Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 48-29 (2015) 019–024

A Virtual Laboratory to Simulate the Control of Parallel Robots

Adrián Peidró, Oscar Reinoso, Arturo Gil, José M. Marín, Luis Payá

Systems Engineering and Automation Department
Miguel Hernández University, Avda. de la Universidad s/n. 03202, Elche (Alicante), Spain
e-mail: {apeidro, o.reinoso, arturo.gil, jmarin, lpaya}@umh.es

Abstract: This paper presents a virtual laboratory to simulate the dynamic control of parallel robots. The objective of the tool is to help robotics students comprehend the different singular and nonsingular methods by which parallel robots can change between different solutions to the forward kinematic problem, and to study how singularities affect the control of the robots when performing these changes. The tool is very intuitive and permits modifying the controller gains of the robots and visualizing the motion of the robot in different spaces.

Keywords: Control, Java, Parallel robots, Simulator, Singularities, Virtual laboratory

1. INTRODUCTION

Parallel robots are closed-chain mechanisms in which the *end-effector* (i.e., the body that carries a tool or a gripper to perform some task) is connected to a fixed base through two or more serial kinematic chains working in parallel. In contrast, serial robots only possess a single open kinematic chain. In general, this closed-loop architecture makes parallel robot faster, stiffer, and more precise than serial robots, which finds applications in surgical robotics, flight simulators and machine-tools, among many other (Merlet 2006).

Besides the previous advantages, parallel robots also have some drawbacks: their workspace is smaller than in serial robots of similar size, their control is more difficult due to the forward kinematic singularities, and their kinematic and dynamic analyses are generally more complex. This higher complexity is due to the closed-loop architecture of the kinematic chains of the robot. To help the robotics students better understand the complex kinematics and dynamics of parallel robots, it is necessary to use simulation tools.

There are many simulation tools that allow the students to easily analyze the kinematics of parallel robots. One of them is GIM (Petuya et al. 2011), a free educational program for intuitively studying the kinematics of general planar and spatial mechanisms. This tool is very flexible, since it permits the student to define mechanisms composed of any number of links connected through different kinds of joints (revolute, prismatic, spherical, etc). After defining the parallel robot, the student can simulate its forward kinematic problem, visualize its workspaces, or perform velocity analyses.

Another tool for studying the kinematics of parallel robots is The CUIK Suite (Porta et al. 2014). This powerful tool solves the position and path planning problems of general closed-chain multibody systems, which can have any architecture defined by the user. In particular, it is especially useful for visualizing the singularities, the configuration space, and the workspaces of parallel robots. This is an advanced research tool that should be used by master's and doctoral students.

Other educational tools are the Java Applets available at http://www.parallemic.org/JavaApplets.html, which can be used to analyze the Euler angles or study the kinematics and singularities of the 5R and 3RRR parallel robots. Another realistic educational simulator for studying the kinematics and singularities of the 5R robot is DexTAR Sim, which can be freely downloaded from http://www.mecademic.com/. This tool can also be used to simulate the off-line programming of this robot, or for programming a real commercially available 5R educational robot. RoboDK (http://www.robodk.com/) is another realistic free tool for simulating the kinematics and off-line programming of industrial robots, including some Delta-like parallel robots.

Although all these tools allow for the simulation of the kinematics of parallel robots (without dynamics), it is important that the students experiment also with the dynamics and control of parallel robots to understand how real (controlled) robots move and how singularities affect the control of the robot. The dynamics of parallel robots can be simulated using general-purpose tools such as ADAMS (Hajimirzaalian et al. 2010, Li and Xu 2009) or the library SimMechanics of MATLAB/Simulink (Koul et al. 2013, Li et al. 2015). SimMechanics can be used to model the dynamics of arbitrary closed-loop mechanisms, which can be combined with the control libraries of Simulink to effectively simulate the dynamic control of parallel robots. Although these tools are powerful, they are proprietary and students must learn how to configure them properly and build the simulations before simulating the control of parallel robots, which may be a difficult task that may hide the actual learning objective. Besides, they do not easily allow for the visualization of singularities at the same time that the robot is simulated, which is of great interest to understand the effect of these singularities in the control of the robot.

To help students understand the different ways to perform changes between different solutions to forward kinematics and the effects of singularities in the dynamic control of parallel robots, we present a virtual laboratory to simulate the control of parallel robots. Our virtual laboratory PAROLA

2405-8963
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2015.11.207

(PArallel RObotics LAboratory), developed with EJS (http://fem.um.es/Ejs/), consists of several Java Applets which allow the students to intuitively simulate the kinematics of some parallel robots, visualize its singularities and modify the geometry of the robots to study the relation between the design of the robot and the singularities or the workspace (Peidró et al. 2015). Now, the laboratory also allows the students to simulate the control of the 5R and 3RRR parallel robots to comprehend the importance of singularities in the control, simulating the dynamic behavior of real controlled parallel robots.

This paper is organized as follows. Section 2 reviews the forward kinematic problem of parallel robots and the singularities. Section 3 presents the tool developed to simulate the dynamic control of two parallel robots. Then, Section 4 presents some examples of the use of this tool to study some problems related to singularities. Finally, Section 5 presents the conclusions and the future work.

2. REDUCED CONFIGURATION SPACE

This section briefly reviews the forward kinematic problem of parallel robots and the singularity problem, following the model and notation introduced in (Thomas and Wenger 2011). For didactic purposes, in this paper we will deal with parallel robots with only 2 degrees of freedom (DOF); if the robot has $f > 2$ actuators, we will assume that $(f - 2)$ actuators remain constant. If we denote by (ω_1, ω_2) the joint coordinates of the robot (the coordinates that are controlled by actuators, such as motors) and by (s, t) the coordinates that define the position and orientation of the end-effector, then the forward kinematic problem consists in calculating (s, t) in terms of (ω_1, ω_2). In parallel robots, this problem has many different solutions: given a pair (ω_1, ω_2), different pairs (s, t) are obtained, which are the *assembly modes* of the robot.

One way of representing the motion capabilities of a parallel robot in a compact form consists in visualizing the *reduced configuration space*. To obtain the reduced configuration space, the joint coordinates ω_1 and ω_2 are varied in their motion ranges, solving the forward kinematic problem for each value of the joint coordinates. Then, one of the variables that define the position and orientation of the end-effector (t, for example) is plotted versus ω_1 and ω_2, for each of the obtained solutions. Plotting all the solutions yields a surface in the (ω_1, ω_2, t) space, which is the reduced configuration space. For example, Fig. 1 shows a reduced configuration space with spherical shape for didactic purposes (in general, the reduced configuration space has a more complex shape).

The reduced configuration space is very useful to plan the movements of the robot and to understand the effect of the singularities in its motion. For example, assume that the current configuration of the robot corresponds to the point A shown in Fig. 1, and the robot must move toward the configuration B, which lies in the southern hemisphere of the reduced configuration space. Solving the inverse kinematics, we determine that the configuration B can be attained with the joint coordinates $({\omega_1}^B, {\omega_2}^B)$, which are introduced into the control loop of the robot so that the actuators move the joint coordinates to these values, performing the trajectory T_1 shown in the plane (ω_1, ω_2) in Fig. 1 (which is the plane of joint coordinates, the plane in which the robot is controlled). However, although the desired joint coordinates are attained properly, the robot reaches a wrong configuration: it moves along the northern hemisphere toward the point C, which is the other solution to forward kinematics for the joint coordinates $({\omega_1}^B, {\omega_2}^B)$.

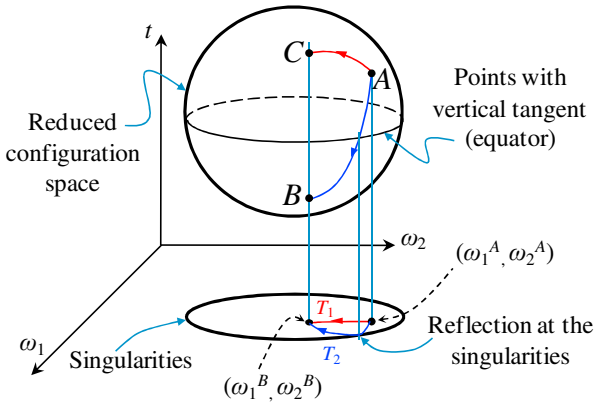

Fig. 1. A spherical reduced configuration space. The points B and C project onto the same point of the (ω_1, ω_2) plane: $({\omega_1}^B, {\omega_2}^B)$. The trajectory AB is singular.

Instead of performing the trajectory AC along the reduced configuration space, we want the robot to perform the trajectory AB, which is accomplished performing the trajectory T_2 in the (ω_1, ω_2) plane (see Fig. 1). Note that this trajectory requires crossing the points of the reduced configuration space which have a tangent that is parallel to the vertical axis t (i.e., the points of the equator of the sphere, in this example). When such points are projected onto the (ω_1, ω_2) plane, the singularities of the robot are obtained. Note that the trajectory T_2, which produces the correct transition between the configurations A and B, moves first toward the singularities and then suffers a reflection, to move toward the point $({\omega_1}^B, {\omega_2}^B)$ in the plane (ω_1, ω_2). Thus, this trajectory crosses a singularity. It is not easy to perform this *singular transition* in practice, because the robot is not controllable at singularities. This can be easily understood using the proposed simulation tool, as it will be shown in the examples in Section 4.

According to the previous example, if the robot needs to perform a transition between the points B and C to change between different assembly modes (i.e., to change between different solutions to forward kinematics for the same joint coordinates), the robot must cross a singularity, which requires performing an uncontrollable motion. However, for some robots for which the singularities form cusp points in the (ω_1, ω_2) plane, it is possible to change between different assembly modes without crossing singularities, i.e., by performing controllable movements. This situation is shown in Fig. 2, in which a trajectory T_3 that encloses the cusp point in the (ω_1, ω_2) plane, produces a transition between the points D and E (which are different solutions to the forward kinematic problem for the same joint coordinates) without crossing points of the reduced configuration space that have vertical tangent (singularities). This is possible thanks to the fold present in the reduced configuration space. For studying

nonsingular transitions, we will focus on cusp points, although this is not the only way to perform these controllable transitions (Bamberger et al. 2008).

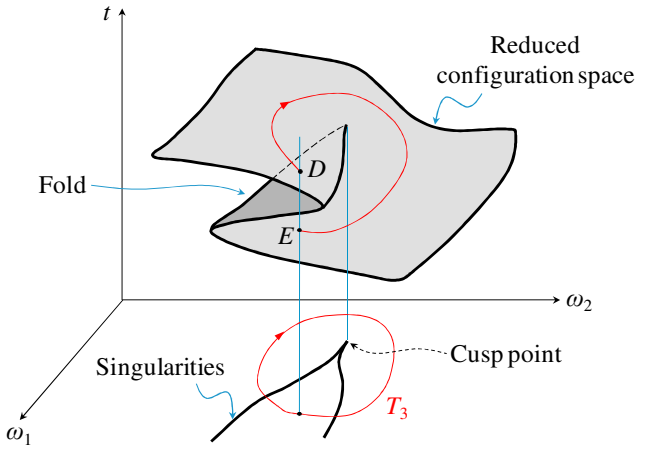

Fig. 2. A reduced configuration space with a fold, which yields a cusp point in the (ω_1, ω_2) plane. Encircling the cusp point produces a nonsingular transition between the solutions D and E. Adapted from (Thomas and Wenger 2011).

In the next section, we present the tool developed to simulate the dynamic control of two parallel robots, with the purpose of studying the movements of the robot along the reduced configuration space and evidencing the difficulty to control the robot when crossing singularities.

3. SIMULATION TOOL

This section describes the Java Applets developed to simulate the control of two well-known parallel robots. The applets can be downloaded from http://arvc.umh.es/parola. The currently implemented robots are the 5R and 3RRR parallel robots, shown in Figs. 3 and 4 respectively.

The 5R robot is a 2-DOF closed-chain mechanism composed of five links connected through five revolute joints. The link AB is fixed, whereas the links {l_1, l_2, l_3, l_4} are mobile. This robot is used for pick and place tasks: controlling the angles θ_1 and θ_2, the position (x, y) of its end-effector (the point P of Fig. 3) can be controlled in the plane. According to the notation used in Section 2, for this robot we have: $\omega_1 = \theta_1$, $\omega_2 = \theta_2$, $s = x$, and $t = y$. The geometry used in the simulations is: $a = 0$, $b = 0.48$ m, and $l_i = 0.2$ m ($i = 1, 2, 3, 4$).

The 3RRR robot is a 3-DOF parallel mechanism which has a triangular end-effector controlled by three parallel kinematic chains of the type RRR (where the underlined joint R means that it is actuated). Controlling the angles θ_1, θ_2, and θ_3, the position and orientation of the end-effector can be controlled. Note that, according to Fig. 4, the position of the end-effector is defined by the angles θ_2 and ψ_2 (which determine the position of point P), whereas its orientation is defined by φ. Since only parallel robots with 2 DOF are considered, one of the joint coordinates must be fixed. For the simulations, it is considered that $\theta_2 = \pi/2$. Thus, according to the notation used in Section 2, for this robot we have: $\omega_1 = \theta_1$, $\omega_2 = \theta_3$, $s = \psi_2$, and $t = \varphi$. For the geometric parameters, the following values are used in the simulator: $a_i = b_i = h = 0.2$ m, $Q_x = 0.62$ m, $Q_y = 0$, $R_x = 0.62/2$ m, and $R_y = 0.62/2 \cdot \sqrt{3}$ (i.e., the fixed supports form an equilateral triangle of side 0.62 m).

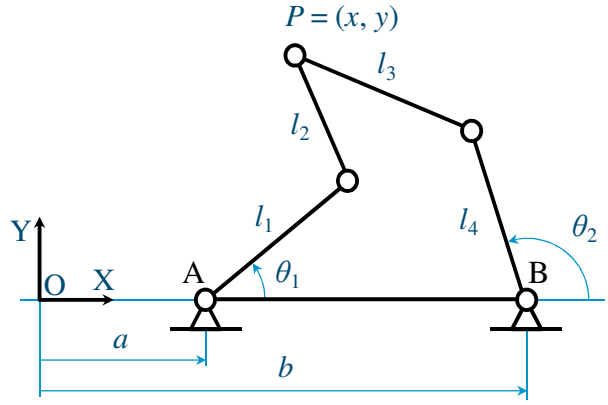

Fig. 3. The 5R parallel robot.

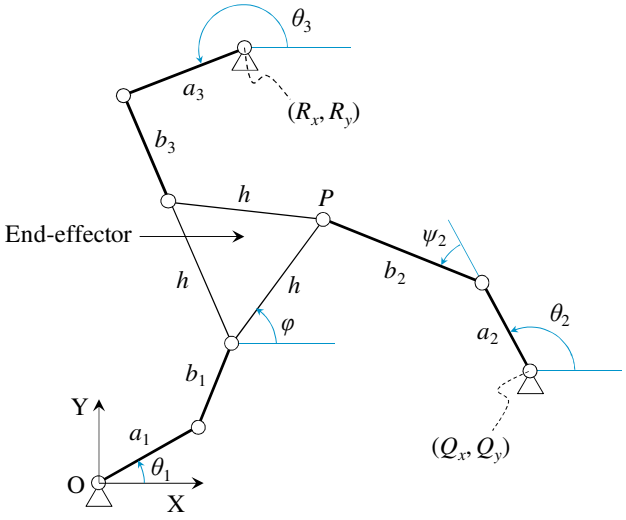

Fig. 4. The 3RRR parallel robot.

Figure 5 shows a screenshot of the simulation tool for the 3RRR robot (the tool has the same structure for the 5R robot). The tool has three windows: a main window (1), a window to visualize the evolution of the joint coordinates along time (2) and a window to visualize the reduced configuration space (3). The main window shows the robot and the plane of joint coordinates (ω_1, ω_2), and it has a control panel with some tabs for performing different analyses. Here, we will focus on the two tabs "Dynamics" and "Control", the functionalities of the remaining tabs are described in (Peidró et al. 2015).

The window (2) shows two plots with the time evolution of the joint coordinates of the robot along the simulation. The desired value for each of the joint coordinates is shown in green color, whereas the actual evolution of the two joint coordinates is represented in red color. The window (3) represents the reduced configuration space in light blue color, and the singularities are also represented in the plane of the joint coordinates in this window, as in Figs. 1 and 2. Recall, from Section 2, that the singularities are the projections of the points of the reduced configuration space with vertical tangent onto the plane of joint coordinates. The current

configuration of the robot is represented as a tiny red square on the reduced configuration space.

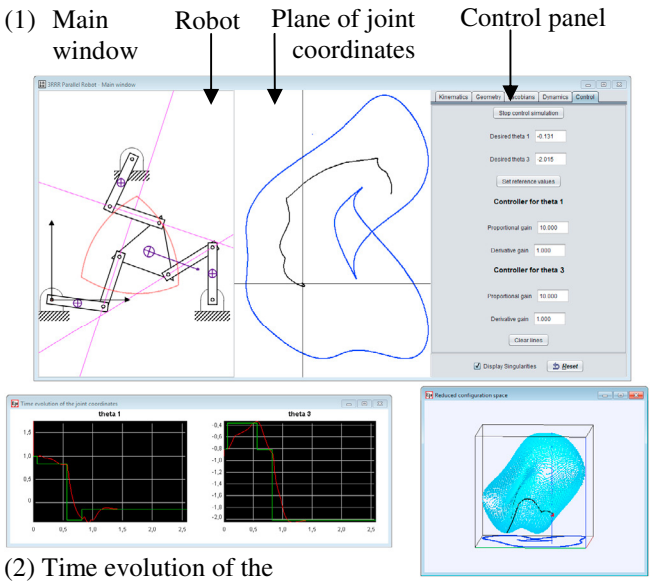

Fig. 5. Tool to simulate the control of parallel robots.

In the "Control" tab of the control panel of the main window (1), the user can simulate the dynamic control of the robot pressing the button "Start control simulation". After this, the tool simulates the dynamics and control of the robot. To control the robot, the following control scheme is applied to each of the joint coordinates:

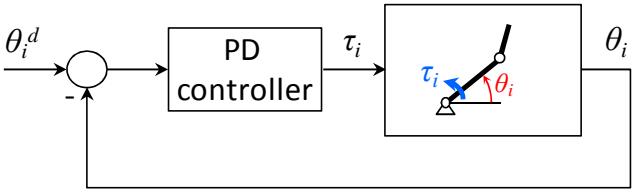

Fig. 6. Scheme to control an independent joint coordinate.

where θ_i^d is the reference or desired value for the joint coordinate θ_i, and τ_i is the control torque applied to the actuated joint. According to Fig. 6, each joint is controlled independently using a simple PD controller. Although the dynamics of the studied robots is highly nonlinear, using a PD controller for each joint independently yields appropriate results for the purposes of the simulator.

The proportional and derivative gains of the PD controllers of each joint can be modified in the "Control" tab. In this tab, the user can also introduce the desired values or references for each joint coordinate. After typing the desired values for the joint coordinates in the provided boxes, the user must press the button "Set reference values" to send these references to the control loop. Also, the reference values for the joint coordinates can be introduced by clicking the desired point in the joint plane (middle panel of the main window), which provides a more intuitive way of specifying the desired joint coordinates. Then, the controller tries to move the joint coordinates toward the desired point, showing the trajectory described by the robot both in the plane of joint coordinates and in the reduced configuration space. These trajectories can be cleared pressing the button "Clear lines" at the bottom of the control panel.

Finally, in the "Dynamics" tab, the user can modify the mass of the links (it is assumed that all the links have uniform mass distributions) or the friction constant of proportionality between adjacent links. The friction is assumed to be viscous, i.e., proportional to the relative angular velocity between adjacent links. The implementation of the dynamics of the robots is based on a Lagrangian formulation in independent coordinates (De Jalón and Bayo 1994).

Next, some simulation examples are presented, in order to study the control of the movements of parallel robots when changing between different assembly modes.

4. EXAMPLES

This section presents some examples of the use of the proposed simulation tool to study some problems related to the singularities of the studied parallel robots. The dynamic and control parameters used for the simulations are the default values, unless otherwise specified. The default values assume a mass of 0.5 kg for all the links and a viscous friction coefficient of $5 \cdot 10^{-4}$ N·m/(rad/s) between all the mobile links. For the proportional and derivative gains of the PD controllers, the default values are 10 and 1, respectively.

4.1 Control of the 5R Robot

Figure 7 shows the initial configuration of the 5R robot at the beginning of the control simulation. With the initial assembly mode, the points of the region R at the top part of its workspace cannot be attained. To reach these points, the robot must cross a singularity and change its configuration to adopt the assembly mode indicated in dashed line in Fig. 7, for which the region R can be accessed. According to the trajectory T_2 of Fig. 1, the robot should travel toward the singularities in the plane of joint coordinates so that the trajectory is reflected at the singularities to produce a change of assembly mode. Next, we will use the simulator to analyze different strategies that can be used to change the assembly mode crossing a singularity in the 5R robot.

First, it can be checked that it is very easy to perform this change manually in the simulator, following the steps described next. Beginning at the initial configuration, click outside the region of the joint plane enclosed by the singularities (point F of Fig. 8.) The points outside the region enclosed by the singularities correspond to forbidden configurations because the robot cannot be assembled at them. In this way, the controller will try to drive the robot toward a forbidden configuration. When the trajectory of the robot arrives at the singularities, it will suffer a reflection that corresponds to a change of assembly mode (the trajectory has crossed points of the reduced configuration space with vertical tangent). After detecting visually the first reflection (change of assembly mode), the user must click a point inside the region enclosed by the singularities (e.g. the point G of Fig. 8) and finally the initial point ($\theta_1 = 0$, $\theta_2 = \pi$) must be introduced as the reference to reach the dashed configuration

of Fig. 7, leaving the robot at a configuration where it can access the region *R* without crossing more singularities.

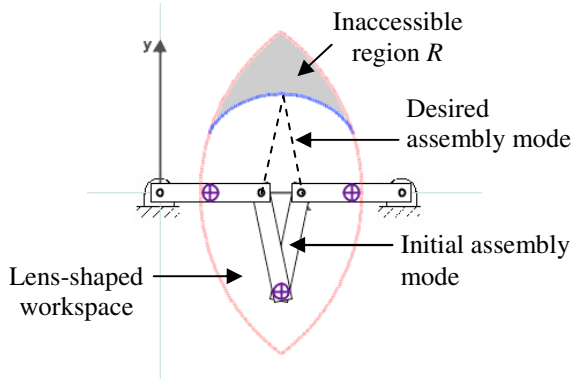

Fig. 7. Workspace and inaccessible region for the 5R robot.

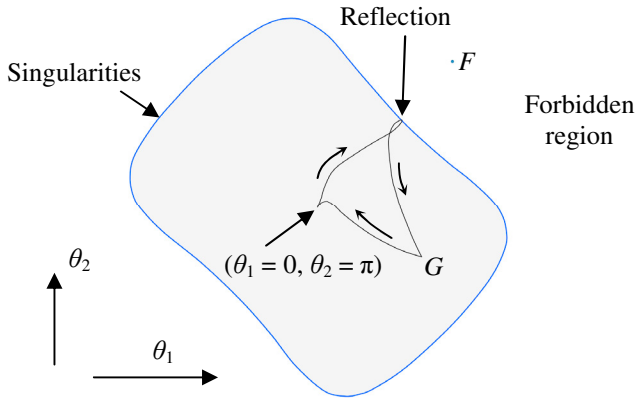

Fig. 8. Performing a singular transition in the 5R robot.

If the user does not introduce a point inside the region enclosed by the singularities after the first reflection at the singularities has occurred, then the controller will continue trying to bring the robot to the forbidden point *F*, and this will produce more impacts and reflections at the singularities, as shown in Fig. 9 (and each reflection will produce a change of assembly mode of the robot). Depending on the used controller, these reflections will decrease in amplitude until the controller stabilizes the robot at a singularity, as shown in Fig. 9. In this stabilized situation, the actuators are pushing the robot and trying to drive it to the forbidden region outside the singularities, a dangerous situation that may end up overheating and damaging the dc motors in a real robot.

The previous strategy for performing a singular change of assembly mode is manual: the user detects that the robot has changed the configuration after the reflection at the singularity occurs, and then introduces another point inside the region enclosed by the singularities to avoid more reflections that produce more changes of assembly mode. This detection may be automated introducing a sensor that detects the first singularity occurrence (for example, detecting that the angle between the links l_2 and l_3 crosses π rad, which is the singularity condition in this robot).

Alternatively, this may also be achieved introducing as the desired point a point inside the region enclosed by the singularities and close to them, instead of a point outside this region. Then, the ratio proportional gain/derivative gain of the controller is increased so that the trajectory presents an overshoot that exceeds the target point, touching the singularities and being reflected at them, producing a change of assembly mode (see Fig. 10). However, this solution is not robust or predictable (especially in a real robot), since sometimes the response may lack the necessary overshoot (not producing the change of assembly mode), or the trajectory may have successive oscillations that may produce several undesired changes of assembly mode, as in Fig. 9.

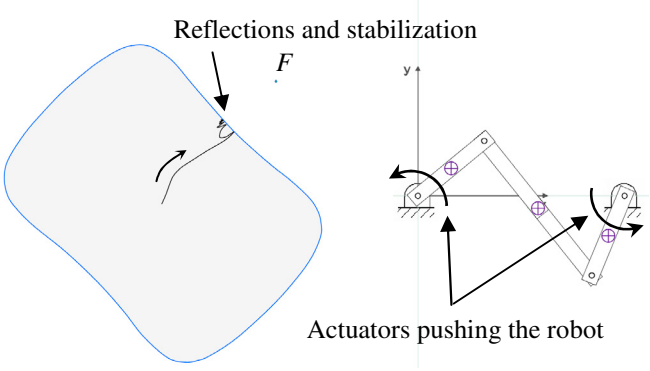

Fig. 9. Stabilization at a singular configuration.

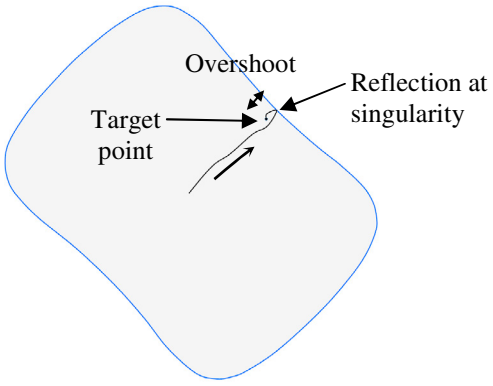

Fig. 10. Singular transition by overshoot.

As the simulator shows, singular transitions are not easy to control. It will be shown next that, in robots like the 3RRR, there are other ways to perform transitions between different assembly modes in a completely controlled manner.

4.2 Control of the 3RRR Robot

According to Fig. 11, the singularities in the plane of joint coordinates of the 3RRR robot have two cusp points. Thus, a transition between different assembly modes can be performed without crossing singularities by encircling one of the cusps. Beginning at the default configuration of the simulator, there are two possibilities: encircling the cusp 1 or the cusp 2. However, two questions arise: 1) which cusp should be encircled? and 2) In which direction should it be encircled: clockwise (CW) or counterclockwise (CCW)? Sometimes, encircling a cusp does not produce a change of assembly mode. And sometimes, it is possible to encircle a cusp in one direction, but not in the opposite direction. However, these questions cannot be answered looking only at

the plane of joint coordinates, the information present in the third dimension of the reduced configuration space (which is lost when the singularities are projected onto the plane of joint coordinates) must also be analyzed.

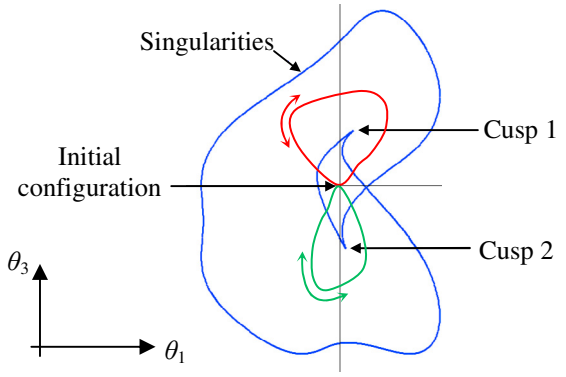

Fig. 11. Joint space of the 3RRR robot, with two cusp points.

For example, it can be checked in the simulator that beginning at the default configuration and encircling the cusp 1 of Fig. 11 both in CW and CCW directions does not produce a change of assembly mode. Encircling the cusp 2 in CW direction is not possible because the robot gets trapped in a singularity. Finally, only encircling the cusp 2 in CCW direction produces a nonsingular change of assembly mode, starting at the default configuration of the simulator (see Fig. 12). To know which trajectories will produce a nonsingular change of assembly mode, the user must carefully study the movements of the robot in the reduced configuration space, evaluating if a trajectory will cross points of the reduced configuration space with vertical tangent (singularities).

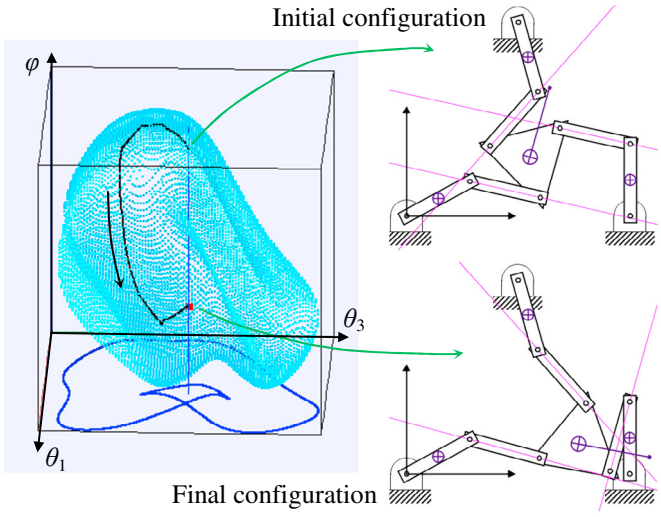

Fig. 12. Trajectory that encircles the cusp 2 of Fig. 11 in CCW direction, producing a nonsingular transition.

5. CONCLUSIONS

This paper has presented a virtual laboratory to simulate the dynamic control of two parallel robots, with the purpose of intuitively studying diverse methods to change between different assembly modes of the robots and to analyze how the control is affected by singularities. In the future, we will make the simulations more flexible, allowing the user to simulate the dynamics of the robots using arbitrary values for the geometric parameters. Also, the dynamics and control of other planar and spatial parallel robots will be implemented.

ACKNOWLEDGEMENT

This work has been supported by the Spanish Ministry of Education through grant FPU13/00413, by the Spanish Ministry of Economy through project DPI2013-41557-P, and by the Generalitat Valenciana through project AICO/2015/021. Also, the authors acknowledge Alfonso Hernández, CompMech, Dept. of Mechanical Eng. (UPVEHU), for the permission to use GIM® software (www.ehu.es/compmech).

REFERENCES

Bamberger, H., Wolf, A., and Shoham, M. (2008). Assembly mode changing in parallel mechanisms. *IEEE Transactions on Robotics*, 24 (4), pp. 765-772.

De Jalón, J.G., and Bayo, E. (1994). *Kinematic and dynamic simulation of multibody systems. The real-time challenge*, Chap. 5. Springer-Verlag, New York.

Hajimirzaalian, H., Moosavi, H., and Massah, M. (2010). Dynamics analysis and simulation of parallel robot Stewart platform. In *Proceedings of the 2nd International Conference on Computer and Automation Engineering*, pp. 472-477. IEEE.

Koul, M.H., Saha, S.K., and Manivannan, M. (2013). Simulation of haptics force law using SimMechanics and Simulink. In *Proceedings of the 1st International and 16th National Conference on Machines and Mechanisms,* pp. 641-648.

Li, M., Bi, D., and Xiao, Z. (2015). Mechanism simulation and experiment of 3-DOF parallel robot based on MATLAB. In Lee, M.S. and Wu, H.T. (ed.), *Proceedings of the 2015 International Power, Electronics and Materials Engineering Conference,* pp. 489-494. Atlantis Press.

Li, Y. and Xu, Q. (2009). Dynamic modeling and robust control of a 3-PRC translational parallel kinematic machine. *Robotics and Computer-Integrated Manufacturing*, 25 (3), pp. 630–640.

Merlet, J.P. (2006). *Parallel robots*. Springer Netherlands.

Peidró, A., Gil, A., Marín, J.M., and Reinoso, O. (2015). A web-based tool to analyze the kinematics and singularities of parallel robots. *Journal of Intelligent & Robotic Systems*, First published online: March 4, 2015. DOI: 10.1007/s10846-015-0220-4.

Petuya, V., Macho, E., Altuzarra, O., Pinto, C., and Hernández, A. (2011). Educational software tools for the kinematic analysis of mechanisms. *Comp. Appl. Eng. Education*, First published online: February 24, 2011. DOI: 10.1002/cae.20532. ISSN: 1061-3773.

Porta, J.M., Ros, L., Bohigas, O., Manubens, M., Rosales, C., and Jaillet, L. (2014). The CUIK Suite: analyzing the motion of closed-chain multibody systems. *IEEE Robotics & Automation Magazine*, 21 (3), pp. 105-114.

Thomas, F. and Wenger, P. (2011). On the topological characterization of robot singularity loci. A catastrophe-theoretic approach. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, pp. 3940-3945. IEEE.