NAME

pid − proportional/integral/derivative controller

SYNOPSIS

loadrt pid [num_chan=*num* **| names=***name1***[,***name2...***]] [debug=***dbg*]

DESCRIPTION

pid は、位置または速度のフィードバックループを制御するために使用される、古典的な比例/積分/微分コントローラーです。 サーボモーターおよびその他の閉ループアプリケーション用。 pid は最大 16 個のコントローラーをサポートします。 実際にロードされる数は、モジュールがロードされるときに num_chan 引数によって設定されます。 または、names =と一意の名前を区切って指定します カンマで。

num_chan =および names =指定子は相互に排他的です。 num_chan =も names =も指定されていない場合、デフォルト値は 3 です。 debug が 1 に設定されている場合（デフォルトは 0）、いくつかの追加の HAL パラメーターがエクスポートされます。これはチューニングに役立つ場合がありますが、それ以外の場合は不要です。

NAMING

ピン、パラメーター、および関数の名前には、次の接頭辞が付いています。

pid.N. for N=0,1,...,num−1 when using **num_chan=num**

nameN. for nameN=name1,name2,... when using **names=name1,name2,...**

pid.N. 以下の説明にフォーマットを示します。

FUNCTIONS

pid.*N*.do−pid−calcs (uses floating-point) Does the PID calculations for control loop *N*.

PINS

pid.*N*.command float in

制御ループに必要な（コマンドされた）値。

pid.*N*.Pgain float in

比例ゲイン。 結果は、エラーに Pgain を掛けたものである出力への寄与になります。

pid.*N*.Igain float in

積分ゲイン。 結果は、エラーに Igain を掛けた積分である出力への寄与になります。 たとえば、10 秒間続いた 0.02 のエラーは 0.2 の積分エラー（errorI）になり、Igain が 20 の場合、積分項は出力に 4.0 を追加します。

pid.*N*.Dgain float in

微分ゲイン。 結果は、エラーの変化率（導関数）に Dgain を掛けたものである出力への寄与になります。 たとえば、0.2 秒で 0.02 から 0.03 に変化したエラーは、0.05 のエラー導関数（errorD）になり、Dgain が 5 の場合、導関数項は出力に 0.25 を追加します。

pid.*N*.feedback float in

エンコーダーなどのセンサーからの実際の（フィードバック）値。

pid.*N*.output float out

モーターなどのアクチュエーターに送られる PID ループの出力。

pid.*N*.command−deriv float in

制御ループの目的の（コマンドされた）値の導関数。 信号が接続されていない場合、導関数は数値的に推定されます。

pid.*N*.feedback−deriv float in

制御ループの実際の（フィードバック）値の導関数。 信号が接続されていない場合、導関数は数値的に推定されます。 フィードバックが量子化された位置ソース（エンコーダーフィードバック位置など）からのものである場合、エンコーダー（9）またはhostmot2（9）の速度出力など、ここでより適切な速度推定を使用することにより、D 項の動作を改善できます。

pid.*N*.error−previous−target bit in

モーションコントローラが期待するように、エラー計算には以前の呼び出しのターゲットと現在の位置を使用します。 これにより、速度に依存する追従誤差を排除することにより、トルクモード位置ループおよび大きなI ゲインを必要とするループの調整が容易になる場合があります。

pid.*N*.error float out

コマンドとフィードバックの差。

pid.*N*.enable bit in

true の場合、PID 計算を有効にします。 false の場合、出力はゼロになり、すべての内部積分器などがリセットされます。

pid.*N*.index−enable bit in

index-enable の立ち下がりエッジで、pid は内部コマンド微分推定を更新しません。 エンコーダインデックスパルスを使用するシステムでは、このピンをインデックスイネーブル信号に接続する必要があります。 これが行われず、FF1 がゼロ以外の場合、入力コマンドのステップ変更により、PID 出力に 1 サイクルのスパイクが発生します。 -deriv 入力の 1 つだけを使用するシステムでは、これは D 項にも影響します。

pid.*N*.bias float in

バイアスは、出力に追加される一定量です。 ほとんどの場合、ゼロのままにしておく必要があります。 ただし、サーボアンプのオフセットを補正したり、垂直方向に移動するオブジェクトの重量のバランスをとったりすると便利な場合があります。 出力の他のすべてのコンポーネントと同様に、PID ループが無効になると、バイアスはオフになります。 PID ループが無効になっている場合でもゼロ以外の出力が必要な場合は、外部 HALsum2 ブロックを追加する必要があります。

pid.*N*.FF0 float in

ゼロ次フィードフォワード項。 FF0 にコマンド値を掛けたものが出力に寄与します。 位置ループの場合、通常はゼロのままにしておく必要があります。 速度ループの場合、FF0 は摩擦またはモーターカウンター EMF を補正でき、適切に使用すればより良いチューニングが可能になる場合があります。

pid.*N*.FF1 float in

一次フィードフォワード項。 FF1 にコマンド値の導関数を掛けた出力への寄与を生成します。 位置ループの場合、寄与は速度に比例し、摩擦またはモーター CEMF を補正するために使用できます。 速度ループの場合、加速度に比例し、慣性を補正できます。 どちらの場合も、適切に使用すると、チューニングが向上する可能性があります。

pid.*N*.FF2 float in

2 次フィードフォワード項。 FF2 にコマンド値の 2 次導関数を掛けた出力への寄与を生成します。 位置ループの場合、寄与は加速度に比例し、慣性を補正するために使用できます。 速度ループの場合、寄与はジャークに比例し、通常はゼロのままにする必要があります。

pid.*N*.FF3 float in

3 次フィードフォワード項。 FF3 にコマンド値の 3 次導関数を掛けたものが出力に寄与します。 位置ループの場合、寄与はジャークに比例し、加速中の残留誤差を補正するために使用できます。 速度ループの場合、寄与は snap（jounce）に比例し、通常はゼロのままにする必要があります。

pid.*N*.deadband float in

「許容可能な」エラーの範囲を定義します。 エラーの絶対値が不感帯よりも小さい場合、エラーがゼロであるかのように扱われます。 本質的に量子化されたエンコーダーなどのフィードバックデバイスを使用する場合、コマンドが 2 つの隣接するエンコーダー値の間にある場合に制御ループが前後にハンチングするのを防ぐために、不感帯は半分のカウントよりわずかに大きく設定する必要があります。 エラーの絶対値が不感帯よりも大きい場合、不感帯のエッジでの伝達関数のステップを防ぐために、ループ計算を実行する前に不感帯の値がエラーから差し引かれます。 （バグを参照してください。）

pid.*N*.maxoutput float in

出力制限。 maxoutput がゼロでない限り、出力の絶対値が maxoutput を超えることはできません。 出力が制限されている場合、ワインドアップとオーバーシュートを防ぐために、エラー積分器は積分する代わりに保持します。

pid.*N*.maxerror float in

P、I、および D に使用される内部エラー変数の制限。エラーが大きい場合（コマンドがステップ変更を行う場合など）に、高い Pgain 値が大きな出力を生成しないようにするために使用できます。 通常は必要ありませんが、非線形システムを調整するときに役立ちます。

pid.*N*.maxerrorD float in

エラー導関数の制限。 Dgain 項で使用されるエラーの変化率は、値がゼロでない限り、この値に制限されます。 Dgain の影響を制限し、コマンドやフィードバックのステップによる大きな出力スパイクを防ぐために使用できます。 通常は必要ありません。

pid.*N*.maxerrorI float in

エラー積分器の制限。 Igain 項で使用されるエラー積分器は、ゼロでない限り、この値に制限されます。 インテグレータのワインドアップと、エラーの発生中または発生後に発生するオーバーシュートを防ぐために使用できます。 通常は必要ありません。

pid.*N*.maxcmdD float in

コマンド派生物の制限。 FF1 で使用されるコマンド導関数は、値がゼロでない限り、この値に制限されます。 コマンドにステップ変更がある場合に、FF1 が大きな出力スパイクを生成するのを防ぐために使用できます。 通常は必要ありません。

pid.*N*.maxcmdDD float in

コマンドの二次導関数の制限。 FF2 で使用されるコマンドの 2 次導関数は、値がゼロでない限り、この値に制限されます。 コマンドにステップ変更がある場合に、FF2 が大きな出力スパイクを生成するのを防ぐために使用できます。 通常は必要ありません。

pid.*N*.maxcmdDDD float in

コマンドの三階導関数の制限。 FF3 で使用されるコマンドの 3 次導関数は、値がゼロでない限り、この値に制限されます。 コマンドにステップ変更がある場合に、FF3 が大きな出力スパイクを生成するのを防ぐために使用できます。 通常は必要ありません。

pid.*N*.saturated bit out

true の場合、現在の PID 出力は飽和しています。 あれは、
output = ± **maxoutput**.

pid.*N*.saturated−s float out

pid.*N*.saturated−count s32 out

true の場合、PID の出力は、この数秒間（saturated-s）または期間（saturated-count）継続的に飽和していました。

PARAMETERS

pid.*N*.errorI float ro (only if debug=1)

エラーの積分。 これは、出力の積分項を生成するために Igain を掛けた値です。

pid.*N*.errorD float ro (only if debug=1)

エラーの派生物。 これは、出力の微分項を生成するために Dgain を掛けた値です。

pid.*N*.commandD float ro (only if debug=1)

コマンドの派生物。 これは、出力の 1 次フィードフォワード項を生成するために FF1 を掛けた値です。

pid.*N*.commandDD float ro (only if debug=1)

コマンドの 2 次導関数。 これは、出力の 2 次フィードフォワード項を生成するために FF2 を掛けた値です。

pid.*N*.commandDDD float ro (only if debug=1)

コマンドの 3 次導関数。 これは、出力の 3 次フィードフォワード項を生成するために FF3 を掛けた値です。

BUGS

一部の人々は、デッドバンドは、エラーがデッドバンド内にある場合はゼロとして扱われ、デッドバンド外にある場合は変更されないように実装する必要があると主張します。 これは、不感帯のサイズに等しい伝達関数のステップを引き起こすため、実行されませんでした。 その振る舞いを好む人は、振る舞いを変更するパラメーターを追加するか、独自のバージョンの pid を作成することを歓迎します ただし、デフォルトの動作は変更しないでください。

負のゲインは、望ましくない動作につながる可能性があります。 状況によっては、負の FF ゲインが理にかなっている可能性がありますが、一般に、すべてのゲインは正である必要があります。 一部の出力が間違った方向にある場合、それを修正するためにゲインを否定するのは間違いです。 代わりに、他の場所でスケーリングを正しく設定してください。