Kinematics Programming for Cooperating Robotic Systems

Cristiane P. Tonetto, Carlos R. Rocha, Henrique Simas, Altamir Dias

Kinematics Programming for Cooperating Robotic Systems

Cristiane P. Tonetto[1], Carlos R. Rocha[1], Henrique Simas[1], Altamir Dias[1]

[1]Federal University of Santa Catarina, Mechanical Engineering Department, P.O. Box 476, University Campus, Trindade, Florianópolis-SC, Brazil, 88040-900
{cris.tonetto, simas, altamir}@emc.ufsc.br,
carlos.rocha@riogrande.ifrs.edu.br

Abstract. This paper presents the kinematics programming for Cooperative Robotic Systems (CRS), based on screw theory approach. It includes a systematic for modeling and programming robotic systems composed by any number of robots (not necessarily identical), working cooperatively to perform different tasks. In order to illustrate the application of the systematic, an example of CRS including four robots is presented. The kinematic computation of a CRS is made through the screw theory approach and its tools, like the Davies method and Assur virtual chains.

Keywords: Cooperative Robotic Systems, Task Programming, Kinematics, Modeling Systematics.

1 Introduction

Some industrial tasks need to use more than one robot to perform some tasks, since one single robot may not complete them alone. So, a Cooperative Robotic System (CRS, in short) can be applied to a task or set of tasks that present some complexities to be performed by a single robot. Some examples of cooperative robots tasks include load sharing, assembling of parts and part reorientation while having another operation over it. Since a CRS is composed by more than one robot, it is necessary to deal with many variables and their interrelationships for the robots programming.

Many researchers have been studying the application of more than one robot for task execution. Lewis [1] introduces the relative Jacobian concept based on the Denavit-Hartenberg convention for two robots. In Tzafestas [2] the system is composed by three identical robots that need to move an object from one position to another. Dourado [3] studies the differential inverse kinematics based on the screw theory for CRS. Owen, Croft and Benhabib [4] applied the relative Jacobian developed by Lewis [1] in a system composed by two planar robots. The cooperative Jacobian concept is introduced by Ribeiro and Martins [5], developed using the Denavit-Hartenberg convention or the screw theory associated with the Davies method. Also, Ribeiro and Martins [5] proposes the cooperative Jacobian for systems composed by an arbitrary number of robots for tasks execution. Some specific case solutions for CRS are found in the literature, but no systematical and general approaches are applied for the kinematics computation and CRS programming.

In this paper, an approach for the CRS programming is proposed. It also presents a general view of the programming process (which is detailed in [6], [7] and [8]). This novel systematical approach intends to ease the programming process for cooperating robots while performing industrial tasks.

This paper is divided in five sections. In Section 2 the main contribution of this work is highlighted. In section 3 it is presented a brief survey of the basic theory for the kinematics computation. In section 4 the CRS concept is analyzed, describing the systematic applied to cooperative robotic. Finally, conclusions are summarized.

2 Contribution to Value Creation

Some questions appeared in the modeling and computing kinematics process of CRS. In the first place, how to generalize the addition of an arbitrary number of robots in the system. Secondly, how different physical structures influence the programming tasks and other variables in the system, and, also, how to combine different number and types of joints in order to perform one or more tasks. Third, since the main purpose of the robot programming process is to have all tasks completed, how to include them in the systematic the task specification(s).

To program a CRS it is needed to know two main components: robots and tasks [8]. Those set of data are better divided into three main environments: the robotic structure, the task environment and the differential kinematics. These environments make possible to split and evaluate robots data and tasks separately, allowing to know in advance how the programming changes when the number of robots and tasks increase in the system's configuration.

Let's take a basic structure for compose a CRS, as a function of the number of robots, and associated to a way of adding the Assur virtual kinematics chains. It gives an initial knowledge over the complexity growth of the kinematics chain resolution involved in the CRS composition. Such systematic is expansible and allows to solve general problems of CRS, like collision avoidance in the workspace of robots and tasks as well as to simulate relative displacement of the robots bases, when necessary in the system.

The strategy to solve the CRS programming allows to simulate CRS composed by any number of robots and tasks. It gives flexibility to the system and allows parameters changes, such as the initial configuration, robots positioning, which robot will perform a task, among others.

All CRS programming logic and strategies proposed in this paper are based on the screw theory, graph theory, the Davies method and the Assur virtual chains. These tools are briefly described in the next section.

3 Base Theory for the CRS's Programming

3.1 Screw Theory

The methodology presented for kinematics computation of CRS is based on the screw theory. In this theory, a screw \$ is a geometric element defined by a directed line in space and by a scalar parameter *h*, that defines the screw pitch [9]. One screw can be decomposed in a magnitude $\dot{q}$ and its normalized axis $\hat{\$}$:

$$\$ = \hat{\$}\dot{q} \quad \text{where } \hat{\$} = \begin{bmatrix} s_i \\ s_{oi} \times s_i + h s_i \end{bmatrix} \tag{1}$$

and also, s_i is an unitary vector with the direction of the axis related to the translation and rotation of the screw displacement. The s_{oi} vector defines the position of the s_i vector related to a fixed coordinate system, *h* is the screw pitch and $s_{oi} \times s_i$ is the cross product of s_{oi} and s_i vectors.

Now, the screw can be adapted to each type of body motion. When the movement is a rotation, the screw step is null (h=0), and Equation (1) leads to $\hat{\$}_{\text{revolute}}$. Moreover, when the movement is a translation, Equation (1) leads to $\hat{\$}_{\text{translation}}$.

$$\hat{\$}_{revolute} = \begin{bmatrix} s_i \\ s_{oi} \times s_i \end{bmatrix}; \quad \hat{\$}_{translation} = \begin{bmatrix} 0 \\ s_i \end{bmatrix} \tag{2}$$

So, the screw movement description may be used to define the differential displacement between two bodies related to a reference coordinate system (based on Chasles theorem and on Mozzi theorem). More details of the screw theory and its applications can be found in the following works: [9], [10] and [11].

3.2 Successive Screw Method

This method can be extended to compute the screw due to the action of a body in another and as well as its coupling among them. The complete approach is presented in the research of Tsai [12], where he deduces the equations that model the rigid body displacements by using the Chasles Theorem. Also, Tsai [12] presents the Rodrigues matrix, written to evaluate the displacement of a rigid body in the space. So, the Rodrigues matrix is given by:

$$A_{\vec{s},\theta} = \begin{bmatrix} (s_x^2 - 1)(1 - \cos\theta) + 1 & s_x s_y(1 - \cos\theta) - s_z \sin\theta & \cdots \\ s_x s_y(1 - \cos\theta) + s_z \sin\theta & (s_y^2 - 1)(1 - \cos\theta) + 1 & \cdots \\ s_x s_y(1 - \cos\theta) - s_y \sin\theta & s_y s_z(1 - \cos\theta) - s_x \sin\theta & \cdots \\ 0 & 0 & \cdots \end{bmatrix}$$

$$\begin{bmatrix} \cdots & s_x s_z(1 - \cos\theta) + s_y \sin\theta & ts_x - s_{o_x}(a_{11} - 1) - s_{o_y} a_{12} - s_{o_z} a_{13} \\ \cdots & s_y s_y(1 - \cos\theta) - s_x \sin\theta & ts_y - s_{o_x} a_{21} - s_{o_y}(a_{22} - 1) - s_{o_z} a_{23} \\ \cdots & (s_x^2 - 1)(1 - \cos\theta) + 1 & ts_z - s_{o_x} a_{31} - s_{o_y} a_{32} - s_{o_z}(a_{33} - 1) \\ \cdots & 0 & 1 \end{bmatrix} \quad (3)$$

By using the screw displacements concept for the kinematics analysis, the screw displacement of *n* successive screw displacements is just the premultiplication of the transformation matrix, given by:

$$A_r = A_1 A_2 A_3 \ldots A_n \quad (4)$$

In this way, the position of the manipulator's end-effector can be computed, while the robot joints move, by using the screws evaluation procedures. In other words, the successive screw technique can be extended over the axis of the joints in order to calculate the robot effector pose. For the CRS, the screw technique is employed to establish the influence of each one of the joints over the others, and, thus, to get the *s* and s_o variables that includes such influence.

3.3 Assur Virtual Chains and Graph Theory

Another tool to study CRS is the Assur virtual chains. The Assur virtual chains, when added to a CRS, help to analyze the displacements of a kinematic chain or even to impose desired movement to a kinematic chain, as described before. By definition, a virtual chain is a kinematic chain composed by virtual links and joints, that can be added to a real kinematic chain without changing the main behavior of a real chain [13]. The virtual chain should be used to describe the relationship among robots links, tasks and parts in the scenario of CRS's planning.

Also, a CRS can be better represented by using graph theory, to visualize and compute interrelations between base, robots and tasks. Such relationships, represented by graphs, can be summarized in a circuit matrix B. This matrix B is a way to describe the presence of each edge in the graph meshes. The matrix is assembled so that each row is reserved to one mesh of the graph and each column to one edge. Each element of the matrix is defined as:

- 0, if the edge is not present on the mesh;
- 1, if the edge is on the mesh and on the same direction of the circuit (arbitrary chosen, but constant);
- -1, if the edge is on the mesh, but in the opposite direction of the circuit.

Together, Assur virtual chains and graph theory are very helpful to build the interrelationships in a CRS study.

3.4 Davies Method

The Davies method is a way to compute and relate the joints' velocities magnitudes of a close virtual chain. The method is an adaptation of the Kirchhoff circuit law and states that the sum of the relative speeds between kinematic pairs throughout any closed kinematic chain is null [10]:

$$\sum_{i=0}^{n} \$_i = \sum_{i=0}^{n} \hat{\$}_i \dot{q}_i = 0 \Leftrightarrow N\dot{q} = 0 \tag{5}$$

in which n is the number of joints of the system and N is the network matrix containing the normalized screws.

To study the kinematic chain behavior it is better to classify the system' joints as primary N_p and secondary N_s. Thus, the Equation 5 can be written as:

$$\left[N_s \vdots N_p\right] \cdot \begin{bmatrix} \dot{q}_s \\ \cdots \\ \dot{q}_p \end{bmatrix} = 0 \Leftrightarrow N_s \dot{q}_s = -N_p \dot{q}_p \tag{6}$$

If N_s is invertible, the magnitude of the secondary joints $\dot{q}_s$ can be computed by the following equation:

$$\dot{q}_s = -N_s^{-1} N_p \dot{q}_p \tag{7}$$

By substituting N_s, N_p and $\dot{q}_s$ on Equation 7, the secondary joints velocities magnitudes turn to be described as a function of the primary joints.

4 Representation of the CRS

The CRS composition is based on several factors related to the tasks, and those associated to the available robots to perform them. Also, other important factor that has to be considered is the environment in which the robots are located, like their physical arrangement around the tasks to be performed and the global workspace where the task will be performed.

In this section a cooperative task example will be presented, in which the CRS is composed by four robots and one composed task. The CRS task programming has fixed and variable data inputs.

So, the CRS system is composed by the robots: ABB IRB 6620, ABB IRB 1600, ABB IRB 140 and ABB IRB 120. Their data are presented on the Tables 1 and 2. On these tables the values of s and s_o for each joint are described. This information is used as input for the robots kinematics programming computation. It is important to notice that the first joint of the ABB IRB 6620 robot is a prismatic joint.

The task will be performed by two positioning robots, one inspection robot, and the last robot must follow a trajectory designed to be painted (write a word - the UFSC

acronym) over the part.

Table 1. Robots data: ABB IRB 6620 and IRB 1600

	IRB 6620		IRB 1600	
Joints	s	s_o (mm)	s	s_o (mm)
1	(0,1,0)	(0, 0, 0)	(0,0,1)	(0, 0, 0)
2	(0,1,0)	(0, 0, 416)	(0,1,0)	(150, 0, 486.5)
3	(0,1,0)	(0, 0, 1371)	(0,1,0)	(150, 0, 961.5)
4	(1,0,0)	(245, 0, 1565)	(1,0,0)	(450, 0, 961.5)
5	(0,1,0)	(880, 0, 1565)	(0,1,0)	(750, 0, 961.5)
6	(1,0,0)	(1076, 0, 1565)	(1,0,0)	(750, 0, 961.5)

Table 2. Robots data: ABB IRB 140 and IRB 120

	IRB 140		IRB 120	
Joints	s	s_o (mm)	s	s_o (mm)
1	(0,0,1)	(0, 0, 0)	(0,0,1)	(0, 0, 0)
2	(0,1,0)	(70, 0, 352)	(0,1,0)	(0, 0, 292)
3	(0,1,0)	(70, 0, 712)	(0,1,0)	(0, 0, 632)
4	(1,0,0)	(260, 0, 712)	(1,0,0)	(156, 0, 632)
5	(0,1,0)	(450, 0, 712)	(0,1,0)	(360, 0, 632)
6	(1,0,0)	(450, 0, 712)	(1,0,0)	(374, 0, 632)

In the Fig. 1 is depicted the configuration of the system on the left and the graph representation with virtual chains addition (CV_i symbol) on the right. In this example system, five PPPS type of virtual chains were added. These virtual chains are spatial and the name PPPS means that they are composed by three prismatic and one spherical joints. The CV_0 virtual chain defines the part movement, related to the fixed coordinate system, and the CV_i, with i=1, 2, 3, 4, describe the movement of the end-effector related to the part.

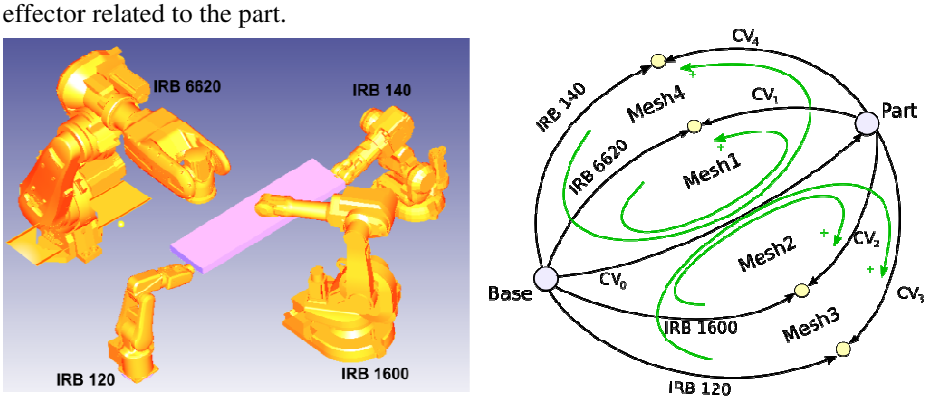

Fig. 1. Cooperative Robotics System and its graph representation

The desired part movement given by CV_0 virtual chain has a trajectory defined by

following values: CV_0=(1200, -530 - 3.75t, 770, 0, 0, 0) where 0<t<40.8s.

The CV_1 virtual chain describes the movement of the IRB 6620 robot end-effector, relative to the part. This movement is defined as the set of points, joined into a trajectory, planned so that it can be used to paint the UFSC acronym over the part.

The CV_2 is related to the inspection procedure, and the IRB 1600 robot is designated for this task. The inspection is given by the following trajectory: CV_2=(100, 500+15t, 200, 0, 0, 0) with 0<t<40.8s, relative to the part's origin.

The CV_3 and CV_4 virtual chains define the movement of the IRB 120 and IRB 140 end-effectors. Since these robots are positioning robots, the end-effector is fixed on the part, thus leading to null trajectories for CV_3 and CV_4.

On Table 3 the positions of the bases relative to the fixed coordinate system and the joints initial configuration for each robot are shown.

Table 3. Robots base position and joints initial configuration

Robots	Base position (mm)	Joints initial configuration (rad)
IRB 6620	(0,-380,0)	(0, 0.52, -0.52, 0, 1.57, 0)
IRB 1600	(2200, 0, 400)	(0.6 , 0, 3.14, 0, -0.52, 0)
IRB 140	(1400, 1150, 200)	(-1,52, 0.26, 0, 0, -0.26, 0)
IRB 120	(1400, -1000, 250)	(1.52, 0.26, 0, 0, -0.26, 0)

The circuit matrix *B* is:

$$B = \begin{array}{c} \\ mesh1 \\ mesh2 \\ mesh3 \\ mesh4 \end{array} \begin{array}{c} \begin{array}{ccccccccc} CV_0 & CV_1 & R_1 & CV_2 & R_2 & CV_3 & R_3 & CV_4 & R_4 \end{array} \\ \begin{bmatrix} 1 & 1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 \end{bmatrix}_{4\times 54} \end{array} \quad (8)$$

The matrix columns are presented in compressed form. For example, the CV_0 column refers to the six joints of the PPPS virtual chain.

The normalized screw matrix *D* is:

$$D = \left[\hat{\$}_{CV_0} \quad \hat{\$}_{CV_1} \quad \hat{\$}_{R_1} \quad \hat{\$}_{CV_2} \quad \hat{\$}_{R_2} \quad \hat{\$}_{CV_3} \quad \hat{\$}_{R_3} \quad \hat{\$}_4 \quad \hat{\$}_{R_4}\right]_{6\times 54} \quad (9)$$

The multiplication of *D* and *B* results in the network matrix *N*:

$$N = D \cdot \mathrm{diag}\{B_i\} \quad (10)$$

where B_i is the vector extracted from the i^{th} row of the *B* matrix.

The CRS network matrix *N* is given by:

$$N = \begin{bmatrix} \hat{\$}_{CV_0} & \hat{\$}_{CV_1} & -\hat{\$}_{R_1} & 0 & 0 & 0 & 0 & 0 & 0 \\ \hat{\$}_{CV_0} & 0 & 0 & \hat{\$}_{CV_2} & -\hat{\$}_{R_2} & 0 & 0 & 0 & 0 \\ \hat{\$}_{CV_0} & 0 & 0 & 0 & 0 & \hat{\$}_{CV_3} & -\hat{\$}_{R_3} & 0 & 0 \\ \hat{\$}_{CV_0} & 0 & 0 & 0 & 0 & 0 & 0 & \hat{\$}_{CV_4} & -\hat{\$}_{R_4} \end{bmatrix}_{24\times54} \tag{11}$$

where $\hat{\$}_{CVi}$ is a 6×6 sized matrix, including the normalized screws of the virtual chains, with $\$_i$=0,1,2,3,4; and $\hat{\$}_{Ri}$ is a 6×6 sized matrix including the normalized screws relative to the robots joints. The network matrix is split into primary matrix N_p and secondary matrix N_s:

$$N_p = \begin{bmatrix} [\hat{\$}_{CV_0}]_{6\times6} & [\hat{\$}_{CV_1}]_{6\times6} & 0 & 0 & 0 \\ [\hat{\$}_{CV_0}]_{6\times6} & 0 & [\hat{\$}_{CV_2}]_{6\times6} & 0 & 0 \\ [\hat{\$}_{CV_0}]_{6\times6} & 0 & 0 & [\hat{\$}_{CV_3}]_{6\times6} & 0 \\ [\hat{\$}_{CV_0}]_{6\times6} & 0 & 0 & 0 & [\hat{\$}_{CV_4}]_{6\times6} \end{bmatrix} \tag{12}$$

$$N_s = \begin{bmatrix} [-\hat{\$}_{R_1}]_{6\times6} & 0 & 0 & 0 \\ 0 & [-\hat{\$}_{R_2}]_{6\times6} & 0 & 0 \\ 0 & 0 & [-\hat{\$}_{R_3}]_{6\times6} & 0 \\ 0 & 0 & 0 & [-\hat{\$}_{R_4}]_{6\times6} \end{bmatrix} \tag{13}$$

The values of $\dot{q}_s$, $\dot{q}_p$, N_s and N_p defined above are replaced on Equation 7 and so, the velocities magnitudes of the robots joints are computed, as a function of the velocities magnitudes defined for each robots end-effector. Fig. 2 shows the robots while performing the designated task, in four different moments, and a top view of the system[1].

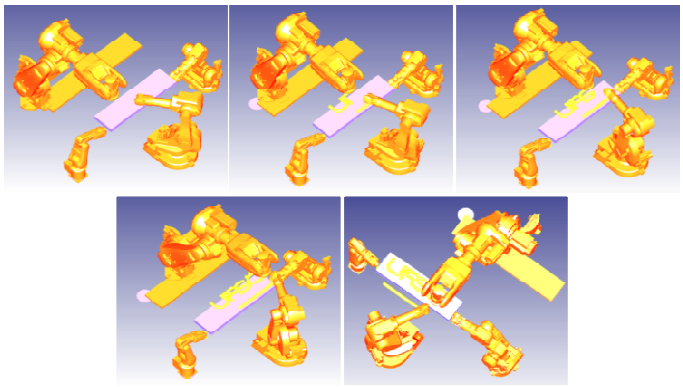

Fig. 2. Robots performing the task cooperatively

[1] A movie of the simulation is available on the youtube channel: http://www.youtube.com/user/UFSCLabCADCAM

The joints positions for each one of the robots are presented in the Fig. 3, with the time as the horizontal axis (0<t<40.8s). For the IRB 6620, only four joints are used, while the 4th and 6th joint values remain null, showing that for this task and initial configuration they weren't necessary. The requirement of joints according to tasks and initial configurations is an interesting topic for further researches.

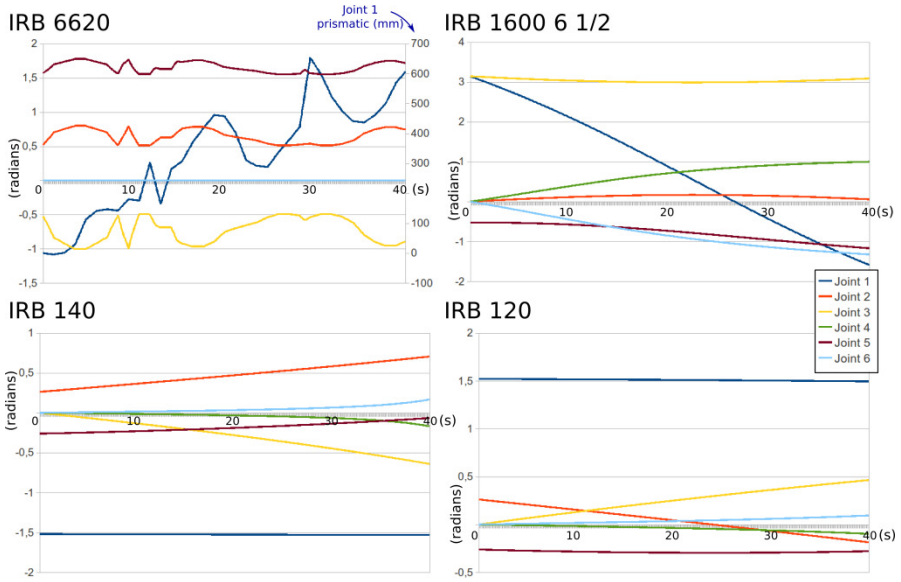

Fig. 3. The joints position for each robot

5 Conclusions

Planning tasks in cooperative robotic system requires more knowledge about the task and how to compose the robotic environment where it will be performed. The proposed methodology for the CRS kinematics computation allows to combine different configurations between robots and parts to be tested.

In this paper one example of application was presented. If one or more robots are added or removed from the system, the programming process is the same as presented, changing only the task specifications and the *N* matrix, since it was represented by the graph and by the normalized screw matrix, its composition is easily changed. The network matrix will have different number of rows and columns according to the number of robots in the system.

The off-line simulation results are important for CRS programming, especially when several robots are included, leading to a great number of parameters to be analyzed. Because of these characteristics, it is very hard to implement a real task without previous simulation. The analytical evaluation could not result in practical results, due to the great number of variables and its inter-dependences. By using simulations, it is possible to have a complete view of the system behavior, to detect

and analyze singularities, collisions and to enhance the performance of the task when it is finally performing the tasks.

Furthermore, the theory presented in this paper can be applied to CRS composed by other kinds of robots, such as: mobile robots, parallel robots and manipulators. Besides that, it is possible to simulate the system looking for collision avoidance of defined joints to obstacles of the environment, while the CRS is performing tasks. A deeper performance analysis of the system is an interesting research field, evaluating which of the available parameters can be adjusted in order to enhance the task performance.

References

1. Lewis, C.L.: Trajectory generation for two robots cooperating to perform a task. Proc. IEEE International Conference on Robotics and Automation. 2, 1626-1631 (1996)
2. Tzafestas C.S., Prokopiou P.A, Tzafestas S.G.: Path Planning and Control of a Cooperative Three-Robot System Manipulating Large Objects. Journal of Intelligent and Robotic Systems. 22, 99 -116 (1998)
3. Dourado A.O.: Cinemática de Robôs Cooperativos. UFSC. Florianópolis-SC, Brazil (2005)
4. Owen W.S., Croft E.A., Benhabib B.: A multi-arm robotic system for optimal sculpting. Robotics and Computer-Integrated Manufacturing 24. 92 - 104, Elsevier Ltd (2005)
5. Ribeiro L.P., Martins D.: Screw-based relative Jacobian for manipulators cooperating in a task using Assur virtual chains. In: Proceedings of 20th International Congress of Mechanical Engineering. ABCM, Brazil (2009)
6. Tonetto C.P., Dias A.: Trajectory Planning with redundant cooperative robotics systems. In: Proceedings of 20th International Congress of Mechanical Engineering. ABCM, Brazil (2009)
7. Tonetto C.P., Dias A.: The screw theory applied to compute the kinematics of multi-robots system in cooperative tasks.In: Proceedings of the 20th International Conference on Flexible Automation and Intelligent Manufacturing. California State University, East Bay, (2010)
8. Tonetto C.P., Rocha C.R., Dias A.: Simulation of Multi-robot Cooperative Systems programming based on a three environment definition In: Proceedings of the 12th Mechatronics Forum Biennial International Conference. Swiss Federal Institute of Technology, Zurich, Switzerland (2010)
9. Hunt K.K.: Don't cross-thread the screw. A Symposium Commemorating the Legacy, Works, and Life of Sir Robert Stawell Ball Upon the 100th Anniversary of A Treatise on the Theory of Screws. 1 - 56 (2000)
10. Davies T.H.: Kirchhoff's circulation law applied to multi-loop kinematic chains. Mechanism and Machine Theory, vol. 16, pp. 171 - 183 (1981)
11. Rocha C.R., Tonetto C.P., Dias A.: A comparison between the Denavit-Hartenberg and the screw-based methods used in kinematic modeling of robot manipulators. Robotics and Computer-Integrated Manufacturing. In: Conference papers of Flexible Automation and Intelligent Manufacturing. vol. 27, pp 723 - 728 (2011)
12. Tsai Lung-Wen,: Robot Analysis: The Mechanics of Serial and Parallel Manipulators. Jonh Wiley & Sons, INC (1999)
13. Campos A., Guenther R., Martins D.: Differential Kinematics of Serial Manipulators Using Virtual Chains. Brazilian Society of Mechanical Sciences and Engineering. ABCM. vol. XXVII, pp 345 - 356. (2005)