Occlusion-Free Visual Servoing for the Shared Autonomy Teleoperation of Dual-Arm Robots

Davide Nicolis, Marco Palumbo, Andrea Maria Zanchettin, and Paolo Rocco

***Abstract*—Visual servoing in telerobotics provides information to the operator about the remote location and assists in the task execution to reduce stress on the user. Occlusions in this scenario can, on the one hand, lead to the visual servoing failure, and, on the other hand, degrade the user experience and navigation performance due to the obstructed vision of elements of interest. In this letter, we consider a teleoperation system composed of two robot arms where one is remotely operated, while the other is autonomous and equipped with an eye-in-hand camera sensor. We propose a reactive unified convex optimization based controller that allows the execution of occlusion-free tasks by autonomously adjusting the camera robot, so as to keep the teleoperated one in the field of view. The occlusion avoidance is formulated inside the optimization as a constraint in the image space, it is formally derived from a collision avoidance analogy and made robust against noisy measurements and dynamic environment. A state machine is used to switch the control policy whenever an occlusion might occur. We validate our approach with experiments on a 14 d.o.f. dual-arm ABB YuMi robot equipped with an red, green, blue (RGB) camera and teleoperated by a 3 d.o.f. Novint Falcon device.**

***Index Terms*—Visual Servoing, telerobotics and teleoperation, optimization and optimal control, occlusion avoidance.**

I. Introduction

ALTHOUGH advancements in robot control and AI are pushing towards fully autonomous robotic agents to replace the human component in many tasks, human intervention remains unavoidable whenever the application demands complex high level decision making, such as exploration in unstructured and cluttered environments, or for security and ethics reasons, such as nuclear plant decommissioning or delicate surgeries. In these cases, telerobotics provides a solution by enhancing the human user capabilities and allowing him/her to operate at a distance, while staying safe from possible hazards.

Under these circumstances, it is of paramount importance to put the operator in condition of understanding what is happening at the remote location with the aid of visual cues, while further exploiting them to improve the system usability. Visual servoing has been extensively considered in teleoperation systems. In [1] the authors used classical visual servoing to aid a physician in the execution of a tele-ecography. Image based visual servoing (IBVS) was instead applied to two robots, one teleoperated and the other autonomous and equipped with a camera, for remote nuclear waste sorting and handling in [2].

Critical issues arise in visual servoing whenever an object of interest is occluded either by itself, by another object, or by the robot arm, or the corresponding image features escape the camera field of view (FoV). When this occurs, the relevant object features are lost, which may result in a control failure. Furthermore, in a teleoperation setup this is also undesirable since it removes direct visual feedback of the object in the scene, making navigation more difficult and less intuitive.

Some authors focused on feature estimation to recover control properties during occlusions, either by estimating point feature depth with nonlinear observers [3], or by reconstructing a dynamic object characterized by point and line features with a geometric approach [4].

Another approach to deal with occlusions is to plan a camera trajectory that would altogether avoid their occurrence. Kazemi *et al.* [5] presented an extensive overview of the main path-planning techniques used in visual servoing to guarantee occlusion-free and collision-free trajectories, as well as to consider FoV limitations.

Other authors used potential fields [6] or variable weighting LQ control laws [7] to preserve visibility and avoid self-occlusions. Although these approaches are suitable for real-time implementation, they may exhibit local minima and possible unwanted oscillations.

Global path-planning can overcome these limitations by computing a priori occlusion-free trajectories, however it often requires long computational time and accurate workspace knowledge, unsuitable for reactive motions in dynamic environments [8].

Time complexity is a major obstacle also in optimization-based path-planning, where the solver computes a suitably parametrized optimal trajectory, while respecting camera constraints [9]. These approaches have the merit of considering arbitrary constraints but are hardly applicable in complex contexts, as the optimization might become non-convex.

In [10] the authors proposed the so called qualitative visual servoing to trade-off feature positioning and visibility. Nevertheless, in a teleoperation framework the resulting solution without proper redundancy management could result in undesired camera motions.

Manuscript received September 9, 2017; accepted December 16, 2017. Date of publication January 11, 2018; date of current version January 25, 2018. This letter was recommended for publication by Associate Editor A. Krupa and Editor F. Chaumette upon evaluation of the Reviewers' comments. *(Corresponding author: Davide Nicolis.)*

The authors are with the Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, Milan 20133, Italy (e-mail: davide.nicolis@polimi.it; marco1.palumbo@mail.polimi.it; andreamaria.zanchettin@polimi.it; paolo.rocco@polimi.it).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LRA.2018.2792143

Folio *et al.* [11] proposed a solution to the occlusion avoidance problem for mobile robots by switching controller whenever an imminent occlusion is detected. The algorithm tries to push the occluding object outside the camera FoV via IBVS, however it is limited to planar camera motions and rotations around the vertical axis.

Quadratic programming techniques have seen increasing popularity in the robotic community due to their ability to assign robot motions via the simple definition of appropriate equality and inequality constraints [12]. These solutions adapt the robot motion in real-time based on sensor measurements. We previously proposed a teleoperation controller based on these algorithms and we showed how FoV visibility constraints can be easily included in the formulation [13]. Other authors employed quadratic optimization in complex systems of high dimensionality, such as the whole-body control of a humanoid robot [14], where visual servoing was integrated with FoV and occlusion constraints, although in a simple example with a wall occluding the robot gaze.

With the present work we aim to provide a solution to the occlusion problem for dual-arm systems where one robot is remotely operated, while the other is equipped with a camera and in charge of maintaining at all times the teleoperated tool and goal visible. We propose a controller that aims at avoiding occlusions altogether. This allows for a more user-friendly teleoperation, since the operator is always aware of the remote location state.

Furthermore, we assume the environment, as well as the operator intentions, to be dynamic and potentially unknown. For this reason we adopt a reactive approach, where risks of occlusion are dealt with on-line, without previous path-planning. The optimization-based controller of [13] is extended with the inclusion of an additional robot arm and occlusion constraints able to model arbitrary convex occluding objects. The constraint formulation is derived by highlighting the analogy between occlusion avoidance in the image space and collision avoidance in the 3D space.

Finally, we define a finite state machine (FSM) that detects the risk of occlusion and switches to a controller that performs an evasive camera motion, while keeping an intuitive display of the remote scene and control of the teleoperated robot for the user.

We validate our approach experimentally with a reaching task executed on a dual-arm 14 d.o.f. ABB YuMi robot equipped with an eye-in-hand RGB camera, and one arm teleoperated by a 3 d.o.f. Novint Falcon device. The robustness of the approach is tested against noise on the image features, as well as by considering dynamic occluding objects and goals. To the best of our knowledge, this is one of the first results that explicitly deal with real-time occlusion avoidance applied to dual-arm teleoperation and considers dynamic environments as well as a priori unplanned teleoperator movements. Possible applications may include handling of hazardous material, and surveying in barely accessible disaster area, where debris can occlude the camera view. Although we focused on telemanipulation systems, we argue that the designed occlusion constraints can be also employed in fully autonomous visual servoing (e.g., where continuous object tracking is required).

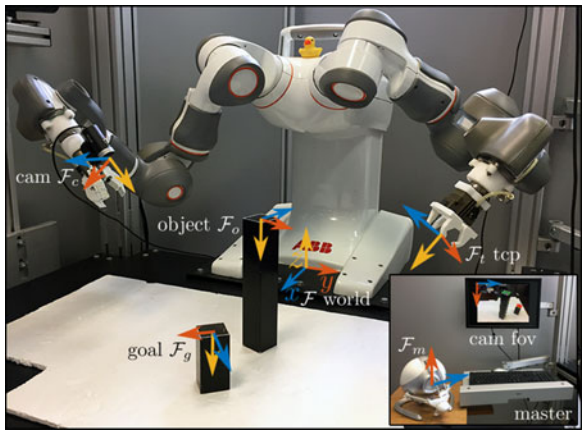

Fig. 1. The experimental setup.

The letter is organized as follows. Section II presents the problem settings along with a short background on visual servoing and optimization-based control, while Section III introduces the occlusion avoidance constraint and its robust formulation. In Section IV we define the FSM responsible for switching the controller and solving the impending occlusion. Finally, Section V shows the validation results on our experimental platform.

II. Problem Setting

The robotic system is composed of two slave robot arms and a master device (Fig. 1). One arm is remotely operated by a user via the master device that controls the robot TCP velocity. The other arm is autonomous and camera-equipped with the purpose of producing an optimal view of the workspace for the user by keeping the teleoperated tool and the goal inside the FoV, as well as avoiding occlusions. The camera feed is the only visual cue available to the user. Overall, the requirements for the system during a remote reaching task are the following:

1) Autonomous continuous camera positioning. The user shouldn't worry about controlling the camera robot.
2) Kinematic coordination (tracking) between master and slave teleoperated robot.
3) Teleoperated TCP and user-selected goal visible at all times in the camera FoV.
4) Autonomous camera repositioning to avoid occlusions due to additional objects in the scene.
5) Natural and intuitive camera motion and reference mapping of the master device in the camera frame. The user should clearly understand what is happening at the remote scene, and how to influence the system behavior.

We define the following frames as shown in Fig. 1: $\mathcal{F}$ is the world frame, $\mathcal{F}_m$ is the master device base frame, $\mathcal{F}_c$ is the camera frame, $\mathcal{F}_t$ is the tool frame of the teleoperated arm, $\mathcal{F}_g$ is the frame of the goal that the user wishes to reach, and $\mathcal{F}_o$ is the frame of an object that can possibly occlude the tool or the goal.

For the camera we adopt the standard pinhole model. $\boldsymbol{\pi}$ identifies the image plane parallel to the (x_c, y_c) plane, and λ is the camera focal length. Given a 3D point $P(x^c, y^c, z^c)$ expressed in $\mathcal{F}_c$, the corresponding point $p(u, v)$ in $\boldsymbol{\pi}$ is given by the

projection equations

$$u = \frac{\lambda x^c}{d_x z^c} + o_u, \quad v = \frac{\lambda y^c}{d_y z^c} + o_v \tag{1}$$

where (d_x, d_y) are the pixels dimensions in the (x_c, y_c) coordinates, and (o_u, o_v) offset the position of the first pixel.

For camera control and occlusion avoidance we employ IBVS. The regulation is done directly at the image feature level, for this reason it is necessary to relate the features movements in the image plane to the robot joint velocities. Given a vector of generic image features of an object $\boldsymbol{s}_o$, its time derivative depends on the joint velocities: $\dot{\boldsymbol{s}}_o = \boldsymbol{L}_o \dot{\boldsymbol{x}}_c^c = \boldsymbol{L}_o \boldsymbol{J}_c^c \dot{\boldsymbol{q}}_c = \boldsymbol{P}_{c,o} \dot{\boldsymbol{q}}_c$, where $\boldsymbol{L}_o$ is the interaction matrix, $\boldsymbol{J}_c^c$ the camera Jacobian expressed in $\mathcal{F}_c$ and $\dot{\boldsymbol{q}}_c$ the camera robot joint velocities.

In the particular case where the teleoperated TCP is considered, the corresponding features are the image point coordinates $\boldsymbol{p}_t = [u_t \ v_t]^T$, and its motion can be compensated since it depends on the arm joint velocities $\dot{\boldsymbol{q}}_t$.

$$\begin{aligned} \dot{\boldsymbol{p}}_t &= \boldsymbol{L}_t \begin{bmatrix} \boldsymbol{v}_c^c - \boldsymbol{v}_t^c \\ \boldsymbol{\omega}_c^c \end{bmatrix} = \boldsymbol{L}_t \boldsymbol{J}_c^c \dot{\boldsymbol{q}}_c + \boldsymbol{L}_t \begin{bmatrix} -\boldsymbol{R}^c & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{J}_t \dot{\boldsymbol{q}}_t \\ &= \begin{bmatrix} \boldsymbol{P}_{c,t} & \boldsymbol{P}_{t,t} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_c \\ \dot{\boldsymbol{q}}_t \end{bmatrix} = \boldsymbol{P}_t \dot{\boldsymbol{q}} \end{aligned} \tag{2}$$

where $\boldsymbol{v}_t^c$ is the tool translational velocity expressed in $\mathcal{F}_c$, $\boldsymbol{J}_t$ the tool Jacobian in $\mathcal{F}$, and $\boldsymbol{R}^c$ the rotation matrix from world to camera frame.

Given the image features reference $\boldsymbol{p}_t^*$, and the error dynamics $\dot{\boldsymbol{p}}_t = \boldsymbol{P}_t \dot{\boldsymbol{q}} = k_p(\boldsymbol{p}_t^* - \boldsymbol{p}_t)$, the classic IBVS controller produces the following control law

$$\dot{\boldsymbol{q}} = k_p \boldsymbol{P}_t^{\dagger} (\boldsymbol{p}_t^* - \boldsymbol{p}_t) + \boldsymbol{N}_t \dot{\boldsymbol{q}}_0 \tag{3}$$

where $\boldsymbol{P}_t^{\dagger}$ is the pseudo-inverse of $\boldsymbol{P}_t$ and $\boldsymbol{N}_t$ its null-space projector, k_p is the controller gain, and $\dot{\boldsymbol{q}}$ the joint reference vector for the low level controllers of the two robots. $\dot{\boldsymbol{q}}_0$ can be used to exploit the remaining degrees of freedom.

For the considered system, the second requirement is kinematic coordination between master and slave devices, which can be implemented via the following error dynamics

$$\dot{\boldsymbol{x}}_t = \begin{bmatrix} \boldsymbol{0} & \boldsymbol{J}_t \end{bmatrix} \dot{\boldsymbol{q}} = k_p(\boldsymbol{x}_t^* - \boldsymbol{x}_t) + \dot{\boldsymbol{x}}_t^* \tag{4}$$

$\boldsymbol{x}_t$ and $\boldsymbol{x}_t^*$ are the teleoperated robot pose and its reference, respectively. Substituting $\dot{\boldsymbol{q}}$ in (4) with (3) yields $\dot{\boldsymbol{q}}_0$ by means of the pseudo-inverse, and the final joint velocities are obtained by substituting back in (3). The resulting control law will move the camera to regulate $\boldsymbol{p}_t$ in the FoV, while allowing the user to teleoperate the robot as desired.

Unfortunately, this standard approach cannot consider specifications that do not require an explicit regulation, but just a generic limitation. This is the case for the considered system, where we want to ensure that the relevant image features remain inside the FoV (visibility) and are not occluded by other objects (occlusion avoidance), but we do not want to explicitly define a reference for these entities. For this reason, in this letter we are making use of an optimization-based formalism that allows the explicit definition of motion constraints via both equalities (master tracking, visual servoing) and inequalities (visibility, occlusion avoidance).

The generic optimization problem is defined as a convex QP as follows

$$\dot{\boldsymbol{q}}^* = \arg \min_{\dot{\boldsymbol{q}}} \sum_{i=1}^{n} \|\boldsymbol{A}_i \dot{\boldsymbol{q}} - \boldsymbol{b}_i\|_{\boldsymbol{W}}^2, \quad \text{s.t. } \boldsymbol{E}\dot{\boldsymbol{q}} \leq \boldsymbol{g} \tag{5}$$

where the cost function is formed with the tasks to be executed, while satisfying the constraints characterized by matrix $\boldsymbol{E}$ and vector $\boldsymbol{g}$. $\boldsymbol{W}$ is a weighting matrix, that plays a role whenever the number of tasks is higher than the available d.o.f., by differently prioritizing them. The features error dynamics and (4) are in the form required by (5) and can be directly used to ensure the visual servoing and master tracking requirements.

Differently from what shown here, the control variables for the proposed controller will be the joint accelerations. This can be achieved simply by considering a second order error dynamics for the quantities to be regulated, or by applying Grönwall's lemma for the constraints [12].

III. Occlusion Avoidance Constraint

In this section we present the formulation of the occlusion avoidance constraint in a form compatible with the presented optimization problem (5).

Let us consider the feature point $\boldsymbol{p}_t$ for which we want to avoid the occlusion, and the feature points characterizing a potentially occluding convex object. Depending on the object geometry, these points are connected to each other and they define a polytope in $\boldsymbol{\pi}$, that $\boldsymbol{p}_t$ is not allowed to enter if its depth is higher than that of the object points. What we need is a formulation of this requirement in terms of the two robots velocities.

In [15], it has been shown how a safety constraint avoiding collision between a robot rigid link and the human can be derived from geometric and kinematic arguments. The authors managed to obtain a minimum distance criterion in the form of an inequality on robot velocities. Here we conjugate this criterion in the image plane. If $\boldsymbol{p}_t$ starts from a visible position, we can avoid the occlusion by having the feature point never "collide" with the forbidden area.

A. Occlusion Avoidance Constraint Definition

Without loss of generality, we consider a single pair of connected object feature points defining the occlusion area boundary, the final set of constraints satisfying the occlusion avoidance requirement will be straightforwardly obtained by applying the following computations for every adjacent pair.

Fig. 2 shows the segment connecting two object feature points $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ along with the point $\boldsymbol{p}_t$ that should avoid the occlusion. For simplicity, we assume the segment's depth to be less than that of $\boldsymbol{p}_t$, so that an occlusion can indeed occur. If this is not the case, we can just consider the sub-segment satisfying this condition.

The position and velocity of the generic point $\boldsymbol{p}_s$ on the segment can be expressed as follows

$$\boldsymbol{p}_s = \boldsymbol{p}_a + s(\boldsymbol{p}_b - \boldsymbol{p}_a), \quad \dot{\boldsymbol{p}}_s = \dot{\boldsymbol{p}}_a + s(\dot{\boldsymbol{p}}_b - \dot{\boldsymbol{p}}_a) \tag{6}$$

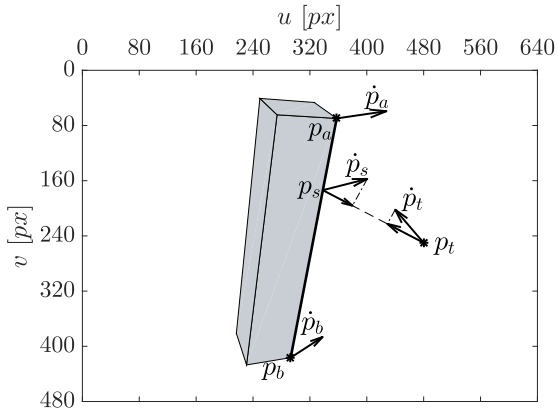

Fig. 2. Pictorial representation of the occlusion avoidance constraint in the image plane. The gray area identifies the projection of the occluding object.

where $s \in [0,1]$. By applying the minimum distance criterion we have

$$(\boldsymbol{p}_t - \boldsymbol{p}_s)^T(\boldsymbol{p}_t - \boldsymbol{p}_s) - t_b(\boldsymbol{p}_t - \boldsymbol{p}_s)^T(\dot{\boldsymbol{p}}_s - \dot{\boldsymbol{p}}_t) \geq 0, \forall s \in [0,1] \tag{7}$$

The first term on the left represents the squared distance between the feature of interest and each of the points on the segment delimiting the occlusion area, while the second one is proportional to the projection of the relative velocity of the two points on the line connecting them. t_b is a design parameter that relates to the maximum time required by the robot to bring to a halt the features in the image. Substituting (6) in (7), for the first term we have

$$\begin{aligned}(\boldsymbol{p}_t - \boldsymbol{p}_s)^T(\boldsymbol{p}_t - \boldsymbol{p}_s) &= \alpha s^2 + \beta s + \gamma \\ \alpha &= (\boldsymbol{p}_b - \boldsymbol{p}_a)^T(\boldsymbol{p}_b - \boldsymbol{p}_a) \\ \beta &= 2(\boldsymbol{p}_a - \boldsymbol{p}_t)^T(\boldsymbol{p}_b - \boldsymbol{p}_a) \\ \gamma &= (\boldsymbol{p}_a - \boldsymbol{p}_t)^T(\boldsymbol{p}_a - \boldsymbol{p}_t)\end{aligned} \tag{8}$$

Similarly, for the second term

$$\begin{aligned}-t_b(\boldsymbol{p}_t - \boldsymbol{p}_s)^T(\dot{\boldsymbol{p}}_s - \dot{\boldsymbol{p}}_t) &= \alpha' s^2 + \beta' s + \gamma' \\ \alpha' &= t_b(\boldsymbol{p}_b - \boldsymbol{p}_a)^T(\dot{\boldsymbol{p}}_b - \dot{\boldsymbol{p}}_a) \\ \beta' &= t_b[(\boldsymbol{p}_a - \boldsymbol{p}_t)^T(\dot{\boldsymbol{p}}_b - \dot{\boldsymbol{p}}_a) + (\boldsymbol{p}_b - \boldsymbol{p}_a)^T(\dot{\boldsymbol{p}}_a - \dot{\boldsymbol{p}}_t)] \\ \gamma' &= t_b(\boldsymbol{p}_a - \boldsymbol{p}_t)^T(\dot{\boldsymbol{p}}_a - \dot{\boldsymbol{p}}_t)\end{aligned} \tag{9}$$

Then the constraint can be rewritten as

$$(\alpha + \alpha')s^2 + (\beta + \beta')s + (\gamma + \gamma') \geq 0, \ \forall s \in [0,1] \tag{10}$$

For the previous equation to be valid $\forall s \in [0,1]$ it is sufficient to check the minima $\overline{s}$ of the parabola in this interval, given the current feature coordinates and velocities.

Note that, since the segment is not a rigid body in the image plane, it is possible to have $\alpha' \neq 0$, thus we cannot simply evaluate (7) by computing $\min_s \|\boldsymbol{p}_t - \boldsymbol{p}_s\|$ and checking the term on the right side only in $s = 0, 1$ as done in [15].

If $\alpha + \alpha' > 0$ then the parabola is convex and the minimum is a point in the $[0,1]$ interval, else if $\alpha + \alpha' < 0$ the parabola is concave and the minimum is necessarily one (or both) of the segment ends. Notice also that $\alpha + \alpha' = 0$ only if $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ are the same point or, given the current velocities, they will overlap in a time t_b.

By applying the relation between feature derivatives and joint velocities to the segment point features $\boldsymbol{p}_a, \boldsymbol{p}_b$, we have

$$\dot{\boldsymbol{p}}_a = \boldsymbol{P}_{c,a}\dot{\boldsymbol{q}}_c, \quad \dot{\boldsymbol{p}}_b = \boldsymbol{P}_{c,b}\dot{\boldsymbol{q}}_c \tag{11}$$

For point $\boldsymbol{p}_t$ we can proceed similarly, but without loss of generality in the following we will consider the case where $\boldsymbol{p}_t$ is the teleoperated tool point feature. By substituting the minimum coordinate $\overline{s}$, (2), and (11) in (10) we obtain the occlusion avoidance constraint in the form (5)

$$\begin{bmatrix} \boldsymbol{E}_c & \boldsymbol{E}_t \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_c \\ \dot{\boldsymbol{q}}_t \end{bmatrix} \geq g \tag{12}$$

$$\begin{aligned}\boldsymbol{E}_c = t_b \big\{ &(\boldsymbol{p}_b - \boldsymbol{p}_a)^T(\boldsymbol{P}_{c,b} - \boldsymbol{P}_{c,a})\overline{s}^2 \\ &+ \big[(\boldsymbol{p}_b - \boldsymbol{p}_a)^T(\boldsymbol{P}_{c,a} - \boldsymbol{P}_{c,t}) \\ &+ (\boldsymbol{p}_a - \boldsymbol{p}_t)^T(\boldsymbol{P}_{c,b} - \boldsymbol{P}_{c,a})\big]\overline{s} \\ &+ (\boldsymbol{p}_a - \boldsymbol{p}_t)^T(\boldsymbol{P}_{c,a} - \boldsymbol{P}_{c,t})\big\}\end{aligned} \tag{13}$$

$$\boldsymbol{E}_t = -t_b\left[(\boldsymbol{p}_b - \boldsymbol{p}_a)^T\overline{s} + (\boldsymbol{p}_a - \boldsymbol{p}_t)^T\right]\boldsymbol{P}_{t,t} \tag{14}$$

$$g = -(\alpha\overline{s}^2 + \beta\overline{s} + \gamma) \tag{15}$$

Note that if $\boldsymbol{p}_t$ were the feature of a generic point, in the previous equation we would have $\boldsymbol{P}_{t,t} = \boldsymbol{0}$ and thus $\boldsymbol{E}_t = \boldsymbol{0}$, this shows that the formulation is general and applicable also to autonomous single-arm VS, without teleoperator. Furthermore, if the occluding object dynamics are known or otherwise estimated, they can be directly considered in (11) by adding the terms $-\boldsymbol{L}_a[\boldsymbol{v}_a^{cT}\ \boldsymbol{0}^T]^T$, $-\boldsymbol{L}_b[\boldsymbol{v}_b^{cT}\ \boldsymbol{0}^T]^T$, where $\boldsymbol{v}_a^c$ and $\boldsymbol{v}_b^c$ are the velocities of the corresponding points on the 3D object. The same can be argued for the goal if it is the considered feature instead of the teleoperated TCP. Nonetheless, we will prove that this knowledge is, to a certain extent, unnecessary, thanks to the inherent robustness of the approach, which is discussed in the next section.

As stated at the end of Section II, the control variables for our system are the joint accelerations. If we apply Grönwall's lemma by defining $f = \boldsymbol{E}_c\dot{\boldsymbol{q}}_c + \boldsymbol{E}_t\dot{\boldsymbol{q}}_t - g$, the inequality $\dot{f} + \delta f \geq 0$ yields an acceleration-based constraint that satisfies and exponentially converges to (12), where δ is the speed of convergence. The final acceleration-based constraint is then the following one

$$\begin{aligned}\begin{bmatrix} \boldsymbol{E}_c & \boldsymbol{E}_t \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_c \\ \ddot{\boldsymbol{q}}_t \end{bmatrix} \geq -\delta \left(\begin{bmatrix} \boldsymbol{E}_c & \boldsymbol{E}_t \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_c \\ \dot{\boldsymbol{q}}_t \end{bmatrix} - g \right) \\ - \begin{bmatrix} \dot{\boldsymbol{E}}_c & \dot{\boldsymbol{E}}_t \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}_c \\ \dot{\boldsymbol{q}}_t \end{bmatrix} + \dot{g} = g'\end{aligned} \tag{16}$$

After repeating the procedure for all the segments defining the occlusion area, all the resulting inequality constraints can be immediately included in an optimization-based controller to prevent the occlusion event.

B. Robust Constraint Reformulation

The obtained constraint is able to ensure the occlusion avoidance in a nominal scenario. In real applications, sources of uncertainty, such as camera calibration and measurement noise, might actually produce a violation of the constraint and possibly an occlusion event. Additionally, the numerical approximations of the optimization discrete implementation, often produce chattering of the control variables, due to intermittent constraint activation. In practice these events make the constraint unreliable, since some non-recoverable occlusions could occur: the feature of interest might enter the forbidden region and remain inside, due to the constraint now working on the wrong side of the segments defining the polytope.

To solve this problem it is necessary to take this uncertainty directly into account in the constraint formulation. Assuming bounded uncertainties, we can follow the approach in [16] to confer robustness to the constraint. Given the vectors $\boldsymbol{\Delta}_p \in \mathbb{D}_p$ and $\boldsymbol{\Delta}_{\dot{p}} \in \mathbb{D}_{\dot{p}}$, respectively modeling the point features relative position and velocity uncertainties, (7) can be rewritten as follows

$$(\boldsymbol{p}_t - \boldsymbol{p}_s + \boldsymbol{\Delta}_p)^T (\boldsymbol{p}_t - \boldsymbol{p}_s + \boldsymbol{\Delta}_p) - t_b (\boldsymbol{p}_t - \boldsymbol{p}_s + \boldsymbol{\Delta}_p)^T (\dot{\boldsymbol{p}}_s - \dot{\boldsymbol{p}}_t + \boldsymbol{\Delta}_{\dot{p}}) \geq 0, \forall s \in [0,1], \forall \boldsymbol{\Delta}_p \in \mathbb{D}_p, \forall \boldsymbol{\Delta}_{\dot{p}} \in \mathbb{D}_{\dot{p}} \tag{17}$$

Then, what we are asking is that the constraint should be satisfied for a whole set of states and controls, an approach often used in robust MPC. Note that this is not the same as simply adding a margin to the nominal constraint. In fact, by expanding the calculation, uncertainty terms that depend on the state and control appear and cannot be easily bounded.

Similarly to the nominal case, we have to find the minimum of this multivariable function to obtain the worst case scenario, and a form compatible with the optimization solver.

Substituting the minimum along with (2) and (11) in (17), and applying Grönwall's lemma gives the new robust constraint $\begin{bmatrix} \tilde{\boldsymbol{E}}_c & \tilde{\boldsymbol{E}}_t \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}}_c \\ \ddot{\boldsymbol{q}}_t \end{bmatrix} \geq \tilde{g}'$, where the tilde indicates matrices accounting for the uncertainty. Notice that, with this approach we are also implicitly making the constraint more robust against unmodeled movements of the considered 3D objects, since they turn out to be additional feature point motions unforeseen by the control. Additionally, interaction matrix uncertainties are partially considered, since we can write $\dot{\boldsymbol{p}} = \boldsymbol{P}\dot{\boldsymbol{q}} = \boldsymbol{P}_n\dot{\boldsymbol{q}} + \delta\boldsymbol{P}\dot{\boldsymbol{q}} = \boldsymbol{P}_n\dot{\boldsymbol{q}} + \boldsymbol{\Delta}_{\dot{p}}$, where $\boldsymbol{P}_n$ is the nominal matrix, and the uncertainty $\delta\boldsymbol{P}$ can be bounded for reasonable estimates of the interaction matrix terms.

IV. Finite State Machine and Controllers

The set of constraints derived in the previous section guarantees that the robot system will not perform a trajectory leading to an occlusion. However, the simple inclusion of such constraints does not ensure a natural movement of the camera for a reaching task. In fact, imagine to control the robot TCP to a point in the image, and teleoperate the arm to induce an occlusion. Upon constraint activation, the controller will produce a camera motion around the corresponding object segment in the 3D space to avoid the occlusion (Fig. 3(a)). Although effective, we have no control over this motion and no guarantees that it will be optimal from the user point of view (e.g., instead of going around the object, we could simply go above it, see Fig. 1).

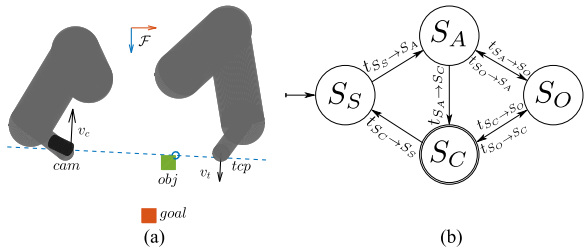

Fig. 3. Left: system in the $x-y$ plane. To avoid the occlusion the camera pivots around the object. Right: the finite state machine for the teleoperation system.

For this reason we divide the reaching task in four different phases, where each one employs its own controller to regulate different sets of quantities and generate a user-friendly camera behavior. To model the phase transitions we employ a finite state machine, as sketched in Fig. 3(b). Notice that since the controllers outputs are the joint accelerations, we can guarantee the continuity of the velocity during the transitions. Therefore, it is not mandatory to employ task sequencing and smoothing techniques [17] to avoid mechanical resonances excitation, although they can be integrated in the proposed approach if also continuous accelerations are required.

In the following we provide a description of each phase.

A. S_S: Setup State

The setup phase is the system starting state. We assume that the user has previously selected a goal from the image, and that at the start it is visible in the FoV.

In this state we autonomously regulate the tool and the goal to predefined positions in the image without any user input. We divide the image in four regions, one quadrant will be occupied by the TCP while the opposite one by the goal. For the tool $\boldsymbol{p}_t$ we choose the nearest region in the plane, while the goal image center of mass (CoM) $\boldsymbol{p}_{g,com}$ position is automatically chosen.

A formulation of these requirements compatible with the optimization problem (5), and with the joint accelerations as control variables, can be obtained by defining second order error dynamics for each regulated quantity. Via the appropriate interaction matrix and robot Jacobian it is possible to make the acceleration dependence explicit within the optimization. This procedure will be used for all the following requirements. As for the constraints, here we include visibility for the goal $(\boldsymbol{E}_{g,fov}, \boldsymbol{g}_{g,fov})$ and robot joint limits $(\boldsymbol{E}_{lim}, \boldsymbol{g}_{lim})$, to ensure that the goal is kept at all times within the camera FoV. The redundancy is solved by minimizing the joint velocities.

In compact notation, the optimization has the following cost function and constraints

$$\text{Cost fun.: } \|\boldsymbol{e}_t\|^2_{\boldsymbol{W}_t} + \|\boldsymbol{e}_{g,com}\|^2_{\boldsymbol{W}_{g,com}}$$

$$\text{Constraints: } (\boldsymbol{E}_{g,fov}, \boldsymbol{g}_{g,fov}), (\boldsymbol{E}_{lim}, \boldsymbol{g}_{lim})$$

The state machine switches to S_A when the regulation errors are below a certain threshold.

$$t_{S_S \to S_A} \ : \ \text{if } \|\boldsymbol{e}_t\| \leq e_{thr} \wedge \|\boldsymbol{e}_{g,com}\| \leq e_{thr}$$

B. S_A: Approach State

After the setup, teleoperation is enabled and the user can start moving the robot arm. The inputs in the master device frame $\mathcal{F}_m$ are mapped into references for the TCP expressed in the camera frame $\mathcal{F}_c$ for an intuitive usage of the platform.

In this state, the objective is to have the camera zoom in on the scene by getting closer as the TCP approaches the goal, thus improving the image resolution for the user. A solution is to keep regulating these two points at the respective references used in S_S. Furthermore, we prevent any camera rotation around the $\mathcal{F}_c$ z-axis to avoid unexpected camera behavior, by regulating the angle ϕ to its starting value.

Thus, if the user does not move the master device, the whole system remains still. Instead, when the user interacts with the master, the teleoperated robot executes the command while the camera moves mostly along its z-axis. Note that in the image the TCP projection never moves.

With respect to the previous controller, we add the camera angle ϕ regulation and the TCP pose $\boldsymbol{x}_{t,tele}$ tracking of the master (4), as well as the tool FoV constraint and occlusion avoidance(17) $(\boldsymbol{E}_{occ}, \boldsymbol{g}_{occ})$

$$\begin{aligned}&\text{Cost fun.: } \|\boldsymbol{e}_t\|^2_{\boldsymbol{W}_t} + \|\boldsymbol{e}_{g,com}\|^2_{\boldsymbol{W}_{g,com}} \\ &\quad + \|\boldsymbol{e}_{t,tele}\|^2_{\boldsymbol{W}_{t,tele}} + \|e_\phi\|^2_{w_\phi} \\ &\text{Constraints: } (\boldsymbol{E}_{fov}, \boldsymbol{g}_{fov}), (\boldsymbol{E}_{lim}, \boldsymbol{g}_{lim}), (\boldsymbol{E}_{occ}, \boldsymbol{g}_{occ})\end{aligned}$$

The state machine switches to S_C when the regulation error is below a certain threshold and the goal area a_g is large enough, so that no further zoom in is necessary anymore.

$$t_{S_A \to S_C} \ : \ \text{if } \|\boldsymbol{e}_t\| \leq e_{thr} \wedge a_g \geq a_{g,thr}$$

C. S_C: Conclusion State

Once the visible goal area is large enough, the camera should stop its approaching motion. Thus, we unlock the tool feature from its reference and allow it to move in the image plane while keeping the FoV constraints. As for the goal, we keep its CoM reference and also regulate its area. This allows the user to perform the final reaching motion, with a clearly visible goal, and, in absence of FoV and occlusion constraint activation, a stationary view of the scene.

The controller at this stage is then the following

$$\begin{aligned}&\text{Cost fun.: } \|\boldsymbol{e}_{g,com}\|^2_{\boldsymbol{W}_{g,com}} + \|\boldsymbol{e}_{t,tele}\|^2_{\boldsymbol{W}_{t,tele}} \\ &\quad + \|e_{g,area}\|^2_{w_{g,area}} + \|e_\phi\|^2_{w_\phi} \\ &\text{Constraints: } (\boldsymbol{E}_{fov}, \boldsymbol{g}_{fov}), (\boldsymbol{E}_{lim}, \boldsymbol{g}_{lim}), (\boldsymbol{E}_{occ}, \boldsymbol{g}_{occ})\end{aligned}$$

In case the user decides to abort the task by moving away from the goal and trying to leave the FoV, we go back to the setup state S_S and regulate again the tool point feature.

$$t_{S_C \to S_S} \ : \ \text{if FoV constraint is active}$$

D. S_O: Occlusion State

Upon activation of the occlusion constraints, the system should be properly controlled to ensure a smooth camera behavior, and put the system in a configuration that makes the reactivation of the constraint unlikely, to avoid frequent camera positioning that may confuse the user.

The FSM enters this state during the teleoperation, whenever an occlusion constraint holds with equality sign

$$t_{S_A \to S_O}, \ t_{S_C \to S_O} \ : \ \text{if occlusion constraint is active}$$

The first action that we can take is to minimize the object area a_o. By reducing the forbidden area, it will be less likely for the TCP to perform a trajectory that passes through the occluding object projection, thus reducing the risk of occlusion. However, since this feature is mostly influenced by the camera motion in its z-axis, we simultaneously limit the camera velocity $v^c_{c,z}$, to avoid losing resolution by moving away. In fact, while retreating along the camera optical axis is a motion compatible with the occlusion constraint, it does not help in dealing with it, as it simply delays its occurrence, since the point-object relative positioning in the image remains the same. Indeed, in this case we want to let the camera exploit the other d.o.f. (e.g., rotation around the object), to reach a more favorable configuration.

At the same time, we also try to move the object projection away from the FoV. To ensure this behavior, we compute its CoM $\boldsymbol{p}_{o,com}$, and set its reference to a region unoccupied by the robot TCP and the goal. In absence of obstacles to the camera motion, either of the remaining free quadrants is a possible choice. Instead, if the camera motion is restricted in one way (e.g., by a table, as in the experiments), the positioning is naturally forced to the remaining region. If the physical obstruction is more complex, the use of a planner based on the reconstructed environment becomes necessary. Clearly, the presented one is a greedy heuristic that solves the occlusion locally, disregarding the effect on objects outside the scene that might later induce a new occlusion.

Notice that by regulating at the same time $\boldsymbol{p}_t$, $\boldsymbol{p}_{g,com}$, and $\boldsymbol{p}_{o,com}$, we cannot further influence the camera motion to satisfy other requirements. In this case we should accept some error in the regulation of the TCP and goal CoM features to tackle the occlusion, and simply ask to minimize their velocities $\dot{\boldsymbol{p}}_t$, $\dot{\boldsymbol{p}}_{g,com}$.

Like in S_A and S_C, we still allow the teleoperation by tracking the master, however we remove the regulation of angle ϕ to allow more freedom to the camera. The optimization problem is thus the following one

$$\begin{aligned}&\text{Cost fun.: } \|\dot{\boldsymbol{p}}_t\|^2_{\boldsymbol{W}_t} + \|\dot{\boldsymbol{p}}_{g,com}\|^2_{\boldsymbol{W}_{g,com}} + \|\boldsymbol{e}_{t,tele}\|^2_{\boldsymbol{W}_{t,tele}} \\ &\quad + \|\boldsymbol{e}_{o,com}\|^2_{\boldsymbol{W}_{o,com}} + \|a_o\|^2_{w_{o,area}} + \|v^c_{c,z}\|^2_{w_{c,z}} \\ &\text{Constraints: } (\boldsymbol{E}_{fov}, \boldsymbol{g}_{fov}), (\boldsymbol{E}_{lim}, \boldsymbol{g}_{lim}), (\boldsymbol{E}_{occ}, \boldsymbol{g}_{occ})\end{aligned}$$

Notice that since in this case we are regulating multiple quantities, it is necessary to accurately choose the weights $\boldsymbol{W}_i$ to achieve the desired behavior. We provide a possible tuning for our system in Section V.

To determine when the camera has dealt with the occlusion, as a heuristic we check the existence of a straight path in the image plane from $\boldsymbol{p}_t$ to a point on the goal $\boldsymbol{p}_g$ and that the occluding area is small enough. From this state the system can switch either to S_A or S_C, depending on the goal area.

$$t_{S_O \to S_A} : \text{if } \exists\, straight\ path\ from\ \boldsymbol{p}_t\ to\ \boldsymbol{p}_g \wedge a_o < a_{o,thr} \wedge \\ \wedge\ occlusion\ constraint\ is\ \underline{not}\ active\ \wedge a_g < a_{g,thr}$$

$$t_{S_O \to S_C} : \text{if } \exists\, straight\ path\ from\ \boldsymbol{p}_t\ to\ \boldsymbol{p}_g \wedge a_o < a_{o,thr} \wedge \\ \wedge\ occlusion\ constraint\ is\ \underline{not}\ active\ \wedge a_g \geq a_{g,thr}$$

V. Experimental Validation

We validated the presented approach on our experimental platform consisting of an ABB YuMi dual-arm robot with 7 d.o.f. for each arm as slave robot, and a Novint Falcon 3 d.o.f. master device. The right arm of YuMi is equipped with a $640px \times 480px$ resolution Microsoft LifeCam RGB webcam. All the image processing, as well as the camera calibration, is done with the OpenCV libraries, while for the optimization problem we employ the qpOASES solver. Due to the limited number of degrees of freedom available on the master, we teleoperated only the position of the slave robot by mapping the master position to the slave TCP velocity expressed in the camera frame. The devices are connected to two real-time Linux PCs and communicate via TCP/IP. The controller runs at a frequency of 250 Hz, and produces the joint reference values for the ABB industrial low-level controller, while the image processing is performed at 20 Hz.

We placed two objects in the robot workspace, a small one acting as the reaching task goal, and a larger potentially occluding object. Since depth information is not provided, we assume to know the 3D model of the two objects to estimate their pose by solving the Perspective-n-Point problem from camera measurements and thus compute the interaction matrices. In the image plane, we set the TCP and the goal CoM reference respectively at a distance of $[80, 60]px$ and $[400, 300]px$ from the FoV limits, depending on the starting configuration of the system, while for the goal reference area in S_C we set $a_g^* = 16500px^2$. In the occlusion state S_O, we used the following task relative weightings and normalizations: $\boldsymbol{W}_t = 1 \cdot 40^{-2}\boldsymbol{I}px^{-2}s^2$, $\boldsymbol{W}_{g,com} = 1 \cdot 40^{-2}\boldsymbol{I}px^{-2}s^2$, $\boldsymbol{W}_{t,tele} = 1 \cdot 0.05\boldsymbol{I}m^{-2}s^2$, $\boldsymbol{W}_{o,com} = 0.8 \cdot diag(640^{-2}, 480^{-2})px^{-2}$, $w_{o,area} = 1 \cdot 62500^{-2}px^{-4}$, $w_{c,z} = 0.1 \cdot 0.05^{-2}m^{-2}s^2$. We set $e_{thr} = 5px$ as a regulation tolerance. $a_{g,thr} = 14025px^2$ was chosen smaller than the reference to prevent the camera from pulling back after the transition to S_C, and $a_{o,thr} = 62500px^2$.

Fig. 4 shows the camera FoV with feature trails and the robot workspace during each phase of one of the experiments.[1] In Fig. 4(a) the image features are regulated to the respective positions (S_S) and the user is ready to start the teleoperation to reach the orange goal (S_A). Notice in Fig. 4(b) how the TCP is moving while its projection is regulated in the FoV. The occlusion constraint activates due to the apparent motion of the object trying to occlude the tool, the system then switches to S_O. Fig. 4(c) shows the trail of the four vertices of the superior face of the object. The camera combines a motion around the occluding segment in the workspace and one upward to avoid the occlusion and minimize the object area while moving it towards the edge of the FoV. The designed behavior brings the camera in a more favorable position to complete the task and return to state S_A. Finally, when the goal resolution is high enough, the final movement begins with the switch to S_C (Fig. 4(d)).

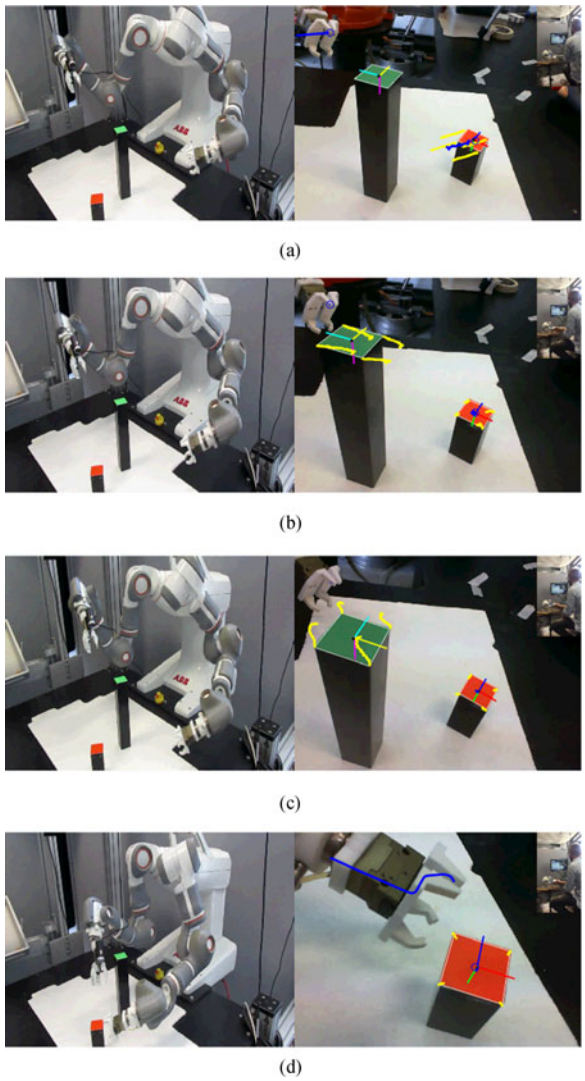

Fig. 4. (a) Position at the end of S_S and at the start of S_A. The blue trails show the TCP and the goal CoM being brought to reference. The TCP starts from outside the FoV. (b) The user tries to go behind an object while in S_A. In the FoV, the object apparent motion towards the TCP activates the constraint and the transition to S_O, while the goal CoM is regulated to reference. (c) During the occlusion the user can still teleoperate the robot, however the controller moves the camera to reduce the risk of occlusion and return to S_A, by minimizing the object area and moving its CoM out of the FoV. The tool and goal CoM displacement from their reference is minimal. (d) When the TCP is close enough to the goal, the system enters S_C. Then the user can freely move the robot in the FoV while the camera provides a high resolution and constant view of the scene.

Master and slave velocities are plotted in Fig. 5. We achieve kinematic coordination even during the occlusion phase, although some small mismatches are expected due to the

[1] Video available at youtu.be/FUMXeglkqdo and in the attached media.

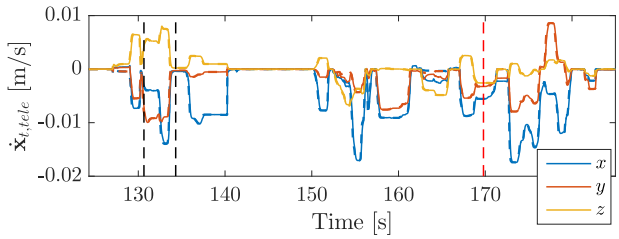

Fig. 5. Master (dashed) and slave (solid) velocities expressed in $\mathcal{F}_c$. The black dashes mark the start and the end of S_O, while the red ones the switch from S_A to S_C.

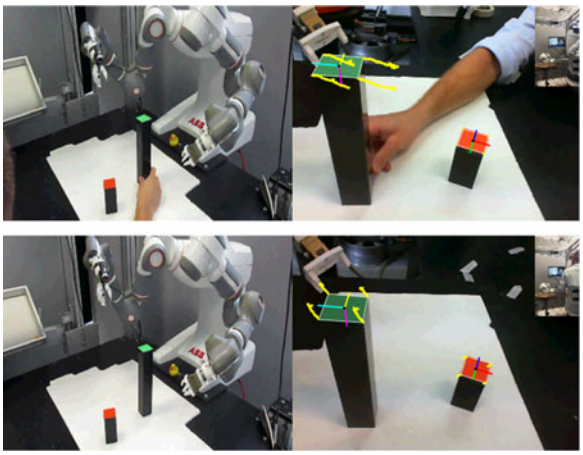

Fig. 6. The object is moved during S_A to induce an occlusion of the TCP. The controller is able to robustly deal with object dynamics not explicitly taken into account in the constraint derivation by moving the camera to bring the object away from the tool. The reaching task can then resume as in the nominal conditions.

weighing used in the optimization, compromising among the tasks when no more d.o.f. are available. The user experience with the system is smooth, since no teleoperation interruption due to the camera control is perceived.

As mentioned in Section III-B, our formulation can implicitly deal with object motions not pre-compensated in (7). Thus, in another experiment we tested the robustness of the approach to unmodeled dynamics, moving the object by hand towards the TCP feature point, causing an artificial occlusion. Fig. 6 shows such experiment demonstrating the approach effectiveness even for perturbed and unexpected conditions.

VI. Conclusion and Future Work

We proposed a unified real-time optimization controller for a visual-servoed dual-arm teleoperation system that allows robust occlusion avoidance in cluttered environments. The designed constraints and FSM ensure a natural interaction between the user and the system, and occlusion-free task execution. The experimental validation also showed the robustness of the approach against measurement noise and unmodeled object dynamics.

In the experiments we assumed to know the objects 3D model for the computation of the respective interaction matrices in the occlusion constraint, however we argue that an accurate model is not necessary. Indeed, even in presence of unknown environment, one can generate a convex volume containing the object (e.g., by fitting a superquadric on a point cloud, as done in [18]), and then discretize it by selecting the points on its surface that will be used in the proposed occlusion constraint, once projected in the image plane. Furthermore, to avoid constraint activation when the point of interest is closer to the camera than the potentially occluding object, a sensor providing depth information is necessary, if no a priori model is available. Finally, we validated our approach with only one occluding object. Multiple objects will require further examination and efficient solvers due to the growing computational time required by the optimization.

References

[1] P. Chatelain, A. Krupa, and N. Navab, "Optimization of ultrasound image quality via visual servoing," in *Proc. 2015 IEEE Int. Conf. Robot. Autom.*, 2015, pp. 5997–6002.

[2] F. Abi-Farraj, N. Pedemonte, and P. R. Giordano, "A visual-based shared control architecture for remote telemanipulation," in *Proc. 2016 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 4266–4273.

[3] A. De Luca, G. Oriolo, and P. Robuffo Giordano, "Feature depth observation for image-based visual servoing: Theory and experiments," *Int. J. Robot. Res.*, vol. 27, no. 10, pp. 1093–1116, 2008.

[4] R. Fleurmond and V. Cadenat, "Handling visual features losses during a coordinated vision-based task with a dual-arm robotic system," in *Proc. IEEE 2016 Eur. Control Conf.*, 2016, pp. 684–689.

[5] M. Kazemi, K. Gupta, and M. Mehrandezh, "Path-planning for visual servoing: A review and issues," *Visual Servoing via Advanced Numerical Methods*. Berlin, Germany: Springer, 2010, pp. 189–207.

[6] Y. Mezouar and F. Chaumette, "Avoiding self-occlusions and preserving visibility by path planning in the image," *Robot. Auton. Syst.*, vol. 41, no. 2, pp. 77–87, 2002.

[7] O. Kermorgant and F. Chaumette, "Dealing with constraints in sensor-based robot control," *IEEE Trans. Robot.*, vol. 30, no. 1, pp. 244–257, Feb. 2014.

[8] M. Kazemi, K. K. Gupta, and M. Mehrandezh, "Randomized kinodynamic planning for robust visual servoing," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1197–1211, Oct. 2013.

[9] G. Chesi and Y. S. Hung, "Global path-planning for constrained and optimal visual servoing," *IEEE Trans. Robot.*, vol. 23, no. 5, pp. 1050–1060, Oct. 2007.

[10] A. Remazeilles, N. Mansard, and F. Chaumette, "A qualitative visual servoing to ensure the visibility constraint," in *Proc. 2006 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 4297–4303.

[11] D. Folio and V. Cadenat, "A controller to avoid both occlusions and obstacles during a vision-based navigation task in a cluttered environment," in *Proc. 44th IEEE Conf. Decis. Control, 2005 Eur. Control Conf.*, 2005, pp. 3898–3903.

[12] O. Kanoun, F. Lamiraux, and P.-B. Wieber, "Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 785–792, Aug. 2011.

[13] D. Nicolis, A. M. Zanchettin, and P. Rocco, "A hierarchical optimization approach to robot teleoperation and virtual fixtures rendering," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 5672–5679, 2017.

[14] D. J. Agravante, G. Claudio, F. Spindler, and F. Chaumette, "Visual servoing in an optimization framework for the whole-body control of humanoid robots," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 608–615, Apr. 2017.

[15] A. M. Zanchettin and P. Rocco, "Path-consistent safety in mixed human-robot collaborative manufacturing environments," in *Proc. 2013 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 1131–1136.

[16] A. M. Zanchettin and P. Rocco, "Robust constraint-based control of robot manipulators: An application to a visual aided grasping task," in *Proc. 2016 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 3634–3639.

[17] F. Keith, P.-B. Wieber, N. Mansard, and A. Kheddar, "Analysis of the discontinuities in prioritized tasks-space control under discreet task scheduling operations," in *Proc. 2011 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 3887–3892.

[18] G. Vezzani, U. Pattacini, and L. Natale, "A grasping approach based on superquadric models," in *Proc. 2017 IEEE Int. Conf. Robot. Autom.*, 2017, pp. 1579–1586.