13

Distributed Architecture for Intelligent Robotic Assembly Part II: Design of the Task Planner

Jorge Corona-Castuera and Ismael Lopez-Juarez

1. Introduction

In previous chapter it has been described the overall architecture for multimodal learning in the robotic assembly domain (Lopez-Juarez & Rios-Cabrera, 2006). The acquisition of assembly skills by robots is greatly supported by the effective use of contact force sensing and objects recognition. In this chapter, we will describe the robot's ability to acquire and refine its knowledge through operations (i.e. using contact force sensing during fine motions) and how a manipulator can effectively learn the assembly skill starting from scratch.

The use of sensing to reduce uncertainty significantly extends the range of possible tasks. One source of uncertainty is that the programmer's model of the environment is incomplete. Shape, location, orientation and contact states have to be associated to movements within the robot's motion space while it is in constraint motion. Compliant motion meets external constraints by specifying how the robot's motion should be modified in response generated forces when constraints are violated. Generalizations of this principle can be used to accomplish a wide variety of tasks involving constrained motion, e.g., inserting a peg into a hole or following a weld seam under uncertainty.

The success of robotic assembly operations therefore, is based on the effective use of compliant motion, 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 within the workspace. However, in reality uncertainties due to manufacturing tolerances, positioning, sensing and control make it difficult to perform the assembly. Compliant motion can be achieved by using passive devices such as the Remote Centre Compliance (RCC) introduced by Whitney (Whitney & Nevis, 1979) or other improved versions of the device (Joo & Miyasaki, 1998). Other alternative is to use Active Compliance, which actually modifies either the position of the manipulated component as a response to constraint forces or the desired force. Some com-

Source: Manufacturing the Future, Concepts - Technologies - Visions , ISBN 3-86611-198-3, pp. 908, ARS/plV, Germany, July 2006, Edited by: Kordic, V.; Lazinica, A. & Merdan, M.

mercial devices have emerged in recent years to aid industrial applications (Erlbacher, 2004).

Active compliance can be roughly divided into fine motion planning and reactive control. Fine motion planning relies on geometrical path planning whereas reactive control on the synthesis of an accommodation matrix or mapping that transform the corresponding contact states to corrective motions. A detailed analysis of active compliance can be found in (Mason, 1983) and (De Schutter & Brussel, 1988). Perhaps, one of the most significant works in fine motion planning is the work developed by Lozano-Perez, Mason and Taylor known as the LMT approach (Lozano-Perez, et al, 1984). The LMT approach automatically synthesizes compliant motion strategies from geometric descriptions of assembly operations and explicit estimates of the errors in sensing and control. Approaches within fine motion planning can also be further divided into model-based approaches and connectionist-based approaches though, some reactive control strategies can be well accommodated within the model-based approach. In either case, a distinctive characteristic in model-based approaches is that these take as much information of the system and environment as possible. This information includes localisation of the parts, part geometry, material types, friction, errors in sensing, planning, and control, etc. On the other hand, the robustness of the connectionist-based approaches relies on the information given during the training stage that implicitly considers all the above parameters.

In this chapter we present a "Task Planner", connectionist-based approach that uses vision and force sensing for robotic assembly when assembly components geometry, location and orientation is unknown at all times. The assembly operation resembles the same operation as carried out by a blindfold human operator. The task planner is divided in four stages as suggested in (Doersam & Munoz, 1995) and (Lopez-Juarez, 2000):

Pre-configuration: From an initial configuration of the hand/arm system, the expected solutions are the required hand/arm collision-free paths in which the object can be reached. To achieve this configuration, it is necessary to recognize invariantly the components and determining their location and orientation.

Grasp: Once the hand is in the Pre-configuration stage, switching strategies between position/force controls need to be considered at the moment of contact and grasping the object. Delicate objects can be broken without a sophisticated contact strategy even the Force/Torque (F/T) sensor can be damaged.

Translation: After the object is firmly grasped, it can be translated to the assembly point. The possibility of colliding with obstacles has to be taken into account.

Assembly Operation: The assembly task requires robust and reactive positions/force control strategies. Mechanical and geometrical uncertainties make high demands on the controller.

The pre-configuration for recognition and location of components as well as the assembly operation are based on FuzzyARTMAP neural network architecture, situated under the connectionist-based approach employing reactive contact forces.
In this approach, the mapping between contact states and arm motion commands is achieved by using fuzzy rules that create autonomously an Acquired-Primitive Knowledge Base (ACQ-PKB) without human intervention. This ACQ-PKB is then further used by the Neural Network Controller (NNC) for compliance learning.

2. Related Work

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 (Lopez-Juarez & Howarth, 1996), (Asada, 1990), (Cervera & del Pobil, 1996), or being implemented on real robots (Cervera & del Pobil, 1997), (Gullapalli, et al, 1994), (Howarth, 1998), (Cervera & del Pobil, 2002). 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 controller was a back-propagation network with 11 inputs. These are the sensed positions and forces: (X, Y, Z, θ_1, θ_2) and (F_x, F_y, F_z, m_x, m_y, m_z). The output of the network was the position commands. The performance of the operation was evaluated by a parameter r, which measured the performance of the controller. r varied between 0 to 1 and was a function of the sensed peg position and the nominal hole location. The network showed a good performance after 150 trials with insertion times lower than 100 time steps (Gullapalli, 1995). Although the learning capability demonstrated during experiments improved over time, the network was unable to generalise over different geometries. Insertions are reported with

both circular and square geometries; however, when inserting the square peg, its rotation around the vertical axis was not allowed, which facilitated the insertion. M. Howarth followed a similar approach, using also backpropagation in combination with reinforcement learning. In comparison with Gullapalli's work, where the reinforcement learning values were stochastic, Howarth's reinforcement value was based on two principles: minimization of force and moment values and continuation of movement in the assembly direction. This implied that whenever a force or moment value was above a threshold, an action (i.e., reorientation), should occur to minimize the force. Additionally, movements in the target assembly direction were favoured. 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 (Howarth, 1998). 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 (Cervera & del Pobil, 1997); 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 with any further improvement. 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.

Other interesting approaches have also been used for skill acquisition within the framework of Robot Programming by Demonstration that considers the characteristics of human generated data. Work carried out by (Kaiser & Dillman, 1996) shows that skills for assembly can be acquired through human demonstration. The training data is first pre-processed, inconsistent data pairs are removed and a smoothing algorithm is applied. Incremental learning is achieved through Radial Basis Function Networks and for the skill refinement;

the Gullapalli's Stochastic Reinforcement Value was also used. The methodology is demonstrated by the peg-in-hole operation using the circular geometry. On the other hand (Skubic & Volz, 2000 b), use a hybrid control model which provides continuous low-level force control with higher-level discrete event control. The learning of an assembly skill involves the learning the mapping of force sensor signals to Single-Ended Contact Formations (SECF), the sequences of SECFs and the transition velocity commands which move the robot from the current SECF to the next desired SECF. The first function is acquired using supervised learning. The operator demonstrates each SECF while force data is collected, and de data is used to train a state classifier. The operator then demonstrates a skill, and the classifier is used to extract the sequence of SECFs and transitions velocities which comprise the rest of the skill.

The above approaches can be divided in two groups, those providing autonomous assembly skill and those which teach the skill by demonstration. These approaches have given some inputs to our research and the work presented here is looking to improve some of their limitations. In Gullapalli's work the hole location has to be known. Howarth improved the autonomy by obviating the hole's location; however, the lengthy training process made this approach impractical. Cervera considered many contact states, which worked well also during the assembly of different type of components. In the case of teaching the skill by demonstration, the method showed by Kaiser and Dillman was lengthy for real-world problems and the work by Skubic and Volz assumes that during supervised training the operator must know which SECF classes to include in the set.
The integration of vision systems to facilitate the assembly operations in uncalibrated workspaces is well illustrated in (Jörg, et al, 2000) and (Baeten, et al, 2003) using eye-in-hand vision for different robotic tasks.

3. Workplace Description

The manufacturing cell used for experimentation is integrated by a KUKA KR15/2 industrial robot. It also comprises a visual servo system with a ceiling mounted camera as shown in figure 1. The robot grasps the male component from a conveyor belt and performs the assembly task in a working table where the female component is located. The vision system gets an image to calculate the object's pose estimation and sends the information to the robot from two predefined zones:

Zone 1 which is located on the conveyor belt. The vision system searches for the male component and determines the pose information needed by the robot.
Zone 2 is located on the working table. Once the vision system locates the female component, it sends the information to the NNC.

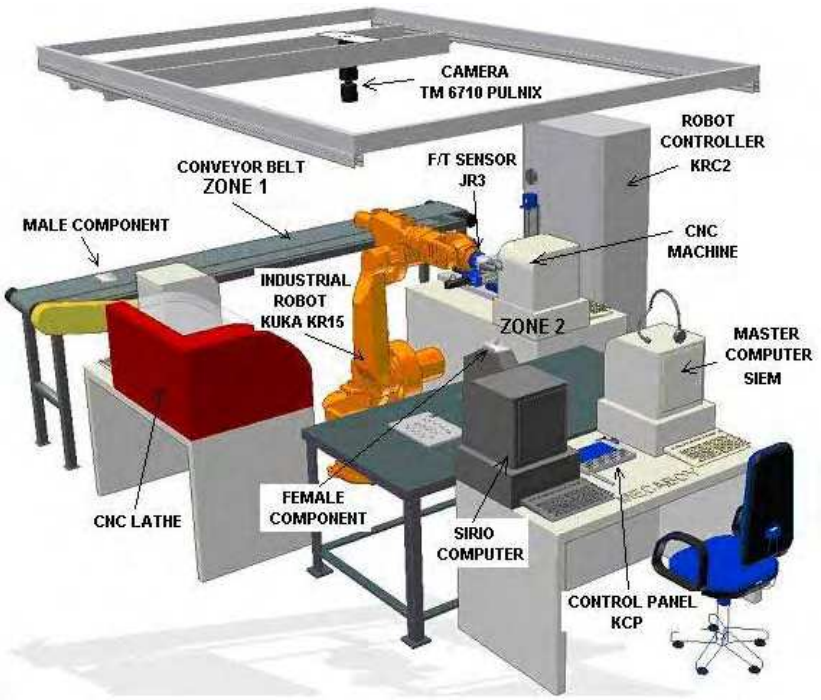

Figure 1. Manufacturing cell

The NNC for assembly is called SIEM (Sistema Inteligente de Ensamble Mecánico) and is based on a FuzzyARTMAP neural network working in fast learning mode (Carpenter, et al, 1992). The vision system, called SIRIO (Sistema Inteligente de Reconocimiento Invariante de Objetos), also uses the same neural network to learn and classify the assembly components (Pena-Cabrera & Lopez-Juarez, 2006). The SIRIO was implemented with a high speed camera CCD/B&W, PULNIX 6710, with 640x480 resolution; camera movements on the X and Y axis were implemented using a 2D positioning system.

For experimental purposes three canonical peg shapes were used: circular, square and radiused-square as it is shown in figure 2. Both, chamfered and chamferless female components were employed during experimentation.

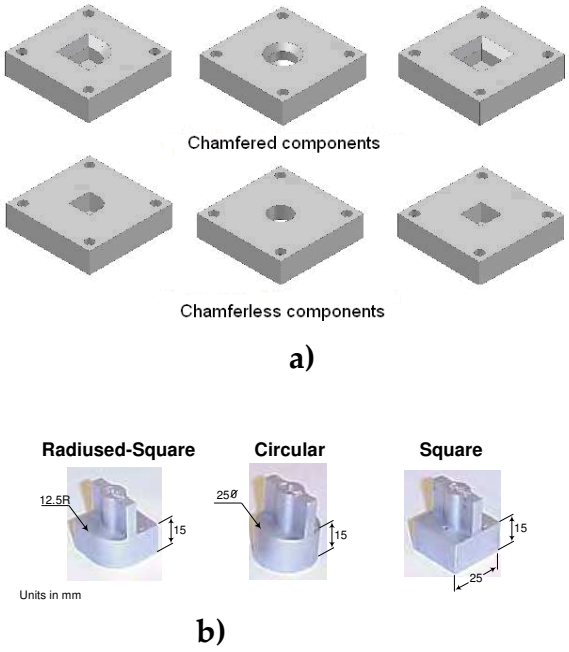

Figure 2. a) Female assembly components, b) Male assembly components

4. Assembly methodology

4.1 Pre-Configuration

4.1.1 Starting from scratch

Initially, the robot system does not have any knowledge. To accomplish the very first assembly the robot has to acquire a Primitive Knowledge Base (PKB) using an interactive method.

a) Given Primitive Knowledge Base (GVN-PKB)
The formation of the PKB basically consists of showing the robot how to react to individual components of the F/T vector. This procedure results in creating the required mapping between contact states and robot motions within the motion space– linear, angular and diagonal movements- , this is illustrated in figure 3. The Given PKB (GVN-PKB) used for the experiments reported in this chapter considered rotation around Z axis and diagonal motions as it is illustrated in figure 4.

Using the above mentioned GVN-PKB to start the learning of the assembly skill, it showed to be effective, however the robot still lacked for autonomy and it was realized that sometimes the robot did not used all the information given in the GVN-PKB and also it was noticed a difference between the taught contact forces the actual forces occurring during assembly so that an autonomously created PKB was needed in order to provide complete self-adaptive behaviour to the robot.

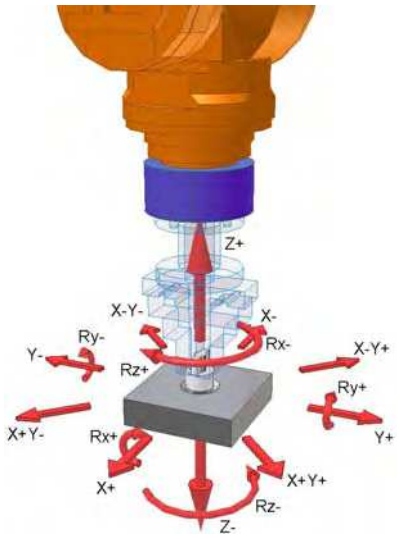

Figure 3. Motion space

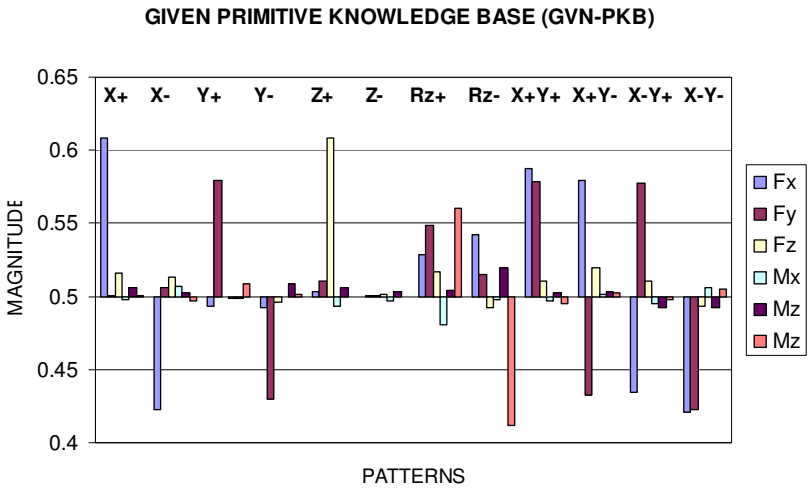

Figure 4. Given PKB (GVN-PKB)

b) Acquired Primitive Knowledge Base (ACQ-PKB)

It was decided to embed a fuzzy logic mechanism to autonomously acquire an initial knowledge from the contact states. That is, learning the mapping from scratch without knowledge about the environment. The only instruction given to the robot was the task – assembly – in order to start moving downwards. When the contact is made the robot starts acquiring information about the contact states following fuzzy rules and autonomously generating the corresponding motion commands and forming the Acquired PKB (ACQ-PKB). During the first contact, the fuzzy algorithm determines the type of operation: chamfered or chamferless assembly and chooses the rules to apply depending of moments and forces magnitude presents in X and Y directions.

Fuzzy logic have proved to be useful to model many decision taking processes in presence of uncertainty or where no precise knowledge of the process exist in an attempt to formalize experience and empiric knowledge of the experts in a specific process. The initial knowledge from our proposal comes from a static and dynamic force analysis when the components are in contact assuming that there is an error in the position with respect to the centre of insertion. With the aid of dynamic simulation software (ADAMS), the behaviour of the contact impact is obtained for different situations which are to be solved by the movements of the manipulator.

There are 12 defined motion directions (X+, X-, Y+, Y-, Z+, Z-, Rz+, Rz-, X+Y+, X+Y-, X-Y+ and X-Y-) and for each one there is a corresponding contact state. An example of these contact states for a chamfered female squared component is shown in figure 5. The contact states for linear motion X+, X-, Y+, Y-, and linear combined motions X+Y+, X+Y-, X-Y+, X-Y- are shown in figure 5(a). In figure 5(b), it is shown a squared component having four contact points. Figures 5(c) and 5(d) provide additional patterns for rotation Rz- and Rz+ respectively when the component has only one point of contact. The contact state for mapping Z+ is acquired making vertical contact between component and a horizontal surface, Z- direction is acquired with the component is in free space. This approach applies also for chamfered circular and radius-squared components as well as the chamferless components.

It is stated to use the following considerations for the generation of the fuzzy rules: a) Number of linguistic values: 2 (minimum, maximum), b) Number of input variables: 12 (*Fxp, Fxn, Fyp, Fyn, Fzp, Fzn, Mxp, Mxn, Myp, Myn, Mzp, Mzn*) and c) Maximum number of rules: $12^2 = 144$ (only 24 were used).

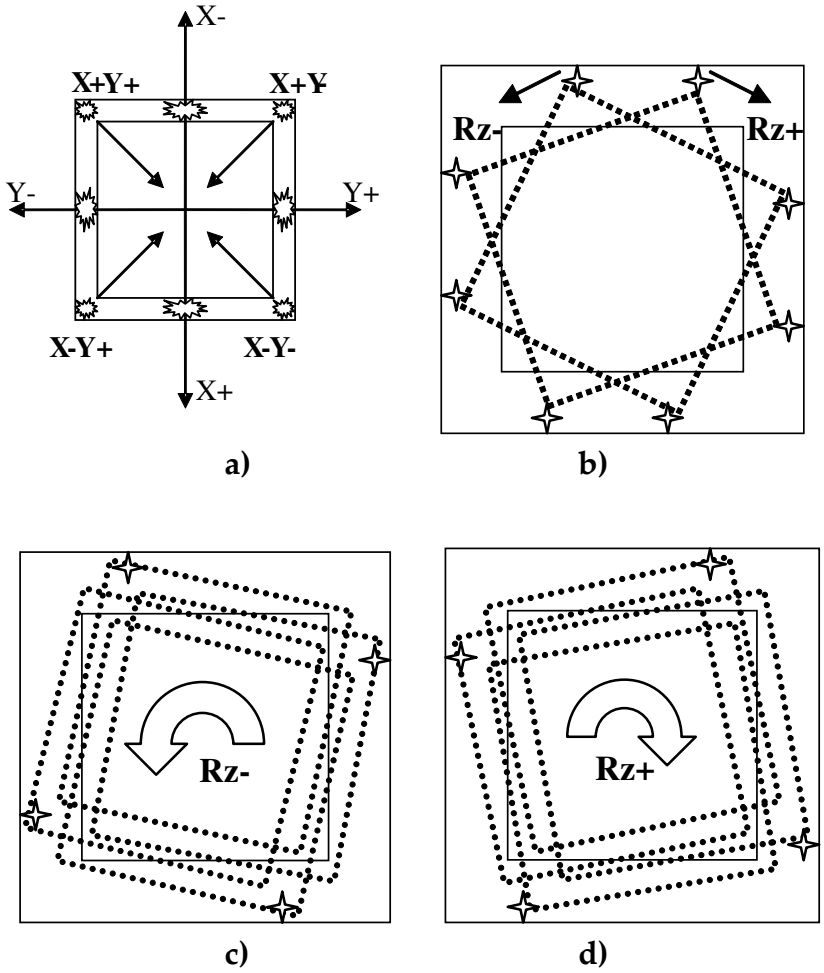

Figure 5. Contacts between chamfered components while acquiring the primitive knowledge base,

a) Linear movements,
b) Pure rotation Rz+ and Rz-,
c) Rotation Rz-,
d) Rotation Rz+.

The membership functions are stated as showed in figure 6. Forces and moments have normalised values between 0 and 1. The normalization was *ad-hoc* and considered the maximum experimental value for both, force and moment values. No belong functions were defined for the output, because our process does not includes defuzzification in the output. The function limit values are chosen heuristically and according to previous experience in the assembly operation.

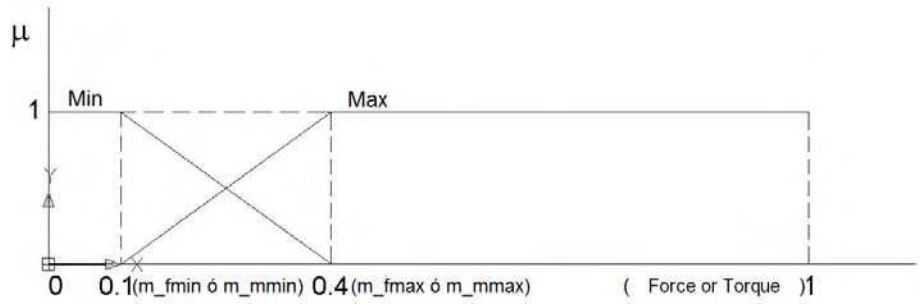

Figure 6. Membership functions

Having those membership values, antecedents and consequents defined, then the Rule Statement can be generated and the ACQ-PKB created. An example of these rules for chamfered assembly is given in table 1.

IF	Fxp	Fxn	Fyp	Fyn	Fzp	Fzn	Mxp	Mxn	Myp	Myn	Mzp	Mzn	THEN	DIR
IF	Max	Min	Min	Min	Max	Min	Min	Min	Max	Min	Min	Min	*THEN*	X+
IF	Max	Min	Min	Min	Max	Min	Max	Min	Max	Min	Min	Min	*THEN*	X+
IF	Min	Max	Min	Min	Max	Min	Min	Min	Min	Max	Min	Min	*THEN*	X-
IF	Min	Max	Min	Min	Max	Min	Min	Max	Min	Max	Min	Min	*THEN*	X-
IF	Min	Min	Max	Min	Max	Min	Min	Max	Min	Min	Min	Min	*THEN*	Y+
IF	Min	Min	Max	Min	Min	Min	Min	Min	Min	Max	Min	Min	*THEN*	Y+
IF	Min	Min	Min	Max	Max	Min	Max	Min	Min	Max	Min	Min	*THEN*	Y-
IF	Min	Min	Min	Max	Max	Min	Min	Min	Min	Min	Min	Min	*THEN*	Y-
IF	Min	Min	Min	Min	Max	Min	Min	Min	Min	Min	Min	Min	*THEN*	Z+
IF	Min	Min	Min	Min	Min	Min	Min	Min	Min	Min	Min	Min	*THEN*	Z-
IF	Min	Min	Min	Min	Max	Min	Min	Min	Min	Min	Max	Min	*THEN*	Rz+
IF	Max	Min	Min	Min	Max	Min	Min	Min	Max	Min	Max	Min	*THEN*	Rz+
IF	Min	Max	Min	Min	Max	Min	Min	Min	Min	Max	Max	Min	*THEN*	Rz+
IF	Min	Min	Max	Min	Max	Min	Min	Max	Min	Min	Max	Min	*THEN*	Rz+
IF	Min	Min	Min	Max	Max	Min	Max	Min	Min	Min	Max	Min	*THEN*	Rz+
IF	Min	Min	Min	Min	Max	Min	Min	Min	Min	Min	Min	Max	*THEN*	Rz-
IF	Max	Min	Min	Min	Max	Min	Min	Min	Max	Min	Min	Max	*THEN*	Rz-
IF	Min	Max	Min	Min	Max	Min	Min	Min	Min	Max	Min	Max	*THEN*	Rz-
IF	Min	Min	Max	Min	Max	Min	Min	Max	Min	Min	Min	Max	*THEN*	Rz-
IF	Min	Min	Min	Max	Max	Min	Max	Min	Min	Min	Min	Max	*THEN*	Rz-
IF	Max	Min	Max	Min	Max	Min	Min	Max	Max	Min	Min	Min	*THEN*	X+Y+
IF	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Min	Min	*THEN*	X+Y-
IF	Min	Max	Min	Max	Max	Min	Min	Max	Min	Max	Min	Min	*THEN*	X-Y+
IF	Min	Max	Min	Max	Max	Min	Max	Min	Min	Max	Min	Min	*THEN*	X-Y-

Table 1. Fuzzy rules for chamfered assembly

For chamferless assembly another knowledge base would have to be generated using similar rules as shown above, but without considering force in axis X and Y. The reason is that these forces in comparison with the moments generated around those axes are very small. The inference machine determines the rules to apply in a given case.

To quantify the fuzzy output response a fuzzy logic membership value is used. For the "AND" connector we used the product criteria (Driankov, et al, 1996), and to obtain a conclusion, the maximum value for the fuzzy outputs in the expression (1) response was used.

$$
\begin{aligned}
&X+ = Fxp_{\max} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\max} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&X+ = Fxp_{\max} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\max} * Mxn_{\min} * Myp_{\max} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&X- = Fxp_{\min} * Fxn_{\max} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\min}\\
&X- = Fxp_{\min} * Fxn_{\max} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\max} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\min}\\
&Y+ = Fxp_{\min} * Fxn_{\min} * Fyp_{\max} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\max} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&Y+ = Fxp_{\min} * Fxn_{\min} * Fyp_{\max} * Fyn_{\min} * Fzp_{\min} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\min}\\
&Y- = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\max} * Fzp_{\max} * Mxp_{\max} * Mxn_{\min} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\min}\\
&Y- = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\max} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&Z+ = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&Z- = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\min} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&Rz+ = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\max} * Myn_{\min} * Mzp_{\max} * Mzn_{\min}\\
&Rz+ = Fxp_{\max} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\max} * Myn_{\min} * Mzp_{\max} * Mzn_{\min}\\
&Rz+ = Fxp_{\min} * Fxn_{\max} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\max} * Mzp_{\max} * Mzn_{\min}\\
&Rz+ = Fxp_{\min} * Fxn_{\min} * Fyp_{\max} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\max} * Myp_{\min} * Myn_{\min} * Mzp_{\max} * Mzn_{\min}\\
&Rz+ = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\max} * Fzp_{\max} * Mxp_{\max} * Mxn_{\min} * Myp_{\min} * Myn_{\min} * Mzp_{\max} * Mzn_{\min}\\
&Rz- = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\max}\\
&Rz- = Fxp_{\max} * Fxn_{\min} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\max} * Myn_{\min} * Mzp_{\min} * Mzn_{\max}\\
&Rz- = Fxp_{\min} * Fxn_{\max} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\min} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\max}\\
&Rz- = Fxp_{\min} * Fxn_{\min} * Fyp_{\max} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\max} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\max}\\
&Rz- = Fxp_{\min} * Fxn_{\min} * Fyp_{\min} * Fyn_{\max} * Fzp_{\max} * Mxp_{\max} * Mxn_{\min} * Myp_{\min} * Myn_{\min} * Mzp_{\min} * Mzn_{\max}\\
&X+Y+ = Fxp_{\max} * Fxn_{\min} * Fyp_{\max} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\max} * Myp_{\max} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&X+Y- = Fxp_{\max} * Fxn_{\min} * Fyp_{\min} * Fyn_{\max} * Fzp_{\max} * Mxp_{\max} * Mxn_{\min} * Myp_{\max} * Myn_{\min} * Mzp_{\min} * Mzn_{\min}\\
&X-Y+ = Fxp_{\min} * Fxn_{\max} * Fyp_{\max} * Fyn_{\min} * Fzp_{\max} * Mxp_{\min} * Mxn_{\max} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\min}\\
&X-Y- = Fxp_{\min} * Fxn_{\max} * Fyp_{\min} * Fyn_{\min} * Fzp_{\max} * Mxp_{\max} * Mxn_{\min} * Myp_{\min} * Myn_{\max} * Mzp_{\min} * Mzn_{\min}
\end{aligned}
\qquad (1)
$$

Once the algorithm values have been generated, a routine which allows the manipulator for autonomous database generation is created. The mapping acquisition between generated contact states-arm motion commands starts from the insertion centre. This information is determined by calculating the centroid of the component by the vision system. Positional errors due to the image

processing are about 1 mm to 2 mm which were acceptable for the experimental work since the assembly was always successful. The manipulator starts moving in every possible direction generating a knowledge database. The results given in this research considered only 24 patterns as indicated in the fuzzy rules shown in table 1, omitting the rotations around the X and Y axis since only straight insertions were considered. Some patterns generated with this procedure for the chamfered and chamferless square peg insertion can be observed in figure 7.

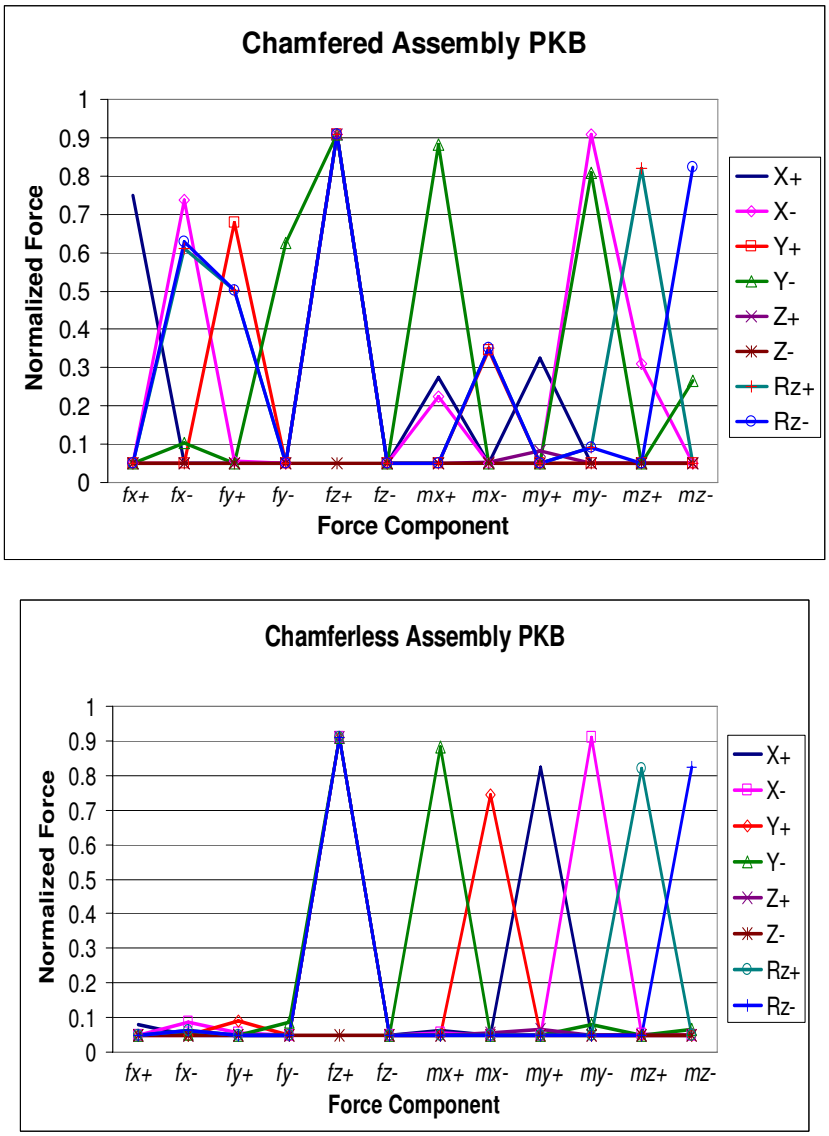

Figure 7. ACQ-PKB, left chamfered assembly, right chamferless assembly

In order to get the next motion direction the forces are read, normalized and classified using the NNC on-line. The F/T pattern obtained from the sensor provides a unique identification. The F/T vector (2) comprises 12 components given by the 6 data values (positive and negative).

$$[Current\,F/T] = \left[fx, fx-, fy, fy-, fz, fz-, mx, mx-, my, my-, mz, mz-\right]^T \quad (2)$$

4.1.2 Acquiring location and component type

The SIRIO system employs the following methodology: a) Finding the region of interest (ROI), b) Calculate the histogram of the image, d) Search for components, e) Centroid calculation, f) Component orientation, g) Calculate Boundary Object Function (BOF), distances between the centroid and the perimeter points, h) Descriptor vector generation and normalization (CFD&POSE) and i) Information processing in the neural network.
The descriptive vector is called CFD&POSE (Current Frame Descriptor and Pose) and it is conformed by (3):

$$\left[CDF\ \&\ POSE\right] = [D_1, D_2, D_3, ..., D_n, X_c, Y_c, \theta, Z, ID]^T \quad (3)$$

Where: $\mathbf{D}_i$ are the distances from the centroid to the perimeter of the object. (180 data values)

- $\mathbf{X}_C$, $\mathbf{Y}_C$, are the centroid coordinates.
- ϕ, is the orientation angle.
- **Z** is the height of the object.
- **ID** is a code number related to the geometry of the components.′

With this vector and following the above methodology, the system has been able to classify invariantly 100% of the components presented on-line even if they are not of the same size, orientation or location and for different light conditions, see (Pena-Cabrera & Lopez-Juarez, 2006) for details.

The CFD&POSE vector is invariant for each component and it is used for classification. The vector is normalized to 185 data dimension and normalized in

the range [0.0 – 1.0]. The normalization of the BOF is accomplished using the maximum divisor value of the vector distance. This method allows having very similar patterns as input vectors to the neural network, getting a significant improvement in the operation system. In our experiments, the object recognition method used the above components having 210 patterns as primitive knowledge to train the neural network. It was enough to recognize the assembly components with $\varrho_a = 0.2$ (base vigilance), $\varrho_{map} = 0.7$ (vigilance map) and $\varrho_b = 0.9$ parameters, however, the SIRIO system can recognize more complex components (Pena-Cabrera, et al, 2005).

4.2 Grasp

At this stage, the PKB has been acquired and the location information sent to the robot. The motion planning from Home position to zone 1 uses the male component given coordinates provided by SIRIO. The robot uses this information and the F/T sensor readings to grasp the piece and to control the motion in Z direction for two stages:

a) The security stage
In the event that position and orientation of the male component, given by SIRIO, have an error larger than 5 mm in *X* or *Y* axis and 10º around Z direction. Sensing is executed during 10 movements in *Z*- direction with manipulator steps of 0.2 mm. In this stage a collision is possible to occur between gripper and components. The system reacts moving to home position when a force limit in Z direction is reached (4 N). The robot continues its trajectory in Z- direction until a distance of 1 mm component is reached.

b) Grasp Component
This sensing stage begins just before the robot touches the component. The sensor is read every 0.1 mm executed by manipulator, this stage ends when the robot touches the component, in this situation the force magnitude in Z direction is at least 4 N, then the condition to grasp (close gripper) is satisfied.

4.3 Translation

The translation is similar to the grasping operation in zone 1. The path to move the robot from zone 1 to zone 2 (assembly point) is accomplished by using the

coordinates given by the SIRIO system. The possibility of collision with obstacles is avoided using bounded movements.

4.4 Assembly Operation

4.4.1 Neural Network Controller (NNC)

a) ART Models

Several works published in the literature inspired ideas about contact recognition and representation (Xiao & Liu, 1998), (Ji & Xiao, 1999), (Skubic &Volz, 1996), however the fuzzy representation appealed to be suitable to expand the NNC capability and further work was envisaged to embed the automatic mechanism to consider contact states that are actually present in a specific assembly operation. It was believed that by using only useful information, compliance learning could be effective in terms of avoiding learning unnecessary contact information, hence also avoiding unnecessary motions within the motion space.

The Adaptive Resonance Theory (ART) is a well established associative brain and competitive model introduced as a theory of the human cognitive processing developed by Stephen Grossberg at Boston University. Grossberg suggested that connectionist models should be able to adaptively switch between its *plastic* and *stable* modes. That is, a system should exhibit plasticity to accommodate new information regarding unfamiliar events. But also, it should remain in a stable condition if familiar or irrelevant information is being presented. An analysis of this instability, together with data of categorisation, conditioning, and attention led to the introduction of the ART model that stabilises the memory of self-organising feature maps in response to an arbitrary stream of input patterns (Grossberg, 1976).

The theory has evolved in a series of real-time architectures for unsupervised learning, the ART-1 algorithm for binary input patterns (Carpenter & Grossberg, 1987). Supervised learning is also possible through ARTMAP (Carpenter, et al, 1991) that uses two ART-1 modules that can be trained to learn the correspondence between input patterns and desired output classes. Different model variations have been developed to date based on the original ART-1 algorithm, ART-2, ART-2a, ART-3, Gaussian ART, EMAP, ViewNET, Fusion ARTMAP, LaminART just to mention but a few.

b) NNC Architecture

The functional structure of the assembly system is illustrated in figure 8. The Fuzzy ARTMAP (FAM) (Carpenter, et al, 1992) is the heart of the NNC. The controller includes three additional modules. The Knowledge Base that stores initial information related to the geometry of the assembling parts and which is autonomously generated. The Pattern-Motion Selection module keeps track of the appropriateness of the F/T patterns to allow the FAM network to be re-trained. If this is the case, the switch SW is closed and the corresponding pattern-action provided to the FAM for on-line retraining. The selection criterion is given by expression (3), discussed next.

Future predictions will be based on this newly trained FAM network. The Automated Motion module basically is in charge of sending the incremental motion request to the robot controller and handling the communication with the Master Computer.

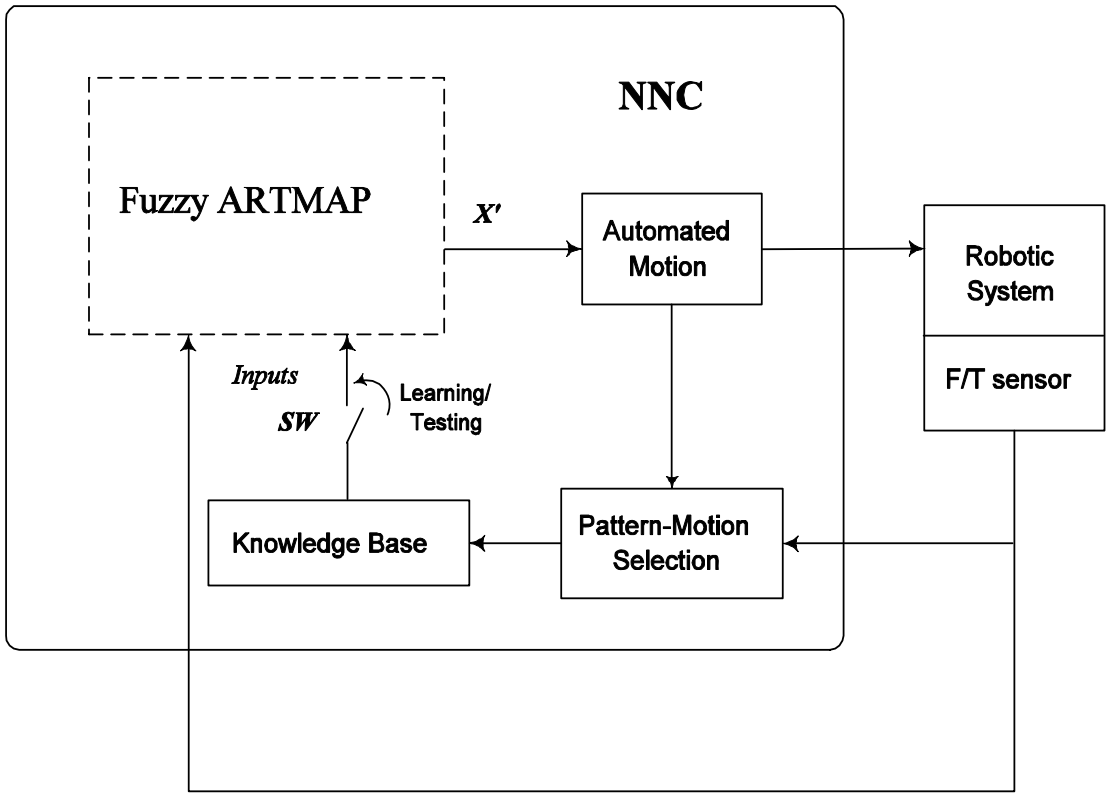

Figure 8. System Structure