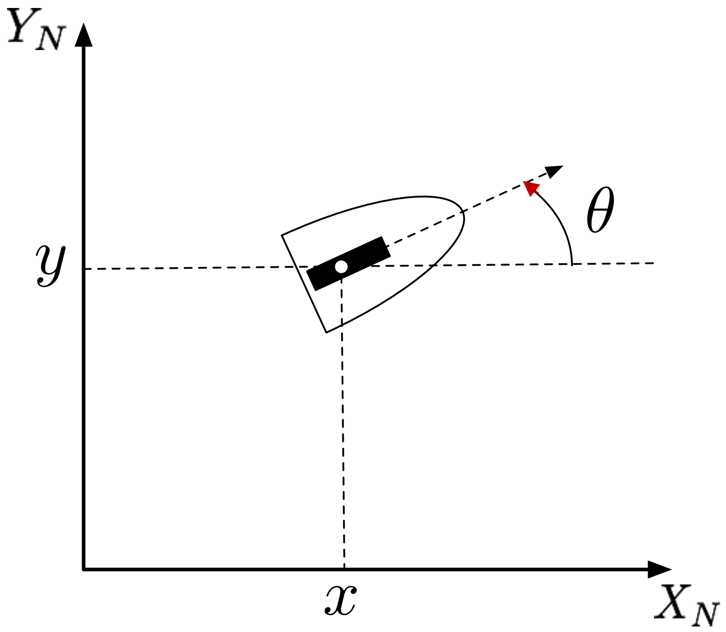
- Main techniques:
  - Hamilton–Jacobi–Bellman equation, dynamic programming
  - Lyapunov analysis

For an in-depth study of this topic, see AA203 "Optimal and Learning-based Control" (Spring 2020)

Closed-loop control: posture regulation

- Consider a differential drive mobile robot

$$\dot{x}(t) = V(t)\cos(\theta(t))$$
$$\dot{y}(t) = V(t)\sin(\theta(t))$$
$$\dot{\theta}(t) = \omega(t)$$

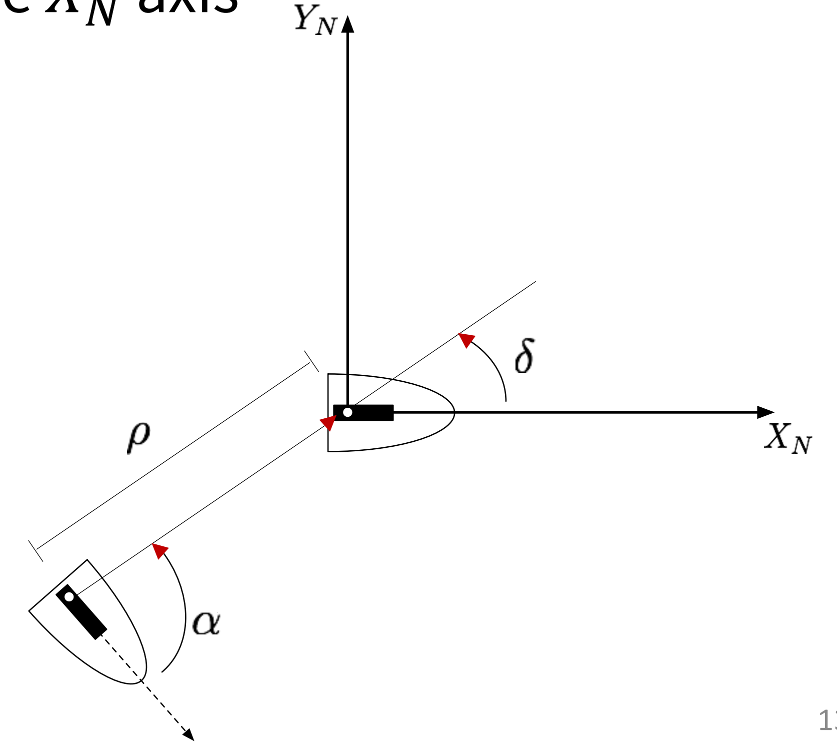


- Inputs: *V* (linear velocity of the wheel) and ω (angular velocity around the vertical axis)
- Goal: drive the robot to the origin *[0, 0, 0]*

Control based on polar coordinates

- Polar coordinates
  - ρ: distance of the reference point of the unicycle from the goal
  - α: angle of the pointing vector to the goal w.r.t. the unicycle main axis
  - δ: angle of the same pointing vector w.r.t. the X_N axis

- Coordinate transformation
  - $\rho = \sqrt{x^2 + y^2}$
  - $\alpha = \text{atan2}(y, x) - \theta + \pi$
  - $\delta = \alpha + \theta$



Equations in polar coordinates

- In polar coordinates, the unicycle equations become

$$\dot{\rho}(t) = -V(t)\cos(\alpha(t))$$
$$\dot{\alpha}(t) = V(t)\frac{\sin(\alpha(t))}{\rho(t)} - \omega(t)$$
$$\dot{\delta}(t) = V(t)\frac{\sin(\alpha(t))}{\rho(t)}$$

- In order to achieve the goal posture, variables (ρ, α, δ) should all converge to zero

Control law

- Closed-loop control law (Problem 2 in pset):

$$V = k_1 \rho \cos(\alpha)$$

$$\omega = k_2 \alpha + k_1 \frac{\sin(\alpha)\cos(\alpha)}{\alpha}(\alpha + k_3 \delta),$$

- If $k_1, k_2, k_3 > 0$, then closed-loop system is globally asymptotically driven to the posture $(0,0,0)$!

- For more details, see M. Aicardi, G. Casalino, A. Bicchi, and A. Balestrino (1995). Closed loop steering of unicycle like vehicles via Lyapunov techniques. IEEE Robotics & Automation Magazine.

Summary

- We covered closed-loop control along two main dimensions
    1. Trajectory tracking (useful to infuse robustness of point-to-point motion)
    2. Posture regulation (useful for final phase of motion)

- We'll see in Pset 2 how the topics of differential flatness, trajectory tracking, posture regulation, and motion planning will lead to an end-to-end trajectory optimization module

Next time: more on direct / indirect methods

$$\dot{\mathbf{x}}^*(t) = \frac{\partial H}{\partial \mathbf{p}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$
$$\dot{\mathbf{p}}^*(t) = -\frac{\partial H}{\partial \mathbf{x}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$
$$\mathbf{0} = \frac{\partial H}{\partial \mathbf{u}}(\mathbf{x}^*(t), \mathbf{u}^*(t), \mathbf{p}^*(t), t)$$