A SIMPLE AND ROBUST HIERARCHICAL CONTROL SYSTEM FOR A WALKING ROBOT

J.R. Kennaway

Keywords: walking robot, hierarchical control, HPCT

Abstract

The present work applies a control architecture proposed by W.T. Powers to several problems in robotics, and suggests that it may have wide practical applicability. The architecture is called (Hierarchical) Perceptual Control Theory, or HPCT, and was proposed by Powers as a possible organisation for living control systems.

1 Introduction

W.T. Powers proposed a certain hierarchical control architecture as a possible organisation for living control systems [3, 4]. We here apply this architecture to some problems in the control of mechanical devices, the primary example being a walking robot. The resulting designs display a high degree of both robustness and simplicity, yet are capable of solving complex multivariable control problems.

2 Example: the inverted pendulum

We shall first illustrate the approach by way of a standard example in control theory, the inverted pendulum shown in Figure 1. We assume that the cart travels on a frictionless track, the pendulum swivels freely at its base through $360^\circ$, and there is an actuator which applies any specified horizontal force to the cart. To keep the pendulum upright and move it to a specified horizontal position by means of this actuator is a complicated task. Nevertheless, it can be achieved by breaking the matter down into simpler tasks. The following design was devised by Powers.

If we had an actuator that could set the bob immediately to any desired position, no control system would be necessary. We don’t have such an actuator; but if we had one which could set the pendulum bob’s horizontal velocity, we could use this to control its position: set the velocity equal to $k_0(r_b - b)$ for some constant $k_0$, where $r_b$ is the demanded position and $b$ is the current position. We don’t have such a velocity actuator, but if we had an actuator that set the bob’s acceleration, we could control the velocity $\dot{b}$ to approach a reference value $r_\dot{b}$ by applying an acceleration $k_1(r_\dot{b} - \dot{b})$. When the pendulum is close to vertical, the acceleration is proportional to the pendulum angle, which is proportional to $\theta = \angle c$, where $c$ is the position of the cart. So we can set the acceleration by setting $\theta$.

We cannot set $\theta$ directly, but we could control it if we could set the cart’s velocity, by setting $\dot{c} = k_2(r_c - c)$, where $r_c$ is the reference cart position. We cannot set $\dot{c}$ directly, but we could control it if we could set the cart’s acceleration: $\ddot{c} = k_3(r_c - \dot{c})$, where $r_c$ is the demanded cart velocity. Finally, we can set the cart’s acceleration by applying a force to the cart, which by hypothesis we are able to do.

The resulting arrangement of four proportional controllers is shown in Figure 2. For suitably chosen values of the gain parameters, it is very stable and robust, although applying a large enough step function to the top-level reference input can make it lose control. Bounding the output of the bob velocity controller (to prevent it demanding an excessive pendulum angle) eliminates this. Although the controller is designed on the assumption of linearity, the physical simulation uses the true differential equations, valid for all pendulum angles. Letting the masses of cart and bob be $M$ and $m$, the force on the cart $f$, an external disturbing force on the bob $d$, and the angle of the pendulum from the vertical $\theta$, the pendulum is described by Equation (1).

$$\begin{align*}
\ddot{c} \cos \theta + l \ddot{\theta} &= g \sin \theta + (d \cos \theta)/m \\
\ddot{c}(M + m) + ml \ddot{\theta} &= f + d + ml \dot{\theta}^2 \sin \theta
\end{align*}$$

(1)

Figure 3 shows the response to a step change in bob reference.
2.1 Analysis of a simple 2-level controller

As an example of the mathematical analysis of these systems, we illustrate a two-level controller, consisting of the bottom two levels of the pendulum controller. The position $x$ is controlled by setting a reference for $\dot{x}$, which is controlled by setting a force, which determines $x$ by Newton’s second law. Letting $x_r$ be the reference position and $v$ the reference velocity, the controllers are described by Equation (2), and the resulting relation between $x$ and $x_r$ is given by Equation (3).

$$v = k(x_r - x)$$
$$m\ddot{x} = k'(v - \dot{x})$$ (2)

$$m\ddot{x} + k'\dot{x} + kk'x = kk'x_r$$ (3)

This is identical to the equation of damped harmonic motion, although there are no physical springs involved. If we write $\rho = k'/mk$ (the ratio of the time constant $1/k$ of the upper controller to $m/k'$, that of the lower controller), then the roots of the characteristic equation are $\frac{1}{2}k(-\rho \pm \sqrt{(\rho - 2)^2 - 4})$. For large $\rho$, these tend to $-k$ and $-k(\rho - 1)$. For $\rho = 4$, the roots coincide at $-2k$, and as $\rho$ approaches zero, they describe circular arcs in the complex plane as in Figure 4. The fastest response is obtained for $\rho$ equal to 4 (defined as the value giving the most negative least negative root).

2.2 Cascade control

The above control scheme closely resembles a standard configuration in process control known as cascade control, although the motivation is somewhat different. In cascade control, as usually motivated, where a single controller produces unacceptable performance, due to the chosen actuator having only a slow effect on the controlled variable, a second controller is introduced which controls some variable (called the secondary variable) which has a more rapid effect on the primary controlled variable. The output of the primary controller connects to the secondary reference (set-point) input, and the secondary output connects to the actuator. Cascades of three or more controllers are possible, but typical practice employs just two controllers.

We find it more instructive to consider each controller as providing a virtual actuator to the next controller up, as suggested by our description of the inverted pendulum controller. In addition, the hierarchical arrangement is capable of much wider application, which we will demonstrate in the main example of this paper.

The value of 4 that we found above for the ratio of upper to lower level time constant agrees with a standard rule of thumb for cascade design, that the secondary controller should have a response about 4 times as fast as the primary.

3 A simplified robot

We shall generalise the above architecture to allow multiple controllers at each level. A minimal example is provided by a simple robot with two legs and one degree of freedom in each leg (Figure 5). Each leg is a linear actuator which has a length $x$ or $y$, and exerts a vertical force $f_x$ or $f_y$. The body has two degrees of freedom of movement: height ($h$) and pitch ($p$). We assume the centre of the body is constrained to a vertical line, and that the pitch remains small, so that we can linearly approximate the kinematics and dynamics by Equation (4).

$$h = \frac{(x + y)}{2} \quad p = \frac{(y - x)}{l}$$

$$\ddot{h} = \frac{(f_x + f_y)}{m} \quad \ddot{p} = \frac{(f_y - f_x)}{I}$$ (4)
To simplify things, we choose units so that the mass $m$ of the body is 1, and its length $l$ and moment of inertia $I$ are 2. There are four proportional controllers, arranged according to the network of Figure 6. The lower level controllers control $\dot{x}$ and $\dot{y}$, and the upper level controls $h$ and $p$. For simplicity we will take the reference inputs $r_h$ and $r_p$ to be zero (measuring $h$, $x$, and $y$ relative to some convenient point above the ground). This results in Equation (5).

\[
\begin{align*}
\dot{x} &= -k_h h + k_p p \\
\dot{y} &= -k_h h - k_p p \\
\dot{h} &= k_x (\dot{x} - \dot{\dot{x}}) \\
\dot{p} &= k_y (\dot{y} - \dot{\dot{y}})
\end{align*}
\]  

(5)

By symmetry it is reasonable to choose $k_h = k_p$ and $k_x = k_y$. By rescaling of time we can choose $k_h = k_p = 1$. Writing $k$ for $k_x$ and $k_y$ we obtain Equation (6), which is the equation for damped harmonic motion for each variable. The optimal value for $k$ is 2.

\[\ddot{h} + 2k \dot{h} + 2kh = 0 \quad \ddot{p} + 2kp + 2kp = 0\]  

(6)

It is instructive to consider what happens if we change the linkage matrix. If we replace the first item of Equation (5) by Equation (7) (that is, replacing $l$ by $\alpha$ in Figure 6 on the line from the $h$ controller to the $\dot{x}$ controller) then Equation (6) for $h$ and $p$ is replaced by Equation (8).

\[
\begin{align*}
\dot{x} &= -k_h \alpha h + k_p p \\
\dot{h} &= k_x (\dot{x} - \dot{\dot{x}}) \\
\dot{p} &= k_y (\dot{y} - \dot{\dot{y}})
\end{align*}
\]  

(7)

\[
\begin{align*}
\ddot{h} + 2k \dot{h} + kh(\alpha + 1) &= 0 \\
\ddot{p} + 2kp + 2kp &= kh(\alpha - 1)
\end{align*}
\]  

(8)

When $\alpha = 1$, this is the original system. For $\alpha = 0$, the height and pitch control interact, but both still reach their reference value. As $\alpha$ approaches $-1$, the response time of the height controller becomes longer and longer. When $\alpha = -1$, the linkage matrix is singular, which means that the height and pitch controllers are attempting to control different variables by means of identical actions. The result is that disturbances to the height of the robot are not controlled. For $\alpha < -1$, the system is unstable.

4 A backhoe excavator

A backhoe excavator, such as that of Figure 7, has three joints, each actuated by a hydraulic cylinder. If one wishes to drive the bucket in a straight horizontal line, keeping it in a constant orientation, one must operate all three actuators in a rather complex way. A control system whose controlled variables are the reach, lift, and inclination of the bucket can provide the operator with the ability to directly drive the bucket straight forwards, backwards, up, and down. It is straightforward to apply the architecture described above to the task, and we have done so, simulating the physics by means of the Vortex physical simulation library1. Some screen-captured movies of the resulting simulations are available at http://www.cmp.uea.ac.uk/~jrk/Robotics/digger.

The control architecture is similar to that of the two-legged robot, but with three proportional controllers at each level. The upper level controls the bucket reach, lift, and slope. The lower level controls the rates of change of the three joint angles. Each actuator is assumed to supply a specified torque to the joint. (We have not modelled actual hydraulic actuators.) The outputs of the top level are linearly combined to give the references of the bottom level.

The linkage matrix connecting the top-level outputs to the bottom-level references is more complicated than for the robot, as it depends on the current state of the machine. When the bucket is far, increasing the dipper angle will raise it, but when it is close, increasing the dipper angle lowers it (Figure 8). When the dipper is vertical, it has no effect on bucket height. The routing of the output from the height controller to the reference input of the dipper controller must therefore depend on the current configuration. For each top-level variable $x$, and

---

each joint \( y \), we have chosen the corresponding element of the linkage matrix to be \( \frac{\partial x}{\partial y} \). That is, the more effect an actuator has on a top-level controlled variable, the more that actuator will be used to control it. The precise values are found not to be critical.

An obvious generalisation of this design is to a backhoe with more joints in its arm. With four joints, as in Figure 9, there is an extra degree of freedom which is not constrained by the position and orientation of the bucket. A simple way of fixing the extra degree of freedom is to add another top-level controller whose controlled variable is \( a_3 - a_2 \) (the difference between the exterior angles at the second and third joints), or more generally, \( a_3 - ka_2 \) for some constant \( k \). This forces joints 2 and 3 to have similar exterior angles, avoiding either of the extreme configurations shown in the Figure. While a four-jointed backhoe may not necessarily be a more practical excavation machine than the standard three-jointed configuration, there are applications to robotic arms requiring extra joints to reach into confined spaces. Arms with five or more joints can be controlled similarly, using extra equalisation controllers to fix all the extra degrees of freedom. Such arms may be beyond the capacity of a human controller to operate effectively by directly driving the joints.

![Figure 8: Dependence of linkage matrix on configuration](image)

5 A four-legged walking robot

This is the main example of this paper. While the previous examples are simple case studies illustrating the method, we believe the robot architecture described here is capable of practical realisation, although as yet it exists only in simulation.

We have constructed a physical simulation of a walking robot (see Figure 10) with four or more legs, and a two-level control hierarchy. At the upper level there is one controller for each degree of freedom of the robot’s body. We assume that its can perceive the height of its body above the ground; perceptions of its other five degrees of freedom are assumed to be defined in terms of the positions of all its feet relative to its body. At the lower level, there is a controller for each of the three degrees of freedom of each leg: one at the knee and two (pitch and yaw) at the shoulder. Each of the lower-level controllers controls the rate of change of joint angle; its output is the torque applied to the joint. These are all proportional controllers.

Each top level controller’s output is directed to some subset of the lower level controllers’ reference inputs. Thus the reference of each lower level controller is a weighted sum of the outputs of the upper level controllers. For an \( n \)-legged robot, the weightings are given by a \( 6 \times 3n \) linkage matrix, in which all of the elements are 1, 0, or \(-1\). (The behaviour of the robot is insensitive to the exact values.) For this example, it is sufficient to choose the weights by straightforward physical intuition. For the robot to lift its body higher, it must decrease the pitch angle at each shoulder. To swing its body to the left, it must swing each shoulder joint to the right. To sway the body towards the right, it must decrease the knee angles on the right, and increase those on the left. And similarly for the other three degrees of freedom: pitch, roll, and forward sway. Figure 11 illustrates the response to a height reference increasing at 2 m/s. (The kink in the shoulder pitch plot at the end of the movement is due to the robot momentarily leaving the ground in its effort to decelerate from 2 m/s to zero.)

![Figure 9: Four-joint backhoe](image)

![Figure 10: Four-legged robot. The blobs are virtual food particles.](image)

![Figure 11: Response of the robot to an increasing height reference. Solid line: reference height (metres). Dashed line: actual height (metres). Dotted line: average angular velocity in shoulder pitch joints (rad/sec).](image)

The architecture works equally well for six, eight, or more legs (the linkage matrix having a general definition that is uniform
in the number of pairs of legs). The robot can continue to stand and balance against disturbing forces even when a leg is removed. The values of the linkage matrix are not critical. One can even replace a few of them by random values and obtain a system that controls almost as well. Provided that there are sufficient degrees of freedom at the lower level, and that different body controllers do not both try to use the same set of signals to the lower level in order to control different perceptions, it is possible for all of the top level controllers to simultaneously achieve good control, despite their interactions.

It is important to the functioning of the robot that the body controllers do not try to set the joint angles, but only their velocities. Computing the angles required to produce a given posture of the body requires complicated inverse kinematic calculations and sensing of the terrain, and errors in the data on which these calculations depend would result in errors of comparable size in the body parameters. The control scheme described avoids this problem by having the body controllers demand rates of change for the joint angles. The negative feedback action of the controllers ensures that the joint angles will arrive at whatever values are required, with residual errors depending only on the tuning of the controllers and the accuracy with which the controlled variables are measured.

As an indication of the simplicity of the control scheme, in a physical construction (which would dispense with the physics simulation code), the most complicated calculations would be the forward kinematic computation of the body position relative to the feet. There is no motion planning, inverse kinematics, learning, adaptation, or modelling by the robot of its environment. The robot is described in more detail in [2].

5.1 Walking and navigation

The four-legged robot as described so far stands and balances on four or more legs, while subject to random external forces. To make it walk, all that is required is for it to repeatedly lift some subset of its legs, swing them forwards, and put them down again. The controller for forwards position relative to the footprint will then pull the body forwards. On uneven terrain, the body pitch and roll controllers will keep the body aligned with the plane of the footprint. Similarly, to turn anticlockwise, it repeatedly lifts a subset of legs, swings them anticlockwise, and puts them down, letting the body heading controller bring the body into alignment with the new footprint. By these means, the robot is able to walk and turn on uneven terrain, and go up and down (shallow) stairs. Adding rudimentary senses to detect the direction of a landmark enables it to navigate towards it by varying the size of the walk and turn actions.

This robot has been implemented in a simulation that can be run from http://www.cmp.uea.ac.uk/~jrk/Archy/Archy.html. The simulation includes an implementation of the dynamics of a rigid body (the robot body excluding the legs), acted on by forces exerted by the legs between the body and the ground. The dynamics of legs lifted from the ground has not been modelled; nevertheless, the control problem, although simplified from reality, is a complex control problem in its own right, which the control architecture we have described is able to solve.

5.2 Improved physical simulation

The physics engine underlying the robot simulation described above was deliberately simplified in order to alleviate the programming task. Using the Vortex physical simulation library, which can handle arbitrary linkages of three-dimensional bodies, we have begun to develop more faithful simulations with the goal of achieving sufficient physical fidelity to justify actual construction. Thus far, we have duplicated the original robot and its ability to control the six degrees of freedom of its body in the presence of disturbing forces and commanded changes of references. (Figure 11 is in fact plotted from data obtained from this more recent version of the robot.) Reimplementing walking for this more accurate simulation is the subject of future work. The control of a leg whose foot is in the air can be achieved by means similar to the backhoe model.

6 The general architecture

A general architecture for the two-level multivariable controllers discussed in this paper is shown in Figure 12, simplified for the present exposition. All lower-case variables are vectors of length $n$, and all upper-case letters are $n \times n$ matrices. $H$ and $K$ are diagonal, i.e. we have $r_{x_i} = h_i(r_{x_i} - x_i)$ and $y_i = k_i(y_i - \dot{y}_i)$. Furthermore, we shall take all the $h_i$ to be equal, and all the $k_i$; thus $H = hI$ and $K = kI$. $L$ is the linkage matrix relating the demanded value of $\dot{x}$ to that of $\dot{y}$. We take the environment relating $y$ to $x$ to be $x = Gy + A$ for some matrix $G$ and vector $A$, where either $G$ and $A$ are constant, or they are the linearisation of the system about its current state. The resulting system is described by Equation (9).

$$
\begin{align*}
\dot{r}_x &= h(r_x - x) \\
\dot{y} &= k(y - \dot{y})
\end{align*}
$$

By rescaling time we can take $h = 1$. If $L$ is chosen to be $G^{-1}$, we obtain Equation (10) for $x$.

$$
\dot{x} + k\dot{x} + kx = k r_x
$$

Figure 12: Multivariable control hierarchy
Each variable independently tracks its own reference. When $LG$ or $GL$ are not diagonal, there is some degree of coupling among the variables. A similar analysis can be given for a hierarchy with three or more levels.

In the backhoe example, $G$ is the Jacobian of the forward kinematics. To minimise computation, we chose $L$ to be the transpose of $G$ rather than its inverse. In practice this gives a sufficiently low degree of coupling between the degrees of freedom of the bucket. Figure 13 plots the bucket coordinates while the reference reach is increased, traversing between the two configurations of Figure 8. The right-hand graph shows, on an increased scale, the errors in the three variables during the movement.

Figure 13: Coupling between degrees of freedom. Left: values; right: errors. Solid line: reference reach; long dashes: reach; short dashes: height; dots: slope.

7 Related approaches

Hierarchical control schemes are ubiquitous in robotics. One such approach, or range of approaches, bearing some relation to the present design, is called subsumption. This is an architecture originally devised by Brooks [1], in which the control problem is, as for HPCT, broken down into a hierarchical arrangement of simpler agents. There are two fundamental differences with HPCT. Firstly, in a subsumption architecture, the agents are not necessarily conceived of as controllers, that is, agents which attempt to produce a certain input by means of their outputs. Secondly, the main principle of the subsumption architecture, for which it is named, is that all of the controllers at all levels act directly on the actuators, controllers at higher levels suspending the actions of controllers at lower levels as necessary, the lower level resuming its operation when the higher level has completed its task. Thus some of the actions taken to balance a legged robot in a standing posture are suspended when a higher-level agent for walking needs to lift some legs off the ground; an agent for walking in a straight line may be suspended by an agent for collision avoidance, and so on. In the HPCT architecture, only the bottom-level controllers send signals to the actuators. Higher level controllers send their outputs only to the reference inputs of controllers at the next level down. In subsumption, higher-level agents operate in instead of lower level agents; in HPCT, higher-level controllers operate by means of lower level controllers. In principle, an HPCT controller could act not only by altering the references of lower level controllers, which is the only mechanism studied here, but also, for example, by altering parameters of their output functions, or the linkage matrix connecting them to their descendants. However, there is never any skipping of levels.

8 Conclusion

We have illustrated a hierarchical scheme for multivariable control based on simple proportional controllers, and demonstrated examples of its functioning and performance. Further mathematical analysis, development of the robot simulation, and application to other complex tasks are the subject of ongoing research.

References


