Anti-Windup, Bumpless, and Conditioned Transfer Techniques for PID Controllers

Youbin Peng, Damir Vrancic, and Raymond Hanus

In this article we give a simple and comprehensive review of anti-windup, bumpless and conditioned transfer techniques in the framework of the PID controller. We will show that the most suitable anti-windup strategy for usual applications is the conditioning technique, using the notion of the realizable reference. The exception is the case in which the input limitations are too restrictive. In this case, we propose the anti-windup method with a free parameter tuned to obtain a compromise between the incremental algorithm and the conditioning technique. We also introduce the new notion of conditioned transfer, and we will show it to be a more suitable solution than bumpless transfer. All the discussions are supported by simulations.

All industrial processes are submitted to constraints. For instance, a controller works in a limited range of 0-10 V or 0-20 mA, a valve cannot be opened more than 100% and less than 0%, a motor driven actuator has a limited speed, etc. Such constraints are usually referred to as plant input limitations. On the other hand, a commonly encountered control scheme is to switch from manual to automatic mode or between different controllers. Such mode switches are usually referred to as plant input substitutions.

As a result of limitations and substitutions, the real plant input is temporarily different from the controller output. When this happens, if the controller is initially designed to operate in a linear range, the closed-loop performance will significantly deteriorate with respect to the expected linear performance. This performance deterioration is referred to as windup. Besides windup, in the case of substitution, the difference between the outputs of different controllers results in a big jump in the plant input and a poor tracking performance. This mode switching, which results in such phenomena, is referred to as bump transfer.

A rational way to handle the problem of windup is to take into account, at the stage of control design, the input limitations. However, this approach is very involved and the resulting control law is very complicated. The nonlinearities of the actuator are not always known a priori. A more common approach in practice is to add an extra feedback compensation at the stage of control implementation. As this compensation aims to diminish the effect of windup, it is referred to as anti-windup (AW).

In the case of mode switching, the method that aims to minimize the jump at the plant input is referred to as the bumpless transfer (BT). Yet to minimize the jump is not always preferable, since this may cause a relatively poor tracking performance. Thus we refer to the method that will not only reduce the jump at the plant input but also keep a good tracking performance as conditioned transfer (CT). This new notion will be shown to be very useful later in this article. An anti-windup strategy is usually implemented as a bumpless transfer technique. Indeed, an anti-windup method will usually diminish the jump at the plant input during mode switching. However, it should be pointed out that anti-windup does not necessarily imply bumpless transfer.

The topic of anti-windup and bumpless transfer has been studied over a long period of time by many authors, and the most popular techniques are described in [2, 4, 7, 10, 11, 16]. However, although the concept of anti-windup and bumpless transfer is introduced in almost every basic control textbook, it is not clearly illustrated and is sometimes misinterpreted. For instance, many authors think that anti-windup is aimed at reducing the output overshoot in its step response, or that anti-windup is a synonym for bumpless transfer, or that the best transfer transition is to eliminate the jump at the plant input. These thoughts need to be corrected. Recently, Kothare et al. [10] have presented a general framework for anti-windup design that is a very useful guide for theoretical researchers. Yet a practical control engineer may still look for a simpler tutorial.

Therefore, the objectives of the present article are as follows. First, we would like to illustrate through simulations the phenomenon of windup and bump transfer. We will limit ourselves to the framework of the PID controller, since it is the most common industrial controller and it frequently experiences windup and bump transfer problems. Then we will review the majority of existing anti-windup methods, illustrate the improved results, and compare those methods by using the notion of the realizable reference. Subsequently, we will investigate the case of mode switching and introduce the new notion of conditioned transfer. It will be shown that conditioned transfer is a more suitable solution than bumpless transfer. Finally, we will include some discussions of practical issues.

Background Materials

As mentioned above, the process input is often limited in practice. The most common types of limitations are magnitude

Peng and Hanus are with the Department of Control Engineering, CP 165, Free University of Brussels, Avenue Franklin D. Roosevelt, 50, B-1050, Brussels, Belgium, fax: +32-2-650-26-77, email: peng@labauto.ulb.ac.be or hanus@labauto.ulb.ac.be. Vrancic is with the Department of Computer Automation and Control, J. Stefan Institute, Jamova 39, 61111 Ljubljana, Slovenia, fax: +386-61-219-385, email: damir.vrancic@ijs.si, home page: http://www-e2.ijs.si/people/Damir.Vrancic.html.

0272-1708/96/$05.00©1996IEEE

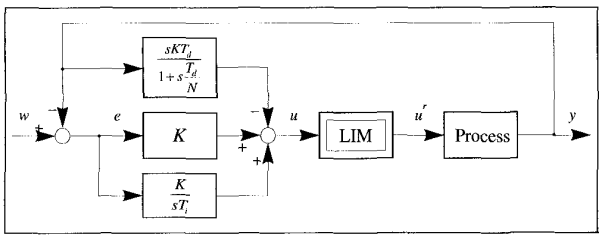

Fig. 1. Limited closed-loop system.

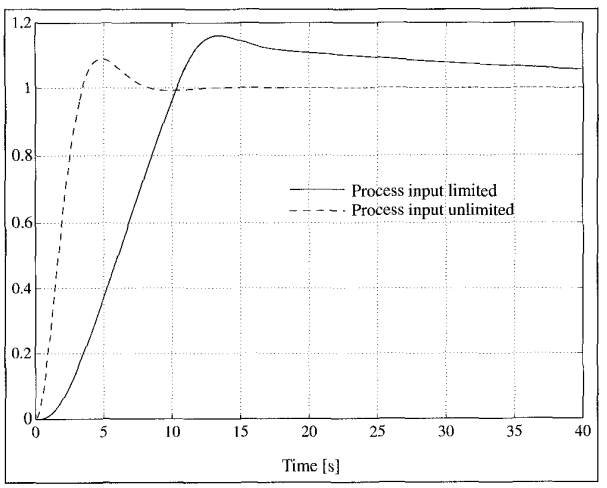

Fig. 2. Process output (y)

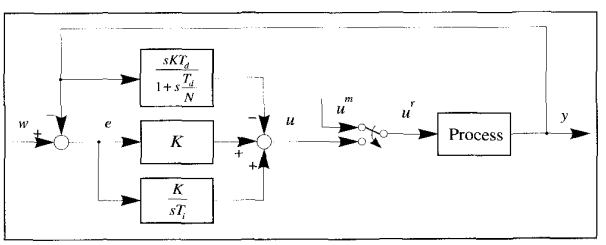

Fig. 3. Bump transfer from manual to automatic mode.

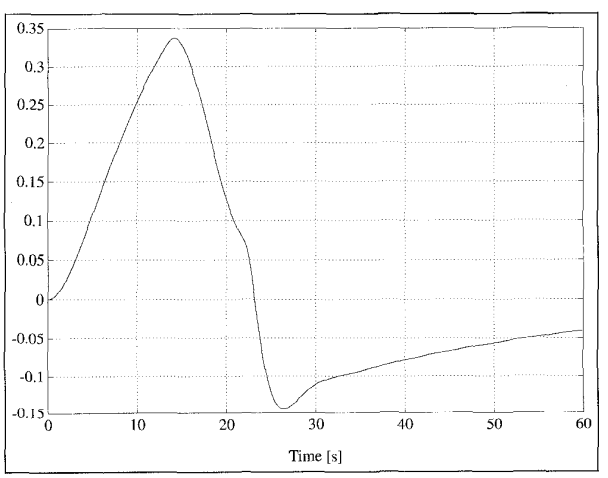

Fig. 4. Process output (y).

and rate limitations. A magnitude limitation and a rate limitation can respectively be described by the following two equations:

$$u^r = \begin{cases} u_{\max} & ; u > u_{\max} \\ u & ; u_{\min} \le u \le u_{\max}, \\ u_{\min} & ; u < u_{\min} \end{cases} \quad (1)$$

$$\frac{\partial u^r}{\partial t} = \begin{cases} v_{\max} & ; \frac{\partial u}{\partial t} > v_{\max} \\ \frac{\partial u}{\partial t} & ; v_{\min} \le \frac{\partial u}{\partial t} \le v_{\max} \\ v_{\min} & ; \frac{\partial u}{\partial t} < v_{\min} \end{cases} \quad (2)$$

where u and u^r are also referred to as the controller output and the real process input, respectively.

A PID controller can be described by the following equation:

$$U(s) = K\left[E(s) + \frac{1}{sT_i}E(s) - \frac{sT_d}{1+s\frac{T_d}{N}}Y(s)\right], \quad (3)$$

where u is the controller output, y is the process output, w is the reference signal, $e = w - y$ is the process tracking error, and the capital letters U, E, and Y denote the Laplace transforms of u, e, and y respectively.The controller parameters are the proportional gain K, the integral time constant T_i, and the derivative time constant T_d. The high frequency gain N is usually set between 7 and 15.

Windup

Consider a closed-loop system containing a PID controller and a magnitude limitation LIM (Fig. 1). (In this article, the block diagram LIM represents the magnitude and/or rate limitations.) Suppose the controller and process are in steady state. A positive step change in w causes a jump in u, so the actuator saturates at high limit if $K > 0$. Thus u^r becomes smaller than u, and y is slower than in the unlimited case. Due to the slower y, e decreases slowly. The integral term increases much more than the one in the unlimited case, and it becomes large. When y approaches w, u still remains saturated or close to saturation due to the large integral term; u decreases after the error has been negative for a sufficiently long time. This leads to a large overshoot and a large settling time of the process output.

To illustrate the above phenomenon, we have made a simulation which is referred to as Sim. 1, with process 1, controller 1 whose parameters are given in the appendix, and the input limitations $u_{max} = 2$, $u_{min} = 0$, $v_{max} = 2$ sec^{-1} and $v_{min} = -2$ sec^{-1}. The closed-loop step responses for both limited and unlimited cases are shown in Fig. 2.

In Fig. 2, we can see a large overshoot and a long process settling time in the limited case as compared to the unlimited case. This closed-loop performance deterioration with respect to the unlimited case is called windup.

Bump Transfer

Let us consider the control scheme with the capability of switching between manual and PID control mode (Fig. 3).

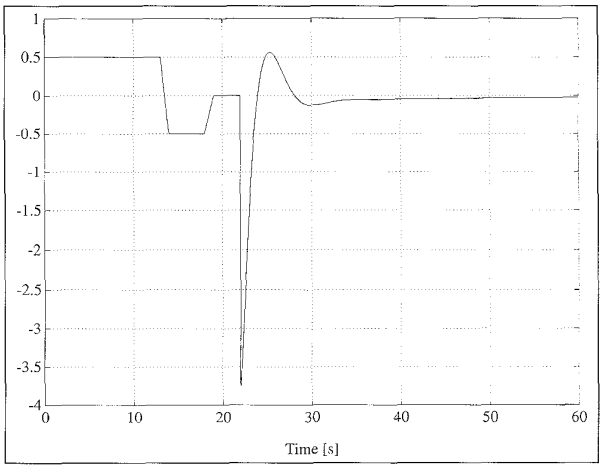

Fig. 5. Process input (u^r).

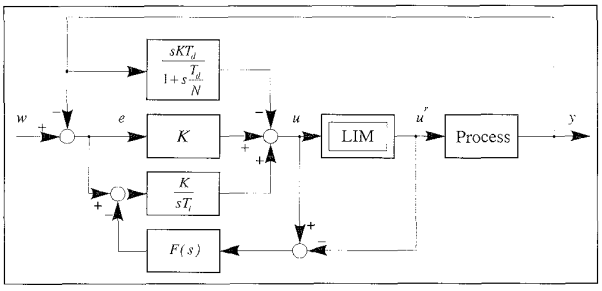

Fig. 6. Limited closed-loop system with AW.

Assume that the switch goes from automatic to manual control. If u^m is such that for some time $e > 0$, then the integral term increases in an uncontrolled way to very high values and u becomes high and much greater than u^m. Now, assume that the switch goes back from manual to automatic control. At that moment, even if $e = 0$, a big jump occurs at u^r, due to the high values of the integral term. Moreover, u decreases only if $e < 0$ for a sufficiently long time. This leads to a long settling time of the process output.

To illustrate the above phenomenon, we have made a simulation which is referred to as Sim. 2, with process 1, controller 1, whose parameters are given in the appendix, and the reference signal taken as 0. The process is manually controlled in the period from 0 to 22 sec. Then, its input is switched to the PID controller. The results of the simulation are shown in Figs. 4 and 5.

From Figs. 4 and 5 it can be seen that, at the instant of switching, a big jump occurs at the process input, and this also causes a long settling time of the process. This mode switching with a jump at the plant input is called bump transfer.

Review of Some Existing AW (BT, CT) Algorithms

Anti-Windup

In fact, windup appears due to the fact that the integral term increases too greatly during saturation. Thus, during saturation, the increase should be slowed down. This can be realized by an extra compensation that feeds back u - u^r to the integral term, through a compensator with transfer function $F(s)$ (Fig. 6). As this compensation aims at reducing the effect of windup, it is called anti-windup.

If the compensator is taken as $F(s) = 1/K_a$ where K_a is a prescribed constant, the scheme described by Fig. 6 is referred to as the linear feedback AW algorithm.

Realizable Reference

The realizable reference w^r is such that if it had been applied to the controller instead of the reference w, the control output u would have been equal to the real plant input u^r obtained with the reference w. In this case, the limitation is not activated.

If w^r is used in the control scheme described by Fig. 6, by definition, the limiter is not activated (u^r is always the same as u), so it can be put away as shown in Fig. 7. Moreover, u^r and y described in Fig. 7 are equivalent to u^r and y described in Fig. 6, respectively. We can see that the control scheme described by Fig. 7 does not include any implicit non-linearity. The non-linearity is hidden in the realizable reference w^r. From Fig. 7, it can clearly be seen that y tracks w^r instead of w, with the expected linear performance.

From the definition of the realizable reference, we have

$$u^r = K\left(1+\frac{1}{sT_i}\right)w^r - K\left(1+\frac{1}{sT_i}+\frac{sT_d}{1+\frac{sT_d}{N}}\right), \tag{4}$$

this yields

$$w^r = \frac{s^2T_iT_d\left(1+\frac{1}{N}\right)+s\left(T_i+\frac{T_d}{N}\right)+1}{(1+sT_i)\left(1+\frac{sT_d}{N}\right)}y + \frac{sT_i}{K(1+sT_i)}u^r. \tag{5}$$

During the limitation, u^r is the same for whatever $F(s)$, and so is y, if the initial conditions are the same. Hence, from (5), it can be seen that during limitation, w^r is the same for whatever $F(s)$. For the linear feedback AW algorithm, we have:

$$u = K\left(1+\frac{1}{sT_i}\right)w - K\left(1+\frac{1}{sT_i}+\frac{sT_d}{1+\frac{sT_d}{N}}\right)y + \frac{K}{sT_i}\left(\frac{u^r-u}{K_a}\right). \tag{6}$$

Subtracting (6) from (4), we can calculate w^r as

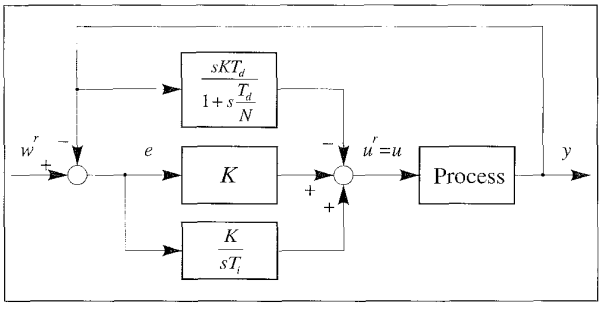

Fig. 7. Equivalent scheme of Fig. 6.

$$w^r = w + \frac{K + sK_aT_i}{KK_a(1+sT_i)}\left(u^r - u\right) = w + G_w(s)\left(u^r - u\right) \ . \tag{7}$$

As $G_w(s)$ is a dynamic transfer function with a pole and a zero, w^r will not become the same as w at the instant when the controller leaves the limitation ($u^r = u$), unless $G_w(s)$ is reduced to a static gain. Indeed, when $K_a = K$, (7) yields:

$$G_w(s) = \frac{1}{K} \ , \tag{8}$$

$$w^r = w + \frac{u^r - u}{K} \ . \tag{9}$$

At the instant the controller leaves the limitation ($u^r = u$), w^r becomes w. This choice thus gives the best tracking performance.

Observer Approach

This AW approach was first presented in [2].

The PID controller described in (3) can also be described by the following state-space equations:

$$\dot{x} = w - y = e \ , \tag{10a}$$

$$u = \frac{K}{T_i}x + Ke - y_d \ , \tag{10b}$$

$$u^r = LIM(u) \ , \tag{10c}$$

where x is the controller state and y_d is the output of D-term described by

$$Y_d(s) = \frac{sKT_d}{1 + s\frac{T_d}{N}}Y(s) \ . \tag{11}$$

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 [1, 2, 13]. To correctly estimate the state when $u^r \neq u$, an observer is introduced. The correction of the state is proportional to the difference between u and u^r through a static gain L:

$$\dot{x}_e = e + L\left(u^r - u\right) \ , \tag{12a}$$

$$u = \frac{K}{T_i}x_e + Ke - y_d \ , \tag{12b}$$

$$u^r = LIM(u) \ , \tag{12c}$$

where x_e is the estimated state.This is the same as the linear feedback AW algorithm by noting $K_a = 1/L$. Indeed, the linear feedback AW has an inherent observer property as pointed out by Walgama and Sternby [17].

Conditioning Technique

This AW approach was first presented in [5] as an extension of the back calculation method proposed by Fertik and Ross [4].

If we apply the realizable reference w^r to the controller instead of w, (10) yields

$$\dot{x}_c = w^r - y \ , \tag{13a}$$

$$u^r = \frac{K}{T_i}x_c + K\left(w^r - y\right) - y_d \ . \tag{13b}$$

where x_c is the new controller state when using w^r.

As w^r is not available a priori, we have to use w to update u as (10b). However, we can use w^r (computed a posteriori) instead of w to update the controller state in order to make it consistent. Thus

$$\dot{x}_c = w^r - y \ , \tag{14a}$$

$$u = \frac{K}{T_i}x_c + K(w - y) - y_d \ , \tag{14b}$$

$$u^r = LIM(u) \tag{14c}$$

By subtracting (14b) from (13b), we have

$$w^r = w + \frac{u^r - u}{K} \ . \tag{15}$$

Substituting (15) for w^r in (14), the controller can be written

$$\dot{x}_c = w - y + \frac{u^r - u}{K} \ . \tag{16a}$$

$$u = \frac{K}{T_i}x_c + K(w - y) - y_d \ , \tag{16b}$$

$$u^r = LIM(u) \tag{16c}$$

This is a special case of the linear feedback AW algorithm in which $K_a = K$. As has been discussed in the sub-section on the realizable reference, this choice results in the best tracking performance. Some extensions of the conditioning technique are given in [8, 9, 16].

Incremental Algorithm

The incremental algorithm is very often used to prevent windup in practice [1, 6]. It is also a relatively simple method to incorporate in a digital controller. Fig. 8 shows a typical discrete-time implementation.

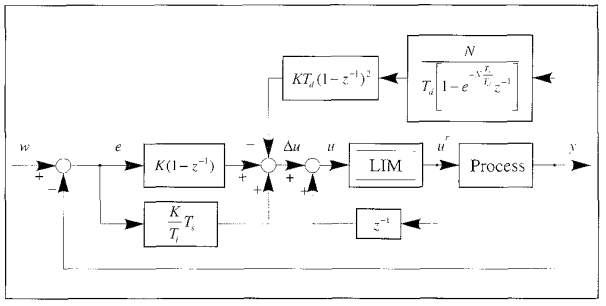

Fig. 8. Incremental algorithm (discrete-time implementation with sampling time TS).

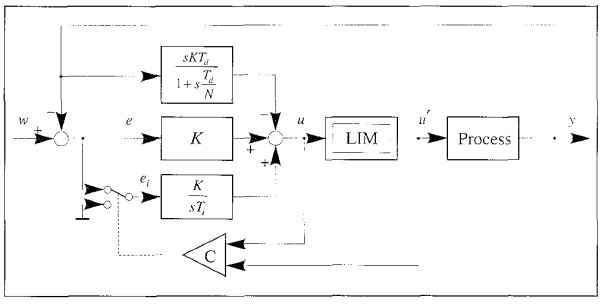

Fig. 10. Conditional integration method.

Using the incremental algorithm, $u(k)$ is updated as

$$u(k) = u^r(k - 1) + \Delta u(k) . \tag{17}$$

If the sampling time is very small, $\Delta u(k)$ and $u(k) - u^r(k)$ are almost zero except at the instant of reference change. By block manipulation, the discrete-time implementation of the incremental algorithm can be expressed in another way, as in Fig. 9.

The discrete-time implementation can be transformed into its continuous-time equivalent by decreasing sampling time T_S:

$$\frac{T_i}{KT_s}z^{-1} \rightarrow \frac{T_i}{KT_s}e^{-sT_S} , \tag{18a}$$

$$\lim_{T_S\rightarrow 0}\left[\frac{T_i}{KT_S}e^{-sT_S}\right] = \lim_{T_S\rightarrow 0}\left[\frac{T_i}{KT_S}\right] \rightarrow \infty . \tag{18b}$$

Thus, the continuous-time implementation can be represented as a special case of the linear feedback AW algorithm with $K_a \rightarrow 0$.

Conditional Integration

Fig. 10 and (19) depict the conditional integration [1, 6].

$$e_i = \begin{cases} e; & u^r = u \\ 0; & u^r \neq u \end{cases} . \tag{19}$$

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,

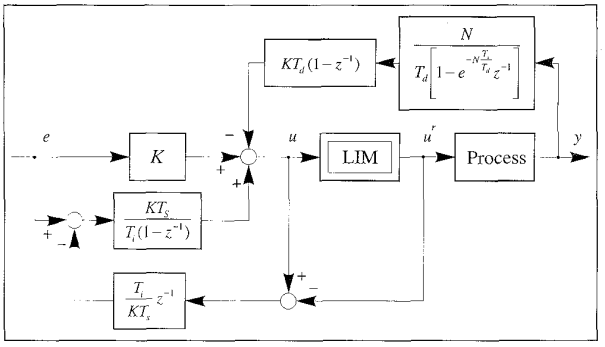

Fig. 9. Discrete-time equivalent of the incremental algorithm.

otherwise ($u^r \neq u$), the comparator switches e_i to 0 (the updating of the integral term stops).As the controller has a non-linear feedback function, the conditional integration is referred to as a non-linear feedback AW method. It can be expressed in a form similar to a linear feedback AW method by setting:

$$K_a \begin{cases} = \dfrac{u - u^r}{e}; & u^r \neq u \\ \neq 0; & u^r = u \end{cases} . \tag{20}$$

Note that K_a is, however, a time-varying term. It can be shown that $0 < K_a < K$ for PI controller [14]. This is also true for the PID controller except that, for non-minimum phase process, it might happen that $K_a > K$. Hence, we can expect that the response of the conditional integration would lie between the responses of the incremental algorithm and the conditioning technique. If it happens that $K_a > K$, then the AW capacity will be reduced, since the AW feedback gain is inversely proportional to K_a.

Comparative Study

To compare the mentioned AW algorithms, we make again the same simulation as Sim. 1, except that different AW strategies are used.

Figs. 11 and 12 show the difference between the AW algorithms. It is clearly seen from Fig. 11 that the conditioning technique gives a w^r that is the closest to w. Fig. 12 demonstrates that y tracks w^r instead of w. It is also seen that the response of the conditional integration lies between the responses of the incremental algorithm and the conditioning technique.

Bumpless Transfer and Conditioned Transfer

Now, let us investigate the case of mode switching. During manual mode ($u^r = u^m$), as the controller is not connected to u^m, its output is usually quite different from u^r. In this case, after switching, a jump will be produced at the plant input (bump transfer). To remove the jump, the controller output u should be made as close as possible to u^m during manual mode. Then the jump at the instant of switching will be minimized. This mode switching is called bumpless transfer. Yet there is no guarantee that the tracking performance will be good after mode switching.

If the controller output u is adjusted so that after switching from manual to automatic control, the plant output y tracks the reference w with the same dynamics as the closed-loop step response, then this mode switching is called conditioned transfer.

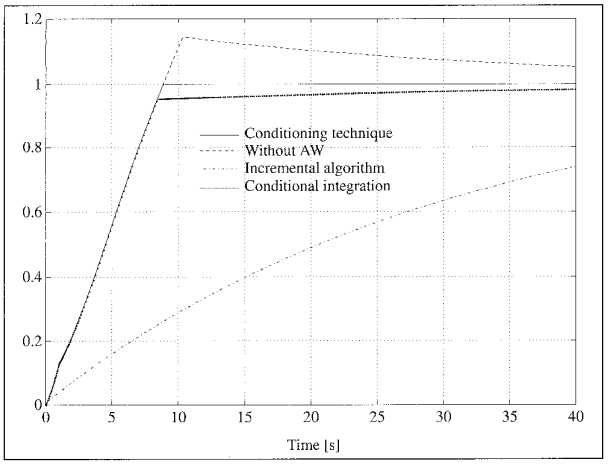

Fig. 11. Realizable reference (w^r).

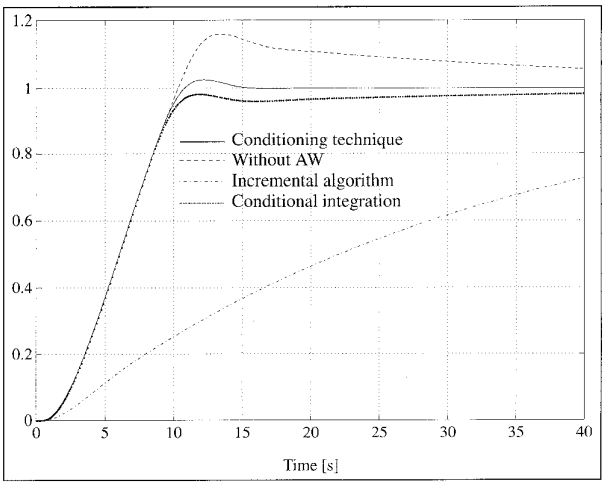

Fig. 12. Process output (y).

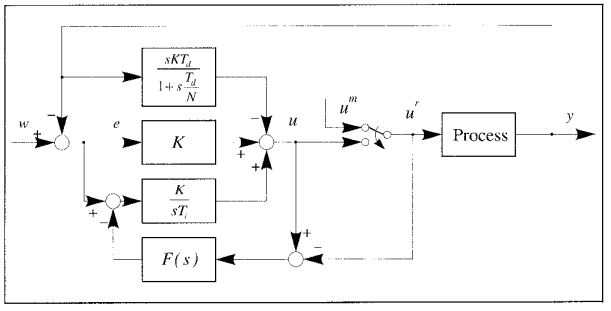

Fig. 13. Bumpless and conditioned transfer from manual to automatic mode.

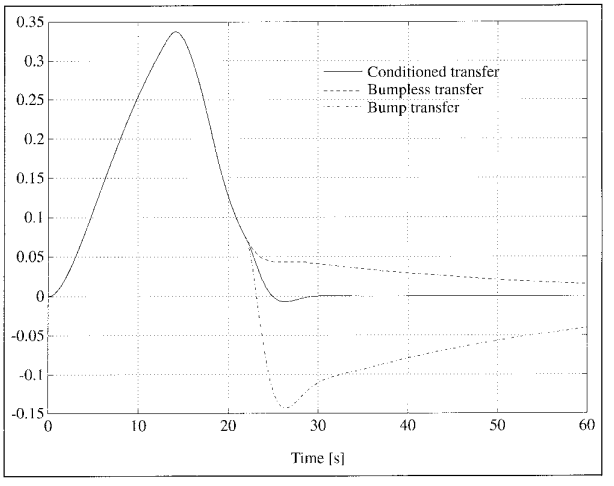

Fig. 14. Process output (y).

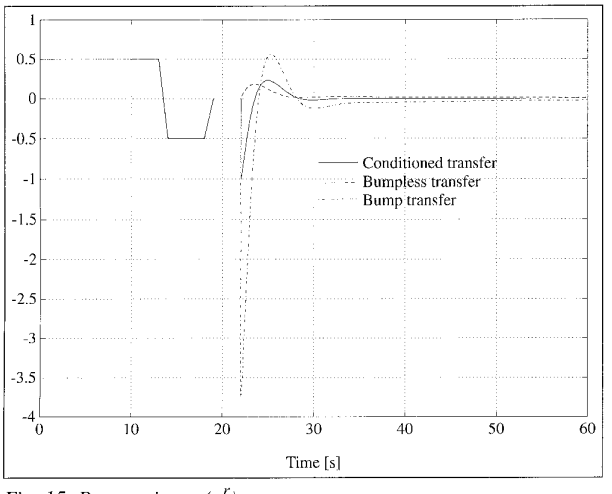

Fig. 15. Process input (u^r).

In other words, after switching, good tracking performance will be assured when using conditioned transfer. Note that the jump is usually small but not minimized in this case.

An anti-windup strategy is usually implemented as a bumpless transfer technique as shown in Fig. 13. However, we will see that only the incremental algorithm is a solution for BT, while the conditioning technique is a solution for CT.

Indeed, using the incremental algorithm ($K_a \rightarrow 0$), during manual mode, referring to (17), we can see that u is made nearly equal to u^r. Thus, the incremental algorithm will not produce a jump at u^r at the time of switching, and can be used as a BT method.

The realizable reference w^r for BT and CT methods can be defined in the same way as for AW scheme. Figs. 13 and 7 then represent the equivalent schemes from the process viewpoint. Note that the realizable reference is defined for all time. During manual mode ($u^r \neq u$), w^r is different from w, and y tracks w^r. After switching to automatic mode ($u^r = u$), we want $w^r = w$ so that y will track w with the same dynamics as the closed-loop step response (CT). The only way to do this at the instant of switching is to use the conditioning technique ($K_a = K$). Usually, 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. Yet if a jump is not tolerable, we can either switch from manual to automatic mode when u is close to

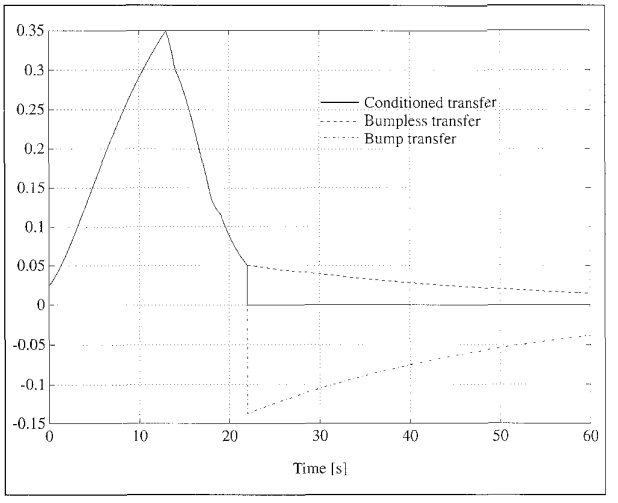

Fig. 16. Realizable reference (w^r).

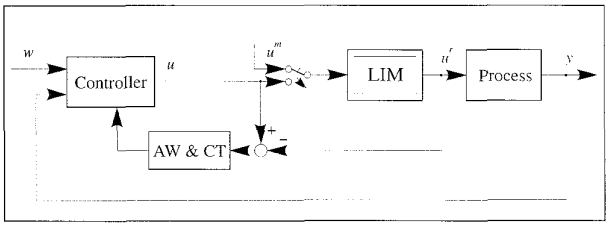

Fig. 17. Anti-windup and conditioned transfer.

u^m (by driving y close to w before switching) or add a rate limitation at the process input [14].

To support the above arguments, we have made again the same simulation as Sim. 2, except that different AW strategies are used. The results are shown in Figs. 14 to 16.

It can be seen that the incremental algorithm (BT) produces no jump at the process input (u^r) at the instant of switching from manual to automatic mode (Fig. 15), but the settling time of the closed-loop response is quite long (Fig. 14). On the other hand, the conditioning technique (CT) yields a short settling time at the cost of producing a small jump at the process input. It can also be seen that only the conditioning technique can make $w^r = w$ at the instant of switching from manual to automatic mode (Fig. 16).

Discussion

Generalization

Fig. 17 shows the way to realize AW and BT or CT in the same time. In fact, the solution can be generalized for all controllers with relatively slow or unstable modes. As pointed out by Doyle et al. [3], those controllers with relatively slow or unstable modes will experience windup problems if there are actuator constraints. In fact, in the case of mode switching, they will experience bump transfer problems too.

Generating the Real Process Input

Measuring the real process input u^r requires additional controller input, cables, and filters that would be too expensive. In most controllers, this problem can be solved by estimating the process input limitations inside the controller. We can put an estimated limitation LIM′ in front of the real limitation LIM (Fig. 18). If LIM′ is the same as LIM, then $u^{r'}$ is already limited by LIM′ and will not be limited again by LIM. Thus, u^r will become the same as $u^{r'}$. LIM′ can be even more "restrictive" than LIM, and $u^{r'}$ would still remain equal to u^r.

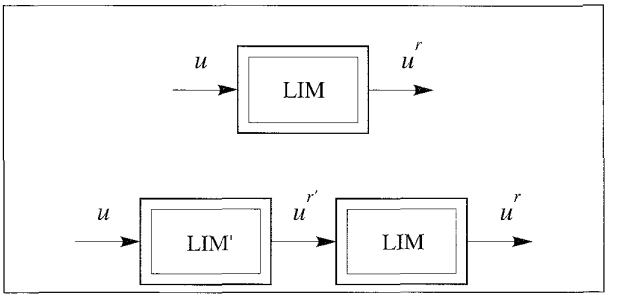

Fig. 18. Estimated limitation and real limitation.

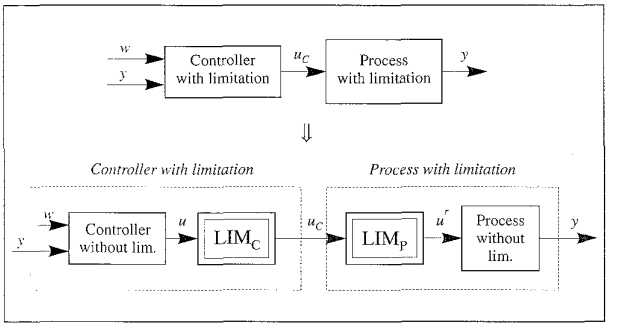

Fig. 19. Representation of the limited system.

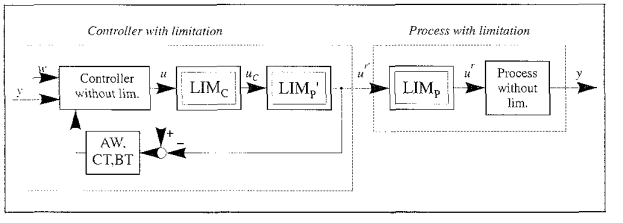

Fig. 20. Estimated process limitation inside controller.

The limited controller and process can be represented as shown in Fig. 19. We can put an estimation of the process limitation (LIM'_P) into the controller as shown in Fig. 20. In this case, the controller output signal ($u^{r'}$) will become the same as u^r as long as LIM'_P is adequate.

Therefore, the controller output signal $u^{r'}$ can be used for AW (BT, CT) algorithms instead of u^r. If LIM_P cannot be estimated accurately, we can use a more "restrictive" estimation. However, if a too "restrictive" estimation is used, the system may become oscillatory, as will be shown shortly.

Dealing with Too Restrictive Process Limitations

As pointed out by Ronnback et al. [12] and Walgama et al. [16], when process input limitations are too restrictive, using the

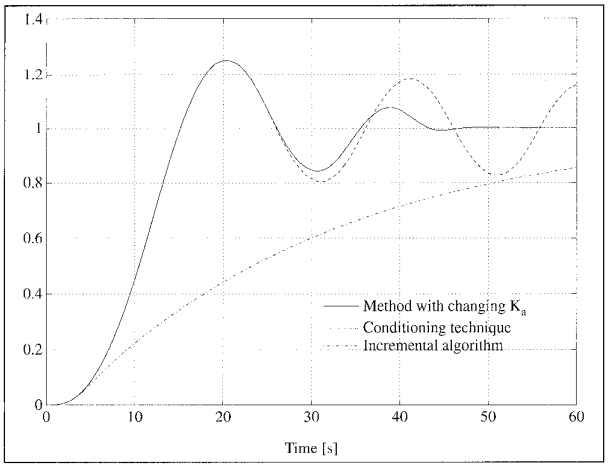

Fig. 21. Process output (y).

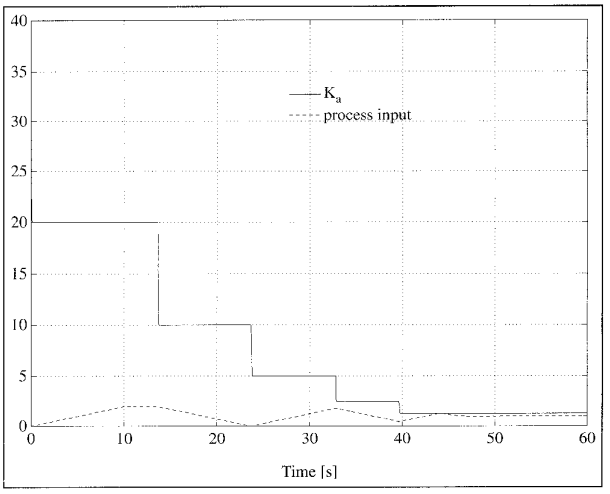

Fig. 22. Method with changing K_a.

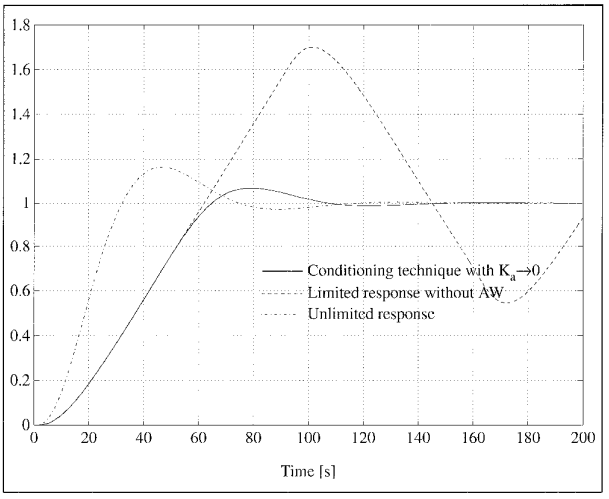

Fig. 23. Process output (y) using I controller.

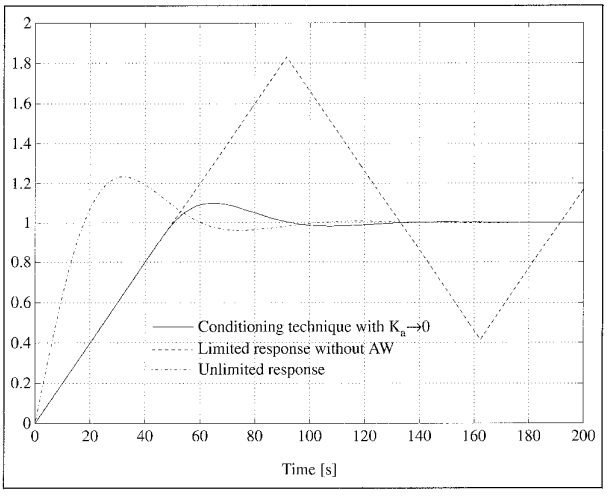

Fig. 24. Process input (u^r) using I controller.

conditioning technique might produce multiple opposite limitations at the process input and thus lead to oscillations at the process output. If this happens, the controller parameters should be tuned so as to decrease the oscillations (more sluggish controller) [14].

Another solution to the above problem is to use an AW algorithm with $K_a << K$ (e.g. the incremental algorithm). The drawback of such a method is the degradation of the tracking performance. This problem could be alleviated by first using the conditioning technique and reducing the constant K_a during multiple opposite saturation. As a rule of thumb, K_a could be reduced by a factor 2 if the limitation changes sign (from positive to negative or vice versa). To illustrate this idea, we make a simulation with the data used for Sim. 1, except that the input limits are more restrictive: $u_{max} = 2$, $u_{min} = 0$, $v_{max} = 0.2$ sec^{-1} and $v_{min} = -0.2$ sec^{-1}.

The results are presented in Figs. 21 and 22. We can see that the tracking performance of the proposed method (with changing K_a) is better than the results obtained with the incremental algorithm.

AW for I Controller

What kind of AW algorithm should be used if we have an I controller instead of a PI or a PID?

For the conditioning technique, K_a is equal to K. As the proportional gain (K) in an I controller is equal to 0, the conditioning technique should be implemented by taking $K_a \to 0$. This corresponds to the incremental algorithm. Thus, for an I controller, the conditioning technique can be realized by the incremental algorithm. Figs. 23 and 24 show an example where an I controller is used. We can see that the AW solution with $K_a \to 0$ works quite well.

In this simulation, we have used process 1 and controller 2 whose parameters are given in the appendix. The input limits are $v_{max} = 0.02$ sec^{-1} and $v_{min} = -0.02$ sec^{-1}.

Conclusions

We have illustrated, through simulations for PID controllers, the phenomenon of windup and bump transfer, and the improved results obtained using the techniques of anti-windup, bumpless transfer, and conditioned transfer. The majority of existing anti-

windup, bumpless, and conditioned transfer techniques have been reviewed in the framework of the PID controller. Using the so-called realizable reference, we have shown that the conditioning technique is the most suitable anti-windup method for usual applications. The exception is the case in which the input limitations are so restrictive that the system output might become oscillatory. In such a case, the controller parameters could be changed in the design stage to damp the oscillations, or an anti-windup method with a tuning parameter K_a (tuned to obtain a compromise between the incremental algorithm and the conditioning technique) could be used.

Two types of transfers in the case of mode switching are described. Although bumpless transfer is a well-known concept, its resulting tracking performance might be degraded. The new notion of conditioned transfer is thus introduced for the first time in this article. Conditioned transfer assures good tracking performance at the cost of a possible small jump at the plant input during mode switching.

Utilizing many other simulations [14], we have tested the above conclusions and found them to be always valid if the closed-loop system demonstrates satisfactory performance in the unlimited case, no matter which process and controller are used.

Acknowledgment

The authors would like to thank Dave Pepper, Michel Kinnaert, and Đani Juricic for carefully correcting the article. They also thank the anonymous reviewers for providing many useful comments.

Appendix

The numerical values of the process and controllers used for simulations:

Process 1:

$$G(s) = \frac{1}{(1+8s)(1+4s)},$$

where $G(s)$ is the process transfer function. Using a pole-placement method [15], we derived the following PID controller for process 1:

Controller 1:

$K = 20$, $T_i = 30$ sec, $T_d = 0.95$ sec, $N = 10$.

By manual tuning [15], we also derived the following I controller for process 1:

Controller 2:

$$G_c(s) = \frac{1}{sT_i},\ T_i = 5 \text{ sec},$$

where $G_C(s)$ is the I-controller transfer function.

References

[1] K.J. Åström and L. Rundqwist, "Integrator Windup and How to Avoid It," *Proceeding of the 1989 American Control Conference*, Pittsburgh, pp. 1693-1698, 1989.

[2] K.J. Åström and B. Wittenmark, *Computer Controlled Systems: Theory and Design*, Prentice Hall, Englewood Cliffs, NJ, 1984.

[3] J.C. Doyle, R.S. Smith, and D.F. Enns, "Control of Plants with Input Saturation Nonlinearities," *Proceeding of the 1987 American Control Conference*, Minneapolis, pp. 1034-1039, 1987.

[4] H.A. Fertik and C.W. Ross, "Direct Digital Control Algorithms with Anti-Windup Feature," *I.S.A. Trans.*, 22nd annual conf. and. exhibit., 6, no. 4, pp. 317-328, Chicago, 1967.

[5] R. Hanus, "A New Technique for Preventing Control Windup," *Journal A*, vol. 21, no. 1, pp. 15-20, 1980.

[6] R. Hanus, "Anti-Windup and Bumpless Transfer: A Survey," *Computing and Computers for Control Systems*, P. Borne et al., eds., IMACS, vol. 4, pp. 3-9, 1989.

[7] R. Hanus, M. Kinnaert, and J.L. Henrotte, "Conditioning Technique, A General Anti-Windup and Bumpless Transfer Method," *Automatica*, vol. 23 (6), pp. 729-739, 1987.

[8] R. Hanus and Y. Peng, "Modified Conditioning Technique for Controller with Nonminimum Phase Zeros and/or Time-Delays," *Preprints of the 13th IMACS World Congress on Computation and Applied Mathematics*, Dublin, vol. 3, pp. 1188-1189, 1991.

[9] R. Hanus and Y. Peng, "Conditioning Technique for Controllers with Time Delays," *IEEE Transactions on Automatic Control*, vol. 37 (5), pp. 689-692, 1992.

[10] M.V. Kothare, P.J. Campo, M. Morari, and C.N. Nett, "A Unified Framework for the Study of Anti-Windup Designs," *Automatica*, vol. 30 (2), pp. 1869-1883, 1994.

[11] M. Morari, "Some Control Problems in the Process Industries," *Essays on Control: Perspectives in the Theory and its Applications*, H.L. Trentelman and J.C. Willems, eds., Birkhäuser, pp. 57-66, 1993.

[12] S. Ronnback, K.S. Walgama, and J. Sternby, "An Extension to the Generalized Anti-Windup Compensator," *Preprints of the 13th IMACS World Congress on Computation and Applied Mathematics*, Dublin, vol. 3, pp. 1192-1196, 1991.

[13] L. Rundqwist, "Anti-Reset Windup for PID Controllers," *Preprints of the 11th IFAC World Congress*, vol. 8, pp. 146-151, 1990.

[14] D. Vrancic, Y. Peng, and Đ. Juricic, "Some Aspects and Design of Anti-Windup and Conditioned Transfer," *Report DP-7169*, J. Stefan Institute, Ljubljana, 1995.

[15] D. Vrancic, J. Petrovcic, and Đ. Juricic, "PID Parameter Settings for the Second Order Processes (in Slovene)," *Report DP-6782*, J. Stefan Institute, Ljubljana, 1993.

[16] K.S. Walgama, S. Ronnback, and J. Sternby, "Generalisation of Conditioning Technique for Anti-Windup Compensators," *IEE Proceedings*, part D, 139, pp. 109-118, 1992.

[17] K.S. Walgama and J. Sternby, "Inherent Observer Property in a Class of Anti-Windup Compensators," *International Journal of Control*, vol. 52, no. 3, pp. 705-724, 1990.

Youbin Peng received the B.S. degree in electrical engineering from Huazhong University of Science and Technology, Wuhan, China, in 1984. He received the M.S. degree in automatic control engineering and the Ph.D. degree in applied sciences from Free University of Brussels (ULB), Brussels, Belgium, in 1991. His research interests include adaptive control, predictive control, fault detection and diagnosis, and nonlinear control (with emphasis on anti-windup design).

Damir Vrancic received the B.S. and M.S. degrees in electrical engineering at the Faculty of Electrical and Computer Engineering, University of Ljubljana, Slovenia, in 1991 and 1995, respectively. Since 1992, he has been with the Department of Computer Automation and Control at J. Stefan Institute. His research interests include anti-windup, PID control, modeling, simulation and design of electronic circuits.

Raymond Hanus graduated in physical engineering and in automatic control engineering at Free University of Brussels (ULB), Brussels, Belgium, in 1970 and 1972, respectively. He received the Ph.D. degree in applied sciences from the same university in 1979. He is currently professor and director of the Department of Automatic Control Engineering at ULB. His main research interests include identification and control of nonlinear systems.

Correction

The figure accompanying Sanjoy K. Mitter's article in our last issue ("Filtering and Stochastic Control: A Historical Perspective," June 1996 *CS*, p. 67) contained several errors. The figure is reproduced correctly here. We regret the errors.—*Ed.*