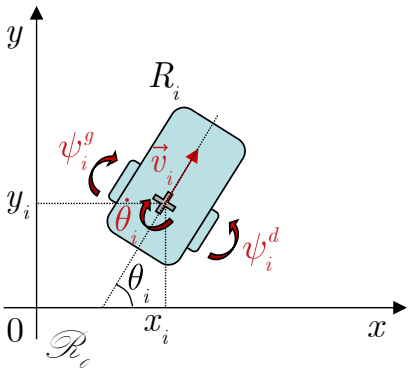
Benchmark Control of a Virtual Axle-Tree

I. Description of the benchmark

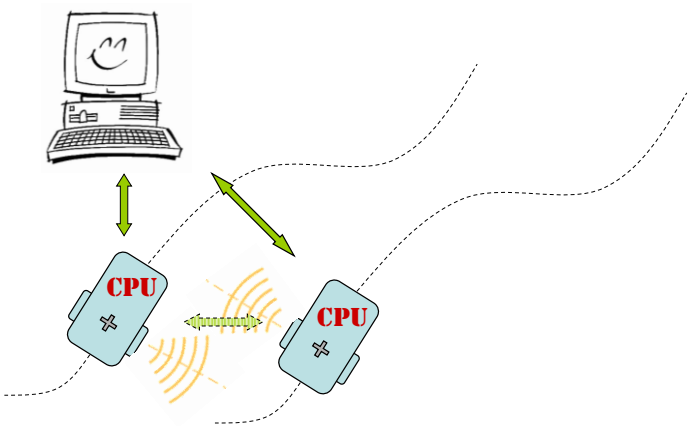
Physical setup

The idea is to drive two mobile robots that need to cooperate in order, for example, to manipulate a load too cumbersome and heavy to be manipulated by only one of the robot. It is called “virtual” axle-tree as there is no physical link between the two robots.

The system is based on:
- two tank-like mobile robot: two motored wheels on each side of the robot, driven in a differential way to deal with its orientation,



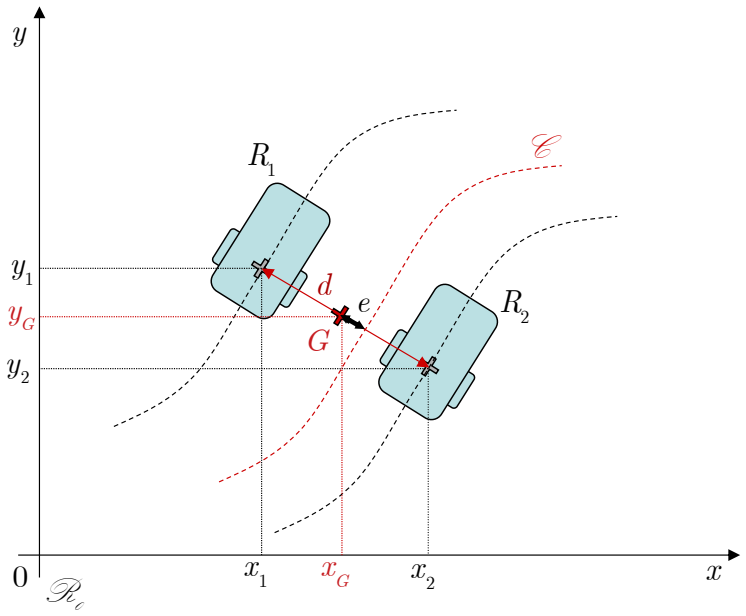
- a supervisor computer: that can measure the absolute position (*x*,*y*) and orientation ϑ of each robot thanks to cameras, and communicate by a wireless mean with each robot. **It is assumed here that robots cannot communicate between them.**



Communication troubles

Measurements of the absolute position of the robots is done thanks to a camera, therefore we can consider that measurement signals are available without delays or packet losses. But the control signals (their physical meaning will be defined after) computed by the supervisor have to be send through a wireless protocol (bluetooth or wifi). Thus here we will have to consider some communication disturbances (delays, packet losses…).

Control Objectives



The control objectives for the two robots (constitutive of the so-called virtual axle-tree) are:

- to drive the barycenter *G* (see figure above) of the two robots along a pre-defined trajectory,

- to ensure an constant interdistance between the two robots, the interdistance *d* being defined here as the euclidian norm between centers (x_1,y_1) and (x_2,y_2).

Modeling

Each robot is represented through a classic kinematic “unicycle” model

$$\begin{cases} \dot{x}_i = v_i \cos\ \theta_i \\ \dot{y}_i = v_i \sin\ \theta_i \\ \dot{\theta}_i = u_i^1 \\ \dot{v}_i = u_i^2 \end{cases} \quad (1)$$

The controlled outputs can be defined as follows:

$$\begin{cases} h_1 = x_G = \dfrac{x_1 + x_2}{2} \\ h_2 = y_G = \dfrac{y_1 + y_2}{2} \\ h_3 = \Delta x = x_1 - x_2 \\ h_4 = \Delta y = y_2 - y_1 \end{cases} \quad (2)$$

II. Control architecture and design

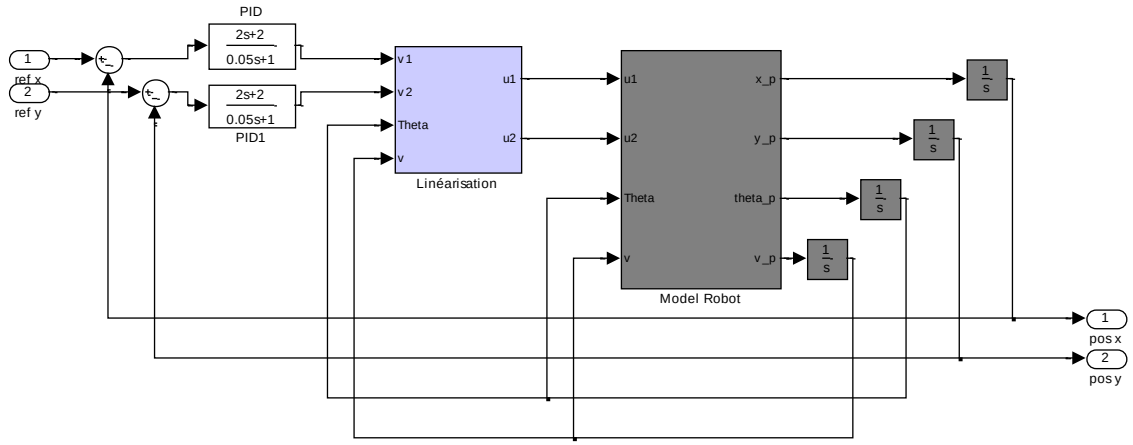
Control architecture

Several architectures could be considered. We propose here, to fit with the assumptions made in the theoretical results developed by Rongyao, to use a two layers nested architecture:

- Each robot is driven locally thanks to a static linearizing feedback control law, to linearize and decoupled the behavior of the longitudinal and lateral dynamics of the robots.

- A MIMO LQ controller (feedback + feedforward static gains) is implemented to drive the 4 controlled outputs (2).

Inner Loop in each robot: local control PD+Linearization:



Model Robot contains equation (1). Objective of this inner loop is first, thanks to the block « Linearization » to obtain the two decoupled integrator chain:

$$\begin{cases} \dfrac{x}{v_1} = \dfrac{1}{s^2} \\ \dfrac{y}{v_2} = \dfrac{1}{s^2} \end{cases} \quad (3)$$

This is based on very classical results on feedback linearization (non-linear control theory). Two new input signals (v_1,v_2) are thus defined. Next, two PD controllers (notice PID in the scheme above), are

implemented to make some pole placements on the two decoupled integrator chains in (3). Here PD are tuned as

$$v_i = (2+2s)e \tag{4}$$

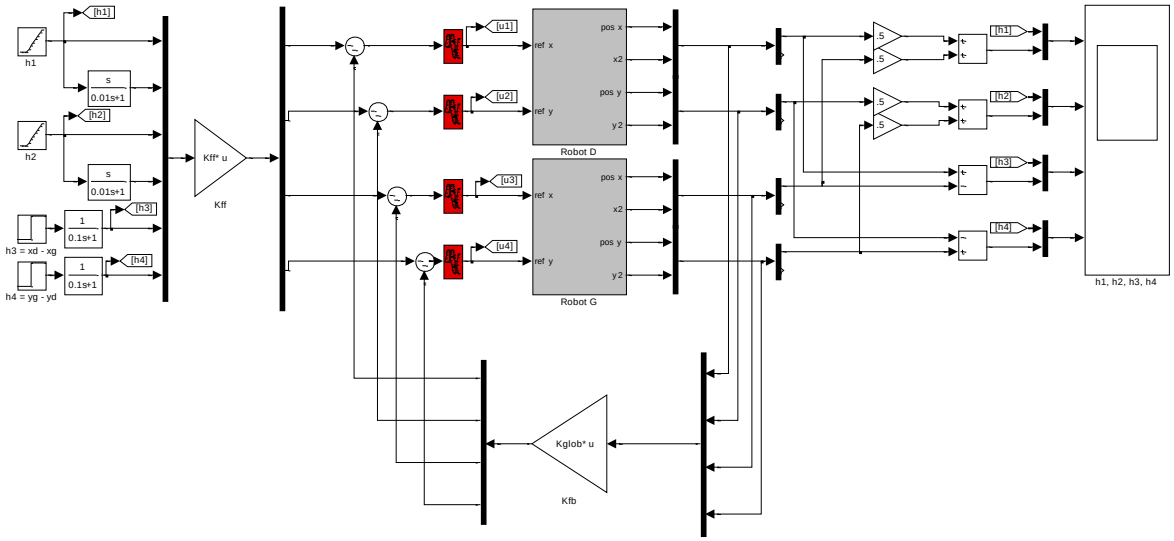
e is the error signal (difference between the measurement signal (position) *x* or *y* and the associated reference). The tuning was made to reach quite fast response without any overshoot. Finally, that leads to two decoupled second order system for one robot:

$$\begin{cases} \dfrac{x_i}{x_i^{ref}} = \dfrac{2+2s}{s^2+2s+2} \\ \dfrac{y_i}{y_i^{ref}} = \dfrac{2+2s}{s^2+2s+2} \end{cases} \tag{5}$$

Notation

In the Matlab code, the input signals $\left(x_i^{ref}, y_i^{ref}\right)$ of this 2 (transfer function) x 2 (robot) = 4 second order model are denoted (wx_i, wy_i) with *i* = *d* for "droite" (Right in French, *i.e.* robot n°2 in the first figure), or *g* for "gauche" (Left in French, *i.e.* robot n°1 in the first figure).

Outer Loop: MIMO H_2/LQ Controller:



The two blocks "Robot D" and "Robot G" are based on the inner closed loop in the previous figure.

Description of the architecture

The external loop is based on a feedback gain K_{fb} and a feedforward gain K_{ff}:

$$w = w_{ff} + w_{fb} \text{ with } \begin{cases} w_{fb} = -K_{fb}X = -K_{fb}\begin{bmatrix}\begin{bmatrix}x_1^1\\x_1^2\end{bmatrix}\\\begin{bmatrix}y_1^1\\y_1^2\end{bmatrix}\\\begin{bmatrix}x_2^1\\x_2^2\end{bmatrix}\\\begin{bmatrix}y_2^1\\y_2^2\end{bmatrix}\end{bmatrix} \\ w_{ff} = -K_{ff}X_{reference} = -K_{ff}\begin{bmatrix}\begin{bmatrix}x_{h1}^1\\x_{h1}^2\end{bmatrix}\\\begin{bmatrix}x_{h2}^1\\x_{h2}^2\end{bmatrix}\\x_{h3}\\x_{h4}\end{bmatrix} \end{cases} \quad (6)$$

with $\begin{bmatrix}x_i^1\\x_i^2\end{bmatrix}$ or $\begin{bmatrix}y_i^1\\y_i^2\end{bmatrix}$, i = 1,2 the states associated to the second order transfer function in (5), and $\begin{bmatrix}x_{hj}^1\\x_{hj}^2\end{bmatrix}$ or x_{hj} , j = 1, 2, 3, 4 the state of the exogenous model of the reference signal h_j^{ref} associated to the controlled output defined in (2).

Communication delays

As depicted in the figure above by the red boxes, the potential delays or packet losses can occur mainly on the 4 input signals $(wx\_i, wy\_i) = (x_i^{ref}, y_i^{ref})$ given that the supervisor where is implemented the MIMO controller is connected to the two robots through a wireless protocol.

Design methodology of K_{fb} and K_{ff}

The design of the feedback gain K_{fb} is obtained by solving the LQ problem considering:

- as state model the four decoupled second order system depicted in (5),

- as controlled outputs, the 4 outputs h_i defined in (2), with the same weighting coefficient q_i.

- as control inputs, the 4 input signals $(wx\_i, wy\_i) = (x_i^{ref}, y_i^{ref})$, with the same weighting coefficient r_i = 1.

The problem is solved in the continuous time framework (even if simulation is done in fixed-step).

The design of the feedforward gain K_{ff} is made by solving the Sylvester equations associated to the *Occultation Principle*; the idea is to find the feedforward gain leading to a reference for the control signals u_{ref} and a reference trajectory for the state x_{ref} making possible to conceal the influence of the exogenous (reference) signals.

This is necessary here as we assumed that the reference signals h_j^{ref} for j =1, 2 (reference for the x and y position of the barycenter G) follow a "Ramp" signal model (*i.e.* a double integral action), and h_j^{ref} for j =3, 4 and assumed as constant, *i.e.* they are modeled by an integral action.

This methodology is often applied in Nantes (*cf.* books of Ph. Chevrel or Ph. De Larminat)… If necessary I can provided the complete problem formulation and resolution.

III. What has to be done…

As far as I understand your theoretical developments, I think you have to defined and model the (MIMO ?) communication channels and associated packet losses that occur on the 4 control input signals $(wx\_i, wy\_i) = (x_i^{ref}, y_i^{ref})$, i = 1, 2.

It is not (yet ?) very clear for me in your file "*Numericalsimulation.m*"…

Next, it will be necessary to re-design the feedback gain, as well as the coder and decoder R and T that will be integrated in the MIMO channels.

An important point is that here, while all the simulation is done at fixed-step at 0.02*s*, both in the Simulink or Matlab version, the design of the feedback and feedforward gains were done in the continuous-time framework (see typically the second order model in (5)). What is the framework of your design methodology ?