Trajectory Generation on Rough Terrain Considering Actuator Dynamics

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

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

Summary. Trajectory generation has traditionally been formulated on the assumption that the environment is flat. On rough terrain, however, deviations of the angular velocity vector from the vertical lead to errors which accumulate in a manner similar to the accumulation of attitude errors in odometry. In practice, feedback control can compensate for these errors in modeling by adjusting the path in real time. In many realistic cases, however, the 3D shape of the terrain is known beforehand, so it is possible to incorporate terrain shape into the predictive model — rather than treat it as an unknown disturbance. This paper presents an algorithm for trajectory generation which compensates for terrain shape in a predictive fashion. The numerical implementation makes it adaptable readily to a broad class of vehicles and even a broad class of predictable disturbances beyond terrain shape. In support of the latter, we demonstrate the ability to invert models of actuator dynamics and wheel slip concurrently with 3D terrain. An example application for a the Rocky 7 Mars rover platform is presented.

Key words: Trajectory Generation, Rough Terrain, Robot Control, Nonholonomic, Mobile Robots

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 posture constraints. Position is defined as a location in space (x,y,z); pose adds orientation (x,y,z,ϕ,θ,ψ); and posture/state includes rates of orientation (x,y,z,ϕ,θ,ψ,$\dot{\phi}$,$\dot{\theta}$,$\dot{\psi}$). The rates of orientation can be expressed in Euler angles ($\dot{\phi}$,$\dot{\theta}$,$\dot{\psi}$) or locally referenced angular velocities (ω_x,ω_y,ω_z). This paper addresses the most general problem of trajectory generation between two arbitrary postures. Among other motivations, the appearance of higher

derivatives in the constraints allows for smoother transitions between adjacent trajectory segments.

1.1 Motivation

Future missions of planetary exploration will require rovers to navigate difficult terrain with limited human supervision [3]. The use of a actuator dynamics and terrain interaction models allows for more capable, reliable, and competent autonomous navigation. Better behaved and more intuitive trajectories are generated, and the need for teleoperation and supervisory control is reduced. By allowing the rover to navigate more accurately and safely through rough terrain, it may even enable a new class of missions.

Research into fast and efficient trajectory generators is important because of limited computing resources on planetary rovers. Current trajectory generation algorithms adopt the flat plane assumption, neglecting the influence of attitude in the solution, and they ignore such matters as steering delay and wheel slip. When incorrect, all of these omissions can lead to infeasible or unsafe generated paths.

Errors induced by rough terrain and dynamics can be treated as disturbances in the system. However, there are cases where, on the scale of a few vehicle lengths, disturbances cannot be compensated by feedback [7]. Such following errors can result in collisions with obstacles near the path or incorrect terminal postures for instrument placement. Conversely, algorithms incorporating sufficiently predictive models for these effects can improve path following performance by compensating for the known component of these models in the generation process.

1.2 Prior Work

Prior work in trajectory generation algorithms involved using polynomial spirals parameterized by arc length to represent curvature. Clothoids (curvature primitives which vary linearly with arc length) have been used to generate trajectories for years; however clothoids have three degrees of freedom, which is insufficient to generally satisfy constraints which would ensure curvature continuity between paths. A method was developed to generate trajectories between postures based on a composite of clothoids in [5], but it required an intermediate posture to be determined by intersecting circles and was not able to solve for a path between any two arbitrary postures.

A method using higher order curvature polynomial spirals was developed by [4]. This method used energy minimization to successively deform a curve until it satisfied the constraints. In [1], a real-time algorithm is proposed which solves the planar trajectory generation problem between two postures by inverting the forward model of the vehicle. This method is adapted here to the 3D problem. A solution for a third-order curvature control parameterized in arc length ($\kappa(s)$=κ_0+as+bs^2+cs^3) is found iteratively by

modifying the guess of parameters ($\underline{p}$=[a,b,c,s]^T), by an amount determined from inverting a linearization of the state equations, until the error in the terminal posture ($\Delta\underline{x}$) is sufficiently small:

$$\Delta\underline{x} = J\Delta\underline{p} \tag{1}$$

$$\Delta\underline{p} = J^{-1}\Delta\underline{x} \tag{2}$$

$$\underline{p}' = \underline{p} + \Delta\underline{p} \; until \; \Delta\underline{x} \cong 0 \tag{3}$$

This method assumed that the robot operated on the x-y plane, which is equivalent to setting the attitude (ϕ,θ) to (0,0). The flat terrain assumption decouples the state equations — satisfying the (z,ϕ,θ,$\dot{\phi}$,$\dot{\theta}$) constraints everywhere. The x and y state equations take the form of Fresnel integrals, so their derivative must be approximated numerically. An approximate Jacobian is found by performing a finite difference of the x and y state equations:

$$\frac{\partial f_{ij}}{\partial p_k} = \frac{f_{ij}(p_k) - f_{ij}(p_k + e)}{e} \tag{4}$$

The problem of path planning in rough terrain was addressed in [2], where arcs calculated on a locally flat plane were used to connect vehicle positions between starting and ending poses. This method globally accounted for rough terrain but locally employed the flat-plane assumption.

2 Models

This section presents a general set of models for predicting the terminal state of a mobile robot from its command inputs. Models are presented in ascending order, beginning with wheel-terrain interaction and actuator dynamics and ending with the terminal pose integral.

2.1 Rate Kinematics, Actuator Dynamics, and Wheel Slip

The purpose of this model is to predict the difference (caused by wheel slip and actuator dynamics) between the requested and actual linear and angular velocities of the vehicle body. The forward rate kinematics model determines the wheel contact point velocity vector $\underline{\Lambda}_{in}$ for each wheel from the body-frame linear and angular velocity inputs ($\underline{V}_{in}$,$\underline{\Omega}_{in}$):

$$\underline{\Lambda} = \left[\underline{v}_{c_1}^T \; \underline{v}_{c_2}^T \; \cdots \; \underline{v}_{c_n}^T \right]^T = h(\underline{V}_{in}, \underline{\Omega}_{in}) \tag{5}$$

A response wheel contact velocity vector $\underline{\Lambda}_{out}$ is determined from the actuator dynamics and wheel slip models $\dot{\underline{\Lambda}}_{out}$=$i$($\underline{\Lambda}_{in}$). The inverse rate kinematics model then determines the body frame linear and angular velocity outputs ($\underline{V}_{out}$,$\underline{\Omega}_{out}$) based on the output wheel velocity vector $\underline{\Lambda}_{out}$. A flowchart describing the model can be seen below in Figure 1:

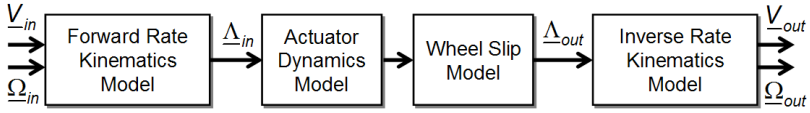

Fig. 1. Actuator Dynamics and Wheel Slip Model: Actuator dynamics and wheel slip is modeled at the wheel level. Forward and inverse rate kinematics are used to transfer between wheel frame and body frame velocities.

2.2 Suspension Kinematics

The previous model determined the linear and angular velocities of the body in the body frame. The purpose of the suspension kinematic model is to determine the attitude of the vehicle under suitable assumptions of terrain contact so that these velocities can be mapped to the world frame in the next step. In general, the attitude cannot be determined in closed form from the terrain. The mechanism used is to start with a forward model, defined to produce wheel contact point coordinates with respect to the body frame, given the suspension variables ($\underline{B}$=β_1,β_2,β_3,$\cdots$). This model is inverted to produce the body attitude by enforcing a terrain contact constraint and determining the robot configuration which minimizes the error between the wheel contact points and the terrain (see for example [6]). The attitude (ϕ,θ) and altitude (z) of a robot is a function of the pose (x,y,ψ), suspension variables ($\underline{B}$),and terrain shape ($z(,)$):

$$\begin{bmatrix} \phi & \theta & z \end{bmatrix}^T = g(x, y, \psi, \underline{B}, z(,)) \tag{6}$$

2.3 Kinetic Motion Model

The kinetic motion model maps linear and angular velocities in the body frame to linear velocities and Euler angle rates in the world frame. This paper follows the SAEJ670e convention, where the x-axis points forward, y-axis to the right, and the z-axis straight down. The Euler angles yaw (ψ), pitch (θ), and roll (ϕ) represent subsequent rotations coinciding with the robot frame about the z, y, and x axis respectively. The world frame velocity $\underline{\dot{R}}$ is found by the product of the rotation matrix and the robot frame velocity $\underline{V}$:

$$\underline{\dot{R}} = rot_z(\psi) rot_y(\theta) rot_x(\phi) \cdot \underline{V} \tag{7}$$

$$\underline{\dot{R}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{8}$$

The angular velocities in the robot fixed frame are determined by transforming the individual rotation rates ($\dot{\phi}$,$\dot{\theta}$,$\dot{\psi}$) from their intermediate frames to the robot-fixed frame:

$$\underline{\Omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + rot_x(\phi) \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + rot_x(\phi) rot_y(\theta) \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{9}$$

$$\underline{\Omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & s\theta \\ 0 & c\phi & -s\phi c\theta \\ 0 & s\phi & c\phi c\theta \end{bmatrix} \cdot \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{10}$$

Roll, pitch, and yaw rates in terms of the angular velocities can be found by inverting this relationship:

$$\underline{\dot{\Psi}} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & t\theta s\phi & -t\theta c\phi \\ 0 & c\phi & s\phi \\ 0 & -\frac{s\phi}{c\theta} & \frac{c\phi}{c\theta} \end{bmatrix} \cdot \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{11}$$

Throughout this paper, $\underline{X}$ will represent the vehicle configuration vector $[\underline{R},\underline{\Psi}]^T$. Equations (8) and (11) can be concatenated to produce the kinetic motion model:

$$\begin{bmatrix} \dot{R} \\ \dot{\Psi} \end{bmatrix} = f(\underline{V}, \underline{\Omega}, X) \tag{12}$$

Simply Actuated Kinematic Motion Model

For a terrain-following mobile robot, the only controllable angular velocity is ω_z, so we will assume that (ω_x,ω_y)=(0,0). Likewise, a body-frame linear velocity aligned with the x-axis of the robot (v_x) will be assumed to be the only controllable linear velocity, so we will assume that (v_y,v_z)=(0,0). Under these assumptions, the attitude and altitude are determined by the suspension kinematic model and the differential equation which governs the pose $\underline{P}$=(x,y,ψ) simplifies to:

$$\underline{\dot{P}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} cos(\psi)cos(\theta)v_x \\ sin(\psi)cos(\theta)v_x \\ \frac{cos(\phi)}{cos(\theta)}\omega_z \end{bmatrix} \tag{13}$$

2.4 Trajectory Kinematics

The forward model of trajectory generation is generated by integrating the kinetic motion model (the world-frame velocities ($\dot{R}$) and robot-frame orientation rates ($\dot{\Psi}$)):

$$\underline{X} = \int f(\underline{V}, \underline{\Omega}, \underline{X}) dt \tag{14}$$

3 Trajectory Generation Algorithm

This section describes an algorithm to solve the two-point boundary value problem generating the control inputs consistent with a desired terminal state.

3.1 Inverting the Trajectory Kinematics

The problem of trajectory generation is that of determining a set of controls that will satisfy a set of posture constraints. A general method for solving the coupled, nonlinear equations in equation (13) is presented in Figure 2. This is a method which will find the nearest local solution. It does not converge in general to the global optimum but we have not had any real difficulty with incorrect local minima.

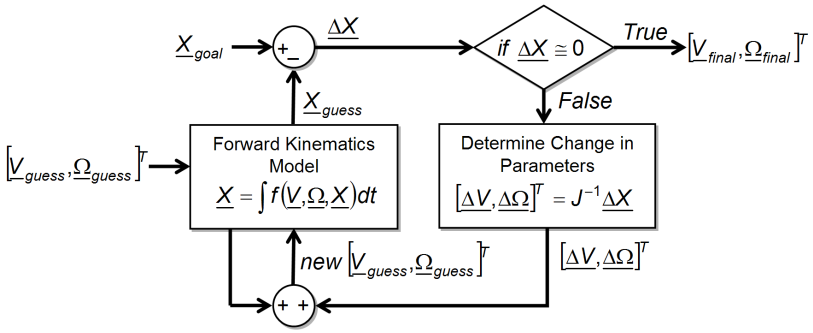

Fig. 2. Inverse Trajectory Generation Solver: From an initial guess of controls ($\underline{V}$,$\underline{\omega}$), the forward trajectory is evaluated and the control is adjusted based on the product of the system Jacobian and the constraint errors. This iterative method continues until the terminal conditions are reached.

In this approach, an initial guess of velocity controls $\underline{V}_{guess}$, $\underline{\Omega}_{guess}$ is modified based on the product of the system Jacobian and the constraint errors ($\underline{\Delta X}$) until the termination requirements are reached. As in [1], partial derivatives are computed numerically using equation (4).

3.2 Control Primitives

The trajectory generation problem reduces to that of finding a set of controls ($v_x(t)$,$\omega_z(t)$) which satisfy the seven constraints (x_f,y_f,ψ_f,v_{x_0},v_{x_f},ω_{z_0},ω_{z_f}). The terminal pose is described as the relative pose of the terminal and initial postures. A third-order polynomial spiral and a linear velocity profile provide enough degrees of freedom to satisfy the seven constraints:

$$\omega_z(t) = \omega_{z_0} + at + bt^2 + ct^3 \tag{15}$$

$$v_x(t) = v_{x_0} + dt \tag{16}$$

Notice that, for systems that do not model actuator delays or wheel slip, three of the constraints (v_{x_0},v_{x_f},ω_{z_0}) are automatically satisfied using this set of control primitives (since d=$\frac{v_{x_f} - v_{x_0}}{t_f}$. For systems that do incorporate such dynamics, the right-pseudo inverse ($J^T(JJ^T)^{-1}$) can be employed in the inverse trajectory generation solver to find for five variables (a,b,c,d,t) that satisfy the four constraints (x_f,y_f,ψ_f,ω_f). The right-pseudo inverse can also be employed to solve for controls which are higher-order polynomial spirals than the ones in equations (15) and (16).

4 Application to the Rocky 7 Rover

This section demonstrates how the general methods discussed in section 2 can be applied to a specific model. The Rocky 7 prototype Mars rover (Figure 3) employs a rocker-bogie suspension to provide contact for all six wheels with most terrains.

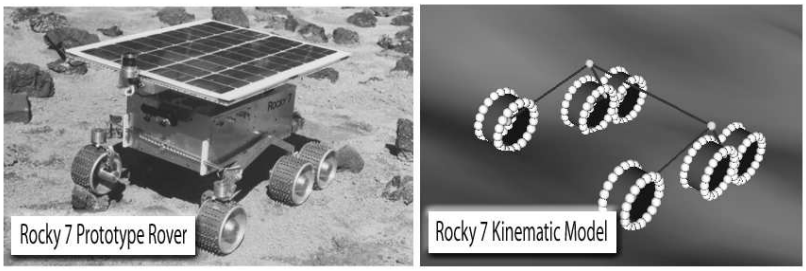

Fig. 3. Rocky 7 Prototype Mars Rover Kinematic Model: On the left is a picture of the Rocky 7 rover, a six-wheeled robot which has a rocker-bogie suspension to navigate complex terrain while maintaining wheel contact. On the right is a representation of our kinematic model of the suspension, which has the three passive joints (ρ,β_1,β_2).

Figure 3 shows how our kinematic model compares to Rocky 7. It was based on the work done in [6], but computed the forward kinematics with respect to the world frame for our suspension model solver. Rocky 7 employs a rocker-bogie suspension that has three degrees of freedom (ρ,β_1,β_2) which correspond to the major and two minor rocker-bogie angles respectively.

4.1 Rocky 7 Rate Kinematics, Actuator Dynamics, and Wheel Slip

The Rocky 7 rover has eight actuators: a steering actuator on each of the front two wheels and six drivable wheels. In order to get a realistic model of actuator dynamics, a first-order lag was assumed.

$$v_{c_{out}}(t) = \frac{\alpha_1 dt}{\tau}[v_{c_{in}}(t) - v_{c_{out}}(t-\tau)] + v_{c_{out}}(t-\tau) \quad (17)$$

$$\psi_{1,2_{out}}(t) = \frac{\alpha_2 dt}{\tau}[\psi_{1,2_{in}}(t) - \psi_{1,2_{out}}(t-\tau)] + \psi_{1,2_{out}}(t-\tau) \quad (18)$$

The wheel slip model assumes that only a fraction of the requested velocity is achieved. Inverse rate kinematics generate the achieved V_{out},Ω_{out} due to the actuator dynamics and wheel slip model.

4.2 Rocky 7 Suspension Kinematics

Just as in the general solution, the suspension model provides a mapping of the linear and angular velocities estimated by the actuator dynamics and wheel slip models from the body frame to the world frame, except now the suspension model is known. Taking advantage of the fact that there are six controls (attitude (ϕ,θ), altitude (z), and the three rocker-bogie angles (ρ,β_1,β_2)) and six constraints (z_{c_1}-z_{c_6}), the same numerical method used in the inverse trajectory generation solver is applied. An error vector $\underline{\Delta z}$=[Δz_{c_1},$\cdots$,Δz_{c_6}]^T is formed from the difference between the elevation of the terrain and the contact point of each of the six wheels. The initial guess of control is adjusted by the product of the inverse Jacobian of the forward kinematics of the contact points with respect to the world frame and the elevation error vector until the terminal conditions are met.

4.3 Rocky 7 Kinetic Motion Model

Since the Rocky 7 rover is a terrain-following mobile robot, the kinetic model follows the form of equation (13). The attitude and altitude are determined from the suspension model for a given pose.

4.4 Rocky 7 Trajectory Kinematics

The forward model of trajectory generation for the Rocky 7 rover is generated by integrating equation (13).

5 Implementation

5.1 Forward Solution of Trajectory Kinematics

In order to determine the terminal pose (x_f,y_f,ψ_f) of the rover, the system dynamics described in equations (14)-(16) must be integrated with respect to time. These state equations are coupled and nonlinear. We have found simple Euler integration to be sufficient:

$$x_{t+\Delta t} = x_t + v_{x_{out_{t+\Delta t}}} cos(\theta(x_t, y_t, \psi_t)) cos(\psi_t) \Delta t \tag{19}$$

$$y_{t+\Delta t} = y_t + v_{x_{out_{t+\Delta t}}} cos(\theta(x_t, y_t, \psi_t)) sin(\psi_t) \Delta t \tag{20}$$

$$\psi_{t+\Delta t} = \psi_t + \frac{cos(\phi(x_t, y_t, \psi_t))}{cos(\theta(x_t, y_t, \psi_t))} \omega_{z_{out_{t+\Delta t}}} \Delta t \tag{21}$$

$$\omega_{z_{out_{t+\Delta t}}} = f(\omega_{z_{in_{t+\Delta t}}}) \tag{22}$$

$$v_{x_{out_{t+\Delta t}}} = g(v_{x_{in_{t+\Delta t}}}) \tag{23}$$

Notice that the output linear and angular velocities of the rate kinematics are used in this model. In this method, a configuration at each new pose must be found. The algorithm can be sped up dramatically by using the previous configuration at time $t - \Delta t$ as the seed for the configuration at the current time t.

It is essential that Δt be small enough to accurately model the integral of the system dynamics and capture the terrain along the path. For example, in trajectory generation over the scale of a few rover lengths, 12-15 iterations may be enough to accurately model the system dynamics but these may still miss small terrain disturbances that may influence the path enough to matter.

5.2 Trajectory Generation using Inverse Trajectory Kinematics

Utilizing the forward model of trajectory kinematics developed in the previous section, along with the controls and methods of section 3.1 and 3.2, an inverse trajectory kinematics solver which accounts for both rough terrain and actuator dynamics is obtained.

5.3 Initialization/Termination

Since the numerical method used in this work does not guarantee global convergence, a heuristic which places the terminal posture of the initial guess near the goal posture is required. It was found that the solution to the two-dimensional trajectory generation problem places the terminal posture of the initial guess within 15% of the goal on a variety of interesting terrains. This is close enough to ensure convergence in most applications. In the case of an exceptional terrain disturbance which incurs a large terminal posture perturbation, line searches or scaling of the change of parameters (Δp) can be implemented to prevent overshoot and divergence from the solution.

The set of termination conditions used for the trajectory generation algorithm were similar to those in [1], stopping iterations when [Δx_f, Δy_f, $\Delta \psi_f$, $\Delta \omega_f$]^T=[$0.001m$, $0.001m$, $0.01rad$, $0.01\frac{rad}{sec}$]^T. Likewise, the suspension model required millimeter residuals in Δz_{c_i}.

6 Results

6.1 Rough Terrain Trajectory Generation Example

This section demonstrates the need of rough terrain trajectory generation by examining an example situation. In this example, the Rocky 7 platform is asked to find a continuous trajectory between two postures as seen in Figure 4. The relative terminal posture $[x_f, y_f, \psi_f, \omega_f]^T$ is equal to $[3.0m, 5.0m, \frac{\pi}{2.0}rad, 0.0\frac{rad}{sec}]^T$. In order to isolate the effect of neglecting the influence of attitude in the trajectory generator, rate kinematics are ignored in this example.

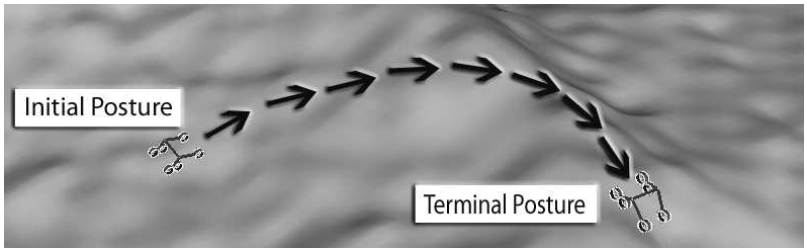

Fig. 4. Example Rough Trajectory Generation: This figure shows an example trajectory generation problem, where a continuous path is desired between an initial posture inside a crater and the final posture just over the lip of the crater.

First, the two-dimensional continuous curvature path is solved to millimeter accuracy and fed into the three-dimensional forward solution. The two-dimensional solution incurs a terminal position error of 6.2% (45.3cm) of the entire arclength of the solution. The three-dimensional trajectory generator finds a new path that is continuous in angular velocity, with an initial and final velocity of 1 m/sec, for milimeter accuracy in only 3 iterations. Figure 5 shows the difference between the two-dimensional system model and three-dimensional system model paths and how the latter generates the correct trajectory.

6.2 General Results

To evaluate the need, performance, and behavior of this algorithm, several thousand tests were run to understand rates of convergence and range of errors to expect. One behavior that was recognized is that even though error would increase dramatically with rougher terrain, the number of iterations required to meet the termination conditions did not. The numerical method that we are using attempts to remove all error in a single iteration, so this behavior suggests that the first order approximation is a good one.

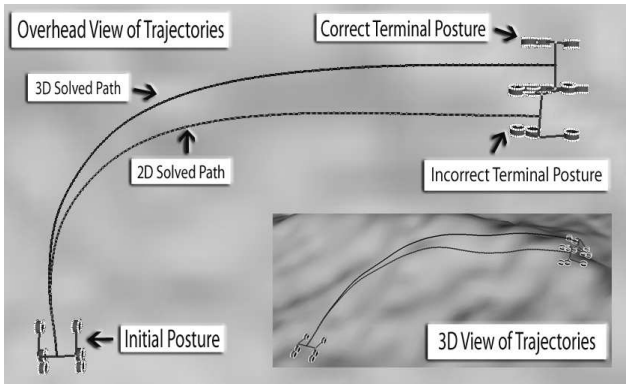

Fig. 5. Trajectory Generation Solution: This figure shows the result of neglecting attitude in the forward model (two-dimensional solution) and the new solution based on a three-dimensional system model. By neglecting attitude, the terminal position is off by 6.2% when compared to the total arclength of the solution.

7 Conclusions

In the context of mobile robots which must already expend significant effort to understand terrain complexity, the use of a flat terrain assumption in trajectory generation is difficult to justify. However, as the paper has shown, the use of terrain information requires a certain amount of effort to develop a more complex generation algorithm. While space was not available to address a computation comparison, the additional computation for 3D models is not a significant factor in practice.

A very general algorithm has been presented which can generate continuous paths for mobile robots obliged to drive over rough terrain while subject to additional nonidealities such as wheel slip and actuator delays. The essential problem is to invert a model of how parameterized control inputs, terrain shape, terrain interaction and actuator dynamic models determine the terminal state of a vehicle at all future times. A numerical technique was adopted due to the assumed inability to express terrain shape in closed form. However, once a numerical approach is adopted, it also means that any forward model can be inverted to produce continuous controls subject only to the capacity of the numerical linearization to converge. In principle, a full Lagrangian dynamics model can be inverted using our technique, for example.

The Rocky 7 prototype rover was used to illustrate the application of the general models of suspension and rate kinematics to a specific robot. For any vehicle, only forward rate kinematics and forward suspension models are needed to use the rest of the algorithm. Our results suggest there is much to gain and little to lose by moving to fully 3D models. Such predictive models lead to improved performance by removing as much model error as possible

at planning time — so that path following controls are used only to compensate for truly unpredictable disturbances.

While the algorithm has been presented in the context of planning computations, it promises to be equally valuable for the generation of corrective trajectories in feedback path following controls. Future work will assess the value of the algorithms for this purpose in the hope of developing short term path followers which maximally exploit the model and terrain information which can be assumed to be present in most present and future mobile robots.

Acknowledgment

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

References

1. Nagy, B., Kelly, A., "Trajectory Generation for Car-Like Robots Using Cubic Curvature Polynomials", Field and Service Robots 2001 (FSR 01), Helsinki, Finland - June 11, 2001.
2. Simeon, T., Dacre-Wright, B., "A Practical Motion Planner for All-terrain Mobile Robots". 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems, Yokohama, Japan - July 26-30, 1993.
3. Iagnemma, K., Shibly, H., Rzepniewski, A., Dubowsky, S., "Planning and Control Algorithms for Enhanced Rough-Terrain Rover Mobility". International Symposium on Artificial Intelligence and Robotics and Automation in Space 2001 (i-SAIRAS 01), St-Hubert, Quebec, Canada - June 18-22, 2001.
4. H Delingette, M. Herbert, and K Ikeuchi, "Trajectory Generation with Curvature Constraint based on Energy Minimization", Proc, IROS, pp 206-211, Osaka, Japan, 1991.
5. 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.
6. Tarokh, M., McDermott, G., Hung, J. "Kinematics and Control of Rocky 7 Mars Rover." Tech Report. August 1998.
7. Volpe, R., "Navigation Results from Desert Field Tests of the Rocky 7 Mars Rover Prototype." International Journal of Robotics Research, 1998.