A Neuro-Fuzzy Solution for Fine-Motion Control Based on Vision and Force Sensors

Yorck von Collani, Jianwei Zhang, Alois Knoll
Technical Computer Science, Faculty of Technology,
University of Bielefeld, 33501 Bielefeld, Germany

Abstract

In this paper the use of a B-spline neuro-fuzzy model for different tasks such as vision-based fine-positioning and force control is presented. It is shown that neuro-fuzzy controllers can be used not only for low-dimensional problems like force control but also for high-dimensional problems like vision-based sensorimotor control. Controllers of this type can be modularly combined to solve a given assembly problem.

1 Multivariate Problems in Modelling and Control

It is well-known that general fuzzy rule descriptions of systems with a large number of input variables suffer from the problem of the "curse of dimensionality." In many real-world applications it is difficult to identify the decisive input parameters and thus to reduce the number of input variables to the minimum. A general solution to building fuzzy models is not only interesting from a theoretical point, it may also extend the range of applications of fuzzy control to more complex intelligent control problems.

In our research work on sensor-based robot control [4], we are faced with many high-dimensional problems concerning a large number of input variables whose importance and inter-dependence are not clearly known.

1.1 Vision-Guided Robot Motion

The classical approaches to robot vision frequently fail in industrial environments for the following reasons:

- The image processing procedures, such as segmentation, feature extraction and classification are not robust enough.
- The algorithms are computationally expensive and hence too slow to meet the usual real-time requirements.

It is one of our long-term research goals to find a general model which transforms raw image data directly into "action values." Our grey-scale images have 111×103 pixels. If no image processing is performed then a control system with about $12,000$ input variables (i.e. one for each pixel) needs to be modelled; the system output would be the motion values for the robot(s).

A special case of vision-action transformation is camera-supported fine-motion control. *Affine Visual Servoing* [9] may be applied to such tasks. The changes in the shape of image contours are the input of a feed-forward controller. Another interesting approach was proposed in [12]: visual servoing for positioning with an uncalibrated camera system. It requires, however, special test motions for performing well.

CMAC neural networks may tackle the problem of dimensionality; in [10] 12 inputs represent four joint positions of the robot, four image parameters and their desired changes. The outputs are the control signals for the four robot joints.

In [11] learning of vision-based positioning based on visual appearance information was introduced. The image data set is compressed using *principal component analysis* to obtain a low-dimensional input space. A parametric eigenspace representation is used for describing the different objects as well as object locations. The positioning problem is thus transformed into finding the minimum distance between a point and a manifold in the eigenspace.

1.2 Existing Solutions to Multivariate Problems

The two main methods to solve the problem of input dimension reduction are *input selection* and *hier-*

archy. Input selection [5, 6] is an experimental method to find the most important input variable in a large set of input variables. With this approach, all the combinatorial possibilities of the low-dimensional fuzzy model are considered and approximately tested. The inputs which result in the best outputs are viewed as the most important ones to build an exact neuro-fuzzy model. The problems with this method are the loss of information and the number of combinations that must be tested.

Hierarchical structuring assumes that the input information can be classified into groups, see [7] for an example. There is, however, no general approach to realise such a grouping.

2 Problem Description

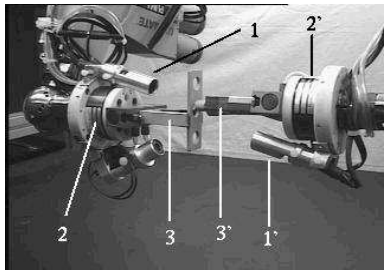
2.1 Experiment Setup

In the problem scenario (see Fig. 1), two cooperating robots are to insert a screw (5) into a wooden ledge (4). The manipulators are installed overhead and can grasp the required assembly components from the assembly table. Each robot is equipped with a force sensor (2,2') on which a pneumatic parallel-jaw gripper (3,3') is mounted. A small camera (1,1'), which observes the scene, is mounted over the gripper at an angle of approximately 30°. The manipulators are two Puma 260. The host computer is a Sun SPARC 5.

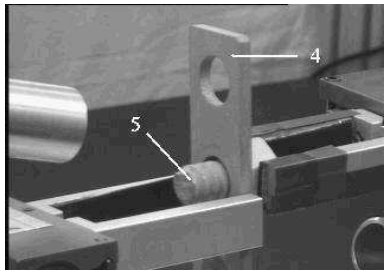
2.2 Problem of Finding a Hole Without Vision

Detecting the contact between screw and ledge, e.g. only through force measurements, is not sufficient to ensure that the screw is in the hole. If there is no support from a camera, the hole has to be searched for by making test movements. In our case, a spiral movement within the vicinity of the hole in the ledge is made. As the ledge presses onto the screw, the screw slides in by traversing the hole.

To make this procedure work, it must be guaranteed that the contact between the ledge and the screw not disappear; especially, when the screw is over the hole. Also the pressure between the ledge and the screw must not become too high because frictional forces that result from search movements might signal the finding of the hole. Force control in the approach direction is therefore inevitable. Otherwise the screw would move over the hole or the force in N- and O-direction may grow too large without the screw being in the hole. In [3] we gave an unsupervised learning approach to building a force controller for this purpose.



(a)



(b)

Figure 1: The experimental setup for assembly. 1,1': hand-camera; 2,2': force/torque sensor; 3,3': parallel jaw-gripper; 4: ledge; 5: screw-head.

2.3 Vision-Based Search

The above method is slow and can fail. A supplementary approach consists in monitoring the scene through the hole with a camera. The screw (or the ledge) may thus be guided to the correct position.

Fig. 2 shows a sequence of typical views of the scene. It is obviously difficult to recognise unambiguous features in the images for the following reasons:

- The background changes permanently. Depending on the orientation of the opposite robot's gripper, different parts of the gripper may be seen apart from the screw.
- Variable illumination. In our setup only daylight and no artifical lighting is used. Especially the lower part of the hole can often hardly be seen. A rotation of the gripper holding the ledge also changes the illumination.
- Different forms of the screw. We use screws with round heads and screws with hexagonal heads.

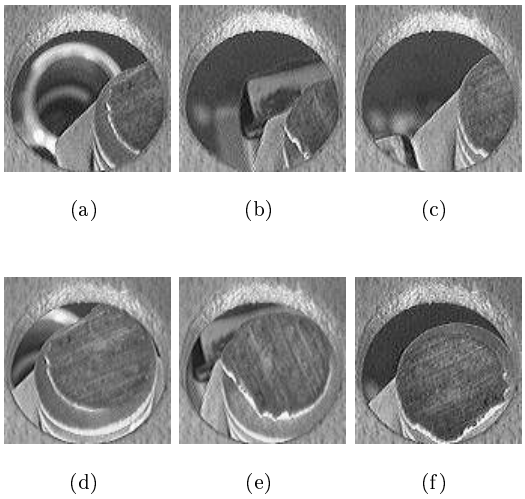


Figure 2: Typical images taken by the hand-camera (Image size: 111 × 103 pixels).

The geometry of the head therefore cannot be used to detect the position of the screw.

We employed the method that extracts features from the whole grey-scale image. It solves this partial problem of vision-guided motion but is in no way limited to this narrow application.

3 B-Spline Neuro-Fuzzy Model

3.1 Basic Principle

The controller for force control can be efficiently realised using the B-spline fuzzy controllers proposed in our earlier work [2, 3]. This type of controller may be characterised by the following features distinguishing it from standard fuzzy controllers:

- B-spline basis functions are employed for specifying the linguistic terms (labels) of the input variables. By choosing the order n of the basis functions, the output is C^{n-2} continuous.
- Each controller output is defined by a set of fuzzy singletons (control vertices). The number of control vertices is equal to the number of the rules and their optimal values can be iteratively found through learning. This adaptation procedure is equivalent to weight adjustment in an Associated Memory Neural Network.
- One problem with learning in conventional fuzzy controllers is that too many parameters must be adjusted. With B-spline fuzzy controllers, a simple modification of control vertices causes the change on the control surface. For supervised learning, if the square error is selected as the quality measure, the partial differential with respect to each control vertex is a convex function. For unsupervised learning, if the error of the cost function is approximately piecewise proportional to the error of the control values, the learning-process descent will also show stable asymptotic behaviour [1].

In [3] we showed the advantages of this learning controller approach. Here, the robot controller learns actively and on-line to control the compensation motion according to measured forces and is quite simple to design. The learning process converges rapidly, the output is smooth if B-spline functions of order 3 or higher are used and the controller performs well.

3.2 Dimension Reduction

If the dimension of the input space is small enough, the input variables can be directly covered by fuzzy sets. Each item of the rule is human readable and may be interpreted as describing a special instance of a general situation. If, however, the image of a camera is regarded as a vector, then this high-dimensional sensor image is too large to build a corresponding rule base. Fortunately, sensor images are often observed in a local context: the complete situation is not of particular interest and a subspace can be found that contains all necessary information for determining the action values.

3.3 Projection into Eigenspace

A well-known technique for dealing with multivariate problems in statistics is the *principal component analysis* (PCA). As shown in [11], this technique is also suitable for reducing the dimension of the input space of a general control problem.

Our approach is to project the grey-scale image into the eigenspace. An eigenvector, denoted as EV_i, is computed as $[a_{1,i}, a_{2,i}, \ldots, a_{m,i}]^T$. The eigenvectors form an orthogonal basis for representing the original individual sensor patterns. Assume that the eigenvectors $EV_1, EV_2, \ldots$ are sorted according to their eigenvalues in a descending order. An eigenspace with a reduced dimension n can be formed with the first n eigenvectors. EV_i defines the ith dimension

in the eigenspace. The projection of an input vector $X = [x_1, x_2, \cdots, x_m]^T$ onto eigenvector EV_i, called the ith principal component, $p_i = a_{1,i}x_1 + a_{2,i}x_2 + \cdots + a_{m,i}x_m$. The complete projection can be represented as:

$$[EV_1, \ldots, EV_n]^T \cdot X = [p_1, \ldots, p_n]^T$$

All projections of the sample data sequence form a manifold in the eigenspace. Such a projection can be viewed as a layer of neural network, see the connection layer of the two left parts of Fig. 3.

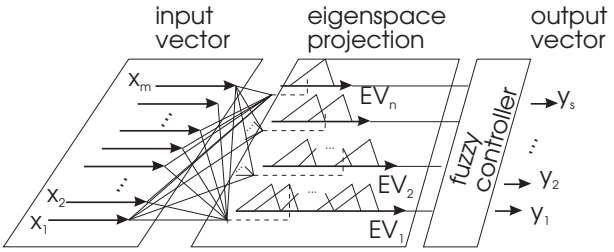


Figure 3: The structure of a fuzzy controller based on eigenspace projection.

We partition the eigenvector with the largest eigenvalue with the finest resolution because the projection on this eigenvector contains the largest variance. If the eigenvectors $\mathrm{EV}_1, \ldots, \mathrm{EV}_n$ are sorted in a descending order according to their eigenvalues, then EV_1 is the first eigenvector. The second eigenvector EV_2 is covered with fewer linguistic terms, and so on.

4 Implementation

4.1 Sampling Training Data

For training, the input data and desired output values have to be recorded. It is desirable that all typical input data be generated.

As outlined above there are the different positions of the screw around the hole (see Fig. 2). For recording, the robot moves to the ideal position for screw insertion. Subsequently, it moves to several other positions. For each of the latter the deviations from the ideal positions are recorded.

4.2 Calculating Eigenvalues and Eigenvectors

After the input data are sampled, the following steps are necessary:

1. The input data are normalised so that the energy of each image becomes 1. As an option the average image can be subtracted.
2. The input variables are stacked into vectors.
3. The covariance matrix of the input vectors is calculated.
4. The eigenvectors and eigenvalues are calculated.
5. Each image is projected into the eigenspace.

4.3 Training the Fuzzy Controller

For the B-spline controller the training procedure is as follows:

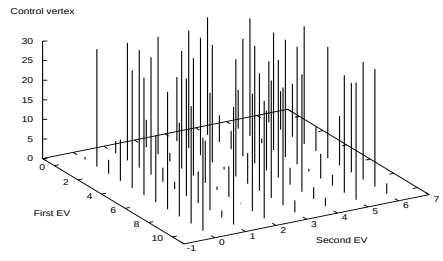
1. Select the n eigenvectors with the largest n eigenvalues denoted as $\mathrm{EV}_1, \ldots, \mathrm{EV}_n$.
2. Select the order of the B-spline basis function for each eigenvector.
3. Determine the knots of the B-spline basis functions for partitioning each eigenvector.
4. Initialise the control vertices for the output.
5. Learn the control vertices with the selected eigenvalues from the images using the gradient descent method.
6. If the result are satisfied, terminate.
7. Modify the knots for eigenvectors, go to 4.

For training the fuzzy controller, the eigenvalues of each of the selected eigenvectors EV_j $j = 1, \ldots, n$ are covered by B-spline basis functions denoted as $X^j_{i_j,k_j}$. The rule for determining the relative location of the screw to the hole can be written as:

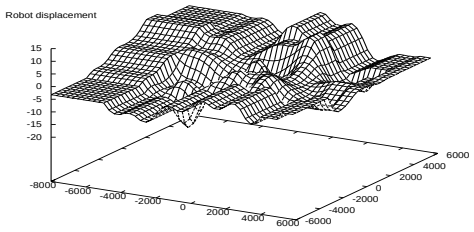
IF $(\mathrm{EV}_1$ IS $X^1_{i_1,k_1})$ and ... and $(\mathrm{EV}_n$ IS $X^1_{i_n,k_n})$
THEN $(x$ IS $X_{i_1,i_2,\ldots,i_n})$

Each rule corresponds to a supporting point for the interpolation in the reduced eigenspace.

It is important to determine the right parameters for the fuzzy set. If too few eigenvectors are used, then the fuzzy controller cannot differentiate all situations. If too many eigenvectors are used, then the memory usage of the fuzzy set is not manageable. Similarly important is the right partition. If the partition is too fine, the fuzzy controller generalises insufficiently.



(a) Control vertices on the plane of the first two eigenvalues.



(b) Control surface with the first two eigenvalues as input and the robot displacement as output.

Figure 4: Control vertices and surface.

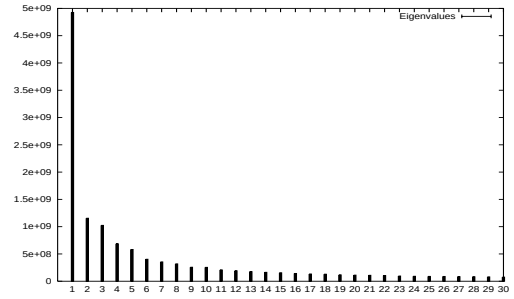


Figure 5: Sorted eigenvalues of covariance-matrix.

Number of used EV's	3	5	7
Number of used B-Splines	$11 \times 7 \times 7$	$11 \times 7 \times 7 \times 5 \times 5$	$11 \times 5 \times 7 \times 7 \times 5 \times 5 \times 3 \times 3$
Robot displacement			
above	5.0	1.4	1.09
above right	6.86	2.29	1.18
below	–	0.2	0
to the left	–	–	3.64
above left	4.86	2.2	2.7

Table 1: Required correction steps in O-direction for two different controllers.

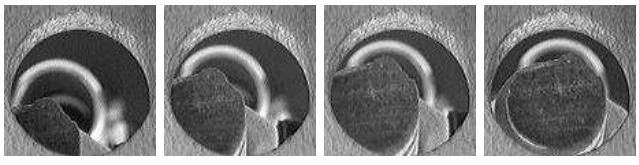
5 Numerical Results

Fig. 4(a) shows the learned control vertices for the first two eigenvectors and 4(b) the resulting control surface. Fig. 5 shows the sorted eigenvalues of the covariance-matrix.

Table 1 shows the average number of the required correction steps in O-direction depending on the controller parameters for some typical displacements. An empty field indicates that the controller cannot manage this situation. A zero means that no correction is made, so that the result for "below" is correct. In this case only a correction in N-direction is necessary.

Our experiment shows that with an increasing number of the eigenvectors fewer steps for correcting the position of the ledge are required. With three and five eigenvectors not all situations can be separated. If, for example, the screw is to the left of the hole, the controller cannot correct the displacement. In this case the motion was made into the wrong direction. Fig. 6 shows a sequence of movements generated by the fuzzy controller.

Combining this approach with the first procedure described in [3] results in a very robust and rapid technique to find the hole and to insert the screw. During the approach phase contact is made between the ledge and the screw, similar to the procedure used for spiral search.



(a) (b) (c) (d)

Figure 6: Correction sequence by using the trained B-spline neuro-fuzzy model.

6 Conclusions

We have shown that high-dimensional problems such as visually guided fine-motion can be solved with neuro-fuzzy controllers. We have also shown that the B-spline model may be utilised for different problems, not only for force control. We have implemented the approach with a two-arm robot system and both kinds of training are used to build the controllers: unsupervised on-line learning for the force controller and supervised off-line for the vision system.

The advantages of our approach are:

- By projecting the high-dimensional input space into a reduced eigenspace, the most significant information for control is maintained. A limited number of transformed inputs can be partitioned with the B-spline model and a sufficient precision can be obtained for determining the robot position correction.
- To solve this special but typical problem the statistical indices provide a suitable solution to describe the information in images with a lot of uncertainties.
- A vector in the eigenspace is directly mapped onto the controller output based on the B-spline model. This makes real-time computation possible.
- Designing the controllers is simple and identical for both low and high dimensional controllers. Both force and vision controllers are of the same type. The B-spline fuzzy controller can be trained in a straightforward manner because modification of control vertices only results in local change of the control surface.

In this approach no complex programming and knowledge about vision are needed. We have shown that this approach is very promising for realising efficient robot assembly skills based on sensorimotor coordinations.

Acknowledgement

Ralf Schmidt's implementation work with PCA is gratefully acknowledged.

References

[1] J. Zhang and A. Knoll. Constructing fuzzy controllers with B-spline models – principles and applications. *International Journal of Intelligent Systems* , 13(2/3):257–286, February/March 1998.

[2] J. Zhang and A. Knoll. Constructing fuzzy controllers with B-spline models. In *IEEE International Conference on Fuzzy Systems*, 1996.

[3] J. Zhang, Y. v. Collani and A. Knoll. On-line Learning of B-Spline Fuzzy Controller To Acquire Sensor-Based Assembly Skills. In Proceedings *IEEE International Conference on Robotics and Automation*, 1997.

[4] A. Knoll, B. Hildebrandt, J. Zhang. Instructing Cooperating Assembly Robots through Situated Dialogues in Natural Language. In Proceedings *IEEE International Conference on Robotics and Automation* 1997.

[5] J. S. R. Jang, C. T. Sun, and E. Mizutani. *Neuro-Fuzzy and Soft Computing*. Prentice Hall, 1997.

[6] S. L. Chiu. Selecting input variables for fuzzy models. *Journal of Intelligent and Fuzzy Systems*, 4:243-256, 1996.

[7] V. Lacrose and A. Tilti. Fusion and hierarchy can help fuzzy logic controller designers. In *IEEE International Conference on Fuzzy Systems*, 1997.

[8] J. Zhang and A. Knoll. Constructing Fuzzy Controllers for Multivariate Problems by Using Statistical Indices. Submitted to *IEEE International Conference on Fuzzy Systems*, 1998.

[9] C. Colombo, B. Allotta, and P. Dario. Affine visual servoing: A framework for relative positioning with a robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 464-471, 1995.

[10] W. T. Miller. Real-time application of neural networks for sensor-based control of robots with vision. *IEEE Transactions on System, Man and Cybernetics*, 19:825–831, 1989.

[11] S. K. Nayar, H. Murase, and S. A. Nene. Learning, positioning, and tracking visual appearance. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 3237–3244, 1994.

[12] B. H. Yoshimi and P. K. Allen. Active, uncalibrated visual servoing. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pp 156–161, 1994.