Coordinate Invariant User-Guided Constrained Path Planning with Reactive Rapidly Expanding Plane-Oriented Escaping Trees

Riddhiman Laha^{*,1}, Ruiai Sun^{*,1}, Wenxi Wu^{*,1}, Dasharadhan Mahalingam², Nilanjan Chakraborty², Luis F.C. Figueredo^{*,1}, and Sami Haddadin¹

Abstract-As collaborative robots move closer to human environments, motion generation and reactive planning strategies that allow for elaborate task execution with minimal easy-to-implement guidance whilst coping with changes in the environment is of paramount importance. In this paper, we present a novel approach for generating real-time motion plans for point-to-point tasks using a single successful human demonstration. Our approach is based on screw linear interpolation, which allows us to respect the underlying geometric constraints that characterize the task and are implicitly present in the demonstration. We also integrate an original reactive collision avoidance approach with our planner. We present extensive experimental results to demonstrate that with our approach, by using a single demonstration of moving one block, we can generate motion plans for complex tasks like stacking multiple blocks (in a dynamic environment). Analogous generalization abilities are also shown for tasks like pouring and loading shelves. For the pouring task, we also show that a demonstration given for one-armed pouring can be used for planning pouring with a dual-armed manipulator of different kinematic structure.

I. INTRODUCTION

We are interested in the problem of a robot manipulating objects in highly constrained, unstructured, possibly cluttered and dynamic environments, in a real-time fashion, and the transfer of the task expertise and constraints to different robotic systems. Take for instance, the shelving task in Fig. 1 where the robot needs to generate a sequence of plans to grasp books from a user and place them in different available spaces in the cupboard while avoiding collision with the cluttered cupboard and dynamic objects in the scene. Designing plans for elaborate tasks such as this requires experienced roboticists to conceptualize and preprogram the robot [1], [2]. It is even more time consuming and limiting when we want to integrate safety aspects, make changes in the environment, structure and/or transfer the task(s) to different robots as seen Fig. 1.

A convenient solution is to teach the robot from human demonstrations. However, existing solutions require multiple demonstrations—which can be cumbersome in cluttered and constrained scenarios—and often have limited capability to generalize to different structure and robots. Furthermore, practical applicability in industry and service robotics, calls for the development of a framework which requires minimal time from demonstration (preferably single) to deployment on the robot [3]. In this work, we propose a reactive planner guided by a single-demonstration that enables the robot to solve such elaborate tasks and that can be straightforwardly

*All authors contributed equally

¹The authors are with Munich Institute of Robotics & Machine Intelligence, Technische Universität München (TUM), Germany. This work was funded by the Lighthouse Initiative Geriatronics by StMWi Bayern (Project X, grant 5140951), LongLeif GaPa gGmbH (Project Y, grant 5140953), "Centre for Tactile Internet with Human-in-the-Loop" (CeTI, grant 390696704) and KLFABRIK Bayern (grant DIK0249). S. Haddadin has a potential conflict of interest as shareholder of Franka Emika GmbH.

²The authors are with the Department of Mechanical Engineering at Stony Brook University (SBU), Stony Brook, USA. The work was partially funded by the grant NSF CMMI 1853454, and a SBU OVPR seed grant.



Fig. 1. Overview of our framework with a shelving task with a singledemonstration and executed in clutter with reactive response to obstacles (top figures), and the generalization of the pouring task towards different placements with collision avoidance and task transfer from a single-arm Panda to a dual-arm Baxter.

transferred to even multiple-arm systems.

These are the key problems that our planning framework solves: First, our planner produces paths through screw linear interpolation (ScLERP) that implicitly maps-and therefore satisfies-all geometric constraints embedded in the singledemonstration. For instance, while pouring a glass of water, the orientation transformations related to the angle deflection along the axis of rotation is coupled with the specific translation position. Our planner explores this information allowing autonomous transfer to new conditions, scenes, new initial or final pose, and even to a completely different robotic system. Note, the user requires no knowledge about the constraints as they are embedded in the topology. Second, our planner shifts interpolated points that can lead to collision through a novel reactive approach ensuring feasibility of the motion. Finally, our planner provides motion generation in real-time with an additional safety layer for guaranteed collision avoidance within the low-level controller.

To the best of our knowledge, this is the first work to integrate this level of reactiveness, collision avoidance and generalization in real-time from a single user demonstration. Furthermore, we demonstrate through real robot experiments the efficiency of our approach and that the coordinate invariant motions generated by our algorithm can be seamlessly transferred to multiple robots with different kinematic structure. *A. Related Work*

The basic approach to make robots autonomous for a particular task is hand-engineering of a controller. Yet the generalization of this task across the workspace (different instances) is extremely hard [1]. A well known strategy, in this regard, is learning from demonstrations (LfD) [4], [5] which is often acquired through teleoperation, kinesthetic or passive observation. A convenient way to encode the

trajectory information is by means of motion primitives, e.g., DMPs [6], [7]. However, adapting DMPs to reach viapoints is not straightforward [8] and they need multiple demonstrations to encode the mean trajectory which can be challenging for rotation. Taking a probabilistic approach such as ProMPs improves generalization [8], yet their convergence is still only guaranteed within the demonstration region. Furthermore, often constraints are embedded in joint-space [9], [10], which limits applicability to variations of the joint to task space mapping, e.g., single changes in the kinematics to changing the robot. Moreover, most of existing approaches have limited generalization capability as well as reactiveness in the presence of dynamic obstacles, and they often require a reasonable amount of demonstrations, which is cumbersome in highly constrained scenarios. To avoid multiple demonstrations, recent focus has been given to oneshot learning or learning from a single example [11]–[13]. Nevertheless, studies in this area often assume the existence of an optimization criterion that defines the tasks [14] or make strong distributional assumptions about the means of task motion generation [13]. Also, previous works in animation and robotics have incorporated user inputs in the configuration space [15]-[19] where handling obstacles in a reactive fashion is not usually the case [20]. Our method on the other hand introduces user kinesthetic guidance in task-space planning.

Overall, we take a fundamentally different approach. We explore the one-time kinesthetic demonstration not to learn the trajectory, imitate the human motion, learn a cost function or a policy, but rather to encode and follow the implicit geometric features underlying the path. Our approach is hinged on the observation that from a given set $\mathfrak{S} \subseteq \text{Spin}(3) \ltimes \mathbb{R}^3$, the set of all rigid body motions, proper subsets can be drawn from geometric structures of interest which are then implicitly captured by the human demonstration and preserved by ScLERP. In other words, we differ from the literature in the sense that our algorithms embed the constraints in the topology of the rigid-body transformations rather than learning the region of attraction of the dynamical system approximation [6], [21]. In this work, we also exemplify the use of one single demonstration to different instances of tasks, constraints, topologies, and even kinematic chains (such as single-arm to multi-arm system). Also, the resulting deterministic path is safe in the sense that it always mimics part of the demonstration thereby producing paths which are intuitive and predictable for human co-workers.

II. PROBLEM FORMULATION

This section presents the definitions and fundamentals of the planning problem. We first briefly recap core concepts regarding the algebra of dual quaternions (DQ) and geometric first-order interpolation properties using screw linear theory. These are the backbone of the proposed approach and rely on unit dual quaternion algebraic and geometric properties. Computational advantages include, e.g., Riemannian geometry, translation and orientation coupling [22]–[25], singularityfree representation, with global convergence controllers [26], [27], that embeds wrenches, twists and primitives, being the universal cover of SE(3) with a simply-connected topology (in contrast to SE(3)), and higher efficiency [28]–[31].

A. Problem and Mathematical Background

Our aim is to explore, adapt or define a path, which consists of a sequence of rigid-body transformations, to complete a geometric constrained task—a motion planning problem. To connect the spatial transformations and generate a smooth curve in SE(3), we must (i) define the rigid body transformation in the Lie group of Spin(3) $\ltimes \mathbb{R}^3$, (ii) describe its Riemannian geometry and geodesics, and finally (iii) define the screw linear interpolation based on the geodesic direction.

An arbitrary **rigid body transformation** can be represented by the unit dual quaternion $\underline{x} \in \text{Spin}(3) \ltimes \mathbb{R}^3$,

$$\underline{\boldsymbol{x}} = \boldsymbol{r} + \frac{1}{2}\varepsilon \boldsymbol{p}\boldsymbol{r},\tag{1}$$

where $\mathbf{r} = \cos(\phi/2) + \sin(\phi/2)\mathbf{n}$ represents a rotation with angle ϕ around the axis \mathbf{n} , in unit quaternions Spin(3) [32], \mathbf{p} is a pure quaternion that represents the translation, and ε is such that $\varepsilon \neq 0$ but $\varepsilon^2 = 0$, [33]. Unit dual-quaternion, Spin(3) $\ltimes \mathbb{R}^3$, is a Lie group with inverse element being $\underline{\mathbf{x}}^* = \mathbf{r}^* + \frac{1}{2}\varepsilon \mathbf{r}^*\mathbf{p}^*$, and identity $\underline{\mathbf{1}}$. Dual quaternion elements can also be described by $\underline{\mathbf{x}} = \mathcal{P}(\underline{\mathbf{x}}) + \varepsilon \mathcal{D}(\underline{\mathbf{x}})$, where $\mathcal{P}(\underline{\mathbf{x}})$ and $\mathcal{D}(\underline{\mathbf{x}})$ are the primary and dual components.

From its differentiable **Riemannian geometry**, it is endowed with a collection of inner products on the tangent space at Spin(3) $\ltimes \mathbb{R}^3$, which in turn builds a Riemannian metric [34], [35]. Once a Riemannian metric is assigned to the manifold—e.g., the length of the path [36]—we explore the minimum curve length, i.e., the geodesics [35], see [34], [36]–[38]. In such manifolds, actions in the geodesics can be expressed by means of the exponential map $\exp_{\underline{x}}$: $\mathcal{T}_{\underline{x}}$ Spin(3) $\ltimes \mathbb{R}^3 \to \text{Spin}(3) \ltimes \mathbb{R}^3$. The $\exp_{\underline{x}}$ locally maps a vector in the tangent space $\mathcal{T}_{\underline{x}}$ Spin(3) $\ltimes \mathbb{R}^3$ (at $\underline{x} \in$ Spin(3) $\ltimes \mathbb{R}^3$) to a point on the manifold following the geodesic through \underline{x} , [39], [40]. The inverse mapping (from manifold to tangent space at the point \underline{x}) is the logarithm map $\log_{\underline{x}}$: Spin(3) $\ltimes \mathbb{R}^3 \to \mathcal{T}_{\underline{x}}$ Spin(3) $\ltimes \mathbb{R}^3$.

map $\log_{\underline{x}} : \operatorname{Spin}(3) \ltimes \mathbb{R}^3 \to \mathcal{T}_{\underline{x}} \operatorname{Spin}(3) \ltimes \mathbb{R}^3$. The mappings $\exp_{\underline{x}}$ and $\log_{\underline{x}}$ are non-trivial to obtain. A solution is to compute them by parallel transport [35], [39], [41]. The parallel transport exploits the exponential function that maps vectors from the tangent space (at the identity) to the manifold [37],

$$\exp_{\underline{x}}(\underline{y}) = \underline{x} \exp(\underline{x}^* \underline{y}), \\ \log_{\underline{x}}(\underline{z}) = \underline{x} \log(\underline{x}^* \underline{z}),$$
(2)

where $\underline{z} \in \text{Spin}(3) \ltimes \mathbb{R}^3$ and \underline{y} is defined in the tangent space at \underline{x} —notice that \underline{y} is not an unit DQ. The exp and log maps from the tangent space, at the identity, i.e., $\mathcal{T}_{\underline{1}}\text{Spin}(3) \ltimes \mathbb{R}^3$ are given by the dual vector representing the axis of screw motion and the dual angle containing both the translation length and the angle of rotation, see further details in [35], [40], [42], [43].

Finally, to describe the screw linear interpolation that connects two points \underline{x}_1 and \underline{x}_2 , and to find points in the path $\underline{x}(\tau) : [0,1] \to \text{Spin}(3) \ltimes \mathbb{R}^3$ with $\underline{x}(0) = \underline{x}_1$ and $\underline{x}(1) = \underline{x}_2$, we exploit (2). First, we map \underline{x}_2 following the geodesic on Spin(3) $\ltimes \mathbb{R}^3$ through \underline{x}_1 onto the tangent space at \underline{x}_1 . Naturally, this mapping yields a vector in the $\mathcal{T}_{\underline{x}_1}\text{Spin}(3) \ltimes \mathbb{R}^3$ corresponding to the geodesic direction of \underline{x}_2 with respect to \underline{x}_1 . Hence,

$$\log_{\boldsymbol{x}_1}(\boldsymbol{x}_2) = \boldsymbol{x}_1 \log(\boldsymbol{x}_1^* \boldsymbol{x}_2), \qquad (3)$$

where $\log_{\underline{x}_1}$ is computed using parallel transport. Notice that (3) is defined in the tangent space of a Riemannian manifold, hence it is a vector space with basis defined by a vector field. Thus, we can linearly interpolate points and compute any point between along the geodesic direction starting from $\log_{\underline{x}_1}(\underline{x}_1)$ towards $\log_{\underline{x}_1}(\underline{x}_2)$, as $(\log_{\underline{x}_1}(\underline{x}_2) - \log_{\underline{x}_1}(\underline{x}_1))\tau + \log_{\underline{x}_1}(\underline{x}_1)$. Note, however that $\log_{\underline{x}_1}(\underline{x}_1) = 0$. Hence, using parallel transport to map the vector in $\mathcal{T}_{\underline{x}_1}$ Spin(3) $\ltimes \mathbb{R}^3$ back to the

unit DQ manifold following the geodesics through \underline{x}_1 yields $\underline{x}(\tau) = \exp_{x_1} (\underline{x}_1 \log(\underline{x}_1^* \underline{x}_2) \tau)$

$$\underline{\boldsymbol{x}}(\tau) = \exp_{\underline{\boldsymbol{x}}_1} \left(\underline{\boldsymbol{x}}_1 \log(\underline{\boldsymbol{x}}_1^* \underline{\boldsymbol{x}}_2) \tau \right) \\ = \underline{\boldsymbol{x}}_1 \exp\left(\log(\underline{\boldsymbol{x}}_1^* \underline{\boldsymbol{x}}_2) \tau \right).$$
(4)

Lemma 1 (ScLERP – [44]): Given two rigid body transformations, \underline{x}_1 and \underline{x}_2 , the ScLERP function,

ScLERP
$$(\tau, \underline{x}_1, \underline{x}_2) = \underline{x}_1 (\underline{x}_1^* \underline{x}_2)^{\tau},$$
 (5)

returns any rigid pose transformation along the geodesic direction from \underline{x}_1 to \underline{x}_2 scaled linearly along $\tau \in [0, 1]$. Taking equally spaced values within τ yields, therefore, a screw linear interpolation from \underline{x}_1 to \underline{x}_2 .

Notice the ScLERP function (5) is the same as the one derived in (4). This can be shown by geometrical exponential [22], [45], and from the scaling of the dual rotation angle about the screw axis—hence the name [44].

Remark 1: The ScLERP interpolation explores the natural parametrization of screw coordinates in terms of 6-DoF displacements [42], [46]. They are particular attractive for coordinate-invariant interpolation which is not possible when decoupling orientation and translation [47], as detailed in [46]. Similar interpolation scheme nonetheless could also be derived from SE(3), and other covering groups that satisfy left-invariance and are based on non-minimal representation of rigid displacements. Hence, it is by no means restricted to the choice of $\text{Spin}(3) \ltimes \mathbb{R}^3$. Still, a matrix-based solution is non-attractive due to the additional computational cost-that can possibly restrict real-time implementation—and due to the efficiency, compactness and intuitiveness of $\text{Spin}(3) \ltimes \mathbb{R}^3$ which can depict wrenches, twists, geometric primitives, constraints and its tangent space with the same algebra. B. Overview of Problem

In this work, we are interested in the motion planning problem that completes a geometric constrained task reactively in a real-time fashion, while responding to unforeseen events, such as dynamic obstacles with guaranteed avoidance behaviour—also in real-time.

The proposed motion generation scheme takes as prior knowledge a single successful task demonstration—defined by a sequence of poses expressed as unit dual quaternion

$$\mathcal{DP} = \{\underline{d}_1, \underline{d}_2, \cdots, \underline{d}_n\}, \ \underline{d}_i \in \text{Spin}(3) \ltimes \mathbb{R}^3$$
(6)

Since the task demonstration is a successful one, it satisfies the task-constraints, and this knowledge is implicitly embedded in the demonstration. As an example, during pouring, the axis and angle of rotation is implicitly present in the demonstration. We can obtain the demonstration, \mathcal{DP} , either by directly sensing the end-effector pose or by recording the joint-space path from the joint encoders and using the forward kinematics map for the manipulator. Kinesthetic teaching along with joint space path recording is used in this paper. A key point here is that no other information, or knowledge, about the task or its constraints are required or needs to be provided.

Given such implicit constraints, our planner aims at finding a sequence of rigid body transformations—herein named the final path and the corresponding robot joint-space actions that takes the manipulator end-effector from an initial configuration \underline{x}_0 to a final goal \underline{x}_f while satisfying the constraints observed in DP. Formally, our problem can be defined as follows:

Problem Definition: Given a single user-demonstrated path \mathcal{DP} , and the new initial and final pose \underline{x}_0 and \underline{x}_f , respectively, find a path from \underline{x}_0 to \underline{x}_f such that

- 1) All constraints that are implicit in DP are satisfied;
- 2) Obstacles that prevent motion feasibility are avoided;

 Motion generation is achieved in real-time with an additional safety-layer for low-level controller guaranteeing collision avoidance.

III. TASK SPACE IMITATION ALGORITHM (TSIA)

In this section, we present a solution to satisfy the first condition in the problem definition.

Problem Statement 1: Compute a path in $Spin(3) \ltimes \mathbb{R}^3$ starting at an initial pose \underline{x}_0 and ending at a goal pose \underline{x}_f while still maintaining the task relevant motion constraints implicit in the user demonstration.

From the human demonstration we obtain a series of unit dual quaternions that encode the rigid body constraints along the path, i.e., \mathcal{DP} as in (6), where \underline{d}_1 corresponds to the starting pose of the demonstration and \underline{d}_n , the final pose.

starting pose of the demonstration and \underline{d}_n , the final pose. Let $\underline{d}'_n (= \underline{x}_f)$ be the new goal pose. To ensure that the relative transformations between the poses in the demonstrated path are preserved on the path to the new goal we want to reach, we replicate the demonstrated motion with respect to the new goal and call this path the *imitated path* (\mathcal{IP}). The \mathcal{IP} reflects the constraints present in the demonstration.

To compute \mathcal{IP} , we first obtain the transformation, $\underline{\delta}_i$, between the last pose on the demonstrated motion $(\underline{d}_n \in \mathcal{DP})$ and every other pose on the demonstrated motion, i.e.,

$$\underline{\delta}_i = \underline{d}_{i-1}^* \underline{d}_n, \ i = 2, \dots, n.$$
(7)

Then, the imitated path $\mathcal{IP} = \{\underline{d}'_1, \underline{d}'_2, \cdots, \underline{d}'_n\}$ can be calculated using,

$$\underline{d'}_{i-1} = \underline{d'}_n \underline{\delta}_i^*, \ i = 2, \dots, n.$$
(8)

Final Path: The calculation of the final path $\mathcal{FP} = \{\underline{d}''_1, \underline{d}''_2, \dots, \underline{d}'_n\}$ from a new initial pose $\underline{d}''_1 (= \underline{x}_0)$ to the new goal pose \underline{d}'_n is performed using ScLERP [48]. Intuitively speaking, our goal is to find a path that blends into the \mathcal{IP} and after blending just follows the \mathcal{IP} to obtain the same geometric constraints during motion as in the \mathcal{DP} .

Let \underline{d}'_i be a guiding pose (in this regard there are various possibilities to explore depending on the scenario) on the \mathcal{IP} and \underline{d}_c be the current task space pose of the manipulator—starting from \underline{x}_0 . We now compute a target pose \underline{d}_t using ScLERP in dual quaternion space, that is,

$$\underline{d}_t(\tau) = \underline{d}_c \cdot \left(\underline{d}_c^* \cdot \underline{d}_i'\right)^{\tau}, \qquad (9)$$

where $\tau \in [0, 1]$ is the time primitive. Different choices of the parameter τ give different target poses in task space. Note that $\tau = 0$ corresponds to \underline{d}_c and $\tau = 1$ corresponds to \underline{d}'_i . We then select an interpolation pose¹ and use that pose as a reference (\underline{x}_d) for our kinematic controller. With the resulting joint velocities \dot{q} —from the closed-loop—the robot takes a new configuration closer to \underline{x}_d which is thereafter set as the new current pose. The process continues taking \underline{d}'_{i+1} as the new guiding pose on \mathcal{IP} . This follows until the goal is reached (we define the error $e(\underline{d}_c, \underline{d}'_n) =$ $\|\operatorname{vec}(\underline{d}_c), \operatorname{vec}(\underline{d}'_n)\| >$ a tolerance). Building our user-guided motion generation approach based on ScLERP is crucial in order to ensure constraint satisfaction during blending, i.e., during approach through interpolation to the desired imitated path. Since ScLERP is based on exponential and logarithmic maps (see (2)), it is clear that translation and orientation of interpolated poses between key-points remain bounded by both boundary pose. Hence, if both \underline{d}'_i and \underline{d}'_{i+1} satisfy a constraint, then the interpolated motion also concurs. Note that any projected subset can also make use of the exponential and logarithm mapping and constraints would

 1As an heuristic, we suggest deploying guiding poses within 20% to 70% of the ${\cal IP}$ length.

Algorithm 1 Reactive Imitation Algorithm		
1:	procedure Get Final Path: $(\underline{d}'_n, \mathcal{DP}, \mathcal{O})$	
2:	$\underline{\delta}_i \leftarrow \text{Compute spatial difference with } \mathcal{DP} (7);$	
3:	$\mathcal{IP} \leftarrow \text{Get imitated path with } (\underline{\delta}_i, \underline{d}'_n)$ (8);	
4:	while $e(\underline{d}_c, \underline{d}'_n)$ >tol and $i \leq n$ do	
5:	$\underline{d}_c \leftarrow \text{Get current pose};$	
6:	$o_c \leftarrow \text{Compute closest obstacle point } (\underline{d}_c, \mathcal{O});$	
7:	if inside detection shell (\underline{d}_c, o_c) then	
8:	$T \leftarrow \text{Escape Tree Generation } (\underline{d}_{\text{cnew}}, o_c);$	
9:	$\underline{d}_{c} \leftarrow \text{Set new current pose } (T);$	
10:	$\underline{d}'_i \leftarrow$ Set guiding pose;	
11:	$\underline{d}_{t} \leftarrow \text{ScLERP} (\underline{d}_{c}, \underline{d}'_{i});$	
12:	$\dot{q} \leftarrow \text{Compute reactive control action } (\underline{d}_t) (12);$	
13:	$i \leftarrow i + 1;$	

also be satisfied. Hence, it is a property intrinsic of our method to implicitly satisfy all geometric constrains defined in the task demonstration without having to explicitly define it—which would require a roboticist with good expertise and geometric reasoning to design.

Finally, note that such constraints cannot be guaranteed without a proper screw linear interpolation. A simple example is the usual approach based on decoupling translation and attitude. Even when using proper attitude spherical interpolation, different key points in a rigid body leads to different unpredicted trajectories, see details in [46].

Remark 2: Note that we are not learning the trajectory itself, but rather we are automatically and implicitly transferring the rigid body displacements which naturally embeds the desired task constraints. This facilitates generalization to different initial and end-goals, topologies, slightly different tasks, and even to different kinematic-chains, such as from single-arm to dual-arm—as seen in Section VI.

IV. REACTIVE OBSTACLE AVOIDANCE

During the real-time motion generation, unforeseen events such as the presence of obstacles may make the path planned by TSIA infeasible. We now present a reactive approach for real-time obstacle avoidance that we use to modify the path generated by TSIA and hence ensure task success. More formally, we present a solution to the following problem:

Problem Statement 2: Compute a collision-free path in $Spin(3) \ltimes \mathbb{R}^3$ while still respecting the constraints embedded in the path in the last step.

For didactic reasons, let us consider a *sphere of influence* around an obstacle with a radius larger than the obstacle. The robot path is deflected only when it is inside this detection shell. To enable integrated real-time collision-free path during planning we introduce the Rapidly Expanding Plane-oriented Escaping Trees (REPET) algorithm.

Escape Tree Generation: As the robot moves along the real-time generated path from the current pose \underline{x}_c towards the detection shell of the closest obstacle, centred at O_{obs} , we start devising the avoidance scheme. First, we obtain the normal vector $\eta(\mathfrak{T}(\underline{x}_c), O_{obs})$, given by $\eta : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3$ based on the current position $\mathfrak{T}(\underline{x}_c) : \operatorname{Spin}(3) \ltimes \mathbb{R}^3 \to \mathbb{R}^3$ and the closest point to the surface of the obstacle described by the shell centre. Then, a tangent plane p_t is computed using the normal information, which is thereafter used to build a vector v orthogonal to η (e.g., using unit vector \hat{i}),

$$\boldsymbol{v} = \hat{i} \times \frac{\boldsymbol{\eta}}{\|\boldsymbol{\eta}\|} \cdot \left(\frac{k_{\eta}}{2}\right),\tag{10}$$

where k_{η} denotes the length of the diagonal of the plane. In the same fashion, we define $\boldsymbol{u} = \mathcal{P}(\boldsymbol{p}_{t}) \times \boldsymbol{v}$ which is used



Fig. 2. Fast Obstacle avoidance using REPET showing all the tangent planes. This collision avoidance path is generated for the stacking problem. along with v to obtain points on sides and edges of the plane.

For efficient collision avoidance, we seek to explore poses along the normal vector in the tangent space of the surface ensuring guaranteed collision free points surrounding convex obstacles or good exploration along non-convex ones. Herein, for brevity, we are focusing on convex ones. The core idea is to integrate such key avoidance points into our motion generation scheme, which is based on a sequence of transformations interpolated through ScLERP. Hence, the key points shift² a sequence of transformations within the \mathcal{FP} building a collision-free path along the tangent space of the closest obstacle's detection shell.

As shown in Alg. 2, we first set a root (based on current pose \underline{x}_c) in the tangent plane, and keep exploring vectors along the orthogonal plane (10) with distance κ_{η} from the shell intersection point for fast building of subsequent planes. The length of κ_{η} defines the avoidance strategy. Large κ_{η} ensures good avoidance but larger deviations from the path, while smaller κ_{η} provide smoother motions yet may not be enough to avoid within one single step. Next, we use the rapid plane generation scheme to grow multiple planes iteratively, thus generating numerous paths exploring different avoidance possibilities which we check for end-effector collisions at each step. This leads to a tree that grows along the new plane-key-points generation, which we call the *escape tree*.

The tree is expanded until either a limited level is reached (stop criterion) or one leaf finds a free Cartesian path towards a free pose within the final path—located after the obstacle shell (exception being if the final goal is within). If we have multiple free path leafs in the breadth, we select the optimal one $\boldsymbol{\varrho}$ in terms of a heuristic cost function. Herein, we are taking the points closer to the goal. If the limited level is reached, we sample random new points in the plane with larger k_{η} . The resulting path follows the tree with our TSIA path generation scheme with new key avoidance poses. An example of REPET is shown in Fig. 2 (Alg. 2) for the real-robot experiment in Section VI.A. Alg. 1 illustrates TSIA combined with the reactive collision avoidance.

V. DQ BASED KINEMATIC CONTROLLER

As stressed in the problem definition, the motion generation needs to be achieved in real-time with an additional layer for low-level controller that guarantees collision avoidance. Essentially, the idea is to have a feedback controller with exponential convergence to a desired reference pose in

²The deflection along the path is deployed in Cartesian coordinates only as we want to disturb the least the constraints. Still, the integrated path remains connected through ScLERP ensuring the remaining constraints are satisfied during avoidance.

Algorithm 2 REPET Algorithm

1:	procedure BUILD PLANE:(origin \underline{x}_{c} , k_{η} , obstacle o)
2:	$\eta \leftarrow \text{Get normal vector } (\underline{x}_{c}, o);$
3:	$\boldsymbol{v} \leftarrow \text{Orthogonal vector defining plane } (\boldsymbol{\eta}, k_{\eta}) \ (10);$
4:	procedure ESCAPE TREE GENERATION: $(\underline{x}_c, \underline{d}'_n, o_c)$
5:	while stopping criteria is not satisfied do
6:	$\underline{x}_{r} \leftarrow \text{Set root } (\underline{x}_{c});$
7:	for Each leaf j of the decision tree level; do
8:	$\boldsymbol{p} \leftarrow \text{Build plane } (\boldsymbol{x}_{r}, k_{\eta}, \boldsymbol{o}_{c});$
9:	$\underline{\vec{c}}_{i} \leftarrow$ Get child from root;
10:	$\underline{cc}_{i} \leftarrow \text{Check for collisions;}$
11:	$\underline{c}_{j}^{*} \leftarrow$ Select best based on criteria ($\underline{cc}_{j}, \underline{d}'_{n}$);
12:	$\boldsymbol{\varrho}, \boldsymbol{\underline{d}}_o \leftarrow \text{Select path, pose } (\boldsymbol{\underline{c}}_{j}^*);$
-	-

Spin(3) $\ltimes \mathbb{R}^3$ without decoupling the translational and rotational components. First, we can define the spatial difference \underline{x}_e as, $\underline{x}_e = \underline{x}_m^* \underline{x}_d$, where \underline{x}_m denotes the measured current pose and \underline{x}_d the desired pose of the end-effector. The error metric \underline{e} can then be defined as $\underline{e} = 1 - \underline{x}_e$. This can be rewritten and mapped back into \mathbb{R}^8 using the Hamilton

operator H (as in [22], [23], [49]) and differentiated as,

$$\operatorname{vec} \underline{\dot{\boldsymbol{e}}} = \boldsymbol{H} \left(\underline{\boldsymbol{x}}_d \right) \boldsymbol{C}_8 \operatorname{vec} \underline{\dot{\boldsymbol{x}}}_m, \qquad (11)$$

where C_8 is a diagonal matrix and vec : $\mathcal{H} \to \mathbb{R}^8$ as defined in [22], [50]. Now, replacing $\underline{\dot{x}}_m$ with the robot Jacobian mapping leads and pseudo-inverse controller leads to

$$\dot{\boldsymbol{q}} = -(\boldsymbol{H}(\underline{\boldsymbol{x}}_d) \boldsymbol{C}_8 \boldsymbol{J})^+ \operatorname{vec} \underline{\dot{\boldsymbol{e}}} = -\mathbf{N}^+ \lambda_e \operatorname{vec} \underline{\boldsymbol{e}}, \qquad (12)$$

where \dot{q} is the joint velocities, \mathbf{N}^+ is extended Jacobian pseudoinverse and λ_e is a positive gain. This controller shows coordinate left-invariance which matches our framework. Due to space limits, we will omit further details or proofs but they can be found in [50]. In this work, we also include additional nullspace tasks such as joint-limit avoidance.

For ensuring avoidance motion only along the constrained obstacle surface within the low-level controller, we continuously modify the desired pose—as it reaches the obstacle detection shell—by means of

$$\underline{\boldsymbol{x}}_{dobs} = \underline{\boldsymbol{x}}_{m} \exp\left(\log(\underline{\boldsymbol{x}}_{m}^{*}\underline{\boldsymbol{x}}_{d}) - \frac{1}{2}\mathfrak{T}\left(\underline{\boldsymbol{x}}_{m}^{*}\underline{\boldsymbol{x}}_{d}\right)\right) + \underline{\boldsymbol{v}}_{ee}\left(\boldsymbol{I} - \mathcal{P}(\boldsymbol{\eta}_{obs})\right) \operatorname{vec}\left(\frac{1}{2}\mathfrak{T}\left(\underline{\boldsymbol{x}}_{m}^{*}\underline{\boldsymbol{x}}_{d}\right)\right)$$
(13)

where $\mathfrak{T}(\underline{x}_m^* \underline{x}_d))$ is the translation element of the spatial difference, \underline{v}_{ee} represents the current linear velocity of the end effector and η_{obs} denotes the normal vector, viewed from the rigid body frame, of the nearest point on the obstacle surface. $\mathcal{P}(\eta_{obs})$ is the orthogonal projection for which η_{obs} belongs to its range space. The last term makes sure that no action will violate the constrained surface, that is, any action along the obstacle normal belongs to the nullspace of the solution. Notice that similar strategy has been used before for SE(3) collision avoidance [36], [51]. However, to the best of the authors' knowledge, this is the first work to extend it to dual quaternion algebra for real-time deployment.

VI. EXPERIMENTS AND ANALYSIS

In this section, we present a set of experiments to evaluate our proposed framework. We show that our reactive userguided motion generation scheme can successfully generalize tasks to different initial and final conditions while ensuring embedded constraints from demonstration are satisfied. Our



Fig. 3. Paths for the pouring task as seen in Fig. 1. Successful generalization for different initial and final conditions without obstacles (magenta curve) and reactive real-time deflection (blue).



Fig. 4. Demonstrated path (blue), final computed path without collision avoidance (magenta), collision free path (cyan) for the stacking task (Fig. 6). Note the integration of the pre-grasp path for placing the blocks even in the presence of obstacles.

planner is able to evade unforeseen obstacles³ in real-time while satisfying task constraints. Finally, task demonstrations given on one manipulator (Franka Emika Panda)⁴ can be transferred to generate motion plans for a different robot, Baxter,⁵ with distinct kinematic structure. Demonstrations given for a one-armed task can also be used to generate a plan for the same task done in a bimanual fashion.

Experiments were performed with the Panda arm, in the first three tasks: stacking, pouring, and shelving books. For showing generalization across different types of robot arms, we used the pouring demonstration on the Panda arm to generate dual-armed pouring motion plan with the Baxter. Note that for all experiments, a single demonstration was used, without any need for adjustments or corrections.

A. Analysis and generalization under different conditions

First, to better understand and validate the proposed framework, we performed two different tasks:⁶ (i) Pouring water from one glass to another with shifting positions of the glasses; (ii) Stacking rectangular wooden cuboids. Both tasks were demonstrated only once, and executed ($\tau = 0.01$) under different conditions as shown in Figs. 1 and 6.

For pouring, the challenge was to embed the angle transformations around a specific axis of rotation—while executing the task in a completely different location with obstacles in the scene. Manually designing the task would

- ⁴Experiments executed at the Technical University of Munich.
- ⁵Experiments conducted at Stony Brook University.
- ^6The guiding pose was set to be 20% of \mathcal{IP} length.

³Herein, we assume full knowledge about obstacle poses, as detection is out of scope for this work which is agnostic to the detection strategy.



Fig. 5. Top view of the shelving task paths showing in Fig. 1. Deflection of the final computed path (blue, green) is crucial for this complex scenario. Note that the plan without collision avoidance (red) is inside the blue obstacle.



Fig. 6. Overview of our approach from single-demonstration (top-left) to generalization and sequential manipulation to reactive collision avoidance (bottom figures)

be challenging, yet our algorithm embeds such constraints in the demonstrated path which is ensured by the imitated path and deployed for the final path through ScLERP. All the trajectories are shown in Fig. 3, which depicts that our reactive framework ensures feasibility of the task while, at the same time, satisfying the demonstrated constraints.

The stacking task is a sequential manipulation problem, where a single demonstration is given but five sequence of motions, from grasping the first block to releasing the last bock needs to be executed. The only additional information is the different end-goals (the height of the blocks). During real-time execution, we included a dynamic obstacle in front of the third, and last, block. The demonstrated path and the executed path for this condition are shown in Fig. 4.

B. Reactive user-guided motion planning in clutter

To highlight the performance and robustness of our approach, we devised a cluttered scene where the objective was to shelve books handed by a human in a cluttered cupboard with a specific attitude transformation. In addition to static obstacles, the robot also avoids two unforeseen obstacles that appear during the placing of the second and the third book. Notice the demonstration was provided in a different shelf (lower shelf) as shown in Fig. 6. Regardless, the proposed framework enabled real-time collision-free solution directly with one single new information: the new goal pose. The resulting trajectory for the three book placements are shown in Fig. 5. The red, blue and green curves depict the paths to place three books in a line. The first path is without any obstacle, the second one is with one obstacle (the larger one) and the third is with two obstacles of different size.

C. Analyze the real-time capabilities

The average computation time to complete all the aforementioned tasks ⁷ are shown in Table I. Notice that for some ⁷The average time refers to the complete task and not to the motiongeneration and control loop which were running under 0.1 ms for all cases.





Fig. 7. Transfer generalization between user-guided demonstration from single-arm to the Baxter dual-arm robot for pouring/transfer task. The screenshots on the top depicts different moments of the task from right to left. The yellow and red poses depict the path for the left and right end-effectors whereas planned path using the single-arm guidance is shown in blue. Notice that both end-effectors need to combine coupled attitude and translation motion to ensure the final pouring path consists of mostly rotation as shown in the single-arm demonstration.

tasks, the collision avoidance led to less computational time. This was due to the deflected trajectory being closer to the goal and highlights the heuristic designed in Section IV. This also highlights the real-time capability of our approach. The computation for each rapidly expanding plane-oriented escaping tree level took approximately 0.0429 ± 0.01094 ms, therefore having the potential to evade dynamic obstacles in real-time. Algorithms were implemented in C++ (not optimized) using the DQ_Robotics [52] library.

D. Transfer to different kinematics: Bi-manual System

The final experiment that we conduct validates one interesting feature of our work – generalizing to multiple robots with different hardware architecture and kinematic chain. As our user guidance based planning is done in Spin(3) $\ltimes \mathbb{R}^3$, it implies that the same user demonstration can be used again for roughly same constrained tasks and/or for different robotic systems. Therefore, we used the demonstration provided on the Panda robot for the pouring task and used the final generated path to a new goal location as an input for the dual-arm Baxter transfer task (Fig. 7) which also inherently had different initial and final poses.

VII. CONCLUSION

This paper presents a novel framework for using human demonstration to generate real-time motion plans for complex tasks that satisfy the embedded implicit constraints. Our approach includes a reactive collision-avoidance algorithm that allows execution in the presence of obstacles that were not present during demonstration. Our technical approach exploits the group structure of rigid body motion and also utilizes the computational efficiency of Spin(3) $\ltimes \mathbb{R}^3$ representation of rigid body configuration. This allows real-time execution. Extensive experimental studies were performed to highlight the generalization capability of our approach. In future works, we plan to integrate vision feedback for unknown environments [53], improve smoothness to C1 trajectories

[46], and introduce multi-modal strategies and human-factors for safe online trajectory adaptation in HRI scenarios as in [54].

REFERENCES

- [1] D. Kragic, J. Gustafson, H. Karaoguz, P. Jensfelt, and R. Krug, "Interactive, collaborative robots: Challenges and opportunities." IJCAI, 2018
- [2] R. Laha, L. F. Figueredo, J. Vrabel, A. Swikir, and S. Haddadin, "Reactive Cooperative Manipulation based on Set Primitives and Circular Fields," in *IEEE International Conference on Robotics and* Automation, Xi'an, China, May 2021.
- [3] R. Laha, "Task-specific motion planning using user-guidance, imitation, and self evaluation," Master's thesis, Stony Brook University, New York, 2018.
- [4] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," Annual Review of Control, Robotics, and Autonomous Systems, vol. 3, pp. 297-330, 2020.
- [5] S. Calinon, "Learning from demonstration (programming by demonstration)," Encyclopedia of robotics, pp. 1-8, 2018.
- [6] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013. A. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with non-
- [7] Inear dynamical systems in humanoid robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2002.
- A. Paraschos, C. Daniel, J. R. Peters, and G. Neumann, "Probabilistic movement primitives," in *Advances in Neural Information Processing* [8] Systems, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26. Curran Associates, Inc., 2013.
- [9] J. Silvério, S. Calinon, L. Rozo, and D. G. Caldwell, "Learning task priorities from demonstrations," IEEE Transactions on Robotics, vol. 35, no. 1, pp. 78-94, 2019.
- [10] J. Silvério, S. Calinon, L. Rozo, and D. G. Caldwell, "Learning competing constraints and task priorities from demonstrations of bimanual skills," arXiv preprint, 2017.
- [11] Y. Wu and Y. Demiris, "Towards one shot learning by imitation for humanoid robots," in *ICRA*. IEEE, 2010. [12] M. Zeestraten, I. Havoutis, J. Silvério, S. Calinon, and D. Caldwell,
- 'An approach for imitation learning on Riemannian manifolds," RA-L, 2017.
- [13] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine, 'One-shot imitation from observing humans via domain-adaptive meta-Learning," in *Robotics:Science & Systems*, 2018.C. Atkeson and S. Schaal, "Learning tasks from a single demonstra-
- [14] tion," in Proceedings of International Conference on Robotics and Automation, vol. 2, 1997, pp. 1706–1712 vol.2.
- [15] P. Praveena, D. Rakita, B. Mutlu, and M. Gleicher, "User-guided offline synthesis of robot arm motion from 6-dof paths," in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 8825-8831
- [16] J. Denny, J. Colbert, H. Qin, and N. M. Amato, "On the theory of user-guided planning," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2016, pp. 4794–4801.
- [17] C. B. Phillips, J. Zhao, and N. I. Badler, "Interactive real-time articulated figure manipulation using multiple kinematic constraints," in *Proceedings of the 1990 Symposium on interactive 3D Graphics*, 1990, pp. 245–250.
- [18] M. Gleicher, "Retargetting motion to new characters," in Proceedings of the 25th annual conference on Computer graphics and interactive techniques, 1998, pp. 33-42.
- [19] F. Islam, O. Salzman, and M. Likhachev, "Online, interactive user guidance for high-dimensional, constrained motion planning," in Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 4921-4928.
- [20] Q.-C. Pham, S. Caron, and Y. Nakamura, "Kinodynamic planning in the configuration space via admissible velocity propagation." in Robotics: Science and Systems, vol. 32, 2013.
- [21] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion with dynamical movement primitives," in Workshop on Robot Programming by Demonstration, IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003.
- [22] B. V. Adorno, "Robot Kinematic Modeling and Control Based on Dual Quaternion Algebra - Part I: Fundamentals - hal-01478225," p. 47, 2017.
- [23] B. V. A, "Two-arm manipulation: From manipulators to end'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM) - Université Montpellier 2, Montpellier, France, 2011. Y. Wu, X. Hu, D. Hu, T. Li, and J. Lian, "Strapdown inertial navigation
- [24] system algorithms based on dual quaternions," IEEE Transactions On Aerospace And Electronic Systems, vol. 41, no. 1, pp. 110-132, 2005.

- [25] L. F. C. Figueredo, "Kinematic control based on dual quaternion algebra and its application to robot manipulators," Ph.D. dissertation, University of Brasilia, Brazil, 2016.
- [26] H. T. Kussaba, L. F. Figueredo, J. Y. Ishihara, and B. V. Adorno, "Hybrid kinematic control for rigid body pose stabilization using dual quaternions," Journal of the Franklin Institute, vol. 354, no. 7, pp. 2769–2787, 2017.
- [27] P. P. Magro, H. T. Kussaba, L. F. Figueredo, and J. Y. Ishihara, "Dual quaternion-based bimodal global control for robust rigid body pose kinematic stabilization," in American Control Conference (ACC), 2017.
- EEEE, 2017, pp. 1205–1210.
 J. Funda and R. Paul, "A computational analysis of screw transformations in robotics," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 3, pp. 348–356, 1990. [28]
- [29] E. Özgür and Y. Mezouar, "Kinematic modeling and control of a robot arm using unit dual quaternions," *Robotics and* Autonomous Systems, vol. 77, pp. 66 – 73, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889015301184
- [30] X. Yang, H. Wu, Y. Li, S. Kang, and B. Chen, "Computationally efficient inverse dynamics of a class of six-dof parallel robots: Dual quaternion approach," *Journal of Intelligent & Robotic Systems*, vol. 94, pp. 101–113, 2019.
- N. A. Aspragathos and J. K. Dimitros, "A Comparative Study of Three [31] Methods for Robot Kinematics," IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 28, no. 2, pp. 135-145, 1998
- [32] J. Kuipers, Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality. Princeton University Press, 1999
- [33] J. M. Selig, Geometric Fundamentals of Robotics, 2nd ed. Springer-Verlag New York Inc., 2005.
- W. M. Boothby, An Introduction to Differentiable Manifolds and [34] Riemannian Geometry, 2nd ed. Academic Press, 2002.
- [35] B. Busam, T. Birdal, and N. Navab, "Camera pose filtering with local regression geodesics on the riemannian manifold of dual quaternions, in 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2436–2445. [36] F. C. Park, "Distance Metrics on the Rigid-Body Motions with
- Applications to Mechanism Design," Journal of Mechanical Design Transactions of ASME, vol. 117, no. 1, pp. 48–54, 1995.
- E. Zacur, M. Bossa, and S. Olmos, "Left-Invariant Riemannian Geodesics on Spatial Transformation Groups," *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1503–1557, jul 2014. Y. L. Sachkov, "Control theory on Lie groups," *Journal of Mathematical* [37]
- [38] Sciences, vol. 156, no. 3, pp. 381-439, 2009.
- J. Gallier and J. Quaintance, Notes on Differential Geometry and Lie [39] Groups. Department of Computer and Information Science University of Pennsylvania, 2016.
- [40] B. Busam, T. Birdal, and N. Navab, "Camera Pose Filtering with Local Regression Geodesics on the Riemannian Manifold of Dual Quaternions," ArXiv e-prints, 2017. [Online]. Available: http://arxiv.org/abs/1704.07072
- [41] M. Lorenzi and X. Pennec, "Geodesics, parallel transport & oneparameter subgroups for diffeomorphic image registration," International Journal of Computer Vision, vol. 105, pp. 111-127, 2013.
- A. Sarker, A. Sinha, and N. Chakraborty, 'On screw linear interpolation for point-to-point path planning," in 2020 IEEE/RSJ International [42] Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 9480-9487.
- X. Wang, D. Han, C. Yu, and Z. Zheng, "The geometric structure of unit dual quaternion with application in kinematic control," *Journal* [43] of Mathematical Analysis and Applications, vol. 389, no. 2, pp. 1352-1364, May 2012.
- L. Kavan, S. Collins, C. O'Sullivan, and J. Zara, "Dual quaternions for rigid transformation blending," Trinity College Dublin, Tech. Rep., [44] Trinity College Dublin 2006.
- M.-j. Kim, M.-s. Kim, and S. Y. Shin, "A compact differential formula [45] for the first derivative of a unit quaternion curve," The Journal of Visualization and Computer Animation, vol. 7, no. 1, pp. 43–57, 1996.
- [46] F. Allmendinger, S. Charaf Eddine, and B. Corves, "Coordinateinvariant rigid-body interpolation on a parametric C1 dual quaternion curve," *Mechanism and Machine Theory*, pp. 731–744, March 2018. [47] R. Grassmann, L. Johannsmeier, and S. Haddadin, "Smooth point-
- to-point trajectory planning in se(3) with self-collision and joint constraints avoidance," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 1–9.
- R. Laha, A. Rao, L. Figueredo, Q. Chang, S. Haddadin, and N. Chakraborty, "Point-to-point path planning based on user guidance [48] and screw linear interpolation," in *Proceedings of the ASME Interna-*tional Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE), August 2021. O. P. Agrawal, "Hamilton operators and dual-number-quaternions in
- [49] spatial kinematics," Mechanism and machine theory, vol. 22, no. 6, pp. 569-575, 1987.

- [50] L. Figueredo, B. Adorno, J. Ishihara, and G. Borges, "Robust kinematic control of manipulator robots using dual quaternion representation," in IEEE International Conference on Robotics and Automation (ICRA),
- (1013, pp. 1949–1955.
 [51] Y. Han and F. Park, "Least squares tracking on the Euclidean group," *IEEE Transactions on Automatic Control*, vol. 46, no. 7, pp. 1127–1132, 2001.
- [52] B. V. Adorno and M. M. Marinho, "Dq robotics: A library for robot
- [52] B. V. Adorno and M. M. Marinho, "Dq robotics: A library for robot modeling and control," *IEEE Robotics Automation Magazine*, 2020.
 [53] C. De Farias, M. Adjigble, B. Tamadazte, R. Stolkin, and N. Marturi, "Dual quaternion-based visual servoing for grasping moving objects," in 2021 *IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 151–158.
 [54] L. Chen, L. F. Figueredo, and M. R. Dogar, "Planning for muscular and peripersonal-space comfort during human-robot forceful collaboration," in 2018 *IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, Nov 2018, pp. 1–8.