Coupling Visual Servoing with Active Structure from Motion

Riccardo Spica, Paolo Robuffo Giordano, François Chaumette

Coupling Visual Servoing with Active Structure from Motion

Riccardo Spica, Paolo Robuffo Giordano, and François Chaumette

***Abstract*— In this paper we propose a solution for coupling the execution of a visual servoing task with a recently developed *active* Structure from Motion strategy able to optimize online the convergence rate in estimating the (unknown) 3D structure of the scene. This is achieved by suitably modifying the robot trajectory in the null-space of the servoing task so as to render the camera motion 'more informative' w.r.t. the states to be estimated. As a byproduct, the better 3D structure estimation also improves the evaluation of the servoing interaction matrix which, in turn, results in a better closed-loop convergence of the task itself. The reported experimental results support the theoretical analysis and show the benefits of the method.**

I. INTRODUCTION

In many applications the state of a robot w.r.t. the environment can only be partially retrieved from its onboard sensors, and online state estimation schemes can be exploited in order to recover the 'missing information' by incrementally processing the sensed data. When considering non-trivial cases, however, one often faces *nonlinear* estimation problems for which the actual robot trajectory plays an important role for a successful estimation convergence. This is, for instance, the case of all Structure from Motion (SfM) problems in which a poor choice of the system inputs (the camera linear velocity) can even make the 3D scene structure non-observable (and regardless of the employed estimation strategy). It is then interesting to study how to *optimize* the trajectory of a robot executing a given task with the aim of facilitating the state estimation process.

The goal of this paper is to propose an *online* solution to this problem in the context of visual control of robot manipulators. We consider, as case study, a classical Image-Based Visual Servoing [1] (IBVS) of a set of point features whose depths represent the unknown states to be estimated during motion. We then show how to couple the IBVS execution with the optimization of the depth estimation convergence. A coupling between visual servoing and SfM estimation was also proposed in [2] but without any *active* optimization of the camera velocity for the sake of enhancing the estimation convergence rate. The optimization action in our proposed solution is based on a recently proposed framework for active SfM [3] and is here projected onto the null-space of the IBVS task considered as the primary objective. Additionally, in order to obtain the largest possible degree of redundancy w.r.t. the IBVS task, we also suitably exploit and extend the redundancy framework introduced in [4] meant to provide a *large* projection operator by considering the *norm* of the visual error as main task. In particular, the controller originally proposed in [4] is here extended to the *second-order* since the active strategy of [3] acts on the camera *linear velocity* and, thus, requires an action at the acceleration level.

We then experimentally validate the proposed active estimation and control strategy. The experiments clearly show the achievement of two goals: (i) the optimization of the camera trajectory during the servoing transient which allows obtaining the fastest possible convergence of the SfM algorithm, and (ii) the concurrent improvement of the IBVS convergence thanks to the better approximation of the interaction matrix from the recovered 3D parameters. We finally stress that the proposed coupling between *task execution* and *trajectory optimization* for improved state estimation is not restricted to the sole class of IBVS problems considered in this work: indeed, one can easily generalize these ideas to other servoing tasks (e.g., considering different visual features or even geometric ones as in PBVS schemes), or apply them to other contexts not necessarily related to visual control (as long as the chosen robot trajectory has an effect on the state estimation task).

The rest of the paper is organized as follows: we start by summarizing in Sec. II the active SfM framework presented in [3]. Then, we present in Sec. III a second-order extension of the strategy described in [4] that allows to increase the degree of redundancy w.r.t. the considered IBVS task to its maximum extent. Finally we report in Sec. IV some experimental results validating our analysis, and we draw some conclusions in Sec. V by also proposing some possible future directions.

II. A FRAMEWORK FOR ACTIVE STRUCTURE FROM MOTION

We start by briefly summarizing the active SfM framework proposed in [3]. Let $\boldsymbol{s} \in \mathbb{R}^m$ be the set of visual features *measured* on the image plane of a (assumed calibrated) camera, $\boldsymbol{\chi} \in \mathbb{R}^p$ a suitable (and locally invertible) function of the unknown structure of the scene *to be estimated* by the SfM algorithm, and $\boldsymbol{u} = (\boldsymbol{v}, \boldsymbol{\omega}) \in \mathbb{R}^6$ the camera linear/angular velocity expressed in the camera frame. With these choices, one can show that the SfM dynamics takes the general form

$$\left\{ \begin{array}{rcl} \dot{\boldsymbol{s}} & = & \boldsymbol{f}_m(\boldsymbol{s}, \boldsymbol{u}) + \boldsymbol{\Omega}^T(\boldsymbol{s}, \boldsymbol{v})\boldsymbol{\chi} \\ \dot{\boldsymbol{\chi}} & = & \boldsymbol{f}_u(\boldsymbol{s}, \boldsymbol{\chi}, \boldsymbol{u}) \end{array} \right. \tag{1}$$

where matrix $\boldsymbol{\Omega}(\boldsymbol{s}, \boldsymbol{v}) \in \mathbb{R}^{p \times m}$ is a *known* quantity such that $\boldsymbol{\Omega}(\boldsymbol{s}, \boldsymbol{0}) \equiv \boldsymbol{0}$. Let now $(\hat{\boldsymbol{s}}\,, \hat{\boldsymbol{\chi}}) \in \mathbb{R}^{m+p}$ be the estimated

R. Spica is with the University of Rennes 1 at Irisa and Inria Rennes Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France riccardo.spica@irisa.fr

P. Robuffo Giordano is with the CNRS at Irisa and Inria Rennes Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France prg@irisa.fr.

F. Chaumette is with Inria Rennes Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France francois.chaumette@irisa.fr

state, and define $\boldsymbol{\xi} = \boldsymbol{s} - \hat{\boldsymbol{s}}$ as the 'visual feedback' error (measured $\boldsymbol{s}$ vs. estimated $\hat{\boldsymbol{s}}$) and $\boldsymbol{z} = \boldsymbol{\chi} - \hat{\boldsymbol{\chi}}$ as the 3D structure estimation error. An estimation scheme for system (1) meant to recover the unmeasurable $\boldsymbol{\chi}(t)$ from the measured $\boldsymbol{s}(t)$ can be devised as

$$\left\{ \begin{array}{rcl} \dot{\hat{\boldsymbol{s}}} & = & \boldsymbol{f}_m(\boldsymbol{s}, \boldsymbol{u}) + \boldsymbol{\Omega}^T(\boldsymbol{s}, \boldsymbol{v})\hat{\boldsymbol{\chi}} + \boldsymbol{H}\boldsymbol{\xi} \\ \dot{\hat{\boldsymbol{\chi}}} & = & \boldsymbol{f}_u(\boldsymbol{s}, \hat{\boldsymbol{\chi}}, \boldsymbol{u}) + \alpha\boldsymbol{\Omega}(\boldsymbol{s}, \boldsymbol{v})\boldsymbol{\xi} \end{array} \right. \tag{2}$$

where $\boldsymbol{H} > 0$ and $\alpha > 0$ are suitable gains. We note that the scheme (2) *does not* require knowledge of $\dot{\boldsymbol{s}}$ (i.e., measurement of velocities on the image plane), but it only needs measurement of $\boldsymbol{s}$ (the 'visual features') and of $(\boldsymbol{v}, \boldsymbol{\omega})$ (the camera linear/angular velocity in the camera frame).

Following [3], it is possible to *characterize* the transient response of the SfM estimation error $\boldsymbol{z}(t) = \boldsymbol{\chi}(t) - \hat{\boldsymbol{\chi}}(t)$, as well as to *affect* it by acting *online* on the camera motion. One can indeed show that the convergence rate of $\boldsymbol{z}(t)$ results dictated by the norm of the square matrix $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$, in particular by its smallest eigenvalue σ_1^2. For a given choice of gain α (a free parameter), the larger σ_1^2 the faster the error convergence, with in particular $\sigma_1^2 = 0$ if $\boldsymbol{v} = 0$ (as well-known, only a translating camera can estimate the scene structure). Being $\boldsymbol{\Omega} = \boldsymbol{\Omega}(\boldsymbol{s}, \boldsymbol{v})$, one has

$$\dot{(\sigma_1^2)} = \boldsymbol{J}_v\dot{\boldsymbol{v}} + \boldsymbol{J}_s\dot{\boldsymbol{s}}, \tag{3}$$

where the Jacobian matrices $\boldsymbol{J}_v \in \mathbb{R}^{1\times 3}$ and $\boldsymbol{J}_s \in \mathbb{R}^{1\times m}$ have a *closed form* expression function of $(\boldsymbol{s}, \boldsymbol{v})$ (known quantities), see [3]. This relationship can then be inverted w.r.t. vector $\dot{\boldsymbol{v}}$ for affecting *online* $\sigma_1^2(t)$ during motion, e.g., in order to maximize its value for increasing the convergence rate of $\boldsymbol{z}(t)$. We note that this step represents the *active* component of the estimation strategy since, in the general case, inversion of (3) will yield a camera velocity $\boldsymbol{v}(t)$ function of the system measured state $\boldsymbol{s}(t)$.

Formulation (1) can be applied to the point feature case (considered in this work) by taking $\boldsymbol{s} = \boldsymbol{p} = (x, y) = (X/Z, Y/Z)$ as the perspective projection of a 3D point (X, Y, Z), and $\chi = 1/Z$ with, thus, $m = 2$ and $p = 1$. Explicit expressions of the above machinery can be found in [3] with, in particular,

$$\left\{ \begin{array}{l} \sigma_1^2 = \boldsymbol{\Omega}\boldsymbol{\Omega}^T = (xv_z - v_x)^2 + (yv_z - v_y)^2 \\ \boldsymbol{J}_v = 2\begin{bmatrix} v_x - xv_z & v_y - yv_z & (xv_z - v_x)\,x + (yv_z - v_y)\,y \end{bmatrix} \\ \boldsymbol{J}_s = 2\begin{bmatrix} (xv_z - v_x)v_z & (yv_z - v_y)v_z \end{bmatrix} \end{array} \right. . \tag{4}$$

III. VISUAL SERVOING COUPLED WITH ACTIVE 3D ESTIMATION

A. Problem Description

We consider the classical situation of a robot manipulator with joint configuration vector $\boldsymbol{q} \in \mathbb{R}^n$ carrying a eye-in-hand camera that measures a set of visual features $\boldsymbol{s} \in \mathbb{R}^m$ to be regulated to a desired constant value $\boldsymbol{s}^*$. As well-known, one has $\dot{\boldsymbol{s}} = \boldsymbol{L}_{\boldsymbol{s}}(\boldsymbol{s}, \boldsymbol{\chi})\boldsymbol{u}$ and $\boldsymbol{u} = \boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{q})\dot{\boldsymbol{q}}$, where $\boldsymbol{L}_{\boldsymbol{s}} \in \mathbb{R}^{m\times 6}$ is the interaction matrix of the considered visual features, $\boldsymbol{\chi} \in \mathbb{R}^p$ is a vector of unmeasurable 3D quantities associated to $\boldsymbol{s}$ (e.g., the depth Z for a feature point), and $\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{q}) \in \mathbb{R}^{6\times n}$ the Jacobian of the eye-in-hand camera w.r.t. the robot joint velocities.

Let $\boldsymbol{J}(\boldsymbol{s}, \boldsymbol{q}, \boldsymbol{\chi}) = \boldsymbol{L}_{\boldsymbol{s}}(\boldsymbol{s}, \boldsymbol{\chi})\boldsymbol{J}_{\boldsymbol{C}}(\boldsymbol{q}) \in \mathbb{R}^{m\times n}$ be the visual task Jacobian and define $\boldsymbol{e} = \boldsymbol{s} - \boldsymbol{s}^*$ as the visual error vector with, thus, $\dot{\boldsymbol{e}} = \boldsymbol{J}\dot{\boldsymbol{q}}$. In case the robot is redundant w.r.t. the visual task ($n > m$), a typical choice for regulating $\boldsymbol{e}(t) \to \boldsymbol{0}$ is to apply the control law [1]

$$\dot{\boldsymbol{q}} = -\lambda\hat{\boldsymbol{J}}^{\dagger}\boldsymbol{e} + (\boldsymbol{I}_n - \hat{\boldsymbol{J}}^{\dagger}\hat{\boldsymbol{J}})\boldsymbol{r}, \qquad \lambda > 0. \tag{5}$$

Here, $\boldsymbol{A}^{\dagger}$ denotes the pseudoinverse of matrix $\boldsymbol{A}$, the task Jacobian $\hat{\boldsymbol{J}}(\boldsymbol{s}, \hat{\boldsymbol{\chi}}, \boldsymbol{q})$ is evaluated on some approximation $\hat{\boldsymbol{\chi}}$ of the unknown true vector $\boldsymbol{\chi}$, e.g., the *value at the desired pose* $\hat{\boldsymbol{\chi}} = \boldsymbol{\chi}^*$, and $\boldsymbol{r} \in \mathbb{R}^n$ is an arbitrary vector projected on the null-space of the main visual task. When applying (5) with $\mathrm{rank}(\boldsymbol{J}) = m$ and $\hat{\boldsymbol{\chi}} = \boldsymbol{\chi}$, one obtains a perfectly decoupled and exponential behavior for the visual error $\boldsymbol{e}(t)$, i.e.,

$$\boldsymbol{e}(t) = \exp(-\lambda(t - t_0))\boldsymbol{e}(t_0). \tag{6}$$

When, instead, $\boldsymbol{\chi}$ is replaced by any approximation $\hat{\boldsymbol{\chi}}$, the ideal closed-loop behavior (6) is no longer obtained.

Since $\boldsymbol{\chi}$ is not directly measurable from visual input, and special approximations such as $\boldsymbol{\chi}^*$ require anyway some 'pre-knowledge' of the scene, another interesting possibility is to exploit the estimation scheme (2) for recovering online a (converging) estimation $\hat{\boldsymbol{\chi}}(t)$ from the measured $\boldsymbol{s}(t)$ (visual features) and known $\boldsymbol{u}(t)$ (camera motion). Observer (2) can indeed run *in parallel* to the servoing controller (5), by treating the camera linear/angular velocities generated by (5) as the inputs $\boldsymbol{u}$ in (2), and by plugging the estimated state $\hat{\boldsymbol{\chi}}$ recovered by (2) in the evaluation of $\hat{\boldsymbol{J}}(\boldsymbol{s}, \hat{\boldsymbol{\chi}}, \boldsymbol{q})$ needed by (5). This way, one can in fact: (i) improve the servoing execution by yielding a closed-loop behavior matching the ideal (6) also when far from the desired pose and without needing special assumptions/approximations of $\boldsymbol{\chi}$, since, as $\hat{\boldsymbol{\chi}}(t) \to \boldsymbol{\chi}(t)$, one has $\hat{\boldsymbol{J}} \to \boldsymbol{J}$, and (ii) obtain, as a byproduct, the concurrent 3D structure estimation of the observed scene by exploiting the motion performed by the camera for realizing the servoing task.

In this conceptual scheme, the estimation of the 3D structure $\boldsymbol{\chi}$ takes then place only during the transient of the servoing task (i.e., as long as the camera is in motion towards its goal location). Being this phase of limited duration, with the camera reaching a full stop at the end of the servoing, one is clearly interested in obtaining the fastest possible convergence for the estimation error. As explained in the previous section, it is possible to 'optimize' the convergence rate of the estimation error by (actively) maximizing the eigenvalue σ_1^2 over time. A natural possibility is to then project such optimization action within the null-space of the main visual servoing task. The next section explains a possible strategy to achieve this goal.

B. Second-order Visual Servoing using a Large Projection Operator

In order to fully take advantage of the optimization of the 3D structure estimation, it is clearly important to exploit the largest possible redundancy w.r.t. the considered visual task.

Indeed, if the visual task constrains most or all the camera dofs, no optimization of the camera motion can be performed. In this sense, the redundancy framework proposed in [4] represents a very convenient possibility: regulation of the visual error vector $\boldsymbol{e}$ can be replaced by the regulation of its norm $\|\boldsymbol{e}\|$ (a 1-dimensional task), thus resulting in a null-space of (maximal) dimension $n-1$ available for additional optimizations. As discussed in [4], this technique becomes singular for $\|\boldsymbol{e}\| \to 0$ or $\boldsymbol{e} \in Ker(\boldsymbol{J}^T)$ and, thus, requires a proper switching to the classical law (regulation of the whole vector $\boldsymbol{e}$) when close to convergence.

Coming to our case, the expression in (3) shows that optimization of $\sigma_1^2(t)$ requires an action at the *joint acceleration level*. Indeed, from (3) and $\boldsymbol{v} = \boldsymbol{J}_C(\boldsymbol{q})\dot{\boldsymbol{q}}$, one has

$$\dot{(\sigma_1^2)} = \boldsymbol{J}_v\dot{\boldsymbol{v}} + \boldsymbol{J}_s\dot{\boldsymbol{s}} = \boldsymbol{J}_v\boldsymbol{J}_C\ddot{\boldsymbol{q}} + \boldsymbol{J}_v\dot{\boldsymbol{J}}_C\dot{\boldsymbol{q}} + \boldsymbol{J}_s\dot{\boldsymbol{s}}. \quad (7)$$

Maximization of σ_1^2 can be obtained by applying, for instance, the following *joint acceleration vector*

$$\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_\sigma = k_\sigma \boldsymbol{J}_C^T\boldsymbol{J}_v^T - (\boldsymbol{J}_v\boldsymbol{J}_C)^\dagger(\boldsymbol{J}_v\dot{\boldsymbol{J}}_C\dot{\boldsymbol{q}} + \boldsymbol{J}_s\dot{\boldsymbol{s}}) \quad (8)$$

with $k_\sigma > 0$. However, the control strategy proposed in [4] only addresses motion control at the *first-order/velocity* level. We then now proceed to its extension at the *second-order/acceleration* level.

Assume that $m \geq n$ and $\mathrm{rank}(\boldsymbol{J}) = n$ (overconstrained visual task with no redundancy left, as in the experiments reported in the next section). We first note that, being $\dot{\boldsymbol{e}} = \boldsymbol{J}\dot{\boldsymbol{q}}$ and, thus, $\ddot{\boldsymbol{e}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}$, the second-order counterpart of the classical law (5) for regulating the whole error vector $\boldsymbol{e}(t)$ to $\boldsymbol{0}$ is simply

$$\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_{\boldsymbol{e}} = \boldsymbol{J}^\dagger(-k_v\dot{\boldsymbol{e}} - k_p\boldsymbol{e} - \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}) \quad (9)$$

with $k_p > 0$ and $k_v > 0$. Note that, as in the first-order case (5), when implementing (9) one should replace $\boldsymbol{J}(\boldsymbol{s}, \boldsymbol{\chi}, \boldsymbol{q})$ with its approximation $\hat{\boldsymbol{J}}(\boldsymbol{s}, \hat{\boldsymbol{\chi}}, \boldsymbol{q})$, and likewise for the evaluation of $\dot{\boldsymbol{e}}$ and $\dot{\boldsymbol{J}}$. However, for the sake of exposition, we assume for now availability of all the needed quantities.

The controller (9) would solve the visual servoing task but, clearly, without any possible additional optimization action. By now letting $\nu = \|\boldsymbol{e}\|$, it is

$$\dot{\nu} = \frac{\boldsymbol{e}^T\boldsymbol{J}}{\|\boldsymbol{e}\|}\dot{\boldsymbol{q}} = \boldsymbol{J}_{\|\boldsymbol{e}\|}\dot{\boldsymbol{q}}, \qquad \boldsymbol{J}_{\|\boldsymbol{e}\|} \in \mathbb{R}^{1\times n},$$

and $\ddot{\nu} = \boldsymbol{J}_{\|\boldsymbol{e}\|}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}_{\|\boldsymbol{e}\|}\dot{\boldsymbol{q}}$. Regulation of $\nu(t) \to 0$ can then be achieved by applying the joint acceleration vector

$$\begin{aligned}\ddot{\boldsymbol{q}} &= \ddot{\boldsymbol{q}}_{\|\boldsymbol{e}\|} = \boldsymbol{J}_{\|\boldsymbol{e}\|}^\dagger(-k_v\dot{\nu} - k_p\nu - \dot{\boldsymbol{J}}_{\|\boldsymbol{e}\|}\dot{\boldsymbol{q}}) + (\boldsymbol{I}_n - \boldsymbol{J}_{\|\boldsymbol{e}\|}^\dagger\boldsymbol{J}_{\|\boldsymbol{e}\|})\boldsymbol{r} \\ &= \boldsymbol{J}_{\|\boldsymbol{e}\|}^\dagger(-k_v\dot{\nu} - k_p\nu - \dot{\boldsymbol{J}}_{\|\boldsymbol{e}\|}\dot{\boldsymbol{q}}) + \boldsymbol{P}_{\|\boldsymbol{e}\|}\boldsymbol{r},\end{aligned} \quad (10)$$

with $k_p > 0$, $k_v > 0$, $\boldsymbol{J}_{\|\boldsymbol{e}\|}^\dagger = \frac{\|\boldsymbol{e}\|}{\boldsymbol{e}^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{e}}\boldsymbol{J}^T\boldsymbol{e}$ and $\boldsymbol{P}_{\|\boldsymbol{e}\|} = \boldsymbol{I}_n - \frac{\boldsymbol{J}^T\boldsymbol{e}\boldsymbol{e}^T\boldsymbol{J}}{\boldsymbol{e}^T\boldsymbol{J}\boldsymbol{J}^T\boldsymbol{e}}$ being the null-space projection operator of the error norm with (at least) rank $n-1$. One can then choose vector $\boldsymbol{r}$ in (10) as

$$\boldsymbol{r} = \ddot{\boldsymbol{q}}_\sigma - k_{d_1}\dot{\boldsymbol{q}}, \qquad k_{d_1} > 0, \quad (11)$$

so as to project the optimization action (8) onto the null-space of the main task $\nu = \|\boldsymbol{e}\|$, together with an additional 'damping' on the joint velocities needed to stabilize motions in the null-space, see [5]. Controller (8)–(10–11) can then realize the visual task by regulating its norm to zero ($\|\boldsymbol{e}\| \to 0$) with the largest possible redundancy (of degree $n-1$)[1].

We note that the Jacobian $\boldsymbol{J}_{\|\boldsymbol{e}\|}$ is singular for $\|\boldsymbol{e}\| = 0$ and, as explained in [4], the projection matrix $\boldsymbol{P}_{\|\boldsymbol{e}\|}$ is not well-defined for $\boldsymbol{e} \to \boldsymbol{0}$. Presence of this (unavoidable) singularity motivates the introduction of a switching from the controller (10) to the classical law (9) when close to convergence. However, the 'first-order' switching strategy proposed in [4] cannot be directly transposed to the second-order case, but some suitable modifications must be taken into account as discussed in the next section.

C. A Second-order Switching Strategy

We start noting that, in closed-loop, controller $\ddot{\boldsymbol{q}}_{\|\boldsymbol{e}\|}$ in (10) imposes the following second-order dynamics to the error norm

$$\ddot{\nu} + k_v\dot{\nu} + k_p\nu = 0. \quad (12)$$

Define $\nu_{\|\boldsymbol{e}\|}(t)$ as the solution of (12) for a given initial condition $(\nu(t_0), \dot{\nu}(t_0))$: $\nu_{\|\boldsymbol{e}\|}(t)$ thus represents the 'ideal' evolution of the error norm, that is, the behavior one would obtain if controller (10) could be implemented $\forall t \geq t_0$.

Let now $t_1 > t_0$ be the time at which the switch from controller (10) to the classical law $\ddot{\boldsymbol{q}}_{\boldsymbol{e}}$ in (9) occurs (e.g., triggered by some threshold on $\|\boldsymbol{e}\|$ as proposed in [4]). For $t \geq t_1$ and under the action of $\ddot{\boldsymbol{q}}_{\boldsymbol{e}}$ one has in closed-loop

$$\ddot{\boldsymbol{e}} + k_v\dot{\boldsymbol{e}} + k_p\boldsymbol{e} = 0. \quad (13)$$

Let $\boldsymbol{e}^*(t)$ be the solution of (13) with initial conditions $(\boldsymbol{e}(t_1), \dot{\boldsymbol{e}}(t_1))$, and let $\nu^*(t) = \|\boldsymbol{e}^*(t)\|$ be the corresponding behavior of the error norm. Ideally, one would like to have

$$\nu^*(t) \equiv \nu_{\|\boldsymbol{e}\|}(t), \quad \forall t \geq t_1. \quad (14)$$

In other words, the behavior of the error norm should not be affected by the control switch at time t_1, but $\nu^*(t)$ (obtained from (13)) should exactly match the 'ideal' evolution $\nu_{\|\boldsymbol{e}\|}(t)$ generated by (12) as if no switch had taken place.

While condition (14) is easily satisfied at first-order [4], this is not necessarily the case at the second-order level. Indeed, when moving to the second-order, condition (14) holds if and only if, at time t_1, vectors $\boldsymbol{e}(t_1)$ and $\dot{\boldsymbol{e}}(t_1)$ are *parallel* (proof in the Appendix). It is then necessary to introduce an intermediate phase before the switch during which any component of $\dot{\boldsymbol{e}}$ orthogonal to $\boldsymbol{e}$ is made negligible.

To this end, let

$$\boldsymbol{P}_{\boldsymbol{e}} = \left(\boldsymbol{I}_m - \frac{\boldsymbol{e}\boldsymbol{e}^T}{\boldsymbol{e}^T\boldsymbol{e}}\right) \in \mathbb{R}^{m\times m}$$

be the null-space projector spanning the $(m-1)$-dimensional space orthogonal to vector $\boldsymbol{e}$. Let also

$$\boldsymbol{\delta} = \boldsymbol{P}_{\boldsymbol{e}}\dot{\boldsymbol{e}} = \boldsymbol{P}_{\boldsymbol{e}}\boldsymbol{J}\dot{\boldsymbol{q}}.$$

[1]Note that also in the case $m < n$ the use of (10) would have increased the redundancy degree from $n-m$ to $n-1$.

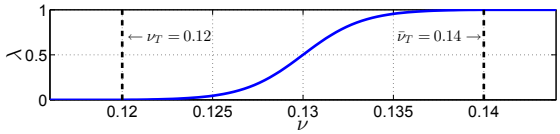


Fig. 1: Switching function $\lambda(\nu)$ with swithching values ν_T and $\bar{\nu}_T$ (dashed lines) such that $\lambda(\nu_T) \approx 0$ and $\lambda(\bar{\nu}_T) \approx 1$.

The scalar quantity $\boldsymbol{\delta}^T\boldsymbol{\delta} \geq 0$ provides a measure of the misalignment among the directions of vectors $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ ($\boldsymbol{\delta}^T\boldsymbol{\delta} = 0$ iff $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ are parallel). One can then minimize $\boldsymbol{\delta}^T\boldsymbol{\delta}$ compatibly with the main task (regulation of the error norm) by choosing vector $\boldsymbol{r}$ in (10) as

$$\boldsymbol{r} = -\frac{k_{d_2}}{2}\left(\frac{\partial \boldsymbol{\delta}^T\boldsymbol{\delta}}{\partial \dot{\boldsymbol{q}}}\right)^T = -k_{d_2}\boldsymbol{J}^T\boldsymbol{P}_{\boldsymbol{e}}\boldsymbol{J}\dot{\boldsymbol{q}}, \qquad k_{d_2} > 0, \tag{15}$$

where the properties $\boldsymbol{P}_{\boldsymbol{e}} = \boldsymbol{P}_{\boldsymbol{e}}^T = \boldsymbol{P}_{\boldsymbol{e}}\boldsymbol{P}_{\boldsymbol{e}}$ were used.

A possible switching strategy for ensuring condition (14) is then:

1) apply the norm controller $\ddot{\boldsymbol{q}}_{\|\boldsymbol{e}\|}$ given in (10) with the null-space vector $\boldsymbol{r}$ defined in (11) as long as $\nu(t) \geq \nu_T$, with $\nu_T > 0$ being a suitable threshold on the error norm. During this phase, the error norm will be governed by the closed-loop dynamics (12) and the convergence rate in estimating $\hat{\boldsymbol{\chi}}$ will be maximized;
2) when $\nu(t) = \nu_T$, keep applying controller $\ddot{\boldsymbol{q}}_{\|\boldsymbol{e}\|}$ but replace (11) with (15) for vector $\boldsymbol{r}$. Stay in this phase as long as $\boldsymbol{\delta}^T\boldsymbol{\delta} \geq \delta_T$, with $\delta_T > 0$ a suitable threshold on the alignment among vectors $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$. Note that, during this second phase, $\nu(t)$ keeps being governed by the closed-loop dynamics (12) since $\boldsymbol{r}$ acts in the null-space of the error norm (i.e., no distorting effect is produced on the behavior of $\nu(t)$ by the change in $\boldsymbol{r}$);
3) when $\boldsymbol{\delta}^T\boldsymbol{\delta} = \delta_T$, switch to the classical controller $\ddot{\boldsymbol{q}}_{\boldsymbol{e}}$ given in (9) until completion of the task. Property (14) will clearly hold since, at the switch time, parallelity among $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ has been enforced by the previous phase.

We finally note that this strategy could cause a discontinuity in the commanded $\ddot{\boldsymbol{q}}$ when passing from phase 1) to phase 2) because of the instantaneous change of vector $\boldsymbol{r}$ from (11) to (15). This discontinuity can be easily dealt with by resorting to a suitable smoothing function $\lambda(\nu)$ as proposed in [4]. Indeed by implementing in both phase 1) and phase 2)

$$\boldsymbol{r} = \lambda(\nu)\left(\ddot{\boldsymbol{q}}_\sigma - k_{d_1}\dot{\boldsymbol{q}}\right) - \left(1 - \lambda(\nu)\right)k_{d_2}\boldsymbol{J}^T\boldsymbol{P}_{\boldsymbol{e}}\boldsymbol{J}\dot{\boldsymbol{q}}, \tag{16}$$

vector $\boldsymbol{r}$ in (11) is gradually replaced by vector $\boldsymbol{r}$ in (15) while $\bar{\nu}_T \geq \nu(t) \geq \nu_T$. Figure 1 shows an illustrative example of $\lambda(\nu)$. As for the switch from phase 2) to phase 3), discontinuities in $\ddot{\boldsymbol{q}}$ are avoided thanks to the alignment among vector $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$.

Before concluding this section we remark that the proposed scheme (active SfM coupled to the second-order visual servoing and associated switching strategy) only requires, as measured quantities, the visual features $\boldsymbol{s}$, the robot joint configuration vector $\boldsymbol{q}$ and the joint velocities $\dot{\boldsymbol{q}}$ (in addition to the usual assumption of intrinsic and camera-to-robot parameters). Indeed knowing $\hat{\boldsymbol{\chi}}$, a (possibly approximated) evaluation of *all* the other quantities entering the various steps of the second-oder control strategy can be obtained from ($\boldsymbol{s}$, $\hat{\boldsymbol{\chi}}$, $\boldsymbol{q}$) and $\dot{\boldsymbol{q}}$ (the only 'velocity' information actually needed). We also note that the level of approximation is clearly a monotonic function of $\|\boldsymbol{\chi} - \hat{\boldsymbol{\chi}}\|$ (i.e., the uncertainty in knowing $\boldsymbol{\chi}$): thus, all the previous quantities will asymptotically match their real values as the estimation error $\boldsymbol{z}(t) = \boldsymbol{\chi}(t) - \hat{\boldsymbol{\chi}}(t)$ converges to zero.

We finally remark that, due to the nonlinear nature of the estimation and servoing schemes, stability of each individual block does not imply stability of their composition (one cannot invoke the separation principle only valid for linear time-invariant systems), and thus some additional analysis should be performed to assess the overall closed-loop stability of our solution[2]. The reported experiments nevertheless showed a promising level of robustness in this sense.

IV. EXPERIMENTAL RESULTS

In this section we report the results of several experiments meant to illustrate the approach described in the previous sections. All experiments were run by making use of a greyscale camera attached to the end-effector of a 6-dofs Gantry robot. The camera has a resolution of 640×480 px and a framerate of 30 fps. Since the robot only accepts velocity commands, the acceleration signal generated by the proposed controller was numerically integrated before being sent to the robot. The controller (and its internal states) was updated at 1 kHz, while the commands were sent to the robot at 100 Hz. All the image processing and feature tracking were implemented via the open-source ViSP library [6].

As visual task, we considered the regulation of $N = 4$ point features $\boldsymbol{p}_i$. We then have $\boldsymbol{s} = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_N) \in \mathbb{R}^m$, and $\boldsymbol{L}_{\boldsymbol{s}} = (\boldsymbol{L}_{\boldsymbol{s}_1}, \ldots, \boldsymbol{L}_{\boldsymbol{s}_N}) \in \mathbb{R}^{m\times 6}$, $m = 8$, with $\boldsymbol{L}_{\boldsymbol{s}_i}$ being the 2×6 interaction matrix associated to the i-th point [1]. As for vector $\boldsymbol{\chi}$, it is $\boldsymbol{\chi} = (\chi_1, \ldots, \chi_N) \in \mathbb{R}^p$, $p = 4$, where $\chi_i = 1/Z_i$ as explained at the end of Sec. II. The points were positioned at the vertices of a square of 0.25 m size.

Each feature point is characterized by its eigenvalue $\sigma_{1,i}^2$ and the associated matrices $\boldsymbol{J}_{v,i}$, $\boldsymbol{J}_{s,i}$, see (4). In order to optimize the estimation of the whole vector $\boldsymbol{\chi}$ (the depth of all points), we simply aimed at maximizing the sum $\sigma^2 = \sum_{i=1}^N \sigma_{1,i}^2$ and thus modified (7–8) as

$$\dot{(\sigma^2)} = \sum_{i=1}^N \boldsymbol{J}_{v,i}\boldsymbol{J}_C\ddot{\boldsymbol{q}} + \sum_{i=1}^N \left(\boldsymbol{J}_{v,i}\dot{\boldsymbol{J}}_C\dot{\boldsymbol{q}} + \boldsymbol{J}_{s,i}\dot{\boldsymbol{s}}_i\right)$$

and

$$\ddot{\boldsymbol{q}}_\sigma = k_\sigma \boldsymbol{J}_C^T \textstyle\sum_{i=1}^N \boldsymbol{J}_{v,i}^T - \left(\sum_{i=1}^N \boldsymbol{J}_{v,i}\boldsymbol{J}_C\right)^\dagger \sum_{i=1}^N \left(\boldsymbol{J}_{v,i}\dot{\boldsymbol{J}}_C\dot{\boldsymbol{q}} + \boldsymbol{J}_{s,i}\dot{\boldsymbol{s}}_i\right). \tag{17}$$

The first set of experiments shows the results of having coupled a servoing controller with an active SfM strategy.

[2]However, we also note that similar theoretical difficulties affect most of the robotics applications in which an estimation step is plugged into the loop (e.g., whenever exploiting an Extended Kalman Filter for feeding a motion controller with the reconstructed state).

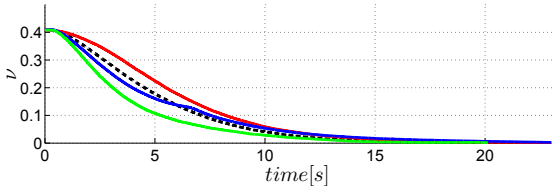


Fig. 2: Behavior of the error norm $\nu(t)$ when using: the full strategy of Secs. III-B and III-C and the estimated $\hat{\boldsymbol{\chi}}(t)$ (case 1 – blue line); the classical controller (9) and the estimated $\hat{\boldsymbol{\chi}}(t)$ (case 2 – red line); the classical controller (9) by employing $\hat{\boldsymbol{\chi}} = \boldsymbol{\chi}^*$ (case 3 – green line); the classical controller (9) by using the real $\boldsymbol{\chi}(t)$ (case 0 – black dashed line)

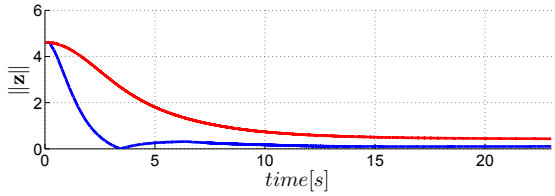


Fig. 3: Behavior of the estimation error $\boldsymbol{z}(t) = \boldsymbol{\chi}(t) - \hat{\boldsymbol{\chi}}(t)$ when actively optimizing the camera motion for improving the estimation convergence rate (case 1 – blue line) and when not optimizing the camera motion (case 2 – red line).

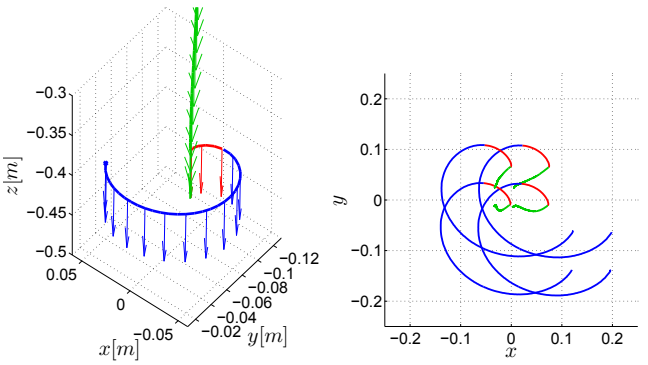
To this end, we report in Fig. 2 the evolution of the error norm $\nu(t)$ for the following four cases:

1) the full strategy (three phases) illustrated in the previous section is implemented. The estimator (2) is concurrently run to provide an estimation $\hat{\boldsymbol{\chi}}(t)$ to all the control terms. This case then involves the active optimization of the camera motion for improving, as much as possible, the convergence rate of the estimation error $\boldsymbol{z}(t) = \boldsymbol{\chi}(t) - \hat{\boldsymbol{\chi}}(t)$ (case 1 – blue line in the plot);
2) controller (9) is implemented for *all* the task duration while observer (2) is still run in parallel for generating $\hat{\boldsymbol{\chi}}(t)$. Thus, in this case no optimization of the estimation error convergence is performed, but $\hat{\boldsymbol{\chi}}(t)$ is still estimated during the resulting camera motion (case 2 – red line in the plot);
3) controller (9) is again implemented for all the task duration, but by now employing $\hat{\boldsymbol{\chi}}(t) = \boldsymbol{\chi}^* = const$, that is, the value of $\boldsymbol{\chi}$ at the desired pose (case 3 – green line in the plot).
4) as a reference 'ground truth', the behavior of controller (9) when using the *real* value of $\boldsymbol{\chi}(t)$ is also reported (case 0 - black line in the plot).

The following gains and thresholds were used in the experiments: $\alpha = 1500$, $k_p = 0.16$, $k_v = 0.8$ in (9) and (10), $k_{d_1} = 27$, $k_{d_2} = 10$, $\bar{\nu}_T = 0.14$, $\nu_T = 0.12$, $k_\sigma = 0.85$, and $\delta_T = 0.02$ (only for case 1). Vector $\hat{\boldsymbol{\chi}}$ was initialized with $\hat{Z} = 0.2$ for all point (for cases 1 and 2),

Finally, Fig. 3 shows the behavior of $\|\boldsymbol{z}(t)\|$ in cases 1 and 2 using the same color code of Fig. 2, while Fig. 4 depicts the camera and feature trajectory for the first experiment (case 1). We also invite the reader to watch the accompanying video for a better visualization of the robot motion.

Let us first consider Fig. 3: we can clearly note how



(a) (b)

Fig. 4: Fig. (a): camera 3D trajectory during case 1 with arrows representing the camera optical axis. The three phases of Sec. III-C are indicated by different colors: blue – phase 1), red – phase 2), green – phase 3). Fig. (b): trajectory of the four point features in the image plane during case 1 with the same color code.

the optimization action present in case 1 (blue line) allows to obtain a significantly quicker convergence of $\|\boldsymbol{z}(t)\|$ w.r.t. case 2 (red line). Indeed, in case 1 $\|\boldsymbol{z}(t)\|$ converges towards zero in about 4 sec and, additionally, it keeps a lower steady-state value w.r.t. case 2 (thus the depths Z_i are estimated faster and more accurately in case 1). A related pattern can be found in Fig. 2: note, in fact, how in case 1 the behavior of $\nu(t)$ (blue line) quickly matches the ideal one of case 0 (black line) because of the fast convergence of $\|\boldsymbol{z}(t)\|$. In case 2 (red line), the matching with case 0 is still obtained, but much later in the plot (at about 10 sec w.r.t. 6 sec). Note also the higher initial overshoot of case 2 compared to case 1. Finally, case 3 (green line) almost never reaches the ideal behavior of case 0 if not at the end of the motion as expected (since only in this case it is $\boldsymbol{\chi}(t) \approx \boldsymbol{\chi}^*$).

The trajectory depicted in Fig. 4 is also helpful in understanding the effects of the optimization action on the estimation error convergence during case 1: note, indeed, how the camera initially moves along an almost planar spiral (blue line) because of the null-space term (17) which imposes a motion maximizing the estimation of the point depths. This also allows to appreciate the benefits of having employed the norm controller (10) during the first phase: thanks to the large redundancy granted by this controller, the camera is free to perform very 'unusual' motions while still guaranteeing correct convergence of the error norm. For completeness, the red line in the plot depicts phase 2) of the switching strategy (the alignment of vectors $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$), while the green line represents phase 3) (use of the classical controller (9)). Finally in Fig. 4b the trajectory of the point features on the image plane are reported with the same color code: clearly only in phase 3), when the full error controller (9) is used, the points correctly move in (approximately) straight lines toward their desired positions.

In the second set of experiments we instead show the importance of having introduced phase 2) in the switching strategy of Sec. III-C (i.e., of having enforced alignment of $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$). To this end, Fig. 5a shows the behavior of the error

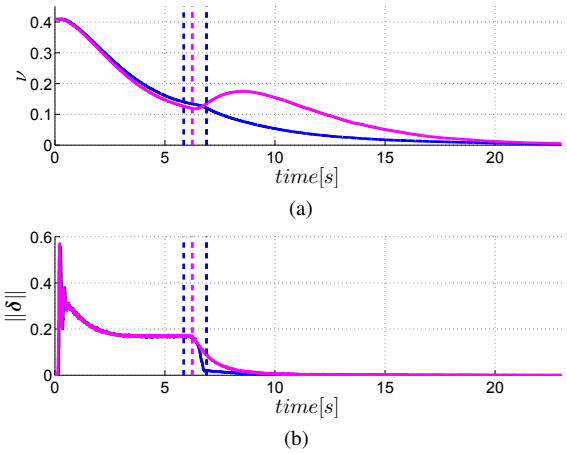


Fig. 5: Behavior of the error norm $\nu(t)$ (Fig. (a)) and of $\|\boldsymbol{\delta}\|$, the measure of misalignment between vectors $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$ (Fig. (b)). In both plots, the blue lines represent the behavior of case 1 (full implementation of the switching strategy of Sec. III-C), while magenta lines represent the direct switch from phase 1) to phase 3) without the action of vector $\boldsymbol{r}$ in (15).

norm $\nu(t)$ of the previous case 1 (blue line) together with the behavior of $\nu(t)$ when *not* implementing phase 2) but just switching from phase 1) to phase 3) (magenta line). The two blue vertical lines represent the switch from phase 1) to phase 2) and from phase 2) to phase 3). Note how, in this second situation, the error norm $\nu(t)$ has a large overshoot when switching to phase 3) due to the misalignment of vectors $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$, an overshoot clearly not present in case 1. Finally, Fig. 5b shows the behavior of $\|\boldsymbol{\delta}\|$, the measure of misalignment among $\boldsymbol{e}$ and $\dot{\boldsymbol{e}}$. Note how, in case 1, $\|\boldsymbol{\delta}\|$ is correctly made negligible at the end of phase 2) thanks to the action of vector $\boldsymbol{r}$ in (15). These results then fully confirm the theoretical analysis of the closed-loop behavior under the control strategy of Sec. III.

V. CONCLUSIONS

In this paper a strategy for optimally coupling a visual servoing task with an active SfM algorithm has been presented. In particular, we showed how to implement an active optimization of the SfM error within the null-space of a IBVS task. To this end, we employed a second-order version of the norm controller originally introduced in [4] for maximizing task redundancy, and provided a thorough analysis of the closed-loop convergence behavior when employing a suitable switching strategy for avoiding the typical singularities of the method. The experiments clearly showed the benefits of the proposed strategy in (i) obtaining a faster convergence of the structure estimation error and (ii) imposing a better closed-loop behavior to the servoing controller w.r.t. an ideal response obtained in perfect conditions.

In the future we plan to apply the proposed general technique to more complex visual servoing tasks and structure estimation problems, possibly involving additional constraints such as presence of obstacles, joint limit avoidance, as well as applications to mobile (ground/flying) robotics.

APPENDIX

Lemma 1.1: Condition (14) holds if and only if, at the switching time t_1, vectors $\boldsymbol{e}(t_1)$ and $\dot{\boldsymbol{e}}(t_1)$ are *parallel*.

Proof: Let $\boldsymbol{\Phi}(t) = [\Phi_{ij}(t)] \in \mathbb{R}^{2\times 2}$ be the state-transition matrix associated to the linear time-invariant system (12). From classical system theory [7], it is $\nu_{\|\boldsymbol{e}\|}(t) = \Phi_{11}(t-t_1)\nu(t_1) + \Phi_{12}(t-t_1)\dot{\nu}(t_1), \quad \forall t \geq t_1$. We note that (13) is governed, component-wise, by the same dynamics of (12). Therefore, the solution of (13) is

$$\boldsymbol{e}^*(t) = \Phi_{11}(t-t_1)\boldsymbol{e}(t_1) + \Phi_{12}(t-t_1)\dot{\boldsymbol{e}}(t_1), \quad \forall t \geq t_1. \tag{18}$$

Assuming $\boldsymbol{e}(t_1)$ and $\dot{\boldsymbol{e}}(t_1)$ are parallel, vector $\dot{\boldsymbol{e}}(t_1)$ can be expressed as

$$\dot{\boldsymbol{e}}(t_1) = \|\dot{\boldsymbol{e}}(t_1)\|\frac{\boldsymbol{e}(t_1)}{\|\boldsymbol{e}(t_1)\|} = \|\dot{\boldsymbol{e}}(t_1)\|\frac{\boldsymbol{e}(t_1)}{\nu(t_1)}. \tag{19}$$

Therefore, (18) becomes

$$\boldsymbol{e}^*(t) = \left(\Phi_{11}(t-t_1) + \Phi_{12}(t-t_1)\frac{\|\dot{\boldsymbol{e}}(t_1)\|}{\nu(t_1)}\right)\boldsymbol{e}(t_1), \quad \forall t \geq t_1,$$

resulting in

$$\begin{aligned}\|\boldsymbol{e}^*(t)\| = \nu^*(t) &= \left(\Phi_{11}(t-t_1) + \Phi_{12}(t-t_1)\frac{\|\dot{\boldsymbol{e}}(t_1)\|}{\nu(t_1)}\right)\|\boldsymbol{e}(t_1)\| \\ &= \left(\Phi_{11}(t-t_1) + \Phi_{12}(t-t_1)\frac{\|\dot{\boldsymbol{e}}(t_1)\|}{\nu(t_1)}\right)\nu(t_1) \\ &= \Phi_{11}(t-t_1)\nu(t_1) + \Phi_{12}(t-t_1)\|\dot{\boldsymbol{e}}(t_1)\|, \quad \forall t \geq t_1.\end{aligned} \tag{20}$$

Now, being $\nu = \|\boldsymbol{e}\|$ and using (19), it is

$$\dot{\nu}(t_1) = \frac{\boldsymbol{e}^T(t_1)\dot{\boldsymbol{e}}(t_1)}{\nu(t_1)} = \|\dot{\boldsymbol{e}}(t_1)\|\frac{\boldsymbol{e}^T(t_1)\boldsymbol{e}(t_1)}{\nu^2(t_1)} = \|\dot{\boldsymbol{e}}(t_1)\|.$$

Plugging $\|\dot{\boldsymbol{e}}(t_1)\| = \dot{\eta}(t_1)$ in (20) finally yields $\nu^*(t) = \Phi_{11}(t-t_1)\nu(t_1) + \Phi_{12}(t-t_1)\|\dot{\nu}(t_1), \quad \forall t \geq t_1$, thus showing that $\nu^*(t) \equiv \nu_{\|\boldsymbol{e}\|}(t)$, i.e. fulfilment of condition (14). ■

REFERENCES

[1] F. Chaumette and S. Hutchinson, "Visual servo control, Part I: Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[2] A. Petiteville, M. Courdesses, V. Cadenat, and P. Baillion, "On-line estimation of the reference visual features application to a vision based long range navigation task," in *2010 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2010, pp. 3925–3930.

[3] R. Spica and P. Robuffo Giordano, "A Framework for Active Estimation: Application to Structure from Motion," in *52nd IEEE Conf. on Decision and Control*, 2013.

[4] M. Marey and F. Chaumette, "A new large projection operator for the redundancy framework," in *2010 IEEE Int. Conf. on Robotics and Automation*, 2010, pp. 3727–3732.

[5] A. De Luca, G. Oriolo, and B. Siciliano, "Robot redundancy resolution at the acceleration level," *Laboratory Robotics and Automation*, vol. 4, no. 2, pp. 97–106, 1992.

[6] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, pp. 40–52, 2005.

[7] T. Kailath, *Linear Systems*. Prentice Hall International, 1998.