Rapid On-Line Learning of Compliant Motion for Two-Arm Coordination

Jianwei Zhang and Markus Ferch

Technical Computer Science, Faculty of Technology,
University of Bielefeld, 33501 Bielefeld, Germany

Abstract

To enable a two-arm manipulator system to perform complex cooperating tasks such as carrying a rigid object, establishing a contact to a surface, grinding, etc., a non-linear multiple input system should be solved. We apply an approach to on-line automatic learning of a B-spline fuzzy controller. This controller model directly connects the sensor inputs to the compensation motion. By using the adaptation of control actions in all possible situations through practising of the robots in the real environment, uncertainties of the robot-object model can be taken into account. The compliant motion controller of the robot system can be adapted to new situations in a short time thanks to the on-line learning approach.

1   Introduction

This work discusses the problem of automatic learning of a controller which guides the motion of one robot arm to cooperate with the other one based on the wrist-mounted force/torque sensor. Examination of this problem is motivated by applications such as transport of objects with large size or heavy weight, and assembly of these objects. The motion of the robots must be controlled so that neither the objects nor the robots are damaged. For example, in the two-arm transportation task, the measured forces and torques on the two manipulator wrists should be kept as small as possible, and in a two-arm grinding task, be kept at a certain desired value.

A lot of research work on this problem employed the approach of modelling the manipulator dynamics and computing torques, e.g. [7]. If the dynamic model of a manipulator is *a priori* known, such an approach supplies an explicit physical interpretation of the motion-force process in form of differential equations. Therefore, stability of such a control system can be analysed. Unfortunately, for compliant motion control using industrial robots, neither the parameters in the dynamic model of the robot is available nor is there a possibility to access the joint controller directly.

Another important approach is the active[1] two-arm coordination based on forces and torques measured in the Cartesian space, e.g. [8, 1]. The compliant motion is realised by adjusting the stiffness of the controller using any type of PID control, the second-order low pass filter algorithm, or in frequency area. Since there are numerous uncertainties in a real robot model, like backlash, object internal tension forces, the imprecise modelling of real inertia parameters, etc., it is desirable that the stiffness of such a compliant motion controller can be automatically adapted to different robots, objects and manipulation tasks.

Robot learning aims at generating robot software in an evolutionary way. Recently, some work using learning has been reported. Off-line supervised learning [4] must utilise data from human demonstration, and it cannot be guaranteed that the optimal controller is trained even if the human instructor demonstrated his best skills. On-line learning does not need the extra training phase anymore. [5] discussed the training of a fuzzy-neural controller for position/force control through back-propagation and gave some simulation results. To train a controller for contour tracking based on force feedback, [6] used a neural network method. Reinforcement learning was applied in one-arm pendulum swing-up problem [3].

In this paper we present a practical approach for learning the nonlinear relation of the forces/torques and the compliant motion in the Cartesian space. For this purpose, the B-spline model [9] is employed, which can be classified into a neuro-fuzzy method. Its basic idea of partitioning input space with overlapping functions coincides with the CMAC [2] principle. Our early experiments with the B-spline model on numerous benchmark problems of modelling and also in mobile robot behaviour learning [10] have shown the good modelling capability of nonlinear relations, smooth output and rapid convergence of learning. In this work, the principle of B-spline controllers is applied to the learning of compliant motion. Our experiments with

---

[1]In comparison with the passive approach with a RCC device.

real robot systems show their suitability in typical two-arm coordination tasks.

2   System and Problem Description

We use two PUMA 260 industry manipulators hanging faced down (Fig. 1). Each manipulator is equipped with a force/torque sensor (type JR3 67M25A) that is mounted between the last (sixth) joint and a pneumatic two-finger gripper.
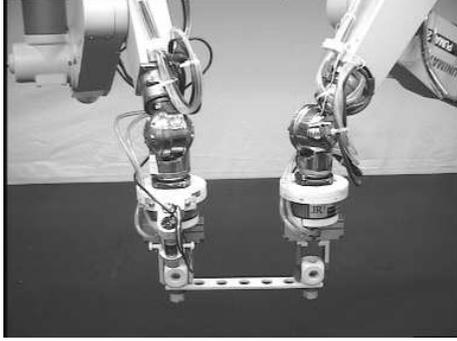

Figure 1: The two Robots holding a rigid object with two parallel jaw-grippers.

2.1   Motion in Closed Kinematic Chain

The basic idea of moving the rigid object itself instead of each robot arm is to put both manipulators in a single kinematic chain which meets the demand of force control. When moving the object through the space without compensation, both manipulators make different position errors which are caused by the joint controllers and the piecewise trajectory generation. Therefore, undesirable forces and torques can be measured as shown in Fig. 3. These forces and torques are to be minimised.

2.2   Compliant Motion

We adopt the principle of compliant motion which makes the manipulators behave similarly to an ideal spring. When s is a displacement and C is a spring constant, the force that is exerted on a body is given by Hook's law:

$$(2.1) \qquad F = -C \cdot s.$$

The minus sign shows that the force tends to move the body back towards the equilibrium position at $s = 0$. For more dimensions (forces and torques) the following compliance function can be defined:

$$(2.2) \qquad \begin{bmatrix} d\vec{x}(t) \\ d\vec{\phi}(t) \end{bmatrix} = \mathcal{C} \begin{bmatrix} \vec{f}(t) \\ \vec{m}(t) \end{bmatrix}$$

Where $\mathcal{C}$ takes the part of the spring constant. It is a 6×6 matrix expressing the programmed compliances in a special coordinate frame.

Let the robot position with respect to the base frame B be $\vec{x}(t)$. It can be modified by $d\vec{x}$ as follows:

$$(2.3) \qquad \vec{x}(t+1) = \vec{x}(t) + d\vec{x}(t).$$

The orientation expressed in the base frame B is $\mathcal{R}(t)$ and has to be modified:

$$(2.4) \qquad \mathcal{R}(t+1) = \mathcal{R}_{d\vec{\phi}}(t)\mathcal{R}(t),$$

with $\mathcal{R}_{d\vec{\phi}}(t) = \text{Rot}(\hat{z}_B, \delta_z(t))\text{Rot}(\hat{y}_B, \delta_y(t))\text{Rot}(\hat{x}_B, \delta_x(t))$ and $d\vec{\phi}(t) = (\delta_x(t), \delta_y(t), \delta_z(t))^T$. More details can be found in [8].

3   Fuzzy Control Based on B-Spline Model

The components of the $\mathcal{C}$ - matrix are unknown and can vary with the robots position and orientation as well as with the environment. Therefore, it is desirable that the nonlinear stiffness parameters can be automatically determined by the controller itself. In the following we briefly show that a B-spline fuzzy controller provides an appropriate model.

In a B-spline fuzzy controller, the membership functions are B-spline basis functions. Details can be found in [9]. B-splines of low orders can be represented explicitly instead of recursively, which allows a much faster calculation. In this work, we employ B-spline basis functions of third order since a). they allow a good balance between calculation speed and smoothness; and b). sufficient locality enables the convergence of learning.

3.1   Convergence of Learning

There are some properties such as "partition of unity", "local support" of the B-spline model that bring the rapid convergence for supervised learning. A general multiple-input B-spline fuzzy controller can be explicitly represented as:

$$(3.1) \qquad y(\vec{x}) = \sum_{i_1=0}^{m_1} \cdots \sum_{i_q=0}^{m_q} d_{i_1 \ldots i_q} \prod_{j=1}^{q} N_{i_j}^{n_j}(x_j)$$

where x_j is the j-th component of $\vec{x}$.

In a B-spline fuzzy controller, there is one control vertex for each rule, which fires to a grade $\prod_{j=1}^{q} N_{i_j}^{n_j}(x_j)$.

For an input vector $\vec{x}$, if the desired output y_d is known, the learning function by using gradient descent based on the quadratic error function $E = \frac{1}{2}(y_r - y_d)^2$ (y_r is the

real output value of the current trained controller) can be derived:

(3.2)
$$d_{i_1,i_2,\ldots,i_q}(t+1) = d_{i_1,i_2,\ldots,i_q}(t) + \Delta d_{i_1,i_2,\ldots,i_q}(t)$$

where

$$(3.3) \quad \Delta d_{i_1,i_2,\ldots,i_q}(t) = -\epsilon \frac{\partial E}{\partial d_{i_1,i_2,\ldots,i_q}(t)}$$
$$= \epsilon \left(y_r - y_d\right) \prod_{j=1}^{q} N_{i_j}^{n_j}(x_j),$$

with the learning rate $0 < \epsilon \leq 1$.

In [9] we showed that the error function E is a convex function for a certain partition of input space. Therefore, the learning of control vertices converges rapidly.

3.2 Rapid Reinforcement Learning

A general unsupervised learning method for multiple-input–single-output system was presented in [10]. Assume that the Z-component of the translational part of one of the manipulators is to be controlled. To design this controller, the desired output for a given input vector is unknown. However, it can be assumed from Eqn. (2.1) that the difference $(y_r - y_d)$ is somehow proportional to $(F_{z,\mathrm{r}} - F_{z,\mathrm{d}})$, the difference between the real and desired force. If such a physical model is embedded in the learning process, the adaptation of control vertices, which is normally a reinforcement learning problem, becomes much simpler.

To initialise the control vertices, the expert's estimation values can be set if they are available. Otherwise, all control vertices are initialised with zero. In every control cycle[2], the output is added to the Z-component of the translational part of the corresponding transformation. By using the feedback information from the measured resulting forces, the control vertices can be improved to get a better result in the next control cycle. The modification of a control vertex is a slight variance from Eqn. (3.3):

(3.4)

$$\Delta d_{i_1,i_2,\ldots,i_n}(t-1) = \epsilon \left(F_{z,\mathrm{r(t)}} - F_{z,\mathrm{d}}\right) \cdot$$
$$\prod_{j=1}^{q} N_{i_j}^{n_j}(x_j(t-1)), \quad \text{with } 0 < \epsilon \leq 1.$$

Because $F_{z,\mathrm{r(t)}}$ is a result of the controller's behaviour in the last control cycle (if there is no delay), the vertices of the last instead of the current cycle are modified. Therefore, the input values of the last control cycle are used.

---

[2]The control cycle for both robots is 20 ms in our experiment.

4 Implementation Issues

4.1 Inputs/Outputs for Compliant Control

Assume that the two robots should carry a rigid object together along a specified trajectory. For such a task we adopt the strategy that both robots are controlled. We design four controllers for each robot: tool- X, Y and Z direction and the rotation around the *approach vector* of the tool coordinate. Since the rotations around tool X- and Y-axes play a secondary role in tasks we investigated, they are neglected to reduce computation burden in a time-critical control cycle.

After experimental factor analysis, the first three inputs of each controller are selected as the related force, torque and the distance between robot shoulder and wrist. The third input can be viewed as an estimation of the moment of inertia which results in different compliant parameters. These inputs are modelled with B-spline basis functions of the third order (Fig. 2) which enables the continuity of the second differentiation of each controller output and can be evaluated within a relatively short computation time. The knots near the zero areas are selected with a rather small distances so that the slight forces can be considered in a finer resolution.
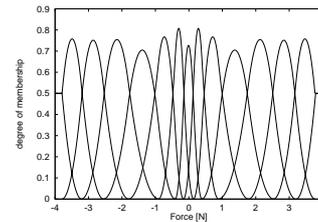

Figure 2: Partitioning the force input with B-spline basis functions of order three for the translational Z controller.

4.2 Anti-Drift Solution for Two Robots

For contact motions, both of the robots should be controlled because not only the internal forces between the manipulators are to be minimised but also the contact forces must be maintained.

Such a controller will have an unpleasing drift effect which is a result of the slightly different calibration. While sensor A detects no force, sensor B might measure a small value. The corresponding robot B will make a little position correction to reach the desired force, which exerts a small force on sensor A. If such an effect continues to exist, both robots may drift from the desired trajectory.

One solution is to limit the value of the integrated controller output. This means that the robots are allowed to

leave the desired trajectory, but only in a certain small range that is just as much as needed for the internal force minimisation. For this purpose, we introduce another feedback to the cost function for learning, i.e. the drift from the desired trajectory. We suggest the following learning function:

$$(4.1) \quad \Delta d_{i_1, i_2, \ldots, i_n}(t-1) = \epsilon \left(F_{z, \mathrm{r}(t)} - F_{z, \mathrm{d}} + \left(\gamma \cdot z(t)^3 \right) \right) \cdot$$
$$\prod_{j=1}^{q} N_{i_j}^{n_j}(x_j(t-1))$$
$$\text{with } 0 < \epsilon \leq 1.$$

If the manipulator is near the desired trajectory, $z(t)$ is very small and the linear part $(F_{z, \mathrm{r}(t)} - F_{z, \mathrm{d}})$ plays the major role in modifying the control vertices. If the manipulator drifts away from the desired trajectory, $z(t)$ increases and the exponential part $(\gamma \cdot z(t)^3)$ becomes more and more significant. γ can be used to adjust the effect of the exponential part for different drift allowances. The variables $z(t)$ as well as $x(t), y(t)$ are used as the fourth input of each controller respectively.

4.3 Repeatedly Practising

Generally, the learning process is performed in the following sequence:

1. Read the input values for the B-spline fuzzy controllers.

2. Calculate the controllers output and updating the $SENSOR$ transformation.

3. Store the input values $x_j(t)$ of the current step in a ring buffer.

4. Modify the control vertices as described in (4.1).

This sequence is repeated in every control cycle until a task is finished. The modified control vertices are used immediately in the next control cycle. The learning procedure for one complete task is called a *practice step*, which should be repeated several times so that for the same task the control vertices are adjusted optimally. Our experiments show that the learning rate ϵ directly influences the convergence speed. If ϵ is selected too small, the learning process needs a large amount of time. If ϵ is too large, the learning procedure can cause oscillation. Our experience of selecting ϵ is that starting with an initial value, e.g. 0.01, ϵ is divided by two or more after a few *practice steps*. A minimum value of ϵ has to be set, e.g. 0.001 in our experiment.

5 Experimental Results

In our early work we tested the on-line learning approach on a one-arm screwing problem, [11]. In the following, we briefly present the results of several typical experiments: a) Two-arm transport of a rigid object; b) Two-arm grinding; c) Two-arm peg-in-hole.

5.1 Two-Arm Motion in Closed Kinematic Chain

The first task is the transport of a wooden ledge with the two robots (Fig. 1). Fig. 3 shows a clear comparison of controlled and uncontrolled motion.


(a)   (b)
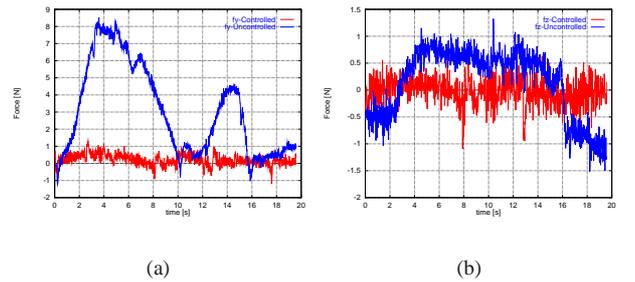
Figure 3: A comparison of the resulting force (a) in Y-direction (*approach vector*) and (b) in Z-direction.

Fig. 4(a) shows that the mean-squared (MS) force error converges. Fig. 4(b) visualises the control surface, the relation between the correction motion in X-direction and two inputs while keeping the other two inputs constant.


(a)   (b)
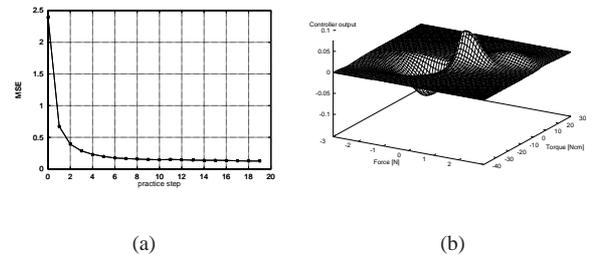
Figure 4: Visualisation of the learning results. (a) Average MS force error of one robot with respect to the *practice step*; (b) The control surface of the compliant controller in X-direction.

5.2 Grinding Motion

The robots are also applied to some contact motions in which the ledge is moved along a metal bar while main-

taining a certain contact force (Fig. 5(a) and (b)). Fig. 5(c) presents the results after several *practice steps*. The force profiles depicted in this figure can be interpreted as four phases. The first one takes about 6 seconds and was logged while the ledge is approaching the bar. The peak of about -23 N indicates that the ledge establishes contact with the bar. It is caused by the control cycle rate of 20 ms and another 20 ms delay. It is noticeable that there is no over-shooting reaction after the peak. During the next 7 to 8 seconds, the ledge is moved along the bar with a given contact force of -5 N. When the ledge is pulled off the bar and moved back to the starting position, the force in Z-direction resumes the value of zero. The small but constant force under contact in X-direction originates from the friction caused by the motion.
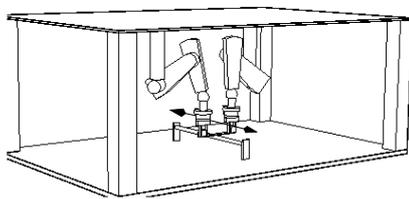

(a)
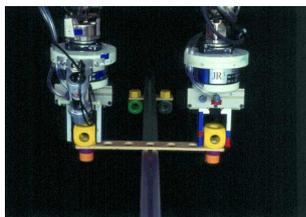
(b)
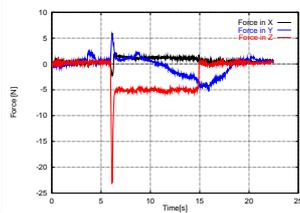
(c)

Figure 5: Moving a wooden ledge along a metal bar. During the contact situation the desired force in Z-direction is -5 N. (a) The global view; (b) The local view; (c) The resulting forces.

5.3 Searching a Hole

The ledge of the last experiments has a hole. Our next experiment is an operation of putting the hole of the ledge onto a bolt (Fig. 6(a)). The operation was splitted into three phases that are characterised by the force profiles in Fig. 6(b). The first phase $[0, 7s]$ is the approach motion that finished with the contact between ledge and bolt, which can

be recognised by the peak. It is followed by a spiral search phase $[7s, 14s]$ to find the bolt. When the desired contact force (-3.5 N in our experiment) shrinks to zero, it is assumed that the hole has been found and the ledge can be sticked on the bolt (the third phase up 17 sec.). Here the termination condition is very critical since the contact between ledge and bolt should not be lost.
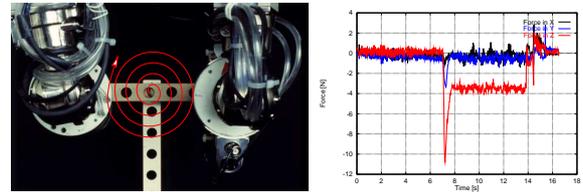

(a)

(b)

Figure 6: Searching for a hole with two arms. (a) A local top view of the experiment system; (b) The resulting forces in Z-direction.

Fig. 7 demonstrates another successful experiment: the two arms carry the ledge with a fastened block, insert it onto a tight slot, then push it along the slot.
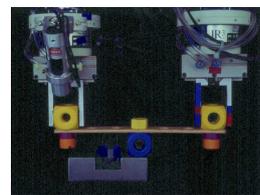

(a)
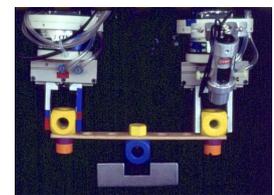
(b)
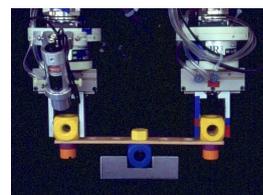
(c)
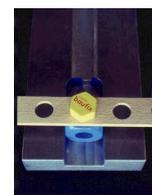
(d)

Figure 7: Insertion and push with tight tolerance. (a) Establishing contact; (b) Insert into a tight slot; (c) and (d) Push along the slot.

6 Discussion

[Learning Speed.] In the compliant motion, the error function is approximately proportional to the change of the controller output. Thanks to the local properties of the B-spline model, the learning process converges very rapidly. If such a condition is not fulfilled, test motions must be introduced. Generally speaking, the number of the test motions is 3^s, where s is the number of outputs, since for each output, three states $(+, 0, -)$ should be tested. In fact, we applied such an approach to the same learning problem, but the learning time is significantly increased. As a conclusion, it can be summarised that if parts of the physical models are available, embedding it in the evaluation (fitness) function can accelerate the learning.

[Generalisation.] The local support property of the B-spline model brings rapid convergence of the learning, but also the disadvantage of weak generalisation ability. That demands a sufficient training data set in case of supervised learning. Fortunately in our on-line learning, the training data are generated in every control cycle. After repeated practices, almost all the typical "situations" constructed by the input forces/torques are experienced by the controller, and the appropriate control action can be learned. In principle, the robots learn lifelong. Obviously, the generalisation of one control skill to another task may be possible if symbolic rules are extracted from the B-spline fuzzy model. That will be one of our major future work.

[Direct Mapping.] This training task can also be realised by an adaptive PID controller to deal with the non-linearity of the stiffness parameters. However, our approach provides a model which maps the inputs (forces and torques) directly onto the outputs (the compensation motions). Such a model can be interpreted with linguistic rules which better reflects human intuitive knowledge.

[Adaptive Model.] Another advantage of this approach it that with the help of the robot's self-practise in the real-environment where it will experience, a good controller can always be found for a specified task. Uncertainties in the robots and the manipulated object can be taken into account in the input–output B-spline model.

Acknowledgement

References

[1] O. M. Al-Jarrah and Y. F. Zheng. Intelligent compliant motion control. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2610–2615, 1996.

[2] J. S. Albus. A new approach to manipulator control: The Cerebellar Model Articulation Contorller (CMAC). *Transactions of ASME, Journal of Dynamic Systems Measurement and Control*, 97:220–227, 1975.

[3] C. G. Atkeson and J. C. Santameria. A comparision of direct and model-based reingorcement learning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3557–3564, 1997.

[4] M. Kaiser and R. Dillmann. Building elementary robot skills form human demonstration. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2700–2705, 1996.

[5] K. Kiguchi and T. Fukuda. Fuzzy neural friction compensation method of robot manipulation during position/force control. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 372–377, 1996.

[6] F. Lange and G. Hirzinger. Learning force control with position controlled robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2282–2288, 1996.

[7] M. H. Raibert and J. Craig. Hybrid position/force control of manipulators. *Journal of Dynamic Systems*, 102:126–133, 1981.

[8] B. R. Shetty and M. H. Ang Jr. Active compliance control of a puma 560 robot. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3720–3725, 1996.

[9] J. Zhang and A. Knoll. Constructing fuzzy controllers with B-spline models - principles and applications. *International Journal of Intelligent Systems*, 13(2/3):257–285, FEB./MAR. 1998.

[10] J. Zhang, K. V. Le, and A. Knoll. Unsupervised learning of control spaces based on B-spline models. In *Proceedings of IEEE International Conference on Fuzzy Systems*, 1997.

[11] J. Zhang, Y. v. Collani, and A. Knoll. On-line learning of B-spline fuzzy controller to acquire sensor-based assembly skills. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 1997.