Vision-based Manipulator Navigation using Mixtures of RBF Neural Networks

Wolfram Blase, Josef Pauli, and Jörg Bruske

Christian–Albrechts–Universität zu Kiel, Institut für Informatik
Preusserstrasse 1-9, D-24105 Kiel, Germany
Fax: (++49)431560481, Email: jpa@informatik.uni-kiel.de

Paper accepted at
International Conference on Neural networks and Brain
October 27-30, 1998, Beijing, China

Vision-based Manipulator Navigation using Mixtures of RBF Neural Networks

Wolfram Blase, Josef Pauli, and Jörg Bruske

Christian–Albrechts–Universität zu Kiel, Institut für Informatik
Preusserstrasse 1-9, D-24105 Kiel, Germany
Fax: (++49)431560481, Email: jpa@informatik.uni-kiel.de

1 Subject of the work

In our behaviour-based robot system a manipulator has been equipped with a monochromatic video camera, fastened onto the robot hand. Based on image processing and neural network learning the system executes goal directed perception-action cycles and thus attains versatile skills. Especially this work reports on close range manipulator navigation, i.e. searching collision-free trajectories of the robot hand to approach and handle goal objects. Neural network learning with *radial basis functions (RBFs)* is involved twofold. First, a function is learned for reconstructing from the optical flow of detected obstacle points their three-dimensional positions. Second, a function of the inverse manipulator kinematics is learned which is used for describing the non-rigid space occupied by the agile manipulator. Furthermore, based on the goal position and the continually detected obstacle positions a vector field is created dynamically by using the gradient of RBFs as basis fields. The vector field simulates attracting and repelling forces for navigating the manipulator hand. To overcome the curse of dimensionality and reach acceptable efficiency in function learning we applied mixtures of RBF neural networks and strongly emphasized divide-and-conquer strategies. The parallel approaches for neural learning (and image processing) are implemented on a four-processor general purpose workstation.

2 Learning to reconstruct from optical flow

For the purpose of online detecting obstacles the camera must be moved through the working space. The SUSAN edge detector [Smith and Brady, 1997] is used for extracting greylevel corners probably arising from obstacles. Based on corresponding features between two successive images the obstacle position must be reconstructed into $3D$ space. The reconstruction function is learned offline using a *hierarchical mixture of expert (HME) networks* [Jordan and Jacobs, 1994] in which RBF networks are arranged in two layers. Figure 1 shows the application of such a mixture of networks for reconstruction from optical flow vectors. Each RBF network of the first layer is trained for a small image area and is used for reconstructing from the optical flow therein the depth coordinate Z. Each RBF network in the second layer is trained for a small range of depth, i.e. ranges of Z, and is used for computing the space coordinates X and Y. The merit of this architecture is twofold. First, the non-linearity of the RBFs takes care for the nonlinear type of reconstruction which is due to significant image distortions. These distortions are a consequence of using a wide-angle objective (lens with small focal length, e.g. $4.2mm$) needed in close range navigation for depicting a wide view volume. Second, the module architecture makes it possible to train each network efficiently by taking only a small subset of the whole training set into account. The output on each of the two layers is calculated by linear combining the respective outputs of a small set of relevant RBF networks. The combination factors are supplied by one gating network for each layer (not shown in the Figure), which are pre-specified in our application.

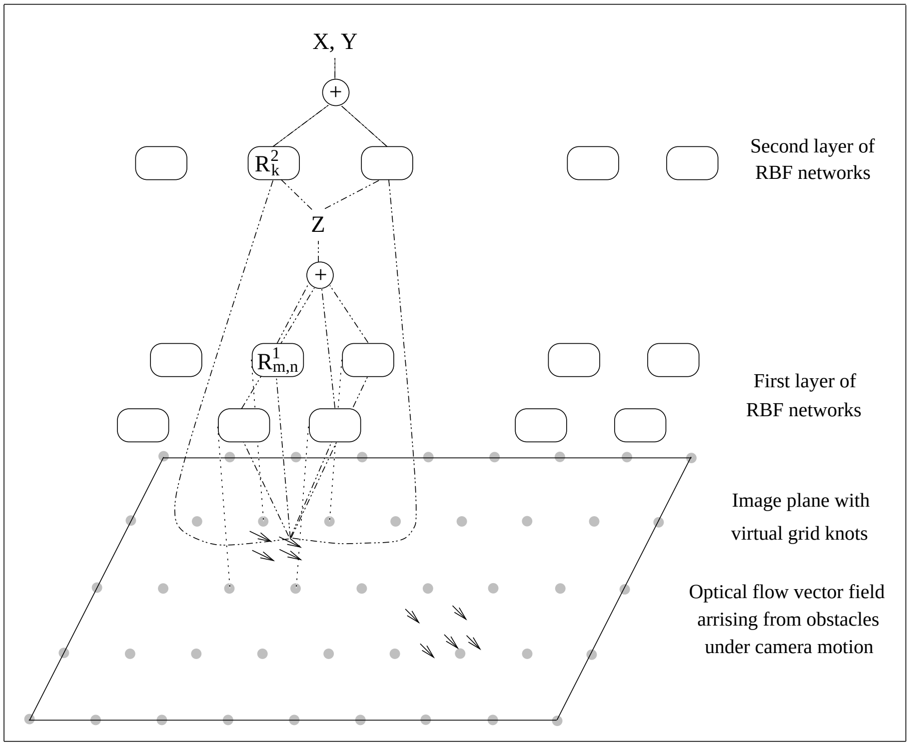

Figure 1: Hierarchical mixture of RBF networks, e.g. two layers.

The reconstruction function f_{rec} is learned as follows.

1. A sheet of paper depicting a regular distributed set of calibration dots is put at a fixed place of a ground plane. Beginning in a near position the camera is moving off the sheet in discrete steps with the optical axis approximately normal to it (e.g. 10 steps of $50mm$ each). At each step an image is taken and the calibration points are extracted. Furthermore these points are determined in the coordinate system of the robot hand which is translating step by step.

2. For every two successive camera positions the corresponding image positions of the calibration points are associated with the Z coordinates of their $3D$ positions (i.e. relative to the second of the two camera positions). A regular grid is defined for the image plane and one RBF network created for each grid knot respectively (first layer in the HME network). Each RBF network is trained efficiently by using a small set of calibration points (more conretely using the flow vectors) located in the neighborhood of the respective grid knot. The ISODATA clustering algorithm is used for defining the hidden nodes and a singular value decomposition (SVD) applied for determining the weights.

3. For each discrete camera position the image positions of the calibration points are associated with the (X, Y) coordinates of their $3D$ positions. One RBF network is defined with respect to each discrete camera position (second layer of the HME network). These RBF networks are trained (using ISODATA and SVD) by taking the respective associations into account.

Let (x_1, y_1) and (x_2, y_2) be corresponding positions of an obstacle in the image before and after camera motion. The second position is used to determine four neighboring grid knots $g_{m,n}$, $g_{m+1,n}$, $g_{m,n+1}$, $g_{m+1,n+1}$. The respective RBF networks $R^1_{m,n}$, $R^1_{m+1,n}$, $R^1_{m,n+1}$, $R^1_{m+1,n+1}$ of the first HME network layer is applied to the optical flow $(x_2 - x_1, y_2 - y_1)$. The linear combined output gives the depth coordinate Z. Just this coordinate is used for selecting two RBF networks R_k^2, R_{k+1}^2 from the second layer, which are most sensitive to Z. They are applied to (x_2, y_2) and the combined output gives the coordinates X and Y of the obstacle.

Figure 2 shows two drinks cans (left and middle) and a beer bottle (right) stored in a refrigerator. The manipulator has to approach the goal object (assuming the can in the middle) by bypassing the obstacle objects (left can and bottle). The SUSAN edge detector has extracted a set of interesting points (see black dots) arising from the imprints of the three objects. Figure 3 shows for these detected image points the $3D$ reconstruction using the mentioned HME network. The X and Z coordinates are shown for points on the goal object (G) and obstacle objects (H_1 and H_2).

Figure 2: Detected grey level corners at two obstacle objects (left and right) and one goal object (middle).

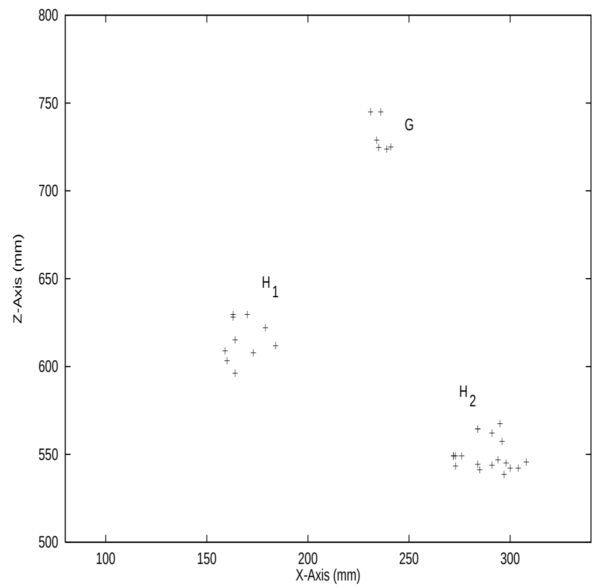

Figure 3: Reconstructed $3D$ coordinates X and Z of detected points from obstacles (H_1, H_2) and goal (G).

3 Learning the inverse manipulator kinematics

Suppose the manipulator must approach a goal position G, but in close neighborhood an obstacle H has been detected. Before approaching the goal it must be determined whether the arm segments will probably collide with H (see Figure 4). This is done by just simulating a movement to G and there describing the occupied space of the manipulator. Figure 5 shows the rotation angle ω_i and position p_i of the joints, and the length l_i and diameter d_i of the links. Assuming that l_i, d_i are known a priori and p_i are computable from l_i and ω_i we easily compute an approximation of the occupied space V_m of the manipulator. Finally it must be checked whether obstacle H is contained in the virtual manipulator space V_m, and if this is not the case, the manipulator actually can appraoch goal G.

The only problem is to solve the inverse manipulator kinematics [Craig, 1989], i.e. determine the mapping of the $3D$ goal position $G = (X, Y, Z)$ into the relevant vector of rotation angles $\Omega = (\omega_1, \cdots, \omega_n)$. We build one layer of RBF networks in which each network is responsible for a certain range of Z and in consequence of that the dimension of the input space is reduced into $2D$

by dropping the Z component. Each RBF network is trained with associations between vectors G and Ω, taking only vectors G with relevant Z values into account. The efficiency of training arises by taking for each RBF network only a subset of the whole training samples into account. In the application phase we determine for an input vector G those RBF networks whose responsible ranges contain the Z value (e.g. two or more networks), apply these networks to the (X, Y) tuple, linear combine the respective outputs, and this gives the vector of rotation angles Ω. According to our experiments the approximation errors in the rotation angles ω_i can be reduced to $1°$ degrees in the mean. Applying forward kinematics this results in a mean positioning error of about $3mm$ which is good enough for checking criteria of obstacle avoidance.

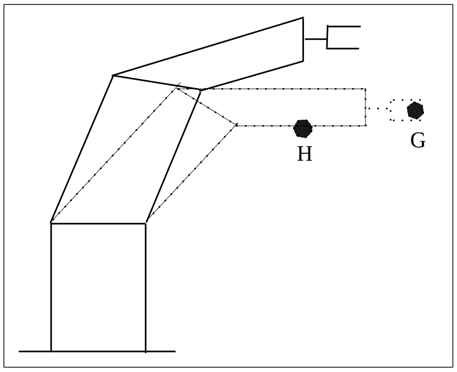

Figure 4: Manipulator and simulated obstacle collision.

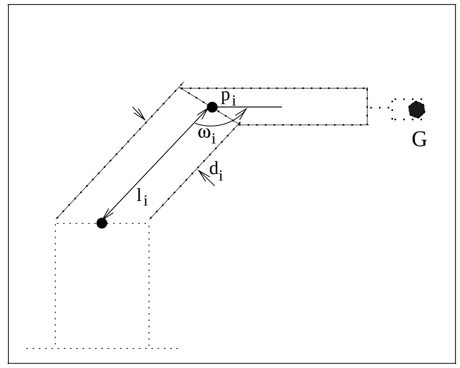

Figure 5: Characterizing the manipulator kinematics.

4 Dynamic construction of a force vector field

The manipulator must navigate towards a goal position while avoiding obstacles. This is done by dynamically constructing a vector field of simulated forces [Mussa-Ivaldi and Giszter, 1992]. The goal position is the center of an attractor field, i.e. in a working space all discretized points specify the origin of a vector which is directed towards the goal (Figure 6, left). As the manipulator begins to move from an arbitrary position it will be attracted from the forces in the goal. Whenever the vision system detects an obstacle a repellor field is created at that position (Figure 6, middle). The summation of attractor and repellor field results in appropriate forces, i.e. the manipulator will be pushed off and thus bypasses the obstacle for approaching the goal (Figure 6, right).

The attractor field is simply defined by vectors of unique length α.

$$AF_G(P) := \alpha \frac{(G - P)}{\| G - P \|} \tag{1}$$

The repellor field is defined by computing the gradient of a negated radial basis function centered at an obstacle position.

$$\Phi_\sigma(P, H) := -exp(-\frac{\| P - H \|^2}{\sigma^2}) \tag{2}$$

$$RF_H(P) := \frac{\partial \Phi_\sigma(P, H)}{\partial P} = 2(P - H)\Phi_\sigma(P, H) \tag{3}$$

The unknown σ value of the gaussian is computed by considering a desired minimal distance from the obstacles and taking the (small) inaccuracy of reconstruction into account.

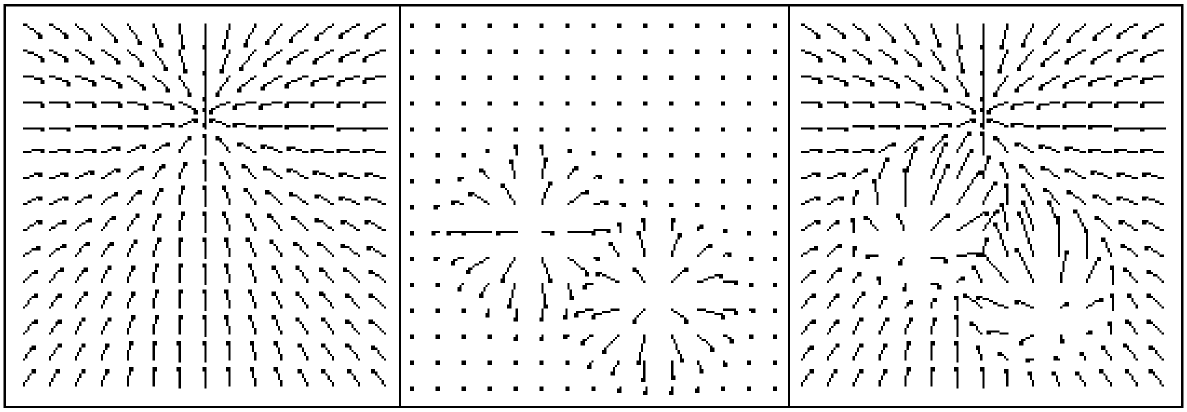

Figure 6: Attractor field for the goal object (left), two repellor fields for two obstacle objects (middle), summation of both fields (right).

In order to exploratory navigate towards the goal position the manipulator is moving step by step, and the vision component is detecting obstacles. In these cases repellor basis fields are constructed, and the force vector field is changed dynamically. During the process locally a set of null vectors can arise which is similar to a local minimum in a potential field. These places are simply treated as obstacles, i.e. putting repellor fields there in order to generate repellent forces. The emerging vector field implicit represents a trajectory towards the goal position. For globally exploring the scene the navigation is repeated for different starting positions and thus an overall force vector field is constructed which implicit represents trajectories towards the goal position starting arbitrarily.

5 Summary

Mixtures of RBF neural networks have been used for vision-based manipulator navigation. A two-layer mixture of RBF networks is appropriate for reconstructing 3D positions of obstacles especially for the case of significant image distortions which result from wide-angle objectives. A one-layer mixture of RBF networks is involved for efficiently solving the inverse manipulator kinematics, which is important for computing the occupied space of the manipulator. A force vector field is constructed dynamically by detecting obstacles and placing repellor fields, which are specified by the gradient of negated RBFs.

References

J. Craig. *Introduction to Robotics.* Addison-Wesley Publishing Company, Massachusetts, 1989.

M. Jordan and R. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

F. Mussa-Ivaldi and S. Giszter. Vector field approximation – A computational paradigm for motor control and learning. *Biological Cybernetics*, 67:491–500, 1992.

S. Smith and J. Brady. SUSAN: A new approach to low level image processing. *International Journal of Computer Vision*, 23:45–78, 1997.