Project Report: EDMMS Temperature Controller

Jeremy Evans, Lorand Mezei, Macallister Armstrong, Anthony Kirkland

1 Purpose

This document introduces the theory of PID control and its programmatic implementation. We will define, at a high level, what a PID controller is, what its purpose is, loop tuning, and its implementation in software. It is meant to serve as an aid for designing the programming logic of a PID controller that utilizes an MSP430 MCU.

Portions of the following text may go into the final report upon approval.

1.1 Version history

Date	Author	Comments	Version
2/3/2021	Jeremy Evans	Initial document	1.0

2 Introduction

A PID controller (proportional-integral-derivative controller) is a feedback control loop mechanism that receives a desired *setpoint* $r(t)$ as input and outputs a *process variable* (PV), denoted as $y(t)$, which is then fed back into the system as input. It continuously calculates an *error value* $e(t)$ as the difference between the desired setpoint and the process variable and applies a correction based on *proportional* (P), *integral* (I), and *derivative* (D) terms, which are summed together to make up the *control variable*, $u(t)$ affecting the value of the process variable, hence the name. In applying this correction over time, it attempts to stabilize the output, i.e., eliminate the oscillation of the error value (or achieve marginal stability, or bounded oscillation, though specifications vary between applications).

Corrections are achieved by multiplying the P, I, and D terms by constants K_p, K_i, and K_d respectively, each chosen through a process known as *loop tuning*. Improperly chosen constants would result in the error value diverging (with or without oscillation), whereas properly chosen constants would have the desired effect of the error value converging.

2.1 Mathematical Definition

PID control is mathematically defined as:

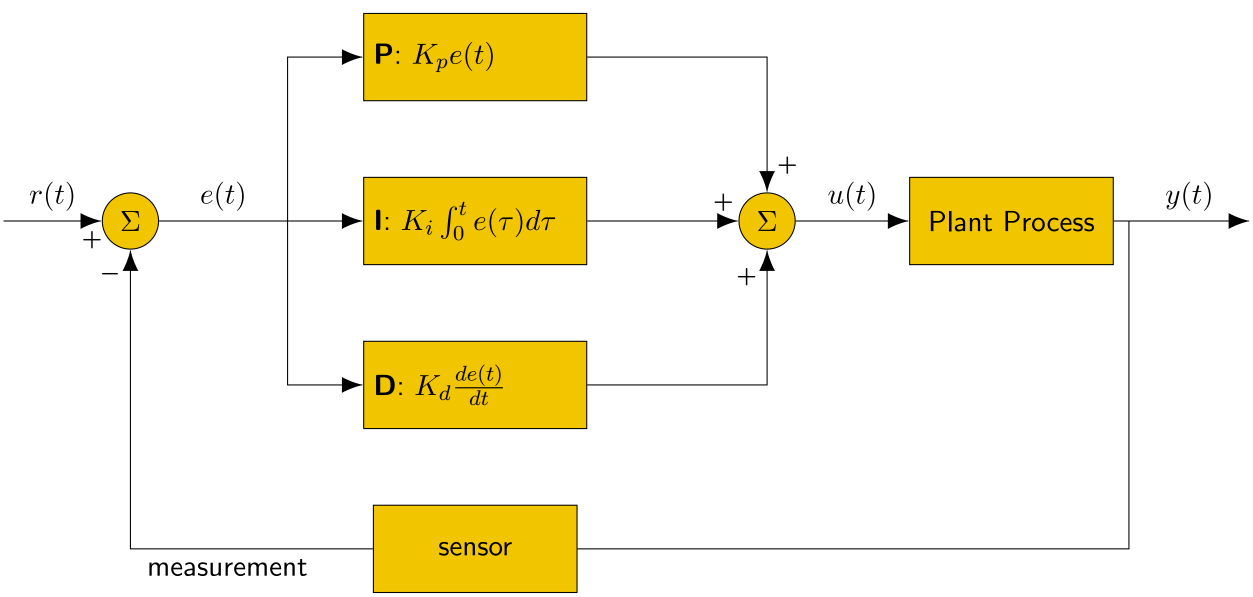
$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d(\tau) + K_d \frac{de(t)}{dt},$$

where

- $u(t)$ is the *control variable*, the parameter that is controlled,
- $K_p \in \mathbb{R} \geq 0$ is the *proportional gain,*
- $K_i \in \mathbb{R} \geq 0$ is the *integral gain,*
- $K_d \in \mathbb{R} \geq 0$ is the *derivative gain,*
- $e(t) = r(t) - y(t)$ is the error ($r(t) = SP$ is the *setpoint*, $y(t) = PV$ is the *process variable*),
- t is the *instantaneous time* (i.e., the current time),
- $\tau \in \mathbb{R}$ are measurements of time in the range $[0, t]$.

2.2 PID Block Diagram

The PID algorithm is given by the following block diagram:



Represented in a high-level model of a heat treatment furnace, $r(t)$, the setpoint, would be the desired furnace temperature; $u(t)$, the control variable, would be the gas flow rate; and $y(t)$, the process variable, would be the measured furnace temperature.

2.3 PID Terms

2.3.1 Proportional

Given by $P_{out} = K_p e(t)$, the proportional term produces an output that is proportional to the current error value $e(t)$; the greater the error value, the greater the control output. This output is multiplied by a gain factor $K_p \in \mathbb{R} \geq 0$ that determines how responsive the controller should be with a given error-value. Large values of K_p result in a large change in the output for a given change in the error. However, exceptionally large values may result in the output *overshooting* the setpoint (exceeding the value of the setpoint) and, in the worst case, destabilizing the system whereupon the error rate diverges.

2.3.2 Integral

Given by $I_{out} = K_i \int_0^t e(\tau) d\tau$

2.3.3 Derivative

Given by $K_i \int_0^t e(\tau) d\tau$

2.4 Loop Tuning