Robotica (2004) volume 22, pp. 309–318.
DOI: 10.1017/S0263574704000025 Printed in the United Kingdom

A method to learn hand grasping posture from noisy sensing information

P. Gorce and N. Rezzoug

LESP EA 31 62 Université de Toulon et du Var, Avenue de l'université, BP 132, 83957 La Garde (France)
E-mail: gorce@univ-tln.fr

(Received in Final Form: October 10, 2003)

SUMMARY

In this paper, we propose a new method to learn a multi-fingered hand grasping posture with little knowledge about the task and few sensing capabilities. The developed model is composed of two stages. The first is dedicated to the finger inverse kinematics learning in order to provide the fingertip-desired position. This function is fulfilled by modular neural network architecture. Following the concept of reinforcement learning, a second neural model dealing with noisy sensing information is used to search the space of hand configuration. Simulation results show a good learning of grasping postures with five fingers and different noise levels.

KEYWORDS: Grasping; Neural networks; Reinforcement learning; Noise and uncertainty.

1. INTRODUCTION

For many years one of the main objectives of robotic research is to reproduce human dexterity and flexibility in an unknown environment. These skills are then transferred to the control of artificial devices of different type, starting from a simple jaw gripper to multi-fingered hands. In the large spectrum of problems related to the use of artificial hands, the definition of a grasping posture associated to an object has been the most challenging, since it implies the consideration of a large amount of constraints. The constraints are, indeed, related not only to the structure of the hand and of the object but also to the requirement of the task and to the state of the environment. Moreover, sensing capability is a crucial point in the characterization of the system state and any attempt to solve the grasp-planning problem must take into account the nature and the quality of sensory signals that may be of low precision and subject to noise.

Several attempts have been lead to classify human prehension in order to better understand the behavioral mechanisms and the different parameters involved in the definition of the hand grasping posture.[1–6] These classifications have been widely used to bring solutions in the robotics field with knowledge-based systems,[7,8] taking advantage of the observation of the human behavior.[9] Another approach, called analytic, consist in the optimization of predefined criteria characterizing grasp (form-force closure and stability).[10–14]

One of the main interests of the neural network approaches in grasp planning is to perform the learning of the underlying rules adopted during the grasp. Thus, Kurperstein and Rubinstein,[15] developed a neural network-based scheme called INFANT in order to perform grasping tasks with a 5 dofs. gripper. Uno *et al.*[16] proposed a two step neural-based computational scheme able to integrate an internal representation of the objects as well as the corresponding hand shape. Taha *et al.*[17] developed a model that produces the preshaping of a planar hand model for circular and rectangular objects.

Most of these studies emphasize the correspondence between an object and a hand shape. One can argue that the same grasping posture can be adopted to grasp objects of various shapes, and one important factor that affects grasping of an object may be to recognize its graspable feature instead of considering its general shape. If a neural model is able to acquire a representation of the graspable features or affordances,[18] this would enable enhanced flexibility for grasp planning. Moreover, one can expect to integrate task requirements through an appropriate choice of the graspable features. Moussa and Kamel[19] followed such an approach and proposed a computational architecture able to learn grasping rules called "generic grasping functions". This representation treats grasping knowledge as a mapping between two reference frames: the object body attached coordinate frame, and the gripper body attached frame.

Since this paradigm seems to be attractive, how can it be applied to a multifingered hand where the mapping between the hand and the object frames may not be sufficient? One of the aims of this paper is to try to give an answer to this last question by taking into account local information linked to a grasp in a neural network-based approach. This new formalism allows us to build the posture of a multifingered hand model just from the knowledge of the desired fingertip position. Moreover, an interesting feature of this model is that it takes into account noisy information coming from sensing devices and uncertainty inherent in any real-world application. Instead of directly considering the graspable features, we use the notion of contact configuration (with a variable number of involved fingers) on the object. In order to make the process tractable (i.e. not considering all the hand degrees freedom in one step), we have separated the resolution in two phases: (i) A neural scheme treats the inverse kinematics mapping of all the fingers; (ii) An optimization of the hand reference frame location and orientation relative to a global reference frame is then carried out.

This paper is organized as follows: In part 2, we first describe the model architecture and the chosen hypotheses. Details about the retained scheme developed to learn the inverse kinematics of the fingers are given in part 3. Then, section 4 describe the procedure used to define and optimize the hand configuration. Finally in section 5, we propose simulation results to demonstrate the efficiency of the proposed tools.

2. MODEL PRESENTATION

2.1. Problem statement

The two main important difficulties encountered in the process of grasping posture learning are the hand high degree of redundancy and the high dimension of the search space. In order to make this problem tractable, we have reduced a large problem to several smaller ones that can be solved more easily.

Firstly, we propose a reasonably simplified hand model composed of 5 articulated rigid chains representing the fingers which are connected to a common body representing the palm (Figure 1). Each finger has 4 degrees of freedom. Using the results from studies on hand anthropometry[20,21], we determine the hand geometrical parameters from the knowledge of its global length and breadth.

The problem statement is: *Given an object and a contact configuration, we want to define all the kinematics parameters of a hand model in such a way that all fingertips can reach a defined contact position on the object surface.* To define the hand posture, we have made some assumptions presented in Figure 2 related to the input and output data of the model.

2.2. Retained approach

In order to define the hand configuration, we have considered two kinds of data: the first is related to the posture of the fingers characterized by joint angles and the second to the location and orientation of the hand model attached coordinate frame. In this way, each finger can be treated separately and we can define two separated processes for fingers and hand postures definition.

The first step is to solve finger inverse kinematics. For each of them, this learning is performed off-line. It is based on a supervised learning method that takes into account the discontinuity of the finger inverse kinematic function with its joint limits (details in section 3).

Then, it is necessary to define the hand configuration in such a way that the fingertips reach their desired location on the object with the minimum error. This task is made difficult because several fingers are involved and a good solution for one finger may not be valid for the others. The second point is that several solutions may exist. In this case, a supervised learning method cannot be used because we do not have a teacher that can provide pre-specified target action or error gradient vectors to specify how the controller has to modify its action so as to improve the performance.[21] Reinforcement learning seems to be well suited to solve this problem because it has the interesting feature of finding the right outputs for a given input through an experimental strategy and also to memorize those outputs. In the reinforcement learning technique, we do not instruct explicitly the controller what action to perform but rather if it is going in the right or in the wrong direction (in the sense of an evaluative criterion) and how to modify its parameters to converge toward a valid solution. Another aspect concerns the choice of the process to infer the appropriate actions from evaluations. Gullapalli[22,23] presented the two major categories, indirect and direct methods. Indirect methods involve constructing a model of the transformation from the controller's action to the evaluation. This model enables to obtain a gradient information used for training the controller.[22] On the other hand, direct methods perform such a task by directly perturbing the process and, from the produced effects on the performance evaluation called a critic (which has a similar function as the teacher in supervised learning), find the right action to apply. The perturbation is usually a random noise with known properties and, as suggested by Gullapalli,[23] a stochastic search for the best action is performed. The direct methods seem to be very attractive, since no model construction is required. In fact, building a process model that can provide gradient information about its performance relative to the learning parameters is even more difficult if we intend to take into account the presence of noise and uncertainty. Moreover, with direct methods, it is the process itself that provides the necessary training data. These are the reasons why we have chosen such a formalism to determine the hand configuration by means of a neural network composed of backpropagation and SRV (*Stochastic Real Valued*) units (detailed in section 4). The main interest of Stochastic Real Valued units is that they permit the learning of functions with continuous outputs using a connectionist network.[23]

2.3. Model structure and principle

The global structure of the model is composed of two neural modules: The main module (Figure 3), called "Hand Configuration Neural Network" (HCNN), is devoted to the determination of the hand global attached frame configuration (location and orientation) relative to a global frame. It is a multilayer feed forward neural network that implements a reinforcement strategy through the use of SRV neurons in the HCNN output layer. Its detailed structure is presented in section 4. The secondary module, called "Finger Configuration Neural Network" (FCNN), is devoted to finger configuration definition. Its output data are used to evaluate the appropriateness of the hand configuration through an evaluative function and the corresponding reinforcement signal is sent back to the HCNN. In section 3, the FCCN modular neural network architecture is presented in details.

The principle of the developed method is now described:

(i) We input to the model the current hand configuration and the rate of change of the hand configuration and, as output, we obtain a new hand configuration within predefined search space bounds,
(ii) Then, using the new hand configuration, we are able to express the contact location in each finger root frame (on

https://doi.org/10.1017/S0263574704000025 Published online by Cambridge University Press

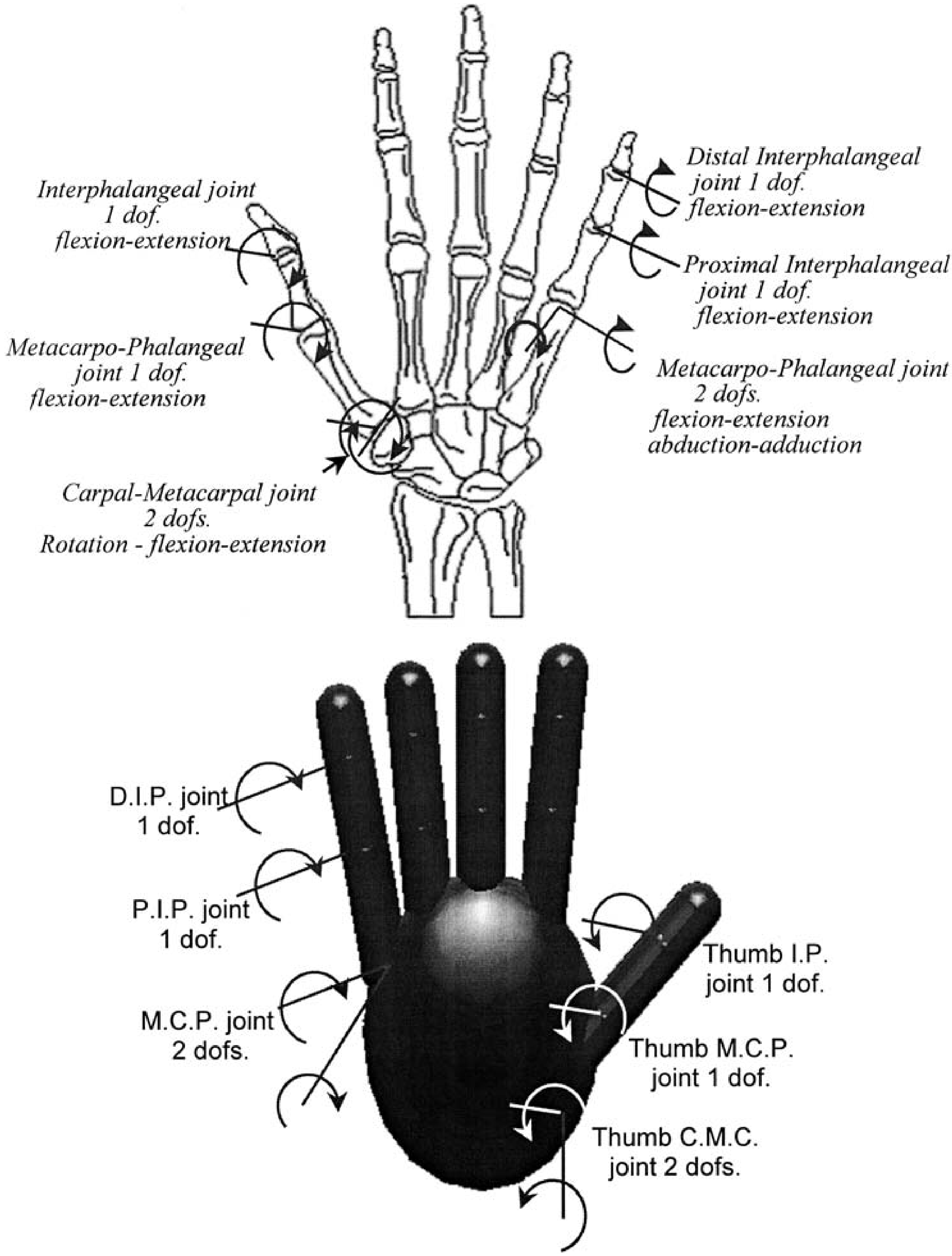

Fig. 1. 3D model of the hand with a representation of the different degrees of freedom associated with each joint. (P.I.P.: proximal interphalangeal, D.I.P.: distal interphalangeal, M.C.P.: metacarpophalangeal, I.P.: interphalangeal, C.M.C.: carpal-metacarpal).

the surface of the object, a particular contact is meant to each finger),

(iii) Using the inverse kinematics scheme, we compute each finger posture. We test them by an evaluative function and compute a reinforcement signal that is used to update the hand configuration neural network. Before performing this last step and, in order to introduce noise and uncertainty, the actual reinforcement is perturbed with a zero mean and predefined variance noise,

(iv) We perform this procedure (i.e. we go back to step (i) until a satisfactory solution is found or until the maximum number of iteration is reached.

https://doi.org/10.1017/S0263574704000025 Published online by Cambridge University Press

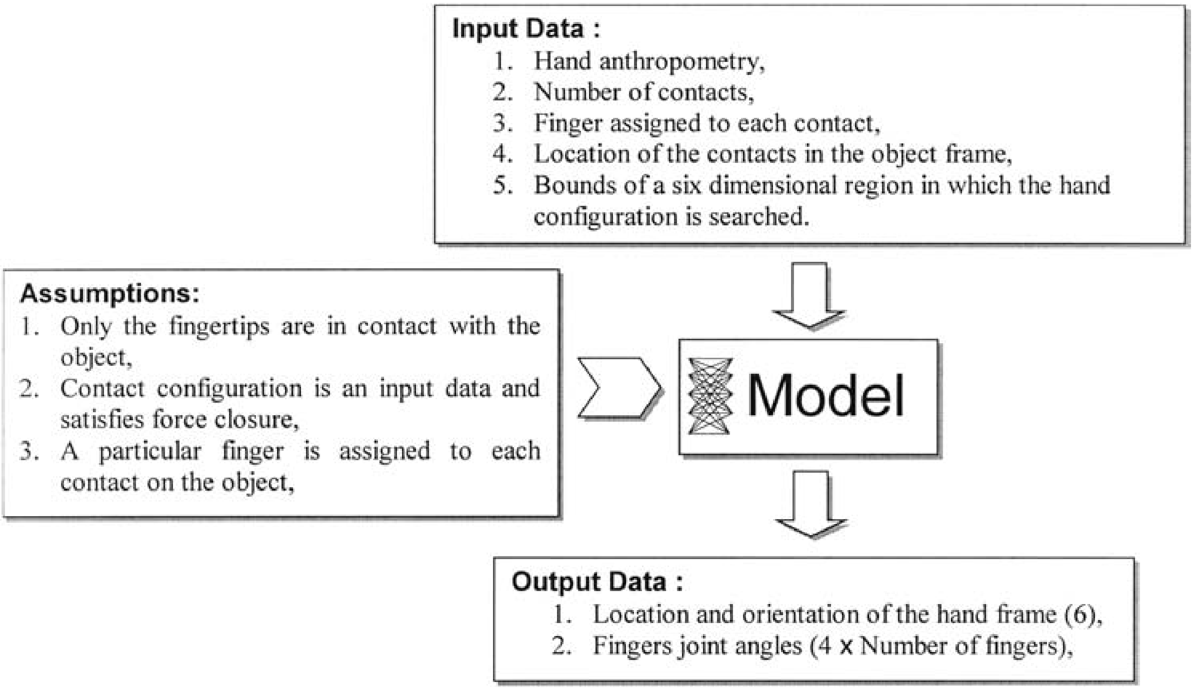

Fig. 2. Assumptions, input and output data of the model.

The evaluative function is based on the computation of the distance between the fingertip position and its desired location on the object. This is an easy task because, in the previous step, we have computed the location of the contact in the finger frame. Once a finger configuration is obtained by inverse kinematics, we compute the fingertip position by forward kinematics.

$$\text{Let } \mathbf{PX}_i^D = \left(x_i^D, y_i^D, z_i^D\right)^T \tag{1}$$

be the vector of the desired fingertip position expressed relative to the root coordinate frame of finger i at step k and

$$\mathbf{PX}_i^M = \left(x_i^M, y_i^M, z_i^M\right)^T \tag{2}$$

be the vector of the actual fingertip position expressed relative to the root coordinate frame of finger i. If n fingers are involved, the total error obtained at step k is:

$$E_k = \sum_{i=1}^{n} \left\|\mathbf{PX}_i^D - \mathbf{PX}_i^M\right\| \tag{3}$$

with $\|\cdot\|$ defining the Euclidean L2 norm.

In order to model the effect of low quality sensing information, this evaluative function is perturbed with a noise

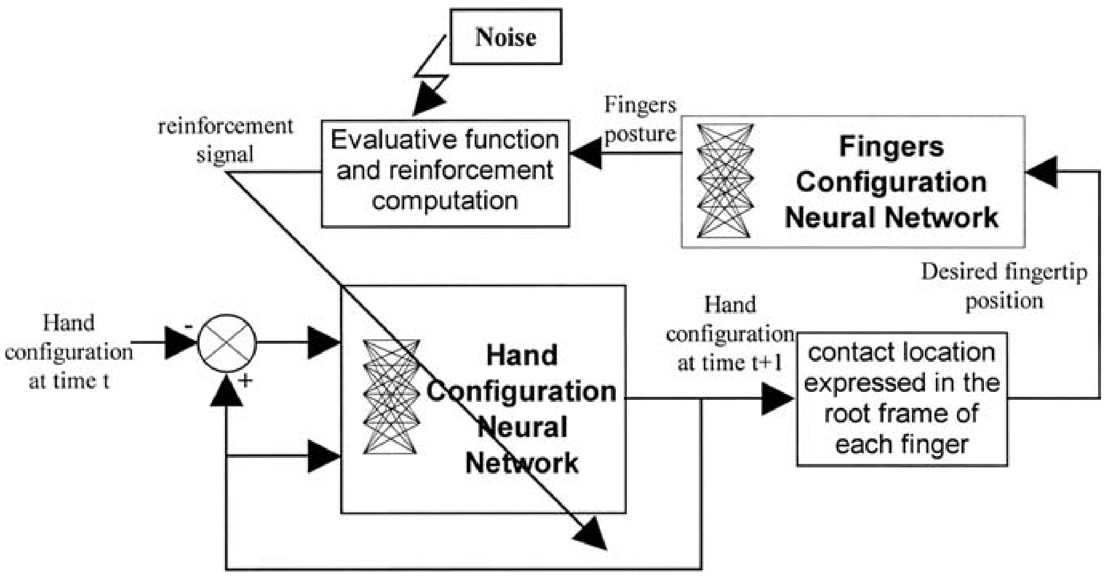

Fig. 3. Architecture of the hand posture definition model.

https://doi.org/10.1017/S0263574704000025 Published online by Cambridge University Press

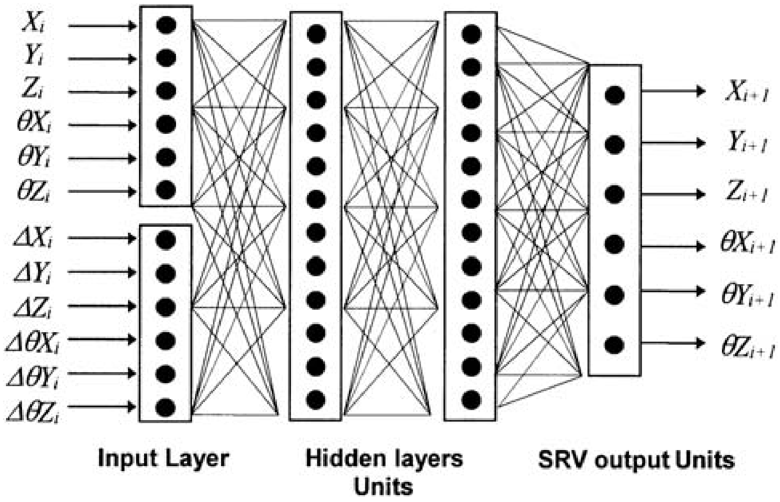

Fig. 4. Structure of the Hand Configuration Neural Network.

of known properties. The chosen procedure is detailed in section 4.

3. FINGER INVERSE KINEMATICS NEURAL SCHEME

From a contact configuration in the object reference frame and a hand configuration (position and orientation) the problem is to compute the finger configuration in the joint coordinate space.[24] We have used a slightly modified version of the modular neural networks architecture defined by Oyama and Tachi.[25,26] The main advantage of this method is that it takes into account the discontinuity of the inverse kinematic system of multi-joints mechanisms with its joint limits. This represents a difficult function to be approximated by a single multi-layers network. The principle of the formalism defined by Oyama and Tachi[25,26] is to learn a discontinuous inverse kinematic function by an appropriate switch of multiple neural networks representative of a sub-set of the whole chain workspace. Therefore, the architecture is composed of several units (called experts) that can learn the inverse kinematics solution over the whole chain workspace. It works as follows:

(i) The desired endpoint position is chosen with a random number generator. Each expert produces an output, which is tested. For each of them, the current chain endpoint position is computed using a forward model. The output error is the Euclidean distance between the current and the desired endpoint position. Then a reaching motion represented by a straight line is constructed in Cartesian and joint spaces from the current to the desired fingertip position. The number of points that defines this motion is a function of the produced error. If the error is large, several points are stored. If the error is below a predefined threshold a single point is considered. Finally, the points of the reaching motion are stored in the training set of the expert that has produced the minimal error among all the experts and learned by the corresponding expert,

(ii) If no satisfactory solution is obtained for the best expert due to joint limits or if the chain reaches a singular position, another expert is selected in an increasing order of the predicted error, and the same procedure as in step (i) is applied until a satisfactory solution is found or all the experts are tested,

(iii) If no satisfactory solution is obtained from steps 1 and 2, an expert is randomly chosen and a reaching motion is performed from the expert representative posture to the desired endpoint position. This procedure is repeated until a solution is found or all the experts are tested,

(iv) If no solution is obtained from steps 1, 2 and 3, a posture is randomly computed and a reaching motion is performed between the chain endpoint and the desired position. This procedure is repeated until a satisfactory motion is generated (i.e. no singular position and joints within their limits). If so, the generated posture is considered as the representative posture of a new expert and is added to the set of experts.

The expert networks have four layers, the activation function of the input and output layers is linear and tangent sigmoid for the other layers. The input layer is composed of three units that correspond to the X, Y and Z position of the fingertip. The output layer has 4 units corresponding to the 4 joints of each finger. The two hidden layers have 35 units each. The error backpropagation algorithm is used to update the neurons' weights. During the simulation, four to six experts were generated for each finger to perform the mapping of their corresponding workspace.

4. HAND CONFIGURATION NEURAL NETWORK

The neural network used to optimize the hand configuration has 2 hidden backpropagation layers and one output layer composed of SRV units (Figure 4). The input layer has 12 units (6 for the location and orientation of the hand attached coordinate frame and 6 for the linear and angular velocities). The 2 hidden layers have 20 units each. The association of SRV and backpropagation units enables are to take advantage of both supervised and reinforcement learning. The whole network still has the benefits of

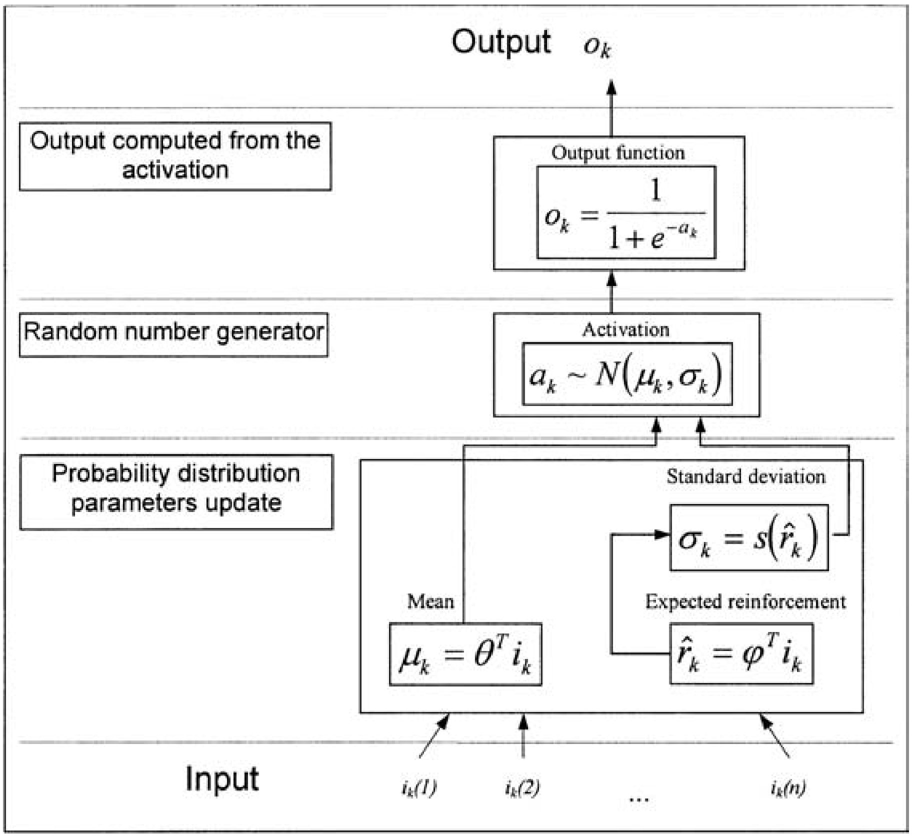

Fig. 5. Input-output relation for a Stochastic Real Valued (SRV) unit, adapted from Gullapalli (1995).

reinforcement learning due to its stochastic search behavior. Also, the BP units in the hidden layer allow us to develop the right internal distributed representation as it is in the case in supervised learning.

The next step is the description of the stochastic real valued (SRV) units, its input-output relation and its stochastic behavior. The SRV units compute their output according to the following procedure. An input vector i_k from $X \subseteq \Re^n$,

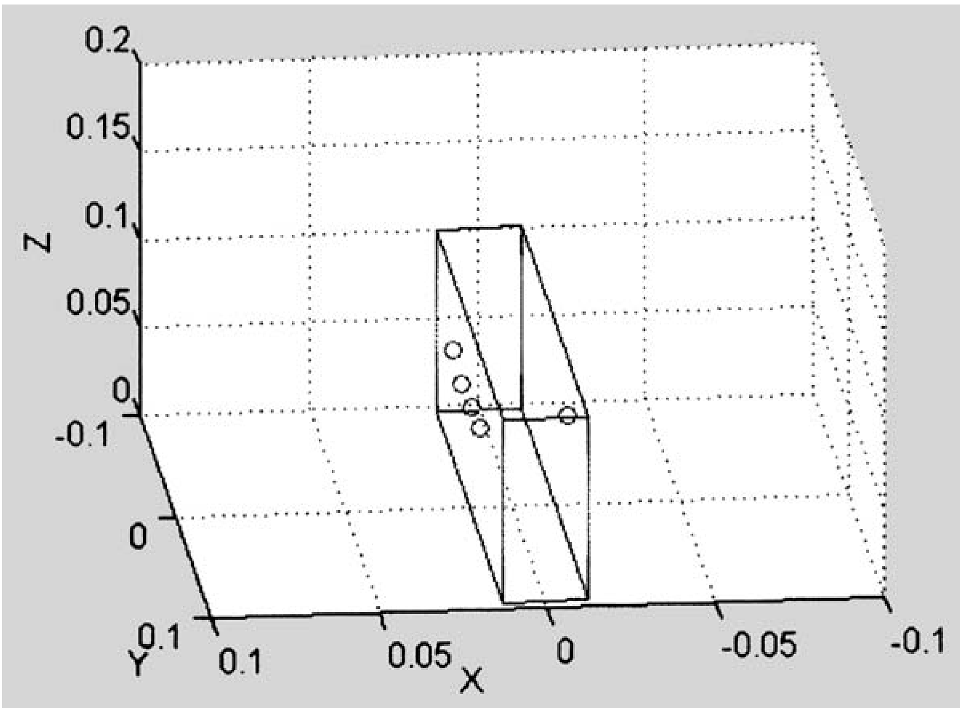

Fig. 6. Defined contact configuration for the five fingered rectangular block grasp.

https://doi.org/10.1017/S0263574704000025 Published online by Cambridge University Press

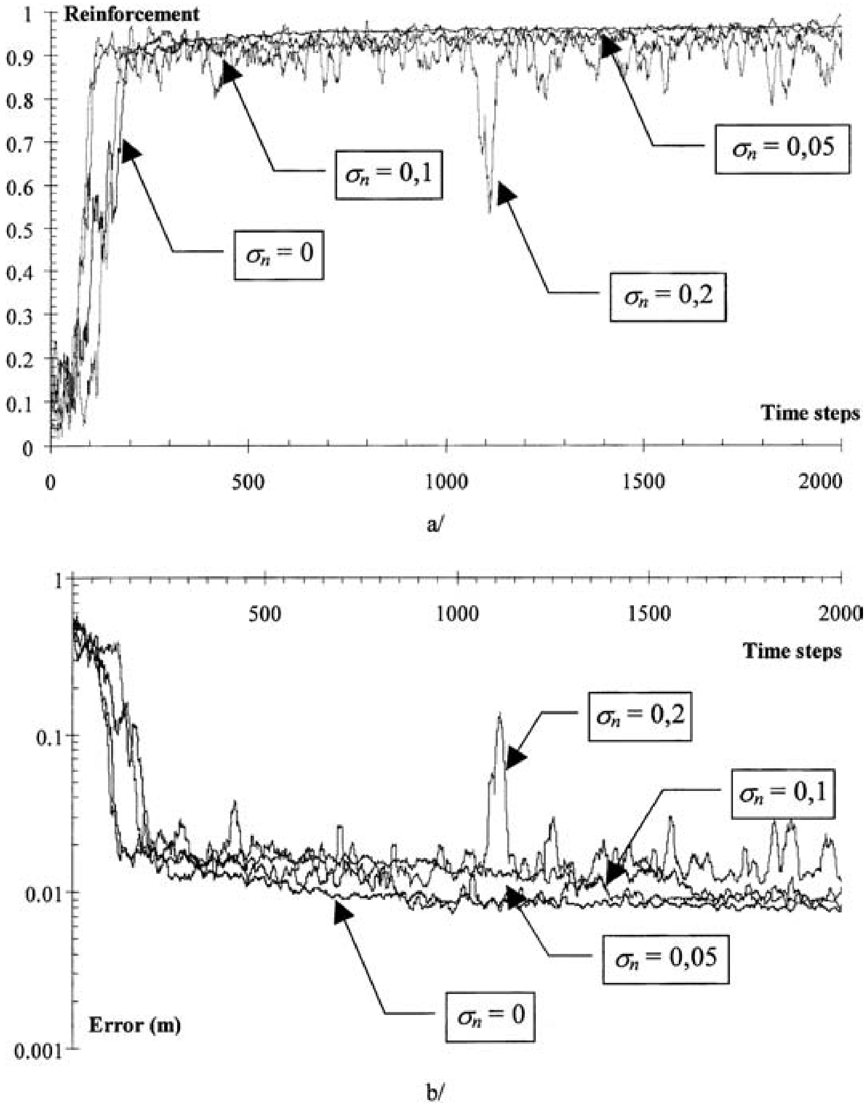

Fig. 7. a/ Reinforcement signal and b/ corresponding error evolution during each trial with four different noise variances.

where $\mathfrak{R}$ is the set of real numbers, is presented to a SRV unit at time step k. The unit produces a random output o_k selected from some internal probability distribution over the interval $O \subseteq \mathfrak{R}$. The SRV unit uses its input i_k to compute the parameters μ_k and σ_k of the internal normal probability distribution (μ_k the mean and σ_k the standard deviation). These parameters computed internally are obtained as the weighted sum of the input i_k with a particular set of weights for each parameter. We summarize the input-output relation for a SRV unit in Figure 5.

In order to obtain a hand configuration within the desired bounds, the network output vector o_k is scaled according to the following equation:

$$\mathbf{X}_{i+1} = \mathbf{X}_{\min} + (\mathbf{X}_{\max} - \mathbf{X}_{\min}) \otimes o_k \tag{4}$$

$\mathbf{X}_{i+1}$ denotes the new hand configuration, $\mathbf{X}_{\min}$ the lower bounds of the search space, $\mathbf{X}_{\max}$ the upper bounds of the search space (Table I), o_k the network output vector and $\otimes$ the outer vector product.

The environment evaluates the new arm configuration $\mathbf{X}_{i+1}$ according to the evaluation function (1–3) and the context of i_k and returns a reinforcement signal $r_k \in R = [0, 1]$, with $r_k = 1$ denoting the maximum possible reinforcement. Therefore, the reinforcement signal value is obtained as follows:

$$r_k = 1 - h(E_k) \tag{5}$$

where E_k (3) corresponds to the error at time step k obtained through the evaluation. h is a monotonic increasing function of the error E_k taking values over the interval [0, 1]. If E_k is large, h tends towards 1 and the network receives a maximum punishment with a reinforcement toward 0. On the contrary, if the error E_k is low, h tends toward 0 and, consequently, the system receives a higher reinforcement through equation (5). In the present case, we have chosen the tangent sigmoid function for h.

Table I. Search space bounds for the rectangular block five fingered grasp.

	X (m)	Y (m)	Z (m)	6X (°)	6Y (°)	6Z (°)
min. value	–0.1	–0.1	0.15	–45	–45	–135
max. value	0.1	0.1	0.25	45	45	–45

https://doi.org/10.1017/S0263574704000025 Published online by Cambridge University Press

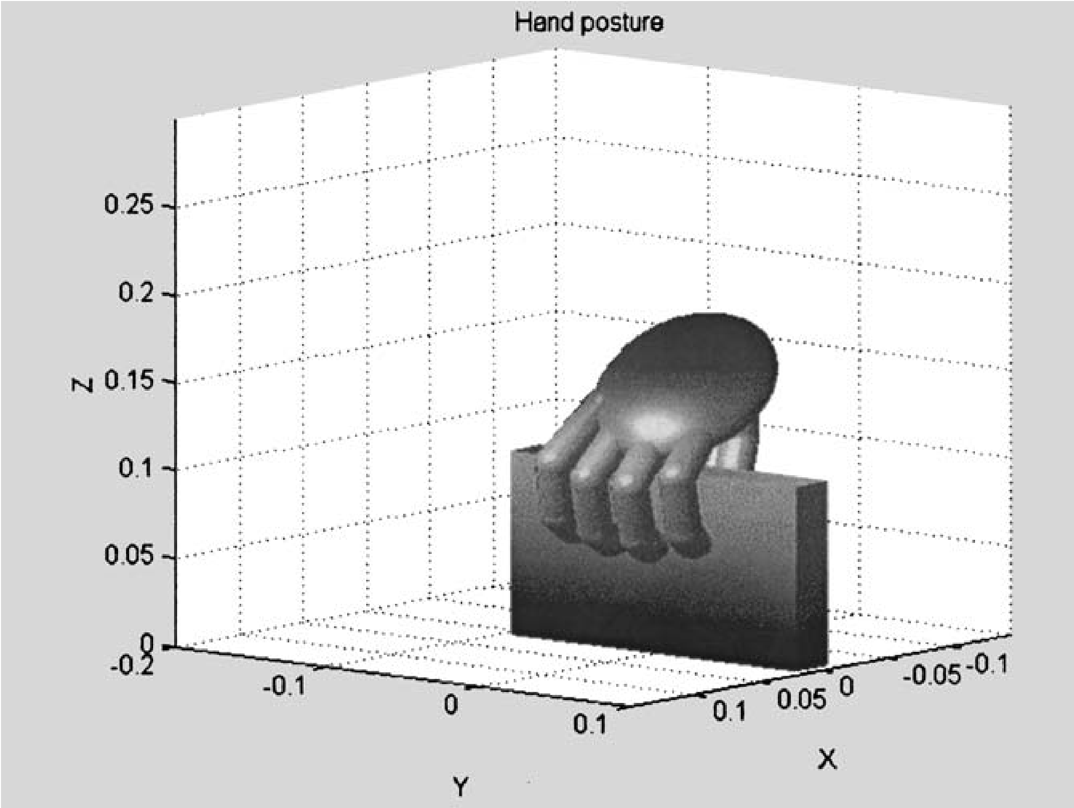

Fig. 8. Obtained hand configuration for a five fingered rectangular block grasp.

In order to model low sensing quality and noise effect, the actual reinforcement is perturbed with a random noise with zero mean and known variance. This noise reflects the quality of hand position sensors providing information relative to the hand global position and orientation as well as to the finger joint position.

To update the two parameters $\theta(k)$ and $\varphi(k)$ used to compute the mean μ_k and standard deviation σ_k, the following learning rules are used:

$$\theta(k+1) = \begin{cases} \theta(k)+\alpha\left((r(o_k,i_k)-\hat{r}_k)\left(\frac{a_k-\mu_k}{\sigma_k}\right)\right)(i_k) & \text{if } \sigma_k>0 \\ \theta(k) & \text{if } \sigma_k=0 \end{cases} \tag{6}$$

$$\varphi(k+1) = \varphi(k) + \rho(r(o_k, i_k) - \hat{r}_k)i_k \tag{7}$$

where

$\mu_k = \theta_k^T i_k$ is the mean of the internal probability distribution,
$\sigma_k = s(\hat{r}_k)$ is the standard deviation of the internal probability distribution,
$\hat{r}_k = \varphi_k^T i_k$ is the expected reward,
$a_k \sim N(\mu_k, \sigma_k)$ is the unit activation drawn from the normal probability distribution,

and $r(o_k, i_k) = r_k$ is the reward or reinforcement obtained from equation (5).

The update rules are designed to produce the following behavior. If the normalized perturbation added to the mean output of the unit (6) leads to a reward that is greater than the expected reward, then, it is likely that the unit produces an output that is closer to the actual output. In other words, the mean should be changed in the direction of the perturbation that has produced a better reward and the unit should update its weights accordingly. In the opposite case, i.e. the reward is less than the expected reward, then the mean should be moved in the opposite direction. In both cases, there is a tendency to increase the reward, by moving the mean in such a direction so as to produce a better reward, thus going away from regions leading to poorer results. The second important point in the learning rule (7) is that the standard deviation depends on the expected reward. In this way, an SRV unit can control the extent of search through the standard deviation value. In fact, as the expected reinforcement increases, the standard deviation decreases (5) and, therefore, the search space is narrowed in the neighborhood of the mean output.

Since SRV output units are used, the error gradient signal is not available because there is no desired output. Moreover, as Gullapalli[22] stated, randomly perturbing the mean output and observing the consequent change in the evaluation enables the unit to estimate the gradient of the evaluation with respect to the output. Therefore to train the backpropagation layers, the actual error is replaced with an estimated error gradient of the following form:

$$\partial_n^{SRV} = \frac{(r(o_k, i_k) - \hat{r}_k)(a_k - \mu_k)}{\sigma_k} \tag{8}$$

where $r(o_k, i_k)$ is the reward, $\hat{r}_k$ the expected reward, a_k the activation and μ_k the mean output. To propagate the error gradient of the *SRV* units back to the *BP* units, we have used

https://doi.org/10.1017/S0263574704000025 Published online by Cambridge University Press

the following equation:

$$\partial_n^{BP} = \Theta^T \cdot \partial_n^{SRV} \tag{9}$$

where $\Theta[i, j]$ is the weight θ used to compute the mean parameter of the ith *SRV* unit from the jth *BP* unit's output (considered as input j of the *SRV* unit). With this properly propagated error gradient (9), we can train the *BP* layers.

5. HAND CONFIGURATION SIMULATION

In this section, we present simulation results. The test case is to construct the hand posture to grasp a rectangular block with five fingers. This is the most difficult case because the algorithm has to find a suitable hand position and orientation taking into accounts all the fingers. The contact configuration is displayed in Figure 6.

In order to identify the influence of uncertainty and noise reflecting low quality sensors, we have considered four levels of random noise with zero mean and standard deviation σ_n of 0.2, 0.1, 0.05 and 0 (which corresponds to a deterministic reinforcement, $r_k \in [0, 1]$). Each trial has a duration of 2000 steps and the bounds of the workspace are defined in Table I. In order to have a satisfying convergence, we use low learning rates (0.01 for BP layers, 0.01 for SRV units mean and standard deviation). In practice, these values have to be low to allow the learning under uncertainty. In Figure 7a, we display the obtained reinforcement with the four noise standard deviation levels and, in Figure 7b, the corresponding error (deterministic). We can clearly see that the algorithm succeeds to find a solution even if the curves are more irregular for large standard deviation (which is understandable). We can also notice that the convergence is of the same order than the trial with deterministic reinforcement attesting the model robustness to noise. Finally, a satisfactory solution is obtained after a relatively low number of time steps.

In Figure 8, we display an obtained hand configuration with $\sigma_n = 0.1$.

CONCLUSION

In this paper, we have proposed a new model that enables the definition of all the kinematics parameters related to a hand configuration. This task is performed in the presence of uncertainty and noise inherent in any real-world application. In the first part, we have proposed a modular scheme based on neural network for the definition of the finger posture. The second stage of the model takes advantage of a reinforcement scheme to perform a search on the hand attached coordinate frame (of dimension six) in order to allow each fingertip to reach its desired location on the object surface. Several simulation results demonstrate the capability of this scheme to construct hand postures with varying noise levels. The fact that no candidate solution is required to start the hand posture construction and that the number of iterations is quite low, are interesting properties of this method. In the near future, instead of considering a hand configuration problem, we plan to work directly at the arm level. Also, further development will be held to define more sophisticated evaluative functions in order to incorporate task and environment constraints.

Acknowledgment

This work was supported by European 5th Framework program grant: BRITE Project Basic Research BE 4505.

References

1. G. Schlesinger, *Der Mechanische Aufbau der Kunstlischen Glieder* (M. Borchardt et al., eds.) (Springer, Berlin, 1919) pp. 321–699.
2. J. R. Napier, "The prehensile movements of the human hand", *Journal of Bone and Joint Surgery* **38**, 902–913 (1956).
3. M. R. Cutkosky, "On grasp choice, grasp Models, and the design of hands for manufacturing tasks", *IEEE Transactions on Robotics and Automation* **5**, 269–279 (1989).
4. T. Iberall, G. Bingham and M. A. Arbib, "Opposition space as a structuring concept for the analysis of skilled hand movements", *Experimental Brain Research* **15**, 158–173 (1986).
5. T. Iberall, "Human prehension and dexterous robot hands", *Int. J. Robotics Research* **16**(3), 285–299 (1997).
6. F. Saito and K. Nagata, "Interpretation of grasp and manipulation based on grasping surfaces", *Proceedings of the 1999 IEEE International Conference on Robotics and Automation* (1999) pp. 1247–1254.
7. R. Tomovic, G. A. Bekey and W. J. Karplus, "A strategy for grasp synthesis with multifingered hand", *Proceedings of the 1987 IEEE International Conference on Robotics and Automation* (1987) **Vol. 1**, pp. 83–89.
8. G. A. Bekey, H. Liu, R. Tomovic and W. J. Karplus, "Knowledge-based control of grasping in robot hands using heuristics from human motor skills", *IEEE Transactions on Robotics and Automation* **9**(6), 709–721 (1993).
9. S. B. Kang and K. Ikeuchi, "Toward automatic robot instruction from perception – Mapping human grasps to manipulator grasps", *IEEE Transactions on Robotics and Automation* **13**(1), 81–95 (1997).
10. J. Kerr and R. Roth, "Analysis of multifingered hands", *Int. J. Robotics Research* **4**, 3–17 (1986).
11. V. Nguyen, "Constructing force-closure grasps", *Proceedings of the 1986 IEEE International Conference on Robotics and Automation.* (1986) pp. 1368–1373.
12. N. Pollard and T. Lozano-Perez, "Grasp stability and feasibility for an arm with an articulated hand", *Proceedings of the 1991 IEEE International Conference on Robotics and Automation.* (1991) pp. 1581–1585.
13. K. Woelfl and F. Pfeiffer, "Grasp strategies for a dextrous robotic hand", **In**: (V. Graefe, ed.), *Intelligent Robots and Systems* (Elsevier Science, 1995) pp. 259–277.
14. P. Gorce, and J. G. Fontaine, "Design methodology for flexible grippers", *Journal of Intelligent and Robotic Systems* **15**(3), 307–328 (1996).
15. M. Kuperstein and J. Rubinstein, "Implementation of an adaptative controller for sensory-motor condition", **In**: (R. Pfeifer, Z. Schreter, F. Fogelman-Soulie and L. Steels, eds.) *Connectionism in Perspective* (Elsevier Science Publishers, 1989) pp. 49–61.
16. Y. Uno, N. Fukumura, R. Suzuki and M. Kawato, "A computational model for recognizing objects and planning hand shapes in grasping movements", *Neural Networks* **8**, 839–851 (1995).
17. Z. Taha, R. Brown and D. Wright, "Modelling and simulation of the hand grasping using neural networks", *Medical Engineering and Physiology* **19**, 536–538 (1997).
18. A. H. Fagg and M. A. Arbib, "Modeling parietal-premotor interactions in primate control of grasping", *Neural Networks* **11**(8), 1277–1303 (1998).
19. M. A. Moussa and M. S. Kamel, "An experimental approach to robotic grasping using a connectionist architecture and generic grasping funtions", *IEEE Transactions on System Man and Cybernetics* **28**, 239–253 (1998).

20. J. W. Garret, “The adult human hand: some anthropometric and biomechanical considerations”, *Human factors* **13**, 117–131 (1971).
21. B. Buchholz, T. J. Armstrong and S. A Goldstein, “Anthropometric data for describing the kinematics of the human hand”, *Ergonomics* **35**, 261–273 (1992).
22. V. Gullapalli, “A stochastic reinforcement learning algorithm for learning real valued functions”, *Neural Networks* **3**, 671–692 (1990).
23. V. Gullapalli, “Direct associative reinforcement learning methods for dynamic systems control”, *Neurocomputing* **9**, 271–292 (1995).
24. N. Rezzoug and P. Gorce, “A neural network architecture to learn hand posture definition”, *IEEE International Conference on Systems Man and Cybernetics* (2001) pp. 3019–3025.
25. E. Oyama and S. Tachi, “Inverse Kinematics Model Learning by Modular Architecture Neural Networks”, *Proceedings of the 1999 IEEE International Joint Conference on Neural Network* (1999) pp. 2065–2070.
26. E. Oyama and S. Tachi, “Modular Neural Net System for Inverse Kinematics Model Learning”, *Proceedings of the 2000 International Conference on Robotics and Automation* (2000) pp. 3239–3246.