A Quick and Dirty PD Position Controller Design Technique

Finding useable starting values for Kp and Kd when a 'good enough' PD position controller is all you need.

Peter Harrison
UK Micromouse and Robotics Society
peterharrison@ukmars.org

I. Introduction

Many systems use relatively position controllers driven by a DC motor. Examples include mobile robot platforms, X-Y positioners and servos. Ideally, these would make use of carefully designed control loops to ensure all the requirements for static and dynamic accuracy, repeatability and response time are met under any foreseeable disturbance. Designing and tuning such control systems can be a challenging and time consuming task.

Practically though, there is often a need for a simpler controller that is just good enough to perform a given task. Often such a controller can be combined with a suitable feed forward loop to get good results from simple and basic hardware and software. Such a case applies to a variety of wheeled mobile robot platforms.

This article describes a method for calculating the constants K_d and K_p in a PD controller used in a wheeled mobile robot. The derived constants are unlikely to be optimal for every task but will provide at least a good, reliable starting point for further empirical tuning.

II. The Controlled System

A robot is driven by a pair of wheels connected to DC motors through a gearbox. There is one motor per wheel. To model the forward motion of the robot, it is convenient to treat it as a unicycle with a single drive motor and a single wheel. When a voltage is applied to the motor, the wheel will turn and the robot will move backwards or forwards at a speed determined by the sign and magnitude of the applied voltage.

After a step change in voltage, the robot cannot instantly move at a new speed. Instead, it will accelerate until the speed reaches some final, steady-state value proportional to the voltage. the constant of proportionality is the *gain* of the system and given the symbol K_m.

There are several factors that determine how quickly the robot will reach its new steady state speed. The speed will increase according to a combination of exponential functions. Each of these has its own time constant.

The greatest influence comes from a combination of the available torque from the motor and the robot's mass. The time constant associated with this will be given the symbol T_m.

There is also a smaller contribution from the electrical characteristics of the motor - its resistance and inductance. For simplicity, you can ignore the electrical time constant for now as it has only a small effect in the modelled example.

III. Control System Blocks

The robot and its controller can be modelled as a pair of blocks, each with its own transfer function. These are shown in Fig. 1.

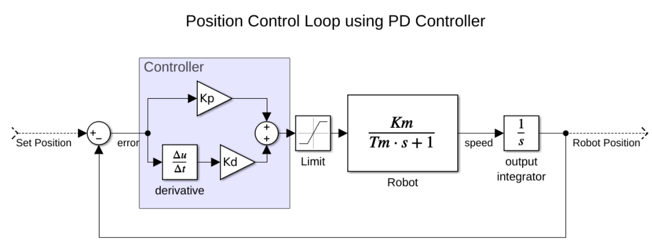

Fig. 1. System Block Diagram

A. Robot

For modelling purposes, you can describe the robot just in terms of its gain, K_m and time constant, T_m. For a small robot like the UKMARSBOT, using 20:1 micrometal gear motors, you can easily determine the gain and time constant. They will have values similar to these:

$$K_m = 265\mathrm{mm\,s^{-1}\,V^{-1}}$$
$$T_m = 0.110\mathrm{s}$$

The basic first-order transfer function describing the *speed* is then

$$\frac{K_m}{T_m s + 1}$$

But the robot needs the *position* to be controlled so there must also be an integrator and so the correct transfer function is

$$G_m = \frac{K_m}{T_m s + 1} \cdot \frac{1}{s}$$

$$G_m = \frac{K_m}{T_m s^2 + s} \quad (1)$$

B. Controller

Although the most common kind of controller described is the PID (Proportional, Integral and Derivative) it is possible in this case to use a simpler variant, the PD controller. The Integral term is generally used when the controlled system must maintain its output with no error over time. In the case of a constantly moving robot, that is not an essential requirement and the simpler PD controller makes analysys and controller design a good deal easier while being more than capable of generating good results.

The controller is defined by just two constants - the proportional gain, K_p and the derivative gain, K_d. These are the two quantities that have to be calculated for a stable and robust control system that also has adeqautely rapid response.

A simple transfer function describes the PD controller:

$$G_c = K_d s + K_p \quad (2)$$

IV. System Transfer Function

The complete system is the product of the robot and controller transfer systems:

$$G_o = G_c G_m$$

$$= (K_d s + K_p) \cdot \frac{K_m}{T_m s^2 + s}$$

$$= \frac{K_m(K_d s + K_p)}{T_m s^2 + s} \quad (3)$$

Equation 3 is the *open loop* transfer function for the system and is used for things like stability analysis. In the actual controlled system G_s is part of a unity feedback loop and the closed loop transfer function of the complete system is

$$G_s = \frac{G_o}{G_o + 1}$$

$$= \frac{\frac{K_m(K_d s+K_p)}{T_m s^2+s}}{\frac{K_m(K_d s+K_p)}{T_m s^2+s} + 1}$$

$$= \frac{\frac{K_m(K_d s+K_p)}{T_m(s^2+s)}}{\frac{K_m(K_d s+K_p)+T_m s^2+s}{T_m s^2+s}}$$

$$= \frac{K_m(K_d s + K_p)}{K_m(K_d s + K_p) + T_m s^2 + s}$$

$$= \frac{K_m(K_d s + K_p)}{T_m s^2 + (K_m K_d + 1)s + K_m K_p}$$

$$= \frac{K_m}{T_m} \cdot \frac{K_d s + K_p}{s^2 + \frac{K_m K_d + 1}{T_m} s + \frac{K_m K_p}{T_m}} \quad (4)$$

Which probably looks awful if you are out of practice. However, eq.4 has a very similar form to a common second-order transfer function and its behaviour will generally be dominated by the denominator of the equation. Specific behaviour will be modified by the terms in the numerator but they will have a smaller effect. Because of its dominance, the denominator is called the characteristic equation and it can be compared directly to one written in standard form:

$$H = A \cdot \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (5)$$

Such equations are well understood and a lot of effort has gone into developing some useful rules of thumb associated with them. It is often a good idea to try and model a system in this form where possible for the convenience of using these rules of thumb. If practical results do not match up with theory then you might need to go back and try a more complex model. A lot of the time though, this will be enough.

V. Settling Time Approximation

Comparing terms in (4) and (5) gives you these relationships.

$$\omega_n^2 = \frac{K_m K_p}{Tm}$$

$$2\zeta\omega_n = \frac{K_m K_d + 1}{Tm}$$

Which can be rearranged to make K_p and K_d the subject.

$$K_p = \frac{T_m \omega_n^2}{K_m} \tag{6}$$

$$K_d = \frac{2\zeta\omega_n T_m - 1}{K_m} \tag{7}$$

For a generic second-order system, there is an approximate relationship between the settling time (T_D), the damping ratio (ζ - pronounced 'zeta') and the natural frequency (ω_n).

$$\omega_n \approx \frac{4}{\zeta T_D} \tag{8}$$

Substituting for ω_n, you can see that it is possible to define the beaviour of the controller in terms of the damping ratio, ζ and the settling time T_D.

The quantities K_m and T_m in (6) and (7) will have been measured or calculated from the properties of the robot. That leaves you with the question of what values to chose for ζ and T_D so that you get a suitable response.

A. Damping Ratio

The damping ratio, ζ, has a significant effect on the amount of overshoot in a system step response. Many systems aim to have $0.5 < \zeta < 1.0$. At the upper extreme, an ideal system would be *critically damped* - it would have no overshoot and take a little longer to get to the set point. At the lower extreme there would be about 16% overshoot but the system would get to the setpoint quite quickly. A convenient value for ζ is 0.707. This value should result in an overshoot of about 5% and it is also handy to note that, at that value, $\zeta^2 = 0.5$

B. Settling time

Although the damping ratio will affect how quickly the output will get to within some error margin of the set point, if there is any significant overshoot, the output can rapidly exceed the error margin. It is common to define the settling time, T_D, as the time taken for the response to stay within 2% of the steady-state value. That value is chosen because it is quite small and corresponds roughly to 4 time-constants for the transient part of the response. That is where the number 4 comes from in (8).

VI. Controller Constant Estimation

With an understanding of the two parameters that you need to choose for the control system, it is time to consider what might be good starting points for their values. Previously, it was mentioned that a convenient value for ζ is 0.707 so you can start with that. The original system had a measured value for its time constant, T_m, so what happens if you make $T_D = T_m$.

Substitute from (8) into (6)

$$K_p = \frac{T_m \omega_n^2}{K_m}$$

$$K_p = \frac{T_m}{K_m} \cdot \left(\frac{4}{\zeta T_D}\right)^2$$

$$= \frac{T_m}{K_m} \cdot \left(\frac{16}{\zeta^2 T_D^2}\right)$$

but $T_D = T_m$ and $\zeta = 0.707$

$$K_p = \frac{T_m}{K_m} \cdot \left(\frac{16}{0.5 T_m^2}\right)$$

$$K_p = \frac{32}{K_m T_m} \tag{9}$$

Similarly, substitute from (8) into (7)

$$K_d = \frac{2\zeta\omega_n T_m - 1}{K_m}$$

$$= \frac{\frac{8\zeta T_m}{\zeta T_D} - 1}{K_m}$$

but $T_D = T_m$

$$K_d = \frac{7}{K_m} \tag{10}$$

VII. System Response

The two equations (9) and (10) seem far too simple to give proper results for nn apparently complex system. Remember that they are estimates - a possible starting point for more tuning. Even so, there are circumstances where they are completely adequate. For example, in a robot position system with good feed forward control and reasonably high encoder resolution, it is likely that no further tuning will be required.

Consider the system defined at the start of the article with $T_m = 0.110$ and $K_m = 265$

First, look at how the system will behave if there is no control scheme in place at all (Fig. 2). This is the step response of the original system without a controller and unity feedback. You can see that the system will eventually settle at the commanded new position but after half a second there is still significant oscillation.

Now the PD controller is added to the system with the values as calculated from (9) and (10). That is, $K_p = 1.098$ and $K_D = 0.0264$. the step response is shown in Fig (3). The response has an overshoot of 17% and a settling time very close to 0.1 seconds. That is a greater overshoot than the 5% expected and a slightly shorter settling time.

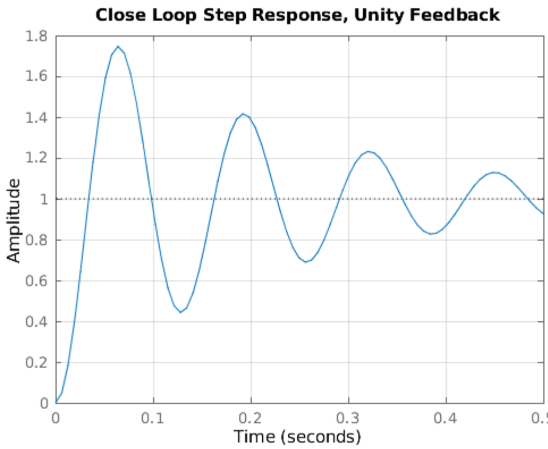

Fig. 2. Step Response With Unity Feedback

The variation is caused by the other terms in the closed loop transfer function and the loose estimates used in the calculations. In spite of that, the overall response is likely to be as good as many might expect after laborious manual tuning experiments.

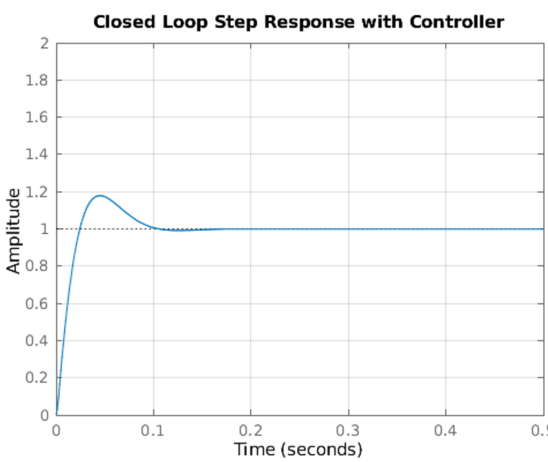

Fig. 3. Step Response With PD Controller

VIII. Tracking Response

The example control system just described was incorporated into a simulink model to test its ability to track a velocity profile. The profile consist of three phases - acceleration at 3000mm/s, constant speed at 1000mm/s and then braking to a halt with deceleration of 3000mm/s.

Figure 4. demonstrates that with no controller and just a unity feedback loop, the ability of the system to track the profile is poor though perhaps better than might be expected from the step response. When used on an actual robot a tracking response like this may appear acceptable to an uncritical eye and it is unlikely that an observer would be able to notice the errors. In practice though, this response is likely to become unstable or oscillatory in the presence of disturbances.

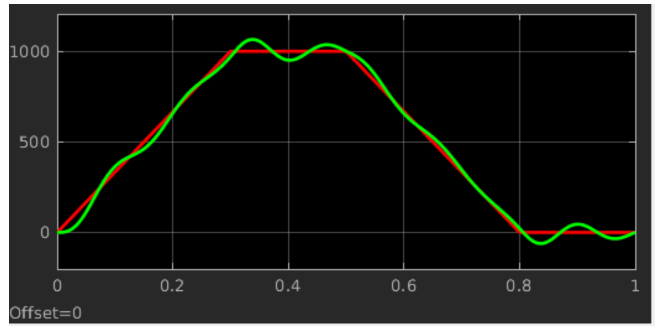

Fig. 4. Velocity Profile Tracking With Unity Feedback

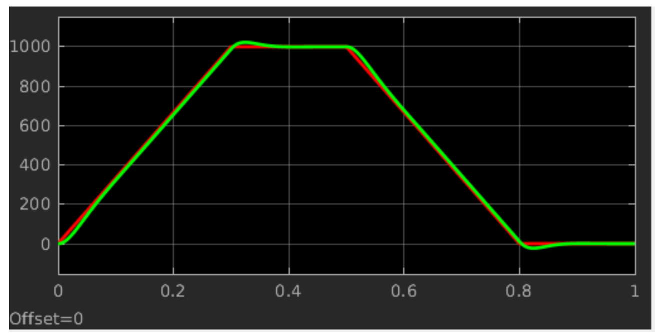

Fig. 5. Velocity Profile Tracking With PD Controller

In figure 5. it is clear that using the calculated controller in the closed loop system, results in very good tracking. Inevitable, but small, errors are present when the speed changes though these will be insignificant in the physical reponse of a real robot.

IX. Conclusion

This article has introduced a very easy way to calculate good estimates for the PD controller constants, K_p and K_d using two simple equations (9) and 10). This is possible because of some simplifying assumptions:

1) A damping ratio, ζ, of 0.707 will give a suitable amount of overshoot in the step response
2) A settling time, T_D, equal to the system time constant is of an acceptable length

These assumptions just make the calculations easier. Other values of ζ and T_D could be chosen and substituted into (6), (7) and (8) to adjust the response to suit the exact application.