gets smaller fast enough that an infinite number of bounces occur in a finite amount of time. A system with an infinite number of discrete events in a finite amount of time is called a **Zeno** system, after Zeno of Elea, a pre-Socratic Greek philosopher famous for his paradoxes. In the physical world, of course, the ball will eventually stop bouncing; the Zeno behavior is an artifact of the model. Another example of a Zeno hybrid system is considered in Exercise 13.

4.2.3 Supervisory Control

A control system involves four components: a system called the **plant**, the physical process that is to be controlled; the environment in which the plant operates; the sensors that measure some variables of the plant and the environment; and the controller that determines the mode transition structure and selects the time-based inputs to the plant. The controller has two levels: the **supervisory control** that determines the mode transition structure, and the **low-level control** that determines the time-based inputs to the plant. Intuitively, the supervisory controller determines which of several strategies should be followed, and the low-level controller implements the selected strategy. Hybrid systems are ideal for modeling such two-level controllers. We show how through a detailed example.

Example 4.8: An automotive **cruise control** system has a supervisory controller (typically a human) that switches the system between several modes of operation. A simplified model of such a system is shown in Figure 4.12. This model has pure inputs, which come from a human driver (or possibly an automaton, in the case of a self-driving car). These inputs are *set*, which turns on the cruise control and sets the speed that the cruise control should match; *cancel*, which reverts to manual control; *coast*, which temporarily disables the cruise control and allows the car to slow down; and *resume*, which resumes with the cruise control matching the last speed set. The model also involves several real-valued variables, which in

summary are:

$$
\begin{aligned}
s(t) &= \text{the speed at time } t \\
p(t) &= \text{the accelerator pedal command from the driver at time } t \\
r(t) &= \text{the resistance due to drag, friction, and slope at time } t \\
d &= \text{the desired speed} \\
G &= \text{the gain for proportional controller}
\end{aligned}
$$

The model shows three modes of operation, off, where the driver is in charge, engaged, where the cruise control is attempting to match the set speed, and coasting, where the car is coasting, slowing down due to drag and friction. In each of these modes, the rate of change of speed $\dot{s}(t)$ is a function of some subset of three values.

The input $p(t)$ represents the driver's command (given via the accelerator pedal). This input is ignored in the engaged and coasting modes, although in a less oversimplified model, instead of being ignored, it should be used to possibly cancel the cruise control and switch to the off mode. A typical automotive cruise control will disengage the cruise control if the driver pushes on the accelerator pedal.

The input $r(t)$ represents the resistance to forward motion that the car encounters due to aerodynamic drag, friction, and the slope of the road. This input tends to slow down the car in all three modes of operation. Typically, the aerodynamic drag will be proportional to speed at low velocities and proportional to the square of the speed at higher velocities.

In the off mode, the rate of change of speed $\dot{s}(t)$ is given by the difference between the driver command $p(t)$ and the resistance $r(t)$. In the coasting mode, it is given by the negative of the resistance, equivalent to the driver not applying any acceleration.

The engaged mode is the most interesting one. This simplified model realizes a proportional controller, which simply multiplies the error by a constant gain G and uses the result as an acceleration command. The error signal is the difference between the desired speed d, which is set upon entering the engaged mode, and the current speed $s(t)$. This command is modified by subtracting the resistance $r(t)$ to determine the rate of change of speed $\dot{s}(t)$. A more sophisticated controller would use a type of controller called a **PID controller** (see Lee and Varaiya (2011)), but for our purposes here, a simple **P controller** is sufficient.

The plant, in this example, is the car, which responds to an acceleration input in a very simple way, by setting the rate of change of speed. The environment in this case is the air around the car and roadway, which together determine the resistance $r(t)$. The sensors provide the measurement of the current speed $s(t)$ of the car, which is an input to the low-level controller in the engaged mode. The controller has two levels. The low-level control is either a human (in the off mode), or the cruise control system (in the engaged and coasting modes). The supervisory control system is modeled by the switching between these modes, whereas the low-level controller is modeled by the differential equations that determine how the speed of the car changes in each of the three modes.

4.3 Summary

Hybrid systems provide a bridge between time-based models and state-machine models. The combination of the two families of models provides a rich framework for describing real-world systems. There are two key ideas. First, discrete events (state changes in a state machine) get embedded in a time base. Second, a hierarchical description is particularly useful, where the system undergoes discrete transitions between different modes of operation. Associated with each mode of operation is a time-based system called the refinement of the mode. Mode transitions are taken when guards that specify the combination of inputs and continuous states are satisfied. The action associated with a transition, in turn, sets the continuous state in the destination mode.

The behavior of a hybrid system is understood using the tools of state machine analysis for mode transitions and the tools of time-based analysis for the refinement systems. The design of hybrid systems similarly proceeds on two levels: state machines are designed to achieve the appropriate logic of mode transitions, and continuous refinement systems are designed to secure the desired time-based behavior in each mode.

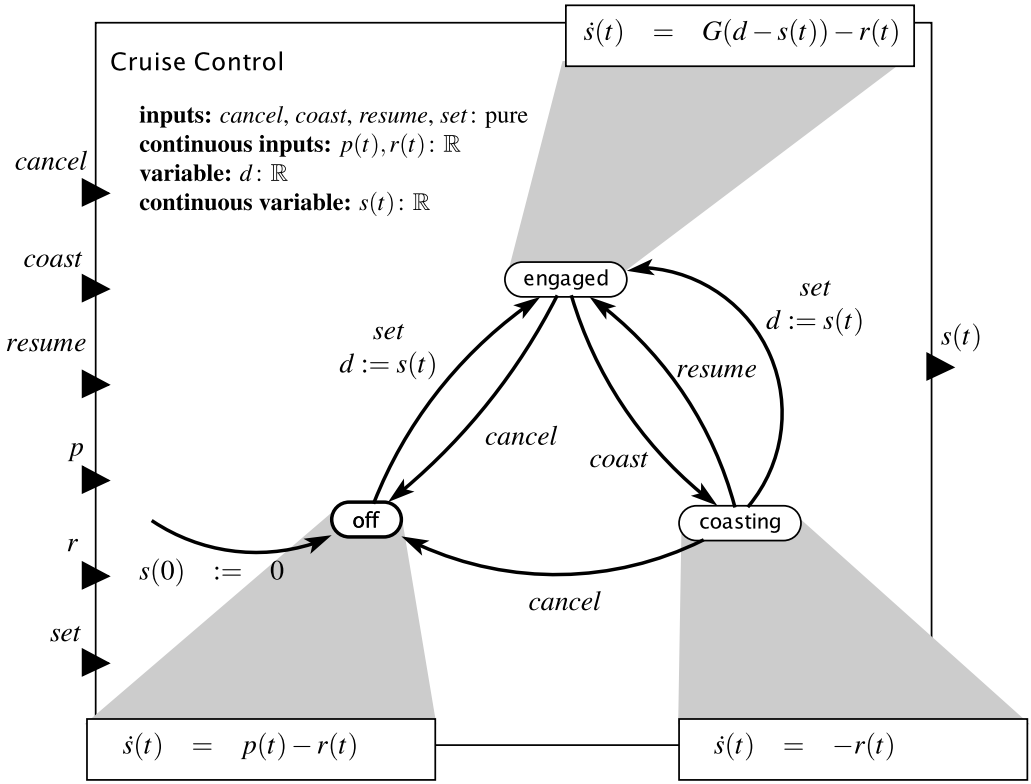

Figure 4.12: Illustration of a supervisory controller for a cruise control system.