# Digital Control: Exercise 4

Xu Fei feixu12@student.hh.se

## 1. Manual control.

In this part, I just get some feeling for the dynamics. I open the file Robot.sq in Sysquake. Then I choose the option manual control, in this control system, $u$ is blue and piecewise constant, and the tracking error $e\,(-y)$ is red one, 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. Figure 1.1 shows the result I almost put the robot on the track (with 0.05 error on the left).

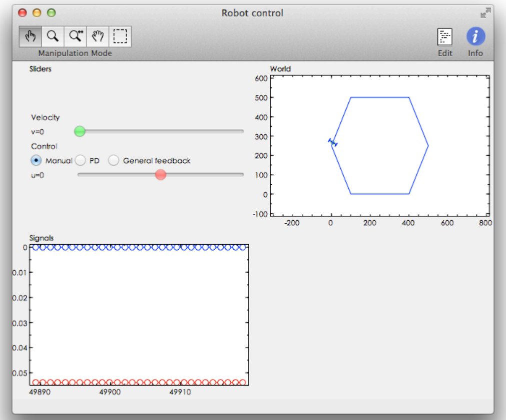


Figure 1.1 Initial the position of the robot on the track

Then I set $v = 0.973$ and let the robot try to follow the track. And I found it's extremely hard to use hand to fix the value. The result is shown in figure 1.2.

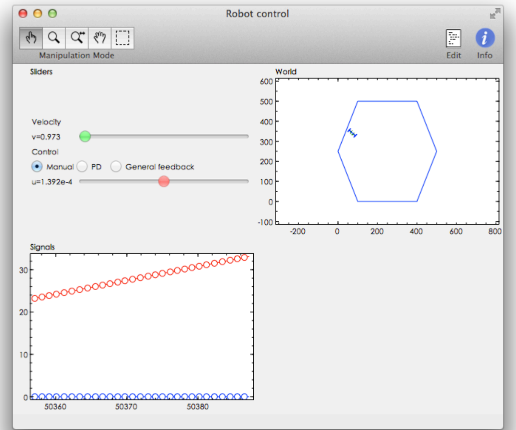


Figure 1.2 Use manual-control to control the robot follow the track

# 2. PD-control.

As we found it's impossible to use hand control the robot following the track interactively, for solve this problem, we choose PD-control instead of it. Now we select PD in the slider bar window. This controller is based on continuous-time control structure:

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

But the implementation is in discrete time.

a) As we know:

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

We replace $\frac{d}{dt} \to \frac{1-q^{-1}}{h}$, where $h$ is sampling period. Then we can get:

$$\frac{T_d}{N}\frac{1-q^{-1}}{h}D(k) + D(k) = K_p T_d \frac{1-q^{-1}}{h}e(k)$$

$$\to \begin{cases} D(k) = \frac{NK_pT_d(1-q^{-1})}{hN + (1-q^{-1})T_d}e(k) \\ P(k) = K_p e(k) \\ u(k) = K_p e(k) + \frac{NK_pT_d(1-q^{-1})}{hN + (1-q^{-1})T_d}e(k) \end{cases}$$

$$\to u(k) = \frac{K_p hN + K_p - K_pT_dq^{-1} + NK_pT_d(1-q^{-1})}{hN + (1-q^{-1})T_d}e(k)$$

$$\to u(k) = \frac{K_p hN + K_p + NK_pT_d - (K_pT_d + NK_pT_d)q^{-1}}{hN + T_d - T_dq^{-1}}e(k)$$

We want to prove that the discretized PD-controller has the structure:

$$u(k) = \frac{S(q^{-1})}{R(q^{-1})}e(k) = \frac{s_0 + s_1q^{-1}}{1 + r_1q^{-1}}e(k)$$

So then we can make

$$\to u(k) = \frac{\frac{K_phN + K_p + NK_pT_d}{hN + T_d} - \frac{(K_pT_d + NK_pT_d)}{hN + T_d}q^{-1}}{1 - \frac{T_d}{hN + T_d}q^{-1}}e(k)$$

Then we can easily get the parameters are:

$$s_0 = \frac{K_phN + K_p + NK_pT_d}{hN + T_d}, \quad s_1 = -\frac{(K_pT_d + NK_pT_d)}{hN + T_d}, \quad r_1 = -\frac{T_d}{hN + T_d}$$

b) Here, I choose the PD-control option and try to find the best value for $K_p$ and $T_d$. Finally I choose $K_p = 0.13$ and $T_d = 4.8$ as the parameter value.

The result is shown in figure 2.1.

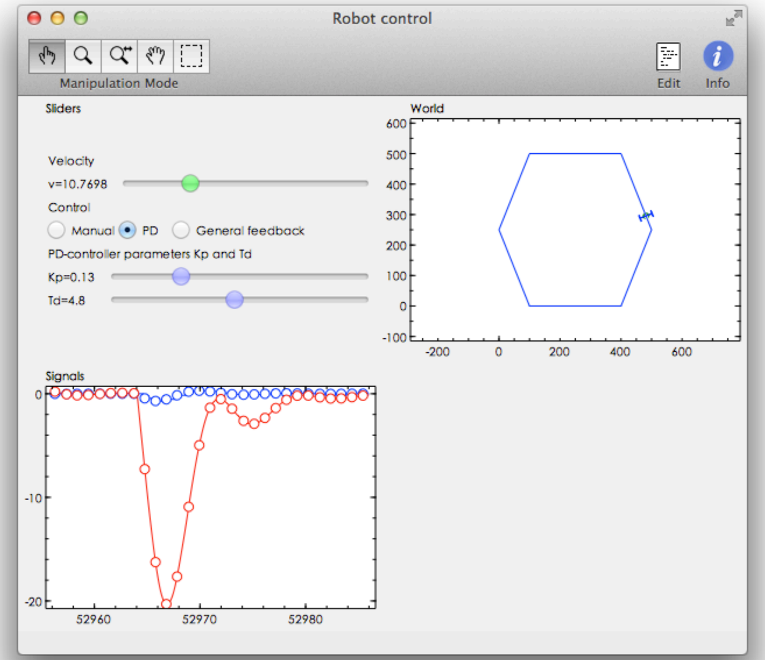


Figure 2.1 PD-control with $K_p = 0.13$ and $T_d = 4.8$

From the result, we can see the signal is quite smooth and doesn't have overshoot situation, also the robot can follow the track automatically.

c) From a), we have got:

$$s_0 = \frac{K_p hN + K_p + NK_p T_d}{hN + T_d}, \quad s_1 = -\frac{(K_p T_d + NK_p T_d)}{hN + T_d}, \quad r_1 = -\frac{T_d}{hN + T_d}$$

Then we know the controller has

$$p_c = -r_1 = \frac{T_d}{hN + T_d}$$

$$z_c = -\frac{s_1}{s_0} = \frac{(K_p T_d + NK_p T_d)}{K_p hN + K_p + NK_p T_d}$$

So if $K_p, h, N, T_d > 0$, then we can easily know that the restriction is:

$$0 < p_c < 1, 0 < z_c < 1.$$

## 3. Discrete-time control design

The continuous-time nonlinear dynamics can be linearized to:

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

Sampling of this system gives the discrete-time model:

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

Now the system is double integrator and unstable, so we need let the gain of controller inversely proportional to the velocity $v$ in order to make the closed-loop sys- tem (for turning) independent on chosen speed. The parameters of the controller using controller gain, pole and zero as:

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

In recursive form, it is implemented as:

$$u(k) = p_c u(k-1) + \left(\frac{K}{v}\right)[e(k) - z_c e(k-1)]$$

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

$$A_c = AR + BS$$

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 $h = 1s$.

As we known:

$$A = 2(1 - q^{-1})^2, \qquad B = v(h^2 q^{-1} + q^{-2})$$

$$S = \frac{K}{v}(1 - z_c q^{-1}), \qquad R = (1 - p_c q^{-1}),$$

$$A_c = AR + BS = 2(1 - q^{-1})^2(1 - p_c q^{-1}) + v(h^2 q^{-1} + q^{-2})\frac{K}{v}(1 - z_c q^{-1})$$

$$\rightarrow A_c = 2(1 - q^{-1})^2(1 - p_c q^{-1}) + K(h^2 q^{-1} + q^{-2})(1 - z_c q^{-1})$$

$$\rightarrow A_c = 2 - 4q^{-1} + 2q^{-2} - 2p_c q^{-1} + 4p_c q^{-2} - 2p_c q^{-3} + Kh^2 q^{-1} + Kq^{-2} - Kh^2 z_c q^{-2} - K z_c q^{-3}$$

$$\rightarrow A_c = 2 + (Kh^2 - 4 - 2p_c)q^{-1} + (2 + 4p_c + K - Kh^2 z_c)q^{-2} - (2p_c + K z_c)q^{-3}$$

$$\rightarrow A_c = 2 + (K - 4 - 2p_c)q^{-1} + (2 + 4p_c + K - K z_c)q^{-2} - (2p_c + K z_c)q^{-3}$$

In order to make a *dead-beat-design*, $A_c$ should be constant, so we can get:

$$\begin{cases} K - 4 - 2p_c = 0 \\ 2 + 4p_c + K - K z_c = 0 \\ 2p_c + K z_c = 0 \end{cases}$$

$$\rightarrow \begin{cases} K = 2.5 \\ p_c = -0.75 \\ z_c = 0.6 \end{cases}$$

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.

Then in *General feedback* mode, I set $K = 2.5$, $p_c = -0.75$ and $z_c = 0.6$. At first I set speed $v = 11.3216$ and the result is shown in figure 3.1. And then I change the speed $v$ to $v = 21.28$, the result is shown in figure 3.2.

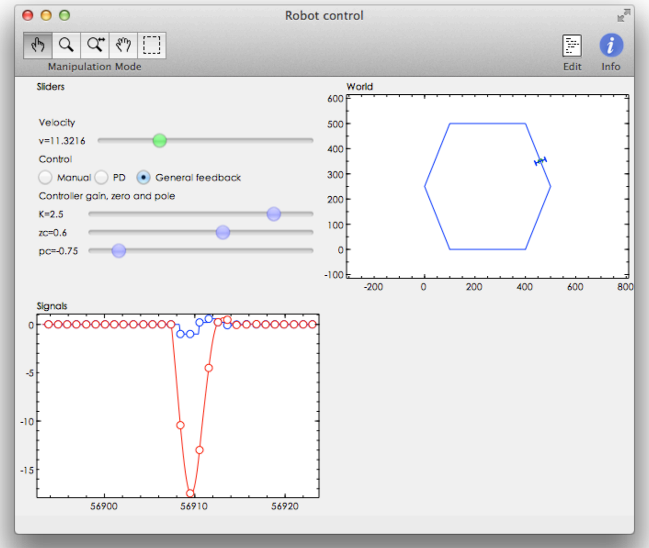


Figure 3.1 General feedback $K = 2.5,\ p_c = -0.75,\ z_c = 0.6$ and $v = 11.3216$

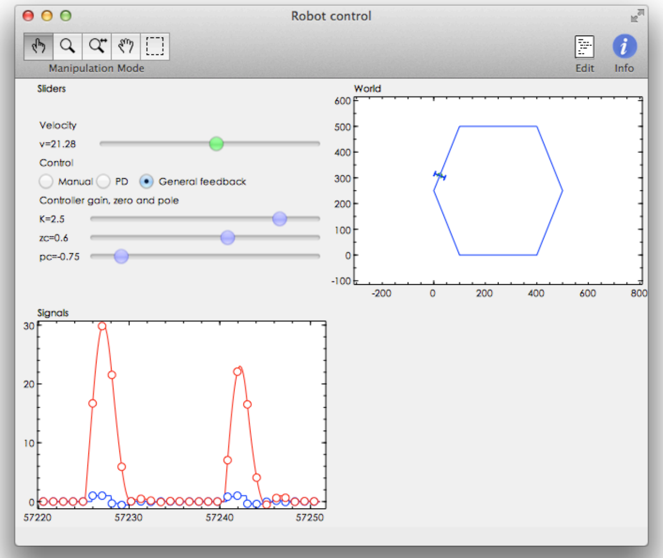


Figure 3.2 General feedback $K = 2.5,\ p_c = -0.75,\ z_c = 0.6$ and $v = 21.28$

From the result we can see the performance has improved a lot then what we have done in 2b). Now, even we change a lot of the speed, the performance is almost the same when the robot is turning.

c) From part 1c), we already known that the restriction is:

$$0 < p_c < 1, 0 < z_c < 1.$$

But, for *dead-beat-design*, $p_c$ will be negative, so it's impossible to achieve dead-beat design with the PD-control structure.