3. A Review of Some Existing AW (BT, CT) Algorithms

In this section, some typical anti-windup algorithms will be described. As the solutions for bumpless and conditioned transfer are similar to those for anti-windup, the presented algorithms can be used for bumpless and conditioned transfer except that the limitation (LIM) should be replaced by a switch, as shown in Figures 2.12 and 2.18.

3.1. AW Algorithms for PID Controllers

3.1.1. Linear Feedback AW Algorithms

In linear feedback anti-windup algorithms, feedback from the difference between u and u^r is connected to the input of the integral term linearly (through a constant gain $1/K_a$) (Figure 2.12). The linear feedback anti-windup algorithm is also known as "tracking or back calculation" technique [Åström and Rundqwist, 1989] [Hanus et al., 1987] [Rundqwist, 1990], proposed by [Fertik and Ross, 1967].

Among linear feedback anti-windup algorithms, the so-called observer approach, conditioning technique and incremental algorithm will be reviewed.

3.1.1.1. Observer Approach

This approach was first presented in [Åström and Wittenmark, 1984]. We review it now in the framework of PID. The PID controller described by the equation 2.3 can also be described by the following state-space equations:

$$\dot{x} = e \tag{3.1}$$

$$u = \frac{K}{T_i} x + Ke - y_d \tag{3.2}$$

$$u^r = LIM(u) \tag{3.3}$$

where is $\dot{x}$ the input of the integral term, y_d is the output of D-term described by

$$Y_d(s) = \frac{sKT_d}{1 + s\frac{T_d}{N}} Y(s) \tag{3.4}$$

The interpretation of the windup phenomenon is that the *state of the controller* does not correspond to the control signal being fed to the process [Åström and Rundqwist, 1989] [Hanus, 1989] [Morari, 1993] [Campo et al., ACC, 1989] [Rundqwist, 1990] [Zheng, 1994]. To estimate correctly the state when $u^r \neq u$, an *observer* is introduced. The correction of the state(s) is proportional to the difference between u and u^r through a gain L:

$$\dot{x}_e = e + L(u^r - u) \tag{3.5}$$

$$u = \frac{K}{T_i} x_e + Ke - y_d \tag{3.6}$$

$$u^r = LIM(u) \tag{3.7}$$

This is the same as linear feedback anti-windup algorithm by taking $K_a = 1/L$. The problem which still exists is how to choose L. Fig. 3.1. represents anti-windup observer approach solution.

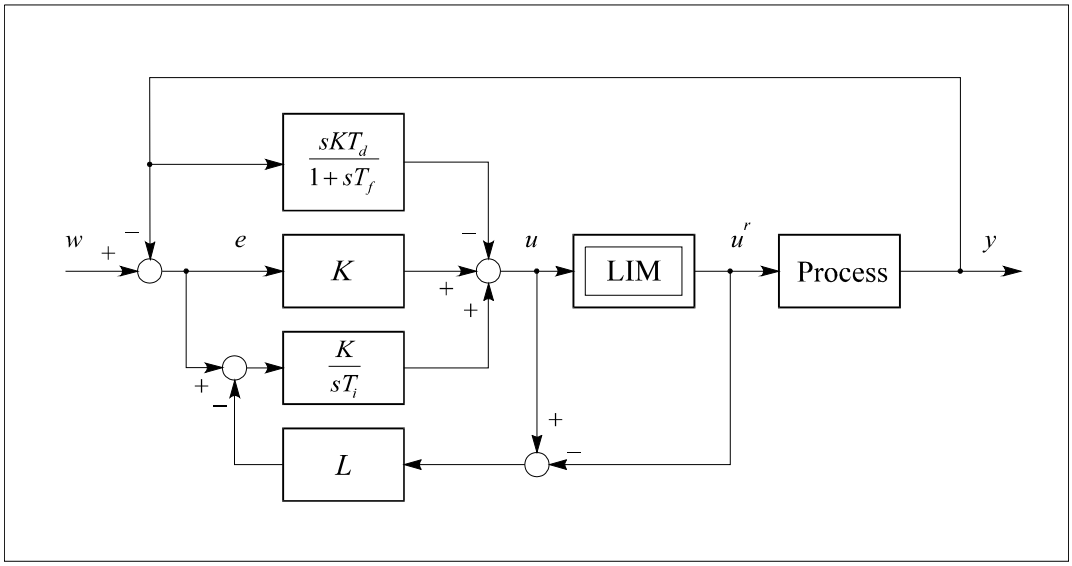

Fig. 3.1. Observer AW approach

3.1.1.2. Conditioning Technique

This approach was first presented in [Hanus, 1980].

Equations (3.1) to (3.3) can be rewritten as:

$$\dot{x} = e = w - y \tag{3.8}$$

$$u = \frac{K}{T_i} x + K(w - y) - y_d \tag{3.9}$$

$$u^r = LIM(u) \tag{3.10}$$

To understand the conditioning technique, it is important to explain a concept of the "*realisable reference*" [Åström and Rundqwist, 1989] [Hanus, 1980] [Hanus 1989] [Hanus et al., 1987] [Hanus and Peng, 1992] [Henrotte, 1989] [Morari, 1993] [Campo et al., C&CE, 1989].

The ***realisable reference*** (w^r) is such that when applied to the controller instead of the reference (w), it results in the control variable (u) which is equal to the process input (u^r) and thus the limitation is not activated.

By above definition, (3.8) and (3.9) yields

$$\dot{x} = w^r - y \tag{3.11}$$

$$u^r = \frac{K}{T_i} x + K\left(w^r - y\right) - y_d \tag{3.12}$$

As w^r is *not available a priori*, we have to use w to update u as (3.9). However, we can use w^r (computed a posteriori) instead of w to update x in order to make controller state consistent:

$$\dot{x} = w^r - y \tag{3.13}$$

$$u = \frac{K}{T_i} x + K\left(w - y\right) - y_d \tag{3.14}$$

$$u^r = LIM(u) \tag{3.15}$$

thus, by subtracting (3.14) from (3.12), we obtain

$$w^r = w + \frac{u^r - u}{K} \tag{3.16}$$

When inserting (3.16) into (3.13) to (3.15), we get

$$\dot{x} = w - y + \frac{u^r - u}{K} \tag{3.17}$$

$$u = \frac{K}{T_i} x + K\left(w - y\right) - y_d \tag{3.18}$$

$$u^r = LIM(u) \tag{3.19}$$

This is a special case of linear feedback anti-windup algorithm with K_a=K (Figure 3.2). Some extensions of the conditioning technique are given in [Hanus and Peng, 1992] [Hanus and Peng, 1991] [Walgama et al. 1992] [Walgama and Sternby, 1990].

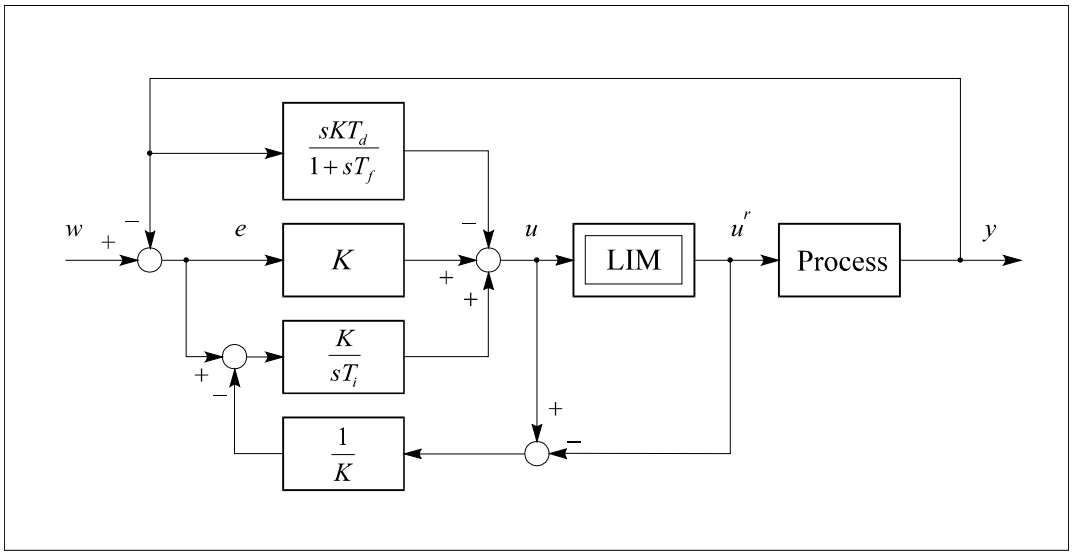

Fig 3.2. Anti-windup solution when using conditioning technique

3.1.1.3. Incremental Algorithm

The incremental algorithm is very often used to prevent windup [Åström and Rundqwist, 1989] [Hanus, 1989]. It is also a relatively simple method to be incorporated in a digital controller. Figure 3.3 shows a typical discrete-time implementation. Here *u(k)* is updated as

$$u(k) = u^r(k-1) + \Delta u(k) \tag{3.20}$$

The difference $\Delta u(k)$ is normally small (except at the instant of reference change), so u tracks u^r very quickly.

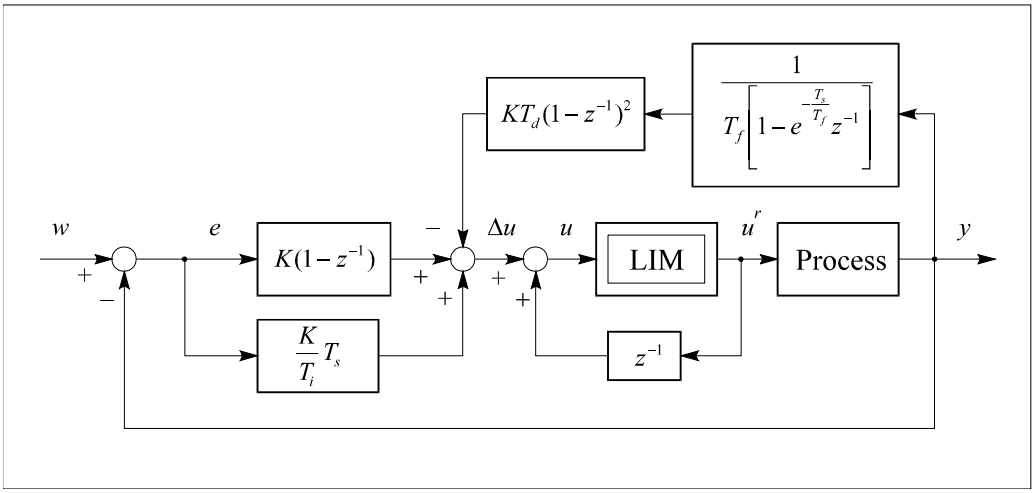

Fig. 3.3. Incremental algorithm (discrete time with sampling time T_S)

The discrete time realisation of the incremental algorithm can be expressed in another way as in Fig. 3.4.

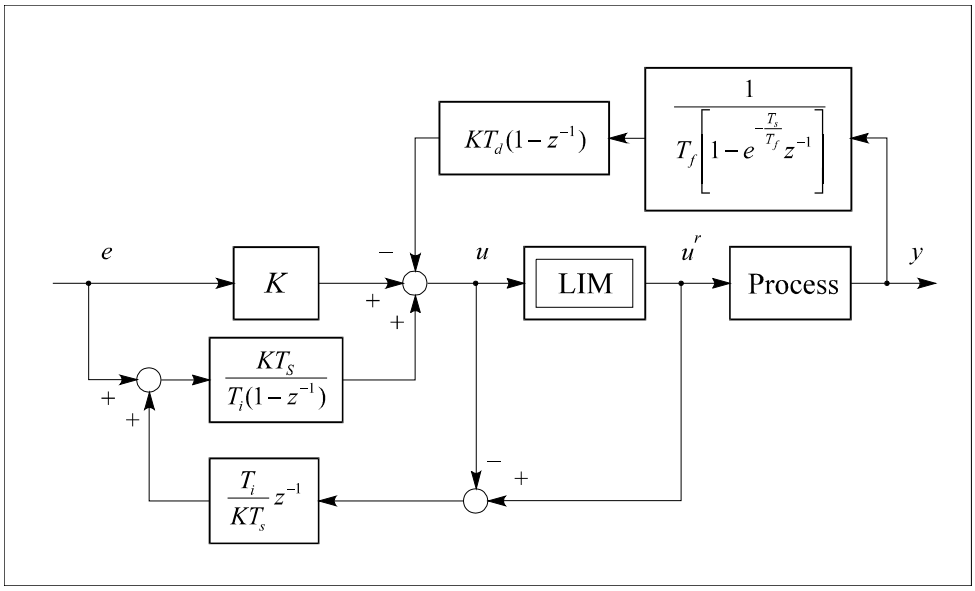

Fig. 3.4. Discrete-time equivalent of the incremental algorithm

It can be transformed into its continuous-time equivalent (Fig. 2.12) by decreasing sampling time T_s:

$$\frac{T_i}{KT_S} z^{-1} \rightarrow \frac{T_i}{KT_S} e^{-sT_S} \tag{3.21}$$

$$\lim_{T_S \rightarrow 0} \left[\frac{T_i}{KT_S} e^{-sT_S} \right] = \lim_{T_S \rightarrow 0} \left[\frac{T_i}{KT_S} \right] \rightarrow \infty \tag{3.22}$$

Thus, the continuous-time realisation can be represented as a special case of the linear feedback anti-windup algorithm with $K_a \rightarrow 0$ (Figure 3.5).

The difference between all presented linear anti-windup algorithms lays in *different value of chosen* K_a (strength of the feedback from limitation back to integrator). *Observer approach* does not define exact value of K_a, *conditioning technique* gives $K_a=K$ and the result of the *incremental algorithm* is $K_a \rightarrow 0$ (small value of K_a).

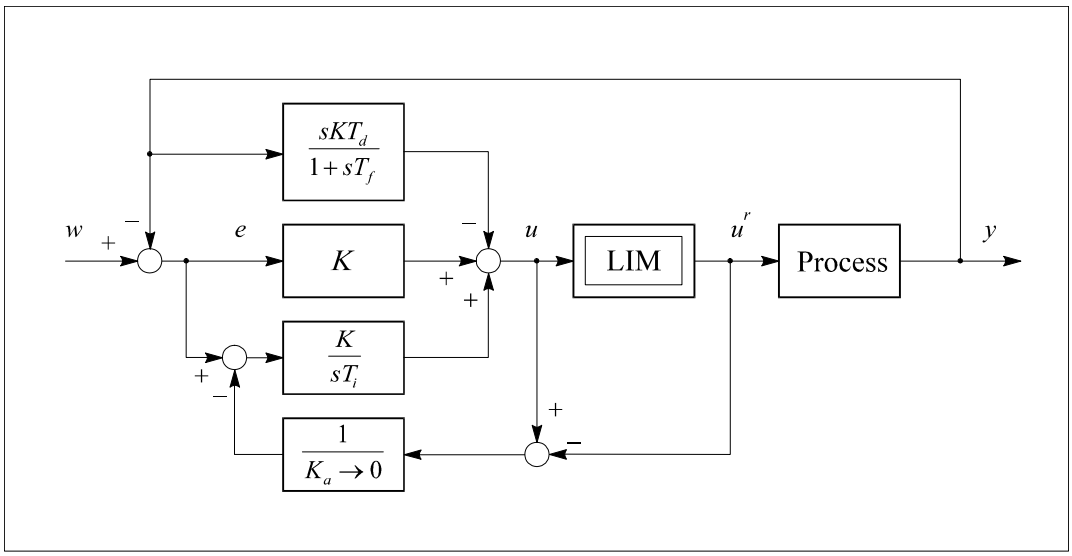

Fig 3.5. Continuous time equivalent of the incremental algorithm

Here we have to point out that in all further experiments continuous-time equivalent of incremental algorithm is used (see Fig. 3.5). Value of K_a=0.01 is used except in cases where different value of K_a are specially given. Therefore, experiments show results of

continuous-time realisation of the incremental algorithm and in some cases results can diverge from original discrete-time realisation.

3.1.2. Non-Linear Feedback AW Algorithms

The representatives of this concept are conditional integration methods [Åström and Rundqwist, 1989] [Hansson et al. 1993] [Morari, 1993] [Rundqwist, 1990] [Shinskey, 1988], which can be classified as follows [Hansson et al. 1993]:

a) Stop integrating when the controller saturates

b) Stop integrating when the control error is large

c) Stop integrating when controller saturates and the control error has the same sign as the control signal

d) Limit the integrator value

e) Stop integrating and assign a predetermined or computed value to the integrator state when a specified condition is true.

In the framework of this work we chose the case a) which can be represented by Fig. 3.6 and equation (3.23). In the following text it will be referred to as ***conditional integration method***.

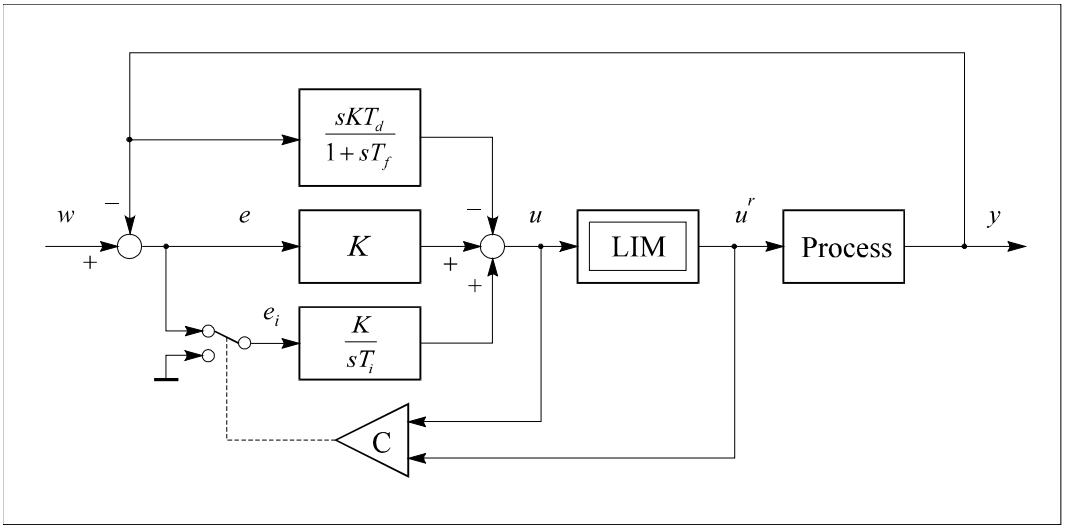

Fig. 3.6. Conditional integration AW method

$$e_i = \begin{cases} e\,; & u^r = u \\ 0\,; & u^r \neq u \end{cases} \tag{3.23}$$

The comparator C switches the input of the integral term according to u and u^r. If the controller works in the linear region ($u^r=u$), the input of the integral term (e_i) is connected to e, otherwise ($u^r \neq u$), the comparator switches e_i to 0 (the updating of the integral term stops).

3.2. AW Algorithms for Generalised PID Controllers

Fig. 3.7. and equation (3.24) represent the solution of anti-windup for generalised PID controllers. Note that the difference from linear anti-windup algorithms for PID controllers is that anti-windup feedback does not feed only integral term. This is due to the fact that D part also contains some kind of "memory" (filter $1+sT_f$).

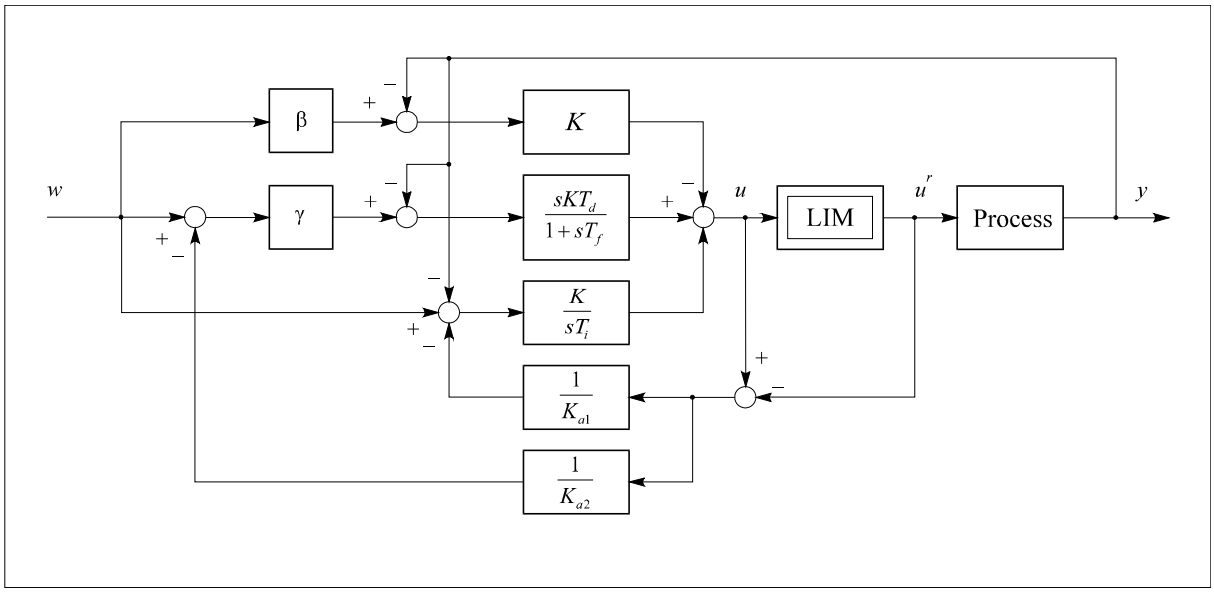

Fig. 3.7. AW method for generalised PID controller

$$U = K\left[\beta + \frac{1}{sT_i} + \gamma\frac{sT_d}{1+sT_f}\right]W - K\left[1 + \frac{1}{sT_i} + \frac{sT_d}{1+sT_f}\right]Y - K\left[\frac{1}{K_{a1}}\frac{1}{sT_i} + \frac{1}{K_{a2}}\gamma\frac{sT_d}{1+sT_f}\right]\left(U - U^r\right) \quad (3.24)$$

Note that K_{a1} and/or K_{a2} can be dynamic transfer function(s).

3.3. AW for Controllers With General Rational Transfer Function

Fig. 3.8. and equation (3.25) represent the solution of anti-windup for controllers with general rational transfer function. To the scheme in Fig. 2.7. (section 2.2) we add an anti-windup feedback (transfer function N_3(s)/D_3(s)) from the limitation to controller output.

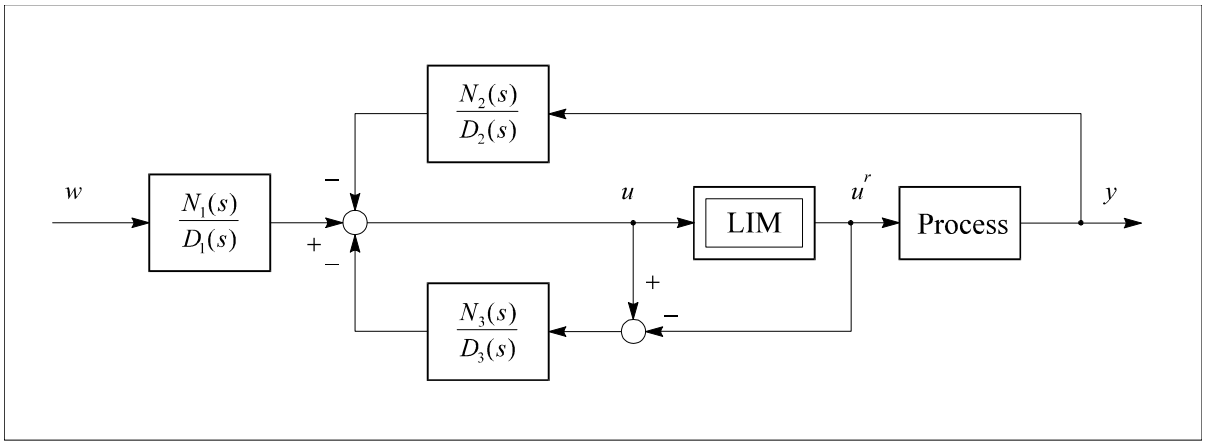

Fig. 3.8. Anti-windup scheme for controllers with general rational transfer function

$$U = \frac{N_1(s)}{D_1(s)}W - \frac{N_2(s)}{D_2(s)}Y - \frac{N_3(s)}{D_3(s)}\left(U - U^r\right) \quad (3.25)$$

3.4. AW for State-Space Controllers

Solutions for state-space controllers are generally covered by observer approach and conditioning technique (sections 3.1.1.1 and 3.1.1.2, where we made a derivation for PID controller). Originally, both methods describe a design of anti-windup for state-space controllers. Fig. 3.9 shows the realisation of anti-windup algorithm for the state-space controller, where matrix G represents an anti-windup feedback from limitation to controller states.

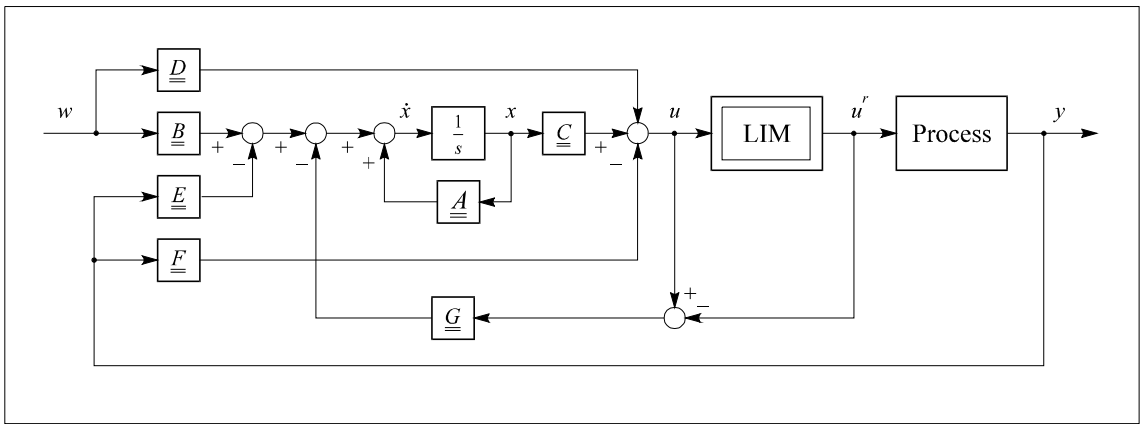

Fig. 3.9. Anti-windup for state-space controllers

$$\begin{aligned} \dot{\underline{x}} &= \underline{\underline{A}}\,\underline{x} + \underline{\underline{B}}\,\underline{u} - \underline{\underline{E}}\,\underline{y} - \underline{\underline{G}}\left(u - u^r\right) \\ \underline{u} &= \underline{\underline{C}}\,\underline{x} + \underline{\underline{D}}\,\underline{w} - \underline{\underline{F}}\,\underline{y} \end{aligned} \tag{3.26}$$