CAM-Rob Postprocessor based on a fuzzified Redundancy Resolution Scheme

Javier Andres
Dept. of Mechanical Engineering and Construction
Universitat Jaume I
Avda Vicent Sos Baynat s/n
12071 Castellon, Spain
Tel.: +34 964728143, Fax: +34 964728106
fandres@emc.uji.es

Luis Gracia
Instituto IDF
Universitat Politècnica de València,
Camino de Vera s/n
46022 Valencia, Spain

Josep Tornero
Instituto IDF
Universitat Politècnica de València,
Camino de Vera s/n
46022 Valencia, Spain

ABSTRACT

This work highlights the applicability of different redundancy resolution schemes to the postprocessing stage from a CAM system to an industrial redundant workcell. The inverse kinematic problem for redundant manipulators is not straightforward and, therefore, it is commonly solved using an iteratively approach based on redundant resolution schemes at the velocity level. In this work, two conceptions of redundancy resolution schemes are evaluated and a novel fuzzy inference system is developed to improve the performance during the toolpath tracking in order to avoid singularities and to maintain a preferred reference posture. For this purpose, the fuzzy inference engine properly adjusts the weight of each joint in the calculation of the performance criterion vectors. The proposed approach is validated in the real prototyping of a windmill blade mold using a KUKA KR15/2 manipulator mounted on a linear track and synchronized with a rotary table. To the authors' knowledge, the proposed method and the results shown are novel in the context of postprocessing techniques from CAM systems to industrial robots devoted to milling works. With the same guidelines, the postprocessor programmed inside the CAM system is expected to be easily applicable not only to other industrial robots, but also for different applications such as welding or painting labors.

Keywords: *Redundant robot, fuzzy logic, robot milling.*

1. Introduction

Rapid prototyping with soft materials is increasing to support the product development process in *industrial design engineering* in order to get physical replicas of CAD (*Computer Aided Design*) defined models. In this context, large prototypes require redundant robotic workcells due to their high flexibility and large working areas. Leading commercial CAM (*Computer Aided Manufacturing*) systems plan off-line the cutting toolpaths as a discrete set of close-enough tool poses. Since the cutter's tracking data are directly related with the desired workpiece finish conditions, these data are mandatory and independent from the machine tool that will manufacture the workpiece (leaving aside calibration efforts [1]). Moreover, this information has to be *postprocessed* (i.e., adapted) from the CAM system to the production system that is going to be used. Therefore, this work evaluates several Redundancy Resolution Schemes (RRS) at the velocity level to deal with the postprocessing stage from the CAM system to the redundant workcell.

The structure of the paper is as follows. After describing the workcell in Section 2, Section 3 introduces several performance criterions and develops their optimal combination by means of Fuzzy Logics (FL). Next, section 4 presents the proposed CAM-Robotics integrated postprocessor for the automatic off-line generation of the robot commands to carry out milling tasks. Subsequently, section 5 tests the designed postprocessor and shows the real prototyping of a windmill blade mold. Finally, some concluding remarks are given.

2. Description of the workcell

At the *Design and Manufacturing Institute* (IDF) of the *Technical University of Valencia*, a sculpturing redundant workcell has been configured to test milling methods for rapid prototyping. As shown in the Fig. 1, an industrial KUKA KR15/2 arm with six revolute (R) joints is mounted on a linear track (where d_L is the angular displacement of this additional prismatic joint, P), and it works over a synchronized rotary table (where θ_M is the angular displacement of this additional R-joint) on which the initial blank of material is set. Since P and R joints allow one *degree of freedom* (DOF), both additional joints plus the six joints of the robotic arm complete a workcell with a *Joint space* ($\Im$) of dimension *n* equal to eight. Moreover, as the milling tool has a symmetry axis that allows rotating the tool without affecting the task, these systems only specify five parameters to carry out the milling task and, hence, at milling works the dimension *l* of the *Task space* (T) is equal to five. For the kinematic analysis, the table can be arbitrary regarded as fixed, while the movable *end-effector* (EE) bears the cutter tool in the Cartesian *Operational workspace* (Ω) whose dimension *m* is equal to six, i.e., the pose of a rigid body in Ω is specified with three linear plus three angular coordinates.

This workcell is *redundant* as $n > l$ $(with\, \mathrm{T} \subseteq \Omega)$ [2], with a *kinematic redundancy degree* r_K of three, i.e., $r_K = n - l = 3$. Therefore, the main difficulty of postprocessing a toolpath from the CAM system to this workcell focuses on managing the redundancy in order to avoid manipulator postures near singularities or limits of range, while reaching the successive cutter poses of the toolpath.

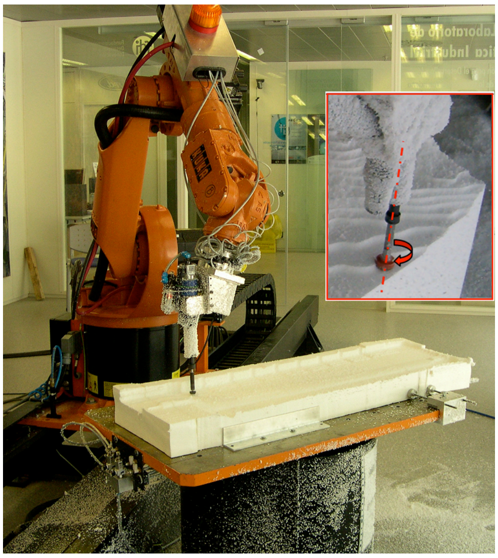

Fig. 1 KUKA workcell at the IDF of the Technical University of Valencia and detail of the irrelevant axis of symmetry of the milling tool while milling an expanded polystyrene.

The *kinematic model* of the robot is the mathematical description required to control adequately the *posture* of the chain and the associated *pose* of the EE while performing a task. On the one hand, the *Direct Kinematic Problem* (DKP) is the mapping from the *Joint space* ($\Im$) to the *Operational space* (Ω), i.e., determining the pose of the EE for a given robot posture. On the other, the *Inverse Kinematic Problem* (IKP) consists of determining the posture of the robot for a given pose of its EE. At the *displacement* level, the DKP is straightforward. Thus, a point in $\Im$ represents a unique pose of the EE referred to the base {B} at Ω. The *standard* Denavit-Hartenberg (DH) model [3], which has widespread acceptance among the scientific community, is considered in this work. It represents the EE pose as a 4×4 homogeneous transformation matrix that results from the operation of the workcell descriptive parameters (a_i, α_i, d_i, θ_i) depicted in Fig. 2 and summarized in Table 1. The joint variable is θ_i for revolute joints and d_i for a prismatic joints. The IKP is essential for CAM systems in order to map the cutter tracking poses from Ω to $\Im$. The IKP at the *displacement* level is more challenging for redundant robots since an infinite number of solutions may exist. The resolution of the IKP for the IDF's workcell is described at [4] taking as entry arguments the current θ_M and d_L values.

Whitney [5] first introduced differential kinematic relationships to solve the so-called *resolved-motion rate* control. The DKP at the velocity level is given by the linear algebraic equation:

$$t = J \cdot \dot{q} \tag{1}$$

where t is EE velocity vector, $\dot{q}$ is the joint velocity vector and J is the Jacobian matrix, or simply *Jacobian*, which is a non-linear function of the joint angles. In this work it will be used the so-called *geometric* Jacobian [7] since it simplifies the computations [8].

Therefore, the Jacobian matrix maps the joint rates, i.e. the 8-dimensional vector $\dot{q} = [\dot{\theta}_M, \dot{d}_L, \dot{\theta}_1, ..., \dot{\theta}_6]^T$, into the *twist* vector $t = [\omega \quad v]^T$, with $\omega = [\omega_x \quad \omega_y \quad \omega_z]^T$ and $v = [v_x \quad v_y \quad v_z]^T$ denoting the angular and linear velocities of the EE's reference frame relative to {B}, respectively. As before, the IKP at the velocity level is useful for path tracking, since obtains the joint velocities corresponding to a given desired twist of the EE. In the case of non-redundant robots, matrix J is square and the solution is given by the inverse of J, i.e. $\dot{q} = J^{-1} \cdot t$. In the case of *redundant* robots, matrix J is non-square J, i.e., it has more columns than rows, and the least-squares solution $\dot{q}$ of equation (1) is given by the *right Moore-Penrose pseudo-inverse* $J^{\dagger} \equiv J^{T}\left(JJ^{T}\right)^{-1}$, i.e., $\dot{q} = J^{\dagger} \cdot t$. Even so, a *homogeneous component* can be added at the cost of giving up the minimum-norm solution:

$$\dot{q} = J^{\dagger} \cdot t \quad + \quad (I_{nxn} - J^{\dagger} J) h \tag{2}$$

from which the manipulator tracks the desired target positions as *primary task* (first term) and could also perform a *secondary task* (second term) by means of the arbitrary vector h which is projected into robot self-motions (chain re-postures but maintaining EE's pose), being $(I - J^{\dagger} J)$ the *projection operator* on the *Null Space* of J, namely $\aleph(J)$. The *performance criterion vector* h can be considered as a *virtual force* that attempts to push the configuration of the manipulator away from a critical area in $\Im$.

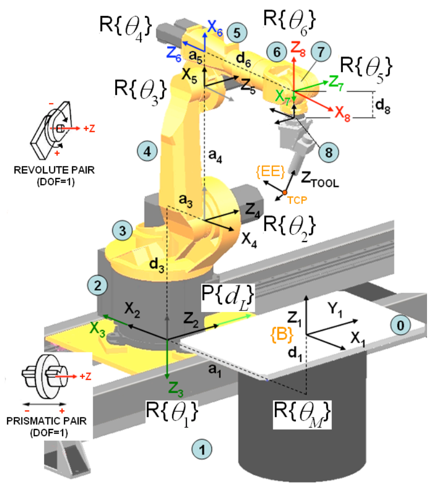

Fig. 2 Denavit-Hartenberg frame assignments for the RP-6R workcell.

3. Robot redundancy

As stated above, the kinematic redundancy degree of the current workcell is three, being one of them a functional redundancy given by the symmetry axis of the milling tool. To capture this functional redundancy, two methods can be used: the *Virtual Joint Method* (VJM), which augments the dimension of $\Im$ by adding a virtual joint on the tool symmetry axis (see Fig. 3) to obtain an extra DOF of redundancy; or the *Twist Decomposition Method* (TDM) [9], which reduces the dimension of the twist vector by eliminating the angular velocity $\omega_{\tau^{\perp}}$ orthogonal to the task subspace (see Fig. 3).

For the VJM equation (2) is rewritten as:

$$\dot{\theta}_v = \begin{bmatrix} [\dot{\theta}_M, \dot{d}_L, \dot{\theta}_1, \ldots, \dot{\theta}_6]^T \\ \dot{\theta}_7 \end{bmatrix} = J_v^{\dagger} \cdot t + (I_{(n+1)x(n+1)} - J_v^{\dagger} J_v) \cdot h \tag{3}$$

where J_v is an *augmented* Jacobian matrix by a virtual joint-rate $\dot{\theta}_7$.

For the TDM equation (2) is rewritten as:

$$\dot{\theta} = \left(J^{\dagger} T\right) \cdot t + J^{\dagger} (I_{6x6} - T) \cdot Jh \, ; \quad T \equiv \begin{bmatrix} (I_{3x3} - ee^T) & 0 \\ 0 & I_{3x3} \end{bmatrix} \tag{4}$$

where matrix T is the *twist projector* and unit vector e denotes the orientation of the tool symmetry axis.

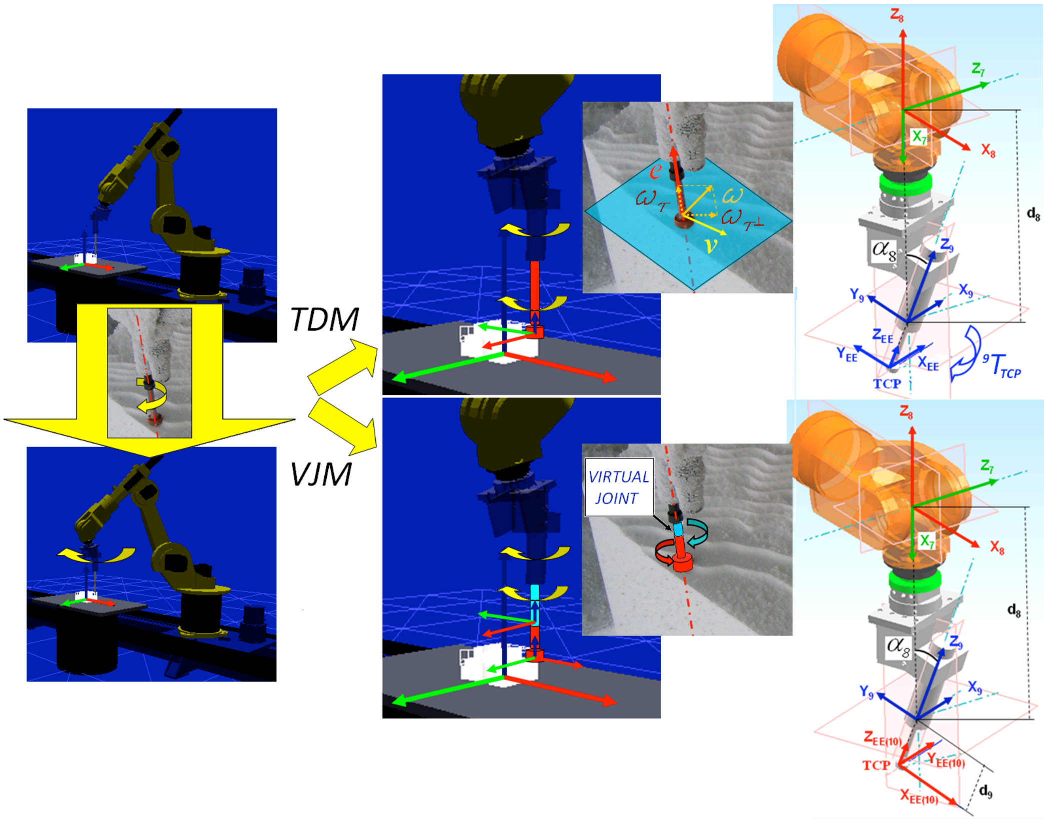

Fig. 3 Two different interpretations of the irrelevant symmetry axis of the milling tool: the VJM and the TDM. Comparison of the DH frame assignments for the VJM (right-down) and the TDM (right-up).

For the workcell at hand, it could be profitable to study the combination of the TDM with a projection on $\aleph(J)$, as below:

$$\dot{\theta} = \left(J^{\dagger} T\right) \cdot t + J^{\dagger} (I_{6x6} - T) \cdot Jh_1 + (I_{nxn} - J^{\dagger} J) \cdot h_2 \tag{5}$$

where h_1 and h_2 are the two feasible *performance criterion vectors* for secondary tasks. To the authors' knowledge, a solution like (5) has not been tested yet. Note that (4) and (2) are a particular case of (5) with $h_2 = 0$ and $T = I_{6x6}$ (i.e., all linear and angular coordinates of the EE are required), respectively.

The most widespread method used to select performance vector h is the *Gradient projection method* (GPM) [10] which minimizes a configuration-dependent scalar, the *performance criterion index p*, by means of its gradient vector:

$$h = -k \cdot \nabla p \;\; ; \;\; \text{with} \;\; \nabla p = \left[\frac{\partial p(q)}{\partial q_1}, \frac{\partial p(q)}{\partial q_2}, \ldots, \frac{\partial p(q)}{\partial q_n} \right]^T \tag{6}$$

where k is an arbitrary constant.

In order to avoid postures with poor *kinematic performance* inside the reachable workspace, i.e., those near to an *internal singularity* [11], the following performance criterion index is considered [12]

$$p_{cond} = \frac{k_F}{2}(q - q_{Ts})^T W_{cond}(q - q_{Ts}) \tag{7}$$

where W_{cond} is the weight diagonal matrix and $k_F(J)$ is the condition number of the *Jacobian* matrix computed using the *Frobenius* norm [8]:

$$k_F(J) = \frac{1}{6}\sqrt{tr(HH^T) \cdot tr[(HH^T)^{-1}]} \;\; ; \quad \text{with} \quad 1 \le k_F < \infty \tag{8}$$

where H is a *homogeneous Jacobian* obtained dividing those elements of J that have units of length by the robot *characteristic length L*, i.e., the normalizing length that renders the condition number of the Jacobian matrix a minimum [13], In particular, the value of L for the KR 15/2 robot, leaving aside the additional joints (θ_M, d_L), is equal to 350.6 mm and the best conditioning achieved was k_F=1.247.

Since it is desirable for the robot to work at any posture minimizing the robot condition number, the index p_{cond} is activated when the value of k_F passes over a preset threshold value ζ. At that instant, the corresponding configuration q_{Ts} is recorded to evaluate the distance to the actual posture at $\Im$:

A certain constant reference arm posture q^{ref} may be desirable for avoiding collision with obstacles or reach the mechanical joint-limits by considering the following performance criterion *index* [9]:

$$p_{jnt} = \frac{1}{2}(q - q^{\mathrm{ref}})^T W_{jnt}(q - q^{\mathrm{ref}}) \tag{9}$$

where W_{jint} is the weight diagonal matrix and the well-known HOME posture $q_0 = [+\pi,\ 0,\ +\pi,\ -\pi/2,\ 0,\ 0,\ +\pi/2,\ 0]^T$ rad (Fig. 2) is taken as the reference posture, i.e., $q^{ref}=q_0$.

Both performance criterions described above could be combined into a unique vector to be applied at VJM (3) in order to simultaneously to maintain the manipulator as close as possible to the q^{ref} posture and as far as possible of bad conditioned postures:

$$h = -\nabla p = -\nabla\left(p_{jnt} + p_{cond}\right) = h_{jnt} + h_{cond} = -\left(W_{jnt}\left(q - q^{\mathrm{ref}}\right) + W_{cond} \cdot k_F \cdot (q - q_{Ts})\right) \tag{10}$$

Similarly, vectors h_1 and h_2 at (5) will be evaluated as the respective sub-tasks of h_{jnt} and h_{cond}.

The choice of the weight matrices W_{cond} and W_{jnt} is a major difficulty when implementing the previous RRS due to subjectivity. The relative importance among the joints in each sub-task is adjusted by tuning both matrices: higher weights are to be assigned to those joints that are supposed to be more reactive when lowering the condition number or being far of the reference posture. This tuning is critical for the performance of the RRS and traditionally it has been made by trial and error and using constant weights [9]. In practice, in case of milling tasks where the tool *pose*, and hence the robot *posture*, changes constantly, it may be desirable to identify an appropriate value for W at each configuration in a reasonable time. In particular, this work proposes to use *Fuzzy Logics* (FL) for designing a *fuzzy inference controller* to assign variable weights at the performance vector h.

4. The proposal

4.1. Software tools

The manufacturing software used in this work is the NX™ of *Siemens*, which integrates the labours of design (CAD), simulation (CAE) and manufacturing (CAM). The CAM module makes possible the planning of milling tasks and interacts with two program codes in TCL (*Tool Command Language*) (which are connected with C++ modules) that manipulate the path data (*Event Handler*) and give the convenient format to the generated output (*Definition File*) [14], see Fig. 4.

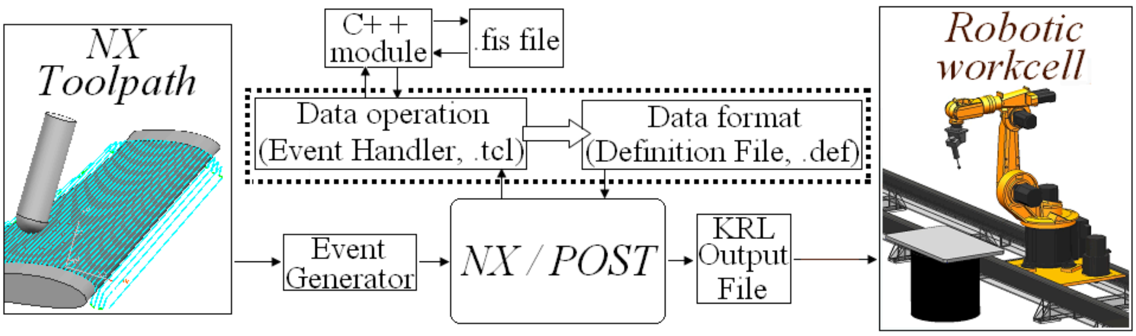

Fig. 4 Post-process flow of the toolpath from NX to KUKA Robot Language (KRL).

All the proposed algorithms have been programmed in *Matlab*® with the aid of the *Hemero* toolbox [15] and the Fuzzy Logic (FL) toolbox [16]. The FL-toolbox generates a *fuzzy inference system* file (.fis) which saves the designed fuzzy inference engine (input/output variables, rules, etc.) and generates a stand-alone fuzzy inference engine in C++. In addition, the software Robomove™ of *Qdesign* [17] was used to display and analyze the robot postures resulting from each implemented RSS.

4.2. Path tracking algorithm

Once the NX-CAM system has generated the trajectory data T_{CAM} as a discrete set of close-enough poses at Ω, the EE of the robot has to track this path. A tangent, normal, and binormal unit vectors can be associated with every sample point of the trajectory, namely the *Frenet-Serret* vectors, indicating the required pose. The joint angles of the robot have to be calculated along this continuous set of poses of the EE. For this purpose, the IKP at the displacement level could be solved at each sampled pose. However, since for redundant manipulators there are infinite solutions, the following alternative iteratively approach, which is based on the robot Jacobian J, is considered [8]:

$$J(q^i)\Delta q^i = \Delta t^i = \begin{bmatrix} Q \cdot vect(Q^T Q_d) \\ \Delta p \end{bmatrix} = \begin{bmatrix} Q \cdot vect(\Delta Q) \\ \Delta p \end{bmatrix} \tag{11}$$

$$\text{with} \qquad vect(M) \equiv \frac{1}{2}\begin{bmatrix} M_{32} - M_{23} \\ M_{13} - M_{31} \\ M_{21} - M_{12} \end{bmatrix} \tag{12}$$

where Q and Q_d represent the current and desired rotation matrices from the base frame to the EE frame and vector Δp is defined as the difference between the prescribed value p_d of the operation point position vector and its current value p. The relationships among the variables $\{Q_d, p_d, Q, p, \Delta Q, \Delta p\}$ can be found in [8].

The following path tracking algorithm with RRS is proposed in this work:

Algorithm 1

1) $q_0 \rightarrow q$ (initial joint position)

for (: each i-point of the trajectory, $\mathrm{T_{CAM}}(i)$)

  2) $\mathrm{T_{CAM}} \Rightarrow \{p_d, Q_d\}$

  while $\|\Delta q\| > \varepsilon$

    3) $\mathrm{DK}(q, \text{DH-Workcell}) \Rightarrow \{p, Q\}$

    4) $Q^T \cdot Q_d \Rightarrow \Delta Q$

    5) $p_d - p \Rightarrow \Delta p$

    6) $\begin{bmatrix} Q \cdot vect(\Delta Q) \\ \Delta p \end{bmatrix} \Rightarrow \Delta t$

    7) $\mathrm{DK}(q, \text{DH-Workcell}) \Rightarrow J_{\text{Workcell}}$

    8) Determination of k_F

      8.1) $\{0, q(4), \ldots, q(8)\} \Rightarrow q_{6R}$

      8.2) $\mathrm{DK}(q_{6R}, \text{DH-KR15/2}) \Rightarrow J_{6R}(q)$

      8.3) $J_{6R} \underset{L}{\rightarrow} H_{6R}$

      8.4) $H_{6R} \rightarrow k_F$

    9) $\mathrm{RRS} \Rightarrow \Delta q$

    10) $q + \Delta q \Rightarrow q$

  end while

end for

where the subindex *Workcell* refers to the complete kinematic chain of the workcell (i.e., including the linear track and the rotary table); the sub-index *6R* refers to the isolated KR15/2 manipulator; and DH-KR15/2 and DH-Workcell refer to the DH models of the KR15/2 manipulator and the complete workcell, respectively.

The above algorithm is customized in the 9th step for each of the two RRS presented in Section 4. Note that the manipulator DH representation depends on the RRS selected. On the one hand, for the VJM (Fig. 3, right-down) an additional row is added in the DH-model (Table 1) due to the additional virtual joint. On the other, the TDM uses the actual DH-model and, therefore, a final constant displacement matrix is required to know the position of the EE (Fig. 3, right-up).

Table 1. Denavit-Hartenberg parameters of the redundant workcell.

Link	α_i (rad)	a_i (mm)	θ_i (rad)	d_i (mm)
1	$\pi/2$	803	θ_M	-305
2	$\pi/2$	0	0	d_L
3	$\pi/2$	300	θ_1	-675
4	0	650	θ_2	0
5	$\pi/2$	155	θ_3	0
6	$\pi/2$	0	θ_4	-600
7	$\pi/2$	0	θ_5	0
8	0.3564	0	θ_6	-443.4
TCP	0	0	$\theta_{7(VJM)}$	-119.7

4.3. Fuzzy weight vector

Two *fuzzy inference engines* have been developed for the adaptive weight assignment of each performance criterion vector, i.e., h_{jnt} and h_{cond}. For this purpose, the expert knowledge is essential. In the workspace region located over the table, the value of k_F is expected to decrease when the robot posture becomes close to the *extended-arm* or *wrist* singularities [18]. In particular, the third and fifth joints (θ_3, θ_5) and the additional joints (θ_M, d_L) have great importance to avoid theses ill-conditioned configurations. Additionally, it is convenient that the joints doing the *gross positioning* ($\theta_1, \theta_2, \theta_3$) work near the reference posture q^{ref} while the *fine orientation* ($\theta_4, \theta_5, \theta_6$) is being done [4], so it makes sense to use different weight assignments for both groups of joints. Based on this reasoning, the output variables of both fuzzy inference engines are those weights associated to the joints mentioned above, while a default value is given to the remaining joints, see

Table 2.

Table 2. Diagonal elements of the *fuzzyfied* weighting matrices: some elements have a fixed value while others are dynamically assigned by the fuzzy inference engine.

	θ_M	d_L	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	$\theta_{7(VJM)}$
W_{jnt}	w_{Mjnt}	w_{Ljnt}	0.01	0.01	w_{3jnt}	0.01	w_{5jnt}	0.01	0.01
W_{cond}	w_{Mcond}	w_{Lcond}	w_{gross}	w_{gross}	w_{gross}	w_{fine}	w_{fine}	w_{fine}	0.01

After studying which robot joints have greater impact on k_F and on the maintenance of q^{ref}, the input variables are structured. In particular, two inputs (θ_3, θ_5) are used for the singularity avoidance and three inputs ($\theta_2, \theta_3, \theta_5$) are used for the maintenance of the reference posture. For these input spaces, a number of clusters (i.e., linguistic etiquettes) have to be assigned according to the experience. In this work, three triangular clusters are considered, see Fig. 5. Note that the functions neither are equidistant nor have identical form, as they are tuned according to experience. Analogously, the output spaces are different for each fuzzy inference system, depending also on the experience.

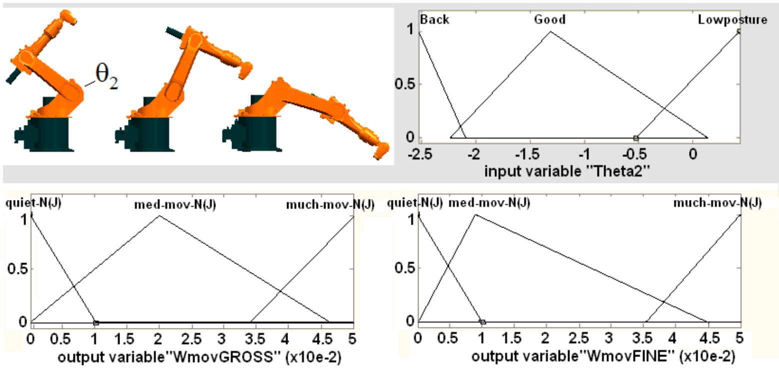

Fig. 5 Up, peak postures of the three corresponding clusters in which the input space θ_2 is divided. Down, output space representation for the assignment of w_{gross} and w_{fine}.

Finally, a few "*if-then*" rules relating both input and output spaces have also to be defined according to the experience. Considering too much rules can be cumbersome and moves away from the desired simplicity of a fuzzy inference system. In this work, two and four rules were created for the singularity avoidance and reference posture criteria, respectively, see Fig. 6.

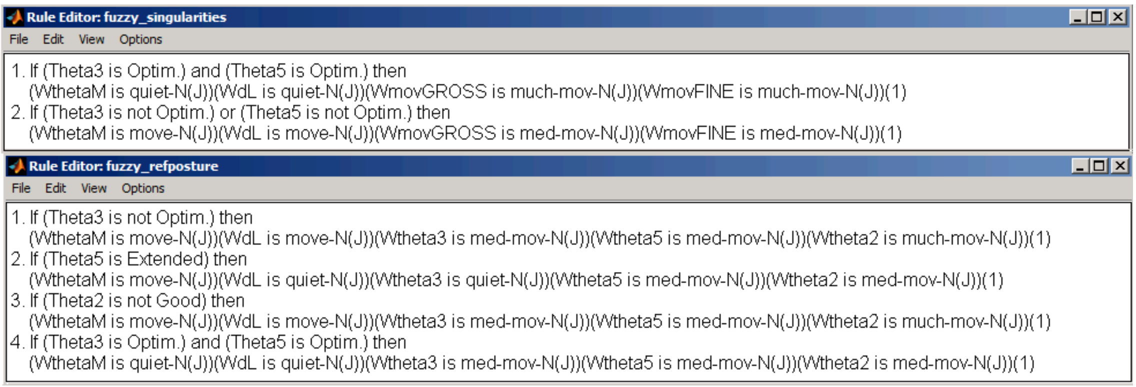

Fig. 6 Rule-base for the singularity avoidance criterion (up) and for the posture criterion (down).

4.4. Improvement of the robot posture

Common milling tasks are made of a sequence of short paths. In this work, the breaks between them are used to revise and improve the posture for the following path. Thus, a periodic posture revision is done in order to obtain a better value for k_F. To practical effects, the non-linear analysis of the KR15/2 posture is done at a set of points using the IKP procedure described in [4] with the current value of the additional joints known. Then, the additional joints θ_M and d_L are moved recursively to improve the value of k_F while maintaining the cutter pose, see Fig. 7. Attending to the manipulator precision, the major improvement is aimed at the table rotation θ_M. After that, a small track displacement d_L is required. Once the robot posture re-adjustment has been finished, the RRS performs the postprocessing for the next period.

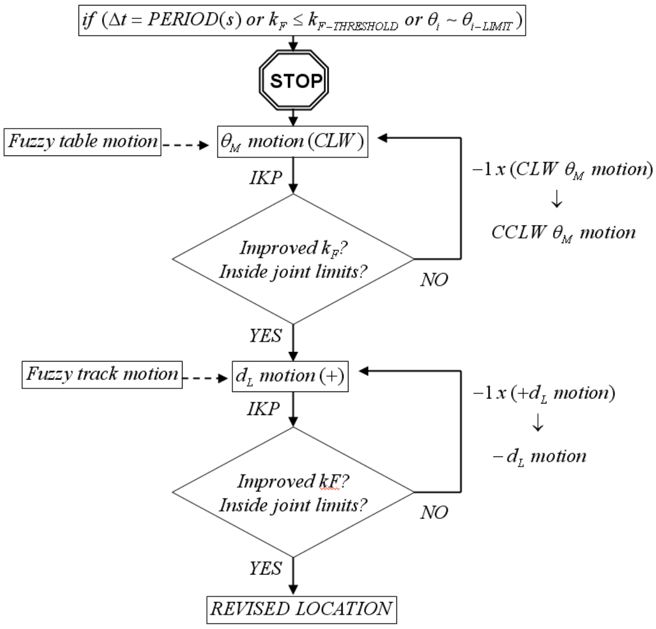

Fig. 7 Periodic revision to improve the robot posture.

5. Practical results

5.1. Postprocessor performance

The postprocessor has been performed with a challenging 5-axis milling that consists of a spherical shape (Fig. 8) to be milled through a continuous spiral path, aiming for the robot posture relocation with the additional joints meanwhile. The orientation of the tool is required to point constantly the center of the sphere, which is located on the rotary table. In particular, the center of the sphere is located at the coordinates C={100, 200, 250} mm in base frame {B} and its radius is equal to 150 mm.

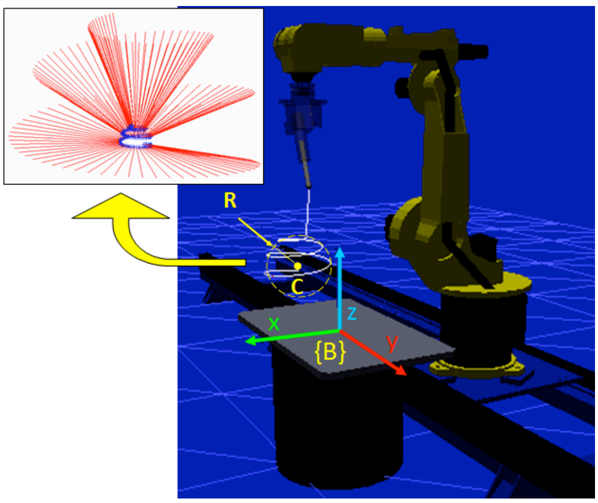

Fig. 8 Workcell at the HOME posture q_0 and representation of the spherical toolpath to be followed. The tool axis, which is represented by the red lines, is maintained perpendicular to the spherical surface.

For the first trial two constant diagonal weighting matrices are considered: $W_{jnt} = W_{cond} \equiv diag(0.01)$. A faster reaction can be expected in the second trial with $W_{jnt} = W_{cond} \equiv diag(0.1)$. Finally, a trial using the fuzzyfied weighting matrices is performed with a threshold value ζ of 0.5 to activate (8). The resulting values of the inverse of k_F during the path tracking for each RRS derived from (5) (VJM, TDM, and TDM with a projection on $\aleph(J)$) is shown in Fig. 9. Clearly, the VJM maintains a more optimum and stable k_F during the path tracking than the TDM implementations, which are more sensible with respect to the weight vectors. As shown in Fig. 10, the joint variable θ_3 reaches its mechanical limit for the VJM with $W_{jnt} = W_{cond} \equiv diag(0.1)$, so the VJM with $W_{jnt} = W_{cond} \equiv diag(0.01)$ seems to be more convenient.

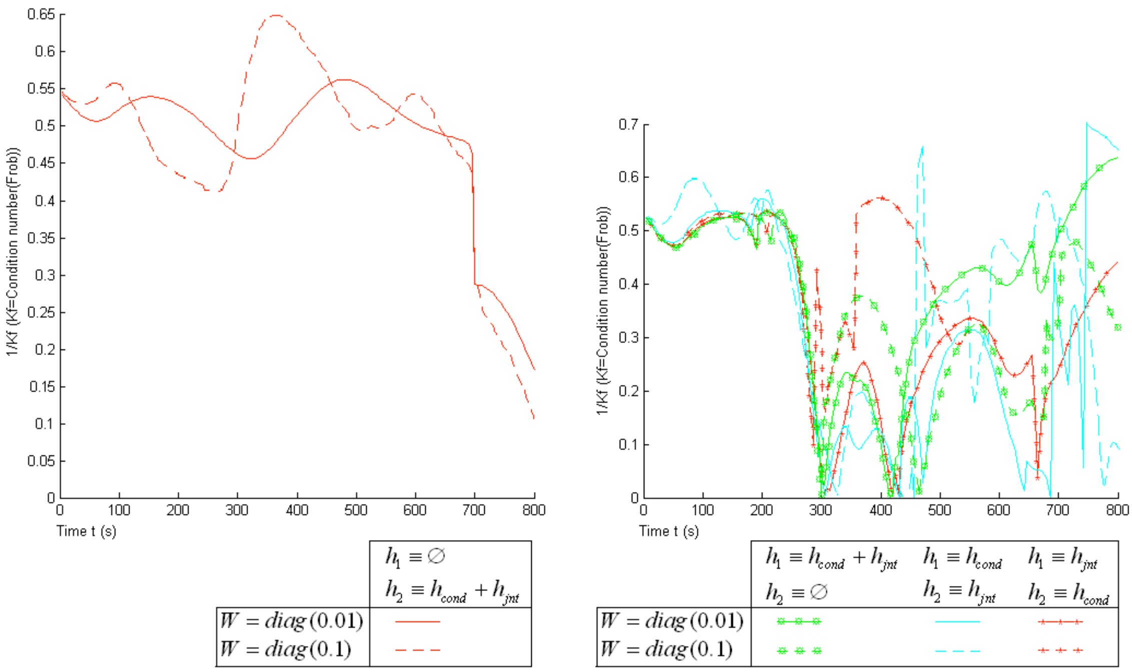

Fig. 9 Evolution of the inverse of the condition number k_F during the spherical path tracking for the VJM (left) and TDM (right).

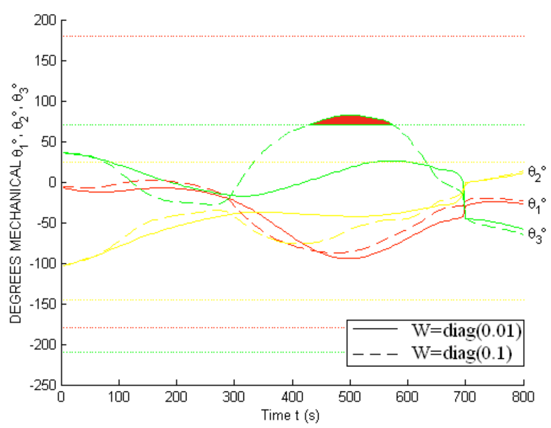

Fig. 10 Behaviour of the gross positioning joints ($\theta_1, \theta_2, \theta_3$) during the spherical path tracking with the VJM.

For the next trial, both weighting matrix W_{jnt} and W_{cond} were tuned with the designed *fuzzy inference controllers*. As shown in Fig. 11, the VJM again has the best and more stable value of $1/k_F$.

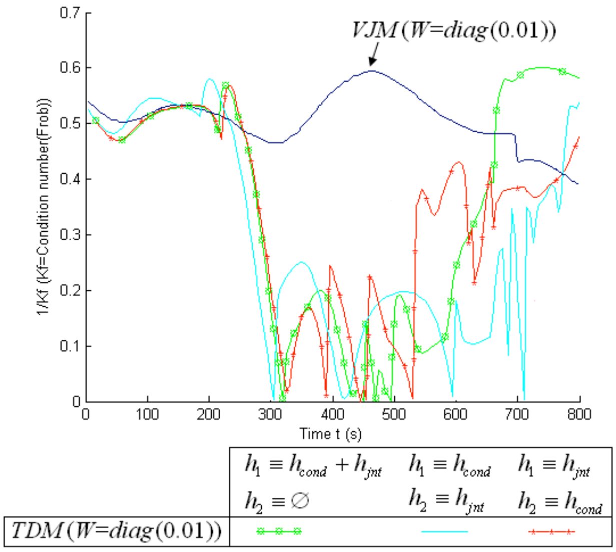

Fig. 11 Evolution of $1/k_F$ for the RRS with the fuzzy adapted weighting vector of Table 2.

Fig. 12 summarizes the conditioning achieved in the tests above with the VJM, which was more robust than the TDM. Note that the use of the fuzzy adapted weighting vector clearly improves the value of $1/k_F$ compared to that obtained with the use of a constant weighting vector. Note also that using the periodic posture improvement (with a period of 100 seconds) the value of $1/k_F$ is significantly improved, especially at the end of the test. It is worth mentioning that, although the worst conditioning value achieved with and without the periodic posture improvement is very similar ($k_F \approx 0.4$), the worst conditioned robot posture achieved with the continuous VJM is more unpleasant (Fig. 13, down) than the one achieved with the periodic posture improvement (Fig. 13, up).

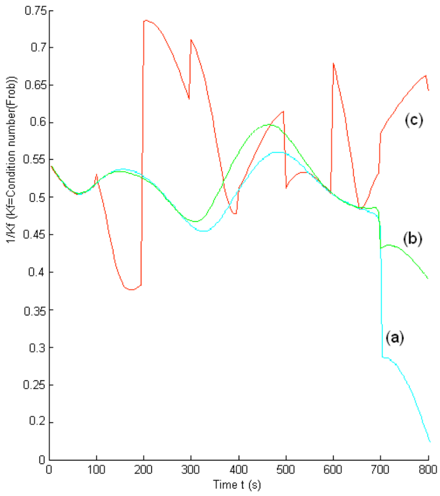

Fig. 12 Comparison of the conditioning achieved using the VJM: (a) with a constant weighting vector; (b) with a fuzzy adapted weighting vector; (c) with a fuzzy adapted weighting vector and a periodic posture improvement.

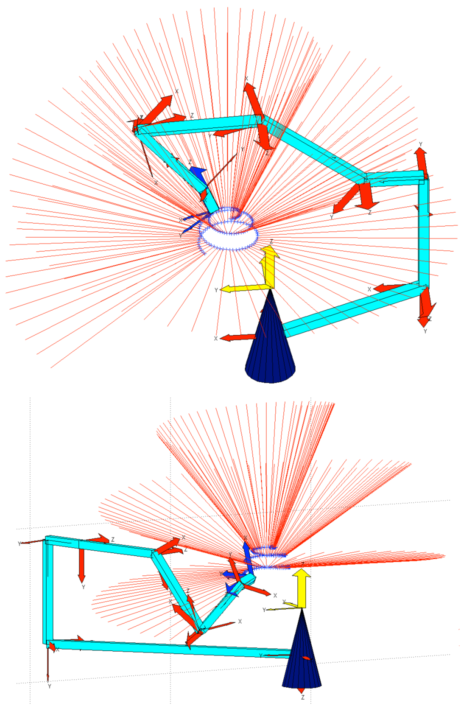

Fig. 13 Worst conditioned postures for the continuous VJM (down) and the VJM with the periodic posture improvement (up).

5.2. Real prototyping of a windmill blade mold

In order to validate the proposed postprocessor, the redundant workcell is devoted to machine a windmill blade mold with expanded polystyrene (EPS). A 5-axes milling operation was planned in order to achieve the specific NACA (*National Advisory Committee for Aeronautics*) profile of this piece. These profiles are airfoil parameterized shapes for aircraft wings developed by the NACA. They allow generating precise cross-sections of the blade and calculate its properties in a CAD/CAE system such as NX^{TM}. Fig. 14 shows the sample toolpath, in which the tool orientation is normal to the surface along the tracking.

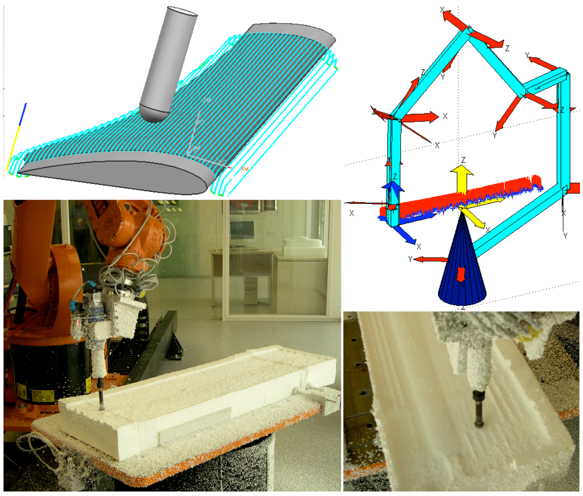

Fig. 14 A windmill blade mold is machined with a 5-axes toolpath to test the postprocessor.

This machining task has been carried out *without* and *with* the proposed postprocessor, starting from the same HOME posture. In the first case, the robot could not reach all the required postures and so the milling task was not possible (Fig. 15, left). In the second case, the robot with the proposed fuzzy postprocessor was able to complete the milling task (Fig. 15, right). In particular, both fuzzy postprocessors without and with the periodic revision of the robot posture were able to mill the mold, but the periodic revision guarantees a higher average of the inverse of the condition number, see Fig. 16. In order to achieve and maintain a better conditioned posture, in both cases all the joints are moved between the allowable limits, see Fig. 20 and Fig. 21. Nevertheless, note that all joints, and particularly the external joints (linear track and rotary table), are widely used with the periodic revision.

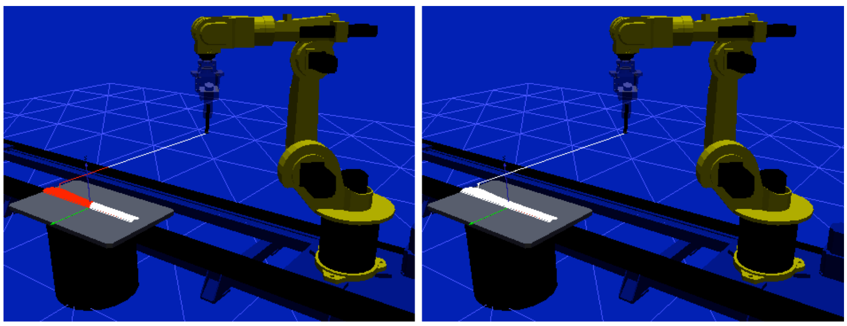

Fig. 15 The white toolpath, representing the area being reachable during the machining process, is enhanced with the programmed algorithm (right), if compared with a fixed table and track (left).

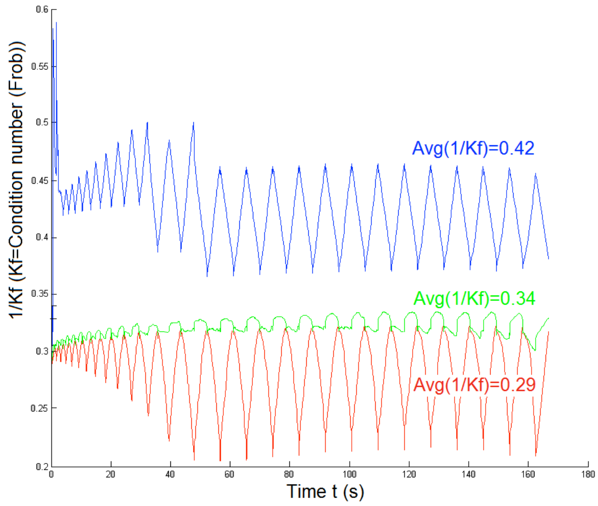

Fig. 16 Evolution of the conditioning of the manipulator while the milling of the windmill blade mold.

6. Conclusions

This work has developed a postprocessor that translates the information generated by a NX-CAM system to the KUKA controller of a redundant workcell devoted to milling tasks. For this purpose, several RRS at the velocity level were tested and optimized. In this sense, an expert *fuzzy systems* has demonstrated to be a good alternative to manage some weight parameters related with experience, opposed to those previously fixed. In particular, a fuzzy inference engine improved the adjustment of the weights of the *performance vectors* for every robot configuration, boosting the performance of the RRS.

The implemented postprocessor has been effectively tested in a graphical simulation of a demanding machining and it has also been successfully validated in a real prototyping of a windmill blade mold with expanded polystyrene by means of 5-axes milling operations.

With the same guidelines, the postprocessor programmed inside the CAM system is expected to be easily applicable not only to other industrial robots, but also for different applications such as welding or painting labors.

Acknowledgments

This research is partially supported by the Technical University of Valencia (PAID-00–09), project PROMETEO 2009/063 of Generalitat Valenciana and research project DPI2009-14744-C03-01 of the Spanish Government.

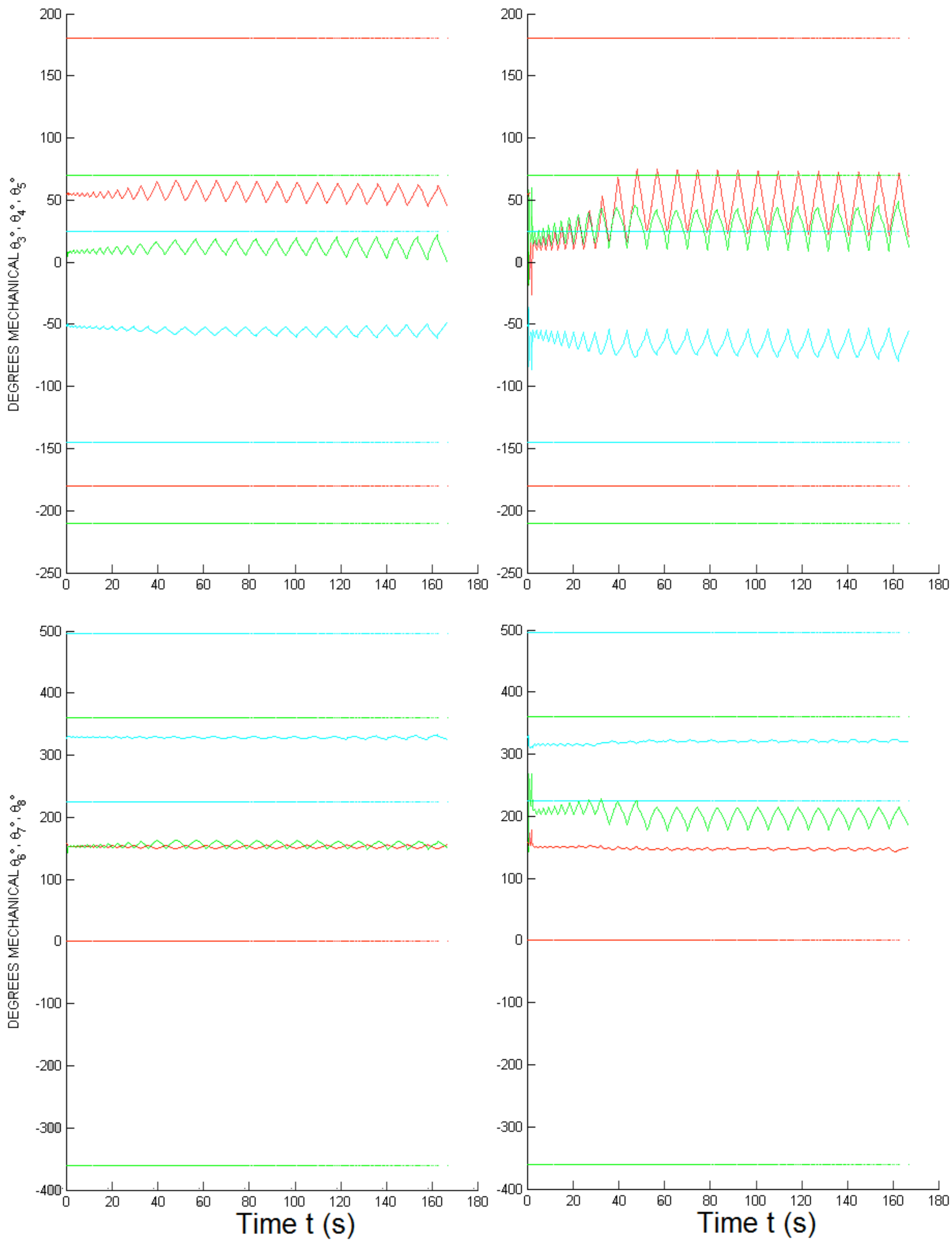

Fig. 17. Joint values at *gross* (up) and *fine* positioning (down) for the postprocessor *without* (left) and *with* (right) periodic revision of the robot posture.

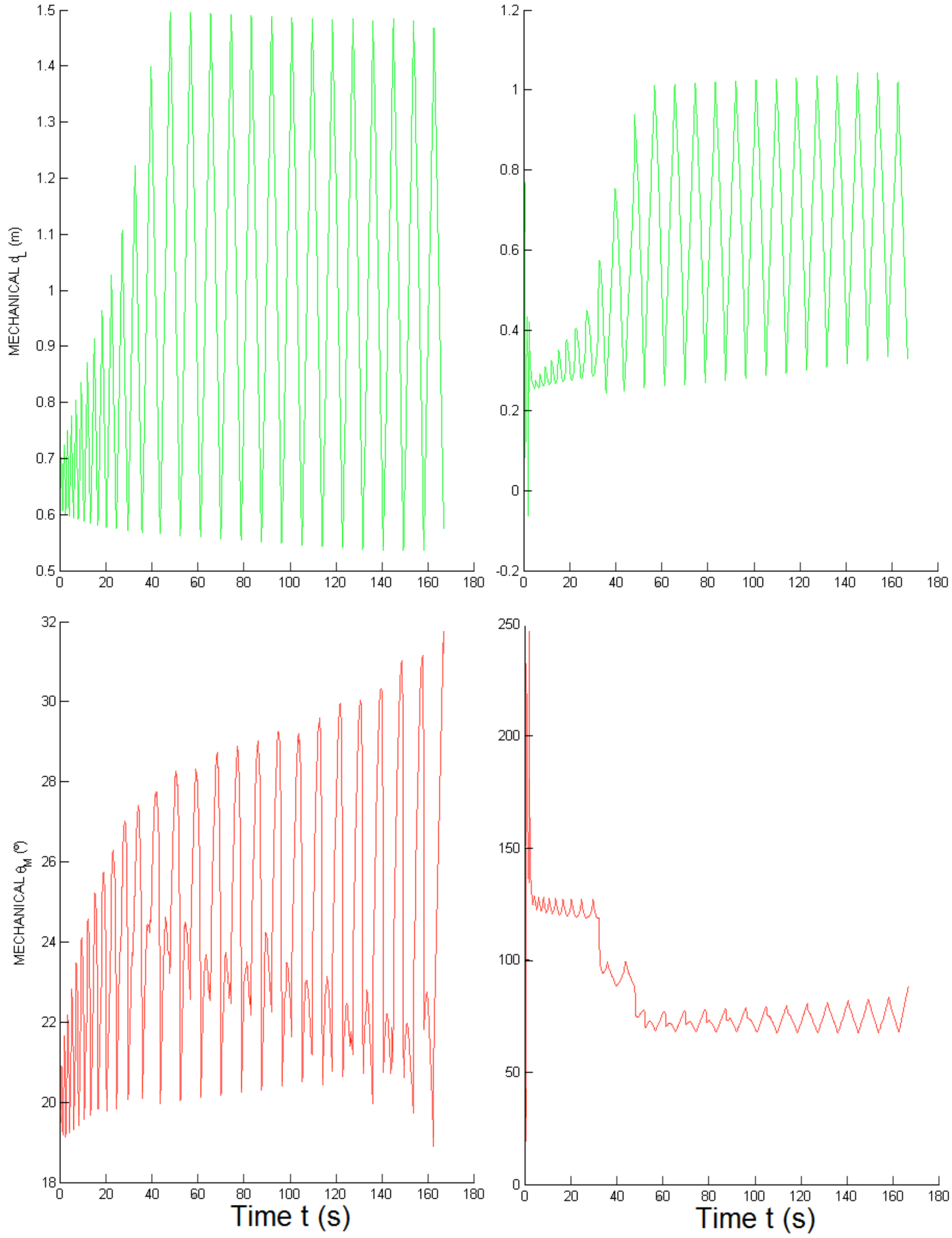

Fig. 18. Values of the external joints (up track, down table) for the postprocessor *without* (left) and *with* (right) periodic revision of the robot posture.

References

[1] Andres J., Gracia L., Tornero J., Calibration and control of a redundant robotic workcell for milling tasks, Int. Journal of Computer Integrated Manufacturing, vol. 24, no. 6, pp. 561–573, 2011.

[2] Patel R.V. and Shadpey F., Control of Redundant Robot Manipulators: Theory and Experiments, Springer, New York, 2005.

[3] Hartenberg R.S. and Denavit J., A kinematic notation for lower pair mechanisms based on matrices, Transactions of the ASME Journal of Applied Mechanics, vol. 77, pp. 215–221, 1955.

[4] Andres J., Gracia L., Tornero J.; Inverse kinematics of a redundant manipulator for CAM integration. An industrial perspective of implementation, Proceedings of the 2009 IEEE International Conference on Mechatronics, Malaga, 2009

[5] Whitney D.E., Resolved motion rate control of manipulators and human prostheses, IEEE Trans. Man-Machine Syst., vol. 10, no. 2, pp. 47-53, 1969.

[6] Sciavicco L. and Siciliano B., Modelling and Control of Robot Manipulators, Springer, London, 2000, pp. 79-87.

[7] Whitney D.E., The mathematics of coordinated control of prosthetic arms and manipulator, ASME J. Dynamics Systems, Measurement and Control, vol. 94, no. 4, pp. 303-309, 1972.

[8] Angeles J., Fundamentals of Robotic Mechanical Systems: Theory, Methods and Algorithms, Springer, New York, 2003.

[9] Huo L. and Baron L., The joint-limits and singularity avoidance in robotic welding, Industrial Robot, vol. 35, no. 5, pp 456-464, 2008.

[10] Liégeois A., Automatic supervisory control of the configuration and behavior of multibody mechanisms, IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-7, pp.245-250, 1977.

[11] Gracia L., Andres J., Tornero J., Trajectory tracking with a 6R serial industrial robot with ordinary and non-ordinary singularities, International Journal of Control, Automation, and Systems, vol. 7, no. 1, pp. 85-96, 2009.

[12] Angeles J., López-Cajún C. S., Kinematic isotropy and the conditioning index of serial robotic manipulators, The International Journal of Robotics Research, vol. 11, no. 6, pp. 560-570, 1992.

[13] Khan Waseem A., Angeles J., The kinetostatic optimization of robotic manipulators: the inverse and the direct problems, Journal of Mechanical Design, vol. 128, no. 1, pp. 168-178, 2006.

[14] Siemens Corp, 2009. NX Documentation. In: {$UGII_base_dir}\UGDOC.

[15] Maza J.I., Ollero A., Herramienta MATLAB-Simulink para la simulación y el control de robots manipuladores y móviles, Actas de las XXI Jornadas de Automática, Sevilla, Spain, 2000.

[16] Roger Jang J.S., Gulley N., Fuzzy logic toolbox: user's guide; revised for version 2.2.7. The Math Works, Inc.; 2008.

[17] Qdesign, S.R.L., 2007. CAD-CAM off line programming for industrial robots. ROBOmove on line help v. 2.0 [online]. Available from: http://www.qdrobotics.com/eng/robomove.php [Accessed 28 November 2011].

[18] Andrés J., Gracia L., Marti H., Tornero J., Toolpath postprocessing for three axes milling in redudant robotic workcells by means of fuzzy integration in a cam platform, Proceedings of the IEEE International Conference on Mechatronics, Malaga, Spain, 2009.