Using General-Purpose Serial-Link Manipulators for Laparoscopic Surgery with Moving Remote Center of Motion

Murilo M. Marinho[*], Mariana C. Bernardes[†], Antonio P. L. Bo[‡]

Department of Electric Engineering, Universidade de Brasília, Brasília, DF, Brazil

Minimally invasive surgical systems are being widely used to aid operating rooms across the globe. Although arguably successful in laparoscopic surgery, the da Vinci robotic system has limitations mostly regarding cost and lack of patient physiological motion compensation. To obtain a more cost-effective alternative, earlier works used general-purpose fully actuated serial-link robots to control instruments in laparoscopic research using constrained Jacobian techniques. In contrast with those works, we present a new technique to solve the laparoscopic constraints for the serial-link manipulator by using a constrained trajectory. This novel technique allows complex 3D remote center-of-motion trajectories to be taken into account. Moreover, it does not have problems related to drifting, and is less prone to singularity related issues as it can be used with redundant manipulators. The proof-of-concept experiments are done by performing artificial trajectories with static and moving trocar points using a physical robot manipulator. Furthermore, the system is tested using user input of 13 medically untrained personnel in an endoscope navigation task. The experiments show that the system can be operated reliably under arbitrary and unpredictable user inputs.

Keywords: Robotic surgery; dual quaternions; kinematic control.

1. Introduction

Minimally Invasive Surgery (MIS) is a surgical technique based on the access to inner cavities via multiple small incisions in the patient's body, being consequently less invasive than its open surgery counterpart. MIS interventions are the preferred approach in many procedures since they offer outstanding advantages like less pain, smaller scars, and faster recovery time [1]. However, as a higher level of surgical skill is required, MIS procedures often have steeper learning curves and require longer procedure times.

The difficulties in performing laparoscopy originate from the motion restriction imposed by the small incisions in the patient's skin. Long and thin tools are inserted through those incisions so that the surgeon can act inside the patient. Those tools include forceps-like instruments to interact with tissues, and also endoscopes that embed a camera to enable visualisation of the inner cavity. To simplify the analysis, the intersection of the trocar axis with the skin incision is usually approximated by a pivoting point, henceforth named as the *trocar point*. With respect to the trocar point, this means that the tool motion is restricted to four degrees-of-freedom (DOF) (see Fig. 1), and that the motion of the outside end of the tool is mirrored at the end inside the patient's cavity. Moreover, the surgeon can no longer interact with the organs and inner structures directly using their hands, thus losing dexterity and tactile feedback. Furthermore, as in many tasks — such as suturing — the surgeon needs both hands to operate tools, an assistant surgeon may be required to operate the endoscope.

Received 4 February 2016; Revised 31 May 2016; Accepted 16 June 2016; Published xx xx xx. This paper was recommended for publication in its revised form by Editor, Iulian Ioan Iordachita.

Email Addresses: [*]*murilomarinho@lara.unb.br*, [†]*bernardes@lara.unb.br*, [‡]*antonio.plb@lara.unb.br*

Current address: Mitsuishi-Sugita laboratory, Department of Mechanical Engineering, The University of Tokyo, Tokyo, Japan.

NOTICE: Prior to using any material contained in this paper, the users are advised to consult with the individual paper author(s) regarding the material contained in this paper, including but not limited to, their specific design(s) and recommendation(s).

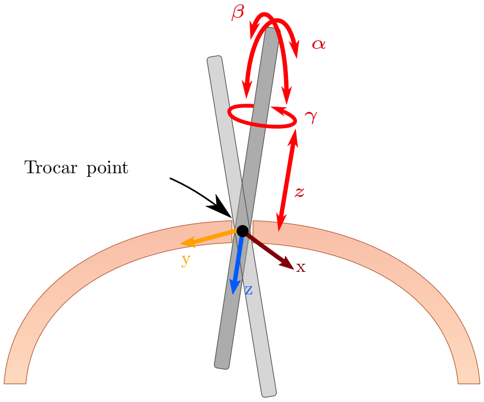


Fig. 1. Laparoscopy motion around the trocar point. As a result, the endoscope can only move in four DOF: α, β and γ, which are rotations around the x, y and z-axis, respectively, and z which is the translation along the z-axis.

In addition to the aforementioned difficulties, patient's physiological movements, such as secondary respiratory motion, further enhance the complexity of MIS procedures. In clinical settings, respiration causes quasi-periodic motion and deformation of organs, being a source of disturbance that affects performance of several types of procedures such as heart surgery, stereotactic surgery, and laparoscopic surgery [2]. Mechanical organ stabilizers are often used in open surgical procedures. However, the stabilizers available for laparoscopic surgery have considerable drawbacks, due to the difficulties of having a stiff structure able to fit through the small incisions. There is the alternative of completely eliminating the source of disturbance through direct influence in the organ, i.e. stopping the heart completely during surgery, but that is an invasive method that may increase morbidity [2–4].

Robots already aid in medical settings by providing tremor canceling, motion scaling, comfort, endurance, improved vision, and so on [5]. The basic concept of robotic assistants in laparoscopy can be reduced to two clear objectives: to move the tool to the position where the surgeon defines, and to generate a *remote center of motion* (RCM), which should have the same instantaneous position as the trocar point. The euclidean distance between the RCM and the trocar point is often called the RCM error, and is used as a performance metric.

The first step towards safe RCM generation is deciding how to handle the kinematic constraints of the trocar. This topic has been considered by myriad works, in which the compensation can be either a mechanical constraint or embedded in the robot control algorithm. The most common approach to solve this issue is through mechanical constraints, basically variants of the parallel bars mechanism used in the da Vinci robot. The da Vinci is arguably the most successful robotic assistant system available, even with a U$1.5M cost figure [5]. Besides the high cost, da Vinci is also known for the lack of force feedback, the large operating room footprint, and the closed architecture. This prompted the research of alternative system designs. For instance, the ALF-X [6] and the RAVEN II [7,8].

Whereas modifying the machine design may allow several aspects to be improved, the use of general purpose serial-link manipulators is another feasible approach. Serial-link manipulators have been reliably used in the medical room for several years, for instance as a component of the CyberKnife [9] and ROSA [10] systems. When applied in laparoscopic surgery, general purpose serial-link manipulators do not account mechanically for the trocar point constraints. Instead, they need a software framework to provide virtual fixtures. One advantage is that, since the manipulator is not mechanically constrained for laparoscopic surgery, it can have a design optimized for different kinds of surgery, allowing the same robot to be used in distinct surgical areas. By using the same robot in distinct surgical procedures across multiple medical fields, the operating costs can be diluted, making the system more economically viable. In addition, robots with more than 6 DOF with the appropriate control software can use the additional DOF to reposition the robot inside the workspace without the need to halt the procedure, which can reduce the amount of times in which the procedure needs to be stopped for manual readjustments of the robotic assistant. Less delays in the operation of the system increase its throughput, reducing patient's exposure, surgeon's fatigue, and associated costs. The most well-known example is the operating table mounted DLR MIRO robot [11]. In the commercial scenario, and with similar motivation, KUKA[a] has developed a medical robot line.

1.1. *Related work*

In this work, we examine a particular approach using fully actuated robotic systems, such as the DLR Miro. Such systems may allow more precise movements, while providing physiological motion compensation. That compensation could be, for instance, of pulmonary and heart function that cause cyclic movements during surgical interventions. As trocars in some cases are placed in areas of low compliance, such as between rib bones, motion compensation could be beneficial. Moreover, minimally invasive heart interventions that traditionally require a full stop of the heart beat could receive beating-heart alternatives, which would be safer for the patient considering the risks and physiological stress caused by a heart-lung machine.

In order to control a robot in laparoscopic procedures, related works using fully actuated robots often derive a constrained Jacobian using the trocar point restrictions.

[a] http://www.kuka-healthcare.com/en/start.htm.

For instance, Osa *et al.* [12] used four ceiling mounted 6 DOF Mitsubishi robots by considering the trocar point constraints a relation between the end effector linear and rotational velocities. Azimian *et al.* [13] used a Mitsubishi PA10-7C robot to control a tool under trocar point constraints by designing a constrained Jacobian. The experiments showed considerable tool-tip deviation (>2 cm), which can be explained by conflicting objectives in their task-priority controller. In both works described before, there is a zero velocity constraint at the trocar point with no feedback loop using the RCM position, making the unavoidable nonzero noisy velocities eventually cause the RCM position to diverge from the trocar position, which may be dangerous since surgical procedures can be rather long. Given the zero velocity constraint, the aforementioned solutions have no direct extension to moving trocar points. In a more recent work, [14] studied another Jacobian considering the trocar point constraints. The main result of the work required trocar point velocities constrained to zero, making it susceptible to the same type of drift as in Azimian *et al.*'s Jacobian. Moreover, as organ motion and deformation modeling can be quite complex, it should be convenient to describe trocar point motion as a 3D position vector, rather than the 2D plane velocities used in From's work.

Workspace singularities are also out of the scope of the preceding works discussed. However, workspace singularities are naturally occurring mechanical related phenomena, that need to be considered when using serial-link robotic manipulators near surgeons and patients. In the case of redundant robots, there can be infinite singular configurations inside the workspace, which can be hard (or impossible) to model beforehand.

In this work, we examine the particular approach of describing the task as a constrained trajectory. Such description is more frequent when describing robotic motion, and may allow a more powerful insight on the inner workings of the control algorithm. Furthermore, by describing the problem as a trajectory instead of modifying the Jacobian, we can use known metrics to evade singular configurations, while also accounting for trocar point motion. Initially, in [15], we explored the trajectory planning for a static trocar point using a dual quaternion finite-point interpolation method in simulations only. That framework, however, is not enough for compensating trocar point motion, and is updated in this work to follow a continuous trajectory. Moreover, there was no singularity algorithm described in that work. Finally, experiments with a physical manipulator were made to evaluate the system effectiveness.

1.2. *Contributions and organization of this work*

Therefore, in this work we wish to improve on two specific points on early works using trajectory description and fully actuated serial-link robots. First and foremost, robot control while considering a trocar point with an arbitrary motion on the workspace, with small RCM and trajectory tracking error (TE). The second contribution is to avoid task-space singularities while performing the necessary task.

We do so by providing an end effector trajectory capable of simultaneously placing the tool at the desired position while considering the trocar point constraints. The continuous trajectory description also allows compensation for arbitrary motion of the trocar point. Note that the proposed framework focuses on properly controlling the robot under a moving trocar point. The physiological motion model itself should be identified online or offline through other means [2–4, 16], which are complimentary to this work. Experiments with a physical robot using artificial trajectories show that the framework allows the control of laparoscopic instruments with near zero RCM error. Moreover, preliminary experiments with 13 users with no medical experience show that the system successfully operates under unpredictable user inputs.

This paper is organized as follows. Section 2 provides a background on kinematic control strategies to introduce some concepts for the unfamiliar reader and to motivate some of our design choices in the context of a laparoscopic application. In Sec. 3, we present the main contributions of our research, introducing the developed control system and its main modules. Then, in Sec. 4 the experimental setup used for validation of the technique was presented followed by the main experimental results in Sec. 5. Finally, in Sec. 6 we present our final considerations and possible future work.

2. Background on Manipulator Kinematic Control

Considering that laparoscopic surgery is described by slow acceleration and precise movements, kinematic control is a feasible approach used by many of the aforementioned related works. With those suppositions in mind, one merit of kinematic control is that it removes a layer of complexity related to robot dynamics, without hindering precision. Another merit is that different robotic manipulator geometries are embedded into the straightforward Denavit–Hartenberg parameters, allowing geometry change without the need of modifying the underlying control law.

While using kinematic control, important design decisions are motivated by the peculiar aspects of serial-link robotic manipulator geometry. Given the broader scope of this work with respect to earlier works, those aspects need to be thoroughly explained. Therefore, in this section we begin by briefly motivating our choice for dual quaternions for rigid body motion description. Afterwards, we introduce kinematic control using dual quaternions in order to elucidate some of the points that

Table 1. Mathematical notation.

Symbol	Meaning	Example
Bold capital	Matrix	$\mathbf{K}$
Bold lower-case	Quaternion	$\mathbf{r}$
Underlined bold lower-case	Dual Quaternion	$\underline{\mathbf{q}}$
Lower-case hat	Unit vectors	$\hat{\imath}$, $\hat{\jmath}$, $\hat{k}$
Lower-case	Natural numbers	n, i
	Vectors	θ, s
	Real numbers	α, β, γ, z, t, s

require most care when using serial-link manipulators: redundancy and workspace singularities. Please refer to Table 1 for the notation adopted in this work.

A proper choice of rigid body motion description provides a strong foundation on which we can develop robot control laws. For instance, the unit dual quaternion framework to represent rigid motions does not have the singularity issues of other minimal representations [17, 18]. Among its advantages, the dual quaternion representation has a straightforward way to obtain geometric parameters, resulting in a more intuitive derivation of controllers and path planners. Moreover, the dual quaternion framework is a compact mathematical tool that presents many advantages in the point of view of feedback control. The definition of both position and orientation in a common vector simplifies the design of the controller and avoids the computation of complex transformations to obtain the orientation error.

With respect to manipulator control, two mappings are necessary to achieve feedback control. Suppose we wish to control a n-DOF serial-link robotic manipulator. The first necessary mapping is between the manipulator joint positions, $\theta(t) \in \mathbb{R}^{n\times 1}$, and the end-effector pose, represented by the dual quaternion $\underline{\mathbf{x}}_{\mathrm{m}}$. That is, given a set of joint positions, obtain the position and orientation of the end-effector. Such mapping can be obtained for any serial-link geometry by using its Denavit–Hartenberg parameters through successive multiplications in dual quaternion space [17]. The second necessary mapping is the inverse kinematics. That is, how to map end effector poses, $\underline{\mathbf{x}}_{\mathrm{m}}$, to manipulator postures,[b] $\theta(t)$. However, a closed-form solution for the inverse kinematics does not exist for the general case, both due to the nonlinearity induced by the manipulator geometry and to problems related to the infinite solutions when mapping a higher dimensionality space, when $n > 6$, to SE(3).

In order to overcome difficulties in obtaining the inverse kinematics model of arbitrary serial-link geometries, it is more convenient to use the mapping between the manipulator joint velocities $\dot{\theta}(t) \in \mathbb{R}^{n\times 1}$ and its generalized end-effector velocity $\operatorname{vec} \dot{\underline{\mathbf{x}}}_{\mathrm{m}}(t) \in \mathbb{R}^{8\times 1}$ given by

$$\operatorname{vec} \dot{\underline{\mathbf{x}}}_{\mathrm{m}}(t) = \mathbf{J}\dot{\theta}(t), \tag{1}$$

where $\mathbf{J} \in \mathbb{R}^{8\times n}$ is called the manipulator analytical Jacobian [19] and depends on the current robot posture. Now, instead of an overly complicated function to invert, we have the linear mapping given by (1). Inverse kinematics can then be obtained by inverting the differential kinematics (1). For instance, it is possible to exponentially reduce the error between the current pose $\underline{\mathbf{x}}_{\mathrm{m}}(t)$ and a constant desired pose $\underline{\mathbf{x}}_d$ by using the closed-loop controller given by

$$\dot{\theta}(t) = \mathbf{K}\mathbf{J}^{\dagger}\operatorname{vec}\left(\underline{\mathbf{x}}_{\mathrm{m}}(t) - \underline{\mathbf{x}}_d\right), \tag{2}$$

with a positive definite $\mathbf{K}$ that affects the rate of convergence and $\mathbf{J}^{\dagger}$ being the *singular value decomposition* (SVD) inverse of the nonsingular $\mathbf{J}$.

Some research has been done in devising controllers directly in dual quaternion space in order to improve the performance of (2). There are results related to cooperative manipulation frameworks [17], and optimum H_∞ control design [18]. One very interesting result of [18] is the introduction of a transformation invariant error. The main idea of such error is to use the dual quaternion conjugation (which can be seen as a *pose difference* operator) instead of a dual quaternion subtraction. Modifying (2) to use such error results in

$$\dot{\theta}(t) = \mathbf{K}\mathbf{N}^{\dagger}\operatorname{vec}\left(1 - \underline{\mathbf{x}}_{\mathrm{m}}^{*}(t)\underline{\mathbf{x}}_d\right) \triangleq \mathbf{K}\mathbf{N}^{\dagger}\operatorname{vec}\underline{\mathbf{e}} \tag{3}$$

in which $\mathbf{N} = \bar{\mathbf{H}}(\underline{\mathbf{x}}_d)\mathbf{C}_8\mathbf{J}$, $\bar{\mathbf{H}}(\underline{\mathbf{x}}_d)$ is the Hamilton operator [20] of $\underline{\mathbf{x}}_d$, $\mathbf{C}_8 = \operatorname{diag}(1, -1, -1, -1, 1, -1, -1, -1)$, and $\underline{\mathbf{e}} \triangleq 1 - \underline{\mathbf{x}}_{\mathrm{m}}^{*}(t)\underline{\mathbf{x}}_d$ is the transformation invariant dual quaternion error. It is important to notice that (3) only considers set-point tasks, and not continuous trajectory tracking. We address continuous trajectory tracking in Sec. 3.4 by adding a feedforward term.

A posture is considered redundant whenever the rank of $\mathbf{J}$ is larger than the task space DOF at that posture. For instance, since our task space is SE(3), a manipulator may have redundant postures whenever $n > 6$. The effect of using the SVD with a redundant $\mathbf{J}$ is the local optimization of the joint velocities in a minimum square sense. Redundancy can also be used to other purposes, such as task space singularity avoidance.

Task space singularities refer to unachievable poses on the workspace, and also have a close connection with the rank of $\mathbf{J}$. The set of the singular postures of a manipulator with a SE(3) task-space can be defined as $\theta_s = \{\theta \,|\, \operatorname{rank}(\mathbf{J}) < 6\}$. The definition is straightforward since whenever a robot is unable to perform an arbitrary motion in space, the linear mapping (1) should (and does) lose rank. For instance, some of the singular configurations are those in which the manipulator is fully stretched, and can indeed not move further out of its own workspace. Other singular configurations, particularly in

[b] A manipulator posture at time t is the configuration of the manipulator given by its joint positions, i.e. $\theta(t)$.

redundant robots, can be rather unintuitive and exist inside the manipulators workspace when two or more DOF align. Those are visible in the Jacobian rank, but they are a mechanical characteristic and not exclusive to this form of control. In fact, the dexterity of a manipulator — i.e. how easily the robot can move to an arbitrary direction given its current posture — can be defined with respect to the singular values of its Jacobian, which also can be used to measure closeness to singularities [21]. That measure can be useful since the problem is not usually the singularity itself, rather only the mathematical manifestation of a mechanical phenomenon. The problem instead lies in the neighborhood of a task-space singularity, since **J** (and thus **N**) becomes ill conditioned, which results in high values for the joint velocities when the SVD inverse is used.

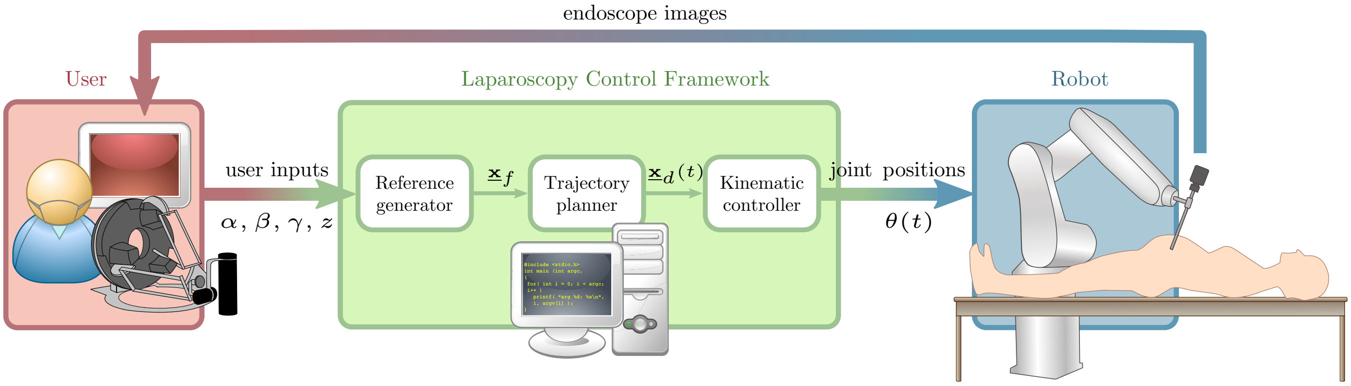
In this light, damped pseudoinverses have been proposed in earlier literature to avoid the velocity peak in the close vicinity of singularities [22]. However, the main side-effect is that the damped inverse is an artificially disturbed version of the real system model, therefore the inversion is imprecise. There are several tweaks that one can apply to the damped pseudoinverse to increase its performance in particular cases. However, if the manipulator is allowed to get close to a singularity, no control law can accurately compensate for this mechanical issue without an associated imprecision for an arbitrary direction if the rank of **J** is less than the task space DOFs. Since such errors cannot be tolerated in medical settings, the most reasonable way to cope with singularities is to avoid getting dangerously close. Whenever a singularity cannot in fact be actively avoided with any technique, manual rearrangement is necessary.

In order to avoid manual rearrangement, a dexterous task space should be found for nonredundant manipulators, or singularity evasion techniques should be used for redundant manipulators. Some works have studied how to optimize the task space for nonredundant manipulators, by offline optimization of measures such as manipulability [23], optimization the global isotropy index [24], or even laparoscopy-specific dexterity optimization [25]. However, due to the ambiguous pairing between θ and $\underline{\mathbf{x}}_{\mathrm{m}}$ when using a redundant manipulator, such measurements cannot be directly used. One solution is to exploit the manipulator redundancy to avoid singularities, by using the nullspace of **J** [26]. That causes the manipulator to use inner motions to avoid singularities, explicitly not disturbing the end effector motion. Note that even though nullspace projection will aid in avoiding singularities, it is not mathematically guaranteed to do so under all circumstances, therefore manual rearrangement may still be necessary in rare cases.

In this work, we use the SVD inversion due to its precision, and address singularity evasion by using an algorithm to locally optimize the condition number of the Jacobian as discussed in Sec. 3.6.

3. The Laparoscopy Controller

With the preceding discussion in mind, in this section we describe the proposed laparoscopy control framework. The overall organization of the system is as described in Fig. 2. In the bird's eye view, we have a surgeon that operates a master device, also having access to image feedback in order to perform the medical procedure. The proposed control system provides the link between surgeon commands and robot motion, and is further divided into three parts. The first component is the reference generator discussed in the following section, where tool control and camera control references are translated from the user interface commands to task space variables. The second component is the trajectory planner described in Sec. 3.2, that receives the desired tool-tip position and computes the constrained trajectory. The last component is the kinematic controller described in Sec. 3.4, which is derived from (3) and modified to account for a time-varying reference trajectory. The singularity evasion component of the controller is discussed in Sec. 3.6.



```
#include <stdio.h>
int main (int argc,
{
 for( int i = 0; i < argc;
 i++ )
   printf( "arg %d: %s\n",
   i, argv[i] );
}
```

Fig. 2. Laparoscopy control framework overview. First, it calculates the end effector final pose $\underline{\mathbf{x}}_f$ from user inputs. Then, it interpolates a trajectory $\underline{\mathbf{x}}_d(t)$ from the initial to the final pose. Finally, the manipulator joint trajectories are derived.

3.1. *Reference generation*

The system provides an user interface to generate four independent control signals $\alpha(t)$, $\beta(t)$, $\gamma(t)$, and $z(t)$ with $\alpha,\ \beta,\ \gamma,\ z(0) = 0$ (see Fig. 1). Each one will control a DOF of the laparoscopic instrument. The trocar point position can be obtained by using estimation techniques such as [2–4]. The result of such estimation is the trocar point position, $\underline{\mathbf{x}}_{\text{rto}}$. For dual quaternion operations and definitions, please refer to Appendix A. More information on dual quaternion algebra can be found in [27].

The trocar position is closely intertwined with the proposed technique. The basic concept is that, using the trocar as the reference frame, the trocar constraint is well described into four distinct DOF. Therefore, we can make use of that uncoupled definition to describe the end-effector motion with respect to the trocar point in the most intuitive manner for the task at hand.

For instance, in order to control a camera, the desired end effector pose can be chosen as

$$\underline{\mathbf{x}}_f(\alpha,\beta,\gamma,z) = \underline{\mathbf{x}}_{\text{tro}}\mathbf{r}\underline{\mathbf{t}}_z, \tag{4}$$

with $\mathbf{r} = \mathbf{r}_\alpha\mathbf{r}_\beta\mathbf{r}_\gamma$ and

$$\begin{aligned}\mathbf{r}_\alpha &= \cos\frac{\alpha}{2} + \hat{\imath}\sin\frac{\alpha}{2}, \quad \mathbf{r}_\beta = \cos\frac{\beta}{2} + \hat{\jmath}\sin\frac{\beta}{2},\\ \mathbf{r}_\gamma &= \cos\frac{\gamma}{2} + \hat{k}\sin\frac{\gamma}{2}, \quad \underline{\mathbf{t}}_z = 1 + \frac{1}{2}\epsilon\hat{k}z;\end{aligned}$$

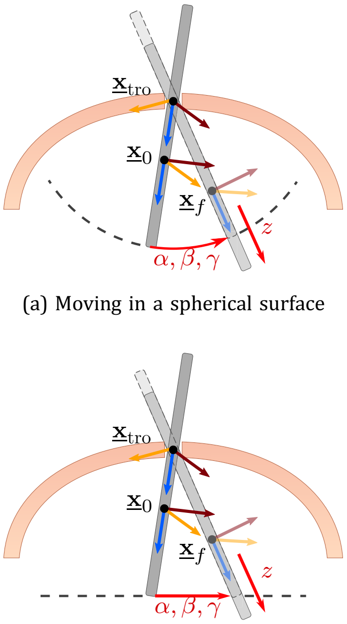
enabling the surgeon to move the endoscope DOFs independently, similar to manually handling the instrument. Note that in this case $\mathbf{r}$ traces a spherical surface centered at the trocar point, while $\underline{\mathbf{t}}_z$ selects the spherical surface radius, allowing full translation description of the workspace while respecting the trocar point constraints (see Fig. (3(a)).

In order to control a tool, tracing a plane may be more intuitive than tracing a sphere. Therefore we may choose a different translation term to (4) given by

$$\underline{\mathbf{t}}_z = 1 + \frac{1}{2}\epsilon\hat{k}z_d, \quad z_d = \frac{z}{\operatorname{vec}\mathbf{a}.\operatorname{vec}\mathbf{b}}; \tag{5}$$

with $\mathbf{a} = \hat{k}$, $\mathbf{b} = \mathbf{r}\mathbf{a}\mathbf{r}^*$, and . being the dot product. This results in tracing a plane surface with $\mathbf{r}$, while selecting the plane depth with $\underline{\mathbf{t}}_z$, as shown in Fig. 3(b).

Note that since the base reference frame is $\underline{\mathbf{x}}_{\text{tro}}$ and only motion in the allowed DOFs is made, references obtained from (4) and (5) are trocar point safe.


$\underline{\mathbf{x}}_{\text{tro}}$
$\underline{\mathbf{x}}_0$
$\underline{\mathbf{x}}_f$
z
α,β,γ
$\underline{\mathbf{x}}_{\text{tro}}$
$\underline{\mathbf{x}}_0$
$\underline{\mathbf{x}}_f$
z
α,β,γ

(a) Moving in a spherical surface

(b) Moving in a plane

Fig. 3. User input references and its effects in two different possibilities. For three given reference angles s_α, s_β and s_γ, the orientation of the final desired pose $\underline{\mathbf{x}}_f$ will be the same, while a given s_z results in different insertion depths depending on the chosen reference strategy.

3.2. *Trajectory planning for a static trocar point*

Every reference pose, $\underline{\mathbf{x}}_f$, is constrained by the trocar point whenever we obtain it from (4). The objective of this Section is to find the constrained trajectory that connects the current end effector pose at time of planning, i.e. when $\underline{\mathbf{x}}_{\text{m}}$ equals some $\underline{\mathbf{x}}_0$, with the reference pose, $\underline{\mathbf{x}}_f$. Such trajectory should be recalculated online whenever $\underline{\mathbf{x}}_f$ changes. In this section, we approach the problem considering a static trocar point, and the moving trocar point is described in the following section.

With that in mind, suppose the laparoscopic tool is at an initial pose $\underline{\mathbf{x}}_0$ and has to reach a feasible final constrained pose $\underline{\mathbf{x}}_f$, e.g. resultant of (4). To properly find the desired trajectory connecting both poses, we find the constrained description of $\underline{\mathbf{x}}_0$ and $\underline{\mathbf{x}}_f$; and next connect them with a zero RCM error trajectory by interpolating their relative rotation and translation.

In order to do so, we first obtain the constrained description of $\underline{\mathbf{x}}_0$: the rotation $\mathbf{r}^0_{\text{tro}}$ and the constrained translation $\underline{\mathbf{t}}^0_{\text{tro}}$ so that $\underline{\mathbf{x}}_{\text{tro}}\mathbf{r}^0_{\text{tro}}\underline{\mathbf{t}}^0_{\text{tro}}$ equals $\underline{\mathbf{x}}_0$. For that purpose, we note that

$$\underline{\mathbf{x}}_0 = \underline{\mathbf{x}}_{\text{tro}}\underline{\mathbf{x}}^0_{\text{tro}}, \tag{6}$$

for some $\underline{\mathbf{x}}^0_{\text{tro}}$ that describes the motion from $\underline{\mathbf{x}}_{\text{tro}}$ to $\underline{\mathbf{x}}_0$. From (6), we can obtain $\mathbf{r}^0_{\text{tro}}$ by noticing that

$$\begin{aligned}\underline{\mathbf{x}}^0_{\text{tro}} &= \underline{\mathbf{x}}^*_{\text{tro}}\underline{\mathbf{x}}_0\\ \therefore \mathbf{r}^0_{\text{tro}} &= \mathcal{P}(\underline{\mathbf{x}}^*_{\text{tro}}\underline{\mathbf{x}}_0).\end{aligned} \tag{7}$$

With (7), we can obtain the constrained translation of $\underline{\mathbf{x}}_0$ as given by

$$\mathbf{t}^0_{\text{tro}} = \text{translation}\left((\underline{\mathbf{x}}_{\text{tro}}\mathbf{r}^0_{\text{tro}})*\underline{\mathbf{x}}_0\right).$$

Due to the RCM constraint, $\mathbf{t}^0_{\text{tro}}$ can only be a translation in the z-axis. However, unmodeled disturbances may cause it to be some $\mathbf{t} = t_x\hat{\imath} + t_y\hat{\jmath} + t_z\hat{k}$, with $t_x \neq 0$ and $t_y \neq 0$. Therefore, we constrain $t_x = t_y = 0$ to obtain

$$\mathbf{t}^0_{\text{tro}} = t_z\hat{k} \Rightarrow \underline{\mathbf{t}}^0_{\text{tro}} = 1 + \frac{1}{2}\epsilon t_z\hat{k}.$$

Now define $\underline{\mathbf{x}}'_0 = \underline{\mathbf{x}}_{\text{tro}}\mathbf{r}^0_{\text{tro}}\underline{\mathbf{t}}^0_{\text{tro}}$, which perfectly respects the trocar point constraint. Using the same procedure, we can obtain

$$\underline{\mathbf{x}}_f = \underline{\mathbf{x}}_{\text{tro}}\mathbf{r}^f_{\text{tro}}\underline{\mathbf{t}}^f_{\text{tro}}.$$

Now that the constrained descriptions of both $\underline{\mathbf{x}}'_0$ and $\underline{\mathbf{x}}_f$ were found, we search for the constrained trajectory between them. Consider that the planning function is called at a time $t = t_i$. We then define a terminal time t_{max} for the trajectory, e.g. $t_{\text{max}} = ||\text{vec}\ (1 - \underline{\mathbf{x}}^*_{\text{m}}(t)\underline{\mathbf{x}}_d)/s$ (see (3)), with $s \in \mathbb{R}^+ - \{0\}$. The parameter s is an intuitive design parameter that can be regarded as the trajectory speed, since $||\text{vec}\ (1 - \underline{\mathbf{x}}^*_{\text{m}}(t)\underline{\mathbf{x}}_d)||$ is a distance between the current pose and the final pose. Using t_i and t_{max}, we define the trajectory planner time, t_p, as

$$\begin{cases} t_p = t - t_i, & t \leq t_{\text{max}} + t_i \\ t_p = t_{\text{max}}, & t > t_{\text{max}} + t_i \end{cases}, \tag{8}$$

therefore t_p increases at the same pace as t in the interval $[0, t_{\text{max}}]$, being saturated at t_{max}. This definition for the planner time will be used so that the planned trajectory begins at $\underline{\mathbf{x}}'_0$ with $t_p = 0$ and ends at $\underline{\mathbf{x}}_f$ with $t_p = t_{\text{max}}$.

With the planner time defined, we can find the necessary time-varying rotation between the current and the target pose, $\mathbf{r}_p(t_p)$. Using the quaternion log operator, we obtain

$$\log \mathbf{r}_p = \frac{t_p}{t_{\text{max}}} \log\left((\mathbf{r}^0_{\text{tro}})^*\mathbf{r}^f_{\text{tro}}\right).$$

$$\therefore \mathbf{r}_p(t_p) = \exp\left(\frac{t_p}{t_{\text{max}}} \log\left((\mathbf{r}^0_{\text{tro}})^*(\mathbf{r}^f_{\text{tro}}\right)\right). \tag{9}$$

The rotation given by (9) can be more easily understood by noticing that if we solve $\mathbf{r}_a = (\mathbf{r}^0_{\text{tro}})^*\mathbf{r}^f_{\text{tro}}$, we obtain $\mathbf{r}_a = \cos\frac{\theta_a}{2} + n_a \sin\frac{\theta_a}{2}$ for some constant θ_a and n_a, therefore

$$\mathbf{r}_p(t_p) = \cos\frac{t_p\theta_a}{2t_{\text{max}}} + n_a \sin\frac{t_p\theta_a}{2t_{\text{max}}}.$$

This means that we have a fixed axis given by $\mathbf{n}_a$, with a linear increasing rotation angle from 0 to θ_a. In a similar manner, the planner translation, $\underline{\mathbf{t}}_p$, is given by

$$\underline{\mathbf{t}}_p(t_p) = (t_p/t_{\text{max}})\left((\underline{\mathbf{t}}^0_{\text{tro}})^*\underline{\mathbf{t}}^f_{\text{tro}}\right). \tag{10}$$

Finally, we compose both (9) and (10) to obtain

$$\underline{\mathbf{x}}_d(t_p) = \underline{\mathbf{x}}_{\text{tro}}\mathbf{r}^0_{\text{tro}}\mathbf{r}_p(t_p)\underline{\mathbf{t}}^0_{\text{tro}}\underline{\mathbf{t}}_p(t_p), \tag{11}$$

note that $\underline{\mathbf{x}}_d(0) = \underline{\mathbf{x}}'_0$, $\underline{\mathbf{x}}_d(t_{\text{max}}) = \underline{\mathbf{x}}_f$, all possible intermediary points preserve the trocar point constraint, and both translation and rotation simultaneously converge to their desired values; as shown in Fig. 4. Concerning implementation, the trajectory does not need to be entirely computed beforehand. At each step, the controlling algorithm can obtain the next step using (11). Such calculation requires few operations, and a change in the velocity parameter s does not alter the computational cost.

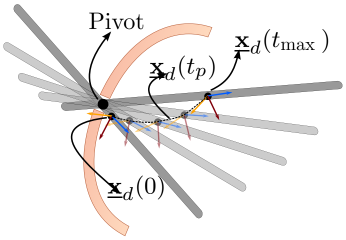

Pivot
$\underline{\mathbf{x}}_d(t_{\text{max}})$
$\underline{\mathbf{x}}_d(t_p)$
$\underline{\mathbf{x}}_d(0)$

Fig. 4. Constrained trajectory $\underline{\mathbf{x}}_d(t_p)$. All points in the trajectory have zero RCM error.

In this section, we examined how to generate a trajectory between two points considering a static trocar point constraints. In the next section, we extend this solution to dynamic trocar positions.

3.3. *Trajectory generation for a moving trocar point*

Although one of the interesting aspects of software generated RCM is the possibility of compensating for physiological motion, this is an issue that lies largely unaddressed in related works. In this section, we discuss how to compensate for trocar point motion using the proposed trajectory planner.

By using some trocar point motion estimation framework [2–4, 28], we obtain the desired RCM position as the time-varying function $\underline{\mathbf{x}}_{\text{tro}}(t)$. We then use the same procedure shown in the last section to define the desired trajectory $\underline{\mathbf{x}}_d$. Notice that now the trajectory exists for all t, while t_p is defined in (8). Therefore

$$\underline{\mathbf{x}}_d(t, t_p) = \underline{\mathbf{x}}_{\text{tro}}(t)\mathbf{r}^0_{\text{tro}}(t)\mathbf{r}_p(t_p)\underline{\mathbf{t}}^0_{\text{tro}}(t)\underline{\mathbf{t}}_p(t_p). \tag{12}$$

Even if the surgeon no longer interacts with the system, i.e. there are no function calls to update t_i, when $t \geq t_{\text{max}}$ in (12) the trajectory planner will compensate for RCM motion as shown in Fig. 5. As the trocar point constrains

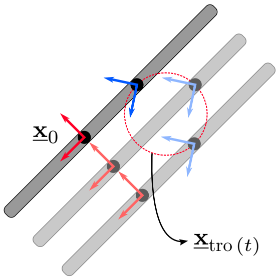

$\underline{\mathbf{x}}_0$
$\underline{\mathbf{x}}_{\text{tro}}(t)$

Fig. 5. The effect on $\underline{\mathbf{x}}_0$ with a time varying $\underline{\mathbf{x}}_{\text{tro}}$.

the motion, the orientation and translation of the tool in the workspace cannot be simultaneously maintained constant in relation to the patient while the RCM is moving. Such issue is inherent to the trocar point constraint and not to the proposed system.

3.4. *Kinematic controller*

The control law (3) considers a point-to-point control of the end effector. This is not the case in our system, since we have a continuous trajectory description. When using a continuous trajectory, the controller needs a feedforward term to be able to predict the perturbation caused by the dynamics of the trajectory. In order to find the feedforward term, we being with the transformation invariant dual quaternion error defined as [18]

$$\underline{\mathbf{e}} = 1 - \underline{\mathbf{x}}_{\mathrm{m}}^{*}\underline{\mathbf{x}}_{d},$$

and differentiate it in time to find

$$\dot{\underline{\mathbf{e}}} = -\dot{\underline{\mathbf{x}}}_{\mathrm{m}}^{*}\underline{\mathbf{x}}_{d} - \underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}.$$

Manipulators are usually joint velocity controlled, therefore $\dot{\theta}$ should explicitly appear on the control law. The first step to do so is to use the vec operator on both sides to find

$$\operatorname{vec}\dot{\underline{\mathbf{e}}} = -\bar{\mathbf{H}}(\underline{\mathbf{x}}_{d})\mathbf{C}_{8}\operatorname{vec}\dot{\underline{\mathbf{x}}}_{\mathrm{m}} - \operatorname{vec}\,\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}. \tag{13}$$

By substituting the differential kinematic equation (1) in (13), we find

$$\begin{aligned} \operatorname{vec}\dot{\underline{\mathbf{e}}} &= -\bar{\mathbf{H}}(\underline{\mathbf{x}}_{d})\mathbf{C}_{8}\mathbf{J}\dot{\theta} - \operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}, \\ \operatorname{vec}\dot{\underline{\mathbf{e}}} &= -\mathbf{N}\dot{\theta} - \operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}. \end{aligned} \tag{14}$$

In (14) we can see that the term $-\operatorname{vec}\,\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}$ is a measurable disturbance in the error dynamics induced by the time-varying desired trajectory. This disturbance is the term we want to eliminate. Since $\dot{\theta}$ is the control signal and $-\operatorname{vec}\,\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}$ depends only on current and desired end effector poses, which are know variables, we can perform a feedback linearization by choosing

$$\dot{\theta} = \mathbf{N}^{\dagger}(\mathbf{K}\operatorname{vec}\underline{\mathbf{e}} - \operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}). \tag{15}$$

The effect of using such control signal can be seen by substituting (15) in (14) to obtain

$$\begin{aligned} &\operatorname{vec}\dot{\underline{\mathbf{e}}} = -\mathbf{N}\mathbf{N}^{\dagger}(\mathbf{K}\operatorname{vec}\underline{\mathbf{e}} - \operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}) - \operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}. \\ &\therefore \operatorname{vec}\dot{\underline{\mathbf{e}}} = -\mathbf{K}\operatorname{vec}\underline{\mathbf{e}}, \end{aligned} \tag{16}$$

which means that this choice of control signal causes the error to exponentially converge to zero, for a positive definite $\mathbf{K}$. Since the generated trajectory between the current and desired points will be completely known, the feedforward term, vec $\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}$, is also known for all times. In the next section, we describe how to find the derivative of the desired trajectory, $\dot{\underline{\mathbf{x}}}_{d}$.

3.5. *Trajectory derivative*

The kinematic controller (15) has a feedforward term given by $\operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}$ in order to compensate for time-varying reference induced errors. In consequence, we require the time derivative of $\underline{\mathbf{x}}_{d}$. For a constant $\underline{\mathbf{x}}_{\mathrm{tro}}$, the derivative of (11) is

$$\dot{\underline{\mathbf{x}}}_{d} = \underline{\mathbf{x}}_{\mathrm{tro}}\mathbf{r}_{\mathrm{tro}}^{0}\dot{\mathbf{r}}_{p}\underline{\mathbf{t}}_{\mathrm{tro}}^{0}\underline{\mathbf{t}}_{p} + \underline{\mathbf{x}}_{\mathrm{tro}}\mathbf{r}_{\mathrm{tro}}^{0}\mathbf{r}_{p}\underline{\mathbf{t}}_{\mathrm{tro}}^{0}\dot{\underline{\mathbf{t}}}_{p},$$

with

$$\dot{\mathbf{r}}_{p} = \frac{\theta_{a}}{2t_{\max}}\left(-\sin\left(\frac{t_{p}\theta_{a}}{2t_{\max}}\right) + \mathbf{n}_{a}\cos\left(\frac{t_{p}\theta_{a}}{2t_{\max}}\right)\right)$$

and

$$\dot{\underline{\mathbf{t}}}_{p} = \frac{1}{2t_{\max}}\epsilon\hat{k}(t_{z}^{f} - t_{z}^{i}),$$

for a $\underline{\mathbf{t}}_{\mathrm{tro}}^{0} = 1 + 0.5\epsilon\hat{k}t_{z}^{f}$.

Whenever $\underline{\mathbf{x}}_{\mathrm{tro}}$ is time-varying, its derivative and the derivatives of $\mathbf{r}_{\mathrm{tro}}^{0}(t)$ and $\underline{\mathbf{t}}_{\mathrm{tro}}^{0}(t)$ should be considered. Concerning computational implementations, note that (12) is computationally inexpensive, therefore numerical differentiation is a viable option.

3.6. *Singularity Evasion*

Recall from Sec. 2 that whenever a manipulator has more than 6 DOF, it will present redundant postures when its task space is SE(3). This means that offline optimization measures of the manipulator dexterity are not directly applicable, which in turn means that the manipulator may have unpredictable singular postures inside its operating workspace. The singular posture is not the only problem. In fact, a region around that posture may also result in imprecisions when moving the robot.

When using a redundant robotic manipulator, instead of the unfeasible offline global optimizations, we can use online local optimization by exploiting the redundant manipulator extra DOFs. This is usually performed by projecting a secondary objective function of the joint variables $c(\theta)$ in the nullspace of the primary task [26]. We may do so by choosing

$$\dot{\theta} = \mathbf{N}^{\dagger}(\mathbf{K}\operatorname{vec}\underline{\mathbf{e}} - \operatorname{vec}\underline{\mathbf{x}}_{\mathrm{m}}^{*}\dot{\underline{\mathbf{x}}}_{d}) + \mathbf{P}\mathbf{K}_{c}\mathbf{J}_{c}(c(\theta) - c_{d}), \tag{17}$$

where $\mathbf{P} = (\mathbf{I} - \mathbf{N}^{\dagger}\mathbf{N})$ is a nullspace projector in which $\mathbf{I}$ is the identity matrix, $\mathbf{K}_{c}$ is a positive definite gain matrix for the secondary objective, and $\mathbf{J}_{c} = \partial c(\theta)/\partial\theta$ is the constraint Jacobian. Also, c_{d} is the desired value for the secondary objective function. By using (17), $c(\theta)$ converges to c_{d} as long as the secondary objective does not conflict with the first [22]. The added term $\mathbf{P}\mathbf{K}_{c}\mathbf{J}_{c}(c(\theta) - c_{d})$ generates inner motions and does not disturb the convergence of (14).

Since we want to use the robot extra DOF to avoid singularities, a good choice for the secondary control objective would be to optimize some sort of dexterity measurement, that is, a metric of how distant the robot pose is from singularities. A measurement of dexterity often used in manipulator control settings is the manipulability [23], which is not applicable to the dual quaternion case since $\det \mathbf{N}\mathbf{N}^T = \det \mathbf{J}\mathbf{J}^T = 0 \quad \forall \quad \theta$. Although computationally inexpensive, manipulability has been questioned for some time given that it might not be the best dexterity measurement [21], as the determinant of a matrix does not always reflect its conditioning. Since then, the computational power available in robotic applications has increased manifold, and it is currently feasible to directly obtain the conditioning number derivative using, for instance, finite differences.

To do so, we begin by defining the reciprocate of the condition number as

$$\kappa(\theta) = \frac{\sigma_6}{\sigma_1}$$

in which σ_1 is the largest singular value of $\mathbf{N}$, and σ_6 is the sixth — and smallest — nonzero singular value of $\mathbf{N}$ whenever it is not singular. Notice that $\max(\kappa) = 1$ for any possible matrix, and represents the best possible conditioning. Consider that each joint θ_i of the robot has a positioning precision δ_i, for $i = 1, \ldots, n$. By defining the selector vectors $s_i \in \mathbb{R}^{n\times 1}$ as

$$\begin{aligned} s_1 &= [1\ 0\ 0\ \cdots\ 0]^T, \\ s_2 &= [0\ 1\ 0\ \cdots\ 0]^T, \\ &\vdots \\ s_n &= [0\ 0\ 0\ \cdots\ 1]^T, \end{aligned}$$

we choose the partial derivatives of the reciprocate of the condition number with respect to each of the joint positions as

$$\frac{\partial \kappa(\theta)}{\partial(\theta_i)} \approx \frac{\kappa(\theta + s_i\delta_i) - \kappa(\theta - s_i\delta_i)}{\delta_i}, \qquad (18)$$

since the *real* θ_i may be any number between $(\theta_i - \delta_i, \theta_i + \delta_i)$. That is a reasonable definition if we suppose that the probability density function of θ is uniform over $(\theta_i - \delta_i, \theta_i + \delta_i)$. Therefore, for singularity evasion we choose

$$\mathbf{J}_c = \left[\frac{\partial(\kappa(\theta))}{\partial(\theta_1)} \cdots \frac{\partial(\kappa(\theta))}{\partial(\theta_n)}\right]^T, \qquad (19)$$

together with $c(\theta) = \kappa(\theta)$ and $c_d = 1$, given that $\kappa = 1$ reflects the best conditioning possible. Although $c_d = 1$ is unreachable for most systems, it is an objective that will — given no conflicts with the first objective — locally increase the conditioning of any $\mathbf{N}$ by using the manipulator nullspace.

4. Evaluation

4.1. *Experimental setup*

In order to evaluate the performance of the proposed control system, a series of experiments were devised. The experimental setup is as shown in Fig. 6 and further described as follows. A 6-DOF COMAU Smart Six Robot[c] was used with an endoscope attached to its last link using a custom 3D printed PLA connection. The selected workspace is a virtual $10\,\text{cm} \times 10\,\text{cm} \times 5\,\text{cm}$ prism inside the custom made acrylics chest phantom, which is reasonably large for a great number of minimally invasive procedures [25]. Whenever a user operates the system, they receive image feedback through the laptop monitor, while operating the robot using a Force Dimension Omega 7 haptics interface. The user cannot see the acrylics chest when seated, so the only source of information is the monitor itself. Finally, by relying on the high rigidity of an industrial-type robot, the ground-truth for the positioning measurement is the forward kinematics of the manipulator.

The reference generation equation used was the one described by (4), which is suitable for controlling an endoscope. The control law used was given by (17). However, since the experimental robot has only 6 DOF, the nullspace projection is dropped to become

$$\dot{\theta} = \mathbf{N}^{\dagger}(\mathbf{K}\text{vec}\ (1 - \underline{\mathbf{x}}_m^*\underline{\mathbf{x}}_d) - \text{vec}\ \underline{\mathbf{x}}_m^*\dot{\underline{\mathbf{x}}}_d).$$

Remember that $\mathbf{K}$ is a positive definite matrix, $\mathbf{N} = \bar{\mathbf{H}}(\underline{\mathbf{x}}_d)\mathbf{C}_8\mathbf{J}$, $\bar{\mathbf{H}}(\underline{\mathbf{x}}_d)$ is the Hamilton operator [20] of $\underline{\mathbf{x}}_d$, $\mathbf{C}_8 = \text{diag}(1, -1, -1, -1, 1, -1, -1, -1)$, $1 - \underline{\mathbf{x}}_{\text{m}}^*\underline{\mathbf{x}}_d$ is the transformation invariant dual quaternion error, and $\underline{\mathbf{x}}_d$ is obtained from the planner equation (11), while the derivative of $\underline{\mathbf{x}}_d$ is obtained through finite differences. The planner described in Section 3.2 is used with $t_{\max} = ||\text{vec}\ (1 - \underline{\mathbf{x}}_m^*\underline{\mathbf{x}}_d)||/s$, with s being the trajectory speed. The whole framework has two design parameters: the gain matrix $\mathbf{K}$ and the trajectory speed scalar s. The latter sets the trade-off between convergence rate to the trajectory and joint velocity increase; whereas the former concerns how fast the reference trajectory is, being inversely proportional to the trajectory time length.

The discrete time implementation of the kinematic control law and the planner were written using the DQ_robotics[d] library in C++. DQ_robotics has the necessary functions to perform the required dual quaternion algebraic operations, while also providing the kinematic control operations such as the forward kinematics model and Jacobian computation. There is a socket connection through which the commands are sent from the haptics interface to the robotic manipulator. Due to the limitations of the robot communication server, the both ways latency is around 150 ms, which is quite high.

[c] Note that even though the experimental robot has 6-DOF, redundant robots can also be used by applying the techniques discussed in Sec. 3.6.

[d] http://sourceforge.net/projects/dqrobotics/.

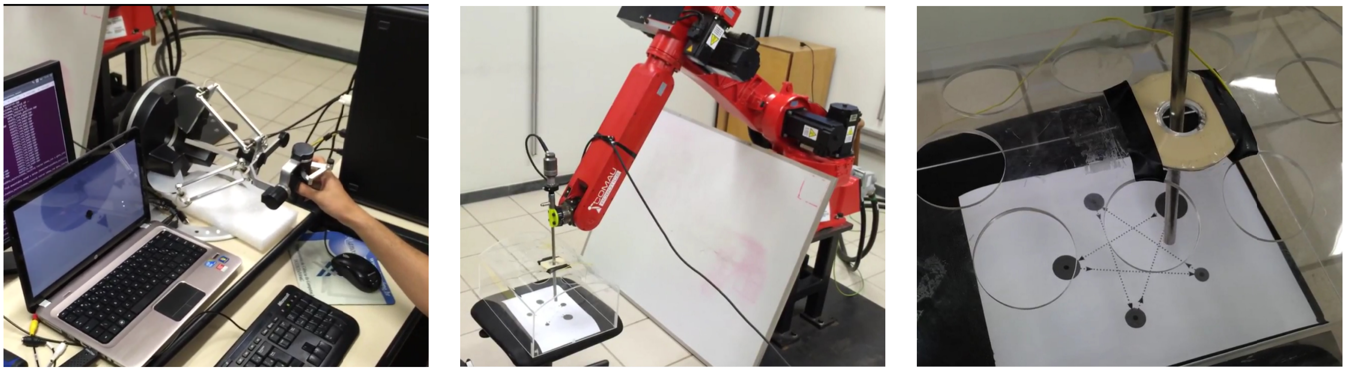


Fig. 6. Experimental setup overview for manual operation. The user commands the robot through the haptic interface, and obtains visual feedback using the monitor (*left*). The robot holds the endoscope that goes into the custom made acrylic box (*center*). A close-up view of the box and the reference path the user should follow (*right*).

4.2. *Experimental protocol*

In total three sets of experiments were devised in order to evaluate the system performance, two sets for artificial inputs and one for user generated inputs. Recall from the last section that the controller has only two design parameters, namely the gain matrix $\mathbf{K}$ and the trajectory speed s. With that in mind, the first set of artificial input experiments; using a static trocar point, were intended to visualize the effects of different choices of the two design parameters in the performance of the system. The second set of artificial input experiments intended to give light about the effects of a moving trocar point with different amplitudes and frequencies on the system performance. The last set regards the user controlled experiments, whose objective was to observe the behavior of the system under the unpredictability of user input. The parameters for this last experiment were constant for all users, chosen based on the performance of the system under artificial inputs.

The artificial inputs were two different trajectories, applied in both the static and dynamic trocar point scenarios. The first trajectory was the round-trip path connecting the two most distant points in the prismatic workspace, as shown in Fig. 7 (*left*). This experiment shows the performance of the system when generating the longest trajectory possible in the workspace. With that information, the parameter s can be tuned since you obtain the time it takes to make the longest motion possible inside the workspace. The second trajectory consisted of a sequence of way-points with varying 3D coordinates, in a star shaped motion as shown in Fig. 7 (*right*). This trajectory allowed a better understanding of the parameters under a larger workspace coverage. The static trocar point experiments were run first to find the effects of the two design parameters without the trocar point motion influence. The moving trocar point experiments intended to show the effects in the system performance of different values for the trajectory speed, s, and differing amplitudes and frequencies for the trocar point motion.

Considering the static trocar point, two sets of parameters were used. The trajectory speeds were $s \in \{0.025, 0.05, 0.1\}$, and the gain matrices are diagonal matrices with the same elements along its main diagonal $\mathbf{K} \in \{2.5, 5, 7.5\}$. All combinations were tested for the total of nine trials for a static trocar point. When considering a moving trocar point, extra scenarios required testing. As the effect of $\mathbf{K}$ is well understood both from literature and from the first experiment, all moving trocar point experiments were held with a constant $\mathbf{K} = 5$. The same set of trajectory velocities was tested, i.e. $s \in \{0.025, 0.05, 0.1\}$. In order to evaluate the effects of different trocar point motions, an artificial trocar point motion was chosen as

$$\underline{\mathbf{x}}_{\text{tro}}(t) = \underline{\mathbf{x}}_{\text{tro}}\left(1 + \frac{\epsilon}{2} a \sin(2\pi f t)\hat{k}\right), \qquad (20)$$

which is a sinusoidal motion with frequency f and amplitude of $2a$, in the inwards-outwards direction with respect to the trocar point. Notice that $\underline{\mathbf{x}}_{\text{tro}}(t)$ can be an arbitrarily complex model describing the trocar point

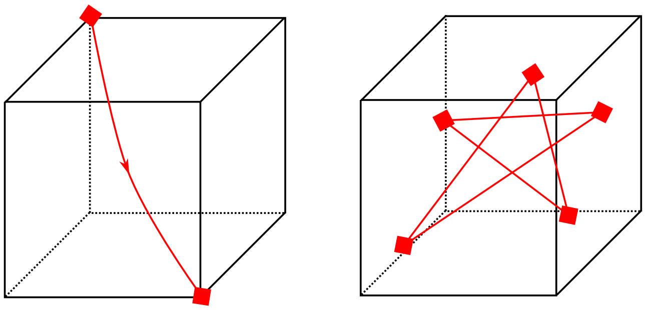


Fig. 7. The way-points used in the experiments involving predefined trajectories: round-trip through the surgical field (*left*) and star shaped (*right*).

motion. In practical scenarios the motion to be compensated should be well modeled and supervised. The experiment evaluated the robot control performance with different amplitudes and frequencies, i.e. (20) with $(a\,[\mathrm{cm}], f\,[\mathrm{hz}]) \in \{(1.0,\ 0.5),\ (2.0,\ 0.5),\ (1.0,\ 1.0)\}$. Those were chosen to represent considerably fast and high amplitude biological movements. By combining the different parameters, nine different trials were run for the moving trocar point.

Using the knowledge acquired in the first two experiments, 13 users with no prior medical experience voluntarily used the system in an endoscope control task. Referring to Fig. 6, the user controlled the system by using the haptics interface. The image feedback is shown through the PC monitor. The monitor had a marker to help centering the camera where the user would deem necessary. The users were requested to, at first, try following the reference star printed in the paper inside the acrylic chest. Afterwards, to freely move the haptics interface. The objective of this experiment was to show that with the appropriate parameters (e.g. $\mathbf{K} = 1.5$ and $s = 0.025$ were used for all users), the system can efficiently operate for a long time and the workspace can be freely explored using user generated commands without singularity and stability problems. The performance measure in this experiment were the RCM error, and the error between the end effector tip pose and the desired end effector tip pose.

Table 2. Static trocar point root mean square error (RCME) and root mean square translation error to planned trajectory (TE).

K	2.5	5	5	5	7.5
s	0.05	0.025	0.05	0.1	0.05
Surgical field round-trip					
RCME [10^{-2}mm]	5.13	0.84	2.80	31.6	2.31
TE [mm]	2	0.16	1.2	8.7	0.88
Star shaped					
RCME [10^{-2}mm]	7.54	1.06	5.14	33.8	3.89
TE [mm]	1.5	0.16	0.90	6.3	0.69

5. Results

5.1. *Static trocar point*

The *root mean square error* of the RCM (RCME) and the TE on both static trocar point experiments are listed in Table 2. The overall performance of the surgical fieldround-trip experiment can be seen in Fig. 8, whereas the performance of the star shaped motion can be seen in Fig. 9. The RCM error was kept below 1 mm for experiments in this set. The trajectory error is mostly a

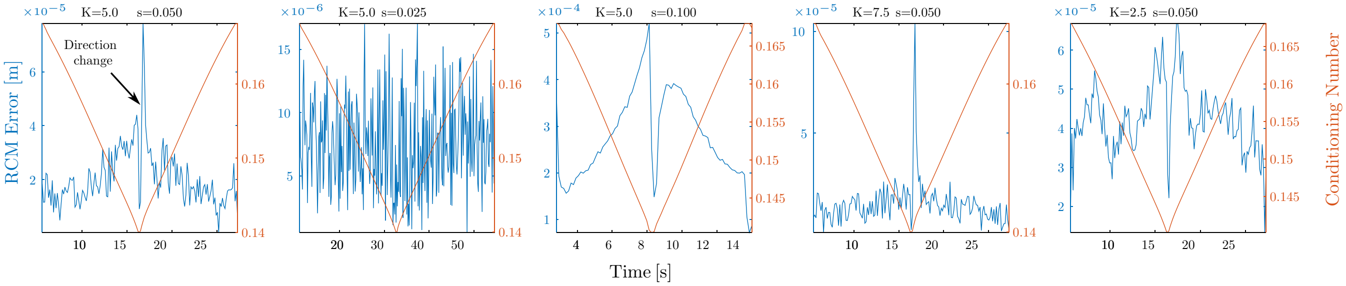


Fig. 8. Results for the surgical field round-trip motion with a static trocar point. RCM errors are below 1 mm for all cases. The peak in the center of most trajectories represents the worst case scenario, in which the robot has to reverse its velocity completely while in the local minimum of dexterity.

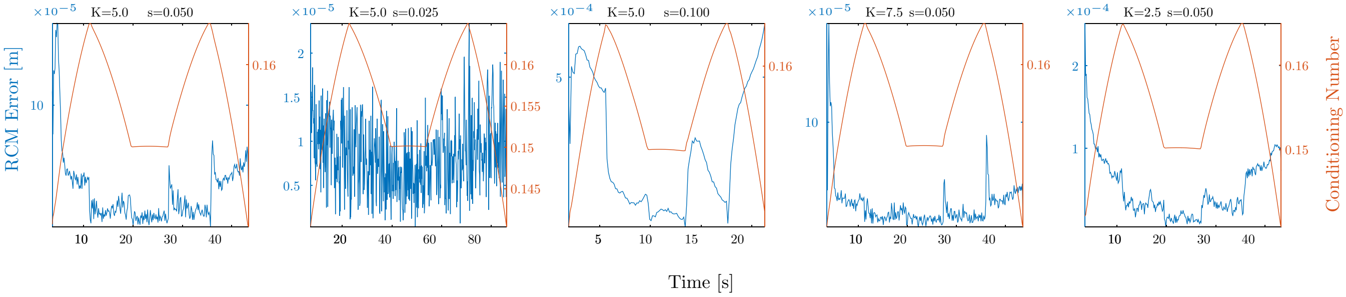


Fig. 9. Results for the star shaped motion with a moving trocar point. RCM error was maintained below 1 mm in all experiments. The peaks in the start shaped motion are less noticeable since the velocity direction change was less sharp. Since the peaks are shorter, the initial error due to the manipulator acceleration is more noticeable.

constant function, with its best value of under 0.2 mm at $\mathbf{K} = 5$ and $s = 0.025$. Moreover, the trajectory was smooth as evidenced by the robot motion and the endoscopic camera image stability. Such results demonstrate in a preliminary form the safety of the concept system.

From the results in Table 2, it is noticeable that the error is in a similar order of magnitude when using the same parameters for the star shaped and extremities experiments. In the case of the star shaped trajectory, the larger error can be explained by the prolonged motion inside a less dexterous workspace. With regard to the overall behavior (refer to Figs. 8 and 9), results show RCM errors under 1 mm for all experiments which are safe for laparoscopic procedures. The most noticeable source of error is the loss of dexterity amplifying the noise, which does not originate from the controlling algorithm itself. An optimized manipulator design for medical procedures, such as done in the DLR MIRO robot [29], may highly increase the overall performance.

Considering the surgical field round-trip case (Fig. 8), the peak in the center of most trajectories represents the worst case scenario, in which the robot has to reverse its velocity completely while in the local minimum of dexterity. The peaks in the star shaped motion (Fig. 9) are less noticeable since the velocity direction change was less sharp, and the manipulator stayed within a more dexterous workspace region. Given that the peaks in the star shaped motion are smaller, the initial error due to the manipulator acceleration is more noticeable. Regardless of peaks, the motion is well behaved and the endoscope shows a stable image, even when the trajectory is repeated several times. When the trajectory velocity was $s = 0.025$, the peaks were not discernible from other sources of noise, but the complete motion took more time to complete. The velocity parameter is inversely proportional to the time of completion, being a rather intuitive design parameter that can be optimized for the workspace volume of each procedure. Moreover, as expected, an increase in the value of $\mathbf{K}$ provides a smaller RMSE error, but at the expense of higher joint velocities and higher error peaks.

Table 3. Moving trocar point RCME and root mean square translation error to planned trajectory (TE).

f [Hz]	0.5	0.5	0.5	0.5	0.5	0.5	1	1	1
a [cm]	1	1	1	2	2	2	1	1	1
s	0.025	0.05	0.1	0.025	0.05	0.1	0.025	0.05	0.1
				Surgical field round-trip					
RCME $[10^{-1}\text{mm}]$	0.554	2.12	8.03	1.37	4.15	15	1.05	3.78	11
TE [mm]	0.14	1.1	8.9	0.15	1.1	9	0.15	1.1	8.9
				Star shaped					
RCME $[10^{-1}\text{mm}]$	0.6	1.94	5.74	0.58	3.7	1.19	0.94	3.54	7.60
TE [mm]	0.16	0.96	4.8	0.93	1	0.14	0.11	0.95	4.9

5.2. *Moving trocar point*

The RCME and the TE on both moving trocar point experiments are listed in Table 3. The overall performance of the surgical field round-trip experiment can be seen in Fig. 10, whereas the performance of the star shaped motion can be seen in Fig. 11. The RCM error

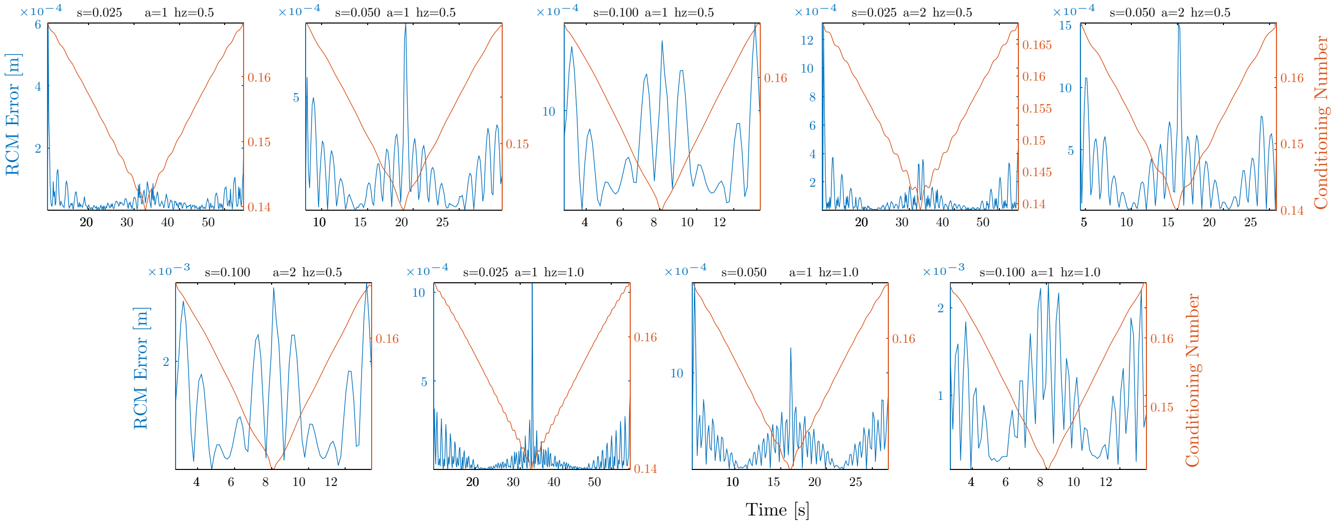


Fig. 10. Results for the surgical field round-trip motion with a moving trocar point. RCM error was within safe margins for all experiments. Note that the dominating error source now is the trocar point motion. Whenever the trocar point motion is opposite to the robot motion, a larger error can be observed. Besides two cases with $s = 0.1$, the RCM error was below 1 mm.

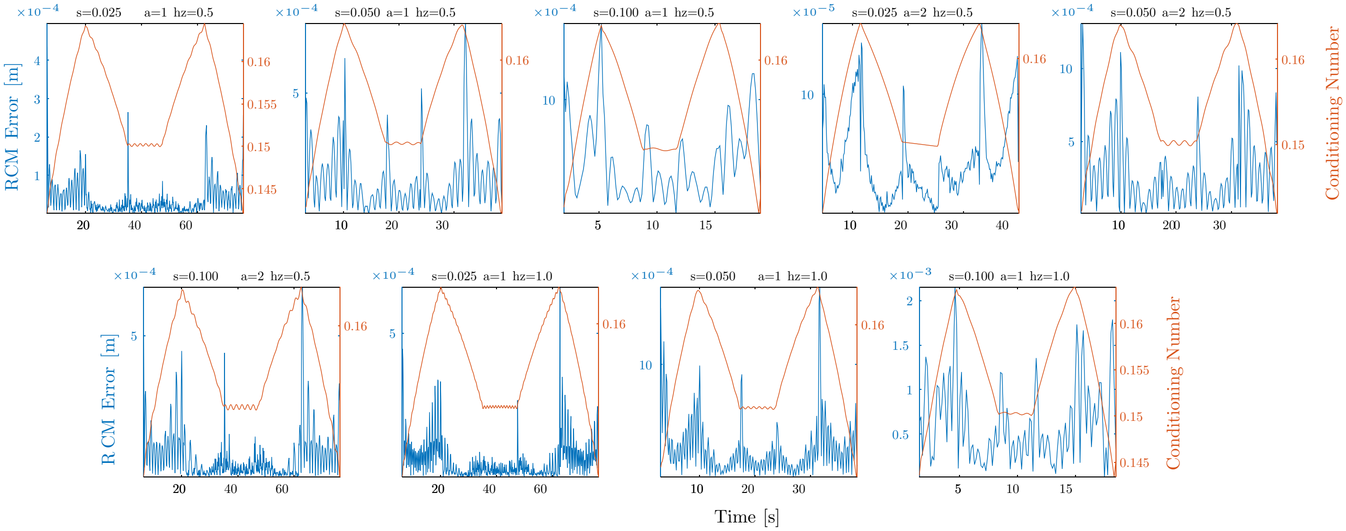


Fig. 11. Results for the star shaped motion with a moving trocar point. RCM error was within safe margins for all experiments. Note that the dominating error sources of error are the trocar point motion and the change in reference. Besides one case with $s = 0.1$, the RCM error was below 1 mm.

was kept mostly below 1 mm for experiments in this set, even with a high amplitude biological motion. The TE is under 0.16 mm when using the trajectory velocity of $s = 0.025$. Moreover, the trajectory was smooth as evidenced by the robot motion and the endoscopic camera image stability. Such results give an initial demonstration of the safety of the proposed framework while performing biological motion compensation.

Also in this case, the system demonstrated similar performance in both star shaped and surgical field round-trip experiments. The effects of the trocar point motion are noticeable on the RCME. More specifically, the RCME in which the trajectory velocity is $s = 0.1$ are specially high (≈1 mm), but still not harmful for the patient if we consider the flexibility of the abdominal wall. The overall behavior of the system also shows the effect of the trocar point motion (refer to Figs. 10 and 11). Naturally, the error increases whenever the trocar point moves in the opposite direction of the end effector trajectory. This problem is further amplified whenever the manipulator moves to a less dexterous region of the workspace. However, if we disregard the trajectories with $s = 0.1$, which proved itself too fast for the system to handle with a dynamic trocar point, the remaining trajectories have all errors below 1 mm.

Table 4. User controlled error.

	RMSE
RCM [mm]	0.0184
Translation [mm]	1.1
Rotation angle [rad]	0.0047
Rotation axis	0.0025

5.3. *Manual control*

In the case of manual control, the RCME, and the TE concerning translation, rotation angle, and rotation axis are shown in Table 4. The overall behavior of the aforementioned variables for all users can be seen in Fig. 13. The run of each user is separated by a vertical orange dotted line, for a total of 13 experiments lasting a total of 75.8 min. The workspace was thoroughly covered by the users as evidenced by Fig. 12.

The behavior of the system was smooth overall, enabling users to enjoy full control of the endoscope. With the choice of parameters $\mathbf{K} = 1.5$ and $s = 0.025$, the

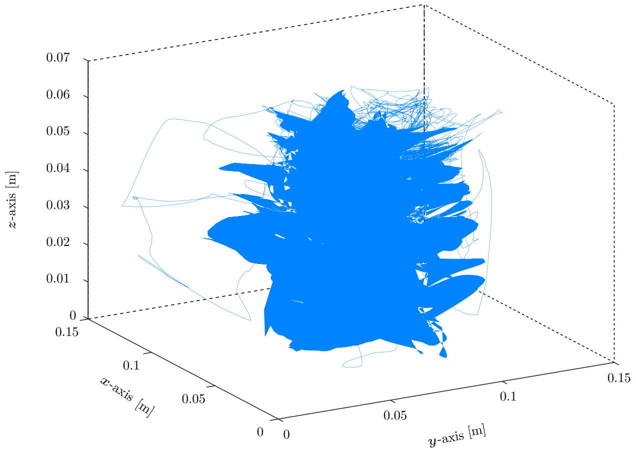


Fig. 12. The workspace coverage of the endoscope in the experiments using user inputs, considering the inputs of all users. The reference point in the plot is the combination of the lowest coordinates achieved by the endoscope during the experiments.

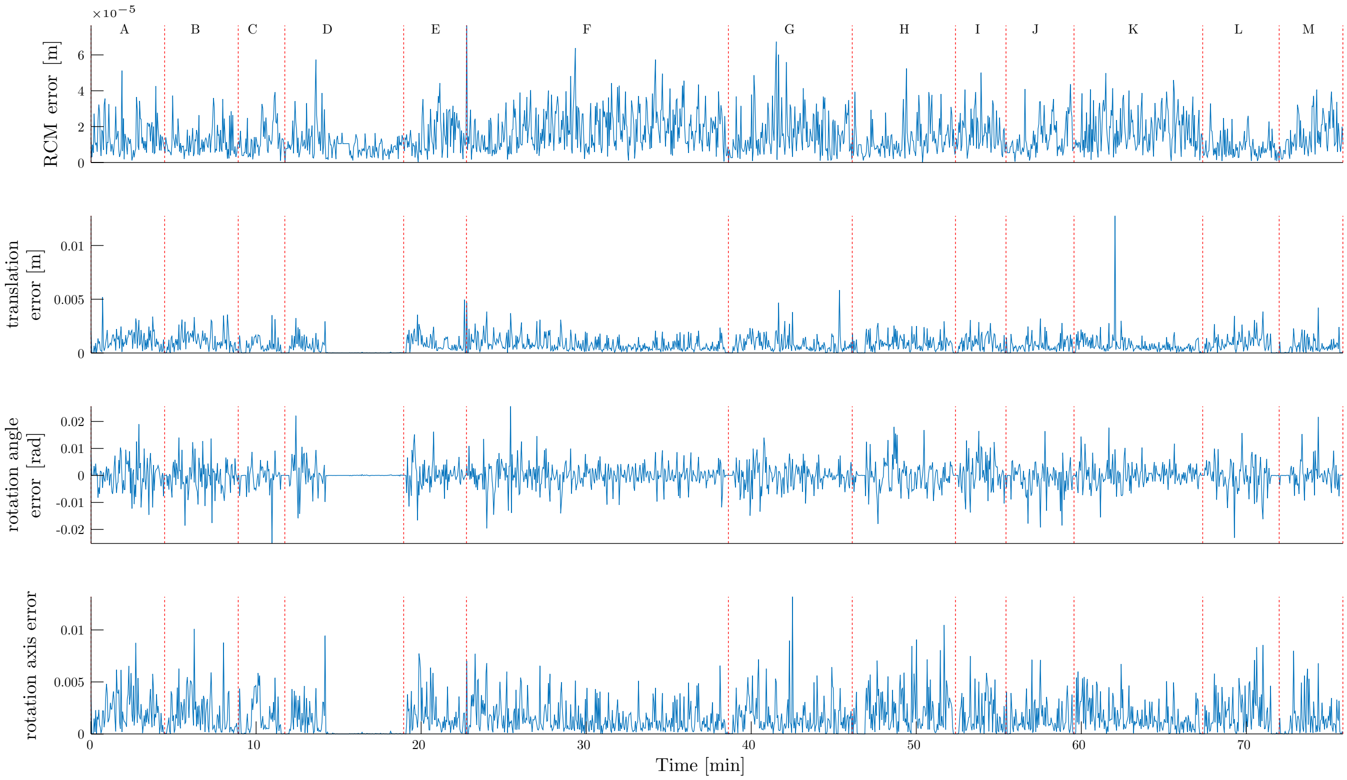


Fig. 13. System performance under manual control. The plots englobe 13 experiments (A-M) performed by untrained users using the parameters $\mathbf{K} = 1.5$ and $s = 0.025$. Note that the RCM error is negligible during the whole procedure. The translation error is mostly under 1 mm and can be smaller by trading-off with the RCM error through a larger **K**. The last part of user D operation included releasing the haptics interface, allowing the position to converge. The converge is important since during real-life surgeries the endoscope should remain still for most of the procedure.

system remained stable during all experiments. The system provided both small RCME $= 1.84\,\mathrm{mm}^{-2}$, and small error in relation to the planned trajectory, even when the user moved the haptics interface abruptly. These preliminary user experiments indicate that the system is not vulnerable to drift phenomena. In fact, user D (between 13 and 20 min in Fig. 13) released the haptics interface and the trajectory errors dropped substantially. Since the RCME is already very small, releasing the interface reduces the RCME less sharply. Note that the RCME was under 0.1 mm for all users. The trajectory error translation-wise was mostly under 2.5 mm with a small RMSE of 1.1 mm, which converged to near-zero when the user stopped generating new references. A system with a smaller communication delay would show even better results.

The trade-off between joint velocities norm and errors — RCME and TE —can be changed by varying the gain **K**. There is, however, a peak value of performance, depending on the sampling rate of the system, after which increases in the gain matrix begin causing more errors. Moreover, the trade-off between trajectory error and trajectory velocity is defined by the velocity variable s. The desired response-time may vary between surgical procedures, and also possibly be dynamic during the procedure. Those effects along with a deeper analysis about telemanipulation performance measures remain for a future work.

6. Conclusions and Future Work

In this work, the use of a fully actuated robotic manipulator to compensate for biological motion in laparoscopic surgery settings was addressed. Despite earlier approaches that employ constrained Jacobian techniques, we evaluated the effects of using a constrained trajectory to allow reliable control of the robotic system. The trajectory is obtained by a path planner that provides a constrained dual quaternion trajectory of the tool end-point given the current trocar point position. Such trajectory allows controlling the laparoscopic tool positioning while compensating for biological motion modeled with respect to the trocar point. Moreover, the technique can be applied for any serial-link manipulator with at least 6 DOF, being applicable also to redundant robotic manipulators.

The system was tested in artificial trajectories with both static and moving trocar points, demonstrating that with a proper choice of parameters both the remote center-of-motion and trajectory errors can be small. The system was also tested under user input, showing its reliability while being operated by human hands. This showed that the system does not show any drifting phenomena, and provided free exploration of the workspace.

The results showed safe errors in both trocar point constraint and trajectory tracking, but the setup had some limitations that directly affected the quality of the results. First, the communication between the controlling computer and the robot had a delay that caused the minimum feasible sampling rate to be at about 150 ms. This is a relevant source of error, and most modern systems can operate at 10 ms and mitigate such effects. Furthermore, the overall proportion between robot and task space was not optimal. COMAU Smart Six horizontal reach is approximately 1.4 m, to perform a task in the millimeter or centimeter scale. Smaller robots such as Kuka's[e] medical robot line could show better results since their scale better couples with the given task. Another source of error is that the trajectory being generated is a constant velocity trajectory, therefore it has discontinuous accelerations in its beginnig and its end. This is not a particular issue in this work, but results can be improved by having a smoother acceleration profile. Regardless of the experimental setup limitations, the system performed good proof-of-concept trials.

In future works, the experimental setup can be improved. Also, experiments should also be devised for operations by surgeons to provide a professional insight of the usability of the system. Moreover, as the experiments were performed in a 6 DOF robotic manipulator, the full exploitation of redundancies in the serial chain by using a redundant manipulator remains as the topic of future works.

Acknowledgments

The authors would like to thank Prof. Bruno Vilhena Adorno, his students, and Universidade de Minas Gerais for providing us the COMAU Smart Six robot and logistical support for the experimental setup at UFMG. We also thank the handy insights in the trajectory tracking problem.

Appendix A

We begin by recalling dual quaternions and their basic algebra when representing rigid transformations to establish the notation used in this work, followed by a brief review of existing work in the kinematic control of robotic manipulators. More information on dual quaternion algebra can be found in [27].

The dual quaternions are the basic building blocks of the kinematic control theory implemented in this work. We begin by defining $\hat{\imath}$, $\hat{\jmath}$ and $\hat{k}$ as the three imaginary components of a quaternion such that $\hat{\imath}^2 = \hat{\jmath}^2 = \hat{k}^2 = -1$ and $\hat{\imath}\hat{\jmath}\hat{k} = -1$.

Hence, a general quaternion $\mathbf{x}$ is given by

$$\mathbf{x} = q_1 + q_2\hat{\imath} + q_3\hat{\jmath} + q_4\hat{k},$$

and its conjugate is defined as $\mathbf{x}^* \triangleq q_1 - q_2\hat{\imath} - q_3\hat{\jmath} - q_4\hat{k}$.

The norm of a quaternion $\mathbf{x}$ is $\|\mathbf{x}\| = \sqrt{\mathbf{x}\mathbf{x}^*}$. An arbitrary rotation of a rigid body by an angle θ around an axis $\mathbf{n} = n_x\hat{\imath} + n_y\hat{\jmath} + n_z\hat{k}$ is represented by the unit norm quaternion

$$\mathbf{r} = \cos\frac{\theta}{2} + \mathbf{n}\sin\frac{\theta}{2}.$$

A translation described by $\mathbf{t} = t_x\hat{\imath} + t_y\hat{\jmath} + t_z\hat{k}$ can be associated to a rotation $\mathbf{r}$ in order to represent the complete rigid motion. This is represented by the unit dual quaternion

$$\underline{\mathbf{x}} = \mathbf{r} + \varepsilon\frac{1}{2}\mathbf{t}\mathbf{r},$$

and its conjugate is defined as $\underline{\mathbf{x}}^* \triangleq \mathbf{r}^* + \varepsilon(\frac{1}{2}\mathbf{t}\mathbf{r})^*$; where ε is nil-potent; i.e., $\varepsilon \neq 0$ but $\varepsilon^2 = 0$.

The translator operator obtains the translation quaternion from the dual quaternion, that is

$$\mathrm{translation}(\underline{\mathbf{x}}) \triangleq \mathbf{t}.$$

The P operator returns a quaternion that represents the pure part of a dual quaternion $\underline{\mathbf{x}}$, that is $P(\underline{\mathbf{x}}) \triangleq \mathbf{r}$. On the other hand, the D operator returns a quaternion that stands for the dual part of a dual quaternion $\underline{\mathbf{x}}$, meaning $D(\underline{\mathbf{x}}) \triangleq \frac{1}{2}\mathbf{t}\mathbf{r}$. The logarithm of $\underline{\mathbf{x}}$ is

$$\log\underline{\mathbf{x}} \triangleq \frac{\theta\mathbf{n}}{2} + \epsilon\frac{\mathbf{t}}{2},$$

while the exponential of $\underline{\mathbf{x}}$ is

$$\exp\underline{\mathbf{x}} \triangleq P(\exp\underline{\mathbf{x}}) + \epsilon D(\underline{\mathbf{x}})P(\exp\underline{\mathbf{x}})$$

with

$$P(\exp\underline{\mathbf{x}}) \triangleq \cos||P(\underline{\mathbf{x}})|| + \frac{\sin||P(\underline{\mathbf{x}})||}{||P(\underline{\mathbf{x}})||}P(\underline{\mathbf{x}}),$$

if $||P(\underline{\mathbf{x}})|| \neq 0$ and $P(\exp\underline{\mathbf{x}}) = 1$ otherwise.

The vec operator maps a given dual quaternion $\underline{\mathbf{x}}$ into an eight-dimensional column vector; i.e.,

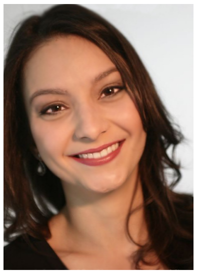
$$\mathrm{vec}\ \underline{\mathbf{x}} \triangleq [q_1\ q_2\ q_3\ q_4\ q_5\ q_6\ q_7\ q_8]^T.$$

[e] http://www.kuka-healthcare.com/en/.

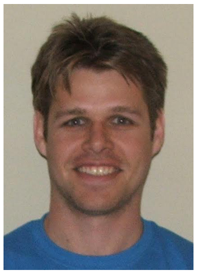
References

1. G. Fichtinger, P. Kazanzides, A. M. Okamura, G. D. Hager, L. L. Whitcomb and R. H. Taylor, Surgical and interventional robotics: Part II, *IEEE Robot. Autom. Mag.* (2008) 94–102.
2. C. N. Riviere, J. Gangloff and M. De Mathelin, Robotic compensation of biological motion to enhance surgical accuracy, *Proc. IEEE* **94**(9) (2006) 1705–1715, doi: 10.1109/JPROC.2006.880722.
3. A. Krupa, J. Gangloff, C. Doignon, M. F. De Mathelin, G. Morel, J. Leroy, L. Soler and J. Marescaux, Autonomous 3-D positioning of surgical instruments in robotized laparoscopic surgery using visual servoing, *IEEE Trans. Robot. Autom.* **19** (2003) 842–853, doi: 10.1109/TRA.2003.817086.
4. T. Ortmaier, Motion compensation in minimally invasive robotic surgery, Thesis.
5. M. Hoeckelmann, I. J. Rudas, P. Fiorini, F. Kirchner and T. Haidegger, Current capabilities and development potential in surgical robotics, *Int. J. Adv. Robot. Syst.* **12** (2015) 61.
6. V. Larocca, F. Marino, A. De Filippis, S. Gidaro and A. Lococo, A new operative telesurgical system: Telelap alf-x-experimental study on animal model, *J. Adv. Biotechnol. Bioeng.* **2**(1) (2014) 12–15.
7. M. J. Lum, D. C. Friedman, G. Sankaranarayanan, H. King, K. Fodero, R. Leuschke, B. Hannaford, J. Rosen and M. N. Sinanan, The raven: Design and validation of a telesurgery system, *Int. J. Robot. Res.* **28**(9) (2009) 1183–1197.
8. B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. N. Kosari and L. White, Raven-ii: An open platform for surgical robotics research, *IEEE Trans. Biomed. Eng.* **60**(4) (2013) 954–959.
9. S. Dieterich, J. Tang, J. Rodgers and K. Cleary, Skin respiratory motion tracking for stereotactic radiosurgery using the cyberknife, *Int. Congress Ser.* **1256** (2003) 130–136.
10. S. Ferrand-Sorbets, M. Delphine Taussig, M. Fohlen, C. Bulteau, G. Dorfmuller and O. Delalande, Frameless stereotactic robot-guided placement of depth electrodes for stereo-electroencephalography in the presurgical evaluation of children with drug-resistant focal epilepsy, *CNS Annual Meeting* (2010).
11. J. Troccaz, U. Hagn, M. Nickl, S. Jörg, G. Passig, T. Bahls, A. Nothhelfer, F. Hacker, L. Le-Tien, A. Albu-Schäffer *et al.*, The dlr miro: A versatile lightweight robot for surgical applications, *Ind. Robot: An Int. J.* **35**(4) (2008) 324–336.
12. T. Osa, C. Staub and A. Knoll, Framework of automatic robot surgery system using visual servoing, *2010 IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, (IEEE, 2010), pp. 1837–1842.
13. H. Azimian, R. V. Patel and M. D. Naish, On constrained manipulation in robotics-assisted minimally invasive surgery, in *2010 3rd IEEE RAS and EMBS Int. Conf. Biomedical Robotics and Biomechatronics (BioRob)* (IEEE, 2010), pp. 650–655.
14. P. J. From, On the kinematics of robotic-assisted minimally invasive surgery, *Model. Identif. Control* **34**(2) (2013) 69–82.
15. M. M. Marinho, M. C. Bernardes and A. P. L. Bo, A programmable remote center-of-motion controller for minimally invasive surgery using the dual quaternion framework (2014).
16. R. Richa, P. Poignet and C. Liu, Three-dimensional motion tracking for beating heart surgery using a thin-plate spline deformable model, *Int. J. Robot. Res.* **32**(1) (2009) 218–230.
17. B. Adorno, A. Bo, P. Fraisse and P. Poignet, Towards a cooperative framework for interactive manipulation involving a human and a humanoid (2011).
18. L. Figueredo, B. Adorno, J. Ishihara and G. Borges, Robust kinematic control of manipulator robots using dual quaternion representation (2013).
19. H.-L. Pham, V. Perdereau, B. V. Adorno and P. Fraisse, Position and orientation control of robot manipulators using dual quaternion feedback, in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems* (2010), pp. 658–663.
20. B. Akyar, Dual quaternions in spatial kinematics in an algebraic sense, *Tubitak* **32** (2008) 373–379.
21. A. Maciejewski and C. Klein, Numerical filtering for the operation of robotic manipulators through kinematically singular configurations, *J. Robot. Syst.* **5** (1988) 527–552.
22. S. Chiaverini, Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, *IEEE Trans. Robot. Autom.* **13**(3) (1997) 398–410, doi: 10.1109/70.585902.
23. T. Yoshikawa, Manipulabilty and redundancy control of robotic mechanisms (1985).
24. R. C. O. Locke, R. V. Patel, Optimal remote center-of-motion location for robotics-assisted minimally-invasive surgery, April 2007, pp. 10–14.
25. H. Azimian, S. Member, R. V. Patel and M. D. Naish, *On constrained manipulation in robotics-assisted minimally invasive surgery*, Japan (2010), pp. 650–655.
26. A. Liegeois, Automatic supervisory control of the configuration and behavior of multibody mechanisms, *IEEE Trans. Syst. Man Cybern.* **7** (1977) 868–871, doi: 10.1109/TSMC.1977.4309644.
27. B. Akyar, Dual quaternions in spatial kinematics in an algebraic sense, *Turkish J. Math.* **32**(4) (2008) 373–391.
28. R. Richa, A. P. Bó and P. Poignet, Towards robust 3d visual tracking for motion compensation in beating heart surgery, *Med. Image Anal.* **15**(3) (2011) 302–315.
29. U. Hagn, M. Nickl, S. Jörg, G. Passig, T. Bahls, A. Nothhelfer, F. Hacker, L. Le-Tien, A. Albu-Schäffer, R. Konietschke, M. Grebenstein, R. Warpup, R. Haslinger, M. Frommberge, and G. Hirzinger, The DLR MIRO: A versatile lightweight robot for surgical applications, *Ind. Robot: An Int. J.* **35**(4) (2008) 324–336.

Murilo M. Marinho received the B.Eng and M.Sc. degrees from the University of Brasília, Brazil in 2012 and 2014, respectively. He is currently a Ph.D. candidate in Mechanics Engineering at The University of Tokyo, Japan, in the Mitsuishi-Sugita Laboratory. His research interests include surgical robotics, manipulator control, and robotics simulation.



Mariana C. Bernardes is Ph.D. in Automatic and Microelectronic Systems from Université Montpellier 2 and in Automation and Electronic Systems from Universidade de Brasília (cotutelle 2012). She has a M.Sc. in Electrical Engineering from Universidade de Brasília (2009) and a B.Sc. in Mechatronic Engineering from the same institution (2005). She has experience in Control and Automation and her research interests are in the fields of medical robotics and human–robot interaction. Currently, she is an Assistant Professor at Electronic Engineering in Universidade de Brasília (UnB) and an Associate Researcher at the Graduate Program in Electronic Systems and Automation (PEGA), also at UnB.



Antônio Padilha L. Bó received the B.Eng. and M.Sc. degrees from the University of Brasília, Brazil, in 2004 and 2007, respectively, and the Ph.D. degree from the University of Montpellier 2, France, in 2010. He is currently Assistant Professor at the Electrical Engineering Department of the University of Brasília, where his research activities are mainly conducted at the Laboratory of Automation and Robotics. His research interests include the application of robots and associated technologies to functional rehabilitation, assistance to disabled persons and surgical procedures. He has been or is involved in different research projects in which there is close collaboration with clinical partners. He has co-authored over 50 peer reviewed publications, including the Best Medical Robotics Paper sponsored by Intuitive Surgical in the *13th International Conference on Medical Image Computing and Computer Assisted Intervention* (*MICCAI 2010*).