Vision-based framework to estimate robot configuration and kinematic constraints

Vision-based framework to estimate robot configuration and kinematic constraints

Valerio Ortenzi[1,*], Naresh Marturi[1,2], *Member, IEEE,* Michael Mistry[3], *Member, IEEE,* Jeffrey Kuo[4] and Rustam Stolkin[1,4], *Member, IEEE*

***Abstract*—This paper addresses the problem of estimating the configuration of robots with no proprioceptive sensors and with kinematic constraints while performing tasks. Our work is motivated by the use of unsensored (industrial) manipulators, currently tele-operated in rudimentary ways, in hazardous environments such as nuclear decommissioning. For such robots, basic proprioceptive sensors are often unavailable. Even if radiation-hardened sensors could be retrofitted, such manipulators are typically deployed on a mobile base, while equipped with powerful end-effector tools for forceful contact tasks, which significantly perturb the robot base. This work contributes a step towards enabling advanced control and increased autonomy in nuclear applications, but could also be applied to mechanically compliant, under-actuated arms and hands, and soft manipulators. Our proposed framework: estimates the robot configuration by casting it as an optimisation problem using visually tracked information; detects contacts during task execution; triggers an exploration task for detected kinematic constraints, which are then modelled by comparing observed versus commanded velocity vectors. Unlike previous literature, no additional sensors are required. We demonstrate our method on a Kuka iiwa 14 R820, reliably estimating and controlling robot motions and checking our estimates against ground truth values, and accurately reconstructing kinematic constraints.**

***Index Terms*—Robots, Robot kinematics, Robot vision systems**

I. Introduction

In robotics research laboratories, accurate proprioception (in the form of rotation encoders at the robot's joints) is usually taken for granted. In contrast, industrial manipulators (*e.g.*, Brokk type arms) that are deployed to perform heavy-duty tasks such as demolition, heavy loading, drilling and grinding of concrete etc. typically lack any proprioceptive sensors. A societally important use of such manipulators is in nuclear decommissioning, safely disposing of nuclear waste to remediate the environment for future generations [1]. The UK alone contains 4.9million tonnes of nuclear waste, which will take more than 100 years and order $200billion to remediate. Worldwide decommissioning costs are of order $trillion. Many contaminated zones cannot be entered by humans, however, a human operator must directly teleoperate the machine, by pushing a separate switch or lever to control each joint, while guessing the inverse kinematics from experience. An inability to automate such operations not only means that task performances are sub-optimal, but also that humans are being exposed to high risks in hazardous environments.Also, it is not considered feasible to retrofit proprioceptive sensors because electronics are vulnerable to different types of radiation; and because the installation of new sensors on trusted machinery would compromise long-standing certification. Moreover, during the execution of tasks, robots must interact with contact surfaces, which are typically unknown a-priori. Understanding which directions of motion are not feasible is important: (i) for not causing damage in a high-consequence environment; and (ii) for enabling planning and execution of trajectories which minimise torques on the robot's joints [2], [3]. Minimising joint torques is critically important in nuclear facilities, where a robot cannot be removed from its working environment for repair, without extremely expensive, difficult and time-consuming decontamination (sometimes impossible).

Our thesis is that adopting external vision offers an effective means to solve these problems. Cameras can be radiation-hardened, and the distance of a remote sensor, away from the radiation source, greatly reduces impact on electronics via inverse-square law. Vision-based proprioceptive feedback can enable advanced trajectory control and increased autonomy. This can help remove humans from harm, improve operational safety, improve task performance, and reduce maintenance costs [4]. We also believe that a much larger community can benefit from this work. This framework can be applied to: underactuated fingers, hands and arms [5]–[7]; inherently backdrivable robots, such as Sybot [8]; and "soft" manipulators, *e.g.*, in [9] or in pneumatic snake-arm type devices such as in [10] or as in [11].

This paper extends our recent work on vision-based proprioception and control [12]; and detection and estimation of kinematic constraints [13]. More specifically, in this paper, we show how to combine these two capabilities in a new overarching control framework to address the more general problem of visually controlling an undersensored robot arm, while simultaneously using visual proprioception to detect, explore and model kinematic constraints arising from obstructions encountered by the robot. The main contributions of this work are summarised as follows:

This work is supported by: EU H2020 RoMaNS, 645582; EU H2020 CogIMon, 644727; EPSRC grant EP/M026477/1, and Innovate UK KTP partnership 9573. It also forms part of the UK National Centre for Nuclear Robotics initiative, funded by EPSRC EP/R02572X/1. Ortenzi was supported by a UK Nuclear Decommissioning Authority (NDA) bursary; Stolkin is supported by a Royal Society Industry Fellowship.

[1]V. Ortenzi, N. Marturi and R. Stolkin are with Extreme Robotics Lab, University of Birmingham, B15 2TT, UK (email: valerio.ortenzi@gmail.com; R.Stolkin@bham.ac.uk). * Corresponding author

[2]N. Marturi is also with KUKA Robotics UK Ltd., Great Western Street, Wednesbury WS10 7LL, UK. (email: nareshmarturi@kuka-robotics.co.uk;)

[3]M. Mistry is with University of Edinburgh, School of Informatics, 10 Chichton Street, UK (email: mmistry@inf.ed.ac.uk).

[4]J. Kuo and R. Stolkin are with National Nuclear Laboratory Ltd., Chadwick House, Warrington, WA3 6AE, UK.

(i) The joint configuration of the robot is estimated by solving an optimisation problem, using vision information along with the robot kinematic model (Sec. III). These vision estimates are then fed to a controller to perform positioning tasks.
(ii) Contacts with the environment are detected analising the desired and the estimated configurations of the robot (Sec. IV). A significant difference is an indicator of a possible contact.
(iii) Finally, we propose a method for estimating the kinematic constraints arising from contacts with the environment (Sec. V). This method also relies only on the vision-based estimates of the robot's configuration and motions.

In the remainder of this paper, in Sec. VI, we report the experiments targeted at showing the behaviour of each module of the proposed framework. For demonstration purposes, we have used the Kuka iiwa 7 degrees of freedom (DoF) collaborative manipulator in our laboratory facilities. Though the used robot is equipped with positional encoders, we have used this information *only as ground truth* in evaluating our framework's performance quantitatively. Finally Sec. VII concludes the paper with directions for future work.

II. Related Work

Our framework solely relies on vision information to estimate the configuration of an undersensored manipulator. Neither visual tracking nor visual servoing form the primary focus of our contribution. Another important difference between our work and most classical visual servoing approaches [14]–[17], is that the latter tend to use joint encoders readings when mapping camera velocities to joint velocities.

This work is more related to [18], but overcomes the redundancy problem by simultaneously tracking multiple parts of the robot, consequently having more relationships constraining the configuration, making our method applicable to high DoF robots. Our approach is marginally related also to [19], however, the major differences are that we simultaneously track parts of an articulated object and separately define an optimisation problem to estimate the joint values using the tracked poses. A variety of ways to track articulated bodies in monocular images can be found in [19]–[22]. In contrast to our work, these authors mainly focused on localising parts of the articulated bodies in each image frame, and not on the estimation of joint angles among the connected parts. Additionally, most of these works focused on tracking robot parts, but made use of information from robot's joint encoders to do so, in contrast to the problem posed in this paper.

A real-time system to track multiple articulated objects using RGB-D and joint encoder information is presented by [23]. A similar approach was used by [24] to track and estimate the pose of a robot manipulator. Recently, Pauwels and Kragic [25] have proposed SimTrack, a framework for real-time robot tracking using RGB-D images from a Kinect along with the recorded joint angles. A marker-tracking method to identify the joint origins of robots was used in [26]. Other notable examples can be found in [27] and [28], where the authors propose to use depth information for better tracking of objects. Recently, an approach based on regression forests has been proposed to directly estimate joint angles using single depth images in [29]. Ma et al. proposed an adaptive penalty-based method using discrete structures to track human body poses in video frames [30], [31]. In summary, the use of depth information alongside standard RGB images can improve the tracking performances. However, it also increases the computational burden and decreases robustness in many real-world applications. Our choice of using simple commercial cameras is motivated by cost, robustness to real-world conditions, and also in an attempt to be as computationally fast as possible.

Recently in [13], we presented a method for reliably estimating the kinematic constraints arising from contacts that limit the free motion space of the robot. There we relied on the kinematic observations derived directly from basic proprioception (rotation encoders at joints). However, in this work we will use vision-based configuration estimates to accomplish the same. Therefore, in contrast to much of the related literature, no added sensors are needed such as force-torque sensors or tactile sensors at the contact points, and we will be using feedback from a single camera to estimate the robot configuration, so neither to detect contacts nor to reconstruct surface normals. In [32], a quadrupedal robot estimates the inclination of a planar surface on which it is trotting, by fusing data from inertial measurement unit (IMU) accelerometers with optical force sensors at each foot and the kinematics of each leg. In the context of hybrid motion/force control [33], a method for estimating the local shape of a constraint surface was proposed in [34], by combining position and end-effector force measurements. Another method for estimating constraints is proposed in [35]; however, this method learns the null space projector of an unknown task constraint from human demonstration. In contrast to the above approaches, our method locally estimates the kinematic constraints due to contacts, without using any additional sensors (force, torque, tactile), only relying on vision-based configuration estimates.

III. Vision-guided State Estimation

Our framework, shown in Fig. 1, is composed of three main components: visual tracking; state estimation; and a controller. We concentrate on state estimation and subroutines of the control component (contact detection and kinematic constraint estimation), and choose available methods for other components.

A. Visual Tracking

We use a very classical approach exploiting fiducial markers. For this purpose, we custom designed a marker that consists of four dots on a plain background, out of which one dot is made deliberately out-of-plane to resolve pose ambiguity. Four such markers are placed on different links of the robot arm as shown in Fig. 2(a). The 3D world coordinates of each dot's centroid are measured in the marker's reference frame *i.e.,* its center point and their corresponding 2D image coordinates are obtained using Freeman chain coding-based blob tracker available in ViSP library [36]. Consequently using

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMECH.2018.2865758, IEEE/ASME Transactions on Mechatronics

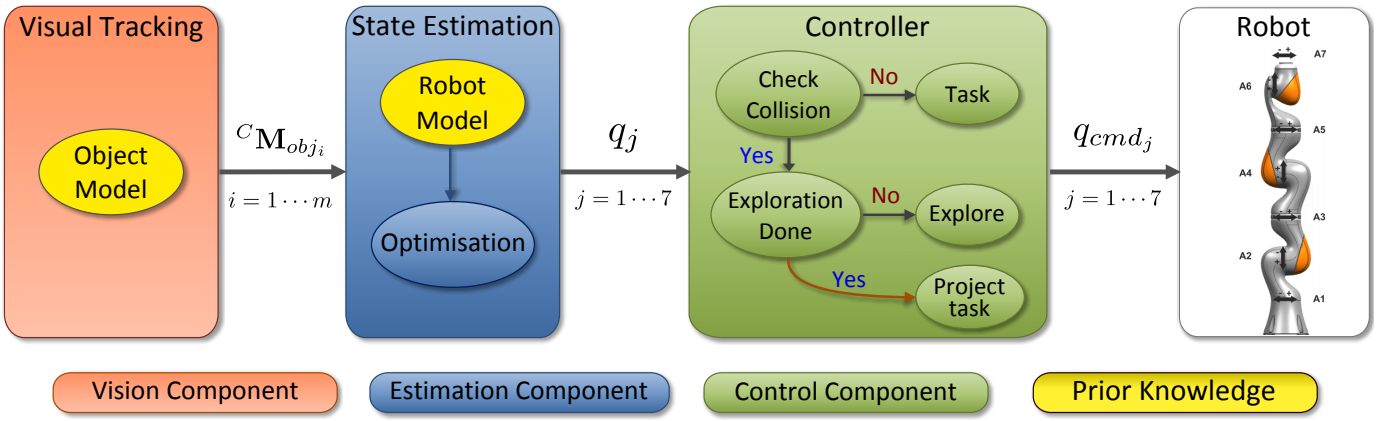


Fig. 1. Illustration of the proposed framework pipeline. The overall schema is divided into 3 main components: vision (orange), estimation (blue) and control (green). The vision component is responsible for tracking robot poses in acquired images. The estimation component uses these poses along with the robot model to solve an optimisation problem to estimate the robot's current joint configuration. These estimates are used by the controller to servo the robot and also to detect possible contacts. When a contact is detected, an exploratory phase starts and the kinematic constraints are estimated. The controller generates the motion control commands to the robot. Here, we assume that the object models to track and the kinematic model of the robot are known beforehand.

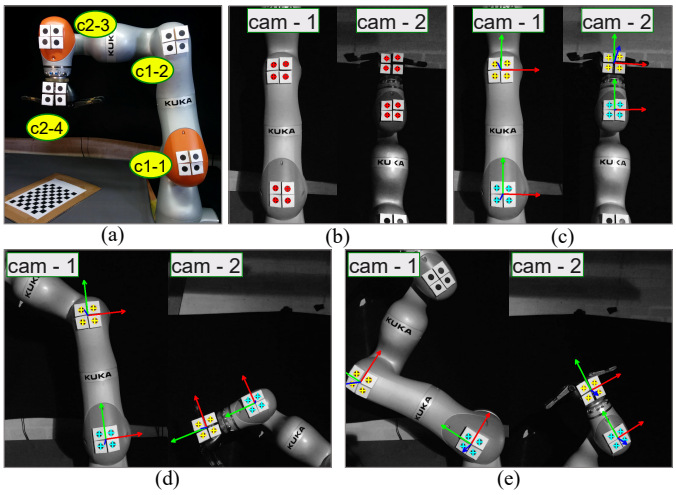


Fig. 2. Illustration of visual tracking and object's pose estimation. (a) Four custom designed markers are placed on different links of the robot. Each marker has four dots (one out-of-plane) whose position with respect to the marker reference frame is known a priori. In the image, markers are named $1 \cdots 4$ and the cameras by which they are tracked are indexed as c1 and c2. (b) Automatically initialised dots for visual tracking in the initial frame. (c)-(e) Tracked poses of each object in later frames while the robot is in motion.

the camera calibration, the corresponding poses of the markers ${}^{c}\mathbf{M}_{o(i)}$ ($i = 1 \cdots 4$ are marker indexes) are computed. The updated poses in each iteration are propagated to the next frame. A more detailed description can be found in [37].

We assume that the robot is always in a pre-defined home position before task initialisation such that the world locations of each marker are known in the first frame of tracking *i.e.*, at $t = 0$. This assumption is not restrictive since a predefined home position for the robot will not add any new constraints. We have used two cameras (each one for two objects) to accomplish tracking in this work. Our choice to use two commercial cameras stems from the fact that it proved difficult to track four markers simultaneously using only one camera, mostly because for the markers to be all visible, the only camera would have to be placed farther from the robot, thus compromising the performances. We could consider using more cameras and markers. A redundancy of sensors and markers might alleviate the problem of possible occlusions, however it would increase the complexity and the cost of the system. Fig. 2b shows automatically initialised tracking in the first frame and Figs. 2c-2e show tracked dots along with individual marker's poses.

B. Robot Configuration Estimation

We use the following key idea to estimate the configuration of the robot. We note that two kinematic paths exist from the camera to each of the markers, *i.e.*, the direct relationship given by the trackers and the kinematic path through the robot, parametrised over the configuration $\mathbf{q}$ of the robot. Thus we can write

$$
\begin{aligned}
{}^{C}\mathbf{M}_{obj_1} &= {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1)\,{}^{1}\mathbf{T}_2(q_2)\,{}^{2}\mathbf{T}_{obj_1} && (1)\\
{}^{C}\mathbf{M}_{obj_2} &= {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1)\,{}^{1}\mathbf{T}_2(q_2)\,{}^{2}\mathbf{T}_3(q_3)\,{}^{3}\mathbf{T}_{obj_2} && (2)\\
{}^{C}\mathbf{M}_{obj_3} &= {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1)\,{}^{1}\mathbf{T}_2(q_2)^2 \\
&\quad \mathbf{T}_3(q_3)\,{}^{3}\mathbf{T}_4(q_4)\,{}^{4}\mathbf{T}_5(q_5)\ {}^{5}\mathbf{T}_6(q_6)\ {}^{6}\mathbf{T}_{obj_3} && (3)\\
{}^{C}\mathbf{M}_{obj_4} &= {}^{C}\mathbf{T}_0\ {}^{0}\mathbf{T}_1(q_1)\ {}^{1}\mathbf{T}_2(q_2)\ {}^{2}\mathbf{T}_3(q_3) \\
&\quad {}^{3}\mathbf{T}_4(q_4)\ {}^{4}\mathbf{T}_5(q_5)\ {}^{5}\mathbf{T}_6(q_6)\ {}^{6}\mathbf{T}_7(q_7)\ {}^{7}\mathbf{T}_{obj_4} && (4)
\end{aligned}
$$

where, ${}^{C}\mathbf{M}_{obj_i}$ is the pose relating marker i to the camera, ${}^{C}\mathbf{T}_0$ is the transformation from camera to world frame and ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$ represents the transformation from world to marker i frame, parametrised over the joint values $\mathbf{q}$, *i.e.*, ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$ embeds the kinematic model of the robot. The state of the robot is now estimated by imposing these equalities, casting an optimisation problem. As already explained, we assume to know the initial configuration of the robot and its kinematic model. The robot's initial configuration is used as a seed for the first iteration of the optimisation problem and the kinematic model is used to compute ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$. The optimisation problem is then stated as

$$
\underset{\mathbf{q}}{\text{minimize}} \sum_i e_i(\mathbf{q}) \quad \text{subject to} \quad |q_j| \leq q_{max_j} \qquad (5)
$$

where,

$$e_i(\mathbf{q}) = v({}^{C}\mathbf{M}_{obj_i} - {}^{C}\mathbf{T}_0 \; {}^{0}\mathbf{T}_{obj_i}(\mathbf{q})) \tag{6}$$

represents an error in the difference of the two kinematic paths, and q_{max_j} is the joint limit for joint j. In order to compute ${}^{0}\mathbf{T}_{obj_i}(\mathbf{q})$, we use the convention of Denavit-Hartenberg [38]. The v operator stands for the sum of the square of each element of the matrix between parentheses.

Because of the better performances achieved in [12], we decided to use the chained method. This method uses each marker to estimate only a subset of joint values. These, in turn, are used as known parameters in the successive estimation problems.

C. Controller

For proof of concept, we implemented a classical joint controller of the form

$$\dot{\mathbf{q}}_r = \mathbf{K}_P \tilde{\mathbf{e}} + \dot{\mathbf{q}}_d \,, \tag{7}$$

to validate our methodology and also to demonstrate how the vision-derived state estimates can be used to servo the robot's end-effector. Here, $\dot{\mathbf{q}}_r$ is the desired/reference velocity. The error $\tilde{\mathbf{e}} = \mathbf{q}_d - \tilde{\mathbf{q}}$ has been defined as the difference between desired and estimated joint positions, $\tilde{\mathbf{q}}$. Finally, $\mathbf{K}_P$ is the proportional gain matrix. We send position commands to the robot, hence (7) is integrated numerically as

$$\mathbf{q}_{cmd} = \tilde{\mathbf{q}} + \Delta t \; \dot{\mathbf{q}}_r \,, \tag{8}$$

where $\mathbf{q}_{cmd}$ are the commands sent to the robot, $\tilde{\mathbf{q}}$ is the estimated robot configuration, Δt is the integration time and $\dot{\mathbf{q}}_r$ is as defined in (7). Note that the availability of only estimates of the configuration - imperfect feedback - jeopardises the convergence of the error $\tilde{\mathbf{e}}$ to zero, which is not guaranteed anymore.

IV. Contact Detection

We implemented two methods to check contacts with the environment. Such checks use the configuration estimates computed as in the previous section. The first method is based on the concept of residuals, while the second method is based on the difference between desired configuration and current configuration. We rely on estimates of the configuration of the robot, which are possibly imprecise and noisy.

A. Residuals

The idea behind the method of the residuals is to model contacts as faults of the system. This method was first proposed in [39], however their definition of residuals assumes knowledge of the dynamic model and assumes that the robot is controlled in torque. Differently, we use only kinematics in our definition. Hence we equivalently define

$$\dot{\mathbf{q}} = \dot{\mathbf{q}}_r - \dot{\mathbf{q}}_C \,. \tag{9}$$

$\dot{\mathbf{q}}_r$ is the commanded velocity, while $\dot{\mathbf{q}}_C$ can be interpreted as a virtual velocity induced by contacts. We define the residuals $\mathbf{r}$ as

$$\mathbf{r} = \mathbf{K}[\int_0^t (\dot{\mathbf{q}}_r - \mathbf{r}) \; dt - \mathbf{q}] \,, \tag{10}$$

with $\mathbf{K} > 0$ and $\mathbf{r} = \mathbf{0}$ at $t = 0$. Differentiating we obtain

$$\dot{\mathbf{r}} = -\mathbf{K}\mathbf{r} + \mathbf{K}\dot{\mathbf{q}}_C \,. \tag{11}$$

Analysing the dynamics of $\mathbf{r}$, $\mathbf{r}$ converges to $\mathbf{0}$ when $\dot{\mathbf{q}}_C = \mathbf{0}$, while this is not true when $\dot{\mathbf{q}}_C \neq \mathbf{0}$.

1) Kinematic Residuals using Estimates: Up until now, the formulations assumed perfect knowledge of the state of the robot, *i.e.*, configurations and velocities. In our scenario, only estimates of the configurations are available. Because of the error in the estimation process, $\mathbf{q}$ is not known but an estimate of it, $\tilde{\mathbf{q}}$, is available. If we express the commanded configuration as $\mathbf{q}_{cmd} = \tilde{\mathbf{q}}_{t-1} + \Delta t \; \dot{\mathbf{q}}_r$, then we model the system as

$$\tilde{\mathbf{q}}_t = \mathbf{q}_{cmd} + \boldsymbol{\epsilon} \,, \tag{12}$$

where,

$$\boldsymbol{\epsilon} = \boldsymbol{\epsilon}_\Delta + \mathbf{q}_C \,, \tag{13}$$

i.e., the mismatch between real configuration and estimates is due to the error in the estimation process ($\boldsymbol{\epsilon}_\Delta$) and a possible contact ($\mathbf{q}_C$). If we substitute $\mathbf{q}_{cmd}$ into (12), we obtain

$$\dot{\tilde{\mathbf{q}}} = \dot{\mathbf{q}}_d + \dot{\tilde{\boldsymbol{\epsilon}}} \,, \tag{14}$$

where, $\dot{\mathbf{q}}_d$ is the desired velocity as in $\dot{\mathbf{q}}_r = \mathbf{K} \; (\mathbf{q}_d - \tilde{\mathbf{q}}) + \dot{\mathbf{q}}_d$ and $\tilde{\boldsymbol{\epsilon}} = \mathbf{K}\Delta t \; (\mathbf{q}_d - \tilde{\mathbf{q}}) + \boldsymbol{\epsilon}_\Delta + \mathbf{q}_C$. If we define the residuals as

$$\mathbf{r} = \mathbf{K} \left[\int_0^t (\dot{\mathbf{q}}_d - \mathbf{r}) \; dt - \tilde{\mathbf{q}} \right] \,, \tag{15}$$

then

$$\dot{\mathbf{r}} = -\mathbf{K}\mathbf{r} + \mathbf{K}\dot{\tilde{\boldsymbol{\epsilon}}} = -\mathbf{K}\mathbf{r} + \mathbf{K}(\boldsymbol{\alpha}((\mathbf{q}_d - \tilde{\mathbf{q}}), \tilde{\boldsymbol{\epsilon}}) + \dot{\mathbf{q}}_C) \,, \tag{16}$$

where, $\boldsymbol{\alpha}((\mathbf{q}_d - \tilde{\mathbf{q}}), \tilde{\boldsymbol{\epsilon}})$ is the derivative over time of $\mathbf{K}\Delta t \; (\mathbf{q}_d - \tilde{\mathbf{q}}) + \boldsymbol{\epsilon}_\Delta$. Hence, the residuals become different than zero when there is an error in the estimates, or when there is a contact, or when both these two situations happen. Thus it is not easy to discern when a contact occur, and the introduction of thresholds for detection is necessary and critical.

B. Sheer difference

The idea behind this method is that if the robot is in a different position with respect to the expected/desired position, then it is possible that a contact has occurred and that such contact has prevented the robot to go to the desired position. We propose to compute the norm of the difference between the current configuration and the configuration the robot should be in to perform the task. When this difference is greater than a threshold, then a contact is detected. In other words

$$|\tilde{\mathbf{q}} - \mathbf{q}_{des}| > \mathbf{q}_{thresh} \rightarrow \text{contact.} \tag{17}$$

This is a naive approach since a contact is not the only possible explanation for such difference, *e.g.*, the robot might have hit a joint limit. However, joint limits mean kinematic constraints, which are also necessary to know.

This method is very inexpensive computationally and represents a further check for collisions. Moreover, the use of configuration estimates affects negatively this check, since it introduces a difference between estimated configuration and desired configuration even when there is no contact. For this reason, the threshold becomes of key importance.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMECH.2018.2865758, IEEE/ASME Transactions on Mechatronics

V. Constraint Estimation

We proposed a method to estimate kinematic constraints in [13]. This work applies those methods to the case where only estimates of the configuration are available. We consider that the contact Jacobian $\mathbf{J}_C(\mathbf{q})$ can be described as

$$\mathbf{J}_C(\mathbf{q}) = \mathbf{\Lambda}\mathbf{J}(\mathbf{q})\,, \tag{18}$$

where $\mathbf{\Lambda}$ is a matrix that specifies which dimension(s) in the task-space is (are) constrained due to contacts and $\mathbf{J}(\mathbf{q})$ is the task Jacobian of the robot. $\mathbf{\Lambda}$ is independent of the dimensionality of the configuration space of the robot and independent of the current configuration of the robot. However, the number of independent rows depends on the number of independent constraints. $\mathbf{\Lambda}$ represents the Cartesian directions of motion which are unfeasible for motion due to the contact(s). It is also possible to derive equivalent equations in the joint space. We define the constraints as

$$\mathbf{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}\,. \tag{19}$$

In this case $\mathbf{\Lambda}^q(\mathbf{q})$ can be regarded also as a constraint Jacobian. $\mathbf{\Lambda}^q(\mathbf{q})$ is dependent on the configuration of the robot, in contrast to the definition of $\mathbf{\Lambda}$.

In real-world tasks involving constraints, it will be non-trivial to compute $\mathbf{\Lambda}$ and $\mathbf{\Lambda}^q(\mathbf{q})$ generally. However, we know that $\mathbf{\Lambda}\dot{\mathbf{y}} = \mathbf{0}$ and equivalently, $\dot{\mathbf{y}}^T\mathbf{\Lambda}^T = \mathbf{0}$. Thus, $\mathbf{\Lambda}^T$ is the solution to the homogeneous system

$$\dot{\mathbf{y}}^T\mathbf{\Lambda}^T = \mathbf{0}\,. \tag{20}$$

Let $\mathbf{B}_{\dot{\mathbf{y}}^T}$ be a set of observed $\dot{\mathbf{y}}^T$, *i.e.*, the end-effector velocities where the end-effector is in contact with the constrained surface. Then the solution set of the homogeneous system in (20) can be found by computing the right null space of $\mathbf{B}_{\dot{\mathbf{y}}^T}$, using singular value decomposition

$$\mathbf{B}_{\dot{\mathbf{y}}^T} = \mathbf{U}\mathbf{S}\mathbf{V}^T\,, \tag{21}$$

where, $\mathbf{U}$ is the matrix of left singular vectors, $\mathbf{S}$ is a diagonal matrix such that $\mathbf{S}_{i,i}$ is the i^{th} largest singular value, and $\mathbf{V}$ is a matrix of the right singular vectors. $\mathbf{\Lambda}^T$ can then be computed by taking the columns of $\mathbf{V}$ with corresponding singular values smaller than a threshold value ϵ.

To compose this $\mathbf{B}_{\dot{\mathbf{y}}^T}$, we propose to perform an exploration when the robot is in contact with the environment. The exploration is a set of predefined motions whose goal is to sample the space. In this implementation, we sample the space uniformly, as in [13]. Hence, there are two possibilities for the expected $\dot{\mathbf{y}}^{exp}$ and the observed $\dot{\mathbf{y}}^{obs}$:

(i) in the free motion subspace, *i.e.*, when the motion is not in the direction of any contact: $\dot{\mathbf{y}}^{exp} = \dot{\mathbf{y}}^{obs}$;
(ii) in the constrained motion subspace, *i.e.*, when the motion is in the direction of any contact: $\dot{\mathbf{y}}^{exp} \neq \dot{\mathbf{y}}^{obs}$.

We collect a set of $\dot{\mathbf{y}}^{obs}$ into $\mathbf{B}_{\dot{\mathbf{y}}^T}$ from the latter case and use the method based on (21) to estimate the selection matrix $\mathbf{\Lambda}$.

In the joint space $\dot{\mathbf{q}}^T\mathbf{\Lambda}^{qT}(\mathbf{q}) = \mathbf{0}$. Similarly to the Cartesian case, predefined exploratory motions can be defined directly in the joint space. Specifically:

(i) in the free motion subspace: $\dot{\mathbf{q}}^{exp} = \dot{\mathbf{q}}^{obs}$;
(ii) in the constrained motion subspace: $\dot{\mathbf{q}}^{exp} \neq \dot{\mathbf{q}}^{obs}$.

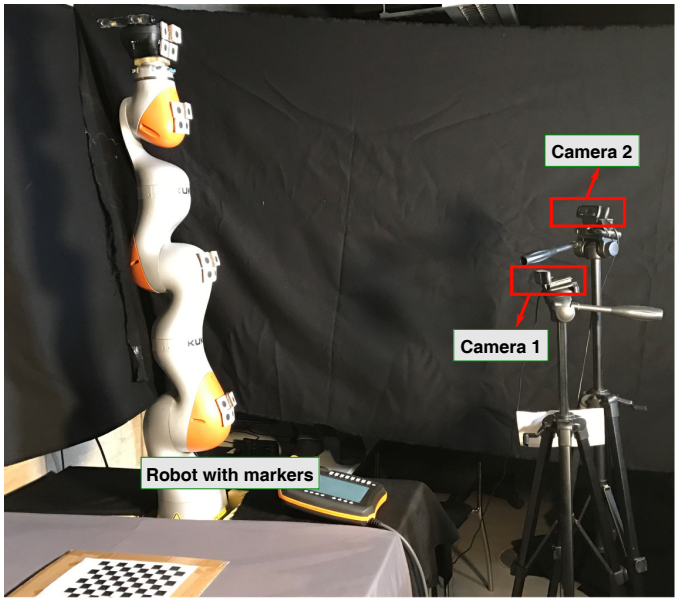


Fig. 3. Experimental set-up illustrating the used KUKA robot with attached markers on its various links and the cameras. The cameras are positioned in the workspace such that each camera can successfully track two markers.

We collect a set of $\dot{\mathbf{q}}^{obs}$ from the latter case, and use singular value decomposition to estimate the selection matrix $\mathbf{\Lambda}^q(\mathbf{q})$.

VI. Experimental Validation

The proposed framework has been evaluated using the experimental setup shown in Fig. 3. As mentioned before, we used the industrial collaborative robot, Kuka iiwa 14 R820 and our vision system consists of two commercial Logitech c920 cameras. The cameras are placed outside the robot task space and each camera can view a set of two fiducial markers. Each marker comprises 4 dots of 20 mm diameter each, glued to cardstocks. One of them has been deliberately made to locate out-of-plane to facilitate pose estimation. All the hardware components of our set-up are interfaced with a work computer (master) running Linux (Intel Core i7-4810MQ CPU with 15.3 GB of RAM) through their respective interfaces. The overall framework has been implemented in C++ on the master computer. We have used ViSP for all matrix computations and NLopt for optimisation. The asynchronous communication between the robot controller running Kuka Sunrise.OS and master has been realised over UDP/IP with a communication speed of ≈ 2 ms.

Overall, we conducted three sequences of experiments which are summarised as follows:

(a) we assess the precision of the proposed framework in estimating the robot configuration and we use the vision-derived state estimates as feedback in a control loop.
(b) Next, we detect contacts occurring when the robot is performing a task.
(c) Finally, we show how our method to estimate constraints due to contact works when using configuration estimates.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMECH.2018.2865758, IEEE/ASME Transactions on Mechatronics

TABLE I
RMSE OF ESTIMATES WITH RESPECT TO GROUND TRUTH VALUES DURING EXPERIMENTS WITH TRAJECTORIES INVOLVING ONLY A JOINT AT A TIME.

joint	RMSE (in degrees (°))						
	traj 1	traj 2	traj 3	traj 4	traj 5	traj 6	traj 7
q_1	2.289	2.687	0.293	0.416	0.212	0.213	0.288
q_2	2.743	0.468	0.076	0.063	0.096	0.071	0.122
q_3	1.129	1.269	2.204	0.813	0.693	0.426	0.496
q_4	4.534	4.091	4.419	1.199	10.648	0.355	0.345
q_5	2.782	5.409	3.029	1.957	3.215	0.636	0.409
q_6	3.400	3.523	5.061	0.950	10.170	0.562	0.228
q_7	1.947	2.148	1.719	2.046	2.511	1.785	5.791

TABLE II
RMSE OF ESTIMATES WITH RESPECT TO GROUND TRUTH VALUES DURING EXPERIMENTS WITH TRAJECTORIES INVOLVING MULTIPLE JOINTS.

Joint	RMSE (in degrees (°))						
	traj 8	traj 9	traj 10	traj 11	traj 12	traj 13	traj 14
q_1	3.633	2.139	0.579	0.180	2.610	1.868	2.803
q_2	2.590	1.170	1.601	0.129	0.568	1.734	1.542
q_3	2.789	1.450	0.794	0.466	1.137	0.867	1.619
q_4	5.278	5.263	4.812	3.878	3.409	5.638	4.180
q_5	2.268	3.209	2.261	1.164	5.208	5.188	6.062
q_6	2.753	4.564	3.802	4.127	3.212	4.347	3.968
q_7	2.415	2.823	7.734	1.622	3.025	3.346	4.135

A. *Vision-based Estimation*

1) Estimating joint configurations: this first set of experiments focus only on the estimation of the robot's configuration. The robot was asked to perform 14 different trajectories. The first 7 trajectories excite only one joint each, *i.e.*, trajectory 1 uses only joint 1, trajectory 2 only joint 2, and so on. The remaining 7 trajectories contain motions for multiple joints at a time. We report each joint root-mean-square error (RMSE) between real values and estimates for each trajectory and also the average RMSE and standard deviation over the RMSE of all the trajectories. Tables I and II show the RMSE values of each joint for each of the 14 trajectories. Table III provides the reader with mean and standard deviation of such RMSEs as computed over the results presented in Tables I and II. Fig. 4 shows results for trajectory 11, where the real trajectory of the joint is in blue, and the estimated trajectory is in green.

Analysis of results: Despite using a different tracking algorithm and a different robot, results are comparable to [12]. The used robot poses a difficult challenge to track using any model-based trackers mainly due to its repetitive appearance and texture-less links, which convinced us to use markers instead. This proves our framework is versatile and usable in distinct situations. From the results, in most of the cases the RMSE values are $< 3°$ except for joint 4, which is the most challenging joint (elbow of the manipulator) to estimate. For very few cases, RMSE values reached $> 5°$, which we believe are due to the problems associated with tracking such as inaccurate tracker intialisations caused by the camera and marker placements, and pose confusions when camera viewing axis becomes orthogonal to marker axis. This is true particularly for the last two markers and results are consequently affected negatively. Also, the errors are higher when the joint values are far from the zero position. Nevertheless, such errors are highly tolerable for the decommissioning tasks that motivated this work.

TABLE III
MEAN AND STANDARD DEVIATION (EXPRESSED IN DEGREES) COMPUTED FOR THE TRAJECTORY RMSE VALUES DEPICTED IN TABLES I AND II.

joint(#)	Mean(°)	*Std.* (°)
q_1	1.4441	1.2409
q_2	0.9270	0.9659
q_3	1.1542	0.6834
q_4	4.1467	2.5694
q_5	3.0573	1.8139
q_6	3.6195	2.4143
q_7	3.0752	1.7520

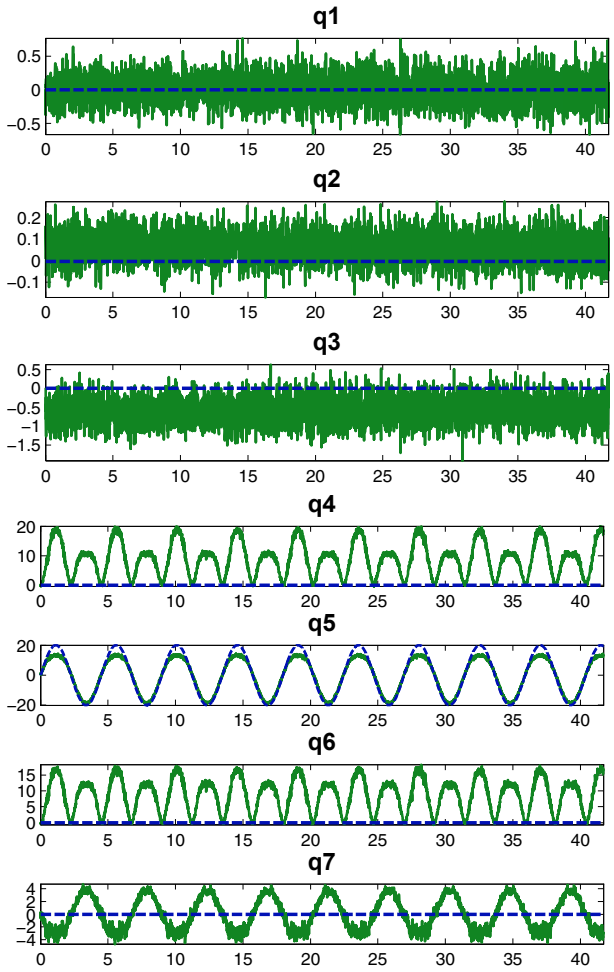


Fig. 4. Trajectory 11. Blue dashed line represents the real trajectory, while the estimated trajectory is in green solid line.

2) Using estimates as feedback in a joint controller: the robot was asked to follow trajectories specified in the joint space. Difficulty of placement for markers 3 and 4 and the relative errors induced on the estimations of joints 4, 5, 6 and 7, and the propagation of the errors in estimating the first 3 joints make the control of the remaining axes of the robot very challenging. Consequently we first focus on the first 2 markers, thus limiting the motion of the robot to the first 3

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMECH.2018.2865758, IEEE/ASME Transactions on Mechatronics

TABLE IV
RMSE OF ESTIMATES WITH RESPECT TO GROUND TRUTH VALUES.

joint	RMSE (in degrees [°])					
	traj 1	traj 2	traj 3	traj 4	traj 5	traj 6
q_1	2.1895	2.3207	2.6822	2.9667	3.6575	2.7492
q_2	2.6638	0.8453	0.7577	2.0913	1.1689	1.6167
q_3	4.2416	3.2690	4.5955	4.5042	5.0706	3.5712

TABLE V
RMSE VALUES OF REAL JOINT VALUES AND REFERENCE MOTION. ERRORS ARE EXPRESSED IN DEGREES.

joint(#)	RMSE (in degrees [°])					
	traj 1	traj 2	traj 3	traj 4	traj 5	traj 6
q_1	2.1665	2.3003	2.6762	2.9561	3.6405	2.7670
q_2	2.6606	0.8421	0.7547	2.1028	1.1752	1.7797
q_3	4.1938	3.2269	4.5817	4.4774	5.0540	3.5266

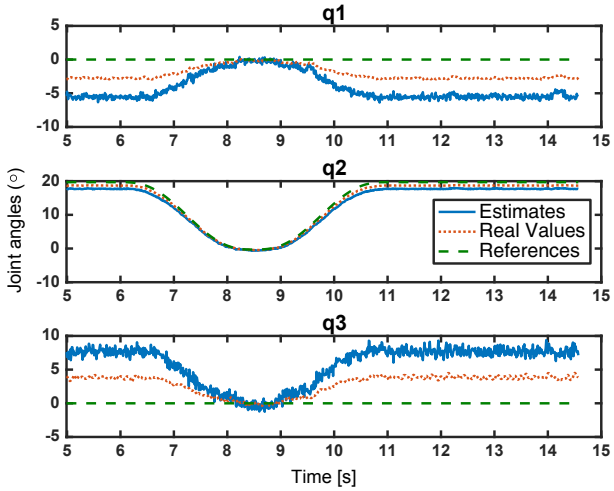
joints, as proof of concept. The robot performs 6 trajectories: the first 3 trajectories exciting each joint singularly. The other 3 trajectories presented simultaneous motions for all the 3 joints. Table IV reports errors between the estimates used in the controller and the ground truth values as read from the encoders. Table V presents task errors computed as difference between reference positions and ground truth values as read from the encoders. Notice that the controller uses only the difference between estimates and reference positions. Fig. 5 shows the evolution of estimates (red), ground truth values (blue) and references (green) during the performance of trajectories 2 and 5.

Analysis of results: estimates can be used to control the robot and task errors induced by errors in the estimates are of the same magnitude of the estimation errors. This is clear comparing Table IV and Table V. Also, task errors do not go to zero because of the errors in the estimations. This is to be expected because, although such a controller guarantees convergence with precise feedback, we are using estimates of the state, thus convergence to zero is not guaranteed anymore. Additionally, we also conducted experiments controlling all the joints. The results are reported in Tables VI and VII. It can be seen from the results that the errors of estimation and task regarding the last 4 joints are higher than the first 3 joints, as we expected. This is due to the difficulty in positioning precisely the two markers (as described earlier). However, errors are never higher than 8 degrees.

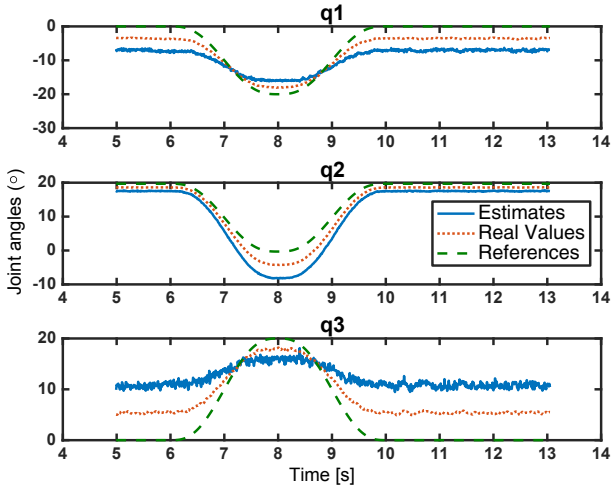
B. Contact Detection

During the experiments, we detect a contact only when both methods *i.e.*, kinematic residuals and sheer difference, detect a contact. This has been done to achieve a higher robustness.

1) With encoder readings: we apply our methods to detect contacts to situations where encoder readings are available. An operator physically pushed the robot (relying on the partial back-drivability of the used robot) for 5 times every 10 seconds starting from 20 seconds in each experiment. Every experiment is 1 minute long, and there are 5 contacts at 20 s, 30 s, 40 s, 50 s and 60 s. Table VIII shows the results of the 5 trials.



(a) Trajectory 2.



(b) Trajectory 5.

Fig. 5. These figures show the evolution of the estimates (blue solid), real values (red dotted) and reference (green dashed) for each experiment.

TABLE VI
RMSE VALUES OF ESTIMATES WITH RESPECT TO GROUND TRUTH VALUES. ERRORS ARE EXPRESSED IN DEGREES.

joint(#)	RMSE (in degrees [°])		
	traj 1	traj 2	traj 3
q_1	2.5402	2.0437	2.5011
q_2	2.3265	1.9597	2.2247
q_3	3.4943	3.3668	3.8382
q_4	4.0870	4.1576	7.4786
q_5	4.5054	3.4631	6.6244
q_6	2.4200	3.3296	5.2717
q_7	5.0569	3.6256	4.9288

False positives detected in those 5 trials are due to the bounces created by compliant reaction of the robot to the operator pushes.

Analysis of results: most of the contacts are reliably detected during the trials. However, our choice of threshold was conservative and it detected additional false positive contacts in 60% of the trials. Since we decided to have an operator push the robot instead of having a real contact with

TABLE VII
RMSE VALUES OF REAL JOINT VALUES AND REFERENCE MOTION. ERRORS ARE EXPRESSED IN DEGREES.

joint(#)	RMSE (in degrees [°])		
	traj 1	traj 2	traj 3
q_1	2.5383	2.0509	2.5088
q_2	2.3439	1.9861	2.2505
q_3	3.4641	3.3279	3.7986
q_4	4.0082	4.0636	7.4306
q_5	4.4579	3.4045	6.5923
q_6	2.2266	3.2488	5.2017
q_7	4.9333	3.4119	4.8382

TABLE VIII
SUMMARY OF RESULTS. THIS TABLE SHOWS THE RESULTS OF THE 5 TRIALS HIGHLIGHTING THE NUMBER OF DETECTED CONTACTS AND THE NUMBER OF FALSE DETECTIONS.

Trial	Detected Contacts	Additional false positives
1	5/5	3
2	3/5	0
3	5/5	1
4	5/5	0
5	4/5	4

the environment, the pushing force was always different. This added to the difficulty of detecting contacts.

2) With vision-based estimates used in the controller: we restrict the robot to be a 3DoF manipulator for this set of experiments. Results of experiments where the robot was asked to perform a trajectory with displacements of $\Delta q_1 = -20$, $\Delta q_2 = -30$, and $\Delta q_3 = 10$ are reported in Fig. 6. Residuals are generally non-zero because of the error in estimating the configuration. We force joint blocks to emulate contacts, *i.e.*, blocks are meant to emulate the result of hitting an object and preventing the robot to move the joint further. Those blocks are effectively joint limits, which produce kinematic constraints to the robot. In detail, we set the block to $-5°$ for joint 1 and whenever the robot was commanded to go to a configuration minor than that, the robot stayed at $-5°$ position. The values for joint 2 and 3 were respectively 15 and 2. Fig. 6(b), 6(c) and 6(d) report the results of performing the same trajectory while enabling blocks on one joint at a time. Every graph reports the residuals and a yellow line has 0 value when no contact is detected and climbs to 100 when a contact is detected. When the block was applied to joints 1 or 2, a contact was detected, as expected. The block in the joint 3 did not trigger any detection.

Analysis of results: the contact in the last experiment has not been detected due to the trajectory displacement being not too important and thus causing the relative residual to not increase enough. This is an unwanted feature of our methods and future work will address it to ameliorate the rate of success of contact detection. Earlier on, we introduced the revised model for the residuals using estimates, and we stated how critical thresholds are in order to correctly detect contacts. Results show that setting the thresholds is particularly hard and it has to depend on: (i) the trajectory and (ii) the accuracy of the estimates. It has to be noticed that a high threshold

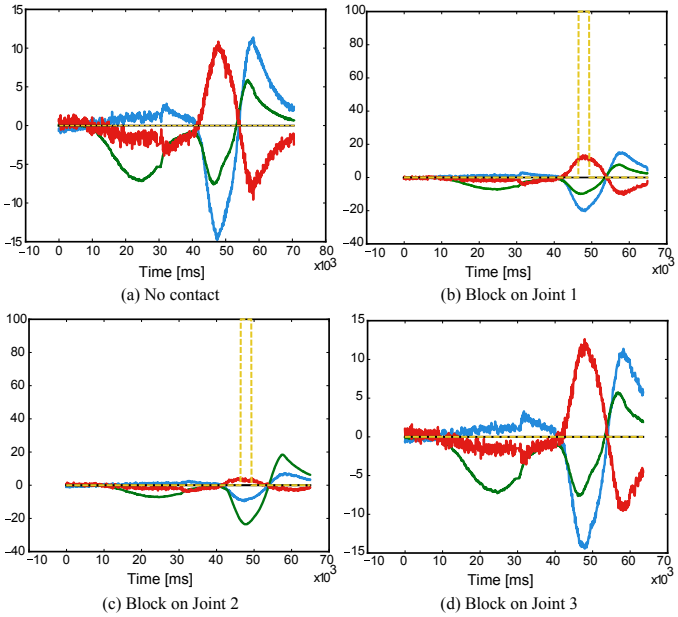


Fig. 6. Contact Detection. (a) Reports the residuals while performing the trajectory in presence of no block. Residuals are non-zero because of imprecision of estimates. (b) - (d) Results of the performance of same trajectory but applying blocks respectively to joint 1, 2 and 3. Residuals are reported in multiple colours and the yellow dashed line represents detection of contacts: when 0 no contact is detected, whilst when 100, a contact is detected. In summary, the block applied on joints caused contact detection, as expected. The block in the joint 3 did not trigger any detection however.

would add tolerance to errors in the estimates, but it would make the system less sensitive to real contacts. On the other hand, low thresholds increase the number of false positives due to the errors in the estimations. A trade-off is complicated but critical to find, and at this moment, it has to be found on a case-by-case manner.

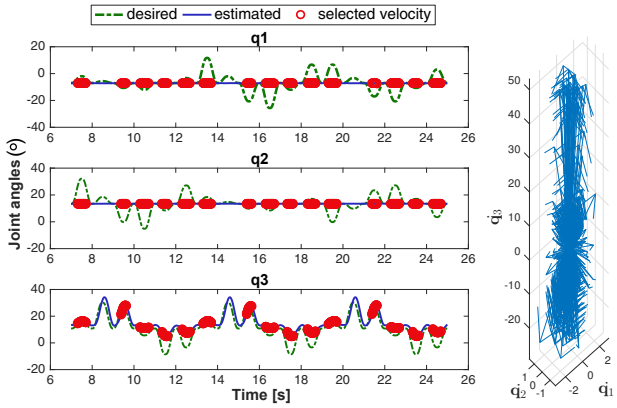
C. Contact Constraint Estimation

We continue the line of experiments described in the previous subsection performing an exploration after detecting a contact. In this series of experiments, the exploration has been defined directly in the joint space and has the goal of estimating the constraint(s) arising from contacts with the environment or blocks such as joint limits. We investigate the behaviour of our framework in presence of bilateral and unilateral constraints. We replicate the blocks introduced in the previous section to emulate unilateral constraints, *e.g.*, when the block was on joint 1, the robot was not able to go to less than $-5°$ but it could go to higher values. In another set of experiments, blocks are hard constraints thus the robot is not permitted to move from the blocked configuration, *e.g.*, the block was set to $-5°$ for joint 1 and joint 1 was not moved at all. This was to emulate bilateral constraints. Three experiments are run to estimate unilateral constraints and three for bilateral constraints. In the first experiment of each set, joint 1 is limited to be more than or equal to $-5°$. In the second experiment joint 2 is blocked to be more than or equal to $15°$. And finally, in the third experiment both blocks are enabled at the same time. This is done to investigate the behaviour of our method in presence of multiple simultaneous constraints.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMECH.2018.2865758, IEEE/ASME Transactions on Mechatronics

TABLE IX
SUMMARY OF RESULTS. THIS TABLE SHOWS THE REAL CONSTRAINT $\mathbf{\Lambda}^q(\mathbf{q})$ AND THE ESTIMATED CONSTRAINT $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$.

Experiment	$\mathbf{\Lambda}^q(\mathbf{q})$	$\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$
1	[1,0,0]	[0.9867, 0.1622, -0.0105]
2	[0,1,0]	[0.2593, -0.9658, -0.0030]
3	$\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$	$\begin{bmatrix}-0.9991 & -0.0414 & 0.0006\\ -0.0414 & 0.9991 & -0.0003\end{bmatrix}$



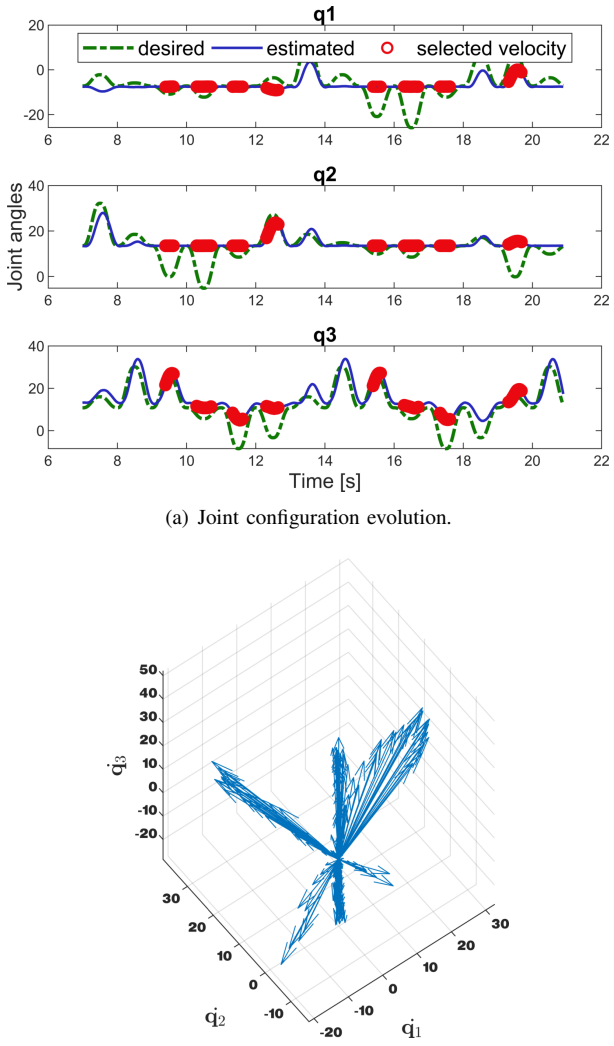
(a) Joint configuration evolution. (b) Velocity set (degrees/sec).

Fig. 7. (a) Evolution of joint configuration and desired configurations during the exploration. Red circles denote when the configuration error between desired and estimated configurations is higher than a threshold thus regarded as due to a contact. (b) Relative velocity set built selecting the estimated velocities when configuration error is higher than the threshold. Since the constraints are bilateral, the set of velocity has components only in the $\dot{q}_3$ direction of the graph. Small components in the other directions are due to configuration estimation errors.

Velocity sets are recorded storing velocity vectors associated to high positional error and analysed offline. The selection criterion of such velocity vectors relies on the assumption that if there is an important positional error this must be due to a contact.

Bilateral constraints: Bilateral constraints fully constrain motion, *i.e.*, motion in a specific direction is not admissible. Table IX reports the estimated $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ (refer to Eq. (19)) and the real $\mathbf{\Lambda}^q(\mathbf{q})$ in all three experiments. Fig. 7 reports the set of joint velocities used to compute the estimate of the constraint in experiment 3. Results show that our method succeeds in estimating the kinematic constraint to a high degree of precision. Moreover, the sign inversion between $\mathbf{\Lambda}^q(\mathbf{q})$ and $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ does not have any influence in this case, since motion is blocked in the entire direction ($\mathbf{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}$), *i.e.*, $[0, 1, 0]$ and $[0, -1, 0]$ represent the same constraint.

Unilateral constraints: in these situations, contacts limit motion only towards the contacts themselves, but the robot is able to leave the contacts and get back to free motion. Table X reports the estimated $\mathbf{\Lambda}^q(\mathbf{q})$ (refer to (19)). Fig. 8 reports the set of joint velocities used to compute the estimate of the constraint in experiment 3. Results show that our method succeeds in estimating the kinematic constraint to a high degree of precision, although in the case of experiment 1 the sign is reversed. A check on the sign would ensure that $\mathbf{\Lambda}^q(\mathbf{q})$ and $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$ have the same sign ($\mathbf{\Lambda}^q(\mathbf{q})\dot{\mathbf{q}} \geq \mathbf{0}$). However due to the errors on the estimates, velocities might have components also in the prohibited direction and a simple check is not enough. Future work includes a more accurate way to test the sign of the constraints.



(a) Joint configuration evolution.

(b) Velocity set.

Fig. 8. (a) Evolution of joint configuration and desired configurations during the exploration. (b) Relative velocity set built selecting the estimated velocities when configuration error is higher than the threshold. Since the constraints are only unilateral, the set of velocity has components not only in the $\dot{q}_3$ direction of the graph, as in the case of Fig. 7(b). However notice that in the $\dot{q}_1$ and $\dot{q}_2$ directions, only positive velocities are present. This is due to the fact that during the exploration, a constraint can be active while the other might not, and vice versa.

TABLE X
SUMMARY OF RESULTS. THIS TABLE SHOWS THE REAL CONSTRAINT $\mathbf{\Lambda}^q(\mathbf{q})$ AND THE ESTIMATED CONSTRAINT $\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$.

Experiment	$\mathbf{\Lambda}^q(\mathbf{q})$	$\tilde{\mathbf{\Lambda}}^q(\mathbf{q})$
1	[1,0,0]	[-0.9922, -0.1225, -0.0233]
2	[0,1,0]	[-0.1558, 0.9853, 0.0705]
3	$\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\end{bmatrix}$	$\begin{bmatrix}0.1039 & 0.9945 & -0.0128\\ 0.9752 & -0.1044 & -0.1949\end{bmatrix}$

Analysis of results: results show that our method succeeds in estimating constraints. Our choice of emulating contacts with software-induced blocks was motivated by our will to be conservative with respect to the robot integrity and to show that our framework can be used to estimate joint limits too. Also because those blocks did not jeopardise the vision part, *i.e.*, blocks did not obstruct any of the trackers to be visible during the entire experiment.

VII. Conclusions

We presented a control framework which is able to estimate the robot's configuration, by tracking markers on robot links using images from commercial cameras. We set an optimisation problem and used the kinematic model of the robot. The joint values are retrieved as those which best fit all the relations. These estimates are used to not only control the robot, but also to check for contacts. Furthermore, whenever a contact was detected, the robot performed a predefined set of exploratory motions, which enable the contact constraints to be estimated, by comparing the difference between commanded and observed velocity vectors. We have presented a variety of experimental results, which showed how our framework can control the robot and when in contact, can estimate the kinematic constraints.

We believe it is possible to achieve more precise results in estimating the joint configuration using more sophisticated tracking methods. Current and future work includes also the use of the whole body of the robot for further experiments on contact detection and constraint estimation. Also, velocity sets will be analysed online, and constraints will be estimated in real-time, so that the robot will be able to resume the task when needed. Finally, we intend to extend this framework to both discover and control the robots dynamics, in addition to the kinematic control described in this paper.

Acknowledgements

This paper is dedicated to the memory of co-author Jeffrey Kuo (1969 - 2017), without whose inspiration this work could not have happened. Dr. Kuo was an important pioneer in the field of nuclear robotics, and was the industrial leader of the industry-academia collaboration that this paper is based on. We are privileged to have known this great gentleman as both a colleague and a friend.

The authors would like to thank Prof. Peter I. Corke of the QUT for his helpful comments and feedback.

References

[1] N. Marturi, A. Rastegarpanah, V. Ortenzi, Y. Bekiroglu, R. Stolkin, and V. Rajasekaran, "Towards advanced robotic manipulations for nuclear decommissioning," in *Robots Operating in Hazardous Environments* (H. Canbolat, ed.), ch. 3, Rijeka: InTech, 2017.

[2] V. Ortenzi, M. Adjigble, J. A. Kuo, R. Stolkin, and M. Mistry, "An experimental study of robot control during environmental contacts based on projected operational space dynamics," in *Proc. IEEE International Conference on Humanoid Robots*, pp. 407–412, IEEE, 2014.

[3] V. Ortenzi, R. Stolkin, J. A. Kuo, and M. Mistry, "Projected inverse dynamics control and optimal control for robots in contact with the environment: A comparison," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 4009–4015, IEEE, 2015.

[4] N. Marturi, A. Rastegarpanah, C. Takahashi, R. Stolkin, and et al., "Towards advanced robotic manipulation for nuclear decommissioning: a pilot study on tele-operation and autonomy," in *IEEE International Conference on Robotics and Automation for Humanitarian Applications*, pp. 1–8, IEEE, 2016.

[5] M. Gabiccini, E. Farnioli, and A. Bicchi, "Grasp analysis tools for synergistic underactuated robotic hands," *Int. J. Rob. Res.*, vol. 32, no. 13, pp. 1553–1576, 2013.

[6] R. R. Ma, J. T. Belter, and A. M. Dollar, "Hybrid deposition manufacturing: design strategies for multimaterial mechanisms via three-dimensional printing and material deposition," *J. Mech. Robot.*, vol. 7, no. 2, p. 021002, 2015.

[7] A. M. Dollar and R. D. Howe, "The highly adaptive sdm hand: Design and performance evaluation," *Int. J Rob. Res.*, vol. 29, no. 5, pp. 585–597, 2010.

[8] CEA Tech, "Sybot." http://sybot-industries.com, 2016. Accessed: 2017-03-21.

[9] H. Wang, B. Yang, Y. Liu, W. Chen, X. Liang, and R. Pfeifer, "Visual servoing of soft robot manipulator in constrained environments with an adaptive controller," *IEEE/ASME Trans. Mechatron.*, vol. 22, pp. 41–50, Feb 2017.

[10] A. Shiva, A. Stilli, Y. Noh, A. Faragasso, I. De Falco, G. Gerboni, M. Cianchetti, A. Menciassi, K. A. Althoefer, and H. A. Würdemann, "Tendon-based stiffening for a pneumatically actuated soft manipulator," *IEEE Robot. Autom. Lett.*, vol. 1, pp. 632–637, 7 2016.

[11] H. Al-Fahaam, S. Davis, and S. Nefti-Meziani, "Wrist rehabilitation exoskeleton robot based on pneumatic soft actuators," in *proc. Int. Conf. Students on Applied Engineering*, pp. 491–496, IEEE, 2016.

[12] V. Ortenzi, N. Marturi, R. Stolkin, J. A. Kuo, and M. Mistry, "Vision-guided state estimation and control of robotic manipulators which lack proprioceptive sensors," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 3567–3574, Oct 2016.

[13] V. Ortenzi, H.-C. Lin, M. Azad, R. Stolkin, J. A. Kuo, and M. Mistry, "Kinematics-based estimation of contact constraints using only proprioception," in *Proc. IEEE International Conference on Humanoid Robots*, pp. 1304–1311, IEEE, 2016.

[14] S. Hutchinson, G. Hager, and P. I. Corke, "A tutorial on visual servo control," *IEEE Trans. Robotics and Automation*, vol. 12, pp. 651–670, 1996.

[15] F. Chaumette and S. Hutchinson, "Visual servo control, part i: Basic approaches," *IEEE Robot. Autom. Mag.*, vol. 13, pp. 82–90, December 2006.

[16] M. Keshmiri and W. F. Xie, "Image-based visual servoing using an optimized trajectory planning technique," *IEEE/ASME Transactions on Mechatronics*, vol. 22, pp. 359–370, Feb 2017.

[17] F. Janabi-Sharifi, L. Deng, and W. J. Wilson, "Comparison of basic visual servoing methods," *IEEE/ASME Transactions on Mechatronics*, vol. 16, pp. 967–983, Oct 2011.

[18] E. Marchand, F. Chaumette, F. Spindler, and M. Perrier, "Controlling an uninstrumented manipulator by visual servoing.," *I. J. Rob. Res.*, vol. 21, no. 7, pp. 635–648, 2002.

[19] K. Nickels and S. Hutchinson, "Model-based tracking of complex articulated objects," *IEEE Trans. Rob. Autom.*, vol. 17, no. 1, pp. 28–36, 2001.

[20] S. Pellegrini, K. Schindler, and D. Nardi, "A generalisation of the icp algorithm for articulated bodies.," in *Proc. British machine vision conference*, September 2008.

[21] A. I. Comport, É. Marchand, and F. Chaumette, "Kinematic sets for real-time robust articulated object tracking," *Image Vis. Comput.*, vol. 25, no. 3, pp. 374–391, 2007.

[22] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 932–946, 2002.

[23] K. Pauwels, V. Ivan, E. Ros, and S. Vijayakumar, "Real-time object pose recognition and tracking with an imprecisely calibrated moving rgb-d camera," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 2733–2740, 2014.

[24] M. Klingensmith, T. Galluzzo, C. Dellin, M. Kazemi, J. A. D. Bagnell, and N. Pollard , "Closed-loop servoing using real-time markerless arm tracking," in *Proc. IEEE Int. Conf. Robot. Autom. (Humanoids Workshop)*, May 2013.

[25] K. Pauwels and D. Kragic, "Simtrack: A simulation-based framework for scalable real-time object pose detection and tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 1300–1307, IEEE, 2015.

[26] N. B. Figueroa, F. Schmidt, H. Ali, and N. Mavridis, "Joint origin identification of articulated robots with marker-based multi-camera optical tracking systems," *Rob. Auton. Syst.*, vol. 61, no. 6, pp. 580–592, 2013.

[27] T. Schmidt, R. Newcombe, and D. Fox, "Dart: dense articulated real-time tracking with consumer depth cameras," *Auton. Robots*, vol. 39, no. 3, pp. 239–258, 2015.
[28] J. Bohg, J. Romero, A. Herzog, and S. Schaal, "Robot arm pose estimation through pixel-wise part classification," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 3143–3150, 2014.
[29] F. Widmaier, D. Kappler, S. Schaal, and J. Bohg, "Robot arm pose estimation by pixel-wise regression of joint angles," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 616–623, IEEE, 2016.
[30] M. Ma, N. Marturi, Y. Li, A. Leonardis, and R. Stolkin, "Region-sequence based six-stream cnn features for general and fine-grained human action recognition in videos," *Pattern Recognition*, vol. 76, pp. 506–521, 2018.
[31] M. Ma, N. Marturi, Y. Li, R. Stolkin, and A. Leonardis, "A local-global coupled-layer puppet model for robust online human pose tracking," *Comput. Vis. Image Underst.*, vol. 153, pp. 163–178, 2016.
[32] C. Gehring, C. D. Bellicoso, S. Coros, M. Blösch, P. Fankhauser, M. Hutter, and R. Siegwart, "Dynamic trotting on slopes for quadrupedal robots.," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 5129–5135, 2015.
[33] V. Ortenzi, R. Stolkin, J. Kuo, and M. Mistry, "Hybrid motion/force control: a review," *Advanced Robotics*, vol. 31, no. 19-20, pp. 1102–1113, 2017.
[34] T. Yoshikawa and A. Sudou, "Dynamic hybrid position/force control of robot manipulators-on-line estimation of unknown constraint," *IEEE Trans. Rob. Autom.*, vol. 9, no. 2, pp. 220–226, 1993.
[35] H.-C. Lin, M. Howard, and S. Vijayakumar, "Learning null space projections.," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 2613–2619, IEEE, 2015.
[36] É. Marchand, F. Spindler, and F. Chaumette, "Visp for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robot. Autom. Mag.*, vol. 12, no. 4, pp. 40–52, 2005.
[37] E. Marchand, H. Uchiyama, and F. Spindler, "Pose estimation for augmented reality: a hands-on survey," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 12, pp. 2633–2651, 2016.
[38] L. Sciavicco and B. Siciliano, *Modelling and control of robot manipulators*. 1439-2232, Springer Science & Business Media, London, 2012.
[39] A. De Luca and R. Mattone, "Sensorless robot collision detection and hybrid force/motion control," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 999–1004, IEEE, 2005.

Valerio Ortenzi is a Postdoctoral Research Fellow at Queensland University of Technology and ARC Centre of Excellence for Robotic Vision since 2017. He recently finished his PhD at the University of Birmingham, UK and holds an undergraduate degree in Automation and Automated Systems Engineering (2010) and a Master in Artificial Intelligence and Robotics (2012), both from Sapienza Universit di Roma. From January to July 2013, Valerio worked as a collaborator at DIAG, Sapienza Universit di Roma. His research focuses on two different directions: visual servoing and control using robot dynamics.

Naresh Marturi (M'11) is a KTP Robot Vision Scientist with KUKA Robotics UK Ltd. and the University of Birmingham, since 2015. He obtained his Ph.D. degree in automatic control from the Université de Franche-Comté, Besançon, France, in 2013 and M.S. degree in robotics and intelligent systems from Örebro University, rebro, Sweden, in 2010. After Ph.D. he spent one year at FEMTO-ST Institute, France as a post-doctoral researcher. His primary research interests are in the fields of machine vision, industrial robotics, human-robot interaction, vision-based robotic control and deep learning. At Kuka, he is majorly responsible for developing industrially robust vision-guided applications by transferring academic knowledge, tools and techniques to industry. He also possesses a solid knowledge and research experience in developing vision-guided techniques at micro and nano scales.

Michael Mistry (M'07) is a Reader in robotics at School of Informatics, University of Edinburgh, where he is also a member of the Edinburgh Centre for Robotics. Michael is broadly interested in legged robotics, particularly in human motion and humanoid robotics. His research focuses on issues relevant to dexterous movement in legged robots, including redundancy resolution and inverse kinematics, operational space control and manipulation, stochastic optimal control, and internal model learning and control, particularly in environmental contact. Previously, Michael has been a Senior Lecturer in robotics at School of Computer Science, University of Birmingham, a postdoc at the Disney Research Lab at Carnegie Mellon University, a researcher at the ATR Computational Neuroscience Lab and a Ph.D. student in Stefan Schaals CLMC lab at the University of Southern California.

Jeffrey Kuo is a Chartered Engineer and Member of the Institute of Mechanical Engineers, with a Ph.D. in robot kinematics and has almost 20 years of experience with in the nuclear industry. He has developed industrial robots systems and remote engineering solutions for nuclear operations and decommissioning. This includes design, test, commissioning and deployment and he has much expertise in plant process control and data acquisition. As the NNL Research Fellow in Robotics, he is actively involved with advancing the state of robot systems for nuclear applications by working with partners within and external to the nuclear sector. This includes promoting industry-led academic research by means of academic and industrial collaboration,supervision of Ph.D.s and lecturing. He is technically leading projects to produce end user robot systems and robot simulators for customers and collaborative R&D projects funded by the European Union's Horizon2020 programme and InnovateUK.

Rustam Stolkin (M'05) is a Royal Society Industry Fellow for Nuclear Robotics and is the Director of A.R.M Robotics Ltd Birmingham UK. He is also Chair of Robotics at University of Birmingham, where he is director of the Extreme Robotics Lab, dedicated to robotics in extreme environments, with a particular focus on nuclear decommissioning. He has been with the University of Birmingham since 2008. His training included undergraduate and masters degrees in Engineering from Oxford University, and a PhD in Robot Vision undertaken between University College London and UK imaging industry. He also worked as an Assistant Professor (Research) at Stevens Institute of Technology, USA, 20042008. Stolkin is highly cross-disciplinary, with patents, publications and projects spanning computer vision, sensor systems, robotic manipulation, robot vehicles, and human-robot interfaces. He is an international leader in the development of robots for nuclear applications, currently leading projects spanning six countries on three continents. He is an active member of the IEEE Technical Committee on Robotics and Automation in Nuclear Facilities.