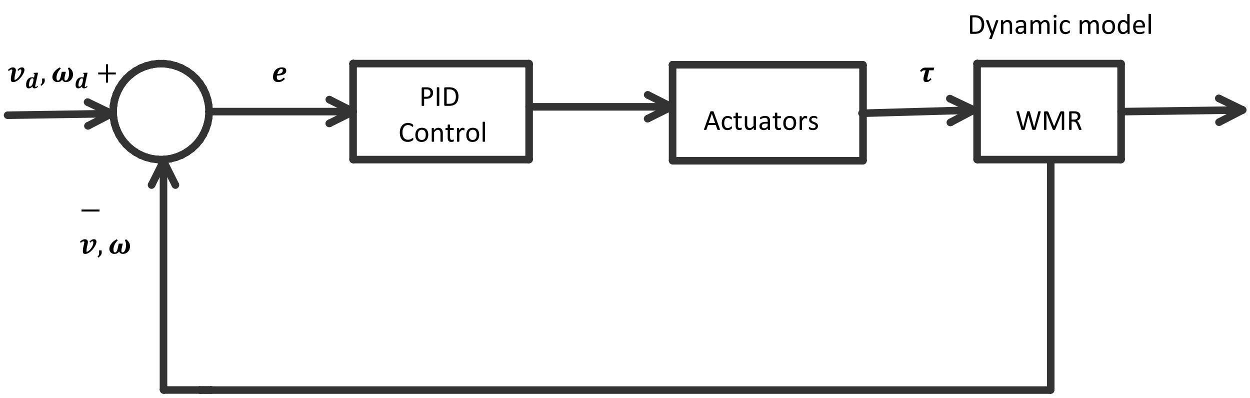
MOTION CONTROL

viernes, 25 de octubre de 2019 11:11 a. m.

- We will be using **velocities** as the **control inputs**. Because:
  - WMRs are in general single-body robots so the dynamic effects are not as important as in multi-body robots. Also the working speed of the robots is not high enough to consider the dynamics.
  - Internally WMRs are already designed to work with velocity as a control input and this is regulated with an internal feedback loop that uses a PID controller over the dynamic model that we cannot access. So if this internal control works correctly **we can directly work on the kinematic model**.
- Internal PID controller:



- Problems, considering the unicycle:
  - Trajectory tracking:
    - Predictable because we expect the robot to converge to the trajectory.
    - Easy
  - Posture regulation (parking):
    - Unpredictable because the robot can, in theory, go anywhere before reaching the desired position.
    - Difficult

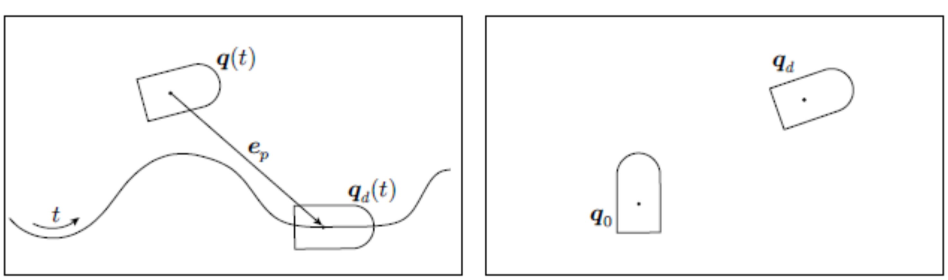
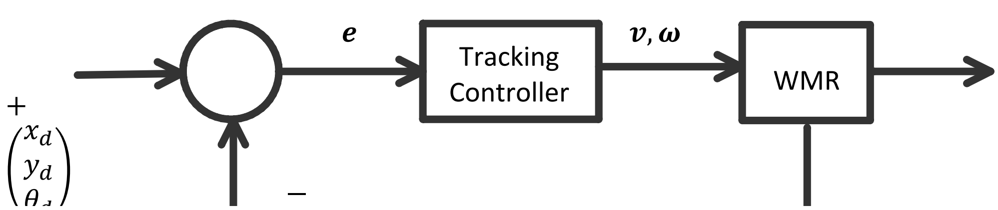
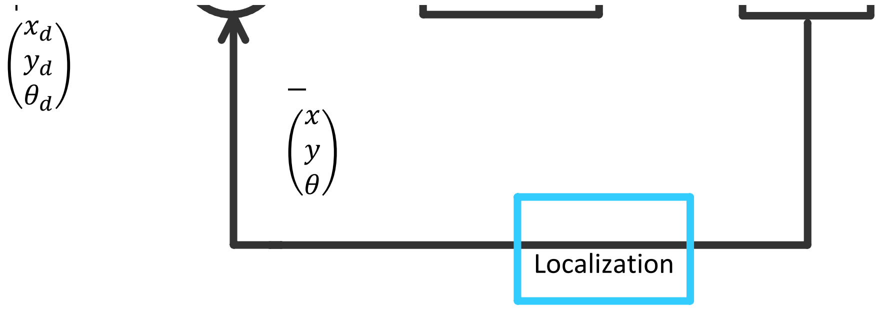


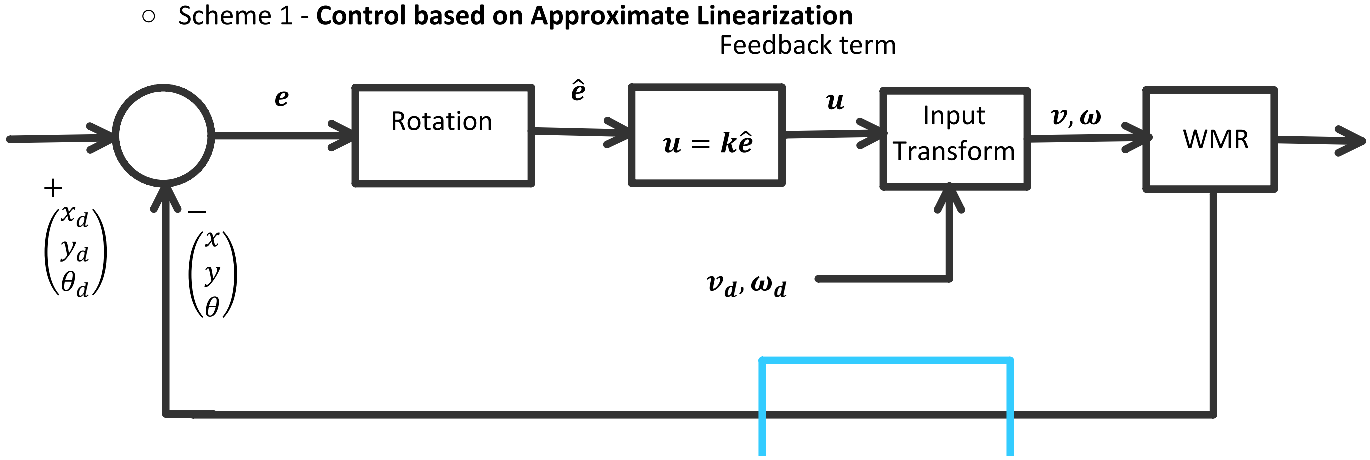
Fig. 11.13. Control problems for a unicycle; *left*: trajectory tracking, *right*: posture regulation

- **Trajectory Tracking**:





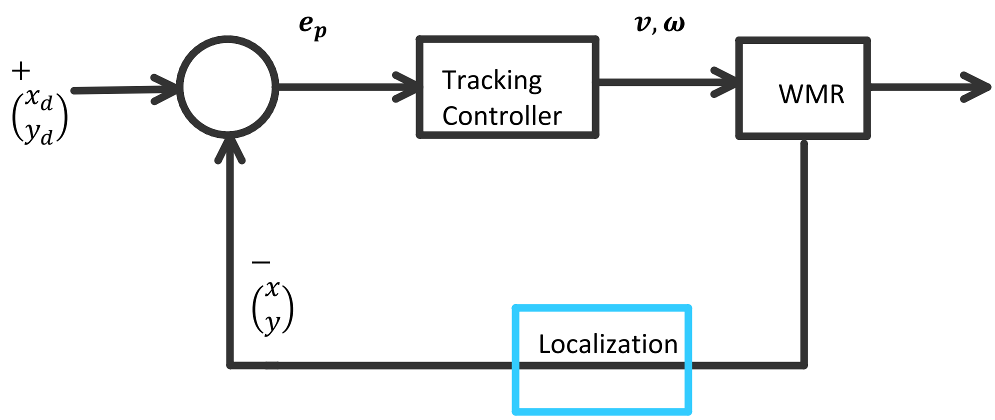
- The desired reference trajectory $(x_d(t), y_d(t))$ must be **admissible**: there must exist v_d and ω_d such that:
  - $\dot{x}_d = v_d \cos\theta_d$
  - $\dot{y}_d = v_d \sin\theta_d$
  - $\dot{\theta}_d = \omega_d$
- Thanks to flatness, we can compute:
  - $\theta_d(t) = ATAN2(\dot{y}_d(t), \dot{x}_d(t)) + k\pi; k = 0,1$
  - $v_d(t) = \pm\sqrt{\dot{x}_d^2(t) + \dot{y}_d^2(t)}$
  - $\omega_d(t) = \dfrac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{\dot{x}_d^2(t) + \dot{y}_d^2(t)}$
- Instead of using the state error $q_d - q$ we use a rotated version:
  - $e = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{pmatrix}$
  - Expanding the equations and using the kinematic model of the unicicle we get:
    - $\dot{e}_1 = v_d \cos e_3 - v + e_2\omega$
    - $\dot{e}_2 = v_d \sin e_3 - e_1\omega$
    - $\dot{e}_3 = \omega_d - \omega$
- Using the input transformation.
  - $v = v_d \cos e_3 - u_1$
  - $\omega = \omega_d - u_2$
- The calculation is simplified and we get:
  - $\dot{e} = \begin{bmatrix} 0 & \omega_d & 0 \\ -\omega_d & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} 0 \\ \sin e_3 \\ 0 \end{bmatrix} v_d + \begin{bmatrix} 1 & -e_2 \\ 0 & e_1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$
- We get a **non-linear** dynamics for the evolution of the error. The first term is linear, while the second and third are nonlinear. Moreover, the first and second terms are in general time-varying because the reference inputs $v_d(t)$ and $\omega_d(t)$ change over time.
- Scheme 1 - **Control based on Approximate Linearization**



- The simplest approach is to linearize the error dynamics around the reference trajectory. $e = 0$
  - $\lim_{e \to 0} \dot{e} = \begin{bmatrix} 0 & \omega_d & 0 \\ -\omega_d & 0 & v_d \\ 0 & 0 & 0 \end{bmatrix} e + \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$
- Now we use the linear feedback:
  - $u_1 = -k_1 e_1$
  - $u_2 = -k_1 e_2 - k_3 e_3$
- That leads to the closed-loop linear dynamics:
  - $\dot{e} = A(t)e = \begin{bmatrix} -k_1 & \omega_d & 0 \\ -\omega_d & 0 & v_d \\ 0 & -k_2 & -k_3 \end{bmatrix}$
- The characteristic polynomial of A is:
  - $p(\lambda) = \lambda(\lambda + k_1)(\lambda + k_3) + \omega_d^2(\lambda + k_3) + v_d k_2(\lambda + k_1)$
- Choosing the gains as:
  - $k_1 = k_3 = 2\zeta a$
  - $k_2 = \dfrac{a^2 - \omega_d^2}{v_d}$
  - With $\zeta \in (0,1)$ and $a > 0$
- We get:
  - $p(\lambda) = (\lambda + 2\zeta a)(\lambda^2 + 2\zeta a\lambda + a^2)$
  - Which is characterized by three constant eigenvalues: one real negative eigenvalue in $-2\zeta a$ and a pair of complex eigenvalues with negative real part.
- However, the system is still time-varying and there is no guarantee that it is asymptotically stable. Except when v_d and ω_d are constant as is the case for circular and rectilinear trajectories.
- This guarantees **local stability** but **not global asymptotic stability**.
- When the error goes to zero then the inputs to the robot become the nominal input to the target vehicle (a unicycle following the desired trajectory).
- It is possible to design a non-linear feedback that guarantees global stability.
- This approach can only be used in **persistent** trajectories, i.e. the velocity cannot be equal to zero at any point. Notice that if $v_d = 0$ a singularity is introduced in the system.
- **NONLINEAR CONTROL**:
  - Considering the error dynamics:
    - $\dot{e}_1 = e_2\omega + u_1$
    - $\dot{e}_2 = v_d \sin e_3 - e_1\omega$
    - $\dot{e}_3 = u_2$
      - With $\omega = \omega_d - u_2$
  - And the following nonlinear version of the control law
    - $u_1 = -k_1(v_d, \omega_d)e_1$
    - $u_2 = -k_2 v_d \dfrac{\sin e_3}{e_3} e_2 - k_3(v_d, \omega_d)e_3$
  - Where k_1 and k_3 are bounded functions with bounded derivatives, and $k_2 > 0$ is constant.
  - If the reference inputs are also bounded and with bounded derivatives, and they do not converge to zero the tracking error **converges to zero**

globally, i.e. for any initial condition.
- Closed loop dynamics:
  - $\dot{e}_1 = e_2\omega - k_1(v_d, \omega_d)e_1$
  - $\dot{e}_2 = v_d \sin e_3 - e_1\omega$
  - $\dot{e}_3 = -k_2 v_d \dfrac{\sin e_3}{e_3} e_2 - k_3(v_d, \omega_d)e_3$
- Proof of Stability in the book

- Scheme 2 - **Input-Output Linearization**:



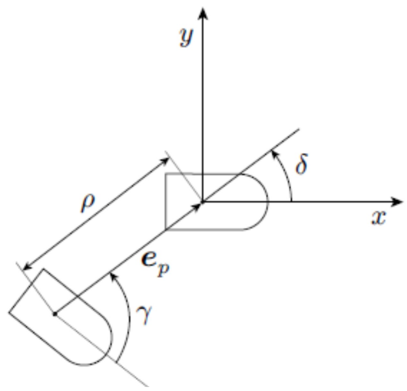
  - The idea is to find an output whose dynamics can be made **linear** via feedback, i.e. with an input transformation.
  - Since the desired trajectory is specified by the flat outputs $(x(t), y(t))$ then tracking the orientation might not be necessary. So we track only the **cartesian error** e_p, not the whole state error. Thanks to flatness the other variables of the state will automatically converge to the desired values.
  - Given a driftless system:
    - $\dot{x} = G(x)u = \sum_{i=1}^{m} g_i(x)u_i$
      - $G(x): n \times m$
    - $y = h(x)$
  - The approach consists in computing the time derivative of the output until the input appears.
    - $\dot{y} = \dot{h}(x) = \dfrac{\partial h}{\partial x}\dot{x} = \dfrac{\partial h}{\partial x}G(x)u = T(x)u$
    - With
      - $T(x) = \dfrac{\partial h}{\partial x}G(x): m \times m$
  - We do an input transformation
    - $v = T^{-1}(x)u$
  - So that the time derivative of the output becomes linear
    - $\dot{y} = T(x)u = T(x)T^{-1}(x)v$
  - From this we can get the outputs with simple integrators
  - Remarks:
    - The matrix $T(x)$ is a decoupling matrix
    - The linearization is **exact,** not an approximation.
  - Example: Unicycle - Slides

- **Regulation**:
  - Posture Regulation via trajectory tracking?

        - Plan a trajectory from an initial configuration q_i to a desired one q_d. The trajectory should obviously stop at the desired configuration q_d.
        - This does not work because controllers based on the error dynamics (as the previous ones) require persistent trajectories.
        - WMRs do not admit universal controllers because they are nonholonomic, so there are no controllers that can stabilize arbitrary trajectories, persistent or not.
        - The root of this problem is **underactuation**, i.e. we have more generalized coordinates than actuators.
    - **Cartesian regulation**: We solve the problem for cartesian position and do not care about orientation.
        - $(x, y) \to (x_d, y_d)$
        - Without loss of generality we define:
            - $x_d = y_d = 0$
        - The error vector becomes:
            - $e_p = \begin{pmatrix} -x \\ -y \end{pmatrix}$
        - The unit vector in the direction of the sagital plane:
            - $n = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$
        - Consider the control law
            - $v = -k_1 e_p^T n = -k_1(x\cos\theta + y\sin\theta)$
            - $\omega = k_2(ATAN2(y, x) - \theta + \pi)$
            - v is proportional to the projection of the error on the unicycle's sagital axis
            - ω is proportional to the **pointing error** (the differrence between the orientation of the error and that of the unicylce.
        - Proof:
            - Consider the Lyapunov-like function:
                - $V = \frac{1}{2}(x^2 + y^2) = \frac{1}{2}e_p^T e_p \geq 0$
                - $V = 0 \Leftrightarrow e_p = 0$
                - **Positive semidefinite** because with it we can only guarantee that the cartesian error goes to zero but we know nothing of the orientation.
            - And its time derivative:
                - $\dot{V} = x\dot{x} + y\dot{y} = v(x\cos\theta + y\sin\theta)$
                - $\dot{V} = -k_1(x\cos\theta + y\sin\theta)^2 \leq 0$
                - Negative semidefinite
            - Barbalat Lemma:
                - If:
                    - $V \geq 0$
                    - $\dot{V} \leq 0$
                    - $\ddot{V}$ is bounded
                - Then:
                    - $\lim_{t\to\infty} \dot{V} = 0$
                - If a function is bounded below, its derivative bounded above, then the function goes to a limit. However this does not mean that its derivative goes to zero unless the second derivative is also bounded.
            - For $\dot{V} = 0$ we get that either
                - $e_p = 0$
                - $e_p \perp n$

- Cannot happen as $t \to \infty$ because the robot will reorient and $\dot{V}$ will increase.
- So, the only option is for the error to converge to zero.

- **Posture Regulation**:
  - To design a controller that is able to regulate the whole configuration vectos it is convenient to convert to polar coordinates
    - $\rho = \sqrt{x^2 + y^2}$
    - $\gamma = ATAN2(y, x) - \theta + \pi$
    - $\delta = \gamma + \theta$
  - This transformation induces a singularity at the origin.



  - Kinematic model:
    - $\dot{\rho} = -v \cos\gamma$
    - $\dot{\gamma} = \dfrac{\sin\gamma}{\rho} v - \omega$
    - $\dot{\delta} = \dfrac{\sin\gamma}{\rho} v$
    - Note that the input vector field associated with v is singular at the origin.
  - Control law:
    - $v = k_1 \rho \gamma$
    - $\omega = k_2 \gamma + k_1 \dfrac{\sin\gamma \cos\gamma}{\gamma}(\gamma + k_3 \delta)$
      - To do this we try to make the derivative of the Lyapunov function negative definite.
    - The driving velocity is exactly the same as the one from the previous controller.
    - The first term of the steering velocity is the same as in the previous case. However, a new term is introduced that depends on the desired final orientation and will bring the orientation error to zero.
  - Proof:
    - Consider the Lyapunov Candidate:
      - $V = \dfrac{1}{2}(\rho^2 + \gamma^2 + k_3 \delta^2) \geq 0$
        - Positive definite
      - $\dot{V} = \rho\dot{\rho} + \gamma\dot{\gamma} + \delta\dot{\delta} \leq 0$
        - Negative semidefinite because the angle δ is not there.
      - Complete
    - Using Barbalat's Lemma again, we need to proof that $\ddot{V}$ is bounded. Can be shown.
    - Finally, this means that:
      - $\dot{V} \to 0$

- However, this only means that $\gamma, \rho \rightarrow 0$. But from the closed loop dynamics of the error we can show that $\delta \rightarrow 0$
- If we use this controller in the original coordinates we get a discontinuous controller because of the singularity at the origin. In fact, due to nonholonomy, posture stabilizers have to be either **discontinuou**s or **time-varying**.

Autonomous and Mobile Robotics

Prof. Giuseppe Oriolo

Wheeled Mobile Robots 4

Motion Control of WMRs: Trajectory Tracking

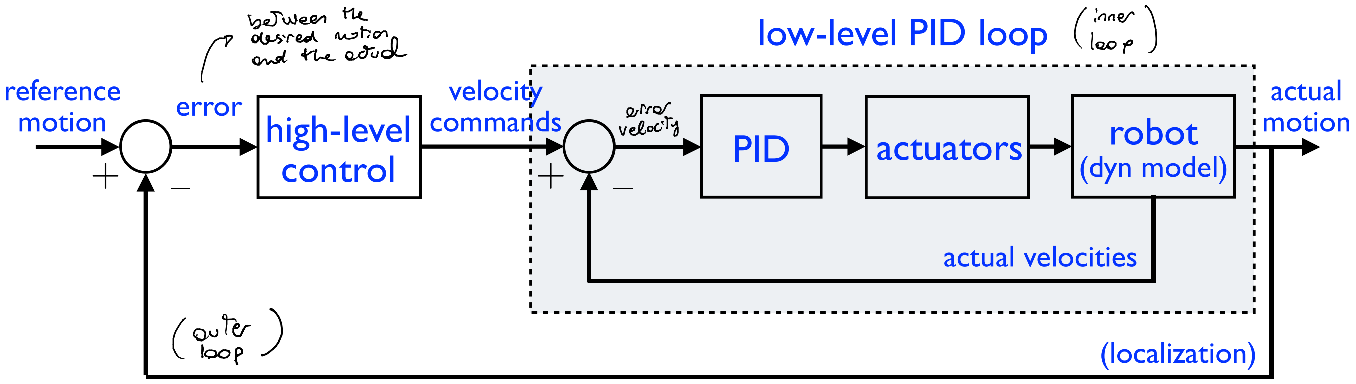
DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

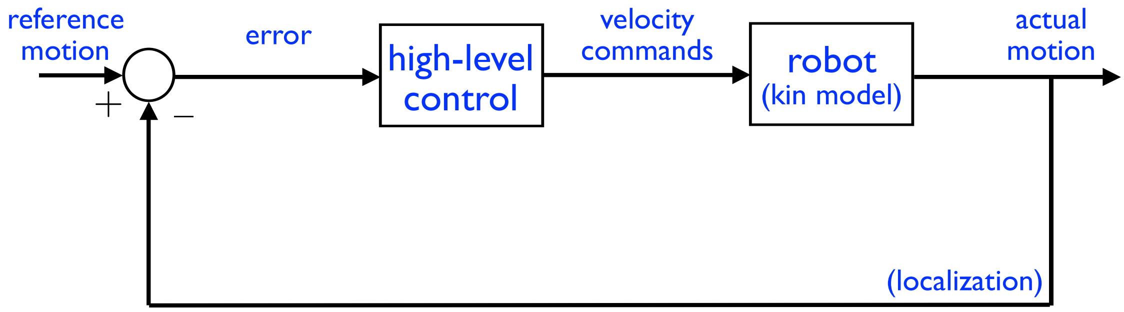
motion control

- a desired motion is assigned for the WMR, and the associated nominal inputs have been computed
- to execute the desired motion, we need feedback control because the application of nominal inputs in open-loop would lead to very poor performance
- dynamic models are generally used in robotics to compute commands at the generalized force level
- kinematic models are used to design WMR feedback laws because (1) dynamic terms can be canceled via feedback (2) wheel actuators are equipped with low-level PID loops that accept velocities as reference

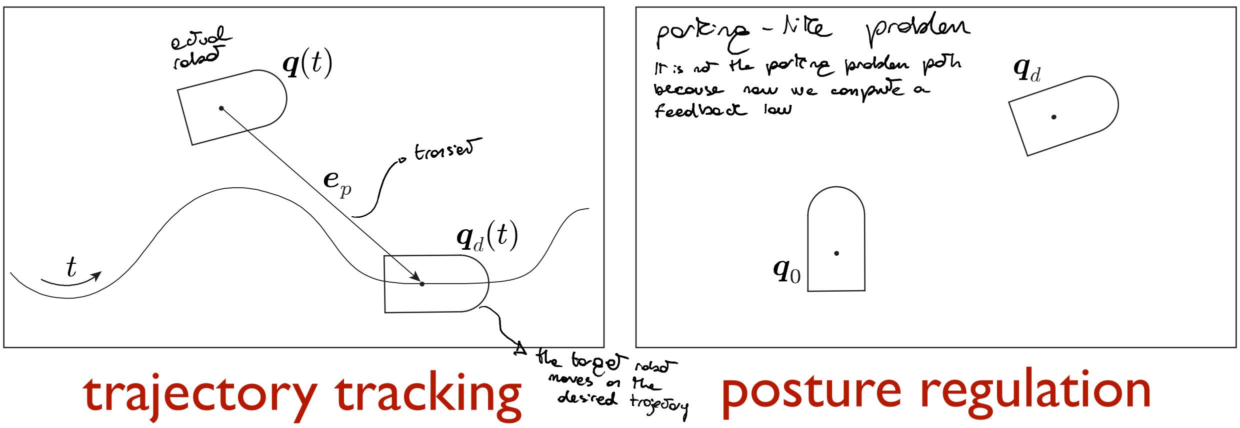
- actual control scheme



- equivalent control scheme (for design)



motion control problems



trajectory tracking (predictable transients)

posture regulation (no prior planning)

- w.l.o.g. we consider a unicycle in the following

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

the bicycle is the same but with one equation more

trajectory tracking: state error feedback

- the unicycle must track a Cartesian desired trajectory $(x_d(t), y_d(t))$ that is admissible, i.e., there exist v_d and ω_d such that

Output cartesian trajectory $\begin{pmatrix} x \\ y \end{pmatrix} \to \begin{pmatrix} x_d(t) \\ y_d(t) \end{pmatrix}$

$$\left.\begin{aligned} \dot{x}_d &= v_d \cos\theta_d \\ \dot{y}_d &= v_d \sin\theta_d \\ \dot{\theta}_d &= \omega_d \end{aligned}\right\}$$

unicycle we want to track → desired trajectory

Here is a couple of outputs from which we can reconstruct the state and the inputs

- thanks to flatness, from $(x_d(t), y_d(t))$ we can compute

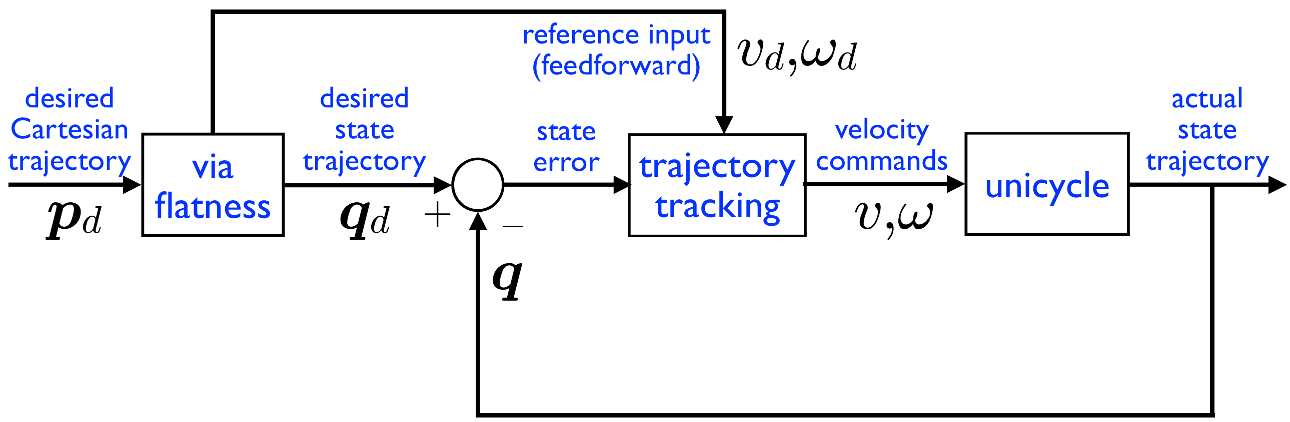
$$\theta_d(t) = \mathrm{Atan2}\left(\dot{y}_d(t), \dot{x}_d(t)\right) + k\pi \qquad k = 0, 1$$

$$v_d(t) = \pm\sqrt{\dot{x}_d^2(t) + \dot{y}_d^2(t)}$$

$$\omega_d(t) = \frac{\ddot{y}_d(t)\dot{x}_d(t) - \ddot{x}_d(t)\dot{y}_d(t)}{\dot{x}_d^2(t) + \dot{y}_d^2(t)}$$

- the desired state trajectory can be used to compute the state error, from which the feedback action is generated; whereas the nominal input can be used as a feedforward term

- the resulting block scheme is



- rather than using directly the state error $\boldsymbol{q}_d - \boldsymbol{q}$, use its rotated version defined as

desired state — actual state

$$\boldsymbol{e} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_d - x \\ y_d - y \\ \theta_d - \theta \end{pmatrix}$$

x, y errors are rotated

equal to θ error

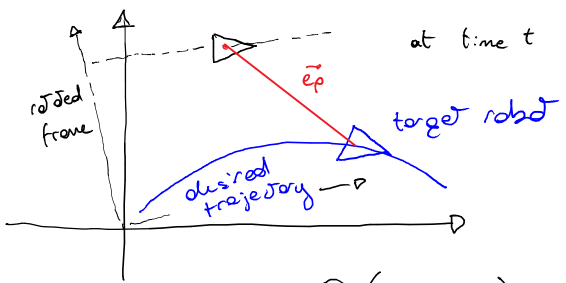
(e_1, e_2) is $\boldsymbol{e}_p$ (previous figure) in a frame rotated by θ

- the error dynamics is nonlinear and time-varying

additional complication v_d & ω_d change with time

$$\dot{e}_1 = v_d \cos e_3 - v + e_2\,\omega$$
$$\dot{e}_2 = v_d \sin e_3 - e_1\,\omega$$
$$\dot{e}_3 = \omega_d - \omega$$

Dynamics of the error



$$e_1 = \cos\theta\,(x_d - x) + \sin\theta\,(y_d - y)$$

error dynamics:

$$\dot{e}_1 = \cos\theta\,(\dot{x}_d - \dot{x}) - \sin\theta\,\dot{\theta}\,(x_d - x) + \sin\theta\,(\dot{y}_d - \dot{y}) + \cos\theta\,\dot{\theta}\,(y_d - y)$$
$$= \cos\theta\,(v_d\cos\theta_d - v\cos\theta) + \sin\theta\,(v_d\sin\theta_d - v\sin\theta) +$$
$$+ \underbrace{\dot{\theta}}_{\omega}\,\underbrace{(-\sin\theta\,(x_d - x) + \cos\theta\,(y_d - y))}_{e_2}$$

$$\dot{e}_1 = v_d\cos\theta\cos\theta_d + v_d\sin\theta\sin\theta_d - v\cos^2\theta - v\sin^2\theta + \omega e_2$$
$$= v_d\cos(\underbrace{\theta_d - \theta}_{e_3}) - v + \omega e_2$$
$$= \boxed{\omega e_2 + v_d\cos e_3 - v}$$

$\dot{e}_2$ = same procedure (no need to compute e_3)

via approximate linearization

- a simple approach for stabilizing the error dynamics is to use its linearization around the reference trajectory (indirect Lyapunov method ⇒ local results)
- to make the reference trajectory an unforced equilibrium for the error dynamics

$$\dot{e}_1 = v_d \cos e_3 - v + e_2\,\omega$$
$$\dot{e}_2 = v_d \sin e_3 - e_1\,\omega$$
$$\dot{e}_3 = \omega_d - \omega$$

new inputs

use the following (invertible) input transformation

$$u_1 = v_d \cos e_3 - v$$
$$u_2 = \omega_d - \omega$$

Brush-up approximate linearization

$\dot{x} = \varphi(x, v)$ is a NL system

assign desired equilibrium (w l o g)

$\dot{x}\big|_{x=0,\,v=0} = 0$ i.e. $\varphi(0,0) = 0$ UNFORCED EQUILIBRIUM

Mc Laurin expansion:

$$\varphi(x,v) = \underbrace{\varphi(0,0)}_{=0} + \underbrace{\frac{\partial \varphi}{\partial x}\bigg|_{\substack{x=0\\v=0}}}_{A}\underbrace{(x-0)}_{\text{variation}} + \underbrace{\frac{\partial \varphi}{\partial v}\bigg|_{\substack{x=0\\v=0}}}_{B}(v-0) + \cancel{\dots} \text{ second order dynamics}$$

$\dot{x} = Ax + Bv$ → linear approximation → Local stability is guaranteed

- we obtain

$$\dot{e}_1 = \omega_d\, e_2 + u_1 - e_2\, u_2$$
$$\dot{e}_2 = -\omega_d\, e_1 + v_d \sin e_3 + e_1\, u_2$$
$$\dot{e}_3 = u_2$$

that is

$$\dot{\mathbf{e}} = \underbrace{\begin{pmatrix} \omega_d\, e_2 \\ -\omega_d\, e_1 + v_d \sin e_3 \\ 0 \end{pmatrix}}_{\boldsymbol{f}(\mathbf{e},t)} + \underbrace{\begin{pmatrix} 1 & -e_2 \\ 0 & e_1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}}_{\boldsymbol{G}(\mathbf{e})\boldsymbol{u}}$$

drift term
nonlinear, time-varying

input term
nonlinear in $\boldsymbol{e}$, linear in $\boldsymbol{u}$

Linearization applying the general approach

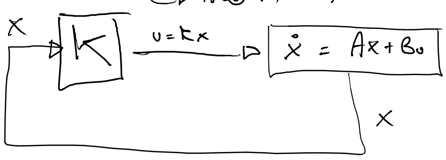
$\varphi(e,u) = f(e,t) + G(e)u$

$$A = \left.\frac{\partial f}{\partial e}\right|_{\substack{e=0\\u=0}} + \underbrace{\left.\frac{\partial (Gu)}{\partial e}\right|_{\substack{e=0\\u=0}}}_{\to 0} = \left.\frac{\partial f}{\partial e}\right|_{\substack{e=0\\u=0}} = \left.\begin{pmatrix} 0 & \omega_d & 0 \\ -\omega_d & 0 & v_d \cos e_3 \\ 0 & 0 & 0 \end{pmatrix}\right|_{\substack{e=0\\u=0}}$$

$$B = \underbrace{\left.\frac{\partial f}{\partial u}\right|_{\substack{e=0\\u=0}}}_{\to 0} + \left.\frac{\partial Gu}{\partial u}\right|_{\substack{e=0\\u=0}} = G(e)\Big|_{\substack{e=0\\u=0}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}$$

Stabilization with feedback

→ matrix K



$\dot{x} = Ax + BKx = (A+BK)$ it must have $Re(\lambda)<0$

The problem is that we have two inputs and not only a scalar one

the input proposed is : $u = Ke = \begin{pmatrix} -k_1 & 0 & 0 \\ 0 & -k_2 & -k_3 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$

- hence, the linearization of the error dynamics around the reference trajectory is easily computed as

$$\dot{\boldsymbol{e}} = \begin{pmatrix} 0 & \omega_d & 0 \\ -\omega_d & 0 & v_d \\ 0 & 0 & 0 \end{pmatrix} \boldsymbol{e} + \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

- define the linear feedback

$$\boldsymbol{u} = \boldsymbol{K}\boldsymbol{e} = \begin{pmatrix} -k_1 & 0 & 0 \\ 0 & -k_2 & -k_3 \end{pmatrix} \begin{pmatrix} e_1 \\ e_2 \\ e_3 \end{pmatrix}$$

- the closed-loop error dynamics is still time-varying!

$$\dot{\boldsymbol{e}} = \boldsymbol{A}(t)\,\boldsymbol{e} = \begin{pmatrix} -k_1 & \omega_d & 0 \\ -\omega_d & 0 & v_d \\ 0 & -k_2 & -k_3 \end{pmatrix} \boldsymbol{e}$$

$\to A+BK$ we need it Hurwitz

$\dot{e}_1 = \omega_d e_2 - k_1 e_1$
$\dot{e}_2 = -\omega_d e_1 + v_d e_3$
$\dot{e}_3 = -k_2 e_2 - k_3 e_3$

closed loop error dynamics

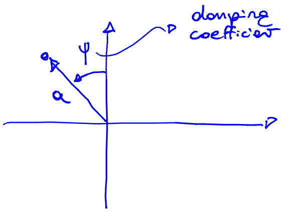
Is $A(t)$ Hurwitz?

$$\lambda I - A = \begin{pmatrix} \lambda + k_1 & -\omega_d & 0 \\ -\omega_d & \lambda & -v_d \\ 0 & +k_2 & \lambda - k_3 \end{pmatrix}$$

suppose $k = k_1 = k_3$

$$\det(\lambda I - A) = \lambda(\lambda+k_1)(\lambda+k_3) + k_2 v_d(\lambda + k_1) + \omega_d^2(\lambda + k_3)$$
$$= (\lambda + k)\underbrace{(\lambda^2 + k\lambda + k_2 v_d + \omega_d^2)}_{\to\ \lambda^2 + 2a\xi\lambda + a^2}$$

$k_2 v_d + \omega_d^2 = a^2$
$k = 2a\xi$



damping coefficient $\sin\psi = \xi$

eigenvalues:

1. $-k$ (real)
2. gains: $k = 2a\xi$ $(k_1 = k_3 = k)$
   $k_2 = \dfrac{a^2 - \omega_d^2}{v_d}$

need $k > 0 \Rightarrow k = 2a\xi$ a should be >0 too
need $a < \xi < 1$
take $a > 0$, $a < \xi < 1$

- letting

$$k_1 = k_3 = 2\zeta a \qquad k_2 = \frac{a^2 - \omega_d^2}{v_d}$$

problem: we cannot use this controller when v_d is 0 (stops are not valid)

with $a>0$, $\zeta\in(0,1)$, the characteristic polynomial of $\boldsymbol{A}(t)$ becomes time-invariant and Hurwitz

$$p(\lambda) = \underbrace{(\lambda + 2\zeta a)}_{\substack{\text{real} \\ \text{negative} \\ \text{eigenvalue}}}\underbrace{(\lambda^2 + 2\zeta a\lambda + a^2)}_{\substack{\text{pair of complex} \\ \text{eigenvalues with} \\ \text{negative real part}}}$$

- caveat: this does not guarantee asymptotic stability, unless v_d and ω_d are constant (rectilinear and circular trajectories); even in this case, asymptotic stability of the unicycle is not global (indirect Lyapunov method)

Comments

- the actual velocity inputs v,ω are obtained plugging the feedbacks u_1, u_2 in the input transformation

- note: $(v,\omega) \to (v_d,\omega_d)$ as $\boldsymbol{e} \to \boldsymbol{0}$ (pure feedforward)

- note: $k_2 \to \infty$ as $v_d \to 0$, hence this controller can only be used with persistent Cartesian trajectories (stops are not allowed)

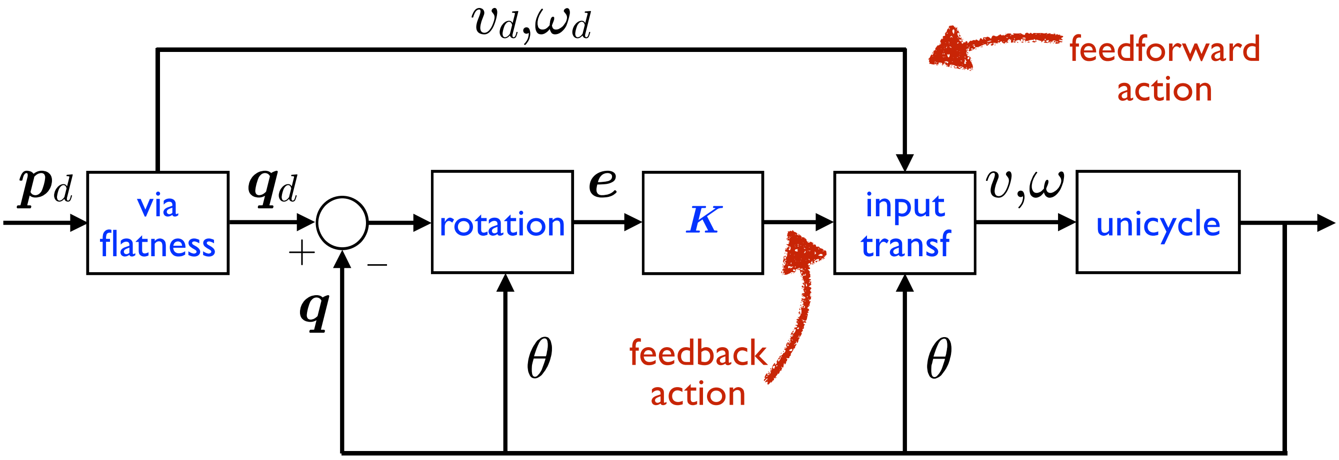
- global stability is guaranteed by a nonlinear version

$$u_1 = -k_1(v_d, \omega_d)\, e_1 \qquad k_2(v_d, e_3)$$

$$u_2 = -k_2\, v_d \frac{\sin e_3}{e_3} e_2 - k_3(v_d, \omega_d)\, e_3$$

if k_1,k_3 bounded, positive, with bounded derivatives

- the final block scheme for trajectory tracking via state error feedback and approximate linearization is

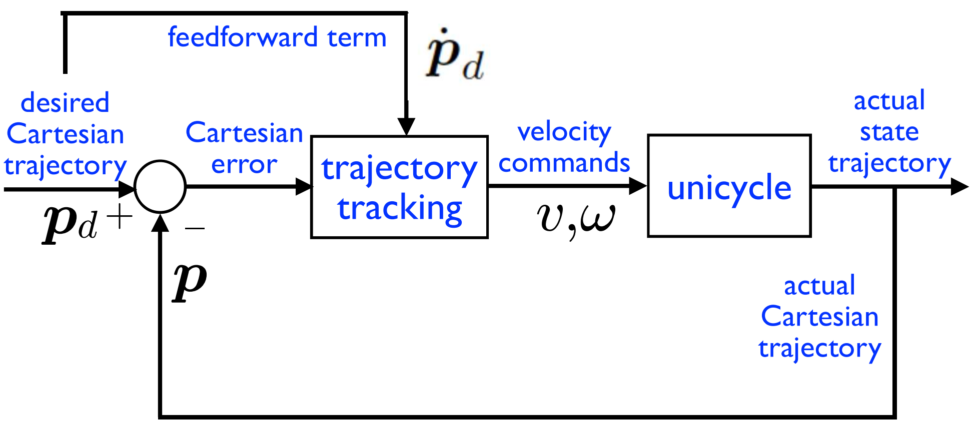


- based on state error
- needs v_d, ω_d
- needs θ also for error rotation + input transformation

trajectory tracking: output error feedback

- another approach: develop the feedback action from the output (Cartesian) error only, without computing a desired state trajectory, while the feedforward term is the velocity along the reference trajectory

- the resulting block scheme is



via exact input/output linearization

- idea: (1) if the map between the available inputs and some derivative of the output is invertible, then (2) by inverting this map the system can be made linear

- however, for the unicycle the map between the velocity inputs and the Cartesian output is singular

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

decoupling matrix is singular

as a consequence, input-output linearization is not possible in this case

Brush-up: input-output linearization

in $\mathbb{R}^n$

$$\begin{cases} \dot{x} = G(x)\,u & \text{Non-Linear driftless system (as kinematic models of WMRs)} \\ y = h(x) & \text{output} \end{cases}$$

We want $y \to y_d$ desired output trajectory

Map:

$$\dot{y} = \frac{\partial h}{\partial x}\dot{x} = \frac{\partial h}{\partial x}G(x)\,u$$

$\frac{\partial h}{\partial x}$: $m \times n$ (states); $G(x)$: $n \times m$ (inputs)

$\frac{\partial h}{\partial x}G(x)$ → Decoupling matrix $T(x)$ $m \times m$

If $T(x)$ is invertible set $u = T^{-1}(x)\,v$

u: original input; v: new input

$$\dot{y} = T(x)\,u = \underbrace{T(x)\,T^{-1}(x)}_{I}\,v = v \qquad \text{I-O map is } \underline{\text{linear}}$$

$$v_1 \to \boxed{\int} \to y_1$$
$$\vdots$$
$$v_m \to \boxed{\int} \to y_m$$

$\boxed{\dot{y} = v}$ ⇝ need to stabilize this to $y_d(t)$

stabilizing a single integrator:

$\dot{x} = u$, $x_d(t)$, $e = x_d - x$

$$\dot{e} = \dot{x}_d - \dot{x} = \dot{x}_d - u \underset{\uparrow}{=} -ke \implies \boxed{u = \dot{x}_d + ke}$$

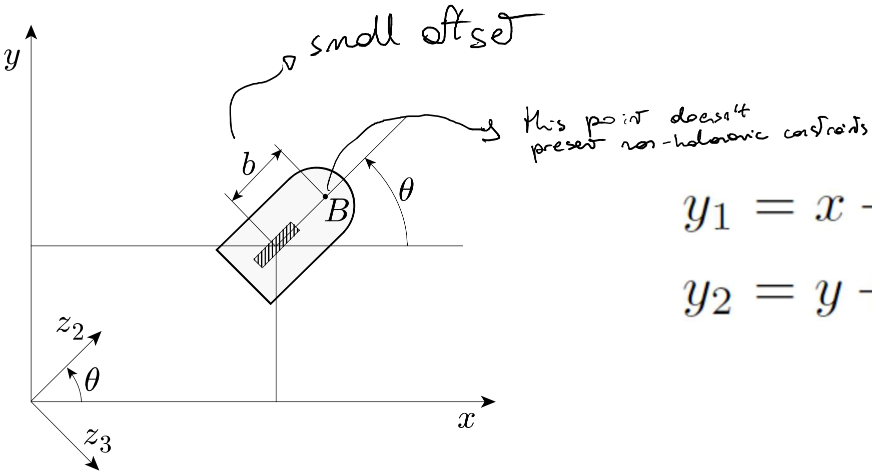
exponential stability

$\dot{x}_d$: feedforward; ke: feedback

Control law stabilizing a single integrator

- solution: change slightly the output so that the new input-output map is invertible and exact linearization becomes possible

- displace the output from the contact point of the wheel to point B along the sagittal axis



$$
\begin{aligned}
y_1 &= x + b\cos\theta \\
y_2 &= y + b\sin\theta
\end{aligned}
$$

- differentiating wrt time

$$
\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \underbrace{\begin{pmatrix} \cos\theta & -b\sin\theta \\ \sin\theta & b\cos\theta \end{pmatrix}}_{\text{determinant} = b} \begin{pmatrix} v \\ \omega \end{pmatrix} = \boldsymbol{T}(\theta) \begin{pmatrix} v \\ \omega \end{pmatrix}
$$

- if $b \neq 0$, we may set

$$
\begin{pmatrix} v \\ \omega \end{pmatrix} = \boldsymbol{T}^{-1}(\theta) \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta/b & \cos\theta/b \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}
$$

obtaining

$$
\begin{aligned}
\dot{y}_1 &= u_1 \\
\dot{y}_2 &= u_2 \\
\dot{\theta} &= \frac{u_2\cos\theta - u_1\sin\theta}{b}
\end{aligned}
$$

- achieve global exponential convergence of y_1, y_2 to the desired trajectory letting

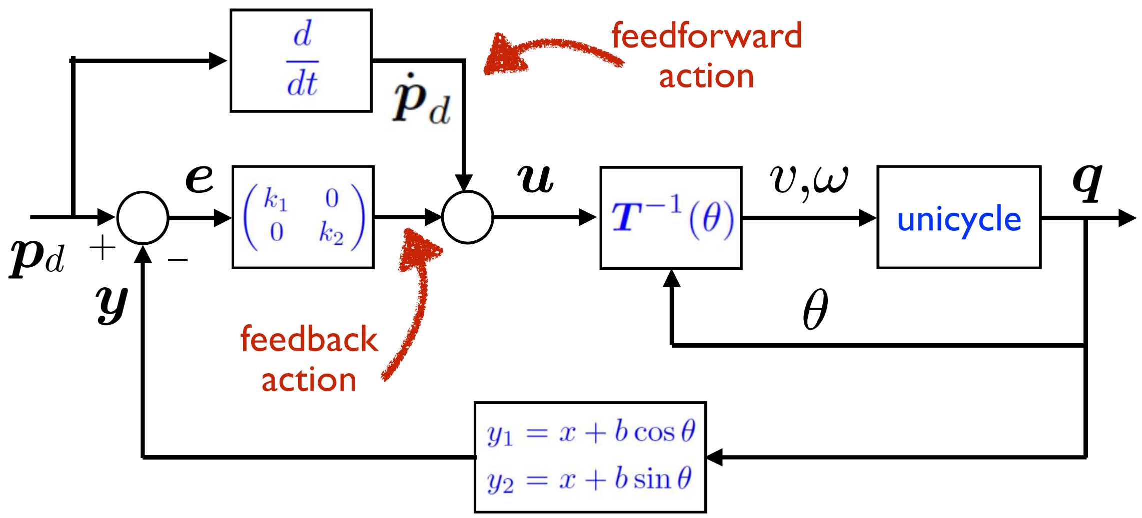
$$u_1 = \dot{y}_{1d} + k_1(y_{1d} - y_1)$$
$$u_2 = \dot{y}_{2d} + k_2(y_{2d} - y_2)$$

with $k_1, k_2 > 0$

- θ is not controlled with this scheme, which is based on output error feedback (compare with the previous)

- the desired trajectory for B can be arbitrary; in particular, square corners may be included

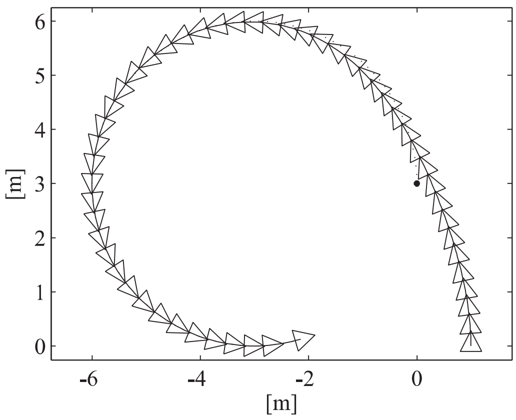
- the final block scheme for trajectory tracking via output error feedback + input-output linearization is

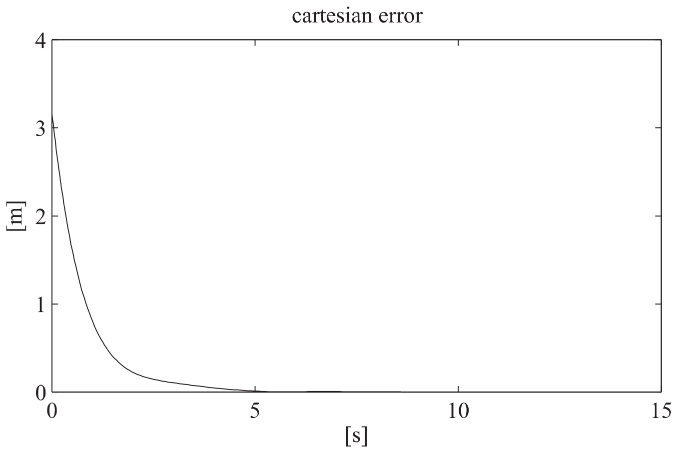


- based on output error
- needs $\dot{\boldsymbol{p}}_d$
- needs x,y,θ for output reconstruction and θ also for input transformation

simulations

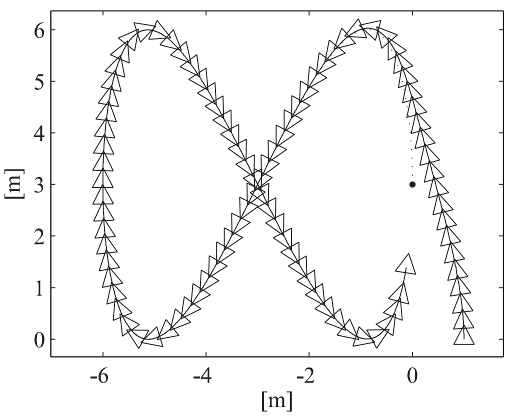
tracking a circle via approximate linearization

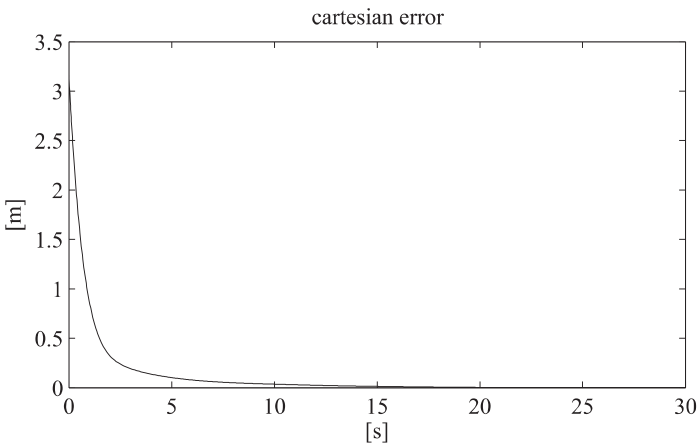




simulations

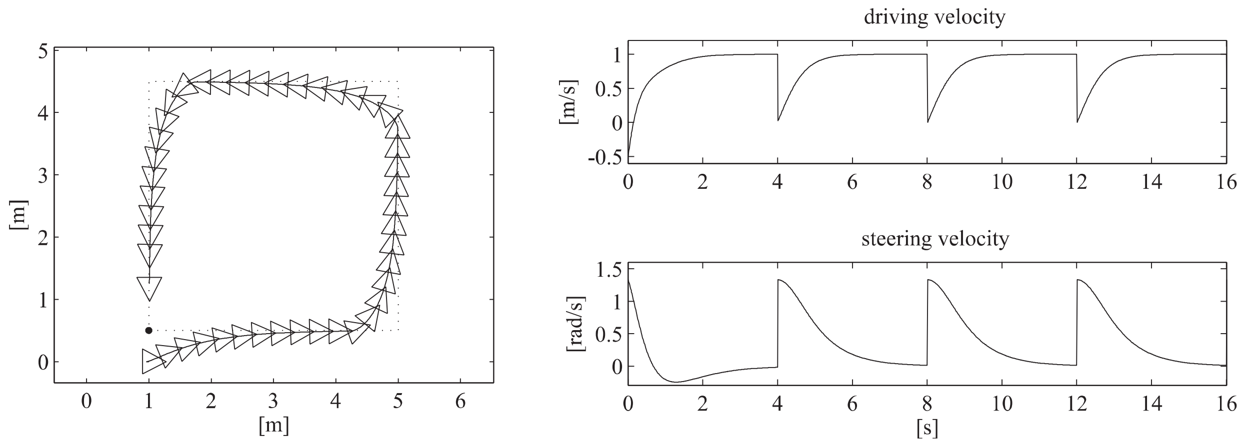
tracking an 8-figure via nonlinear feedback





simulations

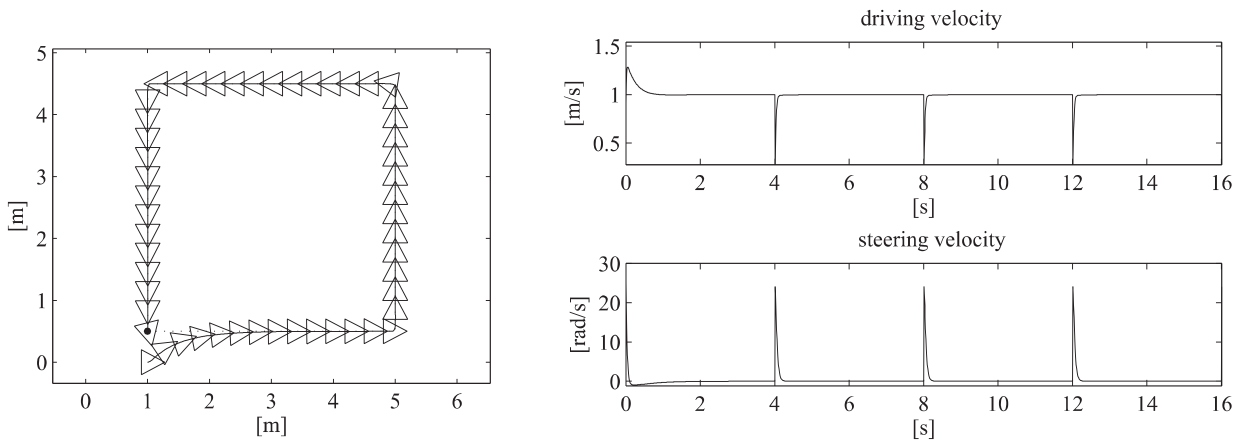
tracking a square via i/o linearization



b=0.75 $\Rightarrow$ the unicycle rounds the corners

simulations

tracking a square via i/o linearization



b=0.2 $\Rightarrow$ accurate tracking but velocities increase

Autonomous and Mobile Robotics

Prof. Giuseppe Oriolo

Wheeled Mobile Robots 5

Motion Control of WMRs: Regulation

DIPARTIMENTO DI INGEGNERIA INFORMATICA
AUTOMATICA E GESTIONALE ANTONIO RUBERTI

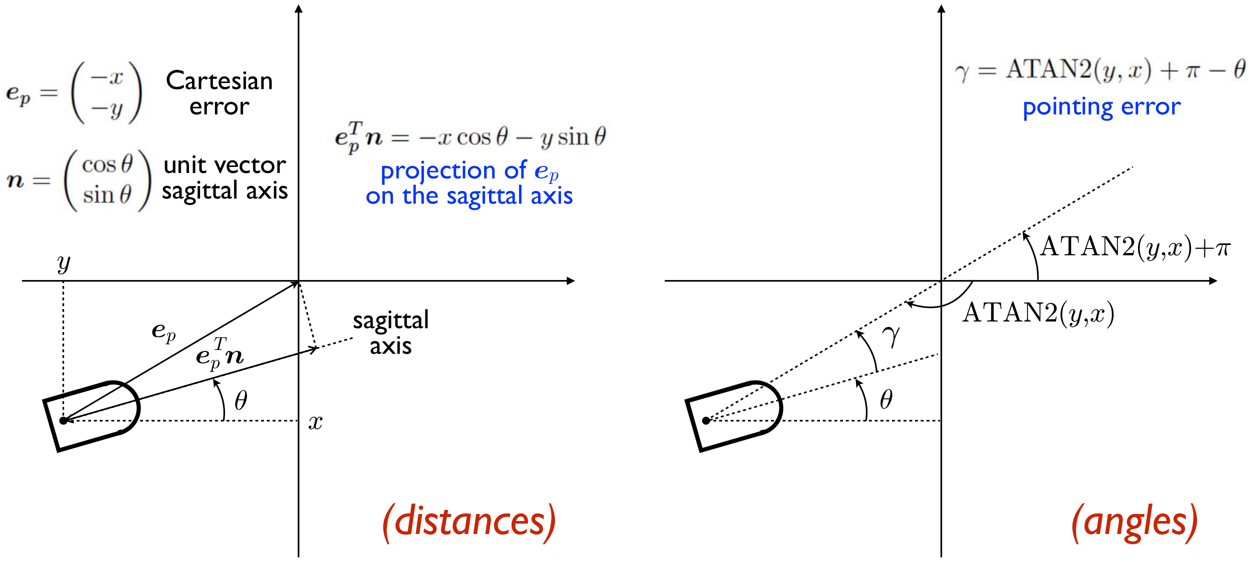
SAPIENZA
UNIVERSITÀ DI ROMA

regulation

- drive the unicycle to a desired configuration $\boldsymbol{q}_d$
- the obvious approach (choose a path/trajectory that stops in $\boldsymbol{q}_d$, then track it via feedback) does not work:
  - linear/nonlinear controllers based on the error dynamics require persistent trajectories
  - i/o linearization leads point B to the destination rather than the representative point of the unicycle
- being nonholonomic, WMRs (unlike manipulators) do not admit universal controllers, i.e., controllers that can stabilize arbitrary trajectories, persistent or not

cartesian regulation

- drive the unicycle to a given cartesian position (w.l.o.g., the origin $(0\ 0)^T$), regardless of orientation
- geometry:



cartesian regulation

- consider the feedback control law

$$v = -k_1(x\cos\theta + y\sin\theta)$$
$$\omega = k_2(\mathrm{Atan2}(y,x) - \theta + \pi)$$

- geometrical interpretation:
  - v is proportional to the orthogonal projection of the Cartesian error e_p on the sagittal axis
  - ω is proportional to the pointing error (i.e., the difference between the orientation of e_p and that of the unicycle)

- Lyapunov-like function

$$V = \frac{1}{2}(x^2 + y^2) \qquad \text{positive semidefinite}$$

$$\dot{V} = -k_1(x\cos\theta + y\sin\theta)^2 \qquad \text{negative semidefinite}$$

- cannot use LaSalle theorem, but Barbalat lemma implies that $\dot{V}$ tends to zero, i.e.

$$\lim_{t\to\infty}(x\cos\theta + y\sin\theta) = 0$$

- under the proposed controller, this implies that the cartesian error goes to zero

$V = \frac{1}{2}(x^2+y^2)$ → positive semidef.

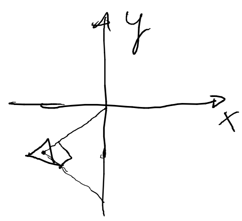
unicycle equations

$\dot{V} = \frac{1}{2}(2x\dot{x} + 2y\dot{y}) = \begin{cases} \dot{x} = v\cos\theta \\ \dot{y} = v\sin\theta \end{cases}$

$= v(x\cos\theta + y\sin\theta)$

now use the control law → $\dot{V} = -k_1(x\cos\theta + y\sin\theta)^2 \le 0$ → neg. semi def.

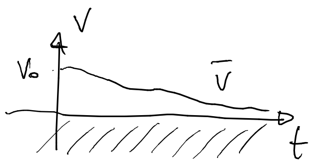
→ 0 when $\begin{pmatrix} x \\ y \end{pmatrix} \perp \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$



Barbalat Lemma:

$V(t) \ge 0$

$\dot{V}(t) \le 0$



then $\lim_{t\to\infty} V(t) = \bar{V}$ ($\bar{V}$ exists)

does this mean that $\dot{V} \to 0$? NOT NECESSARILY

if $V(t) \ge 0$,

$\dot{V}(t) \le 0$

and $\dot{V}(t)$ is bounded,

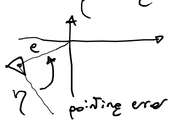
then $\lim_{t\to\infty} \dot{V}(t) = 0$

$\dot{V} \to 0$ means

1. $x, y \to 0$ at Cartesian regulation

~~2.~~ error $e \perp \eta$ sagittal axis

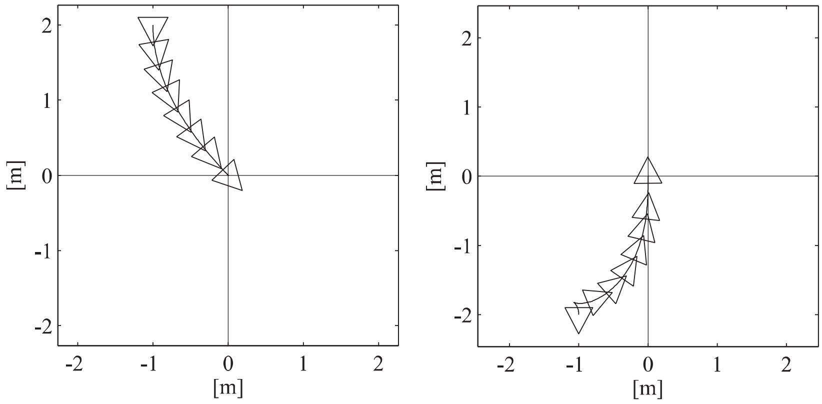
only possibility



is this a possible steady state?

$v = 0$, $\omega \neq 0$ ⇒ NOT A STEADY STATE

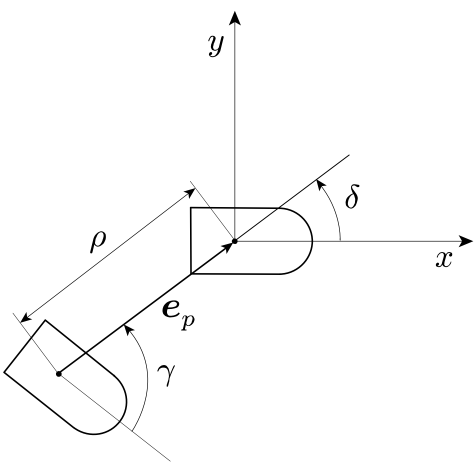
simulation



- final orientation is not controlled
- at most one backup maneuver

posture regulation

- drive the unicycle to a given configuration (w.l.o.g., the origin $(0\ 0\ 0)^T$)
- convert to polar coordinates



$$\rho = \sqrt{x^2 + y^2}$$
$$\gamma = \text{Atan2}(y, x) - \theta + \pi$$
$$\delta = \gamma + \theta$$

- kinematic model in polar coordinates

$$\dot\rho = -v\cos\gamma$$
$$\dot\gamma = \frac{\sin\gamma}{\rho}\,v - \omega$$
$$\dot\delta = \frac{\sin\gamma}{\rho}\,v$$

note the singularity at the origin

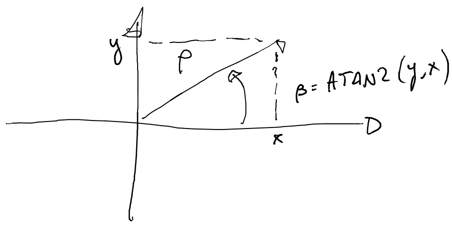
- consider the control law (compare with previous)

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

new term

$$\dot\rho = \frac{1}{2\rho}\left(2x\dot x + 2y\dot y\right) \qquad \rho = \sqrt{x^2+y^2}$$

$$= v\left(\frac{x}{\rho}\cos\theta - \frac{y}{\rho}\sin\theta\right)$$



$$\frac{x}{\rho} = \cos\beta \qquad \frac{y}{\rho} = \sin\beta$$

$$= v\cos(\beta - \theta) = -v\cos(\beta - \theta + \pi) = -v\cos\big(\underbrace{\mathrm{ATAN2}(y,x) - \theta + \pi}_{\gamma}\big)$$

$V = \frac{1}{2}(\rho^2 + \gamma^2 + \delta^2) > 0$ (assume $k_3 = 1$)

$\dot V = \rho\dot\rho + \gamma\dot\gamma + \delta\dot\delta = -\rho v\cos\gamma + v\gamma\frac{\sin\gamma}{\rho} - \gamma\omega + \delta v\frac{\sin\gamma}{\rho}$

$= v\left(-\rho\cos\gamma + \frac{\gamma\sin\gamma}{\rho} + \delta\frac{\sin\gamma}{\rho}\right) - \gamma\omega$

???

$v = k_1\rho\cos\gamma$

$\omega = +k_2\gamma + \alpha$ unknown quantity

$\dot V = -k_1\rho^2\cos^2\gamma - k_2\gamma^2 - \alpha\gamma + \frac{v}{\rho}\sin\gamma(\delta+\gamma)$

zero if

$\alpha = \frac{v}{\rho}\cdot\frac{\sin\gamma}{\gamma}(\delta+\gamma) = k_1\frac{\sin\gamma\cos\gamma}{\gamma}(\delta+\gamma)$

Δ $\boxed{\dot V = -k_1\rho^2\cos^2\gamma - k_2\gamma^2}$ neg. semidef. (δ does not appear)

- Lyapunov candidate

$$V = \frac{1}{2}\left(\rho^2 + \gamma^2 + k_3\,\delta^2\right) \qquad \text{positive definite}$$

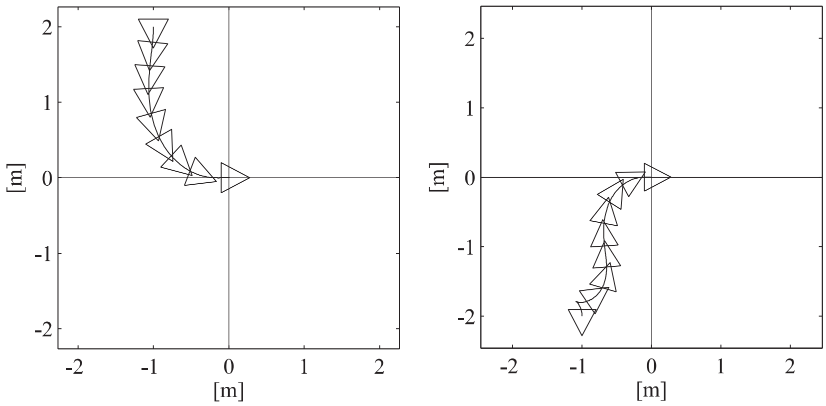
$$\dot V = -k_1\cos^2\gamma\,\rho^2 - k_2\,\gamma^2 \qquad \text{negative semidefinite}$$

- Barbalat lemma implies that ρ,γ and δ go to zero

- the above control law, once mapped back to the original coordinates, is discontinuous at the origin

- it can be shown that, due to the nonholonomy, all posture stabilizers must be discontinuous w.r.t. the state or time-varying (Brockett theorem)

simulation



- final orientation is zeroed as well
- at most one backup maneuver