Robotica (2007) volume 25, pp. 131–145. © 2007 Cambridge University Press
doi:10.1017/S0263574706003262 Printed in the United Kingdom

Image-based visual servoing schemes for nonholonomic mobile manipulators

Alessandro De Luca*, Giuseppe Oriolo and Paolo Robuffo Giordano

Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Via Eudossiana 18, 00184 Roma, Italy.

(Received in Final Form: December 11, 2006)

SUMMARY

We consider the task-oriented modeling of the differential kinematics of nonholonomic mobile manipulators (NMMs). A suitable NMM Jacobian is defined that relates the available input commands to the time derivative of the task variables, and can be used to formulate and solve kinematic control problems. When the NMM is redundant with respect to the given task, we provide an extension of two well-known redundancy resolution methods for fixed-base manipulators (Projected Gradient and Task Priority) and introduce a novel technique (Task Sequencing) aimed at improving performance, e.g., avoiding singularities. The proposed methods are applied then to the specific case of image-based visual servoing, where the NMM image Jacobian combines the interaction matrix and the kinematic model of the mobile manipulator. Comparative numerical results are presented for two case studies.

KEYWORDS: Mobile manipulators; Visual servoing; Kinematic control; Redundancy resolution.

1. Introduction

A mobile manipulator consists of an articulated arm mounted on a mobile platform. Since this mechanical arrangement combines the dexterity of the former with the workspace extension of the latter, it is clearly appealing for many applications, and, in particular, for service robotics.[1,2] The locomotion of the platform is typically obtained through wheels; however, legs (as in walking robots) or gas-jet actuators (for free-flying space robots) are also used. In this paper, we shall focus on the case of wheeled mobile manipulators.

Most wheeled vehicles are subject to nonholonomic constraints arising from the rolling without slipping of the wheels on the ground.[3] These constraints restrict the admissible generalized velocities and thus the instantaneous mobility of the platform, but do not affect the global accessibility of its configuration space, which is still guaranteed through suitable maneuvering with a reduced set of independent velocity commands (the so-called pseudovelocities). On the other hand, fixed-base manipulators are kinematically unconstrained systems, in that arbitrary joint velocities can be imposed at each configuration; at the end-effector level, a loss of instantaneous mobility is only experienced at singular configurations.

A nonholonomic mobile manipulator (NMM) is obtained when the platform providing mobility to the manipulator base is a nonholonomic vehicle. The resulting structure clearly inherits the constraint on the admissible platform velocities—instantaneous mobility in the configuration space is still limited. However, the number of degrees of freedom (dofs) is also increased, typically yielding kinematic redundancy with respect to standard tasks, such as end-effector placement. This means that in general the instantaneous mobility is not constrained at the task level, in spite of the nonholonomic nature of the system. Moreover, the extra dofs can be used to optimize performance criteria and/or to perform auxiliary operations in addition to the primary task.

The study of task-oriented differential kinematics of NMMs has been addressed in the literature following two basic approaches. The first is simply to add the nonholonomic constraints to the relationship between the task velocities and the NMM generalized velocities, resulting in an implicit kinematic model that focuses on the instantaneous loss of mobility.[4,5] Alternatively, we have proposed in ref. [6] a more efficient formulation that explicitly maps the available command inputs (platform pseudovelocities and manipulator joint velocities) to the achievable task velocities; see also refs. [7, 8].

There are two conceptually different problems that can be formulated on the basis of the NMM differential kinematics. The first is *kinematic inversion*, i.e., generating velocity commands that realize an assigned task trajectory, provided that the initial task error is zero. If the NMM is kinematically redundant with respect to a given task, a *redundancy resolution* problem has to be addressed, which can be solved by appropriately extending techniques originally proposed for fixed-base manipulators.[9] The resulting methods are based either on task space augmentation[4,10,11] or on local optimization of cost functions.[6,12]

Similarly to what is done for single-body mobile robots or fixed-base manipulators, the second use of the NMM differential kinematics is to devise *kinematic control* schemes. In this framework, the velocity commands are actually considered as control inputs, and they are designed so as to achieve trajectory tracking or set-point regulation at the task level by combining a feedforward and a feedback action. The presence of the latter allows to recover nonzero initial task errors as well as counteract model perturbations, including numerical drifts.

* Corresponding author. E-mail: deluca@dis.uniroma1.it

The separation between kinematic and dynamic issues entailed by kinematic control approaches is based on the idea that, for a fully actuated mechanical system, any differentiable velocity profile can be realized by an appropriate force/torque controller. It is, therefore, possible to first design velocity commands on the basis of the differential kinematic model, and then compute (e.g., by feedback linearization and backstepping[13]) the dynamic-level inputs that track these commands. Under the assumption that the momentum of the mobile manipulator is limited (i.e., its velocity and/or inertia are low), this decoupled approach will typically result in reasonable actuator demands. In the presence of kinematic redundancy, it is also possible to include the optimization of dynamic performance criteria within a kinematic control problem, e.g., see ref. [14].

When robots operate in unstructured environments (as can be expected for mobile manipulators), it is essential to include exteroceptive sensory information in the control loop. In particular, the use of visual feedback from an onboard camera guarantees accurate positioning, robustness to calibration uncertainties, and reactivity to environmental changes.[15,16] A powerful approach is represented by *visual servoing*, i.e., the specification of a robotic task in terms of image features extracted from a target object and their use for controlling the robot/camera motion through the scene. Two basic approaches have been proposed to deal with visual tasks, namely *position-based visual servoing* (PBVS) and *image-based visual servoing* (IBVS).[16,17]

In PBVS, the image features are processed in order to estimate the relative 3D pose between the camera and the target, which is then used as an error signal for controlling the motion of the robot/camera system toward its desired goal.[18] In IBVS, the error is directly computed in terms of the features, whose motion on the image plane is related to the velocity twist of the camera via the *interaction matrix*. The advantages of IBVS over PBVS are the following: (i) a 3D model of the target is not needed; (ii) performance is robust with respect to perturbations of the robot/camera models, in particular, to calibration errors[19]; (iii) it is easier to devise feature-based motion strategies aimed at keeping the target always in the field of view of the camera.[20] However, there are also some drawbacks to be considered. Apart from situations where the interaction matrix loses rank during the motion, local minima of the task error function[21] may be encountered when trying to impose an (unfeasible) independent motion to a large number of image features.[22] Moreover, the feature depths are unknown in a pure IBVS setting, and must be estimated during servoing in order to compute the interaction matrix (a common choice is to simply use their constant value at the desired pose). Thus, only local stability can be guaranteed for most IBVS schemes.[23]

To alleviate some of the above problems, a number of *hybrid* schemes have been recently proposed.[24–26] In these methods, 3D information (usually obtained from epipolar geometry) is used to control a subset of the camera configuration vector, while the remaining dofs are regulated through an IBVS scheme.

In most works on visual servoing it is assumed, sometimes implicitly, that the camera has full mobility, which means that its velocity twist may be arbitrarily specified at nonsingular configurations (see, e.g., refs. [20, 27]). The visual servoing problem is usually formulated and solved in terms of the camera motion, which is then executed by the robot arm. However, this strategy may not be convenient in general, and, in particular, for two conceptually opposite cases. The first is when the dimension of the visual task exceeds the motion dofs of the camera, e.g., when a 6-dimensional task is specified for a camera/robot system having less than 6 dofs. In this case, a solution that directly relates feature velocities to the actual dofs of the robot and solves the visual servoing problem at that level is computationally more advantageous. For instance, this is what has been done for nonholonomic mobile robots with an onboard fixed camera in ref. [28]. The second case occurs when the visual task is underdimensioned with respect to the camera dofs, so that there are an infinity of camera motions which achieve the task. In addition, the robot manipulator may be itself redundant for the camera positioning task. This distributed redundancy is better exploited in an integrated way at the level of the robot dofs, where it can be effectively used for optimizing configuration-dependent criteria, such as singularity indices, distance from obstacles, or dynamic cost functions.

As a result, with the exception of special situations (6-dof robots executing 6-dimensional tasks), the visual servoing problem should be addressed in terms of velocity commands to the robot, as opposed to the camera velocity twist. This is particularly true for NMMs, which typically possess a redundant number of dofs with respect to the required task. Since the nonholonomy of the platform implies that there are less velocity commands available than generalized coordinates, the concept of redundancy should be properly defined. In the literature, these aspects have been rarely considered; one exception is ref. [29], where the case of a car-like robot carrying a 3R manipulator with a camera mounted on the end-effector was considered. While that work shares the same formulation as ours, it differs in the fact that kinematic redundancy is not addressed.

In this paper, the general class of NMMs is considered. Along the lines of ref. [6], we first introduce a task-oriented differential kinematic model which maps the robot velocity commands to the task velocities and can therefore be used for kinematic inversion as well as kinematic control (Section 2). Based on the introduced NMM Jacobian and with reference to the case of kinematic redundancy, we revisit classical inversion/control techniques such as the Projected Gradient (PG) and the Task Priority (TP) methods[9,30], and present a novel Task Sequencing (TS) strategy, which introduces an additional "artificial" redundancy in the control problem (Section 3). The proposed framework is then applied in Section 4 to the case of visual tasks for NMMs carrying a camera on the end-effector of the arm (*eye-in-hand* configuration). In particular, by defining the image features as task variables, we compute the NMM image Jacobian and show that it does not depend on the mobile platform configuration — a relevant property which allows us to design IBVS schemes based on the previous kinematic control developments. Essential to this step is also the use of a recently developed online estimation algorithm for the feature depths.[31] The design methodology and the

performance of the resulting IBVS schemes are illustrated with two case studies, by considering a two-wheeled differentially-driven platform (with unicycle kinematics) carrying, respectively, a 3R (elbow-type) and a 2R (polar) manipulator arm with an eye-in-hand camera (Section 5).

2. Kinematic Modeling of NMMs

Consider a robotic system made of a nonholonomic mobile platform carrying a manipulator. Let the configuration vector be $q = [q_{\rm p}^{\rm T}\ q_{\rm m}^{\rm T}]^{\rm T}$, where $q_{\rm p} \in \mathbb{R}^{n_{\rm p}}$ and $q_{\rm m} \in \mathbb{R}^{n_{\rm m}}$ are the generalized coordinates of the platform and of the manipulator, respectively. Assume that the task is described in terms of a vector r of s scalar variables. Possible choices for r include the position/orientation of the NMM end-effector in a positioning task, or the location of image features in a visual task. The task variables r are related to the configuration variables q of the NMM by the kinematic map

$$r = f(q_{\rm p},\ q_{\rm m}), \qquad r \in \mathbb{R}^s. \tag{1}$$

The kinematic model of the nonholonomic mobile platform can be expressed as the driftless system

$$\dot{q}_{\rm p} = G(q_{\rm p})u_{\rm p}, \qquad u_{\rm p} \in \mathbb{R}^{\rm p}, \qquad p < n_{\rm p}, \tag{2}$$

where $u_{\rm p}$ are the platform velocity commands (pseudovelocities) and the columns of $G(q_{\rm p})$ span the admissible velocity space at each platform configuration. On the other hand, the manipulator arm is a kinematically unconstrained system, i.e., vector $\dot{q}_{\rm m}$ can be arbitrarily specified at any arm configuration. Thus, we set

$$\dot{q}_{\rm m} = u_{\rm m}, \qquad u_{\rm m} \in \mathbb{R}^{n_{\rm m}}, \tag{3}$$

where $u_{\rm m}$ are the manipulator joint velocity commands.

By differentiating Eq. (1) w.r.t. time, and using Eq. (2), we get

$$\begin{aligned}\dot{r} &= \frac{\partial f}{\partial q_{\rm p}}\dot{q}_{\rm p} + \frac{\partial f}{\partial q_{\rm m}}\dot{q}_{\rm m} = J_{\rm p}(q)G(q_{\rm p})u_{\rm p} + J_{\rm m}(q)u_{\rm m} \\ &= [J_{\rm p}(q)G(q_{\rm p})\ J_{\rm m}(q)]\begin{bmatrix} u_{\rm p} \\ u_{\rm m}\end{bmatrix} = J(q)u,\end{aligned} \tag{4}$$

where $u = [u_{\rm p}^{\rm T}\ u_{\rm m}^{\rm T}]^{\rm T} \in \mathbb{R}^{p+n_{\rm m}}$. Equation (4) is the task-oriented kinematic model of the NMM, relating the instantaneous mobility of whole mobile manipulator to the velocity of the task variables. The $s \times (p + n_{\rm m})$ matrix J in Eq. (4) will be simply referred to as the *NMM Jacobian* for the given task. However, strictly speaking, some elements of J are not partial derivatives, due to the nonholonomic constraint entailed by Eq. (2). Note that, due to the nonholonomy of the platform, the total number of available commands is always less than the number of the NMM generalized coordinates.

The operational advantage of this formulation is that J can be used just as a standard Jacobian matrix in all classical problems addressed for fixed-base manipulators, like kinematic inversion, singularity analysis, or kinematic control. Hereafter, we shall focus on the latter — see ref. [6] for an overview of other problems.

3. Kinematic Control for NMMs

Assume that a reference signal r_d is specified for the task variables r. If $r_d = r_d(t)$, one has a *tracking* problem, whereas a *regulation* problem is assigned when r_d is constant. In the following, we shall use u as the control input for the NMM, assuming that low-level (direct) controllers are in charge of imposing the corresponding velocities to the platform/manipulator system, complying at a fast scale with the NMM dynamics—a classical approach known as *kinematic control*. Therefore, we design kinematic control laws using the task-oriented differential kinematics (4), which should now be viewed as the input–output representation of a driftless control system.

3.1. The nonredundant case

Consider first the case $p + n_{\rm m} = s$, in which the NMM Jacobian is square. Outside singularities, we can realize any task velocity by setting

$$u = J^{-1}(q)\dot{r}. \tag{5}$$

The problem reduces then to choosing $\dot{r}$ in such a way that the control problem is solved.

For a tracking problem, one simply sets in Eq. (5)

$$\dot{r} = \dot{r}_d + K(r_d - r) = \dot{r}_d + Ke, \tag{6}$$

where $K > 0$ is a (diagonal) gain matrix and $e \in \mathbb{R}^s$ is the task error. For a regulation problem, where $\dot{r}_d = 0$, we let

$$\dot{r} = Ke. \tag{7}$$

In both cases, the closed-loop system is described by

$$\dot{e} = -Ke, \tag{8}$$

so that exponential convergence to zero is achieved for each component of the task error.

For regulation tasks, it is also possible to use, in place of Eqs. (5) and (7), the computationally lighter expression

$$u = J^{\rm T}(q)Ke, \tag{9}$$

which still guarantees asymptotic (but no longer exponential) convergence of the task error to zero.[32]

3.2. The redundant case

Assume now that $p + n_{\rm m} > s$, so that the NMM Jacobian matrix has more columns than rows. Since the number of velocity commands (i.e., of dofs) exceeds the dimension of the task, we say that the NMM is *kinematically redundant* with respect to the given task.[1] To compute velocity

[1] One may also define a *static* redundancy property, which occurs when the number of generalized coordinates of the NMM exceeds the dimension of the task ($n_{\rm p} + n_{\rm m} > s$). Clearly, due to the

commands u that realize a given task velocity $\dot{r}$, it is possible to extend redundancy resolution methods originally developed for standard manipulators[9], using the NMM Jacobian in Eq. (4) as a starting point.

In the classical *Projected Gradient* (PG) method, all solutions of Eq. (4) are expressed as

$$u = J^{\dagger}(q)\dot{r} + (I - J^{\dagger}(q)J(q))u_0, \tag{10}$$

where $J^{\dagger}$ is the unique pseudoinverse of the NMM Jacobian, $I - J^{\dagger}J$ is the orthogonal (and symmetric) projection operator into the null-space of J, $\mathcal{N}(J)$, and $u_0 \in \mathbb{R}^{p+n_{\mathrm{m}}}$ is an arbitrary vector usually chosen so as to optimize a given criterion $H(q)$. In general, $J^{\dagger}$ can be computed from the Singular Value Decomposition of J. If rank $(J) = s$, it is $J^{\dagger} = J^{T}(JJ^{T})^{-1}$.[34]

In Eq. (10), we shall use again Eq. (6) or, respectively, Eq. (7), depending on the tracking or regulation nature of the problem. The closed-loop system will be described as before by the task error dynamics $\dot{e} = -Ke$, with the additional presence of an internal dynamics which is unobservable at the output level, i.e., in the task space.

For fixed-base manipulators, $\dot{q}$ and u coincide, and one can directly set $u_0 = \pm\alpha\nabla_q H(q)$ in Eq. (10), where $\alpha > 0$ is a suitable stepsize in the direction of the gradient of H. Care is required when devising a similar scheme in the NMM case, because the available command vector u has a lower dimension than $\dot{q}$. By differentiating $H(q)$ w.r.t. time and using Eqs. (2–3), we have

$$\begin{aligned}\dot{H} &= \frac{\partial H(q)}{\partial q_{\mathrm{p}}} G(q_{\mathrm{p}})u_{\mathrm{p}} + \frac{\partial H(q)}{\partial q_{\mathrm{m}}} u_{\mathrm{m}} \\ &= \nabla_q^T H(q) \begin{bmatrix} G(q_{\mathrm{p}}) & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} u_{\mathrm{p}} \\ u_{\mathrm{m}} \end{bmatrix}.\end{aligned}$$

Therefore, the choice of u_0 that locally realizes the best improvement of $H(q)$ is

$$u_0 = \pm\alpha \begin{bmatrix} G^{\mathrm{T}}(q_{\mathrm{p}}) & 0 \\ 0 & I \end{bmatrix} \nabla_{\mathrm{q}} H(q). \tag{11}$$

In the neighborhood of singular points of the NMM Jacobian, the use of the PG method (10) may result in very high velocity commands which cannot be realized by the low-level actuator controllers. One way to deal with this problem is to use the *Task Priority* (TP) technique.[30] The idea is to reorder the task vector r into μ subtasks $(r_1, \ldots, r_\mu)$, each of dimension s_i $(i = 1, \ldots, \mu$, with $\sum s_i = s)$, and to consider r_i as a task with higher priority than r_j if $i < j$. This is associated to a task-oriented partition of the NMM Jacobian in Eq. (4) of the form

$$\begin{bmatrix} \dot{r}_1 \\ \vdots \\ \dot{r}_\mu \end{bmatrix} = \begin{bmatrix} J_1(q) \\ \vdots \\ J_\mu(q) \end{bmatrix} u.$$

The TP method is formulated in such a way that, whenever J is rank deficient, the lowest priority tasks are relaxed while correctly executing those with highest priority.

For illustration, consider a regulation problem in which the task r is partitioned in $\mu = 2$ subtasks:

$$\begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \end{bmatrix} = \begin{bmatrix} J_1(q) \\ J_2(q) \end{bmatrix} u. \tag{12}$$

By solving the first set of s_1 equations, we get

$$u = J_1^{\dagger}\dot{r}_1 + (I - J_1^{\dagger}J_1)u_0, \tag{13}$$

where $\dot{r}_1 = K_1(r_{1d} - r_1) = K_1 e_1$, with $K_1 > 0$. If rank $J_1 = s_1$ throughout the motion, Eq. (13) ensures that the primary task variables r_1 converge exponentially to their desired values. The auxiliary command u_0 can be chosen so as to minimize the weighted norm of the secondary task error, i.e., the scalar function

$$H_2(q) = \frac{1}{2}e_2^T K_2 e_2 = \frac{1}{2}(r_{2d} - r_2)^{\mathrm{T}} K_2 (r_{2d} - r_2), \quad K_2 > 0. \tag{14}$$

The time derivative of Eq. (14) is

$$\begin{aligned}\dot{H}_2 &= -e_2^T K_2 \dot{r}_2 = -e_2^T K_2 J_2 u \\ &= -e_2^T K_2 J_2 J_1^{\dagger} K_1 e_1 - e_2^T K_2 J_2 (I - J_1^{\dagger}J_1)u_0.\end{aligned}$$

By using the symmetry of $I - J_1^{\dagger}J_1$, we conclude that the choice

$$u_0 = (I - J_1^{\dagger}J_1)J_2^T K_2 e_2 \tag{15}$$

provides the maximum reduction of H_2, subject to the satisfaction of the primary task. By plugging Eq. (15) into Eq. (13), and using the idempotency of $I - J_1^{\dagger}J_1$, we can write the TP kinematic control law as

$$u = J_1^{\dagger}K_1 e_1 + (I - J_1^{\dagger}J_1)J_2^T K_2 e_2. \tag{16}$$

The generalization of Eq. (16) to tracking problems is straightforward but computationally more cumbersome.[9] Different options for interchanging transpose and pseudoinverse of the involved Jacobians and the convergence properties of the associated schemes have been considered in ref. [30].

3.3. Task sequencing for regulation problems

For regulation problems, an alternative way to drive all the task variables to their set-points is to follow a *Task Sequencing* (TS) approach. The idea is to process the μ

nonholonomy of the system, kinematic redundancy implies static redundancy but the opposite is not true. Static redundancy in NMMs is relevant, e.g., when searching for collision-free configurations in motion planning problems with obstacles.[33]

subtasks one at a time. The s_1 subtask variables r_1 are regulated first, then the s_2 subtask variables r_2 are driven toward their desired value without changing r_1, then the s_3 subtask variables r_3 are brought to their desired value without changing r_1 and r_2, and so on. With this approach, an "artificial" redundancy is introduced in the kinematic control process: in particular, the degree of redundancy during the execution of the sequence will be $p + n_{\rm m} - s_1$ in the first phase, $p + n_{\rm m} - (s_1 + s_2)$ in the second phase, and so on. In the last phase, the redundancy degree will be back to its (possibly zero) original value $p + n_{\rm m} - s \geq 0$. This method applies to NMMs that are either redundant or nonredundant w.r.t. the task.

Consider for simplicity the case of $\mu = 2$ subtasks, and assume $p + n_{\rm m} = s = s_1 + s_2$, i.e., the nonredundant case for the global task. Define the two-phase task sequence

$$\begin{cases} \dot{r}_I = \dot{r}_1 = K_1 e_1, & t \in [0,\ T_1] \\ \dot{r}_{II} = \begin{bmatrix} 0 \\ \dot{r}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ K_2 e_2 \end{bmatrix}, & t \in [T_1,\ T_2], \end{cases}$$

where T_i $(i = 1, 2)$ denotes the time of completion of task i, i.e., such that a suitable termination condition is reached.[2] This leads to the kinematic control scheme

$$\begin{cases} u_I = J_1^{\dagger} K_1 e_1 + (I - J_1^{\dagger} J_1) u_0, & t \in [0,\ T_1] \\ u_{II} = (I - J_1^{\dagger} J_1) J_2^T K_2 e_2, & t \in [T_1,\ T_2], \end{cases} \tag{17}$$

where u_0 can be chosen as in Eq. (11) for optimization purposes.

In this basic version, the first task variable r_1 is no longer controlled during the second phase, so that its value may drift, e.g., due to linearization errors (recall that we are using the Jacobian matrix which is a mapping between tangent spaces). One way to counteract this effect is to replace the second-phase control law in Eq. (17) with

$$u_{II} = (I - J_1^{\dagger} J_1) J_2^T K_2 e_2 + J_1^{\dagger} K_1 e_1, \qquad t \in [T_1,\ T_2],$$

i.e., by adding a feedback term aimed at rejecting perturbations on e_1. This second phase of the TS approach becomes then identical to the TP method (16), with the notable difference that e_1 is already very small as a result of the first phase.

The extension these formulas to the case $\mu > 2$ and/or $p + n_{\rm m} > s$ (redundant case) is straightforward. It is also easy to prove that the TS strategy guarantees convergence of the task variables to their set-point (within an arbitrarily small tolerance), provided that the Jacobian of stacked subtasks $\bar{J}_i = [J_1^{\rm T}\ J_2^{\rm T} \ldots J_i^{\rm T}]^{\rm T}$ has full rank during the ith phase of the sequence.

The potential advantage of the TS kinematic control strategy over the classical approach which tries to drive all task variables *simultaneously* to their set-point rests on the artificial redundancy introduced in all but the last phase of the sequence. This can be used to optimize performance indices, and, in particular, to stay away from singular configurations (see Section 5.2). Finally, we mention that a somewhat related TS concept has been proposed in ref. [36], where a layered controller architecture reactively builds and executes a stack of subtasks arising as part of a given (visual) task.

[2] For instance, the termination condition $||r_{id} - r_i(T_i)|| < \epsilon$, for a given $\epsilon > 0$, will result in an arbitrarily small final error. Finite-time convergence to the set-point can be easily obtained by using a *terminal controller*[35] to define each phase of the task sequence.

4. Image-Based Visual Servoing for NMMs

We apply next our approach to the particular case of visual tasks, leading to an IBVS method for NMMs. For this, we recall first the common modeling assumptions of pin-hole cameras.

4.1. Camera model

With reference to Fig. 1, consider a world reference frame $\mathcal{F}_O : \{O;\ \vec{X}_O,\ \vec{Y}_O,\ \vec{Z}_O\}$ and the moving frame $\mathcal{F}_{\rm C} : \{O_{\rm C};\ \vec{X}_{\rm C},\ \vec{Y}_{\rm C},\ \vec{Z}_{\rm C}\}$ associated to a pin-hole camera, with $Z_{\rm C}$ coincident with the camera optical axis. The image plane, perpendicular to the optical axis, lies at a distance λ (the focal length) from $O_{\rm C}$, and is endowed with a 2D reference frame $\mathcal{F}_I : \{O_I;\ \vec{u},\ \vec{v}\}$ with axes parallel to $\vec{X}_{\rm C}$ and $\vec{Y}_{\rm C}$, respectively. In the following, we will assume that all quantities are expressed in the camera frame $\mathcal{F}_{\rm C}$, unless otherwise stated.

An image feature is any real-valued quantity associated to a selected geometric primitive (e.g., the coordinates of a point, the area of an ellipse, the angular coefficient of a line, etc.) in the image plane. Given a generic vector of features $f \in \mathbb{R}^k$, the velocity twist $[V_c^{\rm T}\ \omega_c^{\rm T}]^{\rm T} \in \mathbb{R}^6$ of the camera (expressed in its own frame) is mapped to $\dot{f}$ by a $k \times 6$ matrix $J_f(f,\ \chi)$ called the *interaction matrix*

$$\dot{f} = J_f(f, \chi) \begin{bmatrix} V_c \\ \omega_c \end{bmatrix},$$

where χ is a vector representing 3D information associated to f. It is possible to determine the interaction matrix for many features of interest, see ref. [15] for the case of lines, planes, and circles, and refs. [37, 38] for the set of image moments. In the case of a 3D point feature P with

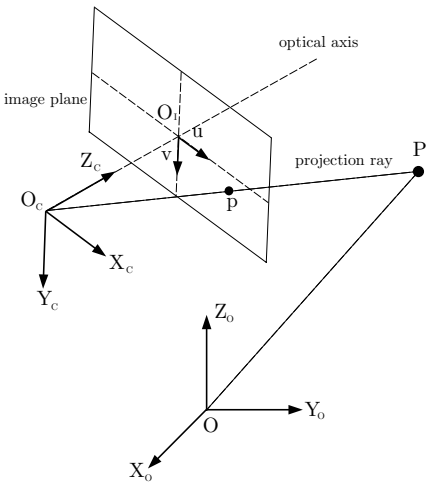

Fig. 1. World and camera frame definitions.

homogeneous coordinates $\bar{P} = [X\ Y\ Z\ 1]^{\mathrm{T}}$, its projection in the image plane is a 2D point feature p with homogeneous normalized coordinates $\bar{p} = [\bar{p}_u\ \bar{p}_v\ 1]^{\mathrm{T}} = [X/Z\ Y/Z\ 1]^{\mathrm{T}}$. The corresponding coordinates in pixels are denoted as $\widetilde{p} = [\widetilde{p}_u\ \widetilde{p}_v\ 1]^{\mathrm{T}} = A\bar{p}$, where A is a nonsingular matrix containing the camera internal parameters, i.e.,

$$A = \begin{bmatrix} \lambda k_u & -\lambda k_u/\tan\delta & u_0 \\ 0 & \lambda k_v/\sin\delta & v_0 \\ 0 & 0 & 1 \end{bmatrix},$$

with $[u_0\ v_0]^{\mathrm{T}}$ the coordinates of the principal point (in pixels), λ the focal length (in meters), k_u and k_v the magnifications in the $\overrightarrow{u}$ and $\overrightarrow{v}$ directions (in pixel/meters), and δ the angle between these axes. Assuming that matrix A is known, i.e., that the camera internal parameters are calibrated, it is possible to transform the measured point $\widetilde{p}$ back to $\bar{p} = A^{-1}\widetilde{p}$. In this 'normalized' space it is easy to derive the expression of the interaction matrix for a single point feature (see ref. [16] for an explicit derivation):

$$\begin{bmatrix} \dot{\bar{p}}_u \\ \dot{\bar{p}}_v \end{bmatrix} = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{\bar{p}_u}{Z} & \bar{p}_u\bar{p}_v & -\left(1+\bar{p}_u^2\right) & \bar{p}_v \\ 0 & -\frac{1}{Z} & \frac{\bar{p}_v}{Z} & 1+\bar{p}_v^2 & -\bar{p}_u\bar{p}_v & -\bar{p}_u \end{bmatrix} \begin{bmatrix} V_c \\ \omega_c \end{bmatrix} = J_f(\bar{p},\ Z) \begin{bmatrix} V_c \\ \omega_c \end{bmatrix}. \tag{18}$$

Note that, in the case of a point feature, the 3D information included in vector χ reduces simply to the point depth Z, which is an unknown parameter in the classical IBVS framework. In view of this, many proposed servoing schemes rely on some approximation of the actual value of Z (e.g., the constant value at the desired pose Z_d). Alternatively, we propose in ref. [31] an online algorithm for estimating Z during motion under the assumption that the target is fixed. In particular, by taking $\widehat{x} = [\widehat{p}_u\ \widehat{p}_v\ 1/\widehat{Z}]^{\mathrm{T}}$ as the current estimation of the partially unknown state $x = [\bar{p}_u\ \bar{p}_v\ 1/Z]^{\mathrm{T}}$ and $y = [\bar{p}_u\ \bar{p}_v]^{\mathrm{T}}$ as the measured output on the image plane, the update law takes the form of a nonlinear observer

$$\dot{\widehat{x}} = \alpha(\widehat{x},\ y) \begin{bmatrix} V_c \\ \omega_c \end{bmatrix} + \beta(\widehat{x},\ y,\ V_c,\ \omega_c), \tag{19}$$

where suitable $\alpha(\cdot)$ and $\beta(\cdot)$ guarantee that $||x(t) - \widehat{x}(t)||$ goes to zero exponentially, as long as the camera is moving with a nonzero linear velocity V_c. Thanks to this result, we will assume that the actual value of Z of each point feature is reconstructed online and made available during the servoing task.

4.2. Visual servoing tasks

We formulate the image-based visual servoing problem for an NMM by choosing as task variables r the vector $f = [\bar{p}_{u1}\ \bar{p}_{v1} \ldots \bar{p}_{uk}\ \bar{p}_{vk}]^{\mathrm{T}} \in \mathbb{R}^{2k}$ of the coordinates of k point features in the image plane. Therefore, the task dimension is $s = 2k$.

The *NMM image Jacobian* J_{image} (ie., the differential mapping between the velocity commands u and the features velocity $\dot{f}$ at a given robot configuration) is simply the specific Jacobian of this task in Eq. (4). However, the computation is simplified by observing that

$$J_{\mathrm{image}} = J_v J_c, \tag{20}$$

where:

– the $6 \times (p + n_{\mathrm{m}})$ matrix J_c gives the linear and angular velocity pair (V_c, ω_c) of the camera mounted on the end-effector, in response to the NMM commands u

$$\begin{bmatrix} V_c \\ \omega_c \end{bmatrix} = J_c(q)u;$$

– the $2k \times 6$ interaction matrix J_v is the stack of k Jacobians J_f of the form (18), each one accounting for one point feature

$$\dot{f} = J_v(f,\ Z) \begin{bmatrix} V_c \\ \omega_c \end{bmatrix} = \begin{bmatrix} J_{f_1}(f_1,\ Z_1) \\ \vdots \\ J_{f_k}(f_k,\ Z_k) \end{bmatrix} \begin{bmatrix} V_c \\ \omega_c \end{bmatrix}, \tag{21}$$

being $Z = [Z_1 \ldots Z_k]^{\mathrm{T}} \in \mathbb{R}^k$ the vector of the depths associated to the k feature points.

4.2.1. Derivation of J_c. For a fixed-base manipulator arm, the pair $(V_c,\ \omega_c)$ is related to $\dot{q}$ through the geometric (basic) Jacobian J_g.[32] For an NMM, this velocity twist is obtained from a suitable camera-oriented NMM Jacobian J_c according to the method of Section 2. In the eye-in-hand case, the velocity twist of the camera coincides with the linear and angular velocity of the NMM end-effector.

Hence, we choose the six-dimensional intermediate task vector

$$r_c = [t_x\ t_y\ t_z\ \phi_1\ \phi_2\ \phi_3]^{\mathrm{T}} = [t^{\mathrm{T}}(q)\ \phi^{\mathrm{T}}(q)]^{\mathrm{T}}, \tag{22}$$

describing the position and orientation of the camera w.r.t. $\mathcal{F}_O$, where ϕ represents a minimal parametrization of the camera orientation by Euler angles. Similarly to Eq. (4), we can compute the linear and angular velocity of the NMM end-effector by

$$\begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & T(\phi) \end{bmatrix} \begin{bmatrix} \dot{t} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & T(\phi) \end{bmatrix} \begin{bmatrix} J_{t\mathrm{p}}G & J_{t\mathrm{m}} \\ J_{\phi\mathrm{p}}G & J_{\phi\mathrm{m}} \end{bmatrix} u, \tag{23}$$

where $T(\phi)$ is the transformation matrix between $\dot{\phi}$ and the camera angular velocity ω, and $J_{t\mathrm{p}} = \partial t/\partial q_{\mathrm{p}}$, $J_{t\mathrm{m}} = \partial t/\partial q_{\mathrm{m}}$, $J_{\phi\mathrm{p}} = \partial\phi/\partial q_{\mathrm{p}}$, $J_{\phi\mathrm{m}} = \partial\phi/\partial q_{\mathrm{m}}$. By means of the rotation matrix R_c, representing the orientation of $\mathcal{F}_{\mathrm{C}}$ with respect to $\mathcal{F}_O$, the pair $(V,\ \omega)$ is then expressed in the camera frame $\mathcal{F}_{\mathrm{C}}$. We get

$$\begin{bmatrix} V_c \\ \omega_c \end{bmatrix} = \begin{bmatrix} R_c^{\mathrm{T}}(q) & 0 \\ 0 & R_c^{\mathrm{T}}(q) \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & T(\phi) \end{bmatrix} \begin{bmatrix} J_{t\mathrm{p}}(q)G(q_p) & J_{t\mathrm{m}}(q) \\ J_{\phi\mathrm{p}}(q)G(q_p) & J_{\phi\mathrm{m}}(q) \end{bmatrix} u = J_c(q_{\mathrm{m}})u. \tag{24}$$

It is worth noting that the Jacobian J_c does not depend on the platform absolute position/orientation q_p, but only on the manipulator variables q_m. The proof of this property, reported in the Appendix, exploits the fact that the velocity twist $(V_c,\ \omega_c)$ of the camera is expressed in the moving camera frame, and it would not hold if other reference frames were used (e.g., world or platform frames). As a result, the computation of $J_c(q_\mathrm{m})$ is independent of the mobile base absolute localization, which is typically obtained from noisy and possibly unreliable data processing algorithms (such as dead-reckoning). Examples of J_c are given in Section 5.

4.2.2. NMM image Jacobian. The final NMM image Jacobian is the $2k \times (p + n_m)$ matrix obtained by combining Eqs. (21) and (24) as

$$\dot{f} = J_v(f,\ Z)J_c(q_\mathrm{m})u = J_\mathrm{image}(f,\ Z,\ q_\mathrm{m})u. \qquad (25)$$

Note that the absolute position/orientation of the platform is still encoded in the values of f and Z upon which J_v depends. However, as explained in Section 4.1, f is measured directly on the image plane and Z is obtained from the processing of visual cues, see Eq. (19). Thus, the platform configuration q_p is completely removed from our control problem. Once the NMM image Jacobian is available, any of the kinematic control methods presented in Section 3 can be used to fulfill visual servoing tasks assigned to the NMM.

Some additional considerations on the singularities of J_image are in order. Even when both J_v and J_c are full rank (which is usually the case), matrix J_image may still be ill-conditioned, i.e., its smallest singular value may be very close to zero. In this case, a motion generation scheme based on inversion or pseudoinversion would require extremely large velocities in response to small task errors.

The NMM image Jacobian can actually lose rank when $g = \dim \mathcal{R}(J_c) - \dim(\mathcal{R}(J_c) \cap \mathcal{N}(J_v)) < 2k$, $\mathcal{R}(J_c)$ being the range space of matrix J_c. When this condition is encountered, it is not possible to impose an arbitrary (controlled) motion to *all* features on the image plane, but only to a subset of dimension g. In turn, any choice for this subset of g features will constrain the motion of the remaining $2k - g$ ones.

This problem cannot be avoided even if we reach a condition where $\mathcal{N}(J_v) = 0$, e.g., by selecting a number $2k > 6$ of features so that $\mathcal{R}(J_v) = 6$, while keeping $p + n_m > 2k$ (the NMM is still redundant for the visual task). Indeed, since in any case, rank $J_\mathrm{image} \leq 6$, the set $\mathcal{S} = \{\dot{f} \in \mathbb{R}^{2k} |\ \dot{f} \notin \mathcal{R}(J_\mathrm{image})\}$ will have dimension $d_\mathcal{S} \geq 2k - 6$, and thus there will still be $d_\mathcal{S}$ independent feature motions unrealizable by any motion of the camera (see ref. [22] for a more thorough analysis).

On the other hand, a convenient way to deal with $k \gg 6$ features is to consider some higher-level image descriptors such as the generalized image moments (area, barycenter, principal axes, etc.). By a proper choice of six of these moments, or of combinations of them, it is possible to derive a 6×6 interaction matrix with good decoupling properties among the linear and angular velocities of the camera.[37,38]

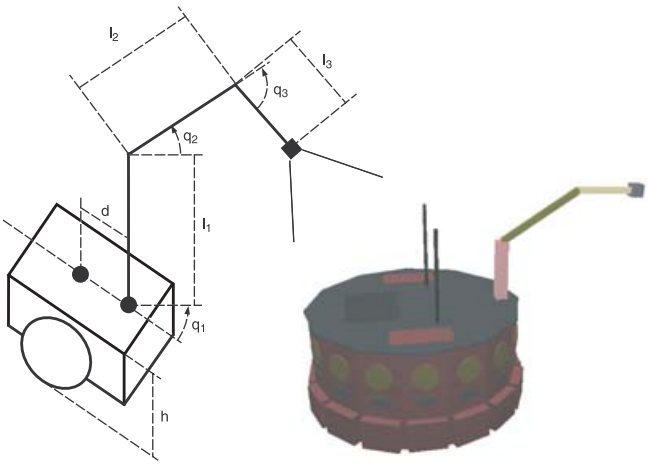

Fig. 2. Unicycle platform with a 3R elbow-type manipulator.

5. Case Studies

The task-oriented kinematic modeling of NMMs and the proposed design of associated IBVS kinematic control laws are illustrated by means of two case studies in which the mobile platform is a two-wheeled differentially-driven vehicle with unicycle kinematics. In the first case, the platform carries a 3R spatial manipulator with eye-in-hand camera and the NMM is redundant for the chosen visual task. In the second case study, a simpler 2R polar manipulator is considered and the NMM is nonredundant w.r.t. the given task. We shall see how the TP and TS strategies are effective in avoiding singularities during visual servoing. Recall that the feature depths needed to compute the interaction matrix are estimated online through the nonlinear observer (19) presented in detail in ref. [31]. Numerical simulations were performed in Webots.[3] Video clips showing the 3D motion of the NMMs in the different cases are available at the website http://www.dis.uniroma1.it/~labrob/research/NMM_Visual_Servoing.html.

5.1. Unicycle platform with 3R arm

5.1.1. Kinematic modeling. Consider an NMM made of a platform with unicycle kinematics carrying a 3R elbow-type manipulator arm, as shown in Fig. 2. For this robot, the configuration vector is $q = [q_\mathrm{p}^\mathrm{T}\ q_\mathrm{m}^\mathrm{T}]^\mathrm{T} \in \mathbb{R}^6$ with $q_\mathrm{p} = [x\ y\ \theta]^\mathrm{T} \in \mathbb{R}^3$ and $q_\mathrm{m} = [q_1\ q_2\ q_3]^\mathrm{T} \in \mathbb{R}^3$.

The kinematic model of the unicycle is described by

$$\dot{q}_\mathrm{p} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} = \widetilde{G}(q_\mathrm{p})\widetilde{u}_\mathrm{p}, \qquad (26)$$

where v and ω are the linear and angular velocity of the platform, and θ is its absolute orientation in the Cartesian plane. In order to obtain an homogeneous representation for the NMM commands, it is useful to rewrite Eq. (26) in terms of the actuated angular velocities $u_\mathrm{p} = [\dot{\phi}_\mathrm{R}\ \dot{\phi}_\mathrm{L}]^\mathrm{T}$ of the right

[3] Webots is a commercial mobile robot simulation software developed by Cyberbotics Ltd. (http://www.cyberbotics.com). Its simulation engine automatically takes into account the presence of image noise as well as motion disturbances.

and left wheel of the platform. This is done by means of the invertible transformation

$$\widetilde{u}_{\mathrm{p}} = Mu_{\mathrm{p}} = \begin{bmatrix} \frac{\rho}{2} & \frac{\rho}{2} \\ \frac{\rho}{w} & -\frac{\rho}{w} \end{bmatrix} u_{\mathrm{p}},$$

where ρ is the wheel radius and w is the axle length, leading to

$$\dot{q}_{\mathrm{p}} = \widetilde{G}(q_{\mathrm{p}})Mu_{\mathrm{p}} = G(q_{\mathrm{p}})u_{\mathrm{p}}.$$

For the manipulator, we take $\dot{q}_{\mathrm{m}} = u_{\mathrm{m}}$ as explained in Section 2. Hence, the velocity command input has dimension $p + n_{\mathrm{m}} = 5$.

With these settings, we obtain the following 6×5 matrix J_c

$$J_c = \begin{bmatrix} \frac{\rho\left(\frac{1}{2}ws_1 - dc_1 - l_3c_{23} - l_2c_2\right)}{w} & \frac{\rho\left(l_3c_{23} + \frac{1}{2}ws_1 + dc_1 + l_2c_2\right)}{w} & -l_2c_2 - l_3c_{23} & 0 & 0 \\ \frac{\rho(2d\,\bar{c}_{123} - 2dc_{123} + ws_{123} - w\bar{s}_{123})}{4w} & \frac{\rho(ws_{123} - w\bar{s}_{123} + 2dc_{123} - 2d\,\bar{c}_{123})}{4w} & 0 & -l_2c_3 - l_3 & -l_3 \\ \frac{\rho(wc_{123} + w\bar{c}_{123} + 2ds_{123} + 2d\,\bar{s}_{123})}{4w} & \frac{\rho(wc_{123} + w\bar{c}_{123} - 2d\,\bar{s}_{123} - 2ds_{123})}{4w} & 0 & l_2s_3 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ -\frac{\rho c_{23}}{w} & \frac{\rho c_{23}}{w} & -c_{23} & 0 & 0 \\ \frac{\rho s_{23}}{w} & -\frac{\rho s_{23}}{w} & s_{23} & 0 & 0 \end{bmatrix},$$

where s_{ijk} and c_{ijk} stand for $\sin(q_i + q_j + q_k)$ and $\cos(q_i + q_j + q_k)$, and $\bar{s}_{123}$, $\bar{c}_{123}$ stand for $\sin(q_1 - q_2 - q_3)$, $\cos(q_1 - q_2 - q_3)$, respectively. Following the analysis developed in ref. [6], it can be shown that this Jacobian is always full rank except when the second link is aligned with the vertical direction, i.e., $q_2 = \pm\pi/2$, where the rank of J_c drops to 4. Note that the presence of an offset $d \neq 0$ between the platform center and the manipulator base (see Fig. 2) is crucial for deleting some of the singularities that affect the manipulator taken alone. For the NMM geometric data, we have set $h = 0.13$, $d = 0.15$, $l_1 = 0.1$, $l_2 = 0.15$, $l_3 = 0.1$, $\rho = 0.1$, and $w = 0.25$ (all units are [m]).

5.1.2. Task description. As visual task, we consider the problem of regulating the position on the image plane of $k = 2$ point features taken from a fixed target. The features are extracted from two (red) markers placed on a cubic target, as shown in Fig. 3. The cube has sides of 0.1 [m], and the two markers are positioned at a distance of 0.03 [m] from their closest edges.

Fig. 3. A view of the target with the two markers.

The cube location in the scene can be described as follows. Consider a frame $\mathcal{F}_{\mathrm{R}}$:$\{O_{\mathrm{R}};\ \vec{X}_{\mathrm{R}},\ \vec{Y}_{\mathrm{R}},\ \vec{Z}_{\mathrm{R}}\}$ centered on the platform and aligned with the platform heading, i.e., with

$$\begin{cases} O_{\mathrm{R}} = [x\ y\ 0]^{\mathrm{T}} \\ \vec{X}_{\mathrm{R}} = [\cos\theta\ \ \sin\theta\ 0]^{\mathrm{T}} \\ \vec{Y}_{\mathrm{R}} = [-\sin\theta\ \ \cos\theta\ 0]^{\mathrm{T}} \\ \vec{Z}_{\mathrm{R}} = [0\ 0\ 1]^{\mathrm{T}}. \end{cases}$$

The pose of the target $(p_t, R_z(\alpha_t)) \in SE(3)$ can then be expressed with respect to frame $\mathcal{F}_{\mathrm{R}}$. Vector p_t points to the center of the cube, while $R_z(\alpha_t)$ is the 3×3 rotation matrix of an angle α_t around the world z-axis. When $\alpha_t = 0$, the cube face with the markers is parallel to the plane $(\vec{Y}_{\mathrm{R}}, \vec{Z}_{\mathrm{R}})$ and looks toward $-\vec{X}_{\mathrm{R}}$. As initial conditions, we have chosen

$$\begin{cases} p_t(t_0) = [1.149\ \ -0.392\ 0.314]^{\mathrm{T}}\ [\mathrm{m}] \\ \alpha_t(t_0) = 0.2473\ [\mathrm{rad}] \end{cases}$$

for the NMM (with respect to the fixed cube target) and

$$\begin{cases} q_1(t_0) = 0 \\ q_2(t_0) = 0.57 \\ q_3(t_0) = -0.57 \end{cases}$$

for the manipulator joint angles (in [rad]). As a result, the initial position of the point features on the image plane is

$$f(t_0) = [74.61 \quad -10.12\ 66.07 \quad -3.5]^{\mathrm{T}},$$

while their desired reference position is

$$f_d = [10.5 \quad -37.53 \quad -18.94 \quad -17.55]^{\mathrm{T}}.$$

As $s = 2k = 4$, the complete 4×5 NMM image Jacobian is computed as

$$\dot{f} = \begin{bmatrix} J_{f_1}(f_1,\ Z_1) \\ J_{f_2}(f_2,\ Z_2) \end{bmatrix} J_c(q_m)u = J_{\mathrm{image}}(f,\ Z,\ q_m)u.$$

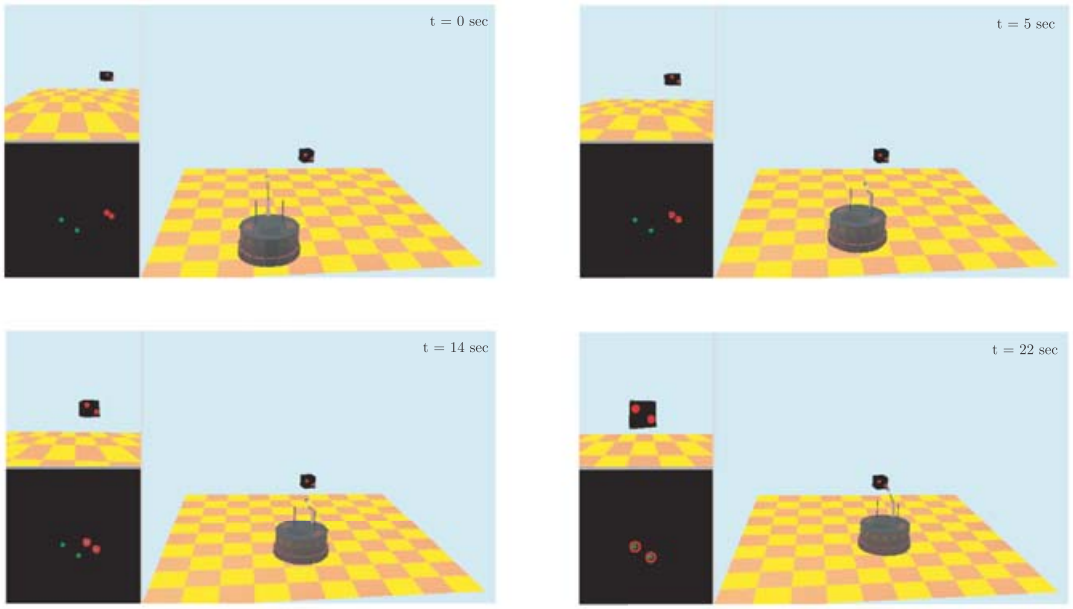

Fig. 4. Snapshots of the visual servoing task with the PG method.

The single degree of kinematic redundancy is used to avoid the singularities of J_c. This is obtained by minimizing the cost function

$$H_1(q) = \frac{1}{\cos^2 q_2},$$

which goes to infinity at the singularities $q_2 = \pm\pi/2$. For redundancy resolution, we adopted the PG method (10) with the control gain matrix $K = 0.2 \cdot I_{2\times 2}$ in Eq. (7) and a stepsize $\alpha = 1$ in Eq. (11), evaluated for $H = H_1(q)$ and using the negative sign.

5.1.3. Simulation results. In Fig. 4, four snapshots taken during the execution of the visual servoing task are shown. In each picture, the small upper-left window shows the current image acquired by the camera, while the lower-left window shows the same image after the feature extraction process. In the same window, two stationary (green) dots represent the desired positions of the features.

The left plot in Fig. 5 shows the path followed by the two feature points on the image plane, starting from the circular markers and ending at the triangle markers. As expected, due to the task decoupling properties of the PG method (with diagonal K), and since no singularities were encountered during visual servoing, the features move on the straight line connecting their initial positions to their goals. The time evolution of $H_1(q)$ in the right part of Fig. 5 confirms that the NMM has been kept away from singular configurations. Finally, the velocity commands of the NMM during the servoing are shown in Fig. 6. For the platform, we reported the $\widetilde{u}_p$ commands instead of the u_p, since linear and angular velocities have a more direct interpretation. We observe that the requested task motion is executed by distributing effort among both the platform and the manipulator.

5.2. Unicycle platform with 2R arm

5.2.1. Kinematic modeling. In this second case study, the NMM has the same previous platform but uses a 2R polar manipulator on board for carrying the camera, see Fig. 7. Note that this arrangement may cover the interesting situation of a wheeled mobile robot equipped with a pan-tilt camera (whose two dofs can be modeled as manipulator joints). In this case, we have $q_\mathrm{p} = [x\ y\ \theta]^\mathrm{T}$, $q_\mathrm{m} = [q_1\ q_2]^\mathrm{T}$, and thus $q \in \mathbb{R}^5$, while the actual commands are $u = [\dot{\phi}_R\ \dot{\phi}_L\ \dot{q}_1\ \dot{q}_2]^T \in$

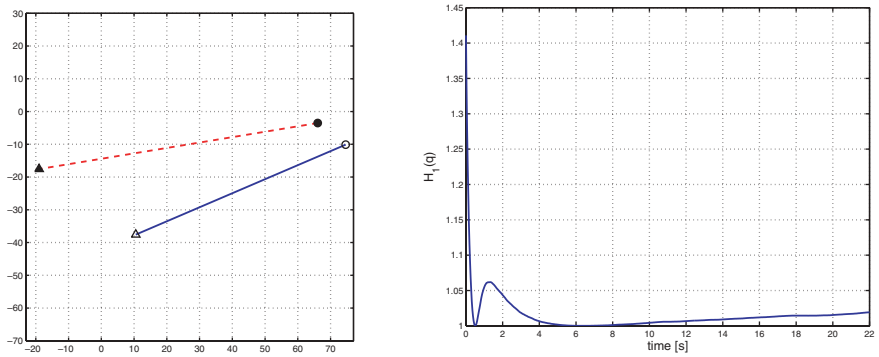

Fig. 5. PG method. *Left*: Motion of point features f_1 (*solid blue line*) and f_2 (*dashed red line*). *Right*: Constrained minimization of index $H_1(q)$.

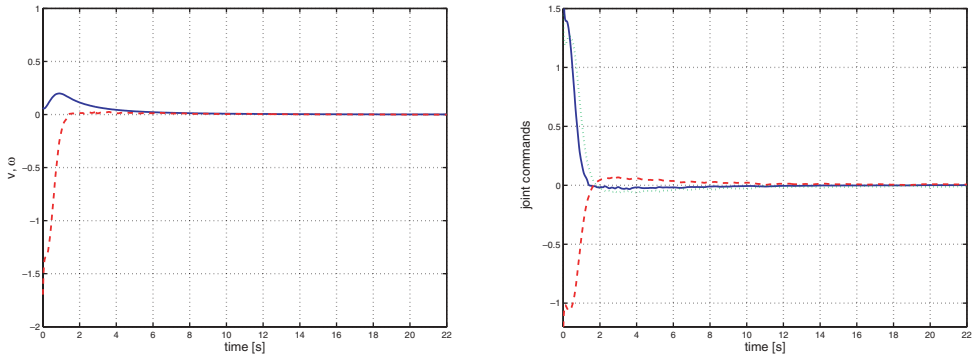

Fig. 6. NMM control commands with the PG method. *Left*: Linear velocity v (*solid blue line*) and angular velocity ω (*dashed red line*) of the platform. *Right*: Arm joint velocities $\dot{q}_1$ (*solid blue line*), $\dot{q}_2$ (*dashed red line*), and $\dot{q}_3$ (*dotted green line*).

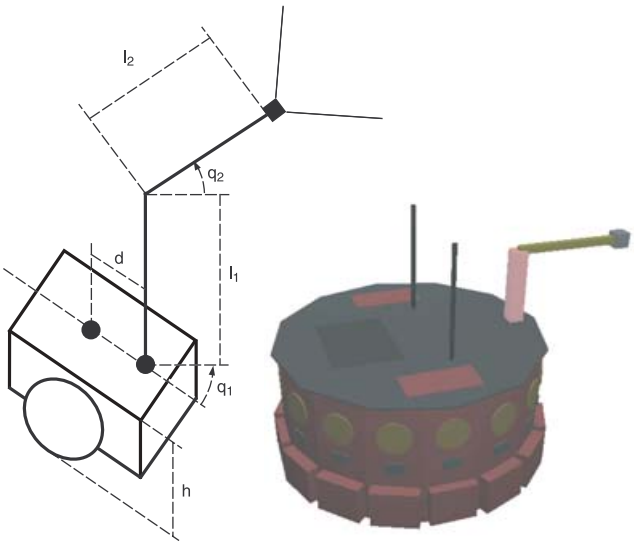

Fig. 7. Unicycle platform with a 2R polar manipulator.

$\mathbb{R}^4$. The geometric data of this NMM are the same as in Section. 5.1 (eliminating the third link).

The associated 6×4 matrix J_c is

$$J_c = \begin{bmatrix} \dfrac{-\rho\left(dc_1 + \frac{w}{2}s_1 - l_2c_2\right)}{w} & \dfrac{\rho\left(dc_1 + \frac{w}{2}s_1 + l_2c_2\right)}{w} & -l_2c_2 & 0 \\ \dfrac{\rho(2d\,\bar{c}_{12} - 2dc_{12} + ws_{12} - w\bar{s}_{12})}{4w} & \dfrac{\rho(-2d\,\bar{c}_{12} + 2dc_{12} + ws_{12} - w\bar{s}_{12})}{4w} & 0 & -l_2 \\ \dfrac{\rho\left(ds_{12} + d\,\bar{s}_{12} + \frac{1}{2}w\bar{c}_{12} + \frac{1}{2}wc_{12}\right)}{2w} & \dfrac{\rho\left(-ds_{12} - d\,\bar{s}_{12} + \frac{1}{2}w\bar{c}_{12} + \frac{1}{2}wc_{12}\right)}{2w} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -\dfrac{\rho c_2}{w} & \dfrac{\rho c_2}{w} & -c_2 & 0 \\ \dfrac{\rho s_2}{w} & -\dfrac{\rho s_2}{w} & s_2 & 0 \end{bmatrix},$$

with $s_{ij} = \sin(q_i + q_j)$, $c_{ij} = \cos(q_i + q_j)$, $\bar{s}_{ij} = \sin(q_i - q_j)$, and $\bar{c}_{ij} = \cos(q_i - q_j)$. It can be shown that this matrix has *always* full rank.

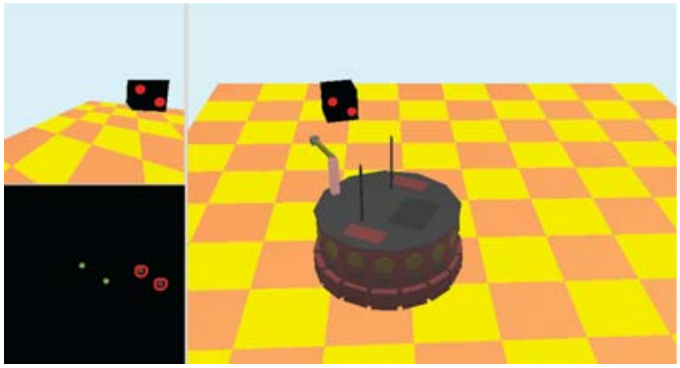

Fig. 8. A situation where the NMM image Jacobian is close to a singularity.

5.2.2. Task description. As in the previous case study, we choose to regulate the position of $k = 2$ point features, obtaining a 4×4 NMM image Jacobian. This NMM is no longer redundant for the given task, and visual servoing can be obtained using the scheme (5) with Eq. (7). However, inversion of the square NMM image Jacobian does not yield in general good results, since this matrix easily becomes ill-conditioned during the servoing task as discussed in Section 4.2.2. Consider, for example, the situation shown in Fig. 8, corresponding to the following initial conditions and

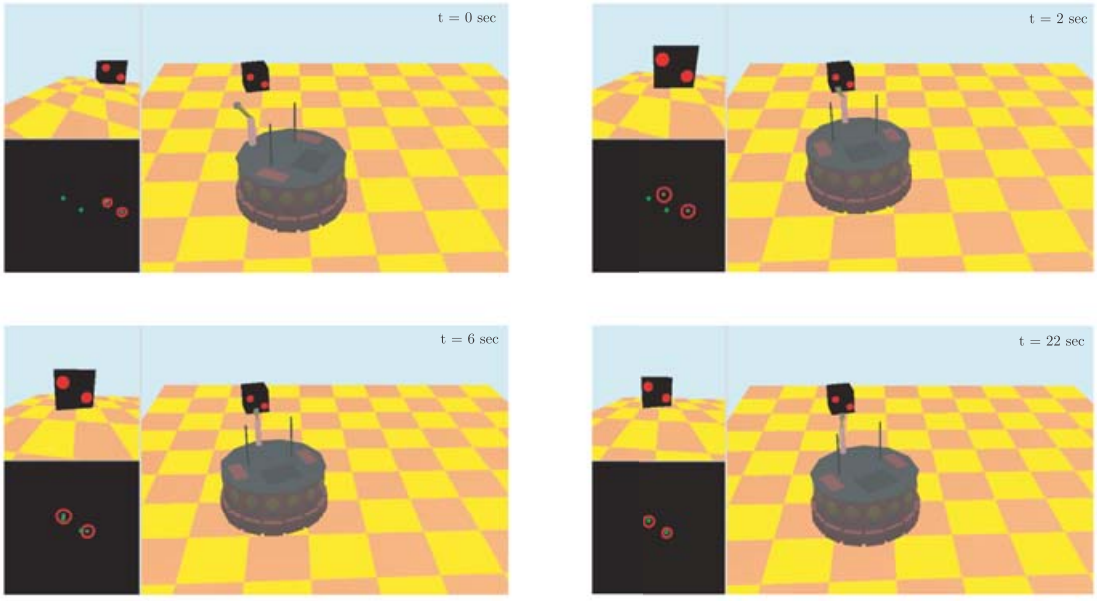

Fig. 9. Snapshots of the visual servoing task with the TP method.

desired position of the two point features:

$$\begin{cases} p_t(t_0) = [0.3956\ -0.453\ 0.2334]^{\mathrm{T}}\ [\mathrm{m}] \\ \alpha_t(t_0) = -0.8571\ [\mathrm{rad}] \\ q_1(t_0) = -0.7025\ [\mathrm{rad}] \\ q_2(t_0) = -0.1368\ [\mathrm{rad}] \\ f(t_0) = [93.09\ -12.02\ 65.57\ 6.17]^{\mathrm{T}} \\ f_d = [16.31\ -8.85\ -17.33\ 13.63]^{\mathrm{T}}. \end{cases}$$

Matrix J_{image} is very close to a singularity and its inversion will generate very large velocity commands for the NMM. However, the two 2×4 blocks J_1 and J_2 in

$$J_{\mathrm{image}} = \begin{bmatrix} J_1 \\ J_2 \end{bmatrix} = \begin{bmatrix} J_{f_1} J_c \\ J_{f_2} J_c \end{bmatrix},$$

which are associated to the positioning task of each single point feature, are well-conditioned in this configuration; in fact, the smallest singular values are $\sigma(J_{\mathrm{image}}) = 0.16$, $\sigma(J_1) = 198.75$, and $\sigma(J_2) = 196.01$, respectively. This implies that the robotic system is too constrained to impose an arbitrary velocity to both features, although it is still possible to move them individually. Therefore, it is interesting to test the performance of the TP and TS methods as they are mainly intended for cases when the (visual) task cannot be directly realized as a whole.

5.2.3. Simulation results. The TP method (12) has been applied with $K_1 = 3$ and $K_2 = 0.02$, and choosing $r_1 = f_1$, i.e., the primary task is the regulation of the lower-right point feature, and $r_2 = f_2$.

Figure 9 shows the motion of the NMM during visual servoing. In this case, the TP method is able to realize the task without encountering any singularity, but the motion of f_2 is no more on a straight line, as can be seen from Fig. 10. This is a direct consequence of the fact that regulation of f_2 is addressed as a secondary task, and thus combined feature motions that would not be realizable are handled by relaxing the constraints on f_2. Nonetheless, the norm of the error on the second feature $\|e_2(t)\| = \|f_2(t) - f_{2d}\|$ is still decreasing (right plot in Fig. 10).

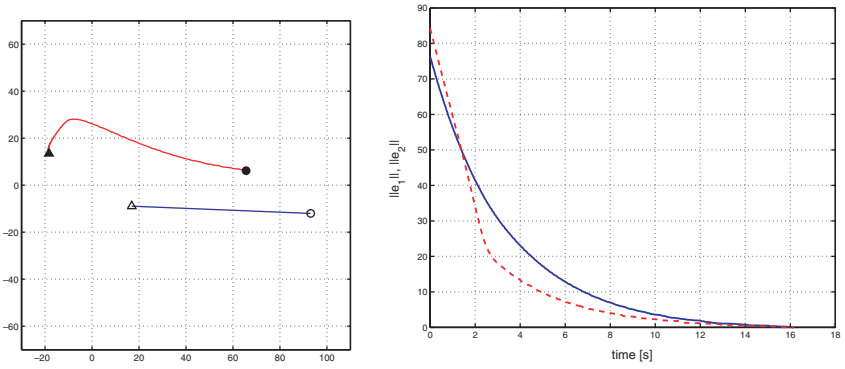

Fig. 10. TP method. *Left*: Motion of point features f_1 (*solid blue line*) and f_2 (*dashed red line*). *Right*: Task errors norms $||e_1||$ (*solid blue line*) and $||e_2||$ (*dashed red line*) versus time.

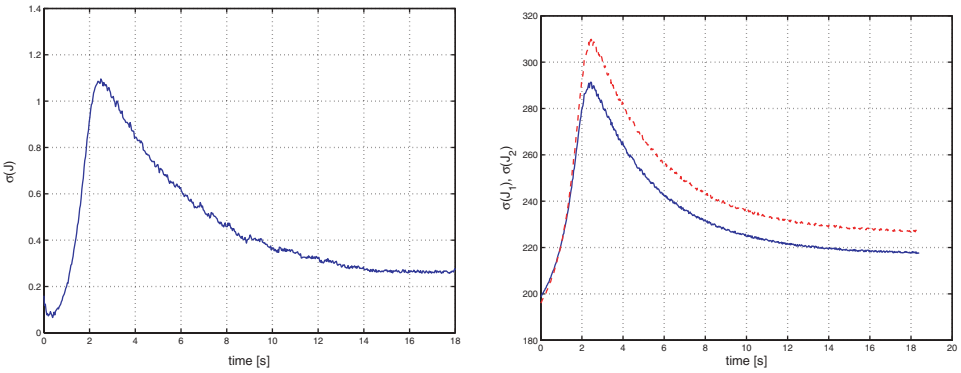

Fig. 11. Singularity analysis during the visual task. *Left*: Time evolution of the smallest singular value $\sigma(J_{\text{image}})$. *Right*: Smallest singular values $\sigma(J_1)$ (*solid blue line*) and $\sigma(J_2)$ (*dashed red line*).

In Fig. 11, we report the evolution of the smallest singular values $\sigma(J_{\text{image}})$, $\sigma(J_1)$, and $\sigma(J_2)$ during the servoing task. Note that matrix J_{image} remains always close to singularity, while J_1 and J_2 are well conditioned throughout the motion. This confirms that a direct inversion of J_{image} would not have provided satisfactory results. The velocity commands for the platform and the manipulator are shown in Fig. 12.

For comparison, we have applied also the TS method (17) and performed the visual task in two phases, achieving two degrees of redundancy during the first phase. In the reported simulation, we have chosen again $r_1 = f_1$ as the subtask to be realized during the first phase, and $r_2 = f_2$ as the completing subtask for the second phase. In order to keep the target as much as possible in front of the NMM, redundancy in the first phase has been used for minimizing the cost function

$$H_{TS}(q) = \frac{1}{2} q_1^2.$$

Denoting by $u_{H_{TS}}$ the expression (11) evaluated for $H = H_{TS}(q)$, the command sequence is implemented as follows:

I. $u_I = J_1^\dagger K_1 e_1 + (I - J_1^\dagger J_1) u_{H_{TS}}$, until $||e_1|| \leq \epsilon_1$ and $H_{TS}(q) \leq \epsilon_2$, where $\epsilon_1 = 1$ and $\epsilon_2 = 0.001$;
II. $u_{II} = (I - J_1^\dagger J_1) J_2^T K_2 e_2 + J_1^\dagger K_1 e_1$, until the end of the servoing task.

In Fig. 13, four snapshots of the NMM motion are shown. In particular, the frame at $t = 4.8$ s in the sequence corresponds to the end of the first phase. The NMM is able to regulate the feature positions by keeping the target in front of the platform, thanks to the optimization of $H_{TS}(q)$ during the first phase. The motion of the features in the image plane is shown in Fig. 14, where the switching point between the two phases is indicated on the motion of f_2. Note that feature f_1 never switches, as its desired value is reached at the end of the first phase and then kept during the second phase. The velocity commands shown in Fig. 15 have a discontinuity at the switching time (vertical dashed line), but are considerably smaller than those produced by the TP method.

6. Conclusions

For manipulators carried by mobile platforms whose instantaneous velocity is restricted by nonholonomic constraints, we have followed a kinematic modeling approach that is based on the task to be performed. The main outcome is that the mobility of a nonholonomic mobile manipulator in performing a task is fully entailed in a matrix, the NMM Jacobian, that relates feasible velocity commands to task velocities. This matrix can been used for singularity analysis, redundancy resolution, and kinematic control, just like the classical Jacobian of fixed-base manipulators.

Two standard redundancy resolution methods (Projected Gradient and Task Priority) have been generalized to NMMs, taking care of the way the gradient of a configuration-dependent criterion is mapped into the available velocity

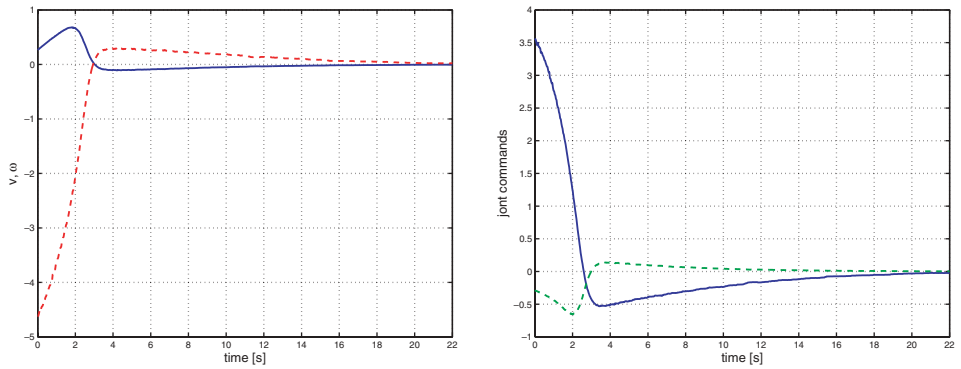

Fig. 12. NMM control commands with the TP method. *Left*: Linear velocity v (*solid blue line*) and angular velocity ω (*dashed red line*) of the platform. *Right*: Arm joint velocities $\dot{q}_1$ (*solid blue line*) and $\dot{q}_2$ (*dashed green line*).

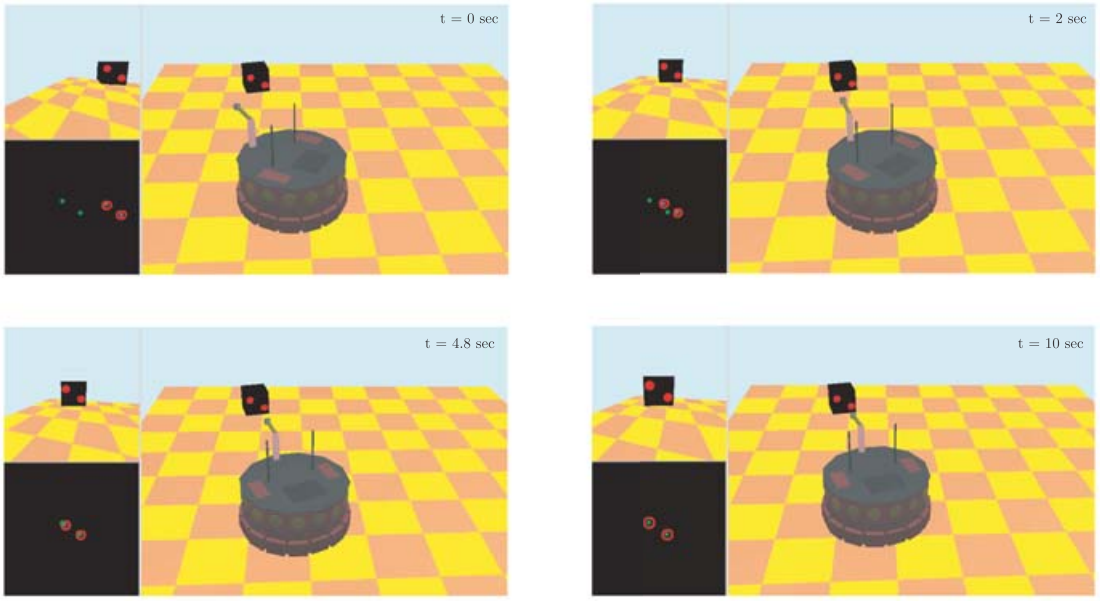

Fig. 13. Snapshots of the visual servoing task with the TS method.

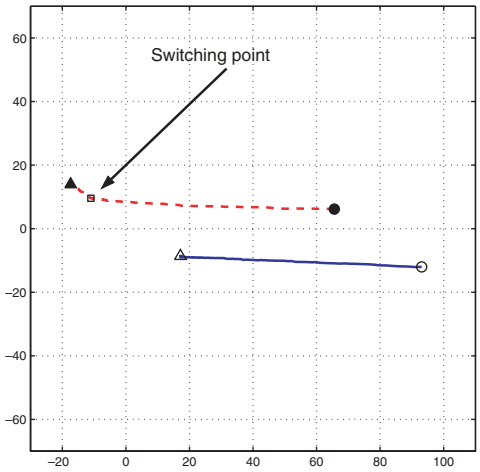

Fig. 14. TS method. Motion of point features f_1 (*solid blue line*) and f_2 (*dashed red line*), with switching point of the latter.

commands. We have also proposed the novel method of Task Sequencing, which introduces additional degrees of redundancy through the time decomposition of a regulation task in a sequence of phases. This turns out to be an efficient way for handling tasks prone to singularities.

As a relevant application of the proposed framework, the image-based visual servoing problem for NMMs equipped with an eye-in-hand camera was considered. By taking the point features in the image plane as task variables, the associated NMM image Jacobian is conveniently obtained by composing the usual interaction matrix of visual servoing with an intermediate, camera-related NMM Jacobian. All methods based on the use of the NMM (image) Jacobian distribute the control effort among the platform and the manipulator velocity commands. The numerical results on two case studies have shown the good performance of the proposed visual servoing control laws, including the optimization of a cost function during task execution and the avoidance of potential singularities on ill-conditioned tasks.

We are currently implementing the proposed visual servoing schemes on the wheeled mobile robot MagellanPro, equipped with a pan-tilt camera (seen as a special 2R polar arm). Future work in visual servoing for NMMs will also explore the use of different sets of features, such as lines, ellipses, or generalized moments, and focus on the case of

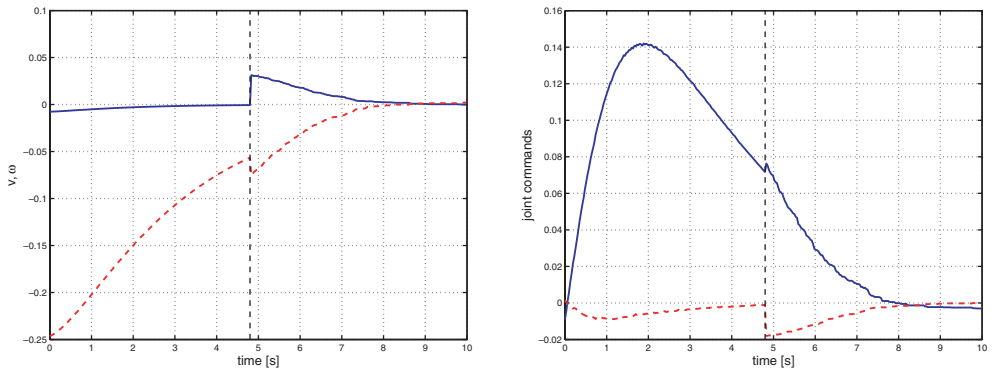

Fig. 15. NMM control commands with the TS method. *Left*: Linear velocity v (*solid blue line*) and angular velocity ω (*dashed red line*) of the platform. *Right*: Arm joint velocities $\dot{q}_1$ (*solid blue line*) and $\dot{q}_2$ (*dashed red line*).

tracking a moving object (e.g., a ball) by estimating online its depth and velocity.

Appendix

In order to show that J_c is independent of q_p, let $R_{z_i}(\alpha)$ be the 3×3 rotation matrix of angle α about the absolute z-axis or about the joint axes $(z_0, \ldots, z_{n_m-1})$ (following the Denavit–Hartenberg convention) of the manipulator arm, and $[x\ y\ h]^{\mathrm{T}}$ be the position of the platform reference point expressed in $\mathcal{F}_O$ (h is the constant platform height). We have

$$t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \begin{bmatrix} x \\ y \\ h \end{bmatrix} + R_z(\theta) e(q_m) \tag{27}$$

where θ is the platform orientation and $e(q_m)$ is the vector pointing from the platform reference point to the camera on the end-effector. Assume that the nonholonomic mobile base is a rigid body that can move with a linear velocity v only along the direction of its orientation θ (this is the case of most wheeled mobile platforms, like those with unicycle or car-like kinematics).

Differentiating Eq. (27), we get

$$\begin{bmatrix} \dot{t}_x \\ \dot{t}_y \\ \dot{t}_z \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} v + R_z(\theta)\dot{e} + \widehat{w}_z(\dot{\theta}) R_z(\theta) e,$$

where

$$\widehat{w}_z(\dot{\theta}) = \begin{bmatrix} 0 & -\dot{\theta} & 0 \\ \dot{\theta} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

is the skew-symmetric matrix associated to the angular velocity $[0\ 0\ \dot{\theta}]^{\mathrm{T}}$. The orientation of frame $\mathcal{F}_{\mathrm{C}}$ w.r.t. frame $\mathcal{F}_O$ is given by the rotation matrix

$$R_c = R_z(\theta) R_0 R_{z_0}(q_1) \ldots R_{z_{n_m-1}}(q_{n_m}) R_e,$$

where $R_{z_k}(q_k)$, $k \in \{0, 1, \ldots, n_m - 1\}$ are the rotations associated to the n_{m} joints of the manipulators arm, R_0 represents the constant orientation of the manipulator first joint axis w.r.t. the platform frame, and R_e is the constant orientation between the camera frame $\mathcal{F}_{\mathrm{C}}$ and the frame located at the arm end-effector. Therefore, the expression of the end-effector linear velocity in the *camera frame*

$$V_c = R_c^{\mathrm{T}} \dot{t} = R_e^{\mathrm{T}} R_{z_{n_m-1}}^{\mathrm{T}}(q_{n_m}) \ldots R_{z_0}^{\mathrm{T}}(q_1) R_0^{\mathrm{T}} \left(\begin{bmatrix} v \\ 0 \\ 0 \end{bmatrix} + \dot{e} + \widehat{w}_z(\dot{\theta}) e \right)$$

is independent from $q_{\mathrm{p}} = [x\ y\ \theta]^{\mathrm{T}}$, having used the property $R_z^{\mathrm{T}}(\theta) \widehat{w}_z(\dot{\theta}) R_z(\theta) = \widehat{w}_z(\dot{\theta})$. Similar arguments can be used to prove independence of the angular velocity ω_c from q_{p}. This result is a direct consequence of having expressed the camera linear/angular velocity in the camera frame, and it would not hold in different frames, as, e.g., the world or the platform frame.

References

1. T. Arai. "Robots with integrated locomotion and manipulation and their future." *Proceedings of the 1996 IEEE/RSJ International Conference on Robots and Intelligent Systems* (1996) pp. 541–545.
2. D. N. Nenchev, Y. Umetani and K. Yoshida. "Analysis of a redundant free-flying spacecraft/manipulator system." *IEEE Trans. Robot. Autom.* **8**(1), 1–6 (1992).
3. G. Campion, G. Bastin and B. D'Andrea-Novel. "Structural properties and classification of kinematic and dynamic models of wheeled mobile robots." *IEEE Trans. Robot. Autom.* **12**(1), 47–62 (1996).
4. H. Seraji. "A unified approach to motion control of mobile manipulators." *Int. J. Robot. Res.* **17**(2), 107–118 (1998).
5. F. G. Pin, K. A. Morgansen, F. A. Tulloch, C. J. Hacker and K. B. Gower. "Motion planning for mobile manipulators with a non-holonomic constraint using the FSP method." *J. Robot. Syst.* **13**(11), 723–736 (1996).
6. A. De Luca, G. Oriolo and P. Robuffo Giordano. "Kinematic modeling and redundancy resolution for nonholonomic mobile robots. *Proceedings of the 2006 IEEE International Conference on Robotics and Automation* (2006) pp. 1867–1873.
7. J. F. Gardner and S. A. Velinsky. "Kinematics of mobile manipulators and implications for design." *J. Robot. Syst.* **17**(6), 309–320 (2000).
8. J.-Y. Fourquet, B. Bayle and M. Renaud. "Manipulability of wheeled mobile manipulators: Application to motion generation." *Int. J. Robot. Res.* **22**(7–8), 565–581 (2003).
9. Y. Nakamura, *Advanced Robotics: Redundancy and Optimization* (Addison-Wesley, Reading, MA, 1991).
10. H. Seraji. "An on-line approach to coordinated mobility and manipulation." *Proceedings of the 1993 IEEE International Conference on Robotics and Automation* (1993) pp. 28–35.
11. F. Lamiraux, B. Bayle, J.-Y. Fourquet and M. Renaud. "Kinematic control of wheeled mobile manipulators." *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2002) pp. 1572–1577.
12. B. Bayle, J.-Y. Fourquet and M. Renaud. "Génération des mouvements des manipulateurs mobiles: Etat de l'art et perspectives." *J. Eur. Syst. Autom.* **35**(6), 809–845 (2001).
13. G. Campion, B. d'Andrea Novel and G. Bastin. "Controllability and State Feedback Stabilizability of Nonholonomic Mechanical Systems." **In**: *Advanced Robot Control* (C. Canudas de Wit, ed.) vol. 162, *LNCIS*, (Springer-Verlag, Berlin, 1991) pp. 106–124.
14. Y. Yamamoto and X. Yun. "Unified analysis on mobility and manipulability of mobile manipulators." *Proceedings of the 1999 IEEE International Conference on Robotics and Automation* (1999) pp. 1200–1206.
15. B. Espiau, F. Chaumette and P. Rives. "A new approach to visual servoing in robotics." *IEEE Trans. Robot. Autom.* **8**(3), 313–326 (1992).
16. S. Hutchinson, G. D. Hager and P. I. Corke. "A tutorial on visual servo control." *IEEE Trans. Robot. Autom.* **12**(5), 651–670 (1996).
17. A. C. Sanderson and L. E. Weiss. "Image based visual servo control using relational graph error signals." *Proceedings of the IEEE International Conference on Cybernetics and Society* (1980) pp. 1074–1077.
18. W. J. Wilson, C. C. Williams Hulls and G. S. Bell. "Relative end-effector control using cartesian position based visual servoing." *IEEE Trans. Robot. Autom.* **12**(5), 684–696 (1996).
19. B. Espiau. "Effect of Camera Calibration Errors on Visual Servoing in Robotics." **In**: *Experimental Robotics III* (T. Yoshikawa and F. Miyazaki, ed.) *LNCIS*, vol. 200, (Springer-Verlag, Berlin 1994) pp. 182–192.

20. P. I. Corke and S. A. Hutchinson. "A new partitioned approach to image-based visual servo control." *IEEE Trans. Robot. Autom.* **17**(4), 507–515 (2001).
21. C. Samson, B. Espiau and M. Le Borgne, *Robot Control: The Task Function Approach* (Oxford University Press, London, 1991).
22. F. Chaumette. "Potential Problems of Stability and Convergence in Image-Based and Position-Based Visual Servoing. **In**: *The Confluence of Vision and Control* (D. Kriegman, G. Hager, and A. Morse, ed.) *LNCIS*, vol. 237, (Springer-Verlag, Berlin) pp. 66–78.
23. E. Malis and P. Rives. "Robustness of image-based visual servoing with respect to depth distribution errors." *Proceedings of the 2003 IEEE International Conference on Robotics and Automation* (2003) pp. 1056–1061.
24. K. Deguchi. "Optimal motion control for image-based visual servoing by decoupling translation and rotation." *Proceedings of the 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems* (1998) pp. 705–711.
25. E. Malis, F. Chaumette and S. Boudet. "2-1/2-D visual servoing." *IEEE Trans. Robot. Autom.* **15**(2), 238–250 (1999).
26. G. Morel, T. Liebezeit, J. Szewczyk, S. Boudet and J. Pot. "Explicit Incorporation of 2D Constraints in Vision Based Control of Robot Manipulators." **In**: *Experimental Robotics VI* (P. Corke and J. Trevelyan, ed.) *LNCIS*, vol. 250, (Springer-Verlag, Berlin 2000) pp. 99–108.
27. F. Chaumette and E. Marchand. "A redundancy-based iterative approach for avoiding joint limits: Application to visual servoing." *IEEE Trans. Robot. Autom.* **17**(5), 719–730 (2001).
28. G. L. Mariottini, G. Oriolo and D. Prattichizzo. "Epipole-based visual servoing for nonholonomic mobile robots." To appear in *IEEE Trans. Robot.*, 2007.
29. R. Pissard-Gibollet and P. Rives. "Applying visual servoing technique to control a mobile hand-eye system." *Proceedings of the 1995 IEEE International Conference on Robotics and Automation* (1995) pp. 166–171.
30. P. Chiacchio, S. Chiaverini, L. Sciavicco and B. Siciliano. "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy." *Int. J. Robot. Res.* **10**, 410–425 (1991).
31. A. De Luca, G. Oriolo and P. Robuffo Giordano. "On-line estimation of feature depth for image-based visual servoing schemes. To appear in *Proceedings of the 2007 IEEE International Conference on Robotics and Automation* (2007).
32. L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators* (Springer, Berlin, 2000).
33. G. Oriolo and C. Mongillo. "Motion planning for mobile manipulators along given end-effector paths." *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (2005) pp. 2166–2172.
34. A. A. Maciejewski and C. A. Klein. "The singular value decomposition: Computation and applications to robotics." *Int. J. Robot. Res.* **8**(6), 63–79 (1989).
35. M. Zak. "Terminal attractors in neural networks." *Neural Netw.* **2**, 259–274 (1989).
36. N. Mansard and F. Chaumette. "Tasks sequencing for visual servoing." *Proceedings of the 2004 IEEE/RSJ International Conference on Robots and Intelligent Systems* (2004) pp. 992–997.
37. F. Chaumette. "Image moments: A general and useful set of features for visual servoing." *IEEE Trans. Robot.* **20**(4), 713–723 (2004).
38. O. Tahri and F. Chaumette. Point-based and region-based image moments for visual servoing of planar objects. *IEEE Trans. Robot.* **21**(6), 1116–1127 (2005).