Continuous Control Primitive Trajectory Generation and Optimal Motion Splines for All-Wheel Steering Mobile Robots

Thomas M. Howard and Alonzo Kelly
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890
{thoward,alonzo}@ri.cmu.edu

ABSTRACT

We present a method for trajectory generation for all-wheel steering mobile robots which can account for rough terrain and predictable vehicle dynamics and apply it to the problem of generating optimal motion splines. There has been little work in trajectory generation for vehicles with all-wheel steering capability compared to the Ackermann, differential-drive, or omnidirectional mobility system models. The presented method linearizes and inverts forward models of propulsion, suspension, and motion to minimize boundary state error given a parameterized set of controls. Our method for generating optimal motion splines between a set of state boundary constraints optimizes the free path heading boundary constraint while meeting position and orientation state constraints. We demonstrate this algorithm on the Rocky 8 rover platform, where parameterized linear velocity, curvature, and path heading controls are generated which satisfy position, orientation, and path heading constraints in rough terrain.

Index Terms – Trajectory Generation, Mobile Robot, All-Wheel Steering, Motion Splines, Rough Terrain.

1 INTRODUCTION

Trajectory generation is the problem of determining a feasible motion (or a set of feasible motions) that will permit a vehicle to move from an initial state to a final state given some model of the associated dynamics. While this two-point boundary value problem is classical and well-studied, it remains quite complex to solve adequately in practice.

In order to generate a smooth, continuous path which satisfies an arbitrary number of constraints (including position, orientation, path heading, velocities and their rates), a nonlinear differential equation that, in 2D, generates the Fresnel integrals, must be solved. The addition of rough terrain further complicates the system by coupling these nonlinear equations of motion. Numerical methods, such as the ones discussed in this paper, are required to solve such problems for arbitrary terrain and vehicle models.

1.1 Motivation

Most trajectory generators for mobile robots are derived from algorithms based on the Ackermann, differential drive, or omnidirectional steering models. Ackerman and differential drive methods require that the velocity vector in the local tangent plane be aligned with the forward axis of the robot and the center of rotation about along an axis aligned with the fixed set of wheels. Omnidirectional methods allow independent motion in translation and rotation [8].

Vehicles with all-wheel steering capability differ from these models in that the velocity vector can be aimed in any direction in the local tangent plane and the center of rotation can be anywhere in the local tangent plane (Figure 1), but the path heading cannot change arbitrarily due to the nonholnomic constraints of the wheels.

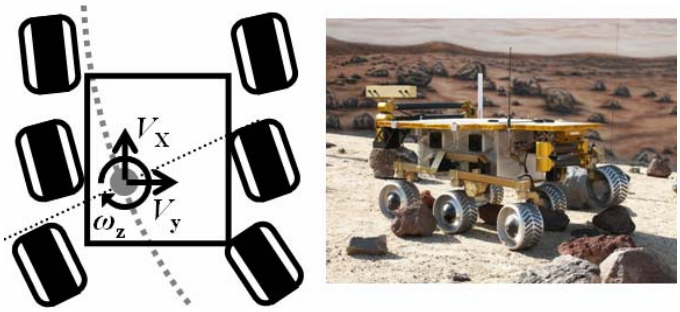

Figure 1: All-Wheel Steering Mobility System. For mobile robots with the property that all of the wheels are steerable (e.g. Rocky 8, shown), the body can rotate about any point in the local tangent plane.

1.2 Related Work

In the context of robot motion planning, most research in trajectory generation has dealt with finding obstacle-free paths subject to nonholnomic constraints assuming flat terrain and simple vehicle models. Most of the work to date falls into one of two categories: graph search via a sequence of low-order geometric primitives or optimization using a single high-order primitive.

Some of the first work in trajectory generation involved composing optimal paths from sequences of line segment arcs [9], clothoids [2][11], and cubic spirals [3]. The desire for higher-order geometric primitives was intended to allow for higher-levels of continuity at the boundary constraints. B-splines have been used to meet arbitrary position and heading

boundary constraints by defining a sequence of knot points along the path [4]. Methods based on sinusoidal and Fourier series input functions also appear throughout the literature [12][13][14]. These methods exploit the geometry of the problem to directly solve for the unknown path parameters. They cannot however generally solve for collision-free paths in an obstacle field.

Some of the most recent work in omnidirectional trajectory generation based on optimal control techniques is discussed in [10], where near-optimal paths are constructed for omnidirectional vehicles using bang-bang optimal control methods. Their methods generate minimum-time omnidirectional trajectories subject to complicated dynamics and actuator models.

Grid-search techniques have been used for a long time in kinodynamic planning. In the context of robot manipulators, optimal joint trajectories were planned in [15] using grid-search. These methods also apply to the problem of solving for obstacle-free and minimum-length paths which satisfy nonholonomic and boundary constraints [16][17][18][19][20]. The drawback of using graph-search techniques for trajectory generation is the resolution lost due to discretization of state space and/or control space. Only a subset of possible motions is expanded at each node and this limits the set of values that can be imposed as boundary constraints.

The methods presented in this article differ from the body of prior work discussed by not requiring that the path heading and vehicle yaw boundary constraints be equal in the wheeled mobile robot trajectory generation problem. This allows us to generate paths which take advantage of the all-wheel steering mobility system to produce more sophisticated and capable maneuvers. Our trajectory generation algorithm is then applied to the problem of generating efficient motion splines for all-wheel steering mobile robots by optimizing the free path heading boundary constraints while meeting position and orientation boundary state constraints.

2 Controls, Boundary Constraints, and Trajectory Prediction

2.1 Notation and Transformations

A state vector $\underline{q}$ is defined to include all degrees of freedom necessary to initialize an integral of the dynamics of the system. Typically, $\underline{q}$ would include information describing global position (x,y,z), orientation (φ,θ,ψ), and body-frame linear and angular velocities ($\underline{V}$,$\underline{\Omega}$). In this formulation, we add the path heading (γ) to the state vector since in all-wheel steering mobility systems, the path heading need not be equal to the body yaw:

$$\underline{q} = \left[x, y, z, \phi, \theta, \psi, \underline{V}, \underline{\Omega}, \cdots\right]^T \tag{1}$$

2.2 Boundary State Constraints

Trajectory generation is a form of two-point boundary value problem and it is convenient to constrain the vehicle state at the boundaries of the path. The most basic types of state constraints include initial and final world-frame position (x,y,z) and orientation (φ,θ,ψ). However, since roll, pitch, and elevation are determined by the pose and the ground for a terrain-following mobile robot, only the position (x,y) or pose (x,y,ψ) boundary constraints can generally be specified.

Typically there are also requirements on the vehicle controls that need to be satisfied at the initial and final state. Linear and angular velocities and accelerations are often constrained for a dynamically feasible motion plan. A state vector is formed for the initial and final states of the robot:

$$\underline{q}_0 = \left[x_0, y_0, \psi_0, v_{x_0}, v_{y_0}, \omega_{z_0}, \cdots\right]^T \tag{2}$$

$$\underline{q}_f = \left[x_f, y_f, \psi_f, v_{x_f}, v_{y_f}, \omega_{z_f}, \cdots\right]^T \tag{3}$$

2.3 Heading vs. Yaw and Continuity Constraints

In the general case, it is necessary to distinguish vehicle heading from vehicle yaw. The former describes the orientation of the body frame with respect to the world frame. The latter describes the orientation of the path tangent vector with respect to the world frame. It is convenient in this analysis to concentrate on the components of vehicle velocity (v_x,v_y) expressed in the body-frame. Only when vehicle velocity is constrained such that $v_y = 0$ do yaw and heading become synonyms. All other vehicles will be called omnidirectional.

A useful set of controls for omnidirectional vehicles includes the vehicle heading (γ) and the speed in the local tangent plane ($|v|$). This is merely a transformation of the original control set (v_x,v_y), so it does not change our kinetic motion model:

$$\begin{aligned} v_x &= |v|\cos(\gamma - \psi) \\ v_y &= |v|\sin(\gamma - \psi) \end{aligned} \quad \rightleftarrows \quad \begin{aligned} |v| &= \sqrt{v_x^{\,2} + v_y^{\,2}} \\ \gamma &= a\tan\left(v_y / v_x\right) \end{aligned} \tag{4}$$

It is convenient in this case to redefine our state vector to include states related to these redefined controls. Initial and final states then include a notion of the vehicle yaw ψ heading γ and speed $|v|$:

$$\underline{q}_0 = \left[x_0, y_0, \psi_0, |v|_0, \gamma_0, \omega_{z_0}, \cdots\right]^T \tag{5}$$

$$\underline{q}_f = \left[x_f, y_f, \psi_f, |v|_f, \gamma_f, \omega_{z_f}, \cdots\right]^T \tag{6}$$

This formulation eliminates an ambiguity that occurs at zero speed because the heading can be determined at zero velocity from a Jeantaud diagram. In such a diagram, lines drawn at the wheels which are perpendicular to the wheel directions intersect at the instantaneous center of rotation. More generally, the direction in which a vehicle will move can be determined from the steer angles of its wheels whether it is moving or not. The steering positions of its wheels also constitute additional state information because they cannot move instantaneously so it is appropriate that heading be in the state vector.

For example, consider the vehicle shown in Figure 2 at zero initial velocity. The path heading can be found from a Jeantaud diagram (in the figure the velocity is oriented 45° from the $+x$ body-frame axis). Also the magnitude of the velocity vector $|v|$ is known to be zero. Equation (4) can be used to find $v_x = 0$ and $v_y = 0$. Conversely, if the only known quantities were the body-frame velocities (v_x, v_y) and they were known to be zero, the path heading could not be determined because the arctangent of (0/0) is not defined. The formulation using heading has removed the ambiguity.

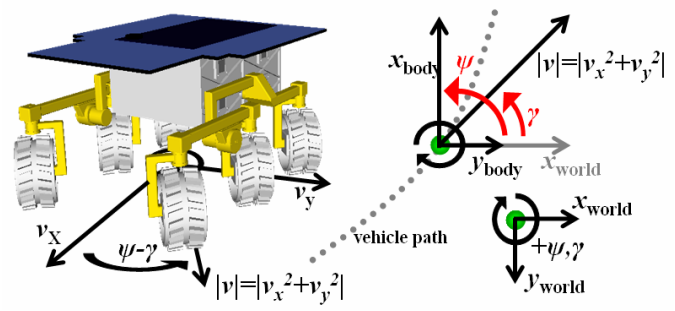

Figure 2: Conversion of Body-Frame Linear and Angular Velocities to Path Heading and Speed. In order to distinguish heading from yaw, the normally independent v_x and v_y are transformed into the speed in the local tangent plane $|v|$ and the path heading (γ), which is the angle to the path tangent from the world-frame x-axis

2.4 Terrain Following Kinetic Motion Model

This model is called kinetic because the system dynamics are developed without reference to forces and mass properties. While this is done for efficiency in the computationally limited context of our planetary rover application, nothing in the formulation prevents the use of more sophisticated models including fully dynamic ones.

For robots constrained to move on a surface, such as slowly moving vehicles, the differential equations which govern the system can take a simplified form. Under assumptions of terrain contact, the unsuspended elements of an omnidirectional mobile robot can only be actuated locally by linear velocities along the x and y axes of the body frame and the angular velocity about the body z axis. The roll rate, pitch rate, and rate of elevation are determined by the local terrain. For such a model, only the 2D pose ($\underline{p}$) of the robot is controllable:

$$\dot{\underline{p}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} (c\psi c\theta)v_x + (c\psi s\theta s\phi - s\psi c\phi)v_y + (c\psi s\theta c\phi + s\psi s\phi)v_z \\ (s\psi c\theta)v_x + (s\psi s\theta s\phi + c\psi c\phi)v_y + (s\psi s\theta c\phi - c\psi s\phi)v_z \\ -\frac{s\phi}{c\theta}\omega_y + \frac{c\phi}{c\theta}\omega_z \end{bmatrix} \quad (7)$$

The terrain following assumption can be used to generate a reduction of the state vector $\underline{q}$, since the roll, pitch, and elevation become merely functions of the global pose:

$$\underline{q} = \left[x, y, \psi, v_x, v_y, \omega_z, \cdots \right]^T \quad (8)$$

The unknown values of v_z, ω_x, and ω_y can be determined by projecting the vehicle state forward in time, inverting a suspension model to enforce terrain contact, and solving for the effective rates. Unlike some of our previous work in rough terrain trajectory generation [1], we do not assume here that these rates are zero.

2.5 Controls

Since the kinetic motion model uses body-frame linear and angular velocities, they are natural choices for our set of controls. This choice is made without loss of generality because a specific implementation might elect to include a mapping from wheel level controls onto body motions before the steps discussed below. Body-frame velocities also satisfy the constraints of rigid body motion by construction and they reduce the dimension of the input space to a minimum. Wheel velocities and steering angles are found by mapping the body frame linear and angular velocities through a kinematic model of the vehicle. For a terrain-following mobile robot, only the body-frame velocity in x and y direction (v_x, v_y) and angular velocity about the z direction (ω_z) are determined in this way. The elevation rate and the angular velocities about the x and y axes are entirely dependent on the terrain shape.

Our trajectory generation method uses parameterized controls. While this control space represents a subset of all feasible motions, an appropriate choice of parameterization can represent nearly all possible paths. The free variables in the control represent knobs that allow the algorithm to change the shape of the control. Each free variable in the control is placed in a control vector ($\underline{ub}$) and the length of this vector represents the number of degrees of freedom in the system.

One effective parameterization is a polynomial function of time, here shown for angular velocity about the z-axis:

$$\omega_z(\underline{u}, t) = a + bt + ct^2 + dt^3 + \ldots \text{ where } \underline{u} = \left[a, b, c, d, \ldots, t_f \right] \quad (9)$$

In the forward simulation of the vehicle, the inputs are modified by models of actuator dynamics, wheel slip, rate and joint limits to produce the response $\underline{u}^*$ to the inputs before integrating the kinetic motion model. For example, if delays are known to exist in the steering system, then the predicted response of the steering actuator is used instead of the requested steering input.

2.6 Trajectory Prediction

The kinetic motion model can be integrated numerically over time to determine the terminal state. This integration is essentially a forward simulation of the robot subject to whatever constraints one chooses to include. Since Equation 8 is a set of coupled nonlinear equations; numerical methods must be used to determine the final state.

$$\underline{q}_{sim} = \left[x_{sim}, y_{sim}, \psi_{sim}, \gamma_{sim}, |v|_{sim}, \cdots \right]^T = \underline{q}_0 + \int \dot{\underline{q}}\left(\underline{q}, \underline{u}^*\right) dt \qquad (10)$$

Any numerical method can be used to integrate this set of differential equations. The Euler forward method is a clear choice because of its simplicity of implementation. The terminal state is accurately predicted so long as the step size is small enough to satisfy the Courant-Friedrich-Levy condition. Other methods, such as the Runge-Kutta RK2 and RK4 methods would produce more accurate integrations for a larger step size, at the cost of complicating the forward solution method.

3 Trajectory Generation

The previous sections has outlined an approach to forward vehicle modelling which computes where the vehicle will go given input controls, initial state, and terrain. Of course, the trajectory generation problem is that of essentially inverting this forward model to determine the controls which will cause the vehicle to go somewhere in particular. The solution developed below will rely on the forward model to encode vehicle specific details but those details will remain encapsulated in the forward model. The rest of the solution is unaware of vehicle specifics.

3.1 Constrained Trajectory Generation

A constrained trajectory generation approach is appropriate when the degrees of freedom in the control vector ($\underline{u}$) are equal to or greater than the number of constraints in the final state vector ($\underline{q}$). Here, we care only about meeting the terminal boundary constraints:

$$\left[\frac{\partial \Delta \underline{q}\left(\underline{u}\right)}{\partial \underline{u}} \right] \Delta \underline{u} = -\Delta \underline{q}\left(\underline{u}\right) \qquad (11)$$

A correction factor for the controls ($\Delta\underline{u}$) can be found by inverting the relationship in equation (). In the event that the Jacobian of the boundary constraint error is non-square, the right-pseudoinverse can be used to evaluate this expression. For such a general interpretation of the inverse:

$$\Delta \underline{u} = -\alpha \left[\frac{\partial \Delta \underline{q}\left(\underline{u}\right)}{\partial \underline{u}} \right]^{-1} \Delta \underline{q}\left(\underline{u}\right) \qquad (12)$$

3.2 Constrained Optimization Trajectory Generation

An optimal control formulation of the problem is appropriate when the length of the control vector ($\underline{u}$) exceeds that of the terminal boundary constraints ($\underline{q}_f$). In this case, it is actually necessary that something be optimized over the trajectory to create the needed extra constraints.

Define an error in the terminal state ($\Delta\underline{q}$) to be the difference in the simulated terminal state ($\underline{q}_{sim}$) and the terminal boundary constraints ($\underline{q}_f$). In the optimal control formulation of the problem, linear and angular velocity controls ($\underline{u}$) must be found which satisfying a set of boundary constraints ($\underline{q}$) and which minimize some utility functional $J(\underline{u})$. Just as in [1][6], 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:

$$\underline{\underline{H}}\left(\underline{u}, \underline{\lambda}\right) = \underline{J}\left(\underline{u}\right) + \underline{\lambda}^T \Delta \underline{q}\left(\underline{u}\right) = \underline{J}\left(\underline{u}\right) + \underline{\lambda}^T \left(\underline{q}_f - \underline{q}_{sim}\left(\underline{u}\right) \right) \qquad (13)$$

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

$$J\left(\underline{u}\right) = \int_0^{t_f} Y\left(\underline{u}, \underline{q}, t\right) dt \qquad (14)$$

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

A correction factor for the controls ($\Delta\underline{u}$) and Lagrange multiplier vector ($\Delta\underline{\lambda}$) is found by inverting the relationship in equation (25):

$$\begin{bmatrix} \Delta \underline{u} \\ \Delta \underline{\lambda} \end{bmatrix} = \alpha \begin{bmatrix} \dfrac{\partial^2 \underline{\underline{H}}\left(\underline{u}, \underline{\lambda}\right)}{\partial \underline{u}^2} & \dfrac{\partial \Delta \underline{q}\left(\underline{u}\right)^T}{\partial \underline{u}} \\ \dfrac{\partial \Delta \underline{q}\left(\underline{u}\right)}{\partial \underline{u}} & \underline{\underline{0}} \end{bmatrix}^{-1} \begin{bmatrix} -\dfrac{\partial \underline{\underline{H}}\left(\underline{u}, \underline{\lambda}\right)^T}{\partial \underline{u}} \\ -\Delta \underline{q}\left(\underline{u}\right) \end{bmatrix} \qquad (15)$$

A scaling factor α is used to ensure numerical stability when initial guesses which are far from the solution. In place of α, the optimal step size could be determined by doing a simple line search.

4 Optimal Motion Splines

4.1 General Optimal Motion Spline Generation

Motion spline generation for a sequence of points is itself an optimal control problem. Similarly to the constrained optimization trajectory generation problem, we optimize some utility functional along the sequence of trajectories by varying free boundary state constraints along the sequence of trajectories.

Another approach is to treat the problem using gradient methods like steepest descent or Newton-Raphson. Using these methods, the Jacobian and Hessian of the system are determined by estimating the partial derivatives of the cost function with respect to the free boundary state constraints.

4.2 Optimal Motion Spline Generation for Instrument Placement and All-Wheel Mobility Systems

This formulation for the optimal motion splines is especially nice for all-wheel steering systems and single-cycle

instrument placement (maneuvering a robot to a position and orientation to deploy an instrument), because typically there are no path heading boundary state constraints. There are usually no constraints on the path heading of the vehicle at the boundary states, only that the path remains obstacle-free.

We can formulate the optimal motion spline problem for all-wheel steering mobility systems for the minimum-time solution with the terminal state path headings are the free variables. We have implemented a simple gradient descent algorithm to demonstrate the effectiveness of this problem setup. Since partial derivative estimation can be expensive, this formulation also has the nice property that some of the trajectories are independent of some of the boundary state constraints, so those partial derivatives will always be zero:

$$\frac{\partial t_f(\underline{u},t)}{\partial \underline{\gamma}_{free}} = \begin{bmatrix} \frac{\partial t_{f,1}}{\partial \gamma_{free,1}} & \frac{\partial t_{f,2}}{\partial \gamma_{free,2}} & 0 & 0 \\ 0 & \frac{\partial t_{f,2}}{\partial \gamma_{free,2}} & \ddots & 0 \\ \vdots & \vdots & \ddots & \frac{\partial t_{f,n}}{\partial \gamma_{free,n-1}} \\ 0 & 0 & \cdots & \frac{\partial t_{f,n}}{\partial \gamma_{free,n}} \end{bmatrix} \tag{16}$$

$$\Delta \underline{\gamma}_{free} = -\alpha \left[\frac{\partial t_f(\underline{u})}{\partial \underline{\gamma}_{free}} \right] \tag{17}$$

$$\underline{\gamma}_{free_{t+1}} = \underline{\gamma}_{free_t} + \Delta \underline{\gamma}_{free} \tag{18}$$

5 Trajectory Generation Results

This section demonstrates some uses of the trajectory generator for all-wheel steering mobility system models and the optimal motion spline generation.

Results are presented for a simulated vehicle because it is the best way to test the algorithm. The main purpose of the algorithms is to invert a model to produce feasible motions that meet the dynamic constraints encoded in the model and the boundary constraints encoded in the problem specification.

To demonstrate the capability to generate motion plans for all-wheel steering vehicles, we plan two different motions in rough terrain which satisfy the same initial and terminal position and orientation constraints (Figure 3). The algorithm assumed a trapezoidal velocity profile, a cubic polynomial curvature primitive, and a linear path heading primitive.

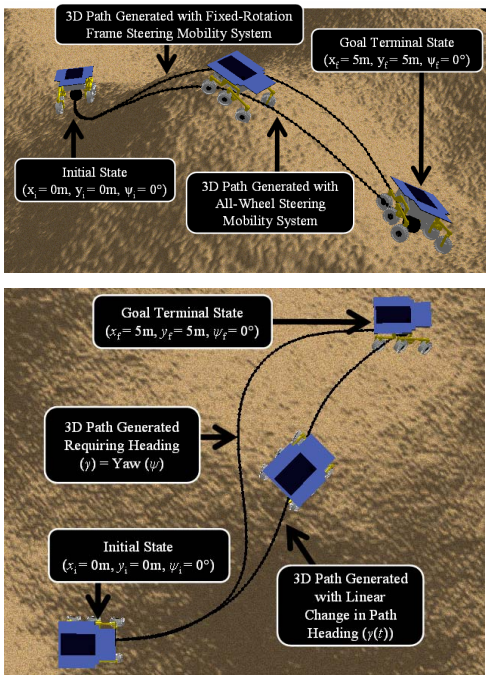

Figure 3: Trajectory Generation in Rough Terrain for All-Wheel Steering Mobile Robots. Two motion plans are generated with the rough terrain trajectory generator, the first requiring the path heading be equal to the path yaw and the second allowing the path heading to vary independently from the vehicle yaw.

The first motion plan required that the path heading be equal to the body yaw at the terminal state, which is the same trajectory generated using traditional trajectory generation methods [1]. The second motion plan allowed the path heading to vary linearly along the path from 0° to 45°. The second motion plan still achieves the correct terminal position and orientation for the boundary states but generates a more efficient motion plan which is 12% shorter by taking advantage of its all-wheel steering capability.

6 Optimal Motion Spline Results

Our methods for all-wheel steering mobility system trajectory generation can be used in conjunction with the optimal motion spline generation algorithm to determine an optimal set of paths for a sequence of waypoint. This is very important for single-cycle instrument placement, because it provides the capability to determine the correct set of trajectories to take while still achieving all of the required boundary states.

To demonstrate the algorithm, we try to plan a sequence of trajectories shown in Figure 4 which meet four position, heading, and curvature boundary state constraints in rough terrain. All of the trajectories shown in Figure 4 were planned making the assumption that the path heading be equal to the body yaw at the terminal boundary constraints.

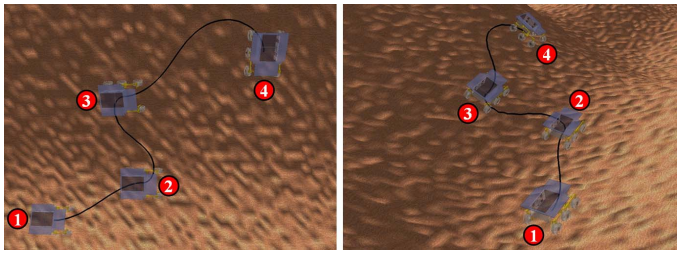

Figure 4: Typical Motion Plan for Single-Cycle Instrument Placement. A motion plan that meets four boundary state constraints is generated which does not utilize the all-wheel steering mobility system.

Our methods for all-wheel steering mobility system trajectory generation and motion spline generation can be used to find the optimal motion sequence for this system. Therefore, we can find the minimum-time path for the sequence of states by allowing the terminal path headings to vary. After just a few iterations of gradient descent, we converge to the optimal solution shown in Figure 5, which is both shorter and smoother than the naively planned path sequence shown in Figure 4.

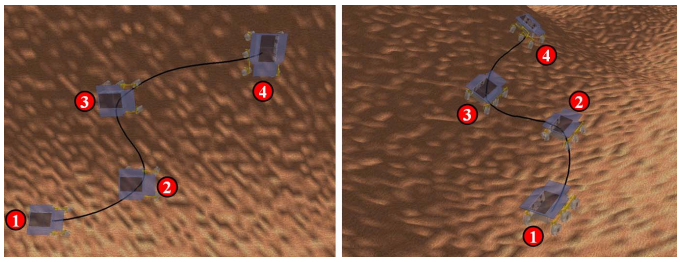

Figure 5: Optimal Motion Plan for Single-Cycle Instrument Placement. An optimal motion plan that meets the four boundary state constraints is generated using our algorithm by allowing the path heading to vary at points 2, 3, and 4 and optimizing for minimum-time. Notice that the overall motion plan is shorter and smoother than the motion plan generated in Figure 4.

7 Conclusions

This paper has presented methods for trajectory generation for all-wheel steering mobility system models and optimal motion spline generation. This algorithm for trajectory generation is the most general version of our trajectory generator to date and can plan motions planes for vehicles with (Rocky 8) or without (Rocky 7, Ackermann, Skid-Steered, Tank-Steer) all-wheel steering mobility systems. The optimal motion spline generation method uses this capability to solve for the minimum-time sequence of trajectories by finding the optimal boundary state path headings at each node. These algorithms can provide all-wheel steering mobile robots and unprecedented capability to generate optimal single-cycle instrument placement motion plans autonomously.

Acknowledgment

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

References

[1] T.M. Howard and A. Kelly. Terrain-Adaptive Generation of Optimal Continuous Trajectories for Mobile Robots. *International Symposium on Artificial Intelligence, Robotics, and Automation in Space 2005* (i-SAIRAS 05). Munich, Germany, September 2005.

[2] Y. Kanayama and N. Miyake. Trajectory Generation for Mobile Robots. In *Proceedings of the International Symposium on Robotics Research*, pages 16-23, Gouvieux, France, 1985.

[3] Y. Kanayama and B.I. Hartman. Smooth Local Path Planning for Autonomous Vehicles. In *Proceedings of the International Conference on Robotics and Automation*, volume 3, pages 1265-1270, Scottsdale, Arizona, 1989.

[4] K. Komoriya and K. Tanie. Trajectory Design and Control of a Wheeled-Type Mobile Robot using B-Spline Curve. In *Proceedings of the IEEE-RSJ International Conference on Intelligent Robots and Systems*, pages 398-405, Tsukuba, Japan, September 1989.

[5] I. Nesnas, M. Maimone, and H. Das. Rover Maneuvering for Autonomous Vision-Based Dexterous Manipulation. In *IEEE Conference on Robotics and Automation*, San Francisco, California, April 2000.

[6] B. Nagy and A. Kelly. Trajectory Generation for Car-Like Robots using Cubic Curvature Polynomials. *Field and Service Robotics 2001* (FSR '01), June 2001.

[7] D. Tilbury, J.P. Laumond, R. Murray, S. Sastry, and G. Walsh. Steering Car-Like Systems with Trailers Using Sinusoids. In the *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pages 1993-1998, May 1992.

[8] F.G. Pin and S.M. Killough. A New Family of Omnidirectional and Holonomic Wheeled Platforms for Mobile Robots. In *Proceedings of the IEEE Conference on Robotics and Automation* 1994 (ICRA 94).

[9] L.E. Dubins. On Curves of Minimal Length with a Constraint on Average Curvature and with Prescribed Initial and Terminal Positions and Tangents. *American Journal of Mathematics*, 79:497-516, 1957

[10] T. Kalmár-Nagy, R. D'Andrea, P. Ganguly. Near-Optimal Dynamic Trajectory Generation and Control of an Omnidirectional Vehicle. *Robotics and Autonomous Systems*, 46:47-64, 2004.

[11] D.H. Shin and S. Singh. Path Generation for Robot Vehicles Using Composite Clothoid Segments. Tech Report, CMU-RI-TR-90-31, The Robotics Institute, Carnegie Mellon University, 1990.

[12] R.W. Brockett. Control Theory and Singular Riemann Geometry. In *New Directions in Applied Mathematics*, pages 11-27, Springer-Verlag, New York, New York, 1981.

[13] D. Tilbury, J.P. Laumond, R. Murray, S. Sastry, and G. Walsh. Steering Car-Like Systems with Trailers Using Sinusoids. In the *Proc. of the IEEE Int. Conf. on Robotics and Automation*, vol. 3, pages 1993-1998, May 1992.

[14] R. Murray and S. Sastry. Nonholonomic Motion Planning: Steering Using Sinusoids. In *IEEE Trans. on Automatic Control*, vol. 38, pages 700-716, 1993.

[15] G. Heinzinger, P. Jacobs, J. Canny, and B. Paden. Time-optimal trajectories for a robot manipulator: A provably good approximation algorithm. In American Control Conference, 1990..

[16] J. Canny, B. Donald, J. Reif, and P. Xaiver. On the Complexity of Kinodynamic Planning. In the *Proc. of the 29th IEEE Symposium on Foundations of Computer Science*, pages 306-318, New York, New York, 1988.

[17] P. Jacobs and J. Canny. Planning Smooth Paths for Mobile Robots. in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 2-7, Scottsdale, Arizona, May 1989.

[18] J. Barraquand and J.C. Latombe. On Non-Holonomic Mobile Robots and Optimal Maneuvering. *Revue d'Intelligence Artificielle*, 3(2): 77-103, 1989.

[19] J.A. Reeds and L.A. Shepp. Optimal Paths for a Car that Goes Both Forward and Backward. *Pacific Journal of Mathematics*, 145(2):367-393, 1990.

[20] J.P. Laumond. Nonholonomic Motion Planning via Optimal Control. In the *Proc. of the Workshop on Algorithmic Foundations of Robotics*, pages 227-238, San Francisco, California, 1995.