A01-37347

AIAA 2001-4155

A RANDOMIZED ATTITUDE SLEW PLANNING ALGORITHM FOR AUTONOMOUS SPACECRAFT

E. Frazzoli, M. A. Dahleh, E. Feron
Massachusetts Institute of Technology, Cambridge, MA
R. P. Kornfeld
Jet Propulsion Laboratory, Pasadena, CA

AIAA Guidance, Navigation, and Control Conference and Exhibit
August 6–9, 2001/Montreal, Quebec, Canada

AIAA 2001–4155

A RANDOMIZED ATTITUDE SLEW PLANNING ALGORITHM FOR AUTONOMOUS SPACECRAFT

E. Frazzoli[*] M. A. Dahleh[†] E. Feron[‡]
Massachusetts Institute of Technology, Cambridge, MA

R. P. Kornfeld[§]
Jet Propulsion Laboratory, Pasadena, CA

The ability to autonomously generate and execute large angle attitude maneuvers, while operating under a number of celestial and dynamical constraints, is a key factor in the development of several future space platforms. In this paper we propose a randomized attitude slew planning algorithm for autonomous spacecraft, which is able to address a variety of pointing constraints, including bright object avoidance and ground link maintenance, as well as constraints on the control inputs and spacecraft states, and integral constraints such as those deriving from thermal control requirements. Moreover, through the scheduling of feedback control policies, the algorithm provides a consistent decoupling between low-level control and attitude motion planning, and is robust with respect to uncertainties in the spacecraft dynamics and environmental disturbances. Simulation examples are presented and discussed.

Introduction

A key enabling technology that could lead to greater spacecraft autonomy is the capability to autonomously and optimally slew the spacecraft to a desired attitude while operating under a number of celestial and dynamic constraints. Large angle attitude slews are typically required for certain space science missions for retargeting of payload instrumentation, e.g. to point a telescope towards a new celestial object. In addition, typical mission profiles for future platforms , e.g. missile-tracking satellites, will require the execution of rapid attitude maneuvers at a very short notice.

In most cases the slew maneuver is subject to several constraints. As an example, consider a spacecraft carrying sensitive science or navigation sensors (e.g. cryogenically cooled telescopes, or star trackers): in this case the attitude reconfiguration must be achieved without directing the sensors along the Sun vector, or along other "bright" regions of the sky. Other constraints can come from the necessity to maintain a communication link with the ground station, from pointing requirements imposed by other spacecraft subsystem (e.g. thermal control), or even from the necessity to reduce the probability of collision of sensitive components of the spacecraft with man-made or natural orbiting objects (e.g. in the case of the crossing of planetary rings).

The planning of such constrained maneuvers can be very challenging, and can result in a costly increase in the workload of the ground segment, and in a reduction of the payload availability and flexibility of the system — for example because of a reduced ability to track targets of opportunity, for which a relatively fast mission replanning is required.

Background

The problem of planning large angle slew maneuvers for autonomous spacecraft, avoiding bright objects has been addressed by McInnes,[1,2] who used an approach based on the definition of an artificial potential function. The potential function guides the spacecraft to the target attitude, while avoiding pointing the payload boresight along the Sun vector. Such an approach suffers from the main drawback associated with potential function methods in robot motion planning: the convergence of the generated trajectory to the desired attitude is not guaranteed in general, as the spacecraft might get trapped at a local minimum of the potential function. While possible in principle, the computation of a potential function that is free of local minima (a *navigation function*[3]) is a very hard problem from the computational point of view, for non-trivial constraint specifications: as a matter of fact the only constraint that is taken into account is Sun vector avoidance by a single body-fixed vector (i.e. the boresight of a tele-

[*]Post-doctoral associate, Laboratory for Information and Decision Systems, email: frazzoli@alum.mit.edu. AIAA member.
[†]Professor, Department of Electrical Engineering and Computer Science, email: dahleh@mit.edu.
[‡]Associate Professor, Department of Aeronautics and Astronautics, email: feron@mit.edu. AIAA senior member.
[§]Staff engineer, Avionic Systems Engineering Section, email: richard.p.kornfeld@jpl.nasa.gov. AIAA member.

scope). Another limitation of this approach (which, as a matter of fact, is common to almost all of the literature on the subject) is due to the fact that the spacecraft is treated as a pointing device, that is a system evolving on the sphere S^2, as opposed to a rigid body, whose attitude evolves on the group of rotations in the three-dimensional space, $SO(3)$. In other words, pointing constraints are imposed only on one axis of the spacecraft; moreover, the control torques are computed in such a way that only one axis of the spacecraft is actually controlled, whereas rotations about this axis can be arbitrary. As a consequence, a direct application of such algorithms to cases in which the pointing constraints involve multiple body-fixed directions is not possible. Dynamical and integral constraints are also not directly addressed.

A constraint monitoring algorithm was developed for use on the Cassini spacecraft.[4] The purpose of this algorithm is to monitor the attitude maneuver requests from the ground for violation of pointing and dynamical constraints (such as maximum angular rates or accelerations), and if necessary provide appropriate constraint avoidance commands. While the detection algorithm is indeed able to check pointing constraints along multiple body-fixed directions, as well as dynamical and integral (or "timed") constraints, the avoidance maneuvers are still computed from a pointing device perspective. As a matter of fact, the avoidance maneuvers are computed in such a way that the "offending axis" is moved away from the avoidance cone. In the case in which multiple constraints are violated at the same time, a linear combination of the avoidance maneuvers is performed. While reportedly successful in many simulations, such a strategy could fail for apparently simple constraint configurations. Moreover, the avoidance maneuvers are computed to avoid constraint violations over a short period of time, and do not guarantee that progress towards the desired attitude is actually made: as in the previous case, the spacecraft might become "trapped" between constraints.

To avoid such a possibility, a global perspective must be retained in the planning process. Hablani[5] considers attitude motion planning in the presence of bright objects, while maintaining a communication link with the ground station. The proposed approach is based on a careful analysis of the vectorial kinematics on the sphere, leading to the avoidance of the bright objects. At the same time, the availability of an additional degree of freedom is exploited to ensure that the communication link with the ground is maintained. While very effective in the case presented by the author, this approach is based on a rather complex logic, which can be cumbersome in implementation and difficult to extend to more complex scenarios, involving several bright objects, and integral and dynamical constraints. In particular, the proposed approach is directly applicable only to cases in which two pointing constraints are active at the same time along the desired path.

The above approaches have been specifically developed for spacecraft attitude slew planning. On the other hand, the problem of attitude slew planning is closely related to the traditional robotics problem of motion planning in the plane or in higher dimensional space, and as such it should be possible to use techniques originally developed in the field of robotics[6–9] (notice that this is the same idea behind McInnes' approach). The main difference lies in the structure of the workspace, which in the case of attitude motion is the group of rotations $SO(3)$, as opposed to $\mathbb{R}^n$.

The solution of general motion planning problems, is computationally very demanding, even in the most basic formulations. The so-called "generalized mover's problem", involving motion planning for holonomic kinematic robots made of several polyhedral parts, among polyhedral obstacles, has been proven by Reif[10] to be PSPACE-hard, and hence at least NP-hard.[11] This — together with the topological difficulties of planning paths on SO(3) — has motivated the development of heuristic algorithms such as those outlined earlier in this section which are able to compute quickly a solution in many cases of interest. On the other hand, no guarantees are available on the ability of such algorithms to find a solution.

Randomized motion planning algorithms

Probably the most exciting recent advance in motion planning research is due to the introduction of randomized motion planning algorithms.[12] Of particular interest for the purposes of this paper is a new class of motion planning algorithms, composed of Probabilistc RoadMap (PRM) planners,[13–18] and their incremental counterparts, pioneered by the Rapidly-exploring Random Tree algorithm.[19–21]

This class of algorithms is particularly interesting since, by relaxing the completeness requirements to probabilistic completeness (i.e. completeness in expectation), it provides computational tractability, while maintaining *formal guarantees* on the behavior of the algorithm. These algorithms are based on the construction of a *graph* of feasible trajectories (i.e. trajectories along which the constraints are satisfied).

Probabilistic RoadMap planners have been proven to be complete in a probabilistic sense, that is, the probability of finding a feasible trajectory to the target attitude (if one exists) approaches unity as the number of nodes — and consequently of trajectory segments — in the graph increases. Moreover, performance bounds have been derived as a function of certain characteristics of the environment (*expansiveness*) which prove that the probability of incorrect termination decreases exponentially fast in the number of nodes.[22] Corresponding bounds have also been found for the incremental version of such algorithms, in the presence of multiple time-varying constraints, dynamical and integral constraints.[23,24]

In this paper, we will discuss how this new class of algorithm can be used to design highly efficient planners for autonomous spacecraft attitude maneuvers; for this purpose we will concentrate on the on-line generation of attitude motion commands, starting from some given initial conditions. Since we will build the graph incrementally, the graph will have the additional structure of a *tree**, rooted at the initial conditions. Nodes in the tree, also referred to as "milestones" represent states of the system along trajectories, edges can be thought of as representing "decisions" (or high-level commands) taken in the construction of the motion plan.

The planner presented in this paper is based on an algorithm introduced by the authors in earlier publications, and applied to motion planning of agile autonomous vehicles.[24,25] Central to this algorithm is the assumption that an *constraint-free* guidance loop is available, which is able to steer the system to a desired attitude at rest, assuming that there are no constraints in the environment. A vast body of literature is available on this subject, as exemplified by the book edited by Wertz.[26]

Since the ability of the planner to compute a feasible trajectory to the target attitude is guaranteed only in a probabilistic sense, it is important to ensure that constraint violation is always prevented, even though a full solution is not found. The algorithm that we will present does indeed provide *safety guarantees* in the sense that all the generated attitude commands will result in feasible trajectories, even in the case a trajectory to the target attitude is not found.

Motion planning framework

This section introduces the elements used to formulate the attitude motion planning algorithm, and to generate the simulation examples. Notice that the main assumption is the availability of an obstacle-free closed-loop attitude control system that ensures the capability to steer the spacecraft to the desired attitude, if no pointing constraints are present. One of the main differences in this paper with respect to earlier publications is the fact that we are dealing with a closed-loop, feedback-controlled system, with its resulting characteristics of robustness with respect to external disturbances and uncertainties.

*A tree is a directed graph, with no cycles, in which all nodes have several outgoing edges, to each of which corresponds another node, called a *child*. A child node can in turn have several other children. All nodes (excluding the root) have one and only one incoming edge, which comes from the *parent*. The root has no parent. A common example of a tree is the directory structure in a computer file system.

System dynamics

For the sake of simplicity, we will consider a simple, rigid body model for the attitude dynamics of the spacecraft, in the absence of a gravitational field or other disturbance torques. More complex spacecraft attitude dynamics models can be used, with no significant changes to the planning algorithm (but a stabilizing attitude control law must be available). Under the stated hypothesis, the configuration of the spacecraft is represented by an element R of the group of rotations in the three-dimensional space, $SO(3)$; in the following we will use a matrix representation of R but other representations are possible — i.e. quaternions or Euler angles.[26] The angular velocity in body axes is given by the vector $\omega \in \mathbb{R}^3$. The state space of the system will consist in the product $\mathcal{X} = SO(3) \times \mathbb{R}^3$, and we can define the state of the system as $x = (R, \omega)$. The attitude dynamics of the spacecraft will be described by the following set of ordinary differential equations (where the explicit dependence on time of R, ω and u is omitted):

$$\left\{ \begin{array}{rcl} \dot{R} & = & R\hat{\omega} \\ J\dot{\omega} & = & -\omega \times J\omega + M(u). \end{array} \right. \tag{1}$$

In the above equations, J is the inertia tensor of the spacecraft, M denotes the moments generated by the controls u, and the skew matrix $\hat{\omega}$ is defined as the unique matrix for which $\hat{\omega}v = \omega \times v$ for all vectors $v \in \mathbb{R}^3$.

In most cases of interest, the dynamics of the spacecraft are limited by constraints due to actuator saturation. Among these we can mention limits on the maximum impulse or maximum thrust available from thrusters, and on the maximum torque and maximum stored momentum for reaction wheel or control moment gyros. Other constraints of the same form might be driven by sensor requirements. For example, limits on the angular rates can derive from star-tracker requirements. These constraints do not depend on the attitude of the spacecraft, and we can call them "flight envelope" constraints. Such constraints can be encoded by inequalities of the form:

$$F(\omega, u) < \phi, \tag{2}$$

or, as in the case of maximum stored momentum, by integral inequalities of the form:

$$\int_{t_0}^{t} F(\omega(\tau), u(\tau))\, d\tau < \Phi. \tag{3}$$

Constraint-free guidance system

The main assumption on which our algorithm is based is the availability of a guidance algorithm which provides feasible trajectories in environments with no "obstacles", or pointing constraints. More specifically, we assume knowledge of a guidance law that can steer the system, in the absence of pointing constraints, to a rest condition at a desired target attitude R_d. This guidance law will be a feedback control policy parametrized by the desired attitude, which will be denoted by $\pi(\cdot, R_d)$ (i.e. the control input will be $u = \pi(x, R_d)$).

Based on the definition of the control policy π, we also require that a Lyapunov function be available, which provides a measure of the "distance" of the initial condition $x = (R, \omega)$ from the target $(R_d, 0)$; such a Lyapunov function can be given for example by the "cost", in terms of time or of control effort, which is necessary to reach the target (or a neighborhood thereof), using the policy π. We will denote such a cost function as $J^* = J^*(x, R_d)$. Last, we assume that a *simulator* of the closed-loop behavior of the spacecraft is available. Such a simulator can be obtained by integrating the differential equations (1) under the action of the policy π.

As an additional remark, notice that in general, it will not be possible to ensure global stability of the closed-loop system using a *smooth*, static, feedback control law, because of the topological structure of the configuration space, $SO(3)$; however, most of the available continuous control laws have a large region of attraction, and the knowledge of this region, denoted by $\mathcal{V}(R_d) \subset SO(3) \times \mathbb{R}^3$, will be enough for the purpose of this paper.

In this paper, we will only consider a simple constraint-free attitude control law, stated in by Bullo and Murray,[27] which has the benefits of a simple mathematical formulation, a relatively large region of attraction, and a readily available Lyapunov function. This control law is appropriate for the case in which the spacecraft is equipped with reaction wheels, providing independent torques on three axes. In practical applications, other attitude control laws — e.g. the ones proposed by Wie *et al.*[28] for control of agile spacecraft — can, and should, be implemented, with no changes to the planning algorithm.

Assume that the objective of the (constraint-free) guidance law is to slew the spacecraft to an attitude R_d, with zero angular rate. The guidance law is based on the definition of a Lyapunov function composed of a term penalizing the attitude error, and a term penalizing angular rates:

$$V(R, \omega, R_d) = \frac{1}{2} k_p \operatorname{Tr}(I - R_d^T R) + \frac{1}{2} \omega^T J \omega, \qquad (4)$$

where k_p is a positive constant (a "proportional" gain), and $\operatorname{Tr}(R)$ denotes the trace of the matrix R, that is the sum of its diagonal elements (or, equivalently, of its eigenvalues). It is easy to check that the derivative of the above function can be made negative (semi)-definite in a set $\mathcal{V}(R_d) = \{(R, \omega) \in SO(3) \times \mathbb{R}(3) \mid V(R, \omega, R_d) < 2k_p\}$, by setting the control moments such that:

$$M(u) = \omega \times J\omega - k_p \operatorname{Skew}(R_d^T R)^\vee - k_d \omega, \qquad (5)$$

where k_d is a positive constant (a "derivative" gain), $\operatorname{Skew}(R) := 1/2(R - R^T)$, and the notation $(\cdot)^\vee$ indicates the inverse of the "hat" operator introduced before.

Environment characterization

We consider an environment in which both fixed and time varying pointing constraints are present, and we assume that the future evolution of the constraints is known in advance. This is not a restrictive hypothesis in the spacecraft case, since the ephemerides of all the celestial bodies of interest are known with great accuracy.

A first class of pointing constraints formulates the necessity to avoid pointing the boresight of sensitive instruments towards the Sun or other bright objects. In this case the constraint is usually formulated as a minimum offset angle α_{min} between the instrument's line of sight and a celestial object. If the boresight of the instrument in body axes is represented by the unit vector v_b, and the direction of the celestial object in inertial coordinates is given by the unit vector w_i, such a constraint can be encoded by the following inequality:

$$Rv_b \cdot w_i < \cos \alpha_{min}. \qquad (6)$$

A second kind of pointing constraints addresses the maintenance of communication links. In this case, the communication antenna must be kept within an angle α_{max} from the ground station direction. Such a constraint can be encoded by the following inequality:

$$Rv_b \cdot w_i > \cos \alpha_{max}. \qquad (7)$$

It is easy to see that Eq. (7) can be reduced to the form (6) by replacing the body-fixed vector v_b by $-v_b$, and the avoidance angle by $\pi - \alpha_{max}$.

The pointing constraints described above must be checked pointwise in time. In some applications, it might be required to impose integral constraints: this is the case, for example, when the heat input to radiators, or other spacecraft surfaces, must be limited to maintain thermal control. Such constraints can be formulated as integral inequalities, which must hold for all times t (these have also been called "timed" constraints.[4] A simple formulation can be the following:

$$\int_{t_0}^{t} \max\{0, \cos(Rv_b \cdot w_i)\}\, d\tau < H. \qquad (8)$$

If the environment is time-varying, constraint violations must be checked on (state × time) pairs (x, t). A constraint violation checking algorithm is assumed to be available, via trajectory sampling or other appropriate method.

Problem formulation

The motion planning problem can now be stated as follows: Given an initial state $x_0 = (R_0, \omega_0) \in SO(3) \times \mathbb{R}^3$, at time t_0, and a target attitude $R_d \in SO(3)$, find a control input $u(t)$ (within the saturation limits of the actuators), that can steer the system the system from x_0 to $(R_d, 0)$, while ensuring that constraints (6),(7), and (8) are satisfied. While the usual formulation of the motion planning problem is concerned only with finding a feasible trajectory, in most applications we are also interested in finding a trajectory minimizing some cost. In this paper, we will assume a cost functional of the following form:

$$J = \int_{t_0}^{t_f} g(R_d^T R(t), \omega(t), u(t))\, dt. \qquad (9)$$

Such cost functional express the cost associated with the time required to complete the maneuver, the total length of the generated attitude path, the required control effort, or combinations thereof.

Constrained attitude maneuver planning

The motion planning algorithm in the presence of pointing constraints is based on the determination of a time-parameterized sequence of "virtual targets" $R_v(t)$ that effectively steers the system to the desired configuration while avoiding constraint violations. Such an approach casts the location of the virtual target as a function of time as a "pseudo-control" input for the system. Since the actual control inputs can be computed from the knowledge of the control policy $\pi(\cdot, R_v)$, this means that the low-level control layer (the layer actually interacting with the vehicle) and the high-level, maneuver planning layer are effectively decoupled, while at the same time ensuring full consistency between the two levels. As a consequence, unlike other approached in constrained motion planning, or constraint monitoring, the planning algorithm can be run at a rate that is much slower than the rate required for the low-level control laws.

Note also that the ideas outlined above can be seen as a motion planning technique through scheduling of Lyapunov functions, where the Lyapunov functions characterize the closed-loop stability of the system under the attitude control laws introduced earlier. While the concept is not entirely new in control theory,[29–31] to the authors' knowledge, this is the first application attitude slew planning with pointing constraints . A fundamental difference can also be seen in the fact that in our algorithm the ordering of Lyapunov functions is performed on-line, whereas in the references the ordering was determined *a priori.*

The basic idea of the algorithm is the following: starting from the initial conditions (R_0, ω_0), we incrementally build a tree of feasible trajectories, trying to explore efficiently the reachable set (i.e. the set of (x, t) couples which can be reached with the available controls, and maintaining fesibility with respect to the constraints).

At each step, we will add a new branch (edge) and a new milestone (node) to the tree. For this purpose, at each tree expansion step we have to address two points: (i) Which node do we want to expand? (ii) In which "direction" shall we explore?. The first incremental randomized motion planning algorithm was recently introduced by LaValle and Kuffner,[20] based on previous work by Kavraki, Svestka *et al.*,[13] with the name of Rapidly Exploring Random Tree (RRT). Another algorithm was introduced by Hsu, Kindel, *et al.*[23] In this paper we will use an algorithm developed by the authors, and detailed in a forthcoming paper,[24] which recovers the most desirable characteristics of both the above mentioned algorithms. It is out of the scope of this paper to give a detailed description of the algorithm, and as a consequence we will give a quick overview of the main features of the algorithm, especially for what pertains to the spacecraft attitude maneuver planning problem.

In our algorithm we roughly proceed as follows:

- pick a target attitude R_v at random, and try to expand all the nodes in the tree in sequence, in order of increasing cost $J^*(R_i, R_v)$ (i.e. starting from the *closest* node, using the measure of distance provided by $J^*(\cdot, R_v)$).

- Apply the optimal control policy $\pi(\cdot, R_v)$, until the system gets to R_v (or to a neighborhood of R_v).

If a feasible trajectory can not be found for all the nodes in the current tree, the candidate milestone is discarded.

The milestones generated according to the above procedure can be regarded as *primary* milestones. We assume that R_{rand} is generated from a uniform distribution on the workspace $SO(3)$. Notice that the fact that the algorithm involves building a tree rooted at some initial condition, allows easy implementation of an integral constraint checker, by simple bookkeeping at the tree nodes and edges.

The above procedure is repeated, and new milestones are added to the tree, until the original target attitude can be reached using the available constraint-free control law (e.g. the control law (5)), at which point the problem is solved. If needed, it is possible to continue the computations, with the objective of improving the total cost of the resulting motion plan. This can be done in a particularly efficient manner if the constraint-free control law is actually optimal (in the absence of constraints) with respect to the cost functional (9).

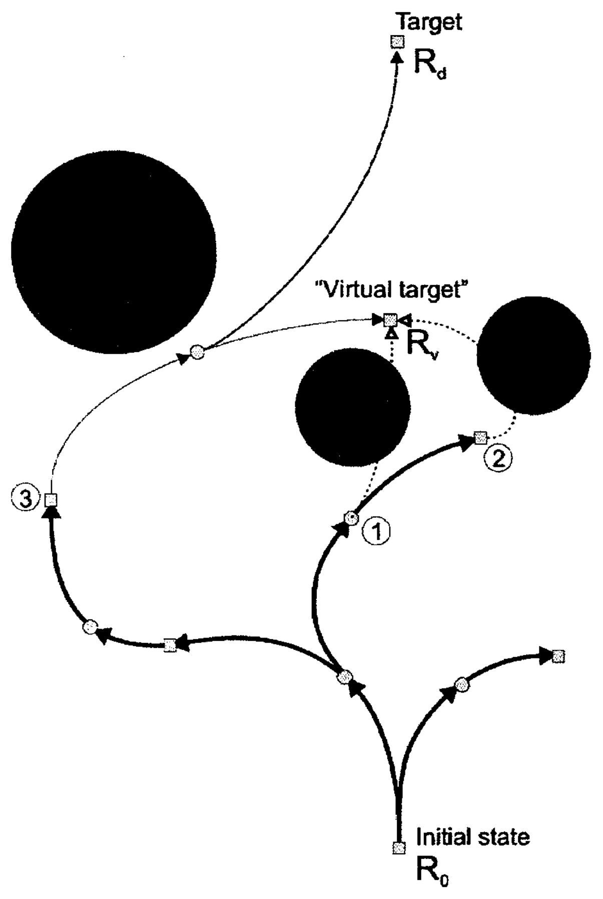

Fig. 1 Example of tree expansion procedure

Improving performance

As it can be easily recognized, the algorithm outlined above consists of jumps from rest conditions to other rest conditions and as such is unlikely to provide satisfactory performance.

Performance may be restored by realizing that the available guidance policy will stabilize the spacecraft about the (virtual) target attitude for all initial conditions in the region $\mathcal{V}(R_v)$. This suggests introducing the following step: consider the tree at some point in time and a newly added milestone to the tree. A *secondary milestone* is defined to be any state of the spacecraft along the path leading from the parent node in the tree to the newly added milestone.

These secondary milestones are added to the tree, and can be selected as the tree node to be expanded in later iterations. Note that all secondary milestones, by construction, have a primary milestone in a child subtree (see Fig. 1), which ensures the fact that the spacecraft can come to rest at an attitude which satisfies all the given constraint. In other words, safety of the generated path is always guaranteed, even in the case a full solution to the target attitude is not found. (More technical details on the safety characterization of generated motion plans in the presence of moving obstacles are available in other papers.[24])

This procedure is perhaps better understood through an example, given in Figure 1. The current tree is depicted as a set of thick lines, the squares represent the primary milestones, and the circles represent secondary milestones. The target attitude R_d is not reachable from any of the milestones currently in the tree by using the policy $\pi(\cdot, R_d)$. Thus, a new candidate milestone R_v is randomly generated, and then attempts are made to join the nodes of the tree to R_v. The closest milestone, in the sense induced by the cost-to-go function $J^*(\cdot, R_v)$ is indicated by the number one. Application of the policy $\pi(\cdot, R_V)$ with initial condition corresponding to this milestone results in a violation of one of the constraints. The same can be said from the second-closest milestone, indicated by the number two. With the third-closest milestone we have better luck, and a new collision-free trajectory is generated. A new primary milestone is created at R_v, a new secondary milestone is created at a random point on the newly generated trajectory, and the two new milestones are connected to the tree through edges encoding the new trajectory. Finally, we check whether or not the target point is reachable from the new milestones. In this case the answer is indeed positive, and the feasibility problem is solved. If this is not the case (or a "better" solution is sought), the iteration would is started again with a new randomly chosen attitude R_v.

Simulation example

In the following example, the case of a (purely imaginary) deep-space spacecraft (see Fig. 2). For the purpose of this example, we assume that the spacecraft carries a telescope with a fixed boresight in body axes, directed along the Z axis. A star tracker is mounted so that its boresight is directed along the Y axis. In order to protect the optics, the telescope axis cannot be less than 30 degrees off any bright object (i.e. the Sun, the Earth, and other bodies in the Solar System), whereas the star tracker axis must stay at least 20 degrees away from the Sun. Moreover, a telemetry antenna is mounted in the X direction: in order to maintain the communication link, the the antenna must be within 60 degrees from the direction of the Earth. Facing in the $-Z$ direction is a radiator, for which the maximum exposure to the Sun is constrained by $H = 120s$ in Eq. (8).

We consider a case in which the spacecraft is commanded to point the telescope to a new target, from an initial condition which would result in a constraint violation if the constraint-free control law were used. In the example, the "bright celestial objects" will be called Sun, Earth and Moon, even though their positions are fictitious. Here we present the results of a challenging scenario, in which the cones of Sun and Moon avoidance (for the telescope) overlap. Moreover, the Earth and the Sun are separated in such a way that the Sun avoidance maneuver would result in the acti-

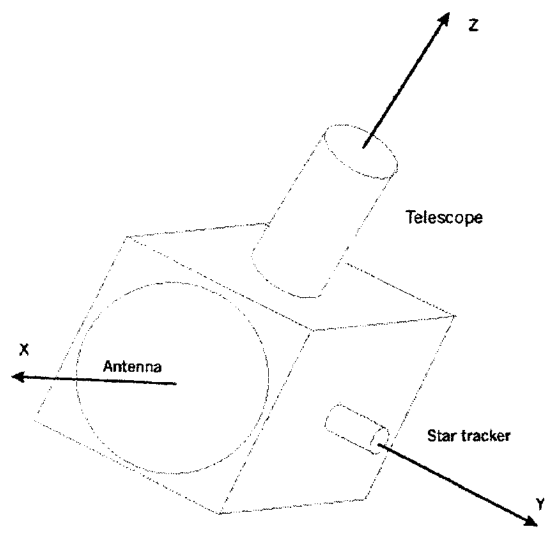

Fig. 2 Spacecraft configuration

vation of the communication link constraint. In other words, an arbitrary maneuver to avoid pointing the telescope to the Sun would result in an interruption of the communications.

In Fig. 3 the trace of pointing direction of the telescope on the tree of attitude plans is depicted. The solution (i.e. the trajectory that is eventually commanded to the spacecraft) is in a darker color, and joins the initial condition with the target point, as expected. It can be noticed that all the trajectories in the tree satisfy the pointing constraints (as well as the dynamical and integral constraints), since the axis of the telescope is always at least 30 degrees away from each of the celestial bodies under consideration.

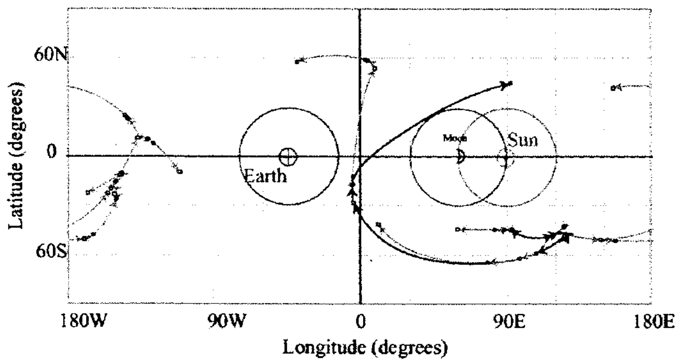

Fig. 3 Trace of the telescope pointing direction. The pointing constraints on the telescope require that it maintains a 30 degrees separation with each one of the bright celestial bodies.

In Fig. 4 the trace of the antenna direction is shown: again, all the trajectories in the tree satisfy the pointing constraints. This time, the axis of the antenna must be within 60 degrees from the Earth. Notice how non-trivial maneuvering is required to satisfy both the telescope and antenna constraints. In this case we are making full use of the capability to plan motions on the rotation group.

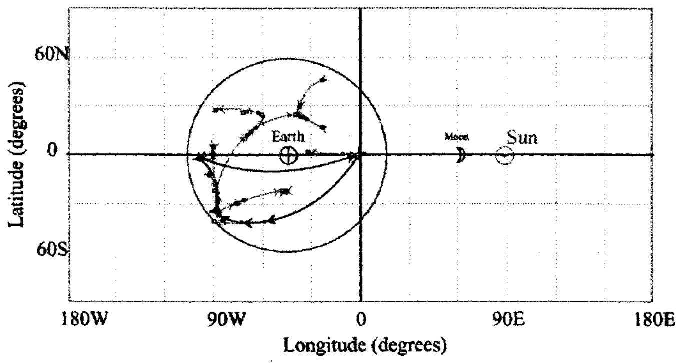

Fig. 4 Trace of the antenna pointing direction. The pointing constraints on the antenna require that it never exceeds a 60 degree separation from the Earth.

For completeness, we also report the trace of the star tracker pointing direction in Fig. 5, even though it is not influential in the determination of the required maneuver (as is the integral constraint in this case).

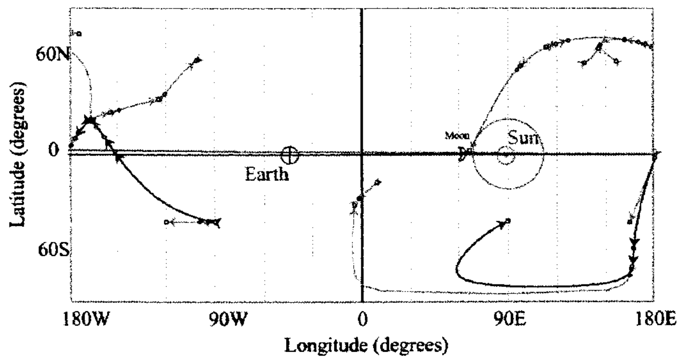

Fig. 5 Trace of the star tracker pointing direction. The pointing constraints on the star tracker require that it maintains a 20 degree separation from the Sun.

A few remarks about performance. The algorithm was implemented in C++, using the LEDA library,[32] on a 300 MHz Pentium II running Linux. Several runs with different constraint and initial/final condition geometries were performed, and a feasible solution was found by the proposed algorithm in *all cases*, in about 1.1 seconds on average (standard deviation 0.6 seconds). This is believed to be due to the fact that in most realistic attitude motion planning problems, the environment has a large expansiveness, and as a consequence the environment can be explored rapidly by the randomized algorithm.

Conclusions

In this paper a randomized attitude slew planning algorithm was presented, based on using a constraint-free guidance systems as local planners in a probabilistic roadmap framework. The main advantage of the algorithm is the capability to address in an efficient and natural fashion the dynamics of the system, multiple pointing constraints (including constraints on the pointing of several different body-fixed vectors), and integral constraints while at the same time providing a consistent decoupling between attitude slew planning and the low-level controls. Formal guarantees on the probabilistic completeness of the algorithm, and on the convergence rate to unity of the probability of correct termination, have already been established, and application to spacecraft attitude motion planning problems yielded a very efficient and reliable planner, in all the cases which have been examined.

References

[1]C. McInnes. Large angle slew maneuvers with autonomous sun vector avoidance. *AIAA J. on Guidance, Control an Dynamics*, 17(4):875–877, 1994.

[2]C. McInnes. Nonlinear control for large angle attitude slew maneuvers. *Proceedings of the Thirs ESA Symposium on Spacecraft Guidance, Navigation, and Control*, pages 543–548, 1996.

[3]R. Rimon and D. E. Koditschek. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–518, October 1992.

[4]G. Singh, G. Macala, E. Wong, and R. Rasmussen. A constraint monitor algorithm for the Cassini spacecraft. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pages 272–282, 1997.

[5]H.B. Hablani. Attitude commands avoiding bright objects and maintaining communication with ground station. *AIAA J. on Guidance, Control an Dynamics*, 22(6):759–767, November-December 1999.

[6]J.-C. Latombe. Motion planning: a journey of robots, molecules, digital actors, and other artifacts. *International Journal of Robotics Research*, 18(11), November 1999.

[7]J. C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[8]Z. Li and J.F. Canny, editors. *Nonholonomic Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1993.

[9]J.-P. Laumond, editor. *Robot Motion Planning and Control*, volume 229 of *Lectures Notes in Control and Information Sciences*. Springer, 1998. Available at `http://www.laas.fr/ jpl/book.html` at the time of writing.

[10]J.H. Reif. Complexity of the mover's problem and generalizations. In *FOCS*, pages 421–427, 1979.

[11]P. Sedgewick. *Algorithms*. Addison Wesley, 2nd edition, 1988.

[12]J. Barraquand and J. C. Latombe. Robot motion planning: A distributed representation approach. *Journal of Robotics Research*, 6(10):628–649, 1991.

[13]L.E. Kavraki, P. Svestka, J.C. Latombe, and M.H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, 1996.

[14]M. H Overmars and P. Svestka. A paradigm for probabilistic path planning. Technical report, Department of Computer Science, Utrecht University, March 1996.

[15]L. E. Kavraki, M. N Kolountzakis, and J.C. Latombe. Analysis of probabilistic roadmaps for path planning. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 3020–3025, 1996.

[16]L. E Kavraki, J.C. Latombe, R. Motwani, and P. Raghavan. Randomized query processing in robot path planning. *Journal of Computer and System Sciences*, 57(1):50–60, August 1998.

[17]D. Hsu, L.E. Kavraki, J.C. Latombe, R. Motwani, and S. Sorkin. On finding narrow passages with probabilistic roadmap planners. In *Proceedings of the 1998 Workshop on Algorithmic Foundations of Robotics*, Houston, TX, March 1998.

[18]D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *Int. J. Comp. Geometry and Applications*, 9(4-5):495–512, 1999.

[19]S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Iowa State University, Ames, IA, Oct. 1998.

[20]S.M. LaValle and J.J Kuffner. Randomized kinodynamic planning. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, 1999.

[21]J.J. Kuffner and S.M. LaValle. RRT-Connect: an efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, 2000.

[22]D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *Proceedings of the 1997 IEEE International COnference on Robotics and Automation*, 1997.

[23]D. Hsu, J.C. Kindel, J.-C. Latombe, and S. Rock. Randomized kinodynamic motion planning with moving obstacles. In *Proc. Workshop on Algorithmic Foundations of Robotics (WAFR'00)*, Hanover, NH, March 2000.

[24]E. Frazzoli, M.A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance, Dynamics and Control*. To appear.

[25]E. Frazzoli, M.A. Dahleh, and E. Feron. Real-time motion planning for agile autonomous vehicles. In *AIAA Conf. on Guidance, Navigation and Control*, Denver, CO, August 2000.

[26]J. Wertz, editor. *Spacecraft Attitude Determination and Control*. Kluwer Academic Publishers, 1978.

[27]F. Bullo and R. M. Murray. Tracking for fully actuated mechanical systems: A geometric framework. *Automatica*, 35(1):17–34, 1999.

[28]B. Wie, D. Bailey, and C. Heiberg. Rapid multi-target acquisition and pointing control of agile spacecraft. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2000.

[29]A. Leonessa, V.S. Chellaboina, and W.M. Haddad. Globally stabilizing controllers for multi-mode axial flow compressors via equilibria-dependent lyapunov functions. In *Proc. IEEE Conf. Dec. Contr.*, San Diego, CA, December 1997.

[30]R. R. Burridge, A. A. Rizzi, and D.E. Koditscheck. Sequential decomposition of dynamically dexterous robot behaviors. *International Journal of Robotics Research,*, 18(6):534–555, June 1999.

[31]M.W. McConley, B.D. Appleby, M.A. Dahleh, and E.Feron. A computationally efficient Lyapunov-based scheduling procedure for control of nonlinear systems with stability guarantees. *IEEE Transactions on Automatic Control*, January 2000.

[32]K. Melhorn and St. Näher. *The LEDA platform of Combinatorial and Geometric Computing*. Cambridge University Press, 1999.