Practical Considerations for PID Control

1.0 Learning Outcomes……1
2.0 Introduction……1
3.0 Practical Considerations…….....…..2
4.0 Summary…….……...7
Recommended Citation……7

1.0 Learning Outcomes

After studying this ACE-Lab learning guide, you should be able to demonstrate the Learning Outcomes listed in Table 1. On completion of the ACE-Lab learning guide, revisit Table 1 and confirm whether each Learning Outcome has been achieved.

Table 1: *Learning Outcomes from the Practical Considerations for PID Control learning guide*

ID	Learning Outcome	Achieved
1.	Identify the main practical factors that can influence the performance of a discrete-time PID controller when implemented on a physical system.	☐
2.	Explain the effects of sampling, measurement noise and filtering, actuator saturation, integral windup, derivative kick, numerical resolution and controller start-up on PID control performance.	☐
3.	Describe practical methods used to reduce or manage these effects, including filtering, anti-windup, appropriate signal limits and suitable controller initialisation.	☐

2.0 Introduction

In practical implementations, the three-term PID controller combines proportional, integral and derivative actions, with each term evaluated at discrete sampling instants and implemented using difference equations suitable for execution on a microcontroller, as illustrated in Figure 1. While this provides the basic computational structure of a discrete-time PID controller, its performance on a physical system is also influenced by a range of practical implementation factors.

A key practical consideration is selecting an appropriate sampling interval, T_s, and tuning the PID gains to achieve the required system response. Other important considerations include measurement filtering, particularly where derivative action may amplify high-frequency noise; actuator saturation and anti-windup, to prevent excessive integral accumulation; signal scaling and limits, such as voltage, PWM or current constraints; and the numerical

implementation of the controller on the chosen microcontroller. Analytical or simulation-based methods may be used as part of the tuning process, but the final controller should be evaluated and refined using the measured response of the physical system. Together, these considerations help ensure that the discrete-time PID controller performs reliably when applied to a physical system.

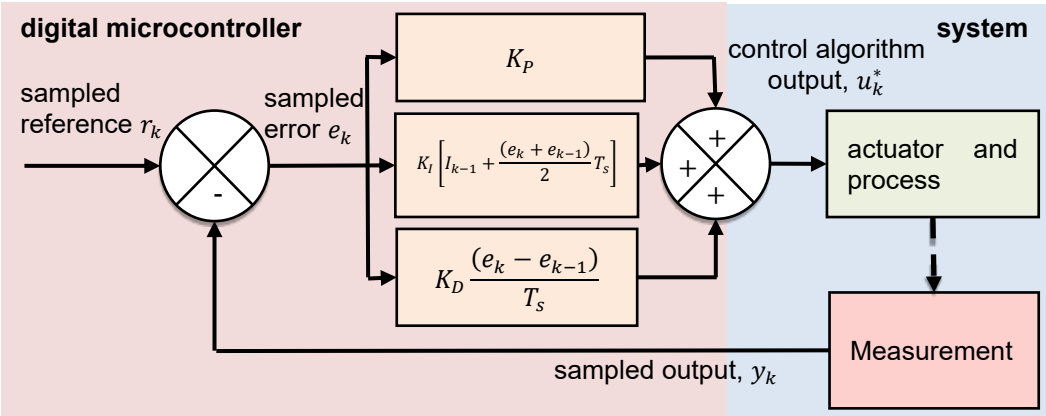


Figure 1: Discrete-Time *Proportional, Integral and Derivative (PID) Control*

3.0 Practical Considerations

Implementing a discrete-time PID controller on a physical system requires more than selecting suitable controller gains. Factors such as the sampling period, measurement quality, actuator limitations, numerical implementation and controller protection mechanisms can all influence the resulting closed-loop performance.

Table 2 summarises the principal practical considerations associated with implementing discrete-time PID control and explains why each is important. The following sections introduce each consideration and describe its influence on the behaviour and implementation of the controller.

Table 2: *Practical Considerations and Why They Matter*

Consideration	Why it matters
Sampling period, T_s	Determines how frequently measurements and control actions are updated.
Gain tuning	Sets the balance between response speed, error removal, damping and control effort.
Measurement filtering	Reduces the influence of sensor noise, especially on derivative action.
Actuator saturation	Represents physical limits such as voltage, current and PWM range.
Anti-windup	Prevents the integral term from accumulating excessively during saturation.
Derivative kick	Can produce a large derivative action following a sudden change in the reference.
Initial conditions / bumpless transfer	Reduces abrupt control changes when starting or switching control modes.
Quantisation and numerical precision	Accounts for finite sensor, PWM and processor resolution.
Execution time	Ensures all calculations complete within each sample period.
Verification	Confirms the measured response and control signal satisfy the requirements.

Sampling Period

The PID controller is evaluated at discrete sampling instants separated by the sampling period, T_s. The choice of T_s determines how frequently the controller reads the measured output, calculates the error and updates the control signal. A sampling period that is too large may fail to capture important system dynamics and can reduce control performance or lead to instability. An unnecessarily small sampling period increases computational demand and, particularly when derivative action is used, can increase sensitivity to measurement noise and quantisation. The sampling period should therefore be sufficiently small to represent the important dynamics of the physical system while allowing the complete control algorithm to execute reliably before the next sample.

Controller Gain Tuning

The PID gains, K_P, K_I and K_D, influence the behaviour of the controller and must be selected for the physical system being controlled. Initial values may be obtained through manual tuning, mathematical modelling, simulation or optimisation, but the final gains should be evaluated using measured results from the physical system.

Changing one gain can influence several features of the response. Controller tuning should therefore consider the complete closed-loop response, including rise time, overshoot, steady-state error, disturbance rejection and the required actuator effort.

Measurement Noise and Filtering

Physical measurements often contain noise. This is especially important when derivative action is used because differentiation responds strongly to rapid changes in the measured signal. Small high-frequency variations can therefore produce relatively large derivative contributions and a rapidly changing actuator command.

A low-pass filter may be applied to the measured signal, or filtering can be incorporated directly into the derivative action. Filtering should be selected carefully because excessive filtering introduces additional delay and can slow the apparent measured response.

Actuator Saturation

Physical actuators cannot produce unlimited control effort. For example, a motor may be constrained by the available supply voltage, current capability or PWM range. The PID controller may calculate an unconstrained control signal, u_k^*, that exceeds these physical limits. The command applied to the actuator, u_k, must therefore be limited to the range that can be produced by the hardware. This can be represented as:

$$u_k = sat(u_k^*) \tag{1}$$

where u_k^* is the control signal calculated by the PID algorithm and u_k is the control signal applied to the actuator after the saturation limits have been imposed. The upper and lower saturation limits should therefore be selected to represent the physical operating range of the actuator. When the calculated control signal remains within these limits, $u_k = u_k^*$. When the calculated control signal exceeds either limit, the applied actuator command is restricted to the corresponding maximum or minimum value.

When integral action is present, prolonged actuator saturation can also lead to integral windup, as discussed in the following section.

Integral Windup and Anti-Windup

Actuator saturation creates an important problem for controllers containing integral action. If the actuator is saturated while a persistent error remains, the integral state can continue to accumulate even though the actuator cannot provide any additional control effort.

This behaviour is known as integral windup. When the system later approaches the reference value, the excessively accumulated integral action must first unwind before the controller returns to its normal operating region. The result can be increased overshoot and a long recovery time.

Anti-windup methods modify the behaviour of the integrator when saturation occurs so that the internal controller state remains consistent with the control effort that can be applied.

Derivative Kick

If derivative action is calculated directly from the error, a sudden change in the reference can create a large change in error between two samples. The derivative term can consequently produce a large, short-duration control action, commonly referred to as derivative kick.

One practical solution is to apply derivative action to the measured output rather than directly to the error. A step change in the reference then does not directly enter the derivative calculation.

Initial Conditions and Controller Start-Up

A discrete-time PID controller contains stored values from previous sampling instants, such as the accumulated integral term, I_{k-1}, and the previous error, e_{k-1}. These values must be assigned suitable initial conditions before the controller begins operating.

For many simple applications, the stored values may initially be set to zero. However, if the controller is enabled while the physical system is already operating, inappropriate initial conditions can produce an abrupt change in the calculated control signal.

The initial controller states should therefore be considered whenever the controller is started, reset or re-enabled. In particular, the initial integral value should be selected so that the controller begins operating from a state that is consistent with the current condition of the physical system.

Bumpless Transfer and Tracking

Some control systems allow operation to switch between manual and automatic PID control, or between different controllers. At the instant of switching, the newly enabled controller may calculate a control signal that differs from the signal currently being applied to the actuator. This can cause a sudden change, or bump, in the actuator command.

Bumpless transfer is used to minimise this sudden change by ensuring that the PID controller output closely matches the existing actuator command when automatic control is enabled.

This is commonly achieved using tracking, where the internal controller state, particularly the integral term, is adjusted so that the calculated controller output follows the control signal already being applied to the actuator. When control is transferred to the PID controller, the output therefore continues from approximately the same value rather than changing abruptly.

PWM and Actuator Resolution

A digital controller ultimately produces a command through hardware with finite resolution. For example, an 8-bit PWM output provides:

$$2^8 = 256 \textit{ possible command values}$$

The calculated PID output must therefore be mapped onto one of the available PWM levels. If the required change in control effort is smaller than the PWM resolution, the physical actuator command may remain unchanged even though the calculated control output has changed. PWM scaling and saturation should be designed together so that the numerical controller output corresponds correctly to the physical actuator range.

Sensor Resolution and Quantisation

The measured output is also represented with finite resolution. Encoders, analogue-to-digital converters and other digital sensors cannot represent infinitely small changes in the physical variable. Small physical changes may therefore produce no change in the measured digital value, while measurements near a quantisation boundary may alternate between adjacent levels.

Quantisation can be particularly noticeable in derivative calculations because the derivative is based on the change between successive samples. Sensor resolution, filtering and sampling should therefore be considered together.

Computational Execution Time

All sensing, filtering, control calculations, communications and actuator updates required by the controller must complete within the selected sampling period. A useful practical requirement is:

$$T_{Execution} < T_s \tag{2}$$

where $T_{Execution}$ denotes the execution time. If the execution time approaches or exceeds the sampling period, controller updates may be delayed or missed. This can change the effective timing of the control algorithm. Execution time should therefore be checked on the target hardware rather than assumed from desktop simulation performance. Ideally, the controller should also execute at consistent intervals, since variation in the actual sampling interval can introduce sampling jitter and affect controller behaviour.

Practical Verification

After implementation, the controller should be evaluated using measured data from the physical system. Verification should consider both the system response and the control signal. A satisfactory output response can sometimes hide undesirable controller behaviour such as repeated actuator saturation, excessive integral accumulation or rapidly changing actuator commands. The following are examples of checks that should be carried out during practical verification:

- Check the measured output against the engineering requirements.
- Inspect the controller output and confirm that actuator limits are respected.
- Check whether the controller spends significant time in saturation.
- Observe the integral state and confirm that windup is controlled.

- Inspect measurement noise and the derivative contribution.
- Refine the controller gains, filtering and protection mechanisms where required.

4.0 Summary

Practical PID implementation requires consideration of factors beyond the controller gains. The sampling period, measurement noise, actuator saturation, integral windup, derivative kick, controller initialisation, numerical resolution and execution time can all influence the behaviour of the physical closed-loop system. Appropriate filtering, signal limits, anti-windup, controller initialisation and implementation choices can be used to manage these effects. The completed controller should then be verified using measured data from the physical system.

Return to the Learning Outcomes at the beginning of this learning guide and consider whether you can now confidently demonstrate each one.

Recommended Citation

If you have used this practical exercise in your teaching, learning or research, please cite it using the reference below, adapting the format as required by your preferred referencing style.

> Pickering, J. E. (2026). Practical Considerations for PID Control. ACE-Lab. Teaching Notes: ACE-Lab Practical Exercises, Version 1.0. Zenodo. https://zenodo.org/records/23121627