Computer Science 773

Robotics and Real-time Control

NEURAL NETWORKS

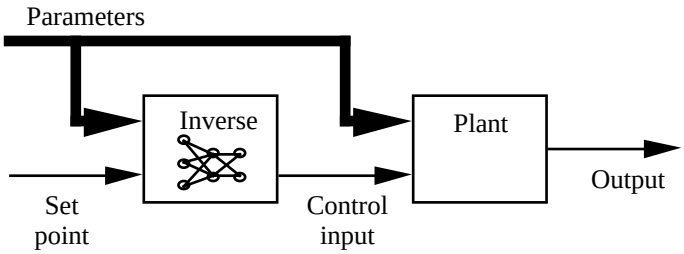
Within the last ten years or so, neural networks have been introduced into control systems in a variety of ways. This sheet describes a few simple examples, but many others have been shown to be useful in different circumstances.

The general principle is to use the network to learn some property of the controlled system which it is difficult to incorporate into the controller in any other way. This is often some sort of non-linearity, and the network is then used to compensate for the intractable plant behaviour so that it can be controlled more easily.

A general reference for the examples discussed here is : D. Sbarbaro-Hofer, D. Neumerkel, K. Hunt : "Neural control of a steel rolling mill", *IEEE Control Systems* **13#3**, 69-75 (June, 1993).

OPEN-LOOP CONTROL.

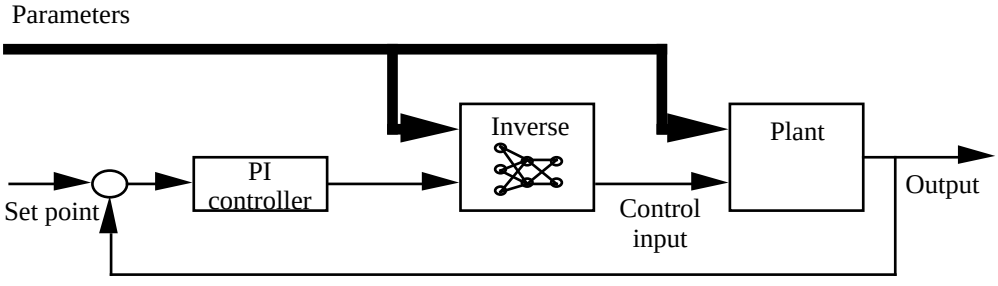
This is perhaps the simplest form of control. A neural network is trained to convert the desired set point into the correct plant control input to produce the required value. As the control input alone is unlikely to determine all the behaviour of the plant, the network must be trained for a range of values of the other plant parameters so that it will work correctly under all circumstances.



This network is regarded as the inverse of the plant, because, while the plant converts the control input into the output, the network converts the desired output into the control input. In practice, the performance is likely to be quite good, but to fall well short of perfect; it is practically impossible to take into account all the factors which determine the plant output, so the inverse can never be exact for all conditions.

CLOSED-LOOP CONTROL.

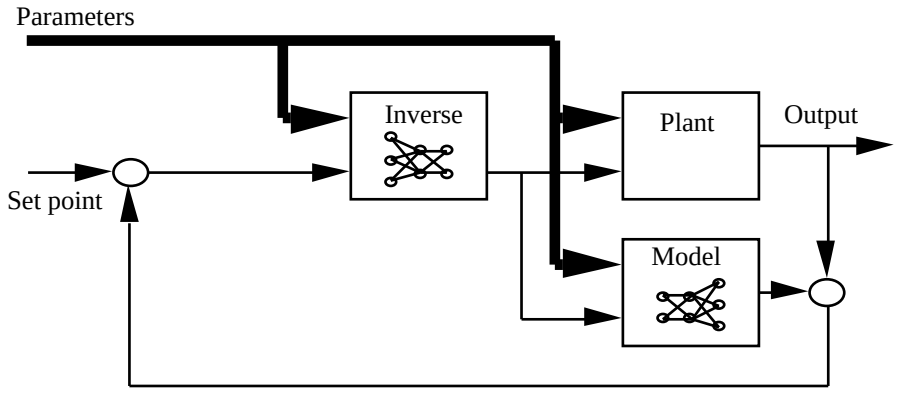
What the open-loop system almost certainly will achieve, though, is to get rather close to the desired value under most circumstances. By fitting a feedback controller round the system, the input to the inverse network can be moved slightly to compensate for the departure from perfection.



Even though the function computed by the network is not an exact inverse, the composite behaviour of the inverse network and the plant, which is what the additional controller sees, is likely to be something reasonably close to an identity function, and any conventional controller will suffice for the control required. I've drawn a PI controller (that's what they put in the original paper), because in practice that's the standard simple controller.

INTERNAL MODEL CONTROL.

Instead of using a controller to correct the smallish errors resulting from the inaccurate inverse, it is possible to try to compensate for them. This is possible by using a second network to model the plant.



First train the model network to reproduce the plant's behaviour as closely as possible; then train the inverse network to be the inverse of the model, which is likely to be easier than to get the inverse of the plant. Now, as the combination of inverse and plant is likely to be quite close to a unit operator, the difference between the outputs of model and plant should be just about the correction to the set point needed to cause the inverse to produce the required control input.

Work it out.

Alan Creak,
March, 1997.