Robotic systems are capable of complex behavior by sequencing simpler skills called primitives (Voyles, Morrow, & Khosla 1997). A primitive is a sensor/actuator mapping robust enough to perform appropriately in various situations. Programming one primitive can be tedious and requires an accurate translation of human knowledge to machine code. Once a sufficient set of primitives is coded, the user must write code to sequence the primitives — also tedious and difficult. Programming by human demonstration addresses these problems of acquiring and combining primitives.

To create primitives, programming by demonstration can be implemented with a supervised learning technique such as artificial neural networks (ANN) to learn a sensor/actuator mapping. Problems exist with such techniques, however, including creating a training set which is comprehensive (for robustness) and concise (for efficient training). Here, we present a method for nonexpert users to collect “good” training data from an intuitive understanding of task behavior, not from knowledge of the underlying learning mechanism.

Good training data includes anomalous situations and corrective behavior. For example, when road-following, data should include examples of how to return to the road if the robot inadvertently strays from the lane. However, if the demonstrator veers off the road to show the robot how to correct itself, the system also learns to veer off the road. Pomerleau’s solution (1992) is to simulate corrective behavior, but this requires task domain knowledge. Our solution, applied to wall-following for indoor mobile robots, is to filter real data, automatically separating good data from bad.

Data from a demonstration consists of sensor and actuator vectors. A sensor vector contains all sensor readings and an actuator vector contains all actuator values at a given timestep. Together, these vectors comprise training data from which the learning method extracts the inherent sensor/actuator mapping. Our filtering process determines which of these vector pairs qualify as good.

We first calculate standard deviation of each sensor across time, providing a measure of consistency. Each sensor whose standard deviation falls below a threshold is labelled a key sensor. For each, the most frequent reading is determined and used as its characteristic reading. The result is the characteristic vector, depicting the desired behavior.

We filter data by taking the vector difference of the characteristic vector and the key sensor readings at each timestep, then we analyze the slope of the smoothed differences across time. A positive slope at a data point indicates the robot is moving away from the desired behavior. This data point is assumed bad and is removed.

Other possible uses for a characteristic vector include: a guide for selecting a subset of sensors for more efficient ANN training; a guide to include or exclude additional data keeping the training set from becoming prohibitively large for on-line learning (similar to that proposed in Pomerleau but without task domain knowledge); and most importantly, as behavior models for Hidden Markov Models (HMMs). The ultimate goal of this work is to create a robotic system capable of learning sequential tasks from human demonstration. HMMs have been used successfully for this purpose in robotics (Pook & Ballard 1993) and are good candidates for success here.

Preliminary experiments on RWI’s ATRV Jr. and Nomadic’s SuperScout resulted in a comprehensive training set with a single, continuous demonstration. Note this method relies on the assumption that key sensors are those with relatively constant readings. This holds for many tasks; nonetheless, we may be able to relax it by using correlation coefficients of sensors and actuators.

Acknowledgements
This work sponsored by Air Force Research Lab under contract F30602-96-2-0240.

References