Marine Control Systems II

Lecture 7: Maneuvering control design

Roger Skjetne

Department of Marine Technology
Norwegian University of Science and Technology

TMR4243

Goals of lecture

- Be able to formulate a control problem as a *maneuvering problem*.
  - Specifying the *Geometric task* and the *Dynamic task*, individually.
- Perform an example maneuvering control design.
- Understand different ways of parameterizing the path, in particular implicit vs. explicit parameterization..
- Parameterize a continuously differentiable path from waypoints.
- Understand the state feedback mechanism in the guidance law, the Gradient update law, that chooses a favorable location of the desired state along the *state path* based on present state of the plant.
  - Exemplified by the double integrator stabilizing the unit circle.
  - Understand that this performance is achieved by the concepts of "*near forward invariance*" and "*near set stability*" of the path – seen as a set to stabilize.
- Be able to formulate a *generalized maneuvering problem*.

Literature

Maneuvering control design:

- Skjetne (2005).
  - Chapters: 1.1, 1.3-1.4, 2, 3.1-3.2
- Lecture presentation.

Example: A tracking design

We want to automatically steer the position $p = (p_x, p_y) \in \mathbb{R}^2$ of a vehicle along a straight-line path.

Solving this as a *tracking problem*, we need to design a desired time-parametrized trajectory.

A straight line can be parametrized by a path variable s as

$$p_d = \begin{bmatrix} a_1 s + b_1 \\ a_2 s + b_2 \end{bmatrix} \in \mathbb{R}^2$$

where a_1, a_2, b_1, b_2 are design constants.

Example cont... Desired trajectory

Suppose further that the vehicle should move along the path with a constant speed reference u_{ref}.
This means that $|\dot{p}_d| = \sqrt{a_1^2 + a_2^2}|\dot{s}| = |u_{ref}|$, and integration yields

$$s(t) = s(0) + \int_0^t \frac{u_{ref}}{\sqrt{a_1^2 + a_2^2}} d\tau = s(0) + \frac{u_{ref}}{\sqrt{a_1^2 + a_2^2}} t =: \upsilon_t(t).$$

Setting $s(0) = 0$ gives the desired trajectory for the vehicle as

$$p_d(t) = \begin{bmatrix} u_{ref} \frac{a_1}{\sqrt{a_1^2 + a_2^2}} t + b_1 \\ u_{ref} \frac{a_2}{\sqrt{a_1^2 + a_2^2}} t + b_2 \end{bmatrix}$$

which satisfies both the path-following and speed objectives in a combined package.

Example cont... State-space representation

For control design, let

$$x_d(t) := \begin{bmatrix} p_d(t) \\ \dot{p}_d(t) \end{bmatrix},$$

and assume the vehicle dynamics is that of a double integrator, $\ddot{p} = u$. Using $x_1 := p$ and $x_2 := \dot{p}$ yields the $\mathbb{R}^4$ state-space system:

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u = Ax + Bu$$

The error dynamics $e := x - x_d(t)$ then becomes

$$\dot{e} = \begin{bmatrix} x_2 - \dot{p}_d \\ u - \ddot{p}_d \end{bmatrix} = Ae + B\left(u - \ddot{p}_d\right).$$

Example cont... CLF and control law

Let a CLF be

$$
\begin{aligned}
V(e) &= e^\top P e \\
\dot{V} &= \dot{e}^\top P e + e^\top P \dot{e} \\
&= \left[Ae + B\left(u - \ddot{p}_d\right)\right]^\top Pe + e^\top P \left[Ae + B\left(u - \ddot{p}_d\right)\right]
\end{aligned}
$$

Let

$$
\begin{aligned}
& K = \begin{bmatrix} k_1 & k_2 \end{bmatrix} \\
& P\left(A - BK\right) + \left(A - BK\right)^\top P = -I
\end{aligned}
$$

Then the tracking control law

$$u = -Ke + \ddot{p}_d(t)$$

gives

$$\dot{V} = e^\top \left(A - BK\right)^\top Pe + e^\top P\left(A - BK\right)e = -e^\top e.$$

The Geometric Task

As presented in Skjetne et al. (2005), *The Maneuvering Problem* is comprised of the two tasks, in prioritized order:

1. **Geometric Task:** -for some absolutely continuous function $s(t)$, force the output y to converge to the desired parametrized path $y_d(s)$, that is,

$$\lim_{t\to\infty} |y(t) - y_d(s(t))| = 0.$$

The Dynamic Task

2. **Dynamic Task:** Satisfy one or more of the assignments:
   1. *Time Assignment:* -force s to converge to a desired time assignment $\tau(t)$,
   $$\lim_{t\to\infty} |s(t) - \tau(t)| = 0.$$
   2. *Speed Assignment:* -force $\dot{s}$ to converge to a desired speed assignment $\upsilon(s,t)$,
   $$\lim_{t\to\infty} |\dot{s}(t) - \upsilon(s(t),t)| = 0.$$
   3. *Acceleration Assignment:* -force $\ddot{s}$ to converge to a desired acceleration assignment $\alpha(\dot{s}(t), s(t), t)$,
   $$\lim_{t\to\infty} |\ddot{s}(t) - \alpha(\dot{s}(t), s(t), t)| = 0.$$

Example: From tracking to maneuvering

For our vehicle control problem, $\ddot{p} = u$, solving this as a *maneuvering problem*, we keep the desired parametrized trajectory as is, i.e.:

The **geometric task** is to converge to the parametrized path

$$p_d(s) = \begin{bmatrix} p_{d,x}(s) \\ p_{d,y}(s) \end{bmatrix} := \begin{bmatrix} a_1 s + b_1 \\ a_2 s + b_2 \end{bmatrix} \in \mathbb{R}^2$$

Example cont... Dynamic task

As the **dynamic task**, the goal was to move along the path with a constant speed u_{ref}.

This means that

$$|\dot{p}_d| = |p_d^s(s)\dot{s}| = \sqrt{p_{d,x}^s(s)^2\dot{s}^2 + p_{d,y}^s(s)^2\dot{s}^2} = \sqrt{a_1^2 + a_2^2}|\dot{s}| = |u_{ref}|$$

The *speed assignment* becomes

$$\dot{s} \to \upsilon(s) := \frac{u_{ref}}{\sqrt{p_{d,x}^s(s)^2 + p_{d,y}^s(s)^2}} = \frac{u_{ref}}{\sqrt{a_1^2 + a_2^2}} =: \upsilon_0.$$

Example cont... Control law

Correspondingly, we define

$$x_d(s) = \begin{bmatrix} x_{d,1}(s) \\ x_{d,2}(s) \end{bmatrix} := \begin{bmatrix} p_d(s) \\ p_d^s(s)\upsilon_0 \end{bmatrix}$$

$$e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} := \begin{bmatrix} x_1 - x_{d,1}(s) \\ x_2 - x_{d,2}(s) \end{bmatrix} = x - x_d(s)$$

For the control law, we now propose:

$$\begin{aligned} u &= -K\left(x - x_d(s)\right) + p_d^{s^2}(s)\upsilon_0 \\ &= -Ke + p_d^{s^2}(s)\upsilon_0. \end{aligned}$$

Example cont... Control law

This gives the closed-loop system

$$\dot{x}_1 = x_2 = x_2 - x_{d,2}(s) + x^s_{d,1}(s)\upsilon_0$$
$$\dot{x}_2 = -k_1\left(x_1 - x_{d,1}(s)\right) - k_2\left(x_2 - x_{d,2}(s)\right) + p^{s^2}_d(s)\upsilon_0$$

$$\dot{x} = (A - BK)\left(x - x_d(s)\right) + x^s_d(s)\upsilon_0$$

or

$$\begin{aligned}\dot{e}_1 &= x_2 - x^s_{d,1}(s)\dot{s} + x_{d,2}(s) - x_{d,2}(s)\\ &= e_2 + x^s_{d,1}(s)\left[\upsilon_0 - \dot{s}\right]\\ \dot{e}_2 &= -k_1 e_1 - k_2 e_2 + x^s_{d,2}(s)\left[\upsilon_0 - \dot{s}\right]\end{aligned}$$

$$\dot{e} = (A - BK)\,e + x^s_d(s)\left[\upsilon_0 - \dot{s}\right]$$

Example cont... CLF

Differentiating the CLF

$$\begin{aligned}V(x,s) &= \left(x - x_d(s)\right)^\top P\left(x - x_d(s)\right)\\ &= e^\top P e\end{aligned}$$

gives

$$\begin{aligned}\dot{V} &= 2e^\top P\left[(A - BK)\,e + x^s_d(s)\left(\upsilon_0 - \dot{s}\right)\right]\\ &= -\left|e\right|^2 + 2e^\top P x^s_d(s)\left[\upsilon_0 - \dot{s}\right]\\ &= -\left|e\right|^2 - V^s(x,s)\left[\upsilon_0 - \dot{s}\right],\end{aligned}$$

where

$$V^s(x,s) = -2\left(x - x_d(s)\right)^\top P x^s_d(s) = -2e^\top P x^s_d(s)$$

Example cont... Choosing update law

We have:

$$\dot{V} = -|x - x_d(s)|^2 - V^s(x,s)\,[v_0 - \dot{s}]$$
$$= -|e|^2 - V^s(x,s)\,[v_0 - \dot{s}]$$

- Assigning

  $$\dot{s} = v_0$$

  solves the dynamic task identically, since then

  $$\dot{V} = -|e|^2$$

  We call it a **Tracking update law**. Why?

Example cont... Choosing update law

We have:

$$\dot{V} = -|x - x_d(s)|^2 - V^s(x,s)\,[v_0 - \dot{s}]$$
$$= -|e|^2 - V^s(x,s)\,[v_0 - \dot{s}]$$

- Assigning

  $$\dot{s} = v_0 - \mu V^s(x,s), \qquad \mu \geq 0,$$

  solves the dynamic task in the limit as $e = x(t) - x_d(s(t)) \to 0$, since

  $$\dot{V} = -|e|^2 - \mu V^s(x,s)^2 \leq -|e|^2$$

  and

  $$V^s(x,s) = -2e^\top P x_d^s(s) \to 0 \quad \text{as} \quad e \to 0.$$

  We call this a **Gradient update law**. Why?

Example cont... Choosing update law

We have:

$$\begin{aligned}\dot{V} &= -\left|x - x_d(s)\right|^2 - V^s(x,s)\left[v_0 - \dot{s}\right] \\ &= -\left|e\right|^2 - V^s(x,s)\left[v_0 - \dot{s}\right]\end{aligned}$$

- Choosing

$$\begin{aligned}\dot{s} &= v_0 - \omega \\ \dot{\omega} &= -\lambda\left(\omega - \mu V^s(x,s)\right), \qquad \lambda, \mu > 0\end{aligned}$$

  solves the dynamic task in the limit as $e = x(t) - x_d(s(t)) \to 0$ and $\omega(t) \to 0$, since for

$$W = V + \frac{1}{2\mu\lambda}\omega^2 = e^\top P e + \frac{1}{2\mu\lambda}\omega^2$$

  we get

$$\dot{W} = -\left|e\right|^2 - V^s(x,s)\omega + \frac{1}{\mu\lambda}\omega\dot{\omega} = -\left|e\right|^2 - \frac{1}{\mu}\omega^2.$$

  We call this a **Filtered gradient update law**. Why?

Example cont... Choosing update law

Note that the vector $x_d^s(s)$ is for a given s the tangent vector for the path at $x_d(s)$. Depending on parametrization (around corners, etc.) this may have varying length. Since this is included in $V^s(x,s)$, this gradient function will have a varying gain along the path.

A modified version of the gradient update law is then to rather use the unit tangent vector:

- Choosing (with $\mu \geq 0$)

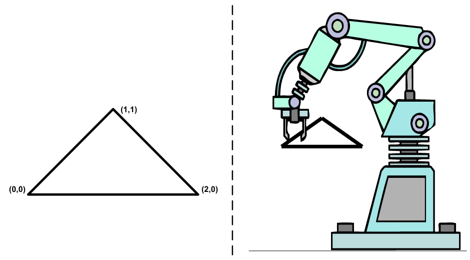
$$\dot{s} = v_0 - \frac{\mu}{\left|x_d^s(s)\right|}V^s(x,s) = v_0 + 2\mu\frac{x_d^s(s)^\top}{\left|x_d^s(s)\right|}P\left(x - x_d(s)\right)$$

  also solves the dynamic task in the limit as $e = x(t) - x_d(s(t)) \to 0$, where $\frac{x_d^s(s)^\top}{\left|x_d^s(s)\right|}$ is the unit tangent vector along the path.

  This is the **Unit-tangent Gradient update law**.

Example: Robotic manipulator

(Skjetne et al., 2004, Example 2) For the robotic manipulator (e.g. a cutting or welding tool):



- The tip of the tool y to trace a desired triangular path.
- Trace the path as fast as possible without exceeding a maximum speed constraint of about $u_{\max} \approx 0.1\,\mathrm{m}\,/\,\mathrm{s}$.
- Deviation from the path always kept less than $10^{-3}\,\mathrm{m}$.

Example cont... Model

This is a typical maneuvering application:

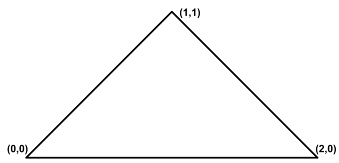
- The depicted triangle is the desired path in $\mathbb{R}^2$.
- Along the path a desired speed profile must be designed that conform to the maximum speed $u_{\max}$ along the edges, while it should slow down at the corners to avoid transients that could exceed the maximum deviation constraint.

Suppose the motion of the tip is represented by

$$M\ddot{x} + D(\dot{x})\,\dot{x} + K(x)\,x = u$$

where $x \in \mathbb{R}^2$ is the position, the force $u \in \mathbb{R}^2$ is the control, $M = M^\top > 0$ is the system inertia matrix, and $D(\dot{x})$ and $K(x)$ are the nonlinear damping and spring matrices.

Example cont... Path



Control objective: Let $y = x$ to trace $y_d(s)$, parametrized as

$$y_d(s) = \begin{cases} [s, \quad s]^\top; & s \in [0,1), \\ [s, \quad 2-s]^\top; & s \in [1,2), \\ [4-s, \quad 0]^\top; & s \in [2,4). \end{cases}$$

Since the path is not $\mathcal{C}^1$, perfect tracing is not possble.

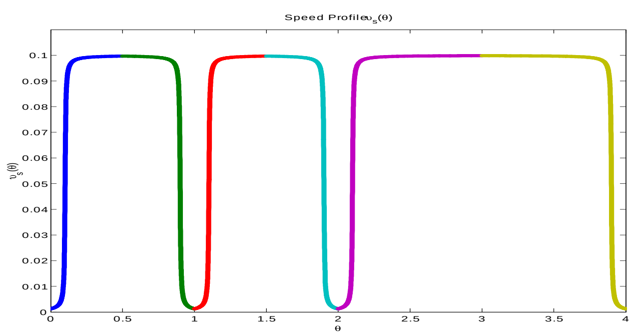
Tracing within the maximum deviation is achievable with the strategy to trace each edge and stop and restart at each corner.
The speed to be small near the corners to avoid transients that could exceed the maximum deviation constraint.

Example cont... Speed assignment

This motivates the speed profile $\upsilon(s)$ between corners k and $k+1$:

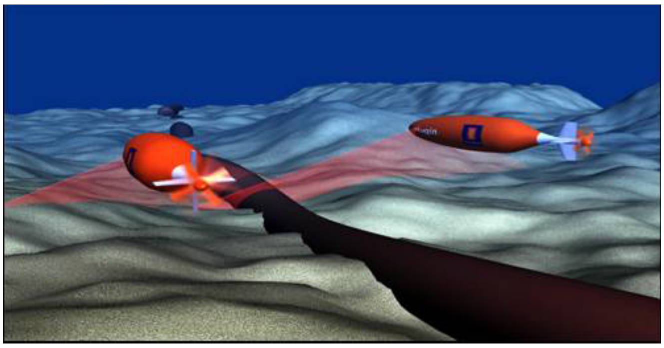
$$\upsilon(s) = \begin{cases} \frac{u_{\max}}{\pi |y_d^s|} \arctan\left(\frac{s - s_k - a_1}{a_2}\right) + \frac{u_{\max}}{2|y_d^s|}; & s \in \left[s_k, \ s_k + \frac{s_{k+1} - s_k}{2}\right) \\ \frac{u_{\max}}{\pi |y_d^s|} \arctan\left(\frac{s_{k+1} - a_1 - s}{a_2}\right) + \frac{u_{\max}}{2|y_d^s|}; & s \in \left[s_k + \frac{s_{k+1} - s_k}{2}, \ s_{k+1}\right) \end{cases}$$

where $s_1 = 0$, $s_2 = 1$, $s_3 = 2$, and $s_4 = 4$.
The parameters a_1 and a_2 are used to shape the corners.



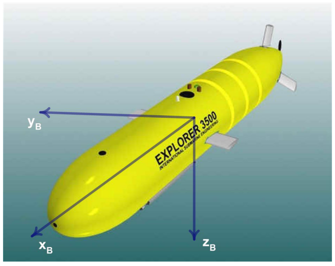
Example: AUV guidance and control

As the oil and gas industry moves production to greater depths, the need for more underwater autonomous control increases. One envisioned task is inspection of underwater pipelines.



An AUV is a 6 DOF system given by the Cartesian coordinates $p = \mathrm{col}(x, y, z)$ in a local Earth-fixed reference frame $\mathcal{E}$ and orientation given by the Euler angles $\Theta = \mathrm{col}(\phi, \theta, \psi)$. The body-fixed velocity vector in $\mathcal{B}$ is then $\nu = \mathrm{col}(u, v, w, p, q, r)$.

Example cont... AUV model



- $J(\Theta) = \mathrm{diag}\left(R(\Theta)_b^e,\ T_\Theta(\Theta)\right)$ is a transformation matrix,
- M is the system inertia matrix,
- C accounts for Coriolis and centripetal terms,
- D is a nonlinear damping matrix,
- g is a vector of restoring forces,
- g_0 accounts for pretrimming and ballast control, and
- τ_i is the control input vector (assumed fully actuated).

Example cont... Path generation

Path generation: We let the overall desired path for the AUV be

$$\eta_d(s) = \mathrm{col}(p_d(s), \Theta_d(s))$$

Here, $p_d(s) \in \mathcal{E}$ is the desired positional curve that typically could be generated by a set of specified setpoints or some other path-generation technique.

$\Theta_d(s)$ is the desired orientation along the curve, given as

$$\begin{aligned}
\phi_d(s) &:= \arctan\left(\frac{z_d^s(s)}{y_d^s(s)}\right) \\
\theta_d(s) &:= \arctan\left(\frac{-z_d^s(s)}{x_d^s(s)}\right) \\
\psi_d(s) &:= \arctan\left(\frac{y_d^s(s)}{x_d^s(s)}\right)
\end{aligned}$$

Example cont... Speed assignment

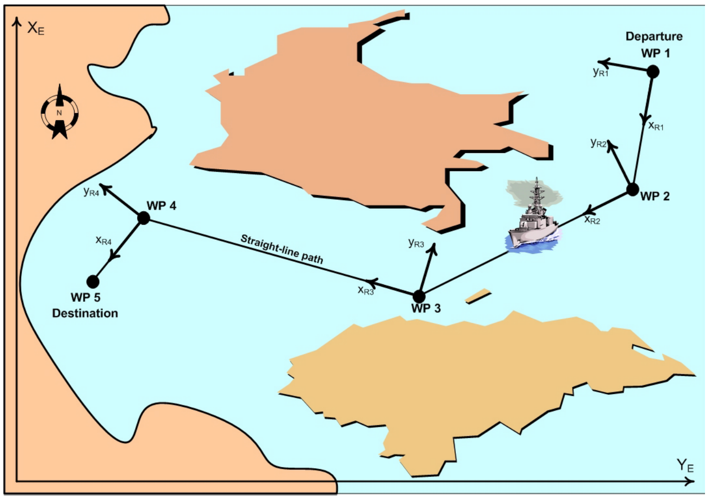
Speed assignment: The speed assignment can be designed according to a desired path speed $u_d(t)$.
In $\mathbb{R}^3$ this gives:

$$v(s,t) := \frac{u_d(t)}{\sqrt{x_d^s(s)^2 + y_d^s(s)^2 + z_d^s(s)^2}}.$$

In this case the AUV is chosen to be aligned with the path, and the speed $u_d(t)$ therefore corresponds to a desired surge speed.

A piecewise linear path

Consider the path in the figure.



Next we discuss different methods for path parametrization based on waypoints (Skjetne et al., 2005, Chapter 2.1).

Discrete parametrization

Let $\mathcal{I} = \{1, 2, \ldots, n\}$ be a set identifying each line-segment l_i of the path.

At each line segment l_i, $i \in \mathcal{I}$, we attach a local path frame $\mathcal{R}_i$ with origin at way-point p_i and x-axis pointed towards p_{i+1}.

Letting $\epsilon_1 := \text{col}(1, 0)$, $\epsilon_2 := \text{col}(0, 1)$, and ψ_i be the angle of rotation of frame $\mathcal{R}_i$ (line segment i) in the local Earth frame $\mathcal{E}$, this gives

$$\psi_i = \arctan\left(\frac{\epsilon_2^\top (p_{i+1} - p_i)}{\epsilon_1^\top (p_{i+1} - p_i)}\right).$$

A vector $q_{R_i} \in \mathcal{R}_i$ is mapped to $\mathcal{E}$ by use of the rotation matrix $R_2(\psi_i) : [-\pi, \pi) \to \mathbb{R}^{2\times 2}$, that is,

$$q_E = R_2(\psi_i) q_{R_i} \in \mathcal{E}, \qquad R_2(\psi_i) := \begin{bmatrix} \cos\psi_i & -\sin\psi_i \\ \sin\psi_i & \cos\psi_i \end{bmatrix}.$$

Discrete parametrization

For a position vector $p \in \mathcal{E}$, let $q \in \mathcal{R}_i$ be defined as

$$q := R_2(\psi_i)^\top (p - p_i) \,,$$

that is, the point p expressed as a vector in $\mathcal{R}_i$.

We then get that $\epsilon_1^\top q$ is the projection of p onto the line segment l_i, and $|\epsilon_2^\top q|$ is the Euclidean distance from p to l_i (*cross-track error*).

The path is defined by those points for which this distance is zero, that is,

$$\mathcal{P} := \left\{ p \in \mathbb{R}^2 : \ \exists i \in \mathcal{I} \text{ s.t. } \begin{array}{c} \epsilon_2^\top R_2(\psi_i)^\top (p - p_i) = 0 \\ \text{and} \\ 0 \le \epsilon_1^\top R_2(\psi_i)^\top (p - p_i) \le \epsilon_1^\top R_2(\psi_i)^\top (p_{i+1} - p_i) \end{array} \right\}.$$

We notice that the path is discretely parametrized by the index i, which identifies all points along line segment l_i.

Continuous parametrization

For a continuous parametrization we define

$$p_d(s) := \begin{cases} p_1 + s\,(p_2 - p_1)\,; & s \in [0, 1) \\ p_2 + (s-1)\,(p_3 - p_2)\,; & s \in [1, 2) \\ \quad \vdots & \\ p_i + (s - i + 1)\,(p_{i+1} - p_i)\,; & s \in [i-1, i) \\ \quad \vdots & \\ p_{n-1} + (s - n + 2)\,(p_n - p_{n-1})\,; & s \in [n-1, n). \end{cases}$$

The path is then simply given by the set

$$\mathcal{P} := \left\{ p \in \mathbb{R}^2 : \ \exists s \in [0, n) \text{ s.t. } p = p_d(s) \right\}.$$

For this path, each point along the path is uniquely determined by a specific value $s \in [0, n)$.

Hybrid parametrization

Using the path parameter to take values $s \in [0, n)$ is unnecessary and can be numerically problematic. Instead, we identify each line segment l_i by $i \in \mathcal{I}$ and continuously parametrize each l_i by $s \in [0, 1)$.

An expression for the line segment l_i is then

$$p_d(i, s) := p_i + s\,(p_{i+1} - p_i)$$

so that $p_d(i, 0) = p_i$ and $\lim_{s \nearrow 1} p_d(i, s) = p_{i+1}$.

The path becomes

$$\mathcal{P} := \left\{ p \in \mathbb{R}^2 : \; \exists i \in \mathcal{I} \text{ and } s \in [0, 1) \text{ s.t. } p = p_d(i, s) \right\}.$$

Notice that each point along the path (except p_{n+1}) is uniquely determined by a pair $(i, s) \in \mathcal{I} \times [0, 1)$.

Generating a C^r path

One method to generate a $\mathcal{C}^r$ path is to specify a set of $n + 1$ way-points (WPs), and then construct a sufficiently differentiable curve that goes through the way-points by using splines and interpolations techniques.

To construct the overall desired path $p_d(s)$ it is first divided into n subpaths $p_{d,i}(s)$, $i = 1, \ldots, n$ between the WPs. Each of these is expressed as a polynomial in s of a certain order. Then the expressions for the subpaths are concatenated at the WPs to assemble the full path.

To ensure that the overall path is sufficiently differentiable at the WPs, the order of the polynomials must be sufficiently high.

Again, the hybrid parametrization is numerically most powerful.

Generating a C^r path: The general case

For simplicity we consider $\mathbb{R}^2$.

Overall desired curve:

$$p_d(s) = \mathrm{col}(x_d(s), y_d(s)), \quad s \in [0, n]$$

Subpaths:

$$p_{d,i}(s) = \mathrm{col}(x_{d,i}(s), y_{d,i}(s)), \quad i \in \mathcal{I}$$

Way-points:

$$p_i = \mathrm{col}(x_i, y_i), \quad i \in \mathcal{I} \cup \{n+1\}$$

A common choice is to let s take an integer value at each WP, starting with $s = 0$ at WP 1 and $s = i - 1$ at WP i.

Generating a C^r path: The general case

The differentiability requirement, $p_d(s) \in \mathcal{C}^r$, means that at the connection of two subpaths, the following must hold:

$$\begin{aligned}
\lim_{s \nearrow i-1} x_{d,i-1}(s) &= \lim_{s \searrow i-1} x_{d,i}(s) & \qquad \lim_{s \nearrow i-1} y_{d,i-1}(s) &= \lim_{s \searrow i-1} y_{d,i}(s) \\
\lim_{s \nearrow i-1} x^{s}_{d,i-1}(s) &= \lim_{s \searrow i-1} x^{s}_{d,i}(s) & \qquad \lim_{s \nearrow i-1} y^{s}_{d,i-1}(s) &= \lim_{s \searrow i-1} y^{s}_{d,i}(s) \\
&\vdots & &\vdots \\
\lim_{s \nearrow i-1} x^{s^r}_{d,i-1}(s) &= \lim_{s \searrow i-1} x^{s^r}_{d,i}(s) & \qquad \lim_{s \nearrow i-1} y^{s^r}_{d,i-1}(s) &= \lim_{s \searrow i-1} y^{s^r}_{d,i}(s)
\end{aligned}$$

for $i \in \mathcal{I} \backslash \{1\}$.

Consider polynomials of order k,

$$\begin{aligned}
x_{d,i}(s) &= a_{k,i}s^k + \ldots + a_{1,i}s + a_{0,i} \\
y_{d,i}(s) &= b_{k,i}s^k + \ldots + b_{1,i}s + b_{0,i}
\end{aligned}$$

where the coefficients $\{a_{j,i},\ b_{j,i}\}$ must be determined.

Generating a C^r path: The general case

For each subpath there are $(k+1)\cdot 2$ unknowns so that there are $(k+1)\cdot 2n$ unknown coefficients in total to be determined for the full path.

Many methods for calculating these coefficients exist:

- Most obvious: to set it up as a large set of $(k+1)\cdot 2n$ linear equations,

$$A\phi = b, \qquad \phi^\top = \left[a^\top, b^\top\right],$$

  for the full path and solve these in a single operation as $\phi = A^{-1}b$. However, as the number n of subpaths increases, this encounters numerical problems in the inversion of A.

- Alternatively: to calculate the coefficients for each subpath independently. To ensure continuity at connection points, assign numerical values which are common for the neighboring subpaths.

A hybrid parametrization

We choose to continuously parametrize each subpath within a fixed interval $\varepsilon \in [0,1)$, and let an index i identify the subpath i, corresponding to the hybrid parametrization.

Assuming that the subpaths are connected at the waypoints, the equations used to calculate the coefficients become:

$\mathcal{C}^0$: Continuity at the WPs gives for $i \in \mathcal{I}$:

$$\begin{aligned} x_{d,i}(0) &= x_i & \quad y_{d,i}(0) &= y_i \\ x_{d,i}(1) &= x_{i+1} & \quad y_{d,i}(1) &= y_{i+1}. \end{aligned}$$

A hybrid parametrization

$\mathcal{C}^1$: The slopes at the first and last WPs are chosen as:

$$\begin{array}{ll} x^s_{d,1}(0) = x_2 - x_1 & y^s_{d,1}(0) = y_2 - y_1 \\ x^s_{d,n}(1) = x_{n+1} - x_n & y^s_{d,n}(1) = y_{n+1} - y_n. \end{array}$$

The slopes at the intermediate WPs are chosen as:

$$\left.\begin{array}{l} x^s_{d,i}(0) = \lambda\left(x_{i+1} - x_{i-1}\right) \\ y^s_{d,i}(0) = \lambda\left(y_{i+1} - y_{i-1}\right) \end{array}\right\} \quad i = 2, \ldots, n$$

$$\left.\begin{array}{l} x^s_{d,i}(1) = \lambda\left(x_{i+2} - x_{i}\right) \\ y^s_{d,i}(1) = \lambda\left(y_{i+2} - y_{i}\right) \end{array}\right\} \quad i = 1, \ldots, n-1$$

where $\lambda > 0$ is a design constant.

A hybrid parametrization

$\mathcal{C}^j$: Setting derivatives of order $j \geq 2$ to zero gives for $i \in \mathcal{I}$:

$$\begin{array}{ll} x^{s^j}_{d,i}(0) = 0 & y^{s^j}_{d,i}(0) = 0 \\ x^{s^j}_{d,i}(1) = 0 & y^{s^j}_{d,i}(1) = 0. \end{array}$$

The result is a hyprid parametrization of the path,

$$\hat{p}_d(i, \varepsilon) = \left[\begin{array}{c} x_{d,i}(\varepsilon) \\ y_{d,i}(\varepsilon) \end{array}\right],$$

where $i \in \mathcal{I}$ and $\varepsilon \in [0, 1)$.

A hybrid parametrization

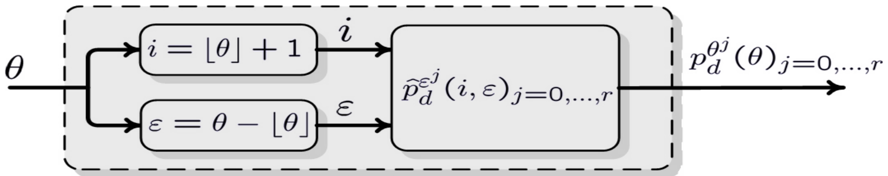
This does not conform with the requirement of a continuous parametrization. However, since

$$\lim_{\varepsilon \nearrow 1} \hat{p}_d^{\varepsilon^j}(i-1,\varepsilon) = \lim_{\varepsilon \searrow 0} \hat{p}_d^{\varepsilon^j}(i,\varepsilon)$$

holds for $j = 0,\dots,r$ and $i = 2,\dots,n-1$, a map $p_d(s)$ can be constructed so that $s \mapsto p_d(s)$ is $\mathcal{C}^r$.

For $s \in [0,n)$, let $i = \lfloor s \rfloor + 1 \in \mathcal{I}$ and $\varepsilon = s - \lfloor s \rfloor \in [0,1)$ where $\lfloor \cdot \rfloor$ is the floor operator. Then we have the continuous $\mathcal{C}^r$ map

$$s \mapsto p_d(s) := \hat{p}_d\left(i(s), \varepsilon(s)\right)$$



The double integrator and the unit circle

We consider the double integrator

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= u\end{aligned} \tag{1}$$

and the task of stabilizing the path

$$\mathcal{P} := \left\{x \in \mathbb{R}^2 : x^\top x = 1\right\} \tag{2}$$

without creating any equilibria in $\mathcal{P}$.
By converse Lyapunov theory (Teel & Praly, 2000) there does not exist a continuous or discontinuous time-invariant state feedback control that renders the unit circle GAS[1]. The reasoning follows...

[1] In the case of discontinuous feedback, this statement applies when the solution concept used is that due to Filippov; for example, see Filippov (1988).

Unit circle cannot be GAS

Conjecture: By converse Lyapunov theory (Teel & Praly, 2000) there does not exist a continuous or discontinuous time-invariant state feedback control that renders the unit circle GAS.

The reasoning follows from contradiction, as follows:

If such a feedback existed there would exist a smooth Lyapunov function demonstrating GAS of the circle. In particular, this function would be positive definite with respect to the circle and would have a directional derivative, in the direction of the closed-loop vector field, that is negative definite with respect to the circle. The Lyapunov function would obtain its minimum on the circle, and it would therefore have a maximum inside the circle. At the maximum, its gradient would be zero and so the directional derivative along the closed-loop vector field could not be negative.

So there must not be a Lyapunov function, which implies that GAS cannot be achieved by such a control.

Modifying our objective

An alternative is to try to stabilize the set

$$\mathcal{A} := \left\{ (x, s) : x - \begin{bmatrix} \cos s \\ -\sin s \end{bmatrix} = 0 \right\} \tag{3}$$

for the system (1) and

$$\dot{s} = 1. \tag{4}$$

A control rendering the set $\mathcal{A}$ GAS will steer (x, s) to the set $\mathcal{P} \times \mathbb{R}$ since

$$\begin{bmatrix} \cos s \\ -\sin s \end{bmatrix}^\top \begin{bmatrix} \cos s \\ -\sin s \end{bmatrix} = 1$$

and[2] thus $\mathcal{A} \subset \mathcal{P} \times \mathbb{R}$.

However, choosing to stabilize $\mathcal{A}$ under the constraint (4) may introduce large transients in the distance to $\mathcal{P} \times \mathbb{R}$. In an attempt to control these transients, we will consider controlling s as well.

[2] While $\mathcal{P} \times \mathbb{R}$ defines a two-dimensional cylindrical surface in $\mathbb{R}^3$, the set $\mathcal{A}$ defines a one-dimensional spiral on the cylindrical surface in $\mathbb{R}^3$.

Control law

We consider the control

$$\begin{aligned} \dot{s} &= 1 + \mu\omega\left(x_1, x_2, s\right), \qquad \mu > 0 \\ u &= \alpha\left(x_1, x_2, s\right) \end{aligned} \tag{5}$$

for the system (1). This will render the set $\mathcal{A}$ UGES and achieve good performance for large μ.
To design the functions ω and α in (5) we select a Hurwitz matrix

$$A := \begin{bmatrix} 0 & 1 \\ -k_1 & -k_2 \end{bmatrix}$$

and let $P = P^\top > 0$ be such that $A^\top P + PA = -Q$ where $Q = Q^\top > 0$. Using

$$V\left(x, s\right) := \left(x - \xi(s)\right)^\top P\left(x - \xi(s)\right) \tag{6}$$

$$\xi\left(s\right) = \begin{bmatrix} \xi_1(s) \\ \xi_2(s) \end{bmatrix} := \begin{bmatrix} \cos s \\ -\sin s \end{bmatrix} \tag{7}$$

and $K := [k_1 \; k_2]$, we assign ω and α in (5) as

$$\omega\left(x, s\right) = -V^s\left(x, s\right) = 2\left(x - \xi(s)\right)^\top P\xi^s(s) \tag{8}$$

$$\alpha\left(x, s\right) = -K\left(x - \xi(s)\right) + \xi_2^s(s). \tag{9}$$

... Control law

Differentiating (6) along the solutions of the resulting closed-loop equations

$$\begin{aligned} \dot{x} &= A\left(x - \xi(s)\right) + \xi^s(s) \\ \dot{s} &= 1 - \mu V^s(x, s) \end{aligned} \tag{10}$$

gives

$$\begin{aligned} \dot{V}(x, s) &= -\left(x - \xi(s)\right) Q\left(x - \xi(s)\right) - \mu V^s(x, s)^2 \\ &\leq -\lambda_{\min}\left(Q\right) \left|x - \xi(s)\right|^2 \leq -cV(x, s), \qquad c := \frac{\lambda_{\min}\left(Q\right)}{\lambda_{\max}\left(P\right)} \end{aligned}$$

It follows that $|x(t) - \xi(s(t))| \leq ke^{-\frac{c}{2}t}\left|x(0) - \xi(s(0))\right|$ holds, where $k := \sqrt{\frac{\lambda_{\max}(P)}{\lambda_{\min}(P)}}$, and the set $\mathcal{A}$ is therefore UGES.

Stability of $\mathcal{P}$, however, has not been achieved. Why?

Near forward invariance of

The set $\mathcal{P}$ is not forward invariant. To show a *"near forward invariance"* property, we induce a two time-scale behavior of the closed-loop system (10) by increasing μ. Letting $\varepsilon = 1/\mu$ be small, allows us to analyze (10) as a singularly perturbed system (Teel et al., 2003; Kokotović et al., 1999; Khalil, 2002).

Let the fast time scale be $t_f = t/\varepsilon$ and define $x' := \frac{dx}{dt_f}$. Then the closed-loop can be written as

$$\begin{aligned} x' &= \varepsilon A\left(x - \xi(s)\right) + \varepsilon \xi^s(s) \\ s' &= \varepsilon - V^s(x, s). \end{aligned}$$

The rapid transient of s is approximately described by the boundary layer system for $\varepsilon = 0$,

$$x' = 0 \tag{11a}$$

$$s' = -V^s\left(x, s\right), \tag{11b}$$

where x is fixed for $\varepsilon = 0$. Clearly, (11) is a continuous gradient algorithm which minimizes V with respect to s for any fixed x.

Local minima of $V(x, s)$

For a given x, the level curves of $\xi \mapsto (x - \xi)^\top P (x - \xi)$ are:

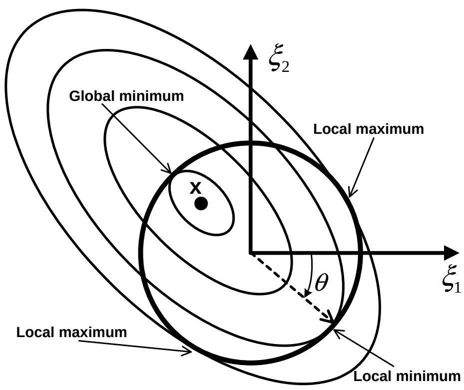


Figure: Replace θ by s.

Four values of s satisfy $V^s(x, s) = 0$, given by the locations where the path $\xi(s)$, indicated as a solid curve in the figure, is tangent to a level set of $\xi \mapsto (x - \xi)^\top P (x - \xi)$.
The four values correspond to two local maxima and two local minima of $s \mapsto V(x, s)$. The global minimum is of interest and denoted $s_V(x)$.

Reduced (quasistatic) system

The function $\xi(s_V(x))$ is substituted into (10) to obtain the *reduced system*

$$\dot{x} = A\left(x - \xi(s_V(x))\right) + \xi^s\left(s_V(x)\right) \tag{12}$$

which **approximately** describes the behavior of x in time-scale t. This motion is restricted to the slow manifold $\mathcal{V}_\varepsilon$, which is ε-close to the manifold $\mathcal{V}_s$ defined by

$$V^s(x, s_V(x)) = -2\left(x - \xi(s_V(x))\right)^\top P\xi^s(s_V(x)) = 0. \tag{13}$$

This is the result of the gradient optimization which rapidly positions $\xi(s)$ to the most favourable position along the desired unit circle.

The subsequent convergence of x is established by differentiating $W(x) := V(x, s_V(x))$ with respect to t along the solutions of the *reduced system* (12). Employing the identity (13), this derivative is

$$\begin{aligned} \dot{W} &= -\left[x - \xi(s_V(x))\right]^\top Q\left[x - \xi(s_V(x))\right] \\ &\leq -\lambda_{\min}(Q)\left|x - \xi(s_V(x))\right|^2 \leq -cW(x(t)) \end{aligned}$$

which is used to show for the *reduced system* that

$$|x(t)|_{\mathcal{P}} \leq k\,|x(0)|_{\mathcal{P}}\, e^{-\frac{c}{2}t}. \tag{14}$$

Near stability

Leaving out some details, it follows that the true solutions of the system is close to the solutions of the reduced system, at least on compact time intervals.

This is made rigorous by enforcing proper initialization, $|x(0)|_{\mathcal{P}}$ sufficiently small and $s(0)$ in the region of convergence of $s_V(x(0))$. Then, for any $\delta > 0$ there exists $\mu^* > 0$ such that $\mu \geq \mu^*$ implies that

$$|x(t)|_{\mathcal{P}} \leq k\,|x(0)|_{\mathcal{P}}\, e^{-\frac{c}{2}t} + \delta, \qquad \forall t \geq 0 \tag{15}$$

holds for the true closed-loop system (10).

The behavior of the solution of the true system (10) will therefore be δ-close to the behavior, characterized by the bound (14), of the reduced system, and this yields a desired near stability property of the unit circle.

Simulation

The figure shows the responses of $x(t)$ and $s(t)$ in a simulation, with $k_1 = 2$ and $k_2 = 1$. Initial positions were $x_0 = [-\frac{\sqrt{2}}{2}\ \frac{\sqrt{2}}{2}]^\top$ (on the circle at the angle $225\,°$) and $s_0 = 0\,°$.

Figures 56 and 57a) show the responses for $\mu = 500$.

In *Run 1,* $V(x_0, \cdot)$ only had one unique initial minimum at $s_V(x_0) = 225\,°$, to which $s(t)$ rapidly converges. Thus, the initial transient in the distance to the circle is small, and $x(t)$ stays close for all time (and eventually converges).

In *Run 2,* on the other hand, we change the P-matrix so that $V(x_0, \cdot)$ has an initial local minimum at $s_{loc.min} = 108\,°$. Since $s(0)$ is in the basin of attraction of $s_{loc.min}$, it rapidly converges to $s_{loc.min}$ and causes the bad transient of $x(t)$.

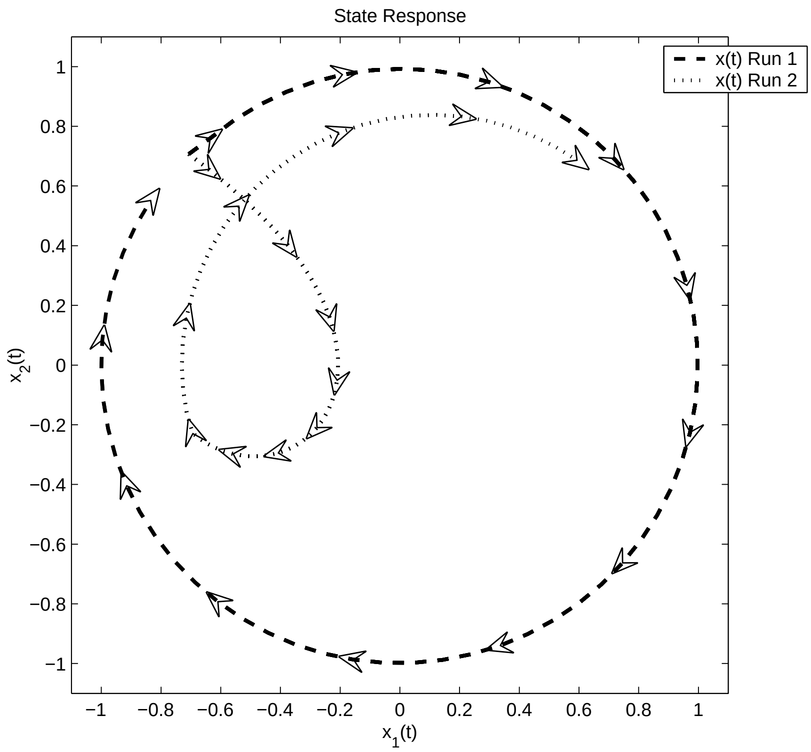
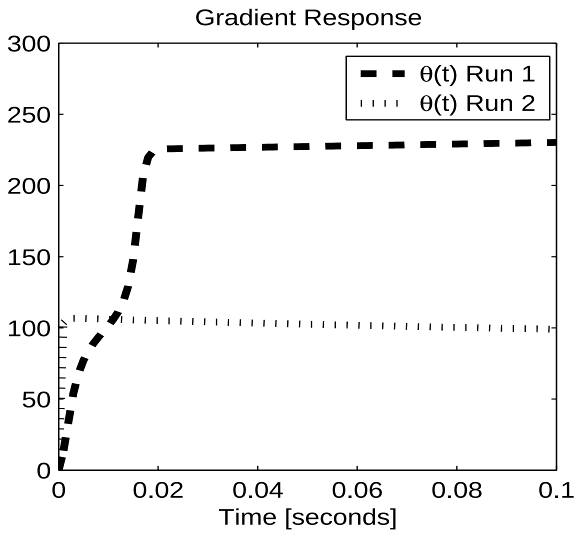
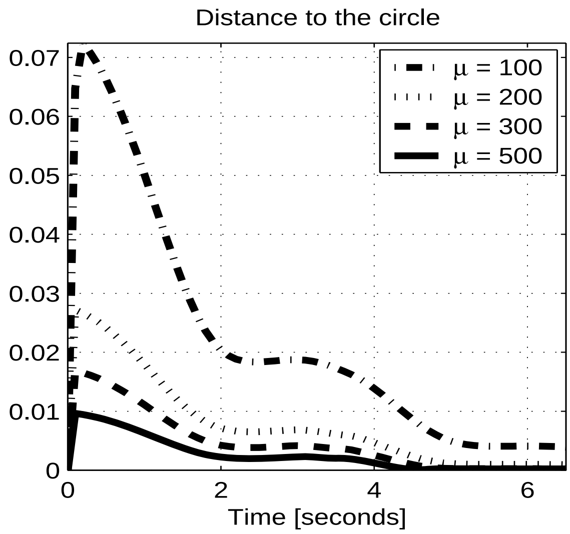


Figure 57 a) shows the responses of $s(t)$ in the two cases (substitute $\theta = s$).





In Figure 57 b), the responses correspond to the unique minimum case in Run 1 for different gains μ. It is seen that the excursion of $x(\,\cdot\,)$ from the circle decreases as μ increases.

Generic maneuvering problem statement

For a system output $y = h(x)$ where $h : \mathbb{R}^n \to \mathbb{R}^m$, the desired manifold is all points represented by the set

$$\mathcal{Q} := \{x \in \mathbb{R}^n : \exists \xi \in \mathbb{R}^q \text{ s.t. } h(x) = h_d(\xi)\}$$

where $q \leq m$ and the map $\xi \mapsto h_d(\xi)$ is sufficiently smooth.

The main generalization is to consider a q-dimensional manifold instead of a 1-dimensional path.

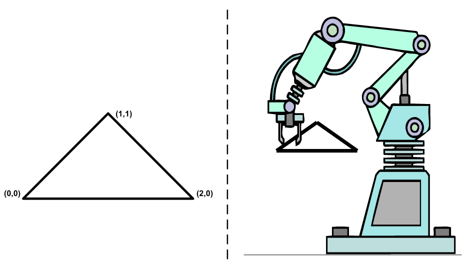
In addition, we want to allow a dynamic assignment f_d that may incorporate feedback from the system output directly.

Generic maneuvering problem statement

As presented in Skjetne et al. (2011), given the parametrization $h_d(\xi)$ of the manifold and a dynamics assignment on the manifold, the **Maneuvering Problem** is comprised of the two tasks:

1. **Geometric task:** For some absolutely continuous function $\xi(t)$, force the output y to converge to the desired manifold $h_d(\xi)$,
$$\lim_{t\to\infty} |y(t) - h_d(\xi(t))| = 0.$$
2. **Dynamic task:** Force $\dot{\xi}$ to converge to a desired dynamic assignment $f_d(\xi, y, t)$,
$$\lim_{t\to\infty} \left|\dot{\xi}(t) - f_d(\xi(t), y(t), t)\right| = 0.$$

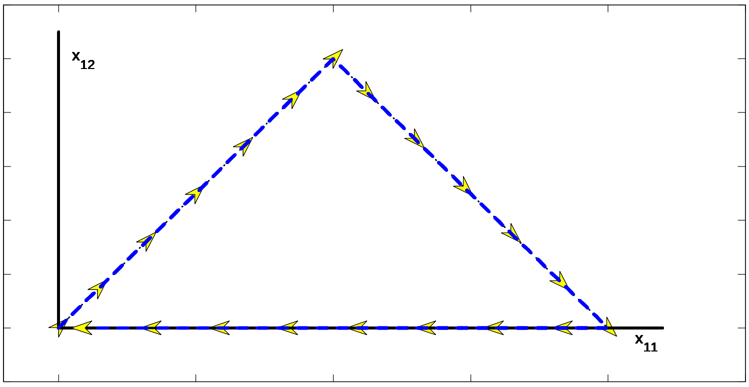
Example: Robotic manipulator

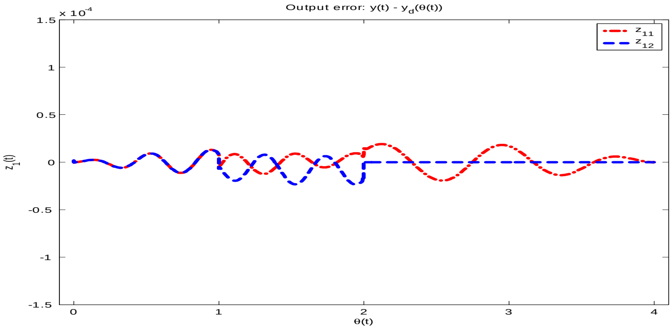


The design procedure yields the control law

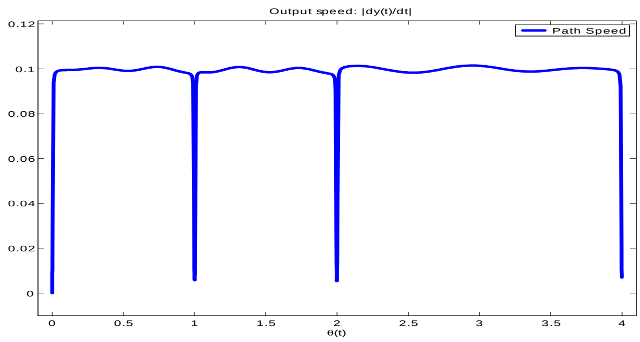
$$\begin{aligned}
\dot{s} &= \upsilon(s) - \omega \\
\dot{\omega} &= -\lambda\omega - 2\mu_1\left(p_1 z_1^\top y_d^s(s) + p_2 z_2^\top M\alpha_1^s\right) \\
u &= -\tfrac{p_1}{p_2} z_1 - K_d z_2 + K_0 x_1 + D_0\alpha_1 + M\sigma_1 \\
&\quad + M\alpha_1^s \upsilon(s) - \tfrac{1}{2}\kappa W(\bar{x}_2) W(\bar{x}_2)^\top p_2 z_2
\end{aligned}$$

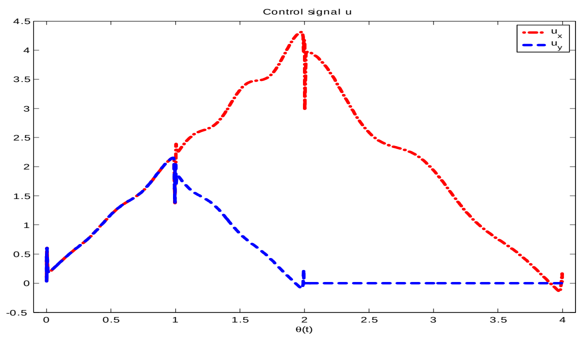
Example cont... Responses





Example cont... Responses



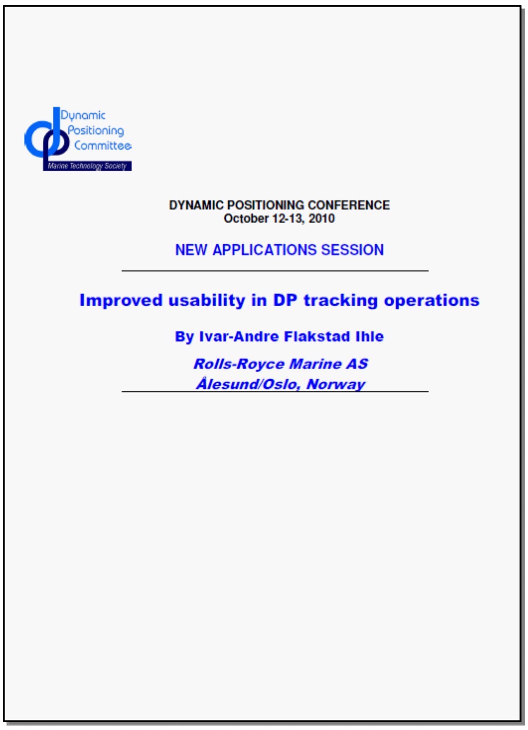


Example cont... Conclusion

This illustrates the gradient-based maneuvering system's capability to always keep the error signals small.

The time spent tracing the path is $51.1\,\mathrm{s}$. If one were able to trace the entire path with speed m_s, then the total time would be $48.3\,\mathrm{s}$ so that the time loss is only 5.8%.

Rolls Royce DP



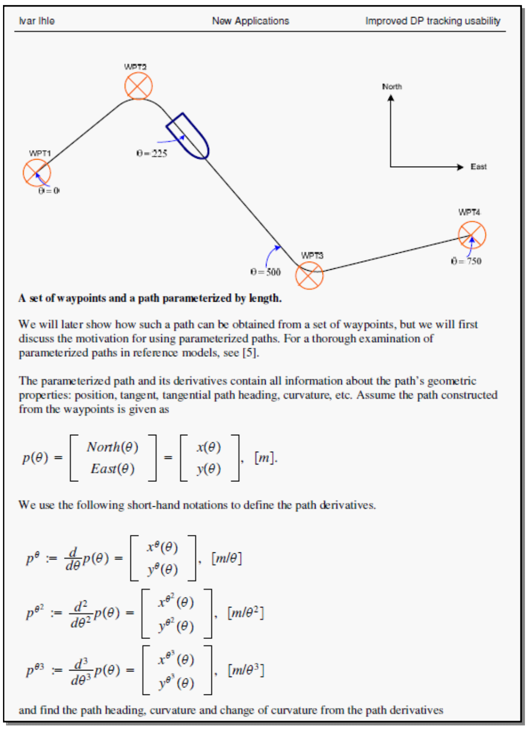
DYNAMIC POSITIONING CONFERENCE
October 12-13, 2010

NEW APPLICATIONS SESSION

Improved usability in DP tracking operations

By Ivar-Andre Flakstad Ihle

Rolls-Royce Marine AS
Ålesund/Oslo, Norway



Ivar Ihle New Applications Improved DP tracking usability

A set of waypoints and a path parameterized by length.

We will later show how such a path can be obtained from a set of waypoints, but we will first discuss the motivation for using parameterized paths. For a thorough examination of parameterized paths in reference models, see [5].

The parameterized path and its derivatives contain all information about the path's geometric properties: position, tangent, tangential path heading, curvature, etc. Assume the path constructed from the waypoints is given as

$$p(\theta) = \begin{bmatrix} North(\theta) \\ East(\theta) \end{bmatrix} = \begin{bmatrix} x(\theta) \\ y(\theta) \end{bmatrix}, \quad [m].$$

We use the following short-hand notations to define the path derivatives.

$$p^{\theta} := \frac{d}{d\theta}p(\theta) = \begin{bmatrix} x^{\theta}(\theta) \\ y^{\theta}(\theta) \end{bmatrix}, \quad [m/\theta]$$

$$p^{\theta^2} := \frac{d^2}{d\theta^2}p(\theta) = \begin{bmatrix} x^{\theta^2}(\theta) \\ y^{\theta^2}(\theta) \end{bmatrix}, \quad [m/\theta^2]$$

$$p^{\theta^3} := \frac{d^3}{d\theta^3}p(\theta) = \begin{bmatrix} x^{\theta^3}(\theta) \\ y^{\theta^3}(\theta) \end{bmatrix}, \quad [m/\theta^3]$$

and find the path heading, curvature and change of curvature from the path derivatives

Ihle (2010)

Preparations for next lecture

DP control system:

- Lecture presentation(s).

Bibliography

Filippov, A. (1988). *Differential Equations with Discontinuous Right-hand sides*. Kluwer Academic Publishers.

Ihle, I.-A. F. (2010). Improved usability in dp tracking operations. In *Dynamic Positioning Conf.* Houston, USA: Marine Tech. Soc.

Khalil, H. K. (2002). *Nonlinear Systems*. New Jersey: Prentice-Hall, Inc, 3 edition.

Kokotović, P., Khalil, H. K., & O'Reilly, J. (1999). *Singular perturbation methods in control*. Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM). Analysis and design, Corrected reprint of the 1986 original.

Skjetne, R., Fossen, T. I., & Kokotović, P. V. (2004). Robust output maneuvering for a class of nonlinear systems. *Automatica*, 40(3), 373–383.

Skjetne, R., Fossen, T. I., & Kokotović, P. V. (2005). Adaptive maneuvering, with experiments, for a model ship in a marine control laboratory. *Automatica*, 41(2), 289–298.

Skjetne, R., Jørgensen, U., & Teel, A. R. (2011). Line-of-sight path-following along regularly parametrized curves solved as a generic maneuvering problem. In *Proc. IEEE Conf. Decision & Control*, volume 50th Orlando, USA: Inst. Electrical and Electronics Engineers.

Teel, A. R., Moreau, L., & Nešić, D. (2003). A unified framework for input-to-state stability in systems with two time scales. *IEEE Trans. Autom. Ctrl.*, 48(9), 1526 – 1544.

Teel, A. R. & Praly, L. (2000). A smooth Lyapunov function from a class-$\mathcal{KL}$ estimate involving two positive semidefinite functions. *ESAIM: Control, Optim. and Calculus of Variations*, 5, 313–367.