4. Notion of Realisable Reference

The term *realisable reference* (w^r) was first mentioned in chapter 3.1.1.2. In this chapter we will show how powerful tool the realisable reference is when comparing different anti-windup methods. Realisable reference is also useful when trying to understand windup and how to avoid it. We will show how to calculate the realisable reference for every mentioned controller and will also make some examples to illustrate the proposed anti-windup structures.

4.1. Realisable Reference for PID Controllers

4.1.1. Realisable Reference for AW Methods

To see the difference between AW methods clearly, the realisable reference signal (w^r) is introduced (see section 3.1.1.2). Let us consider to have such reference w^r instead of w that *resulting controller output u would be the same as realised control variable* u^r *when using reference w*. Such reference is called realisable reference (w^r).

To illustrate the above definition, we have used an example shown in Fig 4.2. Dashed line represents u and full line represents u^r when using reference w (see Fig. 4.1). Fig. 4.3 shows u and u^r when w^r is applied to controller input. We can see u and u^r become the same and equal to u^r in Fig. 4.2.

When using w^r, u^r *would stay the same as when using w*, so also the *process output (y) would not change*. The consequence of equal u and u^r when using the reference w^r is that we can *remove the limitation LIM*. Fig. 4.4 represents the equivalent of Fig 4.1 when using w^r instead of w.

Note that the realisable reference (w^r) can not be computed a priori, but only a posteriori. Therefore it is only a ***tool*** which helps us to understand different anti-windup techniques.

As it was mentioned, u^r and y in Fig 4.1 are the same as in Fig. 4.4, respectively. Moreover, Fig 4.4 *does not include any implicit limitation. The limitation is hidden in the realisable reference* w^r. Because the second scheme has no limitation (it is linear), it can be supposed that ***y tracks*** w^r ***instead of w***.

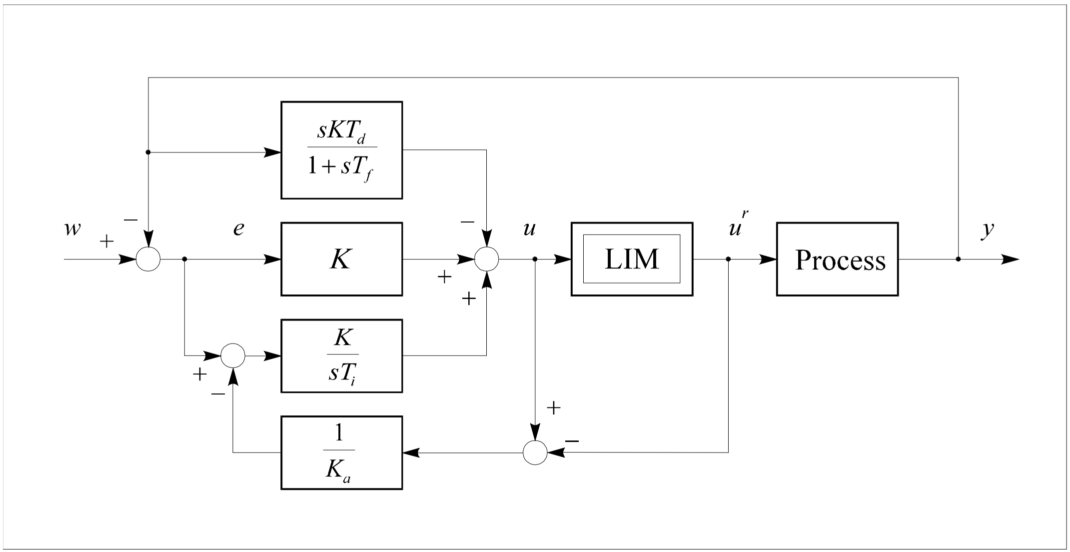

Fig. 4.1. PID controller with anti-windup

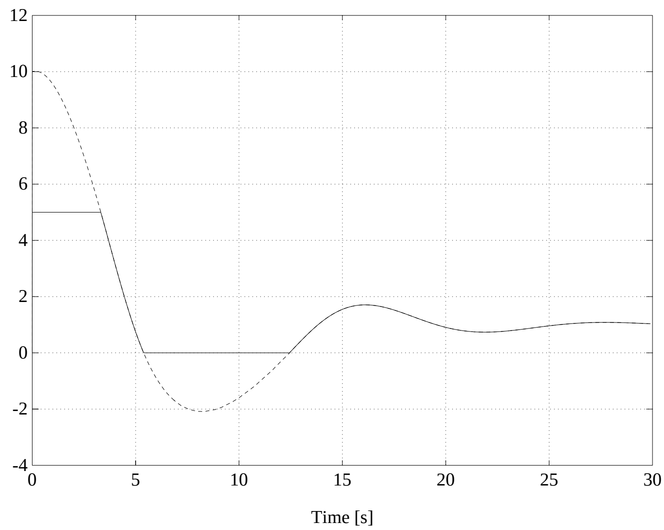

Fig. 4.2. The reference w at controller input;

__ Process input (u^r), -- Controller output (u)

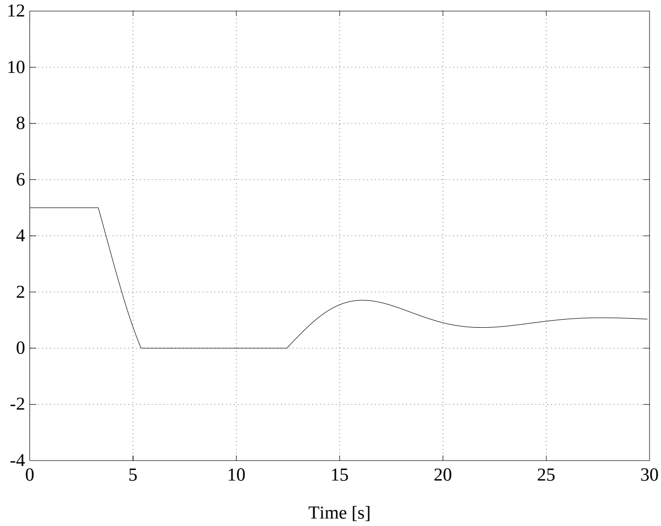

Fig. 4.3. The realisable reference w^r at controller input;

__ Process input (u^r), -- Controller output (u)

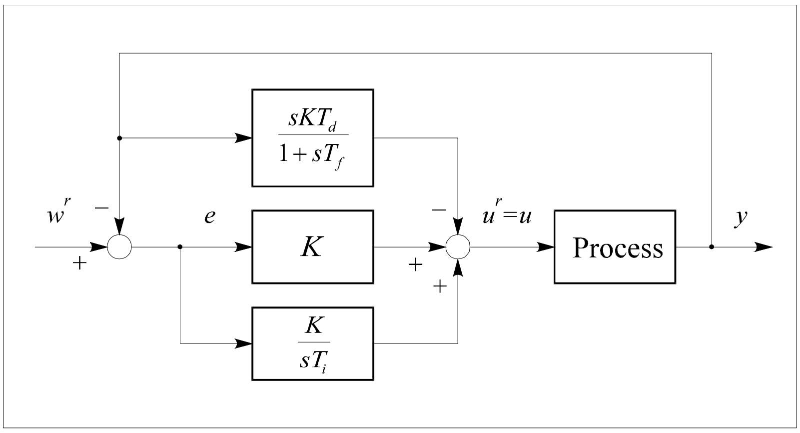

Fig. 4.4. The equivalent scheme of Fig. 4.3. from process' viewpoint

Originally, our goal is to make *y* to track *w*, but *y* actually tracks w^r due to the input limitations. Thus, we would like to have such a realisable reference w^r *that it will be as close as possible to w*.

From the definition of the realisable reference (see Fig. 4.4), we have

$$U^r = K\left(1+\frac{1}{sT_i}\right) W^r - K\left(1+\frac{1}{sT_i}+\frac{sT_d}{1+sT_f}\right)Y \tag{4.1}$$

what yields

$$W^r = \frac{1+sT_i+\dfrac{s^2T_iT_d}{1+sT_f}}{1+sT_i}Y + \frac{sT_i}{K\left(1+sT_i\right)}U^r \tag{4.2}$$

During the limitation, u^r is the same whatever anti-windup method is used. It is valid also for y if the initial conditions are the same. Hence, from equation 4.2 it can be seen that during limitation, *w^r is the same for all anti-windup* (also for bumpless and conditioned transfer) *methods* (the same for whatever K_a). From the linear feedback anti-windup scheme (Fig. 4.1), we have:

$$U = K\left(1+\frac{1}{sT_i}\right) W - K\left(1+\frac{1}{sT_i}+\frac{sT_d}{1+sT_f}\right)Y + \frac{K}{sT_i}\frac{U^r-U}{K_a} \tag{4.3}$$

Subtracting u (4.3) from u^r (4.1), we can calculate w^r as

$$W^r = W + \frac{1}{K_a}\frac{1+s\dfrac{K_aT_i}{K}}{1+sT_i}\left(U^r-U\right) = W + G_w\left(U^r-U\right) \tag{4.4}$$

Now, w-w^r is expressed as a function of u-u^r. Normally, G_w is a *dynamic* transfer function with one pole and zero. So w^r will not become the same as w at the moment when the controller leaves the limitation (u^r=u). We would like to have w^r as close as possible to w. That can be done by tuning K_a. In the case K_a=K, which corresponds to the ***conditioning technique*** (see chapter 3.1.1.2), G_w becomes a static gain:

$$G_w = \frac{1}{K} \tag{4.5}$$

and we obtain

$$w^r = w + \frac{u^r - u}{K} \quad . \tag{4.6}$$

Therefore at the instant when controller leaves the limitation ($u^r = u$), w^r *becomes* w.

Another important conclusion can be made from the last statement. It means, process output (y) *will follow the reference (w) at the instant when system comes out of the limitation*. It also means *system will actually feel no consequences related to the limitation after it leaves it*.

All other methods ($K_a \neq K$) give such w^r that will not become the same as w at the instant when system comes out of the limitation. It means, process (y) will not follow the reference (w) after system leaves the limitation. Consequently such anti-windup system feels the consequences related to the limitation also when it is no more present and therefore the effect of windup was not reduced totally. If $K_a < K$, during the limitation, the integrator is "braked" too much and the opposite is valid if $K_a > K$.

Therefore, the suitable solution for anti-windup is to choose $K_a = K$.

The ***non-linear*** AW method (*conditional integration*), if using PI controller, can be translated into a linear form by replacing (see chapter 3.1.2):

$$K_a = \begin{cases} \dfrac{u - u^r}{e} \; ; & u^r \neq u \\ const. \; ; & u^r = u \end{cases} \tag{4.7}$$

If the initial condition of the integral term is zero and we suppose only one (e.g. upper) limitation happens, we can calculate K_a. When reference changes, process goes toward limitation and input of the integral term becomes zero. From that instant on (till process is limited), a controller has only proportional part:

$$u = Ke \tag{4.8}$$

During limitation, expressions (4.7) and (4.8) give

$$K_a = \frac{u - u^r}{e} = K - \frac{u^r}{e} \tag{4.9}$$

what leads to

$$0 < K_a < K \tag{4.10}$$

We can expect that the response of conditional integration would lay between the incremental algorithm and conditioning technique during the limitation if the initial condition of the integral term would be zero and only one limitation happens.

For PID controller, the situation is more complicated because of derivative term. If supposing the same constraints in calculating K_a (initial condition of the integral term is zero and limitation happens only once), we can see that during limitation u becomes:

$$u = Ke - y_d \tag{4.11}$$

Because u^r is limited, it is absolutely smaller than u and can be calculated as

$$u^r = LIM(u) = \alpha u\,; \quad 0 < \alpha \le 1 \tag{4.12}$$

When inserting (4.11) and (4.12) into (4.7), we obtain

$$K_a = \frac{u - u^r}{e} = \left(K - \frac{y_d}{e}\right)(1-\alpha) \tag{4.13}$$

During limitation if system fits expression

$$y_d > -\frac{\alpha}{1-\alpha} Ke \tag{4.14}$$

what is valid for all minimal phase and some non-minimal phase processes *when process is limited* (process input u^r is constant or is constantly increasing). Than, if inserting (4.14) into (4.13), we have

$$0 < K_a < K \tag{4.15}$$

If we take into account that *during limitation u is* ***positive***, and inserting (4.11) into (4.13), we can get the expression which is valid for all processes and PID controllers if initial condition of the integral term is zero and only one limitation happens:

$$K_a > 0 \tag{4.16}$$

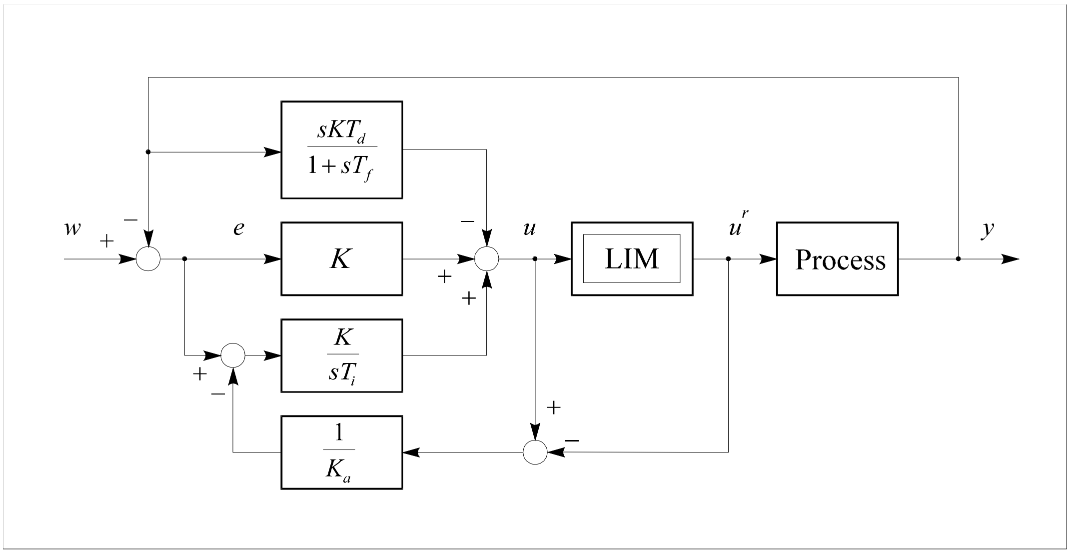

Fig. 4.5. Linear representation of the non-linear scheme (Fig. 3.6)

Hence, the response of the conditional integration will usually lay between the response of incremental algorithm and conditioning technique (4.15) if initial condition of the integral term is zero and one limitation happens. In general, K_a is variable with unpredicted value (see Fig. 4.5).

To compare mentioned AW algorithms, we make a simulation with process

$$G_{PR} = \frac{1}{(1+8s)(1+4s)} \tag{4.17}$$

and controller

$$K = 20,\ \ T_i = 30s,\ \ T_d = 1s,\ \ T_f = 0.1s \tag{4.18}$$

The input is subjected to the following limits:

$$U_{\max} = 2,\ U_{\min} = 0 \tag{4.19}$$

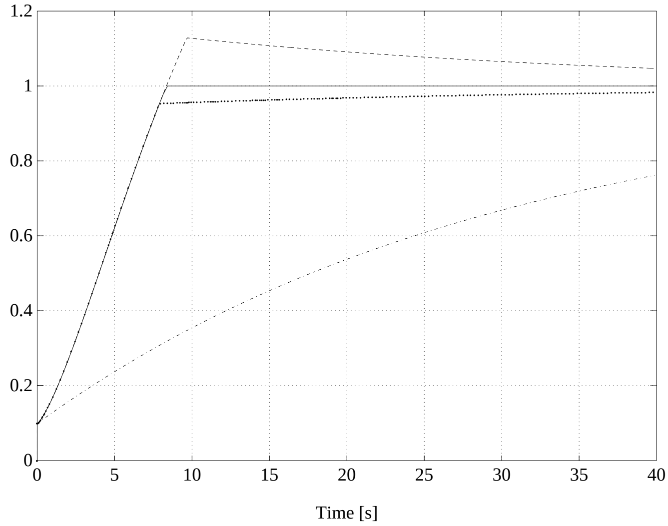

Fig. 4.6. Realisable reference (w^r): __ Conditioning technique,
-- Without AW, -.- Incremental algorithm, ... Conditional integration

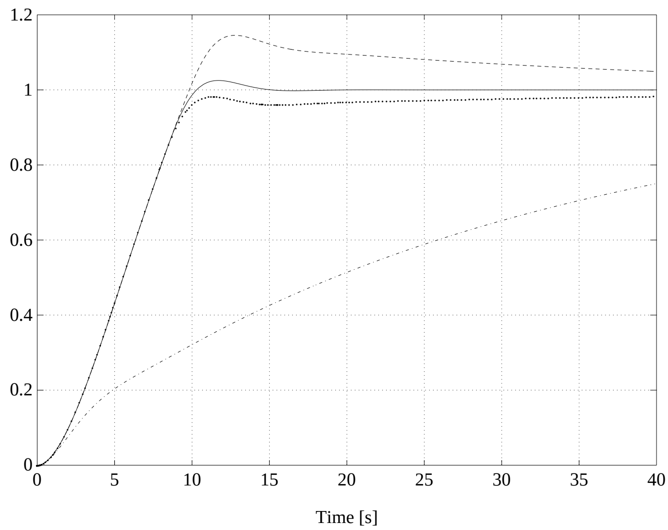

Fig. 4.7. Process output (y): __ Conditioning technique,
-- Without AW, -.- Incremental algorithm, ... Conditional integration

Figures 4.6 and 4.7 show the difference between the described anti-windup algorithms. The reference w goes from 0 to 1 at the time origin. It is clearly seen from Fig. 4.6 that the conditioning technique gives a w^r which is the closest to w. Fig. 4.7 demonstrates that process output (y) tracks w^r instead of w. It is also seen that the response of the conditional integration lays between the responses obtained with incremental algorithm and with the conditioning technique.

4.1.2. Realisable Reference for BT and CT Methods

The ***realisable reference*** (w^r) for bumpless transfer (BT) and conditioned transfer (CT) methods is defined in the same way as in section 4.1.1. So Figures 4.8 and 4.9 represent the equivalent schemes from the process viewpoint. Note that the realisable reference is defined for *all time* (before and after switching). It means that applying w^r to controller instead of w causes the controller output u to be the same as u^r.

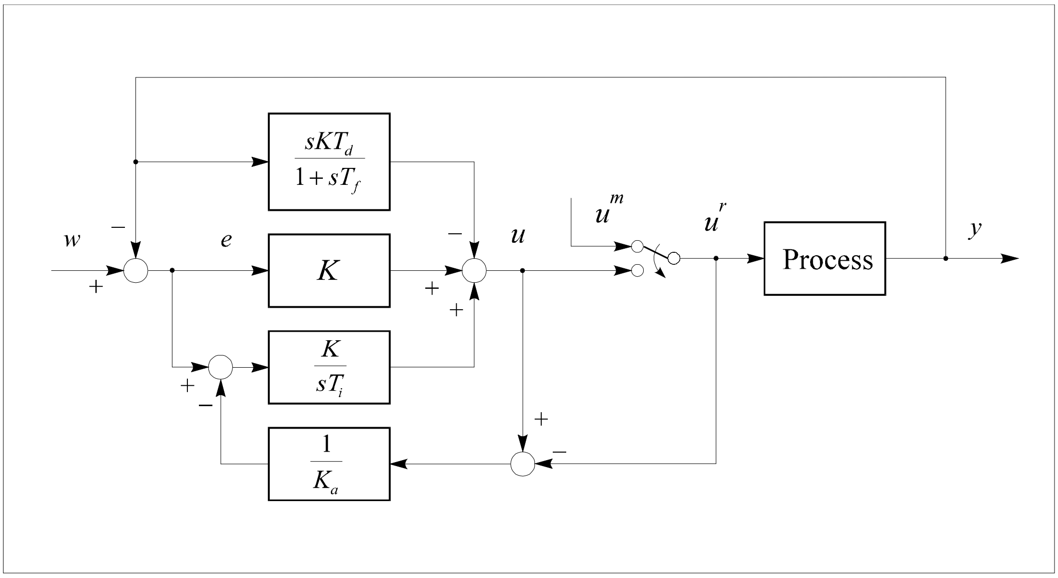

Fig. 4.8. Switching between manual and automatic mode;
the reference w at controller input

Conditioned transfer means that switching from manual to automatic control will cause the controller to make y to follow reference w with the same dynamics as for the closed-loop step response, but y in fact tracks w^r (Figure 4.9). After switching from manual to automatic mode (u^r=u), we want w^r=w. The only way to do that at the instant of switching is to use the *conditioning technique* (K_a=K) (4.6). So *the best conditioned transfer is achieved*. It also means that controller *will not feel the consequences of* *manual mode when it switches again to automatic mode*. If using other anti-windup methods ($K_a \neq K$), process (y) will not follow the reference w after switching to manual mode. System feels the consequences of manual mode and problems are not reduced totally.

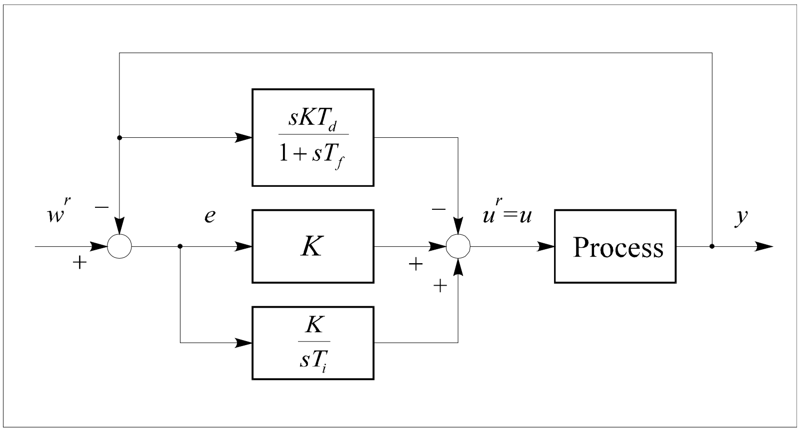

Fig. 4.9. Switching between manual and automatic mode;
the realisable reference w^r at controller input

Now, we can see, the *only anti-windup (conditioned transfer) method which assures no consequences of manual mode when switching back to automatic mode is that which has a static relation between w-w^r and u-u^r*. This method is a *conditioning technique*, which gives the relation represented by equation 4.6.

Therefore, derivation of conditioned transfer methods for other kind of controllers will not be realised separately. The solution is always such kind of anti-windup method, which assures a static relation between w-w^r and u-u^r. It means, when switching to automatic mode, u^r becomes equal to u what results in w^r is the same as w. Process (y) will follow the reference w from the instant of switching to automatic mode.

Usually, conditioned transfer using the conditioning technique will produce a *jump* at the input of the process, because w^r will jump to w when the switching occurs. This is normal, ***as a jump always occurs when the reference has a step change***.

In practice, as in theoretical work, it is often considered that the best solution is to use bumpless methods because of their effect of bump transfer (see chapter 2.4). The goal is often, *because of our fear against bump*, to reduce the change at process input to the minimum when switching from manual to automatic mode. In most cases that kind of solution is not optimal, because it produces inferior tracking performance (long settling time of the process). If we want that, when in automatic mode, *process tracks reference w, let controller do that* by using conditioning technique.

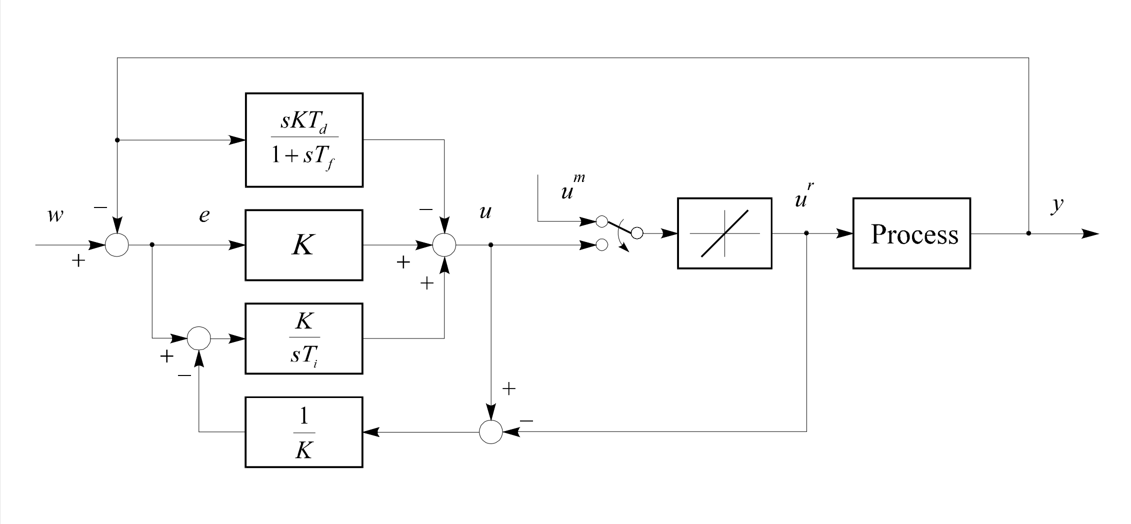

Fig. 4.10. Conditioned transfer with bumpless feature

Yet if a jump is not tolerable, we can either switch from manual to automatic when u is close to u^m (by driving y close to w before switching) or add the rate limitation at the process input (Fig. 4.10).

Incremental algorithm ($K_a \rightarrow 0$) will not produce a jump of u^r (bump) at the time of switching (***bumpless transfer***), because u is equal to u^r (section 3.1.1.3). However, the resulting *tracking performance will not be as good as that produced by the conditioned transfer*.

To support the above arguments, we have made a simulation with process

$$G_{PR} = \frac{1}{(1+10s)^2}, \tag{4.20}$$

controller 1:

$$K = 20, \ T_i = 40.8s, \ T_d = 1.16s, \ T_f = 0.116s \tag{4.21}$$

and controller 2:

$$K = 10, \ T_i = 40s, \ T_d = 1s, \ T_f = 0.1s \tag{4.22}$$

At t=0s, we switch from controller 1 (with reference w_1=0) to controller 2 (with reference w_2=0.1) (see Fig. 4.11). At t=40s controller 1 is switched into closed-loop again. Figures 4.12 to 4.14 show the results obtained when switching between two different controllers with different references.

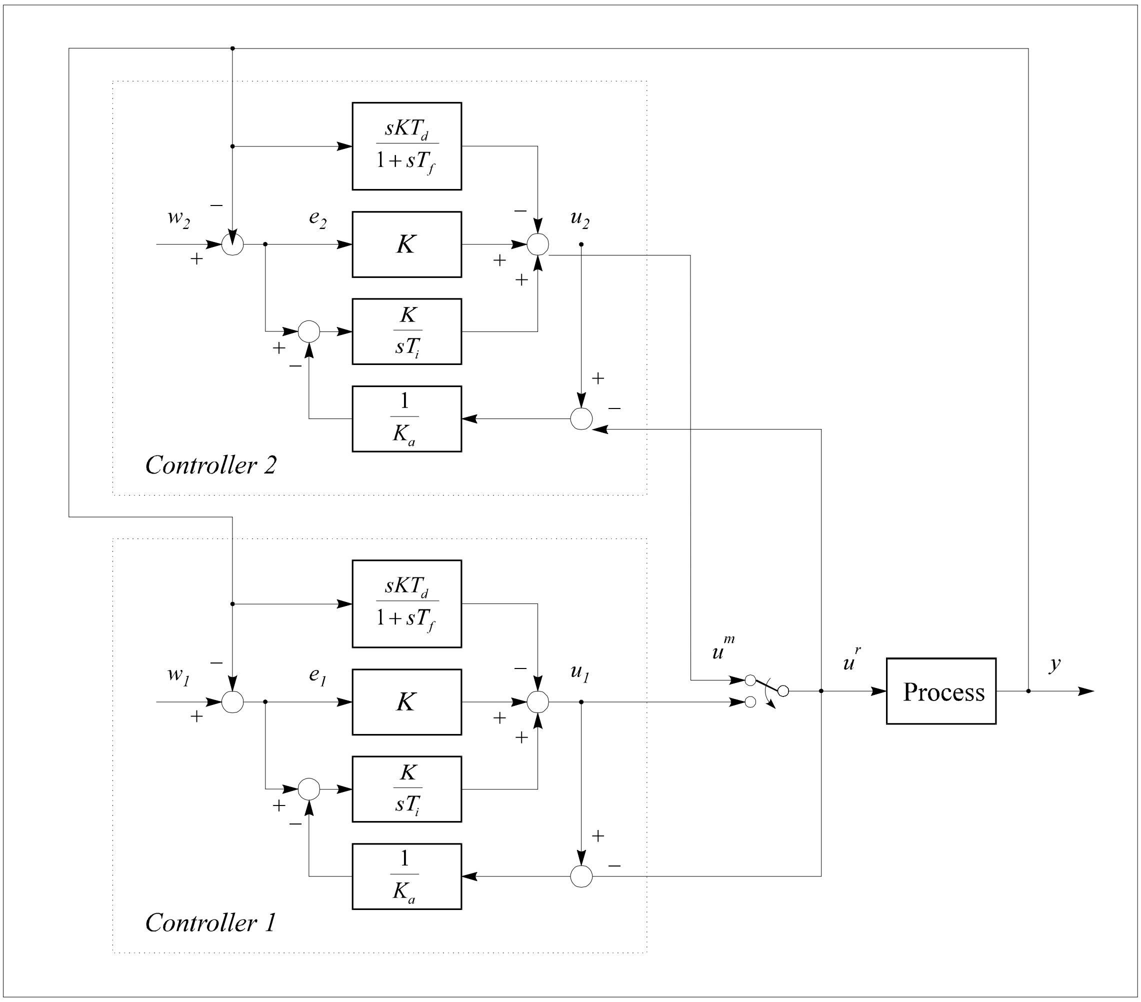

Fig. 4.11. Switching between two controllers

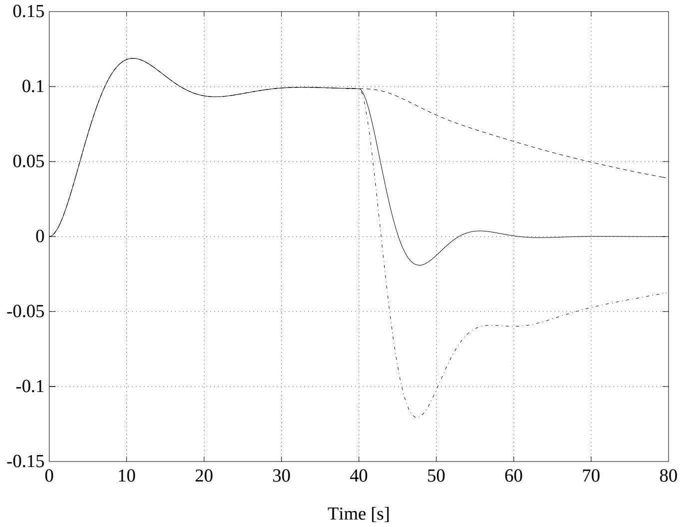

Fig. 4.12. Process output (y);
__ Conditioning technique, -- Incremental algorithm, -.- Without protection

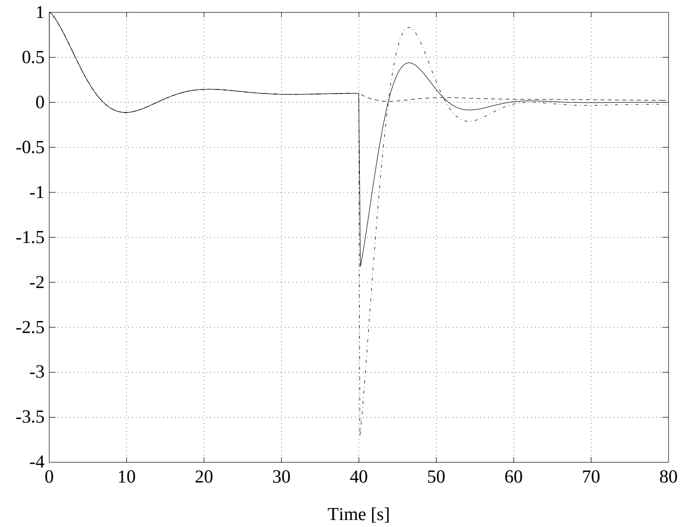

Fig. 4.13. Process input (u^r);
__ Conditioning technique, -- Incremental algorithm, -.- Without protection

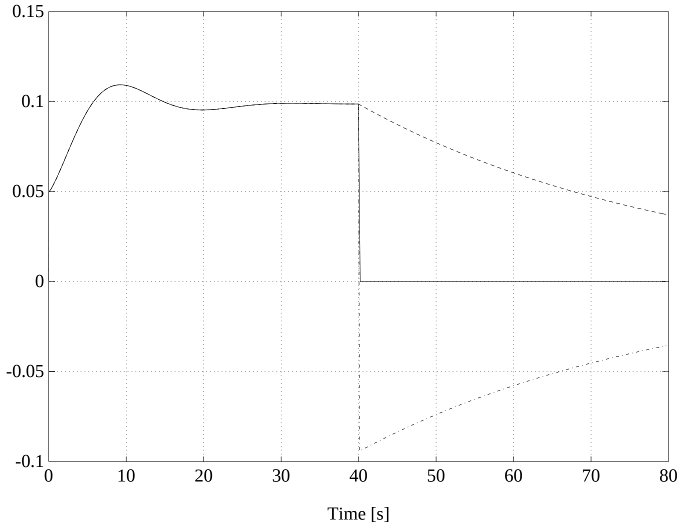

Fig. 4.14. Realisable reference (w^r);
__ Conditioning technique, -- Incremental algorithm, -.- Without protection

We can see that the conditioning technique gives the best tracking performance. Figures 4.15 and 4.16 show the results obtained when using a rate limitation at the process input. In this case, the conditioning technique produces no bump at the process input.

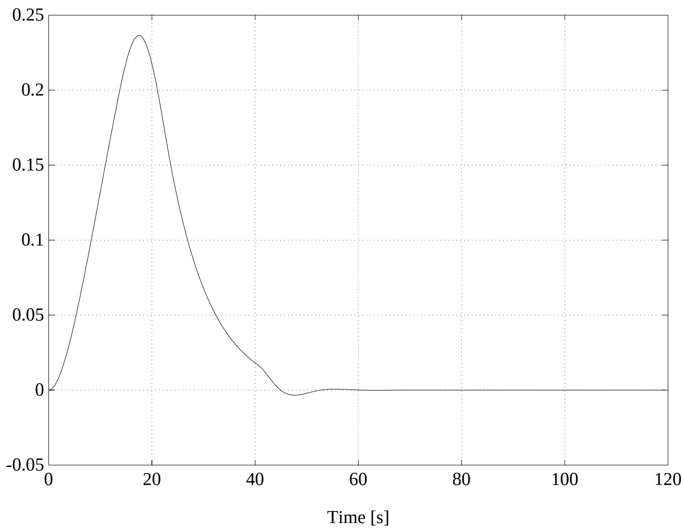

Fig. 4.15. Process output (y) for conditioned transfer with the rate limitation (v_{max}= 0.2, v_{min}= -0.2) (see Fig. 4.10)

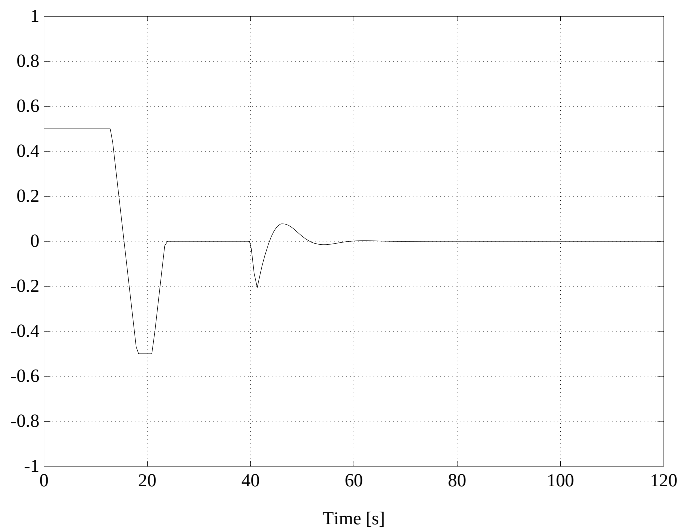

Fig. 4.16. Process input (u^r) for conditioned transfer with the rate limitation (v_{max}= 0.2, v_{min}= -0.2) (see Fig. 4.10)

4.2. Realisable Reference for Generalised PID Controllers

Generalised PID controller with anti-windup is expressed by equation (3.24) (see chapter 3.2)

$$U = K\left[\beta + \frac{1}{sT_i} + \gamma\frac{sT_d}{1+sT_f}\right]W - K\left[1 + \frac{1}{sT_i} + \frac{sT_d}{1+sT_f}\right]Y - K\left[\frac{1}{K_{a1}}\frac{1}{sT_i} + \frac{1}{K_{a2}}\gamma\frac{sT_d}{1+sT_f}\right]\left(U - U^r\right) \quad (4.23)$$

From the definition of the realisable reference (w^r) (see chapters 3.1.1.2 and 4.1.1), we can define U^r as

$$U^r = K\left[\beta + \frac{1}{sT_i} + \gamma\frac{sT_d}{1+sT_f}\right]W^r - K\left[1 + \frac{1}{sT_i} + \frac{sT_d}{1+sT_f}\right]Y \quad (4.24)$$

Subtracting (4.24) from (4.23), we obtain

$$\left(U-U^r\right)\left[1+K\left[\frac{1}{K_{a1}}\frac{1}{sT_i}+\frac{1}{K_{a2}}\gamma\frac{sT_d}{1+sT_f}\right]\right]=K\left[\beta+\frac{1}{sT_i}+\gamma\frac{sT_d}{1+sT_f}\right]\left(W-W^r\right) \quad (4.25)$$

Realisable reference can be expressed from (4.25) as

$$W^r=W+\frac{1+K\left[\frac{1}{K_{a1}}\frac{1}{sT_i}+\frac{1}{K_{a2}}\gamma\frac{sT_d}{1+sT_f}\right]}{K\beta\left(1+\frac{1}{\beta}\frac{1}{sT_i}+\frac{\gamma}{\beta}\frac{sT_d}{1+sT_f}\right)}\left(U^r-U\right)=W+G_{W^r}\left(U^r-U\right) \quad (4.26)$$

Anti-windup system should behave such that realisable reference will be the same as the actual reference (w) at the instant when process comes out of the limitation (see chapters 4.1.1 and 4.1.2). That can happen only if G_{W^r} is a static gain. From (4.26) we can see that it happens in the case when

$$K_{a1}=K_{a2}=K\beta \;\Rightarrow\; w^r=w+\frac{u^r-u}{K\beta} \quad (4.27)$$

The graphical representation of expression (4.23) and solution (4.27) is presented in Fig. 4.17.

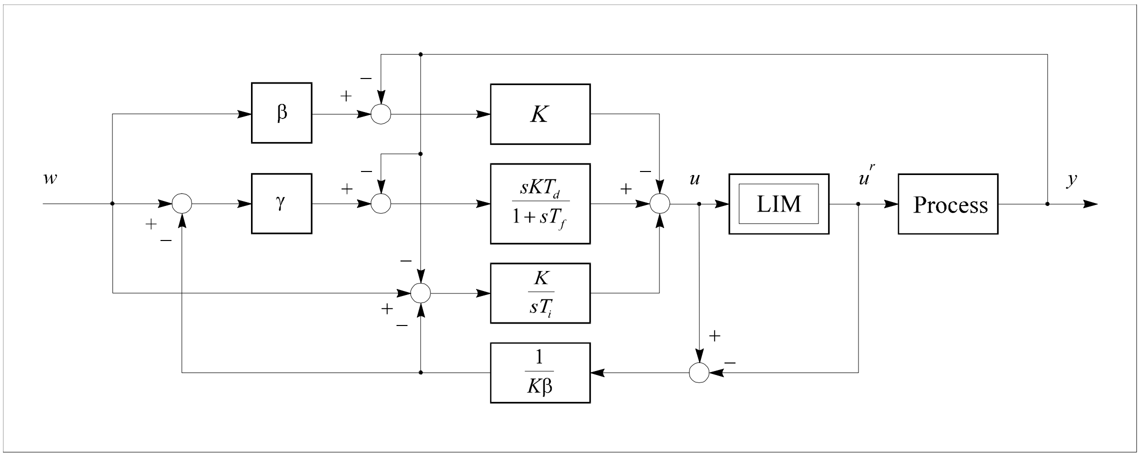

Fig. 4.17. The solution of AW for generalised PID controller. Not useful in digital implementation.

From Fig. 4.17 we can see that some kind of algebraic loop can exist through derivative (*D*) part and anti-windup feedback compensator. In analog controller this phenomenon does not exist, but can be problematic when making digital simulations in *SIMULINK*.

However, there exist solution of K_{a1} and K_{a2} which does not cause algebraic loop:

$$K_{a1} = K\left(\beta + \gamma \frac{T_d}{T_f}\right)$$
$$K_{a2} = -sKT_f\left(\beta + \gamma \frac{T_d}{T_f}\right) \Rightarrow \quad w^r = w + \frac{u^r - u}{K\left(\beta + \gamma \frac{T_d}{T_f}\right)} \tag{4.28}$$

This solution corresponds to ***conditioning technique***.

The graphical representation of expression (4.28) is presented in Fig 4.18.

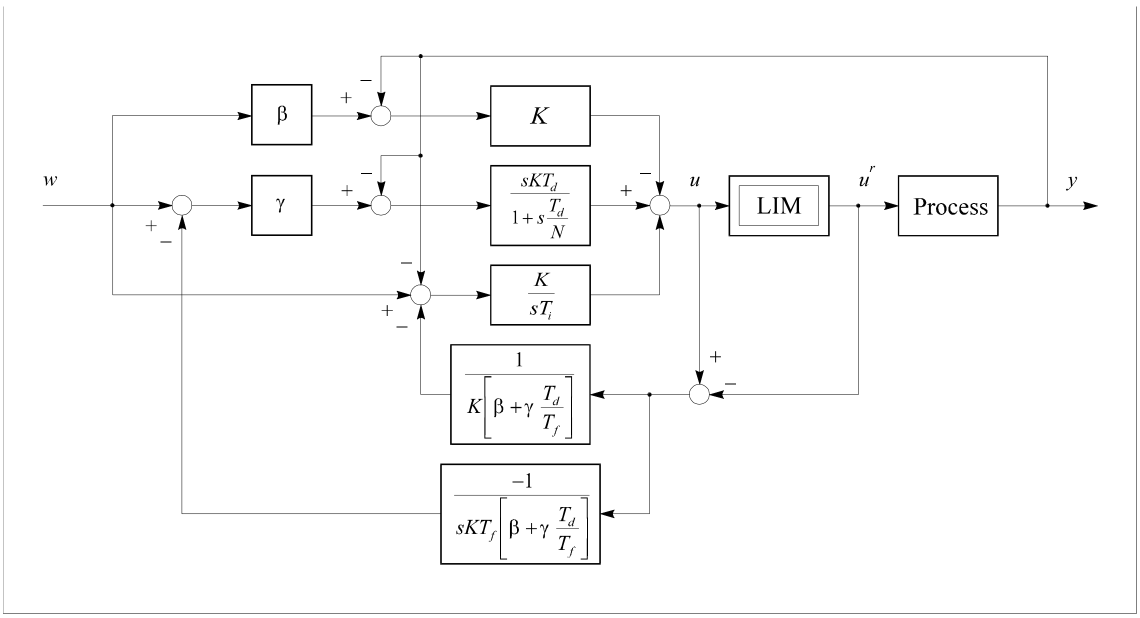

Fig. 4.18. The solution of AW for generalised PID controller (conditioning technique) useful for analog and digital implementation.

To depict above solution, we made a simulation with process

$$G_{PR} = \frac{1}{(1+10s)(1+4s)(1+2s)} \tag{4.29}$$

and controller

$$K = 10,\ \ T_i = 6s,\ \ T_d = 2s,\ \ T_f = 0.2s,\ \ \beta = 0.5,\ \ \gamma = 0.2 \tag{4.30}$$

The input was subjected to the following limits:

$$U_{\max} = 2,\ \ U_{\min} = -2,\ \ v_{\max} = 2s^{-1},\ \ v_{\min} = -2s^{-1} \tag{4.31}$$

The resulting system response is shown in Figures 4.19 to 4.21.

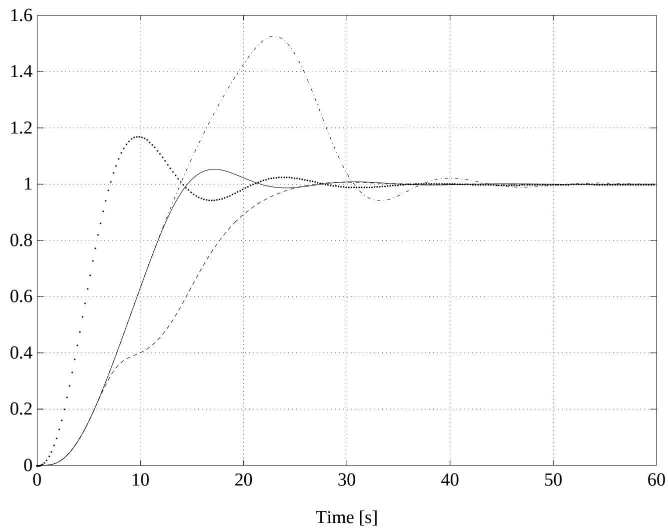

Fig. 4.19. Process output (y);
__ Conditioning technique,
-- Incremental algorithm,
-.- No protection against windup,
... Unlimited response

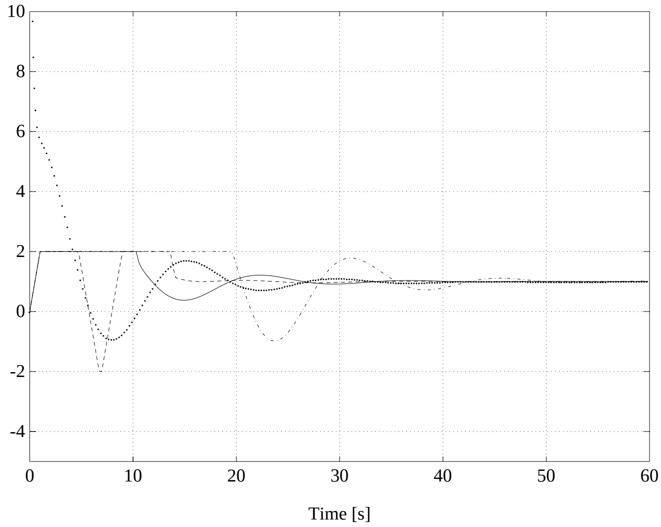

Fig. 4.20. Process input (u^r);
__ Conditioning technique,
-- Incremental algorithm,
-.- No protection against windup,
... Unlimited response

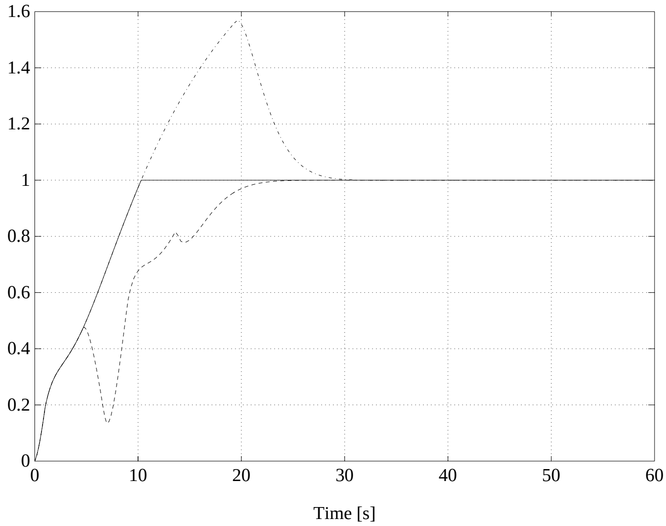

Fig. 4.21. Realisable reference (w^r);
__ Conditioning technique,
-- Incremental algorithm,
-.- No protection against windup,
... Unlimited response

4.3. Realisable Reference for Controllers With General Rational Transfer Function

Controller with general rational transfer function with anti-windup is described by equation (3.25) in chapter 3.3.

$$U = \frac{N_1(s)}{D_1(s)}W - \frac{N_2(s)}{D_2(s)}Y + \frac{N_3(s)}{D_3(s)}\left(U^r - U\right) \tag{4.32}$$

where N_3(s)/D_3(s) denote the anti-windup feedback transfer function. From the definition of realisable reference (see chapters 4.1.1 and 4.1.2), we can express U^r as

$$U^r = \frac{N_1(s)}{D_1(s)}W^r - \frac{N_2(s)}{D_2(s)}Y \tag{4.33}$$

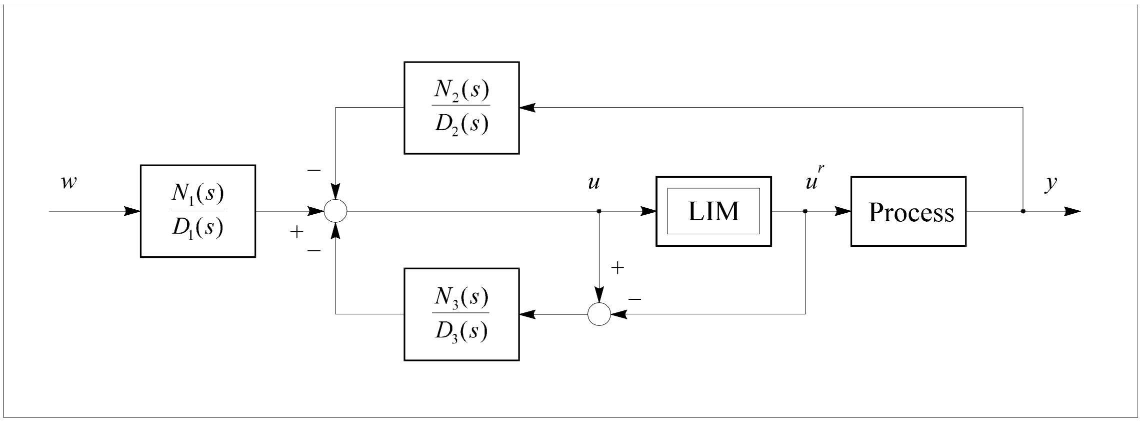

Fig. 4.22. Controller with general rational transfer function and AW

Subtracting (4.33) from (4.32) we can express the realisable reference

$$W^r = W + \frac{D_1(s)}{N_1(s)}\left[1 + \frac{N_3(s)}{D_3(s)}\right]\left(U^r - U\right) = W + G_{w^r}(s)\left(U^r - U\right) \tag{4.34}$$

When process comes out of the limitation, w^r should be the same as w to have no windup effect. Therefore transfer function G_{W^r} should be a static gain:

$$G_{w^r}(s) = K^* \tag{4.35}$$

where K^* represents the constant gain. Inserting (4.35) into (4.34) we get the following expression:

$$K^* = \frac{D_1(s)}{N_1(s)} \frac{D_3(s) + N_3(s)}{D_3(s)} \tag{4.36}$$

From equation (4.36) it can be seen that there exist infinite number of solutions for D_3(s) and N_3(s). To simplify equation (4.36), we can choose

$$D_3(s) = D_1(s) \tag{4.37}$$

Now, we can express N_3(s) from equations (4.36) and (4.37) as

$$N_3(s) = K^* N_1(s) - D_1(s) \tag{4.38}$$

Inserting the solutions for D_3(s) and N_3(s) (equations (4.37) and (4.38)) into equation (4.32), we obtain the following controller with anti-windup compensator:

$$U = \frac{N_1(s)}{D_1(s)} W - \frac{N_2(s)}{D_2(s)} Y + \frac{K^* N_1(s) - D_1(s)}{D_1(s)} \left(U^r - U \right) \tag{4.39}$$

With rearranging equation (4.39), we derive the following representation:

$$U = \frac{1}{K^*} W - \frac{1}{K^*} \frac{D_1(s) N_2(s)}{N_1(s) D_2(s)} Y + \frac{K^* N_1(s) - D_1(s)}{K^* N_1(s)} U^r \tag{4.40}$$

The consequence of equations (4.39) and (4.40) is that there will be *no windup* for arbitrarily chosen K^* ($w^r=w$ at the instant process comes out of the limitation). Even more, the system response will be independent of K^*. The only difference will be in signal u, which depends of K^*. Figures 4.23 and 4.24 show the equivalent realisation of AW described by equations (4.39) and (4.40).

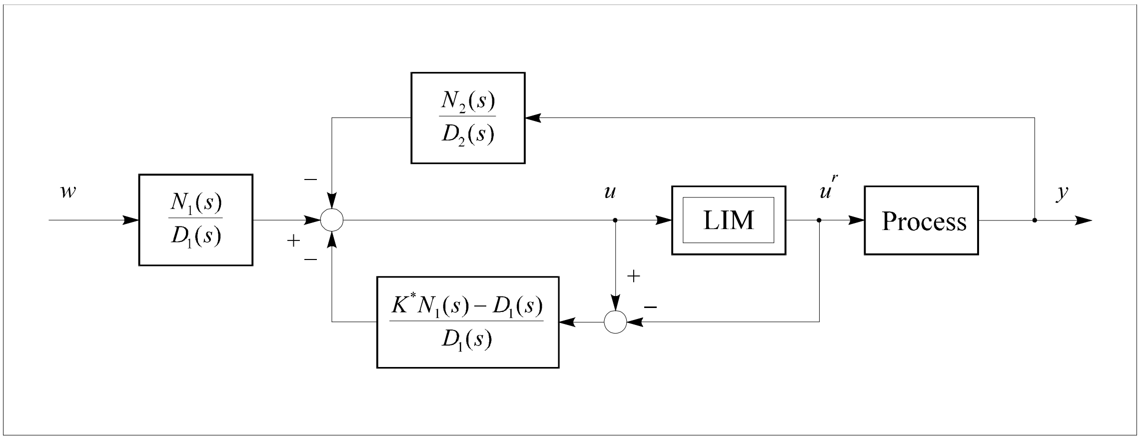

Fig. 4.23. The first representation of AW

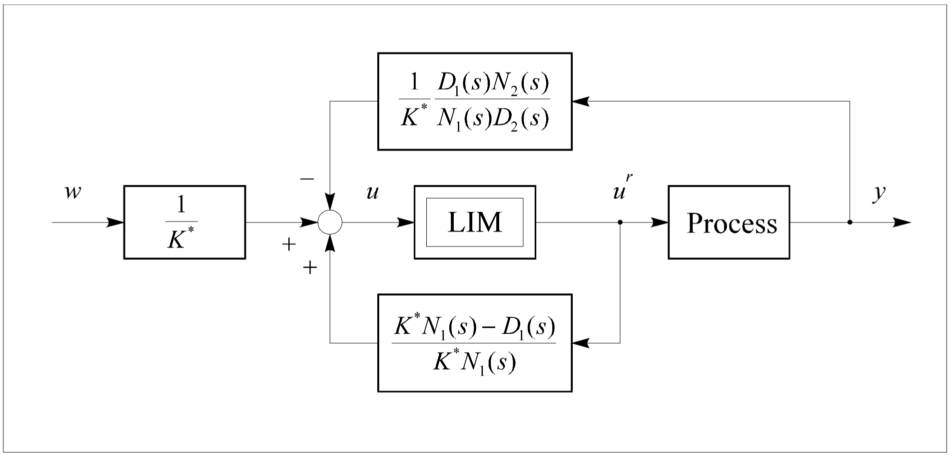

Fig. 4.24. The second representation of AW

Note that in digital implementation, some oscillations could appear when using the anti-windup solution described by (4.39) (see chapter 4.2). Better solution for digital implementation represents equation (4.40). ***Conditioning technique*** solves the problem of algebraic loop (equation (4.39) and Fig. 4.23) automatically by choosing:

$$N_3(s) = N_1(s)\left[\frac{D_1(s)}{N_1(s)}\right]_{s=\infty} - D_1(s) \tag{4.41}$$

with w^r:

$$w^r = w + \left[\frac{D_1(s)}{N_1(s)}\right]_{s=\infty}\left(u^r - u\right) \tag{4.42}$$

Note that *conditioning technique* is a special case of anti-windup algorithm where algebraic loop does not exist. The value of K^* when using conditioning technique is:

$$K^* = \left[\frac{D_1(s)}{N_1(s)}\right]_{s=\infty} \tag{4.43}$$

Two examples have been made to depict chosen anti-windup solution. In the first, the process

$$G_{PR} = \frac{1}{(1+8s)(1+4s)} \tag{4.44}$$

and controller

$$\begin{aligned} N_1 &= 40s + 5 \\ D_1 &= 8s \\ N_2 &= N_1 \\ D_2 &= D_1 \end{aligned} \tag{4.45}$$

are used. The limitations of the system are

$$U_{\max} = 1.5, \quad U_{\min} = 0 \tag{4.46}$$

Two different K^* have been used. Figures 4.25 to 4.28 show the results. Full line represents the solution when K^*=0.2 is used and dashed line the solution for K^*=10. We can see u^r, w^r and y are the same for both K^*. The difference is only in controller output (u) signal.

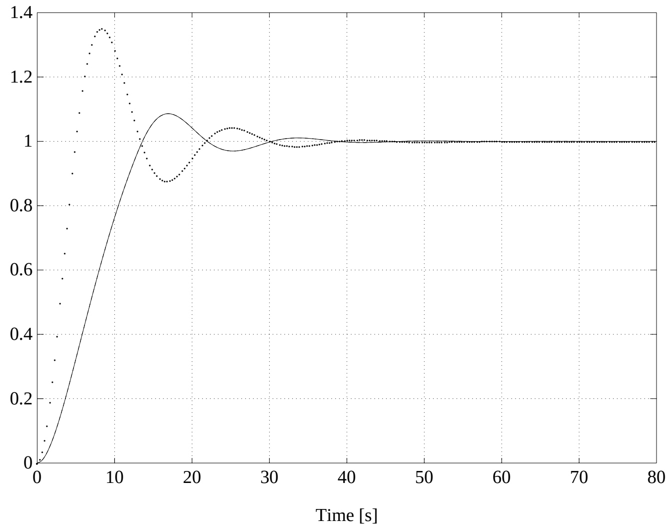

Fig. 4.25. Process output (y); __ K^*=0.2, -- K^*=10, ... Unlimited response

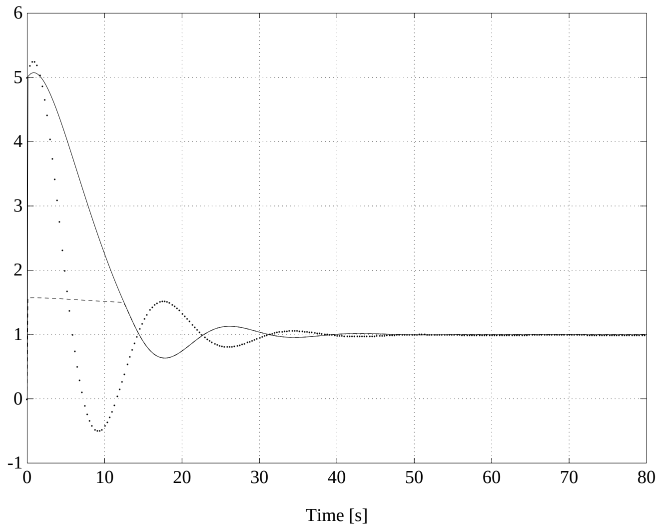

Fig. 4.26. Controller output (u); __ K^*=0.2, -- K^*=10, ... Unlimited response

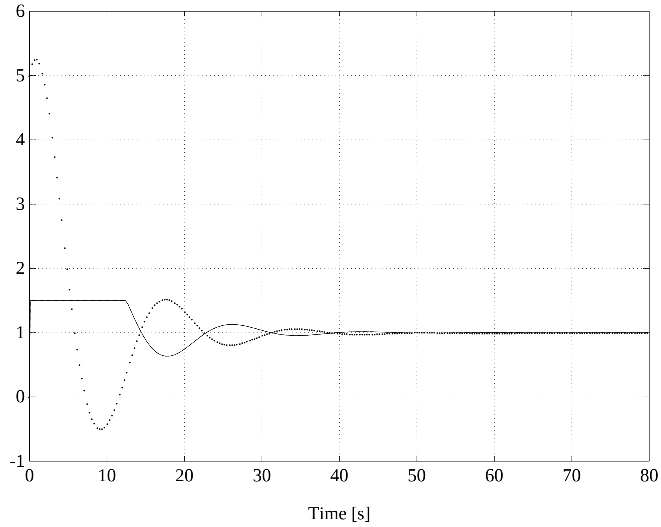

Fig. 4.27. Process input (u^r); __ K^*=0.2, -- K^*=10, ... Unlimited response

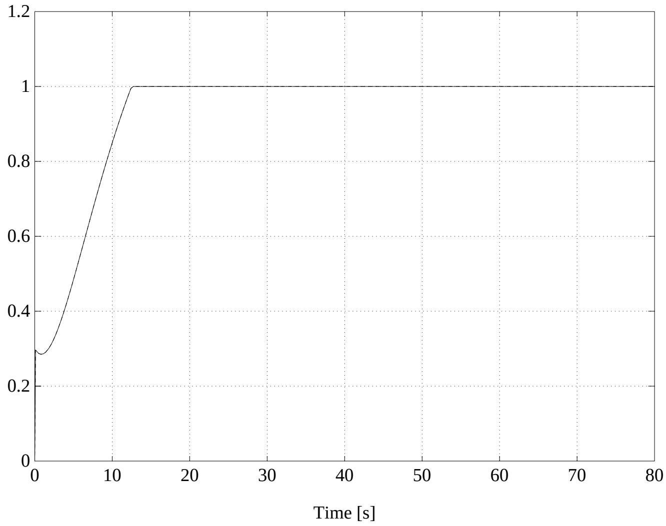

Fig. 4.28. Realisable reference (w^r); __ K^*=0.2, -- K^*=10

For the second example we chose

$$G_{PR} = \frac{1 + 0.2s}{(1 + 4s)(1 + s)^2} \tag{4.47}$$

and

$$\begin{aligned} N_1 &= 8s^2 + 10s + 2 \\ D_1 &= 0.2s^2 + s \\ N_2 &= N_1 \\ D_2 &= D_1 \end{aligned} \tag{4.48}$$

The limitations of the system were

$$U_{\max} = 1.5, \;\; U_{\min} = 0, \;\; v_{\max} = 2s^{-1}, \;\; v_{\min} = -2s^{-1} \tag{4.49}$$

The resulting responses are shown in Figures 4.29 to 4.32. Full line represents the result when using K^*=1 and dashed line for the case K^*=0.1. Again we can see similar situation as before. Resulting process response and realisable reference are quite good.

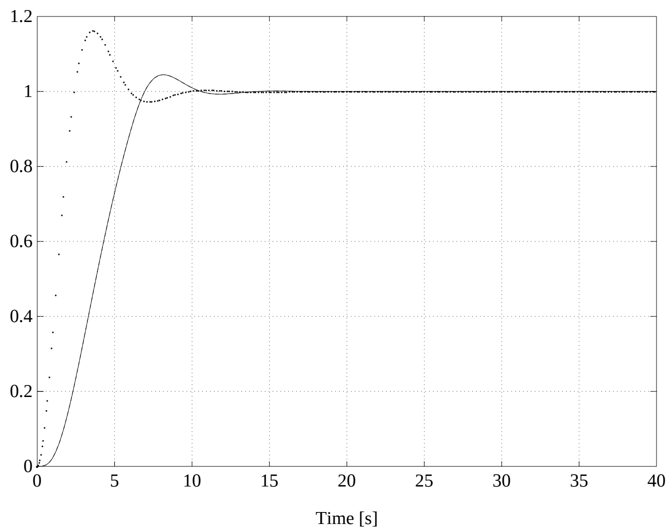

Fig. 4.29. Process output (y); __ K^*=1, -- K^*=0.1, ... Unlimited response

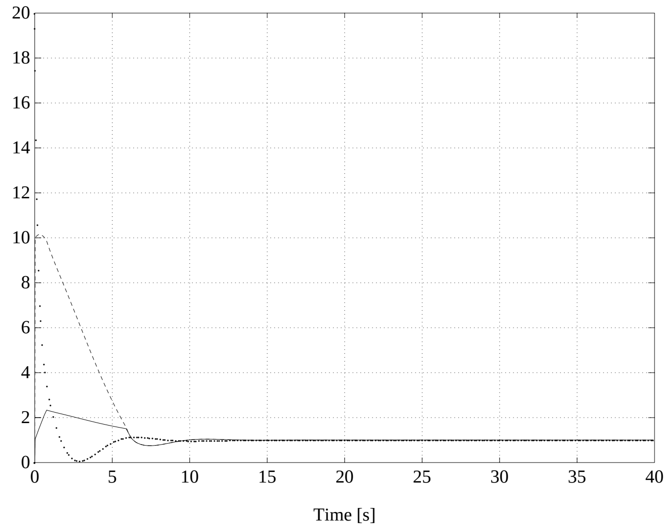

Fig. 4.30. Controller output (u); __ K^*=1, -- K^*=0.1, ... Unlimited response

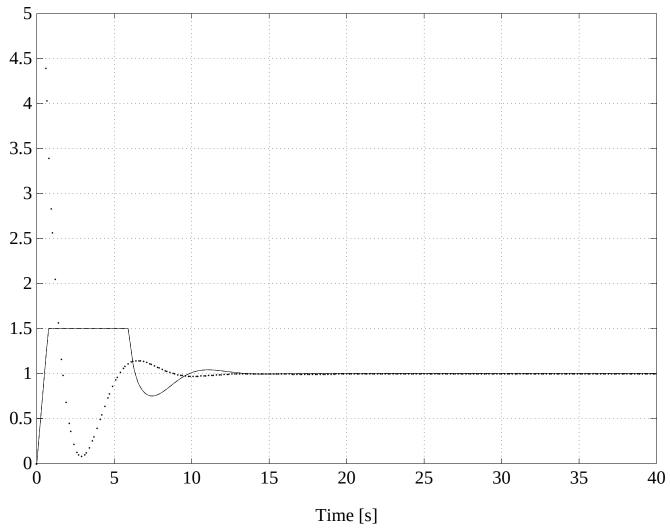

Fig. 4.31. Process input (u^r); __ K^*=1, -- K^*=0.1, ... Unlimited response

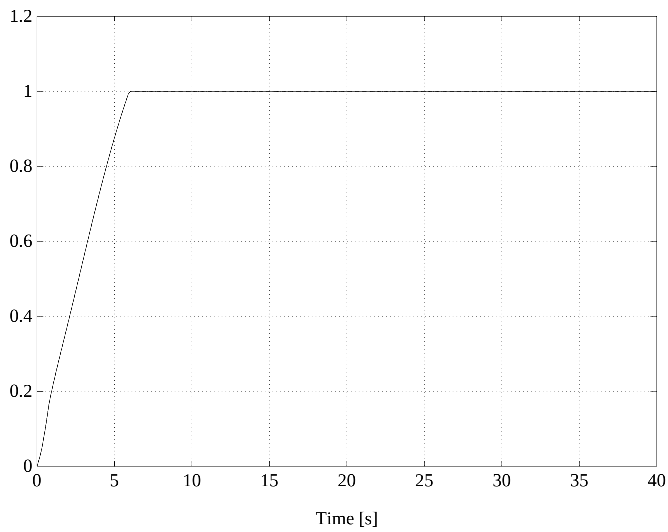

Fig. 4.32. Realisable reference (w^r); __ K^*=1, -- K^*=0.1

4.4. Realisable Reference for State-Space Controllers

Consider the following state-space controller:

$$\begin{aligned}\dot{\underline{x}} &= \underline{\underline{A}}\,\underline{x} + \underline{\underline{B}}\,\underline{w} - \underline{\underline{E}}\,\underline{y} + \underline{\underline{G}}\left(\underline{u}^r - \underline{u}\right)\\ \underline{u} &= \underline{\underline{C}}\,\underline{x} + \underline{\underline{D}}\,\underline{w} - \underline{\underline{F}}\,\underline{y}\end{aligned} \qquad (4.50)$$

where A, B, C, D, E and F denote the matrices of controller given in the state-space form and G is an anti-windup feedback matrix from vectors u^r and u. After Laplace transformation of (4.50) we get

$$\begin{aligned}\underline{X} &= \left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\left[\underline{\underline{B}}\,\underline{W} - \underline{\underline{E}}\,\underline{Y} + \underline{\underline{G}}\left(\underline{U}^r - \underline{U}\right)\right]\\ \underline{U} &= \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\left[\underline{\underline{B}}\,\underline{W} - \underline{\underline{E}}\,\underline{Y} + \underline{\underline{G}}\left(\underline{U}^r - \underline{U}\right)\right] + \underline{\underline{D}}\,\underline{W} - \underline{\underline{F}}\,\underline{Y}\end{aligned} \qquad (4.51)$$

From the definition of the realisable reference, it follows that when using w^r instead of w, u becomes the same as u^r. According to (4.51) it leads to

$$\underline{U}^r = \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\left[\underline{\underline{B}}\underline{W}^r - \underline{\underline{E}}\underline{Y}\right] + \underline{\underline{D}}\underline{W}^r - \underline{\underline{F}}\underline{Y} \tag{4.52}$$

Subtracting u from u^r (equations 4.51 and 4.52) gives

$$\underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{G}}\left(\underline{U}^r - \underline{U}\right) + \left(\underline{U}^r - \underline{U}\right) = \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{B}}\left(\underline{W}^r - \underline{W}\right) + \underline{\underline{D}}\left(\underline{W}^r - \underline{W}\right) \tag{4.53}$$

Let us define

$$\begin{aligned} \underline{\Delta U} &= \underline{U}^r - \underline{U} \\ \underline{\Delta W} &= \underline{W}^r - \underline{W} \end{aligned} \tag{4.54}$$

Equation (4.53) becomes

$$\underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{G}}\underline{\Delta U} + \underline{\Delta U} = \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{B}}\underline{\Delta W} + \underline{\underline{D}}\underline{\Delta W} \tag{4.55}$$

To have no windup, we would like to have such Δw that will become 0 at the instant when process comes out of the saturation ($\Delta u = 0$). To achieve that, we must have static relation between Δw and Δu.

$$\underline{\Delta U} = \underline{\underline{K}}^* \underline{\Delta W} \tag{4.56}$$

where K^* is a constant matrix. When inserting (4.56) into (4.55) we obtain

$$\begin{aligned} \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{G}}\underline{\underline{K}}^* \underline{\Delta W} + \underline{\underline{K}}^* \underline{\Delta W} &= \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{B}}\underline{\Delta W} + \underline{\underline{D}}\underline{\Delta W} \Rightarrow \\ \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{G}}\underline{\underline{K}}^* + \underline{\underline{K}}^* &= \underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{B}} + \underline{\underline{D}} \end{aligned} \tag{4.57}$$

The solution of (4.57) is

$$\underline{\underline{K}}^* = \underline{\underline{D}} \tag{4.58}$$

and

$$\underline{\underline{G}}\,\underline{\underline{D}} = \underline{\underline{B}} \tag{4.59}$$

Equation (4.56)

$$\underline{\Delta U} = \underline{\underline{D}}\,\underline{\Delta W} \tag{4.60}$$

With respect to the *number* of controller *inputs* and *outputs*, we have three possibilities:

a) Number of controller inputs (m) is the same as number of controller outputs (l). D is square matrix and from (4.59) G becomes

$$\underline{\underline{G}} = \underline{\underline{B}}\,\underline{\underline{D}}^{-1} \tag{4.61}$$

b) Number of controller inputs is smaller than number of controller outputs ($m < l$) what means that D is non-square matrix. If we don't want to have windup in the system, equation (4.56) has to be fulfilled. But from (4.60) it is obvious that (l-m) elements of the vector Δu become linearly dependant. In general, it is not the case in any controller.

The solution of such problem can be to add l-m additional inputs to the controller. These inputs (and consequently references for such inputs) should be the same as some of already existing inputs. Matrices D and B changes such that their additional columns are functions of existed columns. Each column in matrices B and D represents one input to the controller. If additional j-th input is connected to existing i-th input, then i-th and j-th columns of matrices B and D must change such that

$$\begin{aligned} \underline{b}_i(old) &= \underline{b}_i(new) + \underline{b}_j \\ \underline{d}_i(old) &= \underline{d}_i(new) + \underline{d}_j \end{aligned} \tag{4.62}$$

where b_i and d_i represent i-th column of matrices B and D respectively. In expression (4.62) we must carry out that new columns d_i and d_j are not linearly dependent to the other columns in matrix D.

Now, matrix D is square with linearly independent new columns and from (4.60) Δu components become independent. So G can be calculated as in expression (4.61).

At the end of this chapter, there is an example which depict such controller design (restructuring).

c) Number of controller inputs is bigger than number of controller outputs ($m > l$). In this case number of *process* inputs is smaller than number of *process* outputs. Process is not controllable and therefore the question of anti-windup solution is irrelevant. The solution could be in reducing the number of *process* outputs which are controlled until $m=l$ or in using some kind of least square solution.

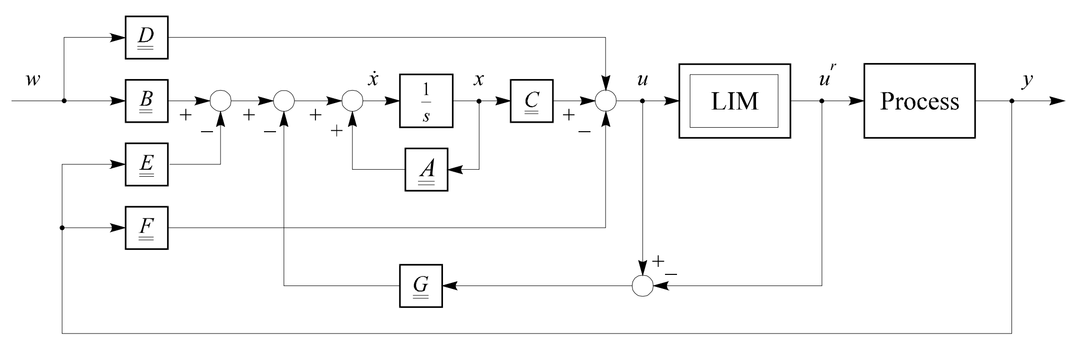

Fig. 4.33. State-space anti-windup scheme

Realisable reference can be expressed from equations (4.52) and (4.55) as

$$\underline{W}^r = \left[\underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{B}} + \underline{\underline{D}}\right]^{-1}\left(\underline{U}^r + \left[\underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{E}} + \underline{\underline{F}}\right]\underline{Y}\right) \tag{4.63}$$

$$\underline{W}^r = \underline{W} + \left[\underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{B}} + \underline{\underline{D}}\right]^{-1}\left[\underline{\underline{C}}\left(s\underline{\underline{I}} - \underline{\underline{A}}\right)^{-1}\underline{\underline{G}} + \underline{\underline{I}}\right]\left(\underline{U}^r - \underline{U}\right) \tag{4.64}$$

To depict above derivations, we have made some examples. In the first (results are shown in Figures 4.34 to 4.44) we have used the following multivariable process:

$$\underline{\underline{G}}_{PR} = \begin{bmatrix} \dfrac{1}{(1+s)(1+2s)} & \dfrac{0.2}{(1+4s)(1+8s)} \\ \dfrac{0.2}{(1+2s)(1+8s)} & \dfrac{1}{(1+s)^2} \end{bmatrix} \tag{4.65}$$

and controller:

$$\begin{aligned} \underline{\underline{A}} &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\ \underline{\underline{B}} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ \underline{\underline{C}} &= \begin{bmatrix} 2.6042 & -1.0417 \\ -0.5208 & 5.2083 \end{bmatrix} \\ \underline{\underline{D}} &= \begin{bmatrix} 10.0016 & -0.0625 \\ -0.125 & 5.0008 \end{bmatrix} \\ \underline{\underline{E}} &= \underline{\underline{B}} \\ \underline{\underline{F}} &= \underline{\underline{D}} \end{aligned} \tag{4.66}$$

System limits were:

$$\begin{aligned} &U_{\max_1} = 2; \quad U_{\min_1} = -2; \quad v_{\max_1} = 2s^{-1}; \quad v_{\min_1} = -2s^{-1} \\ &U_{\max_2} = 2; \quad U_{\min_2} = -2; \quad v_{\max_2} = 5s^{-1}; \quad v_{\min_2} = -5s^{-1} \end{aligned} \tag{4.67}$$

Anti-windup feedback matrix $\underline{\underline{G}}$ was calculated from the expression (4.61):

$$\underline{\underline{G}} = \underline{\underline{B}}\,\underline{\underline{D}}^{-1} = \begin{bmatrix} 0.1 & 0.00125 \\ 0.0025 & 0.2 \end{bmatrix} \tag{4.68}$$

The *response of the unlimited system* was shown in Figures 4.34 and 4.35.

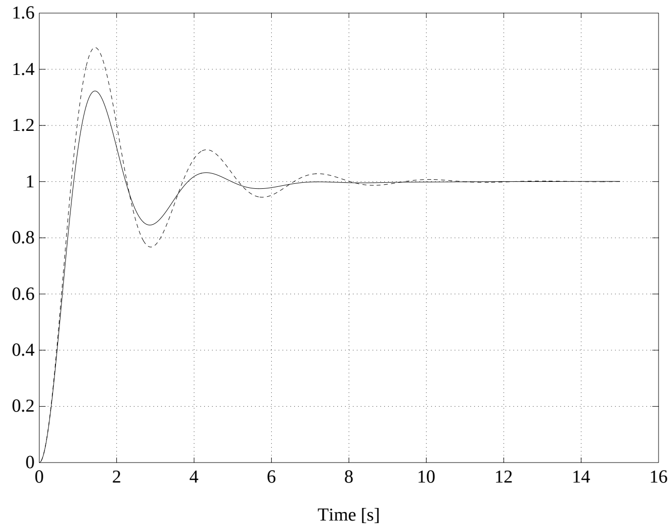

Fig. 4.34. Unlimited response - Process outputs; __ y_1, -- y_2

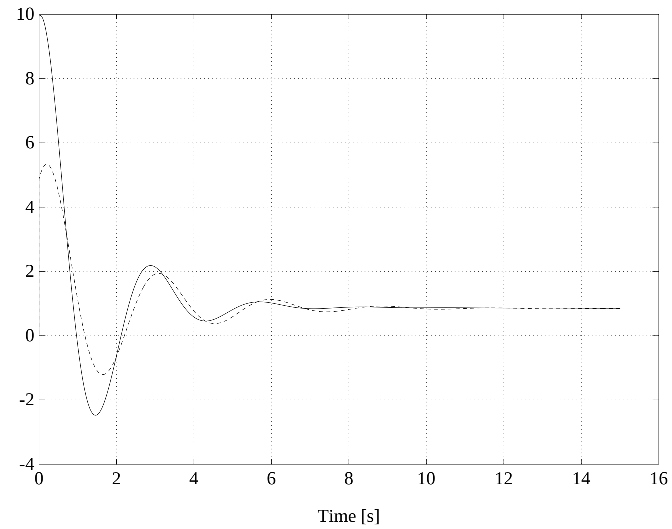

Fig. 4.35. Unlimited response - Process inputs; __ u^r_1 -- u^r_2

The response when using *conditioning technique* (feedback matrix $\underline{\underline{G}}$) is shown in Figures 4.36 to 4.38.

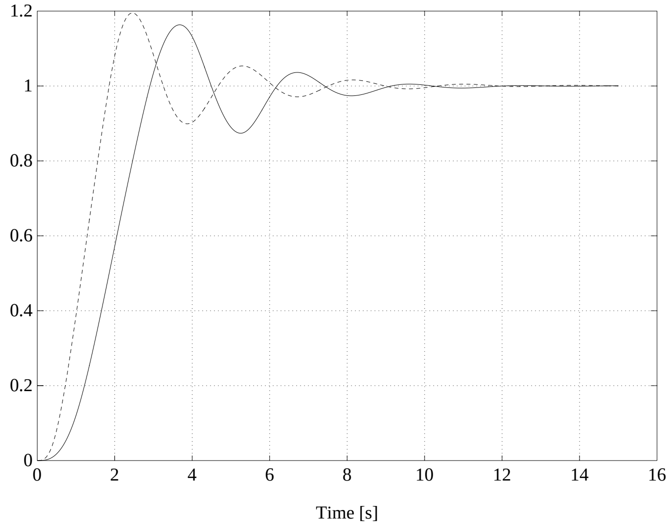

Fig. 4.36. Conditioning technique - Process outputs; __ y_1, -- y_2

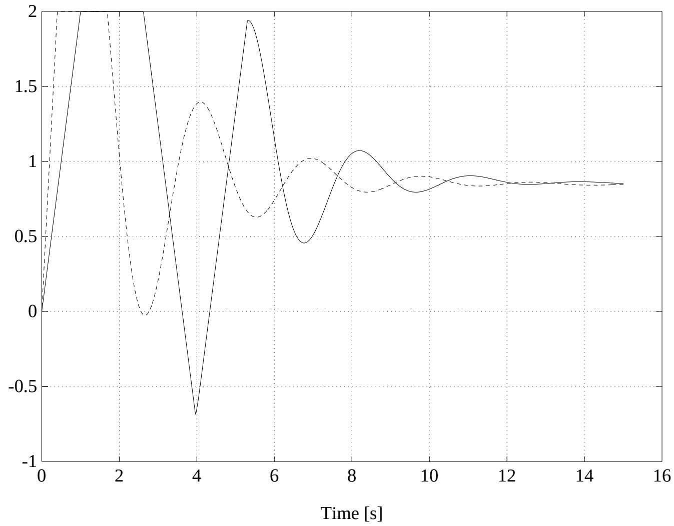

Fig. 4.37. Conditioning technique - Process inputs; __ $u^r{}_1$, -- $u^r{}_2$

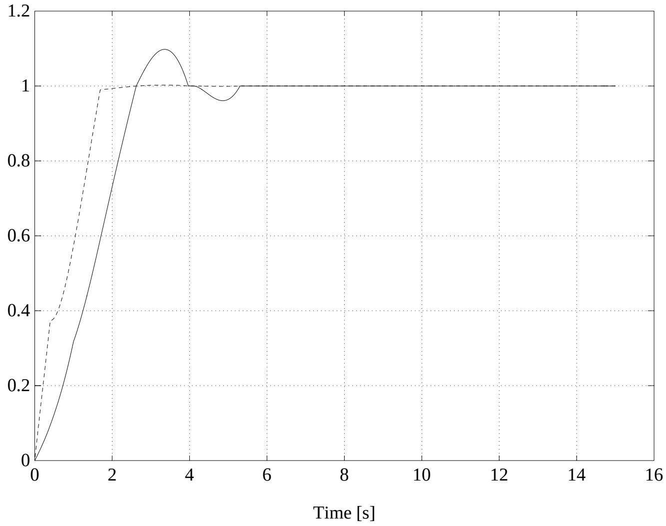

Fig. 4.38. Conditioning technique - Realisable references; __ $w^r{}_1$, -- $w^r{}_2$

The next Figures (4.39 to 4.41) show the experiment made with feedback matrix equal to 100*$\underline{\underline{G}}$, what can serve as an *approximation of the incremental algorithm*.

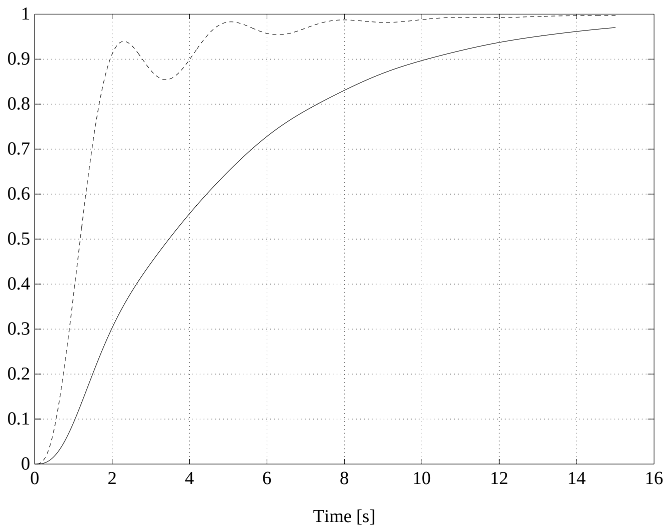

Fig. 4.39. Feedback AW matrix 100$\underline{\underline{G}}$ - Process outputs; __ y_1, -- y_2*

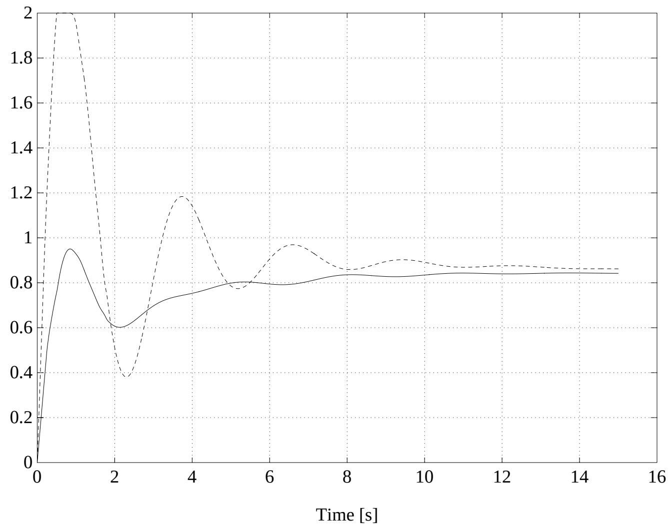

Fig. 4.40. Feedback AW matrix 100$\underline{\underline{G}}$ - Process inputs; __ u^r_1, -- u^r_2*

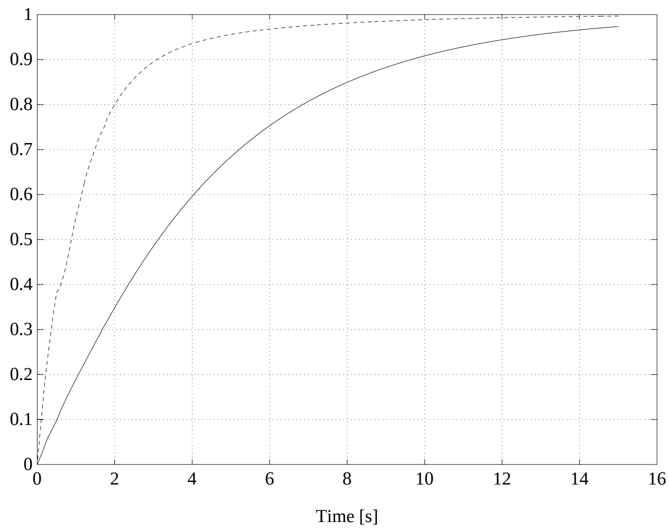

Fig. 4.41. Feedback AW matrix 100$\underline{\underline{G}}$ - Realisable references; __ w^r_1, -- w^r_2*

The last three Figures (4.42 to 4.44) show the system response if there is *no anti-windup protection* in the system.

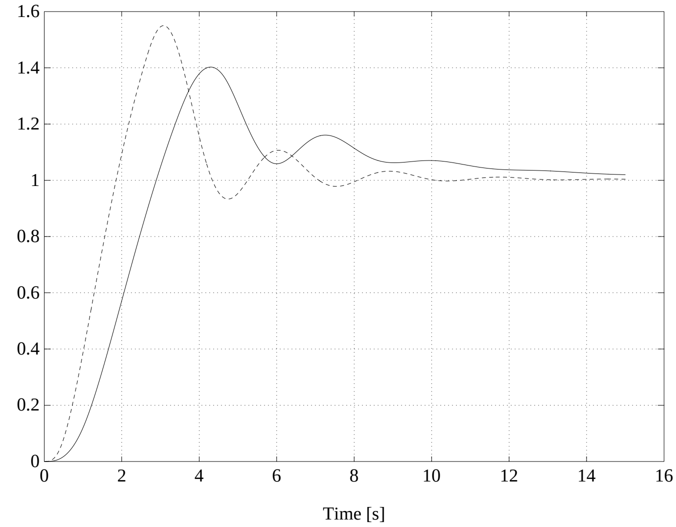

Fig. 4.42. No AW protection - Process outputs; __ y_1, -- y_2

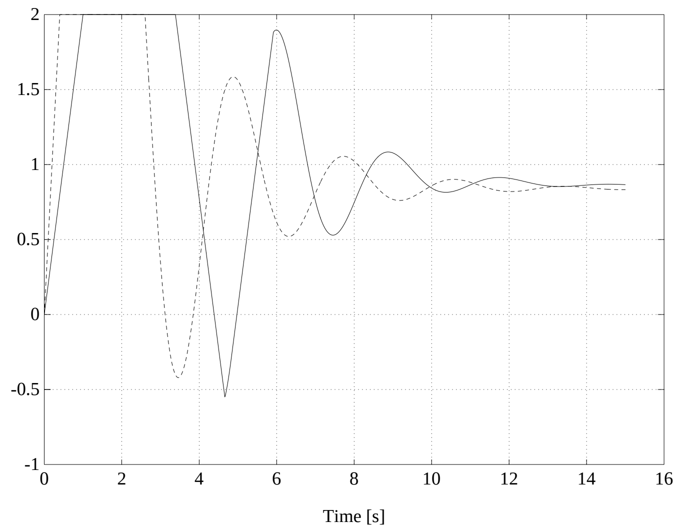

Fig. 4.43. No AW protection - Process inputs; __ u^r_1, -- u^r_2

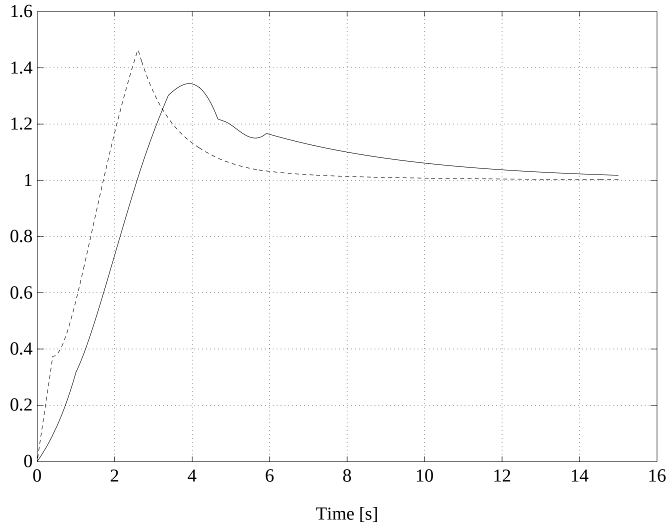

Fig. 4.44. No AW protection - Realisable references; __ $w^r{}_1$, -- $w^r{}_2$

From the results of experiment we can see the effectiveness of anti-windup design when using *conditioning technique*.

The next example show the solution of anti-windup design if system matrix $\underline{\underline{D}}$ is not square. As an example, let us have the following process:

$$G_{PR} = \begin{bmatrix} \frac{1}{(1+s)(1+4s)} & \frac{1}{(1+s)(1+2s)} \end{bmatrix} \tag{4.69}$$

With two inputs and only one output. Therefore the controller has one input and two outputs.

If two PI controllers are used, the controller system matrices become

$$\underline{\underline{A}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\underline{\underline{B}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\underline{\underline{C}} = \begin{bmatrix} \frac{K_1}{T_{i_1}} & 0 \\ 0 & \frac{K_2}{T_{i_2}} \end{bmatrix} \tag{4.70}$$

$$\underline{\underline{D}} = \begin{bmatrix} K_1 \\ K_2 \end{bmatrix}$$

$$\underline{\underline{E}} = \underline{\underline{B}}$$

$$\underline{\underline{F}} = \underline{\underline{D}}$$

where K_1, K_2, T_{i1} and T_{i2} represent proportional gains of the first and second controller and integration time constants of the first and second controller, respectively.

As we can see, the inverse of matrix $\underline{\underline{D}}$ does not exist, so we have to use the procedure being already described in this chapter (equation 4.62) At fist, we have to add additional input to the controller. In our case, we doubled the first (and only one) input. So, the second controller input is in fact the same as first one. In present example, we changed matrices $\underline{\underline{B}}$ and $\underline{\underline{D}}$ (as derived in (4.62)). We chose:

$$\underline{\underline{B}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.71}$$

Note that addition of both columns is the same as the first column in original matrix $\underline{\underline{B}}$. With the similar procedure we can change matrix $\underline{\underline{D}}$, which is chosen as

$$\underline{\underline{D}} = \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \tag{4.72}$$

To show the result of such design, we used:

$$K_{P_1} = 5;\ \ K_{P_2} = 5;\ \ T_{i_1} = 4;\ \ T_{i_2} = 2 \tag{4.73}$$

Matrix $\underline{\underline{G}}$ became

$$\underline{\underline{G}} = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix} \tag{4.74}$$

Process limitations were

$$U_{\max_1} = 2;\ \ U_{\min_1} = 0;\ \ U_{\max_2} = 4;\ \ U_{\min_2} = 0;\ \ v_{\max_2} = 2s^{-1};\ \ v_{\min_2} = -2s^{-1} \tag{4.75}$$

where index 1 stands for the first process input and index 2 for the second process input. In the next Figures we will see the result of simulations when both controller references go from 0 to 1 at time origin.

Figures 4.45 and 4.46 show the *unlimited response* of the mentioned example:

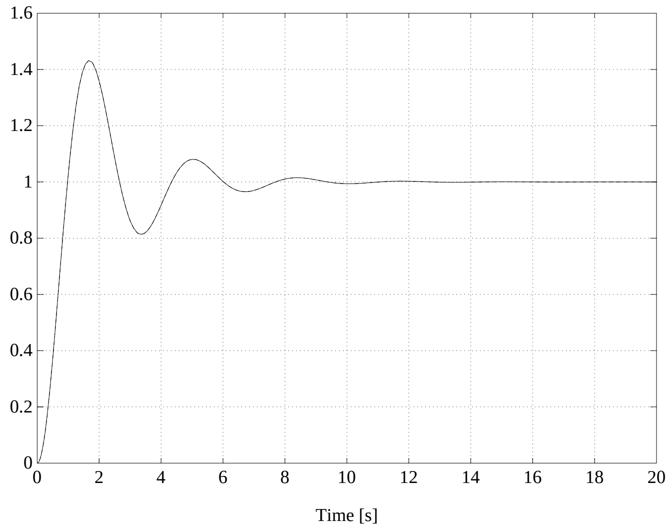

Fig. 4.45. Unlimited response - Process output (y)

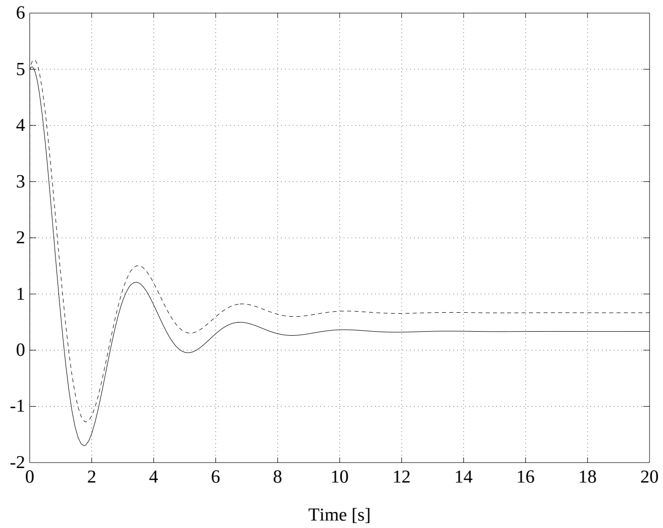

Fig. 4.46. Unlimited response - Process inputs; __ u^r_1, -- u^r_2

Figures 4.47 to 4.49 show results if the system is limited and *conditioning technique* is used.

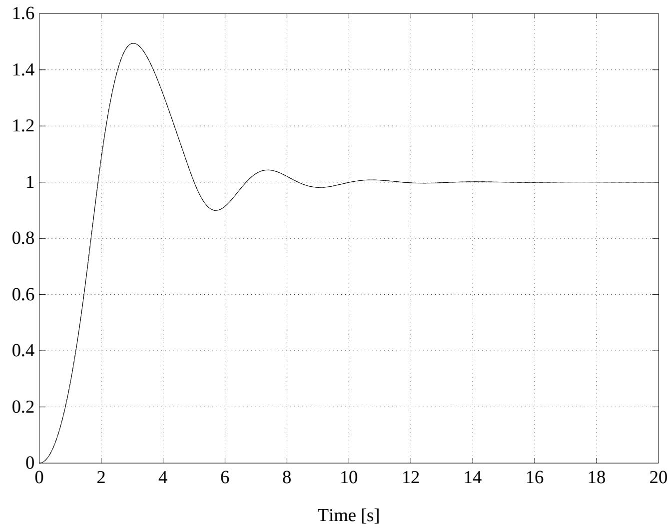

Fig. 4.47. Conditioning technique - Process output (y)

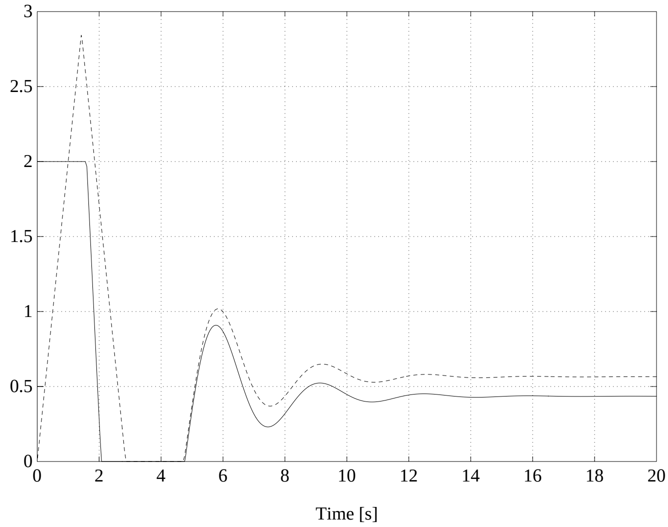

Fig. 4.48. Conditioning technique - Process inputs; __ u^r_1, -- u^r_2

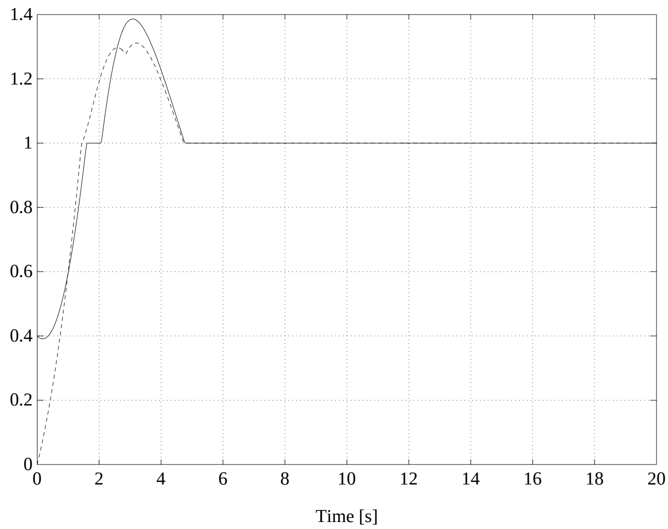

Fig. 4.49. Conditioning technique - Realisable references; __ w^r_1, -- w^r_2

The next response is obtained if the feedback anti-windup matrix 10*$\underline{\underline{G}}$ (Figures 4.50 to 4.52) is used. It can serve as an *approximation of the incremental algorithm*.

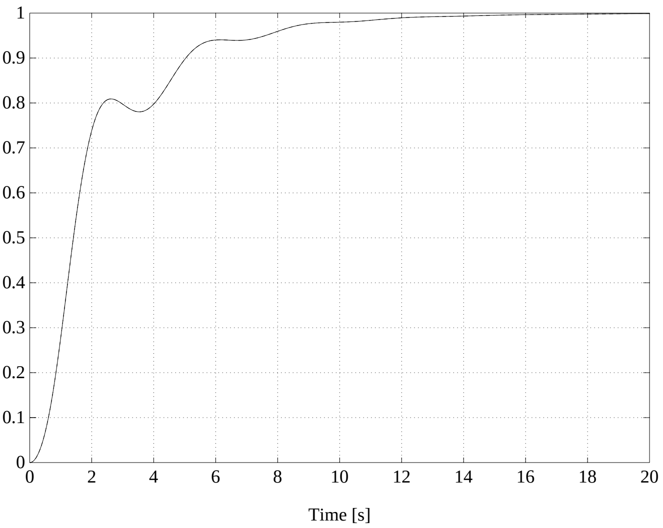

*Fig. 4.50. Feedback matrix 10*G - Process output (y)*

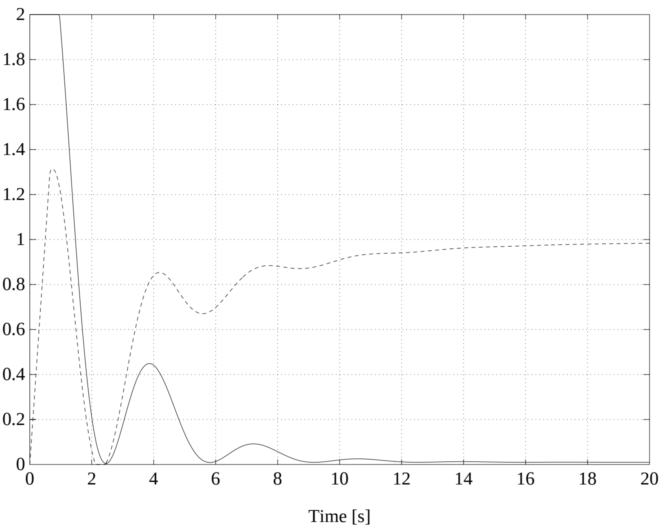

*Fig. 4.51. Feedback matrix 10*G - Process inputs; __ u^r_1, -- u^r_2*

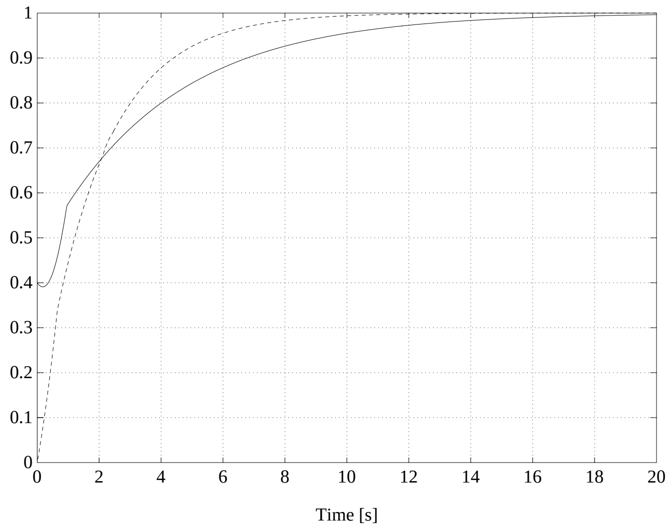

Fig. 4.52. Feedback matrix 10$\underline{\underline{G}}$ - Realisable references; __ w^r_1, -- w^r_2*

And finally, there is a limited system response if *no anti-windup technique* is used. From figures 4.53 to 4.55 we can see that effect of windup is quite strong.

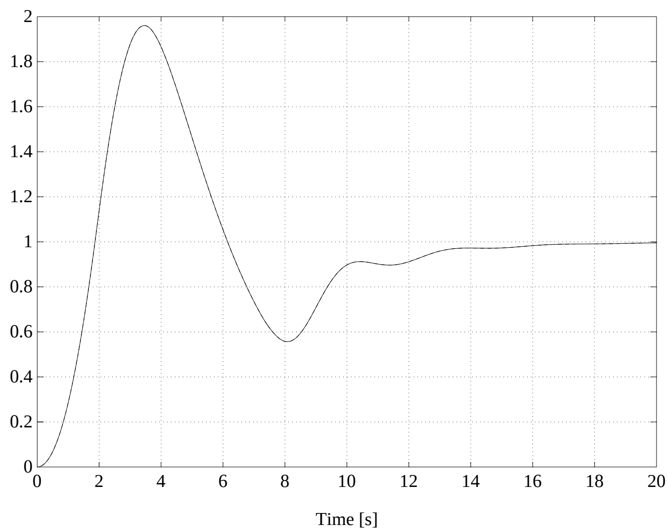

Fig. 4.53. No AW protection - Process output (y)

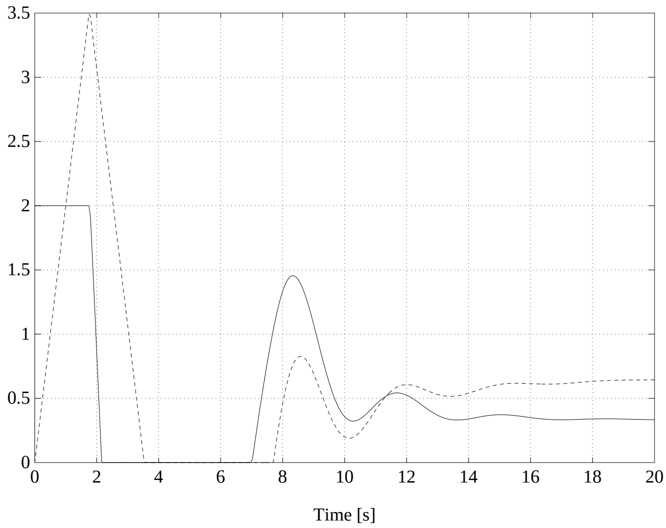

Fig. 4.54. No AW protection - Process inputs; __ u^r_1, -- u^r_2

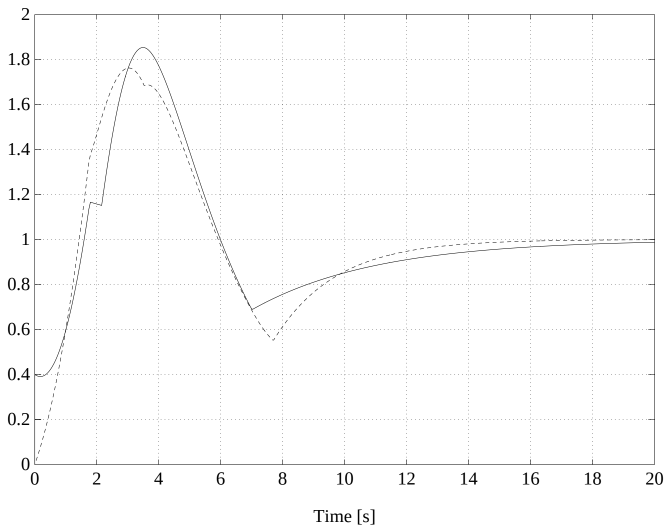

Fig. 4.55. No AW protection - Realisable references; __ w^r_1, -- w^r_2

Simulation results show that conditioning technique gives superiour system response.

The next example is based on the same process as in the previous case (equations (4.69), (4.73) and (4.75)). The difference is only in the way how to obtain matrices $\underline{\underline{B}}$ and $\underline{\underline{D}}$. From equation 4.62, we can see that many possible solutions exist. Here we used the following values of matrices $\underline{\underline{B}}$ and $\underline{\underline{D}}$:

$$\underline{\underline{B}} = \begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix} \quad (4.76)$$

$$\underline{\underline{D}} = \begin{bmatrix} 0.6K_1 & 0.4K_1 \\ 0.4K_2 & 0.6K_2 \end{bmatrix} \quad (4.77)$$

Note that matrix $\underline{\underline{G}}$ did not change from the previous example (4.74).

The results of such system is shown in Figures 4.56 to 4.66. *Unlimited response* of the system is shown in Figures 4.56 and 4.57.

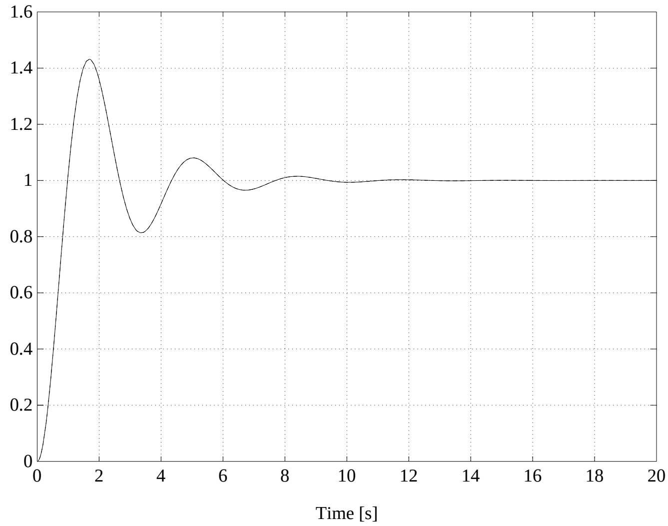

Fig. 4.56. Unlimited response - Process output (y)

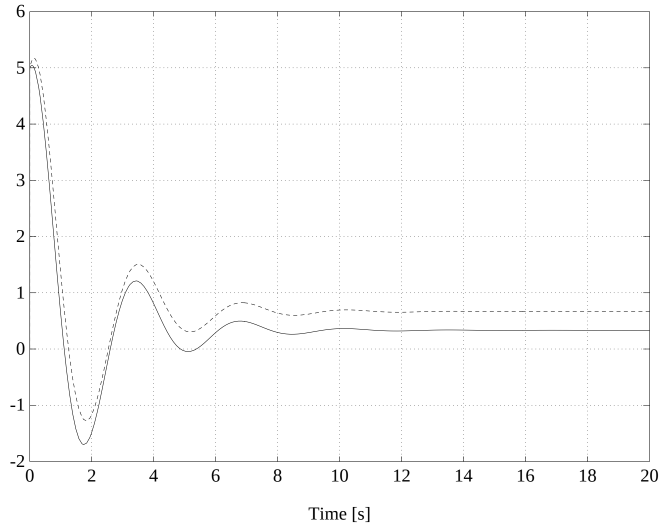

Fig. 4.57. Unlimited response - Process inputs; __ u^r_1, -- u^r_2

The limited response when using *conditioning technique* is shown in Figures 4.58 to 4.60.

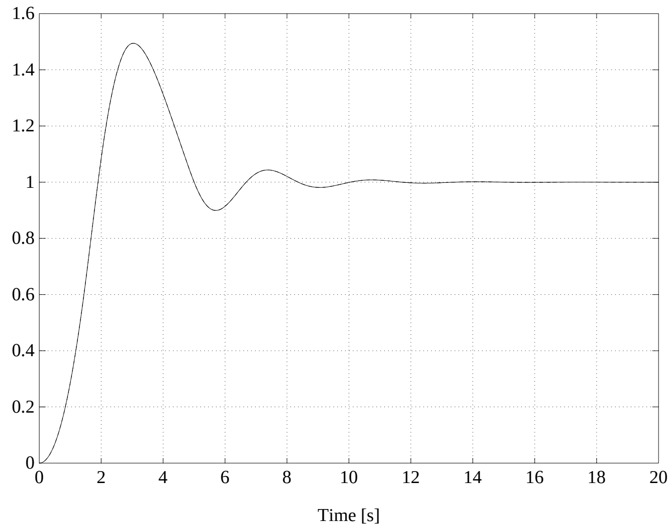

Fig. 4.58. Conditioning technique - Process output (y)

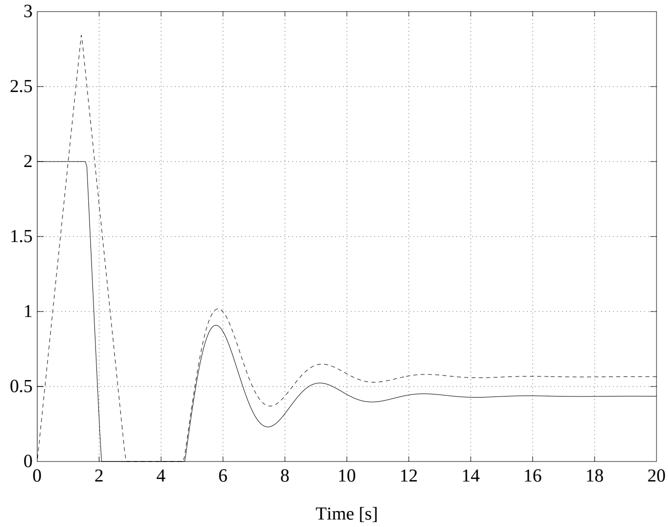

Fig. 4.59. Conditioning technique - Process inputs; __ u^r_1, -- u^r_2

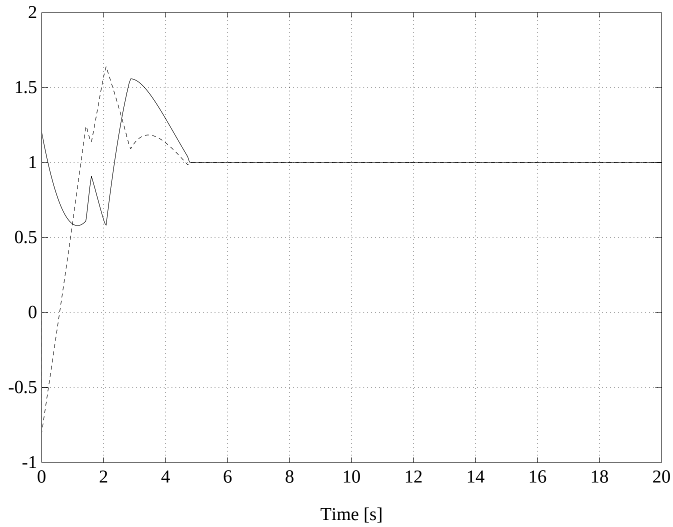

Fig. 4.60. Conditioning technique - Realisable references; __ w^r_1, -- w^r_2

Figures 4.61 to 4.63 show limited response if *10*$\underline{\underline{G}}$ anti-windup feedback matrix* is used.

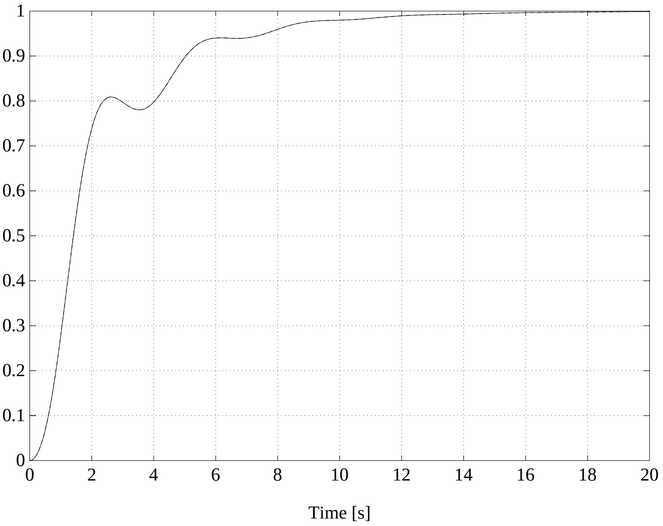

Fig. 4.61. Feedback AW matrix 10$\underline{\underline{G}}$ - Process output (y)*

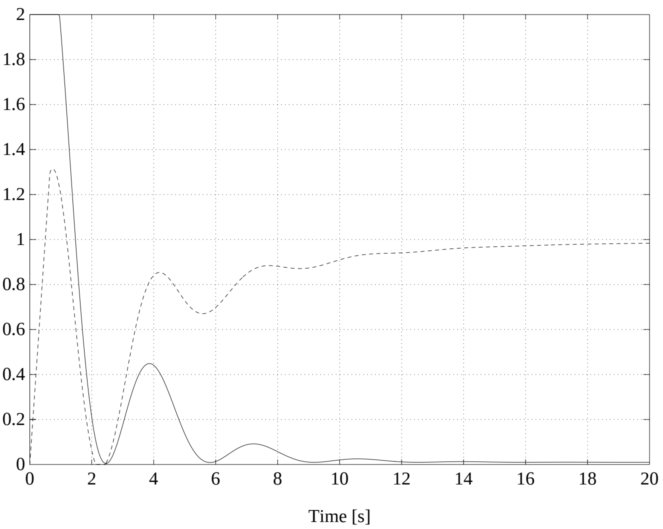

Fig. 4.62. Feedback AW matrix 10$\underline{\underline{G}}$ - Process inputs; __ u^r_1, -- u^r_2*

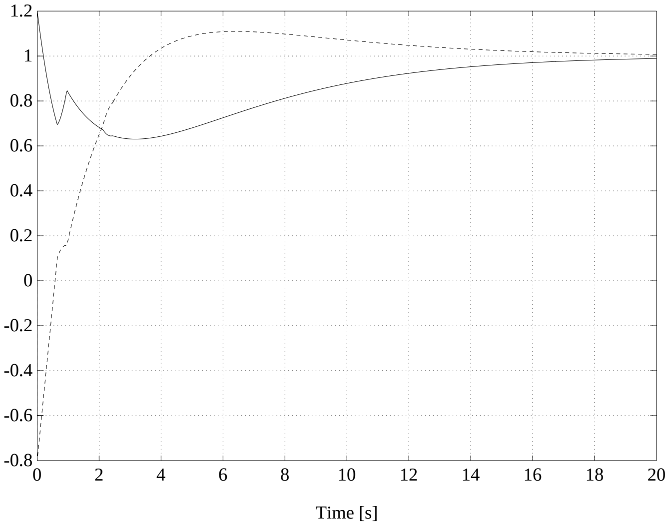

*Fig. 4.63. Feedback AW matrix 10*G̲ - Realisable references; __ w^r_1, -- w^r_2*

And finally the limited response, when using *no anti-windup protection*, is shown in Figures 4.64 to 4.66.

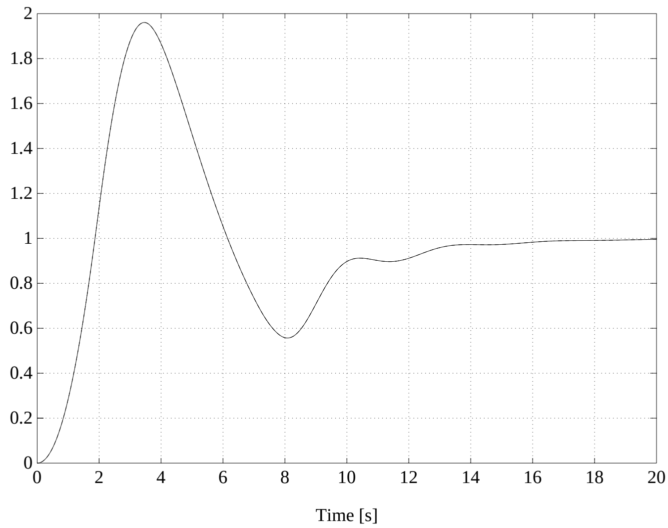

Fig. 4.64. No AW protection - Process output (y)

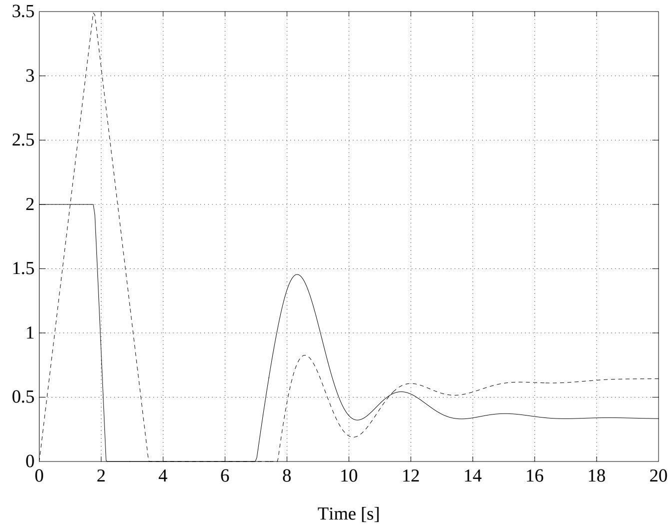

Fig. 4.65. No AW protection - Process inputs; __ $u^r{}_1$, -- $u^r{}_2$

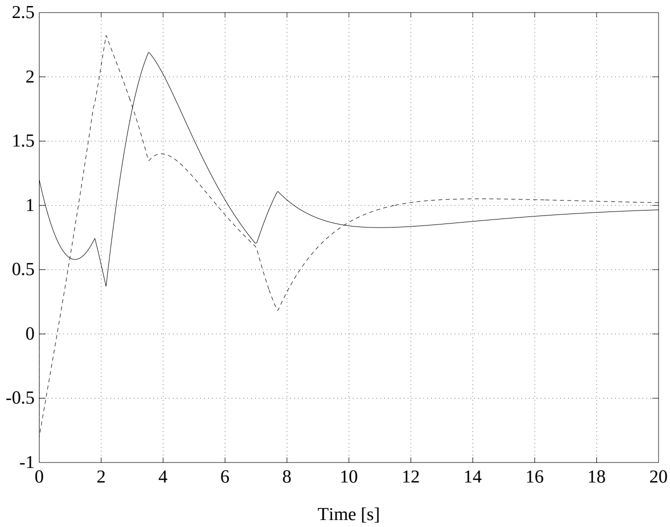

Fig. 4.66. No AW protection - Realisable references; __ $w^r{}_1$, -- $w^r{}_2$

We can see, the process response did not change from previous case. The difference lays only in realisable references. Moreover, the sum of both realisable references ($w^r{}_1$+$w^r{}_2$) is equivalent in both cases.