applied sciences

MDPI

Article

Safety-Aware Optimal Attitude Pointing for Low-Thrust Satellites

Helen Henninger [1,*], James Biggs [2] and Karl von Ellenrieder [1]

[1] Facolta di Scienze e Tecnologie, Free University of Bozen-Bolzano, Piazza Universita' 5, 39100 Bolzano, Italy; Karl.vonEllenrieder@unibz.it
[2] Dipartimento di Scienze e Tecnologie Aerospaziali, Politecnico di Milano, Via La Masa 34, 20156 Milano, Italy; jamesdouglas.biggs@polimi.it
* Correspondence: HelenClare.Henninger@unibz.it

Abstract: In geostationary orbit, long eclipses and the seasonal variations in the direction and intensity of the solar input can cause damage to sensitive equipment during attitude maneuvers, which may inadvertently point the equipment towards the Sun. The requirement that transmitting and receiving antennae remain pointed towards the Earth creates further restrictions to pointing directions. The aim of the study is to construct a novel geometric and reinforcement-learning-based method to determine attitude guidance maneuvers that maintain the equipment in safe and operational orientations throughout an attitude maneuver. The attitude trajectory is computed numerically using the geometric framing of Pontryagin's maximum principle applied to the vehicle kinematics using the global matrix Lie group representation on $SO(3)$, and the angular velocities are shaped using free parameters. The values of these free parameters are determined by a reinforcement learning algorithm to avoid the forbidden areas while maintaining the pointing in operational areas (modeled as subsets of the two-sphere of all possible pointing directions of a particular axis). The method is applied to a model geosynchronous satellite and demonstrated in a simulation.

Keywords: attitude control; satellites; geometric control; reinforcement learning; constrained motion planning

check for updates

Citation: Henninger, H.; Biggs, J.; von Ellenrieder, K. Safety-Aware Optimal Attitude Pointing for Low-Thrust Satellites. *Appl. Sci.* **2021**, *11*, 3002. https://doi.org/10.3390/app11073002

Academic Editors: Silvio Cocuzza, Alberto Doria and Benedetto Allotta

Received: 24 February 2021
Accepted: 24 March 2021
Published: 27 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Earth-pointing satellites are sometimes required to perform attitude maneuvers from one direction in space to another in such a way that, en route, payloads such as telescopes do not see bright objects such as the Sun or Moon, while maintaining the communication with the ground via the beam pencil of a communications antenna. To respect these pointing constraints, we require the satellite orientation to respect a particular *pointing set*. The pointing set is a subset of the satellite's configuration set that guarantees safety (in the case of bright spot avoidance) or achieves a mission outcome (in the case of the pointing of communications antennae). The parallel concept of a safe set in robot guidance traditionally implies solely the robot's $[x, y, z]$ position, but in this paper, we extend this idea to orientation.

Several approaches to the overall constrained attitude motion planning problem are developed. For instance, Reference [1] developed attitude commands from a geometrical perspective, defining exclusion (or keep-out) zones on a unit sphere and determining ideal tangential paths around these zones. In [2], the problem was approached by applying the potential function method. Artificial potential functions guide the satellite during the attitude maneuver and avoid the violation of pointing constraints by overlaying regions of high potential around the forbidden regions. In [3], the unit sphere was discretized into a graph, and an admissible path between attitude keep-out zones was found with the A^* pathfinding algorithm. Randomized path planning algorithms were used by [4]. Here, solution paths were chosen in a random way, and a tree of possible paths was evaluated in the target direction. The all-time low-cost admissible path was chosen. In contrast to

most other methods in the literature, Reference [4] treated the matter directly on the special orthogonal group $SO(3)$. Using $SO(3)$ to represent the spacecraft's rotation allows pointing constraints to be imposed on multiple body-fixed directions. Biggs [5] found reference motions for reorienting a spinning satellite. The path-planning problem with pointing constraints was addressed using optimization of an analytically defined cost function, which includes these parameters. The analytical expression was expressed explicitly on the special orthogonal group $SO(3)$ (instead of employing a local parameterization like Euler angles). However, no explicit way of determining the parameters to respect the pointing constraints was presented.

One way of determining the parameters required is to preserve the pointing set is to "teach" the satellite's guidance system via a learning algorithm to respect the boundaries of the pointing set. In reinforcement learning (RL), an agent (the vehicle) maps observations (states) to actions via a policy function, which can generally be expressed as a neural network to maximize its reward, a scalar feedback signal. The goal of RL is to determine actions that generate a desired system behavior.

The application of RL in obstacle avoidance is not new; the study [6] used an artificial potential field to design the positive rewards of the RL algorithm. According to the different requirements of the task, rewards were modified in the training process to obtain different paths. A control strategy with learning capabilities for unknown environments was demonstrated in [7] using RL with only sparse reward information. A discrete coding of the input space using a neural network structure was presented, which enabled a faster and more efficient convergence of the RL process. Preservation of a safe set is the fundamental concept of motion planning, which avoids catastrophic outcomes by maintaining motion within certain constraints defined by safety. Collision avoidance is the most commonly applied example of this kind of motion planning. A method was described for generating plan-like, reflexive obstacle avoidance behavior in a mobile robot in [8]. The paper showed experiments using a simulated vehicle with a primitive range sensor. As the vehicle explored its surroundings, it adapted its responses to sensory stimuli so as to minimize the negative reinforcement arising from collisions. The problem of a mobile robot learning to navigate an unknown environment while avoiding collisions was considered in [9]: an uncertainty-aware model-based learning algorithm that estimates the probability of collision together with a statistical estimate of uncertainty. In order to learn collision avoidance, the robot must experience collisions at training time. The vehicle accelerates in velocity in settings where it has high confidence. In [10], a tailored design of state and action spaces and a dueling deep Q-network (a deep RL network for autonomous navigation and obstacle avoidance) were proposed for unmanned surface vehicles. In [11], a method for enabling a quadrotor equipped with a monocular camera to autonomously avoid collisions with obstacles in unstructured environments was given. They proposed a deep RL-based method for obstacle avoidance that uses recurrent neural networks with temporal attention and provides better results compared to prior works in terms of distance covered without collisions. The paper [12] focused on the application of RL to obstacle avoidance in dynamic environments for the control of a mobile robot. An intelligent controller was proposed by integrating RL with the behavior-based control architecture and applied to the obstacle avoidance. The problem in which many autonomous systems are designed with a strong reliance on black box predictions from deep neural networks, which tend to be overconfident about unseen data, was considered in [13]. The importance of predictions that are robust to this is evident for safety-critical applications, such as collisions. The paper used an RL framework to form uncertainty-aware navigation around pedestrians.

Our aim in this paper is to use RL in the planning layer of the architecture, to produce a trajectory between two orientations that maintains the spacecraft in the pointing set throughout the maneuver. The approach combines and extends that described in [5]; we describe the system kinematics on $SO(3)$, and then formulate the problem as an optimal kinematic control problem on a matrix Lie group. An application of Pontryagin's maximum

principle (PMP) gives the necessary conditions for optimality in the form of extremal equations, which as in [5] contain a number of unknown parameters. RL is applied to augment the shape of the curve to avoid a forbidden region, by selecting the value of the parameter such that the path remains in the pointing set. These amended parameter values are then used to construct a new trajectory that maintains the satellite in the pointing set.

2. Methods

We considered the case of a satellite in geostationary orbit equipped with a communications antenna with a relatively small beam pencil that needed to be pointed towards the communications ground-station at all times and a sensitive on-board telescope that needed to avoid a bright celestial object. Expressing the attitude kinematics as $R(t) \in SO(3)$, we have:

$$\frac{dR(t)}{dt} = R(t)(\omega_1(t)A_1 + \omega_2(t)A_2 + \omega_3(t)A_3) \tag{1}$$

where $R(t) \in SO(3)$, and:

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

is the standard basis of the Lie algebra $\mathfrak{so}(3)$. We developed a three-step method that solves the attitude guidance problem of determining the desired $\omega_1, \omega_2, \omega_3$ and solving Equation (1) numerically such that the boundary conditions:

$$R(0) = R_0 \quad R(t_f) = R_d \tag{3}$$

(where R_0 is the prescribed initial attitude and R_d the desired final attitude) are satisfied.

2.1. Bright Spot Avoidance

Figure 1 shows the local vertical and local horizontal (LVLH) reference frame, which has its origin at the satellite center of mass and has orthogonal axes (denoted by x, y, z), with the z-axis from the origin of the frame towards the Earth center, the x-axis in the orbital plane in the direction of orbital motion and perpendicular to the z-axis, and the y-axis completing the frame by the right-hand rule. A second, body-fixed, set of axes (denoted by X, Y, Z) and also centered at the satellite center of mass move with the body, and the spacecraft attitude is then defined as the relative angle from the local-level coordinates to the body frame.

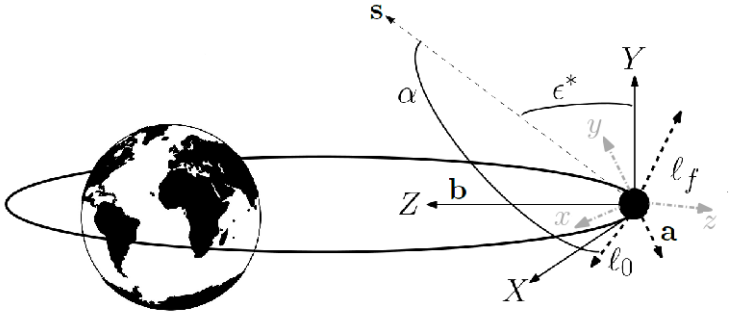


Figure 1. The inertial LVLH frame (X,Y,Z) and the body fixed frame ((x,y,z) shown in grey) with the telescope boresight initial and final directions ℓ_0 and ℓ_f.

The line-of-sight unit vector ℓ_0 is along the initial direction of the boresight, and ℓ_f denotes its final desired direction. The figure depicts a bright object to be avoided by the telescope. The centroid of the bright point, **s** (from the Sun), is specified by an angle (azimuth) α^* in the slew-plane $\ell_0 - \ell_f$ and an elevation angle ϵ^* that measures its angle from the Y-axis. Although the Sun is moving with respect to the Earth-centered frame, we

may assume for our purposes that since the maneuver time is significantly shorter than a solar day, the Sun's position is fixed in this frame. The celestial object is avoided by a minimum specified angle denoted ϵ_{spec}; that is, the dot product of the Sun's position vector **s** with the boresight axis position **a** must fall within the set:

$$\mathcal{S}_1 = \{\mathbf{v} \in \mathbb{R}^3 \mid \mathbf{s} \bullet \mathbf{v} \leq \sin \epsilon_{spec},\ \|\mathbf{v}\| = 1\}. \tag{4}$$

This set describes a region bounded by a cone in R^3, and the intersection of this cone with the sphere $\mathbb{S}^1$ describes the boundary of the forbidden region of the orientation of the boresight axis. In this paper, the boresight axis is denoted by $\mathbf{a} = [2, 2, 1]$, and the trajectory traced by the boresight is then be given by:

$$\mathbf{a}(t) = R(t) \cdot \mathbf{a}. \tag{5}$$

This trajectory $\mathbf{a}(t)$ must lie within the set $\mathcal{S}_1$ at all times.

2.2. Communications Pointing Maintenance

In addition to the telescope in the previous section, we considered that the spacecraft has a communications antenna with a beam pencil centered on the Z-axis with pitch ϕ. We wished to ensure that the beam pencil does not pass out of contact with a ground station directly at the nadir; i.e., the z-axis in the body-fixed frame must remain within $\sin 2\phi$ of the Z-axis in the LVLH frame. This criterion requires that the dot product of the antenna axis must fall within the set defined by the inequality:

$$\mathcal{S}_2 = \{\mathbf{v} \in \mathbb{R}^3 \mid [0, 0, 1] \bullet \mathbf{v} \leq sin2\phi,\ \|\mathbf{v}\| = 1\}, \tag{6}$$

The boundary of this region is a cone in $\mathbb{R}^3$, and the intersection of this cone with the sphere $\mathbb{S}^2$ describes the boundary of the region in which the antenna axis must remain. In this paper, the antenna axis is denoted by $\mathbf{b} = [0, 0, 1]$, and the trajectory traced by the antenna is then given by:

$$\mathbf{b}(t) = R(t) \cdot \mathbf{b}. \tag{7}$$

This trajectory $\mathbf{b}(t)$ must lie within the set $\mathcal{S}_2$ at all times.

In what follows, the maintenance of the pointing set means that:

$$\mathbf{b}(t) \in \mathcal{S}_2 \quad \wedge \quad \mathbf{a}(t) \notin \mathcal{S}_1 \tag{8}$$

for all $t \in [0, 1]$.

The boundaries of these two sets are shown on the sphere in Figure 2.

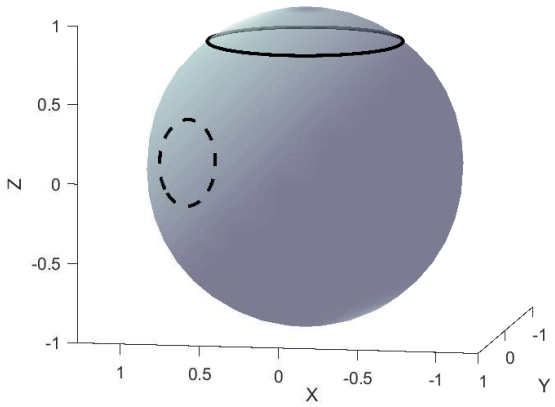


Figure 2. The boundaries of the pointing set. The bright spot set that needs to be avoided ($\mathcal{S}_2$) is the interior of the sphere with the dashed boundary, while the set where the antenna needs to remain ($\mathcal{S}_1$) is the interior of the sphere with the solid boundary.

2.3. Trajectory Generation

We determined the solution $g(t) \in G$ of the left-invariant differential system (1) that satisfies the boundary conditions:

$$R(0) = R_0, \quad R(T_f) = R_F \tag{9}$$

while minimizing the expression:

$$\mathcal{J} = \frac{1}{2}\int_0^{T_f} (c_1\omega_1^2 + c_2\omega_2^2 + c_3\omega_3^2)dt. \tag{10}$$

This cost $\mathcal{J}$ in Equation (10) allows one to define a large class of trajectories where the weights of the cost function can be chosen to alter the shape of the path between two prescribed boundary configurations. A standard geometric control theory approach was used to generate these trajectories to minimize a cost $\mathcal{J}$; for context, as for the nomenclature used in this section, the reader is directed to Appendix A, as well as any reader unfamiliar with geometric control theory.

As stated in Appendix A, to determine the optimal trajectory minimizing the cost $\mathcal{J}$, we required integrating the kinematic equations (Equation (1)) and the extremal equations (Equation (A2)) simultaneously in the real-time interval $[0, T_f]$. To do this, we used numerical integration in the following way: setting:

$$P_1 = M_1, \; P_2 = M_2, \; P_3 = M_3 \tag{11}$$

then by the PMP, the optimal values of the velocities are given by:

$$\omega_1 = P_1/c_1, \quad \omega_2 = P_2/c_2, \quad \omega_3 = P_3/c_3 \tag{12}$$

and for g in $SO(3)$, the solution of the kinematic equation, we may assign:

$$\begin{bmatrix} P_4 & P_5 & P_6 \\ P_7 & P_8 & P_9 \\ P_{10} & P_{11} & P_{12} \end{bmatrix} = \begin{bmatrix} g_{11} & g_{12} & g_{12} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix}. \tag{13}$$

Using the fact that from (1):

$$\dot{R} = R(u_1A_1 + u_2A_2 + u_3A_3), \tag{14}$$

(where $R(t)$ denotes the desired trajectory), we may express the kinematic and extremal equations together in vector form:

$$\begin{aligned} \frac{dP_1}{dt} &= -P_2P_3/c_2 + P_2P_3/c_3 \\ \frac{dP_2}{dt} &= P_1P_3/c_1 - P_1P_3/c_3 \\ \frac{dP_3}{dt} &= -P_1P_2/c_1 + P_1P_2/c_2 \\ \frac{dP_4}{dt} &= P_4 \\ &\vdots \\ \frac{dP_{12}}{dt} &= P_{12}. \end{aligned} \tag{15}$$

The initial value is given by:

$$P_0 = [\lambda_0, R_0^{11}, R_0^{12}, \ldots, R_0^{33}]^T \tag{16}$$

where R_0^{ij} are the i, j components of the matrix R_0 in Equation (9), and $\lambda_0 = [p_1^0, \ldots, p_3^0, M_1^0, \ldots, M_3^0]$, the unknown initial vector for the extremal equations. Integrating (15) starting from P_0 with an initial guess for λ_0 and rearranging the solution $P_7, \ldots, P_{22}$ using the function:

$$\Phi : \mathbb{R}^{16} \to \mathbb{R}^{4\times4}, \quad \Phi([x_1, x_2, \ldots, x_{16}]) = \begin{bmatrix} x_1 & x_4 & x_7 \\ x_2 & x_5 & x_8 \\ x_3 & x_6 & x_9 \end{bmatrix} \tag{17}$$

we then obtained the $\Phi([P_4(t), \ldots, P_{12}(t)]) = R(t)$, the matrix that gives the optimal solution in the configuration space.

Note: we are aware that for integrating Equation (1) developed on a matrix Lie group, symplectic integration is preferred over a numerical integration of (15). However, we calculated the deviation from the Lie group over the time interval $[0, T_f]$ used in the simulations presented in Section 2 by calculating the norm of the difference of $R(t)R(t)^T$ and the identity matrix over the whole of the trajectory to determine if the structure of $SO(3)$ was preserved. Figure 3 shows that the difference is of the order of magnitude of 10^{-13}, while Figures 4 and 5 also strongly suggest this result since the curves remain on the surface of the sphere throughout the maneuver.

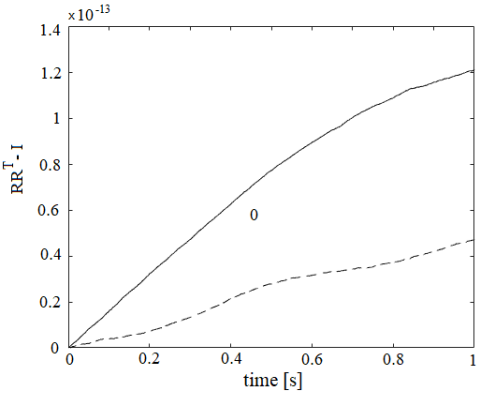
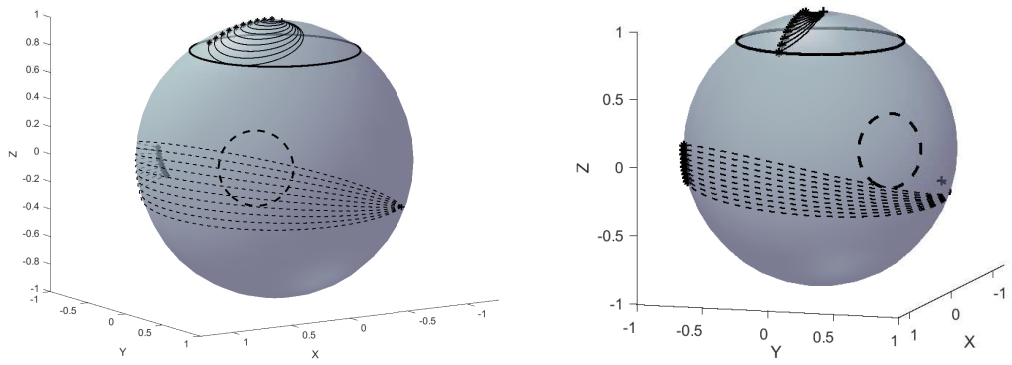


Figure 3. Frobenius norm of the difference between $R(t)R(t)^T$ and the identity matrix for the trajectories shown in Figure 4a (dashed) and Figure 5a (solid).



(**a**) (**b**)

Figure 4. Correcting for a bright spot. The initial trajectories constructed between the points (**a**) in determining the pointing-constraint trajectory described by the parameters in Path 1 and the same trajectories after choosing the c_i values using reinforcement learning (**b**). The initial points are indicted by $+$ and the final points by $*$ (initial and final points from Equation (22).

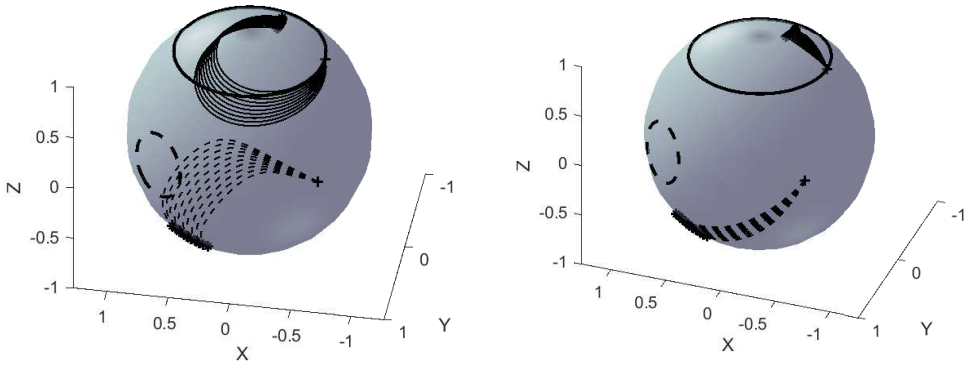


Figure 5. Correcting for antenna pointing. The initial trajectories constructed between the points (**a**) and the same trajectories after choosing the c_i values using reinforcement learning (**b**). The initial points are indicated by + and the final points by $*$ (initial and final points from Equation (23)).

The following steps were then used to solve the optimal control problem of moving the vehicle from the initial to the final configuration:

- Step 1: Determine the solution $R(1, \lambda_0, c_i^0)$ of (1) that minimizes the cost (10) by integrating the equations using (15) in terms of a chosen value $c_i^0 = [1, 1, 1]$ the unknown boundary conditions λ_0 on the (imaginary) time domain $[0, 1]$.
- Step 2: Construct the "error matrix" $\widetilde{R}(1, \lambda_0, c_i^0) = R(1, \lambda_0, c_i^0) - R_D$ and the function:

$$S(\lambda_0) = \|\widetilde{R}(\lambda_0, c_i^0)\|$$

  for the fixed choice of c_i^0.
- Step 3: Find λ_0 that minimizes $S(\lambda_0)$ for the particular choice of $c_i^0, T_f = 1$, using a local minimizer.

After Step 3 was completed, an initial trajectory was determined in terms of the particular choice of variables c_i^0. We went on to apply RL to allow the vehicle to reactively choose a new optimal path that avoided the obstacles by adapting the c_i values away from the chosen initial values c_i^0. The initial choice of one for each c_i value was made, which corresponded in the case that energy was being evaluated to equal inertia values for each axis.

2.4. Reinforcement Learning

In this subsection, we specify the observation and reward functions that we used in conjunction with MATLAB's Reinforcement Learning Toolbox software. In this software, the policy that selects the best action to take is represented in the form of a neural network. In this case, the action is an amendment of the values c_1^0, c_2^0, c_3^0; the action output is added to the c_i^0 values selected in Step 1 in order to choose a new path such that the obstacle is avoided.

The observation sent to the RL block consists of detecting the boundary of the sets $\mathcal{S}_1$ and $\mathcal{S}_2$, and the reward function $r(t)$ is set up in order to ensure that the condition in (8) is maintained. In particular, when the antenna intersects the boundary of the pointing set $\mathcal{S}_2$, the condition (denoting $\mathbf{b} = [x_b, y_b, z_b]$) is expressed by a logical value:

$$obs_2(t) = \begin{cases} 1 & \text{if } x_b^2 + y_b^2 >= r_2^2 \text{ and } z_b >= 0.791 \\ 0 & \end{cases} \tag{18}$$

where $r_2 = \sin(2\phi)$ and $\phi = 0.175$. Since the antenna axis is taken as $[0, 0, 1]$, the boundary of the region is clearly a plane parallel to the x-y-axis, and $z_b = 0.791$ where it intersects the z-axis arises by choosing $\phi = 0.174$; therefore, the coordinates of the boundary projected

on the plane) are given by $X_b = \sin(2\phi)\cos(\theta)$, $Y_b = \sin(2\phi)\sin(\theta)$ (for $\theta \in [0, 2\pi]$), and z_b can be calculated from the inverse stereographic projection through the south pole,

$$z_b = \frac{1 - X_b^2 - Y_b^2}{1 + X_b^2 + Y_b^2}.$$

To determine the boundary of the set $\mathcal{S}_1$, we used the projection from sphere to plane,

$$[X, Y] = \left(\frac{x}{1-z}, \frac{y}{1-z}\right)$$

to map the vector $\mathbf{a} = [x_a, y_a, z_a]$ to the plane where it has the form $[X_a, Y_a]$ and the Sun vector $\mathbf{s} = [x_s, y_s, z_s]$ to the plane where it has the form $[X_s, Y_s]$, and we calculated the value:

$$A = (X_a - X_s)^2 + (Y_a - Y_s)^2.$$

The intersection of $\mathbf{a}$ with the boundary of the set $\mathcal{S}_1$ is then described by:

$$obs_1(t) = \begin{cases} 1 & \text{if } A \leq r_1^2 \\ 0 & \end{cases} \tag{19}$$

where $r_1 = 0.25$. The reward is correspondingly calculated as:

$$r(t) = \begin{cases} -10 & \text{if } obs_1(t) = 1 \ \vee \ obs_2(t) = 1 \\ 10 & \text{if } obs_1(t) = 0 \ \wedge \ obs_2(t) = 0. \end{cases} \tag{20}$$

The RL-block then uses a neural network to calculate the three action outputs based on the values of the reward and observation functions. We chose the simple reward function (20) to avoid the issues of "reward shaping", which may lead the agent to make unnecessarily elaborate maneuvers to increase the reward. A drawback of the Reinforcement Learning Toolbox is that the neural network defining the policy function component is a "black box", and so, we had no real insight into the scaling involved. In this case, it was noticed that the policy was outputting actions to the order of 10^{-3}, so it was required to introduce a scaling of the actions to produce reasonable results. After trial and error, we determined that a scaling of 100 was efficient. Some further insight into the neural networks used in the policy function could have assisted in choosing a scaling term less heuristically.

2.5. Choosing a New Optimal Path Using Reinforcement Learning

After the initial guess c_i, the values c_i^0 were amended to the obstacle-avoidance values chosen by RL (which we denote by c_i^{rl}, which as we noted in the previous section was determined by $c_i^{rl} = c_i^0 + \text{action}$), and we constructed a new trajectory that avoided the obstacle. We followed Steps (1)–(3)with the following amendments in order to construct the final trajectory:

- Step (1a): Determine the solution $R(1, \lambda_0, c_i^{rl})$ of (1) that minimizes the cost (10) using (15) in terms of the unknown boundary conditions λ_0 on the imaginary time domain $[0, 1]$.
- Step (2a): Construct the "error matrix" $\widetilde{R}(1, \lambda_0, c_i^{rl}) = R(1, \lambda_0, c_i^{rl}) - R_{T_f}$ and the function:
$$S(\lambda_0) = \|\widetilde{R}(1, \lambda_0, c_i^{rl})\|. \tag{21}$$
- Step (3a): Find λ_0 that minimizes $S(\lambda_0)$ using a local minimizer.

3. Results

We applied the steps described in Section 2 to create attitude maneuvers for a geostationary satellite from initial configuration R_0 to final configuration R_D. We show two examples:

$$\text{Path 1}: \quad R_0 = \begin{bmatrix} -0.9933 & 0.0997 & -0.0584 \\ 0.0998 & 0.9950 & 0 \\ 0.0581 & -0.0058 & -0.9983 \end{bmatrix}, \quad R_D = \begin{bmatrix} 0.5817 & 0.6517 & 0.4868 \\ 0.8134 & -0.4660 & -0.3481 \\ 0 & 0.5985 & -0.8011 \end{bmatrix} \tag{22}$$

$$\text{Path 2}: \quad R_0 = \begin{bmatrix} -0.6504 & 0.0653 & -0.7568 \\ 0.0998 & 0.9950 & 0 \\ 0.7530 & -0.0756 & -0.6536 \end{bmatrix}, \quad R_D = \begin{bmatrix} 0.21078 & 0.9761 & 0.0525 \\ 0.2947 & -0.0123 & -0.9555 \\ -0.9320 & 0.2170 & -0.2903 \end{bmatrix} \tag{23}$$

Rather than simply plotting these two maneuvers, we ran a series of simulations on an incremented set of the final position of Paths 1 and 2 to create a class of curves starting from the initial point R_0 and going to a set of perturbations of the end point R_D. This showed that the method successfully adjusted each curve in the class to remain within the pointing set, rather than showing a single curve from R_0 to R_D, which could represent an exceptional case.

The incremented sets of the final positions for Paths 1 and 2 are given by:

$$\mathcal{R}_1 = \{\exp(A_2 p_i) R_D \mid p_i = -0.5 + i/18, i \in [0, 1, \ldots, 9]\},$$

$$\mathcal{R}_2 = \{\exp(-2A_3)\exp(\frac{1}{2}A_1)\exp(\frac{\pi}{2}A_2)\exp(A_1 p_i) R_D \mid p_i = -0.2 + i/22.5, i \in [0, 1, \ldots, 9]\}$$

where the rotation matrix used to determine the iterations was chosen such that the curves in the set from the initial point R_0 to the final points in the sets $\mathcal{R}_i$ would violate the conditions stated for each path.

The plots of the maneuvers can be seen in Figures 4a,b and 5a,b. Running 10 examples also allowed us to calculate the average CPU time for each of the two cases: for Path 1, the average CPU time was 1.437 s, and for Path 2, the average CPU time was 2.113 s. The CPU time was calculated as the sum of the CPU time for the MATLAB script used to calculate the curves in Figures 4a and 5a (using Steps 1–3), the Simulink run to determine the new c_i values, and then, the second MATLAB script used to calculate the curves in Figures 4b and 5b (using Steps 1a–3a).

We chose Path 1 where the original optimal trajectory could be seen to cause the telescope to pass across the bright spot S, while in Path 2, the original optimal trajectory designed for the antenna did not fall in the desired pointing region. One can see from the adapted trajectories in Figures 4b and 5b that the c_i^0 values were successfully updated in order to point the telescope $\mathbf{a}(t)$ away from the forbidden region at all times while maintaining the communications antenna curve $\mathbf{b}(t)$ in the desired region.

4. Discussion and Conclusions

This paper presented a method to obtain optimal attitude guidance controls for a spacecraft in geostationary orbit, avoiding pointing sensitive instrumentation towards the Sun during an attitude maneuver while maintaining the direction of a beam pencil. This method used RL to produce an optimal trajectory that reacted to the presence of obstacles to maintain the configuration in the pointing set. The simulations carried out demonstrated that the trajectory that we designed avoided the forbidden pointing direction while maintaining the antenna in the operational pointing direction for two classes of paths, one that violated the bright-spot condition and a second that violated the antenna-pointing condition. When comparing the average CPU times for the two classes of solutions shown here (Path 1, 1.437 s, and Path 2, 2.113 s) with those determined in [5], which perform Lie group-based pointing while avoiding a bright spot (between 0.36 and 0.406 s), the comparison seems unfavorable: however, in their method, the user is required to first

run the program with a "guessed" c_i value and then increment the c_i values manually, each time running and then visually confirming that the trajectory avoids the forbidden region, a process that is certainly less efficient even if it is less computationally demanding. Therefore, a comparison of CPU times between the two papers is not entirely meaningful, and in terms of the ease of operation, ours is certainly preferable.

A possible drawback of our method is that we assumed the attitude to be precisely known. However, in a real mission, there will be delays in the pose estimation. We also assumed that the initial and final orientations were chosen such that the Sun-avoidant and beam pencil vectors fell in the pointing set at the initial and final position, which is reasonable, but a refinement could be to select a suitable orientation to replace the initial and final orientations if they are chosen to fall outside of the pointing set. Another weakness is the "black box" nature of the policy function neural network; also, we constructed a simple reward function and did not perform any reward shaping: it was seen as preferable since a poorly shaped reward function can cause a solution that is not ideal even if it provides the highest reward. This can be seen particularly in the fact that the vehicle passes the obstacle relatively closely and then overcompensates through the rest of the trajectory. Refining of the reward function and the inclusion of time and sensor delays are areas to be addressed in future work.

Author Contributions: H.H. carried out the research during her Postdoc with J.B. on attitude control for satellites and developed a novel geometric method for motion planning on the frame bundle of connected surfaces of arbitrary constant cross-sectional curvature. K.v.E. and H.H. noticed that an improvement could be made to the constrained planning approach used by J.B. in the paper [5] by combining this geometric method with reinforcement learning. H.H. developed the concept, wrote the paper, and carried out the simulations, and J.B. and K.v.E. provided constructive criticism and technical input. All authors read and agreed to the published version of the manuscript.

Funding: This work was sponsored by the European Regional Development Fund (FiRST Lab Project #FESR 1084).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The lead author can be contacted for the data generated during the simulations. No external data sets were used.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; nor in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

RL	Reinforcement learning
PMP	Pontryagin's Maximum Principle
LVLH	Local horizontal-local vertical

Appendix A. Optimal Control Approach

PMP [14] is a necessary condition for optimality, which is most naturally expressed in the language of the geometry of the cotangent bundle T^*G of a matrix Lie group G, where G is the configuration space of a particular motion (in this paper, $G = SO(3)$) [15]. The cotangent bundle T^*G can be trivialized (from the left) such that $T^*G = G \times \mathfrak{g}^*$, where $\mathfrak{g}^*$ is the dual space of the Lie algebra $\mathfrak{g}$. PMP is associated with the system Equation (1) with the cost $\mathcal{J}$ in (10) an optimal Hamiltonian function H on $T^*G = G \times \mathfrak{g}^*$. It follows that the

cotangent bundle of any manifold G is endowed with a canonical symplectic two-form χ_ξ that associates to each function H on T^*G a vector field $\overrightarrow{H}$ defined through the relation:

$$dH_\xi(v) = \chi_\xi(\overrightarrow{H}_\xi, v)$$

and each tangent vector $v \in T_\xi(T^*G)$.

Since in the case of $SO(3)$, $\mathfrak{so}(3)^*$ can be identified with $\mathfrak{so}(3)$ [15], in this case, the Hamiltonian function is a function on $SO(3) \times \mathfrak{so}(3)$. It follows that the Hamiltonian vector fields are given by the equation $\overrightarrow{H}[\cdot,\cdot] = \{.,.\}$, where $\{.,.\}$ denotes the right Poisson bracket. Let $\hat{p}$ denote an element of $\mathfrak{so}(3)^*$. Then, denoting the lift of the basis A_1, A_2, A_3 as $M_i = \hat{p}(A_i)$, an optimal trajectory $R() : [0, t] \to SO(3)$ is a projection of an integral curve $(R(\cdot), M(\cdot))$ of the time-varying Hamiltonian vector field $\overrightarrow{H}$ that satisfies the boundary conditions given in (1) and (10). This is called the projection "downstairs", and $M(\cdot)$ is the extremal curve, or the projection "upstairs" of $(R(\cdot), M(\cdot))$. The projection downstairs satisfies the differential equation:

$$R^{-1}(t)\left(\frac{dR}{dt}\right) = dH, \tag{A1}$$

(where $dH = \sum_{i=1}^{3} \frac{\partial H}{\partial P_i} A_i$, the unique vector field satisfying $dH(v) = \chi(\overrightarrow{H}, v)$), while the projection upstairs can be determined by solving the extremal equations given in the following theorem.

Theorem [16]: The non-canonical Hamiltonian vector fields that are necessary conditions for optimality under the PMP are defined by:

$$\begin{aligned} \frac{dM_1}{dt} &= -\frac{M_2M_3}{c_2} + \frac{M_2M_3}{c_3}, \\ \frac{dM_2}{dt} &= \frac{M_1M_3}{c_1} - \frac{M_1M_3}{c_3}, \\ \frac{dM_3}{dt} &= -\frac{M_1M_2}{c_1} + \frac{M_1M_2}{c_2}. \end{aligned} \tag{A2}$$

While in our particular case, optimality was used to produce a certain class of useful trajectories, we note that from [16], when the weights c_i are fixed to the values $c_i = I_i$ where I_i represents inertia about the i-th axis, the trajectory $(R(\cdot), M(\cdot))$ is an energy-minimal trajectory. Here, however, we used the optimization process to access a class of smooth trajectories described using a global representation of the attitude on the matrix Lie group $SO(3)$.

References

1. Hablani, H.B. Attitude Commands Avoiding Bright Objects and Maintaining Communication with Ground Station. *J. Guid. Control Dyn.* **1999**, *22*, 759–767. [CrossRef]
2. McInnes, C.R. Large Angle Slew Manoeuvres with Autonomous Sun Vector Avoidance. *J. Guid. Control Dyn.* **1994**, *17*, 875–877. [CrossRef]
3. Kjellberg, H.C.; Lightsey, E.G. Discretized Constrained Attitude Pathfinding and Control for Satellites. *J. Guid. Control Dyn.* **2013**, *36*, 1301–1309. [CrossRef]
4. Frazzoli, E.; Dahleh, M.A.; Feron, E.; Kornfeld, R. A Randomized Attitude Slew Planning Algorithm for Autonomous Spacecraft. In Proceedings of the AIAA Guidance, Navigation and Control Conference, Minneapolis, MN, USA, 13–16 August 2012.
5. Biggs, J.D.; Colley, L. Geometric Attitude Motion Planning for Spacecraft with Pointing and Actuator Constraints. *J. Guid. Control Dyn.* **2016**, 1672–1677. [CrossRef]
6. Sun, Y.; Ran, X.; Zhang, G.; Xu, H.; Wang, X. AUV 3D Path Planning Based on the Improved Hierarchical Deep Q Network. *J. Mar. Sci. Eng.* **2020**, *8*, 145. [CrossRef]
7. Macek, K.; PetroviC, I.; Peric, N. A reinforcement learning approach to obstacle avoidance of mobile robots. In Proceedings of the 7th International Workshop on Advanced Motion Control, Maribor, Slovenia, 3–5 July 2002.
8. Prescott, T.; Mayhew, J. Obstacle avoidance through reinforcement learning. *Adv. Neural Inf. Process. Syst.* **1991**, *4*, 523–530.

9. Kahn, G.; Villaflor, A.; Pong, V.; Abbeel, P.; Levine, S. Uncertainty-aware reinforcement learning for collision avoidance. *arXiv* **2017**, arXiv:1702.01182.
10. Wu, X.; Chen, H.; Chen, C.; Zhong, M.; Xie, S.; Guo, Y.; Fujita, H. The autonomous navigation and obstacle avoidance for USVs with ANOA deep reinforcement learning method. *Knowl. Based Syst.* **2020**, *196*, 105201. [CrossRef]
11. Singla, A.; Padakandla, S.; Bhatnagar, S. Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge. *IEEE Trans. Intell. Transp. Syst.* **2019**. [CrossRef]
12. Qiao, J.; Hou, Z.; Ruan, X. Application of reinforcement learning based on neural network to dynamic obstacle avoidance. In Proceedings of the IEEE 2008 International Conference on Information and Automation, Changsha, China, 20–23 June 2008.
13. Lütjens, B.; Everett, M.; How, J.P. Safe reinforcement learning with model uncertainty estimates. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019.
14. Boltyanskii, V.G.E.; Gamkrelidze, R.V.Y.; Pontryagin, L.S. *The Theory of Optimal Processes: I. The Maximum Principle*; TRW Space Technology Labs: Los Angeles, CA, USA, 1960.
15. Jurdjevic, V. *Geometric Control Theory*; Cambridge University Press: Cambridge, UK, 1997.
16. Biggs, J.D. Integrable Hamiltonian Systems on Six Dimensional Lie Groups. Ph.D. Thesis, The University of Reading, Reading, UK, 2007.