TERRAIN-ADAPTIVE GENERATION OF OPTIMAL CONTINUOUS TRAJECTORIES FOR MOBILE ROBOTS

Thomas M. Howard[1], Alonzo Kelly[2]

[1] *Carnegie Mellon University, Robotics Institute, 5000 Forbes Ave., Pittsburgh, PA 15213, USA, thoward@cs.cmu.edu*
[2] *Carnegie Mellon University, Robotics Institute, 5000 Forbes Ave., Pittsburgh, PA 15213, USA, alonzo@ri.cmu.edu*

ABSTRACT

The problem of generating continuous trajectories for motion over general 3D terrain is important. The more naïve and common approach of compensating, in the execution phase, via feedback control, for an incorrect flat terrain assumption, is not always viable. The flat terrain assumption is also almost never necessary since the terrain shape must already be known for autonomous vehicles to operate competently in 3D terrain. We propose a fairly general constrained optimization approach to trajectory generation over arbitrary terrain, for arbitrary vehicles, which optimizes arbitrary utility/cost functionals while satisfying arbitrary constraints. The approach achieves its generality, in part, by numerically linearizing and inverting forward kinematic and dynamic models of propulsion, suspension, and motion prediction. It achieves efficiency by adopting a parametric optimal control approach from earlier related work. An implementation of this algorithm is exhibited using a model based on the Rocky 7 Mars rover platform. Several utility functions minimizing time and/or slope dwell are illustrated, while demonstrating convergence in a variety of terrain shapes.

1. INTRODUCTION AND NOTATION

Trajectory generation for mobile robots is related to the two point boundary problem of classical differential equation theory. It can be defined as the problem of finding a set of controls which satisfy initial and terminal position, pose, or state constraints. Position is defined as a location in space, pose adds orientation, and state includes, for example, rates of orientation:

$$\begin{array}{rl} \textit{position} & (x,y,z) \\ \textit{pose} & (x,y,z,\phi,\theta,\psi) \\ \textit{state} & (x,y,z,\phi,\theta,\psi,\dot{\phi},\dot{\theta},\dot{\psi}) \end{array} \qquad (1)$$

1.1 Motivation

While the present generation of mobile robots are content to move globally from A to B and perhaps avoid obstacles along the way, truly useful machines must interact with the world in ways more general than simply driving over it. Although more or less continuous motion is the core capacity of contemporary mobile robots, future machines will be required to address specific places in the environment, at specific attitudes and headings, and deploy implements to do something ultimately useful.

Competent operations in cluttered environments require the capacity to understand relatively precisely the entire space of feasible motions and search it for a (or a best) solution.

Continuous trajectories can have certain advantages. Often the time to complete the mission or the exposure to risks such as wheel slip increases when the vehicle must stop and even change direction discontinuously.

For autonomous vehicles, trajectory generation algorithms form the basis of any capacity to achieve a designated state. Further, real-time ones are needed to do so in response to information gathered on-the-fly by perception.

In the context of semi-autonomous operations, trajectory generation can be used to drive the vehicle to an operator-designated waypoint. This approach reduces operator workload and potentially provides a better solution than might be achieved otherwise.

For autonomous operations, trajectory generation can be used to acquire specific terminal states when the context is one of acquiring a point goal. When following a path, trajectory generation can correct for path following errors by reacquiring a goal path at some forward position.

Some of our related work involves the use of trajectory generation as a mechanism to encode the connectivity of state space in lattice-like networks such as the one shown later in Fig 4. In this context, trajectory generation is the key to encoding a search space which intrinsically meets all operative mobility constraints. Global path planning thereby becomes reduced to heuristic graph search but the obtained paths are directly executable.

1.2 Background

Traditionally, the trajectory generation problem has been formulated with at least three assumptions: that the environment is flat, that there is a single solution between states, and that the vehicle follows its commands perfectly.

Terrain shape affects the motion of the vehicle because the vehicle steers and moves only in the instantaneous tangent plane of the terrain – not a horizontal plane. While neglecting the influence of attitude (roll and pitch) in motion prediction simplifies the problem, it also leads to large errors in rough terrain. These errors have often been compensated by feedback control, but underactuation and nonholonomic constraints often mean that such model error disturbances cannot be removed entirely after the fact. On the other hand, the terrain shape is often well known and its effect on motion is entirely predictable so such errors are unnecessary. By incorporating terrain shape into the motion prediction model, these errors can be eliminated during generation rather than treating them as an unknown disturbance to be addressed by control.

A fully constrained formulation generates a unique solution. While this is advantageous computationally, it can be limiting in applications where the problem is more complicated than simply achieving the terminal state. If the problem is cast in an optimization context, the machinery of variational methods can be used to search a space of many feasible alternatives for one solution which is best in some overall sense. Within this mathematical context, trajectories which minimize time, energy consumption, risk, wheel slip, slope dwell and any number of other factors, in any combination, can be generated by simply modulating the utility field through which the generator is planning.

Real robots, of course, are unable to follow commands precisely for various reasons – but the degree to which they do not is often predictable. Steering actuators do not move instantaneously, and many interfaces can incorporate significant delay. Once the wheels do achieve the angles and speeds requested, the capacity of the terrain to generate the requested reaction forces is limited and dependent on terrain slope.

1.3 Prior Work

Prior work in [6] approached trajectory generation using a composite of clothoids to satisfy initial and terminal states. Based using an intermediate state at the intersection of two circles, this method was not able to solve for a path between any two arbitrary states. A method using higher-order curvature polynomials was developed in [5]. This method used energy minimization to successively deform a curve until it satisfied the constraints but it was not suitable for real time use.

A near real-time optimal control trajectory generator is presented in [4], which solves eleven first-order differential equations subject to state constraints. In [3], our group developed a fully real-time algorithm which solves the planar trajectory generation problem between two states by inverting a forward model of an idealized, curvature-actuated point vehicle in the plane.

Variational (optimization) methods of trajectory generation are as old as optimal control theory and they have been used in most fields which employ automatic control. For example, in the field of UAVs, [9] proposed a method for fast trajectory generation based on solving an approximate linearized problem (when systems are input-output linearizable). Our real-time method was extended to an optimization context in [1], where a cost function is minimized subject to meeting the terminal state constraints.

An early use of dynamic models of terrain interaction in motion planning is exhibited in [8]. Our method is adapted to rough terrain in [2], where attitude is determined from a particular more realistic vehicle model developed in [7] and dynamic models of such matters as actuator response and terrain interaction can also be accommodated.

1.4 Technical Approach

The methods in [1] and [2] are combined and adapted here to create a variational optimization method which is applicable to both arbitrary terrain and arbitrary vehicle dynamic models. The conversion to accommodate arbitrary terrain leads to a coupling of the motion prediction equations and a shift to differential equation methods rather than quadratures.

The associated increasingly numerical nature of the approach happens also to increase the level of generality with no extra effort. Although the method is broadly applicable, we will present specific results for a fairly complicated planetary rover model while optimizing a few illustrative utility functionals.

2. FORMULATION

2.1 Kinetic Motion Model

As in [2], a kinetic motion model which maps linear and angular velocities in the body frame to linear velocities and Euler angle rates in the world frame based on the SAEJ670e convention is used. The world frame velocity and Euler angle rates are defined as functions of the vehicle attitude (φ,θ), heading (ψ), linear ($\underline{v}$) and angular velocities ($\underline{\omega}$). Our method uses a terrain-following mobile robot whose controls are yaw rate (ω_z) and body-frame linear velocity aligned with the forward axis of the robot (v_x). This assumption of a single component of each velocity simplifies the kinetic motion model to:

$$\underline{P} = \begin{bmatrix} x_0 + \int_0^{t_f} \cos(\psi)\cos(\theta) v_x \, dt \\ y_0 + \int_0^{t_f} \sin(\psi)\cos(\theta) v_x \, dt \\ \psi_0 + \int_0^{t_f} \frac{\cos(\phi)}{\cos(\theta)} \omega_z \, dt \\ v_x(t_f) \\ \omega_z(t_f) \end{bmatrix} \quad \underline{P_f} = \begin{bmatrix} x_f \\ y_f \\ \psi_f \\ v_{x_f} \\ \omega_{z_f} \end{bmatrix} \tag{2}$$

Any rigid body possesses linear and angular velocity, so such a formulation approaches the completely general case. Extensions to admit three arbitrary components of both velocities in the body frame are quite straightforward.

The attitude and elevation (z) are computed by enforcing a terrain contact constraint, at a given pose, using a suspension model. For terminal state, an error vector $\underline{\Delta P}(\underline{q})$ is defined as the difference between the terminal state achieved by the linear and angular velocity controls and the terminal state required by the constraints:

$$\underline{\Delta P}(\underline{q}) = \underline{P}(\underline{q}) - \underline{P_f} \tag{3}$$

In this formulation, linear and angular velocity controls can take the form of any parameterized control primitives with sufficient degrees of freedom. The set of linear and angular velocity controls is designated by $\underline{U}(\underline{q})$. We continue to favour the choice of polynomials as the assumed form of solution:

$$\underline{U}(\underline{q}) = \begin{bmatrix} \underline{v} \\ \underline{\omega} \end{bmatrix} = \begin{bmatrix} a_1 + b_1 t + c_1 t^2 + \cdots \\ a_2 + b_2 t + c_2 t^2 + \cdots \end{bmatrix} \tag{4}$$

2.2 Optimal Control Formulation

In the optimal control formulation of the problem, a (linear and angular velocity) control must be found which satisfies a set of state constraints and minimizes a utility functional $J(\underline{q})$. As shown in [1], this can be accomplished using the method of Lagrange multipliers. The Hamiltonian is defined as the sum of the cost function and the product of the Lagrange multiplier vector with the constraints:

$$H(\underline{q},\underline{\lambda}) = J(\underline{q}) + \underline{\lambda}^T \underline{\Delta P}(\underline{q}) \quad n \text{ eqns} \tag{5}$$

The first-order necessary conditions for optimality are well-known:

$$\begin{aligned} \frac{\partial}{\partial \underline{q}} H(\underline{q},\underline{\lambda}) &= \frac{\partial}{\partial \underline{q}} J(\underline{q}) + \underline{\lambda}^T \frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) = \underline{0}^T \quad && n \text{ eqns} \\ \frac{\partial}{\partial \underline{\lambda}} H(\underline{q},\underline{\lambda}) &= \frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) = \underline{0} && m \text{ eqns} \end{aligned} \tag{6}$$

There are $n+m$ equations for this system, where n is the number of variables in the system (length of $\underline{q}$) and m is the number of constraints in the system.

This system is solved by linearizing the first-order necessary conditions. This is the well-known Newton's method and we will use it again in its fully-constrained form for computing vehicle attitude on the terrain. The initial guess of control parameters $\underline{q}$ and Lagrange multipliers $\underline{\lambda}$ are adjusted at each iteration by $\underline{\Delta q}$ and $\underline{\Delta \lambda}$ respectively until a local optimal trajectory is found when the gradient of the Hessian and the error in terminal states approaches zero. Each iteration involves a solution of:

$$\begin{bmatrix} \frac{\partial^2}{\partial \underline{q}^2} H(\underline{q},\underline{\lambda}) & \frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q})^T \\ \frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) & \underline{\underline{0}} \end{bmatrix} \begin{bmatrix} \Delta \underline{q} \\ \Delta \underline{\lambda} \end{bmatrix} = \begin{bmatrix} -\frac{\partial}{\partial \underline{q}} H(\underline{q},\underline{\lambda})^T \\ -\underline{\Delta P}(\underline{q}) \end{bmatrix} \tag{7}$$

Notice that the Hessian of the Hamiltonian is an n by n matrix. The Hessian of the constraint equations $\underline{P(q)}$ is n by n by m (a third-order tensor), but when multiplied by the m-length Lagrange multiplier vector λ, it reduces to an n by n matrix.

$$\frac{\partial^2}{\partial \underline{q}^2} H(\underline{q},\underline{\lambda}) = \frac{\partial^2}{\partial \underline{q}^2} J(\underline{q}) + \underline{\lambda} \frac{\partial^2}{\partial \underline{q}^2} \underline{\Delta P}(\underline{q}) \tag{8}$$

2.3 Utility Functional

The utility functional $J(\underline{q})$ is a description of what we want to optimize over the path. In general, it takes the form:

$$J(\underline{q}) = \int_0^{t_f} \Upsilon(q,t)\,dt \tag{9}$$

In optimal control, this <u>functional</u> $J(\underline{q})$ is conceived as a line integral of a potentially time-varying utility <u>function</u> $Y(\underline{q},t)$ along an unknown path. Equivalently, the problem can be formulated in terms of cost rather than utility. $Y(\underline{q},t)$ can be consider to be a field over the state vector. It represent any weighted combination of utilities or costs which are properties of a given position. It may include instantaneous energy consumption, wheel slip, loss of mobility, risk, slope, proximity to a position in space, or anything else of interest.

The weights used inevitably represent tradeoffs - like how far the system should be willing to go around an obstacle in order to reduce risk at the cost of lengthening the time to the goal.

2.4 Numerical Approximations of the Jacobian and the Hessian

Quadrature computation of the Jacobian and the Hessian as in [1] cannot be accomplished because of the coupling of attitude in the state equations. We proceed therefore using finite differences to approximate the Jacobian and the Hessian. Where e is a very small number (10^{-6}):

$$\frac{\partial f_{i,j}}{\partial q_k} = \frac{f_{i,j}(q_k) - f_{i,j}(q_k + e)}{e} \tag{10}$$

$$\frac{\partial f_{i,j}}{\partial q_k \partial q_m} = \frac{\dfrac{f_{i,j}(q_k) - f_{i,j}(q_k + e)}{e} - \dfrac{f_{i,j}(q_m + e) - f_{i,j}(q_m + e, q_k + e,)}{e}}{e} \tag{11}$$

This method is based on finding forward solutions. Such computations can be expensive - depending on the complexity of the vehicle model and the terrain roughness. The Hessian is symmetric, so all terms below the diagonal need not be re-computed. $m(n+1)$ forward solutions are required for the numerical approximation of the Jacobian and $m(\sum n+n+1)$ forward solutions are required for the Hessian.

2.5 Inversion of the Trajectory Kinematics

Trajectory generation is the problem of determining the set of controls which will satisfy a set of state constraints, and in this paper, optimize some utility criterion. A general method for solving this set of postures using the optimal control formulation is presented in Fig. 1. Given an initial guess of parameters (defining the input controls parametrically) $\underline{q}$, the motion prediction model is based on integrating the response of the vehicle and the terrain following models. Its predictions are then used to calculate terminal posture error. A correction term $[\underline{\Delta q}, \underline{\Delta \lambda}]^T$ is calculated based on the inversion of the system in Eqn. (7) until the terminal posture errors and the magnitude of the gradient of the Hessian approach zero.

The forward vehicle model may encode a variety of phenomena including models of the suspension, actuator dynamics, and even wheel slip. Given a set of controls $\underline{U}(\underline{q})$, we find the response $\underline{U}(\underline{q})^*$ subject to these models.

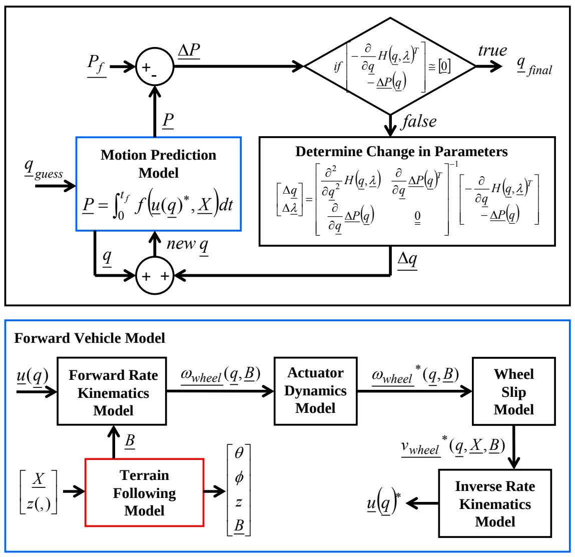
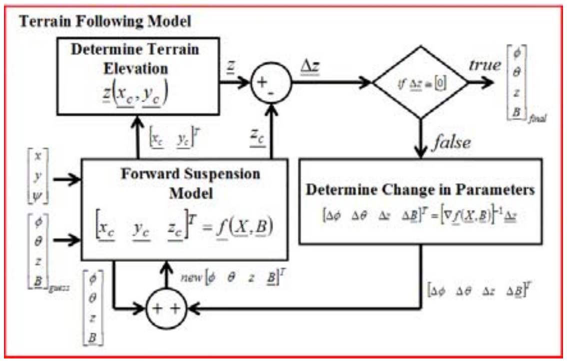


Fig. 1. Trajectory Generation Flow Diagram.

The terrain following model (Fig. 2) models the interaction of the suspension model and the terrain to determine attitude, elevation, and suspension configuration ($\underline{B}$) for a given pose ($\underline{X}$). The forward suspension model determines estimates of the positions of the wheel contact points and measures the distance between the wheel contact point ($\underline{Z_c}$) and the terrain ($\underline{Z}$) at the wheel contact point's ($\underline{X_c}, \underline{Y_c}$). A correction to the attitude ($\underline{\Delta\varphi}, \underline{\Delta\theta}$), elevation ($\underline{\Delta z}$), and suspension configuration ($\underline{\Delta B}$) is determined from the product of the Jacobian of the forward suspension model and the magnitude of the elevation errors ($\underline{\Delta Z}$). The correction terms are added to the guess and this process continues iteratively until the elevation errors approach zero.



Terrain Following Model
Determine Terrain Elevation
$z(x_c, y_c)$
Δz
true
false
Forward Suspension Model
$[x_c \; y_c \; z_c]^T = f(X,B)$
Determine Change in Parameters
$[\Delta\phi \; \Delta\theta \; \Delta z \; \Delta B]^T = [\nabla f(X,B)]^{-1}\Delta z$
new$[\phi \; \theta \; z \; B]^T$

Fig. 2. Terrain Following Flow Diagram.

3. IMPLEMENTATION

3.1 Control Primitives and Satisfying the Terminal Linear and Angular Velocity Constraints

To demonstrate the algorithm, we will use a 4th order polynomial in angular velocity and a linear angular velocity control (Eqn. 11). Higher order controls could also be used with no modifications. The initial linear and angular velocities are known and trivially equal to the a_1 and a_2 unknowns. The vector of control coefficients is represented by $\underline{q} = [b_{1,2}, \ldots t_f]^T$.

$$\begin{aligned} \omega_{z_t} &= \omega_{z_0} + b_1(t_f) + c_1(t_f)^2 + d_1(t_f)^3 + e_1(t_f)^4 \\ v_{x_t} &= v_{x_0} + b_2(t_f) \end{aligned} \tag{12}$$

In this formulation, we will neglect actuator dynamics, and wheel slip models, although including them presents no difficulty. Doing so allows the terminal linear and angular velocity to be determined in closed form from the other variables and the constraints:

$$\begin{aligned} e_1(\omega_{z_0}, \omega_{z_f}, b_1, c_1, d_1, t_f) &= \frac{\omega_{z_f} - \omega_{z_0} - b_1(t_f) - c_1(t_f)^2 - d_1(t_f)^3}{(t_f)^4} \\ b_2(v_{x_0}, v_{x_f}, t_f) &= \frac{v_{x_f} - v_{x_0}}{t_f} \end{aligned} \tag{13}$$

This is advantageous because the terminal linear and angular velocity constraints are satisfied in each iteration. The size of the system is also reduced (three constraints $\underline{P_f} = (x_f, y_f, \psi_f)$ and four controls $\underline{q} = (b_1, c_1, d_1, t_f)$) .

According to our approach of satisfying these constraints in closed form, new values for e_1 must be computed from the terminal angular velocity constraint whenever the Jacobian or Hessian are computed. The b_2 control parameter happens not to depend on any value in $\underline{q}$, so it does not need to be re-computed.

3.3 Terrain Following Model

A terrain following model is required in arbitrary terrain to determine the vehicle roll and pitch at a given pose. We will use a kinematic model based on the Rocky 7 Mars rover prototype to illustrate. This model, depicted in Fig. 3, employs a rocker-bogie suspension with three degrees of freedom ($\underline{B} = [\rho, \beta_1, \beta_2]^T$) corresponding to the major and two minor rocker-bogie angles respectively.

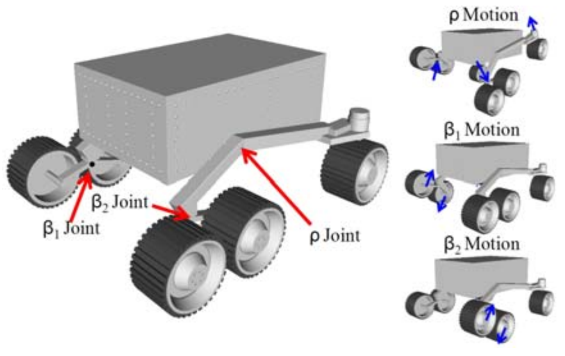


Fig. 3. Suspension Kinematics Model.

3.4 Motion Prediction

Motion prediction takes the form of three coupled, nonlinear equations. Euler integration is employed:

$$\begin{aligned} x_{t+\Delta t} &= x_t + v_x^*(t)\cos(\theta(\underline{X_t}, z(,)))\cos(\psi_t)\Delta t \\ y_{t+\Delta t} &= y_t + v_x^*(t)\cos\left(\theta\left(\underline{\underline{X_t}, z(,)}\right)\right)\sin(\psi_t)\Delta t \\ \psi_{t+\Delta t} &= \psi_t + \omega_z^*(t)\frac{\cos(\phi(\underline{X_t}, z(,)))}{\cos(\theta(\underline{X_t}, z(,)))}\Delta t \end{aligned} \tag{14}$$

It is important to use a step size small enough to both:

a) provide good estimates of the integrals of the linear and angular velocity controls, and
b) capture the effects of the profile of the terrain.

Notice that for flat terrain $((\varphi,\theta)=(0,0))$, the forward solution for the terminal heading integral can be found in closed form. Doing so allows another variable to be determined explicitly in terms of the other controls ($\underline{q}$) and the initial and terminal constraints ($\psi_0, \psi_f, \omega_0, \omega_f$) and it reduces the overall number of constraints to two.

3.5 Initialization/Termination

Ideally, we would like to seed the optimization algorithm with a set of linear and angular velocity controls which satisfy the terminal state constraints. Using heuristics previously developed for the flat-plane trajectory generator in [3], a proper set of controls can be found by solving the fully-determined problem as in [5]. Once this is found, a good initial guess for the Lagrange multiplier vector λ can be determined by solving the underconstrained system in equation 15 using the right-pseudoinverse:

$$\frac{\partial}{\partial \underline{q}} J(\underline{q}) + \underline{\lambda}^T \frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) = \underline{0}$$
$$\underline{\lambda}^T = \left[\left[\frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) \right]^T \left[\left[\frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) \right] \left[\frac{\partial}{\partial \underline{q}} \underline{\Delta P}(\underline{q}) \right]^T \right]^{-1} \right] \left[-\frac{\partial}{\partial \underline{q}} J(\underline{q}) \right] \tag{15}$$

The algorithm considers convergence to have occurred at millimeter accuracy in position and milliradian accuracy in heading. It must also achieve magnitudes of the gradient of the Hessian less than 0.1. The suspension model requires millimeter residuals in wheel contact elevation.

4. RESULTS

In order to demonstrate the advantages of the formulation, minimum time and minimum time/slope dwell utility functionals are illustrated, but in general any utility functional can be accommodated. The minimum time utility functional generates the shortest time path between states. The minimum time/slope dwell formulation shows a trade-off between avoiding high attitudes along the path and finding the shortest path which satisfies the boundary conditions.

4.1 Minimum Time Performance Index

To find a minimum time path, the utility functional $J(\underline{q})$ is defined that minimizes the time integral to achieve the terminal state:

$$J(\underline{q}) = \int_0^{t_f} 1dt = t_f \tag{16}$$

Figure 4 shows the minimum time solution for connecting trajectories in a uniformly spaced lattice overlaid on rough terrain. This result demonstrates that even in the face of large terrain undulations, an optimal continuous path consistent with the assumed polynomial form of the solution controls can be found. The network in Figure 4 has 35 nodes connected via 94 minimum time trajectories. Most achieved millimeter / milliradian error in fewer than 5 iterations of the algorithm with an average runtime per trajectory under 2 seconds.

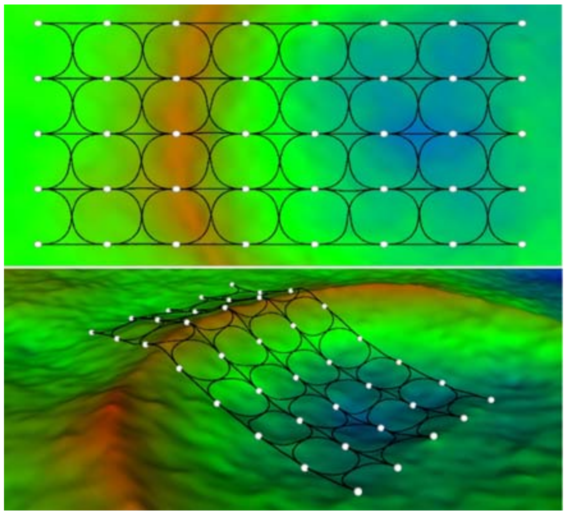


Fig. 4. Connected state lattice generated over rough terrain.

A second test of the algorithm involved solving for a variety of trajectories with different terminal states from a single initial state. Figure 5 shows paths planned to 9 terminal positions at each of 5 distinct terminal headings (the rovers are drawn at one of the 5 headings). Similar runtime and convergence results were achieved in this test.

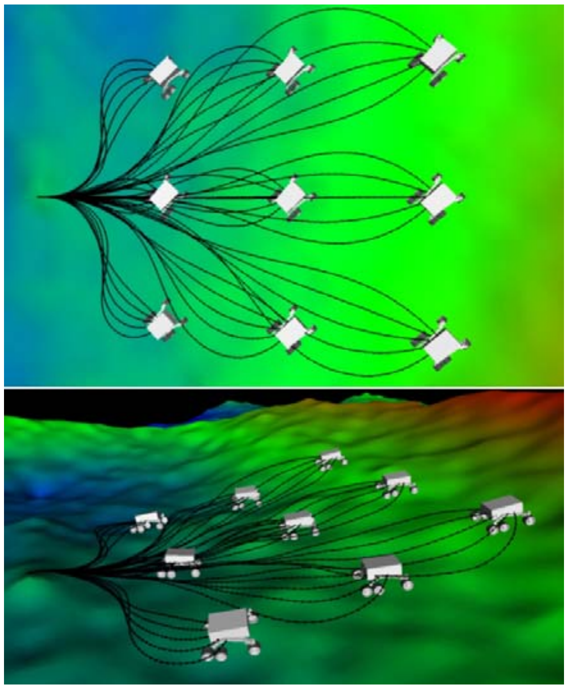


Fig. 5. A variety of terminal states in rough terrain, solved with the present method. Note that terminal positions and headings are correct despite the cumulative effect of terrain undulations. Note also

that the terrain shape compensation leads to paths which are not mirror images in 3D.

4.2 Minimum Slope Dwell Performance Index

For planetary rovers operating in rough terrain, a utility function which also penalizes high roll and pitch values can be used to solve for short paths which avoid slopes:

$$J(\underline{q}) = \int_0^{t_f} 1 + \alpha\left(\phi^2 + \theta^2\right) dt \qquad (17)$$

The "1" term in the integral represents the time that it takes to achieve the terminal posture and the *α* coefficient is a tuning parameter which represents the trade-off between finding the shortest path and the path which minimizes the amount of time spent on slopes – here called "slope dwell". As *α* approaches zero, the minimum-time path will be found. Conversely as *α* approaches infinity, the minimum slope dwell path will be found.

To demonstrate the use of this utility functional, we try to find the shortest path that satisfies the relative terminal state (x,y,ψ) = (4.0,0.0,0.0). A large hill has been placed between the initial and final states and a safe alternate route (Figs. 6 and 7) must be found. The minimum time formulation (α = 0) generates a straight line which drives over the side of the hill. When the tuning parameter is increased (α = 1), the algorithm converges to a path which moves around most of the hill. It still does not plan entirely around it because shortest time also matters. By again increasing the tuning parameter (α = 2), the path avoids the hill entirely at the cost of increased time-to-goal.

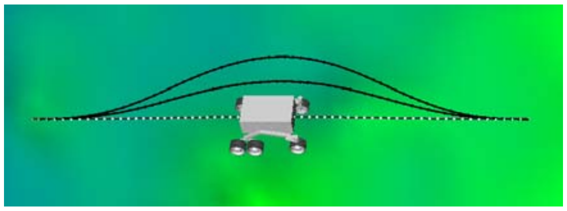


Fig. 6. Top view of solutions to the minimum slope dwell problem with varying α. Here, the α = 0 solution is highlighted in white to show that the minimum time solution is the most direct path to the goal.

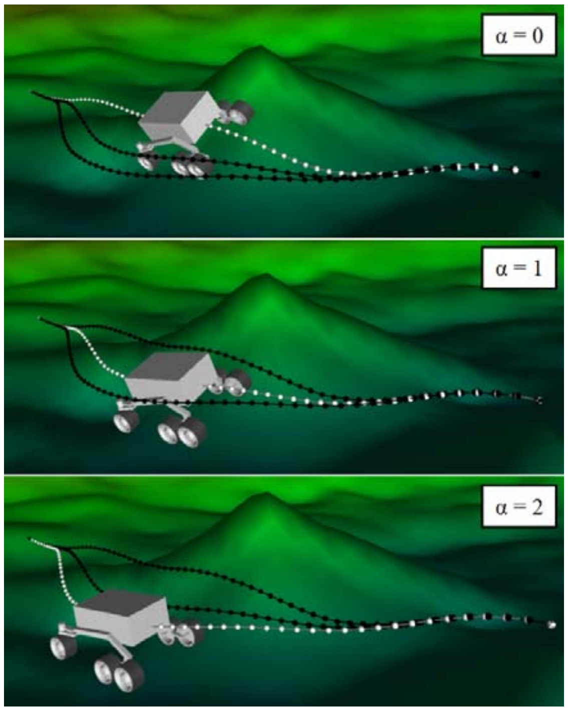


Fig. 7. Perspective view of solutions to the minimum slope dwell formulation with varying α. Notice how as α is increased, the terrain disturbance is avoided more aggressively.

5. CONCLUSIONS

An approach to trajectory generation has been presented which is both highly general and relatively efficient. Here, generality means at least two things. First, the constrained optimization approach subsumes several other problems (like the fully-constrained and unconstrained variants) as special cases. Second, the algorithm admits arbitrary terrain and vehicle models that plug in at specific points in the computation while making no assumptions about their form. The models are required in only their (relatively easy to generate) forward forms, and inversion of these models, when needed, is accomplished numerically. All of the published results of our group leading to this result are special cases of it.

Runtimes under 2 seconds have been achieved on contemporary laptop computers, for a highly complex vehicle and arbitrary terrain, after paying little attention to optimizing the code. Of course, we have also published less capable variants of the method which run in a few milliseconds, by assuming flat terrain, simpler vehicles, and fully-constrained formulations.

Our results indicate that there are no remaining barriers, and there are likely advantages, to implementing such highly capable algorithms in applications even as demanding as planetary rovers. The computations are supportable even on the rover. A new level of intelligence emerges with regard to local motion planning, in terms of understanding the vehicle's capacity to move, the effect of terrain, and the intelligent management of tradeoffs. Motions can be executed in a manner which is best in some overall sense related to increased efficiency and reduced risk.

Our intended future work involves the use of predictive terrain following models in the generation of corrective trajectories in feedback control. We are also pursuing efficient global motion planning results based on state lattices whose states are connected using the techniques described herein.

Acknowledgment

This research was conducted at the Robotics Institute of CMU under contract to NASA/JPL as part of the Mars Technology Program.

6. REFERENCES

1. Kelly, A., and Nagy, B., "Reactive Nonholonomic Trajectory Generation via Parametric Optimal Control", International Journal of Robotics Research, Vol. 22, No. 7-8, 583-601 (2003).

2. Howard, T., and Kelly, A., "Trajectory Generation on Rough Terrain Considering Actuator Dynamics", Field and Service Robotics 2005 (FSR '05), Port Douglas, Australia – July 29th – 31st, 2005.

3. Nagy, B., Kelly, A., "Trajectory Generation for Car-Like Robots Using Cubic Curvature Polynomials", Field and Service Robotics 2001 (FSR '01), Helsinki, Finland – June 11th, 2001.

4. Johannes Reuter, "Mobile Robot Trajectories with Continuously Differentiable Curvature: An Optimal Control Approach", Proceedings of the 1998 IEEE/RSJ Conference on Intelligent Robots and Systems, Victoria, B.C., Canada, October 1998.

5. H. Delingette, M. Herbert, and K Ikeuchi, "Trajectory Generation with Curvature Constrained based on Energy Minimization", Proc, IROS, pp 206-211, Osaka, Japan, 1991.

6. Shin, D.H., Singh, S., "Path Generation for Robot Vehicles Using Composite Clothoid Segments" tech. report CMU-RI-TR-90-31, Robotics Institute, Carnegie Mellon University, December 1990.

7. Tarokh, M., McDermott, G., "Kinematics Modeling and Analyses of Articulated Rovers." Technical Report No. CS/10/2005, January, 2005.

8. M. Cherif, J. Ibanez-Guzman, Ch. Laugier, and T. Goh, "Motion Planning for an All-Terrain Autonomous Vehicle", Int. Conf. on Field and Service Robotics, Pittsburgh, PA, USA, August, 1999

9. Kim, S.K.; Tilbury, D.M., "Trajectory generation for a class of nonlinear systems with input and state constraints", American Control Conference, 2001.