Time-optimal path parameterization of rigid-body motions: applications to spacecraft reorientation

Huy Nguyen[*] and Quang-Cuong Pham[†]

Nanyang Technological University, Singapore 639798

I. Introduction

In this paper, we are interested in finding fast, collision-free, motions for transferring a rigid body (e.g. a rigid spacecraft) from an initial configuration (position and orientation) to a goal configuration, subject to bounds on angular velocities, angular accelerations or torques. This problem is difficult because of the non-trivial natures of the *kinodynamic bounds* [‡] and of the underlying space, the space of rigid body configurations, or SE(3) [§].

When only rotations are involved (no translation), the space of possible motions is SO(3) [¶]. Li and Bainum studied maneuvers based on rotations around a principal axis (or eigenaxis rotations), which they thought to be time-optimal for an inertially symmetric body with independent three-axis control [2]. Bilimoria and Wie showed that, even for inertially symmetric bodies, time-optimal maneuvers actually include significant nutational components [3]. More recently, Bai and Junkins refined Bilimoria and Wie's results by describing precisely various classes of time-optimal maneuvers [4]. The cited works have in common that they try to compute *exact* time-optimal motions by applying directly Pontryagin's minimum principle. While the theoretical interest of such an approach is evident, in practice, it is very computationally intensive and does not seem to be generalizable beyond rest-to-rest motions in obstacle-free environments.

Yet, the ability to move without colliding with the environment is crucial in cluttered space

[*] School of Mechanical and Aerospace Engineering, NTU, Singapore

[†] School of Mechanical and Aerospace Engineering, NTU, Singapore

[‡] In robotics, "kinodynamic constraints" refer to constraints that can only be expressed based on the time derivatives of the robot configuration, for instance, bounds on the robot velocity, acceleration, and force/torque [1].

[§] Special Euclidean group in dimension 3.

[¶] Special Orthogonal group in dimension 3.

stations. In some applications, the satellite or spacecraft must avoid going through some orientations (in order e.g. to keep communication uninterrupted with the base station), which can be also modelled as obstacle in SO(3) or SE(3). Beyond the aerospace field, planning fast collision-free rigid body motions is also important in many robotics or computer graphics applications.

We present an efficient method to find fast maneuvers in SO(3) considering collision avoidance. For that, we adopt the plan-and-shortcut method [5] widely used in the robotics community

1. Plan a collision-free *geometric path* between the initial and the final orientation;

2. Time-parameterize that geometric path to obtain a *trajectory* that respects the kinodynamic constraints (bounds on angular velocities, angular accelerations or torques);

3. Repeatedly apply *shortcuts* to decrease the time duration of the obtained trajectory.

Step 1 of the above algorithm has been addressed, in particular by Kuffner [6], by extending the classical Rapidly exploring Random Tree (RRT) algorithm [1] to SO(3). Regarding steps 2 and 3, the crucial requirement is the ability to optimally *time-parameterize* a given path in SO(3) under kinodynamic constraints. We do so by extending the classical [7–9] Time-Optimal Path Parameterization (TOPP) algorithm to the case of SO(3), which constitutes the main contribution of this paper. We show that, overall, the method we propose can find fast maneuvers for a satellite model in a cluttered environment in less than 10 seconds. We also present an extension to the space of three-dimensional rigid body motions, SE(3).

This paper is organised as follows. In Section II, we recall some backgrounds regarding the problem of interpolation in SO(3) and SE(3). Choosing a good method to interpolate paths between two elements of SO(3) or SE(3) is particularly important: we need paths that are “short” and smooth, with easy-to-compute first and second derivatives, so that the subsequent time-parameterization can efficiently yield high-quality trajectories. In Section III, we present the extension of TOPP to SO(3). In Sections IV, we present the implementation of the planning method and simulation results on several spacecraft maneuver problems. Section V concludes and sketches some future research directions.

II. Interpolation in SO(3) and SE(3)

A. Notations

We first recall some useful notations in the study of SO(3) [10]. Given a vector $\boldsymbol{r} \in \mathbb{R}^3$, one defines the skew-symmetric matrix

$$[\boldsymbol{r}] \overset{\text{def}}{=} \begin{bmatrix} 0 & -r_3 & r_2 \\ r_3 & 0 & -r_1 \\ -r_2 & r_1 & 0 \end{bmatrix}.$$

The exponential of a skew-symmetric matrix defined as above is given by

$$\boldsymbol{e}^{[\boldsymbol{r}]} \overset{\text{def}}{=} \mathbf{I} + \frac{\sin\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|}[\boldsymbol{r}] + \frac{1-\cos\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^2}[\boldsymbol{r}]^2. \tag{1}$$

Next, for any matrix $\mathbf{R} \in \mathrm{SO}(3)$, there exists $\boldsymbol{r}$ such that $\mathbf{R} = \boldsymbol{e}^{[\boldsymbol{r}]}$, and one denotes $\log \mathbf{R} \overset{\text{def}}{=} [\boldsymbol{r}]$. More precisely, if $\mathrm{tr}(\mathbf{R}) \neq 1$, then

$$\log \mathbf{R} = \frac{\phi}{2\sin\phi}(\mathbf{R} - \mathbf{R}^\top), \tag{2}$$

where ϕ satisfies $1 + 2\cos\phi = \mathrm{tr}(\mathbf{R})$. If $\mathrm{tr}(\mathbf{R}) = 1$, then

$$\log \mathbf{R} = \pm\pi[\hat{\boldsymbol{v}}], \tag{3}$$

where $\hat{\boldsymbol{v}}$ is a unit length eigenvector of $\mathbf{R}$ associated with the eigenvalue 1.

B. Interpolation in SO(3)

Consider two orientation matrices $\mathbf{R}_0, \mathbf{R}_1 \in \mathrm{SO}(3)$ and two vectors $\boldsymbol{\omega}_0, \boldsymbol{\omega}_1 \in \mathbb{R}^3$. We are interested in the problem of finding a smooth, "short" curve $(\mathbf{R}(t))_{t\in[0,1]}$ in SO(3) such that

$$\mathbf{R}(0) = \mathbf{R}_0,\ \mathbf{R}(1) = \mathbf{R}_1,\ \boldsymbol{\omega}(0) = \boldsymbol{\omega}_0,\ \boldsymbol{\omega}(1) = \boldsymbol{\omega}_1,$$

where $\boldsymbol{\omega}(t)$ denotes the angular velocity vector computed in the moving (body) frame.

Consider a vector $\boldsymbol{r}_1$ such that $[\boldsymbol{r}_1] = \log(\mathbf{R}_0^\top \mathbf{R}_1)$. One possible interpolation method is given by [10, 11]

$$\mathbf{R}(t) = \mathbf{R}_0 \boldsymbol{e}^{[\boldsymbol{a}_3 t^3 + \boldsymbol{a}_2 t^2 + \boldsymbol{a}_1 t]}, \tag{4}$$

where $\boldsymbol{a}_1, \boldsymbol{a}_2, \boldsymbol{a}_3 \in \mathbb{R}^3$ are constant vectors satisfying

(i) $\boldsymbol{a}_1 = \boldsymbol{\omega}_0$;

(ii) $\boldsymbol{a}_3 + \boldsymbol{a}_2 + \boldsymbol{a}_1 = \boldsymbol{r}_1$;

(iii) $3\boldsymbol{a}_3 + 2\boldsymbol{a}_2 + \boldsymbol{a}_1 = \mathbf{A}^{-1}(\boldsymbol{r}_1)\boldsymbol{\omega}_1$ [the definition of $\mathbf{A}$ is given in equation (7) below].

This interpolation method has many advantages. First, it approximates the minimum acceleration interpolation, which implies that the underlying path is smooth and short. In particular, if $\boldsymbol{\omega}_0 = \boldsymbol{\omega}_1 = \mathbf{0}$, then this interpolation indeed yields the shortest path in SO(3) [11]. This interpolation is also bi-invariant: the resulting orientation trajectories are invariant with respect to the choice of fixed or moving reference frames. Finally, the first and second derivatives are smooth and easy to compute, which is useful for TOPP where we need smooth and easy-to-compute first and second derivatives. The angular velocity and acceleration are indeed given by

$$\begin{aligned} \boldsymbol{\omega}(t) &= \mathbf{A}(\boldsymbol{r})\dot{\boldsymbol{r}}, & (5)\\ \dot{\boldsymbol{\omega}}(t) &= \mathbf{A}(\boldsymbol{r})\ddot{\boldsymbol{r}} + \mathbf{C}(\boldsymbol{r}, \dot{\boldsymbol{r}}), & (6) \end{aligned}$$

where

$$\mathbf{A}(\boldsymbol{r}) \overset{\text{def}}{=} \mathbf{I} - \frac{1 - \cos\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^2}[\boldsymbol{r}] + \frac{\|\boldsymbol{r}\| - \sin\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^3}[\boldsymbol{r}]^2, \tag{7}$$

and

$$\begin{aligned} \mathbf{C}(\boldsymbol{r}, \dot{\boldsymbol{r}}) \overset{\text{def}}{=}\; & \frac{\|\boldsymbol{r}\| - \sin\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^3}\dot{\boldsymbol{r}} \times (\boldsymbol{r} \times \dot{\boldsymbol{r}}) \\ - \; & \frac{2\cos\|\boldsymbol{r}\| + \|\boldsymbol{r}\|\sin\|\boldsymbol{r}\| - 2}{\|\boldsymbol{r}\|^4}\boldsymbol{r}^\top \dot{\boldsymbol{r}}(\boldsymbol{r} \times \dot{\boldsymbol{r}}) \\ + \; & \frac{3\sin\|\boldsymbol{r}\| - \|\boldsymbol{r}\|\cos\|\boldsymbol{r}\| - 2\|\boldsymbol{r}\|}{\|\boldsymbol{r}\|^5}\boldsymbol{r}^\top \dot{\boldsymbol{r}}(\boldsymbol{r} \times (\boldsymbol{r} \times \dot{\boldsymbol{r}})). \end{aligned} \tag{8}$$

C. Interpolation in SE(3)

While SO(3) is the group of 3D rotations, SE(3) is the special Euclidean group of rigid-body motions. SE(3) includes both rotations and translations, and is of the form

$$\begin{bmatrix} \mathbf{R} & \boldsymbol{q} \\ 0 & 1 \end{bmatrix},$$

where $\mathbf{R} \in \mathrm{SO}(3)$ and $\boldsymbol{q} \in \mathbb{R}^3$. There are a number of available approaches to interpolate trajectories between two elements of SE(3). Park and Ravani [10] exploited the Lie group structure of SE(3) to develop an algorithm analogous to the preceding interpolation in SO(3). The resulting motion, however, is a screw motion, which corresponds to a strange motion in physical space. Moreover, in SE(3) there is in general no bi-invariant interpolations [11]. Therefore, we use a simple method, which consists in interpolating orientations and positions separately. For the orientation part, we proceed as in Section B. For the translation part, we interpolate between two pairs (position, linear velocity) by a third degree polynomial. Consider two pairs $(\boldsymbol{p}_0, \boldsymbol{v}_0)$ and $(\boldsymbol{p}_1, \boldsymbol{v}_1)$, the interpolant is given by

$$\boldsymbol{p}(t)_{t\in[0,1]} = \boldsymbol{k}_3 t^3 + \boldsymbol{k}_2 t^2 + \boldsymbol{k}_1 t + \boldsymbol{k}_0, \tag{9}$$

where $\boldsymbol{k}_3, \boldsymbol{k}_2, \boldsymbol{k}_1, \boldsymbol{k}_0$ can be easily found using the boundary conditions $\boldsymbol{p}(0) = \boldsymbol{p}_0, \boldsymbol{v}(0) = \boldsymbol{v}_0, \boldsymbol{p}(1) = \boldsymbol{p}_1, \boldsymbol{v}(1) = \boldsymbol{v}_1$ ($\boldsymbol{v}(t)$ denotes the linear velocity vector). Note that the first and second derivatives of the interpolant are trivial to compute.

III. Time-Optimal Path Parameterization in SO(3)

Consider a path $\mathcal{P}$ – represented as the underlying path of a trajectory $\boldsymbol{r}(s)_{s\in[0,s_{\mathrm{end}}]}$ – in the configuration space. Assume that $\boldsymbol{r}(s)_{s\in[0,s_{\mathrm{end}}]}$ is C^1- and piecewise C^2-continuous. We are interested in *time-parameterizations* of $\mathcal{P}$, which are increasing *scalar functions* $s : [0, T] \to [0, s_{\mathrm{end}}]$, under kinodynamic constraints.

If the constraints can be reduced to the form (note that all vector inequalities in this paper are element-wise)

$$\ddot{s}\boldsymbol{a}(s) + \dot{s}^2\boldsymbol{b}(s) + \boldsymbol{c}(s) \leq \boldsymbol{0}, \tag{10}$$

then efficient methods and implementations allow finding the time-optimal parameterization $s(t)$ (see e.g. [9]).

Consider now a rigid body with three independent actuations, such as a rigid spacecraft whose equation of motion is [2–4]

$$\mathbb{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbb{I}\boldsymbol{\omega}) = \boldsymbol{\tau}, \tag{11}$$

where $\mathbb{I}$ is the 3×3 inertia matrix of the spacecraft and $\boldsymbol{\tau}$ the 3-dimensional torque vectors. The actuation bounds are given by

$$\boldsymbol{\tau}_{\text{min}} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\text{max}}. \tag{12}$$

Consider now an orientation trajectory $\mathbf{R}(s)_{s\in[0,1]} \in \mathrm{SO}(3)$ given by

$$\mathbf{R}(s) = \mathbf{R}_0 \boldsymbol{e}^{[\boldsymbol{r}(s)]}. \tag{13}$$

Substituting the expressions of $\dot{\boldsymbol{r}}$ and $\ddot{\boldsymbol{r}}$ in terms of the path parameter s

$$\dot{\boldsymbol{r}} = \dot{s}\boldsymbol{r}_s, \quad \ddot{\boldsymbol{r}} = \ddot{s}\boldsymbol{r}_s + \dot{s}^2\boldsymbol{r}_{ss} \tag{14}$$

into (5) and (6), one obtains

$$\boldsymbol{\omega} = \dot{s}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s, \tag{15}$$

$$\dot{\boldsymbol{\omega}} = \ddot{s}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s + \dot{s}^2 \left\{\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s)\right\}, \tag{16}$$

$$\boldsymbol{\tau} = \ddot{s}\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s + \dot{s}^2 \left\{\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbb{I}\mathbf{C}(\boldsymbol{r}, \boldsymbol{r}_s) + (\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \times (\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s)\right\}. \tag{17}$$

Thus, the condition $\boldsymbol{\tau}_{\min} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{\max}$ can be put in the form of (10) with

$$\boldsymbol{a} = \begin{bmatrix} \mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s \\ -\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s \end{bmatrix}, \tag{18}$$

$$\boldsymbol{b} = \begin{bmatrix} \mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbb{I}\mathbf{C}(\boldsymbol{r},\boldsymbol{r}_s) + (\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \times (\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \\ -(\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbb{I}\mathbf{C}(\boldsymbol{r},\boldsymbol{r}_s) + (\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s) \times (\mathbb{I}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s)) \end{bmatrix}, \tag{19}$$

$$\boldsymbol{c} = \begin{bmatrix} -\boldsymbol{\tau}_{\max} \\ \boldsymbol{\tau}_{\min} \end{bmatrix}. \tag{20}$$

The motion of the spacecraft is generally constrained, not only by the spacecraft itself (for example, torque limits) but also by task constraints (e.g. bounds on angular accelerations). The form (10) allows easily combining different type of constraints by concatenating the vectors $\boldsymbol{a}(s), \boldsymbol{b}(s), \boldsymbol{c}(s)$ corresponding to the different constraints. For instance, bounds on angular accelerations ($\dot{\boldsymbol{\omega}}_{\min} \leq \dot{\boldsymbol{\omega}} \leq \dot{\boldsymbol{\omega}}_{\max}$) can be put in the form of (10) by noting that $\dot{\boldsymbol{\omega}} = \ddot{s}\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_s + \dot{s}^2 \left\{\mathbf{A}(\boldsymbol{r})\boldsymbol{r}_{ss} + \mathbf{C}(\boldsymbol{r},\boldsymbol{r}_s)\right\}$.

Bounds on angular velocities can be expressed as $\boldsymbol{\omega}^\top\boldsymbol{\omega} \leq \omega_{\max}$, where $\omega_{\max}$ is a scalar. Such constraints were first addressed in [12]. The treatment was completed and implemented in [9].

For time-parameterization a given path in SE(3) under kinodynamic constraints, we separate the process into 2 steps. First, we re-time the orientation part as presented earlier. Regarding the translation part, since we interpolate two pairs $(\boldsymbol{p}_0, \boldsymbol{v}_0)$ and $(\boldsymbol{p}_1, \boldsymbol{v}_1)$ by a third degree polynomial (Section II C), the first and second derivatives are trivial to compute. Therefore, contraints on linear accelerations ($\boldsymbol{\alpha}_{\min} \leq \boldsymbol{\alpha} \leq \boldsymbol{\alpha}_{\max}$) or forces ($\boldsymbol{f}_{\min} \leq \boldsymbol{f} \leq \boldsymbol{f}_{\max}$) can be easily expressed in the form of (10).

IV. Implementation and evaluation

We experimentally evaluated our approach on some spacecraft maneuver problems. Our implementation is open-source and can be found online at [13]. All experiments were run on a machine with an Intel i7 3.40 GHz processor, 4GB RAM. Videos of the resulting simulations can be found at [14].

A. Planning fast, collision-free trajectories in SO(3) under kinodynamic constraints

Here, we present the algorithm to find a fast, collision-free, trajectory connecting two rotations $\mathbf{R}_0$ $\mathbf{R}_1$, subject to velocity and torque bounds. As mentioned earlier, we follow the plan-and-shortcut method [5].

In step 1, we use an RRT-based path planner to find a collision-free piecewise "linear" path that connects $(\mathbf{R}_0, \boldsymbol{\omega}_0)$ and $(\mathbf{R}_1, \boldsymbol{\omega}_1)$ (each "linear" segment is actually a great-circle arc in SO(3)).

Next, in step 2, we optimally time-parameterize each of the "linear" segment. To avoid discontinuities of the velocity vector at the junctions of the "linear" segments, we ensure that the velocities at the beginning and the end of each segment are zero.

In step 3, at each shortcut iteration, we select two random time instants t_0, t_1 along the trajectory, which correspond to two orientation matrices $\mathbf{R}_{t_0}, \mathbf{R}_{t_1} \in \mathrm{SO}(3)$ and two angular velocity vectors $\boldsymbol{\omega}_{t_0}, \boldsymbol{\omega}_{t_1} \in \mathbb{R}^3$. We then find the interpolation path $\mathbf{R}^*(t)_{t\in[0,T]}$, where $T = t_1 - t_0$, such that

$$\mathbf{R}^*(0) = \mathbf{R}_{t_0},\ \mathbf{R}^*(T) = \mathbf{R}_{t_1},\ \boldsymbol{\omega}^*(0) = \boldsymbol{\omega}_{t_0},\ \boldsymbol{\omega}^*(T) = \boldsymbol{\omega}_{t_1}.$$

If this path involves any collision, we discard it. Otherwise, we use TOPP to optimally time-parameterize it, while ensuring that the initial and final velocities of the shortcut portion are unchanged to avoid velocity discontinuities. If the resulting duration is smaller than T then we replace the original trajectory portion by the shortcut.

We tested the proposed algorithm on a reorientation problem for the Messenger spacecraft subject to bounds on angular velocities and torques in a cluttered environment (model downloaded from [15]), see Fig. 1. We ran 100 trials, with 200 shortcut iterations per trial. The average running time for the three steps of the algorithm are shown in Table 1. Overall, the algorithm can find fast trajectories for the satellite model in this cluttered environment in 7.42 ± 4.58s. The average number of successful shortcuts per trial was 11.12 ± 6.31. Fig. 1 shows one typical trajectory found by the algorithm. Fig. 2 shows the corresponding angular velocities and torques as functions of time. Note that, $(\omega^1, \omega^2, \omega^3)$ are the three elements of the angular velocity vector $\boldsymbol{\omega}$, while

(τ^1, τ^2, τ^3) are the three elements of the torque vector $\boldsymbol{\tau}$. One can see that, in agreement with time-optimality, at least one constraint is saturated at any time instant.

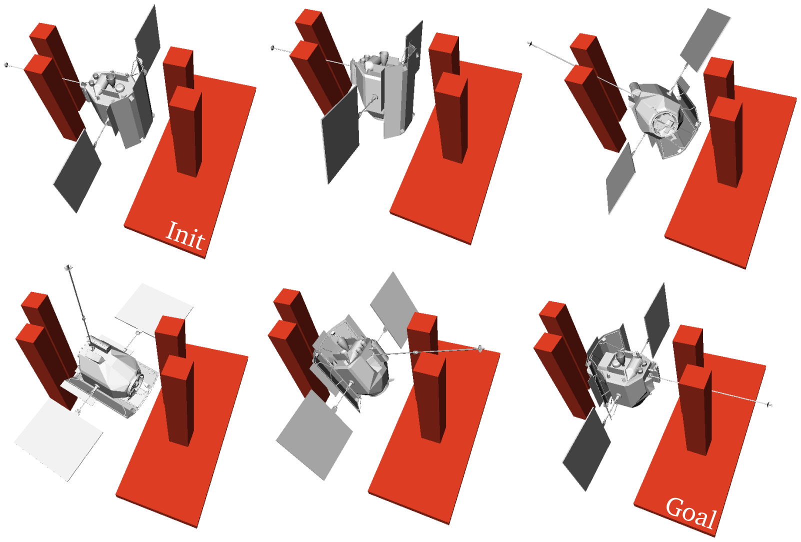

Figure 1. A fast reorientation trajectory in SO(3) for the Messenger spacecraft in a cluttered environment.

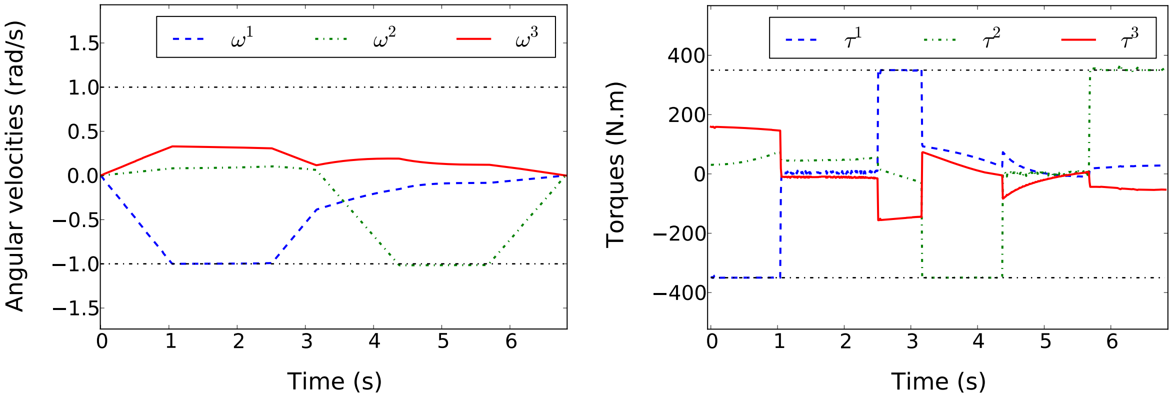

Figure 2. Angular velocities and torques of the trajectory in Fig. 1.

B. Planning fast, collision-free trajectories in SE(3) under kinodynamic constraints

We now consider the case of rigid-body motions. Figure 3 shows the environment we used to test the proposed algorithm. We ran 100 trials, with 200 shortcut iterations per trial. The average running time for the three steps of the algorithm is showed in Table 1. Overall, the algorithm can find fast trajectories for the satellite model in this cluttered environment in 32.21 ± 6.12s. The average number of successful shortcuts per trial was 18.32 ± 7.34. The total running time for problems in SE(3) was longer than in SO(3), because of the higher problem dimension and environment complexity. One typical resulting trajectory found by the algorithm is showed in Figure 3. The corresponding velocities, torques and forces are showed in Figure 4. Note that, (v^1, v^2, v^3) are the three elements of the linear velocity vector $\boldsymbol{v}$, while (f^1, f^2, f^3) are the three elements of the force vector $\boldsymbol{f}$. One can also see here that, in agreement with time-optimality, at least one constraint is saturated at any time instant.

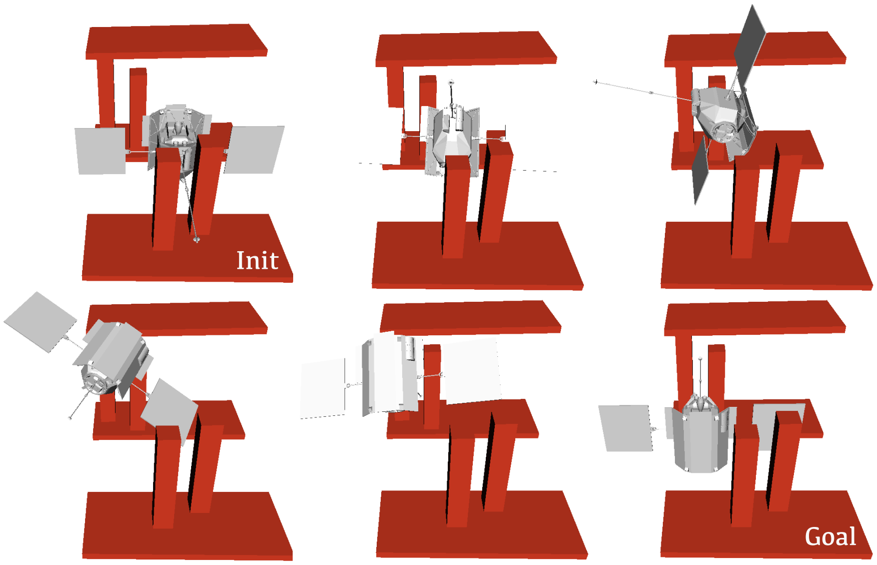

Figure 3. A fast maneuver trajectory in SE(3) for the Messenger spacecraft in a cluttered environment.

	RRT	TOPP on "linear" segments	Shortcutting	Total running time
SO(3)	1.87±3.01	0.09±0.07	5.34±2.35	7.42±4.58
SE(3)	6.18±5.23	0.32±0.20	25.54±6.53	32.21±6.12

Table 1. Average running time, in seconds, for the three steps of the plan-and-shortcut method in SO(3) and SE(3)

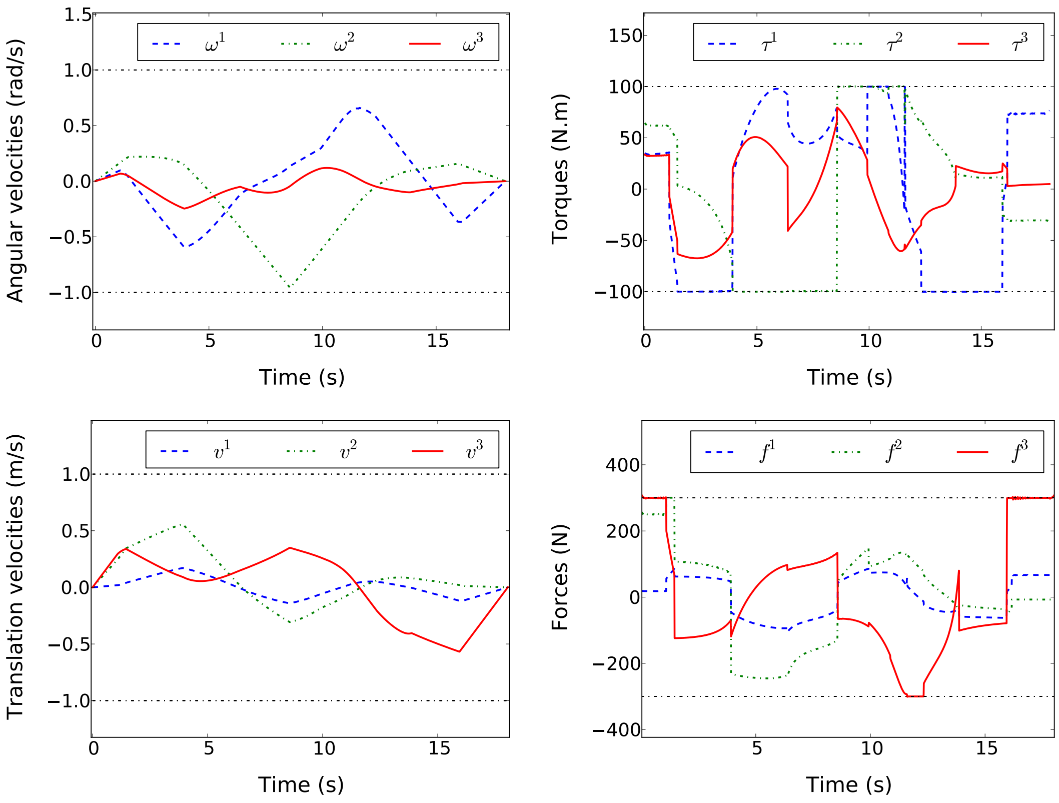

Figure 4. Velocities, torques and forces of the trajectory in Fig. 3.

V. Conclusion

We have addressed the problem of planning fast, collision-free, trajectories in the spaces of three-dimensional rotations SO(3) and three-dimensional rigid body motions SE(3) under kinodynamic constraints in cluttered environments. Our main contribution consisted in extending the classical Time-Optimal Path Parameterization (TOPP) algorithm to SO(3). We integrated the algorithm into a plan-and-shortcut pipeline to yield a complete framework (and open-source implementation) for rigid-body motion planning under kinodynamic constraints. We showed that our implementation could very efficiently find near time-optimal trajectories in a spacecraft maneuver problem. We are currently considering applications to multi-spacecraft coordination, humanoid robots and computer animation.

References

[1] LaValle, S. and Kuffner, J., "Randomized kinodynamic planning," *The International Journal of Robotics Research*, Vol. 20, No. 5, 2001, pp. 378–400.

[2] Li, F. and Bainum, P. M., "Numerical approach for solving rigid spacecraft minimum time attitude maneuvers," *Journal of Guidance, Control, and Dynamics*, Vol. 13, No. 1, 1990, pp. 38–45.

[3] Bilimoria, K. D. and Wie, B., "Time-optimal three-axis reorientation of a rigid spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 3, 1993, pp. 446–452.

[4] Bai, X. and Junkins, J. L., "New results for time-optimal three-axis reorientation of a rigid spacecraft," *Journal of guidance, control, and dynamics*, Vol. 32, No. 4, 2009, pp. 1071–1076.

[5] Geraerts, R. and Overmars, M., "Creating high-quality paths for motion planning," *The International Journal of Robotics Research*, Vol. 26, No. 8, 2007, pp. 845–863.

[6] Kuffner, J. J., "Effective sampling and distance metrics for 3D rigid body path planning," *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, Vol. 4, IEEE, 2004, pp. 3993–3998.

[7] Bobrow, J., Dubowsky, S., and Gibson, J., "Time-optimal control of robotic manipulators along specified paths," *The International Journal of Robotics Research*, Vol. 4, No. 3, 1985, pp. 3–17.

[8] Shin, K. and McKay, N., "Minimum-time control of robotic manipulators with geometric path constraints," *IEEE Transactions on Automatic Control*, Vol. 30, No. 6, 1985, pp. 531–541.

[9] Pham, Q.-C., "A general, fast, and robust implementation of the time-optimal path parameterization algorithm," *IEEE Transactions on Robotics*, Vol. 30, 2014, pp. 1533–1540.

[10] Park, F. and Ravani, B., "Bezier curves on Riemannian manifolds and Lie groups with kinematics applications," *Journal of Mechanical Design*, Vol. 117, No. 1, 1995, pp. 36–40.

[11] Park, F. C. and Ravani, B., "Smooth invariant interpolation of rotations," *ACM Transactions on Graphics (TOG)*, Vol. 16, No. 3, 1997, pp. 277–295.

[12] Zlajpah, L., "On time optimal path control of manipulators with bounded joint velocities and torques," *IEEE International Conference on Robotics and Automation*, Vol. 2, IEEE, 1996, pp. 1572–1577.

[13] Nguyen, H. and Pham, Q.-C., "TOPP for Rigid-Body Motions," https://github.com/dinhhuy2109/TOPP-SO3 [retrieved 30 Oct. 2015].

[14] Nguyen, H. and Pham, Q.-C., "TOPP for Rigid-Body Motions," https://youtu.be/heM7uxGrfVc [retrieved 30 Oct. 2015].

[15] NASA-3D-Resources, "Messenger 3D model," http://nasa3d.arc.nasa.gov/detail/eoss-messenger [retrieved 30 Oct. 2015].