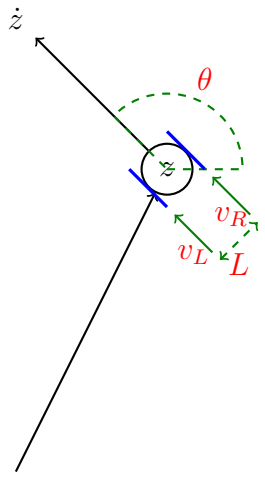
Robot line-tracking control

In this exercise, **Sysquake** will be used to study the control design of a differential-drive robot. A comparison is made between PD-control and design based on a discrete-time model. The circular robot with a left and a right wheel that can change speed independently, is illustrated below



The dynamics can be described using a position z in complex coordinates and a driving direction angle θ as

$$\begin{cases} \dot{\theta} = u \\ \dot{z} = ve^{i\theta} \end{cases}$$

The robot is controlled using two control signals, u and v. The speed v is manually chosen and the turning rate u is to be designed using feedback from the deviation from a line. The real robot is controlled in practice by combining the above two signals to give velocity references to the left and right wheels, respectively, according to

$$\begin{array}{l} v_R = v + \frac{L}{2}u \\ v_L = v - \frac{L}{2}u \end{array} \Rightarrow \dot{\theta} = (v_R - v_L)/L = u$$

Manual control

To get some feeling for the dynamics, open the file `Robot.sq` in Sysquake. Three windows will then show: slide bars for adjustment of parameters, the signals of the control in real-time, and an animation of the robot together with a piece-wise linear path that is supposed to be tracked. If the *Manual* is marked in the slide-bar window, the control signal u can be manipulated interactively. Try to control the robot manually. First select a modest speed v from the slide bar. By keeping u in the middle ($u = 0$) the robot is going straight. If you move u to the left then $u < 0$ and the robot is turning clockwise. With $u > 0$ it turns counter-clockwise. Once you get skilled you may increase the speed v. You will probably find it hard to track the path. This is because the dynamics is open-loop unstable and you have to stabilize the system which can be challenging since you can never rest by leaving the control signal constant.

In the window showing the signals in real-time, u is blue and piecewise constant (it is only changed at sampling instances) and the tracking error e ($= -y$) is red. The error is defined such that it is positive if the robot is to the right of the line and negative when it is to the left. For that reason it may look strange when the robot is moving close to perpendicular towards the line, since it is then not defined what is left or right.

Problem 1 — PD-control

Now select PD in the slide bar window. A PD-control structure is then used to control the robot. This is based on the continuous-time control structure

$$u(t) = P(t) + D(t), \qquad \begin{cases} P(t) = K_p e(t) \\ \frac{T_d}{N}\frac{dD(t)}{dt} + D(t) = K_p T_d \frac{de(t)}{dt} \end{cases}$$

but the implementation is in discrete time.

a) Find this discrete-time approximation based on backward-difference approximation, i.e. replace $\frac{d}{dt} \to \frac{1-\mathrm{q}^{-1}}{h}$, where h is the sampling period. More precisely, prove that the discretized PD-controller has the structure

$$u(k) = \frac{S(\mathrm{q}^{-1})}{R(\mathrm{q}^{-1})} e(k) = \frac{s_0 + s_1 \mathrm{q}^{-1}}{1 + r_1 \mathrm{q}^{-1}} e(k)$$

by calculating the controller parameters r_1, s_0 and s_1 as functions of K_p, T_d, N and h.

b) Try to find, in your opinion, good tuning of the PD-controller by changing K_p and T_d by trial-and-error.

c) The controller has a pole $p_c = -r_1$ and a zero $z_c = -s_1/s_0$. Where are these restricted to be positioned for positive PD-parameters (thus for $K_p, T_d, N, h > 0$)?

Problem 2 — Discrete-time control design

The continuous-time nonlinear dynamics can be linearized to

$$\frac{d^2y(t)}{dt^2} = v \cdot u(t)$$

Sampling of this system gives the discrete-time model

$$A(\mathrm{q}^{-1})y(k) = B(\mathrm{q}^{-1})u(k), \quad \frac{B}{A} = v\frac{h^2}{2}\frac{\mathrm{q}^{-1} + \mathrm{q}^{-2}}{(1-\mathrm{q}^{-1})^2}$$

The system is a double integrator and is therefore unstable. Let the controller gain be inversely proportional to the velocity v in order to make the closed-loop system (for turning) independent on chosen speed. Also, parameterize the controller using controller gain, pole and zero as

$$u(k) = \frac{S}{R}e = \frac{K}{v}\frac{1 - z_c\mathrm{q}^{-1}}{1 - p_c\mathrm{q}^{-1}}e(k)$$

In recursive form, it is implemented as

$$u(k) = p_c u(k-1) + (K/v)[e(k) - z_c e(k-1)]$$

Notice that the gain is inversely proportional to the velocity v making the closed-loop characteristic polynomial

$$A_c = AR + BS$$

independent on v ($B \propto v$ and $S \propto 1/v$).

a) Make a *dead-beat*-design, i.e. choose all closed-loop poles at the origin by letting $A_c = 1$ and calculate the controller parameters K, p_c and z_c. The sampling period is $h = 1$ s.

b) Choose *General feedback* in the slide window. Then tune the three controller parameters K, p_c and z_c according to dead-beat design and test the controller.

c) Why is it not possible to achieve dead-beat design with the PD-control structure?

Report

Document what you do and try to explain the behavior of the controlled robot. Remember that the dynamics that are simulated model the nonlinear nature of the robot. Thus, the simplified linear model that is described and used in the analysis above is only an approximation.