DESIGN OF SINGULARITY-ROBUST AND TASK-PRIORITY KINEMATIC CONTROLLERS IN THE UNIT DUAL QUATERNION SPACE

Murilo Marques Marinho*, Bruno Vilhena Adorno†

*Universidade de Brasília
Brasília, Distrito Federal, Brazil

†*Universidade Federal de Minas Gerais*
Belo Horizonte, Minas Gerais, Brazil

Emails: `murilomarinho@lara.unb.br, adorno@ufmg.br`

Abstract— The great number of possible tasks that can be performed by a robotic manipulator has inspired the development of many different kinematic control strategies in the literature. These classic strategies often use minimal representations for the end-effector pose, leading to representation singularities. Alternatively, in strategies that use non-minimal representations, as homogeneous transformation matrices, it is usually difficult or counterintuitive to extract suitable parameters in order to control the robot. On the other hand, dual quaternions are a suitable representation to unify pose representation and control techniques, and hence this work reviews the design of singularity-robust and task-priority kinematic manipulator controllers and adapts these designs to the unit dual quaternion space. Further, the performance of the designed controllers is evaluated in a software framework composed of ROS, OpenRAVE and the custom DQ_robotics library.

Keywords— robotic manipulator, kinematic control, dual quaternions, ROS, OpenRAVE

1 Introduction

Robotic manipulators are used in a wide range of applications, from simple dish cooking (Yamazaki et al., 2010) to soil sample extraction on Mars[1]. Many of these applications are best described as the end-effector motion, thus motivating the design of controllers directly at the end-effector level. When those motions are performed by robots at considerably low velocities and well behaved drives—allowing the manipulator dynamics to be neglected—kinematic control strategies can be successfully applied.

Naturally, this inherent disparity among the possible tasks to be performed by the manipulator led to the creation of different kinematic control strategies. In these strategies, the controller has to impose the end effector pose according to the task description, while coping with the manipulator joint limits, task-space redundancy, task-space singularities and generic optimization criteria. These issues are very distinct among themselves. Whereas redundancy concerns the manipulator possessing more degrees of freedom than a task requires, resulting in infinite solutions to the inverse kinematics problem, singularities concern the degeneration of a given amount of degrees of freedom when the robot is in specific configurations, in which small velocities in the task space may only be achieved by imposing dangerously high velocities to the robot actuators.

In order to engage this issues systematically, a suitable rigid body motion parameterization must be chosen. Unit dual quaternions have a convincing set of advantages over more widespread representations, such as homogeneous matrices and Euler angles plus Cartesian coordinates (Adorno, 2011; Aspragathos & Dimitros, 1998; Pham et al., 2010). Most controllers, however, must be redesigned in unit dual quaternion space. In order to design and implement these controllers based on dual quaternions in a real system, some efforts have been done towards the development of a unit dual quaternion computational open-source library, called DQ_robotics[2], which is available in Matlab, C, and C++ languages. It implements much of the basic dual quaternion arithmetics and manipulator kinematics functions, but still lacks manipulator robots control algorithms in an easy-to-use interface available to the robotics community. This is a systematic issue in the robotic manipulation research: even though there are many algorithms available, they are seldom made openly available for anyone to use and evaluate.

With respect to the use of custom libraries in physical robotic systems, much attention has been given to the modularity and reusability of the Robot Operating System (ROS)[3]. In this system, sensor and robot interfaces are openly shareable among developers in a systematic and organized way, hastening the development process. One key characteristic of ROS is its capability of incorporating different libraries and frameworks in a unified net of services and topics. This allows the unified integration of both custom libraries and widely used frameworks.

One of these frameworks is the Open Robotics Automation Virtual Environment (OpenRAVE) (Diankov, 2010). OpenRAVE allows the testing and simulation of generic robotic systems in a virtual environment. Moreover, it implements sev-

[1]http://www.nasa.gov/mission_pages/msl/index.html

[2]http://dqrobotics.sourceforge.net/

[3]http://www.ros.org

eral motion planning algorithms which are fully available for use. Naturally, the most reasonable way to make an initial validation of a robot controller would be to start in a simulated environment such as OpenRAVE.

In this work, we tackle both mathematical and computational problems of implementing kinematic controllers, with or without numerical filtering, and the pose controller with position priority in unit dual quaternion space, augmenting the capabilities of the DQ_robotics/C++ library. The controllers are implemented and evaluated in a framework composed of ROS, OpenRave, and the DQ_robotics/C++ library.

2 Mathematical Background

Some mathematical tools are reviewed in this section for better understanding of the controllers devised in unit dual quaternion space. We begin by recalling dual quaternions and their basic algebra when representing rigid transformations, followed by a brief reminder of robotic manipulator kinematic control schemes and some problems that may arise in the execution of tasks represented in the task-space.

2.1 Dual Quaternions

Dual quaternions are the basic building blocks of the kinematic control theory implemented in this work. We begin by defining $\hat{\imath}$, $\hat{\jmath}$ and $\hat{k}$ as the three imaginary components of a quaternion such that (Kuipers, 1999)

$$\hat{\imath}^2 = \hat{\jmath}^2 = \hat{k}^2 = -1 \text{ and } \hat{\imath}\hat{\jmath}\hat{k} = -1.$$

An arbitrary rotation of a rigid body by an angle θ around an axis $\boldsymbol{n} = n_x\hat{\imath} + n_y\hat{\jmath} + n_z\hat{k}$ is represented by the unit quaternion

$$\boldsymbol{r} = \cos\left(\frac{\theta}{2}\right) + \boldsymbol{n}\sin\left(\frac{\theta}{2}\right).$$

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

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

where ε is nilpotent; i.e., $\varepsilon \neq 0$ but $\varepsilon^2 = 0$ (Selig, 2005).

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

$$\operatorname{vec}(\underline{\boldsymbol{p}}) \triangleq \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 & q_7 & q_8 \end{bmatrix}^T,$$

where q_i, $i = 1 \dots 8$, are the coefficients of $\underline{\boldsymbol{p}}$.

2.2 Kinematic Control of Serial Manipulators

The manipulator forward kinematics model (FKM) provides the pose of the end effector from the current values of the joint variables. Usually, the FKM of a manipulator robot can be easily obtained from the Denavit-Hartenberg parameters directly in dual quaternion space (Adorno, 2011).

At a first glance, the inverse kinematics, which is our aim in most tasks, should be as easy as the forward kinematics. However, a closed-form solution for the inverse kinematics does not exist for the general case. A common alternative is to use the inverse differential kinematics. More specifically, consider a serial manipulator with more than six degrees of freedom, which will be redundant for the pose control task. The differential FKM maps the manipulator joint velocities $\dot{\theta} \in \mathbb{R}^{n\times 1}$ into the generalized end-effector velocity $\operatorname{vec}\dot{\underline{\boldsymbol{p}}}_{eff} \in \mathbb{R}^{8\times 1}$:

$$\operatorname{vec}\dot{\underline{\boldsymbol{p}}}_{eff} = J_A\dot{\theta}, \tag{1}$$

where $J_A \in \mathbb{R}^{8\times n}$ is called the manipulator analytical Jacobian (Pham et al., 2010) and depends on the current robot posture. As (1) is a simple linear mapping, it is common practice to invert the Jacobian to obtain the joint velocities for a given desired end effector velocity $\dot{\underline{\boldsymbol{p}}}_d$; that is,

$$\dot{\theta} = {J_A}^{inv}\operatorname{vec}(\dot{\underline{\boldsymbol{p}}}_d), \tag{2}$$

where J_A^{inv} represents a generalized matrix inversion that suits the manipulator posture, number of degrees of freedom and general optimization criteria. This process is called the inverse differential FKM. Problems related to the robot redundancy and robustness to singularities in the task space are often determined by the choice of the generalized inversion J_A^{inv}.

Manipulator redundancy is the existence of more degrees of freedom in the manipulator than the necessary to perform a given task. Mathematically, this redundancy allows (2) to have an infinite number of solutions. In this situation, the matrix inversion is not defined, and we often resort to pseudoinverses (or generalized inverses). The Moore-Penrose pseudoinverse, for instance, gives the minimum-norm solution of the joint velocities, and is an intuitive resource to handle this situation. On the other hand, redundancy allows the controller to use the extra degrees of freedom in the execution of secondary tasks. This is usually performed by projecting the secondary control law in the nullspace of the primary task, as was introduced by Liegeois (1977). In this case, however, one does not obtain the minimum norm solution for the vector of joint velocities.

Singularities correspond to certain configurations in which the rank of the Jacobian in (2) degenerates. This usually happens when, for example, the manipulator is stretched to its limit. In the neighborhood of a singularity, the singular values of the Jacobian become extremely small, which results in high values for the joint velocities when the Moore-Penrose pseudoinverse is directly applied. For this reason, earlier works (Chiaverini, 1997; Maciejewski & Klein, 1988) suggest the use of damped pseudoinverses (with or without numerical filtering) or SVD-based pseudoinverses with truncated singular values.

3 Control Strategies

In the previous section, it was noted that the choice of a suitable generalized inverse J_A^{inv} can be used to deal with manipulator redundancy and singularities in the task space. The use of the task nullspace is another possibility that increases the number of possible controller designs. This section describes controller designs that consider singularities and multiple tasks in unit dual quaternion space.

3.1 Pose controller with damped pseudoinverse

As the differential mapping is fairly simple, the ordinary closed loop solution for the end effector pose control in unit dual quaternion space is

$$\dot{\theta} = K {J_A}^{inv} \operatorname{vec}(\underline{\boldsymbol{p}}_{eff} - \underline{\boldsymbol{p}}_d), \tag{3}$$

being K a positive definite gain matrix that defines the rate in which this controller converges (Pham et al., 2010).

An initial approach to the kinematic singularity issue would be to choose ${J_A}^{inv}$ in (3) as the damped pseudoinverse (Maciejewski & Klein, 1988)

$$J_A^{inv} \triangleq J_A^T (J_A J_A^T + \lambda^2 I)^{-1}, \tag{4}$$

where $\lambda \in \mathbb{R}$ is the damping factor of the inversion. This allows the controller to have some robustness against task space singularities, at the cost of steady state error in the end effector pose. Another reasonable choice in some tasks is to use the Moore-Penrose pseudo-inverse with truncated singular values.

Implementation aspects: when implementing damped inversions such as (4), it is a good practice to exploit the mathematical properties of these matrices for a more efficient algorithm implementation. Notice that $B = (J_A J_A^T + \lambda^2 I)$ is a positive definite matrix for $\lambda \neq 0$ and any real matrix J_A; so the solution of $X = CB^{-1}$, where $C \triangleq J_A^T$ and $X \triangleq J_A^{inv}$, can be found more efficiently using the Cholesky decomposition. Some linear algebra libraries[4] have implementations of this algorithm.

[4]We used the Eigen 3 library in our C++ implementation. http://eigen.tuxfamily.org/

3.2 Pose controller with numerical filtering

To begin the discussion about the pose controller with numerical filtering, consider the singular value decomposition

$$J_A = U \Sigma V^T,$$

where $U \in \mathbb{R}^{8\times 8}$, $V \in \mathbb{R}^{n\times n}$; $U^{-1} = U^T$, $V^{-1} = V^T$ and $\Sigma = \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix}$, where S is the diagonal matrix containing the singular values of J_A in descending order from σ_1 to σ_6. Note that even though $J_A \in \mathbb{R}^{8\times n}$, its maximum rank is 6 due to the fact that unit dual quaternions are not a minimal representation.

As introduced by Chiaverini (1997), choosing a static damping factor λ in (4) causes an inversion error even when the manipulator is not near a task space singularity. Adapting the algorithm for the unit dual quaternion space controller, the solution becomes

$$\begin{cases} \lambda^2 = 0 & ,\sigma_6 > \mu \\ \lambda^2 = (1 - (\frac{\sigma_6}{\mu})^2)\lambda^2_{max} & ,\sigma_6 \leq \mu \end{cases}. \tag{5}$$

The distance from a singularity is defined by $\mu > 0 \in \mathbb{R}$ and the maximum damping factor, λ^2_{max}, is applied when the manipulator is at the singularity. Using this variable damping factor allows the controller to have no steady state error when the manipulator is far from a singularity.

When using (5) in (4), and the controller is near a singular configuration, the inversion is isotropically damped. This means that not only the unfeasible directions of the end-effector velocity are damped, but also the feasible ones. In cases where $\sigma_6 \leq \mu$ and $\sigma_5 > \mu$, the manipulator is at a single kinematic singularity. In this situation, it is possible to successfully filter the damping such that only the unfeasible direction will be damped (Maciejewski & Klein, 1988). This process is called numerical filtering, and we may apply it to the controller by choosing

$$J_A^{inv} \triangleq J_A^T (J_A J_A^T + \lambda^2 u_6 u_6^T)^{-1},$$

where u_6 is the sixth column of the matrix U; that is, u_6 is the column vector in U that corresponds to the sixth singular value.

As it is possible for the controller to encounter more than one singularity at a given posture, an isotropic damping $\beta > 0 \in \mathbb{R}$ where $\beta^2 << \lambda^2$ is added to the inversion, resulting in

$$J_A^{inv} \triangleq J_A^T (J_A J_A^T + \lambda^2 u_6 u_6^T + \beta^2 I)^{-1}, \tag{6}$$

which is then used in (3).

Implementation aspects: even though the variable damping and numerical filtering allow better steady state results, the need for computing the SVD of the Jacobian may hinder the use of the controller in less powerful computational systems.

3.3 *Task-priority based controller using quaternion primitives*

There are some situations (e.g., singularities, joint limits, intrinsic under-actuation) in which some regions in a manipulator configuration space are not reachable and hence the end effector pose cannot be arbitrarily chosen for all regions of the robot workspace. Although this issue is more evident when using manipulators with less than six degrees of freedom, redundant manipulators may also experience loss of dexterity inside its workspace due to singularities or joint limits. On the other hand, some tasks, as pick and place operations with small sealed objects, may only require the complete end effector pose control in specific parts of the tasks trajectory. During the trajectory execution, the end effector position holds more importance than the end effector orientation. Using a task-priority controller with end effector position as the first objective and orientation as the second, the task is fully modeled: full pose control will be achieved when the manipulator is able to comply, or else only the position is effectively controlled while the orientation is stabilized. Clearly, sometimes orientation control is more relevant for the task than position control. However, since the reasoning is the same in both cases, in the following we present only the situation where the position controller has higher priority than the orientation controller.

In order to mathematically model the task-priority controller in dual quaternion space, we begin by defining the position Jacobian, $J_p \in \mathbb{R}^{4\times n}$, such that

$$\operatorname{vec}\dot{\boldsymbol{t}}_{eff} = J_p\dot{\theta}, \tag{7}$$

where $\boldsymbol{t}_{eff}$ is the end effector position and we also define the orientation Jacobian, $J_o \in \mathbb{R}^{4\times n}$, such that

$$\operatorname{vec}\dot{\boldsymbol{r}}_{eff} = J_o\dot{\theta}, \tag{8}$$

where $\boldsymbol{r}_{eff}$ is the end effector orientation (Adorno, 2011).

The complete controller design for task priority with position as the first objective and orientation as the second is given by

$$\begin{aligned}\dot{\theta} &= J_p^{inv}\operatorname{vec}(\boldsymbol{t}_{eff} - \boldsymbol{t}_d)\\ &\quad + N_p J_o^{inv}\operatorname{vec}(\boldsymbol{r}_{eff} - \boldsymbol{r}_d),\end{aligned} \tag{9}$$

where N_p is the nullspace projector (Liegeois, 1977); that is,

$$N_p = (I - J_p J_p^{inv}). \tag{10}$$

The generalized matrix inversion operator in (9) may be chosen as any of the possibilities presented in Sections 3.1 and 3.2. For instance, a simple choice is

$$\begin{cases} J_p^{inv} &= J_p^T(J_pJ_p^T + \lambda_{translation}^2 I)^{-1}\\ J_o^{inv} &= J_o^T(J_oJ_o^T + \lambda_{rotation}^2 I)^{-1}\end{cases}.$$

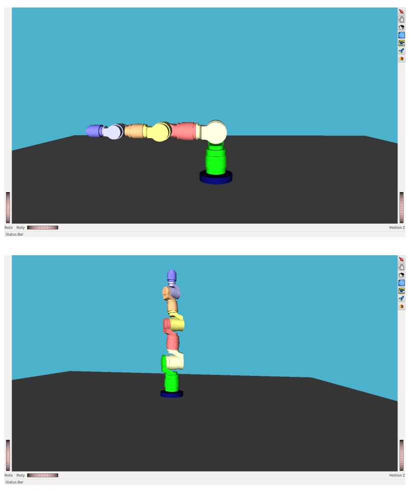


Figure 1: Manipulator configurations during the simulations: initial configuration (*top*) and final configuration (*bottom*).

4 Simulations

In order to evaluate the controllers, a software framework composed of ROS, OpenRave, and DQ_robotics was created and tested in a Ubuntu 12.04 64bits environment. All the controllers explained in the last section were implemented in the DQ_robotics library and are openly available for use. The library was organized to allow the use of any serial link manipulator and has also an integrated data plotting software.

The simulated manipulator chosen for the simulations was a 7 degrees-of-freedom Schunk lwa3. The task, to be performed by all controllers, was to move from an initial singular configuration

$$\theta = \begin{bmatrix}0 & \frac{\pi}{2} & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T$$

to the final singular end effector pose

$$\underline{\boldsymbol{p}}_d = \hat{k} - \varepsilon\frac{1.305}{2}.$$

Figure 1 shows the initial and final configurations of the robot manipulator.

The purpose of placing initial and final configurations inside singular regions was to evaluate the robustness to singularities of all controllers. In order to have a standard reference for comparison, simulations were also performed with a controller which uses the Moore-Penrose pseudoinverse in (3).

The controllers are considered to have reached the target pose when the selected error measurements are within a 10^{-2} tolerance. In the pose controller with damped pseudoinverse, in the one with Moore-Penrose pseudoinverse, and in the one with numerical filtering, this tolerance is

$$\left\|\operatorname{vec}\left(\underline{\boldsymbol{p}}_{eff} - \underline{\boldsymbol{p}}_d\right)\right\| < 10^{-2}$$

and, in the priority controller, this tolerance is

$$\|\mathrm{vec}\,(\boldsymbol{t}_{eff} - \boldsymbol{t}_d)\| < 10^{-2}.$$

The sample rate used in the simulation was $100Hz$. In addition, the robot joint velocities were considered uniform during the sampling interval, simulating a zero order hold of the controllers output in that iteration.

During the simulations, the scalar gains for the controllers based on (4), (6), and the controller with Moore-Penrose pseudoinverse were selected as $K = 0.35$, and the gains for the task-priority controller were chosen as $K_p = 0.25$ and $K_o = 0.05$.

The damping factors were $\lambda^2 = 0.1$ in the pose controller with damped pseudoinverse; $\lambda^2 = 0.1$, $\beta^2 = 0.01$ and $\mu = 0.1$ in the numerical filtered controller, and $\lambda^2_{translation} = 0.15$, $\lambda^2_{rotation} = 0.4$ in the task-priority controller with translation priority. In conjunction with the gains, the damping factors were chosen as to make the upper-bound for the joint velocities equal to $0.5rad/s$, considering that the maximum angular velocity of any joint of the Schunk robot[5] is approximately 0.8 rad/s. Figure 2 presents the performances of each controller.

The results of the controller with damped pseudoinverse and of the numerical filtered controller are hard to distinguish. As the damping is approximately the same for both controllers, the effectiveness in pose control is slightly noticeable from the difference in the convergence time. While the numerical filtered controller converged in 53.51 s, the controller with damped pseudoinverse converged in 53.54 s. This small difference can be explained by the fact that the numerical filtered controller gives a more accurate inverse far from singular configurations. However, despite the fact that only the initial and final poses were at singular configurations, both controllers behaved approximately in the same way, opposed to our initial expectation. This is an issue that we may investigate in future works.

On the other hand, the task-priority controller with translation priority had a different behavior. As we chose smaller gains for this controller in order to obey the constraint of joint velocities (i.e., $< 0.5\,rad/s$), it converged slower than the damped and the numerical filtered controllers (approximately 80 s). In addition, the prioritized scheme was also robust to the singular configurations, suggesting that the partitioning of the Jacobian matrix into smaller matrices improved the numerical conditioning of the whole system. However, aside from the empirical results we did not perform any theoretical study to support this claim. As a consequence, in future works we intend to investigate the supposedly robustness of the prioritized scheme with dual quaternions from a theoretical standpoint.

[5]http://www.schunk.com/schunk_files/attachments/PRL_gesamt_EN.pdf

The last controller, which makes use of a classic Moore-Penrose pseudoinverse, shows the negative effects of the ill-conditioning near singular configurations. Due to the singularity at the initial pose, some joints reached velocities of 28175.7 rad/s in the first iteration, which is completely unfeasible in practice. This caused an impulse response in Figure 2c at 0 s. The manipulator was only able to converge because dynamical aspects were not taken into consideration in the simulation and because we used low gains (for larger gains this usually leads to complete instability). In this way, the manipulator was able to recover itself once outside the singular configuration. However, in practice these extremely large velocities can cause damage to the robot.

5 Conclusion & future work

A mathematical and computational framework for kinematic control of serial manipulators has been developed in this work. The issues of manipulator task redundancy and workspace singularities were undertaken in the controller designs, which exploited the benefits of the unit dual quaternion representation. The simulation results have shown that the proposed controllers are capable of tackling initial and final singular configurations, while maintaining feasible velocities throughout the trajectory. Moreover, all controllers can be used by the community as they are available as an open-source project.

In future works, more challenging controller designs that consider manipulator joint limits and more complex geometric tasks will be developed or adapted for the unit dual quaternion space.

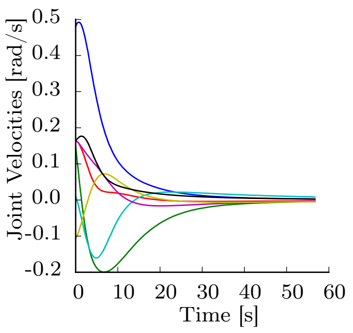
Acknowledgements

The authors want to thank Mateus Martins for his contribution on an earlier version of the DQ_robotics C++ API (Martins, 2012). This work has been supported by the Brazilian agencies CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), FAPEMIG, and CNPq under grant 473056/2011-9.

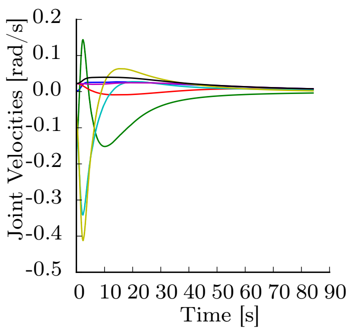
References

Adorno, B. V. (2011). *Two-arm manipulation: from manipulators to enhanced human-robot collaboration [Contribution à la manipulation à deux bras : des manipulateurs à la collaboration homme-robot]*. PhD thesis, Université Montpellier 2, Montpellier, France.

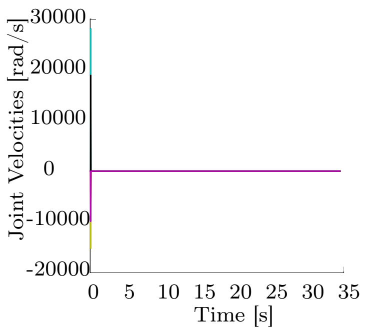
Aspragathos, N. A. & Dimitros, J. K. (1998). A comparative study of three methods for robot kinematics. *IEEE transactions on systems, man, and cy-*



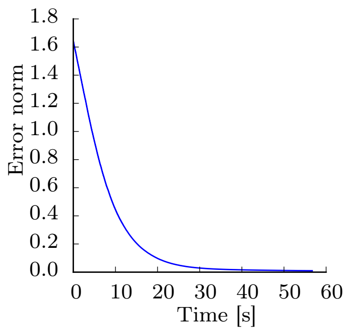
(a) Joint velocities of the controller with damped pseudoinverse (the joint velocities of the numerical filtered pose controller were identical).



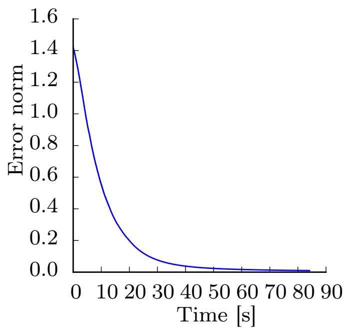
(b) Joint velocities of the task-priority controller.



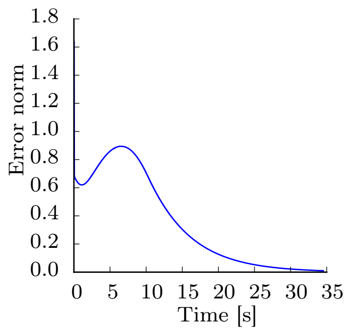
(c) Joint velocities of the controller with Moore-Penrose pseudoinverse.



(d) Dual quaternion error norm of the controller with damped pseudoinverse (the dual quaternion error norm of the numerical filtered pose controller was identical).



(e) Translation error norm of the task-priority controller.



(f) Dual quaternion error norm of the controller with Moore-Penrose pseudoinverse.

Figure 2: Comparison between the performances of the controllers.

bernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, 28(2), 135–45.

Chiaverini, S. (1997). Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3), 398–410.

Diankov, R. (2010). *Automated Construction of Robotic Manipulation Programs.* PhD thesis, Carnegie Mellon University, Robotics Institute.

Kuipers, J. (1999). *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality.* Princeton University Press.

Liegeois, A. (1977). Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man and Cybernetics*, 7, 868–871.

Maciejewski, A. & Klein, C. (1988). Numerical filtering for the operation of robotic manipulators through kinematically singular configurations. *Journal of Robotic Systems*, 5, 527–552.

Martins, M. R. (2012). Desenvolvimento de uma biblioteca em C++ para modelagem e controle cinemático de robôs manipuladores, *Undergraduate Final Project*, Universidade Federal de Minas Gerais, *Defended on Jan 2012.*

Pham, H.-L., Perdereau, V., Adorno, B. V., & Fraisse, P. (2010). Position and orientation control of robot manipulators using dual quaternion feedback. *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 658–663).

Selig, J. M. (2005). *Geometric fundamentals of robotics.* Springer-Verlag New York Inc., 2nd edition.

Yamazaki, K., Watanabe, Y., Nagahama, K., Okada, K., & Inaba, M. (2010). Recognition and manipulation integration for a daily assistive robot working on kitchen environments. *2010 IEEE International Conference on Robotics and Biomimetics*, (pp. 196–201).