Roboitalia
www.roboitalia.com

Il PID facile

Le guide di Roboitalia

Hai deciso di costruire un robot?
Bene!
Iniziamo dalle brutte notizie…
Non e' facile!
Ora le buone notizie…
E' possibile, a patto di applicarsi e studiare.

di Marco Fabbri

Licenza di questo documento

Indice

Premessa

Questa guida non è certo da intendersi come un documento "professionale" e dettagliato come altri documenti che potete trovare in linea ma solo il tentativo di avvicinare, semplificando chi si avvicina per la prima volta ai regolatori PID.
Oltre a quanto si trova in rete e alle informazioni che arrivano dagli amici della rete, l'ispirazione principale per redigere questo documento viene da quanto scritto da Livio S. Orsini nel suo manuale propedeutico sulle tecniche di regolazione che trovate a questo indirizzo
http://www.plcforum.info/didattica/conreg/pagina1.htm

A cosa serve un PID

Ogni volta che un dispositivo deve mantenere costante un determinato valore, ad esempio una velocità, una temperatura, un livello, una rotta ecc. serve un regolatore. Ci serve qualcosa che corregga eventuali ed inevitabili errori rispetto al valore di consegna.
Se pensiamo al pilota automatico di una nave che deve lavorare per mantenere la rotta, ci è facile comprendere come venti, correnti e onde siano fonti di errore che il regolatore deve contrastare. Non è detto che sia solo così, ci può essere anche il caso in cui si debba seguire un valore che cambia come ad esempio seguire un percorso che cambia come succede ad un robot tipo linefollower o se vogliamo ad un missile che segue l'aereo da colpire.

Regolatore Proporzionale

Per il pilota automatico, detto in modo molto banale può sembrare sufficiente qualcosa che legge la bussola e giri il timone dal lato giusto e dell'angolo giusto per correggere; con un comportamento quindi Proporzionale (più la nave è sbandata più intensa sarà la correzione). Un comportamento quindi legato ad una relazione tipo: **correzione=K*errore** dove K è ovviamente da impostare in funzione dei parametri meccanici del timone e della velocità di risposta che vogliamo.
Beh, in effetti questo va già bene, è un regolatore di tipo proporzionale e come credo sia facile intuire la P di PID sta proprio per proporzionale.
Sfortunatamente un regolatore che preveda una correzione solo proporzionale non è sempre ciò che ci serve. Ritornando al nostro esempio sarebbe facile verificare quanto i nostri passeggeri siano in preda ad attacchi di mal di mare e ciò a causa del fatto che la nave continua a virare attorno alla rotta assegnata.
Senza inoltrarci nella teoria del perché proviamo con l'intuito. Intanto è evidente che il regolatore è attivo solo quando c'è un errore e quindi salvo casi eccezionali o brevi transitori avremo sempre un errore inoltre, se si eccede nella intensità della correzione il sistema diventa instabile e inizia a pendolare attorno al valore giusto con oscillazioni sempre più ampie. Giusto per non fare solo chiacchiere ritorniamo alla formula citata prima che mettiamo in forma più seria:

$$P = Kp * \varepsilon$$

P è il valore della correzione che vogliamo dare al nostro sistema.
ε è il valore dell'errore tra il valore della grandezza che stiamo controllando e il valore che desideriamo o valore di consegna
Kp è la costante che imposteremo per regolare la risposte del sistema.

Chi ha fatto un minimo di matematica riconoscera nella formula l'equazione tipica di una retta dove Kp rappresenta la pendenza della retta. In pratica maggiore sarà Kp maggiore sarà "l'intensità" della correzione e più veloce il ritorno verso il valore di consegna e maggiore sarà però il rischio di instabilità. In sostanza la

parte proporzionale è fondamentale per un regolatore in quanto lega la correzione in modo diretto all'errore, da sola però questa parte non riesce a dare stabilità al sistema e se vogliamo nemmeno a prevenire l'errore.

La I di PID, l'integrale o integratore

L'integrale altro non è se non la somma di tutti gli errori nel tempo e in pratica si ottiene semplicemente facendo:

$$\mathbf{I = I + Ki * \varepsilon}$$

Questo parametro "ricordando" gli errori precedenti produce l'effetto di smorzare i penzolamenti tipici del sistema proporzionale. Se prima il sistema ci forniva in uscita una variabile che era: OUT=Kp*ε e che avevamo chiamato P ora abbiamo:

$$\mathbf{OUT = P + I}$$

Con I calcolata come nella formula precedente.
L'effetto di rendere più stabile il sistema è dato da una sorta di ritardo con la quale la variabile out viene integrata. Questo ritardo dipende dalla Ki, costante che dovremo scegliere e bilanciare affinché l'accoppiata P e I sia efficace.
Questo ritardo ha però un prezzo, rende l'insieme meno reattivo e veloce. Per molte applicazioni comunque il tutto potrebbe essere già sufficiente così. Siamo infatti già nella situazione di realizzare un semplice inseguitore di linea o all'inglese linefollower e sarebbe un ottimo metodo per iniziare a mettere in pratica quanto è stato visto fino a questo punto, realizzando ad esempio un percorso rettilineo di alcuni metri e scoprire come attivando solo il regolatore proporzionale ci troveremmo nelle situazioni descritte e aggiungendo l'integratore si possa ottenere un andamento più fluido.

La D di PID, la derivata

La derivata è una funzione matematica che credo conosciate, diversamente vi rimando sui libri di scuola anche se per come la analizzeremo qui non è strettamente necessario conoscerla in modo approfondito.

Viene usata la derivata per ottenere una risposta più rapida e se vogliamo una previsione di quanto sta accadendo. La derivata infatti lavora sulla tendenza dell'errore e non sull'errore assoluto. Questo significa che ad esempio sapendo che l'errore sta crescendo possiamo dare più energia alla correzione di quanto previsto dal solo parametro proporzionale mentre possiamo ridurre la correzione se l'errore sta diminuendo segno che se siamo in fase di arrivo. La derivata per così dire accelera o decelera l'intervento del regolatore in modo dinamico seguendo la tendenza dell'errore e permettendoci di prevedere che, senza altri interventi, alla prossima lettura l'errore sarà minore o maggiore in base appunto alla tendenza.
A livello di formule la derivata la calcoliamo così:

$$D = Kd \bullet \frac{\delta\varepsilon}{\delta t}$$

Qui abbiamo che D è il valore della correzione, Kd è la solita costante che andremo a settare noi in funzione del peso che vogliamo dare alla parte derivativa del nostro regolatore.

δε è la differenza tra i due errori ossia, è chiaro che il nostro sistema dovrà controllare la nostra grandezza ad intervalli regolari e confrontarla con il valore che dobbiamo mantenere per cui ad ogni controllo avremo un errore, se fissiamo in t il momento dell'ultimo controllo avremo che il controllo precedente sarà stato effettuato in t-1, avremo quindi due letture di errore, ε_t ε_{t-1} la differenza tra questi due valori ci dice se

l'errore sta aumentando o diminuendo, avremo quindi un risultato positivo (l'errore aumenta) se l'ultimo errore è maggiore del precedente, negativo al contrario, noterete quindi che D assume un valore positivo o negativo in funzione della tendenza e non del valore assoluto dell'errore.

δt se ripetiamo la misura ad intervalli regolari viene fissato ad 1 non è quindi influente nel nostro calcolo.

Finalmente il PID

Abbiamo finalmente i nostri tre elementi che costituiscono il nostro PID, non ci resta che combinarli.
Beh, dal punto di vista matematico è molto semplice:

OUT = P + I + D

In effetti è tutto quello che serve, i calcoli dei tre parametri li abbiamo fatti prima, li riassumiamo aggiungendo che ovviamente vanno fatti ad ogni ciclo di misura della nostra grandezza da regolare.

$$P=Kp*\varepsilon$$

$$I=I+Ki*\varepsilon$$

$$D = Kd \bullet \frac{\delta\varepsilon}{\delta t}$$

Il difficile in tutto questo è regolare le tre K. Queste determinano infatti il peso dei tre parametri sul totale della nostra regolazione, per fare una cosa scientifica dovremmo avere i parametri meccanici, di risposta e forse anche le previsioni del tempo, ovviamente a livello hobbistico dove spesso si lavora con materiali di recupero non è cosa fattibile, non ci resta che il metodo empirico e cioè la sperimentazione e le prove.
Seppur empirico è un metodo e va fatto non per tentativi casuali (che può funzionare ma si tratta di altro… metodo) ma con cognizione di causa e criterio.
IL suggerimento è quello di impostare un parametro per volta ovvero disabilitare le regolazioni I e D e cercare di ottenere il massimo dell'efficienza dalla sola regolazione proporzionale, aggiungere poi la regolazione I e poi la D cercando sempre di fare piccole variazioni.
Siccome iniziamo dal parametro proporzionale potrebbe risultare necessario tenere questo un po' più basso rispetto alla prova iniziale e questo per il semplice motivo che comunque andiamo ad aggiungere gli altri parametri e ciò potrebbe portare il tutto oltre le possibilità del regolatore o di risposta della macchina.

A questo punto un linefollower sarebbe veramente utile per vedere se ciò che abbiamo fatto ha senso, in particolare ci servirebbe una scheda con un microcontrollore tipo PIC dove scrivere due righe di programma e infilare le formule che abbiamo visto, ah sì, ci serve anche il programma.

Il codice del PID

Riporto pari pari il codice in pseudo C presente nel manuale di Livio S. Orsini visto che è molto ben fatto e comprensibile.

```
int PID (int val_cons)
{
/*La funzione PID è richiamata dall'interrupt del timer di sistema; la chiamata avviene ogni 10 msec., pertanto z-1 = 10 msec. La funzione PID riceve un valore intero che rappresenta il valore di consegna "val_cons ", la funzione restituisce un valore intero "DA_conv" che rappresenta il valore di riferimento per l'attuatore.
  Le variabili:
float Upper_P_limit, Upper_I_limit, Upper_D_limit, Upper_Total_limit
float Lower_P_limit, Lower _I_limit, Lower _D_limit, Lower _Total_limit
float Kp, Ki, Kd
sono globali; sono introdotte e modificate tramite interfaccia HMI
*/
static int AD_Conv = 0;        /*Lettura convertitore A/D: acquisisce la variabile di ingresso*/
static int DA_Conv = 0;        /*Scrittura convertitore D/A: scrive la variabile di uscita*/
static int error = 0;          /*differenza tra valore di consegna e valore reale */
static int old_error = 0;      /*differenza tra valore di consegna e valore reale @ z-1 */
float P=0;                     /* componente proporzionale */
float I=0;                     /* componente integrale */
float D=0;                     /* componente differenziale */
float i_inst = 0;              /* parte istantanea del processo di integrazione*/
float Out = 0;                 /* Totale regolazione */
error = val_cons - AD_Conv;
P = error * Kp;
if (P > Upper_P_Limit) P = Upper_P_Limit;
if (P < Lower_P_Limit) P = Lower_P_Limit;
if Ki > 0 {
i_inst = error * Ki;
I = I + i_inst;
if (I > Upper_I_Limit) I = Upper_I_Limit;
if (I < Lower_I_Limit) I = Lower_I_Limit; }
else
I = 0;
if Kd > 0 {
D = Kd * (error – old_error);
old_error = error;
if (D > Upper_D_Limit) D = Upper_D_Limit;
if (D < Lower_D_Limit) D = Lower_D_Limit;}
else
D = 0;
Out = P + I + D;
if ( Out > Upper_Total_limit) Out = Upper_Total_limit;
if (Out < Lower_Total_limit) Out = Lower_Total_limit;
DA_Conv = Out;
Return (DA_Conv);
}
La routine di gestione dell'interrupt, legata alla scadenza del System Timer, che richiama PID sarà simile a:
void RT_G( )
  {
static int Out_DA = 0;     /*uscita D/A per riferimento)
static int valore_di_consegna = 0;

valore_di_consegna = act_val;
Out_DA  = act_ref_val;
act_ref_val = PID(valore_di_consegna);
}
```

Come potete vedere vengono fissati dei valori massimi e minimi entro i quali ogni parametro può muoversi, viene anche inserita una istruzione if che controlla se la K relativa è impostata a zero, in questo caso il parametro non viene calcolato e impostato a zero, in questo modo è possibile disabilitare le correzioni singolarmente P, I e D in modo da fare la taratura.
IL codice è facilmente convertibile anche in altri linguaggi se escludiamo l'assembler, e ci si può risparmiare anche la gestione dell'interrupt se questo codice diventa un loop infinito che si ripete e all'interno del quale mandiamo le variazioni all'hardware che regola ad esempio i motori.

In questo modo commettiamo un errore che è quello di non rendere esattamente preciso l'intervallo di tempo tra una lettura e la successiva, regolatori particolarmente sofisticati e precisi potrebbero risentire molto di questa incoerenza, nel nostro caso (linefollower) credo sia addirittura necessario mettere un ritardo tra una lettura e la successiva in modo che il robot abbia il tempo di eseguire la correzione che gli stiamo dando e probabilmente qualche micro secondo di differenza nella lettura non sposta di molto le performance.

Conclusioni

Ricordo ancora a tutti i downloader di questo documento che queste righe non sono destinate agli esperti e non hanno la pretesa di essere esaustive, sono solo un piccolo riassunto nemmeno troppo preciso che ha però l'intento di invogliare chi fino ad oggi ha pilotato un motore in ON/OFF a provare qualcosa di un po' più evoluto attraverso il quale fare nuove esperienze.

Volendo approfondire l'argomento vi segnalo alcuni link che trattano dell'argomento:

http://www.plcforum.info/didattica/conreg/conreg.htm
http://xoomer.alice.it/guiott/
http://en.wikipedia.org/wiki/PID_controller

Se volete portare il vostro contributo:
http://www.roboitalia.it