A Singularity-Robust LQR Controller for Parallel Robots

Ricard Bordalba, Josep M. Porta, and Lluís Ros

Abstract— **Parallel robots exhibit the so-called forward singularities, which complicate substantially the planning and control of their motions. Often, such complications are circumvented by restricting the motions to singularity-free regions of the workspace. However, this comes at the expense of reducing the motion range of the robot substantially. It is for this reason that, recently, efforts are underway to control singularity-crossing trajectories. This paper proposes a reliable controller to stabilize such kind of trajectories. The controller is based on the classical theory of linear quadratic regulators, which we adapt appropriately to the case of parallel robots. As opposed to traditional computed-torque methods, the obtained controller does not rely on expensive inverse dynamics computations. Instead, it uses an optimal control law that is easy to evaluate, and does not generate instabilities at forward singularities. The performance of the controller is exemplified on a five-bar parallel robot accomplishing two tasks that require the traversal of singularities.**

I. Introduction

Parallel robots may be advantageous because of their stiffness, precision, and the efficiency of their movements [1]. These assets come at a high cost however: their workspace tends to be limited, and the planning and control of their motions are rather involved. Their closed kinematic chains and passive joints give rise to forward singularities [2], [3], in which the system becomes underactuated [4]. In such configurations, thus, the system will be unable to follow arbitrary accelerations. Actually, for some accelerations, the inverse dynamic problem will yield extremely large motor torques.

It is important to note, however, that singularity-crossing motions can safely be executed if the kinodynamic constraints of the robot are respected [4], [5], [6], [7]. This is a key finding, because the motion capabilities of the robot can be greatly enlarged if such motions are allowed. A recent planner, in fact, is able to generate singularity-crossing motions [8], [9]. This planner relies only on forward dynamics, and thus obtains trajectories that fulfill all kinodynamic constraints, even at the forward singularities. These trajectories are "open loop" though, and thus they need to be stabilized a posteriori with a feedback controller. The purpose of this paper is to complete this task, in order to achieve stable trajectory trackings, even in the presence of disturbances or dynamic model inaccuracies.

As in serial manipulators, the majority of parallel robots are stabilized with computed-torque (CT) control methods [10]. These employ inverse dynamics and feedback linearization to obtain a closed-loop system with easy-to-tune control parameters. CT laws are effective because they tend to produce global basins of attraction towards the desired trajectory. However, the inverse dynamic problem has to be solved at each iteration of the control loop. This is an expensive step that may prevent the design of high-frequency controllers in time-critical tasks. Moreover, CT controllers fail perilously near singularities because, as mentioned before, the inverse dynamics is generally unsolvable at such configurations. A possible workaround is to use multi-control architectures [11], which in the proximity of a singularity switch from a classical CT law to a controller based on virtual constraints. The solution is elegant, but the switching from one controller to the other depends on a heuristic parameter that needs to be tuned experimentally. This process can be costly, or even harmful to the machine.

The controller we introduce does not present the previous limitations. As we shall see, the classical theory of linear quadratic regulators can be extended to cope with closed kinematic chains, providing a single-structure controller that is easy to tune, and does not resort to feedback linearization. Since such a controller does not rely on inverse dynamics, it can be used to stabilize a trajectory even at a forward singularity. The resulting feedback laws, moreover, are cheap to evaluate and produce optimal actions that minimize trajectory errors and control efforts. Our work is inspired on recent developments in humanoid control [12], [13], but our arguments and context of application are different. Notice that in a humanoid robot all joints are actuated, and thus, forward singularities are rarely an issue. Instead, these singularities arise naturally in parallel robots, and must be analyzed and carefully considered.

The rest of the paper is structured as follows. Section II formalizes the problem dealt with in this paper. Section III recalls a few results from the theory of linear quadratic regulators [14], and explains why this theory cannot be directly applied to parallel robots. Section IV shows how to circumvent this issue by extending the theory to the case in which we wish to stabilize the robot at a fixed point. This extension is based on expressing the equations of motion in local coordinates. This paves the way to Section V, which extends the theory to the more complex case of trajectory stabilization. Section VI illustrates the performance of the obtained controllers in several test cases, and compares the results with those of CT control laws. Section VII, finally, provides the paper conclusions.

This work has been partially funded by the Spanish Ministry of Economy and Competitiveness under projects DPI2014-57220-C2-2-P and DPI2017-88282-P.

Ricard Bordalba, Josep M. Porta, and Lluís Ros are with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain {rbordalba,porta,ros}@iri.upc.edu

II. Problem Statement

Let us describe a robot configuration by means of a tuple $\boldsymbol{q} = (\boldsymbol{a}, \boldsymbol{r})$ of n_q generalized coordinates, where $\boldsymbol{a}$ contains the n_a actuated degrees of freedom of the robot, and $\boldsymbol{r}$ encompasses the n_r remaining coordinates in $\boldsymbol{q}$. In parallel robots the coordinates in $\boldsymbol{q}$ are not independent. Instead, they must satisfy a system of n_r nonlinear equations

$$\boldsymbol{\Phi}(\boldsymbol{q}) = \boldsymbol{0} \tag{1}$$

expressing all kinematic loop-closure constraints of the system. By differentiating Eq. (1) with respect to time, and letting $\boldsymbol{\Phi}_{\boldsymbol{q}} = \partial \boldsymbol{\Phi} / \partial \boldsymbol{q}$, we obtain

$$\boldsymbol{\Phi}_{\boldsymbol{q}} \, \dot{\boldsymbol{q}} = \boldsymbol{0}, \tag{2}$$

which delimits the feasible velocity vectors $\dot{\boldsymbol{q}}$ at a given configuration $\boldsymbol{q}$. The system formed by Eqs. (1) and (2) determines the feasible states $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{2n_q}$ of the robot. Hereafter, this system will be compactly written as

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}, \tag{3}$$

and the state space of the robot will be defined as the nonlinear manifold

$$\mathcal{X} = \{\boldsymbol{x} : \boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}\}, \tag{4}$$

which is of dimension $d_{\mathcal{X}} = 2\, n_a$ in general.

Let us now encode the forces and torques of the actuators into an action vector $\boldsymbol{u}$ of dimension n_a. The equations of motion of the parallel robot can then be written in the form

$$\ddot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}), \tag{5}$$

which can be obtained, for instance, from the Euler-Lagrange equations with multipliers [15]. By applying the change of variables $\boldsymbol{v} = \dot{\boldsymbol{q}}$, the state dynamics can be written as a first order differential equation of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \begin{cases} \dot{\boldsymbol{q}} = \boldsymbol{v} \\ \dot{\boldsymbol{v}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \end{cases} \tag{6}$$

Since parallel robots are constrained multibody systems, their time evolution is determined by the differential-algebraic equation formed by Eqs. (3) and (6) [15]. Note that for each value of $\boldsymbol{u}$, Eq. (6) defines a vector field over the $\mathcal{X}$ manifold of Eq. (3).

With the previous definitions, the control problem that we address can be stated as follows. Given time parametric state and action planned trajectories

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{x}_0(t) \\ \boldsymbol{u} &= \boldsymbol{u}_0(t) \end{aligned} \tag{7}$$

satisfying Eqs. (3) and (6) for all time instants $t \in [0, t_f]$, we wish to design a control law

$$\boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x}, t) \tag{8}$$

so that the closed-loop system

$$\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{\pi}(\boldsymbol{x}, t)) \tag{9}$$

is stable along $\boldsymbol{x}_0(t)$. This implies that the integral curves of Eq. (9) must be convergent to $\boldsymbol{x}_0(t)$ for arbitrary initial conditions in the neighborhood of such trajectory.

In the rest of the paper we shall assume that the Jacobian

$$\boldsymbol{F}_{\boldsymbol{x}}(\boldsymbol{x}) = \begin{bmatrix} \Phi_{\boldsymbol{q}} & 0 \\ \frac{\mathrm{d}\Phi_{\boldsymbol{q}}}{\mathrm{d}t} & \Phi_{\boldsymbol{q}} \end{bmatrix} \tag{10}$$

is full rank at all states $\boldsymbol{x}$ of the reference trajectory $\boldsymbol{x}_0(t)$, which is the case for generic robot geometries [2]. This means that the state space $\mathcal{X}$ will have a well-defined tangent space at such points, which is necessary for the control method we propose. Note that this assumption does not exclude forward singularities, i.e., the subjacobian $\boldsymbol{\Phi}_{\boldsymbol{r}} = \partial \boldsymbol{\Phi} / \partial \boldsymbol{r}$ can still be rank deficient.

III. Linear Quadratic Regulators

To address the problem just posed, we propose a controller that, for the system in Eq. (6), minimizes the additive cost function

$$J = \int_0^{t_f} \left[\bar{\boldsymbol{x}}^\top(t)\, \boldsymbol{Q}\, \bar{\boldsymbol{x}}(t) + \bar{\boldsymbol{u}}^\top(t)\, \boldsymbol{R}\, \bar{\boldsymbol{u}}(t) \right] \mathrm{d}t, \tag{11}$$

where $\bar{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{x}_0(t)$, $\bar{\boldsymbol{u}} = \boldsymbol{u} - \boldsymbol{u}_0(t)$, $\boldsymbol{Q}$ is a positive semi-definite matrix penalizing deviations from the planned trajectory $\boldsymbol{x}_0(t)$, and $\boldsymbol{R}$ is a positive-definite matrix penalizing deviations from the planned actions $\boldsymbol{u}_0(t)$.

Unfortunately, for general nonlinear systems $\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$, no solution is known for the mentioned controller. However, for linear time-varying systems

$$\dot{\boldsymbol{x}} = \boldsymbol{A}(t)\boldsymbol{x} + \boldsymbol{B}(t)\boldsymbol{u} \tag{12}$$

with independent variables $\boldsymbol{x}$, the solution is given by the linear quadratic regulator (LQR) [14]. The result is the optimal controller

$$\boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x}, t) = \boldsymbol{u}_0(t) - \boldsymbol{K}(t)\, \bar{\boldsymbol{x}} \tag{13}$$

where $\boldsymbol{K}(t) = \boldsymbol{R}^{-1}\boldsymbol{B}^\top \boldsymbol{S}(t)$, and $\boldsymbol{S}(t)$ is the solution to the differential Riccati equation

$$-\dot{\boldsymbol{S}} = \boldsymbol{S}\boldsymbol{A}(t) + \boldsymbol{A}^\top(t)\boldsymbol{S} - \boldsymbol{S}\boldsymbol{B}(t)\boldsymbol{R}^{-1}\boldsymbol{B}^\top(t)\boldsymbol{S} + \boldsymbol{Q}(t). \tag{14}$$

We can solve for $\boldsymbol{S}(t)$ by integrating this equation numerically backwards in time using the terminal condition $\boldsymbol{S}(t_f) = \boldsymbol{Q}$ [14].

Despite being restricted to the linear case, this solution is powerful, as one can always apply it to the time-varying linearization of $\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ along $\boldsymbol{x}_0(t)$, $\boldsymbol{u}_0(t)$. Such an approach has been shown to provide effective controllers on general nonlinear systems [14].

Note however that the result cannot be directly applied to parallel robots, because their $\boldsymbol{x}$ variables are not independent. In fact, if we obtain Eq. (12) by direct linearization of Eq. (6) we will find that the system is neither controllable nor stabilizable in the traditional sense, since it ignores the constraint in Eq. (3) [12]. One might argue that the differential equations could always be expressed in terms of minimal task- or joint-space coordinates, but then we would be faced with the singularities introduced by such

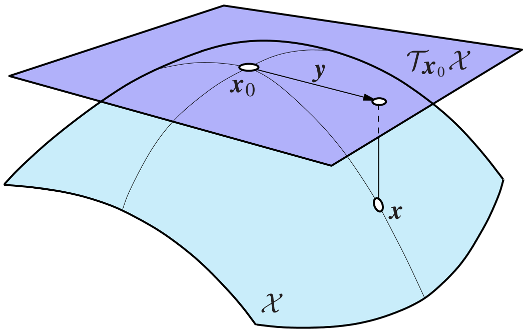

Fig. 1. The tangent space parameterization.

coordinates [2]. The route taken in this paper is robust to such singularities because, as we shall see, we employ a time-varying parameterization of $\boldsymbol{x}$ that is always well-defined.

IV. Stabilization at a Fixed Point

Consider the reference trajectory to be $\boldsymbol{x}_0(t) = \boldsymbol{x}_0 \; \forall t$, where $\boldsymbol{x}_0$ is a fixed point of $\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ for $\boldsymbol{u} = \boldsymbol{u}_0$, i.e., $\boldsymbol{g}(\boldsymbol{x}_0, \boldsymbol{u}_0) = \boldsymbol{0}$. We propose to use the tangent space local parametrization to find a set of minimal coordinates and then apply LQR techniques using such coordinates. Let the tangent space of $\mathcal{X}$ at $\boldsymbol{x}_0$ be

$$\mathcal{T}_{\boldsymbol{x}_0}\mathcal{X} = \{\dot{\boldsymbol{x}} \in \mathbb{R}^{2n_q} : \boldsymbol{F}_{\boldsymbol{x}}(\boldsymbol{x}_0)\,\dot{\boldsymbol{x}} = \boldsymbol{0}\}. \tag{15}$$

The tangent space can be used to define a local diffeomorphism $\boldsymbol{\varphi}$ from an open neighborhood of $\boldsymbol{x}_0$ in $\mathcal{X}$ to and open set of $\mathbb{R}^{d_{\mathcal{X}}}$. The map $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{x})$ provides the local coordinates, or parameters, of $\boldsymbol{x}$. This map is obtained by projecting $\boldsymbol{x}$ orthogonally to $\mathcal{T}_{\boldsymbol{x}_0}\mathcal{X}$ [Fig. 1]:

$$\boldsymbol{y} = \boldsymbol{U}^\top\,\bar{\boldsymbol{x}}, \tag{16}$$

where $\boldsymbol{U}$ is a $2n_q \times d_{\mathcal{X}}$ matrix whose columns provide an orthonormal basis of $\mathcal{T}_{\boldsymbol{x}_0}\mathcal{X}$, i.e., $\boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I}_{d_{\mathcal{X}}}$, where $\boldsymbol{I}_{d_{\mathcal{X}}}$ is the identity matrix of size $d_{\mathcal{X}}$. This basis can be computed efficiently using the QR decomposition of $\boldsymbol{F}_{\boldsymbol{x}}$ at $\boldsymbol{x}_0$. Since the map $\boldsymbol{\varphi}$ is a diffeomorphism, its inverse map $\boldsymbol{\psi} = \boldsymbol{\varphi}^{-1}$ exists and is implicitly determined by the system of nonlinear equations

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{x}) &= \boldsymbol{0}, \\ \boldsymbol{U}^\top\,\bar{\boldsymbol{x}} - \boldsymbol{y} &= \boldsymbol{0}, \end{aligned} \tag{17}$$

which, for a given $\boldsymbol{y} \in \mathbb{R}^{d_{\mathcal{X}}}$, can be solved for $\boldsymbol{x}$ using the Newton-Raphson method (if $\boldsymbol{x}$ is close to $\boldsymbol{x}_0$).

The time derivative of Eq. (16) gives the expression of the vector field defined by Eq. (6) in local coordinates of the tangent space [Fig. 2(a)]:

$$\dot{\boldsymbol{y}} = \boldsymbol{U}^\top\dot{\boldsymbol{x}} = \boldsymbol{U}^\top\,\boldsymbol{g}(\boldsymbol{\psi}(\boldsymbol{y}), \boldsymbol{u}) = \tilde{\boldsymbol{g}}(\boldsymbol{y}, \boldsymbol{u}). \tag{18}$$

The resulting system may not be stable at the fixed point and thus, our goal is to find a feedback law to stabilize it [Fig. 2(b)].

Since we wish asymptotic convergence to the fixed point, we are in the context of infinite-horizon optimal control. Therefore, we set the terminal time t_f of the cost in Eq. (11) to infinity. Moreover, by using $\boldsymbol{y} = \boldsymbol{U}^\top\,\bar{\boldsymbol{x}}$, this cost can be rewritten in terms of the local coordinates $\boldsymbol{y}$:

$$J = \int_0^\infty \left[\boldsymbol{y}^\top(t)\,\tilde{\boldsymbol{Q}}\,\boldsymbol{y}(t) + \bar{\boldsymbol{u}}^\top(t)\,\boldsymbol{R}\,\bar{\boldsymbol{u}}(t)\right] \mathrm{d}t, \tag{19}$$

where $\tilde{\boldsymbol{Q}} = \boldsymbol{U}^\top\boldsymbol{Q}\boldsymbol{U}$.

To obtain an LQR solution, we linearize Eq. (18) around the fixed point to get a linear time-invariant system in tangent space coordinates,

$$\dot{\boldsymbol{y}} = \tilde{\boldsymbol{A}}\boldsymbol{y} + \tilde{\boldsymbol{B}}\bar{\boldsymbol{u}}, \tag{20}$$

where

$$\tilde{\boldsymbol{A}} = \frac{\partial\tilde{\boldsymbol{g}}}{\partial\boldsymbol{y}} = \boldsymbol{U}^\top\,\frac{\partial\boldsymbol{g}}{\partial\boldsymbol{x}}\frac{\partial\boldsymbol{\psi}(\boldsymbol{y})}{\partial\boldsymbol{y}},$$

and

$$\tilde{\boldsymbol{B}} = \frac{\partial\tilde{\boldsymbol{g}}}{\partial\boldsymbol{u}} = \boldsymbol{U}^\top\,\frac{\partial\boldsymbol{g}}{\partial\boldsymbol{u}}.$$

Note that $\frac{\partial\boldsymbol{\psi}(\boldsymbol{y})}{\partial\boldsymbol{y}}$ can be computed in closed form. Consider the mapping

$$\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{x}) = \boldsymbol{U}^\top\,(\boldsymbol{x} - \boldsymbol{x}_0)\,. \tag{21}$$

Lets now use the inverse mapping $\boldsymbol{x} = \boldsymbol{\psi}(\boldsymbol{y})$ to rewrite Eq. (21) as

$$\boldsymbol{y} = \boldsymbol{U}^\top\,(\boldsymbol{\psi}(\boldsymbol{y}) - \boldsymbol{x}_0)\,. \tag{22}$$

If we compute the partial derivative of both side of Eq. (22) with respect to $\boldsymbol{y}$, we have

$$\boldsymbol{I}_{d_{\mathcal{X}}} = \boldsymbol{U}^\top\frac{\partial\boldsymbol{\psi}(\boldsymbol{y})}{\partial\boldsymbol{y}}. \tag{23}$$

Thus, $\frac{\partial\boldsymbol{\psi}(\boldsymbol{y})}{\partial\boldsymbol{y}}$ must be $\boldsymbol{U}$.

From the theory of Section III, the optimal control that minimizes the cost in Eq. (19) for the system in Eq. (20) is given by

$$\boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{y}) = \boldsymbol{u}_0 - \tilde{\boldsymbol{K}}\,\boldsymbol{y}, \tag{24}$$

which can be written in terms of the $\boldsymbol{x}$ variables as

$$\boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x}) = \boldsymbol{u}_0 - \tilde{\boldsymbol{K}}\boldsymbol{U}^\top\,\bar{\boldsymbol{x}}. \tag{25}$$

Since we are solving an infinite horizon control problem, the solution of Eq. (14) reduces to the algebraic Riccati equation [14]

$$\boldsymbol{0} = \boldsymbol{S}\tilde{\boldsymbol{A}} + \tilde{\boldsymbol{A}}^\top\boldsymbol{S} - \boldsymbol{S}\tilde{\boldsymbol{B}}\boldsymbol{R}^{-1}\tilde{\boldsymbol{B}}^\top\boldsymbol{S} + \tilde{\boldsymbol{Q}}, \tag{26}$$

which can be solved using generalized eigenvalue methods [16].

V. Trajectory Stabilization

Now consider the general case of stabilizing an arbitrary trajectory $\boldsymbol{x}_0(t), \boldsymbol{u}_0(t)$. Since this trajectory is now time-varying, the tangent space basis $\boldsymbol{U}(t)$ and the mappings $\boldsymbol{\psi}(\boldsymbol{y}, t)$ and $\boldsymbol{\varphi}(\boldsymbol{x}, t)$ are also time-varying along $\boldsymbol{x}_0(t)$.

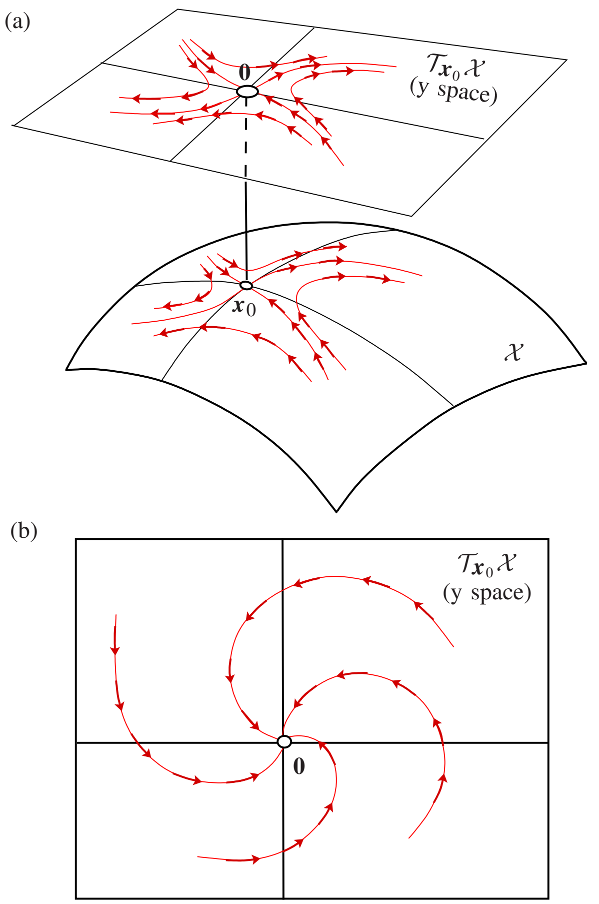
(a)
$\mathcal{T}_{\boldsymbol{x}_0}\mathcal{X}$
(y space)
$\mathbf{0}$
$\boldsymbol{x}_0$
$\mathcal{X}$
(b)
$\mathcal{T}_{\boldsymbol{x}_0}\mathcal{X}$
(y space)
$\mathbf{0}$

Fig. 2. (a) A vector field $\dot{\boldsymbol{x}} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u})$ on $\mathcal{X}$ can be locally expressed in tangent-space coordinates. The field is action varying, but we draw it for a given value of $\boldsymbol{u}$ only. (b) If, as in (a), $\boldsymbol{x}_0$ is a fixed point for some $\boldsymbol{u} = \boldsymbol{u}_0$, we can design a feedback law to stabilize the system at $\boldsymbol{x}_0$.

As in the previous section, we also write the vector field in local coordinates, which can be done by differentiating Eq. (16):

$$\begin{aligned}\dot{\boldsymbol{y}} &= \dot{\boldsymbol{U}}^\top(t)\,\bar{\boldsymbol{x}} + \boldsymbol{U}^\top(t)\,\dot{\bar{\boldsymbol{x}}}\\ &= \dot{\boldsymbol{U}}^\top(t)\;\left[\boldsymbol{\psi}(\boldsymbol{y}, t) - \boldsymbol{x}_0(t)\right] + \\ &\quad\ \boldsymbol{U}^\top(t)\;\left[\boldsymbol{g}(\boldsymbol{\psi}(\boldsymbol{y}, t), \boldsymbol{u}) - \boldsymbol{g}(\boldsymbol{x}_0(t), \boldsymbol{u}_0(t))\right]\\ &= \tilde{\boldsymbol{g}}(\boldsymbol{y}, \boldsymbol{u}, t).\end{aligned}\tag{27}$$

Note that the resulting vector field is now action- and time-varying [Fig. 3(a)], and our goal is to stabilize this system to make it convergent to the desired trajectory [Fig. 3(b)].

Noting that the terminal time t_f is the trajectory time, we rewrite the finite-horizon cost in local coordinates as

$$J = \int_0^{t_f} \left[\boldsymbol{y}^\top(t)\,\tilde{\boldsymbol{Q}}(t)\,\boldsymbol{y}(t) + \bar{\boldsymbol{u}}^\top(t)\,\boldsymbol{R}\,\bar{\boldsymbol{u}}(t)\right] \mathrm{d}t. \tag{28}$$

where $\tilde{\boldsymbol{Q}}(t) = \boldsymbol{U}^\top(t)\,\boldsymbol{Q}\,\boldsymbol{U}(t)$.

Now we linearize Eq. (27) about $\boldsymbol{x}_0(t), \boldsymbol{u}_0(t)$ to obtain:

$$\dot{\boldsymbol{y}} = \tilde{\boldsymbol{A}}(t)\,\boldsymbol{y} + \tilde{\boldsymbol{B}}(t)\,\bar{\boldsymbol{u}}, \tag{29}$$

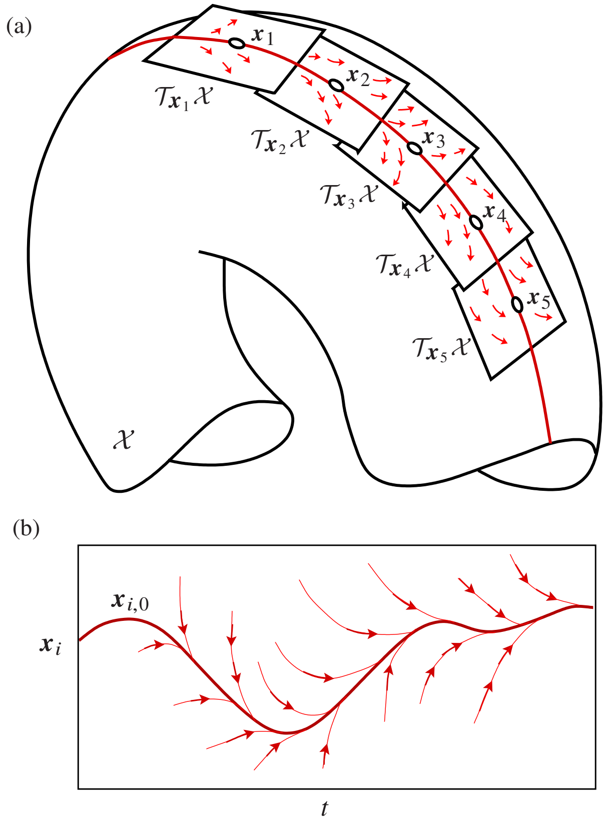
(a)
$\boldsymbol{x}_1$
$\boldsymbol{x}_2$
$\boldsymbol{x}_3$
$\boldsymbol{x}_4$
$\boldsymbol{x}_5$
$\mathcal{T}_{\boldsymbol{x}_1}\mathcal{X}$
$\mathcal{T}_{\boldsymbol{x}_2}\mathcal{X}$
$\mathcal{T}_{\boldsymbol{x}_3}\mathcal{X}$
$\mathcal{T}_{\boldsymbol{x}_4}\mathcal{X}$
$\mathcal{T}_{\boldsymbol{x}_5}\mathcal{X}$
$\mathcal{X}$
(b)
$\boldsymbol{x}_{i,0}$
$\boldsymbol{x}_i$
t

Fig. 3. (a) The linearization of the system along a trajectory yields a time (and action) varying vector field. At each time, the field is drawn for a single value of $\boldsymbol{u}$ only. (b) Our goal is to design a feedback law stabilizing the system along the desired state-time trajectory.

where

$$\tilde{\boldsymbol{A}}(t) = \frac{\partial \tilde{\boldsymbol{g}}}{\partial \boldsymbol{y}} = \dot{\boldsymbol{U}}^\top(t)\,\frac{\partial \boldsymbol{\psi}(\boldsymbol{y}, t)}{\partial \boldsymbol{y}} + \boldsymbol{U}^\top(t)\,\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\boldsymbol{U}(t)$$

and

$$\tilde{\boldsymbol{B}}(t) = \frac{\partial \tilde{\boldsymbol{g}}}{\partial \boldsymbol{u}} = \boldsymbol{U}^\top(t)\,\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}}.$$

Note that $\frac{\partial \boldsymbol{\psi}(\boldsymbol{y}, t)}{\partial \boldsymbol{y}} = \boldsymbol{U}(t)$.

The tangent space basis $\boldsymbol{U}(t)$ could be computed from a QR decomposition of $\boldsymbol{F}_{\boldsymbol{x}}(t)$. However, this approach does not guarantee that the time-varying basis is continuous, which is needed to obtain $\dot{\boldsymbol{U}}(t)$. Therefore, both $\boldsymbol{U}(t)$ and $\dot{\boldsymbol{U}}(t)$ can be computed considering the following identities:

$$\begin{aligned}\boldsymbol{U}^\top(t)\,\boldsymbol{U}(t) &= \boldsymbol{I},\\ \boldsymbol{F}_{\boldsymbol{x}}(t)\,\boldsymbol{U}(t) &= \mathbf{0},\end{aligned}\tag{30}$$

which can be differentiated to obtain:

$$\dot{\boldsymbol{U}}^\top(t)\,\boldsymbol{U}(t) + \boldsymbol{U}^\top(t)\,\dot{\boldsymbol{U}}(t) = \mathbf{0}, \tag{31}$$

$$\dot{\boldsymbol{F}}_{\boldsymbol{x}}(t)\,\boldsymbol{U}(t) + \boldsymbol{F}_{\boldsymbol{x}}(t)\,\dot{\boldsymbol{U}}(t) = \mathbf{0}. \tag{32}$$

From Eq. (32), $\dot{\boldsymbol{U}}(t)$ can be isolated to obtain

$$\dot{\boldsymbol{U}}(t) = -\boldsymbol{F}_{\boldsymbol{x}}^{+}(t)\,\dot{\boldsymbol{F}}_{\boldsymbol{x}}(t)\,\boldsymbol{U}(t), \tag{33}$$

where $\boldsymbol{F}_{\boldsymbol{x}}^{+}(t)$ is the pseudoinverse of $\boldsymbol{F}_{\boldsymbol{x}}(t)$, which requires the Jacobian $\boldsymbol{F}_{\boldsymbol{x}}(t)$ to be full rank $\forall t$. Then, $\boldsymbol{U}(t)$ can be

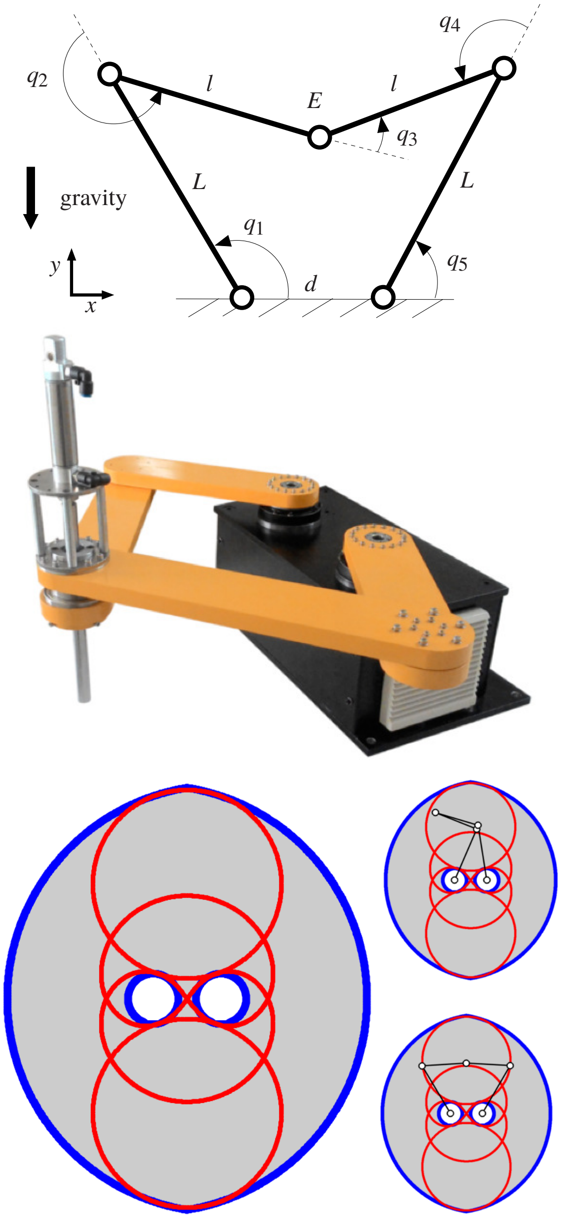

Fig. 4. Top: Geometry of a five-bar robot. Middle: the RAPI-MOD model by Gridbot technologies. Bottom: the workspace of the robot (in grey) and its forward singularity locus (in red). Two near-singular configurations are shown in the right.

obtained integrating Eq. (33) from $\boldsymbol{U}(0)$ enforcing Eq. (30) at each integration step.

Then, from the results in Section III, the optimal control that minimizes the cost in Eq. (28) for the system in Eq. (29) is given by

$$\boldsymbol{u}(t) = \boldsymbol{\pi}(\boldsymbol{y}, t) = \boldsymbol{u}_0(t) - \tilde{\boldsymbol{K}}(t)\, \boldsymbol{y}, \tag{34}$$

which in terms of the $\boldsymbol{x}$ variables results in

$$\boldsymbol{u} = \boldsymbol{\pi}(\boldsymbol{x}, t) = \boldsymbol{u}_0(t) - \tilde{\boldsymbol{K}}(t)\, \boldsymbol{U}^{\top}(t)\, \bar{\boldsymbol{x}}. \tag{35}$$

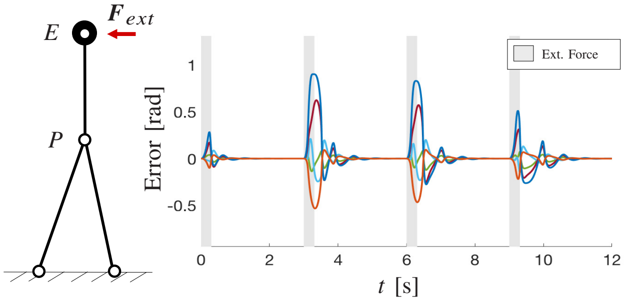

Fig. 5. Left: A weight stabilization task. The robot has to keep an end-effector load at position E, counteracting bounded disturbances applied ($\boldsymbol{F}_{ext}$). Right: The error responses of all joints angles shown over time, drawn with different colors. The force disturbances are applied (and kept constant) during the grey time windows. An animation of the motion can be seen in youtu.be/LNoFYAW209Q.

VI. Test Cases

We next illustrate the performance of the previous LQR controllers in a five-bars robot with the geometry shown in Fig. 4 (top). The angles q_1 and q_5 are actuated, which allow us to control the position E of the end-effector. The remaining angles q_2, q_3, and q_4 are passive. Various versions of this robot have been used both in research and industry [17], [18]. Fig. 4 (middle), for example, shows the RAPI-MOD model from Gridbot technologies. In our version, the base distance is $d = 0.6472$ [m], and the proximal and distal link lengths are $L = 1.14$ [m] and $l = 0.9$ [m]. With these dimensions, the workspace of point E is the grey area of Fig. 4 (bottom-left). This figure also shows the positions of E for which a forward singularity is encountered (in red). We recall from [17] that such singularities appear whenever q_3 is either 0 or π.

Because of the complications introduced by forward singularities, programmers tend to restrict the motion of these robots to singularity-free regions of the workspace [19]. From Fig. 4 (bottom) it is obvious, however, that the motion capabilities would be further enlarged if singularity-crossing motions could be controlled. The following examples show that the proposed LQR controller is successful in achieving such goal. In all examples, the mass of each link is 0.5 [kg] and the end-effector is transporting a load of 1 [kg]. The moments of inertia of the proximal and distal links (with respect to their center of gravity) are of 0.0541 [kg·m^2] and 0.0338 [kg·m^2], respectively.

A. Weight Stabilization at a Forward Singularity

In this example, the robot has to keep its load of 1 [Kg] at the upright configuration of Fig. 5 (left). The two distal links are coincident in such a pose. Thus the robot will move very close to the singularity all of the time, which renders computed-torque methods inapplicable to the situation. The task is difficult because the upper two links can fall down by gravity under the slightest disturbances. Without feedback, therefore, the robot is at an unstable fixed point, and thus the example is ideal to test the controller of Section IV.

To show the robustness of this controller, we simulate the effect of an horizontal disturbance force $\boldsymbol{F}_{ext}$ applied

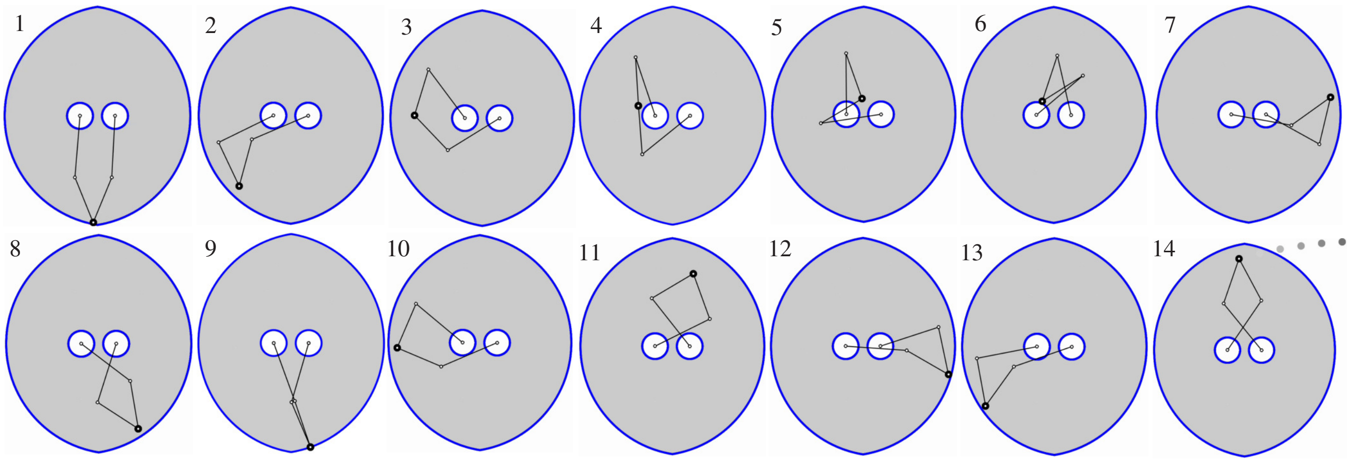

Fig. 6. A weight-throwing task. The robot picks an object in a lower position (snapshot 1) and performs successive swing-up motions to throw it from an upper position (snapshot 14). The robot crosses the forward singularity twice (snapshots 4 and 9). An animation of the motion can be seen in youtu.be/LNoFYAW209Q.

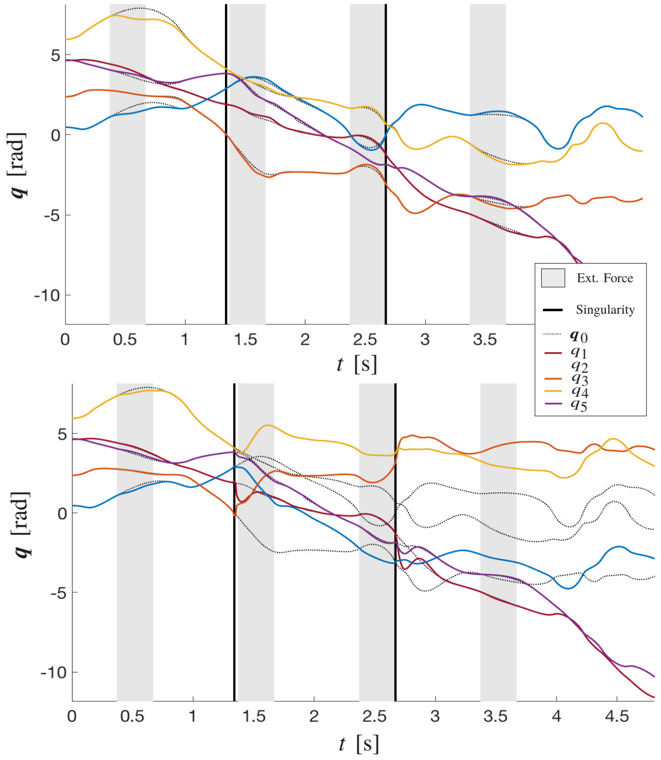

Fig. 7. Comparison of the LQR controller (upper plot) with a standard CT controller (lower plot) in a trajectory stabilization task under disturbances and singularity crossing. An animation of the motion can be seen in youtu.be/LNoFYAW209Q.

to point E every 3 seconds. Every disturbance is maintained during 0.2 seconds, and its magnitude is randomly selected in the range $(-15, 0)$ [N]. Fig. 5 (right) shows the evolution of all joint angle errors during the application of these disturbances, which occur at the time windows shown in grey. As we see, the LQR controller compensates the disturbances and returns the robot to the desired position in all cases. See youtu.be/LNoFYAW209Q for an animation of the task.

The convergence rate of the controller can be tunned by changing the $\boldsymbol{Q}$ and $\boldsymbol{R}$ matrices of Eq. (11). The plot of Fig. 5 was obtained with

$$\boldsymbol{Q} = \begin{bmatrix} k_q \boldsymbol{I}_5 & \boldsymbol{0} \\ \boldsymbol{0} & k_v \boldsymbol{I}_5 \end{bmatrix} \tag{36}$$

and

$$\boldsymbol{R} = k_u \boldsymbol{I}_2, \tag{37}$$

where $k_q = 300$, $k_v = 10$ and $k_u = 2$. The size of these matrices agrees with the fact that $\boldsymbol{q} = (q_1, \ldots, q_5)$ in this robot (and thus $n_q = 5$) and only two joints are actuated (i.e. $n_a = 2$).

B. A Weight Throwing Task

We next compare the performance of the LQR and CT controllers on tracking a trajectory across forward singularities. The reference trajectory $\boldsymbol{x}_0(t)$ and its action history $\boldsymbol{u}_0(t)$ have been obtained with the planner in [8], and correspond to a weight-throwing task illustrated in Fig. 6. The robot has to pick a heavy ball in the lowest position of its workspace (first snapshot) and has to successively increase its velocity so as to throw it from an upper position (last snapshot). The robot goes twice across forward singularities (snapshots 4 and 9). Thus its control may be loosed at such configurations.

The tracking results of the LQR and CT controllers are compared in Fig. 7. In both cases, the robot is subject to the same disturbances. These are applied to the end-effector, and take place during the grey time windows indicated. The time instants in which the singularities are crossed are indicated with a black vertical line. The reference trajectories for all angles q_i are drawn with discontinuous lines. As we see, the CT controller can counteract the applied disturbances before meeting the first singularity. At the forward singularity, however, the inverse dynamic problem produces unbounded torques. The control of the robot is lost, and the rest of the trajectory cannot be tracked anymore. The CT controller has been implemented in the active joint space, and thus $q_1(t)$ and $q_5(t)$ can reliably be tracked, but note that the remaining angles evolve differently because of configuration-space bifurcations. On the contrary, the LQR controller is quite robust to disturbances and singularity crossings. Even if an external force is applied right before the second singularity, the LQR controller is able to converge to the desired trajectory. A video showing the two experiments can be found in youtu.be/LNoFYAW209Q.

VII. Conclusions

This paper proposes the adaptation of an LQR controller to treat the closed-kinematic chains that arise in parallel robots. We use the tangent space local parametrization to find a set of minimal coordinates for the system in order to formulate the LQR controller in such coordinates. With this technique, we are able to stabilize both a fixed point and a preplanned trajectory. This controller can cope with forward singularities, whereas existing CT controllers cannot. Specifically, we are able to stabilize these singularity under significant disturbances at the end-effector. In contrast, the CT method produces unbounded torque at these points, which results in a trajectory that cannot be fully tracked anymore once the singularity is reached.

Our current efforts are focused on the implementation and validation of the proposed approach in a real robot.

References

[1] L.-W. Tsai, *Robot Analysis: the Mechanics of Serial and Parallel Manipulators*. Wiley-Interscience, 1999.

[2] O. Bohigas, M. Manubens, and L. Ros, *Singularities of robot mechanisms: numerical computation and avoidance path planning*, ser. Mechanisms and Machine Science. Springer, 2016, vol. 41.

[3] ——, "Singularities of non-redundant manipulators: A short account and a method for their computation in the planar case," *Mechanism and Machine Theory*, vol. 68, pp. 1–17, 2013.

[4] S. Briot and V. Arakelian, "Optimal force generation in parallel manipulators for passing through the singular positions," *The International Journal of Robotics Research*, vol. 27, no. 8, pp. 967–983, 2008.

[5] C. K. Kevin Jui and Q. Sun, "Path tracking of parallel manipulators in the presence of force singularity," *Journal of dynamic systems, measurement, and control*, vol. 127, no. 4, pp. 550–563, 2005.

[6] S. K. Ider, "Inverse dynamics of parallel manipulators in the presence of drive singularities," *Mechanism and Machine Theory*, vol. 40, no. 1, pp. 33–44, 2005.

[7] M. Özdemir, "Removal of singularities in the inverse dynamics of parallel robots," *Mechanism and Machine Theory*, vol. 107, pp. 71–86, 2017.

[8] R. Bordalba, L. Ros, and J. M. Porta, "Randomized Kinodynamic Planning for Constrained Systems," in *IEEE International Conference on Robotics and Automation*, 2018.

[9] R. Bordalba, J. M. Porta, and L. Ros, "Randomized kinodynamic planning for cable-suspended parallel robots," in *Cable-Driven Parallel Robots*. Springer, 2018, pp. 195–206.

[10] F. Aghili, "A unified approach for inverse and direct dynamics of constrained multibody systems based on linear projection operator: applications to control and simulation," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 834–849, 2005.

[11] R. B. Hill, D. Six, A. Chriette, S. Briot, and P. Martinet, "Crossing type 2 singularities of parallel robots without pre-planned trajectory with a virtual-constraint-based controller," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 6080–6085.

[12] M. Posa, S. Kuindersma, and R. Tedrake, "Optimization and stabilization of trajectories for constrained dynamical systems," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 1366–1373.

[13] S. Mason, N. Rotella, S. Schaal, and L. Righetti, "Balancing and walking using full dynamics lqr control with contact constraints," in *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*, 2016, pp. 63–68.

[14] F. L. Lewis, D. Vrabie, and V. L. Syrmos, *Optimal control*. John Wiley & Sons, 2012.

[15] F. A. Potra and J. Yen, "Implicit numerical integration for Euler-Lagrange equations via tangent space parametrization," *Journal of Structural Mechanics*, vol. 19, no. 1, pp. 77–98, 1991.

[16] W. F. Arnold and A. J. Laub, "Generalized eigenproblem algorithms and software for algebraic Riccati equations," *Proceedings of the IEEE*, vol. 72, no. 12, pp. 1746–1754, 1984.

[17] F. Bourbonnais, P. Bigras, and I. A. Bonev, "Minimum-time trajectory planning and control of a pick-and-place five-bar parallel robot," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 740–749, 2015.

[18] "RAPI-MOD: Ultrafast scara robot." [Online]. Available: http://gridbots.com/rapi_mov.html

[19] R. Bordalba, L. Ros, and J. M. Porta, "Randomized planning of dynamic motions avoiding forward singularities," in *Advances in Robot Kinematics*, 2018, pp. 170–178.