Leggi di controllo PID

Codifica software

Equivalenti discreti dei regolatori

Necessità di «discretizzare»

Per codificare le azioni PID in software eseguibile da un microprocessore o microcontrollore, è necessario prima di tutto ottenerne equivalenti discreti (a tempo discreto). Il microcontrollore esegue l'algoritmo PID a tempo discreto ad ogni passo di campionamento utilizzando i valori delle variabile acquisiti agli istanti di campionamento.

Il codice relativo all'algoritmo deve essere completato per tener conto di situazioni operative diverse da quelle nominali, in particolare la condizione di saturazione della variabile di controllo e la commutazione automatico manuale e viceversa (quando presente).

Equivalenti discreti dei regolatori

Come «discretizzare»

Consideriamo prima il problema della discretizzazione delle equazioni che esprimono le azioni PID. L'operazione può essere fatta secondo vari metodi numerici.

La formula di **Eulero all'indietro** (o **formula implicita** di Eulero) è preferibile. Infatti, la formula di **Eulero in avanti** può dare un equivalente discreto instabile di un sistema continuo asintoticamente stabile, se il periodo di campionamento non è sufficientemente piccolo in rapporto alla più piccola costante di tempo del sistema.
Non è quindi adatta per la discretizzazione del termine derivativo filtrato, in cui la costante di tempo del filtro T_D/N è, per progetto, solitamente piccola, rispetto alla costante di tempo dominante del sistema ad anello chiuso.

La **formula del trapezio**, la più accurata delle tre a parità di periodo di campionamento, ha l'inconveniente di dar luogo a "*ringing*", cioè l'uscita che rimbalza sopra e sotto il valore di equilibrio da un instante di campionamento al successivo, se il periodo di campionamento, che denotiamo con *h,* si avvicina alla costante di tempo del filtro.

Equivalente discreto con formula implicita

Azioni proporzionale e integrale

Azione proporzionale

L'equivalente discreto dell'azione proporzionale $u_P(t) = K_C e(t)$ si ricava semplicemente scrivendo la stessa agli istanti di campionamento:

$u_P(kh) = K_C e(kh)$, essendo h il periodo di campionamento kh l'istante di tempo del *k*-esimo campionamento.

Azione integrale

Scritta in forma differenziale, l'azione integrale è data da:

$$u(t) = \frac{K_C}{T_I}\int_0^t e(r)dr + u(0)$$

$$\frac{du_I(t)}{dt} = \frac{K_C}{T_I} e(t).$$

Applicando la formula implicita di Eulero si trova:

$$u_I(kh) = u_I(kh - h) + K_C \frac{h}{T_I} e(kh).$$

Si noti che per il calcolo al primo istante di campionamento è necessario conoscere $u_I(0)$.

Equivalenti discreti dei regolatori

Azione derivativa e totale PID

Azione derivativa filtrata

L'azione derivativa filtrata è descritta dalla seguente equazione differenziale:

$$\frac{T_D}{N}\frac{du_D(t)}{dt} + u_D(t) = K_C T_D \frac{de(t)}{dt}.$$

$$D_F(s) = \frac{U}{E} = \frac{K_C s T_D}{1 + sT_D/N}$$

Applicando la formula implicita si ottiene:

$$\frac{T_D}{Nh}(u_D(kh) - u_D(kh-h)) = -u_D(kh) + K_C\frac{T_D}{h}(e(kh) - e(kh-h))$$

che, con semplici passaggi, fornisce:

$$u_D(kh) = \frac{T_D}{Nh+T_D}u_D(kh-h) + K_C\frac{NT_D}{Nh+T_D}(e(kh) - e(kh-h)).$$

Per il calcolo al primo istante di campionamento occorre avere $u_D(0)$ e $e(0)$.

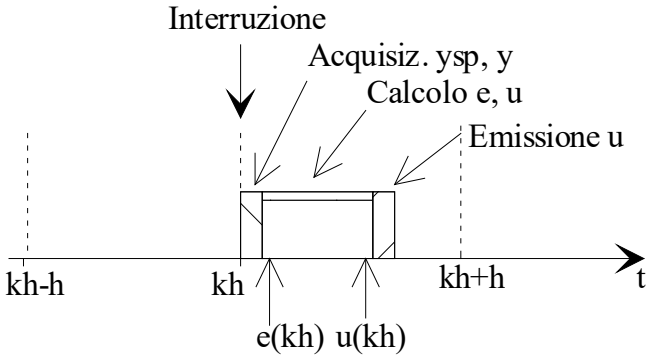
L'equivalente discreto del PID secondo la formula implicita è dato dalla somma degli equivalenti delle tre azioni. Detta *u*(*kh*) l'uscita del regolatore all'istante *kh* si ha:

$$u(kh) = u_P(kh) + u_I(kh) + u_D(kh).$$

Equivalenti discreti dei regolatori

Convenzione sulle notazioni

A causa dei tempi di elaborazione e calcolo, con le notazioni $e(kh)$ e $u(kh)$ si indicano sequenze di numeri che sono calcolati in realtà in istanti diversi nell'ambito di un periodo di campionamento. La convenzione adottata è illustrata in figura.



All'istante di campionamento *kh* un orologio (*clock*) dà il via all'acquisizione degli ingressi, cui segue il calcolo dell'errore $e(kh)$.
Viene eseguito quindi l'algoritmo di controllo ottenendo la variabile di controllo, denominata $u(kh)$ anche se è emessa ad un istante di tempo compreso tra *kh* e *kh+h*.
Per mantenere una maggiore corrispondenza con l'espressione a tempo discreto dell'algoritmo si potrebbe ritardare l'emissione della variabile di controllo fino all'arrivo dell'interruzione per il campionamento successivo *kh+h*. In questo caso la variabile di controllo sarebbe correttamente denotata con $u(kh+h)$, con lo svantaggio però di un inutile maggiore ritardo nell'emissione della variabile di controllo.
E' difficile stabilire, in generale, quale sia la scelta migliore. Le differenze comunque sfumano al diminuire del periodo di campionamento.

Pseudocodice di un regolatore PID

Un possibile pseudocodice, per il calcolo dell'equivalente discreto del PID appena ricavato, e coerente con la convenzione adottata, è il seguente: comprende inizializzazione e parte ricorsiva

Inizializzazione

% Assegnamento e calcolo dei coefficienti costanti

$a_1 = K_c\, h/T_I$

$b_1 = T_D/(Nh + T_D)$

$b_2 = K_C \mathrm{N}\, b_1$

% Inizializzazione dello stato

% si suppone che y_{sp} sia in una memoria interna al regolatore

acquisizione e conversione A/D di y

$e_{old} = y_{sp} - y$

$u_I = u_0 - K_C e_{old} - a_1 e_{old}$

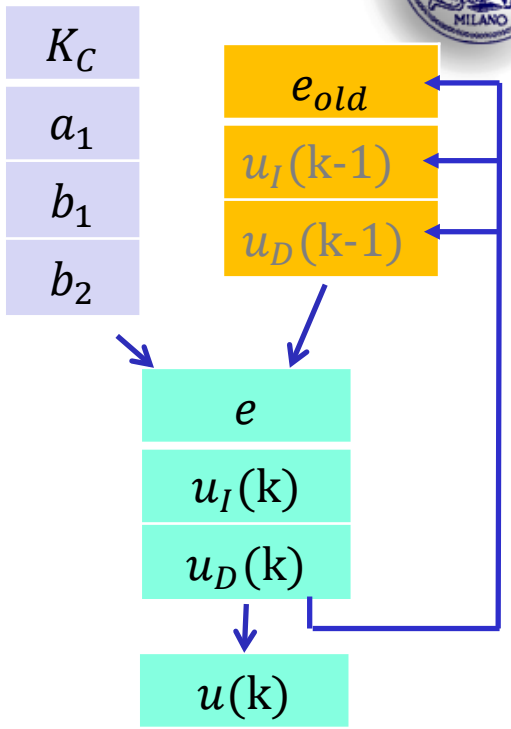
$u_D = 0$

% u_0 corrisponde al valore attuale della variabile di controllo, supposto disponibile. Con l'assegnamento di u_I come sopra, se l'errore rimane invariato, e al campionamento successivo si passa in automatico, la variabile u calcolata sarà uguale a quella (manuale) del campionamento attuale (commutazione M/A *bumpless*).

Pseudocodice di un regolatore PID

Codice ricorsivo

1 attesa attivazione dall'orologio *(clock interrupt)*
2 acquisizione e conversione A/D di y (y_{sp} in memoria)
3 $e = y_{sp} - y$
4 $u_I = u_I + a_1 e$
5 $u_D = b_1 u_D + b_2 (e - e_{old})$
6 $u = K_C e + u_I + u_D$
7 emissione di u e conversione D/A
8 $e_{old} = e$
9 ritorno a 1



Convenzionalmente il simbolo "=" indica assegnamento, ed i simboli di variabili e parametri sono da intendersi come nomi di variabili di programmazione (del codice nel linguaggio software che implementa l'algoritmo).
Come si vede il codice consiste di due parti, come sempre accade in un sistema di controllo in tempo reale: l'*inizializzazione* e il *codice ricorsivo*.
La prima è eseguita una sola volta all'accensione del sistema di elaborazione (o del sistema di controllo), il secondo ad ogni periodo di campionamento, e presenta criticità rispetto ai tempi di calcolo ed all'affidabilità di funzionamento, per cui dev'essere codificato e verificato con metodo e cura particolari.

Pseudocodice di un regolatore PID

Wind-up dell'integratore

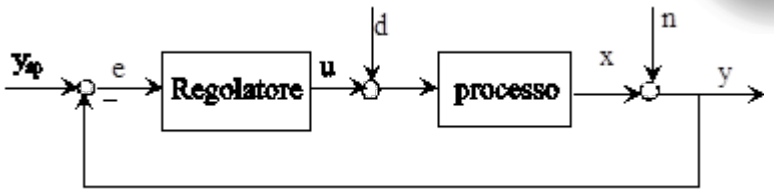
La variabile di controllo di un qualsiasi anello di regolazione è sempre limitata superiormente ed inferiormente. In condizione di regime, se il processo è ben progettato la variabile di controllo assume valori abbastanza lontani dai **limiti di *saturazione***, ma può capitare che li raggiunga in occasione di transitori ampi o rapidi, causati da variazioni rilevanti del setpoint e/o del disturbo di carico.

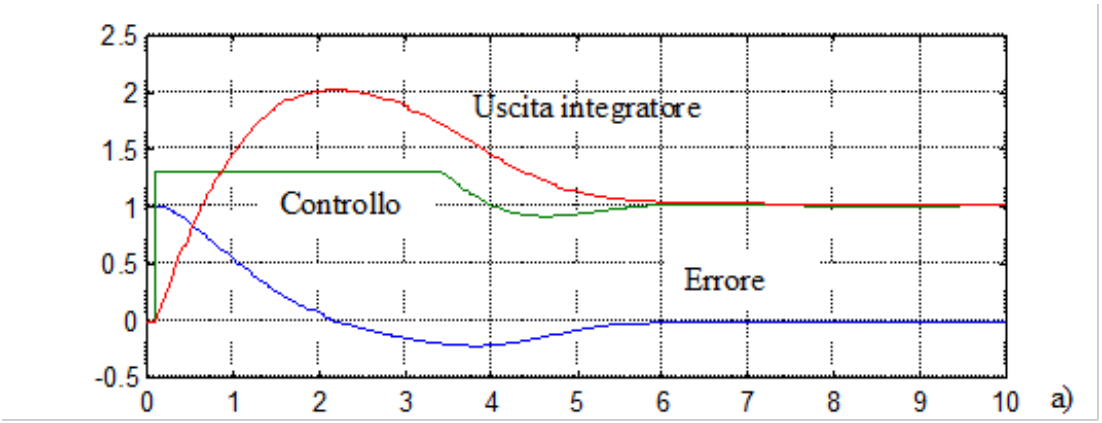
Quando la variabile di controllo è in saturazione, il processo evolve con ingresso costante, indipendentemente dall'azione del regolatore. A questo viene quindi a mancare l'effetto della retroazione sull'errore, e poiché **l'integratore è un sistema dinamico non asintoticamente stabile**, in questa condizione può allontanarsi anche notevolmente dal campo di valori utili per il controllo.

Se così succede, occorre tempo, dopo che l'errore è tornato a valori tali da non richiedere più la saturazione della variabile di controllo, prima che l'integratore, e con lui l'uscita del regolatore, ritorni ai valori utili per il controllo. Si parla in questo caso di "***wind-up***" **dell'integratore**.

Pseudocodice di un regolatore PID

Wind-up dell'integratore. Un esempio: sistema del II ordine con regolatore PI



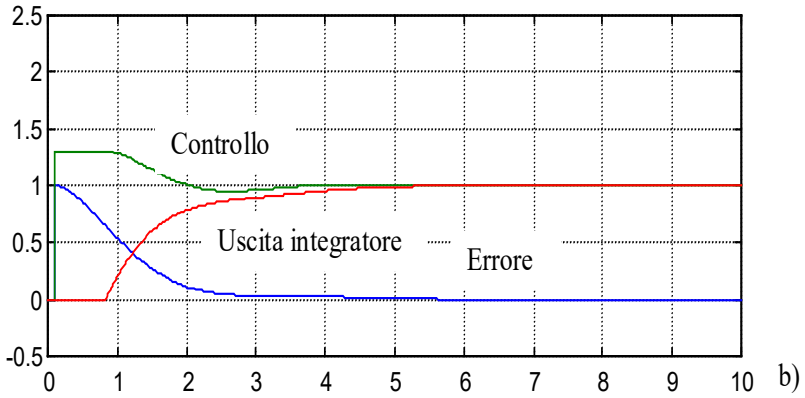


In figura sono riportati gli andamenti dell'errore, della variabile di controllo e dello stato (o uscita) dell'integratore in risposta a uno scalino sul setpoint.
Lo scalino applicato a 0.1 s manda in saturazione la variabile di controllo.
L'azione integrale cresce finché l'errore non cambia segno, arrivando oltre 2, valore ben superiore al limite di saturazione (1.3).
La variabile di controllo esce dalla saturazione dopo che l'errore è diventato negativo e tale è rimasto per circa 1.5 s, il tempo necessario per "riassorbire" la carica integrale, mentre, ragionevolmente, sarebbe dovuta uscire prima che l'errore si annullasse.
Per circa 1.5÷2 s il sistema ha perso la capacità di regolare.

Pseudocodice di un regolatore PID

Anti-Wind-up dell'integratore: integrazione condizionata

Un modo molto utilizzato per evitare il wind-up è quello detto **dell'integrazione condizionata**, e consiste nell'arrestare l'integrazione quando interviene la saturazione. La figura seguente riporta i risultati dell'esperimento precedente eseguito con questo accorgimento.



Si può notare che l'azione integrale interviene solo dopo che la variabile di controllo è rientrata dalla saturazione, con evidente riduzione della durata del transitorio e dell'errore di regolazione.
E' importante osservare che l'effettiva causa del wind-up è la saturazione dell'attuatore. Per condizionare correttamente l'integrazione occorre quindi individuare precisamente tale condizione e il modo migliore per farlo è, in generale, quello di usare dei dispositivi di rilevazione (ad esempio un contatto di fine corsa di un motore).
Per evitare il sensore addizionale, si utilizza spesso la saturazione propria (limiti 0 e 100%) dell'uscita del regolatore, o limiti software più stringenti approssimativamente corrispondenti alla saturazione degli attuatori.
Se la corrispondenza non è precisa, o si limita inutilmente la variabile di controllo o non si realizza il corretto condizionamento dell'integrazione.

Pseudocodice di un regolatore PID

Wind-up in AM v3.1.5 sect. (esempio: cruise control)

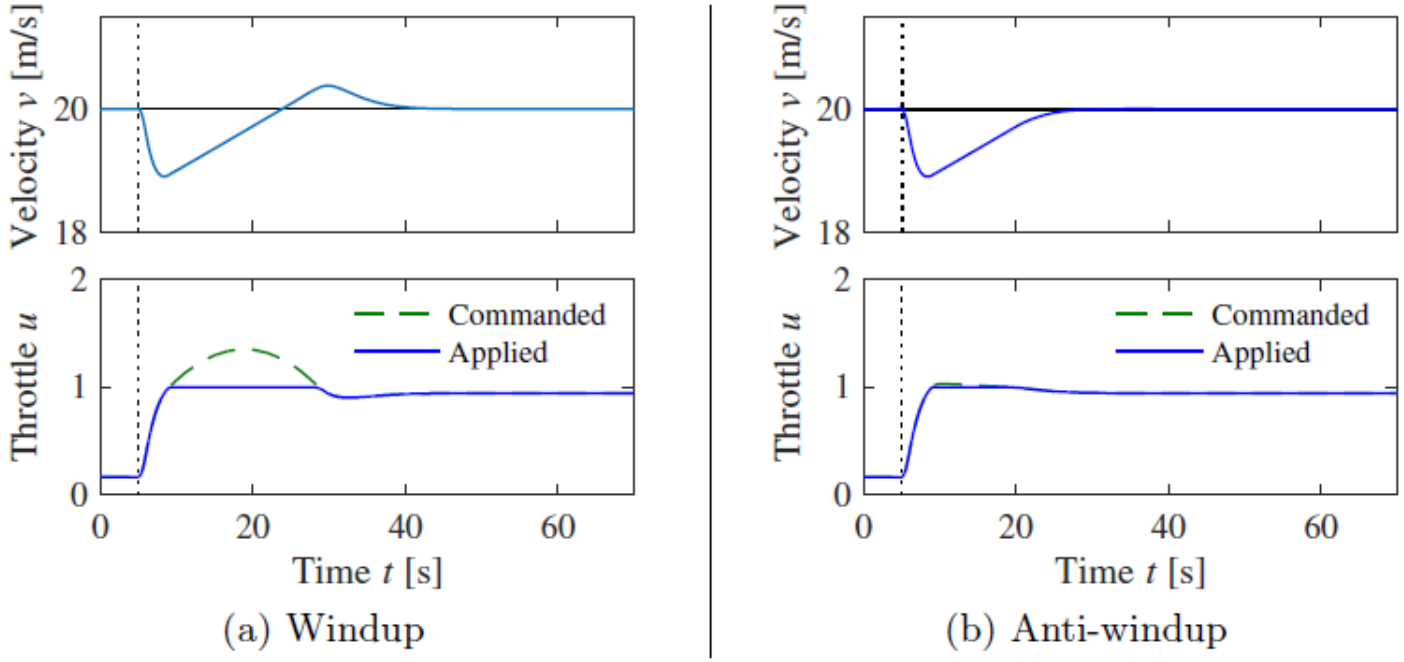
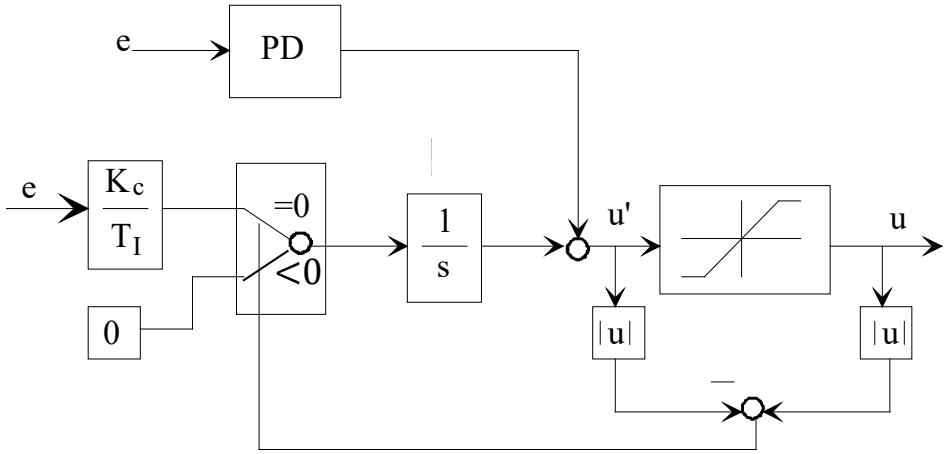


Figure 11.10: Simulation of PI cruise control with windup (a) and anti-windup (b). The figure shows the speed v and the throttle u for a car that encounters a slope that is so steep that the throttle saturates. The controller output is a dashed line. The controller parameters are $k_p = 0.5$, $k_i = 0.1$ and $k_{aw} = 2.0$. The anti-windup compensator eliminates the overshoot by preventing the error from building up in the integral term of the controller.

Pseudocodice di un regolatore PID

Anti-Wind-up dell'integratore: integrazione condizionata

Un regolatore PID con integrazione condizionata dell'errore sulla base dei propri limiti di saturazione è schematizzato in figura. Il calcolo dell'integrale è bloccato applicando zero all'ingresso dell'integratore tramite il blocco selettore.

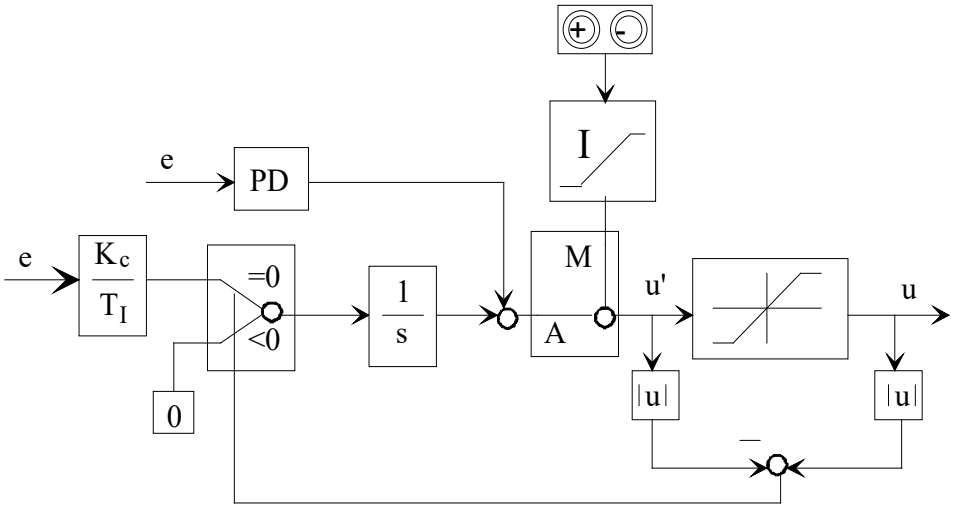


Per catene di regolazione complesse, a più livelli gerarchici o con anelli in cascata, la saturazione di un attuatore o di un regolatore comporta di arrestare l'integrazione in tutti i regolatori della struttura a monte. Il segnale logico di arresto può quindi provenire dal regolatore immediatamente a valle nella gerarchia.

Pseudocodice di un regolatore PID

Commutazione manuale/automatico e automatico/manuale

Gli operatori possono, con rare eccezioni essenzialmente legate a problemi di sicurezza, mettere in qualsiasi momento un anello di regolazione in manuale, oppure comandarne il passaggio in automatico.
Le commutazioni manuale / automatico (M/A) e automatico / manuale (A/M), devono avvenire senza brusche variazioni ("*bumpless*") della variabile di controllo, per evitare transitori indesiderati e danni agli attuatori. Un modo per risolvere il problema è schematizzato in figura.



Quando è selezionato il funzionamento manuale, l'operatore può comandare la variabile di controllo impostandone le variazioni mediante i tasti di *aumenta* / *diminuisci* (+ / -). Le variazioni sono sommate al valore totale attuale nella memoria I, fino all'eventuale raggiungimento dei valori limite (0 o 100%) della variabile di controllo.
La commutazione bumpless si ottiene calcolando, in ogni periodo di campionamento, correttamente lo stato dell'integratore nel funzionamento manuale o il valore della memoria I in automatico.
Ad esempio, nel funzionamento manuale **l'integratore sull'errore di regolazione si dice è messo in "inseguimento" della variabile *u*,** cioè in ogni periodo di campionamento è ricalcolato in maniera tale che, in caso di commutazione in automatico, e nell'ipotesi di errore costante, l'uscita calcolata coincida con la precedente manuale.

Pseudocodice di un regolatore PID

Regolatore PID completo. Codice ricorsivo

Pseudocodice, della sola parte ricorsiva, di un algoritmo PID con integrazione condizionata e la commutazione A/M e M/A senza colpi, con memoria addizionale.

1 attesa attivazione dall'orologio *(clock interrupt)*
2 acquisizione e conversione A/D di y_{sp}, y
3 $e = y_{sp} - y$
4 $u_D = b_1 u_D + b_2(e - e_{old})$
5 if AUTO then $u = K_C e + u_I + u_D$ else $u = u + \delta u_m$ endif
6 if $u > u_h$ then $u = u_h$
  elseif $u < u_l$ then $u = u_l$
  elseif AUTO then $u_I = u_I + a_1 e$
  else $u_I = u - K_C e - u_D$
  endif
7 emissione di u e conversione D/A
8 $e_{old} = e$
9 ritorno a 1

l'aggiornamento dell'integratore è ritardato di un passo, scelta discutibile se il periodo di campionamento è lungo e il contributo dell'azione integrale rilevante.

L'algoritmo si completa con funzioni di autodiagnostica

Le scelte di codifica sono, come spesso accade, frutto di compromesso tra leggibilità ed efficienza del codice. Il valore contenuto nella memoria **I** è denotato con *u*, mentre u_I è lo stato dell'integratore.
Si assume che la variabile AUTO valga *.true.* quando è selezionato il funzionamento in automatico, *.false.* in caso contrario, e che δu_m sia l'incremento (o la diminuzione), in un periodo di campionamento, dell'uscita quando l'operatore tiene premuto il tasto aumenta (o diminuisci). δu_m e i limiti di saturazione inferiore e superiore u_l e u_h dell'uscita devono essere assegnati nell'inizializzazione, così come u_D, u_I e e_{old}.

Pseudocodice di un regolatore PID

Soluzione Anti-wind-up in AM v3.1.5 sect. 11.4 (esempio: cruise control)

La soluzione *anti-windup* proposta in AM è diversa nel calcolo dell'errore in ingresso all'integratore in caso di saturazione. Non viene messo a zero ma ridotto in misura proporzionale all'eccesso di azione di controllo (rispetto al limite di saturazione)

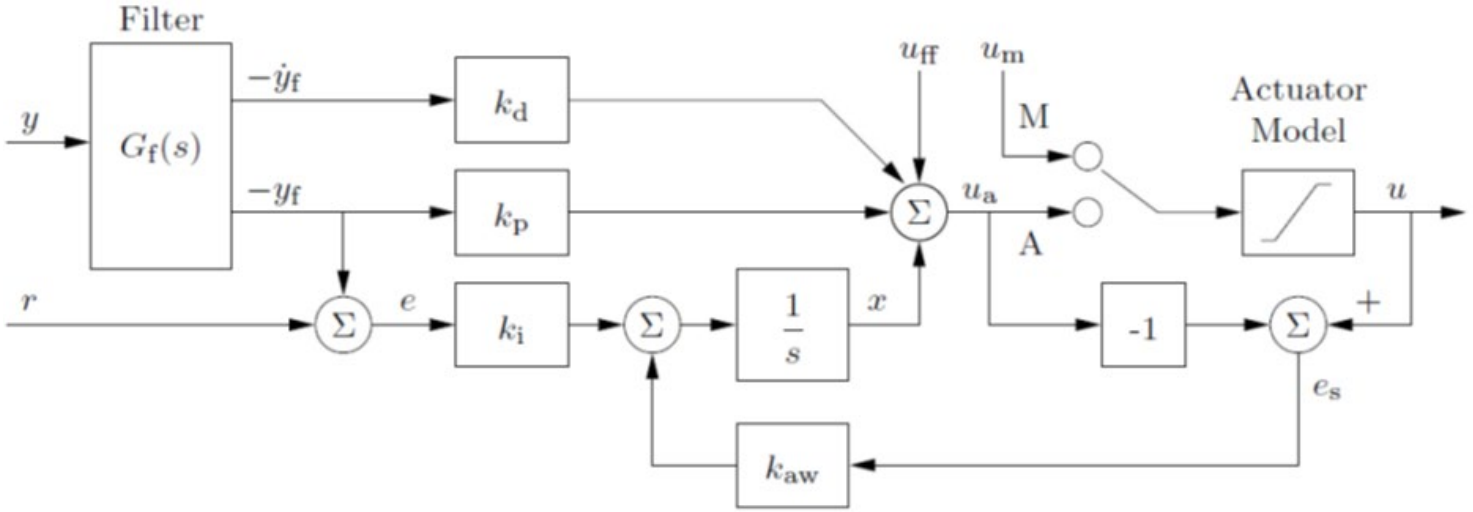


Figure 11.11: PID controller with filtering, anti-windup, and manual control. The controller has filtering of the measured signal, an input u_{ff} for feedforward signal, and another input w for direct control of the output. The switch is in position A for normal operation; if it is set to M the control variable is manipulated directly. The input to the integrator $(1/s)$ has a "reset" term that avoids integrator windup in addition to the normal P, I, and D terms. Notice that the reference r only enters in the integral term.