Application of Dual Quaternion for Bimanual Robotic Tasks

Rohit Chandra

UNIVERSITÉ CLERMONT AUVERGNE
École Doctorale Sciences pour l'Ingénieur
Institut Pascal

Thèse

Présentée par

Rohit Chandra

pour obtenir le grade de
Docteur d'Université

Application of Dual Quaternion for Bimanual Robotic Tasks

Date de la soutenance: 09 Décembre 2019

Devant le jury composé de :

Véronique Perdereau	Examinatrice	Professeur, Sorbonne Université
Omar Tahri	Rapporteur	Maître de conférences, HDR, INSA Centre Val-de-Loire
Philippe Fraisse	Rapporteur	Professeur, Université de Montpellier
Juan CORRALES RAMÓN	Encadrant	Maître de conférences, SIGMA
Youcef MEZOUAR	Directeur de thèse	Professeur des universités, SIGMA

Declaration of Authorship

I, Rohit Chandra, declare that this thesis titled, 'Application of Dual Quaternion for Bimanual Robotic Tasks' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____

Date: _____

Abstract

The classical approach for dual-arm cooperative task space control was revisited and the symmetric formulation of dual arm coordination using *virtual stick*s was implemented using screw-based kinematics with dual quaternion representation. The proposed simultaneous pose control of cooperative task space, i.e. simultaneous control of both position and orientation setpoints of relative and absolute task space was compared against the performance of conventional proportional controller treating position and orientation error separately. The simultaneous pose controller demonstrated better tracking of pose and orientation in terms of accuracy and stability compared to the conventional controller for tasks requiring faster operation in the relative task space of dual-arm manipulators.

The cooperative task space modelling and control approach using screw-based kinematics and dual quaternions were extended for the cooperation modelling of the fingers of an anthropomorphic robotic hand. Additionally, the coupling of joints in the underactuated fingers of the robotic hand were represented with a coupled finger Jacobian. The coupled Jacobian of the robotic finger was used for inverse kinematic control, while allowing easy integration with a robotic arm.

The idea of simultaneous treatment of position and orientation variables was capitalized further with the design of a dynamic trajectory tracker using dual quaternions. The trajectory controller hence designed was capable of tracking pose, velocity and acceleration setpoints for the end-effector using inverse dynamic model of the robot. The dynamic trajectory tracker using a simultaneous resolved rate acceleration controller was found to be capable of tighter trajectory control, specially for error terms related to orientation, compared to the conventional controller that treated the position and orientation setpoints separately and ignored the inherent effect of rotation on translational motion. Additionally, it also led to lower oscillations in the joint torque command when implemented for the control of one of the arms of Baxter dual-arm robot.

Finally, a complete framework for the coordination of bi-arm robotic systems was proposed with the addition of a cooperative task planner. The simplicity of screw theory was exploited additionally for parametrized generation of generalized second order trajectories for tasks requiring simplified motion, like translation, rotation and screw motion around an arbitrary 6D screw-axis given in a known reference frame. The trajectory generation method was extended to represent the constraints related to tasks involving contact between objects using the concept of virtual mechanism.

Acknowledgements

I want to thank my thesis director Professor Youcef Mezouar and co-supervisor Dr. Juan Antonio Corrales Ramón for giving me the opportunity to conduct this doctoral thesis under their supervision. The presented work is the product of Dr. Juan's vision and Prof. Youcef's guidance. I would like to thank them for their unwavering support, encouragement and patience throughout the duration of this thesis.

I would also like to express my gratitude to my colleague Dr. José Sanchez for all the intellectual and philosophical discussions related to robotics, politics and life, in general. I am also appreciative of Dr. Miguel Aranda, who was always available for academic discussions and clearing my theoretical doubts. Additionally, I am also thankful my other contemporary colleagues in Institut Pascal: Dr. Rawan Kalawoun, Dr. Kamal Mohy el Dine, Laurent Lequievre, and Dr. Carlos M. Mateo, to name a few, for all the cheerful and gratifying moments we shared during the last three and a half years together.

Finally, I would like thank my family members: my parents, Shanti Singh and Tanik Prasad Singh, for their unconditional love and support; my cousin Chandan Sharma, who motivated and helped me develop as an individual; and most importantly, my brother Rahul Chandra, who has been an absolute tower of strength throughout my life.

⋆ ⋆ ⋆

This work is funded by the European Union. The European Union gets involved in Auvergne-Rhône-Alpes through the European Regional Development Fund (FEDER). I would like to thank these institutions for the financial support that made this thesis possible.

Contents

List of Figures

List of Tables

Abbreviations

CTS *Cooperative task-space*

HTM *Homogeneous Transformation Matrix*

DH *Denavit-Hartenberg*

UDQ *Unit Dual Quaternion*

DQ *Dual Quaternion*

PD *Proportional Derivative*

VM *Virtual mechanism*

VS *Virtual stick*

SME *Small and medium-sized enterprise*

D *dimensional*

dof *degrees of freedom*

EE *end-effector*

KDL *Kinematics and Dynamics Library*

WTM *wrench transformation matrix*

Notations

- $\hat{\boldsymbol{z}}$: Dual Quaternions, Dual Vectors, or 6D arrays
- $\boldsymbol{z}$: Quaterions and Vectors
- $\hat{z}$: Dual numbers
- $\underline{\hat{z}}$: Dual Number Array
- $\underline{\hat{\boldsymbol{z}}}$: Dual Quaternion or Dual Vector Array
- $\underline{\boldsymbol{z}}$: Quaternion or Vector Array
- ${}^{k}\hat{\boldsymbol{x}}_{i,j}$: Pose of frame j relative to frame i and represented in frame k.
- ${}^{i}\hat{\boldsymbol{x}}_{j}$: Pose of frame j relative to frame i and represented in frame i.

Chapter 1

Introduction

1.1 Motivation

In the past decades, robots have demonstrated great applicability in manufacturing industries for manipulation of heavy parts for assembly, welding and machining operations. Since the focus of previous robotic applications was geared towards high-volume manufacturing, a structured environment with dedicated robotic cells was the prevailing norm in these manufacturing units. But as the market's demand is shifting from mass production towards mass customization [15], as shown in Fig. 1.1, industries are forced to redesign the production lines more frequently. Additionally, manual labor is often needed for intermediate tasks like fastening two parts together, inspection of manufactured parts for defects, *etc.*

To stay relevant in the industries, the robots should evolve to be more skilled with higher cognition abilities, flexible in terms of applicability for different kinds of tasks, and safe enough to work with humans. It is expected that 30% of the robots sold in 2027 will be collaborative robots [16], or as commonly known, cobots, which are designed for safe interaction and collaboration with humans. The take-away message from these trends is that the key drivers of automation in the near future will be the robotics solutions for tasks that are traditionally done by humans, while capable of working close to humans and collaboration with humans in an human-centric environment [17].

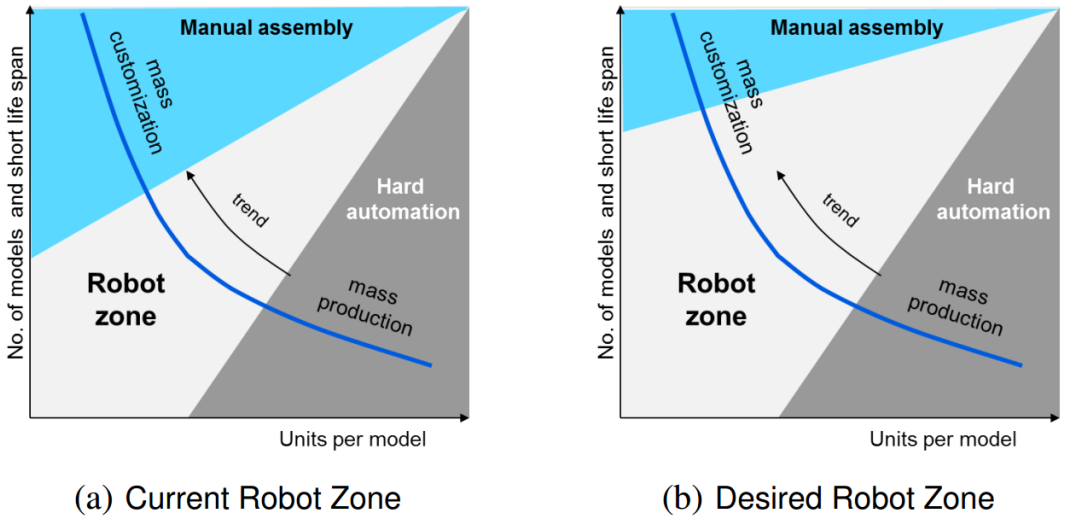


FIG. 1.1 Shift of "Robot zone" towards mass customization zone where manual assembly was prevalent [1].

The robotics community has started to respond to this emerging demand of high mix and low volume production in industries with the development of highly skilled collaborative robots capable of working alongside humans for traditionally manual tasks, such

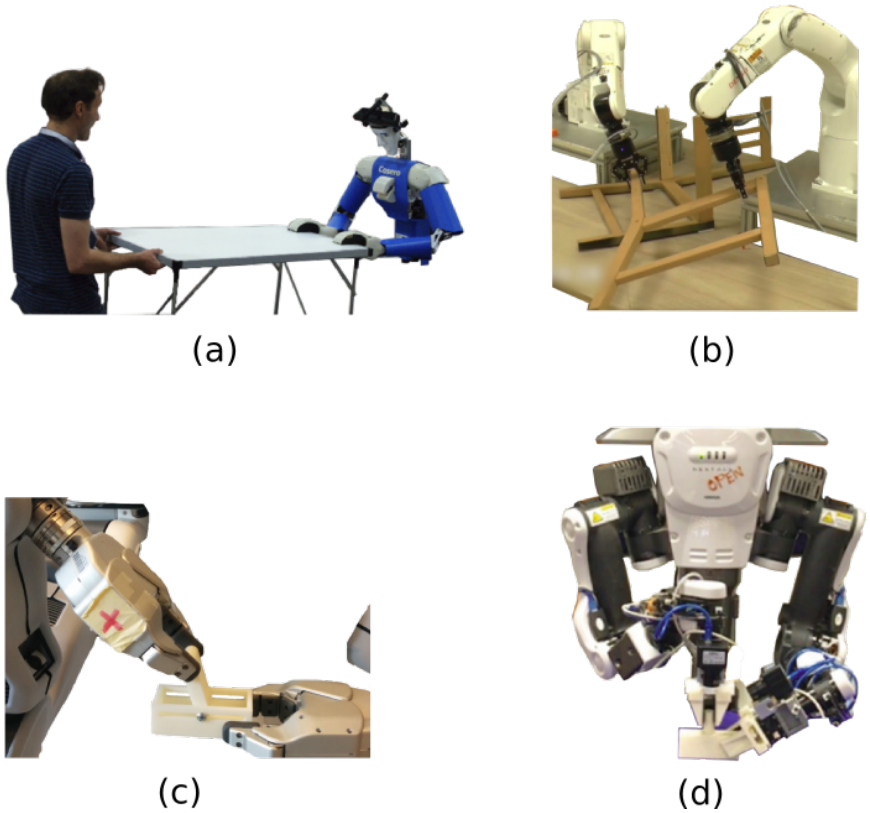


FIG. 1.2 Dual-arm collaborative robots performing industrial tasks: (a) cooperative lifting with human [2]; (b) dual-arm IKEA chair assembly [3]; (c) bimanual folding assembly [4]; (d) bimanual peg-in-hole task [5].

as packaging of goods, small parts assembly, *etc.* Dual-arm robots have been used for cooperative tasks such as: manipulation of heavy objects [18]; bimanual assembly operations [19, 20]; laboratory automation [21, 22]; and, space exploration crew-assistance [23], etc. Some of the recent results in the context of these application scenarios of advanced industrial automation using robots have been depicted in Fig. 1.2.

There is also a significant potential of robotic applications for domestic tasks in a human-centric environment such as service industries, like restaurants and hotels, in addition to medical facilities. Dual-arm robotic platforms can be used for domestic applications like making pancakes [24], and house-keeping tasks such as making a bed, folding laundry, ironing clothes, *etc* [6, 7]. However, one of the most crucial service application of a robot could be taking care of patients and the elderly. Robots can help patients with immobility ease into their wheelchair or to put them to bed or help them stand by providing physical support [25]. They can also be used to dress and undress patients with mental and physical incapabilities [8, 9]. Fig. 1.3 shows some of the robotic solutions for common household tasks, like folding laundry and clothing assistance tasks.

The applicability of dual-arm robots in manufacturing industries, as well as in domestic tasks is undeniable on account of following factors [26, 27]:

- Industrial Applications:
  - *Fixture-less assembly*: Assembly operation involves controlled interaction between two assembly parts. A single manipulator needs a structured fixture to hold one of the parts thus greatly reducing the applicability of manipulators in industries. A dual-arm manipulator can eliminate the requirement of a dedicated setup thus has the potential of increasing the usage of robots in *Small and medium-sized enterprise* (SME)s.
  - *Larger workspace and higher load-bearing capacity*: A dual-arm system enlarges the workspace. Moreover, there are many tasks in industrial settings that involve the manipulation of heavy objects and their interaction with the external environment where dual-arm robots can be of higher utility than single-arm robots.
  - *Flexibility in the operation and task-space*: Sometimes even when the desired tasks can be accomplished with just one arm, the other arm can provide

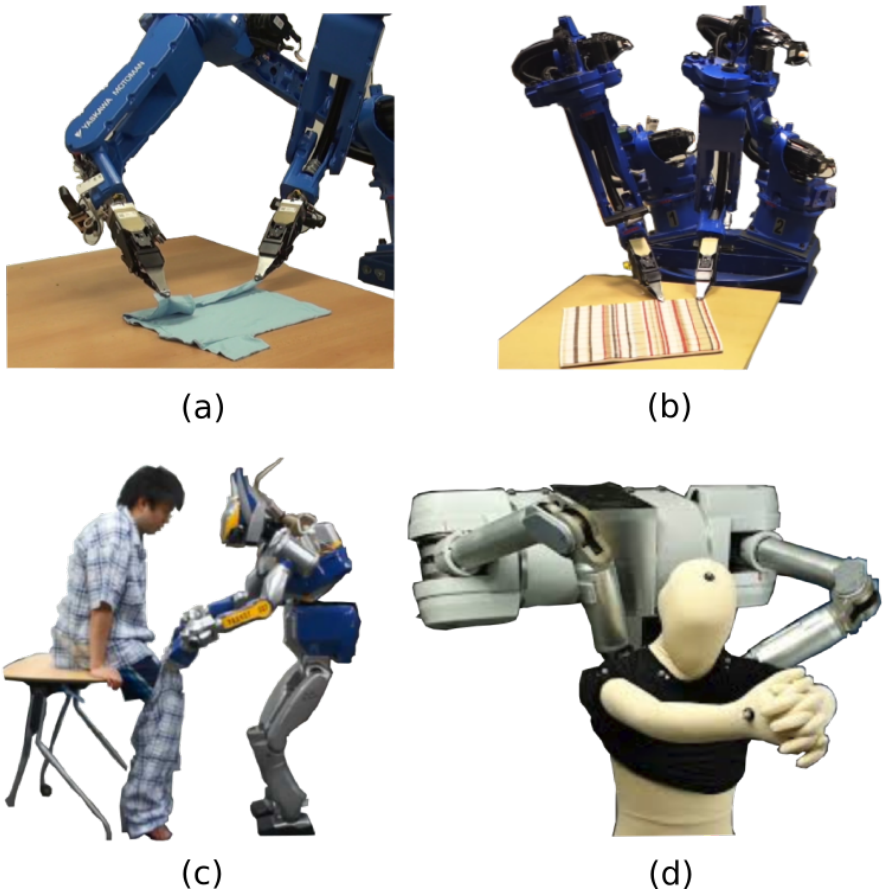


FIG. 1.3 Dual-arm robots performing domestic and assistance tasks: (a) folding clothes [6]; (b) flattening clothes or wrinkle removal [7]; (c) bottom dressing assistance [8]; and, (d) upper body clothing assistance [9].

more flexibility of repositioning of objects, or can bring a camera closer to the task for visual feedback for robust completion of tasks. Additionally, the performance of the task operation can be improved by regrasping and exchanging objects between arms.

- Human-centric applications:
  - *Human form-factor*: The human form-factor of dual-arm robots makes them an ideal choice to employ in service industries owing to the trust factor and thus making dual-arm robots easier to integrate with human-structured environment [28].
  - *Transfer of skills*: Since we humans perform many tasks that involve a combination of single-arm and bimanual operations, a dual-arm robot is more desirable, as human inspired strategies can be easily imitated or transferred [29, 30].

1.2 Scope of the thesis

While dual-arm robots offer a considerable advantage for most of the industrial and domestic applications, they inherit all the single manipulator challenges related to the task description, the abstraction of higher-level inferences from raw sensor data and sensor inaccuracies, uncertainties related to the environment, motion planning and control of the manipulator, and so on. On top of that, dual-arm manipulation presents some new problems such as dual-arm coordination, dual-arm motion planning, role-assignment for each contributing arms, *etc* [26].

This thesis deals with some of the issues related to dual-arm coordination for tasks where cooperative behavior of two arms of a dual-arm robot is paramount. The goal of this thesis has been to apply the principles of screw theory using dual quaternions for the representation of motion variable and spatial transformations, and to improve upon the previous works on cooperative dual-arm manipulation in terms of computational advantage and controller performance. The improvements in computational and storage efficiency can be attributed to the use of UDQ, and controller performance to the consideration of the inherent coupling of linear and angular terms in rigid body motion. The improvement in control performance observed for cooperative task-space control was extended for second-order trajectory tracking for single robotic manipulator.

While this thesis does not address the issues involved in robotic grasping, the kinematic aspects of an anthropomorphic hand that incorporates coupled motion in the last two distal phalanxes similar to human fingers were studied. A coupled Jacobian was proposed which represented this motion coupling and the forward and inverse kinematic model was validated on robotic fingers. Additionally, the concept of relative task-space was applied to control the relative task-space of a unified manipulator consisting of index and thumb, with two end-effectors frames defined at the two fingers tips.

A framework for task definition, trajectory planning and execution for dual-arm manipulation was designed. Screw theory and dual quaternions were used as the tools to define cooperative task-space variables, which refers to the absolute and relative task space of dual-arm robots. In addition, screw-theory were employed to obtain parametrized cooperative task-space trajectories for common industrial and domestic tasks. The task definition approach uses VMs to define physical constraints related to tasks. The goal of

using VM is to parametrize the task definition with task requirements which are either pre-specified or can be obtained from an appropriate sensing module.

1.3 Contributions

The main contribution of this thesis are presented in the following sections.

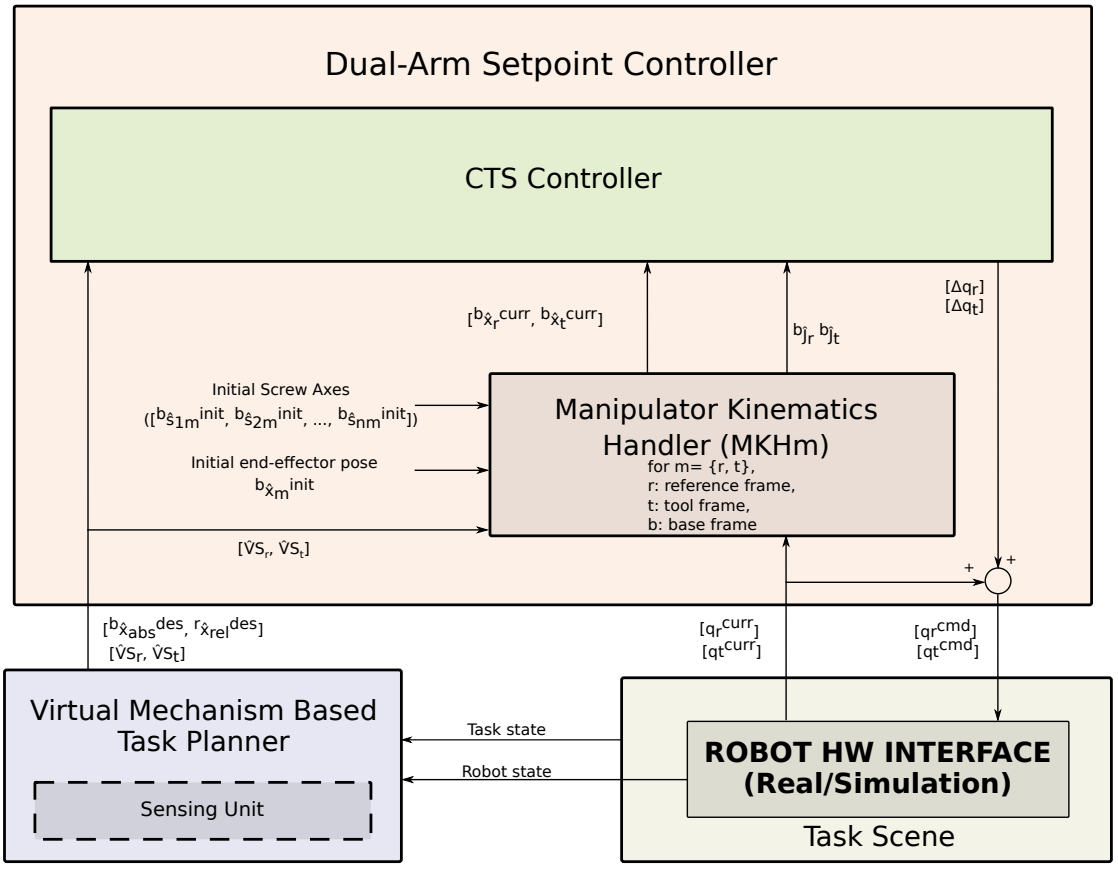


FIG. 1.4 Control architecture for CTS pose control.

1.3.1 Framework for Dual-arm cooperative task execution

A framework for dual-arm cooperative task execution has been proposed that includes a novel formulation of dual-arm coordination using unit dual quaternions, and a task planning strategy for tasks involving the manipulation of rigid, well as articulated and deformable object manipulation. The proposed strategy was validated on Baxter robot for a variety of tasks such as cooperative lifting, deformation of foam for packaging and

manipulation of isometric linear objects like rope. The framework shown in Fig. 1.4 consists of following modules:

1.3.1.1 Dual-arm cooperative task-space formulation

The modelling of dual-arm coordination tasks was remodeled by capitalizing on the efficiency and compactness afforded by unit dual quaternions for pose representation while avoiding representation singularities. The proposed work also takes advantage from the simplicity of screw theory for kinematic modelling, and its capability to represent the natural motion of rigid bodies where linear and angular movements are inherently coupled. The proposed method was compared against the traditional control approach where the pose variables related to linear and angular motion are separately controlled.

1.3.1.2 Cooperative task-planner

A task planner has been proposed for generating motion plans in the cooperative task space, with the a variation of VS concept proposed in [31], that was used for static analysis of a rigid body held by two manipulators. The VS concept was modified for the task modeling of articulated and deformable object manipulation. The implementation of the task planner is based on screw theory that allowed us to parametrize these tasks with minimal parameters, while at the same time generate higher-order trajectories. A comparative analysis of screw theory and *Dual Quaternion* (DQ) based pose controller was carried out for tasks hence defined against the classical control approach involving three-dimensional position and orientation components of motion.

1.3.2 Anthropomorphic finger modeling and control

An anthropomorphic hand was studied that consisted of fingers with human-like mechanical coupling in their middle and distal phalanx motion. The kinematic model was derived analytically while taking this coupling into account, in a way so as to make it easier to integrate with the kinematics of the finger with that of a connected robotic arm. The kinematic model hence derived was validated on an anthropomorphic hand for forward and inverse kinematic control of individual fingers, as well as for relative task-space control of relative motion between thumb and index-finger fingertips.

The inclusion of the relative task space of finger with the cooperative task-space framework has also been proposed and the corresponding architecture has been outlined in the future work section.

1.3.3 Coupled resolved-rate acceleration controller

The trajectory tracking strategy of manipulators using resolved rate acceleration control was studied and a controller was designed which respected the inherent coupling of angular and linear terms in rigid body motion. The proposed trajectory tracker uses a *Proportional Derivative* (PD) control scheme based on spatial dynamics and unit dual quaternion representation of motion variable *viz.* pose, velocity and acceleration. The novel trajectory tracker for robotic manipulation hence designed was compared with a state of the art decoupled controller, where the linear and angular terms of the control law are controlled separately.

A framework for extending the coupled resolved-rate acceleration control for dual-arm handling of a rigid object has also been proposed in the future work section.

1.4 Funding

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

1.5 Publications

The research work conducted in the scope of thesis were disseminated through following conferences:

- Chandra, R., Corrales-Ramon, J. A., & Mezouar, Y. (2019, April). Kinematic modeling of an anthropomorphic hand using unit dual quaternion. In *2019 IEEE/SICE International Symposium on System Integration (SII)* (pp. 433-437). IEEE.

- Chandra, R., Mateo, C. M., Corrales-Ramon, J. A., & Mezouar, Y. (2018, December). Dual-Arm coordination using dual quaternions and virtual mechanisms. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)* (pp. 759-765). IEEE.

- Chandra, R., Corrales-Ramon, J. A., & Mezouar, Y. Resolved-acceleration control of serial robotic manipulators using unit dual quaternions. Manuscript submitted in invited track in *IFAC (International Federation of Automatic Control World Congress)*, 2020, Berlin.

- Chandra, R., Corrales-Ramon, J. A., & Mezouar, Y. (2019). Dual-quaternion based framework for dual-arm manipulation: from coordinated motion control to task planning. Manuscript submitted in *IJRR (International Journal of Robotics Research)*.

1.6 Outline of the thesis

Chapter 2 presents the dual-arm coordination problem in detail and explains the advantages of the proposed dual quaternion based approach compared to state of the art approaches. The comparative validation for different simple cooperative task-space motion of the dual-arm manipulators is also presented in chapter 2. The kinematic modelling and relative task-space control strategy of an anthropomorphic hand is given in chapter 3. The task planning strategy for dual-arm robots, along with experiments with deformable objects has been given in chapter 5.

The new strategy proposed for resolved-rate acceleration control design for trajectory tracking of end-effector, along with the theoretical foundation, related work and comparative validation with state of the arm approaches is described in chapter 4. Future work and perspective and a general conclusion for the entire thesis work is presented in the last chapter, *i.e.* chapter 6.

Chapter 2

Dual-Arm Kinematic Coordination using Dual Quaternion

The treatment of dual-arm coordination available in the literature can be divided into symmetric and asymmetric schemes, where in symmetrical schemes of dual-arm manipulation it is desired to have shared contribution of force and motion from both arms. In asymmetric schemes the pose and desired interaction force setpoints for the master manipulator are used to compute the corresponding variables for the slave arm [32, 33, 34]. Some bimanual tasks demand master-slave control, for example hammering a nail where one of the arms position the nail on the wall, while other handles the hammer. However, for situations where asymmetry is not apparent from the task, it is desired to formulate the cooperation problem symmetrically to have better control over the task and use it to our advantage to modify the cooperative behavior [35, 36]. The reasons for this choice are issues like artificially imposed asymmetry [37], information delay [38], and proper master/slave role assignment during the execution of task [31] with asymmetric coordination scheme.

Uchiyama [31] provided one of the earliest symmetric formulations to deal with dual-arm coordination considering both arms collectively. Their VSs based static analysis of bimanual manipulation of rigid objects provides a good starting point to study dual arm coordination, since it also provides a sound basis to account for different kinematic

cooperative models derived from the study of human bimanual actions [39] to impart different level of contribution to the cooperating robotic arms [35, 40]. In this work however, we only focus on bimanual actions where the cooperating arms are desired to contribute equally, i.e. on parallel bimanual actions.

A screw-based approach using dual-quaternion representation has been presented in this chapter for dual-arm kinematic modelling. One of the goal of this study is to compare such approach with existing methods using other CTS modelling approaches such as DH and DH with DQ representation. Another objective for this work was to analyze the difference in the dual-arm cooperative motion control performance when the position and orientation variables are controlled separately, against the proposed simultaneous pose control law that use screw theory to handle both positional and orientational variables simultaneously.

In following section (section 2.1) the cooperative task space proposed by [31] is briefly presented, which will aid in understanding between different approaches taken in the literature for dual-arm cooperation modelling and control. The related works in the direction of CTS modelling and control is presented in section 2.1, and the motivation and contributions related to this work is presented in section 2.2.2. Based on [31] static analysis of dual arm cooperation, a new method for cooperative task-space modelling is proposed in section 2.3 that uses DQ representation of screw variables of motion, i.e. pose and velocity. The proposed cooperative task-space control architecture and its comparative validation against existing models along with corresponding experiments is given in 2.4. Finally, the conclusion of this work is presented in section 2.5.

2.1 Cooperative task space

Consider two robotic arms equipped with grippers or hands holding an object as shown in Fig. 2.1. Robotic hand attached to *arm 1* exerts force ${}^{o}\boldsymbol{f}_{h1}$ and moment ${}^{o}\boldsymbol{n}_{h1}$ on the object, measured at the origin of EE frame (or force sensor frame) ${}^{o}\boldsymbol{\Sigma}_1$ and given in the base frame $\boldsymbol{\Sigma}_o$. Similarly, the hand attached to *arm 2* exerts force ${}^{o}\boldsymbol{f}_{h2}$ and moment ${}^{o}\boldsymbol{n}_{h2}$ on the object, which are measured at the origin of its EE frame ${}^{o}\boldsymbol{\Sigma}_2$.

For the static analysis VSs can be extended from the origin of both the EEs to the centre of mass of the object, ${}^{o}O_a$, denoted by vectors ${}^{o}\boldsymbol{l}_{h1}$ and ${}^{o}\boldsymbol{l}_{h2}$. Now the forces and

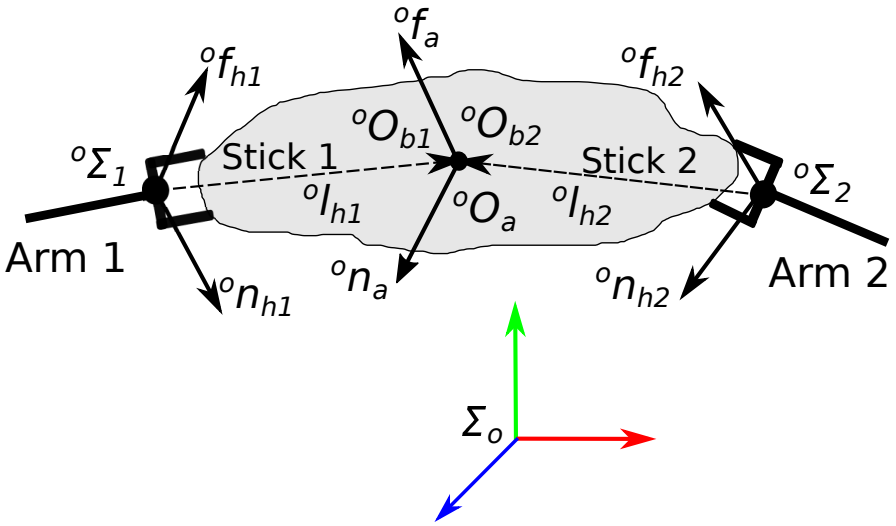


FIG. 2.1 Static analysis of dual-arm handling a single object.

moments at the points of contacts of the hands with the object can be projected to the tips of VSs for static analysis of the system, which we will denote with ${}^{o}\boldsymbol{f}_{bi}$ and ${}^{o}\boldsymbol{n}_{bi}$, for arm i, $(i = 1, 2)$:

$$ {}^{o}\boldsymbol{f}_{bi} = {}^{o}\boldsymbol{f}_{hi} \tag{2.1} $$

$$ {}^{o}\boldsymbol{n}_{bi} = {}^{o}\boldsymbol{n}_{hi} + {}^{o}\boldsymbol{f}_{hi} \times {}^{o}\boldsymbol{l}_{hi}. \tag{2.2} $$

If the object is assumed to be infinitely rigid, the resultant wrench at ${}^{o}O_a$ is given as:

$$ {}^{o}\boldsymbol{f}_a = {}^{o}\boldsymbol{f}_{b1} + {}^{o}\boldsymbol{f}_{b2} \tag{2.3} $$

$$ {}^{o}\boldsymbol{n}_a = {}^{o}\boldsymbol{n}_{b1} + {}^{o}\boldsymbol{n}_{b2} \tag{2.4} $$

By employing generalized force/moment 6-*dimensional* (D) vector ${}^{o}\hat{\boldsymbol{f}}_a$ and 12-D vector ${}^{o}\underline{\hat{\boldsymbol{f}}}_b$, the above expression can be succinctly written as:

$$ {}^{o}\hat{\boldsymbol{f}}_a = \mathbf{W}\ {}^{o}\underline{\hat{\boldsymbol{f}}}_b \tag{2.5} $$

where

$$ {}^{o}\hat{\boldsymbol{f}}_a = \begin{bmatrix} {}^{o}\boldsymbol{f}_a^{\mathsf{T}} & {}^{o}\boldsymbol{n}_a^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \tag{2.6} $$

$$ {}^{o}\underline{\hat{\boldsymbol{f}}}_b = \begin{bmatrix} {}^{o}\boldsymbol{f}_{b1}^{\mathsf{T}} & {}^{o}\boldsymbol{n}_{b1}^{\mathsf{T}} & {}^{o}\boldsymbol{f}_{b2}^{\mathsf{T}} & {}^{o}\boldsymbol{n}_{b2}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \tag{2.7} $$

$$ \mathbf{W} = \begin{bmatrix} \mathbf{I}_6 & \mathbf{I}_6 \end{bmatrix}, \tag{2.8} $$

where, $\mathbf{I}_6$, is 6×6 identity matrix.

The general solution for the EE forces being exerted on the end of the VSs can be obtained by inverting 2.5, and is given as:

$$ {}^{o}\underline{\hat{\boldsymbol{f}}}_b = \mathbf{W}^{\dagger}\ {}^{o}\hat{\boldsymbol{f}}_a + (\mathbf{I}_{12} - \mathbf{W}^{\dagger}\mathbf{W})\ \underline{\hat{\boldsymbol{f}}} \tag{2.9} $$

where $\underline{\hat{\boldsymbol{f}}}$ is an arbitrary 12-D vector and $\mathbf{W}^{\dagger}$ is the Moore-Penrose inverse of $\mathbf{W}$.

Six column vectors corresponding to the orthogonally independent bases belonging to the 6-D null space of $\mathbf{W}$, i.e. $\mathcal{N}(\mathbf{W})$, can be chosen from $\mathbf{V} = [\mathbf{I}_6 \ \ -\mathbf{I}_6]^{\mathsf{T}}$, where six orthogonally independent bases from $\mathcal{N}(\mathbf{W})$ are given by ${}^{o}\hat{\boldsymbol{f}}_r$. This choice of null space resolution implies that both the EE applies equal and opposite force and moment at the end of their corresponding VSs, when the system of dual-arm robot and the object attached to the EEs is at static equilibrium. Thus we get following relations for ${}^{o}\hat{\boldsymbol{f}}_a$ and ${}^{o}\hat{\boldsymbol{f}}_r$, i.e. absolute force and internal force acting on the object:

$$ {}^{o}\hat{\boldsymbol{f}}_a = {}^{o}\hat{\boldsymbol{f}}_{b1} + {}^{o}\hat{\boldsymbol{f}}_{b2} \tag{2.10} $$

$$ {}^{o}\hat{\boldsymbol{f}}_r = 1/2({}^{o}\hat{\boldsymbol{f}}_{b1} - {}^{o}\hat{\boldsymbol{f}}_{b2}) \tag{2.11} $$

Another set of cooperative task space velocity 6-D vectors, containing both linear and angular terms can be constructed, consistent with the cooperative workspace force vectors using the principle of virtual work, and is given as :

$$ {}^{o}\hat{\boldsymbol{\omega}}_a = 1/2({}^{o}\hat{\boldsymbol{\omega}}_{b1} + {}^{o}\hat{\boldsymbol{\omega}}_{b2}) \tag{2.12} $$

$$ {}^{o}\hat{\boldsymbol{\omega}}_r = {}^{o}\hat{\boldsymbol{\omega}}_{b1} - {}^{o}\hat{\boldsymbol{\omega}}_{b2} \tag{2.13} $$

where ${}^{o}\hat{\boldsymbol{\omega}}_{b1}$ and ${}^{o}\hat{\boldsymbol{\omega}}_{b2}$ are the velocities of the frames attached to the ends of the VSs pointing towards the object's centre of mass and assumed to coincide at the starting of the motion.

$$ {}^{o}\hat{\boldsymbol{\omega}}_{bi} = [{}^{o}\boldsymbol{v}_{bi}^{\mathsf{T}} \;\; {}^{o}\boldsymbol{\omega}_{bi}^{\mathsf{T}}]^{\mathsf{T}} \tag{2.14} $$

for $i = 1$ and 2.

Similarly cooperative workspace pose variables can be derived upon integration of 2.12, and assuming that the frames attached to the tips of the VSs (${}^{o}\boldsymbol{\Sigma}_{bi}$, for $i = 1$ and 2) coincides at the beginning of the motion:

$$ {}^{o}\hat{\boldsymbol{p}}_{a} = 1/2({}^{o}\hat{\boldsymbol{p}}_{b1} + {}^{o}\hat{\boldsymbol{p}}_{b2}) \tag{2.15} $$

$$ {}^{o}\hat{\boldsymbol{p}}_{r} = {}^{o}\hat{\boldsymbol{p}}_{b1} - {}^{o}\hat{\boldsymbol{p}}_{b2} \tag{2.16} $$

where ${}^{o}\hat{\boldsymbol{p}}_{bi}$, represents the position and orientation of the frames attached the tips of corresponding VSs. ${}^{o}\hat{\boldsymbol{p}}_{a}$ represents the absolute pose of the grasped object, and ${}^{o}\hat{\boldsymbol{p}}_{r}$ corresponds to relative position and orientation of the frames attached to the tips of VSs.

These cooperative task space variables, i.e. absolute and relative poses, velocities, and absolute and internal forces had been used in the past to control the interaction of dual-arm manipulators with the object, as well as to keep the the internal forces within safe limits. While some of the work focuses entirely on the kinematic aspects of coordination with the assumption of perfect position or velocity control, others have taken the modelling and control error into account and used hybrid force/position controller to accommodate the positioning error to keep the interaction forces between object and manipulators within limits.

In the section below, we provide some of the representative work related to the formulation of cooperative task-space, focussing mainly on the kinematic control problem when using symmetrical formulation of dual-arm coordination. The goal is to highlight the gap in the existing approaches that can be filled with screw-theory and dual-quaternion based approach taken in this work and to understand the corresponding implications.

2.2 Related work and contributions

2.2.1 Related Work

The cooperative task space variables, *i.e.*, absolute and relative velocities, were directly defined by [41] for dual arm coordination for tasks where the robot had not yet grasped the object. The velocities of the tips of the VSs were replaced with those of the EEs to obtain the relative and absolute motion during dual-arm manipulation. These cooperative task space variables were used for inverse kinematics algorithm, which would generate joint trajectories corresponding to the desired absolute and relative trajectories.

The new definition of cooperative task-space variables, proposed in [41], was applied in [42] for internal force and position regulation during handling of an object by a dual arm robot. The concept of VSs was replaced with the concept of *grasp matrix* to project the EE forces on to the object. While, conceptual formulations for both *grasp matrix* and VSs are exactly the same for dual-arm robots, *grasp matrix* allows generalization of the cooperative task-space formulation for multiple manipulators (> 2) cooperation for rigid object manipulation. Moreover, unit quaternions were used for orientation representation in order to avoid representation singularities associated with minimal representation methods such as Euler's angles.

While some bimanual tasks demands coordination over both absolute and relative task space, for example in the case of handling a heavy object the absolute position corresponds to the motion of the object and relative motion to rigidly grasp the object, for some tasks only relative motion is enough. All the assembly tasks such as peg-in-hole and screwing tasks, where the two arms of a dual-arm robot are handling each of the assembly parts, can be accomplished with just the relative motion. Such treatment of bimanual coordination where only relative task space affords higher dimension of null space in the relative task-space which can be used to perform additional tasks such as collision avoidance, keeping the joints positions within safety limits, etc. [43].

Relative Jacobian maps the relative velocity between the two EEs of the cooperating manipulators with their joint velocities. Relative Jacobian was formulated in [44] 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 [45] a weighted pseudoinverse of relative Jacobian was used to control the joint torque contribution of each joints to keep it within safe limits, in addition to controlling the relative motion between two manipulators in machining operations. The trajectories for the manipulators were resolved at the acceleration level using a relative Jacobian, where consideration of just the relative motion rather than complete cooperative task-space afforded extra redundancy for the dual-arm system.

Guenther [46] 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, which can be problematic for control due to the associated representation singularity, as well as, is more expensive in terms of computation and storage. Screw theory and unit dual quaternion algebra were used in [47] for forward as well inverse kinematics problem of cooperative manipulation for dual-arm robots, however, the differential kinematics problem was not addressed.

In [43], 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 [36]. The time derivative of the relative pose was used for the derivation of a modular relative Jacobian which revealed a *wrench transformation matrix* (WTM), a function of mutual position of the cooperating arms' EEs. The modular relative Jacobian composed of WTM hence obtained, demonstrated better trajectory tracking accuracy for high relative angular velocity of the EEs.

The cooperative task space representation of robotic bimanual actions was extended in [35, 40] to account for the three kinematic cooperative models, *viz.* orthogonal, serial and parallel actions, studied and categorized in [39] from the study of human bimanual actions. The orthogonal model corresponded to the movement where two arms are mutually independent or uncoordinated, such as using a keyboard and mouse while operating computers. When two arms share a common task with two hands, for instance, lifting a heavy box, it is referred to as parallel mode of cooperation. Lastly, the coordination

mode where two arms have partial dependence, for example hammering a nail on the wall or cutting vegetables, falls under the serial mode. To address these coordination modes in bimanual action, two new coefficients were introduced, namely, balance and coordination coefficients to modify the balance of shared load, and to represent different bimanual coordination models or to activate and deactivate coordination respectively.

The adjustment of balance coefficient, for a single object manipulation, was akin to shifting the centre of mass of the object in the static analysis for dual-arm cooperation in VSs or *grasp matrix* approach. The purpose of introducing these balance and coordination coefficients was to offer more flexibility in relative tasks. While previous works on relative motion control ([44, 36, 43, 46]) used the kinematic redundancy directly using the null space of relative Jacobian of dual-arm robots, extended cooperative task space representation allowed the adjustment of balance coefficient to perform additional tasks such as increasing manipulability in dual-arm manipulation to avoid singularity, while still controlling the absolute cooperative task space.

Adorno [13] used unit dual quaternions for the forward and inverse kinematic control of dual-arm robots, where the quaternions allowed singularity free representation of orientation and unit dual quaternion representation of pose allowed simultaneous control of translation and rotation variables. The kinematic modelling of the dual-arm robot was based on DH convention. In addition to that, many primitive task Jacobians were proposed, such as relative distance control, absolute rotation control and so on, which allowed to efficiently utilize the available degrees of freedom for a desired task, and to use the redundancy for additional tasks.

While the above mentioned works provided an insight into existing work related to dual-arm coordination at kinematic level, in following section we will highlight some of the shortcomings of previous works related to modelling and control aspects and explain how our approach of formulating cooperative task space using unit dual quaternion representation and screw theory can help in addressing those issues. In addition to that, our proposed method is also compared with another instance of cooperative task space formulation using unit dual quaternion ([13]) existing in literature.

2.2.2 Motivation for unit dual quaternion and screw-theory based cooperative task space formulation

2.2.2.1 Issues related to previous formulation of cooperative task space

- *Usage of DH convention for kinematic modelling*: A DH convention based manipulator modelling requires definition of frames of each involved link. Whereas, screw-based kinematics requires only two frames for the entire chain, one at the base of the robot, while other at the tip of the EE. Although, DH parameter based manipulator modelling is fairly established in robotics, screw-theory can greatly simplify the modelling process for general manipulators [46]. In addition to that, the same approach of screw-based modelling can be applied to parallel manipulators, which can be relevant for tasks where a dual-arm robot can be considered a parallel manipulator, for example, two arms rigidly grasping an object. Moreover, the joint variables can be used to represent actual displacement in contrast to DH-based modelling where actual joints states are generally computed using offsets over the DH parameters [48].

- *Relative Jacobian Formulation*: The earlier definitions of relative Jacobian used individual Jacobians of the manipulators, without the consideration of effect of relative angular velocity on relative linear velocity. However, it was revealed that a WTM, function of relative position vector, exists in the formulation of relative Jacobian. The modular Jacobian consisting of WTM was obtained by taking the time derivative of the relative pose computed by taking one of the EEs frame as base frame and a frame on the other EE as tool frame. The corresponding relative Jacboian demonstrated better trajectory tracking accuracy for high relative angular velocity of the EEs [49, 46]. Screw-based formulation of relative Jacobian naturally takes into account this *wrench transformation matrix* and thus can lead to better trajectory tracking for all kinds of relative tasks.

- *Representation singularity*: The earlier works on cooperative task space control of dual-arm robots used rotation matrices for the representation of orientation, where Euler's angles were derived for the control law formulation. While, rotation matrices can be intuitive, it suffers from representation singularities that may lead to instability in the controller.

More recent works ([42, 35, 40]) employed quaternions to overcome the limitations related to unit quaternions, where the vector part of the error quaternion was used for the formulation of control law. While this approach is relatively simple to use and asymptotic stability of such a control has been proved in [42], deriving angle-axis parameters from the error unit quaternion represents more meaningful definition of error and global exponential convergence behaviour of corresponding control law can be proved [14].

- *Conventional Pose control*: Most of previous works for cooperative task space control controlled the position and orientational error terms separately. While this method is easy to implement, it does not consider the natural motion of rigid bodies where linear component is inherently coupled with the angular component of motion. Controlling linear and angular terms separately may lead to unnatural trajectory and is undesired for applications where both orientation and position tracking is equally important[50].

TABLE 2.1 Computation cost comparison between HTMwDH [12], DQwDH [13] and DQwScrew ([14], A.5) based manipulator kinematics, where n is the number of joints in the manipulator.

	Multiplications	Additions	Trigonometric Functions
	Cost of successive transform (transform product) computation		
HTMwDH	36	27	NA
DQwDH	48	40	NA
DQwScrew	48	40	NA
	Cost of forward kinematics		
HTMwDH	$70n - 64$	$48n - 48$	$4n$
DQwDH	$60n - 48$	$44n - 40$	$4n$
DQwScrew	$61n - 48$	$43n - 40$	$2n$
	Cost of Jacobian computation		
HTMwDH	$155n - 64$	$108n - 48$	$8n$
DQwDH	$189n - 48$	$142n - 40$	$8n$
DQwScrew	$157n - 144$	$123n - 120$	$2n$

TABLE 2.2 Comparison of our screw-theory and unit dual quaternion based modelling of CTS compared to previous works.

Previous Works	**Complete CTS modelling** [31]	**Consideration of WTM in relative Jacobian** [36]	**Advantages of screw-based kinematics** [48]	**Advantages of DQ representation** [14, 51, 52, 53]
[43, 54]	✗	✗	✗	✗
[36, 55, 44]	✗	✓	✗	✗
[46, 49]	✗	✓	✓	✗
[35, 38, 31, 56, 42, 41]	✓	✗	✗	✗
[57, 58, 59]	✓	✓	✗	✗
[13, 60]	✓	✓	✗	✓
Proposed method	✓	✓	✓	✓

2.2.2.2 Comparison with DH parameters and unit dual quaternion based formulation of cooperative task space

A novel approach for forward and inverse kinematics using dual quaternion representation was proposed in [60, 13] which used DH parameter based approach for kinematic modelling of dual-arm robots. The proposed approach was effective in addressing the issue of orientation representation singularity and their control law accounted for both position and orientational error simultaneously. However, use of DH based parameters for kinematic modelling resulted in overly complicated definition of manipulator Jacobian, since it used matrix form of dual-quaternion operations. 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.

The pose error term used in [13] for inverse kinematic control 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 [14]. A meaningful expression of dual quaternion error using dual quaternion product was utilized for a single manipulator control in [61]. Using the modified of error unit dual quaternion, an optimal trajectory tracking controller was proposed in [62], albeit for a single manipulator case. However, the screw parameters based definition of control law

proposed in [14] is still more meaningful since screw error terms can be transformed to different frames, and has global exponential convergence behaviour.

Moreover, the computational efficiency of dual quaternions was underutilized in [13], 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 comparison between three methods, i.e DH and *Homogeneous Transformation Matrix* (HTM) based, DQ and HTM based and DQ and screw-theory based method used in this work, for pose transformation, forward kinematics and Jacobian computation of a serial manipulator, in terms of computation cost has been given in Table 2.1 based on the computational analysis given in A.5. HTMwDH based approach is the classical DH based rotation matrix approach, DQwDH corresponds to the dual quaternion and DH parameters based formulation given in [13] and the last one (DQwScrew) corresponds to dual quaternion kinematic formulation based on screw-theory, proposed in [14]. The explanation for the computation of HTM and DQ method based on DH convention based kinematic modelling of manipulators has been given in [13]. A more detailed analytical and empirical analysis of the computational efficiency of our dual-quaternion and screw-theory based approach concerning forward kinematics and Jacobian computation, and inverse kinematics and control is given in [12, 53].

The presented comparisons show that both dual quaternion based approaches requires less arithmetic and trigonometric operations for obtaining product of transforms and for the computation of forward kinematics compared to HTMwDH based method. While DQwDH based method requires a little less multiplications, it require far more trigonometric computations compared to DQwScrew based method used in our work. In terms of Jacobian computation, HTMwDH method requires less arithmetic ($\times$ and $+$) operations than both dual quaternion method based operations. However, the DQwScrew method requires less trigonometric operations than other two methods, even though it requires a bit more arithmetic operations than HTMwDQ based method. Finally, the DQ based has less memory requirement compared to HTM approach, i.e. DQ based representation of pose needs only 8 floating units, compared to 12 floating units needed for HTM based pose representation.

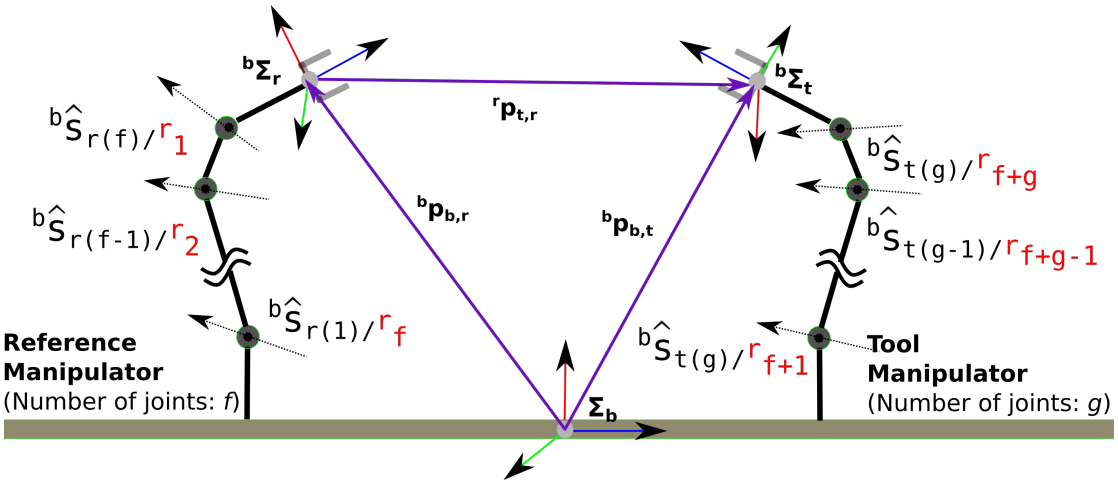


FIG. 2.2 Dual-arm frames and joint axes and virtual joints.

Finally a summary of all the advantage our approach has over other work in the direction of CTS modeling and control has been in Table 2.2.

2.3 Cooperative task space formulation using UDQs

In this section we will use the kinematic formulation for a serial manipulator, developed in [14] and presented in appendix A.5, to formulate the cooperative task space variables related to pose and velocity for dual-arm coordination [31]. A differential kinematic formulation, i.e. relative and absolute Jacobian was also developed to map the joint velocities of the cooperating arms to cooperative task space velocities. The absolute Jacobian has been formulated in the base frame, while the relative Jacobian has been derived in a frame attached to the EE of *reference manipulator*, so as to define the relative task intuitively. If the relative task is defined in the object frame, then the error twist should be transformed from the object to the reference arm's EE frame.

The dual arm set-up along with the different frames used in the following formulations is given in Fig. 2.2. $\boldsymbol{\Sigma_r}$ is the reference frame, $\boldsymbol{\Sigma_t}$ is the tool frame and $\boldsymbol{\Sigma_b}$ is the fixed base frame. The joint screw axes are given as ${}^b\hat{\boldsymbol{s}}_{ri}$ for $i = 1, 2, ..f$, and ${}^b\hat{\boldsymbol{s}}_{tj}$ for $j = 1, 2, ..g$ for the reference and *tool manipulator* respectively, in the base frame. The same screw axes are given as ${}^r\hat{\boldsymbol{s}}_i$ for $i = 1, 2, ..f + g$ for the combined manipulator consisting of the two arms, in the reference end-effector frame.

2.3.1 Relative Jacobian

The relative pose of the tool EE frame $\boldsymbol{\Sigma_t}$ in the reference EE frame $\boldsymbol{\Sigma_r}$ can be computed using the EE poses of tool and *reference manipulator* as:

$$ {}^{r}\hat{\boldsymbol{x}}_t = {}^{b}\hat{\boldsymbol{x}}_r^* \, {}^{b}\hat{\boldsymbol{x}}_t, \tag{2.17} $$

where ${}^{b}\hat{\boldsymbol{x}}_r^*$ is classic dual-quaternion conjugate of ${}^{b}\hat{\boldsymbol{x}}_r$.

The formulation of relative Jacobian, which maps the joint velocities of both the arms to the relative twist of the tool EE with respect to the reference frame, described in the reference frame will follow the same approach discussed in section A.5.1:

$$ \begin{aligned} {}^{r}\hat{\boldsymbol{\xi}}_{t_r} &= \begin{bmatrix} {}^{r}\hat{\boldsymbol{s}}_1 & {}^{r}\hat{\boldsymbol{s}}_2 & \cdots & {}^{r}\hat{\boldsymbol{s}}_{f+g} \end{bmatrix} \begin{bmatrix} \underline{\dot{\boldsymbol{\theta}}_r} & \underline{\dot{\boldsymbol{d}}_r} & \underline{\dot{\boldsymbol{\theta}}_t} & \underline{\dot{\boldsymbol{d}}_t} \end{bmatrix}^T \\ &= {}^{r}\hat{\mathbf{J}} \begin{bmatrix} \underline{\dot{\boldsymbol{\theta}}_r} & \underline{\dot{\boldsymbol{d}}_r} & \underline{\dot{\boldsymbol{\theta}}_t} & \underline{\dot{\boldsymbol{d}}_t} \end{bmatrix}^T, \end{aligned} \tag{2.18} $$

where $\underline{\dot{\boldsymbol{\theta}}_r}$ and $\underline{\dot{\boldsymbol{d}}_r}$ refers to the the general joints velocity array, depending on the type joints for the *reference manipulator*, starting from the joint closest to the reference end-effector. Similarly $\underline{\dot{\boldsymbol{\theta}}_t}$ and $\underline{\dot{\boldsymbol{d}}_t}$ are the joint velocity arrays for the *tool manipulator*, again starting from the joint closest to the reference end-effector.

The screw axes related to the joints will have to be transformed from the base frame to the reference EE frame, $\boldsymbol{\Sigma_r}$, using following relation that transforms a Plücker line, represented as a dual vector:

$$ {}^{r}\hat{\boldsymbol{s}}_i = ({}^{b}\hat{\boldsymbol{x}}_r^*) \, {}^{b}\hat{\boldsymbol{s}}_i \, ({}^{b}\hat{\boldsymbol{x}^*}_r)^* = {}^{r}\hat{\boldsymbol{x}}_b \, {}^{b}\hat{\boldsymbol{s}}_i \, {}^{r}\hat{\boldsymbol{x}^*}_b \tag{2.19} $$

Additionally, either the joint displacements for *reference manipulator* joints, or the corresponding joint axes, have to be taken in negative sense, so as to respect the inverted kinematic chain of the *reference manipulator* used to define a relative task manipulator, which has joint axes ${}^{r}\hat{\boldsymbol{s}}_{t(i)}$ for $i = 1, \ldots, f+g$ (refer the joint screw axes indexing depicted in red in Figure 2.2). In our formulation we have inverted the joint screw of the *tool manipulator* to maintain the traditional representation of relative Jacobian.

Hence the relative Jacobian corresponding relative twist, in the *reference manipulator* frame is given in the matrix form as:

$$\mathbf{J}_{rel} = \begin{bmatrix} -^{r}\boldsymbol{L}_r & \mathbf{0} & ^{r}\boldsymbol{L}_t & \mathbf{0} \\ -^{r}\boldsymbol{M}_r & -^{r}\boldsymbol{L}_r & ^{r}\boldsymbol{M}_t & ^{r}\boldsymbol{L}_t \end{bmatrix}, \tag{2.20}$$

where terms like $\boldsymbol{L}$ and $\boldsymbol{M}$ are arrays of direction vector and the moment vectors of the corresponding manipulator and have meanings similar to A.61.

2.3.2 Absolute Jacobian

All the terms appearing in the following derivation of absolute Jacobian are in the base frame, unless otherwise mentioned. Let ${}^{b}\hat{\mathbf{x}}_r = \boldsymbol{q}_{r\,P} + \epsilon\boldsymbol{q}_{r\,D}$, ${}^{b}\hat{\boldsymbol{x}}_t = \boldsymbol{q}_{t\,P} + \epsilon\boldsymbol{q}_{t\,D}$ and ${}^{b}\hat{\boldsymbol{x}}_{abs} = \boldsymbol{q}_{abs\,P} + \epsilon\boldsymbol{q}_{abs\,D}$ be the unit dual quaternion corresponding to the reference, tool and absolute task space frame, i.e. the frame where the tip of VSs connected to the cooperating manipulators end-effector meet. Let ${}^{b}\boldsymbol{p}_{b,r}$ and ${}^{b}\boldsymbol{p}_{b,t}$ be the position vectors of the reference and tool EEs respectively, which are represented as a pure quaternions. They can be derived from the corresponding poses UDQs as follows:

$${}^{b}\boldsymbol{p}_{b,r} = 2\boldsymbol{q}_{r\,D}\,\boldsymbol{q}^{*}_{r\,P} \tag{2.21}$$

$${}^{b}\boldsymbol{p}_{b,t} = 2\boldsymbol{q}_{t\,D}\,\boldsymbol{q}^{*}_{t\,P} \tag{2.22}$$

where $\boldsymbol{q}^{*}$ is quaternion conjugate of $\boldsymbol{q}$.

Absolute position and orientation is given as ([42]):

$${}^{b}\boldsymbol{p}_{abs} = \frac{{}^{b}\boldsymbol{p}_{b,r} + {}^{b}\boldsymbol{p}_{b,t}}{2} \tag{2.23}$$

$${}^{b}\boldsymbol{q}_{abs\,P} = \boldsymbol{q}_{r\,P}\,{\boldsymbol{q}_{rel\,P}}^{(\frac{1}{2})}, \tag{2.24}$$

where ${\boldsymbol{q}_{rel\,P}}^{(\frac{1}{2})}$ refers to half of the rotation around the same rotation axis corresponding to the relative rotation quaternion ($\boldsymbol{q}_{rel\,P}$) computed in reference frame. The relative rotation quaternion in reference frame is given as:

$${}^{b}\boldsymbol{q}_{rel\,P} = \boldsymbol{q}^{*}_{r\,P}\,\boldsymbol{q}_{t\,P}. \tag{2.25}$$

In order to use the control law using screw parameters of error dual quaternion (see A.73), it is desired to obtain the absolute pose as UDQ. Using relation (2.21) and (2.23), we have

$$
\begin{aligned}
{}^{b}\boldsymbol{p}_{abs} &= \boldsymbol{q}_{r\,D}\,\boldsymbol{q}^{*}_{r\,P} + \boldsymbol{q}_{t\,D}\,\boldsymbol{q}^{*}_{t\,P} \\
\text{or,}\quad 2\boldsymbol{q}_{abs\,D}\boldsymbol{q}^{*}_{abs\,P} &= \boldsymbol{q}_{r\,D}\,\boldsymbol{q}^{*}_{r\,P} + \boldsymbol{q}_{t\,D}\,\boldsymbol{q}^{*}_{t\,P}
\end{aligned}
\tag{2.26}
$$

Therefore,

$$
\boldsymbol{q}_{abs\,D} = \frac{(\boldsymbol{q}_{r\,D}\,\boldsymbol{q}^{*}_{r\,P} + \boldsymbol{q}_{t\,D}\,\boldsymbol{q}^{*}_{t\,P})\boldsymbol{q}_{abs\,P}}{2} \tag{2.27}
$$

The absolute angular velocity in the base frame in the matrix-vector form, using (A.59) is given as:

$$
\begin{aligned}
\boldsymbol{\omega}_{abs} &= \frac{\boldsymbol{\omega}_r + \boldsymbol{\omega}_t}{2} \\
&= \frac{\boldsymbol{L}_r\,\underline{\dot{\boldsymbol{\theta}}_r} + \boldsymbol{L}_t\,\underline{\dot{\boldsymbol{\theta}}_t}}{2}
\end{aligned}
\tag{2.28}
$$

And absolute linear velocity is given as:

$$
\boldsymbol{v}_{abs} = \frac{\boldsymbol{v}_r + \boldsymbol{v}_t}{2} \tag{2.29}
$$

An absolute Jacobian is required to map joint velocities of the dual-arm robot to screw twist corresponding to absolute frame motion, i.e $\{\boldsymbol{v}_{abs0}, \boldsymbol{\omega}_{abs}\}$ pair. Here $\{\boldsymbol{v}_{abs0}$ refers to the linear velocity of a point attached to the absolute frame, currently coinciding with the base frame. Since the screw-based control law has attractive features like global stability and exponential convergence [14], it is desired to obtain the absolute Jacobian in the same form as well. Using (A.67) we have:

$$
\boldsymbol{v}_{abs0} = \boldsymbol{v}_{abs} - \boldsymbol{\omega}_{abs} \times \boldsymbol{p}_{abs}. \tag{2.30}
$$

After expanding (2.30) using (A.67), (2.23), (2.28), (2.29) we get:

$$\begin{aligned}\boldsymbol{v}_{abs0} &= \frac{\boldsymbol{v}_r + \boldsymbol{v}_t}{2} - \frac{\boldsymbol{\omega}_r + \boldsymbol{\omega}_t}{2} \times \frac{(\boldsymbol{p}_r + \boldsymbol{p}_t)}{2} \\ &= \frac{\boldsymbol{v}_{r0} + \boldsymbol{\omega}_r \times \boldsymbol{p}_r + \boldsymbol{v}_{t0} + \boldsymbol{\omega}_t \times \boldsymbol{p}_t}{2} \\ &- \frac{\boldsymbol{\omega}_r + \boldsymbol{\omega}_t}{2} \times \frac{(\boldsymbol{p}_r + \boldsymbol{p}_t)}{2} \\ &= \frac{2(\boldsymbol{v}_{r0} + \boldsymbol{v}_{t0}) + \boldsymbol{\omega}_r \times (\boldsymbol{p}_r - \boldsymbol{p}_t) + \boldsymbol{\omega}_t \times (\boldsymbol{p}_t - \boldsymbol{p}_r)}{4}\end{aligned}$$

Using matrix-vector representation given in (A.60), we have:

$$\begin{aligned}\boldsymbol{v}_{abs0} &= \frac{\boldsymbol{M}_r\ \underline{\dot{\boldsymbol{\theta}}_r} + \boldsymbol{L}_r\ \underline{\dot{\boldsymbol{d}}_r} + \boldsymbol{M}_t\ \underline{\dot{\boldsymbol{\theta}}_t} + \boldsymbol{L}_t\ \underline{\dot{\boldsymbol{d}}_t}}{2} \\ &+ \frac{\boldsymbol{L}_r\ \underline{\dot{\boldsymbol{\theta}}_r} \times (\boldsymbol{p}_r - \boldsymbol{p}_t)}{4} \\ &+ \frac{\boldsymbol{L}_t\ \underline{\dot{\boldsymbol{\theta}}_t} \times (\boldsymbol{p}_t - \boldsymbol{p}_r)}{4}\end{aligned} \tag{2.31}$$

Writing (2.28) and (2.31) in the matrix form, we get:

$$\begin{bmatrix}\boldsymbol{\omega}_{abs} \\ \boldsymbol{v}_{abs_0}\end{bmatrix} = \mathbf{J}_{abs} \begin{bmatrix}\underline{\dot{\boldsymbol{\theta}}_r} & \underline{\dot{\boldsymbol{d}}_r} & \underline{\dot{\boldsymbol{\theta}}_t} & \underline{\dot{\boldsymbol{d}}_t}\end{bmatrix}^T \tag{2.32}$$

where,

$$\mathbf{J}_{abs} = \begin{bmatrix}\frac{\boldsymbol{L}_r}{2} & \mathbf{0} & \frac{\boldsymbol{L}_t}{2} & \mathbf{0} \\ \frac{2\boldsymbol{M}_r + \boldsymbol{L}_r \times (\boldsymbol{p}_r - \boldsymbol{p}_t)}{4} & \frac{\boldsymbol{L}_r}{2} & \frac{2\boldsymbol{M}_t + \boldsymbol{L}_t \times (\boldsymbol{p}_t - \boldsymbol{p}_r)}{4} & \frac{\boldsymbol{L}_t}{2}\end{bmatrix} \tag{2.33}$$

2.4 Implementation of dual-arm CTS pose control

The CTS variables and corresponding Jacobians obtained earlier in section 2.3 were used for the control of absolute and relative poses of the CTS of a dual-arm robot. The simultaneous pose control approach, that uses screw parameters of the error UDQ, was compared against the conventional approach of controlling position and orientation separately. The proposed method was validated with different cooperative tasks, that required both absolute and relate pose control, on Baxter dual-arm robot [63]. The control architecture for implementation of CTS control is explained in section 2.4.1. The robotic platform used for the validation along with its modeling parameters are given in

section 2.4.2. The cooperative tasks, the classical CTS approach used for comparative validation, and the trajectory generation strategy is given in section 2.4.3. The details of the experimental validation with real robot and discussion about the results has been detailed in section 2.4.4.

2.4.1 Control Architecture

The CTS kinematic formulations using UDQ representation was implemented for a general dual-arm system consisting of two redundant (more than 6dofs) manipulators. The controller architecture of the complete system capable of CTS inverse kinematics control is shown in Fig. 2.3.

A trajectory descriptor was implemented for generating desired pose trajectory in absolute and relative task-space for simple tasks like rotation and translation along a pre-defined screw axis defined in a known reference frame. The trajectory generation strategy has been detailed in section 2.4.3.2. The desired absolute (${}^{b}\hat{\boldsymbol{x}}_{abs}^{des}$) and relative pose (${}^{r}\hat{\boldsymbol{x}}_{rel}^{des}$) is used along with the current CTS poses (${}^{b}\hat{\boldsymbol{x}}_{abs}^{curr}$ and ${}^{r}\hat{\boldsymbol{x}}_{rel}^{curr}$) to compute the corresponding error UDQs. The current CTS poses are computed from the current poses of left and arms' EEs, i.e. ${}^{b}\hat{\boldsymbol{x}}_{r}^{curr}$ and ${}^{b}\hat{\boldsymbol{x}}_{t}^{curr}$ respectively, using the current joint positions and initial configuration of joint screw axes of the dual-arm robot in the *Dual-Arm CTS Handler* block in given architecture.

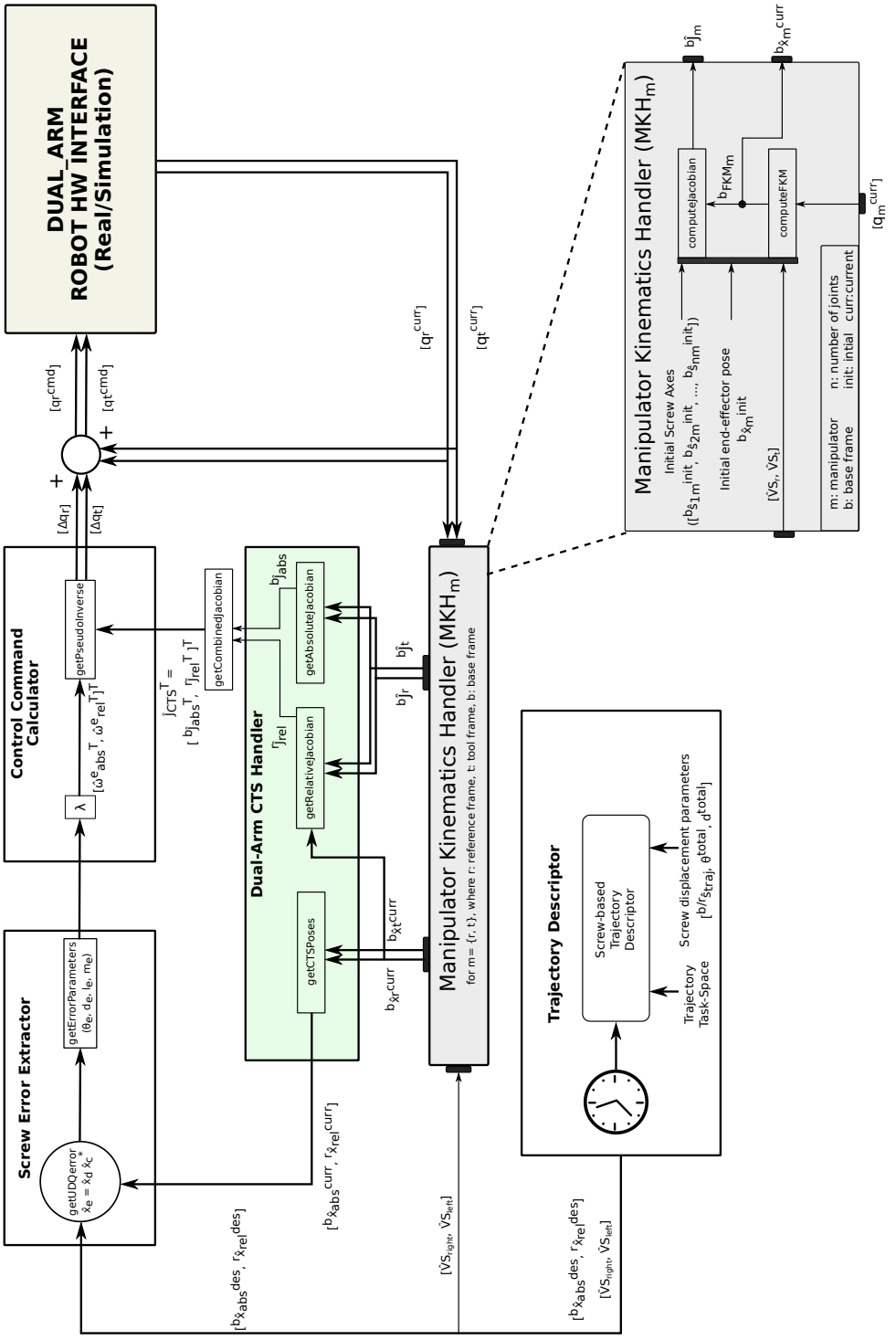


FIG. 2.3 Control architecture for CTS pose control.

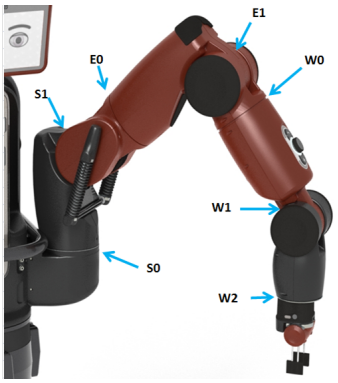


FIG. 2.4 Baxter dual-arm robot's joints [10]. The robot has two 7-dof arms. There are two shoulder joints (S), two elbow joints (E), and three wrist joints (W).

The two end-effectors are extended with VSs pointing to the centre of the mass of the held object, or a desired articulation points for objects like scissors. Note that the current and desired absolute poses are given in the static base frame of the robot, while the desired and current relative poses are given in the *reference manipulator*'s EE's pose. In the current implementation a frame attached to the right arm's EE was chosen as the reference frame for relative task-space.

The CTS Jacobians, ${}^{b}\mathbf{J}_{abs}$ and ${}^{r}\mathbf{J}_{rel}$, were computed from the individual Jacobians (${}^{b}\mathbf{J}_{r}$ and ${}^{b}\mathbf{J}_{t}$) in the *Dual-Arm CTS Handler* block, and a damped pseudo-inverse was used to compute the joint velocity command for the dual-arm robot. The screw-based control law utilizes screw parameters obtained from the error UDQ ($\hat{\boldsymbol{x}}_e$) obtained from the desired ($\hat{\boldsymbol{x}}^{des}$) and current ($\hat{\boldsymbol{x}}^{curr}$) poses related to the CTS as given in *Screw Error Extractor* block in the proposed architecture (see (A.5.3)). The controller was tuned to obtain a stable response for CTS pose control with a suitable gain (λ). An integrator approach was chosen to compute desired joint position commands, $[\boldsymbol{q}_r{}^{cmd}]$ and $[\boldsymbol{q}_t{}^{cmd}]$, in the control loop using the current joint positions, $[\boldsymbol{q}_r{}^{curr}]$ and $[\boldsymbol{q}_t{}^{curr}]$, and computed joint velocity, $[\Delta\boldsymbol{q}_r]$ and $[\Delta\boldsymbol{q}_t]$.

2.4.2 Baxter Dual-arm platform

Baxter robot is an industrial collaborative robot built by Rethink Robotics [63, 64], and is equipped with two compliant 7 *dof*s robotic arms. The joints for one of the arms is shown in Fig. 2.4. The novel actuation mechanism provides protection against shock

TABLE 2.3 Baxter's left and right arm's joints parameters for starting configuration of the robot when all the joints are at home or zero position.

Joints	3D Screw axis ($\boldsymbol{l}$)	Point on axis ($\boldsymbol{p}$)
S0_right	0, 0, 1	0.0640272, −0.259027, 0.129626
S1_right	0.707108, 0.707105, 0	0.112818, −0.307818, 0.399976
E0_right	0.707105, −0.707108, 0	0.184942, −0.379943, 0.399976
E1_right	0.707105, 0.707108, 0	0.370501, −0.565502, 0.330976
W0_right	0.707108, −0.707105, 0	0.44375, −0.638751, 0.330976
W1_right	0.707105, 0.707108, 0	0.635163, −0.830166, 0.320976
W2_right	0.707105, −0.707108, 0	0.71717, −0.912173, 0.320976
S0_left	0, 0, 1	0.0640272, 0.259027, 0.129626
S1_left	−0.707108, 0.707105, 0	0.112818, 0.307818, 0.399976
E0_left	0.707105, 0.707108, 0	0.184942, 0.379943, 0.399976
E1_left	−0.707105, 0.707108, 0	0.370501, 0.565502, 0.330976
W0_left	0.707108, 0.707105, 0	0.44375, 0.638751, 0.330976
W1_left	−0.707105, 0.707108, 0	0.635163, 0.830166, 0.320976
W2_left	0.707105, 0.707108, 0	0.71717, 0.912173, 0.320976

loads [65] and meet the requirements of a collaborative robot that is Power and Force Limited by Inherent Design as described in ISO 10218-1: 2011, section 5.10.5. The robot is imprecise in terms of positioning accuracy of the EEs since the robot uses serial elastic actuators (SEAs) for the actuation of the joints of the arms.

The robot has been used for assembly tasks like bimanual peg-in-hole tasks [66], as well for domestic applications like collaborative human-robot manipulation of deformable objects like bed sheet [67]. The integration of the robot control with robot operating system (ROS) [68] through Baxter software developers kit (SDK) provides a perfect platform to design and test new bimanual applications, and it has been utilized for the validation of the novel methods developed during this thesis in the scope of dual-arm cooperative manipulation.

The joint axes parameters for both the arms of Baxter is given in Table. 2.3 for the initial configuration of the robot with zero displacement in all the joints. They are used to construct 6-D Plücker line as shown in Fig. A.2 for the computation of forward and inverse kinematics, as explained in section A.5.1.

2.4.3 Cooperative dual-arm tasks and comparative validation strategy

The validation of the proposed strategy of CTS pose controller designed based on screw theory was done with tasks that required motion in relative and absolute task space.

TABLE 2.4 Baxter's left and right arm's EE pose in the base-frame of the robot for zero joint displacements.

Arm	EE Position $\{x, y, z\}$ (in m)	EE Orientation $q_0 + q_1\boldsymbol{i} + q_2\boldsymbol{j} + q_3\boldsymbol{k}$
Right Arm	0.7975, 0.9925, 0.3210	0.653281, 0.270599, 0.653281, −0.270599
Left Arm	0.7975, 0.9925, 0.3210	0.653281, −0.270599, 0.653281, 0.270599

We generated the CTS trajectories at different speeds to compare the performance of screw-based CTS Jacobians and screw-error based control law with the conventional CTS Jacobians where the *wrench transformation matrix* is not included in the relative Jacobian. We describe below the CTS Jacobians and control law for the conventional control method used for comparison.

2.4.3.1 CTS Jacobians and Control law for conventional CTS pose controller

The manipulator Jacobians for decoupled controller were obtained using *Kinematics and Dynamics Library* (KDL) library [69, 70].

The conventional cooperative Jacobians, i.e. absolute and relative Jacobians are defined as follows [41, 42]:

$$\mathbf{J}_{abs}^{KDL} = \frac{1}{2}\begin{bmatrix}\mathbf{J}_{right}^{KDL} & \mathbf{J}_{left}^{KDL}\end{bmatrix} \tag{2.34}$$

$$\mathbf{J}_{rel}^{KDL} = \begin{bmatrix}\mathbf{J}_{left}^{KDL} & -\mathbf{J}_{right}^{KDL}\end{bmatrix} \tag{2.35}$$

where $\mathbf{J}_i^{KDL}$ for $i \in \{left, right\}$ is the conventional Jacobian relating the joint velocities to the end-effector frame conventional velocity, rather than screw velocity (refer section A.6.1). Note that, the relative Jacobian for conventional pose control has been given in the base frame of the robot. Hence, the error twist between the current and desired relative pose had to be transformed to the base frame for a relative task defined in the right arm's EE frame, for which relation A.50 can be used, the position and orientation terms to be used for conventional pose controller can be derived (see A.77). The control law for pose control used in the conventional pose controller is given as:

$$\hat{\boldsymbol{\xi}}_{KDL} = \boldsymbol{\lambda}(\theta_e\, \boldsymbol{l}_e + \varepsilon\, \boldsymbol{p}_e) \tag{2.36}$$

Controller Type	**Jacobian Type**	**Control Law**	**Absolute CTS Proportional Gain** $\begin{bmatrix} \boldsymbol{\lambda}_{absP_p}^{3\times3} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \boldsymbol{\lambda}_{absP_o}^{3\times3} \end{bmatrix}$	**Relative CTS Proportional Gain** $\begin{bmatrix} \boldsymbol{\lambda}_{relP_p}^{3\times3} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \boldsymbol{\lambda}_{relP_o}^{3\times3} \end{bmatrix}$
Simultaneous pose Controller	Equation 2.33, 2.20	Equation A.73	$\begin{bmatrix} \mathbf{250} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{250} \end{bmatrix}$	$\begin{bmatrix} \mathbf{250} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{250} \end{bmatrix}$
Conventional Pose Controller	Equation 2.34, 2.35	Equation 2.36	$\begin{bmatrix} \mathbf{300} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{300} \end{bmatrix}$	$\begin{bmatrix} \mathbf{100} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{500} \end{bmatrix}$

TABLE 2.5 Summary of the controllers used for comparative analysis of dual-arm CTS control

where θ_e, and $\boldsymbol{l}_e$ are angle-axis parameters corresponding to the error quaternion obtained from the quaternions related to the desired and actual orientations, and λ is the controller gain.

$$\boldsymbol{q}_e = \boldsymbol{q}_d \boldsymbol{q}_c^*. \tag{2.37}$$

$\boldsymbol{p}_e$ is the relative position error between desired and actual pose, for both relative and absolute task space, and can be computed from the error unit dual quaternion given in (A.72) using (A.70).

The controller gains for the control of CTS using both simultaneous pose control and conventional pose control approach is given in Table 2.5, where $\boldsymbol{\lambda}_{absP_p}$ refers to the proportional gain corresponding to linear part of the error twist, while $\boldsymbol{\lambda}_{absP_o}$ refers to the proportional gain corresponding to angular part for absolute task-space control. Similarly, $\boldsymbol{\lambda}_{relP_p}$ and $\boldsymbol{\lambda}_{relP_o}$, refers to the linear and angular part of the error twist in the relative task space. The gains were chosen by tuning the conventional pose controller for a stable and bounded performance for the pose tracking tasks for Baxter-dual arm robot, where both the controllers were operating at 100 *Hz*.

TABLE 2.6 Cooperative tasks used for comparative validation of CTS control strategy.

Trajectory task-space	Motion type	Screw axis	Reference frame	Displacement desired ($\hat{\theta}_{task_f}$)
Absolute	Translation	0, 0, 1	Initial Absolute frame	0.20 *meter*
Relative	Rotation	1, 0, 0	Initial right EE VS tip frame	40 *degrees*

2.4.3.2 Trajectory generation for CTS kinematic control

All the tasks start with a pre-defined position of the left and right arms' EEs such that the tips of respective VSs are assumed to be coinciding in the middle of the two EEs and the respective frames are pre-oriented to have same orientation at the beginning of the task. The screw axes for the cooperative tasks are constructed by choosing one of the orthogonal axes in the corresponding reference frames, i.e. absolute frame for absolute task, and a frame attached to the tip of the VS attached to the *reference manipulator*'s end-effector frame. The orthogonal axis chosen for the validation of motion in relative and absolute task-space has been provided in Table. 2.6.

A screw displacement can be parametrized with a dual number, $\hat{\theta}_{task} = \theta_{task} + \varepsilon d_{task}$. Given total time and the current time, t_f and t respectively, a trajectory can be specified by varying the displacement parameters represented by the dual number $\hat{\theta}_{task}$ linearly with time:

$$\hat{\theta}_{task}(t) = \hat{\theta}_{task_f}\left(\frac{t}{t_f}\right) \tag{2.38}$$

The UDQ corresponding to the desired pose in the reference frame can then be computed using the screw displacement parameters hence obtained using A.36.

2.4.4 Experiments

In the following sections we will present the plots corresponding to the performance of the proposed screw-based CTS Jacobians and simultaneous pose control technique along

with conventional CTS Jacobians and control law. The two experiments chosen for the comparative validation given in Table 2.6 were carried out with different time allotted for the entire motion, which were: 8, 4.5 and 1.5 seconds. The goal of performing the same motion at different speed was to study the effect of *wrench transformation matrix* consideration in the proposed screw-based CTS Jacobian om the controller performance.

2.4.4.1 Absolute frame translation in z-axis of absolute frame

Baxter robot performing translation of absolute pose in the positive z-axis of absolute frame (fixed at the centre of the end-effectors) can be seen in Fig. 2.5. The yellow lines represent the initial desired state, while magenta lines represent the final desired state of the virtual sticks. The trajectory for the right (*reference manipulator*), left (*tool manipulator*) and the absolute frame is represented with red dots. The relative pose of the two cooperating robotic arms was desired to be fixed during the entire motion.

The simultaneous pose (DQ based) controller and conventional (KDL based) pose controllers performance during the motion in absolute task space for a duration of 8 *sec.* in terms of DQ terms for absolute and relative task space desired and achieved poses are given in Fig. 2.6 and Fig. 2.7, respectively. Similarly the performance of the controllers for the motion duration of 1.5 *secs.* is given in Fig. 2.8 and Fig. 2.9 for absolute and relative task space, respectively.

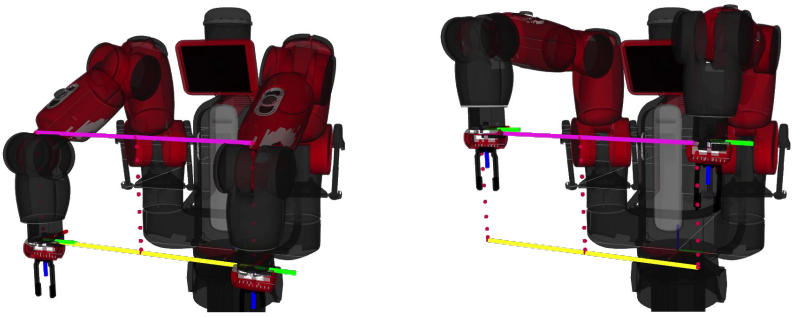


FIG. 2.5 Initial and final configuration of robot performing rotation in the x-axis of relative task frame.

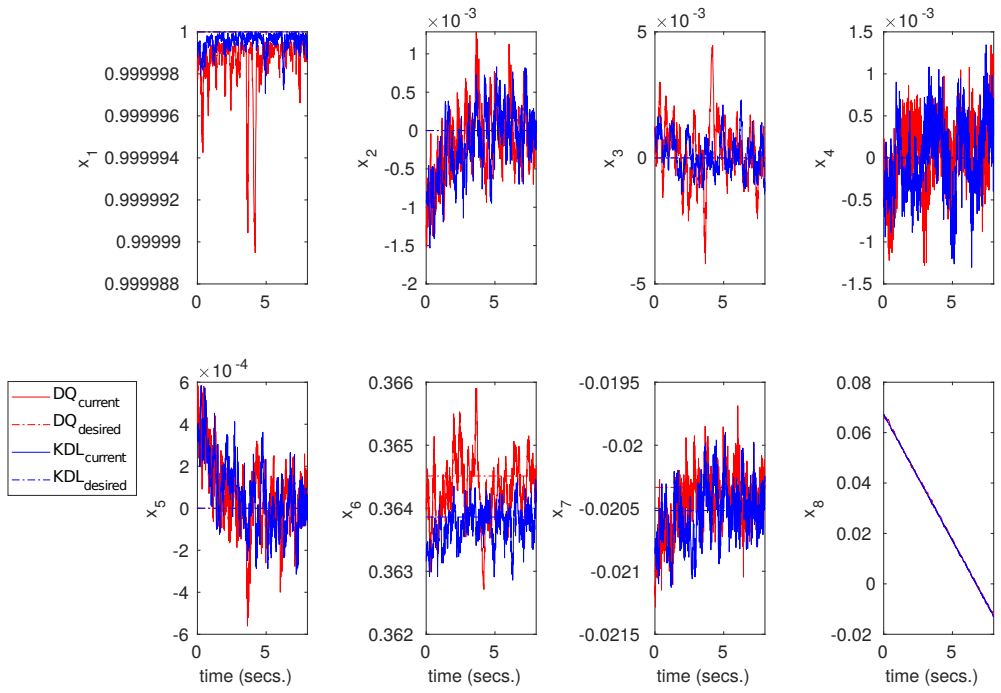


FIG. 2.6 **Absolute Translation**: Pose tracking performance for absolute task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 8 *secs*..

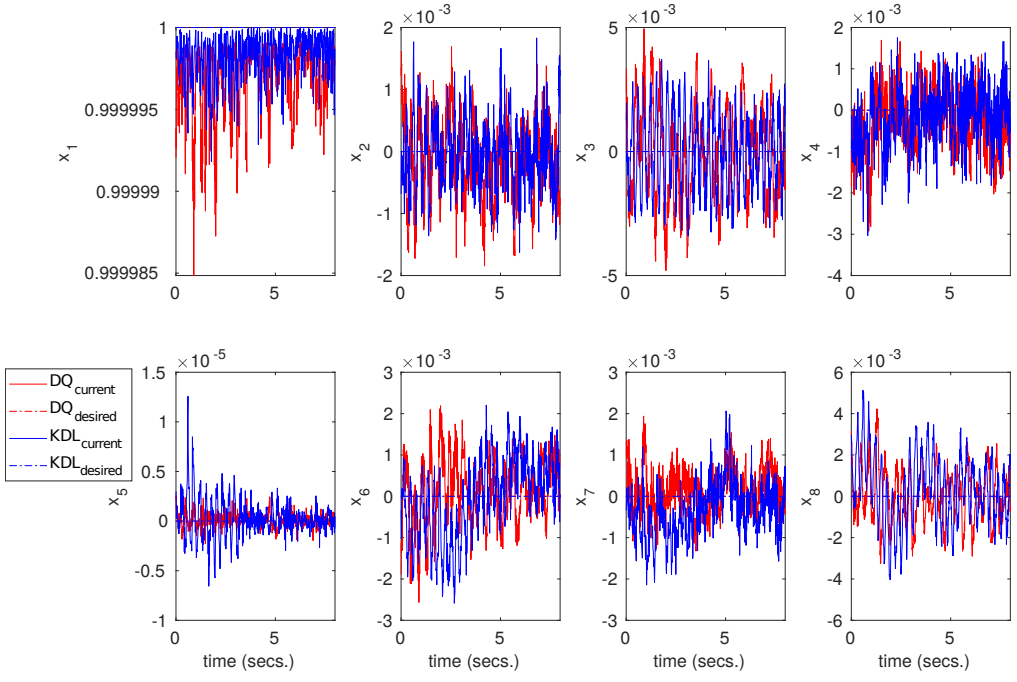


FIG. 2.7 **Absolute Translation**: Pose tracking performance for relative task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 8 *secs*..

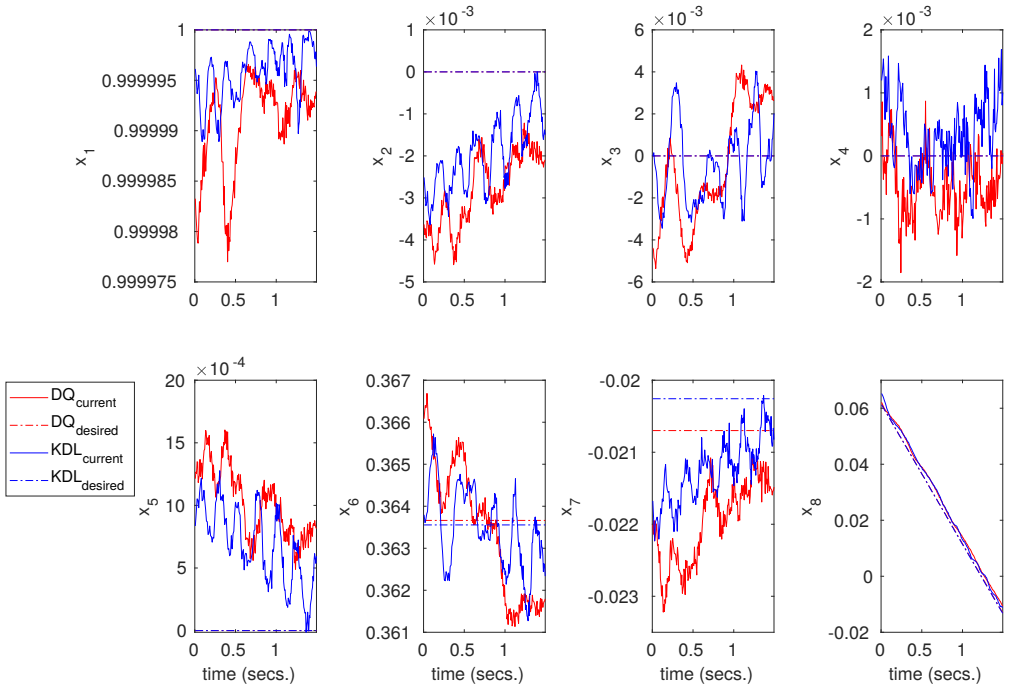


FIG. 2.8 **Absolute Translation**: Pose tracking performance for absolute task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 1.5 *secs*..

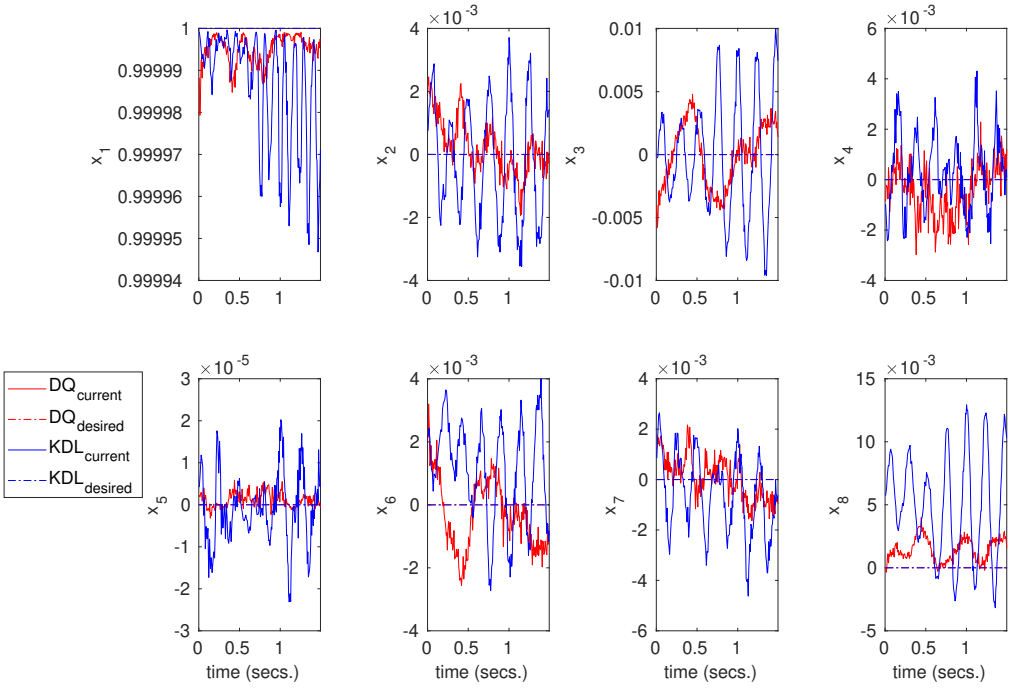


FIG. 2.9 **Absolute Translation**: Pose tracking performance for relative task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 1.5 *secs*..

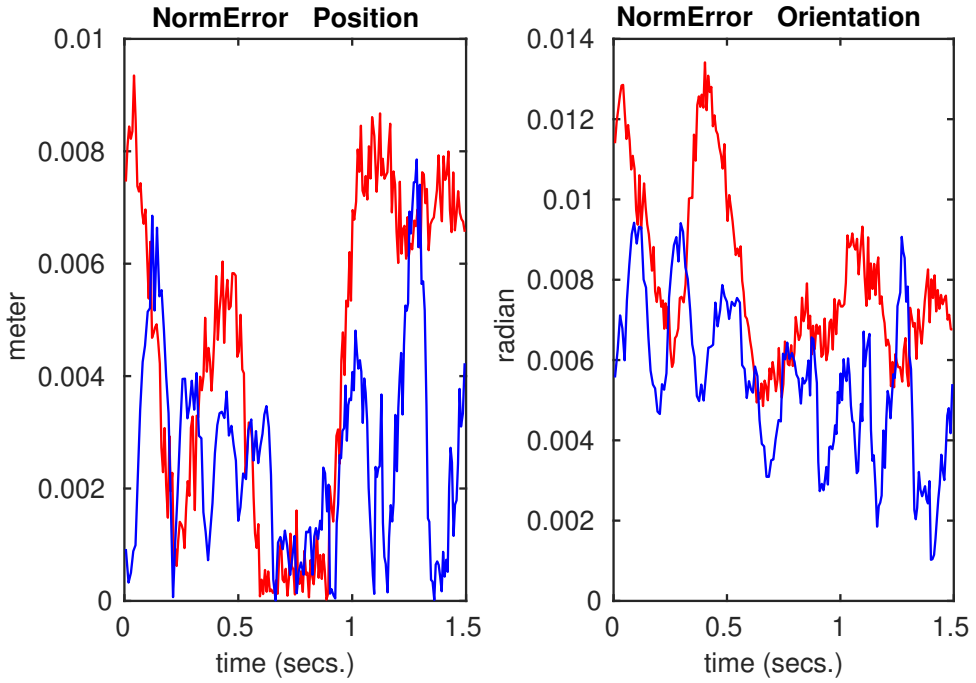


FIG. 2.10 **Absolute Translation**: Evolution of error in terms of the norm of position and orientation error in absolute task space for motion duration 1.5 *secs*. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

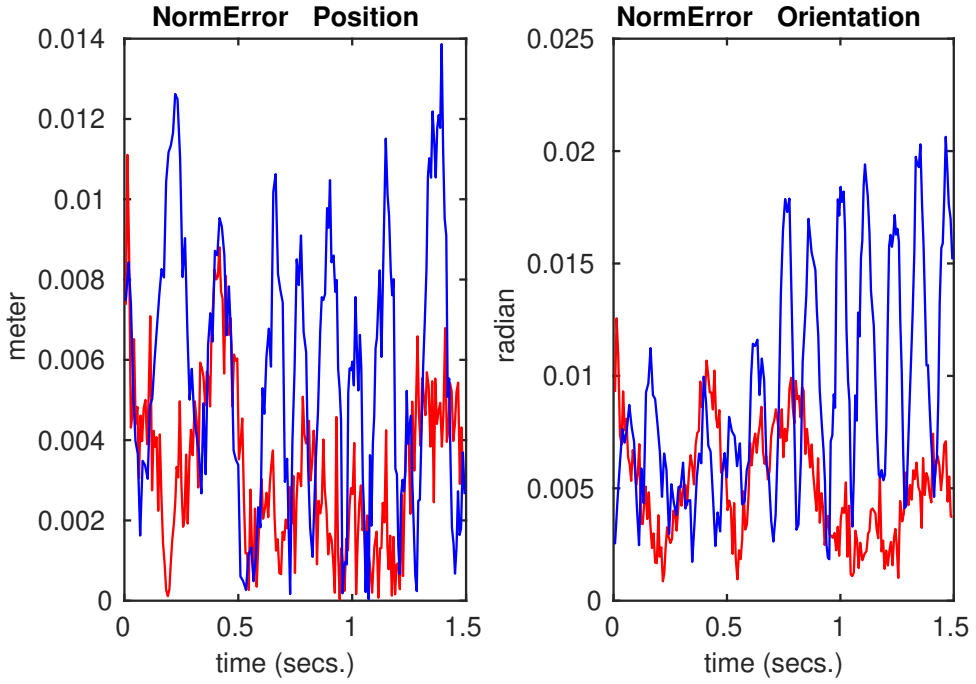


FIG. 2.11 **Absolute Translation**: Evolution of error in terms of the norm of position and orientation error in relative task space for motion duration 1.5 *secs*. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

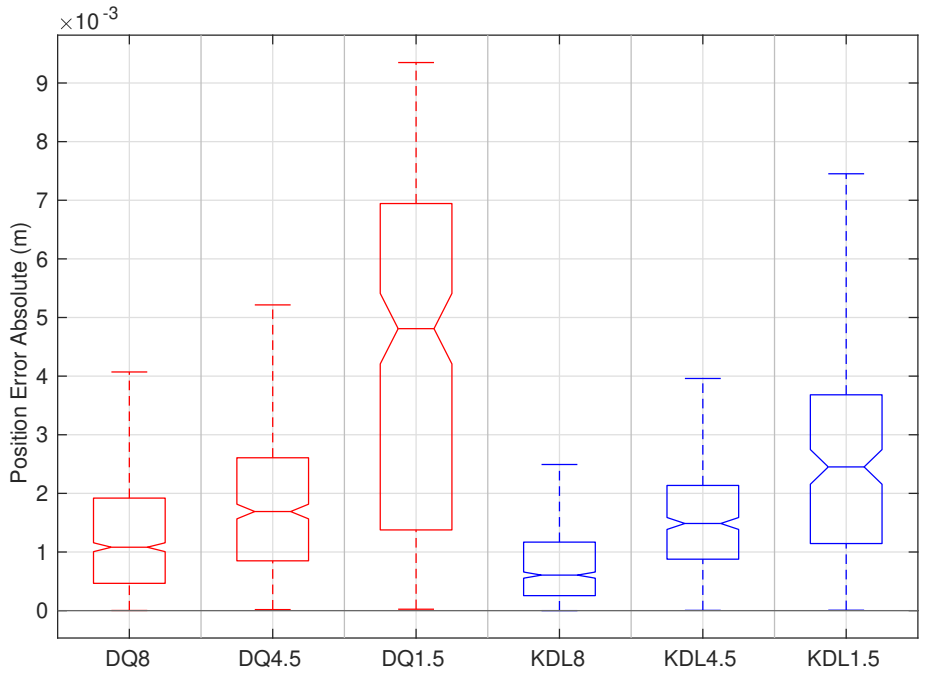


FIG. 2.12 **Absolute Translation**: Box plot corresponding to the mean and standard deviation for position error in absolute task space for the motion duration of 8, 4.5 and 1.5 secs. (——) simultaneous pose control approach and (——) to conventional pose control approach.

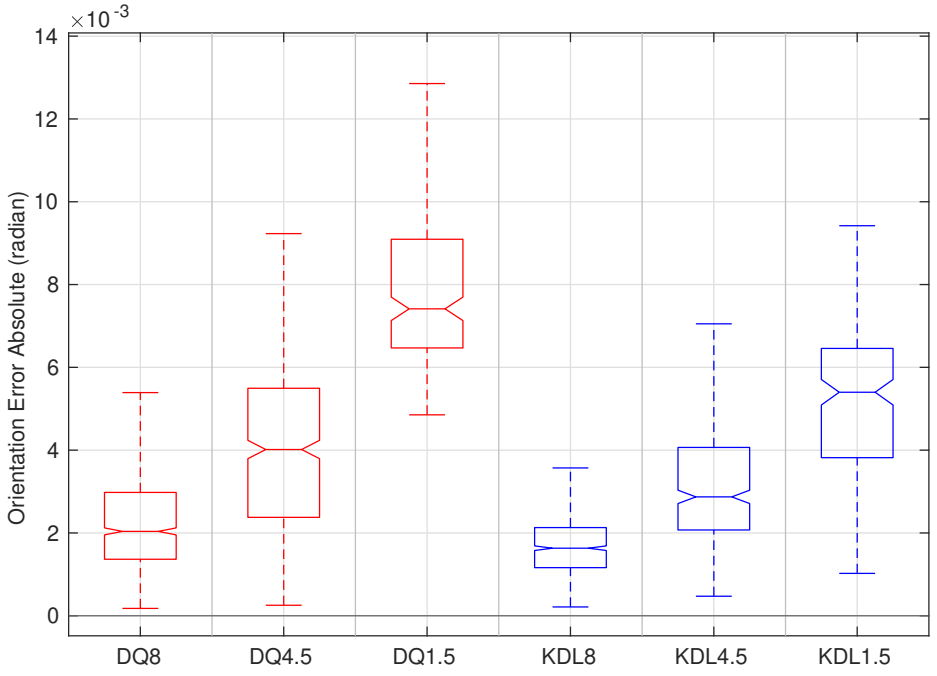


FIG. 2.13 **Absolute Translation**: Box plot corresponding to the mean and standard deviation for orientation error in absolute task space for the motion duration of 8, 4.5 and 1.5 secs. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

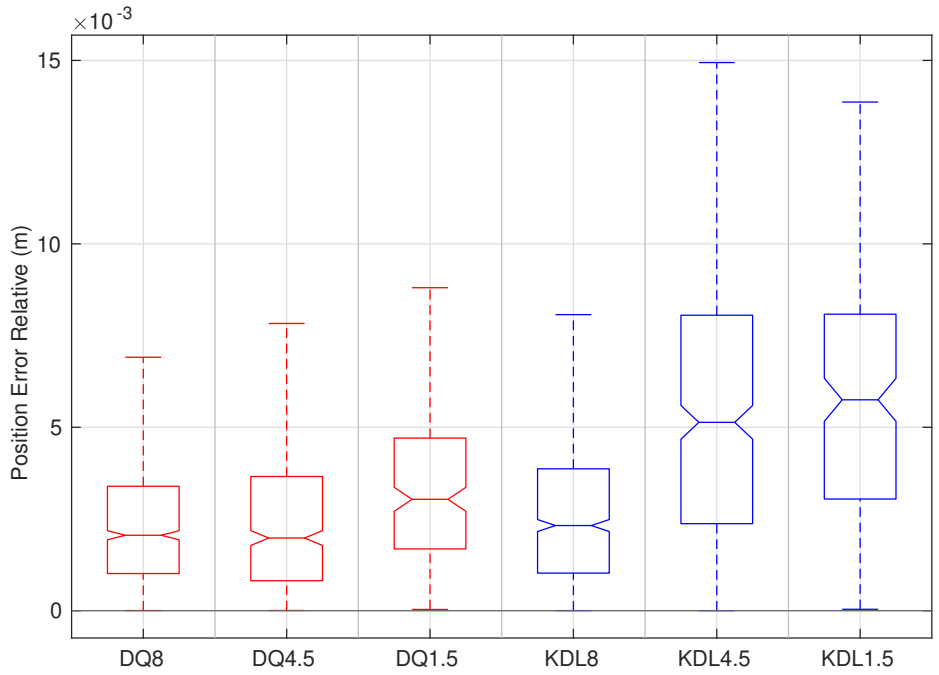


FIG. 2.14 **Absolute Translation**: Box plot corresponding to the mean and standard deviation for position error in relative task space for the motion duration of 8, 4.5 and 1.5 secs. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

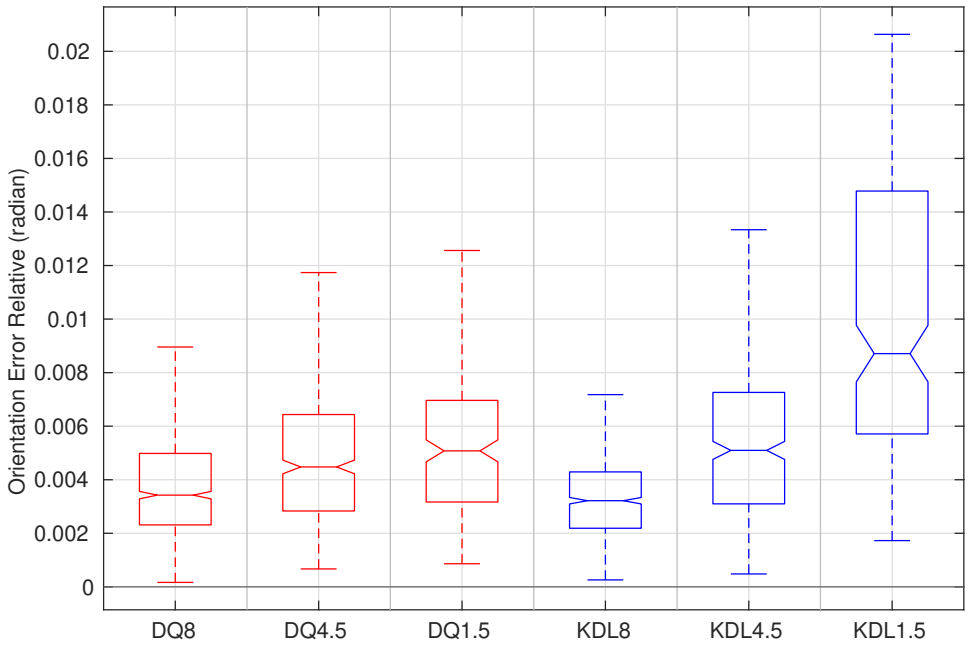


FIG. 2.15 **Absolute Translation**: Box plot corresponding to the mean and standard deviation for orientation error in relative task space for the motion duration of 8, 4.5 and 1.5 secs. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

The evolution of error in terms of norm of position variables ($\sqrt{x_e^2 + y_e^2 + z_e^2}$) and orientation variables ($\|\theta_e\|$) for the motion duration 1.5 *secs.* is given in Fig. 2.10 and 2.11 for absolute and relative task space respectively. Finally, the mean root mean square errors and corresponding standard deviations during CTS pose tracking control for all three time durations are given in Fig. 2.12, 2.13, 2.14, 2.15, for the norm of position and orientation components of pose error in absolute and relative task space. Note that in the desired absolute pose for the simultaneous pose control approach and (——) to conventional pose control approach and conventional pose control approach(see x_6 and x_7 terms in Fig. 2.6) are different because the desired trajectories for the experiments were derived from the initial configuration of the manipulators. Since the validation platform Baxter dual-arm robot consists of non-stiff joints, it is not possible to start the experiments at exactly the same configuration even after setting the joint states initially.

While the performance of the two controllers were similar for both relative and absolute task space for the motion duration 8 *secs.* (Fig. 2.6, 2.7), the conventional pose controller had slightly better performance in absolute task-space, even for the shorter durations of 4.5 and 1.5 *secs*, as evident in Fig. 2.12, 2.13. However, the DQ based simultaneous pose controller demonstrated consistently better performance for position as well orientation tracking (see Fig. 2.14 and 2.15). Additionally, the oscillations observed in KDL based conventional controller in Fig. 2.9 and 2.11 for shortest duration experiment (1.5 sec.) was not observed DQ based simultaneous pose controller.

2.4.4.2 Relative rotation around x axis of right arm's EE frame

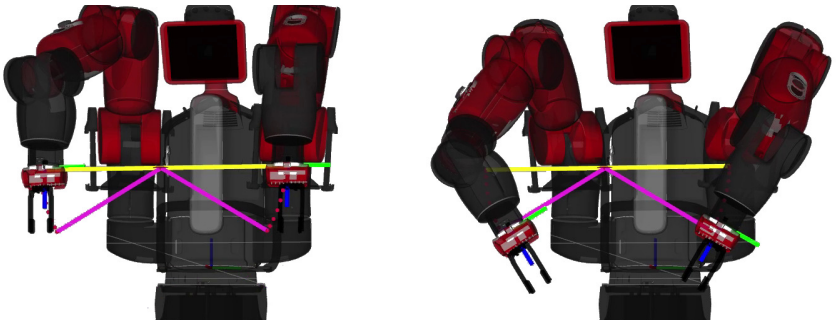


FIG. 2.16 Initial and final configuration of robot performing rotation in the x-axis of relative task frame.

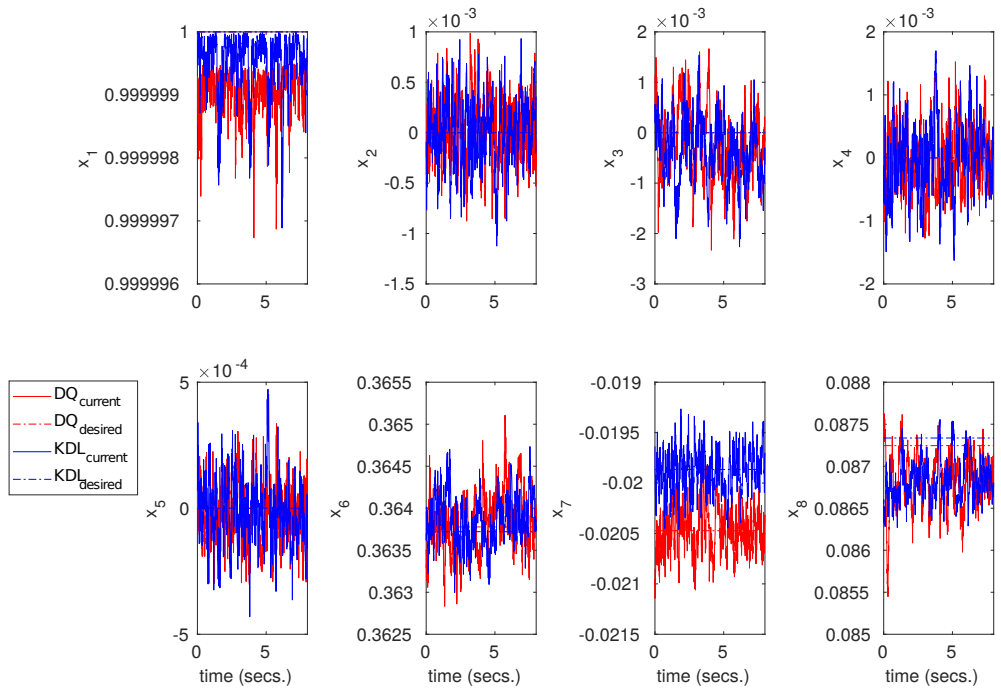


FIG. 2.17 **Relative Rotation**: Pose tracking performance for absolute task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 8 *secs*..

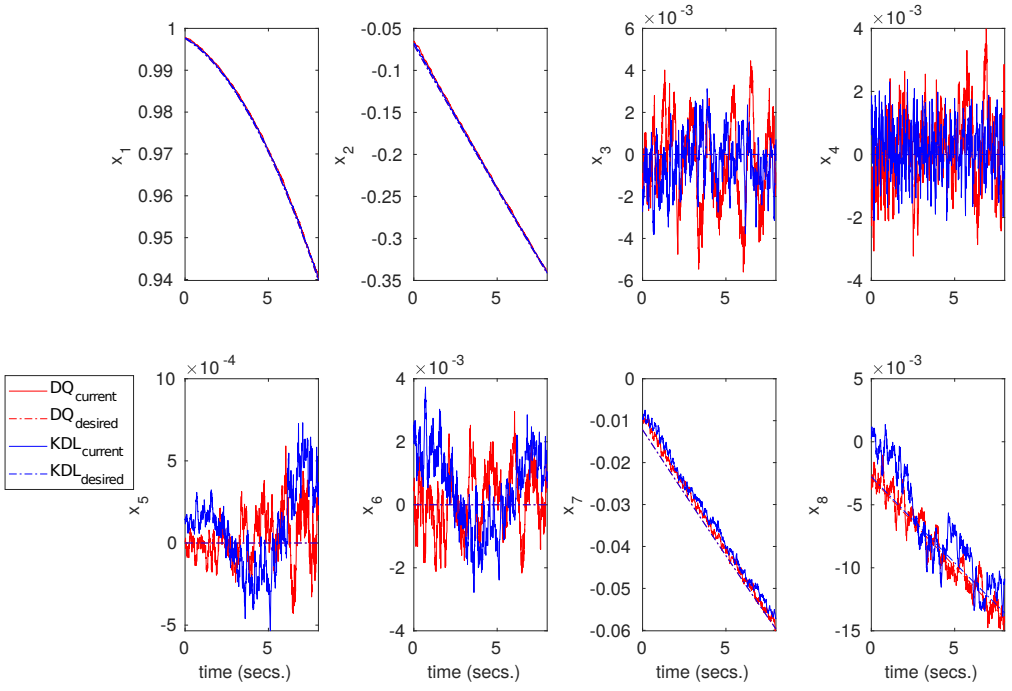


FIG. 2.18 **Relative Rotation**: Pose tracking performance for relative task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 8 *secs*..

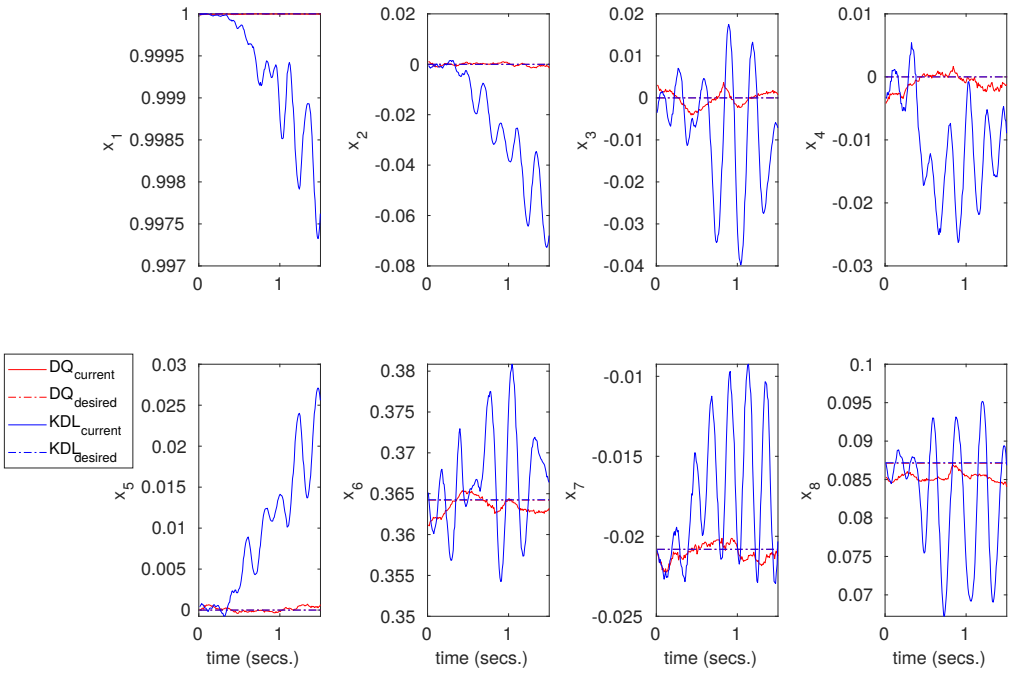


FIG. 2.19 **Relative Rotation**: Pose tracking performance for absolute task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 1.5 *secs*..

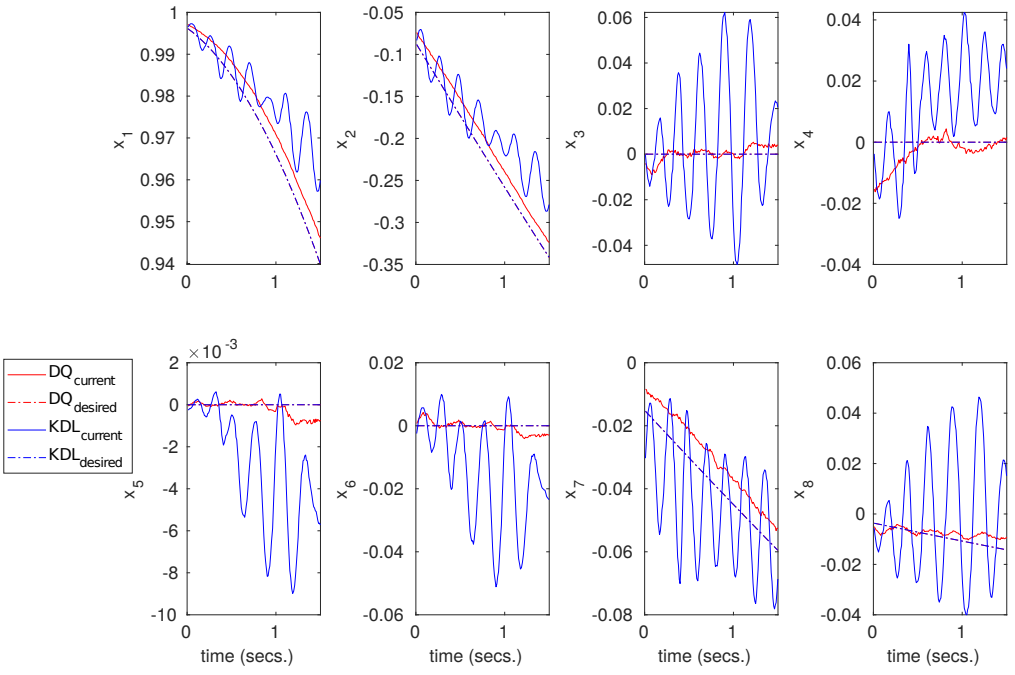


FIG. 2.20 **Relative Rotation**: Pose tracking performance for relative task space in DQ $(x_1, x_2, ..., x_8)$ terms for motion duration 1.5 *secs*..

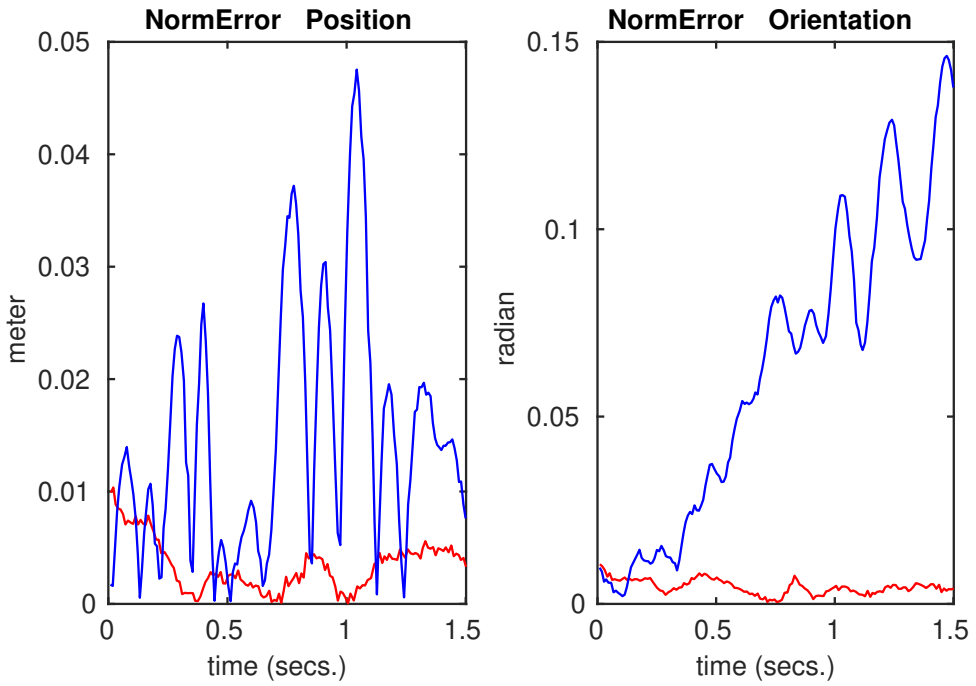


FIG. 2.21 **Relative Rotation**: Evolution of error in terms of the norm of position and orientation error in absolute task space for motion duration 1.5 *secs*.. . (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

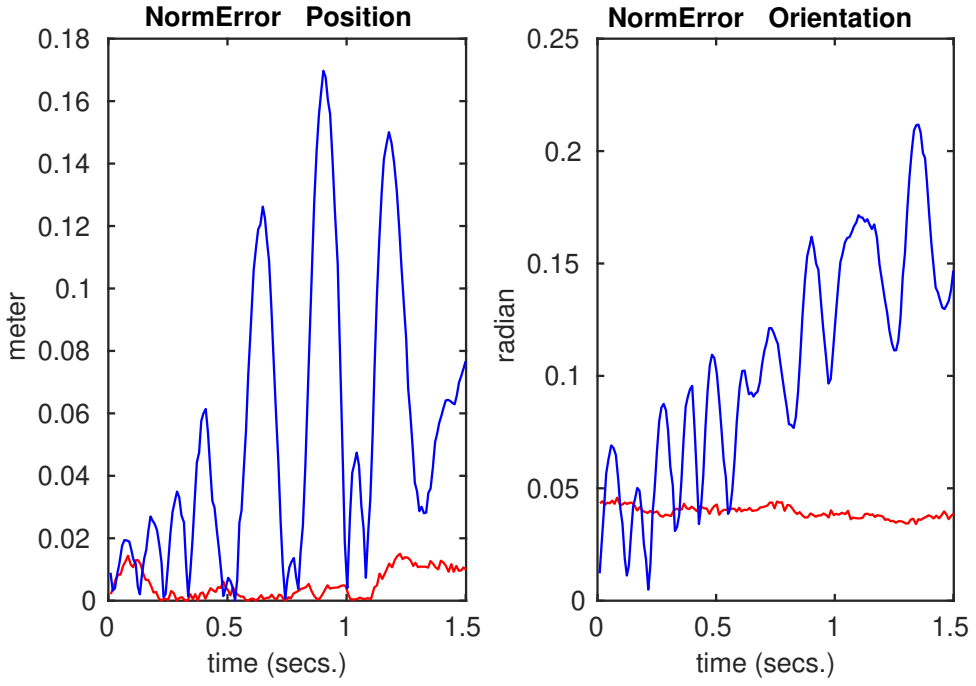


FIG. 2.22 **Relative Rotation**: Evolution of error in terms of the norm of position and orientation error in relative task space for motion duration 1.5 *secs*.. . (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

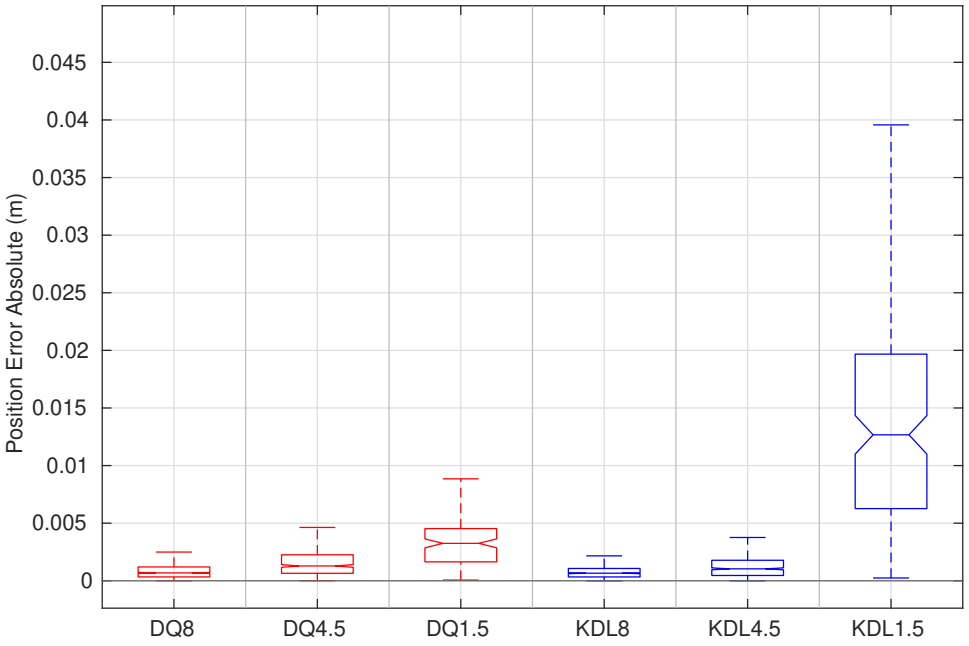


FIG. 2.23 **Relative Rotation**: Box plot corresponding to the mean and standard deviation for position error in absolute task space for the motion duration of 8, 4.5 and 1.5 secs. (—) refers to simultaneous pose control approach and (—) to conventional pose control approach.

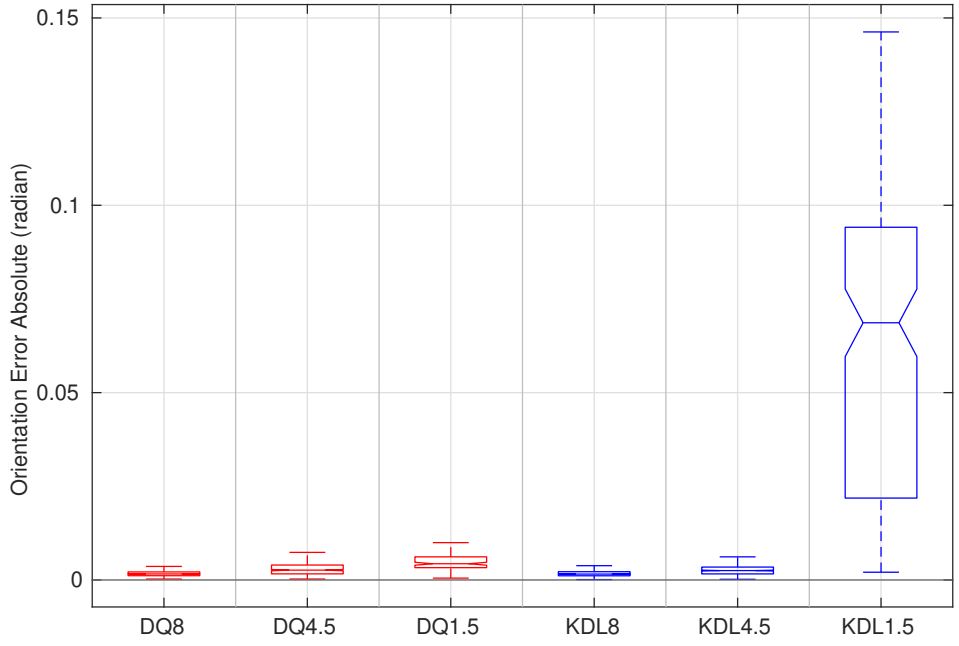


FIG. 2.24 **Relative Rotation**: Box plot corresponding to the mean and standard deviation for orientation error in absolute task space for the motion duration of 8, 4.5 and 1.5 secs. (—) refers to simultaneous pose control approach and (—) to conventional pose control approach.

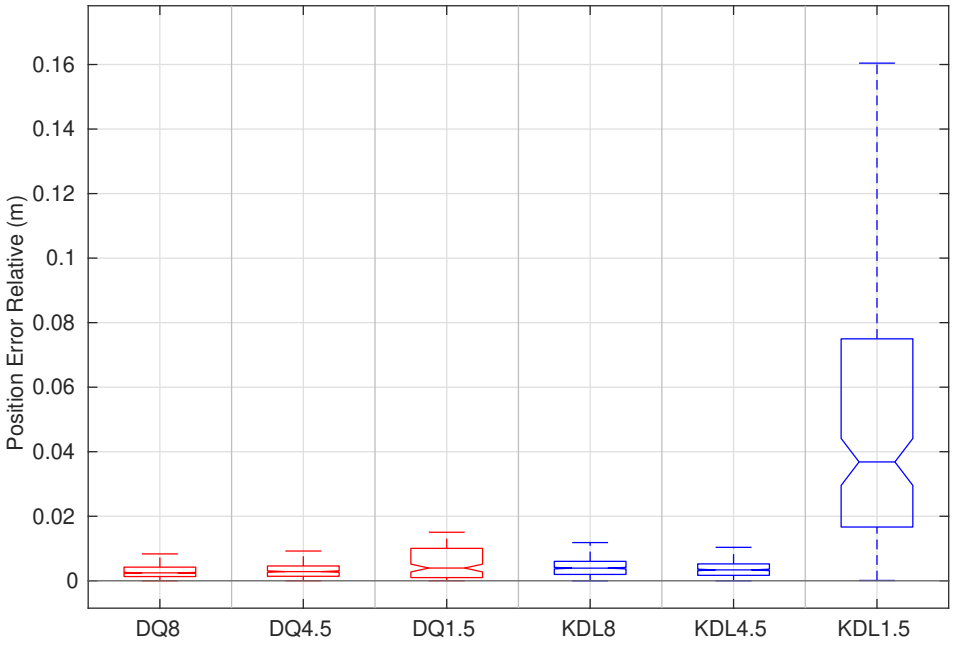


FIG. 2.25 **Relative Rotation**: Box plot corresponding to the mean and standard deviation for position error in relative task space for the motion duration of 8, 4.5 and 1.5 secs. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

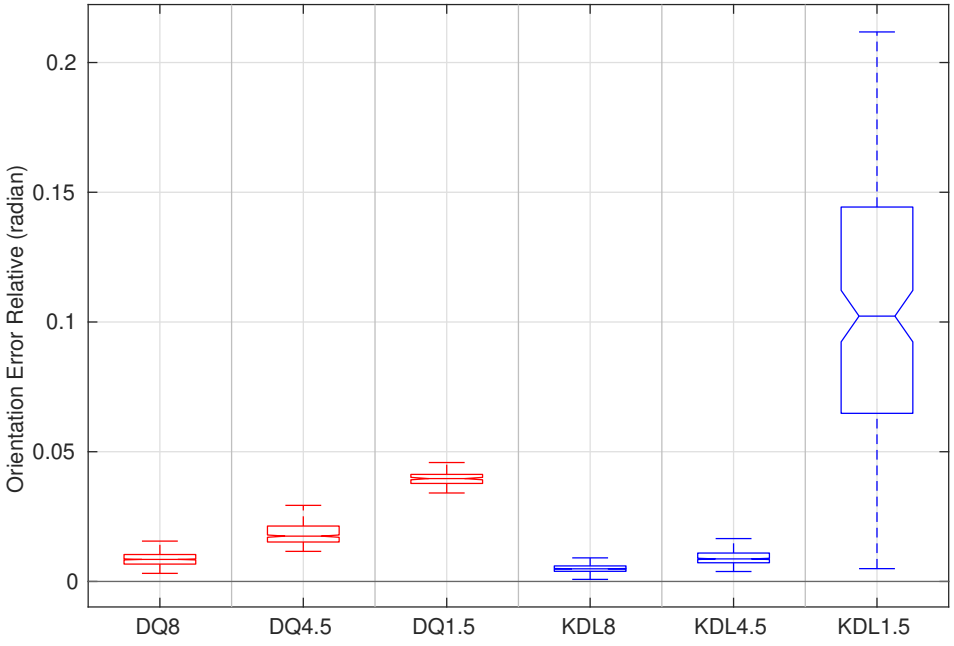


FIG. 2.26 **Relative Rotation**: Box plot corresponding to the mean and standard deviation for orientation error in relative task space for the motion duration of 8, 4.5 and 1.5 secs. (——) refers to simultaneous pose control approach and (——) to conventional pose control approach.

Baxter robot performing relative rotation between the tips of the virtual sticks connected to the end-effectors of two arms around x-axis of the tip of *reference manipulator*'s VS frame can be seen in Fig. 2.16. The trajectory for the right and left end-effectors is represented with red dots. The absolute pose at the centre of the two cooperating robotic arms was desired to be fixed during the entire motion.

The simultaneous pose controller and conventional pose controllers performance in terms of DQ terms for absolute and relative task space desired and achieved poses are given in Fig. 2.17 and Fig. 2.18, respectively. Similarly the performance of the controllers for the motion duration of 1.5 *secs.* is given in Fig. 2.19 and Fig. 2.20 for absolute and relative task space, respectively. The two controllers performance during the motion in relative task space for a duration of 8 *sec.* in terms of DQ terms related to the absolute and relative task space desired and achieved poses are given in Fig. 2.17 and Fig. 2.18, respectively. Similarly the performance of the controllers for the motion duration of 1.5 *secs.* is given in Fig. 2.20 and Fig. 2.20 for absolute and relative task space, respectively. The evolution of error in terms of norm of position variables and orientation variables for the motion duration 1.5 *secs.* is given in Fig. 2.21 and 2.22 for absolute and relative task space respectively. Finally, the mean root mean square errors and corresponding standard deviations during CTS pose tracking control for all three time durations are given in Fig. 2.23, 2.24, 2.25, 2.26, for the norm of position and orientation components of pose error in absolute and relative task space.

The conventional controller had slightly better performance in absolute task-space for the relative rotation for a duration of 8 and 4.5 *secs.* (Fig. 2.17, 2.18, 2.23, 2.24). However, the DQ based simultaneous pose controller demonstrated consistently better performance for position as well orientation tracking in relative task-space control (see Fig. 2.25 and 2.26). For tasks with total duration 1.5 secs. high oscillations can be observed in relative and absolute task space, as is evident from Fig. 2.19, 2.20, 2.21 and 2.22.

The performance of conventional for position and orientation control in relative as well as absolute task space deteriorated for faster relative task due to high oscillations, as shown in Fig. 2.23, 2.24, 2.25, 2.26. The performance of simultaneous pose control is also affected from high relative motion, however, the controller remains stable and error

were significantly less for position and orientation control in absolute and relative task space.

The above results confirms that consideration of *wrench transformation matrix* for relative task-space control is important for tasks requiring faster motion in relative task space. Additionally, it demonstrates that the absolute pose tracking is reliant on the stability and performance of relative task space, since oscillations were also observed in absolute task space for conventional control.

2.5 Conclusion

The classical approach for dual-arm cooperative task space control was revisited and the symmetric formulation of dual arm coordination using VSs was implemented using dual quaternion method. The proposed proportional simultaneous pose control of cooperative task space, i.e. simultaneous control of both position and orientation was compared against the performance of a proportional conventional controller that used KDL library for forward kinematics and Jacobian computation. The simultaneous pose controller demonstrated better tracking of pose and orientation in terms of accuracy and stability compared to KDL based controller for tasks requiring faster operation in CTS.

One of the future goal is to define various task Jacobians, for relative and absolute task-space, like rotation or translation along a desired axis, and control only those aspects of the motion which are absolutely necessary for the completion of the task, while using the redundancy to perform additional task, like increasing manipulability, avoiding singularities and joint limits, etc. Although, the simultaneous pose approach for kinematic control of CTS showed improved performance over conventional conventional approach, still the deviation of actual CTS poses from desired set-points reveals the need to consider the interaction forces for cooperative manipulation, specially in the case of rigid objects. So, the first goal for the screw-based implementation for manipulation is to verify its applicability for single arm case and compare the hybrid force/position control performance with screw-based Jacobian. Afterwards, it is desired to do the same for dual-arm robots, where the motion and forces are controlled for relative task-space. A similar approach was taken for dual-arm impedance control in [71], where a modular

relative Jacobian was formulated for relative task-space control. It will allow the controller to control the interaction between the two arms during dual-arm manipulation, if the grasps are assumed to be infinitely rigid.

Chapter 3

Kinematics of the fingers of an anthropomorphic robotic hand with coupling

3.1 Introduction

The operation of robotic hands performing grasping tasks employs requires two kinds of operations: bringing the fingers of the hand to an appropriate grasping position, and closing the fingers until the tactile sensors on the fingertips senses a contact. The availability of a kinematic model allows for the control of the fingers for each of the above operations. In this work we present the kinematic model of an anthropomorphic hand, consisting of fingers with coupled joints similar to humans.

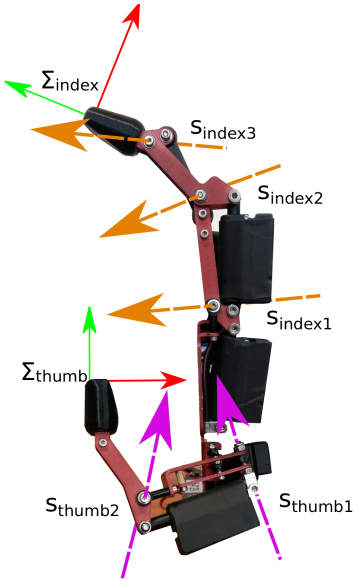
Underactuated hands provides a cheaper and lighter solution for grasping, in addition to self-adapting capability for different kinds of objects [72]. A four-bar linkage comprising middle and distal phalanx in a robotic finger can effectively imitate the coupling that exists between the proximal and distal interphalangical joints (*i.e.* PIP and DIP joints) of the human fingers. Additionally, the higher load bearing capacity of four-bar linkages [73] makes them a natural choice for the purpose of effort transmission in fingers.

However, the mechanism employing these systems often presents additional complexity to the kinematic formulation, owing to the non-linear relation between the actuation

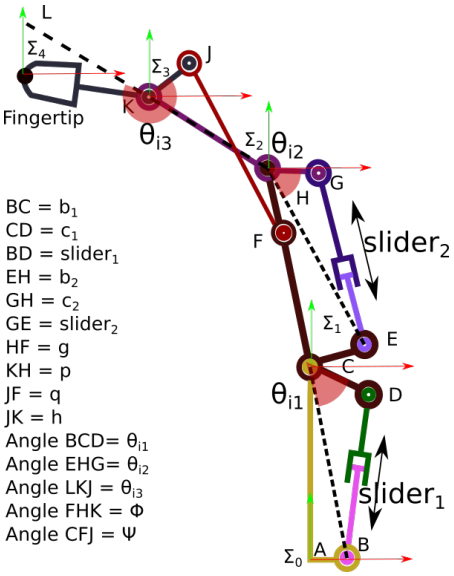
and the motion of the finger joints. Some existing kinematic formulations for these kind of fingers use a simplified linear coupling model for the relation between PIP and DIP joints at the cost of accuracy, or a complex actuation mechanism as a trade-off for the simplicity of the design [74].

A geometrical approach was taken to obtain the relation between the finger joints actuation and the motion of finger consisting of such mechanisms using a coupling Jacobian. The goal of such formulation was the easier integration of the fingers fo the robotic hand with the arm at the kinematic level. This will allow us in the future for a combined forward and inverse-kinematics control of the robotic arm and hand, to ensure reachability of the fingertips to the grasping point, and pre-plan the grasping tasks in terms of the finger movement.

In addition to that, a relative Jacobian has been derived to control the relative configuration of index finger and thumb fingertips (see Fig. 3.1a) using a screw-based method described in 2, with the aim of using this model for grasping and in-hand manipulation. In addition to the advantages of the screw-based treatment of cooperative task space strategy using UDQ (see Table 2.1 and 2.2), the kinematic model can easily be integrated with the kinematics of a robotic arm in order to benefit from the added redundancy.



(A) Index finger and thumb of AR10 hand, with corresponding axes and frames.



(B) Index finger mechanism: Frames defined on the base of the index finger ($\mathbf{\Sigma_0}$), on each rotational joints (i.e. $\mathbf{\Sigma_1}$, $\mathbf{\Sigma_2}$, $\mathbf{\Sigma_3}$), and on the fingertip ($\mathbf{\Sigma_4}$).

FIG. 3.1 AR10 hand, and index finger mechanism.

The model hence obtained is implemented and validated on an anthropomorphic robotic hand, AR10 from Active8 Robots[75] shown in Fig. 3.1a, installed at the end-effector of one of the arms of Baxter robot. Though AR10 hand provides a versatile yet relatively cost-effective solution for performing grasping and manipulation task, a kinematic model for the underactuated finger mechanism for forward and inverse kinematics currently does not exist, as far as the author knows.

In the following sections we describe the formulation of finger Jacobian using unit dual quaternion formulation. The use of screw based kinematics helps in avoiding the complexity of kinematic modelling associated with the traditional Denavit Hartenberg convention based method. Section 3.2.1 provides the detail of the formulation of kinematics and corresponding Jacobian for fingers with coupled joints, whereas the formulation of relative Jacobian in section 3.2.2. The validation of the formulations obtained on AR10 hand is explained in sec. 3.3. Afterwards, a summary of insights and future goals is presented in sec. 3.4.

3.2 Kinematic modelling of the fingers

3.2.1 Formulation of index finger Jacobian

The kinematic mechanism corresponding to the index finger of AR10 hand (Fig. 3.1a) is given in Fig. 3.1b. The initial configuration of the joint angles, and screw axes are computed and stored for a desired starting configuration, with respect to the base frame of the index finger, *i.e.*, $\boldsymbol{\Sigma_0}$. The relation between the finger configuration and the slider positions is presented below.

The index finger consists of three joints where the last two joints are coupled. The corresponding joints are $\underline{\hat{\boldsymbol{\theta}}}_{index} = [\hat{\theta}_{i1} \;\; \hat{\theta}_{i2} \;\; \hat{\theta}_{i3}]^T$. The proximal phalanx lead screw mechanism ($slider_1$) rotates $\boldsymbol{\Sigma_1}$ around its z-axis, and thus changing θ_{i1}. Using cosine rule for the ΔBCD, θ_{i1} is computed as:

$$\theta_{i1} = cos^{-1}\left(\frac{{b_1}^2 + {c_1}^2 - {slider_1}^2}{2\, b_1\, c_1}\right) \tag{3.1}$$

θ_{i2}, and consequently the rotation of $\mathbf{\Sigma_2}$ caused by $slider_2$, is also computed using (3.1), using the corresponding parameters, i.e., b_2, c_2 and $slider_2$. The change in the $slider_2$ caused by the middle phalanx movement leads to the change in θ_{i2} as well as θ_{i3}, owing to the four-bar mechanism of the distal phalanx and middle phalanx defined by the segments HF, JF, JK and KH. As it can be seen in Fig. (3.1b), the change in θ_{i2}, also corresponds to change in ϕ, that is the angle between the segment FH and HK. θ_{i3} corresponding to a given ϕ is obtained using the constraint of fixed length of the link JK, and the two ways to define the position of point K from point H, i.e by using the position constraint $HK = HF + FJ + JK$.

$$\theta_{i3} = tan^{-1}\left(\frac{q\,sin(\psi) - p\,sin(\phi)}{g + q\,cos(\psi) - p\,cos(\phi)}\right) - \phi \tag{3.2}$$

where, ψ is given as:

$$\begin{aligned}
\psi(\phi) &= tan^{-1}\left(\frac{B}{A}\right) - cos^{-1}\left(\frac{C}{\sqrt{A^2 + B^2}}\right) \\
A(\phi) &= 2\,p\,q\,cos(\phi) - 2\,g\,q, \\
B(\phi) &= 2\,p\,q\,sin(\phi) \\
C(\phi) &= g^2 + p^2 + q^2 - h^2 - 2\,p\,g\,cos(\phi).
\end{aligned}$$

The parameters p, q, g, h used in above equation have been shown in Fig. 3.1b. The relation between the derivative of ϕ or θ_{i2}, and θ_{i3} is given as ([76]):

$$f(\theta_{i2}, \psi, \theta_{i3}) = \frac{\dot{\theta_{i3}}}{\dot{\phi}} = \frac{\dot{\theta_{i3}}}{\dot{\theta_{i2}}} \tag{3.3}$$

$$= \frac{p\,sin(\psi - \phi) - h\,sin(\phi + \theta_{i3} - \psi)}{h\,sin(\phi + \theta_{i3} - \psi)}. \tag{3.4}$$

Since change in θ_{i2} corresponds to the change in ϕ, the coupling Jacobian is given as $f(\theta_{i2}, \psi, \theta_{i3})$, assuming the initial offset between θ_{i2} and ϕ is known. Finally, the unit dual quaternion based Jacobian for the index finger is given as (see Fig. 3.1a):

$$J_{index} = \begin{bmatrix} \hat{\boldsymbol{s}}_{index1} & (\hat{\boldsymbol{s}}_{index2} + \hat{\boldsymbol{s}}_{index3} \cdot f(\theta_{i2}, \psi, \theta_{i3})) \end{bmatrix} \tag{3.5}$$

Note that the relations (3.2) to (3.5) are only valid for the *crossed* configuration of four-bar mechanism given in Fig. 3.1b ([77], [76]). It has been assumed that the configuration

will be *crossed* throughout the operation, which is, in fact, true for the particular hand we have chosen for validation of the strategy.

The error unit dual quaternion $\hat{\boldsymbol{e}}$ in the base frame, can be obtained as the relative displacement between the desired and current pose of the fingertip, *i.e.* $\hat{\boldsymbol{x}}_d$ and $\hat{\boldsymbol{x}}_c$, respectively computed in the base frame.

$$\hat{\boldsymbol{e}} = \hat{\boldsymbol{x}}_c \; \hat{\boldsymbol{x}}_d^* \tag{3.6}$$

where, $\hat{\boldsymbol{x}}_d^*$ is the classical quaternion conjugate of $\hat{\boldsymbol{x}}_d$. The control law for kinematic control for screw-based method using unit dual quaternion representation for a given gain λ, is:

$$\begin{aligned} \hat{\boldsymbol{\xi}} &= -\lambda \; \ln(\hat{\boldsymbol{e}}) \\ &= -\lambda(\theta_e \; \boldsymbol{l}_e + \varepsilon \; (\theta_e \; \boldsymbol{m}_e + d_e \; \boldsymbol{l}_e)) = -(\boldsymbol{\omega}_e + \varepsilon \boldsymbol{v}_{oe}) \end{aligned} \tag{3.7}$$

where, $\{\theta_e, \; d_e, \; \boldsymbol{l}_e, \; \boldsymbol{m}_e\}$ are screw displacement parameters 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 [14].

3.2.2 Formulation of relative Jacobian

The thumb consists of two lead screw mechanisms in series, with three links for each joints connected as a triangle to convert the translational motion of lead-screw mechanism to rotational motion of the joints. The geometrical arrangement is similar to the first two mechanisms of the index finger, *i.e.*, ΔBCD and ΔEHG. However, the rotation axes of the thumb, *i.e.* $\hat{\boldsymbol{s}}_{thumb1}$ and $\hat{\boldsymbol{s}}_{thumb2}$, are not coplanar, as was the case with the index finger (refer to Fig. 3.1b). The forward kinematics for the thumb is computed using (3.1).

The relative Jacobian maps the intermediate joint displacement to the relative screw displacements between the index fingertip and thumbtip frames. The index fingertip is taken as the reference frame for the relative pose control (Fig. 3.1a). The relative Jacobian defined in the index fingertip frame is given as:

$$J_{rel} = \begin{bmatrix} (\hat{\boldsymbol{s}}_{i2} + \hat{\boldsymbol{s}}_{i3} \cdot f(\theta_{i2}, \psi, \theta_{i3})) & \hat{\boldsymbol{s}}_{i1} & \hat{\boldsymbol{s}}_{t1} & \hat{\boldsymbol{s}}_{t2} \end{bmatrix} \tag{3.8}$$

Note that in the above equation the subscript *index* has been replaced with i for the joint screw axes of index finger, for the sake of compactness, whereas subscripts *t1* and *t2* refers to the joint screw axes of the thumb joints. It is important to point out that all the joint screw axes, which were computed at the base frame, $\boldsymbol{\Sigma_0}$, have to be transformed to the frame at the index fingertip ($\boldsymbol{\Sigma_{index}}$), during the starting of the computation. The joint displacement array corresponding to the relative Jacobian defined in (3.8) is:

$$\delta\hat{\underline{\boldsymbol{\theta}}}_{rel} = \begin{bmatrix} -\delta\hat{\theta}_{i2} & -\delta\hat{\theta}_{i1} & \delta\hat{\theta}_{t1} & \delta\hat{\theta}_{t2} \end{bmatrix}^T \tag{3.9}$$

Again, note that the joint displacements for the index finger joints are negative, in order to account for the reverse sense of displacement as seen from the index finger tip frame, compared to the displacement observed from the base frame $\boldsymbol{\Sigma_0}$.

3.3 Experimental validation and results

The Jacobian formulated in (3.5) for the finger mechanism was used for the inverse kinematic control for the index finger of AR10 hand (see (3.7)). Using the formulations discussed in subsection 3.2.1, the initial values of θ_1, θ_2, and θ_3 were obtained. The subsequent angular displacements were computed using the current slider positions for forward kinematics. The slider commands for inverse kinematic control were computed by using the inversion of (3.1) and (3.2), using the initial and the computed values of θ_1, θ_2, and θ_3.

The parameters given in Fig. (3.1b) and similarly for the thumb, i.e. the dimensions of all the involved segments, were computed using manual measurements. In the following experiments we express the error of inverse kinematics with the assumption of perfect knowledge of those parameters. Since the fingers have very limited workspace, a pre-identified achievable configuration for the index fingertip was chosen as desired position of the fingertip. As index finger is a planar mechanism, the XY plane trajectory taken by the finger tip have been depicted in Fig. 3.2.

The exponential convergence of the norm of the error screw parameters used in the controller (see (3.7)) is computed as:

$$ScrewError_{norm} = \sqrt{\boldsymbol{\omega}_e^2 + {\boldsymbol{v}_{oe}}^2}, \tag{3.10}$$

is shown in Fig. 3.3. Among the three controller gains tested, the best performance was achieved for the gain $\lambda = 0.5$. The convergence is the slowest for $\lambda = 0.1$, while some oscillation can be seen in the *Finger tip trajectory* plot in Fig. 3.3 for $\lambda = 1$. The error at the end of the inverse kinematic control, i.e. at the steady state is given in Table 3.1, which demonstrates that for the gains of $\lambda = 0.5$ and 1, the error is under one *mm*.

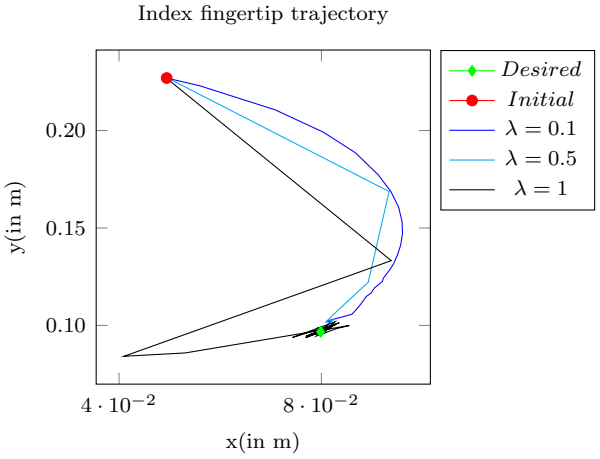


FIG. 3.2 Trajectory of the fingertip of the index finger.

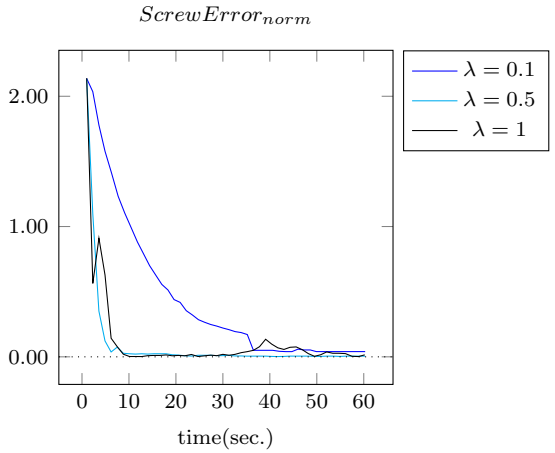


FIG. 3.3 Performance of the inverse kinematics controller.

The performance of the relative pose control between index and thumb tips are given in Fig. 3.4 and Fig. 3.5. The results for only those gains that provided satisfactory result

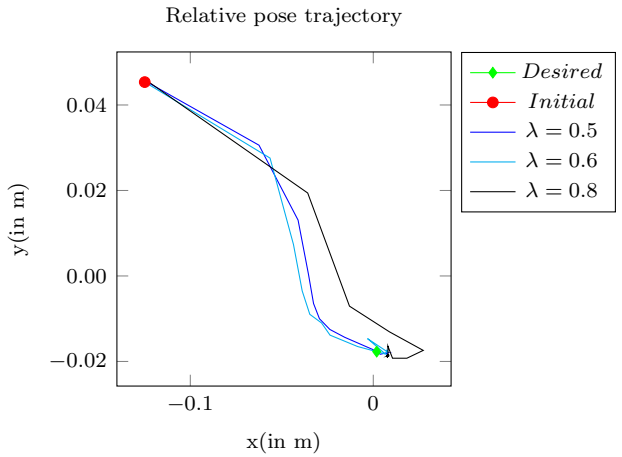


FIG. 3.4 Trajectory of the relative pose between index fingertip and thumbtip in the index fingertip frame.

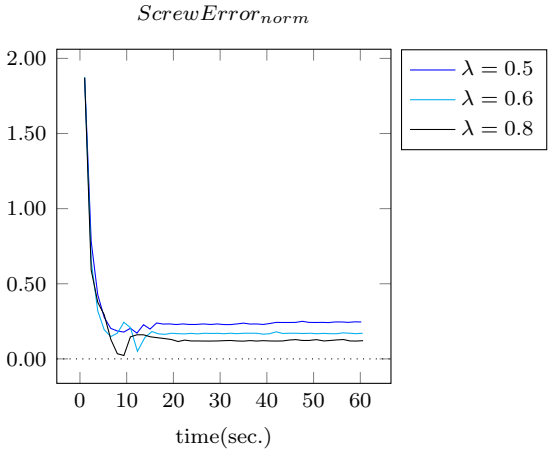


FIG. 3.5 Performance of the inverse kinematics controller for relative pose control.

TABLE 3.1 Final error of the index finger IK computation.

Gain	$\mathbf{X}_{error}$ (in mm.)	$\mathbf{Y}_{error}$ (in mm.)
0.1	−1.37181	−3.89763
0.5	−0.196327	−0.373781
1	−0.598718	−0.714857

TABLE 3.2 Final error of the relative pose IK computation.

Gain	$\mathbf{X}_{error}$ (in mm.)	$\mathbf{Y}_{error}$ (in mm.)
0.5	-5.65922	0.44209
0.6	-5.52501	-0.206231
0.8	-5.86298	-0.206231

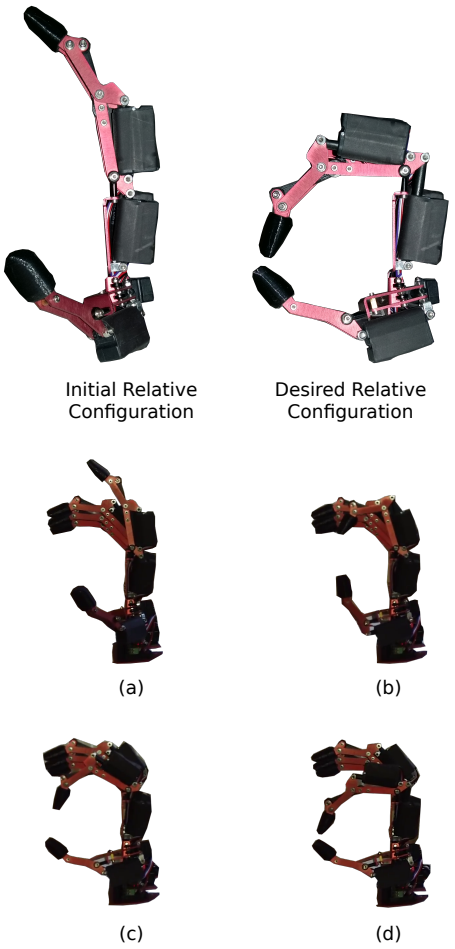


FIG. 3.6 Initial configuration, final desired configuration and state of fingers at different stages during the inverse kinematics control of relative pose between thumb and index fingertips.

have been provided. Similar to the index finger inverse kinematic control, the desired pose was obtained from a known achievable pose. The initial, desired relative configuration and different stages of robotic fingers motions from initial to the final desired relative configuration has been shown in Fig. 3.6. As can be seen in Fig. 3.4, the gain of $\lambda = 0.6$ was found to be most suited for relative pose control. However, an offset can be observed in the $ScrewError_{norm}$ in the Fig. 3.5, which corresponds to 5.5 mm and 0.2 mm in x and y axis respectively, as shown in Table 3.2. This is attributed to the fact that the relative task space consists of 6 dofs, while the combined system of index and thumb can only afford 4 dofs. Therefore it is important to reduce the task to lower dimension, for example by only giving position set-points instead of proving complete position and orientation set-points. Another issue encountered during relative pose control between index and thumb fingertips was the synchronization of the motion

of the fingers. It can be observed in Fig. 3.5 where even after achieving a lower error at 10 *sec.* mark, the error ($ScrewError_{norm}$) settles at a higher value. The synchronization between the fingers' motion can be achieved by setting a control frequency higher than the response time of both the fingers. In the current implementation the relative task-space inverse kinematics controller operates at 2 *Hz*.

3.4 Conclusion

The coupling in the motion of middle and distal phalanxes of a finger in an anthropomorphic robotic hand effected by a four-bar mechanism was modelled and inverse kinematic controller was designed. Additionally, the cooperative task-space modelling and control strategy based on screw-theory and UDQ developed in Chapter 2 was extended to the anthropomorphic hand for relative task space control between the fingertips of index and thumb for application in grasping tasks. The finger Jacobian comprising of coupled joints used in the inverse kinematics controller demonstrated exponential error convergence for the desired end-effector position. The relative pose control between index and thumb fingertip gave promising results, however demonstrated the need for further work, so that it can be used for real applications like in-hand manipulation. For the future work, we have identified following areas of improvements:

- *Calibration of fingers*: As we noted in the experimental validation section, the parameters used in the modeling were manually measured, since the manufacturer have not provided an accurate model. A vision based calibration method can be employed to estimate the joint screw axes parameters and the fingertips poses [78].
- *Task specific Jacobian*: Since the fingers have limited *dof*s, special care has to be taken about the control objectives. For example, positional distance Jacobian can be employed [13], since it is not really important to control the orientation of the frames in the case of grasping.
- *Integration with a robotic arm*: The kinematic model for the fingers developed in this work can easily be integrated with the kinematic model of a robotic arm, for example Baxter robot [64], to increase the applicability of the hand.

  A strategy for integrating the cooperative task-space control of the anthropomorphic hand with that of the arm it is attached to (see Fig. 3.7) can be proposed to

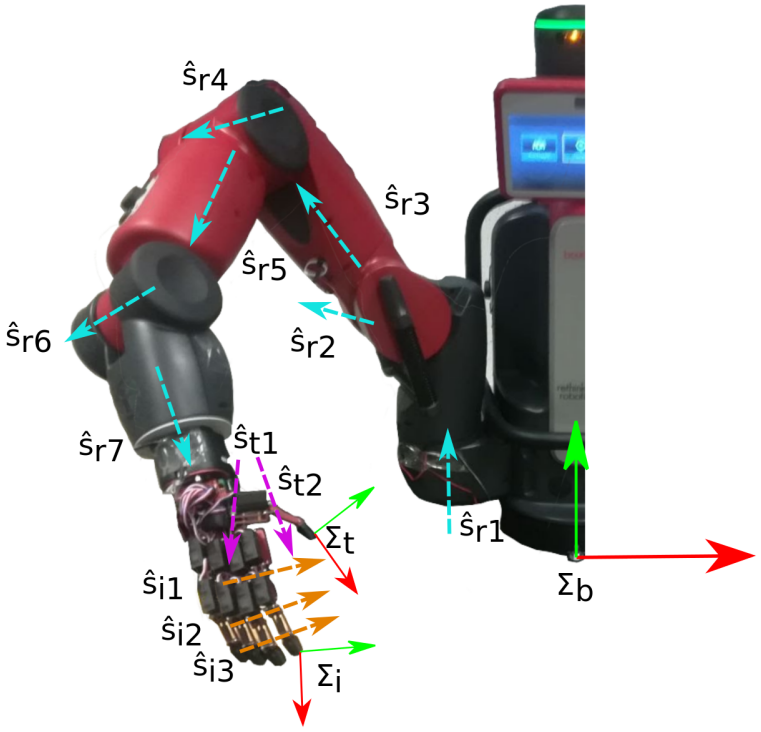


FIG. 3.7 Coordinate frames and screw axes for robotic hand-arm system consisting of Baxter robot and AR10 hand.

deal with the low dof of actuation available in the corresponding fingers. All the screw axes are given in the base frame of the robot, and it can be assumed that the pose transformation between the arm's end-effector frame $\boldsymbol{\Sigma_r}$, given as ${}^{r}\hat{\boldsymbol{x}}_{bh}$ is accurately known. To simplify the subsequent modeling, we assume that the robotic arms and robotic hands consists only of rotational joints.

The strategy to obtain relative pose, absolute pose, and relative Jacobian remains the same as given in section 2.3, while still keeping the index fingertip frame $\boldsymbol{\Sigma_i}$ as the reference frame. The absolute Jacobian utilizing the combined dof of hand and arm system can be derived as follows:

Let the Jacobian of the robotic arm $\boldsymbol{r}$ in base frame $\boldsymbol{\Sigma_r}$ is given as (refer A.5.2):

$$ {}^{b}\hat{\boldsymbol{J}}_r = \begin{bmatrix} \boldsymbol{L}_r & \boldsymbol{0} \\ \boldsymbol{M}_r & \boldsymbol{L}_r \end{bmatrix} \tag{3.11} $$

Similarly the Jacobian of the index finger ($\boldsymbol{i}$) and thumb ($\boldsymbol{t}$) of the anthropomorphic robotic hand in base frame $\boldsymbol{\Sigma_r}$ given in 3.2.1 can be given in simplified form as:

$$ {}^{b}\hat{\boldsymbol{J}}_i = \begin{bmatrix} \boldsymbol{L}_i & \boldsymbol{0} \\ \boldsymbol{M}_i & \boldsymbol{L}_i \end{bmatrix}, \tag{3.12} $$

and

$$ {}^{b}\hat{\boldsymbol{J}}_t = \begin{bmatrix} \boldsymbol{L}_t & \boldsymbol{0} \\ \boldsymbol{M}_t & \boldsymbol{L}_t \end{bmatrix} \tag{3.13} $$

We can assume the robotic arm as two manipulators $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$ for deriving the absolute Jacobian between these two manipulators, while in reality they are the same manipulators, each consisting of one of the fingers. Now the absolute velocity for these two extended manipulators can be given similar to (2.32) as:

$$ \begin{bmatrix} \boldsymbol{\omega}_{abs} \\ \boldsymbol{v}_{abs_0} \end{bmatrix} = \mathbf{J}_{abs} \begin{bmatrix} \underline{\dot{\boldsymbol{\theta}}_{r_a}} & \underline{\dot{\boldsymbol{\theta}}_i} & \underline{\dot{\boldsymbol{\theta}}_{r_b}} & \underline{\dot{\boldsymbol{\theta}}_t} \end{bmatrix}^T = [\underline{\dot{\boldsymbol{\theta}}_{total}}]^T \tag{3.14} $$

where,

$$ \mathbf{J}_{abs} = \begin{bmatrix} \frac{\boldsymbol{L}_{r_a}}{2} & \frac{\boldsymbol{L}_i}{2} & \frac{\boldsymbol{L}_{r_b}}{2} & \frac{\boldsymbol{L}_t}{2} \\ \frac{2\boldsymbol{M}_{r_a}+\boldsymbol{L}_{r_a}\times(\boldsymbol{p}_i-\boldsymbol{p}_t)}{4} & \frac{2\boldsymbol{M}_i+\boldsymbol{L}_i\times(\boldsymbol{p}_t-\boldsymbol{p}_i)}{4} & \frac{2\boldsymbol{M}_{r_b}+\boldsymbol{L}_{r_b}\times(\boldsymbol{p}_i-\boldsymbol{p}_t)}{4} & \frac{2\boldsymbol{M}_t+\boldsymbol{L}_t\times(\boldsymbol{p}_t-\boldsymbol{p}_i)}{4} \end{bmatrix} \tag{3.15} $$

The terms $\boldsymbol{p}_t$ and $\boldsymbol{p}_i$ refers to the position of thumb and index fingertip frames in the base frame.

Now, using (3.15) in (3.14) and noting that manipulators $\boldsymbol{r}_a$ and $\boldsymbol{r}_b$ are indeed the same, we can obtain the absolute Jacobian for robotic hand and arm system as:

$$ \mathbf{J}_{abs} = \begin{bmatrix} \boldsymbol{L}_r & \frac{\boldsymbol{L}_i}{2} & \frac{\boldsymbol{L}_t}{2} \\ \frac{2\boldsymbol{M}_r+\boldsymbol{L}_r\times(\boldsymbol{p}_i-\boldsymbol{p}_t)}{2} & \frac{2\boldsymbol{M}_i+\boldsymbol{L}_i\times(\boldsymbol{p}_t-\boldsymbol{p}_i)}{4} & \frac{2\boldsymbol{M}_t+\boldsymbol{L}_t\times(\boldsymbol{p}_t-\boldsymbol{p}_i)}{4} \end{bmatrix}, \tag{3.16} $$

where $\underline{\dot{\boldsymbol{\theta}}_{total}}$ is redefined as:

$$ [\underline{\dot{\boldsymbol{\theta}}_{total}}]^T = \begin{bmatrix} \underline{\dot{\boldsymbol{\theta}}_r} & \underline{\dot{\boldsymbol{\theta}}_i} & \underline{\dot{\boldsymbol{\theta}}_t} \end{bmatrix}^T. \tag{3.17} $$

Chapter 4

Resolved Acceleration Control Using Dual Quaternion

4.1 Introduction

Traditional control of robotic manipulators involves decoupling of position and orientation trajectory set-points, i.e. linear and angular position, velocity and acceleration terms of motion, results in an unnatural trajectory of the controlled end-effector manipulating an object [79]. A tighter control of position and orientation with consideration of the coupling effect inherent in a rigid body motion is vital for the success of certain manipulation tasks, such as welding of a curved surface. During such tasks the tool attached to the end-effector has to follow the surface while maintaining a specific orientation throughout the operation. A simultaneous pose control design can also improve performance in applications such as collision avoidance in a clattered environment [80], as it can be beneficial for reducing the *swept volume* of the end-effector [81], i.e. the volume generated as a result of the motion of end-effector [81].

The basis of such simultaneous pose controller defined on *SE(3)*, the Euclidean group of rigid motions, along with a strategy to deal with simultaneous pose trajectory tracking problem for a rigid body using homogeneous transformation matrices and exponential coordinates was given in [82]. The geometric properties of Lie groups (and corresponding Lie algebras) were exploited to generalize the classical PD control in a coordinate-free way. The difference in the convergence behaviour of simultaneous pose and conventional

control of linear and angular terms was reported in [80] for the motion on $SE(2)$ of a ground omnidirectional robot and it was concluded that the simultaneous treatment is a better choice when the synergy of position and orientation is important, i.e. applications where the position and orientation set points are desired to be achieved simultaneously.

This works presents the first instance of simultaneous pose computed torque controller design for task-space control of a manipulator which considers the inherent coupling between the translational and rotational aspects of a rigid body motion for controller design, rather than controlling the translational motion of a point on the rigid body and the angular motion of the rigid body separately. Dual quaternions have proved to be a compact and singularity-free alternative to homogeneous transformation matrices and are equally capable for handling the natural coupling between the translation and rotational motion of rigid bodies [83, 84, 85, 86, 50, 53, 13, 14]. The computational and storage advantages of using unit dual quaternions for a manipulator's kinematic modelling and control have been established in Table 2.1. This work extends the UDQ and screw-based kinematics discussed in A.5 and Chapter 2 for the tracking of acceleration, velocity and pose set-points for end-effector of a robotic manipulator.

Previous works in the direction of task space regulation of manipulators have been presented in section 4.2. Both simultaneous pose and conventional task space regulation schemes have been presented in section 4.3. The validation strategy for the proposed simultaneous pose tracking scheme for task space regulation has been presented in 4.4, where the conclusions and future work are presented in section 4.5.

4.2 Related work on task space regulation for manipulators

In this section we will revisit the previous works related to second order trajectory trackers design in the domain of manipulators and will highlight potential improvements simultaneous pose control might over these methods.

There are mainly two methods available in literature related to trajectory tracking of a manipulator, which are: computed torque control with resolved-acceleration control [87, 88, 89, 90, 91];, and operational space control [92, 93]. These approaches differ

mainly in their method of treating singularities and redundancies. The redundancy of manipulators is treated as a problem of resolving the end-effector desired motion into joint motions with respect to some criteria, such as avoidance of joint limits, kinematic singularities etc in resolved-acceleration control. Whereas in operational space framework the manipulator is treated as a redundant mechanism in the neighbourhood of a singular configuration with respect to its end-effector motion in the subspace of operational space. The subspace of the operational space is orthogonal to its singular direction and can be obtained using the kinematic characteristic of the manipulator's Jacobian matrix.

One of the first implementations for task-space regulation of a serial manipulator was given in [87], where joint torques were obtained for the tracking of a desired trajectory by a manipulator using inverse dynamic model of the manipulator. While the dynamics of position error was obtained in a straightforward manner, the dynamics of orientation errors was resolved only for small orientational error with the assumption that only one of the axis of desired coordinates is misaligned.

A new approach for resolved-rate and resolved-acceleration controllers capable of suppressing high joint velocities near singularities was proposed in [88]. By the virtue of damped pseudo-inverse of manipulator Jacobian, the proposed control scheme reduces the joint rates while inducing minimal errors in the end-effector motion. However, this controller scheme is incapable of dealing with singularities where the rate of convergence in the singular directions is reduced. Potential functions composed of position and orientation were constructed for stability and convergence analysis. A family of potential functions for orientational error that would bring the manipulator from current to the desired rotated configuration were proposed, although authors assumed the orientation error to remain bounded during closed-loop control.

Resolved-acceleration technique for the tracking control problem of robot manipulators in the task space was reviewed in [90], where different forms of orientation error were compared in terms of tracking performance and representation singularity avoidance. An alternative method of extracting Euler terms for orientation from mutual rotation matrix was advocated over classical method of getting the difference of corresponding Euler angles. The proposed approach only presented singularity problems for high orientation error. The authors also proposed quaternion based orientation error dynamics along

with angle-axis based approach. While quaternion based approach was found to be capable of singularity avoidance, it led to non-linear closed-loop system. The angle-axis approach suffered from singularity. The tracking performance for quaternion feedback and the angle-axis feedback schemes was found to be similar to the proposed Euler angles based method.

Owing to its non-singular representation of orientation, the quaternion feedback based method has since been the preferred method for formulate orientation dynamics [91, 59, 94] for resolved-acceleration controller design, as well as for operation space formulation of manipulator task-space regulation problem [93]. However, none of the above work have addressed the inherent coupling of rotational and linear motion for a rigid body for designing the control law for end-effector trajectory tracking.

In the current work we take advantage of singularity-free attribute of quaternions, compactness and efficiency of screw-based based kinematics and simultaneous treatment of angular and linear motion variables of second-order trajectory using UDQ to propose a new resolved acceleration control and compare its trajectory tracking performance with above mentioned conventional controllers. In the following section we will describe the design of both control strategies in detail.

4.3 Controller design for task-space regulation

In this section controller design for conventional task space regulator has been presented, which would later serve as a benchmark for the performance analysis of the proposed controller.

4.3.1 Resolved Acceleration Control

The dynamic model of a serial manipulator consisting on n joints in joint space is given as:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\Gamma}, \tag{4.1}$$

where, $\boldsymbol{H}(\boldsymbol{q})$ is an $(n \times n)$ symmetric and positive definite inertial matrix and varies with the joint positions $\boldsymbol{q}$. $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is an $(n \times 1)$ vector of Coriolis and centrifugal torques, whereas $\boldsymbol{G}(\boldsymbol{q})$ consists of gravity torques. $\boldsymbol{\Gamma}$ is an $(n \times 1)$ vector of joint driving torques.

The differential kinematics relation for a manipulator that relates the joint velocities $\hat{\underline{\boldsymbol{\theta}}}$ to the end-effector twist $\hat{\boldsymbol{\omega}}$ is given as:

$$\hat{\boldsymbol{\omega}} = \hat{\boldsymbol{J}}\, \dot{\hat{\underline{\boldsymbol{\theta}}}} \tag{4.2}$$

The end-effector acceleration $\hat{\boldsymbol{a}}$ is obtained by differentiating (4.2) with respect of time:

$$\hat{\boldsymbol{a}} = \dot{\hat{\boldsymbol{\omega}}} = \hat{\boldsymbol{J}}\, \ddot{\hat{\underline{\boldsymbol{\theta}}}} + \dot{\hat{\boldsymbol{J}}}\, \dot{\hat{\underline{\boldsymbol{\theta}}}} \tag{4.3}$$

where the twist and acceleration terms consists of both linear and angular terms:

$$\hat{\boldsymbol{\omega}} = \begin{bmatrix} \boldsymbol{\omega}^{\mathsf{T}} & \boldsymbol{v}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \tag{4.4}$$

$$\hat{\boldsymbol{a}} = \hat{\boldsymbol{a}}_{cmd} = \begin{bmatrix} \boldsymbol{\alpha}_{cmd}^{\mathsf{T}} & \boldsymbol{a}_{cmd}^{\mathsf{T}} \end{bmatrix}^{\mathsf{T}} \tag{4.5}$$

Subsequently, the acceleration of the joints of the manipulator can be computed from acceleration control command as:

$$\ddot{\hat{\underline{\boldsymbol{\theta}}}}_{cmd} = \ddot{\boldsymbol{q}}_{cmd} = \hat{\boldsymbol{J}}^{-1} \left(\hat{\boldsymbol{a}}_{cmd} - \dot{\hat{\boldsymbol{J}}}\, \dot{\hat{\underline{\boldsymbol{\theta}}}} \right). \tag{4.6}$$

Now $\hat{\boldsymbol{a}}_{cmd}$ can be used to compute the joint torque commands for the inverse dynamic model given in (4.1). Assuming the perfect knowledge of the involved dynamic parameters and inverting (4.3) and using $\hat{\boldsymbol{a}}_{cmd}$ as the control input in (4.1), the driving torque can be computed as:

$$\boldsymbol{\Gamma} = \boldsymbol{H}(\boldsymbol{q}) \hat{\boldsymbol{J}}^{-1} (\hat{\boldsymbol{a}}_{cmd} - \dot{\hat{\boldsymbol{J}}}\, \dot{\hat{\underline{\boldsymbol{\theta}}}}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) \tag{4.7}$$

Since Jacobian inverse is not defined for singular configurations, a damped least-squares inverse can be adopted for robustness in the neighbourhood of kinematic singularities, whereas a pseudo-inverse can be used in the redundant case ($n > 6$) in conjunction with

a suitable term in the null space of the Jacobian describing the internal motion of the manipulator.

Note that the dual angle $\hat{\theta}_i$ used in (4.3) is converted to q_i in (4.1), depending on the type of joint (see (A.52)).

4.3.2 Conventional and Simultaneous Pose Task-Space Regulator Design

As discussed in the last section, the inverse dynamic model can be used to obtain joint torques for the input $\hat{\mathbf{a}}_{cmd}$, commonly referred to as resolved acceleration. It is desired to construct a control dynamics for positional and orientation trajectory with a proper input command, that will ensure a stable and reliable tracking of the desired trajectory. In this section we present the conventional position and orientation task-space regulation design from the existing works, and then present the strategy for simultaneous pose task-space regulation inspired from [50].

4.3.2.1 Conventional controller design

Consider a desired trajectory for a manipulator's end-effector motion given by $\hat{\boldsymbol{x}}_d$, $\hat{\boldsymbol{\omega}}_d$, and $\hat{\boldsymbol{a}}_d \in \mathbf{R}^6$, which are desired pose, velocity and acceleration terms containing both positional and orientation components and given in a static base frame. Note that these terms considered as 6D arrays in for the explanation of conventional controller design in this section.

$$\hat{\boldsymbol{x}}_d = \begin{bmatrix} \boldsymbol{o}_d{}^\intercal & \boldsymbol{p}_d{}^\intercal \end{bmatrix}^\intercal \tag{4.8}$$

$$\hat{\boldsymbol{\omega}}_d = \begin{bmatrix} \boldsymbol{\omega}_d{}^\intercal & \dot{\boldsymbol{p}}_d{}^\intercal \end{bmatrix}^\intercal \tag{4.9}$$

$$\hat{\boldsymbol{a}}_d = \begin{bmatrix} \dot{\boldsymbol{\omega}}_d{}^\intercal & \ddot{\boldsymbol{p}}_d{}^\intercal \end{bmatrix}^\intercal \tag{4.10}$$

Similarly the terms for actual trajectory related to end-effector motion, i.e. $\hat{\boldsymbol{x}}_c$, $\hat{\boldsymbol{\omega}}_c$ can be defined. It is important to define rotation matrices corresponding to the orientation terms of desired and actual trajectory, which we denote with $\boldsymbol{R}_d$ and $\boldsymbol{R}_c$, respectively. The orientation components $\boldsymbol{o}_c{}^\intercal$ and $\boldsymbol{o}_d{}^\intercal$ can be derived from the corresponding rotation matrices based on the representation chosen from the 12 possible definitions of the Euler

angles, or by using A.41 and A.46 given in the appendix A.4.2.1 if angle-axis method of orientation representation is used.

Position tracking scheme:

Given a position trajectory, the position error is given with the difference of desired and current position vectors as:

$$\boldsymbol{p}_e = \boldsymbol{p}_d - \boldsymbol{p}_c \tag{4.11}$$

For the following choice of resolved linear acceleration:

$$\boldsymbol{a}_{cmd} = \ddot{\boldsymbol{p}}_d + \boldsymbol{\lambda}_{Vp}\dot{\boldsymbol{p}}_e + \boldsymbol{\lambda}_{Pp}\boldsymbol{p}_e \tag{4.12}$$

an exponentially stable closed loop dynamics is obtained, as long as the proportional and derivative gains corresponding to closed-loop control of translational motion, i.e. $\boldsymbol{\lambda}_{Pp}$ and $\boldsymbol{\lambda}_{Vp}$, have positive elements.

$$\ddot{\boldsymbol{p}}_e + \boldsymbol{\lambda}_{Vp}\,\dot{\boldsymbol{p}}_e + \boldsymbol{\lambda}_{Pp}\,\boldsymbol{p}_e = \boldsymbol{0} \tag{4.13}$$

Orientation tracking scheme:

While the choice of position error is straightforward to obtain an stable closed loop dynamics, there are many different possibilities for orientation error selection.

- *Euler Angle feedback*: The most common approach to represent a relative orientation is rotation matrices based on Euler angles. While this approach of representing rotation is intuitive, it suffers from representation singularity which means that for certain configurations the extraction of Euler angles is not possible. $\boldsymbol{\lambda}_{Po}$ and $\boldsymbol{\lambda}_{Vo}$ represent the proportional and derivative gains corresponding to closed-loop control of angular motion of the end-effector in the subsequent equations.
  - *Classical operational space approach*: A classical approach to obtain stable closed-loop orientation error dynamics is by deriving the Euler angles from the rotation matrices of desired and actual trajectory and getting the numerical

difference.

$$\boldsymbol{o}_e = \boldsymbol{o}_d - \boldsymbol{o}_c \tag{4.14}$$

The resolved angular acceleration is given as:

$$\boldsymbol{\alpha}_{cmd} = \boldsymbol{T}(\boldsymbol{o}_e)(\ddot{\boldsymbol{o}}_d + \boldsymbol{\lambda}_{V_o}\dot{\boldsymbol{o}}_e + \boldsymbol{\lambda}_{P_o}\boldsymbol{o}_e) + \dot{\boldsymbol{T}}(\boldsymbol{o}_e, \dot{\boldsymbol{o}}_e)\dot{\boldsymbol{o}}_e \tag{4.15}$$

where, $\boldsymbol{T}(\boldsymbol{o}_e)$ is a transformation matrix that relates the angular velocity error to the time derivative of the error represented using Euler angles:

$$\boldsymbol{\omega}_e = \boldsymbol{T}(\boldsymbol{o}_e)\dot{\boldsymbol{o}}_e \tag{4.16}$$

$$\dot{\boldsymbol{\omega}}_e = \dot{\boldsymbol{T}}(\boldsymbol{o}_e, \dot{\boldsymbol{o}}_e)\dot{\boldsymbol{o}}_e + \boldsymbol{T}(\boldsymbol{o}_e)\ddot{\boldsymbol{o}}_e \tag{4.17}$$

The above resolved angular acceleration scheme becomes ill-conditioned when the actual or desired end-effector orientations are near their corresponding representation singularities.

– *Alternative Euler angles feedback*: An alternative feedback scheme for orientation tracking was proposed in [90] to mitigate the issues with representation singularities in classical Euler feedback scheme. An error rotation matrix in the actual end-effector frame can be first derived from the desired and current rotation matrices:

$${}^{c}\boldsymbol{R}_e = \boldsymbol{R}_c^{\mathrm{T}}\boldsymbol{R}_d \tag{4.18}$$

Using this error rotation matrix and with proper choice of Euler representation the representation singularity will only come into effect when there are large orientation error, as compared to last approach where there is chance of running into representation singularity while deriving orientation components from current as well as desired rotation matrices.

Now, similar to classical Euler Angle feedback, after deriving the Euler angles ($\boldsymbol{o}_e$) from the error rotation matrix ${}^{c}\boldsymbol{R}_e$ and taking time derivative of 4.18 and using expression 4.16 after computing the transformation matrix $\boldsymbol{T}_e(\boldsymbol{o}_e)$

in the static base frame:

$$\boldsymbol{T}_e(\boldsymbol{o}_e) = \boldsymbol{R}_c\, {}^c\boldsymbol{T}_e(\boldsymbol{o}_e), \tag{4.19}$$

following feedback scheme for closed-loop orientation control can be adopted:

$$\boldsymbol{\alpha}_{cmd} = \ddot{\boldsymbol{\omega}}_d - \dot{\boldsymbol{T}}_e(\boldsymbol{o}_e\dot{\boldsymbol{o}}_e)\dot{\boldsymbol{o}}_e + \boldsymbol{T}_e(\boldsymbol{o}_e)(\boldsymbol{K}_{Vo}\dot{\boldsymbol{o}}_e + \boldsymbol{K}_{Po}\boldsymbol{o}_e), \tag{4.20}$$

where $\boldsymbol{T}_e(\boldsymbol{o}_e)$ is defined similar to (4.16), and

- *Angle-axis approach*: The axis-angle representation of a rotation represents a rotation in 3D Euclidean space by a unit vector $\boldsymbol{r}$ representing the axis of rotation and an angle θ describing the magnitude of the rotation about the axis, and can be represented as a rotation matrix as follows:

$$\boldsymbol{R} = \mathbf{I} + (\sin\theta)[\boldsymbol{r}]_\times + (1 - \cos\theta)[\boldsymbol{r}]_\times^2 \tag{4.21}$$

where $[\boldsymbol{r}]_\times$ is the cross-product operator matrix corresponding to the axis of rotation $\boldsymbol{r}$ and $\mathbf{I}$.

Two equivalent versions of orientation error were defined in [87]:

$$\boldsymbol{o}_e = \frac{1}{2}([\boldsymbol{n}_c]_\times \boldsymbol{n}_d + [\boldsymbol{s}_c]_\times \boldsymbol{s}_d + [\boldsymbol{a}_c]_\times \boldsymbol{a}_d),\ \textit{and} \tag{4.22}$$

$$= \sin(\theta_e)\boldsymbol{r}_e \tag{4.23}$$

where $\boldsymbol{n}_d$, $\boldsymbol{s}_d$, $\boldsymbol{a}_d$ and $\boldsymbol{n}_c$, $\boldsymbol{s}_c$ $\boldsymbol{a}_c$ are the unit vectors of the axes corresponding to the rotation matrices ($\boldsymbol{R} = [\boldsymbol{n}\ \boldsymbol{s}\ \boldsymbol{a}]$) related to the desired and actual pose, respectively. In the second expression, $\boldsymbol{r}_e$ and θ_e are the angle-axis parameters related to the angle-axis representation of mutual rotation in the common reference frame, given as:

$$\boldsymbol{R}_e = \boldsymbol{R}_d\boldsymbol{R}_c^\intercal \tag{4.24}$$

Taking the first and second time derivative of (4.22), following resolved angular acceleration can be chosen:

$$\boldsymbol{\alpha}_{cmd} = \boldsymbol{L}^{-1}(\boldsymbol{L}^\intercal\dot{\boldsymbol{\omega}}_d + \dot{\boldsymbol{L}}^\intercal\boldsymbol{\omega}_d - \dot{\boldsymbol{L}}\boldsymbol{\omega}_c + \boldsymbol{K}_{Vo}\dot{\boldsymbol{o}}_e + \boldsymbol{K}_{Po}\boldsymbol{o}_e) \tag{4.25}$$

where

$$\boldsymbol{L} = -\frac{1}{2}([\boldsymbol{n}_d]_\times[\boldsymbol{n}_c]_\times + [\boldsymbol{s}_d]_\times[\boldsymbol{s}_c]_\times + [\boldsymbol{a}_d]_\times[\boldsymbol{a}_c]_\times). \tag{4.26}$$

The above mentioned control command $\boldsymbol{\alpha}_{cmd}$ is stable in the interval $\theta_e = (-\pi/2,\, \pi/2)$, but is ill-defined when $\boldsymbol{L}$ is singular, which occurs at $\theta_e = \pm\pi/2$. In addition to that, another drawback of this approach is the high computation cost of obtaining $\boldsymbol{\alpha}_{cmd}$ in (4.25).

- *Quaternion feedback*: The representation singularity issues related to the above mentioned approaches for the computation of resolved angular acceleration can be mitigated by using the vector part of the quaternion related to the mutual rotation matrix between desired and actual orientation represented in the base frame in (4.24).

  Let quaternion $\boldsymbol{q}_e = s_e + \boldsymbol{v}_e$ corresponds to the rotation matrix $\boldsymbol{R}_e$ in (4.24). Then the resolved angular acceleration can be given as:

$$\boldsymbol{\alpha}_{cmd} = \dot{\boldsymbol{\omega}}_d + \boldsymbol{K}_{Vo}\boldsymbol{\omega}_e + \boldsymbol{K}_{Po}\boldsymbol{v}_e \tag{4.27}$$

  where $\boldsymbol{\omega}_e$ the angular velocity error of the end-effector frame is given as:

$$\boldsymbol{\omega}_e = \boldsymbol{\omega}_d - \boldsymbol{\omega}_c. \tag{4.28}$$

  Lyapunov analysis was invoked for the stability analysis of the above mentioned control law in [90]. Although quaternion feedback approach does not present any representation singularity issues, the proposed control law is non-linear.

4.3.2.2 Unit Dual Quaternion based Simultaneous Pose Controller Design

Given current ($\hat{\boldsymbol{x}}_c$) and desired ($\hat{\boldsymbol{x}}_c$) poses represented with UDQ and both expressed in a static base frame, there are four possible permutations for error computation.

The error UDQ used in our formulation is:

$$\hat{\boldsymbol{x}}_e = \hat{\boldsymbol{x}}_d\, \hat{\boldsymbol{x}}_c^* = {}^b\hat{\boldsymbol{x}}_{c\to d}. \tag{4.29}$$

The screw axis related this error represents a screw displacement vector, directed from the current frame c to the desired frame d, expressed in the base frame b. Which means, if we transform the error screw axis derived from $\hat{\boldsymbol{x}}_e$ using expressions given in A.4.2.1, from the base frame to the current frame using $Ad_{({}^b\hat{\boldsymbol{x}}_c^*)}$:

$$\begin{aligned} {}^c\hat{\boldsymbol{s}}_e &= Ad_{({}^b\hat{\boldsymbol{x}}_c^*)}\, {}^b\hat{\boldsymbol{s}}_e = {}^b\hat{\boldsymbol{x}}_c^*\, {}^b\hat{\boldsymbol{s}}_e\, {}^b\hat{\boldsymbol{x}}_c \\ &= Ad_{({}^c\hat{\boldsymbol{x}}_b)}\, {}^b\hat{\boldsymbol{s}}_e = {}^c\hat{\boldsymbol{x}}_b\, {}^b\hat{\boldsymbol{s}}_e\, {}^c\hat{\boldsymbol{x}}_b^*, \end{aligned} \tag{4.30}$$

and then postmultiply the corresponding UDQ, obtained using (A.36), to the current frame pose $\hat{\boldsymbol{x}}_c$, we will achieve the desired pose, *i.e.*:

$$\hat{\boldsymbol{x}}_d = \hat{\boldsymbol{x}}_c \exp\left(\frac{{}^c\hat{\theta}_e}{2}\, {}^c\hat{\boldsymbol{s}}_e\right). \tag{4.31}$$

The same interpretation can also be applied to unit quaternion errors. For example, in [90], the vector part of the quaternion derived from the current and desired rotation matrices was used, which is equivalent to ($\boldsymbol{q}_e = \boldsymbol{q}_c^*\boldsymbol{q}_d = {}^c\boldsymbol{q}_{c\to d}$), and thus had to be transformed to the base frame. Instead, (${}^b\mathbf{q}_{c\to d} = \boldsymbol{q}_d\boldsymbol{q}_c^*$) could have directly been used with the same effect.

We follow the feedback linearisation approach developed in for the attitude and position tracking problem of a rigid body. The goal is to obtain an $\hat{\boldsymbol{a}}_{cmd}$ for obtaining the appropriate joint driving torques, such that the pose and spatial velocity of the end-end effector of serial manipulator will converge to a desired pose with a desired spatial velocity.

Taking the derivative of the error UDQ (4.29), we obtain:

$$\dot{\hat{\boldsymbol{x}}}_e = \dot{\hat{\boldsymbol{x}}}_d\hat{\boldsymbol{x}}_c^* + \hat{\boldsymbol{x}}_d\dot{\hat{\boldsymbol{x}}}_c^*. \tag{4.32}$$

Now, it has been proved in [80] that:

$$\dot{\hat{\boldsymbol{x}}} = \frac{1}{2}\hat{\boldsymbol{x}}\hat{\boldsymbol{\omega}}, \tag{4.33}$$

where, $\hat{\boldsymbol{\omega}}$ is the screw velocity of a rigid body given in the reference frame, whose pose of the rigid body is given with UDQ $\hat{\boldsymbol{x}}$. The time derivative of its conjugate $\hat{\boldsymbol{x}}^*$ is given

as:

$$\dot{\hat{\boldsymbol{x}}}^{*} = -\frac{1}{2}\,\hat{\boldsymbol{x}}^{*}\hat{\boldsymbol{\omega}}. \tag{4.34}$$

Similarly to [80], following can be derived:

$$\dot{\hat{\boldsymbol{x}}}_e = \frac{1}{2}\hat{\boldsymbol{x}}_e\hat{\boldsymbol{\omega}}_e, \tag{4.35}$$

where,

$$\hat{\boldsymbol{\omega}}_e = Ad_{(\hat{\boldsymbol{x}}_e^{*})}\hat{\boldsymbol{\omega}}_d - \hat{\boldsymbol{\omega}}_c \tag{4.36}$$

The adjoint operation used in the above expression is given in 4.30.

Taking the derivative of 4.36, we obtain:

$$\dot{\hat{\boldsymbol{\omega}}}_e = \dot{Ad}_{(\hat{\boldsymbol{x}}_e^{*})}\hat{\boldsymbol{\omega}}_d + Ad_{(\hat{\boldsymbol{x}}_e^{*})}\dot{\hat{\boldsymbol{\omega}}}_d - \dot{\hat{\boldsymbol{\omega}}}_c \tag{4.37}$$

After expanding the involved terms in the above expression we obtain:

$$\dot{\hat{\boldsymbol{\omega}}}_e = Ad_{(\hat{\boldsymbol{x}}_e^{*})}\dot{\hat{\boldsymbol{\omega}}}_d - \dot{\hat{\boldsymbol{\omega}}}_c + \widehat{\boldsymbol{vec}}_{\hat{\boldsymbol{\omega}}_c} Ad_{(\hat{\boldsymbol{x}}_e^{*})}\hat{\boldsymbol{\omega}}_d \tag{4.38}$$

where, $\widehat{\boldsymbol{vec}}_{\hat{\boldsymbol{x}}_1}\hat{\boldsymbol{x}}_2$ operation refers to dual quaternion product $\hat{\boldsymbol{x}}_1\hat{\boldsymbol{x}}_2$, but with null scalar terms in both real and dual parts.

If following $\hat{\boldsymbol{a}}_{cmd}$ is chosen for feedback linearisation:

$$\begin{aligned}\hat{\boldsymbol{a}}_{cmd} =& 2\,\boldsymbol{\lambda}_P \ln \hat{\mathbf{x}}_e + \boldsymbol{\lambda}_V\hat{\boldsymbol{\omega}}_e \\ &+ \widehat{\boldsymbol{vec}}_{\hat{\boldsymbol{\omega}}_c} Ad_{(\hat{\mathbf{x}}_e^{*})}\hat{\boldsymbol{\omega}}_d + Ad_{(\hat{\boldsymbol{x}}_e^{*})}\dot{\hat{\boldsymbol{\omega}}}_d,\end{aligned} \tag{4.39}$$

and substituted in the place of $\dot{\hat{\boldsymbol{\omega}}}_c$ in 4.38, we obtain the following error dynamics:

$$\dot{\hat{\boldsymbol{\omega}}}_e + \boldsymbol{\lambda}_V\hat{\boldsymbol{\omega}}_e + 2\,\boldsymbol{\lambda}_P \ln \hat{\boldsymbol{x}}_e = 0 \tag{4.40}$$

The asymptotic stability of equilibrium point $(\ln(\hat{\boldsymbol{x}}_e), \hat{\boldsymbol{\omega}}_e) = (\hat{\mathbf{0}}, \hat{\mathbf{0}})$ for the above system was established in [50] for an appropriate choice of the gains $\boldsymbol{\lambda}_P$ and $\boldsymbol{\lambda}_V$. Therefore, the end-effector motion will eventually converge to the desired trajectory for the control

command obtained in (4.39). The *two equilibria problem* for dual quaternion has been discussed in [50], where the system (4.40) has two identical equilibria at $\hat{\boldsymbol{x}}_e = (\mathbf{I}, \mathbf{0})$ and $(-\mathbf{I}, \mathbf{0})$, which was resolved by multiplying the error UDQ with -1 if the scalar part of the real quaternion related to the error UDQ is negative.

The expression $2\,\boldsymbol{\lambda}_P \ln \hat{\boldsymbol{x}}_e$ is the same control law used in chapter 2 for kinematic control of manipulators, and refers to the product of dual angle and unit dual vector pertaining to the axis of the screw obtained by deriving the screw parameters from the error UDQ $\hat{\boldsymbol{x}}_e$ (see section A.4.2.1).

4.3.2.3 Formulation of Jacobian Derivative

In order to compute the driving torque command in 4.1 from the control law $\hat{\boldsymbol{a}}_{cmd}$ obtained in 4.39, the computation of $\dot{\hat{\boldsymbol{J}}}$ is required (see 4.7).

Derivative of the Jacobian $\hat{\boldsymbol{J}}$ of a serial robotic manipulator can be obtained as in [95], assuming a joint screw axis $\hat{\boldsymbol{s}}_i$ represented as unit dual line, is fixed on the child link i:

$$\begin{aligned}\dot{\hat{\boldsymbol{J}}} &= \begin{bmatrix}\dot{\hat{\boldsymbol{s}}}_1 & \dot{\hat{\boldsymbol{s}}}_2 & \cdots & \dot{\hat{\boldsymbol{s}}}_n\end{bmatrix} \\ &= \begin{bmatrix}\widehat{\boldsymbol{vec}}_{\hat{\boldsymbol{\omega}}_1}\hat{\boldsymbol{s}}_1 & \widehat{\boldsymbol{vec}}_{\hat{\boldsymbol{\omega}}_2}\hat{\boldsymbol{s}}_2 & \cdots & \widehat{\boldsymbol{vec}}_{\hat{\boldsymbol{\omega}}_n}\hat{\boldsymbol{s}}_n\end{bmatrix},\end{aligned} \tag{4.41}$$

where, $\hat{\boldsymbol{\omega}}_i$ represents the screw velocity of an i_{th} link, and can be computed by summing the twists caused by the joints between the base frame and the i_{th} link in a way similar to 4.2.

4.4 Experimental Validation

The simultaneous pose resolved acceleration control obtained in section 4.3.2.2 was validated on one of the redundant arms of Baxter dual arm collaborative robot [63]. Note that the presented implementation does not address redundancy resolution for the extra joint in the used robotic arm. Instead, the trajectory is defined such that the self-motion of some of the joints due to redundancy does not affect the manipulator motion due to imperfect gravity compensation.

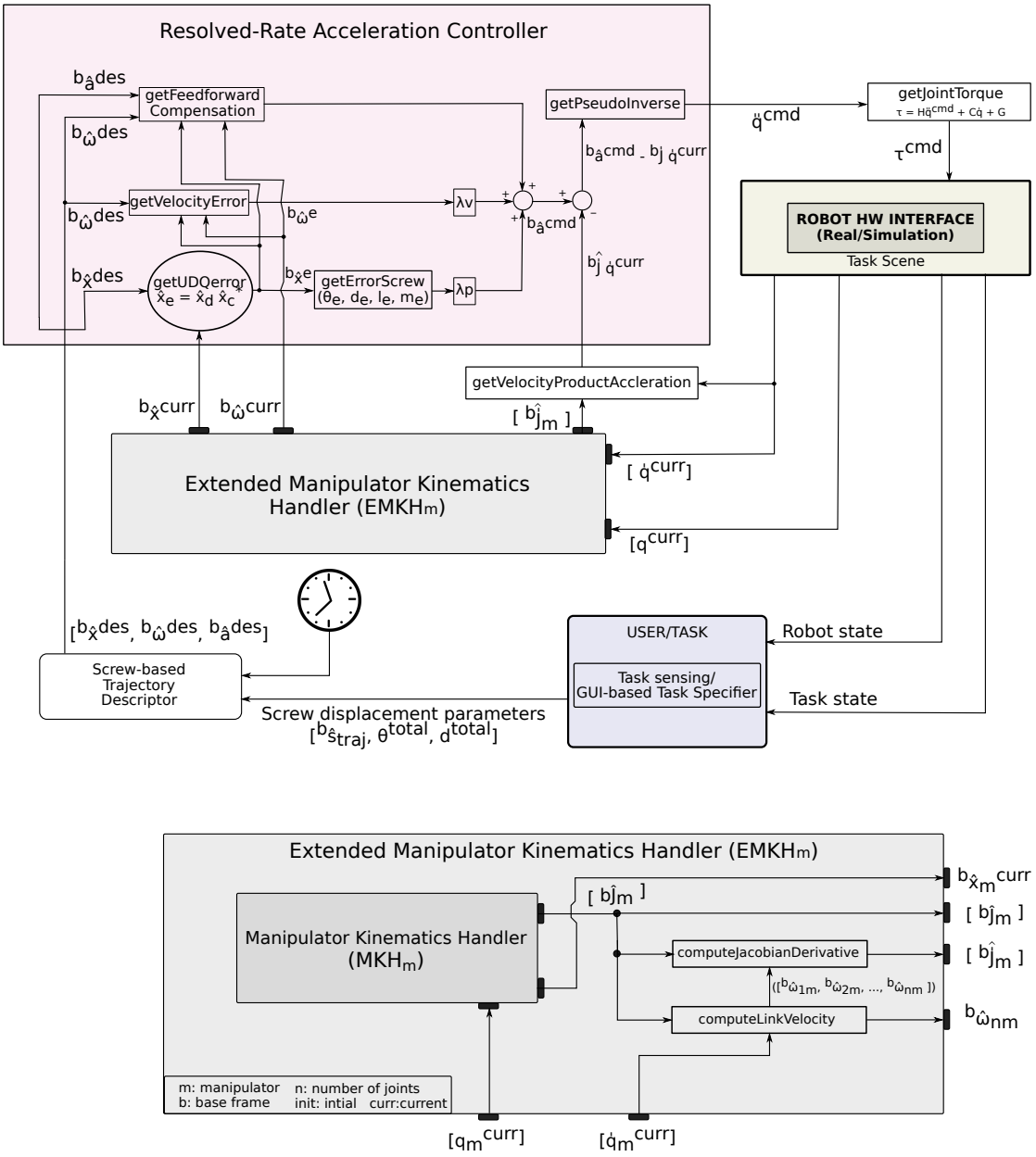


FIG. 4.1 Resolved rate acceleration control strategy for single manipulator.

The control architecture implemented for the comparative performance analysis of PD controller described in previous section using dual quaternions has been shown in Fig. 4.1. Note that compared to the CTS architecture shown in Fig. 2.3 in chapter 2 an *Extended Manipulator Kinematics Handler* is used. It computes Jacobian derivative from the initial screw axes, current joint positions and velocities using 4.41, in addition to the manipulator Jacobian.

The *Resolved-Rate Acceleration Controller* module implements the control law discussed in section 4.3.2.2 to obtain the acceleration command in 4.39. The required joint torque

command was obtained using inverse dynamic model of the manipulator in *getJoint-Torque* module using pseudo-inverse of the manipulator Jacobian and *velocity-product acceleration* obtained using the Jacobian derivative and joint speeds.

While in the future it is desired to receive the set-points for the desired end-effector trajectory from a sensing module and task-planner, in the current implementation we have used user-defined trajectory to validate the proposed trajectory tracking strategy for the manipulator. Concepts from screw theory were used to generate parametrised trajectory in *Screw-based Trajectory Descriptor* module to generate pose, velocity and acceleration set-points to be used in the *Resolved-Rate Acceleration Controller* module. The trajectory generation module uses a third-order polynomial for interpolation of pose and velocities and is capable of generating second order trajectories such as rotation, translation and screw motion, i.e. desired pose, velocity and acceleration. The cubic function parametrized by parameters of screw motion, and initial desired pose ensures zero initial and final velocity given the total displacement and the time allotted for the motion [96].

Third order polynomial trajectory

Since a screw displacement is defined as a rotation (θ_{task}) and translation (d_{task}) along same screw axis, it can be parametrized with a dual number, $\hat{\theta}_{task} = \theta_{task} + \varepsilon d_{task}$, containing both terms. The position and velocity terms in the desired motion are the functions of $\hat{\theta}_{task}$ and, $\dot{\hat{\theta}}_{task}$, as given in A.51 and A.58. Given total time and the current time, t_f and t respectively, a trajectory may be specified by assigning initial and final conditions for a dual parameter containing both translational and angular displacement terms. The four desired boundary conditions for position and velocity require the screw displacement parameter, $\hat{\theta}_{task}(t)$, to be a cubic polynomial function of time.

$$\hat{\theta}_{task}(t) = \hat{a}_0 + \hat{a}_1 t + \hat{a}_2 t^2 + \hat{a}_3 t^3 \tag{4.42}$$

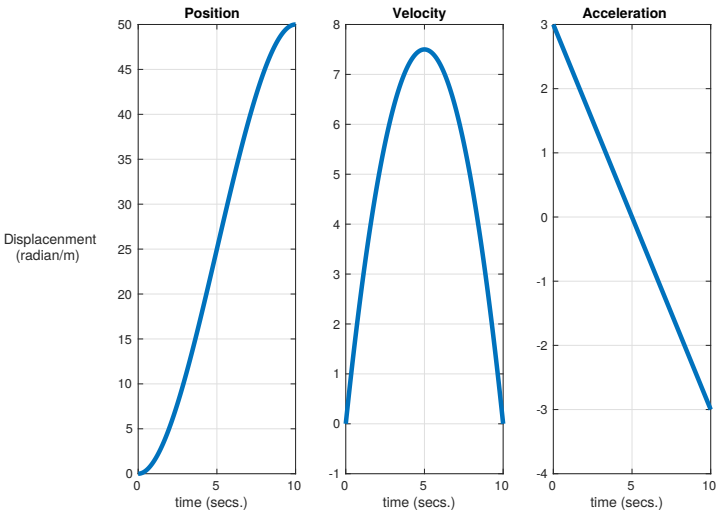


FIG. 4.2 Visualization of position, velocity and acceleration trajectories.

Solving for the boundary conditions reveals following expressions for the coefficients $\hat{a}_0$, $\hat{a}_1$, $\hat{a}_2$, $\hat{a}_3$.

$$\begin{aligned}
\hat{a}_0 &= \hat{\theta}_{task_i}, \\
\hat{a}_1 &= \dot{\hat{\theta}}_{task_i}, \\
\hat{a}_2 &= \frac{-3(\hat{\theta}_{task_i} - \hat{\theta}_{task_f}) - (2\dot{\hat{\theta}}_{task_i} + \dot{\hat{\theta}}_{task_f})t_f}{{t_f}^2}, \\
\hat{a}_3 &= \frac{2(\hat{\theta}_{task_i} - \hat{\theta}_{task_f}) + (\dot{\hat{\theta}}_{task_i} + \dot{\hat{\theta}}_{task_f})t_f}{{t_f}^3},
\end{aligned} \tag{4.43}$$

The evolution of position, velocity and acceleration of the one of the parameters belonging to the dual screw displacement ($\hat{\theta}_{task}$), i.e. θ_{task} or d_{task}, generated using above method has been plotted in Fig. 4.2 for a time period of 10 seconds, where the initial and final velocity is zero and initial and final positions are 0 and 50 respectively. Now, a second order trajectory can be generated, given a 6D screw axis, i.e. a Plücker line in an known frame, and the third order polynomial trajectory generator, as discussed in section A.5.1 and A.5.2.

Comparative performance analysis of simultaneous pose and conventional controllers

The controller performance was compared with the control law used in [90], where the translational and orientational components of the trajectory were treated separately.

Controller Type	**Jacobian Type**	**Control Law**	**Proportional Gain** $\begin{bmatrix} \boldsymbol{\lambda}_{P_p}^{3\times3} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \boldsymbol{\lambda}_{P_o}^{3\times3} \end{bmatrix}$	**Derivative Gain** $\begin{bmatrix} \boldsymbol{\lambda}_{V_p}^{3\times3} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \boldsymbol{\lambda}_{V_o}^{3\times3} \end{bmatrix}$
Simultaneous pose Controller	Screw-based Jacobian	Equation 4.39	$\begin{bmatrix} \mathbf{190} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{190} \end{bmatrix}$	$\begin{bmatrix} \mathbf{6} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{6} \end{bmatrix}$
conventional Controller	KDL Jacobian	Equation 4.44	$\begin{bmatrix} \mathbf{270} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{270} \end{bmatrix}$	$\begin{bmatrix} \mathbf{7.5} & \mathbf{0}^{3\times3} \\ \mathbf{0}^{3\times3} & \mathbf{7.5} \end{bmatrix}$

TABLE 4.1 Comparison of simultaneous pose and conventional controllers for resolved-rate acceleration control

The Jacobian of the manipulator for the conventional controller were obtained from the KDL library. This Jacobian maps the joint velocities of the manipulator to the conventional end-effector velocity and not spatial velocity (see discussion in appendix A.6). Additionally the derivative of this Jacobian was also obtained from KDL library.

The quaternion based control law used for comparison, as given in [90]:

$$\boldsymbol{a}_{cmd} = \ddot{\boldsymbol{p}}_d + \boldsymbol{\lambda}_{Vp}\,\dot{\boldsymbol{p}}_e + \boldsymbol{\lambda}_{Pp}\,\boldsymbol{p}_e, \tag{4.44}$$

$$\boldsymbol{\alpha}_{cmd} = \boldsymbol{\omega}_d + \boldsymbol{\lambda}_{Vo}\,\boldsymbol{\omega}_e + \boldsymbol{\lambda}_{Po}\,\boldsymbol{v}_e, \tag{4.45}$$

where, $\boldsymbol{p}_e$ refers to the position error of the end-effector frame, for a given desired position $\boldsymbol{p}_d$. $\boldsymbol{v}_e$ refers to the vector part of the error quaternion computed as $\boldsymbol{q}_e = \boldsymbol{q}_d\boldsymbol{q}_c^*$, from which the orientation error in the base frame can be derived. $\boldsymbol{a}_{cmd}$ and $\boldsymbol{\alpha}_{cmd}$ are used to obtain the combined acceleration command, similar to $\hat{\boldsymbol{a}}_{cmd}$, to be used in an expression similar to (4.7).

Both the controllers were tuned for a stable profile in velocity and to obtain the best tracking performance. The controller gains for the two controllers used in the comparative analysis have been given in Table. 4.1. The control loop frequency for both the controllers was set 200 *Hz*.

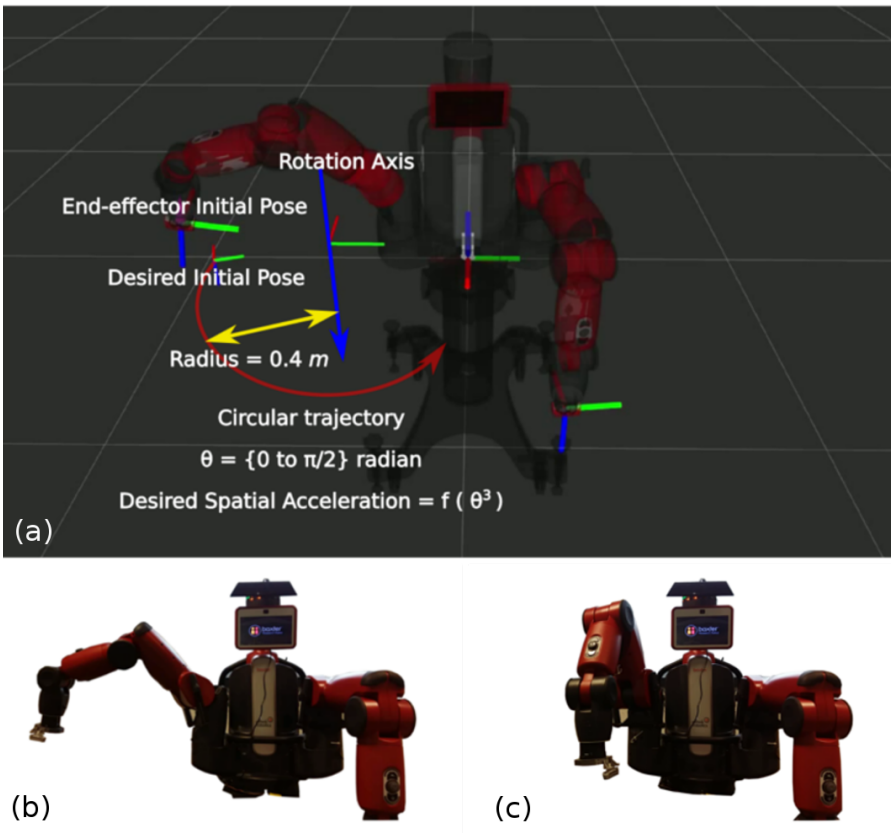


FIG. 4.3 Trajectory generation strategy and Baxter robot performing trajectory tracking task.

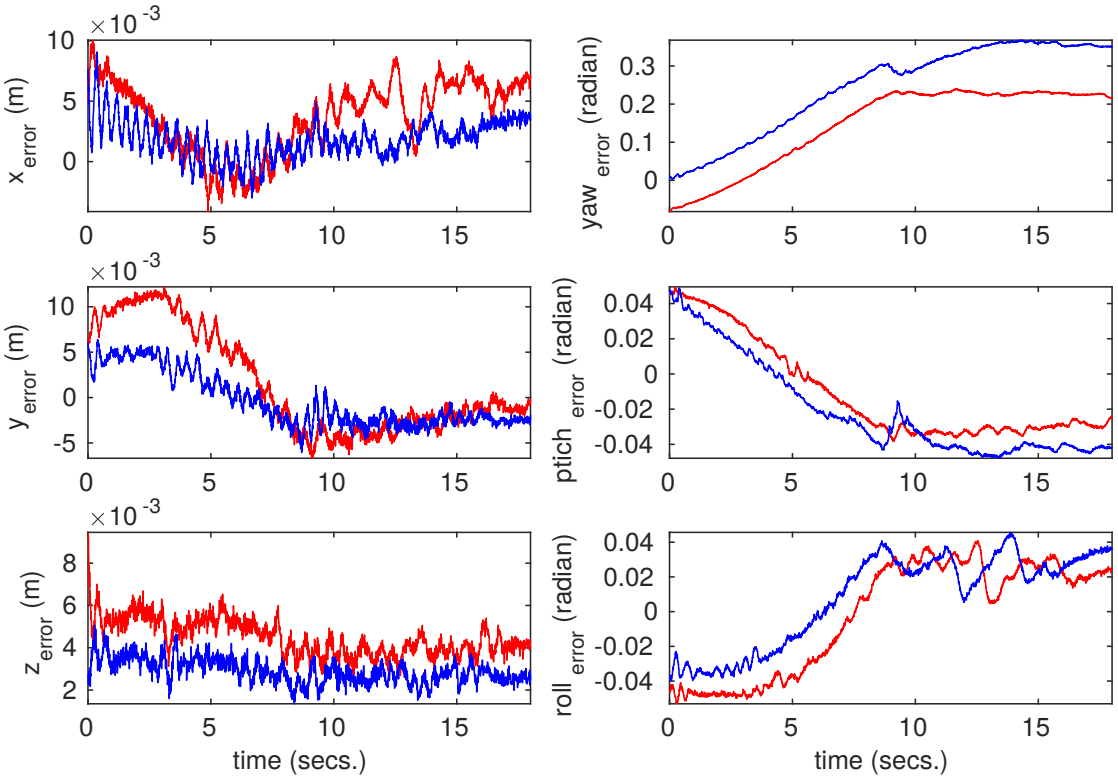


FIG. 4.4 Pose error ($Desired - Current$) plot for the two controllers. (——) refers to the simultaneous pose controller and (——) to conventional controller.

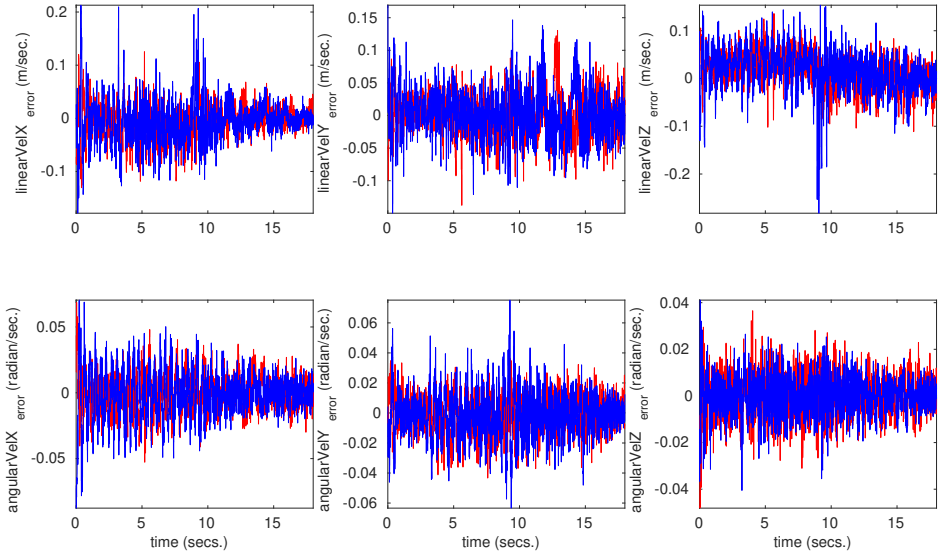


FIG. 4.5 Velocity error ($Desired - Current$) plot for the two controllers. Linear velocity components are given in $m/sec.$, and angular velocity error in $radian/sec.$ (——) refers to the simultaneous pose controller and (——) to conventional controller.

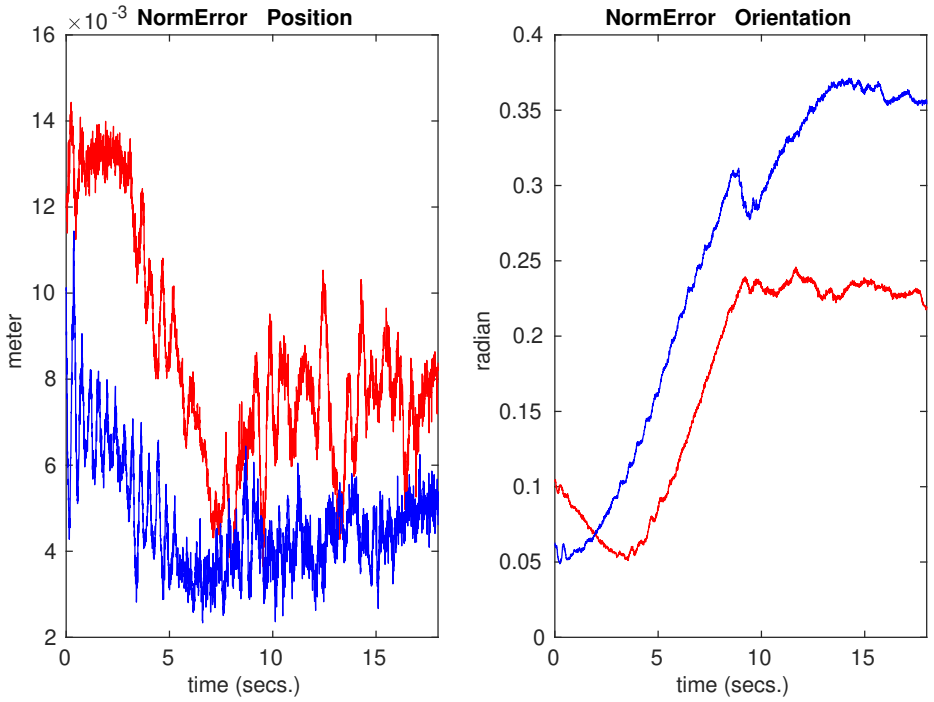


FIG. 4.6 Norm of position and orientation error. (——) refers to the simultaneous pose controller and (——) to conventional controller.

The end-effector was desired to rotate around a pre-selected line defined in the base frame of the robot, while keeping one of the axis attached to the end-effector frame always pointing towards the line, thus requiring both translational and orientation control of the end-effector. The acceleration of the screw motion thus generated was parametrized by $\theta_{traj} = \{0\ to\ \frac{\pi}{2}\}$ and $d_{traj} = 0$, using the trajectory generation strategy discussed in section 4.4. The goal of such trajectory was to start and end with a zero screw velocity. The total duration of the simulation was 18 seconds.

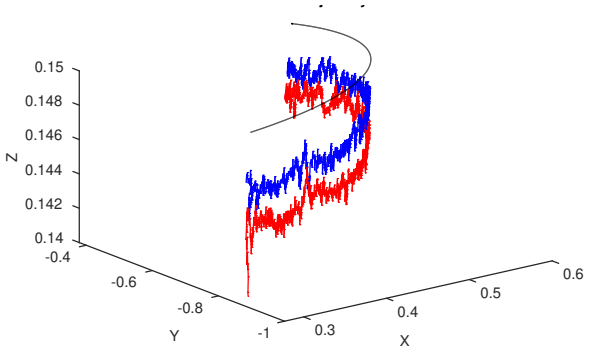


FIG. 4.7 Desired and tracked trajectory of the end-effector position. (——) refers to the simultaneous pose controller and (——) to conventional controller.

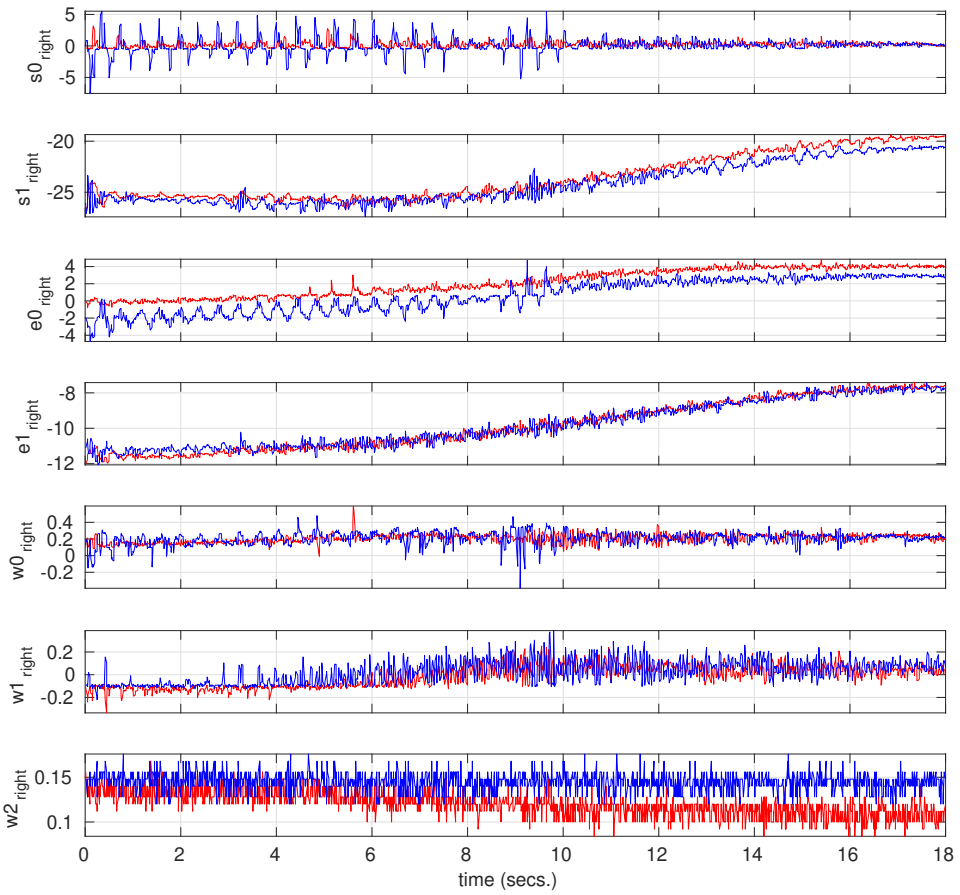


FIG. 4.8 Joint Effort plot for the two controllers. (——) refers to the simultaneous pose controller and (——) to conventional controller.

The $\hat{\boldsymbol{a}}_{cmd}$ obtained in 4.39 calculated using position and velocity errors was used to compute $\ddot{\boldsymbol{q}}_{cmd}$ using 4.6. The $\ddot{\boldsymbol{q}}_{cmd}$ hence obtained was substituted in 4.1 to get the joint driving torques. The joint inertia matrix, Coriolis and centrifugal torques, and the gravity torques were obtained using KDL library [70], and $\dot{\boldsymbol{q}}$ was provided by the Baxter ROS interface.

The robot performing the trajectory tracking task can be seen in Fig. (4.3), and performance of both the controllers for pose and velocity tracking has been given in Fig. (4.4) and (4.5), respectively. While the performance of both the simultaneous pose and conventional controllers is identical for position tracking (see Fig. 4.7), the simultaneous pose controller performed better in terms of orientation tracking, as is evident from Fig. 4.6. The performance is identical in terms of velocity error and the commanded joint

torques for all the joints, given in Fig. 4.8, however higher oscillations can be observed with conventional controller for commanded joint torques.

4.5 Conclusion and Future Works

A new controller for resolved acceleration control of robotic manipulators was proposed for the trajectory tracking of serial robotic manipulator using screw theory and concepts from spatial dynamics. Representation of the motion variables with dual quaternions allowed simultaneous treatment of translational and orientational components of trajectory tracking error. Comparison with a conventional controller revealed better orientation tracking and less oscillations in commanded joint torques, while achieving identical performance for the translational components of the trajectory. However, an appropriate redundancy resolution strategy is needed to utilize the additional degrees of freedom for redundant manipulators, as during the current implementation special attention was taken during the definition of trajectory. Additionally, we would like to extend this method trajectory tracking for dual-arm manipulation.

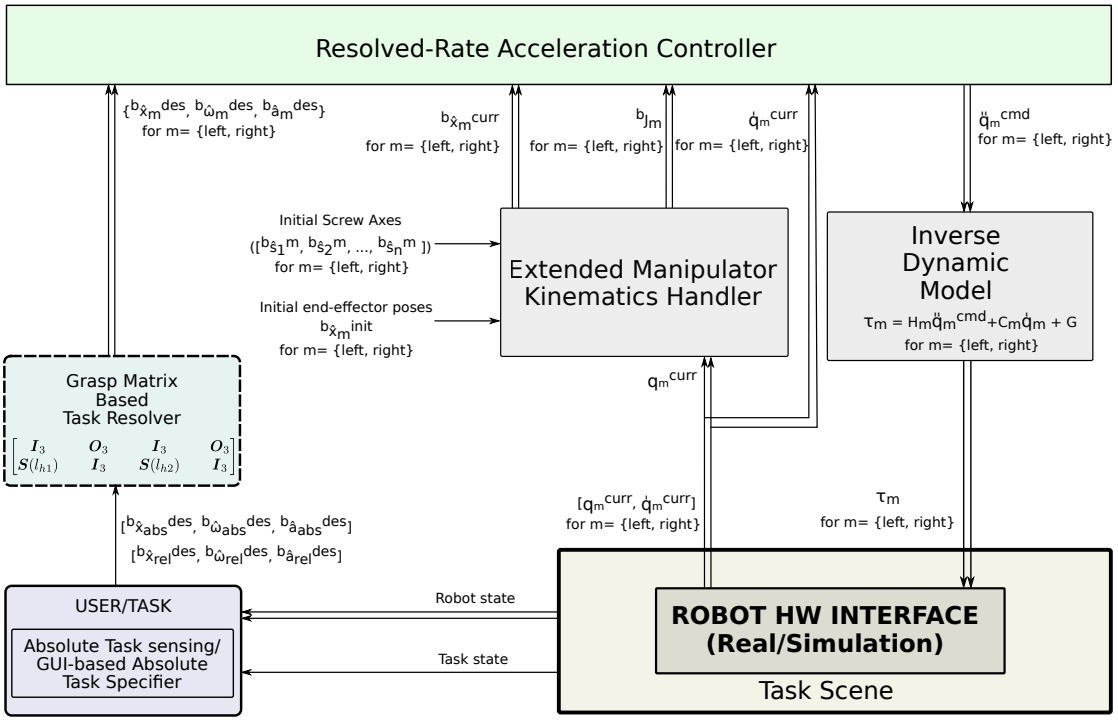


FIG. 4.9 Pose control loop for dual-arm cooperative manipulation of single object.

In the future, we wish to extend the simultaneous pose resolved-rate acceleration controller for dual-arm manipulation similar to the approach taken in [42, 97, 59], where the desired trajectory of individual manipulators were obtained from the CTS trajectories. If the two robotic arms are manipulating a single object and the constraint of rigid grasp is assumed, then the relative motion can be assumed to be negligible, and the corresponding trajectories for the the cooperating manipulators can be derived from the geometry of the grasp [59]. Since, the simultaneous pose approach of trajectory tracking demonstrated better tracking for single arm, it will be interesting to see if this performance carries over into dual-arm cooperative manipulation. In addition to that, a hybrid force/position force control or impedance control strategy is needed to ensure the interaction forces between the object and the robotic arms' end-effectors, and internal force acting on the object under acceptable limits. The dual-arm architecture, proposed as an extension to the architecture presented in Fig. 4.1, incorporating the above mentioned strategy for the inner position control loop, has been given in Fig. 4.9.

Chapter 5

CTS Planner for Shape-Servoing and Contact-based Assembly Tasks

In the previous chapters, we established the advantages of screw theory and DQ based kinematic modeling of CTS of dual-arm robots (chapter 2) and anthropomorphic hands with underactuated fingers (chapter 3), in terms of computational and storage efficiency, as well as in terms of ease of kinematic modelling. The coupled treatment of position and orientation variables in resolved acceleration control of manipulators demonstrated improvement in trajectory tracking performance with regards to tracking accuracy and reduced oscillations in the joint torque commands, as shown in chapter 4. Additionally, the approach to extend the resolved acceleration control for bimanual control was proposed as future work.

In this chapter, we add another dimension to our framework for dual-arm control with the addition of a task planner. When the two end-effectors attached to the arms of a dual-arm of a robot are interacting through a single rigid or articulated object, or with two objects held in each arm, there are situations which require complex interaction between the cooperating robotic arms. These interactions are related both to the kind the task desired, as well as to physical constraints related to the tasks. For example when handling a rigid object, the relative pose between the arms should be kept constant so as to not exert excessive internal force on the object, whereas the absolute force will counter

gravity and perform manipulation and control interaction with the environment. In the case of interaction of the grasped object with environment the type of contact the held rigid object makes with the environment has to be taken into account while generating absolute task-space motion, i.e. the motion for the object to keep the interaction forces arising due to the contact of object with the environment within limits. Similarly, if the dual-arm robot is performing an assembly operation with two mating parts held in each arm, the involved contact state between the objects also limits the motion permissible for the relative task-space.

A cooperative task-planner that is capable of representing these constraints related to external (with the environment) and internal (between cooperating manipulators) interaction can be instrumental to generate cooperative motion plans. In this chapter we propose a task planner that is capable of representing such tasks and related constraints with the application of virtual mechanisms. We generate virtual mechanisms in relative and absolute task space to represent different tasks with constraints, and adjust the VSs attached to each of the arms as the task evolves thus giving the same inputs to the control architecture presented in 2.4.1 (Fig. 2.3). The objective of readjusting the virtual sticks is to extend the static analysis of bimanual rigid body grasping [31] explained in chapter 2.1 for tasks requiring interaction between two objects and also for the handling of objects with articulations, like a pair of scissors. In the current work we assume that the object/objects are at static equilibrium throughout the task, the object model and corresponding pose sensing, as well as manipulator pose control, is precise enough to keep the interaction forces within limits.

The implementation of the task planner is based on screw theory, which allows us to easily define these tasks with minimal parameters, while at the same time is also capable of generating higher-order trajectories. A higher-order trajectory is desired to generate smooth end-effector motion in situations when there are abrupt changes in the pose setpoints. The task-planner will be validated with tasks requiring variation of virtual sticks, by deforming a linear elastic body in different shapes with intuitive approach, and with tasks requiring virtual mechanisms to represent multiple intermediate articulations, like deforming a rope using hinges. Although we have limited the experiments to deformable objects, one of the objective of the these experiments is also to demonstrate how contact states arising in an assembly tasks can be represented with the proposed task-planner.

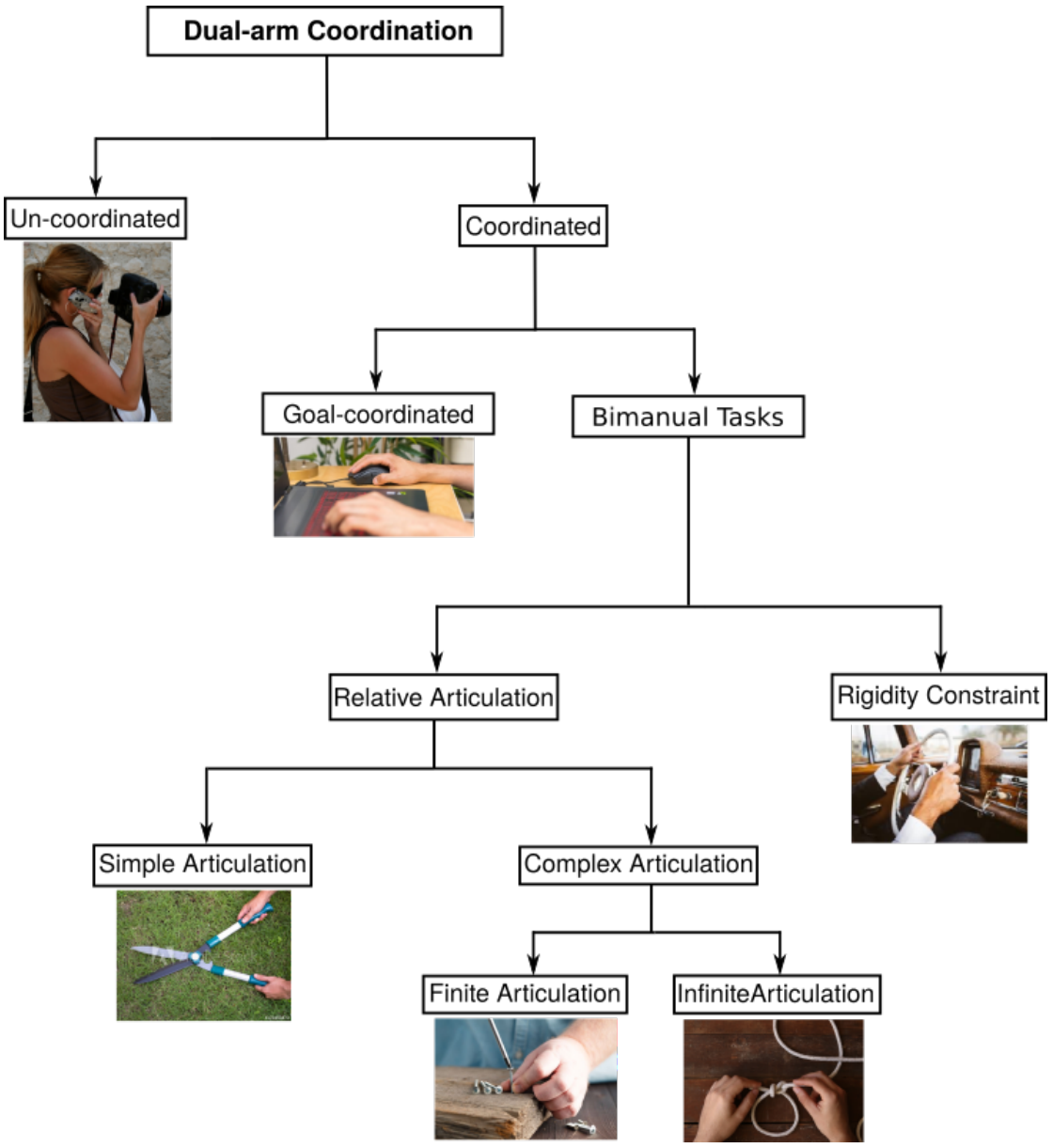


FIG. 5.1 Proposed levels of dual-arm coordination hierarchy.

In the following section (section 5.1) the global motivation for the development of cooperative task-space planner based on VM is presented. The framework for defining and executing CTS based tasks is explained in section 5.2. Afterwards, the experimental setup and corresponding results are discussed and future directions for the proposed method are presented in section 5.3. The concluding remarks related to this work and future work will be presented in section 5.4.

5.1 Motivation

The applicability of VM based task planners can be demonstrated with a new categorization of dual-arm tasks as shown in Fig. 5.1. There are a variety of tasks that can be accomplished by a dual-arm robot, which can roughly be divided into coordinated (goal-coordinated and bimanual tasks) and uncoordinated tasks [26]. Uncoordinated tasks are relatively simpler to handle since they are an extension of single-arm tasks, in spite of added complexity of collision awareness of the other arm's operation. Coordinated tasks, especially bimanual tasks, where coordination between the arms is inherent requirement of the task, for example bimanual lifting or bimanual assembly tasks. Bimanual cooperative structures have been further classified into parallel and serial model of cooperation [35], taking a cue from the study of human bimanual actions [39].

In the proposed new categorization we define new sub-categories of bimanual tasks, different from [35] categorization based on the dependency between arms for bimanual tasks. The new categorization is based on the degree of complexity in the definition of cooperative tasks and extends the concept of CTS for performing assembly tasks, and to control the shape of deformable objects using two arms. Although single object manipulation presents challenges related to internal force control and interaction of the grasped object with the environment, the rigid body constraint assumed during cooperative manipulation design makes it comparatively easier to define the task. In the manipulation of objects with fixed articulation, for instance, scissors, telescope, etc., constraint related to the articulation joint should also be taken into account. On the other hand, contact-based dual-arm assembly tasks like bimanual peg-in-hole and fastening using screws, and shape-control of deformable objects presents additional task modeling challenges due to the higher degree of articulation related to assembly tasks and shape control of deformable objects.

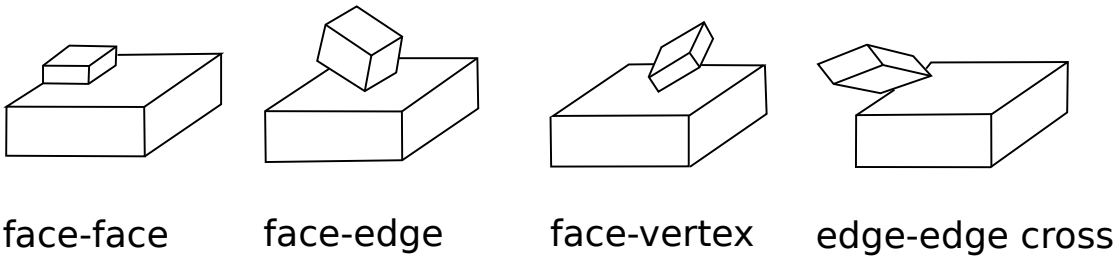


Fig. 5.2 Principal contacts between two polyhedra [11].

Assembly tasks deals with achieving a desired contact configuration between two assembly parts, where these parts go through a controlled sequence of different contact formations. Some these contact states for two polyhedra are given in Fig. 5.2. During assembly sequence the mating parts are required to perform motion within these contact states, in addition to transitioning between different contact states [11] which can also be represented with finite articulation between assembly parts. Robotic manipulation of these assembly parts has to respect the constraints imposed by the contact, and the knowledge of these constraints are important to not only for maintaining the contact, but also for keeping the interaction forces within limits [98].

The constraints related to a different state of contact during assembly tasks can be modeled using VM, with the exclusion of the dynamic aspects such as interaction force control. VM has been used for contact modelling in assembly tasks in [99], and for modeling finger contact model in [100]. Additionally, the same approach can be used to model bimanual skills for the task-frame formalism of assembly tasks [15, 101]. While there are more complicated issues involved for reliable and safe execution of assembly tasks, such as proper sensing of contact state, compliance control, etc. [102], in the current work, we have limited our study to the definition of kinematic constraints involved in dual-arm coordination for assembly tasks. The definition of VM for some common assembly tasks has been given in Fig. 5.3.

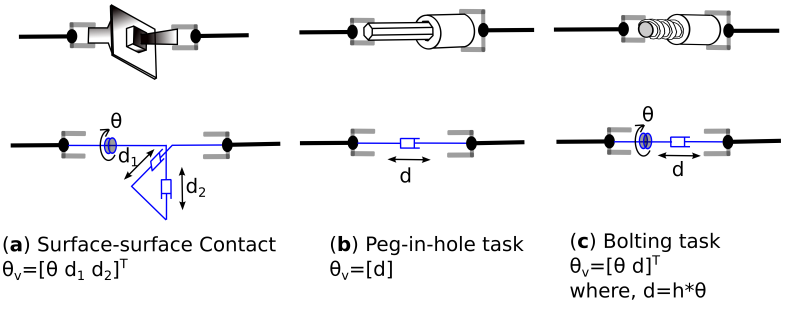


FIG. 5.3 Virtual mechanism equivalents of different assembly tasks: q_v encapsulates the joint displacement of the virtual mechanism. (a) Surface-surface contact task can be represented with a 3-dof planar mechanism, (b) peg-in-hole task with a one-dof prismatic joint, (c) and bolting task with 2-dof screw joint mechanism, where the prismatic joint displacement is equal to the revolute joint displacement (θ) multiplied by the pitch (h) of the bolt.

Deformable object manipulation presents some of the most complex scenarios for defining dual-arm cooperative motion which arises due to the high degree of articulations associated with deformable objects [103]. The degree of articulations of a deformable linear object often outnumbers the degree of control the cooperating arms have over

these articulations, which are only twelve dimensions of CTS, half of them for relative and the other half for absolute task-space control. Nonetheless, the description of certain linear deformable objects using VMs will not only allow us to obtain an approximate model of the shape of objects but also provide a plan for the deformation. In fact, there are some previous works related to the deformation control of isometric linear objects have already employed similar method of modeling the shape of the rope as multi-link system [104, 105], however, without the consideration of cooperative task-space for the shape control of rope.

5.2 Design of Cooperative Task Planner

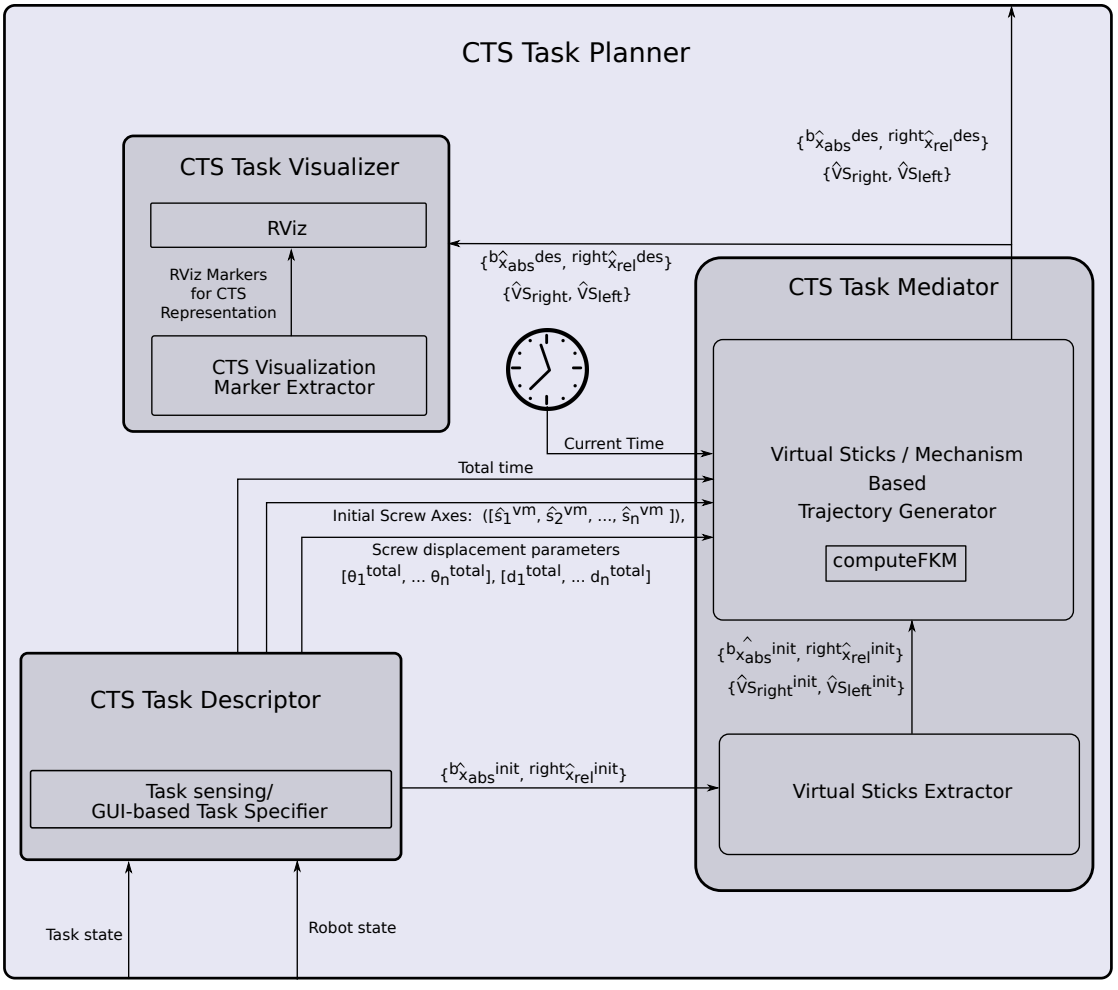


Fig. 5.4 Cooperative task planner architecture.

This section presents architecture for the implementation of an efficient and compact way of parametrized definition of assembly and shape-servoing tasks using virtual links,

and is given in Fig. 5.4. The cooperative task planner consists of a *CTS Task Descriptor* module that is capable of defining VMs based on the sensor input, for example, an image of the initial state of the linear object, and with the desired resolution. The same can be derived for a desired state of the object, and a plan for deformation can thus be obtained by identifying the nodes/joints of deformation and moving the closest joints in the initial VM representation to achieve an approximation of the desired state. The proposed task planner in its current state of implementation does not close the shape control loop with sensorial information, but we demonstrate the effectiveness of this approach by achieving some simple shapes obtained with user-defined tasks in *CTS Task Descriptor* module. The joint screw axes of the virtual mechanism, total desired joint displacements and total time for the operation are the outputs of *CTS Task Descriptor* module and given as an input to *CTS Task Mediator* module.

The task is initialized once the object is held by both hands by recording the current relative configuration. While in the original formulation by [31] the absolute pose was assumed to coincide with the centre of mass for a rigid body held by two manipulators. However, for the shape control of linear elastic body like a foam we can assume negligible mass of the object and can a variable absolute pose. The implication this assumption is that VS sticking from the end-effectors attached to the cooperating robotic arms are coinciding with the new absolute pose. Thus the relative motion between the tips of the virtual sticks can be used to apply a desired internal force for shape control. In the current implementation the new absolute pose can be defined by the user in *CTS Task Descriptor* module, while the relative pose at the beginning of a task is always a unity transformation matrix, which means absolute frame and the frames at the tips of the virtual sticks are all coinciding at the beginning of the task.

A VM is defined in *Virtual Sticks/ Mechanism Based Trajectory Generator* using initial screw axes DQ given in the reference frame and the current relative pose. In the current implementation, the absolute pose in *reference manipulator*'s end-effector frame is connected to the link exactly at the centre of the virtual mechanism. The *Virtual Sticks/ Mechanism Based Trajectory Generator* uses the *computeFKM* block to compute the evolution of the virtual mechanism in the *reference manipulator*'s end-effector frame. The desired absolute pose is then computed from the current state of virtual mechanism defining the desired relative configuration using the method described in Algorithm 2. Finally new virtual sticks are computed from the desired relative and absolute frame

hence computed using Algorithm 3. The complete method used in *CTS Task Mediator* to generate the CTS poses and virtual sticks is given in Algorithm 1.

Algorithm 1 VM based CTS task planner

procedure VM BASED CTS TASK PLANNING

Initial Conditions:

- ${}^{b}\hat{\boldsymbol{x}}_{r}^{init} \leftarrow$ right end-effector pose in *base* frame
- ${}^{b}\hat{\boldsymbol{x}}_{t}^{init} \leftarrow$ left end-effector pose in *base* frame

Input:

- $\boldsymbol{l}^{vm} \in \{\boldsymbol{l}_x, \boldsymbol{l}_y, \boldsymbol{l}_z\}$: initial VM screw axes direction in *reference manipulator*'s end-effector (ref) frame
- n_{VM}: number of joints desired for the VM
- $\underline{\hat{\boldsymbol{q}}}_{total}^{vm} = [q_{total}^{vm}, ...] \in \mathrm{D}^{n_{VM}}$: total joint motion (dual angle consisting of translational and rotational motion) desired for VM joints
- $\hat{\boldsymbol{s}}^{abs} \in \{\hat{\boldsymbol{s}}_x, \hat{\boldsymbol{s}}_y, \hat{\boldsymbol{s}}_z\}$: screw axes direction in absolute frame for desired trajectory
- $[\theta_{abs}^{total}, d_{abs}^{total}]$: total screw displacement desired for absolute frame given in the initial absolute frame
- $totalTime$: total time allotted for CTS task

Output:

- ${}^{b}\hat{\boldsymbol{x}}_{abs}^{des}$: desired absolute pose in *base* frame
- ${}^{r}\hat{\boldsymbol{x}}_{rVS}^{des}$: VS_{ref} in *reference manipulator*'s end-effector (ref) frame
- ${}^{t}\hat{\boldsymbol{x}}_{tVS}^{des}$: VS_{tool} in *tool* end-effector frame

${}^{r}\hat{\boldsymbol{x}}_{vm}^{init} = {}^{r}\hat{\boldsymbol{x}}_{t}^{init} \leftarrow multiplyDQ(getDQClassicConjugate({}^{b}\hat{\boldsymbol{x}}_{r}^{init}), {}^{b}\hat{\boldsymbol{x}}_{t}^{init})$

${}^{r}\hat{\boldsymbol{x}}_{abs}^{init} \leftarrow computeAbsPoseInRefFrame({}^{r}\hat{\boldsymbol{x}}_{t}^{init})$ (Algorithm 2)

initialize VM

- divide total length of relative position into $n_{VM} + 1$ parts
- $\begin{bmatrix} \hat{\boldsymbol{s}}_1^{vm} & \hat{\boldsymbol{s}}_2^{vm} & \dots & \hat{\boldsymbol{s}}_{n_{VM}}^{vm} \end{bmatrix} \leftarrow$ initial screw axes

$currentTime \leftarrow \emptyset$

while $currentTime < totalTime$ **do**

$timeFraction \leftarrow currentTime/totalTime$

$\underline{\hat{\boldsymbol{q}}}_{now}^{vm} \leftarrow \underline{\hat{\boldsymbol{q}}}_{total}^{vm} * timeFraction$

${}^{r}\hat{\boldsymbol{x}}_{vm}^{now} \leftarrow FKM(\underline{\hat{\boldsymbol{q}}}_{total}^{vm}, {}^{r}\hat{\boldsymbol{x}}_{vm}^{init}, n_{VM})$

${}^{r}\hat{\boldsymbol{x}}_{abs}^{now} \leftarrow FKM(\underline{\hat{\boldsymbol{q}}}_{total}^{vm}, {}^{r}\hat{\boldsymbol{x}}_{abs}^{init}, n_{VM}/2 + 1)$

$[\theta_{abs}^{now}, d_{abs}^{now}] \leftarrow [\theta_{abs}^{total}, d_{abs}^{total}] * timeFraction$

${}^{b}\hat{\boldsymbol{x}}_{abs}^{des} \leftarrow multiplyDQ({}^{b}\hat{\boldsymbol{x}}_{abs}^{init}, screwToDQ([\theta_{abs}^{total}, d_{abs}^{total}, \hat{\boldsymbol{s}}^{abs}]))$

$[{}^{r}\hat{\boldsymbol{x}}_{rVS}^{des}, {}^{t}\hat{\boldsymbol{x}}_{tVS}^{des}] \leftarrow computeVSFromAbsoluteRelative({}^{b}\hat{\boldsymbol{x}}_{abs}^{des}, {}^{r}\hat{\boldsymbol{x}}_{vm}^{now})$ (Algorithm 3)

end while

end procedure

Algorithm 2 computeAbsPoseInRefFrame

procedure GET ABSOLUTE POSE FROM RELATIVE POSE

Input:

- ${}^{r}\hat{\boldsymbol{x}}_t$: *tool* frame pose in ref frame

Output:

- ${}^{r}\hat{\boldsymbol{x}}_{abs}$: absolute pose in ref frame

$[\theta, d, \boldsymbol{l}, \boldsymbol{m}] \leftarrow DQToScrew({}^{r}\hat{\boldsymbol{x}}_t)$

$\boldsymbol{p} \leftarrow getPositionFromDQ({}^{r}\hat{\boldsymbol{x}}_t)$

${}^{r}\boldsymbol{orientation}_{abs} \leftarrow angleAxisToQuaternion(\theta/2, \boldsymbol{l})$

${}^{r}\boldsymbol{translation}_{abs} \leftarrow getTranslationQuaternion({}^{r}\boldsymbol{orientation}_{abs}, \boldsymbol{p/2})$

${}^{r}\hat{\boldsymbol{x}}_{abs} \leftarrow \begin{bmatrix} {}^{r}\boldsymbol{orientation}_{abs} & {}^{r}\boldsymbol{translation}_{abs} \end{bmatrix}$

end procedure

Algorithm 3 computeVSFromAbsoluteRelative

procedure GET VSs FROM ABSOLUTE AND RELATIVE POSES

Input:

- ${}^{b}\hat{\boldsymbol{x}}_{abs}$: absolute pose in ref frame
- ${}^{r}\hat{\boldsymbol{x}}_t$: *tool* pose in ref frame

Output:

- ${}^{r}\hat{\boldsymbol{x}}_{rVS}$: VS_{ref} in ref end-effector frame
- ${}^{t}\hat{\boldsymbol{x}}_{tVS}$: VS_{tool} in *tool* end-effector frame

$[{}^{b}\hat{\boldsymbol{x}}_r, {}^{b}\hat{\boldsymbol{x}}_t] \leftarrow getRefToolFromAbsRelative({}^{b}\hat{\boldsymbol{x}}_{abs}, {}^{r}\hat{\boldsymbol{x}}_t)$ (Algorithm 4)

${}^{r}\hat{\boldsymbol{x}}_{rVS} \leftarrow multiplyDQ(getDQClassicConjugate({}^{b}\hat{\boldsymbol{x}}_r), {}^{b}\hat{\boldsymbol{x}}_{abs})$

${}^{t}\hat{\boldsymbol{x}}_{tVS} \leftarrow multiplyDQ(getDQClassicConjugate({}^{b}\hat{\boldsymbol{x}}_t), {}^{b}\hat{\boldsymbol{x}}_{abs})$

end procedure

Algorithm 4 getRefToolFromAbsRelative

procedure GET END-EFFECTORS POSES FROM ABSOLUTE AND RELATIVE POSES

Input:

- ${}^{b}\hat{\boldsymbol{x}}_{abs}$: absolute pose in *base* frame
- ${}^{r}\hat{\boldsymbol{x}}_t$: *tool* end-effector pose in ref end-effector frame

Output:

- ${}^{b}\hat{\boldsymbol{x}}_r$: ref pose in *base* frame
- ${}^{b}\hat{\boldsymbol{x}}_t$: *tool* pose in *base* frame

${}^{r}\hat{\boldsymbol{x}}_{abs} \leftarrow computeAbsPoseInRefFrame({}^{r}\hat{\boldsymbol{x}}_t)$

${}^{b}\hat{\boldsymbol{x}}_r \leftarrow multiplyDQ({}^{b}\hat{\boldsymbol{x}}_{abs}, getDQClassicConjugate({}^{r}\hat{\boldsymbol{x}}_{abs}))$

${}^{b}\hat{\boldsymbol{x}}_t \leftarrow multiplyDQ({}^{b}\hat{\boldsymbol{x}}_r, {}^{r}\hat{\boldsymbol{x}}_t)$

end procedure

5.3 Experimentation Validation and Result

The proposed framework for CTS task planning and control was validated on a Baxter dual-arm platform for the manipulation of a deformable foam and a non-elastic rope. The variation of virtual sticks, obtained by changing the initial absolute pose in the reference frame, was validated first with an elastic linear foam and by analysing the deformed shape. It is hypothesized that static analysis valid for rigid body control, where the relative pose is kept constant to avoid excessive strain on the object due to internal force, can be used to achieve the desired deformation for quasi-static motion between the robotic arms by providing a relative motion.

In the first set of experiments we will validate the variation of virtual sticks with deformation tasks with linear deformable foam, so that the effect of changing the absolute pose is visible. Afterwards, the VM based trajectory generation method was validated with constraint based deformation of a rope. The controller gains for the simultaneous pose controller used in this approach were the same chosen in chapter 2 and has been presented in Table 2.5. The total duration of the experiment was 8 seconds and the controller was operating at ≈ 100 *Hz*. The following sections describe the experiments designed for the validation of the VS (section 5.3.1) and VM based task specification (section 5.3.2) in more detail.

5.3.1 Variation of Virtual sticks

In the first set of experiments shown in Fig. 5.5 and 5.6, the linear foam is at static equilibrium when it is held between both the two cooperating arms. For the *BC* (Bend Centre) experiment absolute pose was fixed exactly at the centre of the foam, while for the *BOC* (Bend Off Centre) experiment absolute pose was fixed exactly at the centre of the foam. A relative rotational motion of 80 degrees was generated at the tips of the VSs in the frame of the tip of the *reference manipulator* while keeping the absolute pose constant with respect to the base frame. The screw-based rotational trajectory planning was achieved using the approach given in section 2.4.3.2.

The performance of the controller has been shown in terms of DQ representation on the desired and actual pose for both experiments in Fig. 5.9 and 5.10. Whereas, in Fig. 5.8 and 5.7 the controller performance has been represented in terms of norm of the position,

i.e. the modulus of distance ($\sqrt{x_e^2 + y_e^2 + z_e^2}$) error, and in terms of orientational distance ($\|\theta_e\|$) error related to angle-axis approach of orientation error representation.

FIG. 5.5 Bend at the centre: The absolute pose was fixed at the centre of the foam. The initial desired state of VSs is shown in yellow, current desired state in blue, and final desired configuration in magenta.

FIG. 5.6 Bend off centre: The absolute pose was shifted 10 cm to the left from the centre of foam.

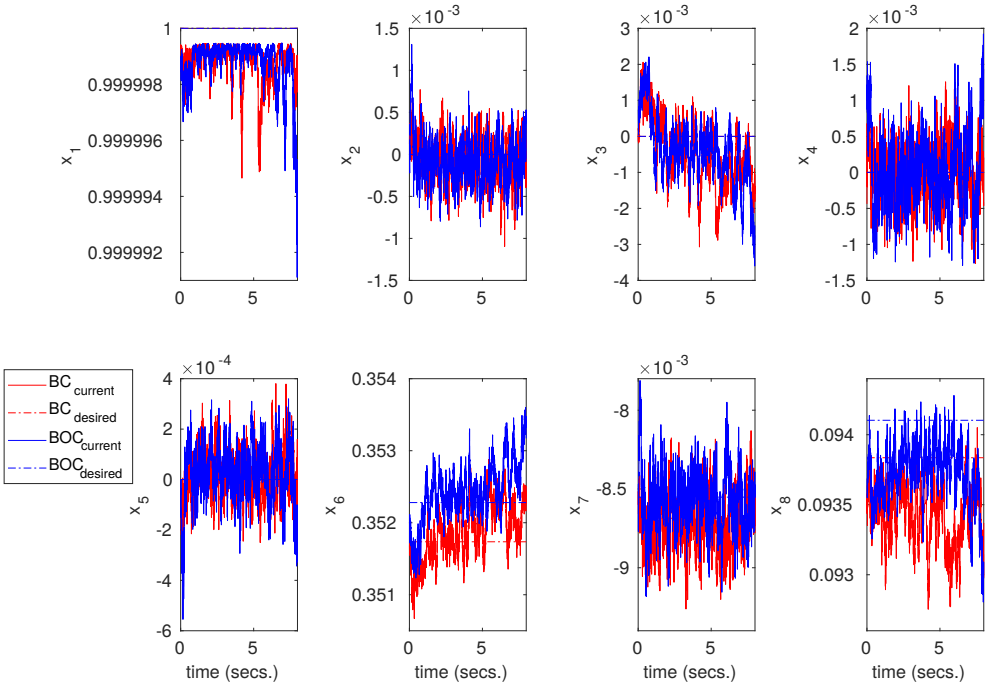


FIG. 5.7 Pose tracking performance for absolute task space in DQ $(x_1, x_2, ..., x_8)$ terms for two relative rotation tasks with variation of absolute pose, which are BC(Bend Centre) and BOC(Bend Off Centre).

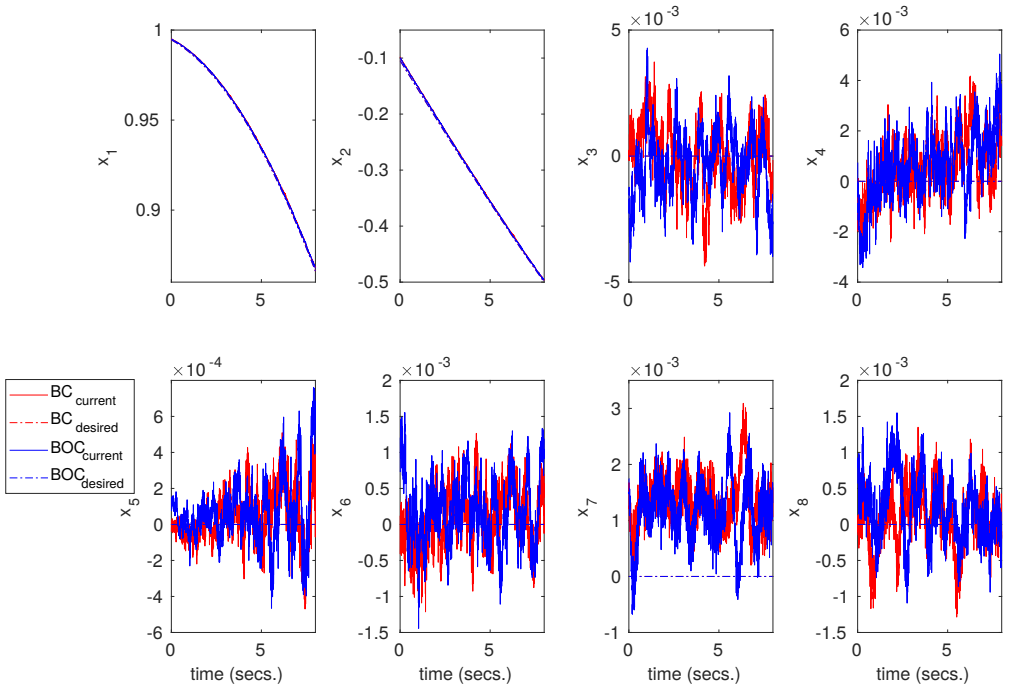


FIG. 5.8 Pose tracking performance for relative task space in DQ $(x_1, x_2, ..., x_8)$ terms for two relative rotation tasks with variation of absolute pose, which are BC(Bend Centre) and BOC(Bend Off Centre).

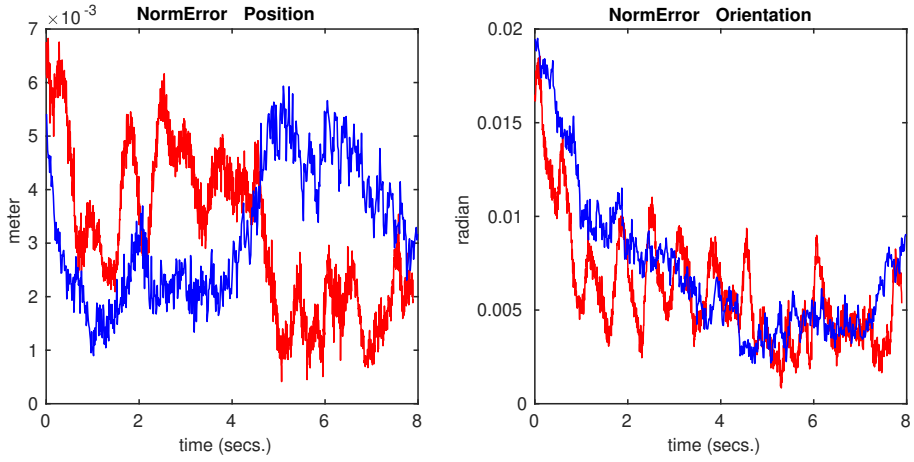


FIG. 5.9 Evolution of error in terms of the norm of position and orientation error in absolute task space. (——) refers to BC experiment and (——) to BOC experiment.

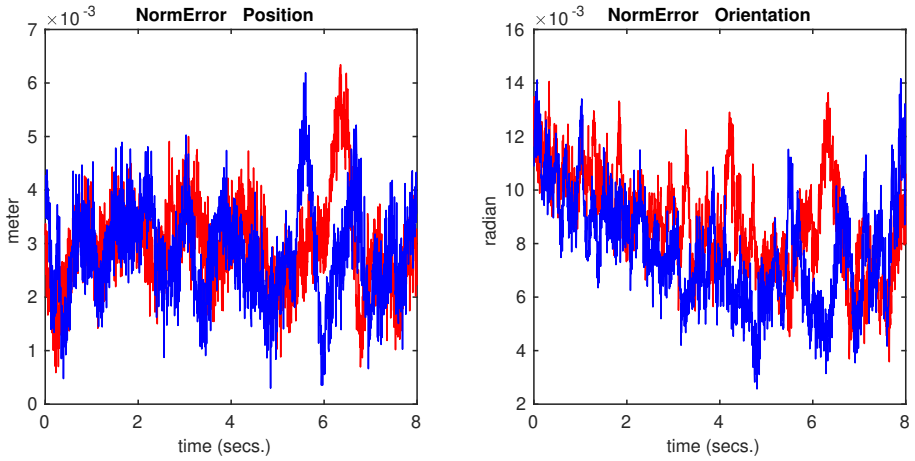
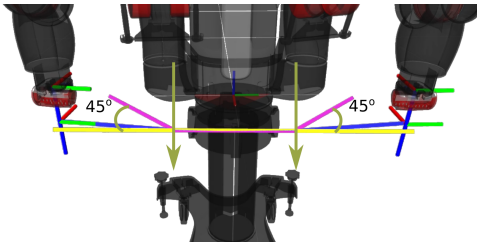


FIG. 5.10 Evolution of error in terms of the norm of position and orientation error in relative task space. (——) refers to BC experiment and (——) to BOC experiment.

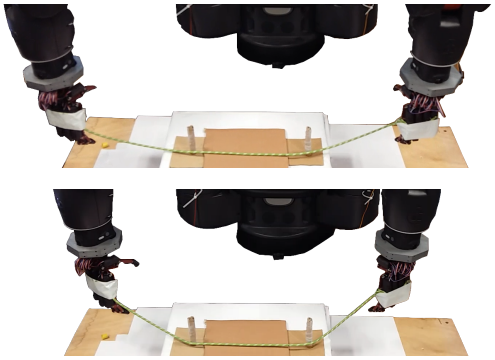
For both the experiments high pose tracking accuracy was achieved both in relative and absolute task space and the maximum pose error was around 7 mm, while a maximum rotation error of 0.02 radian was recorded. The deformable foam was at equilibrium throughout the experiment and the deformation was very controlled. This confirms that our static equilibrium assumption can be beneficial for controlled deformation of such objects.

5.3.2 Relative task definition using VMs

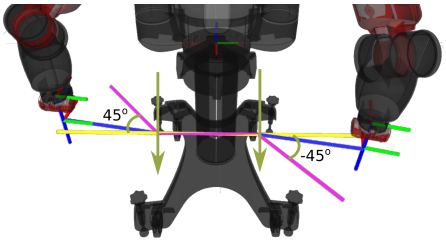
The VM based definition of task in relative task space explained in 1 was validated with experiments related to constrained deformation of a rope. Two kinds of experiments were designed to validate the VM based trajectory generation module. The experiment starts with a rope held in the undeformed elongated state by both arms of the manipulator. In the first experiment, the rope has to be wound around two vertical lines in the U-shape, while keeping the absolute pose static. The distances between the hinges were a third of the total length of the rope. The motion of the arms was generated by using two rotational axes in the same direction. An equal rotation of 45 degrees is given to both the joints, while the initial absolute pose was maintained. For the next experiment, one of the joints was given an equal and opposite rotation, while maintaining the initial absolute pose by redefining the VSs for each iteration of the control loop. The robot performing the task, as well as the corresponding VM is given in Table 5.1.



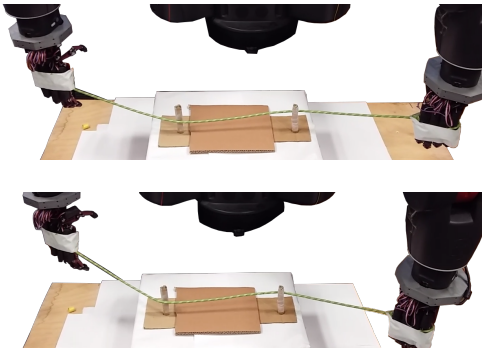
(a) Initial and desired state of U-shape VM.



(b) Robot performing U-shaped deformation.



(c) Initial and desired state of S-shape VM.



(d) Robot performing S-shaped deformation.

TABLE 5.1 U and S shaped deformation trajectory details and real robot performing the tasks.

The performance of the controller has been shown in terms of DQ representation on the desired and actual pose for both experiments in Fig. 5.11 and 5.12. Whereas, in Fig. 5.13 and 5.14 the controller performance has been represented in terms of the modulus of distance error, and in terms of orientational distance error related to angle-axis approach of orientation error representation.

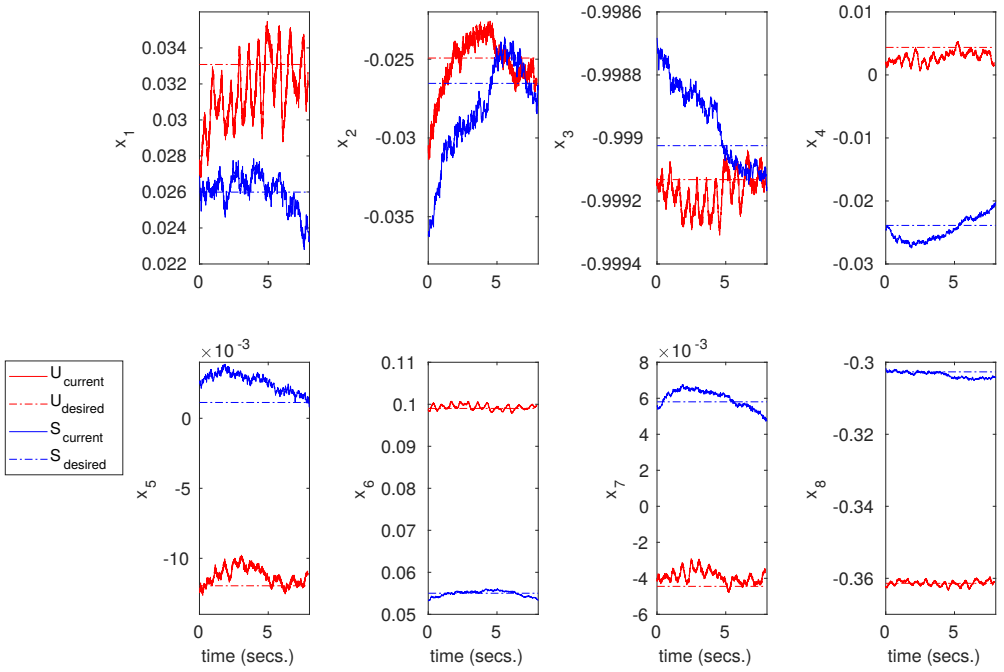


FIG. 5.11 Pose tracking performance for absolute task space in DQ $(x_1, x_2, ..., x_8)$ terms for the rope deformation task to achieve U and S shape.

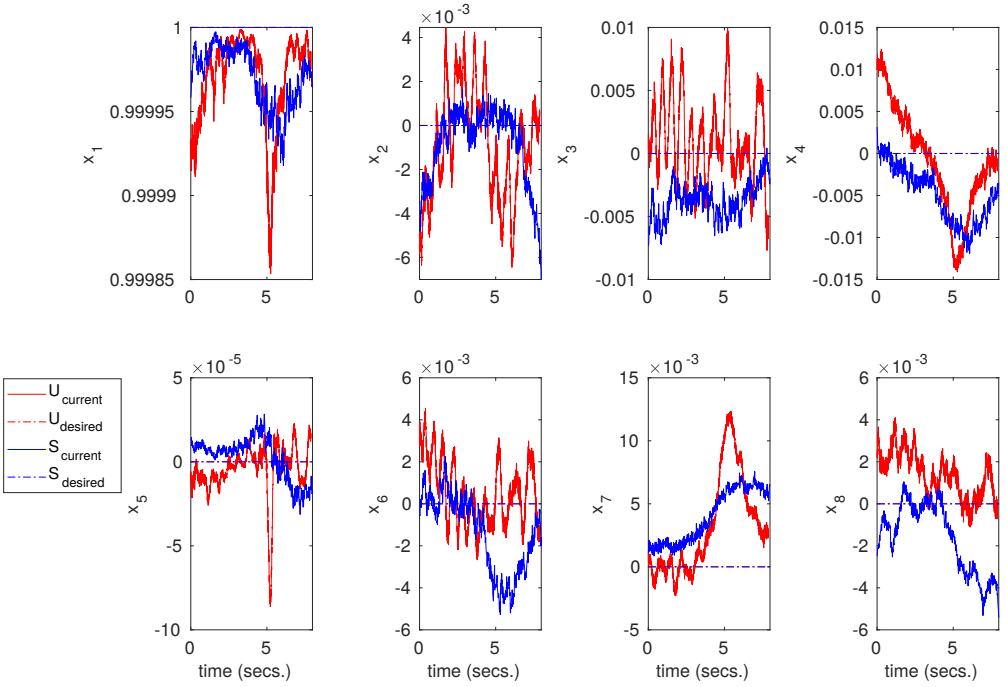


FIG. 5.12 Pose tracking performance for relative task space in DQ $(x_1, x_2, ..., x_8)$ terms for the rope deformation task to achieve U and S shape.

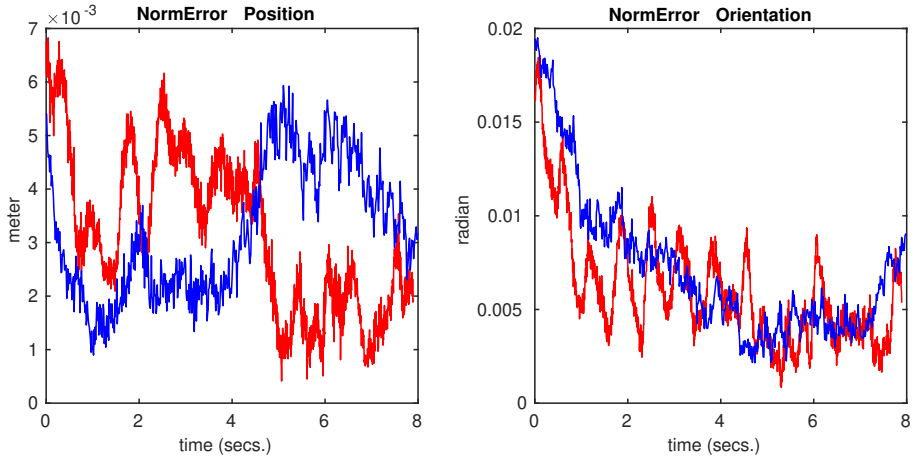


FIG. 5.13 Evolution of error in terms of the norm of position and orientation error in absolute task space for the rope deformation task to achieve U (——) and S (——) shape.

While the trajectory profile for both the experiments is stable, significant error (≈ 2.5 cm) was recorded for the U-shape task in relative task space. This significant error can be attributed to the measurement errors related to the dimension of the rope and the distance between the hinges, and to the fact that the rope is rigidly attached to the robot and the joints of Baxter robot are compliant. Note that the desired absolute as well relative pose represented as DQ in Fig. 5.13 and 5.14 is constant throughout the task. This is because instead of varying the desired relative pose, we are computing the corresponding virtual sticks, as explained in Algorithm 1.

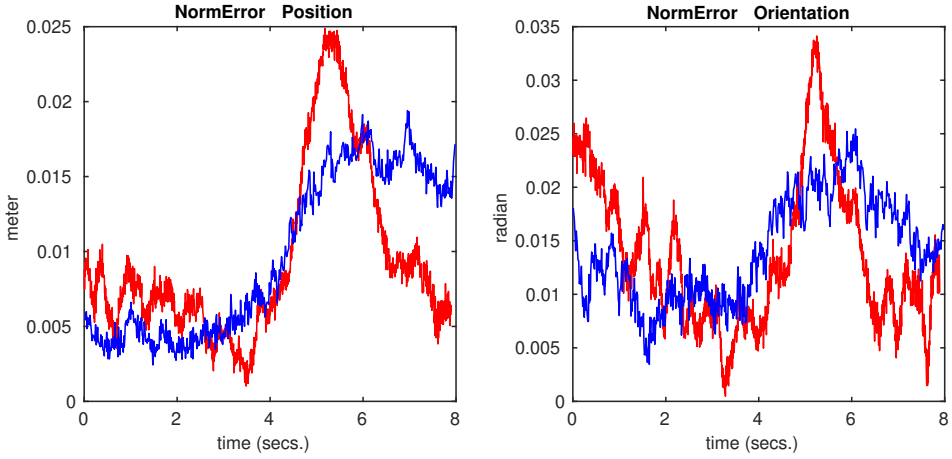


FIG. 5.14 Evolution of error in terms of the norm of position and orientation error in relative task space for the rope deformation task to achieve U (——) and S (——) shape.

5.4 Conclusion

This chapter presented the application screw theory based kinematics with DQ representation for constraint modeling and trajectory generation using virtual mechanisms. The static analysis for rigid body handling using dual-arm robots was extended for the definition of a deformation task for linear semi-rigid objects with elastic properties. The implementation of the task planner consisted of planning as well as visualization modules, which was found to be useful for visualizing the task before executing it with the robot.

The task modeling approach using screw theory provided a basis for compact parametrization of a wide range of tasks. While conventional approaches require a frame for every link of the virtual mechanism along with the axis of joint motion, the screw-based approach requires only the screw axes of the joints of the VM. Forward kinematics of the virtual mechanism is also more efficient than HTM based method since DQ multiplication is computationally less expensive, as shown in section 2.2.2.2. Moreover, the comparative evaluation of DQ pose controller using screw parameters of pose error against the conventional control of position and orientation error, proved that the simultaneous pose control was found to be better at pose tracking in relative task space.

Chapter 6

Conclusion

In this thesis we developed a complete framework for dual-arm coordination and task definition. We used screw-based kinematics for the modelling of manipulators and their coordination, and employed VM for task definition. A simultaneous pose control approach was taken for the CTS control of dual-arm that operated on the screw error between the desired and current poses, and was compared with the conventional approach of minimizing position and orientation errors separately. Preliminary results of the implementation of these two controllers, simultaneous pose controller and conventional pose controller, on a dual-arm robot, demonstrated that simultaneous pose controllers are better at tracking CTS pose set-points, with clear improvement for orientation control.

To summarize the contributions of this thesis:

- *CTS modelling and control*: We combined screw-based kinematics with DQ representation for the modelling and control of CTS poses for dual-arm coordination in chapter 2. The compactness and computational efficiency of dual quaternion representation of pose (Table 2.1), and generalized velocity were utilized for the forward kinematics and control law design for dual-arm cooperation. Usage of screw-based kinematics allowed natural consideration of *wrench transformation matrix* and thus more stable and accurate tracking in *relative* task space, even for tasks requiring fast relative motions (Table 2.2).

- *Anthropomorphic hand modelling*: We extended the relative Jacobian modelling for the cooperation modelling of the fingers of an anthropomorphic robotic hand in chapter 3. The coupling of joints in the underactuated fingers of the robotic hand were represented with a coupled finger Jacobian. The coupled Jacobian of the robotic finger was used for inverse kinematic control, while allowing easy integration with a robotic arm. Additionally, the relative task-space between the fingertips of index finger and thumb was modelled using screw theory, similar to dual-arm robots. The inverse kinematic control to achieve a desired relative pose between the thumb tip and the index finger tip was validated on a robotic hand AR10.

- *Resolved-rate acceleration control*: The idea of simultaneous pose treatment of position and orientation variables was taken further in chapter 4 with the design of a second-order trajectory tracker using DQs. The trajectory controller hence designed was capable of tracking pose, velocity and acceleration setpoints for the end-effector using inverse dynamic model of the robot. The simultaneous pose resolved rate acceleration controller was implemented for the control of one of the arms of Baxter dual-arm robot, and was found to be capable of tighter trajectory control, specially for error terms related to orientation, compared to the conventional controller ([90]) that treated the position and orientation setpoints separately and ignored the inherent effect of rotation on translational motion. Additionally, it also led to lower oscillations in the joint torque command.

- *CTS Task Planner*: Finally, a complete framework for the coordination of bi-arm robotic systems was proposed with the addition of a cooperative task planner in chapter 5. The simplicity of screw theory was exploited additionally for parametrized generation of generalized second order trajectories for tasks requiring simplified motion, like translation, rotation and screw motion around an arbitrary 6D screw-axis given in a known reference frame. The trajectory generation method was extended to represent the constraints related to tasks involving contact between objects using the concept of VM. Again, screw theory was found to be a suitable choice for constructing such mechanisms on the fly, and computation of desired set points for relative and absolute task-space for dual arm coordination. Additionally, the VMs were also used to parametrize different kinds of operations of

semi-deformable objects with evident elastic properties like foam and with highly articulated objects like rope where we constrain the rope to achieve desired shapes.

6.1 Future directions

The current implementation of the framework for dual-arm manipulation is capable of:

- Generating simple trajectories like rotation, translation and screw motion around a 6D screw axes. Additionally, trajectories can also be generated using VM for single arm, as well as for relative and absolute task space. The parametrized trajectory generator is also capable of generating second-order trajectory with null velocity at the starting and end of the trajectory, which can easily be extended to higher order if is desired to include the boundary conditions for acceleration.
- Tracking of pose trajectory hence generated for single arm, and for CTS control of dual-arm robots using both UDQ based coupled controller, as well as for KDL based decoupled controller.
- Trajectory tracking of a single manipulator, when the end-effector has to follow a desired pose, velocity and acceleration set-points, for both position and orientation.

The significance of this framework can be further enhanced by:

- *Combining hand kinematics with the robotic arm*, so that the complete robotic hand-arm system can be made to achieved an initial configuration with an object to be grasped. Then the relative task-space modelling for fingertips can be used to obtain task-specific Jacobian [13] for tasks like grasping and un-grasping by increasing or decreasing the distance between finger tips. We have proposed CTS Jacobians for cooperation control between robotic fingers of an anthropomorphic hand attached to a robotic arm in the conclusion section (3.4) of chapter 3.
- *Extension of coupled resolve-rate acceleration control of manipulators for dual-arm manipulation*: In the future we would like to extend the coupled resolved-rate acceleration controller for dual-arm manipulation similar to the approach taken in [42, 97, 59], where the desired trajectory of individual manipulators were obtained

from the CTS trajectories. We have proposed an architecture for this extension of resolved-rate acceleration control for tasks where the two robotic arms are manipulating a single object and the constraint of rigid grasp is assumed in Fig. 4.9 in chapter 4.

- *Identifying and formalizing dual-arm robotics skills for deformation control and assembly tasks.* Since today's industries rely on constantly changing products to spur the market demands, a flexible framework for robotic application in the production line is desired. One of most common approaches to address this challenge of easy reconfigurability of robotic applications has been to implement a skill-based system [15]. While there are multitude of aspects involved in the design of such a system that are outside the scope of this thesis, such as force control, uncertainty handling, etc., other than the additional components of task state identification and state transition, a parametrized description of motion during these tasks constitute the core of such systems. As has been demonstrated in the current chapter, screw-based definition of tasks provides a compact and general parametrization of motion. Some common deformation and assembly tasks can be encoded as dual-arm skill primitives using the tools such as task definition and dual-arm motion control, provided in the current framework of cooperative task modelling and control.

Appendix A

Dual Quaternion and Spatial Dynamics Basics

A.1 Quaternions

Quaternions can be regarded as pairs of a scalar and a vector $(s, \boldsymbol{v})$, where $s \in \mathbb{R}$ and $\boldsymbol{v} \in \mathbb{R}^3$, or as a 4-tuple:

$$\boldsymbol{q} = s_q + \boldsymbol{v}_q = s_q + v_{q_x}\boldsymbol{i} + v_{q_y}\boldsymbol{j} + v_{q_z}\boldsymbol{k} \tag{A.1}$$

$$= (s_q,\, v_{q_x},\, v_{q_y},\, v_{q_z}). \tag{A.2}$$

$\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ are the unit vectors along the x, y, z axis, respectively, for which following relations was defined by William Rowan Hamilton in 1843 [106]:

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1 \tag{A.3}$$

A pure quaternion has null scalar component, i.e. $s = 0$.

A.1.1 Quaternion operations

- *Addition*: Addition of two quaternions $\boldsymbol{p} = s_p + \boldsymbol{v}_p = s_p + v_{p_x}\boldsymbol{i} + v_{p_y}\boldsymbol{j} + v_{p_z}\boldsymbol{k}$ and $\boldsymbol{q} = s_q + \boldsymbol{v}_q = s_q + v_{q_x}\boldsymbol{i} + v_{q_y}\boldsymbol{j} + v_{q_z}\boldsymbol{k}$ is defined as:

$$\boldsymbol{p} + \boldsymbol{q} = (s_p + s_q) + (v_{p_x} + v_{q_x})\boldsymbol{i} + (v_{p_y} + v_{q_y})\boldsymbol{j} + (v_{p_z} + v_{q_z})\boldsymbol{k} \tag{A.4}$$

- *Multiplication*: Multiplication of two quaternions $\boldsymbol{p} = s_p + \boldsymbol{v}_p$ and $\boldsymbol{q} = s_q + \boldsymbol{v}_q$ is performed as:

$$\boldsymbol{pq} = (s_p s_q - \boldsymbol{v}_p \cdot \boldsymbol{v}_q) + (s_p \boldsymbol{v}_q + s_q \boldsymbol{v}_p + \boldsymbol{v}_p \times \boldsymbol{v}_q) \tag{A.5}$$

  The multiplication between quaternions is associative but not commutative.

- *Conjugate*: The conjugate of a quaternion $\boldsymbol{q} = s_q + \boldsymbol{v}_q$ is given as:

$$\boldsymbol{q}^* = s_q - \boldsymbol{v}_q \tag{A.6}$$

  The product of a quaternion and its conjugate will result in the square sum of its elements, i.e.:

$$\boldsymbol{q}\boldsymbol{q}^* = \boldsymbol{q}^*\boldsymbol{q} = {s_q}^2 + v_{q_x}^2 + v_{q_y}^2 + v_{q_z}^2 \tag{A.7}$$

- *Norm*: The norm of a quaternion is defined as:

$$\|\boldsymbol{q}\| = \sqrt{\boldsymbol{q}\boldsymbol{q}^*} = \sqrt{\boldsymbol{q}^*\boldsymbol{q}} \tag{A.8}$$

  A quaternion with unity norm, i.e $\|\boldsymbol{q}\| = 1$ is called unit quaternion and can be used to represent rotation operation akin to rotation matrix.

- *Inverse*: The inverse of a quaternion is defined as:

$$\boldsymbol{q}^{-1} = \frac{\boldsymbol{q}^*}{\|\boldsymbol{q}\|^2} \tag{A.9}$$

  The inverse of a unit quaternion is its conjugate as can be verified from above relation.

A.1.2 Rotation representation using unit quaternions

A rotation of angle $\theta \in [0.\pi]$ around a unit vector $\boldsymbol{u}$ can be represented using a unit quaternion as (see Fig. A.1):

$$\boldsymbol{q} = e^{\frac{\theta}{2}(u_x\boldsymbol{i}+u_y\boldsymbol{j}+u_z\boldsymbol{k})} = \cos\frac{\theta}{2} + (u_x\boldsymbol{i} + u_y\boldsymbol{j} + u_z\boldsymbol{k})\sin\frac{\theta}{2} \tag{A.10}$$

A rotation represented by $\boldsymbol{q}$ can be applied to a vector $\boldsymbol{v} = v_x\boldsymbol{i} + v_y\boldsymbol{j} + v_z\boldsymbol{k}$, and the vector thus obtained is given as:

$$\boldsymbol{v'} = \boldsymbol{q}\boldsymbol{v}\boldsymbol{q}^* \tag{A.11}$$

A.2 Dual Numbers

Dual numbers, proposed by Clifford [107], is defined with a real and dual part, a and b respectively, as:

$$\hat{d} = a + \varepsilon b, \tag{A.12}$$

where $a, b \in \mathbb{R}$ and ε is a *dual unit*, such that $\varepsilon^2 = 0$.

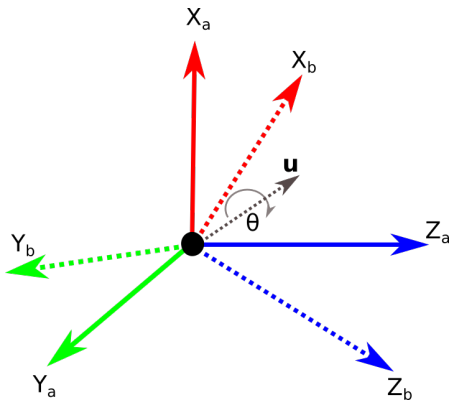


FIG. A.1 Frame rotation using unit quaternion.

A.2.1 Dual number operations

- *Addition*: Addition of two dual numbers $\hat{d}_1 = a_1 + \varepsilon b_1$ and $\hat{d}_2 = a_2 + \varepsilon b_2$ is given as:

$$\hat{d}_1 + \hat{d}_2 = (a_1 + a_2) + \varepsilon(b_1 + b_2). \tag{A.13}$$

- *Multiplication*: The multiplication of two dual number is given as:

$$\hat{d}_1 \hat{d}_2 = a_1 a_2 + \varepsilon(a_1 b_2 + a_2 b_1). \tag{A.14}$$

- *Inverse*: Inverse of a dual number $\hat{d} = a + \varepsilon b$ is defined for the condition $a \neq 0$, as:

$$\hat{d}^{-1} = a^{-1}(1 - \varepsilon b a^{-1}). \tag{A.15}$$

- *Function of dual number*: A function of a dual number $\hat{d} = a + \varepsilon b$ is defined using Taylor expansion yields:

$$f(a + \varepsilon b) = f(a) + \varepsilon b f'(a), \tag{A.16}$$

since all terms with two or higher power of ε vanish.

A.3 Dual Vectors

Dual vectors have 3D vectors as both real and dual parts, or can also be defined as vectors where each entry of the vector is a dual number. Dual vectors with a unit vector as real part and orthogonal real and dual parts are the representation of spatial lines in $\mathbb{R}^3$, also known as Plücker lines, as shown in Fig. A.2:

$$\hat{\boldsymbol{s}} = \boldsymbol{l} + \varepsilon \boldsymbol{m} \tag{A.17}$$

A.3.1 Dual vectors operations

- The product of a dual number $\hat{d}$ and a dual vector $\hat{\boldsymbol{d}} = (\hat{d}_1, \hat{d}_2, \hat{d}_3)$ is given as:

$$\hat{d}\hat{\boldsymbol{d}} = (\hat{d}\hat{d}_1, \hat{d}\hat{d}_2, \hat{d}\hat{d}_3) \tag{A.18}$$

- The inner product of two dual vectors $\hat{\boldsymbol{d}} = (\hat{d}_1, \hat{d}_2, \hat{d}_3)$ and $\hat{\boldsymbol{e}} = (\hat{e}_1, \hat{e}_2, \hat{e}_3)$ is defined as:

$$\hat{\boldsymbol{d}} \cdot \hat{\boldsymbol{e}} = \hat{d}_1\hat{e}_1 + \hat{d}_2\hat{e}_2 + \hat{d}_3\hat{e}_3 \tag{A.19}$$

- The cross product of two dual vectors $\hat{\boldsymbol{d}}$ and $\hat{\boldsymbol{e}}$ is given as :

$$\hat{\boldsymbol{d}}\hat{\boldsymbol{e}} = \begin{pmatrix} \hat{d}_2\hat{e}_3 - \hat{d}_3\hat{e}_2 \\ \hat{d}_3\hat{e}_1 - \hat{d}_1\hat{e}_3 \\ \hat{d}_1\hat{e}_2 - \hat{d}_2\hat{e}_1 \end{pmatrix} \tag{A.20}$$

If $\boldsymbol{l}$ is a unit vectors then, it can regarded as the direction vector of the line and the dual part $\boldsymbol{m}$ is the moment of the line about the origin of the reference frame, given as $\boldsymbol{m} = \boldsymbol{p} \times \boldsymbol{l}$, where $\boldsymbol{p}$ can be any point on the line. The inner product of two Plücker lines yields the cosine of a dual angle $\hat{\theta} = \theta + \varepsilon d$, where d is the relative distance between the lines and θ is the angle between the two Plücker lines.

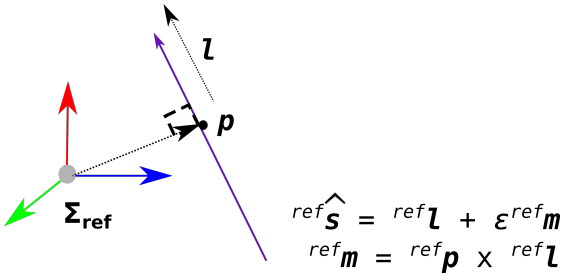


FIG. A.2 Plücker line in reference frame: Represented by a dual vector $\hat{\boldsymbol{s}}$ with direction unit vector $\boldsymbol{l}$ as the real part and moment of the line $\boldsymbol{m}$ computed with respect to the origin of the reference frame $\boldsymbol{\Sigma}_{ref}$.

A.4 Dual Quaternions

Dual quaternions are extension of dual numbers with real and dual quaternion components which combines Hamilton quaternion algebra with the dual number theory [107]. A dual quaternion can be written in following forms:

$$\hat{\boldsymbol{x}} = \boldsymbol{q}_r + \varepsilon \boldsymbol{q}_d \tag{A.21}$$

$$= \hat{s} + \hat{\boldsymbol{v}} = (s_p + \varepsilon s_q) + (\boldsymbol{v}_p + \varepsilon \boldsymbol{v}_q) \tag{A.22}$$

$$= (s_r,\, v_{rx},\, v_{ry},\, v_{rz},\, s_d,\, v_{dx},\, v_{dy},\, v_{dz}) \tag{A.23}$$

The first dual quaternion representation has quaternions where $\boldsymbol{q}_r$ and $\boldsymbol{q}_d$ are real and dual components of dual quaternion $\hat{\boldsymbol{x}}$, respectively. The second representation has $\hat{s}$ and $\hat{\boldsymbol{v}}$ as the dual number and dual vector components of the given dual quaternion. The last representation considers a dual quaternion as an 8-tuple.

A.4.1 Dual quaternion operations

- *Addition*: Given two dual quaternions $\hat{\boldsymbol{x}}_1 = \boldsymbol{q}_{r_1} + \varepsilon \boldsymbol{q}_{d_1}$ and $\hat{\boldsymbol{x}}_2 = \boldsymbol{q}_{r_2} + \varepsilon \boldsymbol{q}_{d_2}$, addition operation of dual quaternions is defined as:

$$\hat{\boldsymbol{x}}_1 + \hat{\boldsymbol{x}}_2 = (\boldsymbol{q}_{r_1} + \boldsymbol{q}_{r_2}) + \varepsilon(\boldsymbol{q}_{d_1} + \boldsymbol{q}_{d_2}). \tag{A.24}$$

- *Multiplication*: The product of two dual quaternions $\hat{\boldsymbol{x}}_1$ and $\hat{\boldsymbol{x}}_2$ is defined as:

$$\hat{\boldsymbol{x}}_1 \hat{\boldsymbol{x}}_2 = \boldsymbol{q}_{r_1} \boldsymbol{q}_{r_2} + \varepsilon(\boldsymbol{q}_{r_1} \boldsymbol{q}_{d_2} + \boldsymbol{q}_{d_1} \boldsymbol{q}_{r_2}) \tag{A.25}$$

- *Conjugate*: There are three conjugates that can be defined for a dual quaternion, which are used for different operations like Plücker line transformation, or point transformation.
  - *Classical quaternion conjugate*: This conjugates refers to the representation of dual quaternion as a variation of dual numbers with quaternions as real and dual components and is used for line transformation.

$$\hat{\boldsymbol{x}}^* = \boldsymbol{q}_r^* + \varepsilon \boldsymbol{q}_d^*. \tag{A.26}$$

– *Dual conjugate*: Dual conjugate is similar to the conjugate of a dual number and is given as:

$$\overline{\hat{\boldsymbol{x}}} = \boldsymbol{q}_r - \varepsilon \boldsymbol{q}_d. \tag{A.27}$$

– *Combined conjugate*: is used for point transformation and is defined as:

$$\overline{\hat{\boldsymbol{x}}}^* = \boldsymbol{q}_r^* - \varepsilon \boldsymbol{q}_d^*. \tag{A.28}$$

The conjugate of the conjugate of $\hat{\boldsymbol{x}}$, for all the conjugates introduced above, is itself. The conjugate of the product of dual quaternions equals the product of the individual conjugates of these dual quaternions in the reverse order.

- *Norm*: The norm of a dual quaternion is obtained using classical quaternion conjugate and is given as:

$$\|\hat{\boldsymbol{x}}\| = \sqrt{\hat{\boldsymbol{x}}\hat{\boldsymbol{x}}^*} = \sqrt{\hat{\boldsymbol{x}}^*\hat{\boldsymbol{x}}} \tag{A.29}$$

$$= \sqrt{(s_r^2 + \boldsymbol{v}_r \cdot \boldsymbol{v}_r) + \varepsilon\, 2(s_r s_d + \boldsymbol{v}_r \boldsymbol{v}_d)} \tag{A.30}$$

A unit dual quaternion has unity norm, i.e.:

$$\|\hat{\boldsymbol{x}}\| = \sqrt{\hat{\boldsymbol{x}}\hat{\boldsymbol{x}}^*} = \sqrt{\hat{\boldsymbol{x}}^*\hat{\boldsymbol{x}}} = 1, \tag{A.31}$$

which implies that real part quaternion $\boldsymbol{q}_r$ should be a unit quaternion and it should be orthogonal to the dual part $\boldsymbol{q}_d$ as 4-tuples:

$$s_r{}^2 + v_{rx}{}^2 + v_{ry}{}^2 + v_{rz}{}^2 = 1, \tag{A.32}$$

$$s_r s_d + v_{rx} v_{dx} + v_{ry} v_{dy} + v_{rz} v_{dz} = 0. \tag{A.33}$$

The above equations shows that unit dual quaternions belongs to a six-dimensional manifold and can be used to represent rigid body displacement.

A.4.2 Transformations using unit dual quaternions

A unit dual quaternion can be used to represent rigid body motion or a directed line. They are also useful for transforming geometrical entities like lines, points, etc.

A.4.2.1 Rigid body motion and pose representaion

A rigid body motion can be represented using unit dual quaternions a single screw motion as well as with a series of transformation similar to an order sequence of translation and rotational motion. Both forms of the representation of rigid body motion is equally useful and has been explained below, along with the methods to obtain the motion parameters when the corresponding UDQ is given.

- *HTM equivalent rigid body motion using UDQ*: A rigid body displacement equivalent of homogeneous transformation matrix is given as:

$$\hat{\boldsymbol{x}} = \boldsymbol{q}_r + \varepsilon \frac{1}{2} \boldsymbol{t} \boldsymbol{q}_r \tag{A.34}$$

The above unit dual quaternion represents a sequence of rigid body motion, where a frame attached to the rigid body is first translated by $\boldsymbol{t}$ and then rotated using the unit quaternion $\boldsymbol{q}_r$, as demonstrated in Fig. A.3.

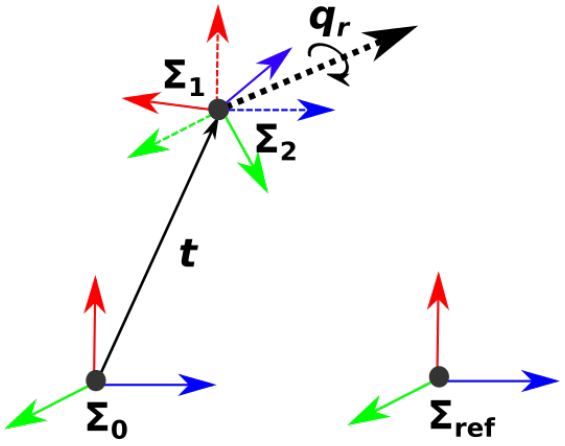


FIG. A.3 Rigid body motion akin to one represented using homogeneous transformation matrix. The rigid body frame ($\boldsymbol{\Sigma}_0$) is first translated by a vector $\boldsymbol{t}$ to $\boldsymbol{\Sigma}_{ref}$ to $\boldsymbol{\Sigma}_1$ frame (shown with dotted axes) and rotated using a unit quaternions $\boldsymbol{q}_r$ to $\boldsymbol{\Sigma}_2$ frame, where all the motion and frames are computed and represented in the reference frame. $\boldsymbol{\Sigma}_{ref}$.

The rotation parameter corresponding to a unit dual quaternion ($\hat{\boldsymbol{x}} = \boldsymbol{q}_r + \varepsilon \boldsymbol{q}_d$) in this representation or rigid body motion is derived in the same way as the screw based representation. The translation component as a pure quaternion is given as:

$$\boldsymbol{t} = 2\boldsymbol{q}_d \boldsymbol{q}_r^* \tag{A.35}$$

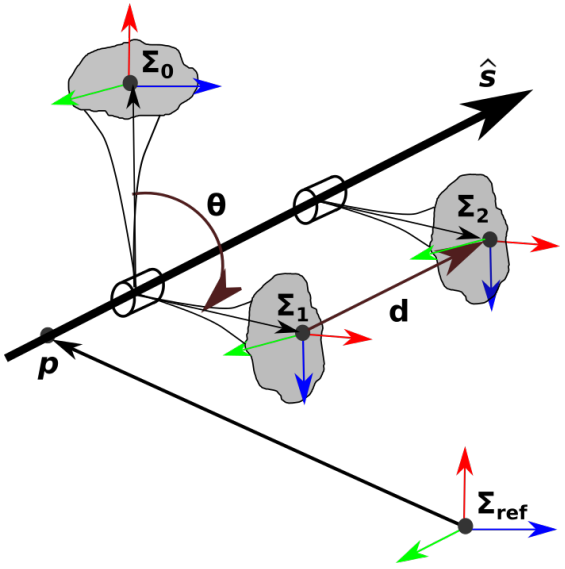


FIG. A.4 Screw displacement of a rigid body: $\boldsymbol{p}$ is any point lying on the screw axis $\hat{\boldsymbol{s}}$, both computed and represented in reference frame $\boldsymbol{\Sigma}_{ref}$. The rigid body first undergoes a rotation by θ along $\hat{\boldsymbol{s}}$ and then a translation by d along the same axis.

The method for deriving the angle-axis parameters can be derived from the rotation quaternion, i.e. the primary part of the given dual-quaternion, will be the same for both HTM equivalent representation and screw-displacement representation, which is given below.

- *Screw displacement*: Chasles's theorem states that the general spatial displacement of a rigid body is screw displacement, i.e. a rotation about an axis and a translation along the same axis, as depicted in Fig. A.4. The corresponding unit dual quaternion, with real component as rotation unit quaternion, is given as:

$$\hat{\boldsymbol{x}} = \exp\left(\frac{\hat{\theta}}{2} \cdot \hat{\boldsymbol{s}}\right) \tag{A.36}$$

$$= \cos\left(\frac{\hat{\theta}}{2}\right) + \hat{\boldsymbol{s}} \sin\left(\frac{\hat{\theta}}{2}\right) \tag{A.37}$$

$$= \left(\cos\left(\frac{\theta}{2}\right) + \boldsymbol{l}\sin\left(\frac{\theta}{2}\right)\right) + \varepsilon\left(-\frac{d}{2}\sin\left(\frac{\theta}{2}\right) + \boldsymbol{l}\frac{d}{2}\cos\left(\frac{\theta}{2}\right) + \boldsymbol{m}\sin\left(\frac{\theta}{2}\right)\right) \tag{A.38}$$

where, $\hat{\theta}$ is a dual angle containing the screw rotation and translation parameters, and $\hat{\boldsymbol{s}}$ is a Plücker line given as a dual vector (see Fig. A.2).

$$\hat{\theta} = \theta + \varepsilon d \tag{A.39}$$

$$\hat{\boldsymbol{s}} = \boldsymbol{l} + \varepsilon \boldsymbol{m} \tag{A.40}$$

$\boldsymbol{l}$ is the unit vector along the axis related to the screw displacement and $\boldsymbol{m} = \boldsymbol{p} \times \boldsymbol{l}$ corresponds to the moment of the directed line (Plücker line) in the reference frame at which the screw displacement is computed. $\boldsymbol{p}$ is a vector from the origin of the reference frame to any point lying on the directed line.

Computational cost of representation of screw-based displacement using UDQ: The computation of UDQ from given screw parameters, $\boldsymbol{l}$, $\boldsymbol{m}$, d, and θ using expression A.38 will require:

- Two trigonometric operations for the computation of $\sin\left(\frac{\theta}{2}\right)$ and $\cos\left(\frac{\theta}{2}\right)$.
- 13 multiplication ($\times$) operation for the computation of $d/2(1\ \times)$, $\theta/2(1\ \times)$, $\boldsymbol{l}\sin\left(\frac{\theta}{2}\right)(3\ \times)$, $\frac{d}{2}\sin\left(\frac{\theta}{2}\right)(1\ \times)$, $\frac{d}{2}\cos\left(\frac{\theta}{2}\right)(1\ \times)$, $\boldsymbol{l}\frac{d}{2}\cos\left(\frac{\theta}{2}\right)(3\ \times)$, and $\boldsymbol{m}\sin\left(\frac{\theta}{2}\right)(3\ \times)$.
- Three addition ($+$) operations for the computation of $\left(\boldsymbol{l}\frac{d}{2}\cos\left(\frac{\theta}{2}\right) + \boldsymbol{m}\sin\left(\frac{\theta}{2}\right)\right)$.

UDQ to screw parameters : Given a unit dual quaternion ($\hat{\boldsymbol{x}} = \hat{s} + \hat{\boldsymbol{v}} = (s_p + \varepsilon s_q) + (\boldsymbol{v}_p + \varepsilon \boldsymbol{v}_q)$) corresponding to a rigid body motion, the computation of screw parameters (θ, d, $\boldsymbol{l}$ and $\boldsymbol{m}$) requires different approaches depending if there is rotation involved during the motion.

First, the rotation angle is derived from the scalar part of the real quaternion component of the unit dual quaternion (comparing A.22 and A.37):

$$\theta = 2\arccos(s_p). \tag{A.41}$$

If there is no rotation involved, i.e. $s_p = 1$ or $\theta = 0$, the dual quaternion corresponds to pure translation and the screw components are given as:

$$d = 2 \left\| \hat{\boldsymbol{v}}_q \right\| \tag{A.42}$$

$$\boldsymbol{l} = 2\frac{\boldsymbol{v}_q}{d} \tag{A.43}$$

$$\boldsymbol{m} = [0,\ 0,\ 0]^T . \tag{A.44}$$

If $\theta \neq 0$ and $0 < \theta < 2\pi$, screw parameters are given as:

$$d = -2\frac{s_q}{\sin(\theta/2)} \tag{A.45}$$

$$\boldsymbol{l} = \frac{\boldsymbol{v}_p}{\sin(\theta/2)} \tag{A.46}$$

$$\boldsymbol{m} = \left(\boldsymbol{v}_q - s_p \frac{d}{2} \boldsymbol{l} \right) \frac{1}{\sin(\theta/2)}. \tag{A.47}$$

A.4.2.2 Point tranformation

A 3D point in cartesian coordinates $\boldsymbol{p}$, given as pure quaternion, can be represented as a unit dual quaternion:

$$\hat{\boldsymbol{p}} = 1 + \varepsilon \boldsymbol{p} \tag{A.48}$$

A point $\boldsymbol{p}$, represented in unit dual quaternion form ($\hat{\boldsymbol{p}}$), as shown above undergoing a transformation represented using unit dual quaternion $\hat{\boldsymbol{x}}$ can be given as:

$$\hat{\boldsymbol{p}}_b = \hat{\boldsymbol{x}} \hat{\boldsymbol{p}}_a \overline{\hat{\boldsymbol{x}}}^* \tag{A.49}$$

where $\overline{\hat{\boldsymbol{x}}}^*$ is the combined conjugate of the transformation unit dual quaternion $\hat{\boldsymbol{x}}$.

A.4.2.3 Line tranformation

A directed line or Plücker line represented using dual vector (see sec. A.3) is a unit dual quaternion, since it satisfies both conditions of unit quaternion as real part and orthogonal relation between real and dual part of the dual quaternion given in A.32.

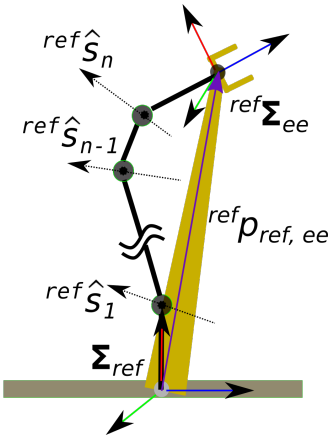


FIG. A.5 A serial manipulator and its joint screw axes in the reference frame $\boldsymbol{\Sigma}_{ref}$. The current end-effector position computed and represented in the reference frame is given by the vector ${}^{ref}\boldsymbol{p}_{ref}$.

A directed line represented by a dual vector $\hat{\boldsymbol{s}}_a$ undergoing a transformation represented by unit dual quaternion $\hat{\boldsymbol{x}}$ can be given as:

$$\hat{\boldsymbol{s}}_b = \hat{\boldsymbol{x}}\hat{\boldsymbol{s}}_a\hat{\boldsymbol{x}}^* \tag{A.50}$$

where $\hat{\boldsymbol{x}}^*$ is the classical quaternion conjugate of unit dual quaternon $\hat{\boldsymbol{x}}$.

A.5 Serial manipulator kinematics using unit dual quaternions

A forward kinematic model of a manipulator is used to obtain the position of its constituent links and end-effector for a given joint configuration. Inverse kinematics deals with the computation of the joint positions for a desired end-effector pose and the most common method is through the use of inverse Jacobian of the manipulator. This section presents a dual quaternion based forward and inverse kinematics of serial manipulator, where the theory developed in [14] is revisited. All the variables related to pose and directed lines such as joint screw axes are computed and represented in a common reference frame $\boldsymbol{\Sigma}_{ref}$.

A.5.1 Forward Kinematics of a serial manipulator

Let $\hat{\boldsymbol{s}}_i$ be the directed line corresponding to the current joint screw axes of a serial manipulator with n joints (shown in Fig. A.5), represented as a dual vector for $i = 1, ..., n$. Let $\hat{\boldsymbol{x}}_{ee_0}$ be the initial pose of the end-effector with respect to the base frame, for the initial configuration of the joints given as an array of dual numbers $\underline{\hat{\boldsymbol{\theta}}}_0 = \begin{bmatrix} \hat{\theta}_{1_0} & \hat{\theta}_{2_0} & \cdots & \hat{\theta}_{n_0} \end{bmatrix}^T \in \mathrm{D}^{n\times 1}$. The end-effector pose $\hat{\boldsymbol{x}}_{ee}$ relative to the base frame, $\boldsymbol{\Sigma}_{ref}$, for a joint configuration $\underline{\hat{\boldsymbol{\theta}}} = \begin{bmatrix} \hat{\theta}_1 & \hat{\theta}_2 & \cdots & \hat{\theta}_n \end{bmatrix}^T \in \mathrm{D}^{n\times 1}$ is given as:

$$\begin{aligned} \hat{\boldsymbol{x}}_{ee} &= \hat{\boldsymbol{\delta}}_T \; \hat{\boldsymbol{x}}_{ee_0}, \\ \hat{\boldsymbol{\delta}}_T &= \hat{\boldsymbol{\delta}}_1 \; \hat{\boldsymbol{\delta}}_2 \; \cdots \; \hat{\boldsymbol{\delta}}_n, \\ \hat{\boldsymbol{\delta}}_i &= \exp\left(\frac{\hat{\theta}_i}{2} \; \hat{\boldsymbol{s}}_{i_0}\right). \end{aligned} \tag{A.51}$$

where,

- $\hat{\boldsymbol{s}}_{i_0}$ is the initial joint screw axis described using unit vector along the axis (i.e. $\boldsymbol{l}_{i_0}$), and moment of the joint screw axis about the base frame of the manipulator (i.e. $\boldsymbol{m}_{i_0}$),
- dual vector $\hat{\boldsymbol{\delta}}_i$ is the displacement effected on the end-effector by the ith joint of the manipulator due to joint displacement $\hat{\theta}_i$, and
- $\hat{\theta}_i$ is a joint displacement from the home position $\hat{\theta}_{i_0}$

$$\begin{aligned} \hat{\theta}_i &= \Delta\theta_i, && \text{for revolute joints,} \\ \hat{\theta}_i &= \varepsilon \; \Delta d_i, && \text{for prismatic joints.} \end{aligned} \tag{A.52}$$

Computational cost for the forward kinematics computation using screw-based method using dual-quaternion representation: The product of DQ consists of 3 quaternion multiplications and a sum of two quaternions. Multiplication of two quaternions involves one dot product (3 × operations and 2 + operations), one cross product (6 × operations, and 3 + operations), and two vector scaling operations (6× operations) and one scalar

multiplication ($1 \times$ operation) and 7 additions ($7 +$ operations). Thus total computation requirements for pose transformation using UDQ is $3(3+6+6+1) = 48$ multiplication operations, and $3(2+3+7)+4 = 40$ addition operations.

Now, the computation of forward kinematics for a serial manipulator with n joints involve n times computation of UDQ representation of screw-based displacement using A.38. It also involves $(n-1)$ UDQ multiplications. Thus the total computation of forward kinematics requires $48(n-1)+13n = (61n-48)$ multiplication $(\times)$ operations, and $3n+40(n-1) = 43n-40$ addition/substraction $(+)$ operations and $2n$ trigonometric operations.

A.5.2 Jacobian computation of a serial manipulator

The screw twist of the end-effector frame in a reference frame is given as:

$$^{ref}\hat{\boldsymbol{\xi}}_{ee_o} = \boldsymbol{\omega}_{ee} + \varepsilon\ \boldsymbol{v}_{ee_o}. \tag{A.53}$$

where, $\boldsymbol{\omega}_{ee}$ is the rotational velocity of the end-effector frame, and $\boldsymbol{v}_{ee_o}$ is the translational velocity of a point which is attached to the end-effector frame and is instantaneously coincident with the origin of reference frame $\boldsymbol{\Sigma}_{ref}$ (see Fig. A.5). $\boldsymbol{\omega}_{ee}$ is defined as a pure quaternion non-dual part, and $\boldsymbol{v}_{ee_o}$ is defined as a pure quaternion dual part of the dual quaternion $^{ref}\hat{\boldsymbol{\xi}}_{ee_o}$.

The twist of the end-effector in the base frame is given as:

$$^{ref}\hat{\boldsymbol{\xi}}_{ee_o} = \begin{bmatrix} \hat{\boldsymbol{s}}_1 & \hat{\boldsymbol{s}}_2 & \cdots & \hat{\boldsymbol{s}}_n \end{bmatrix} \dot{\underline{\hat{\boldsymbol{\theta}}}} = \hat{\boldsymbol{J}}'\dot{\underline{\hat{\boldsymbol{\theta}}}} \tag{A.54}$$

Therefore, the corresponding Jacobian is given as:

$$\hat{\boldsymbol{J}}' = \begin{bmatrix} \hat{\boldsymbol{s}}_1 & \hat{\boldsymbol{s}}_2 & \cdots & \hat{\boldsymbol{s}}_n \end{bmatrix}. \tag{A.55}$$

The current configuration of an intermediate joint screw of the manipulator represented

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

$$\hat{\boldsymbol{s}}_i = \hat{\boldsymbol{\delta}}_{T_{(i-1)}} \; \hat{\boldsymbol{s}}_{i_0} \; \hat{\boldsymbol{\delta}}^*_{T_{(i-1)}} \tag{A.56}$$

$$\hat{\boldsymbol{\delta}}_{T_{(i-1)}} = \hat{\boldsymbol{\delta}}_1 \; \hat{\boldsymbol{\delta}}_2 \; \cdots \; \hat{\boldsymbol{\delta}}_{i-1}. \tag{A.57}$$

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

A matrix-vector representation of (A.55) was given in [14] as:

$$^{ref}\boldsymbol{\xi}_{ee_o} = \begin{bmatrix} \boldsymbol{\omega}_{ee} \\ \\ \boldsymbol{v}_{ee_o} \end{bmatrix} = \begin{bmatrix} \boldsymbol{L} & \boldsymbol{0} \\ \\ \boldsymbol{M} & \boldsymbol{L} \end{bmatrix} \begin{bmatrix} \underline{\dot{\boldsymbol{\theta}}} \\ \\ \underline{\dot{\boldsymbol{d}}} \end{bmatrix} \tag{A.58}$$

$$\boldsymbol{\omega}_{ee} = \boldsymbol{L} \; \underline{\dot{\boldsymbol{\theta}}}, \tag{A.59}$$

$$\begin{aligned} \boldsymbol{v}_{ee_o} &= \boldsymbol{M} \; \underline{\dot{\boldsymbol{\theta}}} + \boldsymbol{L} \; \underline{\dot{\boldsymbol{d}}} \\ &= \boldsymbol{P} \otimes \boldsymbol{L} \; \underline{\dot{\boldsymbol{\theta}}} + \boldsymbol{L} \; \underline{\dot{\boldsymbol{d}}} \end{aligned} \tag{A.60}$$

where,

$$\underline{\dot{\boldsymbol{\theta}}} = \begin{bmatrix} \dot{\theta}_1 & \dot{\theta}_2 & \cdots & \dot{\theta}_n \end{bmatrix}^T \in \mathbb{R}^{n\times 1}, \tag{A.61}$$

$$\underline{\dot{\boldsymbol{d}}} = \begin{bmatrix} \dot{d}_1 & \dot{d}_2 & \cdots & \dot{d}_n \end{bmatrix}^T \in \mathbb{R}^{n\times 1}, \tag{A.62}$$

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{l}_1 & \boldsymbol{l}_2 & \cdots & \boldsymbol{l}_n \end{bmatrix} \in \mathbb{R}^{n\times 3}, \tag{A.63}$$

$$\boldsymbol{P} = \begin{bmatrix} \mathrm{p}_1 & \mathrm{p}_2 & \cdots & \mathrm{p}_n \end{bmatrix} \in \mathbb{R}^{n\times 3}, \tag{A.64}$$

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{m}_1 & \boldsymbol{m}_2 & \cdots & \boldsymbol{m}_n \end{bmatrix} \in \mathbb{R}^{n\times 3}, \; and \tag{A.65}$$

$$\otimes \text{ is element-wise cross-product operator.} \tag{A.66}$$

Let ${}^{ref}\boldsymbol{p}_{ref,ee} = \boldsymbol{p}_{ee}$ be the position of end-effector frame origin computed and given with respect to reference frame, $\boldsymbol{\Sigma}_{ref}$. The relation between the linear velocity of a point on the end-effector frame ($\boldsymbol{v}_{ee}$), and the velocity of a point attached to the end-effector but instantaneously coincident with the reference frame, i.e. $\boldsymbol{v}_{ee_0}$ is:

$$\boldsymbol{v}_{ee} = \boldsymbol{v}_{ee_o} + \boldsymbol{\omega}_{ee} \times \boldsymbol{p}_{ee}. \tag{A.67}$$

(A.67) can be written in a matrix form using (A.58) as:

$$\boldsymbol{v}_{ee} = (\boldsymbol{P} - \boldsymbol{P}_{ee}) \otimes \boldsymbol{L}\ \underline{\dot{\boldsymbol{\theta}}} + \boldsymbol{L}\ \underline{\dot{\boldsymbol{d}}}, \tag{A.68}$$

where,

$$\boldsymbol{P}_{ee} = \begin{bmatrix} \boldsymbol{p}_{ee} & \boldsymbol{p}_{ee} & \cdots & \boldsymbol{p}_{ee} \end{bmatrix} \in \mathbb{R}^{(n)\times 3}. \tag{A.69}$$

$\boldsymbol{p}_{ee}$ can be derived as a vector from the current pose of the end-effector frame ($\hat{\boldsymbol{x}}_{ee}$) computed in (A.51), as:

$$\boldsymbol{p}_{ee} = 2\boldsymbol{q}_t\boldsymbol{q}_r^*, \tag{A.70}$$

where $\hat{\boldsymbol{x}}_{ee} = \boldsymbol{q}_r + \epsilon\boldsymbol{q}_t$, and $\boldsymbol{q}_r^*$ is the quaternion conjugate of $\boldsymbol{q}_r$.

Computational cost for the screw-based Jacobian computation for a serial manipulation using dual-quaternion representation: In order to compute the Jacobian as given in (eq. (A.55), (A.56) and (A.57)), we need to first compute the forward kinematics and we need two additional dual quaternion multiplication for each joints to transform the screw axes (A.56) except the first joint screw axis, to obtain the DQs corresponding the joint screw axes for the current joint configurations. So we need totally $(2*(n-1)*48+61n-48 = 157n-144)$ multiplication operations, and $2*(n-1)*40+43n-40 = 123n-120$ addition operations, in addition to 2 trigonometric operations for the computation of forward kinematics.

A.5.3 Proportional kinematic control using screw parameters

Given two poses, i.e. current ($\hat{\boldsymbol{x}}_c$) and desired ($\hat{\boldsymbol{x}}_d$), both expressed in the same reference frame, there are following two choices for error unit dual quaternion computation.

$$\hat{\boldsymbol{x}}_{e1} = \hat{\boldsymbol{x}}_c^* \cdot \hat{\boldsymbol{x}}_d = {}^{c}\hat{\boldsymbol{x}}_{c\to d} \tag{A.71}$$

$$\hat{\boldsymbol{x}}_{e2} = \hat{\boldsymbol{x}}_d \cdot \hat{\boldsymbol{x}}_c^* = {}^{ref}\hat{\boldsymbol{x}}_{c\to d} \tag{A.72}$$

Since joint screw axes, and hence the Jacobian has been computed in the reference frame in A.55, the definition of error given in A.72 is the appropriate choice for the computing joint velocity as control input to track a desired pose. ${}^{ref}\hat{\boldsymbol{x}}_{c\to d}$ refers to a screw displacement vector directed from the current frame c to the desired frame d, expressed in the reference frame $\boldsymbol{\Sigma}_{ref}$.

The control law for kinematic control was given in [14] in terms of the logarithm of the error unit dual quaternion:

$$\begin{aligned}\hat{\boldsymbol{\xi}} &= \lambda \ \ln(\hat{\boldsymbol{x}}_{e2}) \\ &= \lambda(\theta_e \ \boldsymbol{l}_e + \varepsilon \ (\theta_e \ \boldsymbol{m}_e + d_e \ \boldsymbol{l}_e))\end{aligned} \tag{A.73}$$

where, $\{\theta_e,\ d_e,\ \boldsymbol{l}_e,\ \boldsymbol{m}_e\}$ are screw displacement parameter related to the error unit dual quaternion $\hat{\boldsymbol{x}}_{e2}$ (see A.4.2.1), and λ is a positive scalar gain for the controller. The global exponential stability of the above mentioned control law ($\hat{\boldsymbol{\xi}}$) was proved for $-\pi \geq \theta_e \geq \pi$ in [14] using Lyapunov analysis.

The norm of the screw error, which has been used throughout the work for the validation of screw theory and dual quaternion based manipulation based controllers is computed as:

$$screwNorm_{error} = \|\boldsymbol{l}_e\theta_e\| + \|\boldsymbol{l}_e d_e + \boldsymbol{m}_e\theta_e\| \tag{A.74}$$

A.6 Spatial Vectors: Velocity and Acceleration

A screwing motion along a directed line is the most general motion of a rigid body, which is referred to as spatial motion by Featherstone in [108, 95]. A screw motion is

characterised with an angular magnitude, linear magnitude and a directed line. Spatial motion vectors include spatial velocity, or *twist velocity*, and spatial acceleration, which is the derivative of spatial velocity. Spatial motion vectors along with force vectors form two dual 6D vector spaces called $\mathbf{M}^6$ and $\mathbf{F}^6$, which are defined in the same way as a *screw twist*. Spatial forces, or *wrench* and spatial momentum are the members of $\mathbf{F}^6$.

In this section the relation of spatial variables, like spatial velocity and spatial acceleration with conventional twist and conventional acceleration is discussed.

A.6.1 Spatial Velocity

Consider a point P on a rigid body, moving with an angular velocity $\boldsymbol{\omega}$ and linear velocity $\boldsymbol{v}_P$ given in a reference frame $\boldsymbol{\Sigma}_{ref}$ with origin O. The spatial acceleration in 6D vector form is given as:

$$\begin{bmatrix} \boldsymbol{\omega} \\ \\ \boldsymbol{v}_o \end{bmatrix} = \begin{bmatrix} \boldsymbol{\omega} \\ \\ \boldsymbol{v}_P + \overrightarrow{OP} \times \boldsymbol{\omega} \end{bmatrix} \tag{A.75}$$

The same can be represented as a dual vector as:

$$\hat{\boldsymbol{\omega}} = \boldsymbol{\omega} + \varepsilon \boldsymbol{v}_o \tag{A.76}$$

Thus spatial velocity of a point attached to the rigid body but instantaneously passing through the origin of the reference frame. Velocity motor and twist velocity are other common names for spatial velocity.

Conversely, the dual vector of the velocity of a point P on a rigid body in 6D form can be obtained from the spatial velocity and the position vector from the origin of the reference frame to the point P on the rigid body:

$$\begin{bmatrix} \boldsymbol{\omega} \\ \\ \boldsymbol{v}_P \end{bmatrix} = \begin{bmatrix} \boldsymbol{\omega} \\ \\ \boldsymbol{v}_o - \overrightarrow{OP} \times \boldsymbol{\omega} \end{bmatrix} \tag{A.77}$$

A.6.2 Spatial Acceleration

Spatial acceleration is the derivative of screw velocity and defines a helicoidal vector field [109].

$$\hat{\boldsymbol{a}} = \dot{\hat{\boldsymbol{\omega}}} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \dot{\boldsymbol{v}}_o \end{bmatrix} \tag{A.78}$$

The angular acceleration $\dot{\hat{\boldsymbol{\omega}}}$ of a body in *classical* sense and spatial dynamics is the same. However, the linear acceleration of a point attached to the body is quite different in both paradigm. Below, we provide the steps to obtain classical acceleration of a general point attached to a rigid body from its spatial acceleration and vice-versa.

A.6.2.1 Spatial to classical acceleration

Let $\hat{\boldsymbol{a}}_{c_o} = \dot{\boldsymbol{\omega}} + \varepsilon\ \boldsymbol{a}_{c_o}$ represents the *conventional* acceleration of a frame c_o rigidly attached to the end-effector and instantaneously coincident with the origin of the base frame. Given spatial acceleration $\boldsymbol{a} = \dot{\boldsymbol{\omega}} + \varepsilon\dot{\boldsymbol{v}}_o$, and spatial velocity $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega} + \varepsilon\boldsymbol{v}_o$, $\hat{\boldsymbol{a}}_{c_o}$ is computed as:

$$\hat{\boldsymbol{a}}_{c_o} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \boldsymbol{a}_{c_o} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \dot{\boldsymbol{v}}_o + \boldsymbol{\omega} \times \boldsymbol{v}_o \end{bmatrix} \tag{A.79}$$

Note that the dual part of $\boldsymbol{a}_{c_o}$, *i.e.* $\boldsymbol{a}_{c_o}$ is not the derivative of $\boldsymbol{v}_{c_o}$ (or $\boldsymbol{v}_o$). In fact, $\boldsymbol{a}_{c_o}$ "refers to the acceleration of an individual body-fixed point at the moment when it happens to be passing through the origin" [109].

The dual classical acceleration comprising the linear acceleration of a general point, say P, *i.e.* origin of a frame on the end-effector, can be obtained as:

$$\hat{\boldsymbol{a}}_{c_P} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \boldsymbol{a}_{c_P} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \boldsymbol{a}_{c_o} + \dot{\boldsymbol{\omega}} \times \overrightarrow{\mathrm{OP}} + \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \overrightarrow{\mathrm{OP}}) \end{bmatrix} \tag{A.80}$$

Thus (A.79, A.80) can be used to convert spatial acceleration of a rigid body to the classical acceleration of a know point on the rigid body, if the *twist* related to its motion is known.

A.6.2.2 Classical to spatial acceleration

The classical acceleration of a point attached to the rigid body and instantaneously coincident with the origin of the reference frame is derived first from the classical acceleration of a point P attached to the rigid body.

$$\hat{\boldsymbol{a}}_{c_o} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \boldsymbol{a}_{c_o} \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \boldsymbol{a}_{c_P} - \dot{\boldsymbol{\omega}} \times \overrightarrow{\mathrm{OP}} - \boldsymbol{\omega} \times (\boldsymbol{\omega} \times \overrightarrow{\mathrm{OP}}) \end{bmatrix} \tag{A.81}$$

Then inverting (A.79), the spatial acceleration can be obtained as:

$$\hat{\boldsymbol{a}} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \dot{\boldsymbol{v}}_o \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \dot{\boldsymbol{v}}_o + \boldsymbol{\omega} \times \boldsymbol{v}_o \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \\ \boldsymbol{a}_{c_o} - \boldsymbol{\omega} \times \boldsymbol{v}_o \end{bmatrix} \tag{A.82}$$

References

[1] Stefan Zeiß. *Manipulation Skill for Robotic Assembly.* Master's thesis, 2014.

[2] Jörg Stückler and Sven Behnke. Following human guidance to cooperatively carry a large object. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 218–223. IEEE, 2011.

[3] Francisco Suárez-Ruiz, Xian Zhou, and Quang-Cuong Pham. Can robots assemble an ikea chair? *Science Robotics*, 3(17):eaat6385, 2018.

[4] Diogo Almeida and Yiannis Karayiannidis. Folding assembly by means of dual-arm robotic manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3987–3993. IEEE, 2016. 00000.

[5] Ching-Yen Weng, Wei Chian Tan, and I-Ming Chen. A survey of dual-arm robotic issues on assembly tasks. In *ROMANSY 22–Robot Design, Dynamics and Control*, pages 474–480. Springer, 2019.

[6] Jan Stria, Daniel Prusa, Vaclav Hlavac, Libor Wagner, Vladimir Petrik, Pavel Krsek, and Vladimir Smutny. Garment perception and its folding using a dual-arm robot. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, sep 2014.

[7] Li Sun, Gerardo Aragon-Camarasa, Simon Rogers, and J Paul Siebert. Accurate garment surface analysis using an active stereo robot head with application to dual-arm flattening, 2015.

[8] Kimitoshi Yamazaki, Ryosuke Oya, Kotaro Nagahama, Kei Okada, and Masayuki Inaba. Bottom dressing by a dual-arm robot using a clothing state estimation based on dynamic shape changes. *International Journal of Advanced Robotic Systems*, 13(1):5, 2016.

[9] Tomoya Tamei, Takamitsu Matsubara, Akshara Rai, and Tomohiro Shibata. Reinforcement learning of clothing assistance with a dual-arm robot. In *Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on*, pages 733–738. IEEE, 2011.

[10] Hardware specifications - sdk-wiki. http://sdk.rethinkrobotics.com/wiki/Hardware_Specifications. (Accessed on 06/11/2019).

[11] Jing Xiao and Xuerong Ji. Automatic generation of high-level contact state space. *The International Journal of Robotics Research*, 20(7):584–606, 2001.

[12] Neil Dantam. Practical exponential coordinates using implicit dual quaternions, 12 2018.

[13] Bruno Vilhena Adorno. *Two-arm manipulation: From manipulators to enhanced human-robot collaboration.* Ph.D. thesis, Université Montpellier II-Sciences et Techniques du Languedoc, 2011.

[14] Erol Özgür and Youcef Mezouar. Kinematic modeling and control of a robot arm using unit dual quaternions. *Robotics and Autonomous Systems*, 77:66–73, 2016.

[15] Arne Wahrburg, Stefan Zeiss, Björn Matthias, Jan Peters, and Hao Ding. Combined pose-wrench and state machine representation for modeling robotic assembly skills. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 852–857. IEEE, 2015.

[16] Collaborative robot market forecast 2018 - interact analysis. https://www.interactanalysis.com/collaborative-robot-market-2018/. (Accessed on 05/31/2019).

[17] Danica Kragic, Joakim Gustafson, Hakan Karaoguz, Patric Jensfelt, and Robert Krug. Interactive, collaborative robots: Challenges and opportunities. In *IJCAI*, pages 18–25, 2018.

[18] Wen-Chung Chang and Van-Truong Nguyen. Control of cooperative dual-arm mobile robots in a vision-based intelligent space. In *Emerging Trends In Mobile Robotics*, pages 296–304. World Scientific, 2010.

[19] Jingguo Ge, Zhaofu Chi, and Qingwei Li. Small part assembly with dual arm robot and smart camera. In *ISR/Robotik 2014; 41st International Symposium on Robotics; Proceedings of*, pages 1–6. VDE, 2014.

[20] Siwen Fang, Xinlong Huang, Heping Chen, and Ning Xi. Dual-arm robot assembly system for 3c product based on vision guidance. In *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 807–812. IEEE, 2016.

[21] Xianghua Chu, Heidi Fleischer, Norbert Stoll, Michael Klos, and Kerstin Thurow. Application of dual-arm robot in biomedical analysis: Sample preparation and transport. In *Instrumentation and Measurement Technology Conference (I2MTC), 2015 IEEE International*, pages 500–504. IEEE, 2015.

[22] Heidi Fleischer, Robert Ralf Drews, Jessica Janson, Bharath Reddy Chinna Patlolla, Xianghua Chu, Michael Klos, and Kerstin Thurow. Application of a dual-arm robot in complex sample preparation and measurement processes. *Journal of laboratory automation*, 21(5):671–681, 2016.

[23] E Zereik, A Sorbara, G Casalino, and F Didot. Autonomous dual-arm mobile manipulator crew assistant for surface operations: force/vision-guided grasping. In *Recent Advances in Space Technologies, 2009. RAST'09. 4th International Conference on*, pages 710–715. IEEE, 2009.

[24] Michael Beetz, Ulrich Klank, Alexis Maldonado, Dejan Pangercic, and Thomas Rühr. Robotic roommates making pancakes - look into perception-manipulation loop. In *IEEE International Conference on Robotics and Automation (ICRA), Workshop on Mobile Manipulation: Integrating Perception and Manipulation*, pages 529–536, May, 9–13 2011.

[25] Toshiharu Mukai, Shinya Hirano, Hiromichi Nakashima, Yo Kato, Yuki Sakaida, Shijie Guo, and Shigeyuki Hosoe. Development of a nursing-care assistant robot riba that can lift a human in its arms. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5996–6001. IEEE, 2010.

[26] Christian Smith, Yiannis Karayiannidis, Lazaros Nalpantidis, Xavi Gratal, Peng Qi, Dimos V Dimarogonas, and Danica Kragic. Dual arm manipulationa survey. *Robotics and Autonomous systems*, 60(10):1340–1353, 2012.

[27] Sukhan Lee. Dual redundant arm configuration optimization with task-oriented dual arm manipulability. *IEEE Transactions on Robotics and Automation*, 5(1):78–97, 1989.

[28] Albert Albers, Sven Brudniok, Jens Ottnad, Christian Sauter, and Korkiat Sedchaicharn. Upper body of a new humanoid robot-the design of armar iii. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 308–313. IEEE, 2006.

[29] Joao Silvério, Leonel Rozo, Sylvain Calinon, and Darwin G Caldwell. Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 464–470. IEEE, 2015.

[30] Lucia Pais Ureche and Aude Billard. Constraints extraction from asymmetrical bimanual tasks and their use in coordinated behavior. *Robotics and autonomous systems*, 103:222–235, 2018.

[31] 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*, pages 350–356 vol.1. IEEE, April 1988. ISBN 978-0-8186-0852-0. 00216.

[32] Lei Yan, Zonggao Mu, Wenfu Xu, and Bingsong Yang. Coordinated compliance control of dual-arm robot for payload manipulation: Master-slave and shared force control. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 2697–2702. IEEE, 2016.

[33] Jian M Tao, JYS Luh, and Yuan F Zheng. Compliant coordination control of two moving industrial robots. *IEEE transactions on Robotics and Automation*, 6(3):322–330, 1990.

[34] Fuhai Zhang, Lei Hua, Yili Fu, and Bin Guo. Dynamic simulation and analysis for bolt and nut mating of dual arm robot. In *Robotics and Biomimetics (ROBIO), 2012 IEEE International Conference On*, pages 660–665. IEEE, 2012. 00001.

[35] H Andy Park and CS George Lee. Extended cooperative task space for manipulation tasks of humanoid robots. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 6088–6093. IEEE, 2015.

[36] Rodrigo S. Jamisola, Petar Kormushev, Darwin G. Caldwell, and Frank Ibikunle. Modular relative jacobian for dual-arms and the wrench transformation matrix. In *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 181–186. IEEE, 2015. 00003.

[37] Stanley A Schneider and Robert H Cannon. Object impedance control for cooperative manipulation: Theory and experimental results. *IEEE Transactions on Robotics and Automation*, 8(3):383–394, 1992.

[38] Meiling Wang, Minzhou Luo, Tao Li, and Marco Ceccarelli. A unified dynamic control method for a redundant dual arm robot. *Journal of Bionic Engineering*, 12(3):361–371, 2015.

[39] Yves Guiard. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *Journal of motor behavior*, 19(4):486–517, 1987.

[40] H Andy Park and CS George Lee. Dual-arm coordinated-motion task specification and performance evaluation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 929–936. IEEE, 2016.

[41] Pasquale Chiacchio and Stefano Chiaverini. Pd-type control schemes for cooperative manipulator systems. *Intelligent Automation & Soft Computing*, 2(1):65–72, 1996.

[42] F. Caccavale, P. Chiacchio, and S. Chiaverini. Task-space regulation of cooperative manipulators. *Automatica*, 36(6):879–887, June 2000. 00052.

[43] Jinoh Lee, Pyung Hun Chang, and Rodrigo S Jamisola. Relative impedance control for dual-arm robots performing asymmetric bimanual tasks. *IEEE transactions on industrial electronics*, 61(7):3786–3796, 2014.

[44] C.L. Lewis. Trajectory generation for two robots cooperating to perform a task. volume 2, pages 1626–1631. IEEE, 1996. ISBN 978-0-7803-2988-1. 00030.

[45] William S Owen, Elizabeth A Croft, and Beno Benhabib. Acceleration and torque redistribution for a dual-manipulator system. *IEEE transactions on robotics*, 21(6):1226–1230, 2005.

[46] Raul Guenther and Daniel Martins. Screw-based relative Jacobian for manipulators cooperating in a task. In *ABCM Symposium Series in Mechatronics*, volume 3, pages 276–285, 2008. 00016.

[47] Emre Sariyildiz and Hakan 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*, 20(4):607–628, 2012.

[48] CR Rocha, CP Tonetto, and A Dias. A comparison between the denavit–hartenberg and the screw-based methods used in kinematic modeling of robot manipulators. *Robotics and Computer-Integrated Manufacturing*, 27(4):723–728, 2011.

[49] Raul Guenther and Daniel Martins. Screw-based relative Jacobian for manipulators cooperating in a task USING ASSUR VIRTUAL CHAINS. In *ABCM Symposium Series in Mechatronics*, volume 3, pages 276–285, 2008. 00000.

[50] Xiangke Wang and Changbin Yu. Unit dual quaternion-based feedback linearization tracking problem for attitude and position dynamics. *Systems & Control Letters*, 62(3):225–233, 2013.

[51] Janez Funda and Richard P Paul. A computational analysis of screw transformations in robotics. *IEEE Transactions on Robotics and Automation*, 6(3):348–356, 1990.

[52] Nicholas A Aspragathos and John K Dimitros. A comparative study of three methods for robot kinematics. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(2):135–145, 1998.

[53] Xiangke Wang and Huayong Zhu. On the comparisons of unit dual quaternion and homogeneous transformation matrix. *Advances in Applied Clifford Algebras*, 24(1):213–229, 2014.

[54] Bailin Cao, Gordon I Dodds, and George W Irwin. Redundancy resolution and obstacle avoidance for cooperative industrial robots. *Journal of Robotic Systems*, 16(7):405–417, 1999.

[55] Rodrigo S. Jamisola and Rodney G. Roberts. A more compact expression of relative Jacobian based on individual manipulator Jacobians. *Robotics and Autonomous Systems*, 63:158–164, January 2015. 00006.

[56] Masaru Uchiyama and Pierre Dauchez. Symmetric kinematic formulation and non-master/slave coordinated control of two-arm robots. *Advanced Robotics*, 7(4):361–383, January 1992. 00052.

[57] Fabrizio Caccavale and Luigi Villani. An impedance control strategy for cooperative manipulation. In *Advanced Intelligent Mechatronics, 2001. Proceedings. 2001 IEEE/ASME International Conference On*, volume 1, pages 343–348. IEEE, 2001. 00014.

[58] F. Caccavale, P. Chiacchio, A. De Santis, A. Marino, L. Villani, F. Caccavale, and A. Marino. An experimental investigation on impedance control for dual-arm cooperative systems. In *2007 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 1–6, September 2007. 00002.

[59] F. Caccavale, P. Chiacchio, A. Marino, and L. Villani. Six-DOF Impedance Control of Dual-Arm Cooperative Manipulators. *IEEE/ASME Transactions on Mechatronics*, 13(5):576–586, October 2008. 00083.

[60] Bruno Vilhena Adorno, Philippe Fraisse, and Sébastien Druon. Dual position control strategies using the cooperative dual task-space framework. In *IROS'10: International Conference on Intelligent Robots and Systems*, pages 3955–3960. IEEE, 2010.

[61] LFC Figueredo, Bruno Vilhena Adorno, João Yoshiyuki Ishihara, and Geovany Araujo Borges. Robust kinematic control of manipulator robots using dual quaternion representation. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1949–1955. IEEE, 2013.

[62] MM Marinho, LFC Figueredo, and Bruno Vilhena Adorno. A dual quaternion linear-quadratic optimal controller for trajectory tracking. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 4047–4052. IEEE, 2015.

[63] Rethink Robotics. Baxter collaborative robots for industrial automation, 2017.

[64] Rethink Robotics. Baxter research robot. *Retrieved April*, 26:2018, 2013.

[65] Gill A Pratt and Matthew M Williamson. Series elastic actuators. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 1, pages 399–406. IEEE, 1995.

[66] Xianmin Zhang, Yanglong Zheng, Jun Ota, and Yanjiang Huang. Peg-in-hole assembly based on two-phase scheme and f/t sensor for dual-arm robot. *Sensors*, 17(9):2004, 2017.

[67] Daniel Kruse, Richard J Radke, and John T Wen. Collaborative human-robot manipulation of highly deformable materials. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 3782–3787. IEEE, 2015.

[68] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. Ros: an open-source robot operating system. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*. Kobe, Japan, May 2009.

[69] KDL wiki — The Orocos Project. http://www.orocos.org/kdl. (Accessed on 06/14/2019).

[70] Ruben Smits, H Bruyninckx, and E Aertbeliën. Kdl: Kinematics and dynamics library. *Avaliable: http://www. orocos. org/kdl*, 2011.

[71] Jinoh Lee, Pyung Hun Chang, and Rodrigo S. Jamisola. Relative Impedance Control for Dual-Arm Robots Performing Asymmetric Bimanual Tasks. *IEEE Transactions on Industrial Electronics*, 61(7):3786–3796, July 2014. 00026.

[72] Gert A Kragten, Frans CT Van der Helm, and Just L Herder. A planar geometric design approach for a large grasp range in underactuated hands. *Mechanism and Machine Theory*, 46(8):1121–1136, 2011.

[73] Arun K Natesan. Kinematic analysis and synthesis of four-bar mechanisms for straight line coupler curves, 1994.

[74] Ivan Godler, Kohtaroh Hashiguchi, and Takashi Sonoda. Robotic finger with coupled joints: A prototype and its inverse kinematics. In *Advanced Motion Control, 2010 11th IEEE International Workshop on*, pages 337–342. IEEE, 2010.

[75] AR10 Robotic Hand — Active8Robots. https://www.active8robots.com/robots/ar10-robotic-hand/. (Accessed on 05/28/2019).

[76] J Michael McCarthy. *Geometric design of linkages*, volume 11. Springer Science & Business Media, 2006.

[77] Godfried Toussaint. Simple proofs of a geometric property of four-bar linkages. *The American mathematical monthly*, 110(6):482–494, 2003.

[78] Haixia Wang, Shuhan Shen, and Xiao Lu. A screw axis identification method for serial robot calibration based on the poe model. *Industrial Robot: An International Journal*, 39(2):146–153, 2012.

[79] Dapeng Han, Qing Wei, Zexiang Li, and Weimeng Sun. Control of oriented mechanical systems: A method based on dual quaternion. *IFAC Proceedings Volumes*, 41(2):3836–3841, 2008.

[80] Da-Peng Han, Qing Wei, and Ze-Xiang Li. Kinematic control of free rigid bodies using dual quaternions. *International Journal of Automation and Computing*, 5(3):319–324, 2008.

[81] Karim Abdel-Malek, Jingzhou Yang, Denis Blackmore, and Ken Joy. Swept volumes: fundation, perspectives, and applications. *International Journal of Shape Modeling*, 12(01):87–127, 2006.

[82] Francesco Bullo and Richard M Murray. Proportional derivative (pd) control on the euclidean group. In *European Control Conference*, volume 2, pages 1091–1097, 1995.

[83] Xiangke Wang and Changbin Yu. Feedback linearization regulator with coupled attitude and translation dynamics based on unit dual quaternion. In *Intelligent Control (ISIC), 2010 IEEE International Symposium on*, pages 2380–2384. IEEE, 2010.

[84] Xiangke Wang, C Yu, et al. Unit-dual-quaternion-based pid control scheme for rigid-body transformation. In *Proceedings of 18th World Congress International Federation of Automatic Control*, pages 9296–9301, 2011.

[85] Jianying Wang and Zhaowei Sun. 6-dof robust adaptive terminal sliding mode control for spacecraft formation flying. *Acta Astronautica*, 73:76–87, 2012.

[86] Xiangke Wang, Dapeng Han, Changbin Yu, and Zhiqiang Zheng. The geometric structure of unit dual quaternion with application in kinematic control. *Journal of Mathematical Analysis and Applications*, 389(2):1352–1364, 2012.

[87] J Luh, M Walker, and R Paul. Resolved-acceleration control of mechanical manipulators. *IEEE Transactions on Automatic Control*, 25(3):468–474, 1980.

[88] CW Wampler and LJ Leifer. Applications of damped least-squares methods to resolved-rate and resolved-acceleration control of manipulators. *Journal of Dynamic Systems, Measurement, and Control*, 110(1):31–38, 1988.

[89] Fabrizio Caccavale, Stefano Chiaverini, and Bruno Siciliano. Second-order kinematic control of robot manipulators with jacobian damped least-squares inverse: Theory and experiments. *IEEE/ASME Transactions on Mechatronics*, 2(3):188–194, 1997.

[90] Fabrizio Caccavale, Ciro Natale, Bruno Siciliano, and Luigi Villani. Resolved-acceleration control of robot manipulators: A critical review with experiments. *Robotica*, 16(5):565–573, 1998.

[91] Bin Xian, Marcio S de Queiroz, D Dawson, and I Walker. Task-space tracking control of robot manipulators via quaternion feedback. *IEEE Transactions on Robotics and Automation*, 20(1):160–167, 2004.

[92] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[93] Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757, 2008.

[94] Fabio Vigoriti, Fabio Ruggiero, Vincenzo Lippiello, and Luigi Villani. Tracking control of redundant manipulators with singularity-free orientation representation and null-space compliant behaviour. In *Human Friendly Robotics*, pages 15–28. Springer, 2019.

[95] Roy Featherstone. A beginner's guide to 6-d vectors (part 2)[tutorial]. *IEEE robotics & automation magazine*, 17(4):88–99, 2010.

[96] Luigi Biagiotti and Claudio Melchiorri. *Trajectory planning for automatic machines and robots*. Springer Science & Business Media, 2008.

[97] F. Caccavale and L. Villani. Impedance control for multi-arm manipulation. In *Decision and Control, 2000. Proceedings of the 39th IEEE Conference On*, volume 4, pages 3465–3470. IEEE, 2000. 00007.

[98] Jean-Pierre Merlet. C-surface applied to the design of an hybrid force-position robot controller. In *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, volume 4, pages 1055–1059. IEEE, 1987.

[99] Herman Bruyninckx, Sabine Demey, Stefan Dutré, and Joris De Schutter. Kinematic models for model-based compliant motion in the presence of uncertainty. *The International journal of robotics research*, 14(5):465–482, 1995.

[100] Hiroaki Kobayashi. Control and geometrical considerations for an articulated robot hand. *The International Journal of Robotics Research*, 4(1):3–12, 1985.

[101] Ingo Kresse. *A semantic constraint-based Robot Motion Control for Generalizing Everyday Manipulation Actions*. Ph.D. thesis, Technische Universität München, 2017.

[102] Herman Bruyninckx, Tine Lefebvre, Lyudmila Mihaylova, Ernesto Staffetti, Joris De Schutter, and J Xiao. A roadmap for autonomous robotic assembly. In *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning (ISATP2001). Assembly and Disassembly in the Twenty-first Century.(Cat. No. 01TH8560)*, pages 49–54. IEEE, 2001.

[103] Jose Sanchez, Juan-Antonio Corrales, Belhassen-Chedli Bouzgarrou, and Youcef Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research*, page 0278364918779698, 2018.

[104] Takayuki Matsuno, Toshio Fukuda, and Fumihito Arai. Flexible rope manipulation by dual manipulator system using vision sensor. In *2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics. Proceedings (Cat. No. 01TH8556)*, volume 2, pages 677–682. IEEE, 2001.

[105] Yuji Yamakawa, Akio Namiki, and Masatoshi Ishikawa. Motion planning for dynamic knotting of a flexible rope with a high-speed robot arm. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 49–54. IEEE, 2010.

[106] William Rowan Hamilton. Xi. on quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 33(219):58–60, 1848.

[107] Clifford. Preliminary sketch of biquaternions. *Proceedings of the London Mathematical Society*, s1-4(1):381–395, nov 1871.

[108] Roy Featherstone. A beginner's guide to 6-d vectors (part 1). *IEEE robotics & automation magazine*, 17(3):83–94, 2010.

[109] Roy Featherstone. The acceleration vector of a rigid body. *The International Journal of Robotics Research*, 20(11):841–846, 2001.