

Dual-Arm Coordination Using Dual Quaternions and Virtual Mechanisms

Rohit Chandra, Carlos Mateo, Juan Antonio Corrales Ramon, Youcef Mezouar

▶ **To cite this version:**

Rohit Chandra, Carlos Mateo, Juan Antonio Corrales Ramon, Youcef Mezouar. Dual-Arm Coordination Using Dual Quaternions and Virtual Mechanisms. 2018 IEEE International Conference on Robotics and Biomimetics (ROBIO), Dec 2018, Kuala Lumpur, Malaysia. pp.759-765, 10.1109/ROBIO.2018.8665292 . hal-02098256

HAL Id: hal-02098256
https://uca.hal.science/hal-02098256

Submitted on 12 Apr 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Dual-arm coordination using dual quaternions and virtual mechanisms

Rohit Chandra, Carlos M. Mateo, Juan Antonio Corrales-Ramon and Youcef Mezouar

***Abstract*— A novel approach to the kinematic coordination problem for dual-arm robots and for the definition of bimanual tasks is proposed, where both modelling and control aspects of the problem are handled using dual quaternion representation of pose and screw-based manipulator Jacobian. The proposed formulation is advantageous in terms of computation and storage efficiency compared to other formulations of dual-arm manipulator forward kinematics and Jacobian computation. Unit dual quaternion representation is also capable of avoiding representational singularities related to orientation.**

The coordination problem for dual-arm manipulation is addressed using relative Jacobian, and parametrized task definition is achieved using virtual mechanisms and task-specific Jacobians. The framework hence developed has been validated using a dual-arm Baxter robot for steering wheel rotation and bimanual pouring tasks, thus validating the position controller with both relative and global task-space motions. The proposed experiments use a visual sensing strategy to detect and follow the manipulated objects during the task.

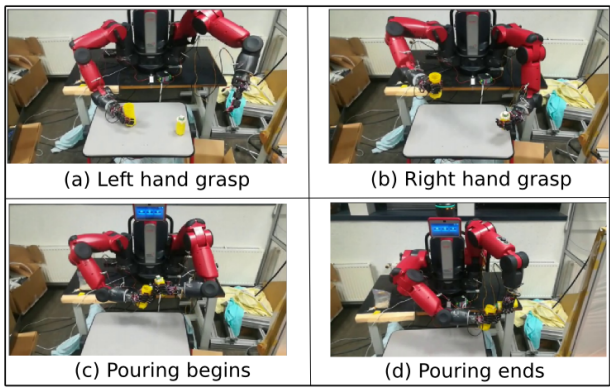


Fig. 1. Baxter performing bimanual beans pouring task.

I. INTRODUCTION

A dual-arm robot performing coordinated bimanual tasks can be considered as a closed chain mechanism, where the cooperating end-effectors are connected by the kinematic constraints of the task. Two kinds of scenarios exists for such dual-arm coordination: handling a single object; and tasks where both arms are handling two different objects and some coordination is required between them, for example, assembly tasks. These coordination scenarios has been referred to as parallel and serial mode of coordination, respectively, in [1]. While the relative cooperative behaviour is common to both serial and parallel cooperative structures, the parallel cooperative tasks can also be defined as a variation of serial tasks, i.e. by defining the relative configuration of the two arms and simultaneously deriving the configuration for one of the arms from the common task.

This work aims at formulating the kinematic modelling and control problem of dual-arm robots for the above mentioned cooperative tasks using dual quaternion representation of pose and screw-based manipulator Jacobians. The computational and storage efficiency of unit dual quaternion ([2]), and singularity free representation of orientation [3] has been utilized for the pose representation. Inverse kinematics control has been achieved using screw-based differential kinematics. Screw-based kinematics has the advantage of flexibility of the choice of reference frames on the links and manipulator starting configuration, over Denavit–Hartenberg (DH) convention based kinematics [4]. It also provides a geometrically intuitive representation of manipulator Jacobian. An object tracking technique based on points cloud ([5], [6]) has been used to add the perception faculty to the framework for the beans pouring task, to validate the method with a more realistic setup.

The contributions of this work are:

- Usage of dual quaternion representation, based on screw-theory, for the computation of relative Jacobian of cooperating arms for bimanual manipulation. Thus facilitating better relative pose tracking, even in the case of high relative angular velocity [7].
- A parametrized task definition, achieved using virtual mechanisms attached between two frames, which can either be fixed to the two end-effectors, or between one of the end-effectors and a global frame, or both. These virtual mechanisms eases the description of certain tasks where some of the degrees of freedom in task space is constrained, e.g. steering wheel rotation task (Fig. 6).

Section II provides the literature review of the available approaches for dual-arm coordination modelling. Section III provides the mathematical basis for kinematic modelling and control. Control architecture of the proposed framework is presented in section IV. Object tracking pipeline and experimental evaluation of the framework is presented in section V, and finally, some concluding remarks about our work and future goals are discussed.

II. RELATED WORK AND CONTRIBUTION

Uchiyama [8] provided one of the earliest symmetric formulation to deal with dual-arm coordination considering

All authors are with Université Clermont Auvergne, SIGMA Clermont, Institut Pascal BP 10448, F-63000 Clermont-Ferrand, France.
Corresponding author is ol.chandra@gmail.com

both arms collectively, and introduced two task space variables, namely absolute and relative velocity, for the static analysis of an object jointly held by the end-effectors of two manipulators. These were the average and the difference of the velocities of the tips of the *virtual sticks*, *i.e.* vectors connecting the two end-effectors to the centre of the object.

Relative Jacobian was formulated in [9] using DH convention based frame definition of manipulator links and rotation matrix representation of orientation for the trajectory generation for two robots cooperating to perform assembly tasks. It allowed the cooperating manipulators to be treated as a single manipulator, while facilitating higher dimensional null space.

In [10], a modular relative Jacobian was obtained by using the individual Jacobians of the cooperating manipulators, and used it for impedance control between two arms. A slightly different formulation of relative Jacobian was obtained in [7]. The time derivative of the relative pose was used for the derivation of a modular relative Jacobian which revealed a *wrench transformation matrix*, a function of mutual position of the cooperating arms' end-effectors. The modular relative Jacobian composed of *wrench transformation matrix*, hence obtained, demonstrated better trajectory tracking accuracy for high relative angular velocity of the end-effectors.

Adorno [11] provided dual quaternion representation of task space variables based on DH convention. While theoretical foundation of the relative Jacobian formulation was given for a general manipulator, complete formulation of relative Jacobian was limited to dual-arm robots with rotational joints. While most of the modern manipulators have rotational joints, differential kinematics formulation for both rotational as well as prismatic joints becomes especially relevant when virtual mechanisms are used to represent task constraints, e.g. the translational motion in peg-in-hole tasks.

Moreover, the computational efficiency of dual quaternions was underutilized in [11], as their computation of Jacobian required more mathematical operations than the homogeneous transformation matrix approach, attributed to the use of a large (8×8) dual quaternion Hamilton matrix to compensate for the non-commutative nature of dual quaternion multiplication. The pose error term used in [11] was the numerical difference between the unit dual quaternion representing desired and current frames for relative pose control, which does not represent a meaningful displacement variable [2].

Guenther [4] provided a new formulation of relative Jacobian based on screw theory that required only two frames, *i.e.* reference and tool frame, thus eliminating the need of defining frames on each involved link for the corresponding DH convention based formulation. However, they used the rotation matrices to represent the orientation of the screw axis. Screw theory and unit dual quaternion algebra were used in [3] for forward as well inverse kinematics problem of cooperative manipulation for dual-arm robots, however, the differential kinematics problem was not addressed.

Forward kinematic displacement computation and Jacobian formulation was proposed for kinematic control of a serial manipulator using unit dual quaternion based on screw theory in [2]. The exponential convergence of the control law, using the screw parameters derived from the error unit dual quaternion, was proved using a Lyapunov candidate function.

In this work we propose a new formulation of dual-arm relative differential kinematics, i.e. relative Jacobian which is based on [2] and [4]. In order to control the task defined with respect to a global frame, the pose of one of the arms, referred as reference manipulator in this work, is controlled, in addition to relative pose of the end-effectors. Task-specific Jacobians have been used to define various control modes, such as *relative pose* control, *relative plus absolute pose* control, *relative orientation* control, etc. Virtual mechanisms have been defined using similar dual quaternion formulation to model task constraints and to compute desired trajectories to be used for bimanual tasks. Additionally, an exponentially convergent proportional control law has been utilized for pose control.

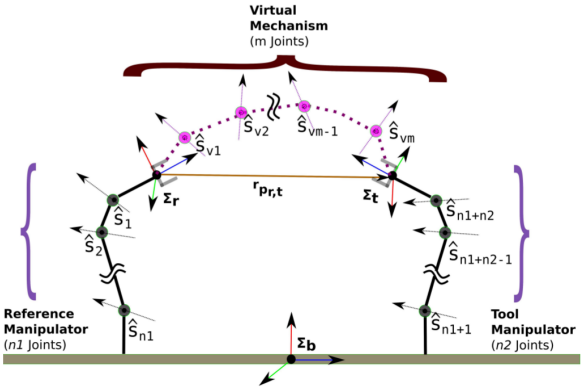


Fig. 2. Dual-arm frames and joint axes and virtual joints.

III. Mathematical Formulation for Dual-Arm Coordination

The dual arm set-up along with the different frames used in the following formulations is given in Fig. 2. $\boldsymbol{\Sigma_r}$ is the reference frame, $\boldsymbol{\Sigma_t}$ is the tool frame and $\boldsymbol{\Sigma_b}$ is the fixed base frame.

In the following explanation, dual quaternions and dual vectors have been notated with bold characters with hat (e.g. $\hat{\mathbf{x}}$ and $\hat{\mathbf{s}}$, respectively), where dual vectors are dual quaternions with pure quaternions[1] as dual and non-dual part. Dual numbers have been referred with variables with hat (e.g. $\hat{\theta}$), whereas dual number array has been notated with underlined bold characters with hat (e.g. $\hat{\underline{\boldsymbol{\theta}}}$). Real number arrays are referred using underlined bold characters (e.g. $\underline{\boldsymbol{\theta}}$) and vectors and quaternions are notated with bold characters (e.g. $\boldsymbol{l}$).

[1]A pure quaternion, $\boldsymbol{q}$ is defined as $\boldsymbol{q} = q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k}$.

A. Dual Quaternion Pose Representation

Pose representation using unit dual quaternion is given as:

$$\hat{\mathbf{x}} = \exp\left(\frac{\hat{\theta}}{2}\ \hat{\mathbf{s}}\right) = \cos\left(\frac{\hat{\theta}}{2}\right) + \hat{\mathbf{s}}\ \sin\left(\frac{\hat{\theta}}{2}\right) \tag{1}$$

where, $\hat{\theta} \in \mathrm{D}$ is a dual angle and $\hat{\mathbf{s}} \in \mathrm{D}^{3\times 1}$ is a directed line represented using a unit dual vector.

$$\hat{\theta} = \theta + \varepsilon\ d, \qquad \hat{\mathbf{s}} = \boldsymbol{l} + \varepsilon\ \boldsymbol{m}, \quad \varepsilon^2 = 0, \qquad \varepsilon \neq 0, \tag{2}$$

where, θ is a rotation around the screw axis, d is translation along the same axis, $\boldsymbol{l}$ is the unit direction vector of the screw axis, and $\boldsymbol{m}$ is the moment of this screw axis with respect to the origin of the reference frame ($\boldsymbol{m} = \boldsymbol{p} \times \boldsymbol{l}$), and $\boldsymbol{p}$ is an arbitrary point lying on the screw axis.

B. Relative Forward Kinematics

In Fig. 2, $\hat{\mathbf{s}}_i$ is the dual vector of joint screw joint for $i = 1, ..., (n1 + n2)$, corresponding to the dual arm manipulator and $\hat{\mathbf{s}}_{vi}$ for $vi = 1, ..., m$, corresponds to the virtual mechanism, computed and given in $\boldsymbol{\Sigma_r}$. Joint screw axes for the dual-arm robot are counted from 1 to $(n1+n2)$, where joint screw axis 1 is attached and closest to the reference frame. ${}^{r}\boldsymbol{p}_{r,t} = \boldsymbol{p}_t$ is the position of tool frame origin computed and given with respect to $\boldsymbol{\Sigma_r}$.

Let $\hat{\mathbf{x}}_{t_0}$ be the initial relative pose of the tool end-effector with respect to the reference end-effector, for the initial joint configuration $\underline{\hat{\boldsymbol{\theta}}}_0 = \begin{bmatrix}\hat{\theta}_{1_0} & \hat{\theta}_{2_0} & \cdots & \hat{\theta}_{(n_1+n_2)_0}\end{bmatrix}^T \in \mathrm{D}^{(n_1+n_2)\times 1}$. The tool end-effector pose $\hat{\mathbf{x}}_t$, relative to the reference frame, $\boldsymbol{\Sigma_r}$, for joint configuration $\underline{\hat{\boldsymbol{\theta}}} = \begin{bmatrix}\hat{\theta}_1 & \hat{\theta}_2 & \cdots & \hat{\theta}_{(n_1+n_2)}\end{bmatrix}^T \in \mathrm{D}^{(n_1+n_2)\times 1}$ is given as:

$$\begin{aligned}
&\hat{\mathbf{x}}_t = \hat{\boldsymbol{\delta}}_T\ \hat{\mathbf{x}}_{t_0}, \\
&\hat{\boldsymbol{\delta}}_T = \hat{\boldsymbol{\delta}}_1\ \hat{\boldsymbol{\delta}}_2\ \cdots\ \hat{\boldsymbol{\delta}}_(n_1 + n_2), \\
&\hat{\boldsymbol{\delta}}_i = \exp\left(\frac{\hat{\theta}_i}{2}\ \hat{\mathbf{s}}_{i_0}\right).
\end{aligned} \tag{3}$$

where $\hat{\theta}_i$ is a joint displacement from the home position $\hat{\theta}_{i_0}$, and,

$$\begin{aligned}
\hat{\theta}_i &= \Delta\theta_i, && \text{for revolute joints,} \\
\hat{\theta}_i &= \varepsilon\ \Delta d_i, && \text{for prismatic joints.}
\end{aligned}$$

$\hat{\mathbf{s}}_{i_0}$ is the initial joint screw axis described using $\boldsymbol{l}_{i_0}$ and $\boldsymbol{m}_{i_0}$, which are the unit dual vector for the initial axis and dual vector moment of the joint screw axis about the base frame of the manipulator, and $\boldsymbol{m}_{i_0}$ is computed as:

$$\boldsymbol{m}_{i_0} = \boldsymbol{p}_{i_0} \times \boldsymbol{l}_{i_0}, \tag{4}$$

where $\boldsymbol{p}_{i_0}$ is the initial position of a point on i_{th} joint axis.

C. Relative Jacobian

The relative twist between the reference and tool frame is given as:

$${}^{r}\hat{\boldsymbol{\xi}}_{rt} = \boldsymbol{\omega}_r + \varepsilon\ \boldsymbol{v}_{r_0}. \tag{5}$$

where, $\boldsymbol{\omega}_r$ is the rotational velocity of the tool frame, and $\boldsymbol{v}_{r_0}$ is the translational velocity of a point which is attached to the tool frame and is instantaneously coincident with the origin of reference frame. $\boldsymbol{\omega}_r$ is defined as a pure quaternion non-dual part, and $\boldsymbol{v}_{r_0}$ is defined as a pure quaternion dual part of the dual quaternion ${}^{r}\hat{\boldsymbol{\xi}}_{rt}$.

The relative twist of the tool end-effector is given as:

$${}^{r}\hat{\boldsymbol{\xi}}_{rt} = \begin{bmatrix}\hat{\mathbf{s}}_1 & \hat{\mathbf{s}}_2 & \cdots & \hat{\mathbf{s}}_{(n_1+n_2)}\end{bmatrix} \dot{\underline{\hat{\boldsymbol{\theta}}}} = \hat{\mathbf{J}'}_r \dot{\underline{\hat{\boldsymbol{\theta}}}} \tag{6}$$

Therefore, relative Jacobian is given as:

$$\hat{\mathbf{J}'}_r = \begin{bmatrix}\hat{\mathbf{s}}_1 & \hat{\mathbf{s}}_2 & \cdots & \hat{\mathbf{s}}_{(n_1+n_2)}\end{bmatrix}. \tag{7}$$

The current dual vector of joint screw, $\hat{\mathbf{s}}_i$, for the i_{th} joint, can be obtained from its initial value $\hat{\mathbf{s}}_{i_0}$, by transforming it using the total displacement caused by the previous $(i-1)$ joints.

$$\hat{\mathbf{s}}_i = {}^{b}\hat{\boldsymbol{\delta}}_{T_{(i-1)}}\ \hat{\mathbf{s}}_{i_0}\ {}^{b}\hat{\boldsymbol{\delta}}^{*}_{T_{(i-1)}} \tag{8}$$

${}^{b}\hat{\boldsymbol{\delta}}^{*}_{T_{(i-1)}}$ is the classical dual quaternion conjugate of ${}^{b}\hat{\boldsymbol{\delta}}_{T_{(i-1)}}$.
${}^{b}\hat{\boldsymbol{\delta}}_{T_{(i-1)}}$ can be obtained as:

$${}^{b}\hat{\boldsymbol{\delta}}_{T_{(i-1)}} = \hat{\boldsymbol{\delta}}_1\ \hat{\boldsymbol{\delta}}_2\ \cdots\ \hat{\boldsymbol{\delta}}_{i-1}. \tag{9}$$

Please note that in the case of first joint, *i.e.* $\hat{\mathbf{s}}_1$, the dual vector of joint screw is constant with respect to the reference frame.

$\hat{\mathbf{J}'}_r$ maps joint velocities to the translational velocity of a point attached to the tool end-effector and instantaneously coincident with the origin of reference frame ($\boldsymbol{v}_{r_0}$), while they map the same rotation velocity. In order to compute the compute relative Jacobian ($\hat{\mathbf{J}}_r$), that maps the joint velocities to the relative translational velocity of origin of the tool end-effector frame ($\boldsymbol{v}_r$), following relation can be used(refer Fig. 2):

$$\boldsymbol{v}_r = \boldsymbol{v}_{r_0} + \boldsymbol{\omega}_r \times \boldsymbol{p}_t, \tag{10}$$

where $\boldsymbol{p}_t$ (same as ${}^{r}\boldsymbol{p}_{r,t}$) is the current position of the tool end-effector with respect to the current pose of the reference end-effector frame given in current reference frame. $\boldsymbol{p}_t$ can be derived as a vector from the current relative pose of tool frame as:

$$\boldsymbol{p}_t = 2\boldsymbol{q}_t\boldsymbol{q}_r^*,$$

where $\boldsymbol{q}_r$, and $\boldsymbol{q}_t$ are the rotational and translational component of $\hat{\mathbf{x}}_t$, respectively, such that $\hat{\mathbf{x}}_t = \boldsymbol{q}_r + \varepsilon\boldsymbol{q}_t$, and $\boldsymbol{q}_r^*$ is the quaternion conjugate of $\boldsymbol{q}_r$.

The relative Jacobian ($\hat{\mathbf{J}}_r$) can now be represented using modified screw axes $\hat{\mathbf{s}}_{mod_i} \forall i = 1, \ldots, (n_1 + n_2)$, as:

$$\hat{\mathbf{J}}_r = \begin{bmatrix}\hat{\mathbf{s}}_{mod_1} & \hat{\mathbf{s}}_{mod_2} & \cdots & \hat{\mathbf{s}}_{mod_{(n_1+n_2)}}\end{bmatrix}, \tag{11}$$

where $\hat{\mathbf{s}}_{mod_i}$ is given as:

$$\hat{\mathbf{s}}_{mod_i} = \boldsymbol{l}_i + \varepsilon\ (\boldsymbol{p}_i - \boldsymbol{p}_t) \times \boldsymbol{l}_i = \hat{\mathbf{s}}_i - \varepsilon\ (\boldsymbol{p}_t \times \boldsymbol{l}_i) \tag{12}$$

D. Task Modeling using Virtual Mechanisms

The different kinematic constraints related to different tasks can be represented using virtual mechanisms. Different tasks with their respective virtual mechanisms used for modelling the constraints have been depicted in Fig. 4 and 5. Please note that both dual-arm manipulators and the virtual mechanism connecting the two end-effectors share the same reference and tool frame (Fig. 2). Since both the arms have been assumed to be redundant, hence a generalized virtual kinematic chain can be attached to the serial chain of the manipulator individually as well as in relative task space. Let,

$$\underline{\hat{\boldsymbol{\theta}}}_V = \begin{bmatrix}\hat{\theta}_{v1} & \hat{\theta}_{v2} & \cdots & \hat{\theta}_{vm}\end{bmatrix}^T \tag{13}$$

$$\hat{\mathbf{J}}_{V0} = \begin{bmatrix}\hat{\mathbf{s}}_{v1_0} & \hat{\mathbf{s}}_{v2_0} & \cdots & \hat{\mathbf{s}}_{vm_0}\end{bmatrix} \tag{14}$$

be the array of dual angle representing desired joint displacements of the virtual mechanism with m joints, and the initial dual vectors of joint screws corresponding the virtual mechanism.

The desired pose for a task represented using the virtual mechanism can be computed using (3), and (6), respectively, if $\underline{\hat{\boldsymbol{\theta}}}_{\boldsymbol{V}}$, i.e. the configuration of the virtual mechanism related to the desired task is given.

IV. Control Architecture

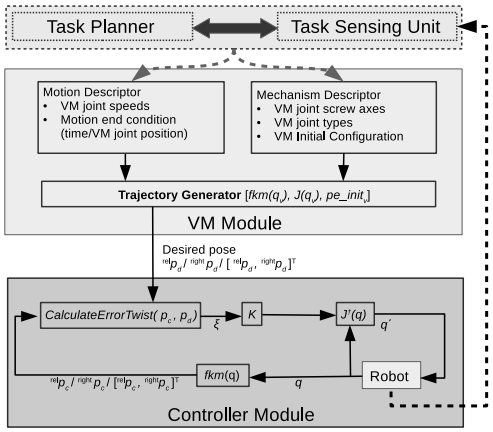


Fig. 3. Controller architecture for inverse kinematic velocity control for virtual mechanism based task definition.

The kinematic formulations based on screw theory with dual quaternion representation discussed in III was implemented for a general dual-arm system consisting of two redundant (more than 6 *dof*s) manipulators. Fig. 3 presents the controller architecture of the complete system capable of inverse kinematic control for a general virtual mechanism based task. The whole framework is divided in three layers. The top layer is composed of *Task Planner* and *Task Sensing Unit*, which are related to the task desired. In the current implementation we assume that we already have the information about the virtual mechanism that defines the task, and we send this information directly to *VM Module* using a *yaml* configuration file, consisting of joint screw axes, a point on the screw axes and initial configuration of virtual mechanisms. Desired joint rate of the virtual joints and motion end condition is also defined in the same configuration file.

The output of *VM Module* is the the desired pose of the *tool* end-effector relative to *reference* end-effector. The trajectory is given as input to the inverse kinematics controller illustrated in the *Controller Module* layer. The controller computes the error screw displacement parameter from the current and desired pose unit dual quaternions and computes the desired positions of the joints of a dual-arm system for next time step of the control iteration. These position commands are then subsequently passed to an existing joint position controller of the robot.

The error unit dual quaternion $\hat{\boldsymbol{e}}$, can obtained as the relative displacement between the desired and current pose, *i.e.* $\hat{\mathbf{x}}_d$ and $\hat{\mathbf{x}}_c$ respectively.

$$\hat{\mathbf{e}} = \hat{\mathbf{x}}_c \, \hat{\mathbf{x}}_d^* \tag{15}$$

where, $\hat{\mathbf{x}}_d^*$ is the classical dual quaternion conjugate of $\hat{\mathbf{x}}_d$. The control law for kinematic control shown in the *Controller Module* layer in Fig. 3, *i.e. CalculateErrorTwist*(p_c, p_d) along with the gain λ, is computed as:

$$\begin{aligned}\hat{\boldsymbol{\xi}} &= -\lambda \, \ln(\hat{\mathbf{e}}) \\ &= -\lambda(\theta_e \, \boldsymbol{l}_e - \varepsilon \, (\theta_e \, \boldsymbol{m}_e + d_e \, \boldsymbol{l}_e))\end{aligned} \tag{16}$$

where, $\{\theta_e, \, d_e, \, \boldsymbol{l}_e, \, \boldsymbol{m}_e\}$ are screw displacement parameter related to the error dual quaternion $\hat{\boldsymbol{e}}$, and λ is a positive scalar gain for the controller. The global exponential convergence of the above mentioned control law ($\hat{\boldsymbol{\xi}}$) has been proved for $-\pi \geq \theta_e \geq \pi$ in [2], which has been used for relative task-space kinematic control for the different control modes discussed below.

A. Control modes

Several control modes were defined to deal with different tasks related to dual-arm robots, where a specialized Jacobians and error vectors were defined for all of them. The screw axis dual vectors ($\hat{\mathbf{s}}_i \forall i = 1 \; to \; (n_1 + n_2)$) were obtained for all the joints in the base frame, and were transformed accordingly as required for defining Jacobians for different control modes, using following line transformation relation using unit dual quaternion:

$${}^a\hat{\mathbf{s}}_i = {}^a\hat{\mathbf{x}}_b \; {}^b\hat{\mathbf{s}}_i \; {}^a\hat{\mathbf{x}}_b^* \tag{17}$$

where, a line dual vector in frame b is being transformed to frame a using the unit dual quaternion ${}^a\hat{\mathbf{x}}_b$, i.e. pose of frame b in a frame.

The control modes used in the current implementation and following experiments are (refer Fig. 2):

1) *Right control mode*: The Jacobian for *right* control mode, used for controlling right arm, is given as:

$$\hat{\mathbf{J}}'_{right} = \begin{bmatrix}\hat{\mathbf{s}}_{n_1} & \hat{\mathbf{s}}_{n_1-1} & \cdots & \hat{\mathbf{s}}_{(1)}\end{bmatrix}, \tag{18}$$

where all the screw axis vectors are given in base frame.

2) *Left control mode*: The Jacobian for *left* control mode, used for controlling left arm, is given as:

$$\hat{\mathbf{J}}'_{left} = \begin{bmatrix} \hat{\mathbf{s}}_{n_1+1} & \hat{\mathbf{s}}_{n_1+2} & \cdots & \hat{\mathbf{s}}_{(n_1+n_2)} \end{bmatrix}, \quad (19)$$

where all the screw axis vectors are given in base frame.

3) *Relative control mode*: The Jacobian for *relative* control mode, to control only the relative configuration between the two end-effectors, is given as:

$$\hat{\mathbf{J}}'_{relative} = \begin{bmatrix} \hat{\mathbf{s}}_1 & \hat{\mathbf{s}}_2 & \cdots & \hat{\mathbf{s}}_{(n_1+n_2)} \end{bmatrix}, \quad (20)$$

where all the screw axis vectors are given in right end-effector (i.e. reference) frame.

4) *RightRelative control mode*: The Jacobian for *RightRelative* control mode, which is used to control both the right arm, as well as relative pose between the two end-effectors, is given as:

$$\hat{\mathbf{J}}'_{refRel} = \begin{bmatrix} \begin{bmatrix} \hat{\mathbf{J}}'_{ref} & \mathbf{0}^{8\times n2} \end{bmatrix} \\ \hat{\mathbf{J}}'_{r} \end{bmatrix}$$

where,

$$\hat{\mathbf{J}}'_{ref} = \begin{bmatrix} \hat{\mathbf{s}}_1 & \hat{\mathbf{s}}_2 & \cdots & \hat{\mathbf{s}}_{n_1} \end{bmatrix} \quad (21)$$

$$(22)$$

where the screw axis vectors are given in right end-effector (i.e. *reference*) frame.

5) *RightOrientationRelative control mode*: The Jacobian for *RightOrientationRelative* control mode, used to control just the orientation of the right arm, in addition to the relative configuration of the two end-effectors is given as:

$$\hat{\mathbf{J}}'_{refORel} = \begin{bmatrix} \begin{bmatrix} \hat{\mathbf{J}}''_{ref} & \mathbf{0}^{4\times n2} \end{bmatrix} \\ \hat{\mathbf{J}}'_{r} \end{bmatrix}$$

where, $\hat{\mathbf{J}}''_{refoRel}$ is the first four rows of $\hat{\mathbf{J}}'_{ref}$, pertaining to the rotation part.

In addition to the Jacobians related to reference and relative task space, the control law has to combined in a similar fashion, so that the exponential convergence property of the control law given in (16) can be exploited. For instance, for *RightRelative* control mode, the error unit dual quaternions are combined as,

$$\hat{\boldsymbol{\xi}}_{refRel} = \begin{bmatrix} \hat{\boldsymbol{\xi}}_{ref} \\ \hat{\boldsymbol{\xi}}_{rel} \end{bmatrix} \quad (23)$$

V. Experimentation Validation and Results

The above mentioned methodology was implemented on Baxter dual arm collaborative robot [12], which is equipped with two seven degrees of freedom arms. Baxter software developers kit (SDK) provides an interface to ROS to give joint velocity commands for both the manipulators. In order to compute torque command to execute the desired velocity, gravity compensation torques for each joint were internally computed and added by the Baxter SDK.

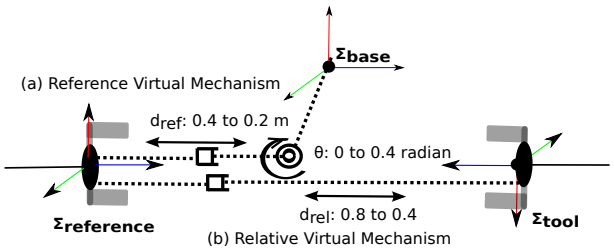


Fig. 4. Virtual mechanisms used for grasping and rotating the steering wheel using two arms.

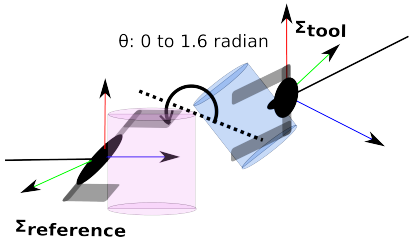


Fig. 5. Virtual mechanism used to define bimanual beans pouring task.

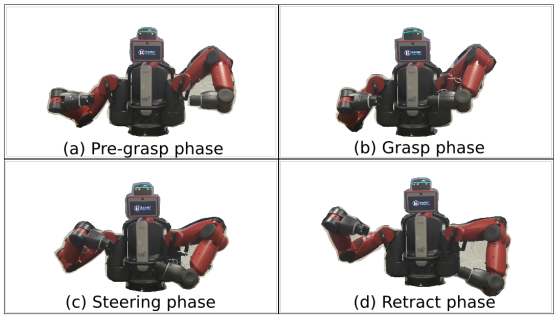
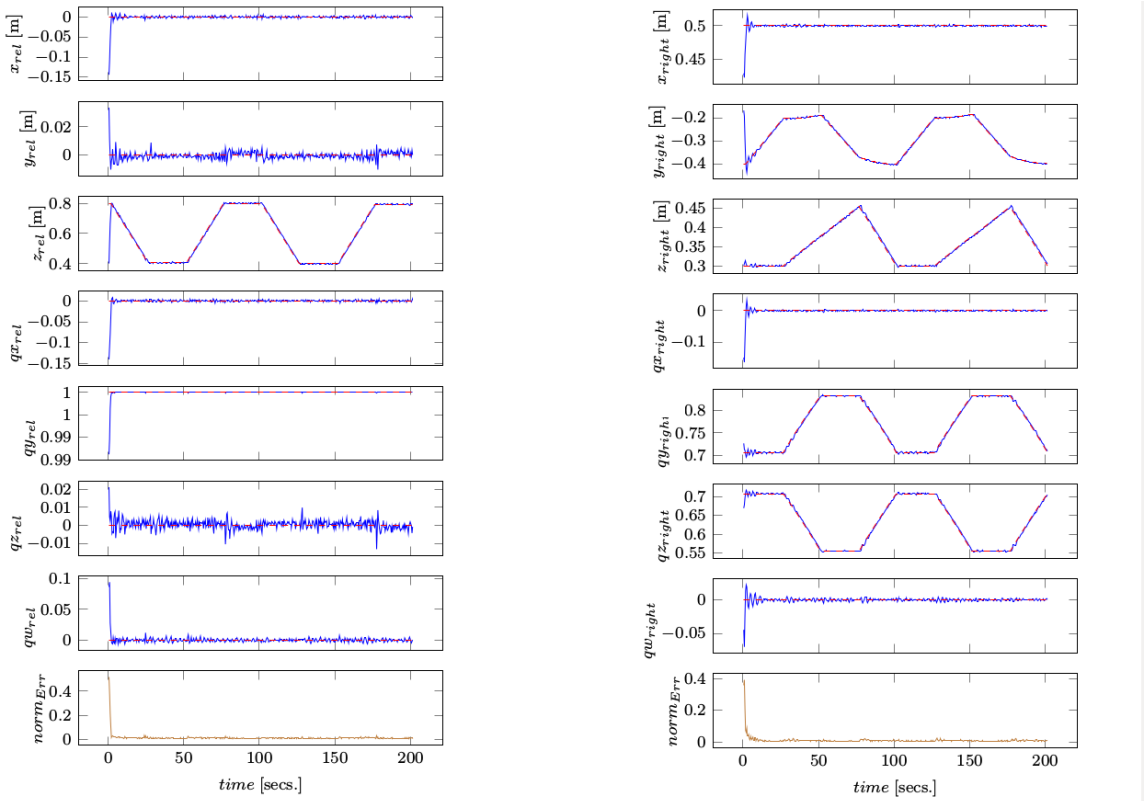


Fig. 6. Baxter performing steering wheel rotation task.

In order to demonstrate the validity of the proposed kinematic formulations and the virtual mechanism based trajectory generator, we designed two different tasks: steering wheel rotation; and bimanual beans pouring task.

A. Steering wheel rotation task

The formulation of the steering wheel rotation task in the context of this paper deals only with the kinematic aspect of the problem. The authors acknowledge the need of at least a static analysis to successfully complete the task because even a small kinematic error can lead to high internal force on the grasped steering wheel. In order to successfully accomplish this task, the reference manipulator end-effector ($\boldsymbol{\Sigma_r}$) had to rotate with respect to the base frame ($\boldsymbol{\Sigma_b}$), and at the same time it was required to keep the tool end-effector ($\boldsymbol{\Sigma_t}$) at a constant relative pose with respect the reference end-effector ($\boldsymbol{\Sigma_r}$). Additionally, we also formulated grasping and retracting motion for both the manipulators using two prismatic joints, in *reference* and *relative* task space. Fig. 4 provides the virtual mechanisms for both relative as well as reference task space to emulate



(a) Relative Task Space Performance (b) Reference(Right) Task Space Performance

Fig. 7. Performance for steering wheel rotation task. - - - represents desired trajectory, whereas ——— represents actual trajectory.

the steering wheel rotation motion. *RightRelative* control mode was used for the trajectory tracking for this task and execution of task by Baxter robot is illustrated in Fig. 6.

B. Bimanual beans pouring task

The goal of this experiment is to pick up two glasses which are placed on a table in front of the robot and to pour the content of one of them to the other one. To accomplish the task, it is required that one of the arms holding the glass containing beans rotate along the central axis of another cup held by other robot (see Fig. 5). In order to provide perception capability, an object recognition and tracking process was implemented in addition to a simple grasping strategy. This experiment uses two AR10 robotic hands ([13]), attached to each of the arms and an external RGB-D sensor to get visual information. The complete operation required *Left*, *Right* and *RightOrientationRelative* control modes. In fact, the task is quite challenging to achieve using *RightRelative* mode due to limited workspace of the robot, and the absence of a strategy to use redundancy of the manipulators, which is outside the scope of this implementation. *RightOrientationRelative* control mode provides the robot the flexibility to change the position of the right arm, without changing the orientation, and thus we were able to complete the task. The virtual mechanism used to generate task trajectory is depicted in Fig. 5.

Object recognition and tracking process. KLD-sampling adaptive particle filter [5], [6] built in the Point Cloud Library's [14] was used to track the objects in the scenario during the manipulation task. KLD-sampling is a stochastic approach to increase the efficiency of the particle filter tracker. To initialize the pose of each tracked object the tabletop object recognition pipeline is used, which find the points cluster of objects previously trained, i.e. stored objects in a database with an associated object reference frame, centred on the bottom of the object. This pipeline consists of two operations: segmentation; and detection. The first one detects the table by finding the dominant plane in the points cloud using RANSAC, then, points above the table are considered to belong to graspable objects. The second one uses an iterative fitting technique (ICP) to see how well it corresponds to each cluster an object stored in the database, and if a good fit is found, the cluster is labelled as the detected object. Note that now, the reference frame of the cluster, that is its pose, is the same that for the recognized object. Once initialized, KDL-sampling tracker returns a pose for each object over the sequence. In this work the grasping points are defined as fixed points on the trained objects, and grasping posture were manually configured. Fig. 1 depicts the different stages of the task performed by the robot.

C. Results

The controller was manually tuned to get a stable execution of trajectory for the experiments. The trajectory

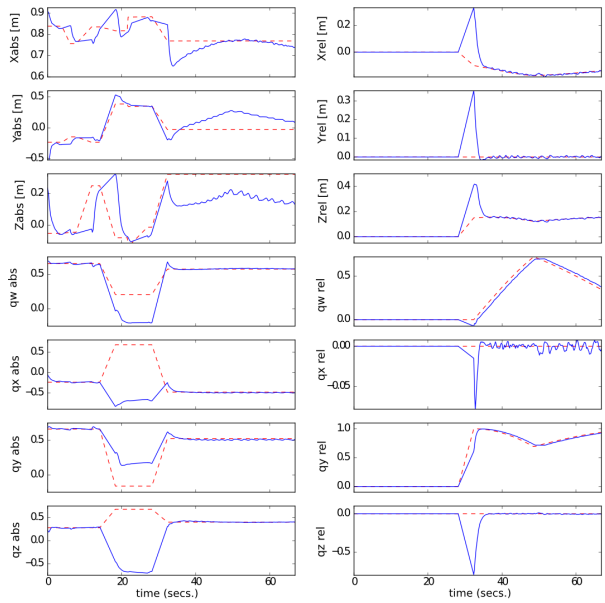


Fig. 8. Performance for beans pouring task. - - - represents desired trajectory, whereas ——— represents actual trajectory.

generated and the response of the trajectory controller for both relative and reference task space, for steering wheel rotation task is presented in Fig. 7. The root mean squared error and the standard deviation of the position for the entire operation is given in Table I. The root mean squared error is highest along the z axis in the relative task space, in the order of millimetres and the controller performance demonstrates an standard deviation in the order of $10^{-2}m$ for both relative and reference task space.

The performance of the vision based framework for beans pouring task is given in Fig. 8. While the errors are considerably higher than steering wheel rotation task, the task was successfully completed. Apart from the added load to the manipulators (AR10 hand and glass full of beans) without calibrating the arm, faster operation of the arm also contributed to the error.

TABLE I

CONTROLLER PERFORMANCE FOR STEERING WHEEL ROTATION TASK

Controller Gain, $\lambda = 4$		Relative	Reference
Root mean squared error [m]	X	6.3998e-04	5.3517e-04
	Y	4.1589e-04	-5.0470e-04
	Z	0.0018	-3.8197e-04
Standard Deviation [m]	X	0.0089	0.0053
	Y	0.0028	0.0154
	Z	0.0231	0.0030

VI. CONCLUSION AND FUTURE WORK

The trajectory tracking performance demonstrated the effectiveness of the proposed framework using unit dual quaternion expression of screw kinematics for dual-arm coordination along with the exponential convergence of the norm of the screw error. The complete system was implemented as a C++ library for general dual-arm systems, and is compatible with ROS.

The next step envisioned, in the context of this work, is to compare this method with the performance of another dual quaternion based implementation available in literature ([11]). Additionally, we would like to extend the framework to include the absolute Jacobian, which can be used for the complete description for cooperative task space. In addition to that, we plan to use to use the null space of the Jacobian to perform additional task, for examples keeping the joint positions of the two cooperating arms within the joint limits, etc. Finally, an extended cooperative task space formulation based on [1] is desired to complete the kinematic description for dual-arm cooperation using unit dual quaternion representation of screw motion.

ACKNOWLEDGMENT

This work is funded by the European Union. The European Union gets involved in Auvergne-Rhône-Alpes through the European Regional Development Fund (FEDER).

REFERENCES

[1] H. A. Park and C. G. Lee, "Extended cooperative task space for manipulation tasks of humanoid robots," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 6088–6093.

[2] E. Özgür and Y. Mezouar, "Kinematic modeling and control of a robot arm using unit dual quaternions," *Robotics and Autonomous Systems*, vol. 77, pp. 66–73, 2016.

[3] E. Sariyildiz and H. Temeltaş, "A new formulation method for solving kinematic problems of multiarm robot systems using quaternion algebra in the screw theory framework," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 20, no. 4, pp. 607–628, 2012.

[4] R. Guenther and D. Martins, "Screw-based relative jacobian for manipulators cooperating in a task," in *ABCM Symposium Series in Mechatronics*, vol. 3, 2008, pp. 276–285.

[5] D. Fox, "Kld-sampling: Adaptive particle filters," *Advances in Neural Information Processing Systems*, vol. 14, no. 1, pp. 713–720, 2002.

[6] ——, "Adapting the sample size in particle filters through kld sampling," *International Journal of Robotics Research*, vol. 22, no. 12, pp. 985–1004, 2003.

[7] R. S. Jamisola and R. G. Roberts, "A more compact expression of relative jacobian based on individual manipulator jacobians," *Robotics and Autonomous Systems*, vol. 63, pp. 158–164, 2015.

[8] M. Uchiyama and P. Dauchez, "A symmetric hybrid position/force control scheme for the coordination of two robots," in *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*. IEEE, 1988, pp. 350–356.

[9] C. L. Lewis, "Trajectory generation for two robots cooperating to perform a task," in *Robotics and Automation, 1996. Proceedings., 1996 IEEE International Conference on*, vol. 2. IEEE, 1996, pp. 1626–1631.

[10] J. Lee, P. H. Chang, and R. S. Jamisola, "Relative impedance control for dual-arm robots performing asymmetric bimanual tasks," *IEEE transactions on industrial electronics*, vol. 61, no. 7, pp. 3786–3796, 2014.

[11] B. V. Adorno, "Two-arm manipulation: from manipulators to enhanced human-robot collaboration," Ph.D. dissertation, Université Montpellier II-Sciences et Techniques du Languedoc, 2011.

[12] "Baxter collaborative robots for industrial automation — rethink robotics," http://www.rethinkrobotics.com/baxter/, (Accessed on 09/12/2017).

[13] "Ar10 humanoid hand. https://www.active8robots.com/robots/ar10-robotic-hand/." [Online]. Available: https://www.active8robots.com/robots/ar10-robotic-hand/

[14] R. B. Rusu and S. Cousins, "3d is here: point cloud library," in *IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1—4.