PID Algorithm Basics

1. Introduction

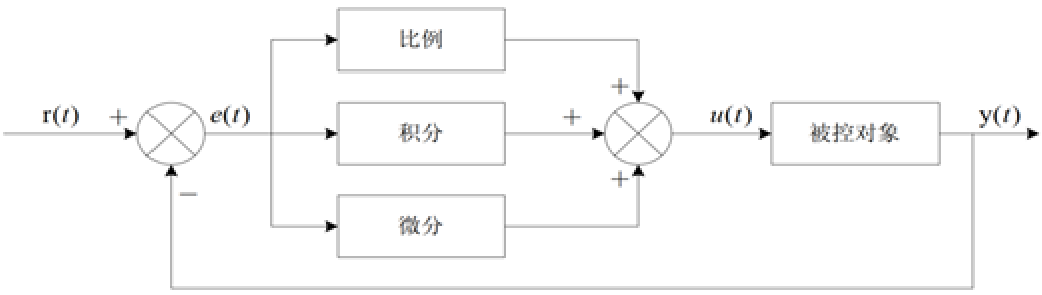
PID performs proportional, integral, and derivative operations on the input deviation, and the superposition result of the operations controls the actuator. The formula is as follows:

$$u(t) = K_p[e(t) + \frac{1}{T_i}\int_0^t e(t)dt + Td\frac{de(t)}{dt}]$$

It consists of three parts:

- P is proportional, which is the input deviation multiplied by a coefficient;
- I is integral, which performs integral operation on the input deviation;
- D is derivative, which performs derivative operation on the input deviation.

As shown in the figure below is a basic PID controller:



(1) Proportional Part

```
    The mathematical expression of the proportional part is: Kp*e(t)

    In analog PID controllers, the role of the proportional link is to react
instantly to deviations. Once a deviation occurs, the controller immediately
produces a control action that changes the control quantity in the direction of
reducing the deviation. The strength of the control action depends on the
proportional coefficient. The larger the proportional coefficient, the stronger
the control action, the faster the transition process, and the smaller the static
deviation of the control process; however, the larger it is, the more likely it
is to produce oscillations, destroying system stability. Therefore, the selection
of the proportional coefficient must be appropriate to achieve the effects of
short transition time, small static deviation, and stability.

    Advantages: Adjust the system's open-loop proportional coefficient to improve
the system's steady-state accuracy, reduce system inertia, and speed up response
time.

    Disadvantages: Using only a P controller, an excessively large open-loop
proportional coefficient not only increases the system's overshoot but also
reduces the system's stability margin, possibly causing instability.
```

(2) Integral Part

The mathematical expression of the integral part is:

$$\frac{K_p}{T_i}\int_0^t e(t)dt$$

```
    The larger the integral constant, the weaker the cumulative effect of
integration. At this time, the system will not oscillate during transition;
however, increasing the integral constant will slow down the process of
eliminating static errors, and the time required to eliminate deviations is
longer, but it can reduce overshoot and improve system stability. When Ti is
smaller, the integral effect is stronger. At this time, the system may oscillate
during the transition time, but the time required to eliminate deviations is
shorter. Therefore, Ti must be determined according to the specific requirements
of actual control.
```

Advantages: Eliminate steady-state errors.

Disadvantages: The addition of an integral controller affects system stability and reduces the system's stability margin.

(3) Derivative Part

The mathematical expression of the derivative part is:

$$K_p * Td\frac{de(t)}{dt}$$

```
    The role of the derivative link is to prevent changes in deviation. It
controls based on the trend of deviation changes (rate of change). The faster the
deviation changes, the larger the output of the derivative controller, and it can
make corrections before the deviation value becomes larger. The introduction of
the derivative effect will help reduce overshoot, overcome oscillations, and make
the system tend toward stability, which is particularly beneficial for high-order
systems as it speeds up the system's tracking speed. However, the derivative
effect is very sensitive to input signal noise. Generally, derivative control is
not used for systems with high noise, or the input signal is filtered before the
derivative effect takes effect. The effect of the derivative part is determined
by the derivative time constant Td. The larger Td is, the stronger its effect of
suppressing deviation changes; the smaller Td is, the weaker its effect of
resisting deviation changes. The derivative part obviously has a great effect on
system stability. Properly selecting the derivative constant Td can make the
derivative effect optimal.

    Advantages: Makes the system's response speed faster, reduces overshoot,
alleviates oscillations, and has a "prediction" effect on dynamic processes.
```

2. Selection of PID Algorithms

```
    Digital PID control algorithms can be divided into positional PID and
incremental PID control algorithms. Before deciding which PID algorithm to use,
we should first understand its principle:
```

- Positional PID Algorithm:

$$u(k) = K_p e(k) + K_I \sum_{i=o} e(i) + K_D[e(k) - e(k-1)]$$

```
    e(k): user-set value (target value) - current state value of the controlled
object

    Proportional P: e(k)

    Integral I: Σe(i) cumulative error

    Derivative D: e(k) - e(k-1) current error - previous error

    That is, positional PID performs PID control based on the deviation between
the current actual position of the system and the expected position you want to
achieve

    Because of the error integral Σe(i), which keeps accumulating, the current
output u(k) is related to all past states, using the cumulative value of errors;
(error e will have error accumulation), the output u(k) corresponds to the actual
position of the actuator. Once the control output is wrong (the current state
value of the controlled object has problems), large changes in u(k) will cause
large changes in the system. When the integral term of positional PID reaches
saturation, the error will still continue to accumulate under the integral
effect. Once the error starts to change in the opposite direction, the system
needs a certain time to exit the saturation zone. Therefore, when u(k) reaches
maximum and minimum, the integral effect should be stopped, and there should be
integral limiting and output limiting. Therefore, when using positional PID, we
generally directly use PD control, while positional PID is suitable for objects
whose actuators do not have integral components, such as servos and balance car
upright and temperature control system control

    Advantages: Positional PID is a non-recursive algorithm that can directly
control actuators (such as balance cars). The value of u(k) corresponds one-to-
one with the actual position of the actuator (such as the current angle of the
car), so it can be well applied in objects whose actuators do not have integral
components

    Disadvantages: Each output is related to past states, and e(k) needs to be
accumulated during calculation, resulting in a large computational workload.
```

- Incremental PID:

$$\Delta u(k) = u(k) - u(k-1) = K_P[e(k) - e(k-1)] + K_i e(k) + K_D[e(k) - 2e(k-1) + e(k-2)]$$

```
    Proportional P: e(k)-e(k-1) current error - previous error

    Integral I: e(k) error

    Derivative D: e(k)-2e(k-1)+e(k-2) current error - 2*previous error + error
before previous

    Incremental PID can be clearly seen from the formula. Once KP, TI, and TD are
determined, as long as the deviation of the last three measurement values is
used, the control increment can be calculated from the formula. The resulting
control amount $Δu(k)$ corresponds to the increment of recent position errors,
not the deviation corresponding to the actual position. There is no error
accumulation, meaning that incremental PID does not need accumulation. The
determination of the control increment $Δu(k)$ is only related to the most recent
3 sampling values, making it easy to obtain better control effects through
weighting, and when system problems occur, incremental will not seriously affect
system work

    Summary: Incremental PID takes the increment of positional PID. At this time,
the controller outputs the difference between the position values calculated at
two adjacent sampling moments. The result is the increment, that is, on the basis
of the previous control amount, the control amount needs to be increased
(negative value means reduced).
```

Advantages: ① Small impact when malfunctioning, and if necessary, logical judgment methods can be used to remove erroneous data.

```
      ② Small impact during manual/automatic switching, facilitating bumpless
switching. When computer fails, it can still maintain the original value.

      ③ No accumulation required in the formula. The determination of control
increment Δu(k) is only related to the most recent 3 sampling values.
```

Disadvantages: ① Large integral truncation effect, with steady-state errors;

```
      ② Large impact of overflow. Some controlled objects are not suitable for
incremental;
```

- Differences Between Incremental and Positional

  ```
  (1) Incremental algorithms do not need accumulation. The determination of
  control increment is only related to the most recent few deviation sampling
  values, and the impact of calculation errors on control quantity calculation
  is small. Positional algorithms need to use the accumulated value of past
  deviations, which easily produces larger accumulation errors.

  (2) Incremental algorithms produce the increment of control quantity. For
  example, in valve control, only the change part of valve opening is output.
  The impact of malfunctioning is small, and if necessary, the current output
  can be limited or prohibited through logical judgment without seriously
  affecting system work. Positional output directly corresponds to the
  object's output, thus having a larger impact on the system.
  ```

(3) Incremental PID controls the increment of control quantity and has no integral effect, so this method is suitable for objects whose actuators have integral components, such as stepper motors, while positional PID is suitable for objects whose actuators do not have integral components, such as electro-hydraulic servo valves.

```
    (4) When performing PID control, positional PID needs integral limiting and output limiting, while incremental PID only needs output limiting. Positional PID and incremental PID are just two implementation forms of digital PID control algorithms, essentially identical. The main difference is the storage method of integral terms. Positional PID stores integral terms separately, while incremental PID stores integral terms as part of the output. Various players online also have their unique insights and views on the use of positional and incremental forms. Specifically, we still need to see which algorithm is suitable for our specific application scenario.
```

3. Tuning of PID Parameters

```
    There are many methods for selecting PID controller parameters, such as trial and error method, critical proportion method, expanded critical proportion method, etc. However, for PID control, parameter selection is always a very cumbersome task, requiring constant adjustment to obtain satisfactory control effects. Based on experience, the general steps for determining PID parameters are as follows:
```

- Determine Proportional Coefficient Kp

When determining the proportional coefficient Kp, first remove the integral and derivative terms of PID, can set Ti=0, Td=0, making it

```
    pure proportional control. Input is set to 60% to 70% of the system's allowed maximum output. The proportional coefficient Kp gradually increases from 0 until the system oscillates; then conversely, gradually decrease the proportional coefficient Kp from this point until the system oscillation disappears. Record the proportional coefficient Kp at this time, and set the PID's proportional coefficient Kp to 60% to 70% of the current value.
```

- Determine Integral Time Constant Ti

  ```
  After the proportional coefficient Kp is determined, set a larger integral time constant Ti, then gradually decrease Ti until the system oscillates, then conversely, gradually increase Ti until the system oscillation disappears. Record the Ti at this time, and set the PID's integral time constant Ti to 150% to 180% of the current value.
  ```

- Determine Derivative Time Constant Td

The derivative time constant Td generally does not need to be set, can be 0, at which time PID control converts to PI control. If setting is needed, it is the same as the method for determining Kp, taking 30% of its value when not oscillating.

- System No-load, Load Joint Debugging

  ```
  Fine-tune PID parameters until performance requirements are met.
  ```

```
Of course, this is just my personal debugging method and may not necessarily
be suitable for everyone and every environment. It is only provided for
everyone's reference; however, there are also classic trial-and-error
formulas for debugging PID circulating online. I have also posted them for
everyone's reference:

Parameter tuning to find the best, check in order from small to large.

 First proportional then integral, finally add derivative.

 Curve oscillates frequently, proportional dial should be enlarged.

 Curve floats around large bends, proportional dial should be turned
smaller.

 Curve deviates and recovers slowly, integral time should be decreased.

 Curve fluctuation period is long, integral time should be increased.

 Curve oscillation frequency is fast, first reduce derivative.

 Large dynamic error and slow fluctuation, derivative time should be
increased.

 Ideal curve has two waves, front high back low four to one.

 Look, adjust, and analyze more, regulation quality won't be low.
```

PID is proportional (P), integral (I), derivative (D) control algorithm, and it's not necessary to have all three algorithms simultaneously. It can also be PD, PI, or even only P algorithm control. My previous most basic idea for closed-loop control was only P control - feedback the current result, subtract from the target, if positive, decelerate, if negative, accelerate. Of course, this is just the simplest closed-loop control algorithm. Going back to the summary of positional and incremental from the previous section, we need to refer to our current control environment specifically, because each control system is different, and the parameters that can make our system achieve the most stable effect are naturally OK.