ELSEVIER

Information Sciences 145 (2002) 89–111

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Knowledge acquisition and learning in unstructured robotic assembly environments

I. Lopez-Juarez [a,*], M. Howarth [b]

[a] *CIATEQ A.C., Centro de Tecnologia Avanzada, Manantiales 23A, Fracc. Ind. B.Q., CP 76246 El Marques, Queretaro, Mexico*

[b] *Sheffield Hallam University, School of Engineering, Sheffield s1 1WB, England, UK*

Received 4 July 2001; received in revised form 8 October 2001; accepted 28 November 2001

Abstract

Mechanical assembly by robots has traditionally depended on simple sensing systems and the robot manufacturers programming language. However, this restricts the use of robots in complex manufacturing operations. An alternative to robot programming is the creation of self-adaptive robots based on the adaptive resonance theory (ART) artificial neural network (ANN).

The research presented in this paper shows how robots can operate autonomously in unstructured environments. This is achieved by providing the robot with a primitive knowledge base (PKB) of the environment. This knowledge is gradually enhanced on-line based on the contact force information acquired during operations.

The robot resembles a *blindfold* person performing the same task since no information is provided about the localisation of the fixed assembly component. The design of a novel neural network controller (NNC) based on the Fuzzy ARTMAP network and its implementation results on an industrial robot are presented, which validate the approach.

Keywords: Robotic assembly; Neural network controller; Force control; On-line learning; Adaptive resonance theory; Skill acquisition; Knowledge discovery

* Corresponding author. Fax: +52-442-216-9963.
E-mail addresses: ilopez@ciateq.mx (I. Lopez-Juarez), m.howarth@shu.ac.uk (M. Howarth).

0020-0255/02/$ - see front matter
PII: S0020-0255(02)00225-6

1. Introduction

Industrial robots are useful and reliable machines for assembly. The success of the operation is based on the accuracy of the robot itself and the precise knowledge of the environment, i.e. information about the geometry of the assembly parts and their localisation in the workspace. Combining these elements and using the manufacturers programming language efficient programs can be written. When parameters change, for instance part geometry, the robot program has to be amended to take into account new conditions. The adaptation to these new conditions is explicitly given by the programmer. Industrial robots are currently being programmed using this technique, hence, robots are still unable to be self-adaptive to varying conditions and this is possibly one of the major drawbacks that has limited their extensive use in manufacturing.

Techniques are sought to provide self-adaptation for robots. Robot manipulators operate in real world situations with a high degree of uncertainty and require sensing systems to compensate from potential errors during operations. Uncertainties come from a wide variety of sources such as robot positioning errors, gear backlash, arm deflection, ageing of mechanisms and disturbances. Controlling all the above aspects would certainly be a very difficult task, therefore a simpler approach based on force control is preferred. By using force control the overall effect of the contact force between the environment (assembly parts) and the manipulator are considered as a whole.

1.1. Force control, connectionist models and prior work

Force control can be roughly divided in Model-based and Connectionist-based approaches. The model-based approach takes as much information of the system and environment as possible. This information includes localisation of the parts, geometry of the parts, materials, friction, etc. The connectionist-based approach is based on connectionist models and its robustness relies on the information given during the training stage that implicitly considers all of the above parameters. Model-based methods do not offer a complete solution due to the uncertainties associated during assembly as mentioned earlier. On the other hand, connectionist-based techniques have proved to work reliably when uncertainty is involved due to their generalisation property.

The use of connectionist models in robot control to solve the problem under uncertainty has been demonstrated in a number of publications, either in simulations [1–3], or being implemented on real robots [4–6]. In these methods, reinforcement learning (RL), unsupervised and supervised type networks have been used.

The reinforcement algorithm implemented by V. Gullapalli demonstrated to be able to learn circular and square peg insertions. The network showed a good

performance after 150 trials with insertion times lower than 100 time steps [7]. Although the learning capability demonstrated during experiments improved over time the network is unable to generalise over different geometries. Insertion are reported with both circular and square geometries, however, when inserting the square peg, its rotation around the vertical axis was restricted otherwise the insertion would not have been possible. M. Howarth followed a similar approach, using Backpropagation in combination with reinforcement learning. During simulation it was demonstrated that 300 learning cycles were needed to achieve a minimum error level with his best network topology during circular insertions [6]. A cycle meant to be an actual motion that diminished the forces acting on the peg. For the square peg, the number of cycles increased dramatically to 3750 cycles. These figures are important, especially when fast learning is desired during assembly. On the other hand, E. Cervera using SOM networks and a Zebra robot (same used by Gullapalli) developed similar insertions as the experiments developed by Gullapalli. Cervera in comparison with Gullapalli improved the autonomy of the system by obviating the knowledge of the part location and used only relative motions. However, the trade-off with this approach was the increment of the number of trials to achieve the insertion [4], the best insertions were achieved after 1000 trials. During Cervera's experiments the network considered 75 contact states and only 8 out of 12 possible motion directions were allowed. For square peg insertions, there were needed 4000 trials to reach 66% success of insertion and that did not improve any further. According to Cervera's statement, "We suspect that the architecture is suitable, but the system lacks the necessary information for solving the task". The situation clearly recognises the necessity to embed new information in the control system as it is needed, which is likely to be achieved with an architecture such as ART.

1.2. Original work

The objective of the research presented in this paper is to create self-adapting robots able to perform mechanical assembly with a minimum of instructions and information. This is achieved by resembling the behaviour of a *blindfold* person developing the operation. For a blindfold person the only available information is the contact force while attempting to insert the workpiece since the localisation of the parts is unknown. In addition, the other basic information a person would possess in this situation is the information stored in his/her brain on how to react to *primitive* constraint forces. In other words, the intrinsic attitude of a person to react to a force that impedes the insertion and by moving to the opposite direction.

In a similar way, the robot is provided only with contact force information and a primitive knowledge base (PKB), which is an initial contact force–action mapping that bias its initial reactions to constrained forces. No

information is given about the localisation of the parts. The arm increases its knowledge on-line based on the success of the predicted motion. The robot actually increases and enhances its knowledge as the operation progresses. The time that the robot takes to complete a similar operation is reduced and also mistakes made earlier do not recur, which demonstrates the new expertise of the robot.

The design of the novel neural network controller (NNC) is founded on the strength of ART networks to learn incrementally. The new information is acquired as the operation develops without affecting the knowledge that was previously learnt. The Fuzzy ARTMAP algorithm is used and the NNC training made on-line. The number of contact force patterns that the NNC can accommodate in its knowledge is limited only to memory storage. The switching mechanism of the NNC is regulated by the development of the operation. New knowledge information is only accepted in the knowledge base when it has strongly contributed towards the success of the assembly. The resulting enhanced knowledge base (EKB) at the end of the assembly can be used for similar operations. Results on an industrial robot demonstrates that the robot's skill improves effectively and the insertion times and the errors diminish over time. Furthermore this is, to the best knowledge of the authors, the first time the Fuzzy ARTMAP network has been applied to an industrial robot manipulator.

2. Host–slave architecture

The hardware architecture is formed by the robot, slave computer, robot controller, supervisory host computer and the Force/Torque (F/T) sensor as illustrated in Fig. 1.

Main units of the robot system are the controller and the robot arm itself. Power and data are transmitted between the two units through two interconnecting cables. The controller houses the components that control and power the robot arm. The supervisory host computer communicates with the controller via two serial ports: the "supervisory" mode in which high-level commands are sent to the controller and the ALTER mode through the slave computer for incremental fine motions. The ALTER mode information handling is very strict since once the data interchange has been started, the supply of motion requests by the computer must continue otherwise an error message will appear and the communication will be terminated.

The slave computer is a 486-based computer running under DOS. This computer was used to release the host computer from the low-level communication with the robot controller. With this architecture, the host only needs to request the motion to the slave computer and start a new process or monitor

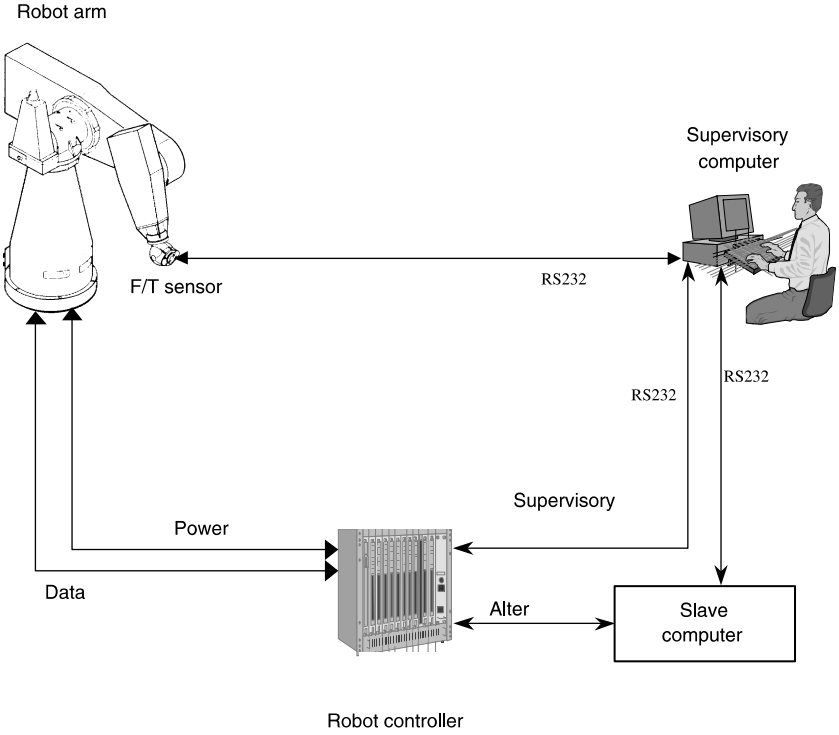


Fig. 1. Host–slave architecture.

the F/T signal while the arm's position is being modified. Both processes, force monitoring and arm positioning, can be made in parallel and in real-time.

3. The problem and nature of forces

Fig. 2(a) shows a typical peg in hole insertion, which is the most common operation in assembly and a canonical operation for performance assessment. The force traces occurring during this type of operation are given in Fig. 2(b).

This type of signal is normally acquired by using a F/T sensor mounted in the robot's wrist. The sensor provides the required input information to the NNC. The signal patterns contain information regarding the force and torque "felt" at the robot wrist. With this information is possible to determine how much force is being applied to the end-effector or gripper [1] and how these

[1] The gripper is a mechanical device to grasp and hold the assembly part, which normally consist of two or more fingers.

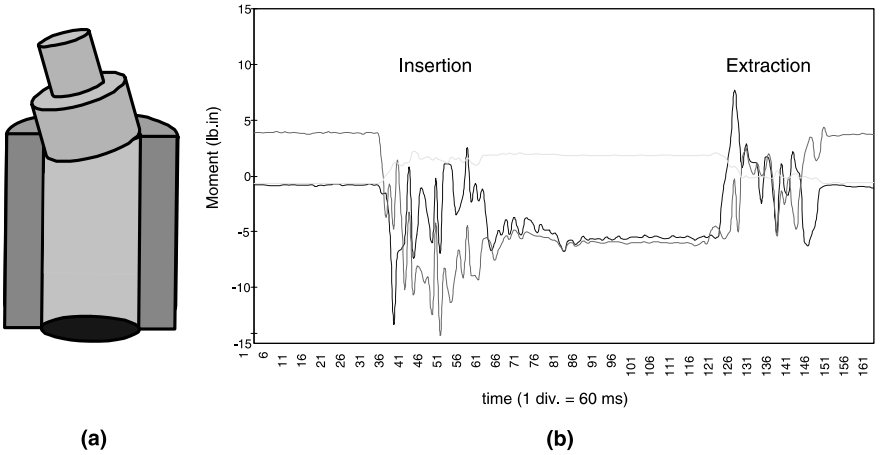


Fig. 2. (a) Peg-in-hole insertion and (b) contact froce.

forces affect the orientation of the peg by means of the moment value and sign. Although the graph only shows the moment information, the information available to the NNC is a F/T vector containing six elements, i.e. *fx*, *fy*, *fz*, *mx*, *my* and *mz*.

4. Adaptive resonance theory

The adaptive resonance theory (ART) [8] was developed by Stephen Grossberg and Gail Carpenter at Boston University to solve the called *stability–plasticity* dilemma. That is, the system is sensitive to novelty capable of distinguishing between familiar and unfamiliar events (plastic) and still remain stable. Different model variations have been developed to date based on the original ART-1 algorithm for binary input patterns [9], ART 2-A for analogue and binary input patterns [10], and ART 3 based on chemical transmitters. Supervised learning is possible through ARTMAP [11] that uses two ART modules and its variants, Fuzzy ARTMAP [12], Gaussian ARTMAP [13] and ART-EMAP [14] even though there are many other variants adapted for specific applications [15]. In the following section a brief explanation of the mechanics of ART-1 and Fuzzy ARTMAP is given followed by the description of the NNC.

4.1. ART-1

The ART-1 architecture consists of two parts: attentional subsystem and orienting subsystem as illustrated in Fig. 3.

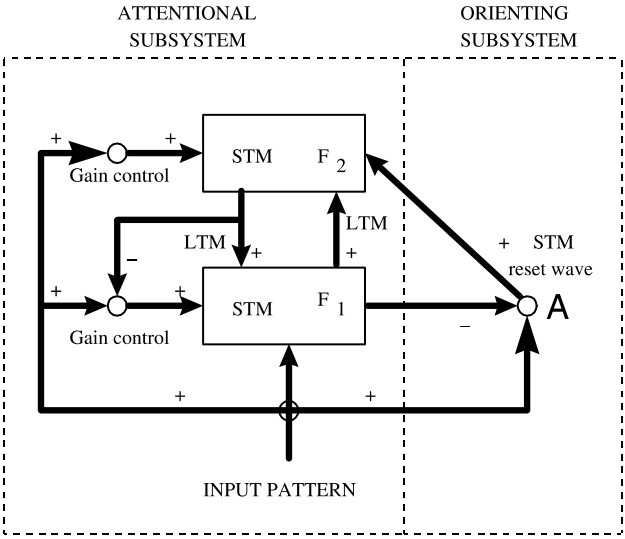


Fig. 3. ART architecture.

The attentional subsystem is made up of two layers of nodes F_1 and F_2. In an ART network, information in the form of processing-element output reverberates back and forth between layers. If a stable resonance takes place learning or adaptation can occur. On the other hand, the orienting subsystem is in charge of resetting the attentional subsystem when an unfamiliar event occurs.

A *resonant state* can be attained in one of two ways. If the network has learned previously to recognise an input vector, then a resonant state will be achieved quickly when that input vector is presented. During resonance, the adaptation process will reinforce the memory of the stored pattern. If the input vector is not immediately recognised, the network will rapidly search through its stored patterns looking for a match. If no match is found, the network will enter a resonant state whereupon the new pattern will be stored for the first time. Thus, the network responds quickly to previously learned data, yet remains able to learn when novel data is presented, hence solving the so-called *stability–plasticity* dilemma. The activity of a node in the F_1 or F_2 layer is called *short-term* memory (STM) whereas the adaptive weights are called *long-term* memory (LTM). Gain controls handle the discrete presentation of the input signals. A vigilance parameter ρ measures how much mismatch is tolerated between the input data and the stored patterns, which can be used to control the category coarseness control of the classifier.

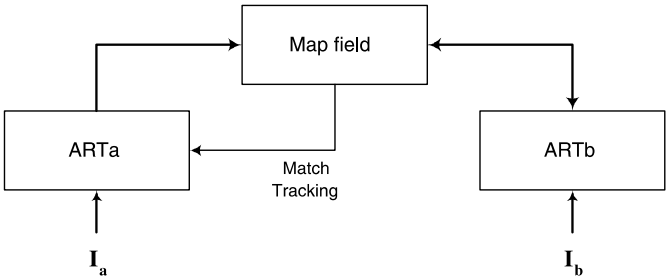


Fig. 4. Fuzzy ARTMAP architecture.

4.2. Fuzzy ARTMAP

In the Fuzzy ARTMAP (FAM) network there are two modules ARTa and ARTb and an inter-ART module "*Map field*" that controls the learning of an associative map from ARTa recognition categories to ARTb categories. This is illustrated in Fig. 4.

The map field module also controls the match tracking of ARTa vigilance parameter. A mismatch between Map field and ARTa category activated by input Ia and ARTb category activated by input Ib increases ARTa vigilance by the minimum amount needed for the system to search for, and if necessary, learn a new ARTa category whose prediction matches the ARTb category. The search initiated by the inter-ART reset can shift attention to a novel cluster of features that can be incorporated through learning into a new ARTa recognition category, which can then be linked to a new ART prediction via associative learning at the Map field.

5. Neural network controller

The functional structure of the assembly system is illustrated in Fig. 5. The FAM is the heart of the NNC. The controller includes three additional modules. The knowledge base that stores the initial information related to the geometry of the assembling parts. This information is used only during the first assembly operation, later this is enhanced by patterns that favour the assembly and whose inclusion is regulated by the pattern–motion selection module. This module keeps track of the F/T patterns and verifies whether the action is good enough to allow the FAM network to be retrained. If this is the case, the switch SW is closed and the corresponding pattern–action provided to the FAM for on-line retraining.

Future predictions will be based on this newly trained FAM network. The Automated Motion module basically is in charge of sending the incremental

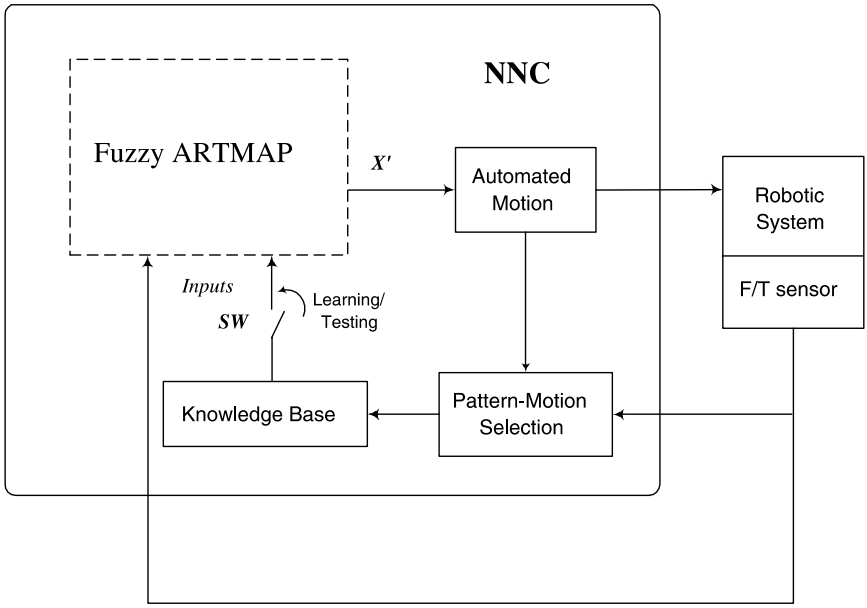


Fig. 5. System structure.

motion request to the robot controller and handling the communication between the slave computer and the FAM network output. External components to the NNC are the robot controller, the manipulator itself and the F/T sensor that provides the pattern information. The programs for the NNC were created using Visual C++ 5.0 and implemented in a 100 MHz Pentium PC.

5.1. Initial training and PKB formation

The formation of the PKB basically consists of showing the robot how to react to individual components of the F/T vector. The influence of each vector component requires a motion opposite to the direction of the applied force to diminish its effect. The procedure is illustrated in Fig. 6. For simplicity, only the lower arm of the manipulator has been shown.

Every motion of this type is referred to as a primitive motion (PM) and the idea is to teach the robot *where* to move when single F/T components, i.e. fx, fy, fz, mx, my, or mz are applied to the workpiece. Fig. 6(a)–(c) illustrate the PM needed to diminish the corresponding constraint force in the X, Y or Z axis. Note that in Fig. 6(c), when the arm is in free-space the PM will be in $-Z$ direction since this was the condition (minimum constraint forces) to proceed downwards during this assembly operation. Generally speaking, the training consists of moving the workpiece against a rigid object to produce the

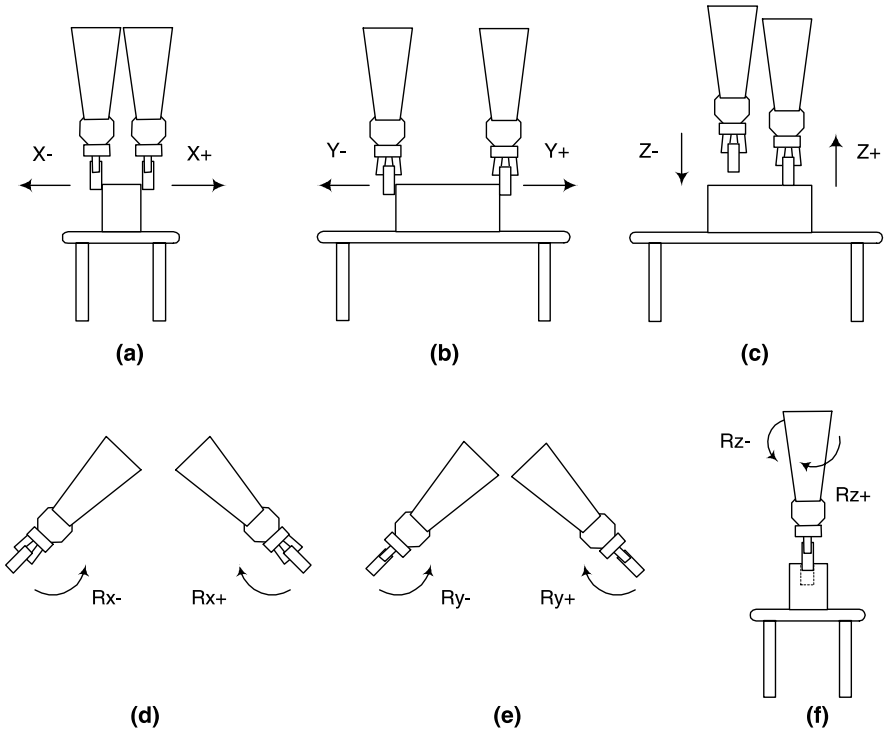


Fig. 6. Training procedure.

appropriate component force and consequently determine the corresponding PM. The magnitude of the constraint forces applied to the workpiece is bounded by the force limit selected in the NNC program. The PM corresponding to the rotation in X and Y axis (Fig. 6(d) and (e)) were assigned after rotating the arm in free-space at an angle so that a single *mx* or *my* component was produced. The PM, Rz, was given to the network using a square peg into a square hole producing a moment around the Z axis as illustrated in Fig. 6(f).

At this time and while the arm is in constraint motion, the F/T pattern will be acquired in the knowledge base and will be associated with the selected motion. The storage of the F/T vector and the PM will form the PKB that is required to start the assembly for the very first time. Once the first insertion has been completed, the robot may possibly have increased its knowledge. If so, the PKB is enhanced and an enhanced knowledge base (EKB) version will be used during the following insertion.

The PKB used during our experiments is shown in Fig. 7. The F/T data from the sensor was scaled to the range $[0, 1]$, where the extreme values 0 and 1 corresponded to a force of -15 and $+15$ lb respectively. Negative values were

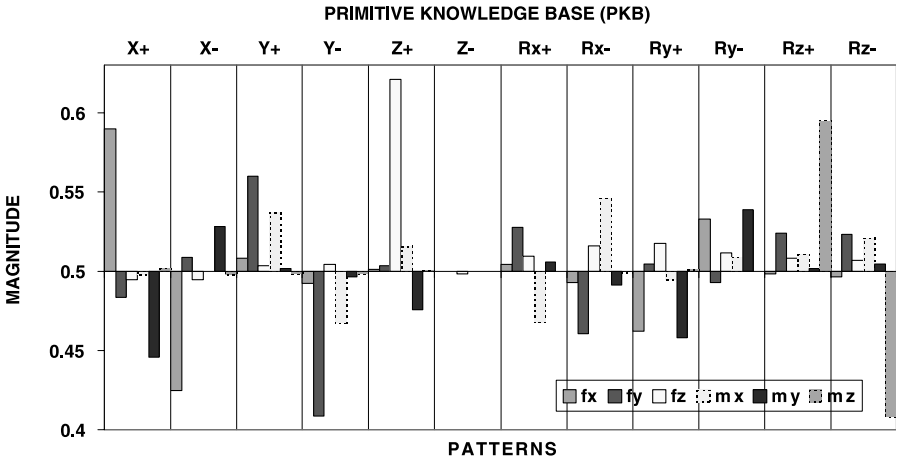


Fig. 7. PKB.

assigned to the interval $[0, 0.5)$ and positive values were assigned to the interval $(0.5, 1]$. It should be noted that the origin in the graph is set to 0.5, where positive and negative values are represented in the upper and lower halves of the graph respectively. Every column corresponded to an input vector to the network. The corresponding assigned output vector is shown at the top of the graph for each pattern.

6. Knowledge discovery

The first stage of the NNC was initially implemented using the ART-1 network in fast learning mode [9]. Results showed the incremental learning capability of the network by quickly acquiring the contact force patterns. However, since the ART-1 network only takes binary values an encoding was necessary. Information about initial results and details on the encoding can be founded in [16]. With the use of supervised Fuzzy ARTMAP algorithm the NNC can now take analogue values and provide the required prediction capability.

6.1. Pattern–motion selection and knowledge enhancement

There are potential problems associated with the learning mechanism which are solved by the pattern–motion selection module. The robot should continue moving in the insertion direction if, and only if, a minimum force value has been reached. This situation should trigger the learning mechanism in order to allow the acquisition and learning of the pattern–action pair that produced such a situation. In the event of continual learning after having reached this

point, the performance of the NNC might decay. This situation is similar to what is known as overtraining, overfitting or overlearning in ANNs. At this point the learning should be stopped because if the robot learns other patterns under the above circumstances, eventually the minimum force value will be different leading to wrong motions. The same applies to the condition when the end-effector meets a force higher than the force limit. There should not be any further learning during this situation since learning a higher force would probably damage the sensor.

The above situations can be resumed in three fundamental questions:

1. What is a good motion?
2. How to recover from errors?
3. Which motions should or should not be learned?

Having an assembly system which is solely guided by contact force states, the criterion to decide whether the motion was good enough to be learnt is based on the following expression:

$$F_{\text{after}} < 0.1 * F_{\text{before}}. \tag{1}$$

F_{after} and F_{before} are computed using the following equation:

$$F = \sqrt{fx^2 + fy^2 + fz^2 + mx^2 + my^2 + mz^2}. \tag{2}$$

Expression (1) means that if the total force after the incremental motion is significantly reduced then that pattern–action will be considered good to be included in the knowledge base. Experiments showed that if this threshold value was set higher (i.e. $\geqslant 0.3 * F_{\text{before}}$) the network became very sensitive and showed overtraining behaviour.

Forces that are higher than the value given by $0.1 * F_{\text{before}}$ and lower than the F_{limit} are still good values. However, the corresponding pattern–action pair will only be used during network recall. This situation is illustrated in Fig. 8 that shows three possible situations: learning, recall and error recovery.

The third area is a situation where $F \geqslant F_{\text{limit}}$. In this situation the user is alerted and asked to reposition the arm.

There will be ambiguous situations in which learning should not be permitted. This applies to patterns in the insertion direction (usually Z direction). Consider downward movements in the $Z-$ direction. At the time the peg makes

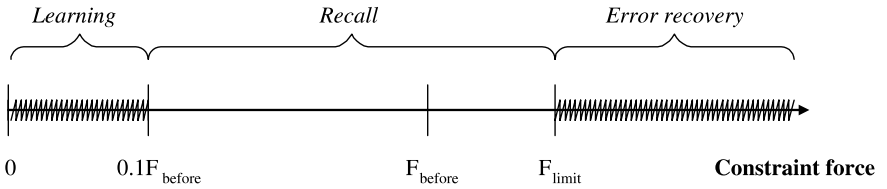


Fig. 8. Learning, recall and error recovery.

contact with the female block, there may well be a motion prediction in the $Z+$ direction. This recover action will certainly diminish the contact forces and will satisfy the condition given by the expression (1) in order to learn the force–action pair. However, this situation is redundant since it was given when the PKB was formed and it is likely that it will corrupt the PKB. Similarly, learning should not be allowed when the arm is in free-space. In this situation, F_{after} and F_{before} will be very similar and again learning another pattern in the $Z-$ direction will be redundant. Both situations were tested experimentally by the author and revealed that an unstable situation may appear if further learning is allowed in the insertion direction.

After the pattern–action has satisfied expression (1) and the prediction direction is not in the Z direction, the pattern is allowed to be included in the new "expertise" of the robot, the EKB. Patterns that do not satisfy expression (1) and whose values are lower than the F_{limit} will only be used to recall previous knowledge. The knowledge refinement process will continue in the NNC until the end-condition is satisfied.

7. Results

Several tests were carried out to assess the performance of the NNC using aluminium pegs with different cross-sectional geometry: circular, square and radiused-square. The diameter of the circular peg was 2.5 cm and the side of the square peg was also 2.5 cm. The dimensions of the non-symmetric part, termed radiused-square because it was an square peg with one corner rounded to a radius of 1.25 cm. Clearances between pegs and mating pairs were 0.1 mm. The assembly was ended when 3/4 of the body of the peg were inside the hole. This represented 50 motion steps in the $-Z$ assembly direction. A typical assembly operation is shown in Fig. 9.

The Fuzzy ARTMAP network parameters during experiments were set for fast learning (learning rate = 1). The base vigilance $\overline{\rho_a}$ had a low value since it has to be incremented during internal operations. ρ_{map} and ρ_b were set much higher to make the network more selective creating as many clusters as possible.

The vigilance parameters used for the experiments reported in this article are as follows:

$$\overline{\rho_a} = 0.2 \text{ (base vigilance)},$$

$$\rho_{\text{map}} = 0.7,$$

$$\rho_b = 0.9.$$

Typical results on three different geometries are summarised in Table 1.

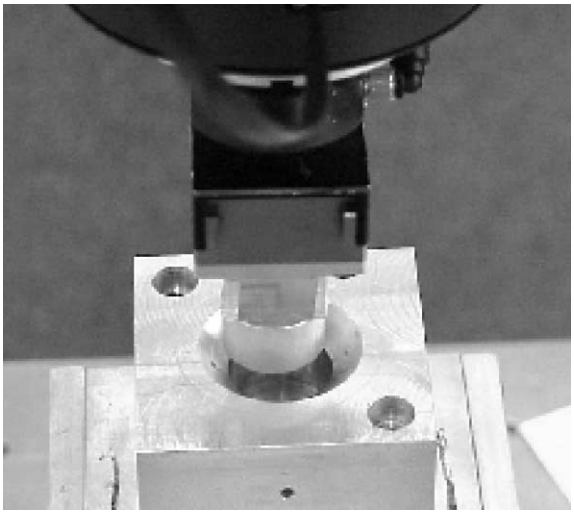


Fig. 9. Typical assembly operation.

At the start of the operation positional offsets were given as indicated in the second column. During the first insertion, learning enabled (ON status), the network learned three new patterns and this operation required 58 incremental motions and only eight alignment motions. The learned patterns were $X+$, $X-$ and $Y-$ as indicated in the comments field. The processing time for the whole insertion was 5.17 s. This time considered only the processing of the patterns, training and testing of the network. The actual insertion time was longer since a delay of 1 s was added to avoid the transient stage after every incremental motion. Considering this delay, the first assembly was accomplished in approximately 63.17 s. Subsequent assemblies were carried out and the number of learned patterns decreased to only one. The processing time showed only small fluctuations for insertions using the same offset. Table 1 also shows the "expertise" acquired by the robot during the operations. After nine insertions the NNC had learnt 12 additional patterns. This implied that these patterns were good enough to be learned. This EKB reinforced the prediction capability of the network since the new patterns were actually generated by the particular geometry of the parts, i.e. circular. The type of learned patterns at every insertion is indicated in the comments field. With a larger offset (insertions 10–14), the NNC learned only one additional pattern indicating that the network had already acquired the necessary knowledge about the chamfer and used this information effectively. As the starting point was further from the end-condition, the time to complete the insertion was proportionally longer. This is reflected in both the number of alignment motions and the total number of actions. Another interesting result is that the NNC also predicted rotation

Table 1
Insertion results

Insertion	Offset $(\delta x, \delta y, \delta \mathrm{Rz})$ (mm, mm, °)	Learning	New patterns	Alignment motions	Total Motions	Process. time (s)	Comments
Circular chamfered peg insertion							
1	$(-0.8, -0.4, 0.0)$	ON	3	8	58	5.17	$X+, X-, Y-$
2	$(-0.8, -0.4, 0.0)$	ON	2	8	58	5.23	$X+, Y-$
3	$(-0.8, -0.4, 0.0)$	ON	0	4	54	4.70	
4	$(-0.8, -0.4, 0.0)$	ON	1	4	54	5.00	$X+$
5	$(-0.8, -0.4, 0.0)$	ON	1	6	56	4.94	$X+$
6	$(-0.8, -0.4, 0.0)$	ON	2	6	56	5.18	$X+, Y-$
7	$(-0.8, -0.4, 0.0)$	ON	1	6	58	4.93	$X+$
8	$(-0.8, -0.4, 0.0)$	ON	1	6	56	5.09	$X+$
9	$(-0.8, -0.4, 0.0)$	ON	1	6	56	4.84	$X+$
10	$(-2.5, -2.5, 0.0)$	ON	1	14	64	5.47	$Y+$ (Incl. $Rx-$)
11	$(-2.5, -2.5, 0.0)$	ON	0	13	63	5.44	
12	$(-2.5, -2.5, 0.0)$	ON	0	16	66	5.60	
13	$(-2.5, -2.5, 0.0)$	ON	0	14	64	5.42	
14	$(-2.5, -2.5, 0.0)$	ON	0	13	63	5.49	
15	$(-2.5, -2.5, 0.0)$	OFF	0	25	85	7.35	$Z+(14)$ $Ry-(3)$
16	$(-2.5, -2.5, 0.0)$	OFF	0	24	84	7.27	$Z+(15)$ $Ry-(3)$
Square chamfered peg insertion							
17	$(-2.5, -2.5, 0.0)$	ON	1	39	89	7.76	$X+$
18	$(-2.5, -2.5, 0.0)$	ON	12	41	91	8.17	$Y+, Y-, X+, X-$
19	$(-2.5, -2.5, 0.0)$	ON	6	15	65	5.66	$X+, Y+, Y-$

Table 1 (*continued*)

Circular chamfered peg insertion							
Insertion	Offset $(\delta x, \delta y, \delta \mathrm{Rz})$ $(\mathrm{mm}, \mathrm{mm}, ^\circ)$	Learning	New patterns	Alignment motions	Total Motions	Process. time (s)	Comments
Radiused-square chamfered peg insertion							
20	$(2.44, 0.19, 0)$	ON	3	14	64	5.56	$X-, Y-$
21	$(2.44, 0.19, 0)$	ON	5	17	67	5.95	$X-, Y-, X+$
22	$(-1.56, 2.38, 0)$	ON	2	22	72	6.17	$Y+$

about the X axis during insertion 10, which indicates that information from the original PKB was still used if appropriate, as occurred in this situation. A further test was undertaken during insertions 15–16 which will be discussed later in Section 7.1. During square and radiused-square peg insertions the offsets were given as indicated using the same PKB. In all cases, the NNC performance was satisfactory.

7.1. Expertise test

A further test was conducted during insertions 15–16 using the PKB and the incremental learning capability inhibited (OFF status). For comparison purposes, the graphs corresponding to insertions 14 and 15 using the same offset are shown in Figs. 10 and 11 respectively. In both figures, the upper graph represents the force traces whereas the motion directions commanded by the NNC are given in the lower graph. In the motion direction graph, the horizontal axis corresponds with the $Z-$ direction. Bars above the horizontal axis represent linear alignments and below the horizontal axis represent angular alignments. Despite that the offset was the same, the number of alignment motions and insertion time were higher. With the learning inhibited, the robot was not allowed to learn contact states within the chamfer hence the NNC generated motions based only on its initial PKB. This resulted in motions that produced an excessive fz. As a result, the NNC predicted a series of compensatory movements in $Z+$ and $Ry-$ to recover from these situations. The robot was ultimately able to insert the workpiece, however the performance was poorer in terms of alignment and consequently speed. It can clearly be observed that the same operation with the same offset can be achieved more efficiently and faster if the robot uses the EKB. In other words, the robot shows its *dexterity* when it is allowed to use its expertise.

8. Discussions

8.1. Density of data and knowledge acquisition

The capability of generalisation and knowledge acquisition of the NNC has been demonstrated. Patterns that reduce significantly the contact forces during manipulations were acquired into the knowledge base and learnt. A representative learning example was shown in Section 7 with the circular chamfered insertion. In this example, the network was initially trained with the PKB containing the 12 possible patterns associated with the robot's 6 DOF. This information biased the initial learning by creating 12 categories to allocate every possible motion direction. From these results, it was verified that subsequent patterns corresponding to contact states within the

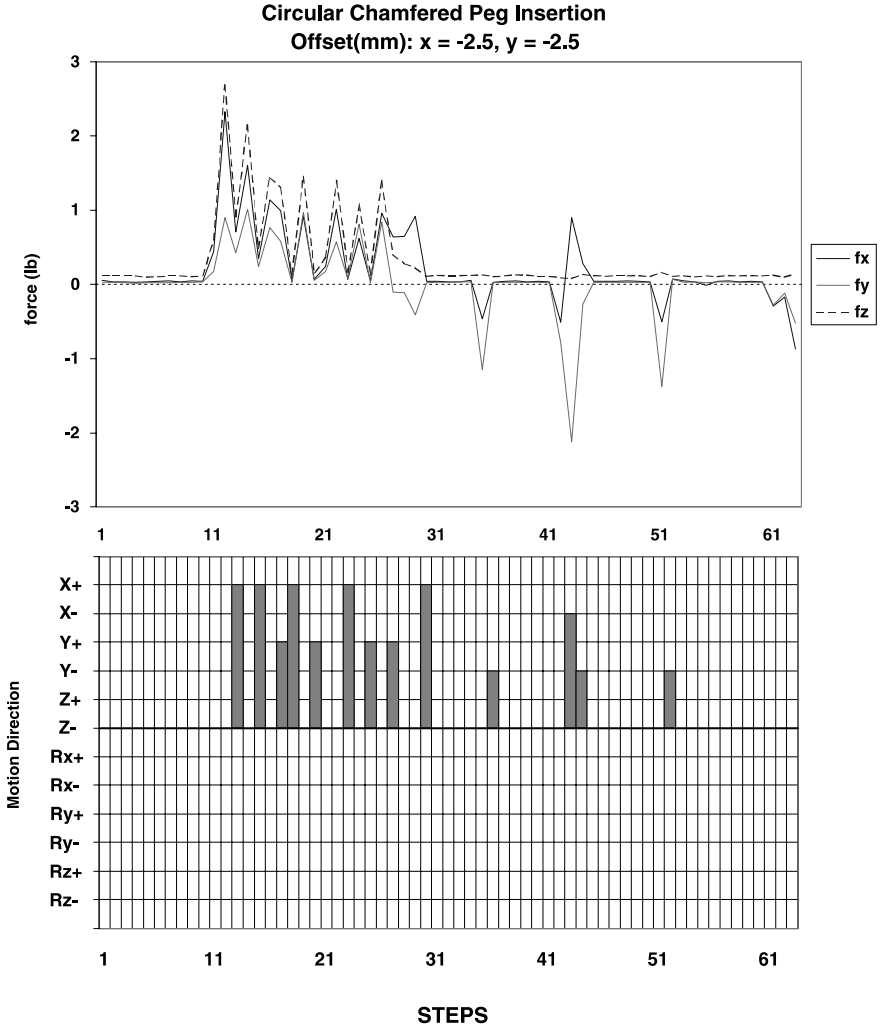


Fig. 10. Insertion with learning enabled.

chamfer were effectively allocated into these categories. However, the pattern population within certain categories produced high density of data within regions in the feature space. For instance in $X+$ direction, which is explained below.

During the chamfered circular peg insertion only four patterns were learnt. These patterns corresponded to the $X+$, $X-$, $Y+$ and $Y-$ (see Table 1). The new patterns were valuable to speed up the insertion and to improve the

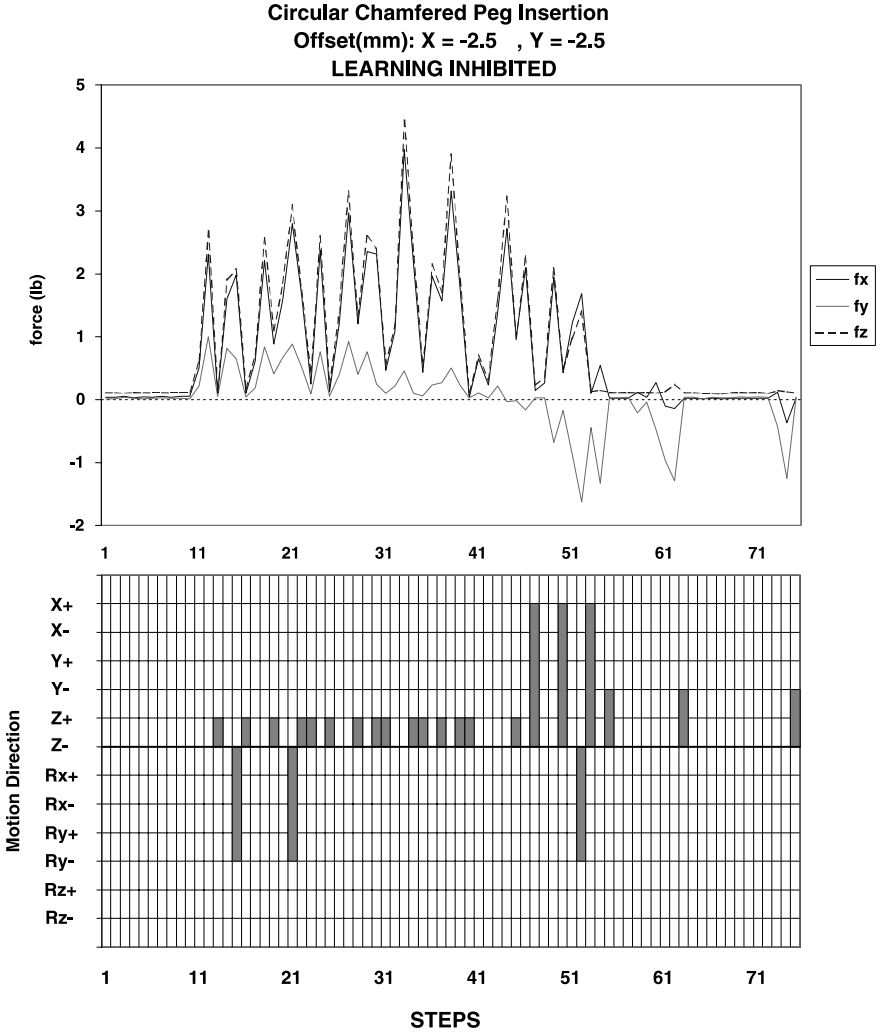
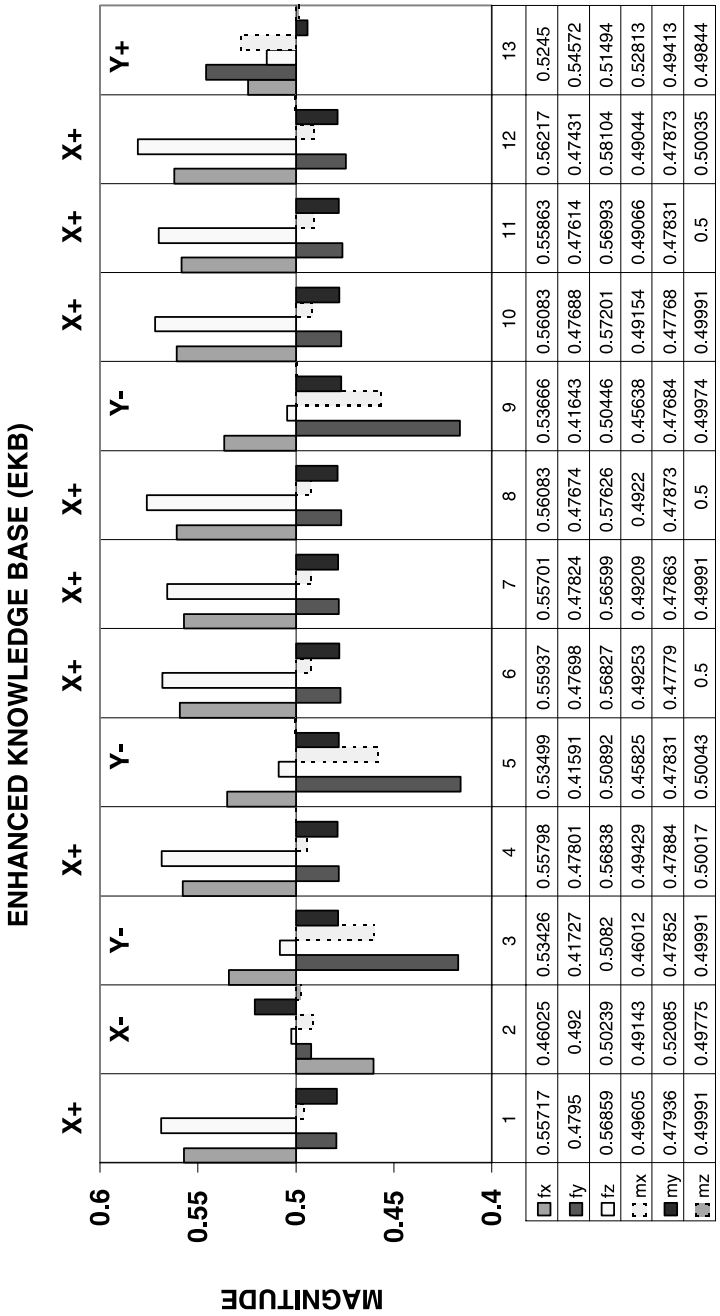


Fig. 11. Insertion with learning inhibited.

insertion trajectory as it was shown during the test. However, these patterns were present within the data more than once and a total of 13 patterns were acquired after 14 insertions which implied that certain categories were more populated. This can be appreciated in Fig. 12 that shows the nature of learned patterns.

As it can be seen, patterns belonging to the same category were very similar. The patterns corresponding to the $X+$ direction were allowed to be learnt eight



	1	2	3	4	5	6	7	8	9	10	11	12	13
fx	0.55717	0.46025	0.53426	0.55798	0.53499	0.55937	0.55701	0.56083	0.53666	0.56083	0.55863	0.56217	0.5245
fy	0.4795	0.492	0.41727	0.47801	0.41591	0.47698	0.47824	0.47674	0.41643	0.47688	0.47614	0.47431	0.54572
fz	0.56859	0.50239	0.5082	0.56838	0.50892	0.56827	0.56599	0.57626	0.50446	0.57201	0.56993	0.58104	0.51494
mx	0.49605	0.49143	0.46012	0.49429	0.45825	0.49253	0.49209	0.4922	0.45638	0.49154	0.49066	0.49044	0.52813
my	0.47936	0.52085	0.47852	0.47884	0.47831	0.47779	0.47863	0.47873	0.47684	0.47768	0.47831	0.47873	0.49413
mz	0.49991	0.49775	0.49991	0.50017	0.50043	0.5	0.49991	0.5	0.49974	0.49991	0.5	0.50035	0.49844

Fig. 12. Learned patterns during the circular chamfered insertion.

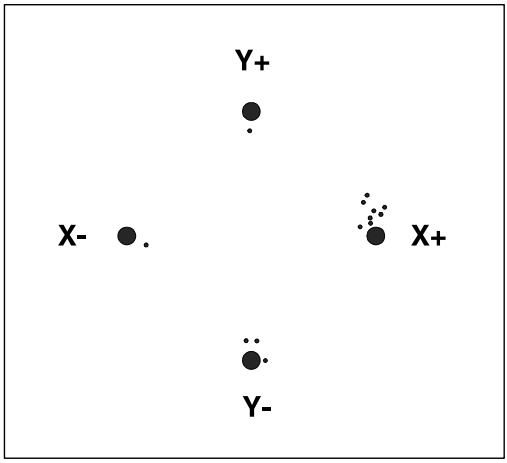


Fig. 13. Data density.

times. This implied that the contact forces were significantly reduced in eight occasions. This high number of patterns populated more the feature space in that area, which is represented in Fig. 13.

For simplicity, only four major areas of action ($X+$, $X-$, $Y+$ and $Y-$) are represented. Initially, the main groups are formed, this is represented by the big black dots as illustrated at the beginning of the operation using what has been termed PKB. The smaller dots represent additional patterns that have been clustered within the same major region. As it is observed, the region belonging to the $X+$ direction was more populated than in the others. The high density of data only implies that there are more data in the region and the cost is memory space. However, since the criteria to learn new patterns was the condition given by the expression $F_{\text{after}} < 0.1 * F_{\text{before}}$, then as the learning progresses, a reduction in contact forces is expected, as it was demonstrated during the experiments, since the robot became more skillful. Being this statement true, it is also true that the knowledge acquisition becomes more strict. This obeys to the fact that forces are smaller as the robot is more skillful and from the above expression forces have also to be smaller to be accepted into the EKB.

Also, as the robot's dexterity improved, the trend in the number of patterns that were accepted into the EKB decreased as it was shown in Table 1. The above expression for allowing the patterns to be learnt resulted to be a criterion to stop automatically the learning.

With this reasoning in mind, it can be demonstrated that the density does not corrupt the selectivity of the NNC, but only affects the memory resources to allocate the learned patterns.

9. Conclusions

Results from our experiments demonstrate that industrial manipulators can learn manipulative skills on-line using only contact force information. The location of the parts was unknown and the information from the environment minimal. The knowledge was enhanced according to the part geometry and provided the required adaptation to the robot to learn a new assembly and improve its skills from experience.

An alternative technique towards the creation of autonomous robots is the use of neural network driven controllers as the NNC presented in this paper. The novel method developed here is generic and can easily be built onto other industrial robots.

Ongoing work is looking at improving the speed of the actual insertion times by reducing the delay imposed in the communication link with the robot controller and the study of components with different geometry. Additionally, future directions have also been envisaged in terms of automating the generation of the PKB, which will create complete autonomous robots for assembly. The core idea is to embed into the robot system a "primitive reflex system" analogous to the reflexes in a human being. Human reflexes are involuntary responses that occur automatically in the presence of certain stimuli. Many of these reflexes are critical for survival and unfold naturally as a part of the infant's development. For instance, the rooting reflex that causes newborn babies to turn their heads toward things that touch their cheeks or, the Babinski reflex, which is the fanning out the baby's toes that happen when the outer edge of the sole of his foot is stroked. These primitive reflexes are lost after few months of life, since new knowledge is being acquired during development which allow babies to develop complex motions. Similarly, a step further in robot development will be to allow the development of the PKB based on these primitive reflexes. So far, the robot has not been able to react automatically, but to recall from what it has been taught. The idea is to allow it to react to completely new experiences and learn from them building up the PKB. This step is feasible and the design will involve contact localisation on the parts to be assembled based on their geometry and generate the corresponding vector motion that diminish the constraint force. These reflexes are needed only at the very beginning of a new operation and once all the 12 primitive motions have been generated creating the PKB, then these reflexes will not be necessary unless the robot needs to learn a completely new operation.

Acknowledgements

The authors wish to thank Rolls Royce & Associates for providing the manipulator. The support of Nottingham Trent University and CONACyT to Dr. Lopez-Juarez in his research work is also acknowledged.

References

[1] I. Lopez-Juarez, M. Howarth, K. Sivayoganathan, Robotics and skill acquisition, in: A. Bramley, T. Mileham, G. Owen (Eds.), Advances in Manufacturing Technology X, 1996, pp. 166–170 (ISBN 1 85790 031 6).
[2] H. Asada, Teaching and learning of compliance using neural nets, IEEE Int. Conf. Robotics and Automation (1990) 1237–1244.
[3] E. Cervera, A.P. del Pobil, Learning and classification of contact states in robotic assembly tasks, in: Proceedings of the 9th International Conference on IEA/AIE, 1996, pp. 725–730.
[4] E. Cervera, A.P. del Pobil, Programming and learning in real-world manipulation tasks, IEEE/RSJ International Conference on Intelligent Robot and Systems (1997) 471–476.
[5] V. Gullapalli, J.A. Franklin, H. Benbrahim, Acquiring robot skills via reinforcement learning, IEEE Control Systems (1994) 13–24.
[6] M. Howarth, An investigation of task level programming for robotic assembly, Ph.D. Thesis, The Nottingham Trent University, 1998.
[7] V. Gullapalli, Skillful control under uncertainty via direct reinforcement learning, Robotics and Autonomous Systems (1995) 237–246.
[8] G.A. Carpenter, S. Grossberg, Adaptive resonance theory (ART), in: M.A. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, MIT Press, Cambridge, MA, 1995, pp. 79–82.
[9] G.A. Carpenter, S. Grossberg, Computer vision, graphics, and image processing, in: A Massively Parallel Architecture for a Sel-Organizing Neural Pattern Recognition Machine, Academic Press, New York, 1987, pp. 54–115.
[10] G.A. Carpenter, S. Grossberg, D.B. Rosen, ART 2-A: an adaptive resonance algorithm for rapid category learning and recognition, Neural Networks 4 (1991) 493–504.
[11] G.A. Carpenter, S. Grossberg, J.H. Reynolds, ARTMAP: supervised real-time learning and classification of nonstationary data by self-organizing neural network, Neural Networks (1991) 565–588.
[12] G.A. Carpenter, S. Grossberg, N. Markunzon, J.H. Reynolds, D.B. Rosen, Fuzzy ARTMAP: a neural network architecture for incremental learning of analog multidimensional maps, IEEE Trans. Neural Networks 3 (5) (1992) 698–713.
[13] J.R. Williamson, Gaussian ARTMAP: a neural network for fast incremental learning of noisy multidimensional maps, Neural Networks 9 (5) (1996) 881–897.
[14] G.A. Carpenter, W.D. Ross, ART-EMAP: a neural network architecture for object recognition by evidence accumulation, IEEE Trans. Neural Networks 6 (4) (1995) 805–818.
[15] A. Nigrin, Neural Networks for Pattern Recognition, MIT Press, Cambridge, MA, 1993.
[16] I. Lopez-Juarez, M. Howarth, K. Sivayoganathan, An adaptive learning approach to control contact force in assembly, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'98), 3, 1998, pp. 1443–1448.