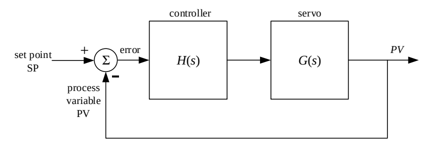
5.4 Proportional, Integral, and Derivative Controllers

Proportional Control

The goal of servo tracking is to make the system output, *PV* follow the set point, *SP* as close and as quickly as possible. So, the *system* gain should approach 1, and its time constant should be as small as practical.

For operations in the Laplace domain to be valid, the system must be *linear*, that is the controller is never sent to its maximum or minimum. This just requires that changes are kept small, and that the controller output, *CO*, is monitored before declaring that a result is valid.

A *proportional* controller outputs a signal that is proportional to its input.



$$\boldsymbol{\frac{PV}{SP} = \frac{GH}{(1 + GH)}}$$

$$CO = k_{\mathrm{p}} \times error$$

This means $H = k_{\mathrm{p}}$ where k_{p} is a constant

A proportional controller is easy to implement. In hardware, it is just a noninverting op amp amplifier. In software, it's just a multiplication.

Assuming that the system under control is a first order lag element, like the motor of many previous examples

$$G = \frac{m}{\tau s + 1}$$

$$\frac{PV}{SP} = \frac{GH}{1 + GH} = \frac{\frac{m}{\tau s + 1} \times k_{\mathrm{p}}}{1 + \frac{m}{\tau s + 1} \times k_{\mathrm{p}}}$$

$$\frac{PV}{SP} = \frac{\frac{mk_{\mathrm{p}}}{\tau s + 1}}{1 + \frac{mk_{\mathrm{p}}}{\tau s + 1}}$$

$$\frac{PV}{SP} = \frac{\frac{mk_{\mathrm{p}}}{\tau s + 1}}{\frac{\tau s + 1}{\tau s + 1} + \frac{mk_{\mathrm{p}}}{\tau s + 1}} = \frac{\frac{mk_{\mathrm{p}}}{\tau s + 1}}{\frac{\tau s + 1 + mk_{\mathrm{p}}}{\tau s + 1}}$$

$$\frac{PV}{SP} = \frac{mk_{\mathrm{p}}}{\tau s + 1 + mk_{\mathrm{p}}}$$

$$\frac{PV}{SP} = \frac{mk_{\mathrm{p}}}{\tau s + \left(mk_{\mathrm{p}} + 1\right)}$$

The resulting servo tracking system is still first order lag. The standard transfer function form has a denominator of $\tau s+1$. Divide the numerator and denominator by mk_p+1.

$$\frac{PV}{SP} = \frac{\dfrac{mk_{\mathrm{p}}}{mk_{\mathrm{p}} + 1}}{\dfrac{\tau}{mk_{\mathrm{p}} + 1}s + 1}$$

The open loop motor's transfer function gain, *m*, may vary considerably which caused a similar variation in performance. But the servo tracking closed loop's gain is

$$A_{\mathrm{prop}} = \frac{mk_{\mathrm{p}}}{mk_{\mathrm{p}} + 1} \leq 1$$

This could be very close to 1. The steady state *PV* is close to *SP* and is not very sensitive to changes in the motor's *m*.

The open loop motor's transfer function time constant, dictates how quickly the motor responds. But the servo tracking closed loop's time constant is

$$\tau_{\mathrm{prop}} = \frac{\tau}{mk_{\mathrm{p}} + 1}$$

which may be much smaller than the motor's time constant alone.

From Example 4-2

m= 206 RPM/V
τ= 0.362 sec
ω_0 = 277 RPM

$$G = \frac{mV}{(\tau s + 1)} + \frac{\tau\omega_0}{(\tau s + 1)}$$

Example 5-2

Verify these performance improvements for the motor used in the previous examples. To assure the system remains linear, set

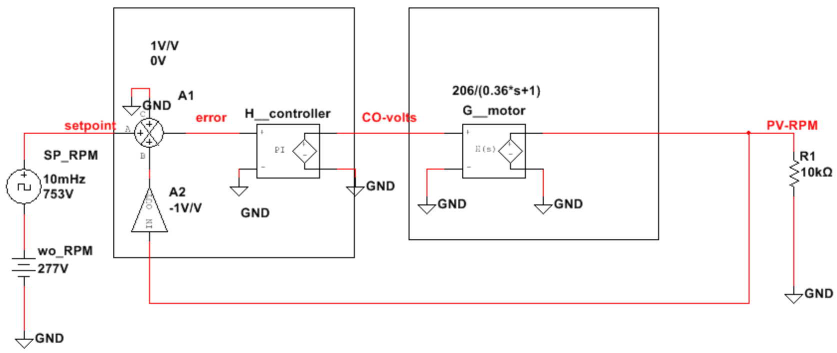
$$\omega_0 = 277\ \mathrm{RPM}$$

$$\Delta SP = 1030\ \mathrm{RPM} - 277\ \mathrm{RPM} = 753\ \mathrm{RPM}$$

Solution

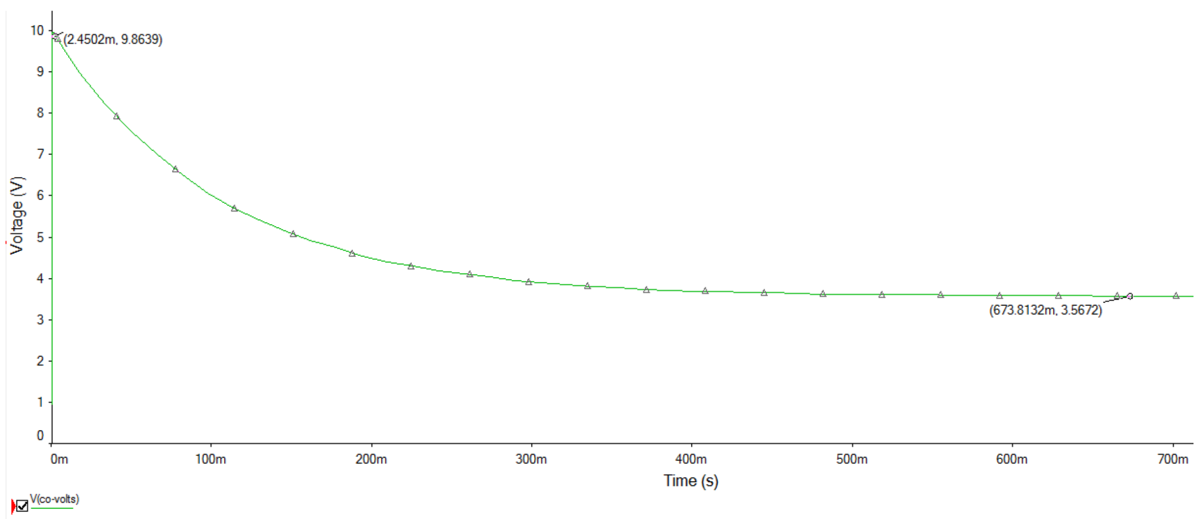
The schematic and that analysis are shown in Figure 5-13.

MULTISIM



(a) Schematic for Example 5-2

Laplace transform math requires that the system remain linear. So, k_p must be adjusted to assure that the controller output never hits its maximum ($\pm$ 10 V for this example). After a few attempts the largest k_p without sending the controller to its maximum is $k_p = 0.012$. Figure 5-13(b) is a plot of the controller output in response to the set point step. *CO* < 10 V.



(b) Controller output remains < 10 V, the controller's maximum

Figure 5-13 Setup for Example 5-2 *Proportional* servo tracking $k_p = 0.012$

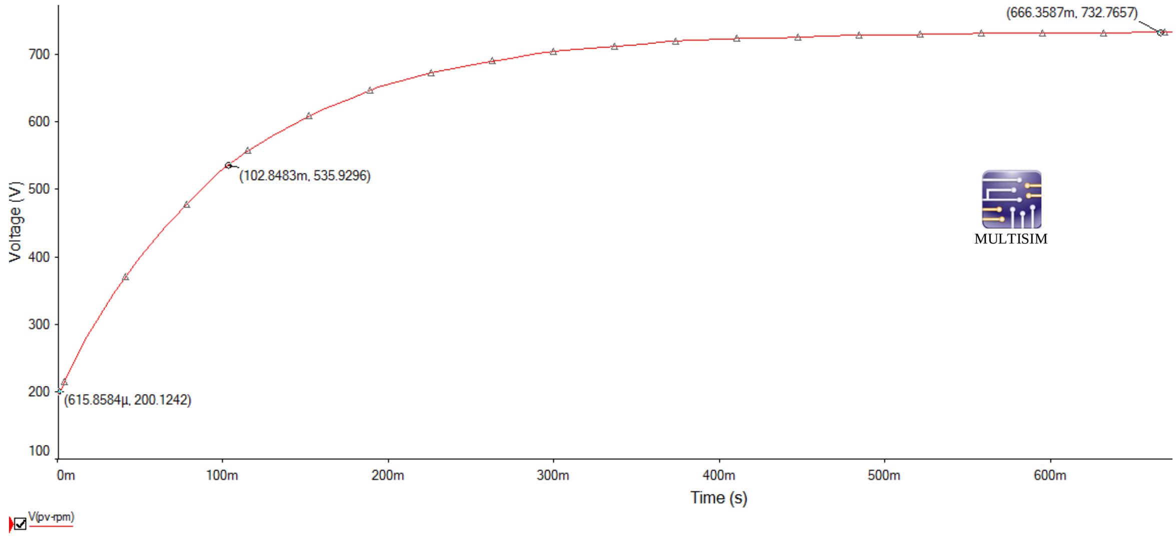


Figure 5-14 Motor proportional control wit $k_p = 0.012$, $m = 206$ RPM/V, $\tau = 0.36$sec

$$\Delta PV = 733 \text{ RPM} - 200 \text{ RPM} = 533 \text{ RPM}$$

$$A_{\text{prop-sim}} = \frac{\Delta PV}{\Delta SP} = \frac{533 \text{ RPM}}{753 \text{ RPM}} = 0.71$$

$$A_{\text{prop-theory}} = \frac{mk_{\text{p}}}{mk_{\text{p}} + 1} = \frac{206 \times 0.012}{206 \times 0.012 + 1} = 0.71$$

Theory and simulation gains match.

$$\Delta PV_{1\,\tau-sim} = 0.63 \times 533 \text{ RPM} = 336 \text{ RPM}$$

$$PV_{1\,\tau-sim} = 336 \text{ RPM} + 200 \text{ RPM} = 536 \text{ RPM}$$

From 5-14, the time to $PV_{1\,\tau\text{-sim}} = \tau_{\text{sim}} = 103$ msec

Theory indicates that

$$\tau_{\text{prop}} = \frac{\tau}{mk_{\text{p}} + 1} = \frac{0.36 \text{ sec}}{206 \times 0.012 + 1} = 104 \text{ msec}$$

Theory and simulation time constants correlate well.

The *PV* can never perfectly track the *SP*, because

$$\frac{PV}{SP} = A_{\mathrm{prop}} = \frac{mk_{\mathrm{p}}}{mk_{\mathrm{p}} + 1} \leq 1$$

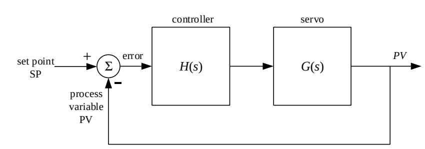
The error in the steady state *PV* is call *residual error*.

$$residual\ error = \frac{SP - PV}{SP} \times 100\%$$

With a little algebra, the gain equation can be manipulated to give

$$residual\ error = \frac{1}{mk_{\mathrm{p}} + 1} \times 100\%$$

Increasing the proportional constant, k_p lowers this error, allowing the output to more closely track the set point. But, be careful. Raising k_p increases the size of the controller's output. For a large step, or large k_p the controller could be driven to its maximum value. At that point, all of the linear math fails. Even though the simulations in Figure 5-10 through 5-12 drive *CO* to its limits and produce good results, in practice it is a *bad* idea to allow part of a control system to max out. With the controller at its maximum value, the system can no longer respond to further increases, meaning that control has been lost. Additionally, often elements take an inordinately long time to come out of saturation. During that time, the system spins on at full speed with no supervision.



The largest acceptable k_p can be calculated. The system is driven to its maximum output, PV_{max} , when the controller is at it's maximum, CO_{max}. If the system's gain may change, use the smallest gain to assure that the controller can push the system to its limit.

$$PV_{\mathrm{max}} = m_{\mathrm{min}} \times CO_{\mathrm{max}}$$

In a proportional controller system, CO_{max} is

$$CO_{\mathrm{max}} = k_{\mathrm{p\ max}} \times (SP_{\mathrm{max}} - PV_{\mathrm{min}})$$

This worst case comes at startup, when the set point is asking for full output, but the servo is just beginning to respond, $PV_{min} = 0$.

$$CO_{\mathrm{max}} = k_{\mathrm{p\ max}} \times SP_{\mathrm{max}}$$

Substituting this into the original equation for PV_{max}

$$PV_{\max} = m_{\min} \times k_{\text{p max}} \times SP_{\max}$$

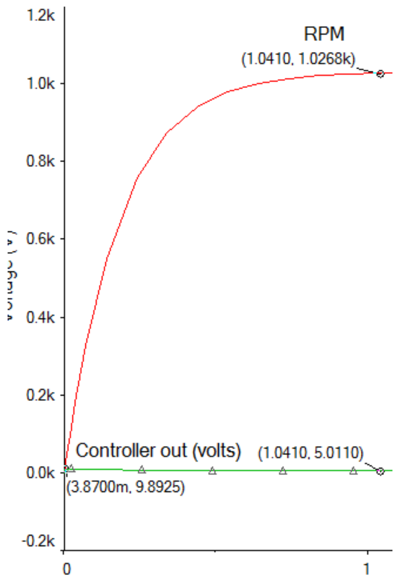
$$\frac{PV_{\max}}{SP_{\max}} = m_{\min} \times k_{\text{p max}}$$

The whole goal of negative feedback is to cause *PV* = *SP*.

$$1 = m_{\min} \times k_{\text{p max}}$$

Worst case k_p

$$k_{\text{p max}} = \frac{1}{m_{\min}}$$



Proportional only control
SP = 2060 RPM

k_p = 1/m produces fast response with 50% residual error

This is a pretty small value and results in 50% residual error. When the system does not experience full-scale transients or errors, then these calculations should be completed with the practical limits for that system.

Integral Control

The proportional controller can never produce a perfect result. Increasing k_p reduces the residual error, until the output of the controller is driven to its maximum, and control is lost. There's a better way.

The *integral* controller's output depends on the integral of the error.

$$co = k_i \int error\ dt$$

where *co* and *error* are functions of *time*.

To understand why this relationship can drive the *error* to zero, differentiate both sides of the equation

Integral controllers drive the residual error to zero.

How rapidly the controller output, *co*, <u>changes</u> depends on the *error* and its constant, k_i. Right after a step in set point, the *error* is large and the controller output <u>changes</u> quickly. It continues to change, though more slowly, until the *error* is zero. If there is *error* the output changes.