Control Tutorial

Benjamin Kuipers

January 26, 2004

This introduces some qualitative concepts from control theory, for the student with a basic knowledge of calculus. Our examples will be most closely tied to the domain of mobile robot control See also [Hahn and Edgar, 2001] for a good tutorial tied to chemical process control.

1 The System Being Controlled

A huge variety of systems[1] can be controlled, but we will be thinking mostly about controlling robots, especially mobile robots.

The intrinsic "physics" of the system is represented by the equation

$$\dot{\mathbf{x}} = F(\mathbf{x}, \mathbf{u}), \tag{1}$$

where $\mathbf{x}$ represents the state of the system, $\mathbf{u}$ represents the control input to the system, $\dot{\mathbf{x}}$ represents the time derivative $d\mathbf{x}/dt$ of the system's state.[2]

When F is linear, we assume it is of the form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}. \tag{2}$$

Where the state variable $\mathbf{x}$ has n components, and $\mathbf{u}$ has k components, the matrix $\mathbf{A}$ is $n \times n$ and $\mathbf{B}$ is $n \times k$.

It is often the case that the state $\mathbf{x}$ cannot be directly sensed, but there is sensor output

$$\mathbf{y} = G(\mathbf{x}) \tag{3}$$

that tells us indirectly about the state of the system.[3]

To close the loop, we need a control law

$$\mathbf{u} = H_i(\mathbf{y}) \tag{4}$$

which takes sensor output and determines the motor input to the system.[4] I have written H_i with a subscript to emphasize that the current control law is selected from some larger set of possible control laws.

[1] The system is often called the "plant" by control theorists, revealing the origins of control theory in factory automation.

[2] Bold-faced lower-case letters represent vectors. Bold upper-case letters represent matrices. Non-bold letters represent scalars. For many of our examples, we will assume single-input, single-output (SISO) systems, in which most variables are scalars.

[3] The problem of *observability* is to assess how much information **y** provides about **x**. For our purposes, we will assume that **y** provides plenty of information, and may often be redundant.

[4] Notice that for factory automation, it seems natural to talk about sensor "output" and motor "input", while for a robot it may seem more natural to talk about sensor "input" and motor "output". So it goes.

For a given selection of H_i, the combination of equations (1), (3) and (4) defines a dynamical system

$$\dot{\mathbf{x}} = F(\mathbf{x}, H_i(G(\mathbf{x}))) \tag{5}$$

describing the evolution of the state $\mathbf{x}$ of the system.

Control theory and control engineering are about the design and selection of the controllers H_i, determining the dynamical system that determines the system's behavior. Traditionally, a single H_i is carefully designed to control the system thoughout its behavior, but we will take a more modern approach in which the controller must decide when to replace H_i with some other H_j.

1.1 The Mobile Robot

For a mobile robot moving in the plane, $\mathbf{x} = [x, y, \theta]$ is the robot's state, and $\mathbf{u} = [v, \omega]$ specifies its forward and angular velocities. This model of motor control hides the transient accelerations and decelerations on the way to steady-state velocity.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = F(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix} \tag{6}$$

If we need a more sophisticated model of mobile robot dynamics, we can define the motor commands as forward and angular accelerations $\mathbf{u} = [a, \alpha]$, and expand the state vector to $\mathbf{x} = [x, y, \theta, v, \omega]$.

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = F(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \\ a \\ \alpha \end{bmatrix} \tag{7}$$

For most of our purposes in this tutorial, we will assume that $\mathbf{x}$ is directly measurable. Later, we will consider *observer* processes that estimate $\mathbf{x}$ given $\mathbf{y}$.

2 Bang-Bang Control

The goal of control is to bring a variable x to some desired value or setpoint x_{set}, and keep it there. (This variable could be a state variable, but it might also be some other function of the state.) The primary input to a control law is the *error term*:

$$e = x - x_{set}. \tag{8}$$

A particularly simple way to do this is to test whether the current value is high or low, and then pick an action that will push the system in the direction opposite the error.

$$\begin{array}{llll} \text{if} & e < -\epsilon & \text{then} & u := \mathit{on} \\ \text{if} & e > +\epsilon & \text{then} & u := \mathit{off} \end{array} \tag{9}$$

The constants *on* and *off* are chosen to guarantee that $\dot{x}$ is positive (so x is increasing) or negative (so x is decreasing), respectively. The $\pm\epsilon$ bounds around $e = 0$ are chosen to avoid "chatter", where $x \approx x_{set}$ and the system switches as rapidly as physically possible between *on* and *off*.

The standard household thermostat is a good example of this type of controller. The thermostat itself is slightly more complex than equation (9) because the state variable is the heat content of the house (not directly measurable), while the measurable variable is temperature.

The major disadvantage of the bang-bang controller (so-called because of the sudden changes in u) is that the value of x does not converge to x_{set}, but oscillates continually around it. Repeated discontinuous changes between $u = on$ and $u = off$ can also be damaging to the system.

The major advantage of the bang-bang controller is its simplicity. It essentially responds to $sign(e)$, using it to select between constant actions. However, there is a good deal more information in e, and the controller can take advantage of it.

3 Proportional Control

A proportional controller pushes the system in the direction opposite the error, with a magnitude that is proportional to the magnitude of the error:

$$\begin{aligned} e &= x - x_{set} \\ u &= -k_1 e + u_b \end{aligned} \tag{10}$$

where u_b is set so that $e = 0 \Rightarrow \dot{x} = 0$. That is, the controller pushes harder for larger error, and the system becomes quiescent when the error is zero.

However, in case of a continuing disturbance (which adds a new term into equation (2)) or a change in the setpoint x_{set}, the proportional controller will exhibit *steady-state offset*: reaching quiescence with $e \neq 0$.

The constant k_1, the *controller gain*, can be tuned so the system returns to the setpoint as quickly as desired, without responding so forcefully as to cause other problems.[5]

3.1 Exponential Approach to the Setpoint

As a simple demonstration, let's solve for the explicit behavior of a system combining the proportional controller in equation (10) with a one-dimensional instance of equation (1).

$$\begin{aligned} \dot{x} &= ax + bu \\ &= ax + b(-k_1 e + u_b) \\ &= ax + b(-k_1(x - x_{set}) + u_b) \\ &= -(k_1 b - a)x + (k_1 x_{set} + u_b)b \\ &= -\alpha x + \beta \end{aligned}$$

We use α and β to abbreviate large constant terms to avoid distraction as we solve the differential equation $\dot{x} = -\alpha x + \beta$. For this system to be stable, we must have that $\alpha = k_1 b - a > 0$. (Also recall that C is carried through the derivation as an unknown constant to be set later from initial conditions, so it need not have a consistent value from line to line.)

$$\dot{x} = -\alpha x + \beta$$

[5] *In section N.N, we will show how very high controller gains can conflict with modeling assumptions in other parts of the system and cause unexpected types of behavior.*

$$
\begin{aligned}
\int \frac{\dot{x}\, dt}{-\alpha x + \beta} &= \int 1\, dt \\
-\frac{1}{\alpha} \ln(-\alpha x + \beta) &= t + C \\
\ln(-\alpha x + \beta) &= -\alpha t + C \\
-\alpha x + \beta &= C e^{-\alpha t} \\
x &= C e^{-\alpha t} + \frac{\beta}{\alpha}
\end{aligned}
$$

Note that the constant β/α term ensures that $\dot{x}(\infty) = 0$. Once we plug in the constants, we get the more obscure actual solution:

$$x(t) = Ce^{-(k_1 b - a)t} + \frac{(k_1 x_{set} + u_b)b}{k_1 b - a}.$$

To ensure that $x(\infty) = x_{set}$, we must set $u_b = -\frac{a}{b} x_{set}$.

4 Non-Linear "Proportional" Control

The proportional control law in equation (10), $u = -k_1 e + u_b$, has the advantages of simplicity and analytical tractability. However, we can get other benefits if we generalize it to a class of qualitatively described non-linear control laws:

$$u = -f(e) + u_b \text{ where } f \in M_0^+. \tag{11}$$

M_0^+ is the set of monotonically increasing, continuously differentiable functions passing through $(0, 0)$ [Kuipers, 1994].[6]

Using qualitative simulation [Kuipers, 1994], we can show that any system satisfying the qualitative description

$$\dot{e} = -f(e) \text{ where } f \in M_0^+ \tag{12}$$

will come to rest at $e = 0$ for any $f \in M_0^+$. Since any such function f will work, we can pick the f that has the performance properties we want. Non-linear monotonic functions f provide many options not available with linear functions.

1. If $f(e)$ has a vertical asymptote, response u is unbounded and error e is bounded by the asymptote value.[7]

2. If $f(e)$ has a horizontal asymptote, then response u is bounded while error may be unbounded. This can be an advantage when there is a serious cost to a strong response over long periods of time.

3. The linear controller approaches its setpoint exponentially (perhaps with overshoot and oscillation), reaching quiescence only at $t = \infty$. It is possible for a non-linear controller to reach quiescence at the setpoint at finite time.

4. If a controller is composed from fragments local to neighborhoods in the state space, it may be easier to guarantee weak qualitative properties like monotonicity than to guarantee strong global properties like linearity.

[6] The complete definition of M_0^+ includes a couple of other smoothness conditions.

[7] *But see section N.N where we discuss the problems caused by very high controller gains.*

5 The Stopping Controller

An important basic skill is to stop smoothly and firmly at a desired point.

Suppose that the robot starts at $t = 0$ at position $x = 1$, moving toward the origin at speed $v_0 > 0$, so $v(0) = -v_0 < 0$. We want the robot to stop at $x = 0$, coming to rest at $t = t_s$. State variables y and θ can be ignored.

Assuming that position x is directly sensed and velocity v is directly controlled, we can specify a very simple linear dynamical system:

$$\begin{aligned} \dot{x} &= v \\ v &= -kx \end{aligned}$$

We can solve the equation of motion explicitly, to get $x(t) = e^{-kt}$, which approaches its rest position asymptotically at $t_s = \infty$. Asymptotic approach can be awkward with real motors, which work poorly at very low speeds.

In order to allow the robot to move more quickly to $x = 0$, we want to choose a controller in which $f(x)$ is concave down. An obvious choice is

$$\dot{x} = -k\sqrt{x}. \tag{13}$$

We can solve the dynamical system (13) analytically, with the initial condition $x(0) = 1$, getting the predicted behavior

$$x(t) = (1 - kt/2)^2.$$

This is a parabolic (rather than exponential) drop from $x(0) = 1$ to $x(t_s) = 0$, reaching quiescence in finite time at $t_s = 2/k$.

Differentiating for velocity gives $v(t) = k^2t/2 - k$. The initial condition $v(0) = -v_0$ implies that $k = v_0$, so

$$v(t) = \frac{v_0^2 t}{2} - v_0. \tag{14}$$

Acceleration is constant, at $v_0^2/2$, so the controller behaves like a bang-bang controller, but one in which the constant motor signal is acceleration, not velocity (e.g., equation (7)). The acceleration value can be compared with the maximum permissible acceleration of the robot, to see whether the stopping goal is feasible.

Although $\dot{x} = -v_0\sqrt{x}$ has the qualitative properties we wanted, we may be able to find an even better shape for the response curve than $\sqrt{x}$, for example by training a neural net function approximator.

Implementation note: We have assumed that velocity can be directly and instaneously influenced, which is only an approximation. All we actually need is the ability to achieve a constant deceleration of $v_0^2/2$. It may also be necessary to have a special method to handle the final moment of stopping, to prevent velocity from overshooting zero.

6 Types of Control Loops

Feedback Control The straight-forward implementation of the stopping control law (13) is to sense $x(t)$, and update $\dot{x}(t)$ appropriately, at each time-step t.

This feedback control law is robust against sensor and motor errors, but has the cost, both in computation and in sensory delay, of sensing the world at each time step.

Open-Loop Control Since we can explicitly solve for the time-course of the control signal $v(t)$, given observations of $x(0)$ and $\dot{x}(0)$, we can simply follow equation (14) as a fixed policy, independent of further observations.

The open loop formulation also avoids any problems that might arise because $f'(x) \to \infty$ as $x \to 0$.

While the open loop control law clearly minimizes sensor effort and delay, it leaves the robot without feedback to compensate for sensor and motor errors.

Model-Predictive Control We can get the best of both worlds by repeating the derivation of equation (14) every time we get new observations of $x(t)$ and $\dot{x}(t)$.

If sensing is slow compared with the motor control loop, the open-loop policy provides an output signal to send whenever the control loop needs one. But if new sensory information becomes available, the model (equation (14)) can be updated, compensating for sensory or motor errors, or for inaccuracies in the model.

7 Adaptive Control

The behavior of a controller is determined by the values of certain constant parameters such as k, x_{set}, and u_b for the P-controller. An *adaptive controller* includes a slower control loop that adjusts the value of a parameter over a longer time-scale.

Once a proportional controller is tuned by setting k and u_b for a particular situation, there are certain kinds of changes it still cannot compensate for. For example, suppose I have a P-controlled heater that keeps my house near its setpoint temperature during the winter. The tuning parameters take into account the amount of heat that is lost through the walls of my house. If someone leaves a window open, the amount of heat loss will increase, so the control signal to the heater must increase to compensate. The P-control law can only increase the control signal if the error term e increases, so the new system reaches equilibrium at some distance from the setpoint, in order to have a non-zero error term. This is *steady-state offset*.

We can solve this problem by having a secondary control loop that "adapts" the parameter u_b in the basic loop at a slower rate. This transforms the P-controller to

$$\begin{aligned} u &= -k_1 e + u_b \\ \dot{u_b} &= -k_2 e \text{ where } k_2 \ll k_1 \end{aligned} \tag{15}$$

Since k_2 is small relative to k_1, u_b changes more slowly than x. Recall that in the P-controller, u_b was set to ensure that $e = 0 \Rightarrow \dot{x} = 0$. In the adaptive controller (15), the system reaches quiescence (i.e. $\dot{x} = \dot{u_b} = 0$) only when $e = 0$ and the value of u_b is such that $\dot{x} = 0$. In the home-heater example above, when the window is left open, $\dot{x}$ approaches zero when e is still significantly negative. The adaptive control loop then causes $\dot{u_b} > 0$, so u_b and hence u are gradually increased until they compensate for the added heat loss. When the window is closed, a persistent high value of e will decrease u_b again.

"Habituation" is the phenomenon when the human body, often the sensory system, gets used to a certain stimulus and responds progressively less. We can emulate this phenomenon quite easily by applying the adaptive control loop to x_{set} rather than u_b.

$$\begin{aligned} u &= -k_1 e + u_b \\ \dot{x_{set}} &= +k_h e \text{ where } k_h \ll k_1 \end{aligned} \tag{16}$$

If e is persistently high, then the slower habituation loop will increase the setpoint x_{set}, which in turn decreases the error term. The same argument about quiescence applies to both adaptive controllers. Habituation is valuable when the cost of the control respnse may be larger than the cost of the error. This arises very clearly in the complex of different control loops involved in regulation of blood pressure [Guyton, 1991]. The body takes dramatic action to avoid short-term loss of blood pressure, but those actions can be very damaging over the long term, so they habituate away.

8 Proportional-Integral (PI) Control

The adaptive controller of equation (15) can be considered a special case of a proportional-integral (PI) controller. Note that

$$\dot{u}_b = -k_2 e \text{ implies that } u_b(t) = -k_2 \int_0^t e\ dt$$

which allows us to rewrite equation (15) as:

$$u = -k_1 e - k_2 \int_0^t e\ dt \tag{17}$$

The PI controller avoids steady-state offset, but can exhibit *overshoot* and *oscillation*. Divergent oscillation is a dramatic failure mode for a controlled system. However, a properly tuned PI controller will usually exhibit some overshoot and oscillation. If it has no oscillations, it is probably not tuned aggressively enough.

9 Proportional-Derivative (PD) Control

Viscous friction (damping) opposes motion with a force that increases with velocity (perhaps non-linearly). In case a controlled system has a tendency to overshoot, we can add a derivative term to provide "frictional damping" on the motion of the system, decreasing and perhaps eliminating overshoot. See [Kuipers and Ramamoorthy, 2002] for examples of positive and negative damping in control laws for pumping up a pendulum and balancing it in the inverted position.

$$u = -k_1 e - k_3 \dot{e} \tag{18}$$

A problem is that estimating the derivative term from sensor input is particularly vulnerable to noise in the sensor readings.

10 Proportional-Integral-Derivative (PID) Control

Combining all three terms gives the PID controller, which is the workhorse of industrial control engineering.

$$u = -k_1 e - k_2 \int_0^t e\ dt - k_3 \dot{e} \tag{19}$$

Clearly, PI and PD controllers are special cases of the PID controller. In most texts, the parameters in a PID controller have special significance, and are not as cleanly separated as k_1, k_2, and k_3. There are many sophisticated methods for tuning the constants in PID controllers.

11 Spring Models and their Properties

We can get insight into the behavior of simple dynamical systems, and learn how to tune simple controllers, by considering the familiar mass-spring system. The key fact about springs is Hooke's Law, which says that the restoring force exerted by a spring is proportional to its displacement from its rest position. If x represents the spring's displacement from rest, then

$$F = ma = m\ddot{x} = -kx.$$

This model is friction-free. A damping friction force adds a term proportional to $\dot{x}$ and opposite in direction, and gives

$$F = ma = m\ddot{x} = -kx - k_f\dot{x}.$$

Rearranging and combining constants, we get the linear spring model:

$$\ddot{x} + b\dot{x} + cx = 0. \tag{20}$$

Any elementary differential equations text shows that the solutions to equation (20) are all of the form:

$$x(t) = Ae^{r_1 t} + Be^{r_2 t} \tag{21}$$

where r_1 and r_2 are the roots of the *characteristic equation* $x^2 + bx + c = 0$ derived from (20). Thus,

$$r_1, r_2 = \frac{-b \pm \sqrt{b^2 - 4c}}{2}. \tag{22}$$

We can gain insight into the qualitative properties of the solutions to equation (20), and hence to the behavior of some controllers, by considering the nature of the roots r_1 and r_2, and the interpretation of the parameters b and c.

- For the system to converge to $(x, \dot{x}) = (0, 0)$, both roots r_1 and r_2 must have negative real parts. This requires that both $c > 0$ and $b > 0$.
- $c > 0$ means that the system has a restoring spring force obeying Hooke's Law, so it tends to bring the mass back to the rest position. When $c < 0$, the system behaves like an "anti-spring", with an outward force that increases with distance, so the mass diverges to infinity.
- $b > 0$ means that the system has a damping force proportional to velocity and opposing it in direction. Along with $c > 0$, the model behaves like a damped spring. Exactly how it behaves depends on the sign of the discriminant $D = b^2 - 4c$ in equation (22).
  - For sufficiently small values of b, such that $D = b^2 - 4c < 0$, the system is *underdamped*, oscillating but with decreasing amplitude, asymptotically converging to the rest position. Whether it converges or not, the system will oscillate if r_1 and r_2 have non-zero imaginary parts.
  - For sufficiently large values of b, such that $D = b^2 - 4c > 0$, the system is *overdamped*, so x moves slowly to the rest position (as if through molasses) without overshooting.
  - For particular values of b and c on the boundary between these two regions, $D = b^2 - 4c = 0$, the system is *critically damped*, meaning that it moves as quickly as possible to the rest position without overshooting.

- If $b < 0$, the system has negative damping: it is driven by an "anti-friction" force that is proportional to velocity and in the same direction. If there is a regular spring restoring force ($c > 0$), then the "anti-friction" force can pump the system to higher and higher (divergent) oscillations. Otherwise, the system simply diverges.

- The undamped spring ($c > 0$ but $b = 0$) simply exhibits periodic oscillation.

[8]

In order to tune a second-order system, we formulate it in the damped spring framework, and set the parameters to make it critically damped.

11.1 The Monotonic Damped Spring Model

Just as we did in section 4, we can generalize the linear terms in equation (20) to possibly non-linear monotonic functions.

$$\ddot{x} + f(\dot{x}) + g(x) = 0, \text{ where } f, g \in M_0^+ \tag{23}$$

We can show by qualitative simulation [Kuipers, 1994] that any system satisfying this qualitative model must converge to a stable fixed-point at $(x, \dot{x}) = (0, 0)$. The linear model (20) has only two qualitatively distinct classes of solutions: the direct convergence of the overdamped and critically damped cases, and the infinite decreasing oscillations of the underdamped case. However, the non-linear model also has solutions with any finite number of half-cycles of oscillation before overdamped convergence to $(0, 0)$.

11.2 Eigenvalues

The intuitions based on damped springs and their parameters a and b are limited to second-order systems whose models can be formulated to look like equation (20) or (23). However, the analysis of the roots r_i of the characteristic equation can be generalized to higher-order systems.

An *eigenvalue* of a linear system $\mathbf{y} = \mathbf{A}(\mathbf{x})$ is a scalar value λ for which there is some vector $\mathbf{v}$ such that $\mathbf{A}(\mathbf{v}) = \lambda\mathbf{v}$. The eigenvalues for a matrix $\mathbf{A}$ are the solutions λ to the characteristic equation:

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0.$$

An *eigenvector* for a given eigenvalue λ is a vector $\mathbf{v}$ for which $\mathbf{A}(\mathbf{v}) = \lambda\mathbf{v}$.

Qualitative properties of a controller can be determined from its eigenvalues. The two properties we have seen earlier that generalize are:

- If all eigenvalues have negative real parts, the system is stable.

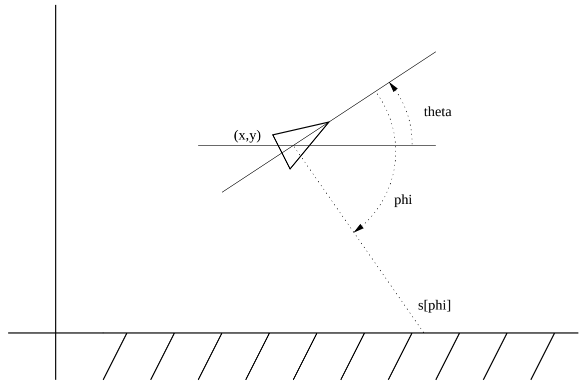
- If any eigenvalue has non-zero imaginary part, the system will oscillate.

Many sophisticated ways to tune controllers and other dynamical systems use the properties of the system's eigenvalues.

[8] *Need b-c phase diagram for qualitative behaviors, with separating curve.*

12 The Wall-Following Control Law

One of the most important robot control laws allows the robot to follow a wall or corridor. It assumes that the robot is moving forward at constant linear velocity v, and sets angular velocity ω in response to positional error $e = y - y_{set}$ and orientation error θ. The same control law can be used to follow the midline of a corridor by defining $e = d_{right} - d_{left}$. This is a slightly simplified version of the wall-following controller from [van Turennout *et al.*, 1992].



The robot is at position (x, y) and orientation θ. The range sensor in direction ϕ senses distance s_ϕ, but in this paper we assume that y and θ are sensed directly.

We assume the following model of the robot (equation (6)).

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = F(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} v\cos\theta \\ v\sin\theta \\ \omega \end{bmatrix}$$

We want to define the control law so that the behavior of the system will be described by a "damped spring" model:

$$\ddot{e} + k_\theta \dot{e} + k_e e = 0. \tag{24}$$

where the constants k_e and k_θ are tuned to make the system behave well (i.e., critically damped convergence $e \to 0$).

For small values of θ,

$$\begin{aligned} \dot{e} &= v\sin\theta \approx v\theta \\ \ddot{e} &= v\cos\theta\,\dot{\theta} \approx v\omega \end{aligned} \tag{25}$$

By substituting (25) into equation (24) and solving for ω as a function of the values of the observed variables e, θ and v, we get the wall-following control law:

$$\omega = -k_\theta \theta - \frac{k_e}{v} e. \tag{26}$$

While this is not particularly aggressive[9], we can force equation (24) to be critically damped by requiring

$$k_e = \frac{k_\theta^2}{4} \text{ or equivalently, } k_\theta = \sqrt{4k_e}. \tag{27}$$

[9] *Provide an explanation of proper tuning methods.*

13 Further Readings

- Pumping and balancing the inverted pendulum: [Kuipers and Ramamoorthy, 2002].
- Nonlinear robotics control: [Slotine and Li, 1991].
- State space analysis: [Brogan, 1990].
- Optimal control: [Lewis and Syrmos, 1995].
- Process control: [Seborg *et al.*, 1989].

References

[Brogan, 1990] W. L. Brogan. *Modern Control Theory*. Prentice Hall, Englewood Cliffs, NJ, third edition, 1990.

[Guyton, 1991] A. C. Guyton. Blood pressure control – special role of the kidneys and body fluids. *Science*, 252:1813–1816, 1991.

[Hahn and Edgar, 2001] J. Hahn and T. F. Edgar. Process control systems. In R. A. Meyers, editor, *Encyclopedia of Physical Science and Technology*. Academic Press, NY, third edition, 2001.

[Kuipers and Ramamoorthy, 2002] B. Kuipers and S. Ramamoorthy. Qualitative modeling and heterogeneous control of global system behavior. In C. J. Tomlin and M. R. Greenstreet, editors, *Hybrid Systems: Computation and Control*, volume 2289 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2002. http://www.cs.utexas.edu/users/qr/papers/Kuipers-hscc-02.html.

[Kuipers, 1994] B. J. Kuipers. *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*. MIT Press, Cambridge, MA, 1994.

[Lewis and Syrmos, 1995] F. L. Lewis and V. L. Syrmos. *Optimal Control*. John Wiley, NY, second edition, 1995.

[Seborg *et al.*, 1989] D. E. Seborg, T. F. Edgar, and D. A. Mellichamp. *Process Dynamics and Control*. John Wiley, NY, 1989.

[Slotine and Li, 1991] J.-J. Slotine and W. Li. *Applied Nonlinear Control*. Prentice Hall, Englewood Cliffs NJ, 1991.

[van Turennout *et al.*, 1992] P. van Turennout, G. Honderd, and L. J. van Schelven. Wall-following control of a mobile robot. In *IEEE International Conference on Robotics and Automation*, pages 280–285, 1992.