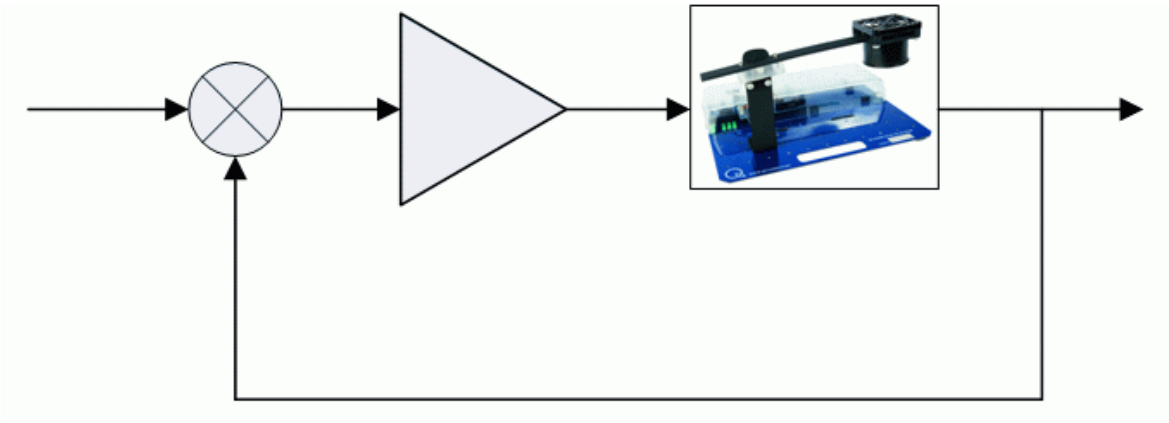
QNET: HVACT, DCMCT, ROTPENT, MECH-KIT, VTOL, and MYOELECTRIC

Quanser Engineering Trainer for NI-ELVIS

QNET Practical Control Guide



Acknowledgments

Thanks to Karl J. Åström for his immense contribution to this project.

Table of Contents

1. Introduction

Feedback has many useful properties, for example: it permits design of good systems from poor components, unstable systems can be stabilized, and effects of disturbances can be reduced. Combining these nice properties with the advances in computing and software, which has made design simpler and implementation cheaper, it is easy to understand why applications of control are expanding rapidly. The concepts of control are also essential for understanding natural and man-made systems. A recent panel [1] gives the following recommendation: *Invest in new approach to education and outreach for the dissemination of control concepts and tools to nontraditional audiences. The panel report goes on to say: As a first step toward implementing this recommendation, new courses and textbooks should be developed for experts and non experts. Control should also be made a required part of engineering and science curricula at most universities including not only mechanical, electrical, chemical, and aerospace engineering, but also computer science, applied physics and bioengineering. It is also important that these courses emphasize the principles of control rather than simply providing the tools that can be used for a given domain. An important element of education and outreach is the continued use of experiments and the development of new laboratories and software tools. This is much easier to do than ever before and also more important. Laboratories and software tools should be integrated into the curriculum.*

The laboratories described in this book are designed to implement some of the recommendations. Since control is a systems field, to get a full appreciation of control it is necessary to cover both theory and applications. The skill base required in control includes modeling, control design, simulation, implementation, commissioning, tuning, and operation of a control system [1], [2]. Many tasks can be learned from books and computer simulations but laboratory experiments are necessary to obtain the full range of skills and a deeper understanding. The experiments in this book can be used in a self-contained course. They can also be used to augment traditional text books such as [3], [4], [5] and [6] with laboratories. The experiments can be used in many different ways, in structured classes as well as for demonstrations and for self-study. Further, they can be used to give students from science and biology an introduction to control.

A careful selection was made to obtain a set of experiments that illustrate essential ideas, typical processes and applications. Process control and motion control are two common application areas. In process control a typical task is to keep a process variable constant in spite of disturbances. This type of problem is called a *regulation problem*. In motion control a typical task is to make an object move in a specified manner. This task is called a *servo problem*.

Typical examples of regulation are found, in process industries such as petrochemical and pulp-and paper, in heating ventilation and air-conditioning, and in laboratory systems. In process control it is often difficult or time consuming to develop mathematical models of the processes. The information required to control the process is therefore often deduced directly by experiments on the process. Controllers can also be installed and tuned by experiments without resorting to a model. We have chosen a heating process as a typical process to illustrate regulation problems.

Typical examples of motion control are found in the manufacturing industry, in scanners, printers, cameras, robots, CD players, vehicles, and instrumentation. A characteristic feature of motion control is that it is often possible to obtain mathematical models of the systems from first principles, possibly with a few complementary experiments. We have chosen a simple DC motor to illustrate motion control. Typical experiments are to control the speed or the motor angle in desired ways.

Even if there are many applications of regulation and servoing, there are many other types of control problems. Stabilization of an unstable system is one task, the transporter Segway is a typical example. Damping of a swinging load on a crane, motion planning for a moving robot, traction control of cars are other examples. These control tasks are typically more difficult than regulation and servoing and they may require more advanced modeling and control. In spite of this we judged that it was important to have a simple demonstration of *task-based control*. We have chosen a rotary pendulum and a vertical take-off and landing device as examples of task-based control. The pendulum is a classic system that has been used to teach dynamics and control for a long time. Even if the processes are simple they illustrate many real-life control systems. The vertical take-off and landing device presents a different set of modeling and control challenges geared towards aerospace applications.

Although it does involve controlling an actual device, instrumentation is a very important aspect in controls. This involves understanding how to use different types of sensors and switches. Various sensors are found in all types of industry. Magnetic field transducers are used to detect the throttle in vehicles. Sonar and infrared sensors are often used in mobile robots to measure the distance of surrounding objects. We have chosen a mechatronic sensors device to illustrate how to use different types of sensors as well as switches and

light-emitting diodes.

The processes are controlled using a PC running the National Instruments programming environment LabVIEW. These processes are manufactured by Quanser and are called Quanser Engineering Trainers for NI ELVIS, or QNET for short. The processes are connected to the computer using the National Instruments Educational Laboratory Virtual Instrumentation Suite (NI ELVIS). As shown in Figure 1.1, the QNET board slides into the NI ELVIS II device. The QNET is compatible with both the traditional NI ELVIS, called NI ELVIS I, and the ELVIS II. The ELVIS I workstation is connected to a multi-function data acquisition card that is installed inside the PC. On the other hand, the NI ELVIS II has its own DAQ device that connects to the PC via USB. Using LabVIEW it is possible to implement user interfaces that are easy to used. For example, Figure 1.2 shows the front panel for the PI Control of the QNET HVAC Trainer process. It is also possible to look under the hood and see precisely how the controllers and the user interface are implemented.

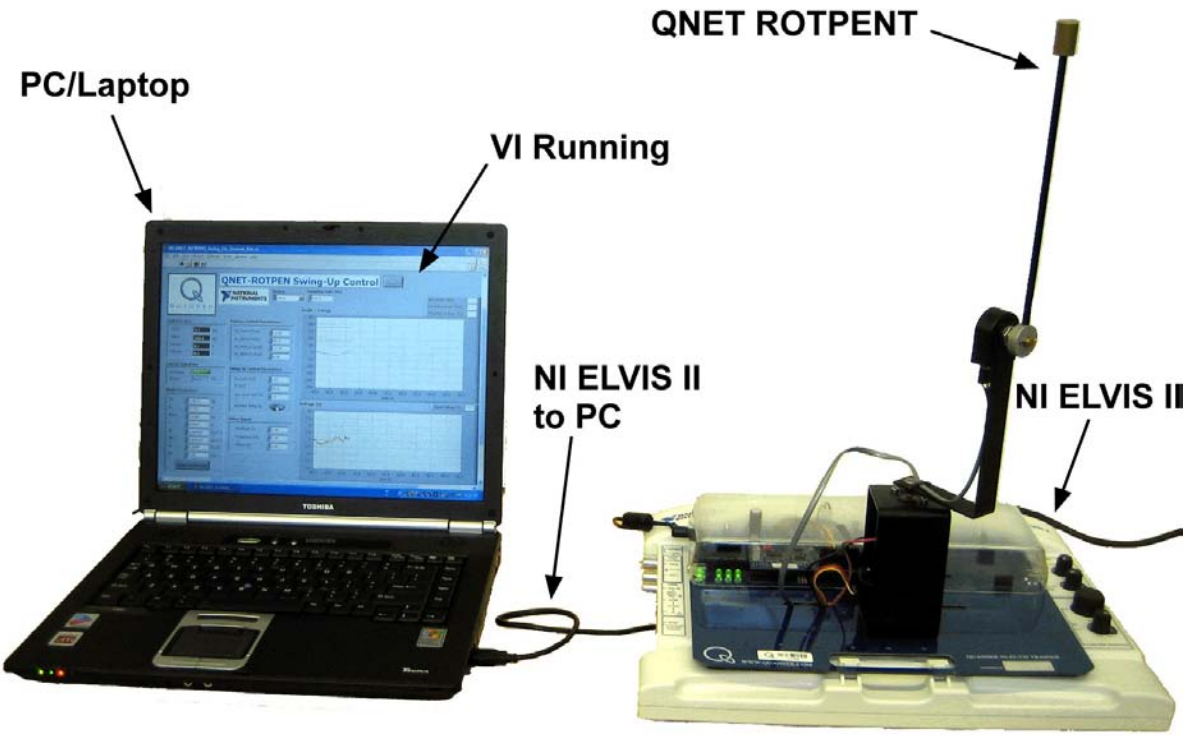


Figure 1.1: QNET Rotary Pendulum Trainer and NI-ELVIS II setup.

The experiments can be performed in many different ways but there are two extremes: the traditional laboratory mode and the guide mode. The traditional mode is to have pre-lab assignments, lab execution, and report writing. It is the more analytical and detailed

approach. In the more intuitive guide mode, the laboratories involve immediately running the VI for the experiment and following a procedure outlined in the laboratory manual. The laboratory manuals is a set of brief step-by-step instructions that takes the user through some experiments. It also includes exercises to test the student. The guide mode is also recommended for quick demonstrations (for instance, using a projector) and for students who can work with less structured instruction. Intermediate forms of instruction can be made by combining the modes. Some experiments can be made in guide mode and others in the traditional mode.

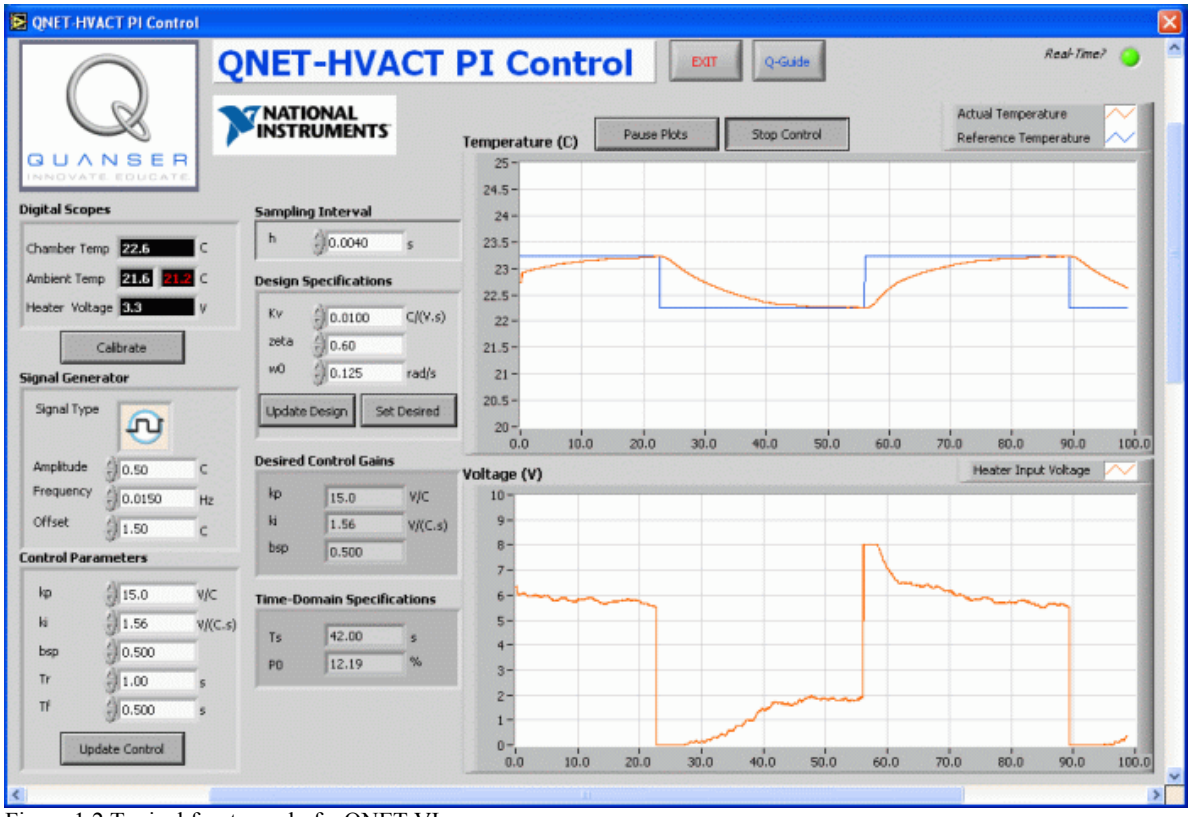


Figure 1.2 Typical front panel of a QNET VI.

The book is organized as follows. Chapter 2 gives a few hints about practical issues in control. In Chapter 3, an overview of on-off and PID controllers is given. This material is essential for the experiments, but it can be replaced with similar material in the textbooks that the students use. Chapter 4 is a short introduction to LabVIEW. It demonstrates how controllers can be implemented in LabVIEW and how LabVIEW can be used to simulate control systems. Short descriptions of the QNET processes and the experiments are given in Chapters 5, 6, 7, 8, and 9. References are given in Chapter 11.

2. Control Practice

Control is a well developed discipline with a good design methodology that is well supported by software. A control system consists of a process, sensors, actuators and a controller. The control law is an algorithm which describes how the signal sent to the actuator are obtained by processing the signals from the sensors. The control algorithm is typically specified as a differential- or a difference equation. The control algorithm is typically implemented as a program in the computer. It is highly advantageous to make an integrated design of the complete system including process design, location of sensors and actuators. However, a control engineer if often asked to control a process with specified sensors and actuators.

There are two different approaches to obtain a practical solution to a control problem: empirical or analytical. tuning. When using empirical tuning a standard controller is connected to the sensors and actuators and the parameters are obtained by empirical adjustment. In analytical tuning a mathematical model of the process is first developed and the control algorithm is then obtained by a variety of analytical procedures. In practice it is quite common that the two approaches are combined. Even if empirical tuning is used it is essential to know the system well before control is attempted.

Although practicing industrial control engineers do not typically derive models of the system, they are controlling (the authors have seen heuristic manual tuning performed in some of the most demanding applications). This experiment stresses the importance of "**knowing the system before you control it**". This is also necessary to have a broader understanding of control. The students derive the theoretical open-loop model of the system and assess its performance limitations. The system is designed in such a way that a good model can be derived from first principles. The physical parameters can all be determined by simple experiments. Using VIs and the QNET, the students perform experiments with its inputs and observe its outputs. Open-loop tests are performed and system parameters are estimated using static and dynamic measurements. A first-order simulation of the derived model is run in parallel with the actual system and a bump-test is performed to assess the validity of the estimated model.

The procedure used when applying empirical tuning can be summarized in the following steps:

- Understand the system
- Choose a controller and connect it to the system
- Commission the system
- Run and evaluate

The crucial step in empirical tuning is to choose the control algorithm. A first cut of this choice is very easy because fortunately a PI or PID controller is often sufficient at least for processes with one input and one output. It is therefore important that any user of control has a good understanding of the PID controller. This is the reason why the PID controller is covered extensively in the experiments. Design of more complex controllers require more knowledge than is covered in introductory courses in control. Design of such controllers is however simplified by the availability of good software. Experience indicates that it is difficult to adjust more than two parameters empirically. This is one reason why most industrial controllers are based on PI control, derivative action is used rarely.

Analytical tuning can be described by the following steps

- Understand the system
- Develop a mathematical model
- Design a controller
- Simulate and validate
- Implement the controller
- Commission the system
- Run and evaluate

Analytical tuning has more steps and is more complicated. However, it has the significant advantage that it is possible to find the factors that limits the achievable performance. When using empirical tuning we never know if it is possible to get better results by using a more complicated controller. Traditional control courses give much emphasis on design of controllers, they also cover modeling and simulation. Availability of systems like LabVIEW makes it very easy to implement controllers because there are standard blocks for PID control. A controller specified by a differential equation or a difference equation can also be implemented easily.

Notice that there are several steps in both empirical and analytical tuning that are not covered in typical control courses, namely

- Understand the system
- Commission the system
- Run and evaluate

The purpose of this book and the associated experiments are to cover these aspects. Since control covers so many fields the first step is very domain dependent. In this particular case we require that the students develop a good understanding of the particular laboratory systems. This is also a good opportunity to review other courses in engineering.

Control is a systems subject. It is when a system has to be commissioned that all pieces of a system come together and it is a challenge to make sure that everything works. To commission a system it is necessary to have a good understanding of all the elements, process, sensor, computer, software and actuator. Commissioning a large system can be quire a scary task, but when it is mastered it also gives the engineer a lot of pride: I made it work! A system seldom works the first time and it is necessary to develop skills in finding the faults. Large companies have engineers who specialize in this task. Control laboratories can be a good introduction to commissioning. Commissioning is a typical skill that is best learned in the tutor/apprentice mode. A few guidelines can be given. A good system should have a stop button so that it can be immediately disconnected if something goes wrong. To start with it is useful to make sure that the control signals are influencing the plant and that the sensors give reasonable signals. For stable systems it is a good idea to make small changes in the control variable in open loop and to observe how the system and the signals react. Make sure that all signs are correct. If the loop is broken at the controller output you can also see how the controller is reacting to the signals. Finally the loop can be closed, with small controller gains. The system can be gently prodded by changing reference values and disturbances.

3. On-Off And PID Control

The idea of feedback is to make corrective actions based on the difference between the desired and the actual value. This idea can be implemented in many different ways. In this chapter we will describe on-off and PID control which are common ways to use feedback.

3.1. On-Off Control

A simple feedback mechanism can be described as follows:

$$u = \begin{cases} u_{max} & 0 < e \\ u_{min} & e < 0 \end{cases} \qquad [3.1]$$

where

$$e = r - y \qquad [3.2]$$

is the control error which is the difference between the reference signal, r, and the output of the system, y. The control law implies that maximum corrective action is always used, which explains the name on-off control.

A system with on-off control will always oscillate, in many cases the amplitude of the oscillations is so small that they can be tolerated. The amplitude of the oscillations can also be reduced by changing the output levels of the controller. This will be discussed in Chapter 5 which deals with temperature control. The relay characteristics of the on-off controller can also be modified by introducing a dead-zone or hysteresis, as shown in Figure 3.1.

On-off control can also be used to obtain information about the dynamics of a process. This is used in the auto-tuner for PID control discussed in Chapter 5.

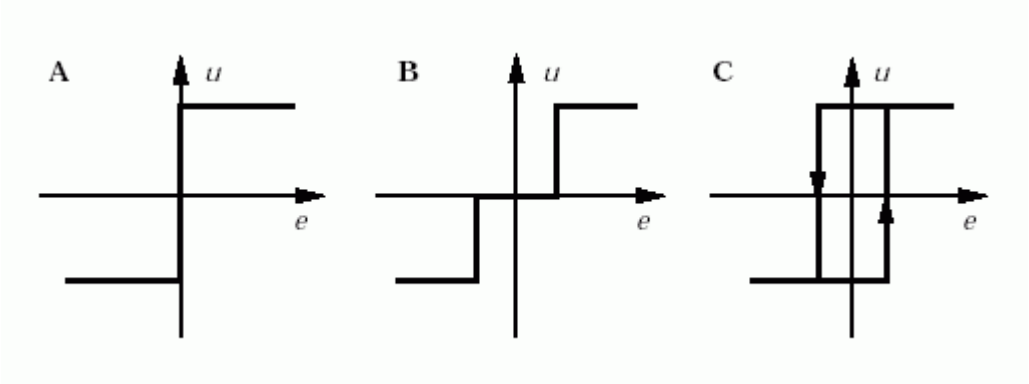


Figure 3.1 On-off control variations.

3.2. PID Control

The proportional-integral-derivative, or PID, controller is very useful. It is capable of solving a wide range of control problems. It is quoted that about 90% of all control problems can be solved by PID control. Moreover, a good majority these problems can be solved using only PI controllers because derivative action is not so common.

The reason why on-off control often gives rise to oscillations is that the system over-reacts, a small change in the error will make the manipulated variable change over the full range. This effect is avoided in proportional control where the characteristic of the controller is proportional to the control error for small errors. This can be achieved by making the control signal proportional to the error:

$$u = \begin{cases} u_{max} & e_{max} < e \\ u_{min} & e < e_{min} \\ k\,e & e_{min} \le e \textbf{ and } e \le e_{max} \end{cases} \qquad [3.3]$$

where k is the controller gain, e is defined in Equation [3.2],

$$e_{min} = \frac{u_{min}}{k} \qquad [3.4]$$

and

$$e_{max} = \frac{u_{max}}{k} \qquad [3.5]$$

.

The interval (e_{min}, e_{max}) is called the *proportional band* because the behaviour of the

controller is linear when the error is in this interval. The linear behavior of the controller is simply

$$u = (k(r - y) = ke) \quad [3.6]$$

Proportional control has has the drawback that the process variable often deviates from its reference value. This can be avoided by making the control action proportional to the integral of the error:

$$u(t) = k_i \int_0^t e(\tau)\, d\tau \quad [3.7]$$

This control form is called integral control and k_i is the integral gain. It follows from Equation [3.7] that if there is a steady state where the control signal and the error are constant, i.e. $u(t) = u_0$ and $e(t) = e_0$ respectively, then

$$u_0 = k_i e_0 t \quad [3.8]$$

This equation is a contradiction unless $e_0 = 0$ and we have thus proven that there is no steady state error if there is a steady state. Notice that this conclusion is true for any process and any controller that has integral action. The catch is that there may not necessarily be a steady state because the system may be oscillating. This property, which we call the *Magic of Integral Control*, is one of the reasons why PID controllers are so common.

An additional refinement of the controller is to provide it with an ability for anticipation. Future errors can be predicted by linear extrapolation. The predictor is

$$e(t + T_d) \approx e(t) + T_d\left(\frac{d}{dt}e(t)\right) \quad [3.9]$$

and provides an estimate of the error T_d time units ahead.

Combining proportional, integral, and derivative control we obtain a controller that can be expressed mathematically as follows:

$$u(t) = k\, e(t) + k_i \int_0^t e(\tau)\, d\tau + k_d\left(\frac{d}{dt}e(t)\right) \quad [3.10]$$

The control action is thus a sum of three terms referred to as proportional (P), integral (I) and derivative (D). As illustrated in Figure 3.2, the proportional term is based on the present error, the integral term depends on past errors, and the derivative term is a prediction of future errors. Advanced model-based controllers differ from the PID controller by using a

model of the process for prediction.

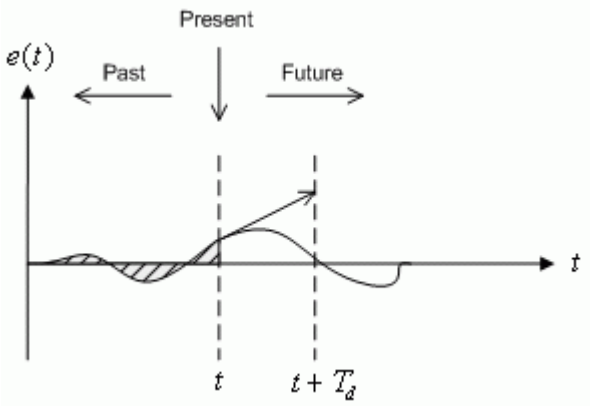


Figure 3.2 PID controller actions based on past, present, and future control errors.

The controller [3.10] can also be described by the transfer function

$$C(s) = k\,s + \frac{k_i}{s} + k_d\,s \qquad [3.11]$$

.

Further, it is common in industry to parametrize the PID control in [3.10] as follows:

$$u(t) = k\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)\,d\tau + T_d\left(\frac{d}{dt}e(t)\right)\right), \qquad [3.12]$$

where k is the proportional gain, T_i is the integral time, and T_d is the derivative time.

The PID controller described by [3.10] or [3.11] is the ideal PID controller. Attempts to implement these formulas do not lead to good controllers. Most measurement signals have noise and taking the differentiation of a noisy signal gives very large fluctuations. In addition, many actuators have limitations that can lead to integrator *windup*. In addition, the response to reference signals can be improved significantly by modifying the controller. These effects will now be discussed separately.

3.3. Peak Time and Overshoot

The *standard second-order* transfer function has the form

$$\frac{Y(s)}{R(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \qquad [3.13]$$

where ω_n is the natural undamped frequency and ζ is the damping ratio. The properties of its response depend on the values of the ω_n and ζ parameters. Consider when a second-order system, as shown in Equation [3.13], is subjected to a step of

$$R(s) = \frac{R_0}{s} \qquad [3.14]$$

with an amplitude of $R_0 = 1.5$. The obtained response is shown in Figure 3.3, below, where the red trace is the output response, y(t), and the blue trace is the reference step, r(t).

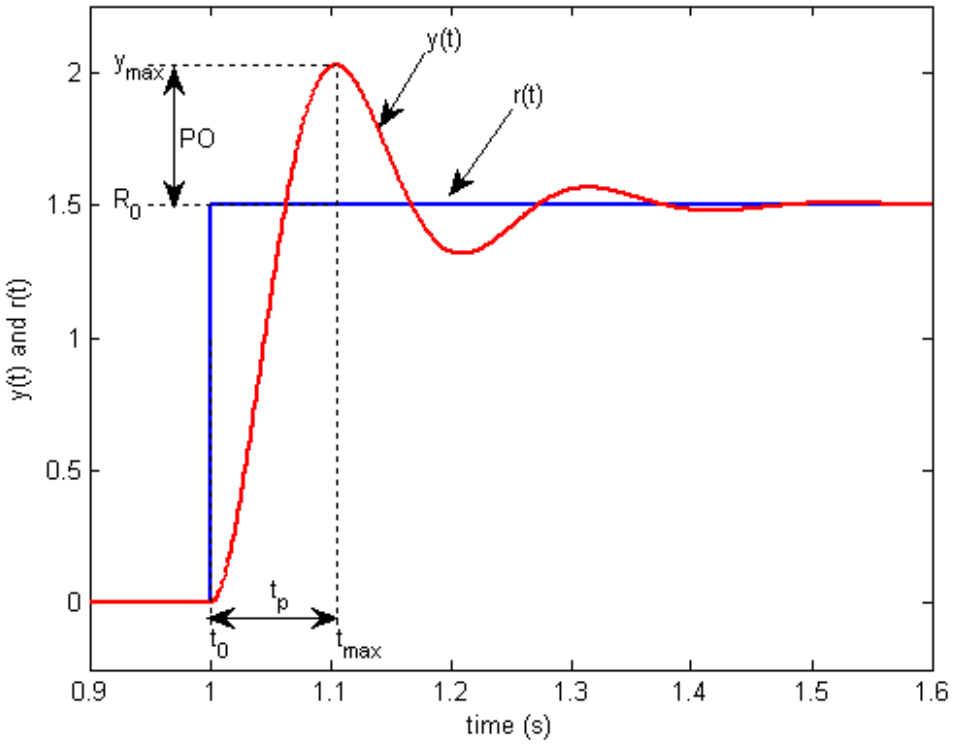


Figure 3.3: Standard second-order step response.

The maximum value of the response is denoted by the variable y_{max} and it occurs at a time t_{max}. For a response similar to Figure 3.3, the *percentage overshoot* is found using the equation

$$PO = \frac{100\,(y_{max} - R_0)}{R_0} \quad [3.15]$$

From the initial step time, t_0, the time it takes for the response to reach its maximum value is

$$t_p = t_{max} - t_0 \quad [3.16]$$

This is called the *peak time* of the system.

In a second-order system, the amount of overshoot depends solely on the damping ratio parameter and it can be calculated using the equation

$$PO = 100\,\mathbf{e}^{\left(-\frac{\pi\,\zeta}{\sqrt{1-\zeta^2}}\right)} \quad [3.17]$$

The peak time depends on both the damping ratio and natural frequency of the system and it can be derived that the relationship between them is

$$t_p = \frac{\pi}{\omega_n\sqrt{1-\zeta^2}} \quad [3.18]$$

Generally speaking then, the damping ratio affects the shape of the response while the natural frequency affects the speed of the response.

3.4. Filtering

A drawback with derivative action is that differentiation has very high gain for high frequency signals. This means that high frequency measurement noise will generate large variations of the control signal. The effect of measurement noise can be reduced by replacing the derivative action term k_d*s in [3.11] by

$$D_a = -\frac{k_d\,s}{1 + T_f s} \quad [3.19]$$

This can be interpreted as an ideal derivative that is filtered using a first-order low-pass filter system with the time constant T_f. For small s the transfer function is approximately k_d*s and for large values of s it is equal to k_d/T_f. Thus the approximation acts as a derivative for low-frequency signals and as a constant gain of k_d/T_f for the high-frequency signals. The filtering time is chosen as

$$\frac{k_d}{k\,N} = \frac{T_d}{N} \qquad [3.20]$$

where N in the range of 2 to 20. The transfer function of a PID controller with a filtered derivative is

$$\mathrm{C}(s) = k + \frac{k_i}{s} + \frac{k_d\,s}{1 + s\,T_f}. \qquad [3.21]$$

The high-frequency gain of the controller is $k*(1+N)$, which is a significant improvement over the ideal PID controller .

Instead of only filtering the derivative, it is also possible to use an ideal controller and filter the measured signal. The transfer function of such a controller using a second-order filter is then

$$\mathrm{C}(s) = \frac{k + \frac{k_i}{s} + k_d\,s}{1 + s\,T_f + \frac{1}{2}s^2\,T_f^2}. \qquad [3.22]$$

3.5. Set-point Weighting

The controllers described so far are called controllers with *error feedback* because the control action is based on the error, which is the difference between the reference r and the process output y. There are significant advantages to have the control action depend on the reference and the process output and not just on the difference between this signals. A simple way to do this is to replace the ideal PID controller in [3.10] with

$$\mathrm{u}(t) = k\,(b_{sp}\,\mathrm{r}(t) - \mathrm{y}(t)) + k_i \int_0^t \mathrm{r}(\tau) - \mathrm{y}(\tau)\,d\tau - k_d\left(\frac{d}{dt}\mathrm{y}(t)\right) \qquad [3.23]$$

where the parameter b_{sp} is called set-point weight or the reference weight. In this controller the proportional action only acts on a fraction b_{sp} of the reference and there is no derivative action on the set-point. Integral action continues to act on the full error to ensure the error

goes to zero in steady state. Closed-loop systems with the ideal PID controller [3.10] or the PID controller with set-point weighting in [3.23] respond to disturbances in the same way, but their response to reference signals are different.

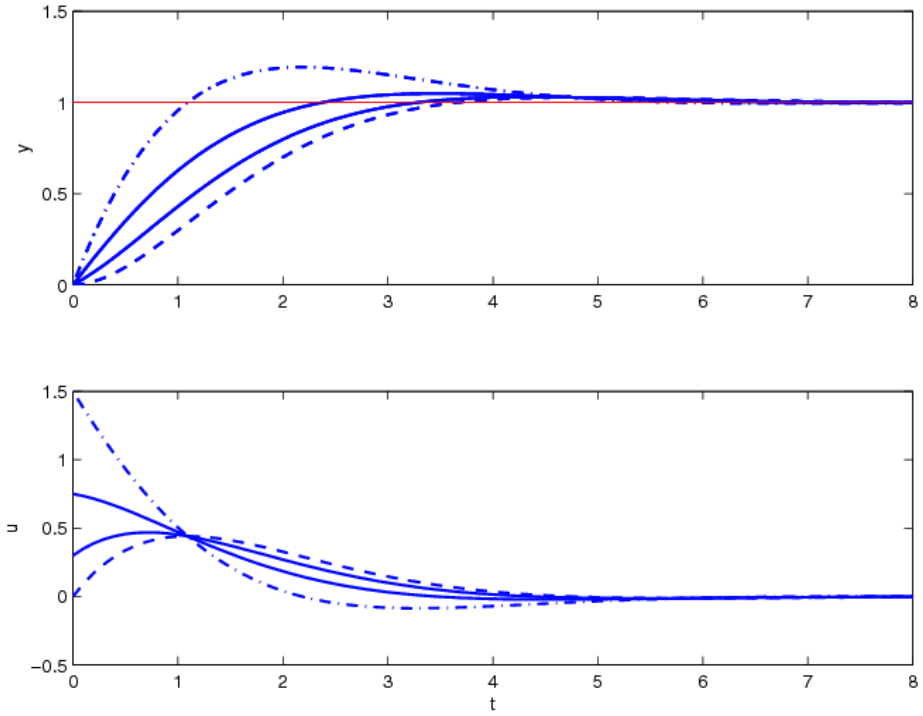


Figure 3.4 Set-point weighting effect on step reponse.

Figure 3.4 illustrates the effects of set-point weighting on the step response of the process,

$$P(s) = \frac{1}{s} \qquad [3.24]$$

with the controller gains $k_p = 1.5$ and $k_i = 1$. As shown in Figure 3.4, the overshoot for reference changes is smallest for b_{sp}=0, which is the case where the reference is only introduced in the integral term, and increases with increasing b_{sp}. The set-point weights in Figure 3.4 are: b_{sp}=0 on the dashed plot trajectory, b_{sp}=0.2 and b_{sp}=0.5 on the two solid lines, and b_{sp}=1 on the dash-dot response. The set-point parameter is typically in the range of 0 to 1.

3.6. Integral Windup

Many aspects of a control system can be understood from linear models. There are, however, some nonlinear phenomena that are unavoidable. There are typically limitations in the

actuators: a motor has limited speed, a valve cannot be more than fully opened or fully closed, etc. For a control system with a wide range of operating conditions, it may happen that the control variable reaches the actuator limits. When this happens the feedback loop is broken and the system runs in open loop. The actuator remains at its limit independently of the process output as long as the actuator remains saturated. If the integral term is large, the error must change sign for a long period before the integrator winds down. The consequence is that there may be large transients. This phenomena is called *integrator windup* and it appears in all systems with actuator saturation and controllers having integral action.

The *windup* effect is illustrated in Figure 3.5 by the dashed red line. The initial reference signal is so large that the actuator saturates at the high limit. The integral term increases initially because the error is positive. The output reaches the reference at around time t=4. However, the integrator has built-up so much energy that the actuator remains saturated. This causes the process output to keep increasing past the reference. The large integrator output that is causing the saturation will only decrease when the error has been negative for a sufficiently long time. When the time reaches t=6, the control signal finally begins to decrease while the process output reaches its largest value. The controller saturates the actuator at the lower level and the phenomena is repeated. Eventually the output comes close to the reference and the actuator does not saturate. The system then behaves linearly and settles quickly. The *windup* effect on the process outpu is therefore a large overshoot and a damped oscillation where the control signal flips from one extreme to the other as in relay oscillations.

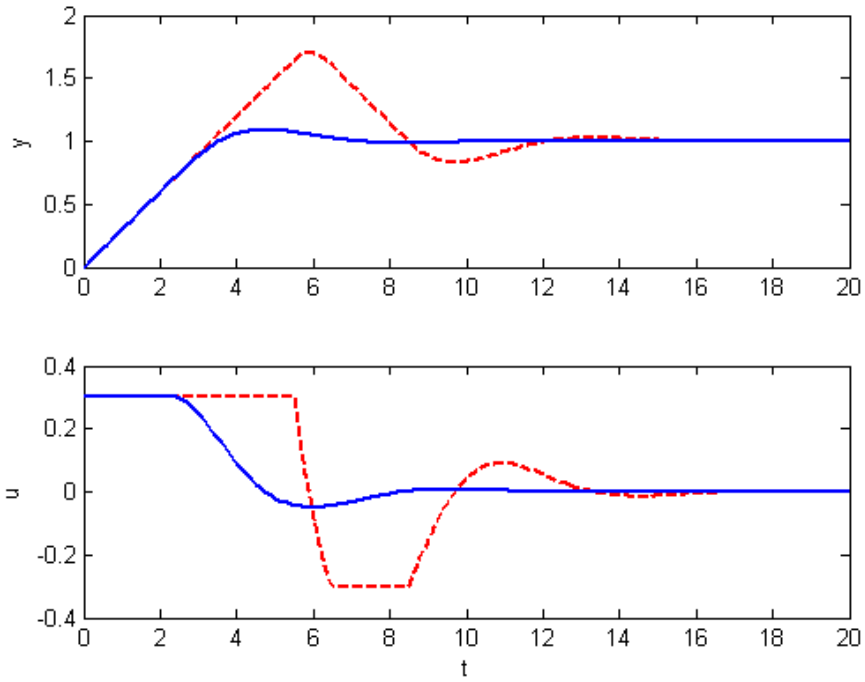


Figure 3.5 Illustration of integrator windup. The dashed curves shows the results for a controller without windup protection. The full curves show the results for a controller with windup protection. Top curves show output y and reference r. Bottom curves show control signal u.

There are many ways to avoid windup, one method is illustrated in Figure 3.6. The system has an extra feedback path that that sets the integrator to a value so that the controller output is always close to the saturation limit. This is accomplished by measuring the difference e_s between the actual actuator output and feeding this signal to the integrator through gain $1/T_r$.

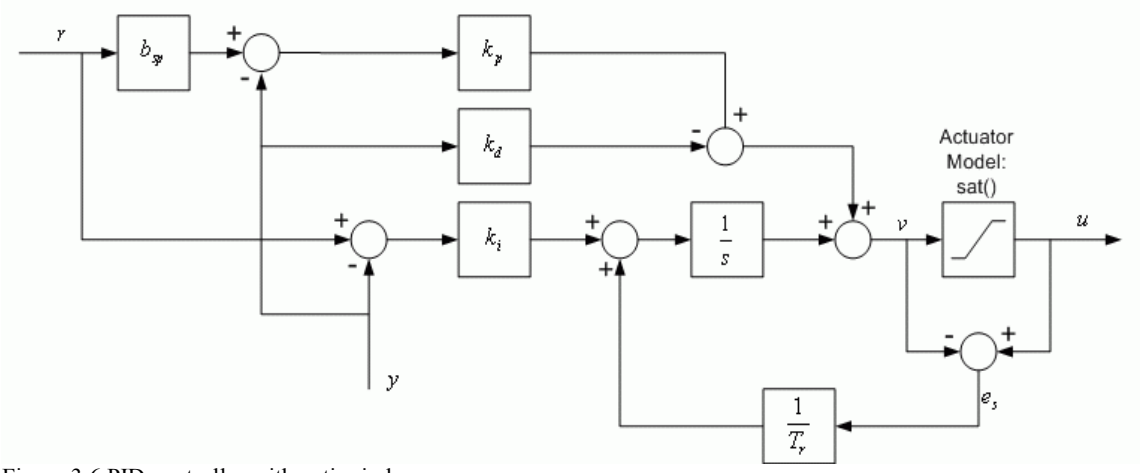


Figure 3.6 PID controller with anti-windup.

The signal e_s is zero when there is no saturation and the extra feedback loop has no effect on the system. When the actuator saturates, the signal e_s is different from zero. The normal feedback path around the process is broken because the process input remains constant. The feedback around the integrator will act to drive e_s to zero. This implies that controller output is kept close to the saturation limit and integral windup is avoided.

The rate at which the controller output is reset is governed by the feedback gain, $1/T_r$, where the *tracking time constant*, T_r,determines how quickly the integral is reset. A long time constant gives a slow reset and a short time constant a short reset time. The tracking time constant cannot be too short because measurement noise can cause an undesirable reset. A reasonable compromise is to choose T_r as a fraction of the integrator reset time T_i for proportional control and

$$T_r = \sqrt{T_i T_d} \qquad [3.25]$$

for PID control. The integrator reset time T_i and the derivative reset time T_d are defined in the parametrized PID controller shown in [3.12].

The solid curves in Figure 3.5 illustrates the effect of anti-windup. The output of the integrator is quickly reset to a value such that the controller output is at the saturation limit, and the integral has a negative value during the initial phase when the actuator is saturated. Observe the dramatic improvement of using windup protection over the ordinary PI controller that is represented by the dashed lines in Figure 3.5.

4. LabVIEW

LabVIEW is a is a graphical programming environment that was originally developed by National Instruments to implement virtual instruments. The basic idea was to describe how data flows from a sensor to a display and to quickly generate nice looking displays. LabVIEW which first appeared in 1986 has been developed continuously, currently substantial efforts are made to make extensions to control applications. A key feature is that programming is done graphically by cut-and-paste and that the user interface is an integral part of the system. LabVIEW programs, called VI's (Virtual Instruments), consists of two parts called the front panel and the block diagram. The front panel is the graphical user interface which has indicators, dials and knobs. The block diagram describes how how data is connected to sensors and actuators and how data flows from the sensors to the actuators and the front panel. The front panel and the block diagram are coupled, if an instrument is pasted on the block diagram it also appears on the block diagram. The computations are described by a graph which has nodes or vertices's connected by edges or arcs. The nodes represents computations and the arcs represents data that flows between the computations. There are many different types of nodes for simple tasks such as adding two signals or more complicated tasks like making an FFT computation or solving a differential equation. The language is typically extended by adding different types of nodes. Programming of the block diagram is also done graphically. There is a data-flow language G, hidden from the used who only has access through the graphical user interfaces. There is semantics to ensure that data has represented by the arcs has the correct type. Conceptually a LabVIEW program can be thought of as a way of describing how data flows from sensors via computations to actuators, which is a natural concept for control. In addition LabVIEW provides the tools for building the user interface.

Good information about LabVIEW is found on the site
http://www.ni.com/labview/power\#3

The article
http://www.ni.com/devzone/lvzone/view_archived1.htm
written by LabVIEW's creator Jeff Kodosky gives a short insightful presentation of the

philosophy behind LabVIEW.

4.1. PID Controllers in LabVIEW

There are several ways to implement controllers in LabVIEW. A simple method is to use a simulation node, which permits a high-level description in terms of block diagrams and transfer functions. Such a description can be entered graphically by cut-and-paste using a simulation node. The graph of a PID controller with set-point weighting, a filtering of the derivative, and windup protection is shown in Figure 4.1. Notice the strong similarity with the block diagram used in textbooks.

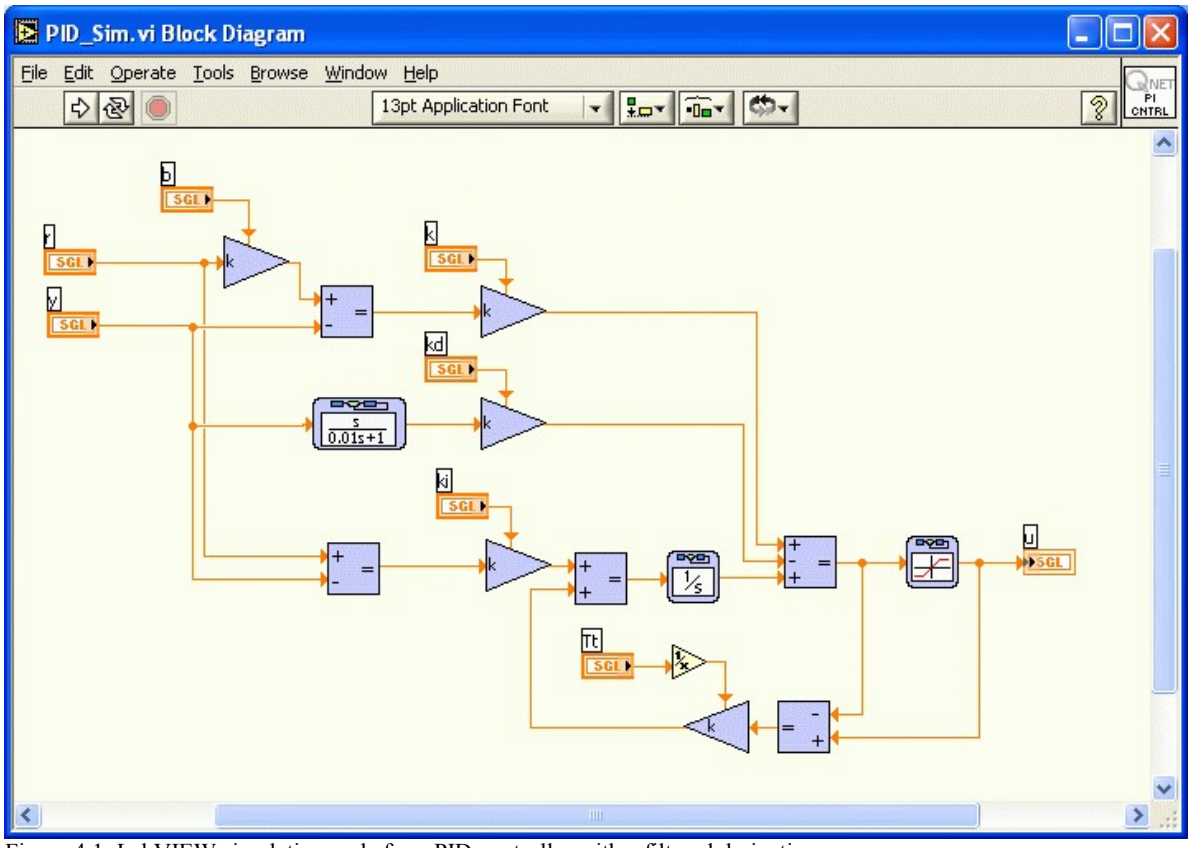


Figure 4.1: LabVIEW simulation node for a PID controller with a filtered derivative.

The representation in Figure 4.1 corresponds to a nonlinear differential equation. A digital computer can do algebraic operations but it cannot integrate differential equations. It is therefore necessary to go through several steps to obtain an approximation of the differential equations that can be handled by the computer. This is done automatically in LabVIEW when using a simulation node. When implementing a continuous-time control

law, such as a PID controller, on a digital computer it is necessary to approximate the derivatives and the integral that appear in the control law. The nonlinear differential equation is then approximated by a difference equation that can be implemented in a formula loop. There are several ways to do the approximation.

The proportional term is

$$P(t) = k_p (b\, r(t) - y(t)) \quad [4.1]$$

This term is implemented simply by replacing the continuous variables with their sampled versions. Given the analog-to-digital converters receive values of reference r and process output y at sampling time t_k, the proportional term of the PID controller is given by

$$P(t_k) = k_p (b\, r(t_k) - y(t_k)) \quad [4.2]$$

As shown, there are no approximations required for the proportional action.

The integral term is approximated by

$$I(t_{k+1}) = I(t_k) + k_i\, h\, e(t_k) \quad [4.3]$$

where

$$h = t_{k+1} - t_k \quad [4.4]$$

is the sampling period and

$$e(t_k) = r(t_k) - y(t_k) \quad [4.5]$$

is the error in discrete time.

The derivative term with a first-order filter is represented by the transfer function

$$D(s) = -\frac{k_d\, s\, Y(s)}{1 + s\, T_f} \quad [4.6]$$

The derivative term is thus given by the differential equation

$$T_f\left(\frac{\partial}{\partial t} D\right) + D = -k_d\left(\frac{\partial}{\partial t} y\right) \quad [4.7]$$

Notice that the derivative only acts on the process output. This equation can be approximated in the same way as the integral term. If the derivative in Equation [4.7] is approximated by a backward difference, the following equation is obtained:

$$\frac{T_f(\mathrm{D}(t_k) - \mathrm{D}(t_{k-1}))}{h} + \mathrm{D}(t_k) = -\frac{k_d(\mathrm{y}(t_k) - \mathrm{y}(t_{k-1}))}{h} \quad [4.8]$$

Solving for the $D(t_k)$, the expression becomes:

$$\mathrm{D}(t_k) = \frac{T_f \mathrm{D}(t_{k-1})}{T_f + h} - \frac{k_d(\mathrm{y}(t_k) - \mathrm{y}(t_{k-1}))}{T_f + h} \quad [4.9]$$

If the filter time-constant $T_f = 0$, the derivative term reduces to a simple difference of the output. When $T_f > 0$, the difference will be filtered. Observe that $T_f/(T_f + h)$ in Equation [4.9] is always in the range of 0 and 1. This implies the approximation is always stable.

Introducing the state

$$\mathrm{x}(t) = \mathrm{D}(t) + \frac{k_d \mathrm{y}(t)}{T_f + h} \quad [4.10]$$

,

and substituting Equation [4.9] in the discretized version of [4.10] gives

$$\mathrm{x}(t_k) = \frac{T_f \mathrm{x}(t_{k-1})}{T_f + h} + \frac{k_d h \mathrm{y}(t_{k-1})}{(T_f + h)^2} \quad [4.11]$$

.

Summarizing, the PID controller can be represented by the difference equations:

$$\begin{aligned} \mathrm{u}(t_k) &= b k_p r - \left(k_p + \frac{k_d}{T_f + h}\right)\mathrm{y}(t_k) + I(t_k) + \mathrm{x}(t_k) \\ I(t_k) &= I + k_i h(\mathrm{r}(t_{k-1}) - \mathrm{y}(t_{k-1})) \\ \mathrm{x}(t_k) &= \frac{T_f \mathrm{x}(t_{k-1})}{T_f + h} + \frac{k_d h \mathrm{y}(t_{k-1})}{(T_f + h)^2} \end{aligned} \quad [4.12]$$

The PID controller has two states: I and x, and seven parameters: proportional gain k_p, integral gain k_i, derivative gain k_d, set point weight b, filter time constant T_f, tracking time constant T_t, and sampling period h.

The difference equations in [4.12] can be implemented using a formula node as illustrated in the VI shown in Figure 4.2. Timing can be provided by including the formula node in a timed block. We have also added anti-windup based on a saturation model. Figure 4.2

also shows how the computations can be made faster by pre-computing some parameters. These calculations are only required when parameters are changed. Notice that only 6 multiplications and 7 additions are required in each iteration.

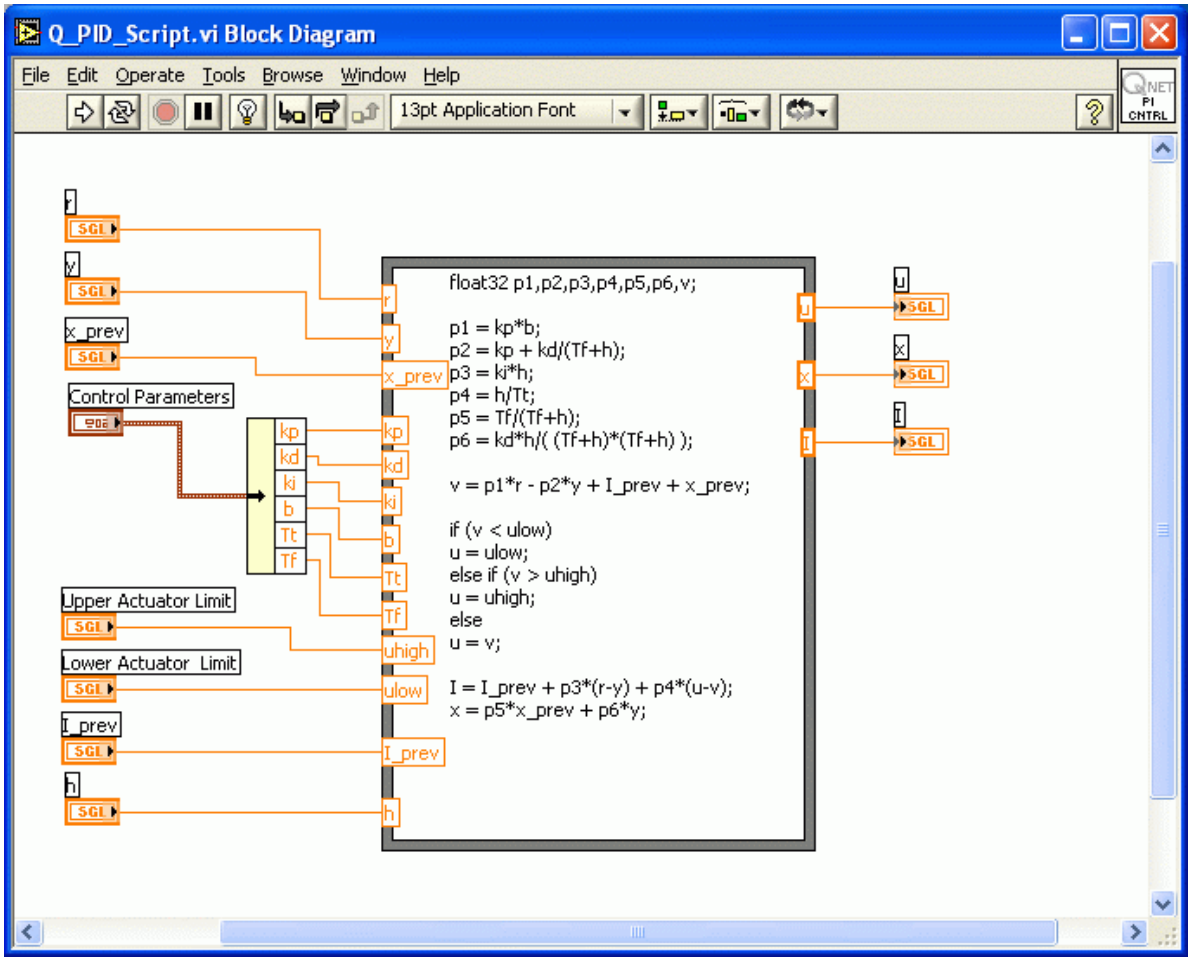


Figure 4.2: Formula node implementation of PID controller.

There are many other ways to make the approximations. Typically there is little difference in the performance of the different approximations if the sampling rate is faster then the dynamics of the system. However, there may be performance differences in extreme situations. The PID controllers discussed have a constant gain at high frequencies but higher-order filtering should be considered in systems with considerable sensor noise.

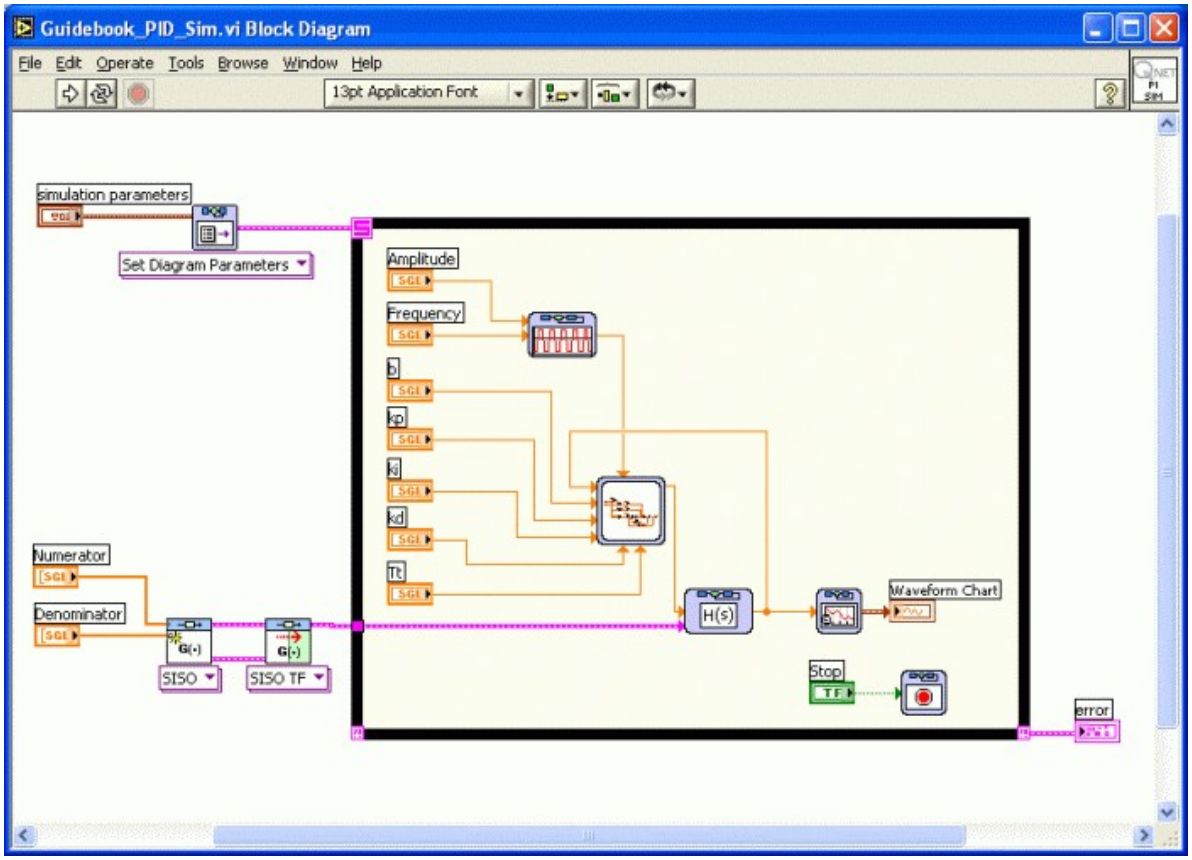


Figure 4.3: LabVIEW VI for simulation of a PID controller.

LabVIEW can also be used for simulation. Figure 4.1 shows a simulation node for a PID controller. The complete simulation is obtained when adding the node to a time-loop containing a simulation of the process along with a signal generator to generate the setpoint. Figure 4.3 shows a complete simulation of a PID controller.

5. Process Control

The QNET-012: heating and ventilation trainer (HVACT) is shown in Figure 5.1. The system consists of a plexiglass duct, with a heater in one end and a blower in the other end. The heater is a halogen lamp and the blower is a variable-speed fan. There is a thermistor sensor placed inside the duct to measure the temperature of the chamber and another thermistor sensor outside the chamber to measure the room temperature.

Figure 5.1: The QNET heating and ventilation trainer (HVACT).

The temperature measured at the thermistor inside the chamber is to be controlled using the

heater voltage while the fan is ran at a constant speed. Heat is transferred to the thermistor by radiation from the heater and by convection from the air stream. Radiative heat transfer is highly nonlinear and it is therefore difficult to model the system by first principles. As a result, empirical tuning will be used to control the system. This heat transfer plant is very similar to the systems that are used to control wafer temperature in semiconductor manufacturing.

There are two experiments: on-off control and PI control. The experiments can be performed independently.

5.1. On-Off Control

On-off control or relay feedback is one of the simplest control strategies. The heater is switched on when the temperature is lower than the desired value, and the heater is switched off when the temperature is higher than the desired value. To avoid rapid switches it is common to introduce a hysteresis in the relay switch. A block diagram of a system with relay feedback is shown in Figure 5.2.

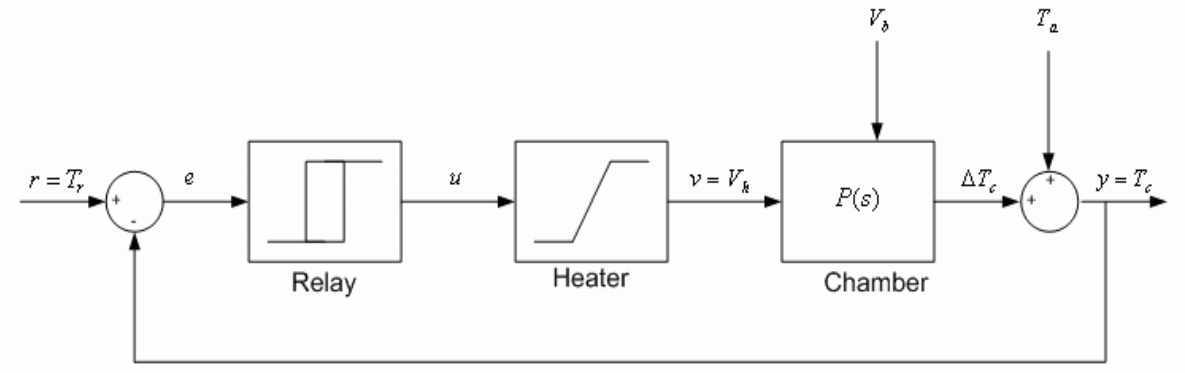


Figure 5.2 Block diagram of the heater system with relay feedback.

The error, variable *e* in Figure 5.2, is the difference between the reference temperature, T_r, and the actual chamber temperature, T_c. The on-off controller is implemented using a relay switch with hysteresis, as shown in Figure 5.3. The heater actuator is represented by a saturation block and the chamber plant is represented by the transfer function *P(s)*.

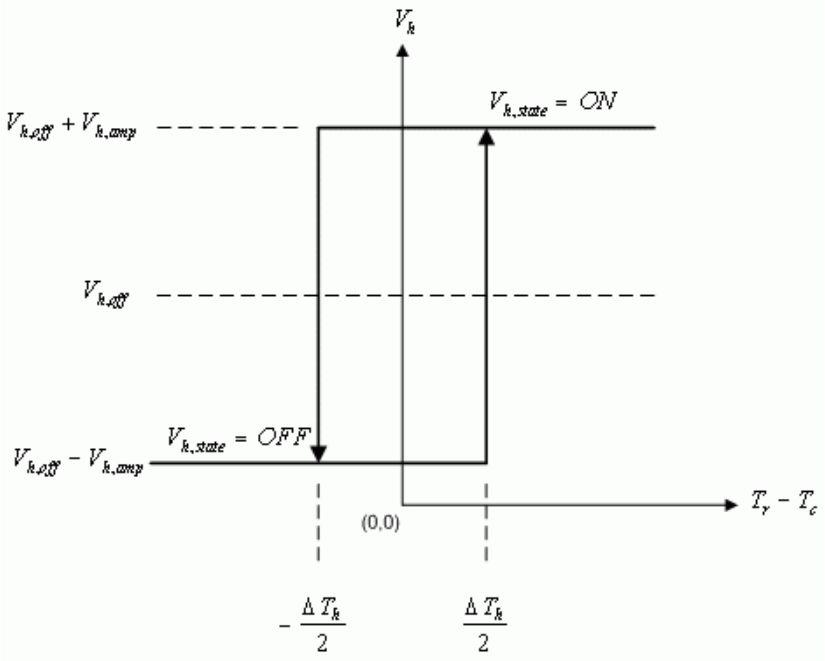


Figure 5.3 Input and output relation for an on-off controller with hysteresis

The hysteresis width, ΔT_h in Figure 5.3, has to be chosen such that a large measurement noise does not generate any unintentional switches. As depicted in Figure 5.3, the output control signal voltage of the on-off controller can be adjusted using a mean or offset, $V_{h,off}$, and an amplitude, $V_{h,amp}$.

In the experiment, the behavior of the heater system will be investigated for different values of controller parameters. More specifically, the control signal and the measured temperature will be observed.

The LabVIEW virtual instrument for the on-off control is shown in Figure 5.4.

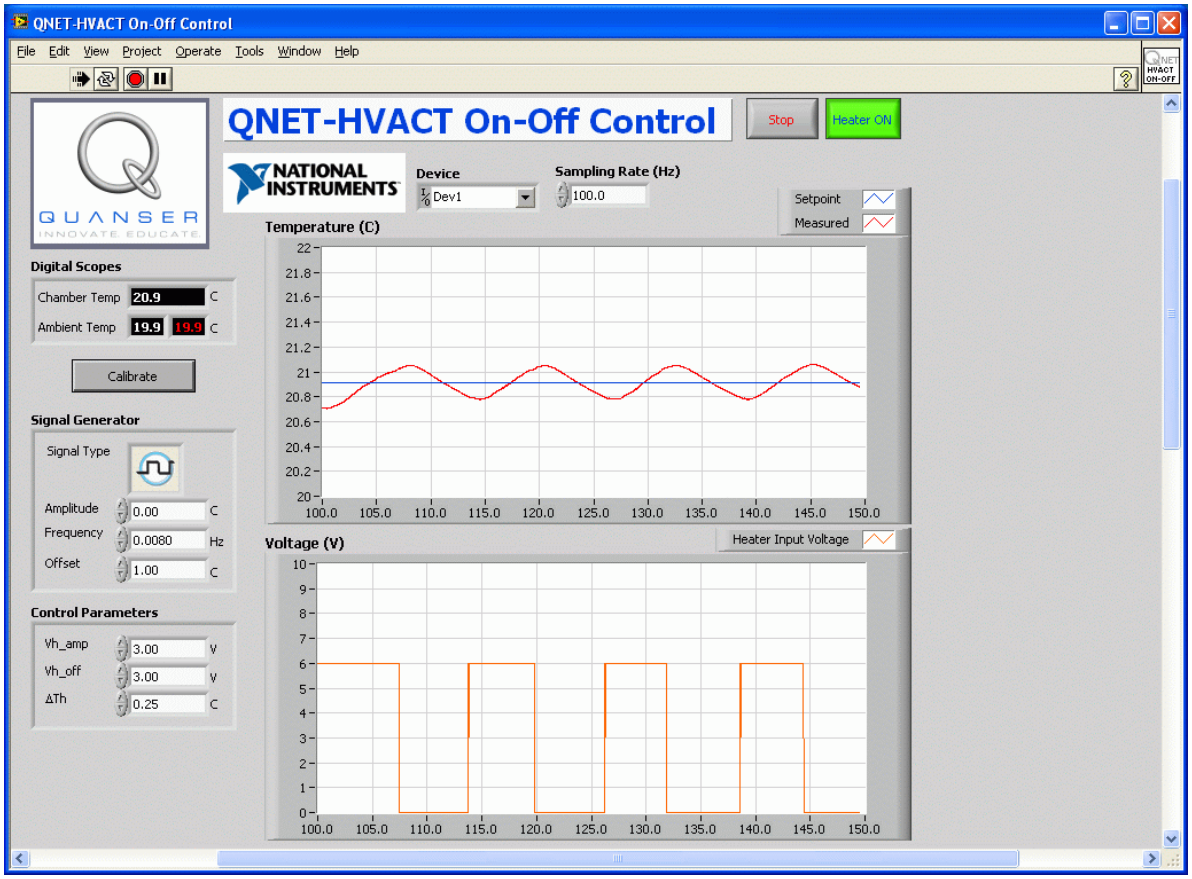


Figure 5.4 Virtual instrument for on-off heater control.

The on-off control input and the measured temperature output from the experiment shown in Figure 5.4 have an interesting property that makes it possible to find a simple model for the process. The temperature response is a ramp due a voltage step therefore the temperature is the integral of the voltage. Under the conditions shown in Figure 5.4 the process can be modeled by the simple transfer function

$$P(s) = \frac{K_v}{s}, \qquad [5.1]$$

where the parameter K_v is the slope of the ramp.

See *Wikipedia* for more information on relay, hysteresis, mathematical model, transfer function, and LTI system theory.

5.2. PI Control

The oscillations that occur with on-off control can be avoided by using a linear proportional and integrating controller. To design such a controller analytically a simple model representing the actual plant is needed. Since the conditions shown in Figure 5.4 are representative for what happens when the temperature is controlled, transfer function [5.1] can be used for the model-based approach to find the controller. The block diagram of the closed loop system is shown in Figure 5.5.

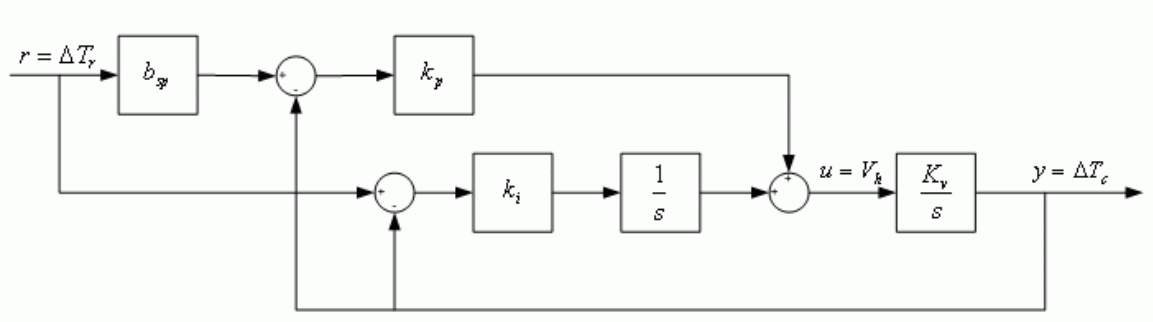


Figure 5.5 Block diagram of heater PI closed-loop system.

The process transfer function is the transfer function in Equation [5.1] and the input-output relation for a PI controller with set-point weighting is

$$U(s) = k_p (b_{sp} R(s) - Y(s)) + \frac{k_i (R(s) - Y(s))}{s} \quad [5.2]$$

The closed loop transfer function from the relative temperature reference, $\Delta T_r = T_r - T_a$, to the output temperature measured relative to the ambient temperature, $\Delta T_c = T_c - T_a$, is

$$G_{\Delta T_c, \Delta T_r}(s) = \frac{K_v (k_p s b_{sp} + k_i)}{s^2 + K_v k_p s + K_v k_i} \quad [5.3]$$

The closed-loop system has the characteristic polynomial

$$s^2 + K_v k_p s + K_v k_i \quad [5.4]$$

and the desired closed loop characteristic polynomial is

$$s^2 + 2\zeta \omega_0 s + \omega_0^2, \quad [5.5]$$

where ω_0 is the undamped closed loop frequency and ζ is the damping ratio. The

characteristics equation in [5.4] matches equation [5.5] with the proportional control parameter

$$k_p = \frac{2\zeta\omega_0}{K_v} \qquad [5.6]$$

and the integral control gain

$$k_i = \frac{\omega_0^2}{K_v} \qquad [5.7]$$

.

Large values of ω_0 give large values of controller gain. This implies noise will create large variations in the control signal. The set-point weight parameter b_{sp} can be used to adjust the overshoot of the response.

The sensor signal is noisy and it is therefore necessary to filter the measured signal. A simple first-order filter has the transfer function

$$T_c = \frac{T_{c,\,meas}}{T_f s + 1} \qquad [5.8]$$

,

where $T_{c,meas}$ is the measured temperature from the thermistor and T_f is the transfer function time constant. Increasing T_f decreases the cutoff frequency and minimizes noise in the signal at the expense of changing the shape of the signal.

Temperature control typically admits high controller gains. A consequence of this is that the controller output may saturate and result in integrator windup. The heater is therefore useful to illustrate the usefulness of integrator feedback.

The LabVIEW virtual instrument that implements the heater PI control is shown in Figure 5.6. The control parameters k_p, k_i, b_{sp}, the anti-windup tracking time constant T_r, and the filter time constant T_f can all be adjusted.

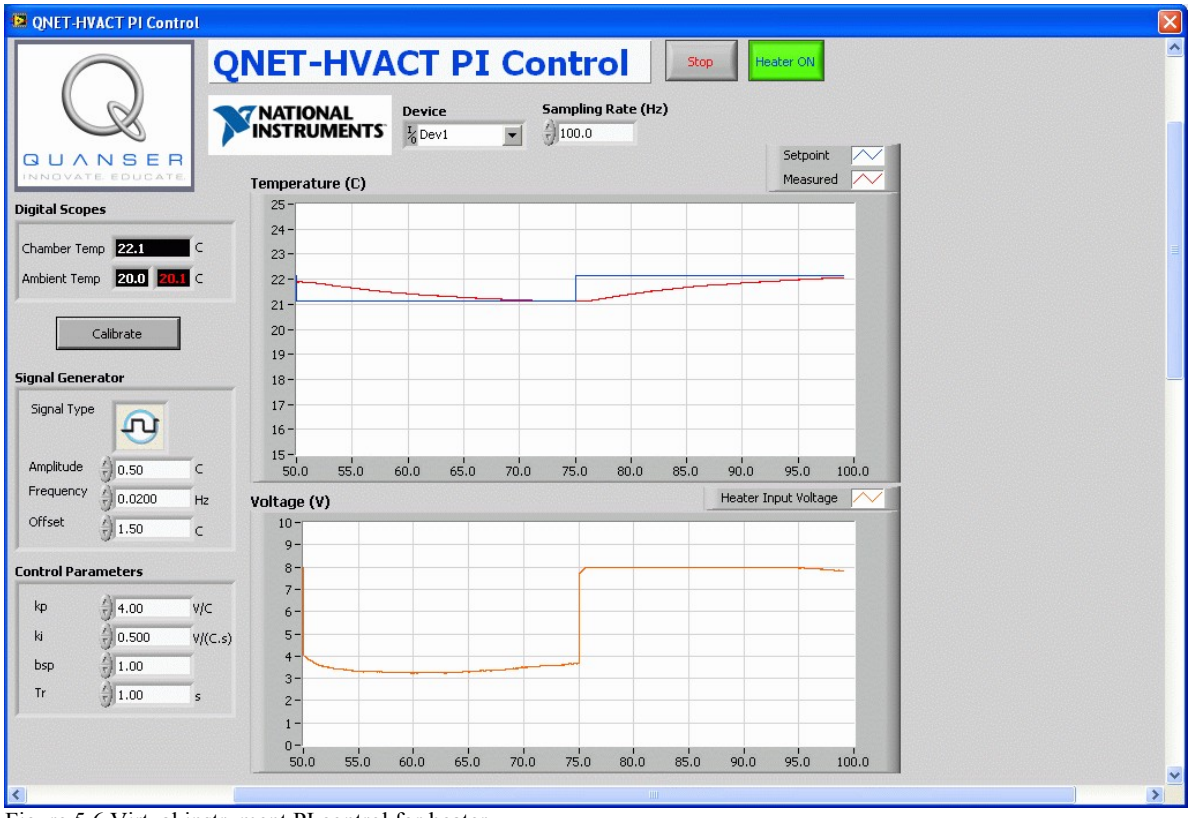


Figure 5.6 Virtual instrument PI control for heater.

See *Wikipedia* for more information on process control, control theory and PID.

6. Motion Control

The DC Motor Control Trainer is shown in Figure 6.1. The system consists of a direct-current motor with an encoder and an inertia wheel on the motor shaft. The motor is driven using a pulse-width modulated (PWM) power amplifier. The power to the amplifier is delivered using the QNET power cable from a wall transformer and the encoder is powered by the ELVIS unit. Signals to and from the system are available on a header and on standard connectors for control via a Data Acquisition (DAQ) card. The control variable is the voltage to the drive amplifier of the system and the output is either the wheel speed or the angle of the wheel. Disturbances can be introduced manually by manipulating the wheel or digitally through LabVIEW.

Figure 6.1: The QNET DC motor control trainer (DCMCT).

There are three experiments: modeling, speed control, and position control. The experiments can be performed independently.

6.1. Modeling

The motor trainer is very well suited to physical modeling. The moment of inertia of the wheel can be determined by measuring its dimensions and weighting the wheel. The equations of motion for the motor are determined by mechanics and electromagnetics. The key parameters are the motor constant and the electrical resistance of the motor armature. They can be determined by simple experiments. The resulting model is a transfer function from voltage to motor speed:

$$G_{\omega,V}(s) = \frac{K}{\tau s + 1} \qquad [6.1]$$

where K is the steady-state gain and τ is the time constant.

6.1.1. Bumptest Method

The bumptest is a simple test based on a step response for a stable system. It is carried out in the following way. A constant input is chosen. A stable system will then reach an equilibrium. The input is then changed rapidly to a new level and the output is recorded.

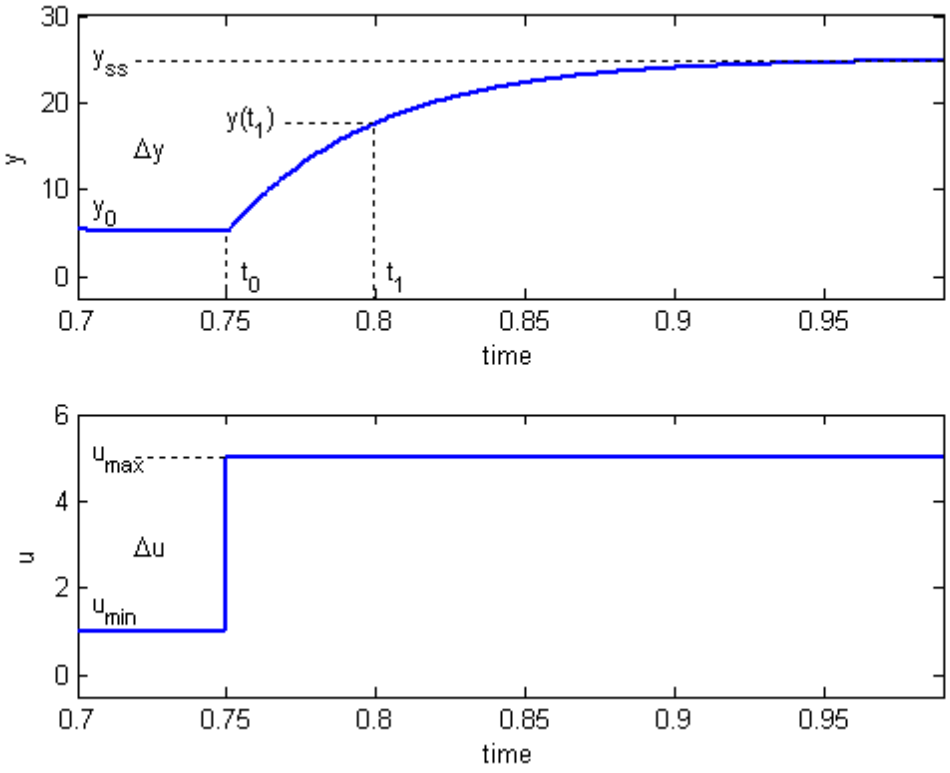


Figure 6.2: Input and output signal used in the bumptest method.

The step response shown in Figure 6.2 is generated using the transfer function

$$\frac{Y(s)}{U(s)} = \frac{K}{\tau s + 1} \qquad [6.2]$$

with the parameters

$$K = 5.0 \left[\frac{rad}{s\,V} \right] \qquad [6.3]$$

and

$$\tau = 0.05\,[s] \qquad [6.4]$$

The input signal, *u*, is a step that begins at time t_0. The input signal has a minimum value of u_{min} and a maximum value of u_{max}. The resulting output signal is initially at y_0. Once the step is engaged, the output eventually settles to its steady-state value y_{ss}. From the output and input signals, the steady-state gain is

$$K = \frac{\Delta y}{\Delta u} \quad [6.5]$$

where

$$\Delta y = y_{ss} - y_0 \quad [6.6]$$

and

$$\Delta u = u_{max} - u_{min} \quad [6.7]$$

In order to find the model time constant, τ, the output signal at $y(t_1)$ must be measured. It is defined

$$y(t_1) = 0.632\, y_{ss} + y_0 \quad [6.8]$$

and the time is

$$t_1 = t_0 + \tau \quad [6.9]$$

From this, the model time constant is

$$\tau = t_1 - t_0 \quad [6.10]$$

6.1.2. Model Validation

When the modeling is complete it can be validated by running the model and the actual process in open-loop. That is, the open-loop voltage is fed to both the model and the actual device such that both the simulated and measured response can be viewed on the same scope. The model can then be adjusted to fit the measured motor speed by fine-tuning the modeling parameters. The LabVIEW virtual instrument for modeling is shown in Figure 6.3.

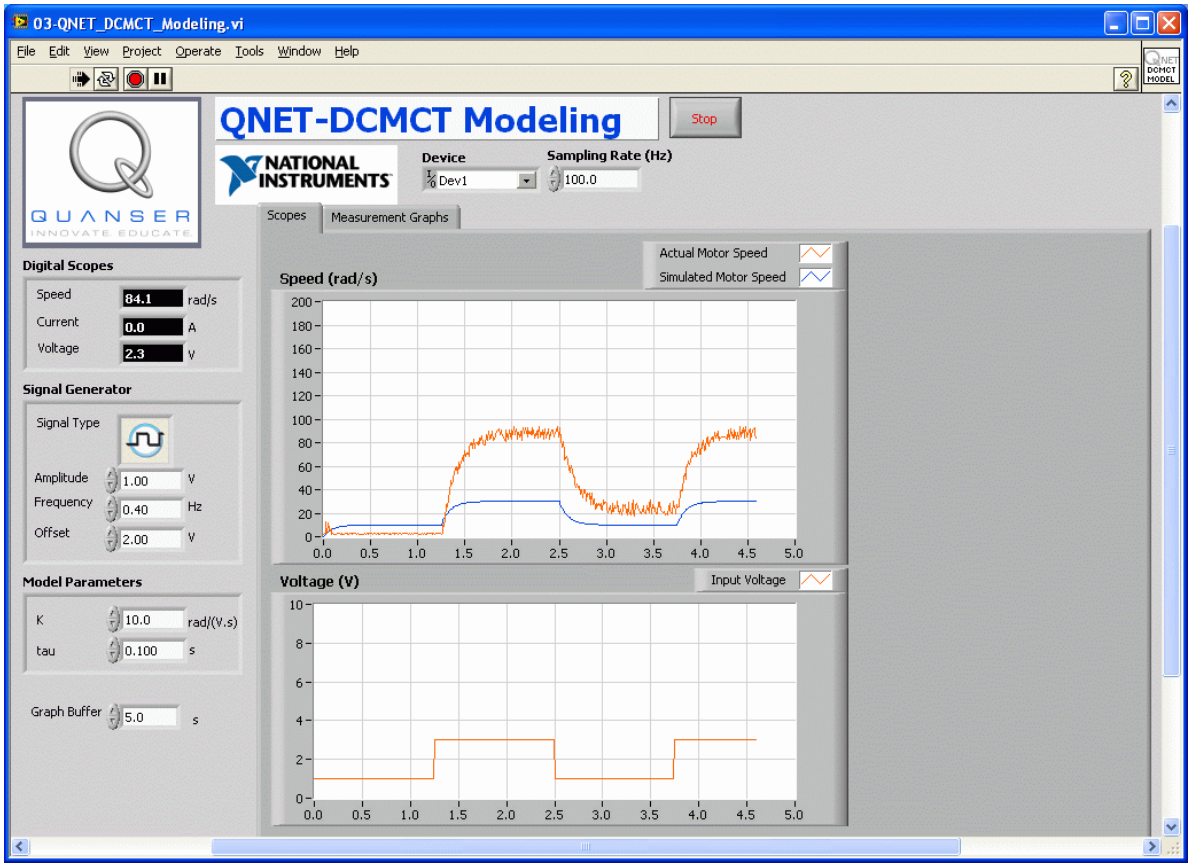


Figure 6.3 LabVIEW VI for modeling QNET DC motor.

See *Wikipedia* for more information on electric motor , mathematical model, transfer function, and LTI system theory.

6.2. Speed Control

The speed of the DC motor is controlled using a proportional-integral control system. The block diagram of the closed-loop system is shown in Figure 6.4.

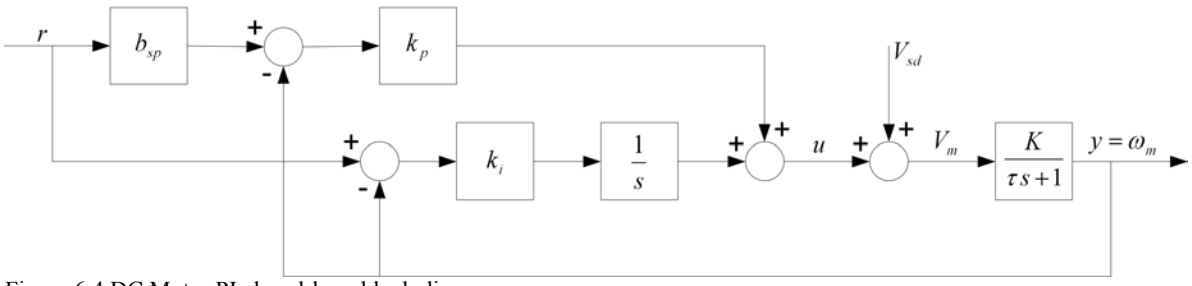


Figure 6.4 DC Motor PI closed-loop block diagram.

The transfer function representing the DC motor speed-voltage relation in Equation [6.1] is used to design the PI controller. The input-output relation in the time-domain for a PI controller with set-point weighting is

$$u = k_p (b_{sp} r - y) + \frac{k_i (r - y)}{s} \quad [6.11]$$

where k_p is the proportional gain, k_i is the integral gain, and b_{sp} is the set-point weight. The closed loop transfer function from the speed reference, r, to the angular motor speed output, ω_m, is

$$G_{\omega,r}(s) = \frac{K(k_p s b_{sp} + k_i)}{s^2 \tau + (K k_p + 1) s + K k_i} \quad [6.12]$$

The standard desired closed loop characteristic polynomial is

$$s^2 + 2\zeta \omega_0 s + \omega_0^2 \quad [6.13]$$

where ω_0 is the undamped closed loop frequency and ζ is the damping ratio. The characteristic equation in [6.12], i.e. the denominator of the transfer function, can match the desired characteristic equation in [6.13] with the following gains:

$$k_p = \frac{-1 + 2\zeta \omega_0 \tau}{K} \quad [6.14]$$

and

$$k_i = \frac{\omega_0^2 \tau}{K} \quad [6.15]$$

Large values of ω_0 give large values of controller gain. The damping ratio, ζ, and the set-point weight parameter, b_{sp}, can be used to adjust the speed and overshoot of the response to reference values.

There is no tachometer sensor present on the QNET DC motor system that measures the speed. Instead the amplifier board has circuitry that computes the derivative of the encoder signal, i.e. a digital tachometer. However to minimize the noise of the measured signal and increase the overall robustness of the system, the first-order low-pass filter

$$\omega_m = \frac{\omega_{meas}}{T_f s + 1}, \qquad [6.16]$$

is used. Parameter T_f is the filter time constant that determines the cutoff frequency and ω_{meas} is the measured speed signal.

Tracking a square wave with various PD gains are discussed in the laboratory as well as the effects of set-point weighting and integrator windup. The steady-state errors due to triangular references are also assessed. The virtual instrument for speed control is shown in Figure 6.5.

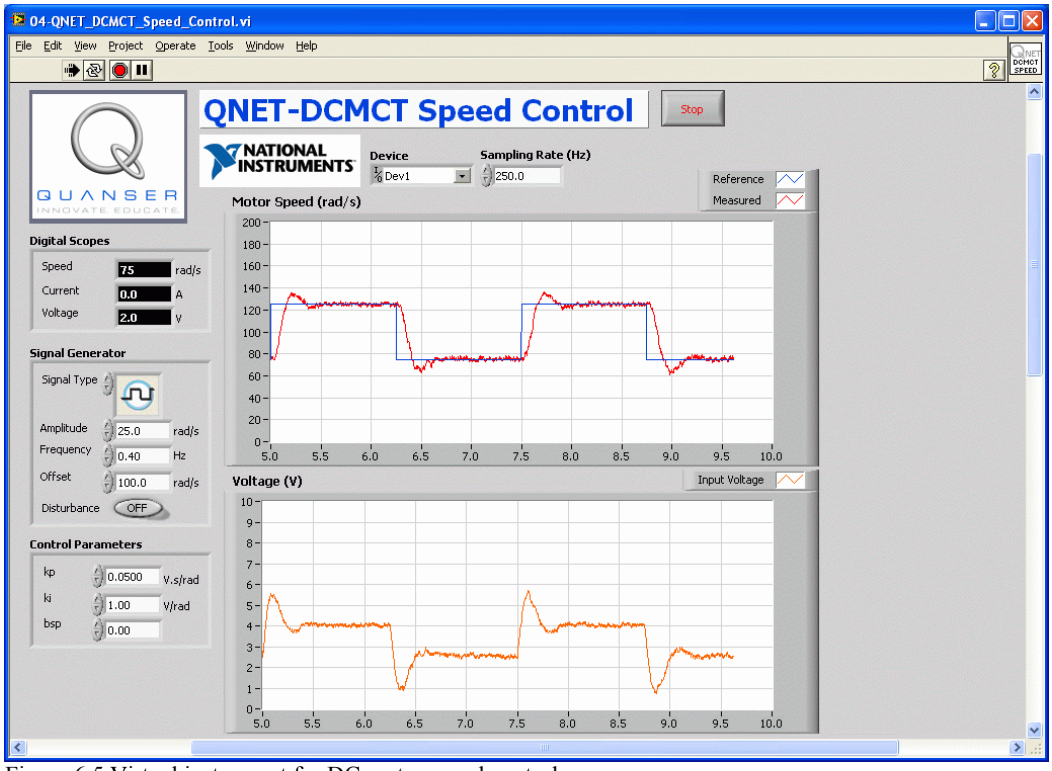


Figure 6.5 Virtual instrument for DC motor speed control.

6.3. Position Control

Control of motor position is a natural way to introduce the benefits of derivative action. In this experiment a proportional-integral-derivative controller is designed according to specifications. The closed-loop PID control block diagram is shown in Figure 6.6.

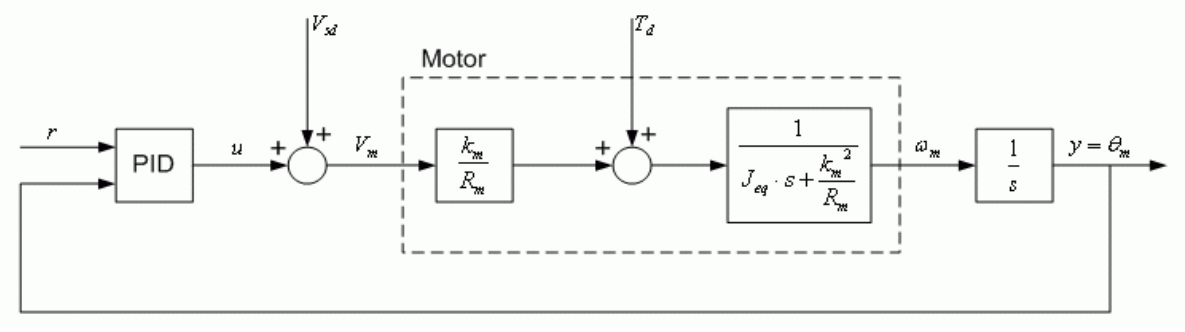


Figure 6.6 Block diagram of closed-loop motor position system using PID.

The two-degree of freedom PID transfer function inside the PID block in Figure 6.6 is

$$u(t) = k_p\,(b_{sp}\,r(t) - y(t)) + k_i \int_0^t r(\tau) - y(\tau)\,d\tau + k_d\left(b_{sd}\left(\frac{d}{dt}r(t)\right) - \left(\frac{d}{dt}y(t)\right)\right) \qquad [6.23]$$

where k_p is the position proportional control gain, k_d is the derivative control gain, k_i is the integral control gain, b_{sp} is the set-point weight on the reference position $r(t)$, and b_{sd} is the set-point weight on the velocity reference of $r(t)$. The dotted box labeled *Motor* in Figure 6.6 is the motor model in terms of the back-emf motor constant k_m, the electrical motor armature resistance R_m, and the equivalent moment of inertia of the motor pivot J_{eq}. The direct disturbance applied to the inertial wheel is represented by the disturbance torque variable T_d and the simulated disturbance voltage is denoted by the variable V_{sd}.

6.3.1. PD Control Design

The behaviour of the controlling the motor position is first analyzed using a PD control. By setting $k_i = 0$ in the PID control equation [6.23] and taking its Laplace transform, the PD transfer transfer function is

$$u = k_p\,(r - y) + k_d\,s\,(b_{sd}\,r - y) \qquad [6.24]$$

Combining the position process model

$$\frac{\Theta_m(s)}{V_m(s)} = \frac{K}{s\,(\tau\,s + 1)} \qquad [6.25]$$

with the PD control [6.24] gives the closed-loop transfer function of the motor position system

$$G_{\theta,r}(s) = \frac{(s\,k_d\,b_{sd} + k_p)\,K}{s^2\,\tau + (K\,k_d + 1)\,s + K\,k_p} \quad [6.26]$$

Similarly to the speed control laboratory, the standard characteristic function shown in [6.13]can be achieved by setting the proportional gain to

$$k_p = \frac{\omega_0^2\,\tau}{K} \quad [6.27]$$

and the derivative gain to

$$k_d = \frac{-1 + 2\,\zeta\,\omega_0\,\tau}{K} \quad [6.28]$$

6.3.2. Response to Load Disturbance

Next, the behaviour of the PID closed-loop system when it is subjected to a disturbance is examined. The block diagram shown in Figure 6.7 represents the load disturbance to position response when b_{sp} and b_{sd} in the PID are both set to 1.

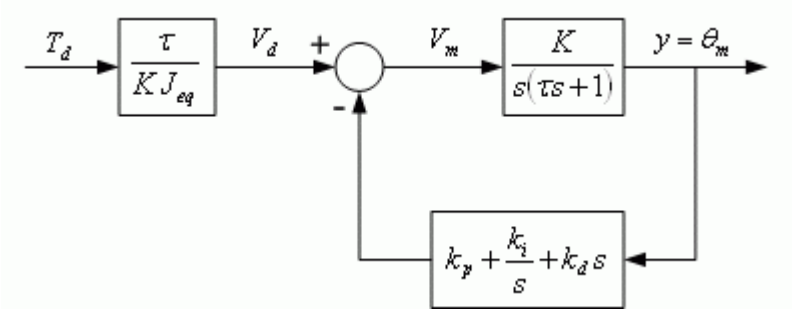


Figure 6.7: PID closed-loop block diagram to a load disturbance input.

The closed-loop disturbance to position transfer function is

$$G_{\theta,T}(s) = \frac{s\,\tau}{J_{eq}\,(s^3\,\tau + (K\,k_d + 1)\,s^2 + K\,k_p\,s + K\,k_i)} \quad [6.29]$$

Given a step disturbance with an amplitude of T_{d0}

$$T_d(s) = \frac{T_{d0}}{s} \quad [6.30]$$

the steady-state angle of the closed-loop system is

$$\theta_{ss} = T_{d0}\left(\lim_{s \to 0} G_{\theta,T}(s)\right) \quad [6.31]$$

.

The steady-state angle of the PD control, that is when $k_i = 0$ in [6.29], is

$$\theta_{ss\_PD} = \frac{\tau\, T_{d0}}{J_{eq}\, K\, k_p} \quad [6.32]$$

,

and the steady-state angle with integral action is

$$\theta_{ss\_PID} = 0 \quad [6.33]$$

.

Thus when the system is subjected to a disturbance, a constant steady-state error is observed when using the PD control system. However, the disturbance is rejected when integral control is used and the steady-state angle eventually goes to zero.

PID control design involves using the standard characteristic equation for a third-order system

$$(s^2 + 2\zeta\omega_0 s + \omega_0^2)(s + p_0) = s^3 + (2\zeta\omega_0 + p_0)s^2 + (\omega_0^2 + 2\zeta\omega_0 p_0)s + \omega_0^2 p_0 \quad [6.34]$$

where ω_0 is the natural frequency, ζ is the damping ratio, and p_0 is a zero. The characteristic equation of the closed-loop PID transfer function, i.e. the denominator of the transfer function [6.29], is

$$s^3 + \left(\frac{K k_d}{\tau} + \frac{1}{\tau}\right)s^2 + \frac{K k_p s}{\tau} + \frac{K k_i}{\tau} \quad [6.35]$$

.

The PID characteristic equation [6.35] matches [6.34] using the proportional gain

$$k_p = \frac{\omega_0 \tau (\omega_0 + 2\zeta p_0)}{K} \quad [6.36]$$

,

the derivative gain

$$k_d = \frac{-1 + 2\zeta\omega_0\tau + p_0\tau}{K} \quad [6.37]$$

,

and the integral gain

$$k_i = \frac{\omega_0^2 p_0 \tau}{K} \quad [6.38]$$

By varying the zero location, p_0, the time required by the closed-loop response to recover from a disturbance is changed.

Tracking a reference position square wave using PID control is first examined in this laboratory. Then, disturbance effects using PD and PID are studied through direct manual interaction or a simulated using a control switch in the VI. The LabVIEW virtual instrument for position control is shown in Figure 6.8.

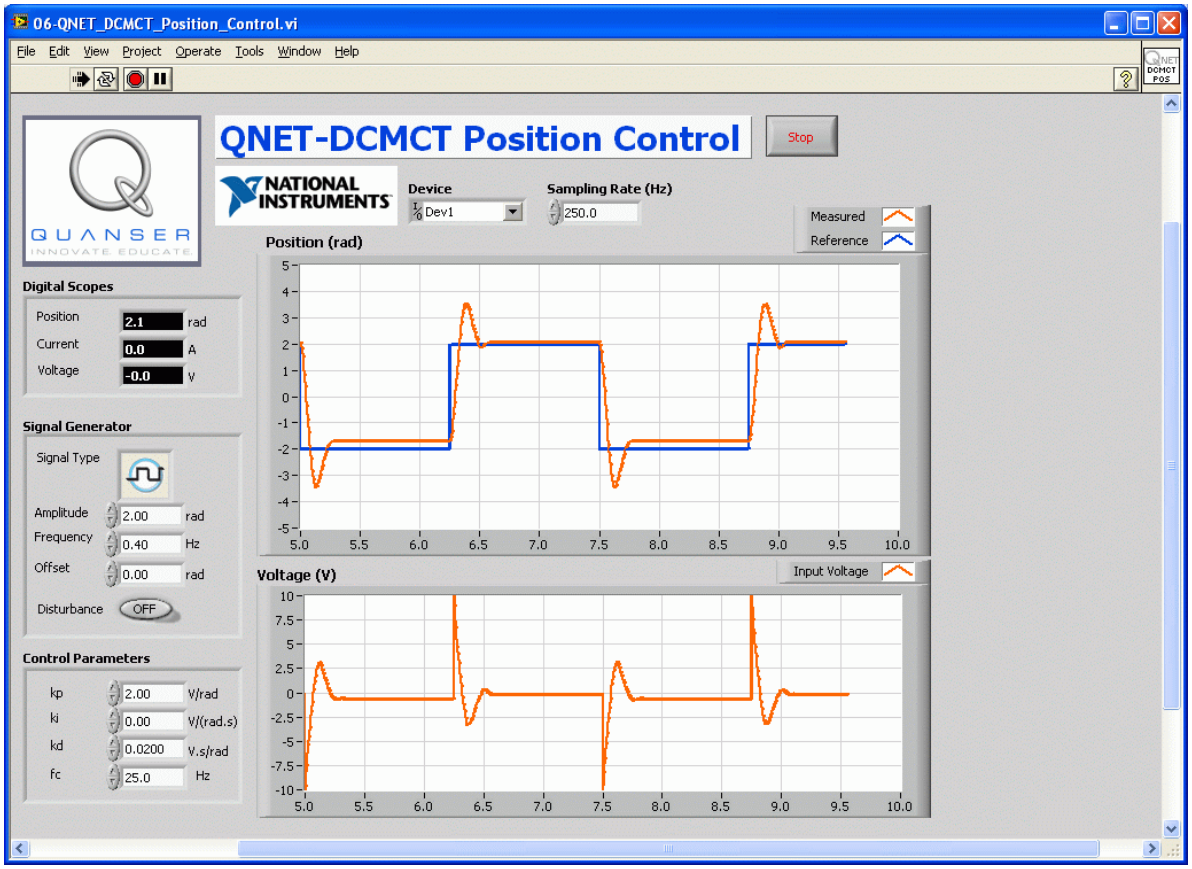


Figure 6.8 Virtual instrument for DC motor position control.

See *Wikipedia* for more information on motion control, control theory and PID.

7. Task-Based Control

Regulation and servo problems are very common, but feedback can be used in many other useful ways. The name task-based control is used as a common classification of a wide variety of problems. For instance, stabilization of an unstable system can be considered a task-based problem. However, it is a borderline example since it can also be viewed as a regulation problem. The Segway transporter is a typical example where stabilization is a key task. In that case stabilization is also merged with the steering functions. Other examples are damping of a swinging load on a crane, stabilization of a rocket during take-off, and the human posturing systems. There are many examples of task-based control in aerospace such as automatic landing and orbit transfer of satellites. Robotics is a rich field for task-based control with challenges such as collision avoidance, motion planning, and vision based control. Task-based control is typically more complicated than regulation and servoing but they may contain servo and regulation functions as sub-tasks. We have chosen the rotary pendulum system to illustrate task-based control

The QNET rotary inverted pendulum trainer is shown in Figure 7.1. The motor is mounted vertically in a metal chamber. An L-shaped arm is connected to the motor shaft and pivots between ±180 degrees. A pendulum is suspended on a horizontal axis at the end of the arm. The pendulum angle is measured by an encoder. The control variable is the input voltage to the pulse-width modulated amplifier that drives the motor. The output variables are the angle of the pendulum and the angle of the motor. Some of these components were used in the motion control experiment in Chapter 6.

Figure 7.1: The QNET rotary inverted pendulum trainer (ROTPENT).

7.1. Gantry Crane

This experiment illustrates some control tasks for gantry cranes. The gantry is a moving platform or trolley that transports the crane about the factory floor or harbor. The load hangs from the crane using wires and is moved by the gantry crane. Typically the problem is to move the load quickly and move it to the correct position. The fast motion necessary for production makes it more difficult to move the load to the correct location given the swinging motions of the crane. This problem can be mimicked using the rotary pendulum system by viewing the tip of the L-shaped arm as the moving trolley and the pendulum tip as the load being carried.

In this experiment we will begin by modeling the system and determine strategies to dampen the oscillations of the system.

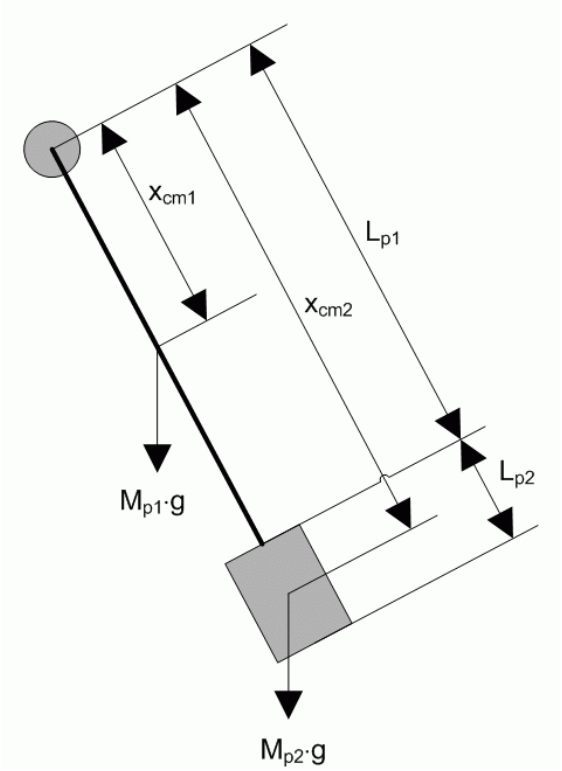


Figure 7.2 Free-body diagram of pendulum assembly.

Figure 7.2 shows the free-body diagram of the pendulum assembly that is composed of two rigid bodies: the pendulum link with mass M_{p1} and length L_{p1}, and the pendulum weight with mass M_{p2} and a length L_{p2}. The center of mass of the the pendulum link and the pendulum weight are calculated separately using the general expression

$$x_{cm} = \frac{\int p\, x\, dx}{\int p\, dx} \qquad [7.1]$$

where x is the linear distance from the pivot axis and p is the density of the body. The circle in the top-left corner of Figure 7.2 represents the axis of rotation or the pivot axis that goes into the page.

The pendulum system is then expressed as one rigid body with a single center of mass, as shown in Figure 7.3.

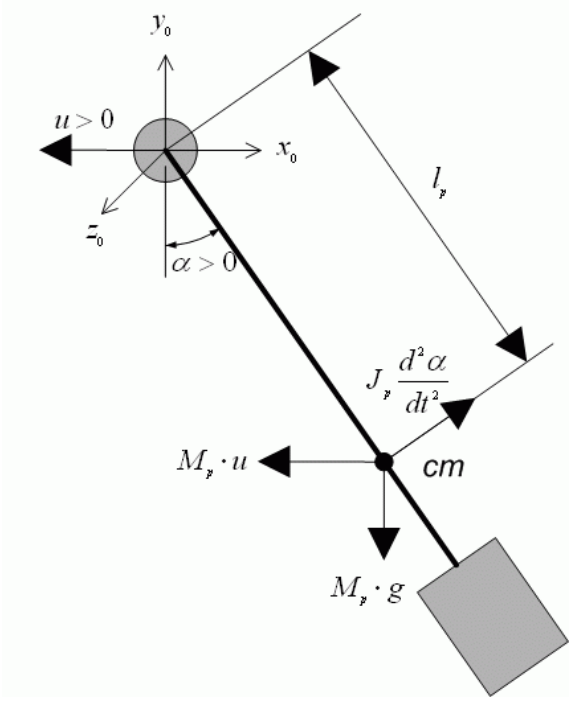


Figure 7.3 Free-body diagram of composite pendulum.

The center of mass of a composite object that contains *n* bodies can be calculated using

$$x_{cm} = \frac{\sum_{i=1}^{n} m_i x_{cm,i}}{\sum_{i=1}^{n} m_i} \qquad [7.2]$$

where $x_{cm,i}$ is the known center of mass of body *i* and m_i is the mass of body *i*.

From the free-body diagram in Figure 7.3, the resulting nonlinear equation of motion of the pendulum is

$$J_p \left(\frac{d^2}{dt^2} \alpha(t) \right) = M_p g l_p \sin(\alpha(t)) + M_p u l_p \cos(\alpha(t)) \qquad [7.3]$$

,

where J_p is the moment of inertia of the pendulum at the pivot axis z_0, M_p is the total mass of the pendulum assembly, *u* is the linear acceleration of the pivot axis, and l_p is the center of mass position as depicted in Figure 7.3. Thus as the pivot accelerates towards the left the inertia of the pendulum causes it to swing upwards while the gravitation force $M_p g$ and the applied force $M_p u$ (the left-hand terms in Equation [7.3]) pull the pendulum downwards.

The moment of inertia of the pendulum can be found experimentally. Assuming the pendulum is unactuated, linearizing Equation [7.3] and solving for the differential equation gives the expression

$$J_p = \frac{1}{4}\frac{M_p g l_p}{\pi^2 f^2}, \quad [7.4]$$

where f is the measured frequency of the pendulum as the arm remains rigid. The frequency is calculated using

$$f = \frac{n_{cyc}}{\Delta t}, \quad [7.5]$$

where n_{cyc} is the number of cycles and Δt is the duration of these cycles. Alternatively, J_p can be calculated using the moment of inertia expression

$$J = \int r^2 \, dm, \quad [7.6]$$

where r is the perpendicular distance between the element mass, dm, and the axis of rotation.

In addition to finding the moment of inertia, this laboratory investigates the *stiction* that is present in the system. The rotor of the DC motor that moves the ROTPEN system requires a certain amount of current to begin moving. In addition, the mass from the pendulum system requires even more current to actually begin moving the system. The friction is particularly severe for velocities around zero because friction changes sign with the direction of rotation.

The virtual instrument for studying the physics of the pendulum when in the gantry configuration is shown in Figure 7.4.

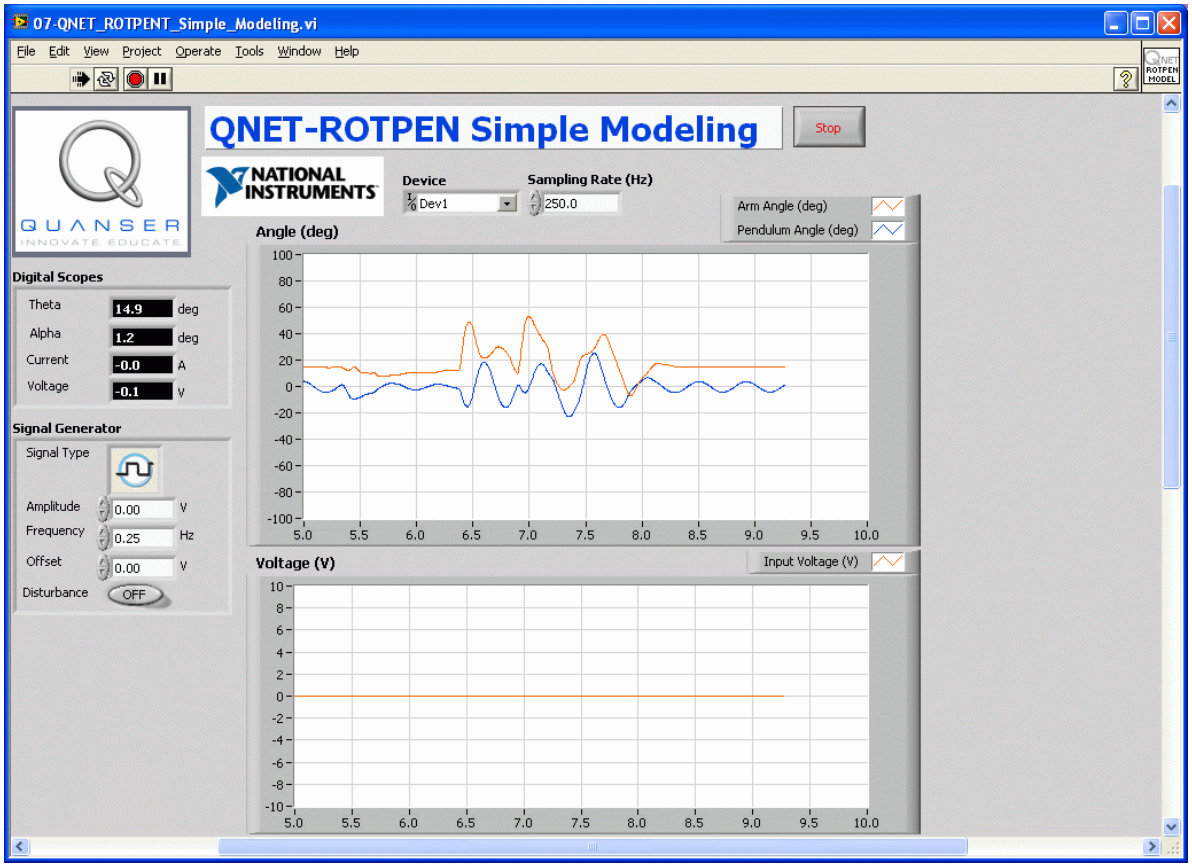


Figure 7.4: LabVIEW virtual instrument for QNET ROTPENT simple modeling.

See *Wikipedia* for more information on center of mass, inertia, pendulum, friction.

7.2. Balancing

Balancing is a common control task. In this experiment we will find control strategies that balance the pendulum in the upright position while maintaining a desired position of the arm. When balancing the system the pendulum angle, α, is small and balancing can be accomplished simply with a PD controller. If we are also interested in keeping the arm in a fixed position a feedback from the arm position will also be introduced. The control law can then be expressed as

$$u = -k_{p,\theta}(\theta - \theta_r) - k_{p,\alpha}\alpha - k_{d,\theta}\left(\frac{\partial}{\partial t}\theta\right) - k_{d,\alpha}\left(\frac{\partial}{\partial t}\alpha\right), \qquad [7.7]$$

where $k_{p,\theta}$ is the arm angle proportional gain, $k_{p,\alpha}$ is the pendulum angle proportional gain,

$k_{d,\theta}$ is the arm angle derivative gain, and $k_{d,\alpha}$ is the pendulum angle derivative gain. The desired angle of the arm is denoted by θ_r and there is no reference for the pendulum angle because the desired position is zero.

There are many different ways to find the controller parameters. As discussed in Section 7.5, one method is based on LQR-optimal control. Initially, however, the behaviour of the system will be explored using default parameters.

When balancing the pendulum over a fixed point, the arm tends to oscillate about that reference because of the friction present in the motor. Due to friction, the motor will not move until the control signal is sufficiently large and the generated torque is larger than the stiction (see Section 7.1 for more details). This means that the pendulum has to fall a certain angle before the motor moves and the net result is an oscillating motion.

Friction can be compensated by introducing a Dither signal at the input voltage of the DC motor. The Dither signal used has the form

$$V_d = A_d \sin(f_d t) + V_{d0}, \qquad [7.8]$$

where A_d is the voltage amplitude, f_d is the sinusoid frequency, and V_{d0} is the offset voltage of the signal.

See *Wikipedia* for more information on PID and friction.

7.3. Energy Control

If the arm angle is kept constant and the pendulum is given an initial position it would swing with constant amplitude. Because of friction there will be damping in the oscillation. The purpose of energy control is to control the pendulum in such a way that the friction is constant. The potential energy of the pendulum is

$$E_p = M_p g l_p (1 - \cos(\alpha(t))) \qquad [7.9]$$

and the kinetic energy is

$$E_k = \frac{1}{2} J_p \left(\frac{d}{dt} \alpha(t) \right)^2. \qquad [7.10]$$

The potential energy is zero when the pendulum is at rest at $\alpha = 0$ in Figure 7.3, and equals $2 \cdot M_p \cdot g \cdot l_p$ when the pendulum is upright at $\alpha = \pm\pi$. The sum of the potential and kinetic energy of the pendulum is

$$E = \frac{1}{2} J_p \left(\frac{d}{dt} \alpha(t) \right)^2 + M_p g l_p (1 - \cos(\alpha(t))) \quad [7.11]$$

Differentiating expression [7.11] results in the differential equation

$$\frac{\partial}{\partial t} E = \left(\frac{d}{dt} \alpha(t) \right) \left(J_p \left(\frac{d^2}{dt^2} \alpha(t) \right) + M_p g l_p \sin(\alpha(t)) \right) \quad [7.12]$$

Substituting Equation [7.3] for pendulum acceleration into Equation [7.12] gives

$$\frac{\partial}{\partial t} E = M_p u l_p \cos(\alpha(t)) \left(\frac{d}{dt} \alpha(t) \right) \quad [7.13]$$

Since the acceleration of the pivot is proportional to current driving the arm motor and thus also proportional to the drive voltage we find that it is easy to control the energy of the pendulum. The proportional control law

$$u = (E_r - E) \cos(\alpha(t)) \left(\frac{d}{dt} \alpha(t) \right) \quad [7.14]$$

drives the energy towards the reference energy E_r. Notice that the control law is nonlinear because the proportional gain depends on the pendulum angle, α. Also, notice that the control changes sign when $d\alpha/dt$ changes sign and when the angle is ±90°.

However, for energy to change quickly the magnitude of the control signal must be large. As a result the following swing-up controller is implemented in the LabVIEW VI

$$u = sat_{u_{max}} \left(\mu (E - E_r) \operatorname{Sign} \left(\cos(\alpha(t)) \left(\frac{d}{dt} \alpha(t) \right) \right) \right) \quad [7.14]$$

where μ is a tunable control gain and the sat_{umax} function saturates the control signal at the maximum acceleration of the pendulum pivot, u_{max}.

See *Wikipedia* for more information potential energy, kinetic energy, control theory, and nonlinear control.

7.4. Hybrid Swing-Up Control

The energy swing-up control in [7.14] can be combined with the balancing control law in [7.7] to obtain a control law which performs the dual tasks of swinging up the pendulum and balancing it. As illustrated in Figure 7.5, this can be accomplished by switching between the two control systems.

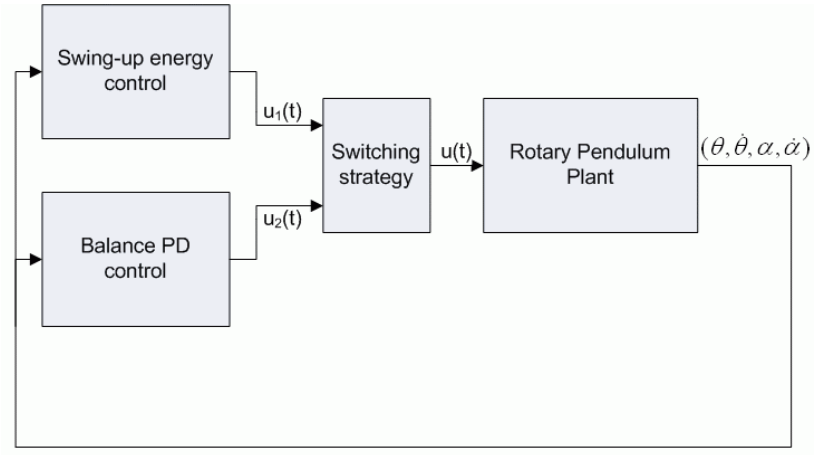


Figure 7.5 Swing-up hybrid control.

This system can be modeled as a hybrid system. Hybrid systems are systems with both continuous and discrete parts. There are two continuous part : the closed-loop system using the swing-up energy controller and the closed-loop system using the PD balance controller. The switching strategy is the discrete element that chooses which controller, or system, to run. The switching logic can be obtained by determining a region in state space where the balancing works well. Balancing control is then used inside this region and energy control is used outside the region. Figure 7.6 is a called a hybrid automaton and, for this specific task, can be used to describe the system model and the switching logic.

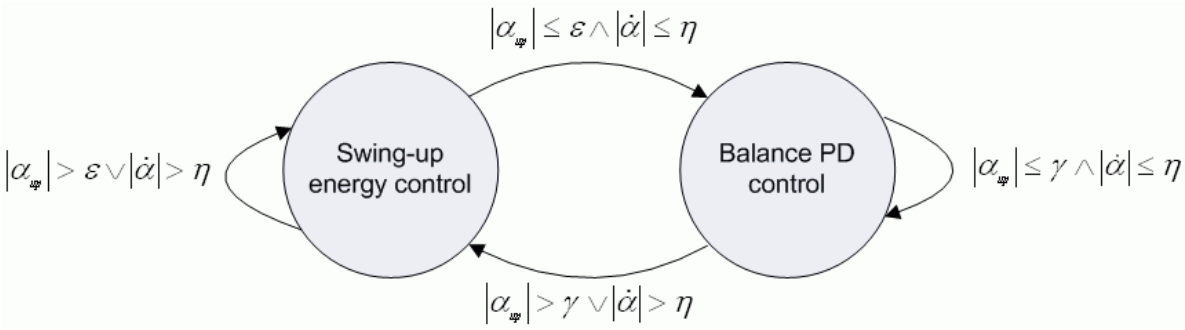


Figure 7.6 Hybrid swing-up controller automaton.

The circles in Figure 7.6 are called *locations* and represent the two different continuous system. The arrows are called *edges* and represent the discrete jumps taken when certain condition are satisfied. The angle used in the switching logic in Figure 7.6 is called the upright angle. It is defined as zero when the pendulum is about its upright vertical position and expressed mathematically using

$$\alpha_{up} = \left| \mathrm{Mod}(\alpha, 2\pi) \right| - \pi \quad . \qquad [7.15]$$

The various switching parameters can then be set as :

$$\varepsilon = 2\,[deg]\,,$$

$$\eta = 720\left[\frac{deg}{s}\right], \text{ and} \qquad [7.16]$$

$$\gamma = 30\,[deg]\,.$$

Given that the pendulum starts in the downward vertical position, it is in the swing-up location of the hybrid automaton. The swing-up controller pumps energy into the pendulum until it swings within ±2° of its upright vertical position. Once the pendulum is within that that range and does not exceed 720 °/s in either direction, the edge is taken to engage the balance controller. It remain in the *Balance PD control* location until the pendulum goes beyond the ±30° position range or beyond ±720 °/s.

The virtual instrument used to run the balance and swing-up controllers on the QNET rotary pendulum system is shown in Figure 7.7.

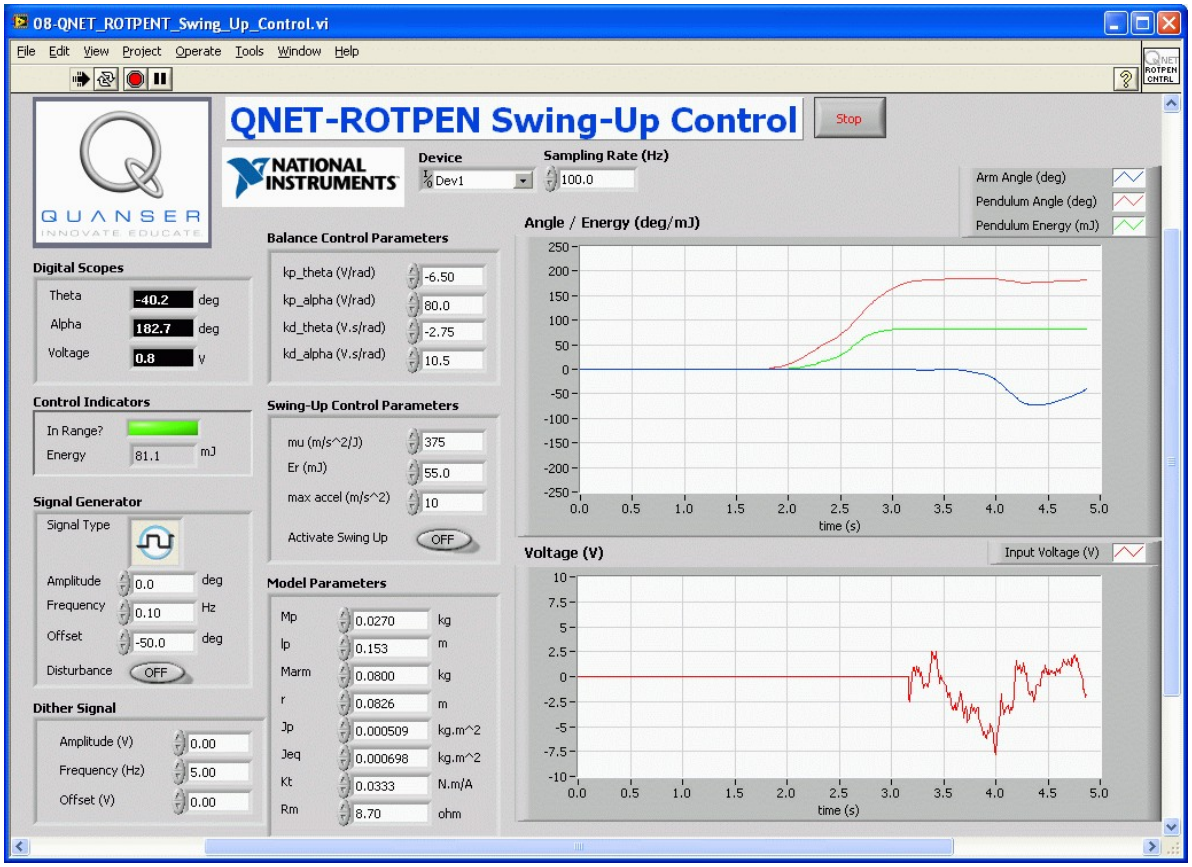


Figure 7.7: LabVIEW virtual instrument for QNET-ROTPENT swing-up control.

7.5. Optimal Balancing Control

A rich collection of methods for finding parameters of control strategies have been developed. Several of them have also been packaged in tools that are relatively easy to use. Linear Quadratic Regulator (LQR) theory is a technique that is suitable for finding the parameters of the balancing controller in [7.7]. Given that the equations of motion of the system can be described in the form

$$\frac{\partial}{\partial t} x = A\,x + B\,u \qquad [7.17]$$

the LQR algorithm computes a control task, u, to minimize the criterion

$$J = \int_0^\infty \mathrm{x}(t)^T Q\, \mathrm{x}(t) + \mathrm{u}(t)^T R\, \mathrm{u}(t)\, dt \qquad [7.18]$$

.

The matrix Q defines the penalty on the state variable and the matrix R defines the penalty on the control actions. Thus when Q is made larger, the controller must work harder to minimize the cost function and the resulting control gain will be larger. In our case the state vector x is defined

$$x = \left[\theta, \alpha, \frac{\partial}{\partial t}\theta, \frac{\partial}{\partial t}\alpha \right]^T \qquad [7.19]$$

.

Since there is only one control variable, R is a scalar and the control strategy used to minimize cost function J is given by

$$u = \left(-Kx = -k_{p,\theta}\,\theta - k_{p,\alpha}\,\alpha - k_{d,\theta}\left(\frac{\partial}{\partial t}\theta\right) - k_{d,\alpha}\left(\frac{\partial}{\partial t}\alpha\right)\right) \qquad [7.20]$$

which only differs from Equation [7.7] by having a different reference value for the arm angle.

The LQR theory has been packaged in the LabVIEW *Control Design and Simulation Module*. Thus given a model of the system in the form of the state-space matrices A and B and the weighting matrices Q and R, the LQR function in the *Control Design Toolkit* computes the feedback control gain automatically. In this experiment, the model is already available. In the laboratory, the effect of changing the Q weighting matrix while R is fixed to 1 on the cost function J will be explored.

See *Wikipedia* for more information on optimal control.

8. VTOL Control

The QNET vertical take-off and landing (VTOL) trainer is shown in Figure 8.1. The system consists of a variable-speed fan with a safety guard mounted on an arm. At the other end of the arm, an adjustable counterweight is attached. This allows the position of the weight to be changed, which in turn affects the dynamics of the system. The arm assembly pivots about a rotary encoder shaft. The VTOL pitch position can be acquired from this setup.

Figure 8.1: The QNET vertical take-off and landing (VTOL) trainer.

Some examples of real-world VTOL devices are helicopters, rockets, balloons, and harrier jets. Aerospace devices are typically more difficult to model. Usually this will involve using software system identification tools to determine parameters or actual dynamics. Due to their inherent complexity, flight systems are usually broken down into different subsystems to make it more manageable. These subsystems can be dealt with individually and then integrated to provide an overall solution.

There are three experiments: current control, modeling, and flight control.

8.1. Cascade Control

The VTOL device is broken down into two subsystems: the voltage-current dynamics of the motor and the current-position dynamics of the VTOL body. The cascade control implemented in the VTOL trainer is depicted in Figure 8.2, below. A PI current controller, the inner loop, is designed to regulate the current inside the motor according to a desired current reference. This current reference is generated from the outer-loop controller: a PID compensator that controls the pitch of the VTOL trainer.

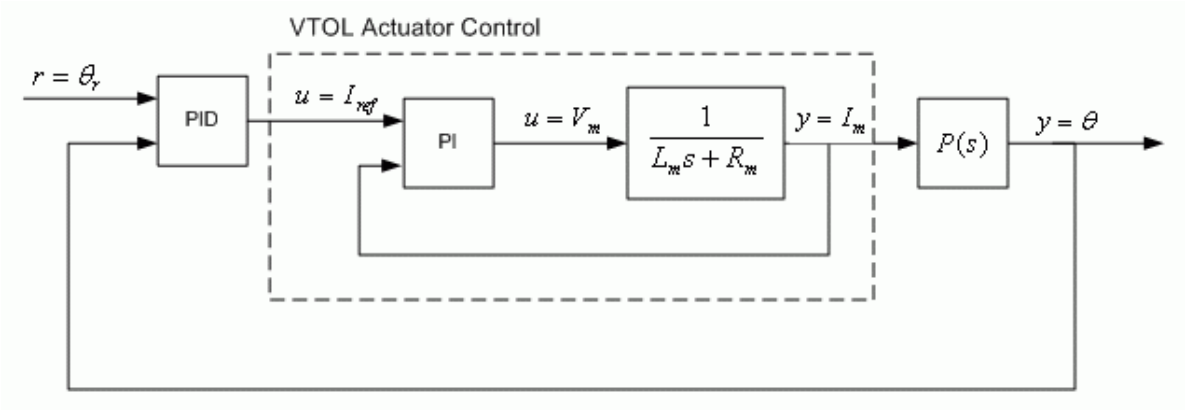


Figure 8.2: VTOL trainer cascade control system.

8.2. Current Control

In cases where the actuator has relatively slow dynamics, such as an electromagnet with a large inductance, it is favorable to design a current controller. Typically a proportional-integral compensator is used to regulate the current flowing in the load. This basically makes the actuator dynamics negligible and simplifies the control design of the outer-loop.

In this case, the voltage-current relationship of the VTOL trainer motor can be described, in the time-domain, by the equation

$$v_m(t) = R_m\, i_m(t) + L_m \left(\frac{d}{dt} i_m(t) \right) \qquad [8.1]$$

and by the transfer function

$$I_m(s) = \frac{V_m(s)}{R_m + L_m s} \quad [8.2]$$

Figure 8.3 shows the VTOL current control system implemented. The PI compensator computes the voltage necessary to reach the desired current.

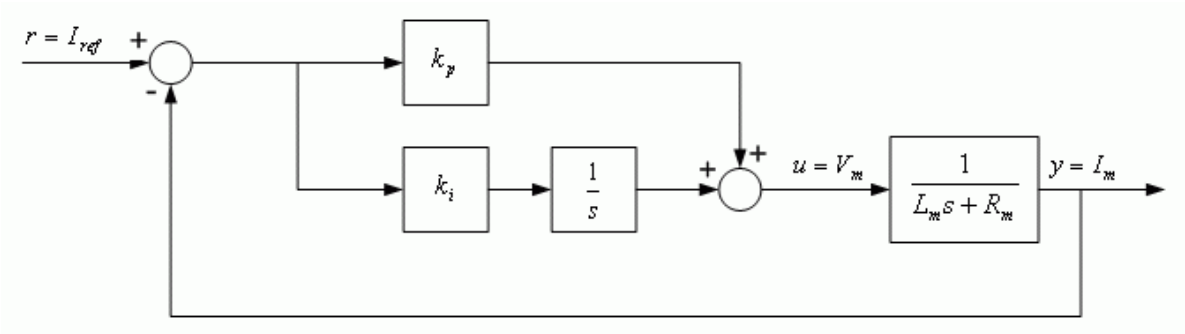


Figure 8.3: VTOL motor PI current control loop.

Using the PI controller

$$v_m(t) = k_{p,c}(i_{ref}(t) - i_m(t)) + k_{i,c}\int i_d(t) - i_m(t)\,dt \quad [8.3]$$

we obtain the following closed-loop transfer function

$$G_{I_{ref}, I_m}(s) = \frac{k_{p,c}\,s + k_{i,c}}{s^2 L_m + (k_{p,c} + R_m)\,s + k_{i,c}} \quad [8.4]$$

To match the *standard second-order characteristic equation* shown in [5.5], we need a proportional gain of

$$k_{p,c} = -R_m + 2\zeta\omega_n L_m \quad [8.4]$$

and an integral gain of

$$k_{i,c} = \omega_n^2 L_m \quad [8.4]$$

These gains can then be designed according to a desired natural frequency, ω_n, and damping ratio, ζ.

The virtual instrument used to run the current controller on the QNET vertical take-off and landing system is shown in Figure 8.4.

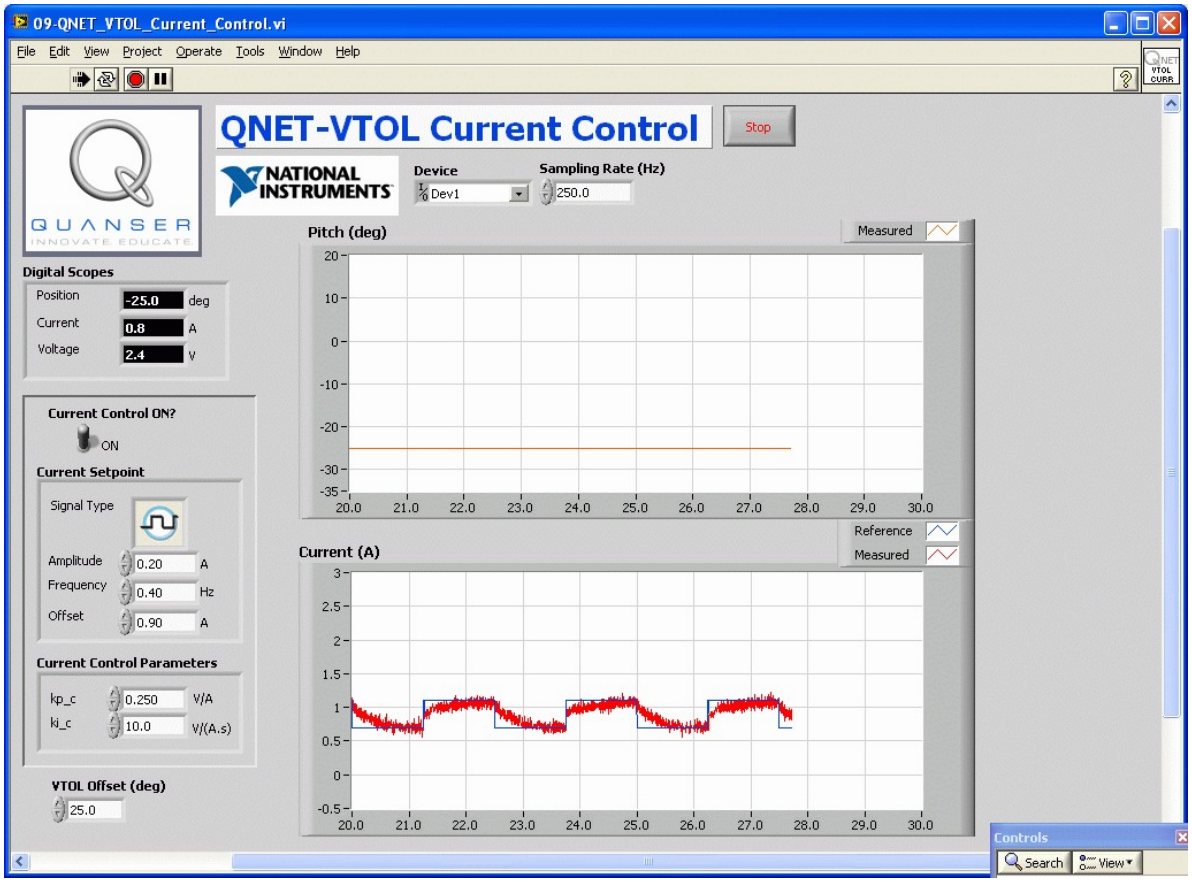


Figure 8.4: LabVIEW virtual instrument for QNET-VTOL trainer current control

8.3. Modeling 1-DOF VTOL

Unlike a DC motor, this system has to be characterized with at least a second-order model. The equation of motion is derived from first principles and then used to obtain the transfer function representing the current to position VTOL dynamics.

Various methods can be used to find the modeling parameters. In the laboratory, the parameters are first found manually by performing a few experiments and taking measurements. Thereafter, the *LabVIEW System Identification Toolkit* is used to automatically find the model. This demonstrates how to use software tools to identify parameters or even entire models (especially important for higher-order systems). The

modeling is then validated by running the obtained model in parallel with the actual system.

8.3.1. Torques acting on the VTOL

The free-body diagram of a 1-DOF Vertical Take-Off and Landing device that pivots about the pitch axis is shown in Figure 8.5.

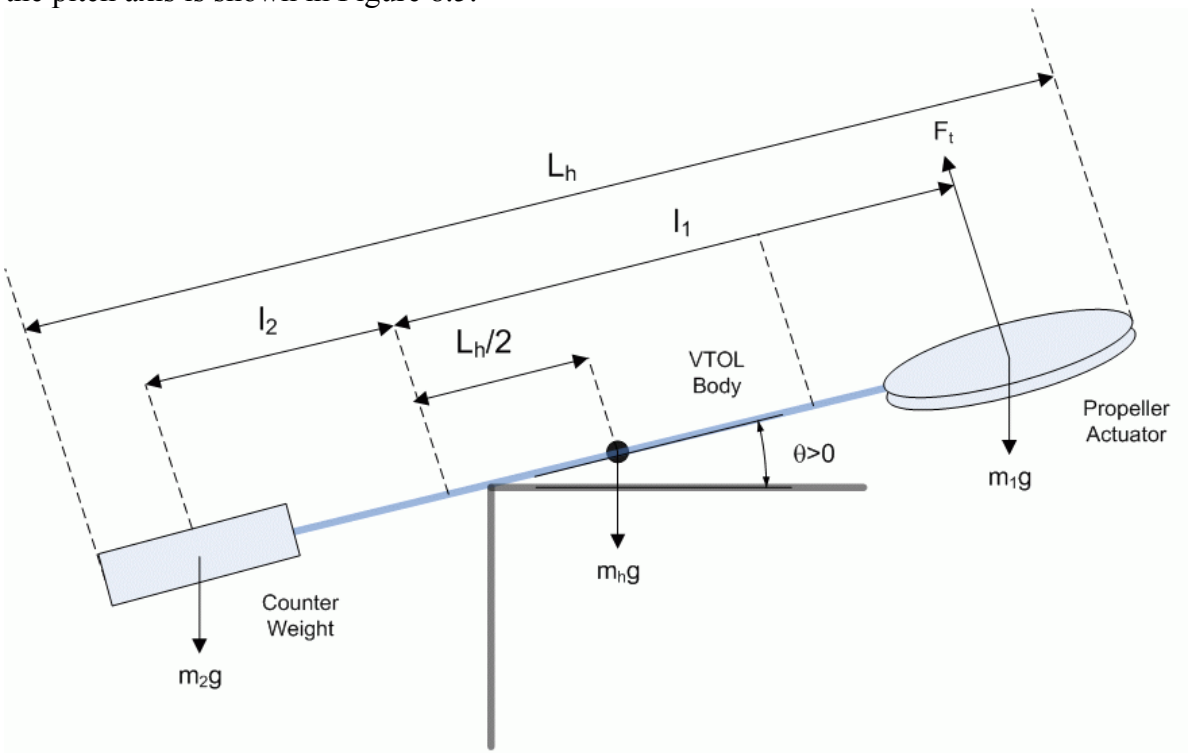


Figure 8.5: Free-body diagram of 1-DOF VTOL.

As shown in Figure 8.5, the torques acting on the rigid body system can be described by the equation

$$\tau_t + m_2 g \cos(\theta(t)) l_2 - m_1 g \cos(\theta(t)) l_1 - \frac{1}{2} m_h g \cos(\theta(t)) L_h = 0 \quad [8.1]$$

The thrust force, F_t, is generated by the propeller and acts perpendicular to the fan assembly. The thrust torque is given by

$$\tau_t = F_t l_1 \quad [8.2]$$

where l_1 is the length between the pivot and center of the propeller, as depicted in Figure 8.5. In terms of the current, the thrust torque equals

$$\tau_t = K_t I_m \quad [8.3]$$

where K_t is the thrust current-torque constant. With respect to current, the torque equation becomes

$$K_t I_m + m_2 g \cos(\theta(t)) l_2 - m_1 g \cos(\theta(t)) l_1 - \frac{1}{2} m_h g \cos(\theta(t)) L_h = 0 \quad [8.4]$$

The torque generated the propeller and the gravitational torque acting of the counter-weight act in the same direction and oppose the gravitational torques on the helicopter body and propeller assembly.

We define the VTOL trainer as being in *equilibrium* when the thrust is adjusted until the VTOL is horizontal and parallel to the ground. At equilibrium, the torques acting on the system are described by the equation

$$K_t I_{eq} + m_2 g l_2 - m_1 g l_1 - \frac{1}{2} m_h g L_h = 0 \quad [8.5]$$

where I_{eq} is the current required to reach equilibrium.

8.3.2. Equation of Motion

The angular motions of the VTOL trainer with respect to a thrust torque, τ_t, can be expressed by the equation

$$J\left(\frac{d^2}{dt^2}\theta(t)\right) + B\left(\frac{d}{dt}\theta(t)\right) + K\theta(t) = \tau_t \quad [8.6]$$

where θ is the pitch angle, J is the equivalent moment of inertia acting about the pitch axis, B is the viscous damping, and K is the stiffness. With respect to current, this becomes

$$J\left(\frac{d^2}{dt^2}\theta(t)\right) + B\left(\frac{d}{dt}\theta(t)\right) + K\theta(t) = K_t I_m \quad [8.7]$$

In Section 7.1, we showed how to find the moment of inertia of an object by integrating over a continuous body. However, when finding the moment of inertia of a composite body with n point masses its easier to use the formula

$$J = \sum_{i=1}^{n} m_i r_i^2 \quad [8.8]$$

where for object i, m_i is its mass and r_i is the perpendicular distance between the axis of rotation and the object.

8.3.3. Process Transfer Function Model

The transfer function representing the current to position dynamics of the VTOL trainer is

$$P(s) = \frac{K_t}{J\left(s^2 + \frac{Bs}{J} + \frac{K}{J}\right)} \quad [8.9]$$

This is obtained by taking the Laplace transform of Equation [8.7] and solving for $\Theta(s)/I_m(s)$. Notice that the denominator

$$s^2 + \frac{Bs}{J} + \frac{K}{J} \quad [8.10]$$

matches the characteristic second-order transfer function

$$s^2 + 2\zeta\omega_n s + \omega_n^2 \quad [8.11]$$

By determining the natural frequency of the system, one can find the stiffness using

$$K = \omega_n^2 J \quad [8.12]$$

The virtual instrument used to validate a transfer function model on the QNET VTOL trainer is shown in Figure 8.6. This VI can also be used to find the VTOL device transfer function using the *System Identification Toolkit*.

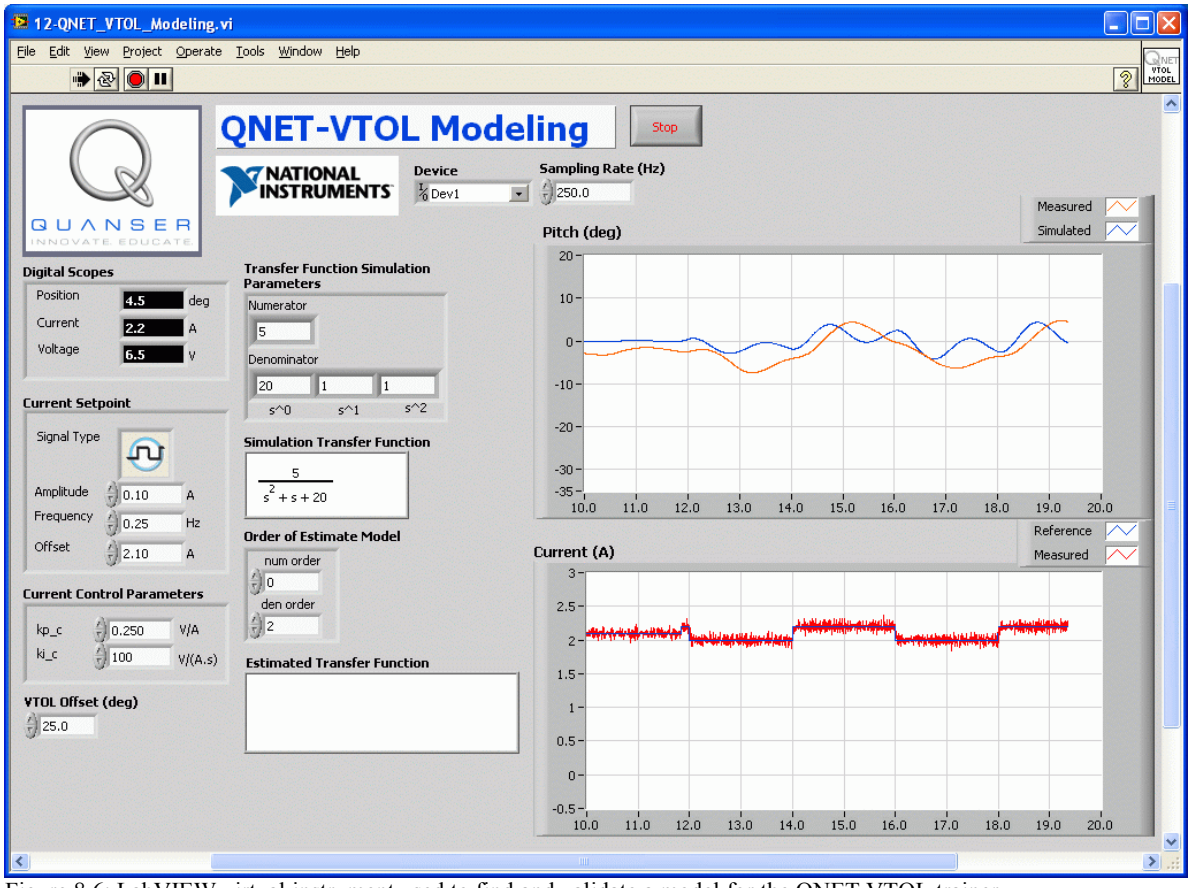


Figure 8.6: LabVIEW virtual instrument used to find and validate a model for the QNET VTOL trainer.

8.3.3.1. Natural Frequency of a Response

See Section 7.1 for the equation of how to find the natural frequency.

8.4. Flight Control

8.4.1. Steady-state Error Analysis

Steady-state error is the difference between the reference and output signals after the system response has settled. Thus for a time t when the system is in steady-state, the steady-state error equals

$$e_{ss} = r_{ss}(t) - y_{ss}(t) \qquad [8.13]$$

where r_{ss} is the value of the steady-state reference and y_{ss} is the steady-state value of the process output.

The block diagram shown in Figure 8.7 is a general unity feedback system with a compensator C(s) and a transfer function representing the plant, P(s). The measured output, Y(s), is supposed to track the reference signal R(s) and the tracking has to yield to certain specifications.

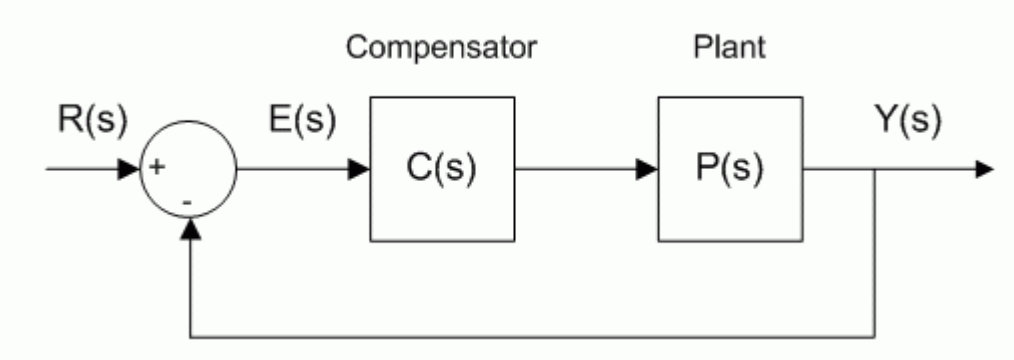


Figure 8.7: Unity feedback system.

The error of the system shown in Figure 8.7 is

$$E(s) = R(s) - Y(s) \qquad [8.14]$$

and by solving for E(s) the resulting closed-loop transfer function

$$E(s) = \frac{R(s)}{1 + C(s)\,P(s)} \qquad [8.15]$$

is obtained.

The error transfer function of the VTOL trainer when subject to a step of

$$R(s) = \frac{R_0}{s} \qquad [8.16]$$

and using the PID compensator

$$C(s) = k_p + k_d s + \frac{k_i}{s} \qquad [8.17]$$

is

$$\mathrm{E}(s) = \frac{R_0}{s\left(1 + \frac{\left(k_p + k_d s + \frac{k_i}{s}\right) K_t}{J\left(s^2 + \frac{B s}{J} + \frac{K}{J}\right)}\right)} \quad [8.18]$$

If the transfer function is stable, then the steady-state error can be found using the final-value theorem (FVT):

$$e_{ss} = \lim_{s \to 0} s\,\mathrm{E}(s) \quad [8.19]$$

Using FVT, the steady-state error of the VTOL trainer closed-loop PID step response is

$$e_{ss} = R_0 \left(\lim_{s \to 0} \frac{s\,(s^2 J + B s + K)}{s^3 J + B s^2 + s K + K_t k_p s + K_t k_d s^2 + K_t k_i} \right) \quad [8.20]$$

8.4.2. PID Control Design

The PID control loop used for the VTOL device is depicted in Figure 8.8.

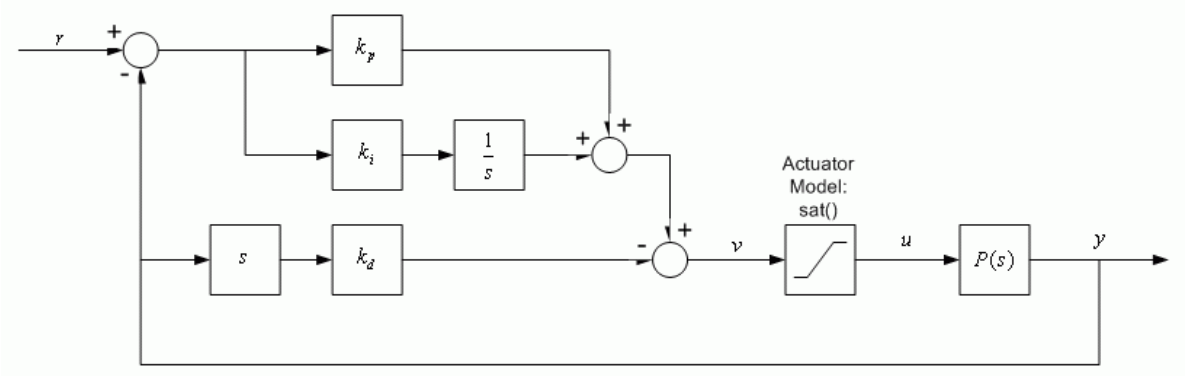


Figure 8.8: VTOL PID Control Loop.

The transfer function representing the VTOL trainer position-current relation in Equation [8.9] is used to design the PID controller. The input-output relation in the time-domain for a PID controller is

$$u(t) = k_p(\theta_d(t) - \theta(t)) + k_i \int \theta_d(t) - \theta(t)\,dt - k_v\left(\frac{d}{dt}\theta(t)\right), \quad [8.21]$$

where k_p is the proportional gain, k_i is the integral gain, and k_v is the velocity gain. Remark that only the measured velocity is used, i.e. instead of using the derivative of the error. The closed loop transfer function from the position reference, r, to the angular VTOL position output, θ, is

$$G_{\theta,r}(s) = \frac{K_t(k_p s + k_i)}{s^3 J + (B + K_t k_v)s^2 + (K + K_t k_p)s + K_t k_i}. \quad [8.22]$$

The prototype third-order characteristic polynomial is given in Section 6.3. The characteristic equation in [8.22], i.e. the denominator of the transfer function, can match the desired characteristic equation in Equation [6.34] with the following gains:

$$k_p = \frac{-K + 2p_0\zeta\omega_n J + \omega_n^2 J}{K_t} \quad [8.23]$$

$$k_i = \frac{p_0\omega_n^2 J}{K_t} \quad [8.24]$$

and

$$k_v = \frac{-B + p_0 J + 2\zeta\omega_n J}{K_t}. \quad [8.25]$$

The virtual instrument used to run the flight controller on the QNET vertical take-off and landing system is shown in Figure 8.9.

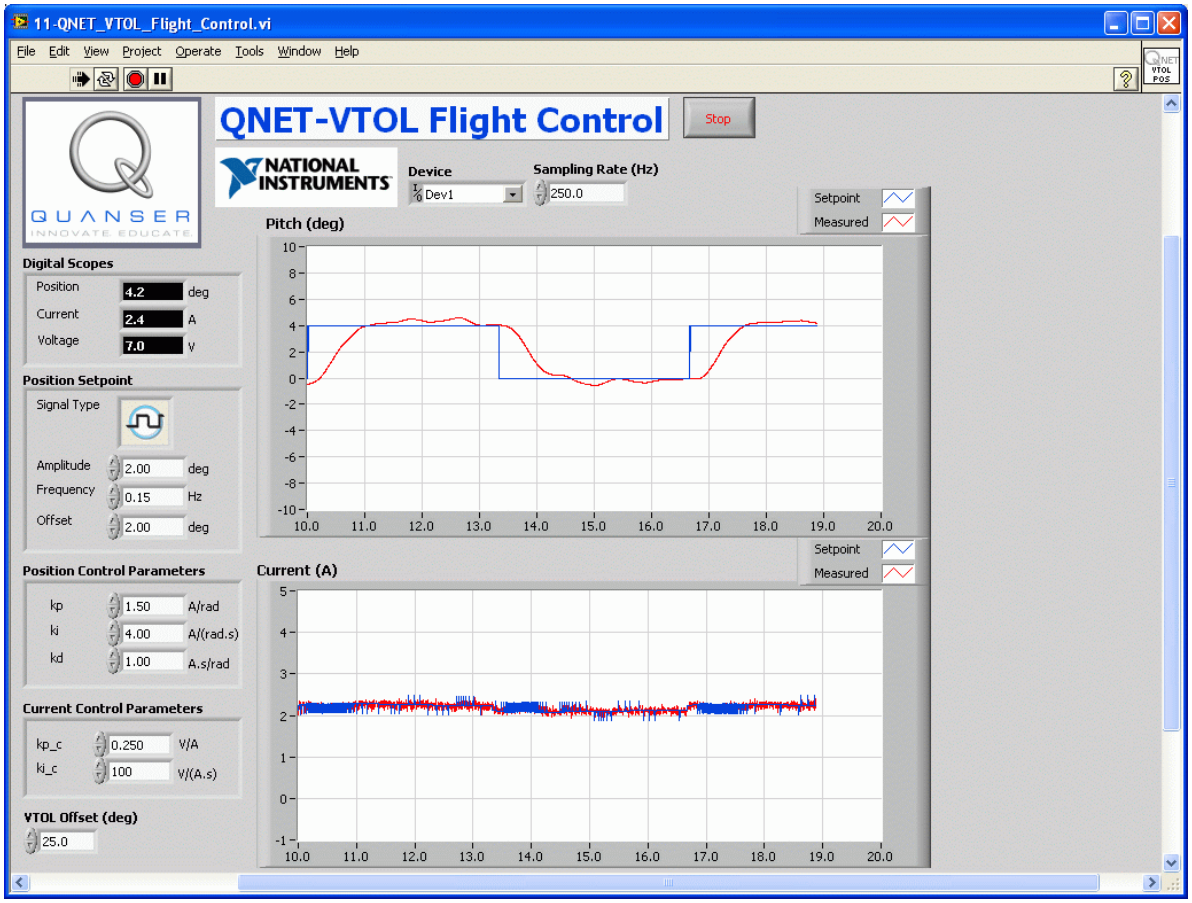


Figure 8.9: LabVIEW virtual instrument used to run VTOL trainer flight control.

9. Mechatronic Sensors

Mechatronics engineering is a cross-disciplinary field that combines mechanical and electronic design in control systems architecture though the application of computer programming. One of the most useful topics that can be covered in an introductory mechatronics course is the understanding and application of sensors. Various sensors are used in all types of industries. For example, in the automotive industry magnetic field transducers are used for throttle, pedal, suspension, and valve position sensing. In assembly line and machine automation, optical sensors are used for non-contact position sensing and safety. Piezo film sensors are installed in packages to log vibration history of a shipment.

The QNET-015 mechatronics sensors (MECHKIT) trainer is shown in Figure 9.1. It has ten types of sensors, two types of switches, a push button, and two LEDS. This QNET module can be used to teach the physical properties of most sensors used today, and the techniques and limitations of their application.

Here is a list of the components on the QNET-MECHKIT:

- Strain gauge to measure deflection
- Piezo film sensor to measure vibration.
- Rotary potentiometer to measure position.
- Pressure and thermistor sensors.
- Long range sensors: sonar and infrared.
- Short range sensors: magnetic field and optical.
- Micro switch, push button, and optical switch.
- Two light emitting diodes (LEDs)
- Encoder

Figure 9.1: The QNET mechatronic sensors trainer.

9.1. Sensor Properties

This section discusses various sensor properties that are often found in technical specifications.

9.1.1. Resolution

The resolution of a sensor is the minimum change that can be detected in the quantity that is being measured. For instance a sensor that measures angular position of a motor shaft may only be able to detect a 1 degree change. Thus if the motor moves 0.5 degrees, it will not be detected by the sensor. Depending on the precision needed for the application, this may be adequate.

9.1.2. Range

Range sensors can only take measurements of a target within a certain operating range. The operating range specifies a maximum, and sometimes also a minimum, distance where the target can be from the sensor in order to obtain an accurate measurement. Sensors with a

small range are the magnetic field and optical position sensors. Sensor with a relatively larger range are infrared and sonar.

9.1.3. Absolute and Incremental

Absolute sensors detect a unique position. Incremental sensors measure a relative position that depends on a prior position or last power on/off. For example, if an incremental rotary encoder is used to measure the position of wheel, the encoder will measure zero every time its power is reset. If an absolute sensor such as a rotary potentiometer is used, then it will detect the same angle regardless if it has just been powered.

9.1.4. Analog Sensor Measurement

Analog sensors output a signal that correlates to the quantity it is measuring. The relationship between the output signal of the sensor and the actual measurement varies depending on the type of sensor. For example, the voltage measured by a potentiometer is directly proportional to the angle it is measuring. However, the resistance of a thermistor decreases exponentially as the temperature increases.

Some of the different ways to characterize analog sensors is illustrated in Figure 9.2.

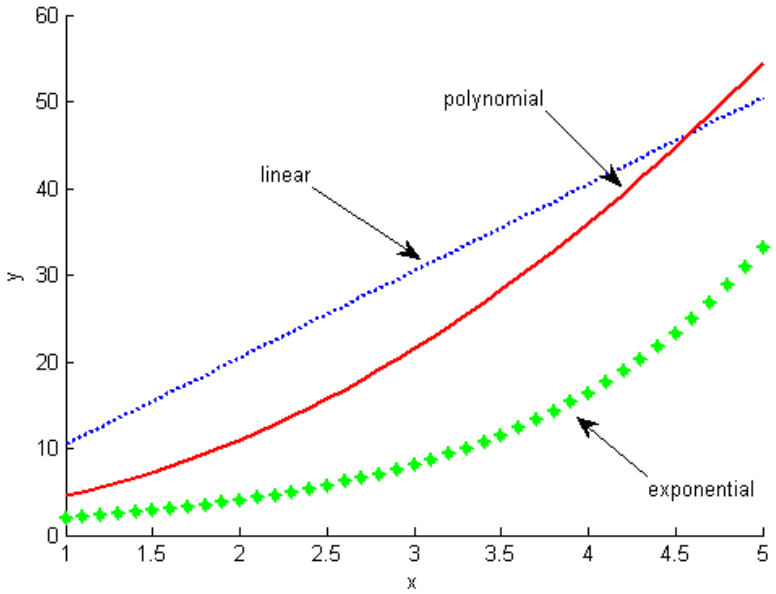


Figure 9.2: Different sensor responses.

Linear sensors can be modeled using the equation

$$y = a\,x + b \quad , \qquad [9.1]$$

where a is the rate of change and b is the offset. Variable x is the sensor output signal and y is the measurement, e.g. for the potentiometer x would be the voltage measured by the sensor and y would be the angular measurement (in either degrees or radians). Other types of sensors need to be characterized by more complex relationship such as polynomial

$$y = a\,x^2 + b\,x + c \qquad [9.2]$$

or exponential

$$y = a\,\mathbf{e}^{(b\,x)} \quad . \qquad [9.3]$$

9.2. Mechatronic Sensors Trainer Components

This section gives an overview of the different components on-board the QNET mechatronic sensors (MECHKIT) trainer.

9.2.1. Vibration and Deflection Sensors

9.2.1.1. Piezo

Piezo sensors measure vibration. The piezo sensor on the QNET-MECHKIT trainer, shown in Figure 9.3, is connected to a plastic band that has a brass disc weight at the end.

Figure 9.3: Piezo sensor on QNET mechatronic sensors trainer.

9.2.1.2. Strain Gage

A strain gage measures strain, or deflection, of an object. As shown in Figure 9.4, in the QNET mechatronic sensors trainer a strain gage is used to measure the deflection of a flexible link. As the link bends, the resistance of the strain gage changes.

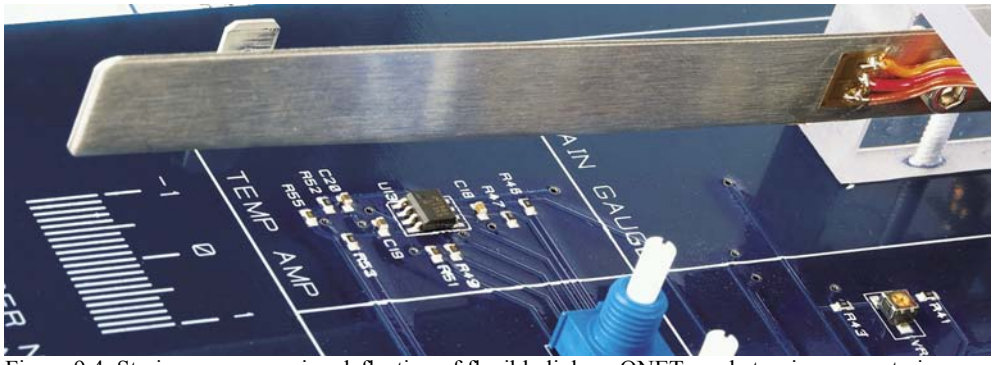


Figure 9.4: Strain gage measuring deflection of flexible link on QNET mechatronic sensors trainer.

Strain gages have many applications. They can be used in buildings to measure movement over a long span of time or in robotics for force control.

9.2.2. Measuring Angle

9.2.2.1. Rotary Potentiometer

Rotary potentiometers are absolute analog sensors used to measure angular position, such as a load shaft of a motor. They are great to obtain a unique position measurement. However, caution must be used as their signal is discontinuous. That is, after a few revolutions potentiometers will reset their signal back to zero. The potentiometer on the QNET MECHKIT board is shown in Figure 9.5.

Figure 9.5: Potentiometer knob on QNET mechatronic sensors trainer.

9.2.2.2. Incremental Optical Rotary Encoder

Similarly to rotary potentiometers, encoders can also be used to measure angular position. There are many types of encoders but one of the most common is the rotary incremental optical encoder, e.g. see the optical shaft encoder in Figure 9.6. Unlike potentiometers, encoders are relative. The angle they measure depends on the last position and when it was last powered. It should be noted, however, that absolute encoders are available.


Figure 9.6: US Digital incremental rotary optical shaft encoder.

The encoder has a coded disk that is marked with a radial pattern. As the disk rotates (with the shaft), the light from an LED shines through the pattern and is picked up by a photo sensor. This effectively generates the A and B signals shown in Figure 9.7. An index pulse is triggered once for every full rotation of the disk, which can be used for calibration or "homing" a system.

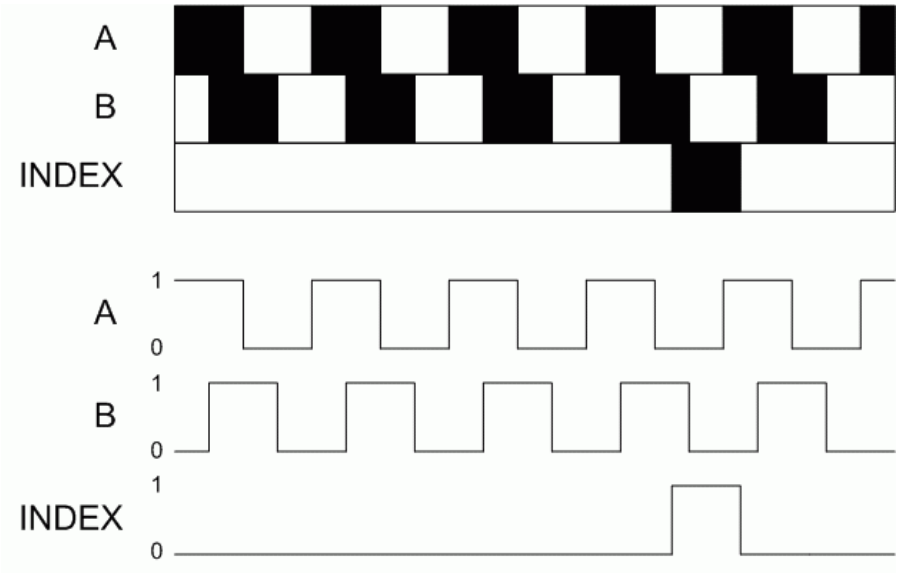


Figure 9.7: Optical incremental encoder signals.

The A and B signals that are generated as the shaft rotates are used in a decoder algorithm to generate a count. The resolution of the encoder depends on the coding of the disk and the decoder. For example, an encoder with 1024 lines on the disk can generate a total of 1024 counts for every rotation of the encoder shaft. However, in a quadrature decoder the number of counts quadruples, therefore the encoder would generate 4098 counts per revolution.

The encoder knob on the QNET mechatronic sensors trainer is pictured in Figure 9.8 and the corresponding A, B, and Index signals are displayed on the LEDs shown in Figure 9.9.

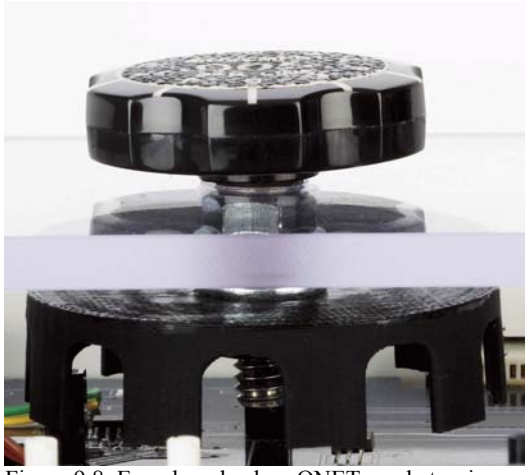


Figure 9.8: Encoder wheel on QNET mechatronic sensor trainer.

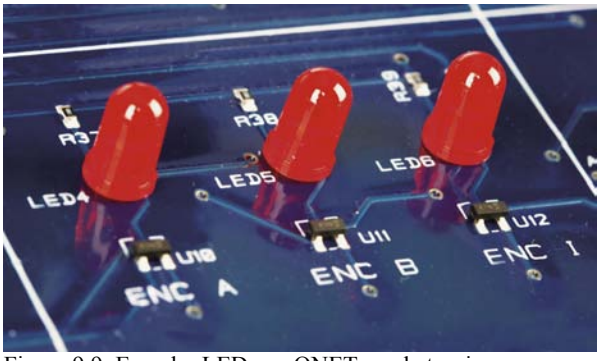


Figure 9.9: Encoder LEDs on QNET mechatronic sensor trainer.

9.2.3. Pressure Sensor

A pressure sensor is attached to the plunger on the QNET mechatronic board shown in Figure 9.10. This is a gage pressure sensor and its measurements are relative to the atmospheric pressure. The voltage signal generated is proportional to the amount of pressure in the vessel of the plunger. So as the plunger is pushed further, the air inside the vessel becomes more compressed and the reading increases.

Figure 9.10: Pressure sensor on QNET mechatronic sensors trainer.

Pressure sensors can also be used to indirectly measure other values. For example, in the QNET mechatronics board the position of the plunger head is measured. It can also be used to measure the amount of volume in a reservoir or the altitude of an aerial vehicle.

9.2.4. Temperature Sensors

As described in [13], there are several different types of transducers available to measure temperature: the thermocouple, the resistance temperature detector (RTD), the thermistor, and the integrated circuit (IC). Each have their own advantages and disadvantages. The Thermocouple has a wide temperature range and is easy to use but is the least stable and sensitive. The RTD, on the other hand, is most stable and accurate of the sensors but is slow and relatively more expensive. The IC is the only linear transducer, has the highest output, but is slow. The thermistor responds very quickly but has a limited temperature range. The thermistor on the mechatronic sensors board is shown in Figure 9.11.

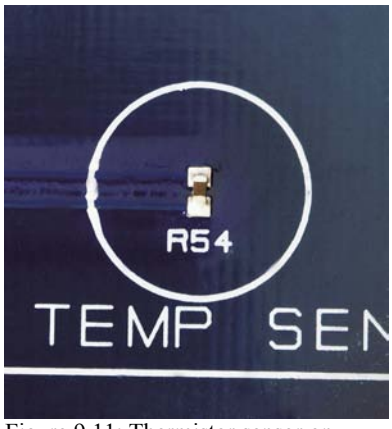


Figure 9.11: Thermistor sensor on QNET mechatronic sensors trainer.

The thermistor is a resistor that changes value according to the temperature. As given in [12], the relationship between the resistance of the thermistor and the temperature, *T*, can be described using the B-parameter equation

$$R = R_0 \, \mathbf{e}^{\left(B \left(\frac{1}{T} - \frac{1}{T_0} \right) \right)} \quad [9.4]$$

The resistance is R_0 when the temperature is at T_0. For the thermistor on the mechatronic sensors trainer, the sensor resistance is

$$R_0 = 47000. \ [\Omega] \quad [9.5]$$

when the temperature is at 25 degrees Celsius, or

$$T_0 = 298.15 \ [K] \quad [9.6]$$

Thermistors are typically part of a circuit. In QNET the mechatronic sensors trainer, the thermistor is in the circuit shown in Figure 9.12 and labeled by *R*.

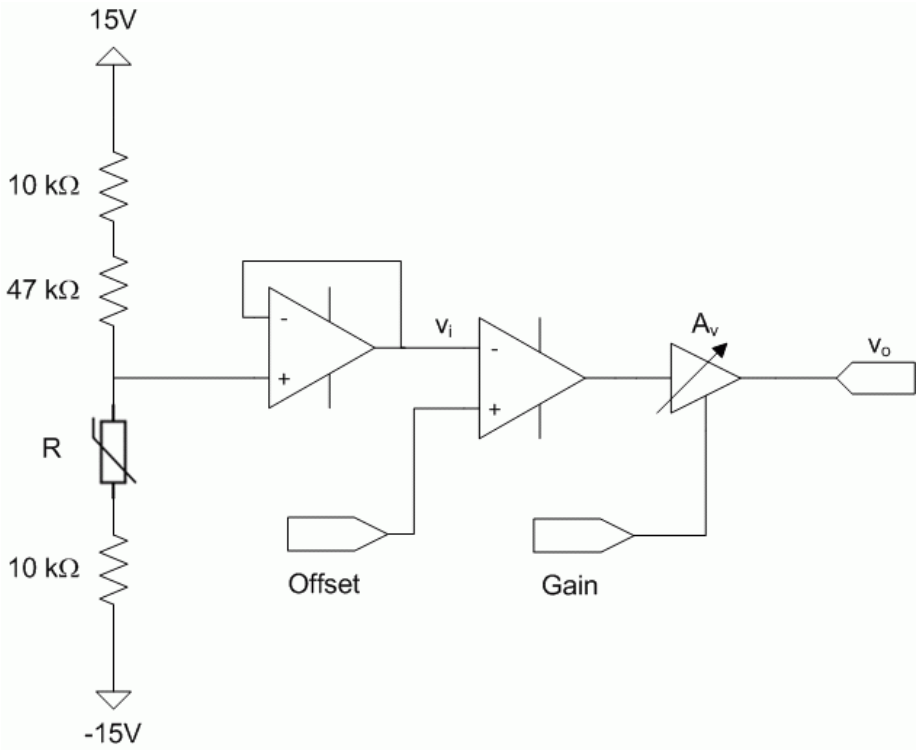


Figure 9.12: Thermistor circuit on QNET mechatronic sensors module.

Using the voltage divider rule, the voltage entering the negative terminal of the second operation amplifier, i.e. the offset op amp, is

$$v_i = \frac{30\ (R + 10000.)}{67000. + R} - 15 \qquad [9.6]$$

.

The output voltage of the circuit is

$$v_o = A_v\ (v_{off} - v_i) \qquad [9.7]$$

,

where v_{off} is the voltage adjusted using the *Offset* potentiometer and A_v is the amplifier gain that can be changed externally using the *Gain* potentiometer. The *Gain* and *Offset* potentiometers are on the QNET mechatronic sensor trainer and shown in Figure 9.13.

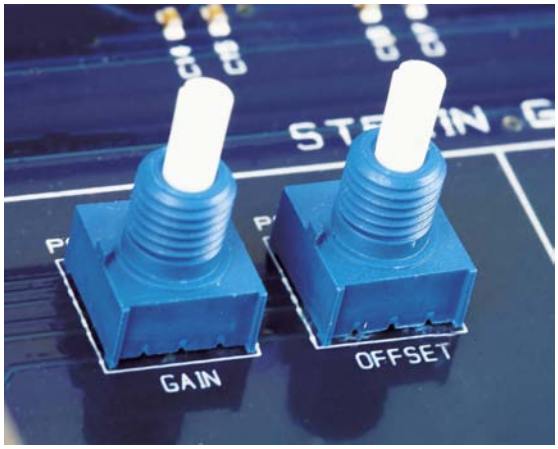


Figure 9.13: Thermistor Gain and Offset potentiometers on QNET mechatronic sensors trainer.

9.2.5. Long Range Distance Sensors

These are non-contact sensors that measure the distance of a target using either sound (sonar) or light (infrared).

9.2.5.1. Sonar Sensor

Often used in mobile robotics, sonar sensors are fitted with an emitter that generates ultrasonic waves and a receiver that captures them after hitting a target. A timer calculates how long it takes for the signal to return and, given the speed of sound in air, the distance of the remote target is measured. The sonar ranger on the mechatronic trainer is pictured in Figure 9.14.

Figure 9.14: Sonar sensor on mechatronic sensors trainer.

Sonar sensors are great for long-distance measurements. For example, the one mounted on mechatronic board can go up to 21 feet. However, in general, these devices do not have good close-range measurements and their resolution can be relatively coarse.

9.2.5.2. Infrared Sensor

Infrared (IR) sensors are widely used in robots, automotive systems, and various other applications that require an accurate, medium-range non-contact position measurement. An IR sensor is typically composed of an infrared emitting diode (IRED), a position sensing detector (PSD), and a signal processing circuit. It outputs a voltage the correlates to the distance of the remote target. The infrared distance measuring sensor on the QNET MECHKIT board is shown in Figure 9.15.

Figure 9.15: IR sensor on QNET mechatronic sensors trainer.

Infrared-based distance sensors typically have a smaller maximum range than sonar but the resolution is better.

9.2.6. Short Range Distance Sensors

In both the sonar and infrared sensors, the target must be a minimum distance away in order for a correct measurement to be taken. These are therefore not suitable in applications where short-range non-contact measurements are required. For these applications, magnetic field and optical position sensors are used to measure the distance of a target in close proximity with the sensor.

9.2.6.1. Optical Position

Optical position sensors are used in applications such as assembly lines, machine automation, and even edge detection in robots. The optical position sensor on the QNET mechatronic sensors board is the black plastic housing located at the bottom of Figure 9.16.

An infrared emitting diode and an NPN silicon phototransistor are mounted side-by-side and are used to measure the position of a target. This sensor has a range of 0.25 inches.

Figure 9.16: Optical position sensor (bottom) and target position knob (top) on QNET mechatronic sensors trainer.

9.2.6.2. Magnetic Field

A magnetic field transducer outputs a voltage proportional to the magnetic field that is applied to the target. The magnetic field sensor is the chip located on the bottom of Figure 9.17. It applies a magnetic field perpendicular to the flat screw head. The position of the screw head is changed by rotating the knob. This magnetic field transducer has a similar range to the optical position sensor.

Figure 9.17: Magnetic field transducer on QNET mechatronic sensors trainer.

9.2.7. Switches

Different applications call for different types of switches. For example, a micro switch may be used to detect mobile robot hitting a wall whereas an optical switch could be used to detect an edge. The push button is the most common type of switch mechanism. A switch that is active high means the output is high, e.g. 5.0 V, when the switch is triggered (e.g. pressed down). Active low means the signal is high, e.g. 5.0 V, when the switch is not engaged (e.g. not pressed down).

The different switches on the QNET mechatronic sensors trainer are introduced in this section followed by a discussion about *debouncing*.

9.2.7.1. Micro Switch

The micro switch is an active low device and is shown in Figure 9.18.

Figure 9.18: Micro switch on QNET mechatronic sensors kit.

9.2.7.2. Push Button

The push button on the QNET MECHKIT is pictured in Figure 9.19 and is active high.

Figure 9.19: Push button on QNET mechatronic sensors trainer.

9.2.7.3. Optical Switch

The optical switch, shown in Figure 9.20, is a photo-microsensor that includes transmissive and reflective components. As opposed to the push button and micro switch, this is a non-contact triggering solution. It is triggered when the reflective sensor does not detect any light, i.e. when an object is placed between the components. It goes low when no object is detected.

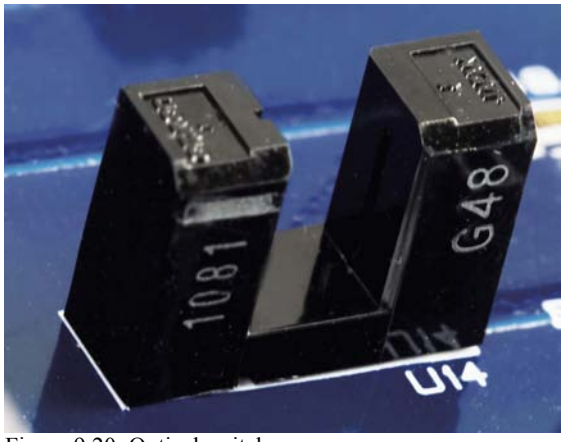


Figure 9.20: Optical switch.

9.2.7.4. Debounce

When implemented digitally, debounce is a type of signal conditioning algorithm that ensures the switch, button, or sensor does not trigger anything due to unexpected conditions.

For example, consider a high-powered cart that is mounted on a track. Proximity sensors are installed that detect when the cart goes beyond a safety limit, in which case the amplifier is deactivated. However due to the high-frequency switching in the motor leads, the proximity switches are sometimes unexpectedly triggered – even when the cart is in the safe zone. The raw signal from the proximity sensor is shown in the top plot of Figure 9.21. To avoid this, the output signal from the sensor is passed though a debounce switch and the resulting signal is shown in the bottom plot of Figure 9.21.

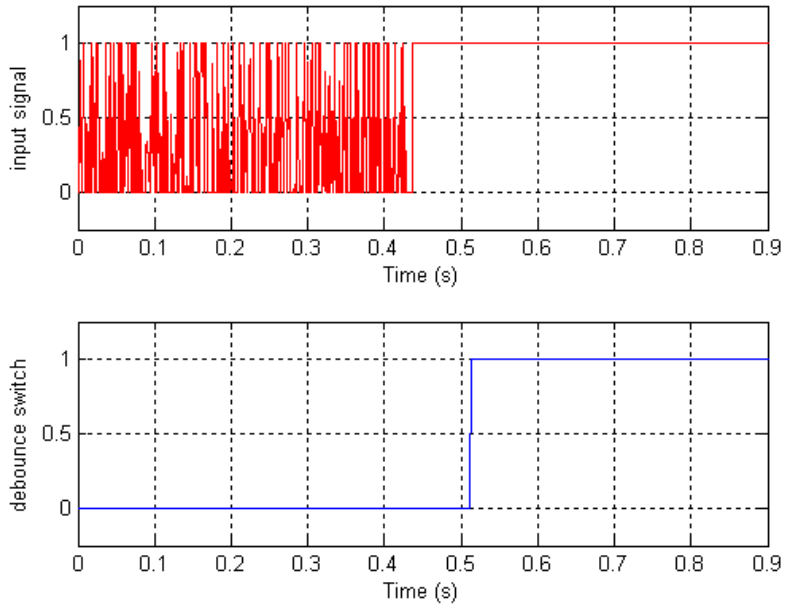


Figure 9.21: Sample debounce response.

9.2.8. Light Emitting Diodes (LEDs)

A light-emitting diode, or LED, is a low-energy and robust indicator that is used in many applications. The LEDs on the mechatronic sensors trainer, labeled LED7 and LED8, are pictured in Figure 9.22. They are connected to digital output lines from which they can be turned on and off. As with switches, LEDs can be wired to be active high or active low.

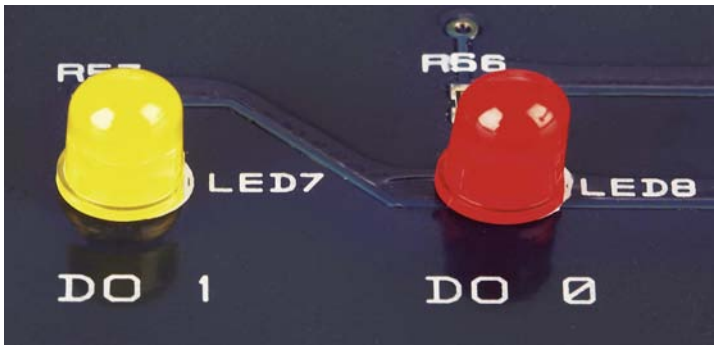


Figure 9.22: LEDs on QNET mechatronic sensors trainer.

10. EMG Signal Processing

Electromyography, or EMG, involves acquiring and studying the electrical activity of muscles. The instrument used to measure the contraction of a muscles is called an *electromyograph* but the term EMG sensor is often used as well. An electromyograph measures the electric potential generated by muscle cells and this recorded voltage is called an *electromyogram*. EMG signals are of interest to the developers of prosthetic devices, such as artificial limbs, and this is called myoelectric prosthesis. EMG is also found in bio-instrumentation, as a clinical diagnostic tool to identify neuromuscular diseases, assisted control in aircrafts, and unvoiced speech recognition.

The QNET Myoelectric trainer shown in Figure 10.1 includes a two-electrode electromyograph with a grounding strap and a servo. The on-board processed EMG signal can be measured and the servo can be driven by the PWM. Through EMG signal processing and control, the clamp on the servo can be opened and closed through muscle contraction, similarly to myoelectric prosthesis.

Figure 10.1: The QNET myoelectric trainer.

10.1. EMG Signal

The electromyogram acquired from the EMG is very qualitative. It depends greatly on how the sensor is placed, how close it is to the muscle, and what muscle is being measured. A typical EMG signal is shown in the first plot of Figure 10.2. As illustrated, EMG signals are very noisy and have a small amplitude, usually ranging around 5 mV. It can contain frequencies ranging from 10 Hz to 1 kHz.

To remove some of the noise, the electrodes on the QNET myoelectric trainer include a differential amplifier as well as a local band-pass filter. See Reference [14] for the common mode rejection ratio (CMRR) and filter specifications of the electromyograph. The EMG signal received from the instrument is isolated and amplified on the QNET myoelectric trainer circuit, as described in Reference [14].

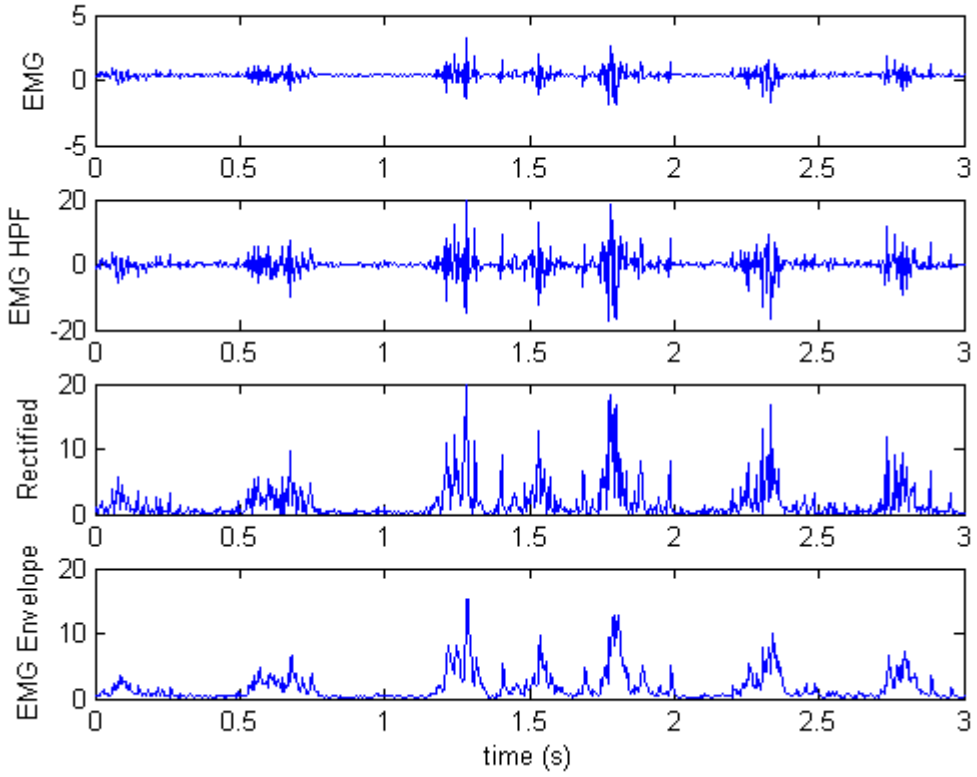


Figure 10.2: Measured and processed EMG signal.

10.2. EMG Processing

The signal acquired from the EMG sensor and amplified through the on-board QNET circuit is shown in the first two blocks of Figure 10.3. This signal is in the +/- 10 V range but still includes a lot of noise. The signal must be processed further, using either an analog circuit or digitally through a PC, in order for it to be used. A signal processing method knows as *linear envelope* is used to do this. As illustrated in Figure 10.3, this involves rectifying the signal and passing it through a low-pass filter. A high-pass filter (HPF) may

also be used to remove any low-frequency components. Choosing the filter cutoff frequency of the high-pass and low-pass filter is important.

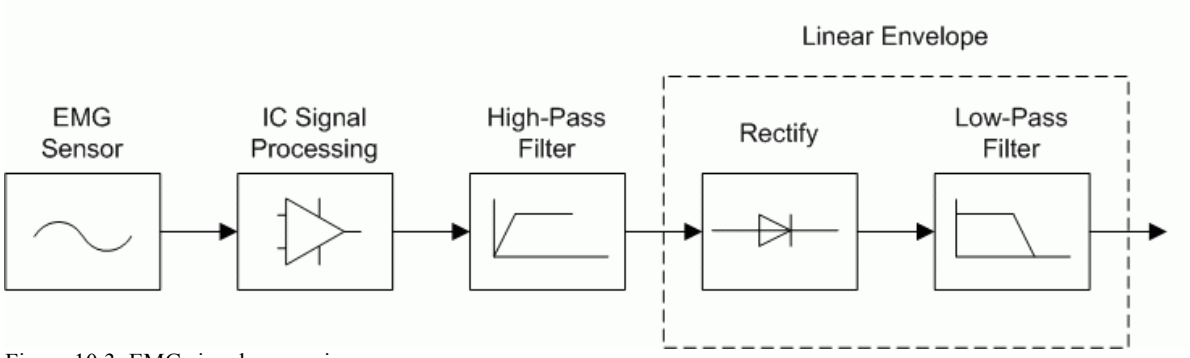


Figure 10.3: EMG signal processing.

The result when the measured EMG signal is passed through a high-pass filter is shown in the second plot in Figure 10.2, above. For the QNET myoelectric, choose a low HPF cutoff (e.g. around 0.25 Hz) to ensure only the DC component is removed and the remainder of the signals are kept. Setting the cutoff too high can make the signal too noisy.

In order to take a "running average" of the EMG the signal is passed through a linear envelope. Rectifying the signal means taking its absolute value. In electronics, a full-wave rectifier circuit is used. As illustrated in the third plot in Figure 10.2, above, the obtained signal is always positive.

The low-pass filter makes the signal smooth and generates the "envelope" of the signal, as shown in the last plot of Figure 10.2. There is a tradeoff when setting its cutoff frequency. If the cutoff is too low, the envelop will be too slow. If its set too high, then the envelope will be less smooth. The resulting signal from the linear envelope can potentially be used to check for muscular failure, rehabilitation, myoelectric prosthesis, and so on.

10.3. Myoelectric Control

The EMG envelope signal discussed in Section 10.2 is used in the control algorithm to open and close the clamp on the servo. The servo is driven by the on-board PWM amplifier.

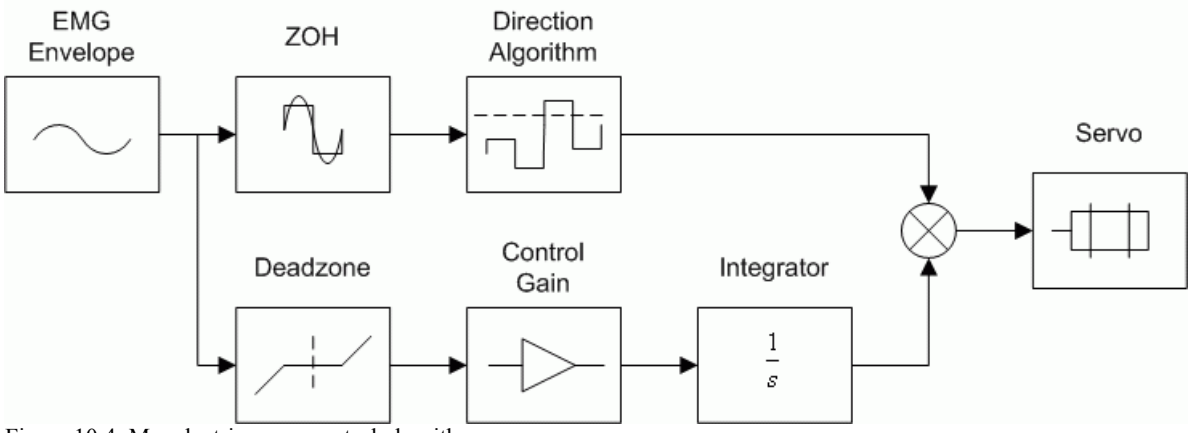


Figure 10.4: Myoelectric servo control algorithm.

In order to open/close the clamps at different positions, this task-based control system has two parts: direction control and position control.

10.3.1. Servo Direction Control

The direction algorithm uses the EMG envelope signal to choose when to open or close the clamps and is illustrated in the top portion of Figure 10.4. To change the direction of the servo, i.e. from open to close or close to open, the signal must exceed a set threshold value for a certain time. Thus when the muscle is contracted more than usual, it should trigger a direction change. For example, if the signal is above the pre-defined threshold value of 0.5 V for at least 0.1 seconds then the servo direction should change.

Zero-order hold (ZOH) is typically used to reconstruct digital signals. It holds its input signal for a specified sampling period. The equation is defined

$$y(t) = \begin{cases} u(t_i) & t - t_i < 0 \text{ and } t_i - t \le 0 \\ u(t_k) & t_k - t \le 0 \text{ and } t - t_{k+1} < 0 \end{cases}, \qquad [10.1]$$

where t is the current simulation time, t_i is the initial simulation time, and for $k = 0, 1, 2, \ldots$ and the ZOH sampling time T_s,

$$t_k = t_i + k\,T_s \,. \qquad [10.2]$$

Figure 10.5, below, shows how the EMG envelope in sampled when passed through a ZOH with a sampling period of 0.1 seconds. If the ZOH signal exceeds a specified threshold, then the current servo direction is reversed.

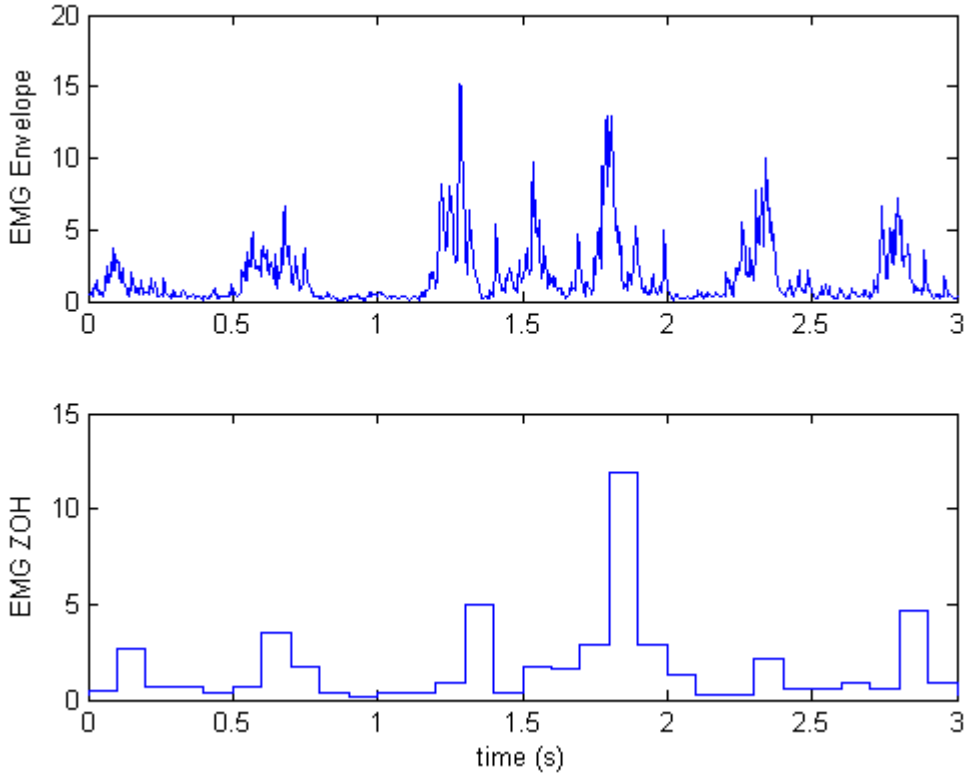


Figure 10.5: Zero-order hold of EMG envelope.

The direction function is then

$$dir = \begin{cases} 1 & \varepsilon < y_{env}(t) \\ -1 & y_{env}(t) \leq \varepsilon \end{cases} \qquad [10.3]$$

where ε is the threshold and $y_{env}(t)$ is the envelope of the EMG signal.

10.3.2. Servo Position Control

The position of the servo/clamp is proportional to the voltage fed to the PWM amplifier that drives it (i.e. no feedback design required). As shown in Figure 10.4, a dead zone is used to remove any small amplitude signals from the EMG envelope that may cause the servo to move from minor muscle contraction. Basically this prevents the servo from drifting when the muscles are at rest.

As shown in Figure 10.4, the envelope of the EMG signal is amplified by an control gain

and passed through an integrator. This generates a voltage that controls the position of the servo. The gain has to be tuned according to the EMG signal. Saturation limits on the integrator ensure the clamp does not close or open passed its limits. The voltage command to the servo can be defined

$$u(t) = \begin{cases} V_{high} & V_{high} < u \\ V_{low} & u < V_{low} \\ \dfrac{dir\, k_i y_{env}(t)}{s} & otherwise \end{cases}, \qquad [10.4]$$

were the upper integrator saturation is V_{high}, the lower integrator limit is V_{how}, the integral gain is k_i, the EMG envelope signal is $y_{env}(t)$, and the *dir* function is defined in Equation [10.3].

11. References

[1] Murray, R.M., Åström, K.J., Boyd, S.P., Brockett, R.W., and Stein, G. Future directions in control in an information rich world. IEEE Control Systems Magazine, 2003, 23:2: pp 20—33.

[2] Murray, R.W. (editor) Control in an Information Rich World. Report of the Panel on Future Directions in Control, Dynamics and Systems. SIAM 2003.

[3] Dorf, R. C Modern Control Systems (10th Edition) Prentice Hall 19XX

[4] Nise, N. S. Control Systems Engineering. Prentice Hall 19XX.

[5] Franklin, G. F. Powell, D. J. and Emami-Naeini, Feedback Control of Dynamic Systems. Prentice Hall 19XX

[6] Ogata, K. Modern Control Engineering, 4th Edition

[7] Åström, K.J. and Östberg, A.-B. (1986) A teaching laboratory for process control, IEEE Control Systems Magazine, 1986, 6:5: pp 37—42.

[8] Åström, K.J. and Lundh, M. (1992) Lund Control Program Combines Theory with Hands-On Experience, IEEE Control Systems Magazine, 12:3, pp 22—30.

[9] Bristol, E.H. (1986) An industrial point of view on control teaching and theory. IEEE Control Systems Magazine, 1986, 6:1: pp 24—27.

[10] Kheir, N.A., Åström, K.J., Auslander, D., Cheok, K.C., Franklin G.F., Masten, M., and Rabins, M. (1996) Control Systems Engineering Education, Automatica, 1996, 32:2, pp 147—166.

[11] Åström, K.J. and Hagglund, T. Advanced PID Control. Instrument Society of America, 2005.

[12] Wikipedia. Thermistor. http://en.wikipedia.org/wiki/Thermistor.

[13] Agilent Technologies. Practical Temperature Measurements (Application Note 290). 2008.

[14] Quanser. QNET User Manual.

[15] William Sellers. Introduction to EMG.

[16] University of Utah. Electromyogram (EMG) Detector with Audiovisual Output.