J.D. Biggs, L. Colley
Geometric Attitude Motion Planning for Spacecraft with Pointing and Actuator Constraints
Journal of Guidance Control and Dynamics, Vol. 39, N. 7, 2016, p. 1669-1674
doi:10.2514/1.G001514

Geometric Attitude Motion-Planning for Spacecraft with Pointing and Actuator Constraints

James D. Biggs*,

Politecnico di Milano, 20156 Milano, Italy.

Lucy Colley† ,

University of Strathclyde, Glasgow, G1 1XJ, UK.

I. Introduction

Constrained attitude motion-planning for spacecraft is necessary in many mission scenarios, for example, when the spacecraft has a sensitive instrument which may become damaged if pointed within a certain angle of the Sun. Several approaches to the general constrained attitude motion-planning problem have been developed. For example, Hablani[1] developed attitude commands from a geometric perspective, defining exclusion (or keep-out) zones on a unit sphere and determining ideal tangential paths around these zones. McInnes[2] approached the problem by applying the potential function method. Artificial potential functions guide the satellite during the attitude maneuver and avoid violation of pointing constraints by overlaying regions of high potential around the forbidden regions. Henri et al.[3] discretized the unit sphere into a graph and an admissible path between attitude keep-out zones is found with the A* pathfinding algorithm. Frazzoli et al.[4] use randomized path planning algorithms. Here, solution paths are chosen at random and a tree of possible paths are evaluated to the target direction. The lowest cost admissible path is chosen. In contrast to almost all other methods in the literature Frazzoli et al.[4] treat the problem directly on the Special Orthogonal Group SO(3) whose elements $R(t) \in$ SO(3) satisfy the orthonormal frame constraints $R(t)^T R(t) = I_{3\times3}$ where $I_{3\times3}$ is the identity matrix and $\det(\mathrm{R(t)}) = 1$. Using SO(3) to represent the spacecraft's rotation allows pointing constraints to be imposed on multiple body-fixed directions and to represent the spacecraft's motion in a unique and singularity free way.

*Associate Professor, Department of Aerospace Science & Technology, jamesdouglas.biggs@polimi.it.
†Under-Graduate Student, Department of Mechanical & Aerospace Engineering, lucy.colley@strath.ac.uk.

This Note presents a semi-analytical method for motion-planning with pointing and dynamic constraints in two stages: Firstly, the path-planning problem with pointing constraints is addressed using parameter optimization of an analyticaly defined cost function on the virtual domain $t \in [0, 1]$, and secondly dynamic constraints are addressed on the real time domain $\tau \in [0, T_f]$ where T_f is the final time, by adjusting the speed at which the spacecraft motions along the derived path. The analytical formulation of the problem allows a simple method for re-shaping the path between two prescribed rotations so that it can avoid forbidden regions. The approach described in this Note has the advantage that it is simple to implement, deterministic (if the optimizer used is deterministic), does not require discretization or integration, is easily adjusted to satisfy actuator constraints and expressed explicitly on the Special Orthogonal Group SO(3) (as opposed to using a local parameterization such as Euler angles).

The proposed method uses a shape-based approach commonly used in problems of inverse dynamics. For example, simple functions such as polynomials are often used as basis functions for motion-planning and optimized using inverse dynamics. In particular, attitude motion-planning on the unit quaternions has been undertaken using exponentials of polynomial functions[5] to minimize time and direct normalization of polynomials[6] to design smooth feasible motions to reduce spill-over for flexible spacecraft. These shape-based methods have not been adapted to the rotation group SO(3) which has the additional complexity of shaping nine components of the rotation matrix and satisfying the orthonormal frame constraints. This Note presents an analytically defined set of paths on SO(3) that are defined in terms of free parameters that can be adjusted to match the boundary conditions and re-shape the path to avoid a forbidden region. The analytically defined paths on SO(3) are derived from an optimal control problem whose quadratic cost function is a weighted integral function of the angular velocities. Exploiting the symmetries of the problem a special case is solved analytically and used to develop the motion-planning method.

II. Previous Results on Optimal Kinematic Control on SO(3)

In this Note the paths on SO(3) are derived from a special case of an optimal kinematic control problem on SO(3).[7] Moreover, a fixed end point problem is considered where $R_0 = R(0), R_{T_f} =$

$R(T_f)$ where T_f is the fixed final time subject to the kinematic constraint

$$\dot{R}(t) = R(t)\Omega, \tag{1}$$

where $\Omega = \omega_1 A_1 + \omega_2 A_2 + \omega_3 A_3$ where $\omega_1, \omega_2, \omega_3$ are the components of the angular velocity vector $\omega = [\omega_1, \omega_2, \omega_3]^T$ and A_1, A_2, A_3 are the basis of the Lie algebra $\mathfrak{so}(3)$, the space of 3×3 skew-symmetric matrices with the additional structure of a Lie bracket defined by $[X, Y] = XY - YX$ where $X, Y \in \mathfrak{so}(3)$. The choice of the basis $A_1, A_2, A_3 \in \mathfrak{so}(3)$ is

$$A_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix}, \quad A_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \tag{2}$$

where physically A_1, A_2, A_3 define the infinitesimal rotations in the roll, pitch and yaw directions respectively. The cost function J_1 to be minimized is a quadratic function of the angular velocity components

$$J_1 = \frac{1}{2} \int_0^1 c_1\omega_1^2 + c_2\omega_2^2 + c_3\omega_3^2 dt, \tag{3}$$

where $c_1, c_2, c_3 > 0$ are arbitrary weights and $t \in [0, 1]$ is the virtual domain. This cost function allows one to define a large class of motion where the weights of the cost function can be chosen to alter the shape of the path between two prescribed boundary configurations. This problem can be solved via an application of the co-ordinate free Maximum Principle[8] to yield the necessary conditions for optimality[7]

$$\begin{aligned} \dot{M}_1 &= \tfrac{c_2 - c_3}{c_2 c_3} M_2 M_3 \\ \dot{M}_2 &= \tfrac{c_3 - c_1}{c_1 c_3} M_1 M_3 \\ \dot{M}_3 &= \tfrac{c_1 - c_2}{c_1 c_2} M_1 M_2, \end{aligned} \tag{4}$$

where M_1, M_2, M_3 are the extremal curves that define the optimal angular velocities $\omega_1^*, \omega_2^*, \omega_3^*$ via the expressions

$$\omega_1^* = M_1/c_1, \omega_2^* = M_2/c_2, \omega_3^* = M_3/c_3. \tag{5}$$

Numerical shooting has been proposed as a method to solve this optimal kinematic control problem and match the boundary conditions on the rotation.[7] However, for the rotation group extensive numerical shooting is required to match the nine components of the rotation matrix at the prescribed final time. Furthermore, using numerical shooting to solve the necessary conditions for optimality subject to the given boundary conditions[7] does not consider the possibility of the inclusion of forbidden regions or the dynamic feasibility of tracing the path. In this Note we use a special analytic solution of the extremal curves Eq. (4) to develop an attitude motion-planner that can incorprate pointing and actuator constraints through an iterative process of parameter optimization and inverse dynamics respectively.

III. Analytic Derivation of a Class of Rotational Motions

The functions that will be used to develop the motion-planner are a special solution of the necessary conditions for optimailty described by Eq. (4). In order to derive these functions the conserved quantities

$$H = \frac{1}{2}(\frac{M_1^2}{c_1} + \frac{M_2^2}{c_2} + \frac{M_3^2}{c_3}), \quad M = (M_1^2 + M_2^2 + M_3^2), \tag{6}$$

are used. There are a number of particular solutions of Eq. (4) that can be analytically defined. For example, when $c_1 = c_2 = c_3$ the extremal curves are constant. However, setting this condition means that c_1, c_2, c_3 are fixed and cannot be freely set to re-shape the path if the initial path intersects the forbidden region. Another special solution of the necessary conditions for optimality are the heteroclinic connections of Eq. (4) which can be expressed in terms of hyperbolic and trigonometric functions.[9] Setting the particular value for the weight $c_3 = \frac{M}{2H}$ then using Eq. (4) and Eq. (6) it can be shown that:

$$M_1 = A\text{sech}(\gamma t + C), \quad M_2 = B\text{sech}(\gamma t + C), \quad M_3 = D\tanh(\gamma t + C), \tag{7}$$

where $D = s_1 s_2 \sqrt{M}$, $A = s_1\sqrt{\frac{c_1(2Hc_2 - M)}{c_2 - c_1}}$, $B = s_2\sqrt{\frac{c_2(M - 2Hc_1)}{c_2 - c_1}}$ where s_1, s_2 are $s_1 = \text{sgn}(M_1(0))$, $s_2 = \text{sgn}(M_2(0))$ and where $\gamma = \sqrt{\frac{-(M-2Hc_1)(M-2Hc_2)}{c_1 c_2 M}}$ is given by substitution into Eq. (4). The extremals Eq. (4) can be expressed in vector form $\dot{L} = \nabla H \times L$ where $L = \nabla M$ and $L, \nabla H \in \mathbb{R}^3$

where ∇ is the gradient. Equivalently Eq. (4) can be expressed in Lax Pair form on the Lie group SO(3) where $\dot{L} = [L, \nabla H]$ where $L, \nabla H \in \mathfrak{so}(3)$.[8] Moreover, the solution to the Lax Pair equations are of the general form:

$$L(t) = R(t)^{-1} L(0) R(t). \tag{8}$$

A particular solution $R_p(t)$ can then be chosen such that

$$R_p(t) L(t) R_p(t)^{-1} = \sqrt{M} A_3, \tag{9}$$

then substituting

$$R_p(t) = \exp(\phi_1 A_3) \exp(\phi_2 A_2) \exp(\phi_3 A_3), \tag{10}$$

into Eq. (9) and Eq. (1) the following expressions can be obtained through algebraic manipulation:

$$\begin{aligned} \cos\phi_2 &= \tfrac{M_3}{\sqrt{M}}, \sin\phi_2 = \tfrac{\sqrt{M-M_3^2}}{\sqrt{M}} \\ \cos\phi_3 &= \tfrac{M_1}{\sqrt{M-M_3^2}}, \sin\phi_3 = \tfrac{M_2}{\sqrt{M-M_3^2}}, \end{aligned} \tag{11}$$

which for the extremal curves of hyperbolic type described by Eq. (7) are explicitly:

$$\begin{aligned} &\cos\phi_2 = s_3 \tanh(\gamma t + C), \quad \sin\phi_2 = \operatorname{sech}(\gamma t + C) \\ &\cos\phi_3 = \tfrac{A}{\sqrt{M}}, \quad \sin\phi_3 = \tfrac{B}{\sqrt{M}}. \end{aligned} \tag{12}$$

Note that $\phi_2 \in [0, \pi]$ as $\operatorname{sech}(\gamma t + C) \geq 0$. Substituting Eq. (10) into Eq. (1) and using the expressions in Eq. (12) with $c_3 = M/2H$ leads to:

$$\phi_1 = \frac{2H}{\sqrt{M}} t + \beta, \tag{13}$$

where β is a constant of integration which can be set to zero without loss of generality due to the rotational symmetry. Then substituting Eq. (13) and Eq. (12) into Eq. (10) yields the particular rotation matrix $R_p(t)$. A general rotation matrix $R(t)$ can then be expressed for any initial condition

$R(0)$ as:

$$R(t) = R(0)R_p(0)^{-1}R_p(t), \tag{14}$$

with $R_p(t) = (\mathbf{x} \quad \mathbf{y} \quad \mathbf{z})$ where the orthonormal vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$ are defined by:

$$\mathbf{x} = \frac{1}{\sqrt{M}} \begin{bmatrix} A\cos\phi_1 - Bs_1s_2\tanh(\gamma t + C)\sin\phi_1 \\ A\sin\phi_1 + Bs_1s_2\tanh(\gamma t + C)\cos\phi_1 \\ B\text{sech}(\gamma t + C) \end{bmatrix} \tag{15}$$

$$\mathbf{y} = \frac{1}{\sqrt{M}} \begin{bmatrix} -As_1s_2\tanh(\gamma t + C)\sin\phi_1 - B\cos\phi_1 \\ As_1s_2\tanh(\gamma t + C)\cos\phi_1 - B\sin\phi_1 \\ A\text{sech}(\gamma t + C) \end{bmatrix} \tag{16}$$

$$\mathbf{z} = \begin{bmatrix} \sin\phi_1\text{sech}(\gamma t + C) \\ -\cos\phi_1\text{sech}(\gamma t + C) \\ s_1s_2\tanh(\gamma t + C), \end{bmatrix} \tag{17}$$

where $C = \tanh^{-1}(M_3(0)/(s_1s_2\sqrt{M}))$. Note that these functions are expressed completely in terms of t, c_1, c_2 and the free parameters $M_1(0), M_2(0), M_3(0)$. Therefore, given a final time $t = 1$ and the prescribed weights c_1, c_2, the parameters $M_1(0), M_2(0), M_3(0)$ can be optimized to match the boundary conditions $R_0 = R(0)$ and $R_1 = R(1)$ on the virtual domain as described in the following section.

IV. Attitude Motion-Planning with Constraints

This section introduces the general motion-planning method in the presence of pointing and actuator constraints. Sub-section A describes the general path-planning procedure on the virtual domain $t \in [0, 1]$. Sub-section B extends this method to include a forbidden pointing direction. By iterating the relative weights of the cost function the path is re-shaped between the prescribed boundaries. Sub-section C considers the motion-planning problem that uses time-parameterization to augment the angular velocity along the path in SO(3) and through inverse dynamics ensure that the actuator torque limits and rate limits are respected.

A. Path-Planning Algorithm on the Rotation Group

To describe the implementation of the basic path-planner given the final time $t = 1$ and the prescribed weights c_1 and c_2, we define the vector $\mathbf{X} = [M_1(0), M_2(0), M_3(0)]^T$, where $M_1(0), M_2(0)$ and $M_3(0)$ are the initial conditions of the extremal curves. The Hamiltonian and Casimir in Eq. (6) for $c_3 = \frac{M}{2H}$ can be expressed in terms of $\mathbf{X}$ such that

$$H = \frac{1}{2}\left(\frac{M_1(0)^2}{c_1} + \frac{M_2(0)^2}{c_2} + \frac{M_3(0)^2(M_1(0)^2/c_1 + M_2(0)^2/c_2)}{M_1(0)^2 + M_2(0)^2}\right), \tag{18}$$

and

$$M = M_1(0)^2 + M_2(0)^2 + M_3(0)^2. \tag{19}$$

To match the boundary conditions $R_0 = R(0)$ and $R_1 = R(1)$, $R(0)$ is simply stated in equation (14) and defining R_d to be the desired rotation at $t = 1$ and $I_{3\times3}$ the identity matrix, then $\mathbf{X}$ is optimized such that the rotation error defined by

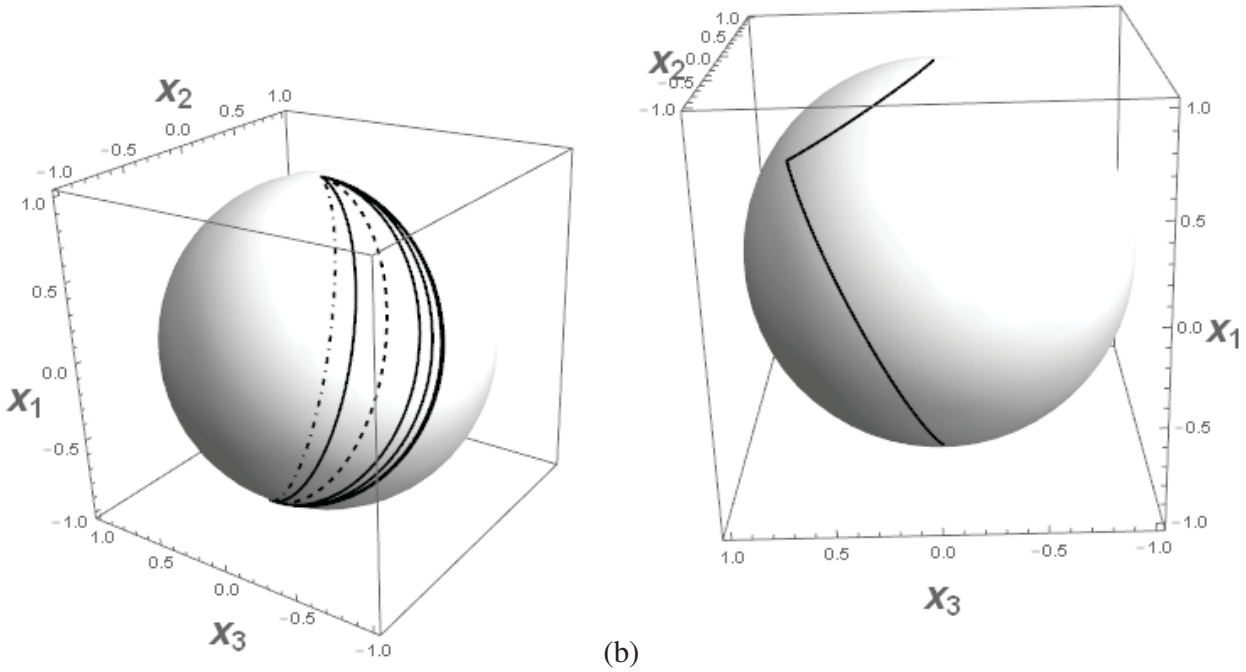
$$\|R_e\| = \mathrm{tr}[I_{3\times3} - R(1)^T R_d)] \tag{20}$$

is minimized within some prescribed error tolerance. To demonstrate this we set the boundary conditions to $R(0) = I_{3\times3}$ and

$$R_d = \begin{bmatrix} -0.782 & 0 & 0.623 \\ -0.623 & 0 & -0.782 \\ 0 & -1 & 0 \end{bmatrix}. \tag{21}$$

The error (20) is a nonlinear function of $\mathbf{X}$ and must be solved numerically. The numerical problem can be formulated as a parameter optimization problem and $\mathbf{X}$ can be chosen using an appropriate numerical method to minimize $\|R_e\|$. Initially the problem was optimized for fixed $c_2 = 1$ and c_1 was varied between 0.9 and 0.1. A number of optimizers were used to test their suitability for solving this problem. In Mathematica the Nelder-Mead (NM), Differential Evolution (DE), Simulated Annealing (SA) and Random Search (RS) minimization methods were used.[10] For

$c_1 = 0.9$ every method converged to the minimum with $\mathbf{X} = [1.71, -0.61, -1.612]$ and $\|R_e\| = 1 \times 10^{-14}$. The projection of this curve onto the unit sphere is represented by the dashed line in Fig. 1a. The CPU times in seconds were 0.36 for NM, 0.59 for DE, 0.406 for SA and 0.5 for

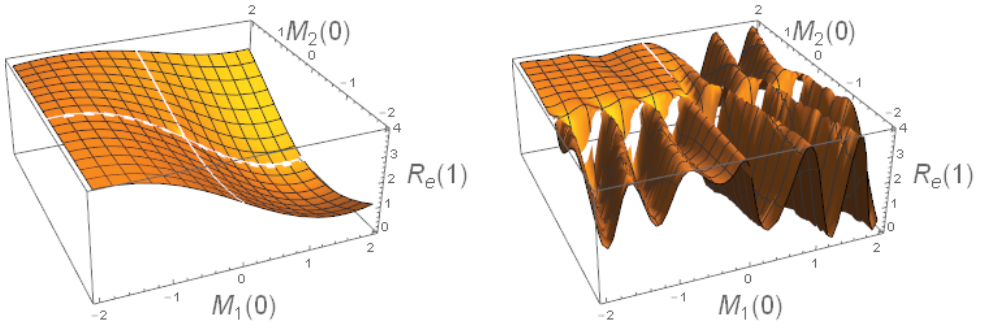


(a) (b)

Figure 1. Pointing direction projected onto the unit sphere: (a) displays a variety of paths between two fixed end-points in SO(3) for varying c_1 (b) displays a two-maneuver path between the same initial and final end points in SO(3)

RS implemented on a standard 2 GHz, dual-core PC. Varying c_1 down to 0.17 indicated similar relative performance results of the optimizers. However, below these values there was a significant variation in their performance. For, example at $c_1 = 0.1$ the following errors were achieved 0.978 for NM, 1×10^{-12} for DE, 1×10^{-14} for SA and RS. The projection of the curve for SA and RS onto the unit sphere are represented by the dot-dashed line in Fig. 1a. The reason for this discrepency between the numerical optimizers is illustrated in Fig. 2. In particular, for $c_1 = 0.9$ the value of $\|R_e\|$ as a function of $M_1(0)$ and $M_2(0)$ with the value of $M_3(0)$ given by the previous optimization is shown in Fig. 2a. In this case we can see that there is only one minima and each of the methods converge to it independently of any required initial guess. However, when $c_1 = 0.1$ it can be seen in Fig. 2b that there are many local minima and that only the RS and SA methods find the global minima of this surface. The poor performance of the NM method is due to the

fact that it is a local optimization method and is heavily dependent on the initial guess. SA and RS work by generating a population of random initial guesses in the search space and find a local minimum from each point. The best local minimum is chosen to be the solution. Thus, SA and RS, as implemented in Mathematica, were more robust to this problem involving many local minima. This was true for all other selections for R_d that were selected in the cost function (Eq. (20)). This initial analysis highlights the need to employ an optimizer within the path-planning method that is robust to minimizing functions with a large number of local minima.



(a) (b)

Figure 2. The error $\|R_e\|$ as a function of $M_1(0)$ and $M_2(0)$ with fixed $c_1, c_2, M_3(0)$: (a) corresponds to $c_2 = 1, c_1 = 0.9, M_3(0) = -1.6124$ and (b) to $c_2 = 1, c_1 = 0.1, M_3(0) = -0.116$

B. Path-Planning with Pointing Constraints

In the case that the generated geometric path intersects a forbidden region a simple iterative process can be used to augment the shape of the curve to avoid it. For example, fixing $c_2 = 1$, c_1 can be adjusted incrementally and at each step $\mathbf{X}$ can be selected to minimize the function $\|R_e\|$. Figure 1a illustrates different pointing directions projected onto the unit sphere for different values of c_1. From the left-most curve to the right most curves in Fig. 1a the values are $c_1 = 0.9, 0.5, 0.1, 1.1, 5, 50$. Any further increase in c_1 has a negligable effect on the deviation of the new path from the right-most one. All paths in Fig. 1a match the boundary condition $R_1 = R(1)$ within an error tolerance of $\|R_e\| < 1 \times 10^{-14}$. This procedure could potentially be useful for avoiding small forbidden regions of the pointing direction while making accurate single three-axis maneuvers. For example, if an initial curve for $c_1 = 0.1$ intersects a small forbidden

region then c_1 can be increased iteratively (avoiding the singularity $c_1 = c_2$) until the forbidden region is avoided. Recall, for example, that the RS method requires a CPU time of approximately 0.5 seconds to undertake a single optimization. Then, the total time to compute a path to avoid the obstacle will be the number of iterations divided by two. The number of iterations will depend on the specified increments of c_1 used. Moreover, with larger increments the total computation time will be smaller but within a specified region there will be less curves to select from. In the case that no curve avoids the forbidden region then a simple two (or more) maneuver problem can be undertaken using this approach. This is illustrated in Fig. 1b where an intermediate boundary condition $R_1 = R(1)$ is included such that the boundary conditions are $R_0 = R(0)$,$R_1 = R(1)$ and $R_{T_f} = R_d$ where $T_f = 2$. In Fig. 1b $c_1 = 0.1$ and can be fine tuned if required.

These derived paths are kinematically feasible and are defined on a virtual domain $t \in [0, 1]$. In order to address the dynamic feasibility of tracking these paths, the speed at which the path in SO(3) is traced can be adjusted through a simple time-parameterization and the torque profile checked through inverse dynamics.

C. Motion-Planning via Time-Parameterization

An inverse dynamics approach can be used to check if the control torque $\mathbf{u}(t)$ required to trace the path in SO(3) is feasible given the actuator constraints by observing the inverse dynamics

$$\mathbf{u}(t) = J\frac{d\omega}{dt} - J\omega \times \omega, \tag{22}$$

where J is the positive definite, symmetric inertia tensor. In general the planned path on the virtual domain will not be dynamically feasible and time must be parameterized to alter the speed at which the path in SO(3) is traced. Moreover, the virtual time t is expressed in a new time co-ordinate which we call real time τ defined by $t = F(\tau)$. A new rotation is then defined as $R^*(\tau) = R(F(\tau))$ and it follows by a simple application of the chain rule that the corresponding real angular velocity $\omega^*(\tau)$ can be expressed as:

$$\omega^*(\tau) = \omega(F(\tau))\frac{dF(\tau)}{d\tau}. \tag{23}$$

For simplicity we denote $\omega^*(\tau)$ by ω^* and then the required torque to induce this motion can be expressed as

$$\mathbf{u}^*(\tau) = J\frac{d\omega^*}{d\tau} - J\omega^* \times \omega^*, \tag{24}$$

where $\mathbf{u}^*(\tau) = [u_1, u_2, u_3]^T$. For example to reduce the maximum required torque we can set $t = \frac{\tau}{T_f}$ with $\tau \in [0, T_f]$. The final time T_f can then be increased until the required maximum torque falls within the maximum torque range of the actuators. Note that the geometric path will remain unchanged during time-parameterization and thus will still avoid the original forbidden region without the need to repeat the path-planning procedure.

Furthermore, we note that in general the angular velocities (Eq. (5)) corresponding to the geometric path on SO(3) will be non-zero at the end points of the motion and the actuators will need to supply an instantaneous torque at these boundaries, for example, natural rigid-body motions in general describe motions with non-zero angular velocities at the boundaries and these can be matched using instantaneous torque (an impulse).[11] Such a situation is closely realizable with jet thrusters. However, in the case where an instantaneous torque cannot be supplied, or it is not desirable to do so, the parameterization function should be carefully chosen. Assuming that we are working on the real time (in seconds) domain $\tau \in [0, T_f]$, then real time can be parameterized again such that $\tau = F(\kappa)$ where $\kappa \in [0, T_f]$. For example, to avoid the need for instantaneous torques at the end points of the motion one could choose the parameterization $\frac{dF(\kappa)}{d\kappa} = 1 - \cos\left(\frac{2\pi\kappa}{T_f}\right)$ to guarentee a rest-to-rest attitude motion such that

$$F(\kappa) = \kappa - \left(\frac{T_f}{2\pi}\right)\sin\left(\frac{2\pi\kappa}{T_f}\right) \tag{25}$$

the angular velocity components Eq. (5) in real time $\kappa \in [0, T_f]$ will then be

$$\begin{aligned}
\omega_1^* &= \tfrac{A}{T_f c_1}\mathrm{sech}(\gamma(\tfrac{\kappa}{T_f} - \left(\tfrac{1}{2\pi}\right)\sin\left(\tfrac{2\pi\kappa}{T_f}\right)) + C)(1 - \cos\left(\tfrac{2\pi\kappa}{T_f}\right)),\\
\omega_2^* &= \tfrac{B}{T_f c_2}\mathrm{sech}(\gamma(\tfrac{\kappa}{T_f} - \left(\tfrac{1}{2\pi}\right)\sin\left(\tfrac{2\pi\kappa}{T_f}\right)) + C)(1 - \cos\left(\tfrac{2\pi\kappa}{T_f}\right)),\\
\omega_3^* &= s_3\tfrac{\sqrt{M}}{T_f c_3}\tanh(\gamma(\tfrac{\kappa}{T_f} - \left(\tfrac{1}{2\pi}\right)\sin\left(\tfrac{2\pi\kappa}{T_f}\right)) + C)(1 - \cos\left(\tfrac{2\pi\kappa}{T_f}\right)).
\end{aligned} \tag{26}$$

To demonstrate the effect of time-parameterization we select the initial path on $t \in [0, 1]$ with

$c_1 = 0.9, c_2 = 1$ and $\mathbf{X} = [1.71, -0.61, -1.612]$. The case $T_f = 1$ s is equivalent to the initial problem on the virtual domain. For the final time $T_f = 1$ s, the effect of the time-parameterization in Eq. (25) is demonstrated so that the original motion on the virtual domain can be compared directly with the parameterized path. The corresponding pointing direction is plotted against time along with their angular velocities as shown in Fig. 3. This has the effect of smoothing the angular velocities at the end-points and thus eliminating the requirement to use jet thrusters to match the boundary conditions.

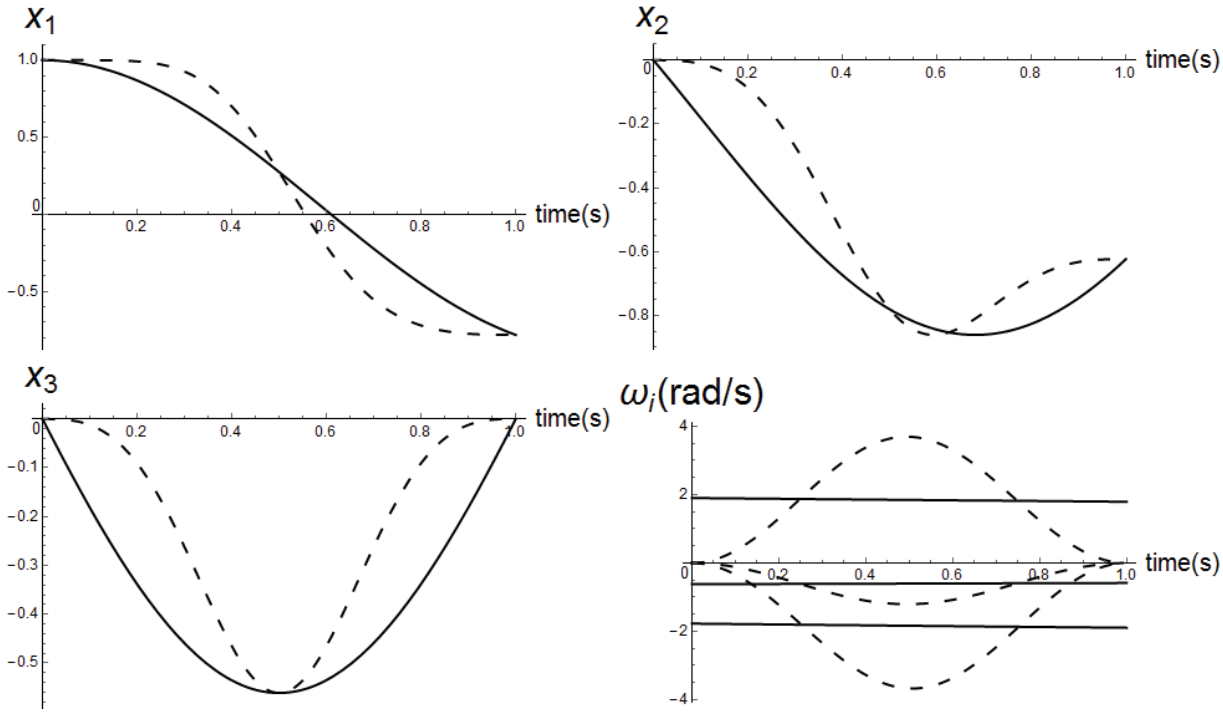


Figure 3. Pointing direction (the first normalized column vector of the rotation matrix) and angular velocity against time; with parameterization of (25) (dashed-line) and without parameterization (solid-line)

The final time T_f is then adjusted to ensure that the maximum required torque is less than the maximum torque that can be produced by the actuators. For example, for a micro-satellite we assume the principal moments of inertia $I_1 = 19, I_2 = 19.5, I_3 = 2.6$ (kg-m^2) and the products of inertia to be zero in Eq. (24) with the angular velocity Eq. (26) and a maximum torque of 0.1 N-m for micro-satellite reaction wheels. If we set $T_f = 80$ s to give the spacecraft a reasonable amount of time to perform the motion the corresponding angular velocity and torque profile is shown in Fig. 4 and are clearly feasible within the actuator constraints.

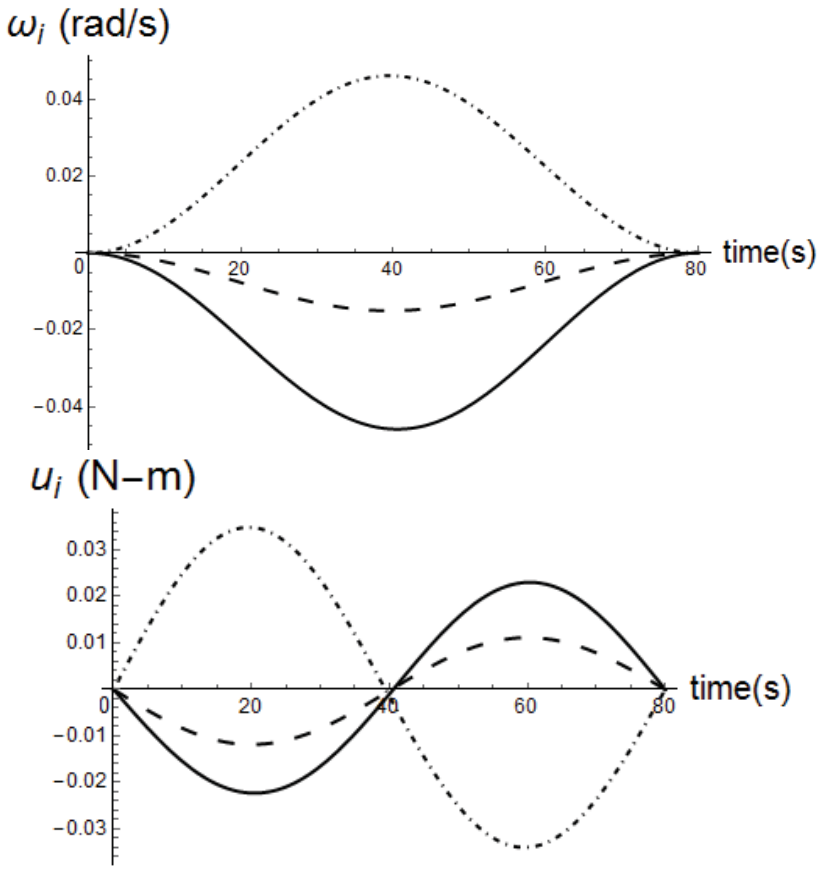


Figure 4. Final angular velocity and torque components over time: Dot - Dashed line corresponds to the angular velocity ω_1 and the torque component u_1, the dashed line to ω_2 and u_2 and the solid line to ω_3 and u_3

V. Conclusion

This Note presents a geometric method for path-planning on SO(3). The method uses an analytic solution of an optimal kinematic control problem whose cost function is an integral of a quadratic function of the angular velocities with arbitrary weights. For prescribed relative weights and initial configuration, three parameters of the analytically defined rotation matrix can be optimized to match the boundary condition on the prescribed final configuration. In addition the relative weights of the cost function can be iteratively changed and optimized at each stage in order to reshape the path between the prescribed boundary conditions. This process is potentially useful for avoiding small forbidden regions in multiple axis if the original paths intersects it. Furthermore, large forbidden regions could be avoided by introducing intermediate boundary conditions and interpolating between them. Due to the analytic nature of the paths the time can be parameterized

to account for limit and rate limit constraints of the actuators. The method does, however, rely on the use of a robust parameter optimization method that can accurately detect the global minima in the presence of many local minima inherent in the problem for some particular weights of the cost function.

VI. Acknowledgments

This work has been supported by the Marie Curie fellowship PITN-GA-2011-289240 AstroNet-II.

References

[1] Hablani, H. B.,"Attitude Commands Avoiding Bright Objects and Maintaining Communication with Ground Station," *Journal of Guidance, Control and Dynamics*, Vol. 22, No. 6, 1999, pp. 759-767.
doi: 10.2514/2.4469

[2] McInnes, C. R., "Large Angle Slew Manoeuvres with Autonomous Sun Vector Avoidance," *Journal of Guidance, Control and Dynamics*, Vol. 17, No. 4, 1994, pp. 875-877.
doi: 10.2514/3.21283

[3] Kjellberg, H. C., and Lightsey, E. G., "Discretized Constrained Attitude Pathfinding and Control for Satellites," *Journal of Guidance, Control, and Dynamics*, Vol. 36, No. 5 2013, pp. 1301-1309.
doi: 10.2514/1.60189

[4] Frazzoli, E., Dahleh, M. A., Feron, E., and Kornfeld, R., "A Randomized Attitude Slew Planning Algorithm for Autonomous Spacecraft, *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, Montreal, Quebec, 2001.

[5] Boyarko, G., Romano, M., and Yakimenko, O., "Time-Optimal Reorientation of a Spacecraft Using an Inverse Dynamics Optimization Method," *Journal of Guidance, Control, and Dynamics*, Vol. 34, No. 4, 2011, pp. 1197-1208.
doi: 10.2514/1.4944

[6] Caubet, A., and Biggs, J., "A motion planning method for spacecraft attitude maneuvers using single polynomials," *AAS/AIAA Astrodynamics Specialist Conference*, At Vail, CO, August 2015.

[7] Spindler, K., “Optimal attitude control of a rigid body,” *Applied Mathematics and Optimization*, No. 34, 1996, pp. 79-90.
doi:10.1007/BF01182474

[8] Jurdjevic, V., *Geometric Control Theory*. Advanced Studies in Mathematics, Cambridge University Press, 52, 1997. pp. 362-401.

[9] Biggs, J., Maclean, C., and Caubet, A., “Heteroclinic optimal control solutions for attitude motion-planning,” In: *Australian Control Conference*, Perth. 2013. pp. 218 - 223
doi: 10.1109/AUCC.2013.6697276

[10] Wolfram Research, Inc., *Mathematica*, Version 10.2, Champaign, IL, 2015

[11] Maclean, C., Pagnozzi, D., Biggs, J., ”Planning natural repointing manoeuvres for nano-spacecraft,” *IEEE Transactions on Aerospace and Electronic Systems*, 50 (3). pp. 2129-2145.
doi:10.1109/TAES.2014.130417.