

Optimal path planning for the motion of a wheel

Jonathan M. Cameron Wayne J. Book

George W. Woodruff School of Mechanical Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332 U.S.A.

Abstract

In this paper, we examine optimal path planning for the motion of a wheel which minimizes some combination of time and effort expended. Several techniques are examined but a combination of continuation and multi-dimensional Newton-Raphson shooting is shown to be effective, within limits.

1 Introduction

Planning the motion of autonomous vehicles is critical for full utilization of their potential, especially when fuel (battery charge) is limited and much maneuvering is to be done. These vehicles may be asked to perform more challenging, less structured tasks in the future than the wire guided vehicles of the past. Examples include inspection of radiation hazards, maintenance of remote installations and manipulation from a mobile base. Most vehicles in common use have nonholonomic behavior, which greatly complicates motion planning. When moving in a structured environment the nonholonomic behavior may not be significant. This research addresses the cases where the nonholonomic constraints are significant. The model of a single wheel explored explicitly here is representative of commercial "omnidirectional" vehicles (e.g. Denning and Cyber Motion vehicles) in common use today for some of these tasks. More conventional front wheel steering is not examined here, but the concepts are relevant with added complexity that obscures the principles to be conveyed.

Unless power is provided by a tether, a mobile robot can only operate for limited periods of time without stopping to recharge. To improve productivity, it is desirable to operate as long as possible without interrupting operations. But it is also important to operate quickly. So in some situations, it is desirable to minimize the time required to complete a motion. In others, it is desirable to minimize the energy used. It

is a worthwhile goal to determine path plans which balance these two objectives and produce useful motions.

In this paper, we consider the motions of a vehicle as represented by a single wheel. We would like to plan optimal motions for the wheel from a starting configuration to a goal configuration. A configuration is determined by three values: two for position and one for angle.

2 Background

Other researchers have developed path planning techniques for wheeled vehicles that address different but related concerns. Most of these works construct paths by piecing together canonical subtrajectories which satisfy some goal of smoothness. Dubins showed that minimum-length paths without cusps can be constructed using combinations of straight line segments and circular arcs [3]. Reeds and Sheep extended this to paths with cusps [8]. *Clothoid* curves provide a smooth transition in steering angles during the transitions from straight lines to circular arcs and have been used extensively to design highway layout. Kanayama introduced *cubic spirals*—which are curves which minimize the integral of the derivative of the curvature along the path [6]. Paths constructed of cubic spirals apparently minimize the side-to-side acceleration during maneuvers.

3 The optimal path planning problem for the wheel

3.1 General problem derivation

The wheel rolls on the x - y plane without slipping (or tipping) but can turn about the point it contacts the plane. The conventions for its coordinates are shown in Figure 1. The forward speed of the wheel

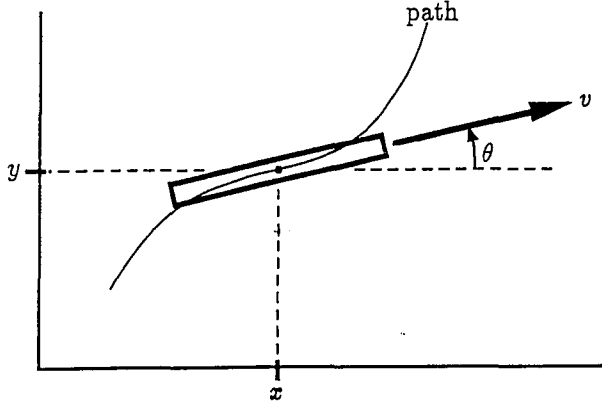


Figure 1: Kinematics of a wheel

is v . The steering angle of the wheel is θ .

The constraint that the wheel can't slip sideways can be represented by two kinematic equations of motion:

$$\dot{x} = v \cos \theta \quad (1)$$

$$\dot{y} = v \sin \theta \quad (2)$$

Note that the motion allowed by this constraint is equivalent to the motion of a keeled sailboat, a pizza cutter, and a knife. In order to put these equations of motion into vector form, let v and $\dot{\theta}$ be inputs, \mathbf{u} , and restate the equations of motion in the form $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})\mathbf{u}$, or more explicitly:

$$\frac{d}{dt} \underbrace{\begin{Bmatrix} x \\ y \\ \theta \end{Bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix}}_{\mathbf{f}(\mathbf{x})} \underbrace{\begin{Bmatrix} v \\ \dot{\theta} \end{Bmatrix}}_{\mathbf{u}} \quad (3)$$

We would like to construct an optimal path from an initial configuration (\mathbf{x}_i) to a goal configuration (\mathbf{x}_g) while minimizing some combination of time and energy expended. We assume the initial configuration is zero ($x_i = 0, y_i = 0, \theta_i = 0$) without loss of generality since the goal configuration can always be found relative to the initial configuration.

A total cost function which trades off the competing desires of minimizing time and minimizing energy is:

$$J = \int_0^{t_f} \left(a + \frac{1}{2}(1-a)(b_1 v^2 + b_2 \dot{\theta}^2) \right) dt \quad (4)$$

where the parameter a ($0 \leq a < 1$) determines whether the tradeoff is biased towards minimizing time (with smaller a) or energy (larger a). In general, minimizing the integral of the sum of the squares of the velocities does not always minimize the energy

expended, but is useful since it is more mathematically tractable and is related to the energy expended. If friction in each degree of freedom is proportional to the velocity for that degree of freedom, then minimizing the integral of the sum of the squares of the velocities is directly related to minimizing the energy expended. This is also a common technique for minimizing the maximum levels of control inputs [2, p. 149].

Following the usual optimal controls approach (see any optimal controls text, such as Bryson and Ho [2]), the cost functional is:

$$L = a + \frac{1}{2}(1-a)(b_1 v^2 + b_2 \dot{\theta}^2) \quad (5)$$

Construct the Hamiltonian by adjoining L with $\mathbf{f}(\mathbf{x})$ by a vector of Lagrange multipliers, $\boldsymbol{\lambda}$ (which are called costates):

$$H = L + \boldsymbol{\lambda}^T \mathbf{f}(\mathbf{x}) \quad (6)$$

$$= a + \frac{1}{2}(1-a)(b_1 v^2 + b_2 \dot{\theta}^2) + \lambda_1 v \cos \theta + \lambda_2 v \sin \theta + \lambda_3 \dot{\theta} \quad (7)$$

In order to optimize the path, it is necessary that $\dot{\boldsymbol{\lambda}} = -(\partial H / \partial \mathbf{x})^T$. Evaluating this produces:

$$\dot{\lambda}_1 = -\frac{\partial H}{\partial x} = 0 \quad (8)$$

$$\dot{\lambda}_2 = -\frac{\partial H}{\partial y} = 0 \quad (9)$$

$$\dot{\lambda}_3 = -\frac{\partial H}{\partial \theta} = v(\lambda_1 \sin \theta - \lambda_2 \cos \theta) \quad (10)$$

Equation (8) implies that λ_1 is constant. Similarly, Equation (9) implies that λ_2 is constant. However λ_3 varies over the path.

For the control to be optimal, it is necessary that $\partial H / \partial \mathbf{u} = 0$ at each instant along the optimal path. This leads to:

$$\frac{\partial H}{\partial v} = 0 = (1-a)b_1 v + \lambda_1 \cos \theta + \lambda_2 \sin \theta \quad (11)$$

$$\frac{\partial H}{\partial \dot{\theta}} = 0 = (1-a)b_2 \dot{\theta} + \lambda_3 \quad (12)$$

Solving Equation (11) for the optimal velocity gives:

$$v^* = -\frac{\lambda_1 \cos \theta + \lambda_2 \sin \theta}{(1-a)b_1} \quad (13)$$

Likewise, solving Equation (12) for the optimal steering rate gives:

$$\dot{\theta}^* = -\frac{\lambda_3}{(1-a)b_2} \quad (14)$$

The costates λ_1 , λ_2 , and λ_3 are influence functions [2, p. 49]: They generate the optimal inputs. Therefore, the full set of optimal kinematic equations of motion are the state equations Equations (1) and (2) and the costate Equations (14) and (13). Integrating this set of equations together guarantees optimal motions. But getting an optimal motion to a desired configuration requires finding exactly the right initial conditions and final time.

Notice that there are four unknowns: λ_1 , λ_2 , λ_{3i} (the initial value of λ_3), and t_f (the final time). The final time must be free since we want to optimize it. Since the final time is free, the Hamiltonian must be zero on the optimum path. Specifically, it must be zero at the initial instant. Letting $\theta = \theta_i = 0$, $v = v_i$ and $\dot{\theta} = \dot{\theta}_i$ in Equation (7) produces:

$$a + \frac{1}{2}(1-a)(b_1 v_i^2 + b_2 \dot{\theta}_i^2) + \lambda_1 v_i + \lambda_3 \dot{\theta}_i = 0 \quad (15)$$

Substituting in the optimal velocities from Equations (13) and (14) and manipulating the result leads to:

$$\frac{\lambda_1^2}{b_1} + \frac{\lambda_{3i}^2}{b_2} = 2a(1-a) \quad (16)$$

which is a quadratic surface in λ_1 and λ_{3i} . Solving this for λ_{3i} and doing a few trial integrations to determine the necessary sign gives:

$$\lambda_{3i} = -\text{sgn}(x_g) \sqrt{b_2(2a(1-a) - \lambda_1^2/b_1)} \quad (17)$$

Since λ_{3i} can be determined from λ_1 and λ_2 , the problem really has only three unknowns: λ_1 , λ_2 , and t_f . To find a solution, it is necessary to determine the values of λ_1 , λ_2 , and t_f which produce an optimal path that ends with $\mathbf{x}_f = \mathbf{x}_g$.

3.2 Straight line solution

In the special case that the wheel is pointing directly towards the goal at the initial instant and the desired final orientation (θ_g) is in the same direction, the optimal path is a straight line and can be solved in closed form. Since the goal is straight ahead, $\theta = 0$ along the entire path. Which means $\dot{\theta} = 0$. That implies $\lambda_3 = 0$ which implies $\lambda_{3i} = 0$. Also, since θ is constant, it is clear from Equation (13) that v is constant during the motion. Solve Equation (13) for λ_1 (letting $\theta = 0$) and substitute it into Equation (15) and solve the result for v^* :

$$v^* = \text{sgn}(x_g) \sqrt{2a/(1-a)b_1} \quad (18)$$

3.3 The nature of the general problem

By appropriate manipulation of Equations (10), (13), and (14), everything can be put in terms of θ in one, higher order, differential equation:

$$\ddot{\theta} = C_1 \sin 2\theta + C_2 \cos 2\theta \quad (19)$$

where C_1 and C_2 are functions of the problem parameters (a , b_1 , b_2 , x_g , y_g , and θ_g). This ordinary differential equation does not have a closed-form solution.

This has important implications. In most of the previous work (as discussed in Section 2), it is possible to solve the underlying problem in a closed form. In our case, it is not possible to construct canonical subtrajectories since there is no closed-form solution to the underlying problem (except in special cases such as a straight line path).

This leads to two possible approaches: (1) solve the problem beforehand (for all relevant cases) and save the solutions for later reuse, or (2) solve the problem on the fly (on a case-by-case basis).

If we solve the problem before hand and create a library of solutions, it is important to know how complex the solution space is. If it is complex it will require significant, possibly excessive, storage. If the solution space simple, the storage requirements are reduced significantly.

If we solve the system on the fly, it is critical to find a technique which can find a solution quickly.

3.4 Example paths

In order to show how the shape of the optimal path depends on the values of the costates, it is useful to see some sample paths. Figure 2 shows a grid of optimal paths for various choices of λ_1 and λ_2 (all with the same final time $t_f = 8$ seconds). The graphs in each column show the path for the same value of λ_1 —which is noted at the bottom of the column. Similarly, every graph in each row shows the path for the same value of λ_2 —which is noted on the left end of the row. Every graph covers the same area ($0.5 \leq x \leq 3.5$ and $0.5 \leq y \leq 3.5$) and is drawn to the same scale for comparison purposes. Each path is optimal because it is generated by integrating the equations of motion which incorporate the optimal inputs.

An examination of this figure reveals several things:

- There is a significant degree of symmetry. The paths to (x, y) have the same shape as paths to $(x, -y)$ —but are flipped about the x -axis and the costates have opposite signs. Therefore, it is adequate to be able to find the path to anywhere in the first or second quadrants since paths to the

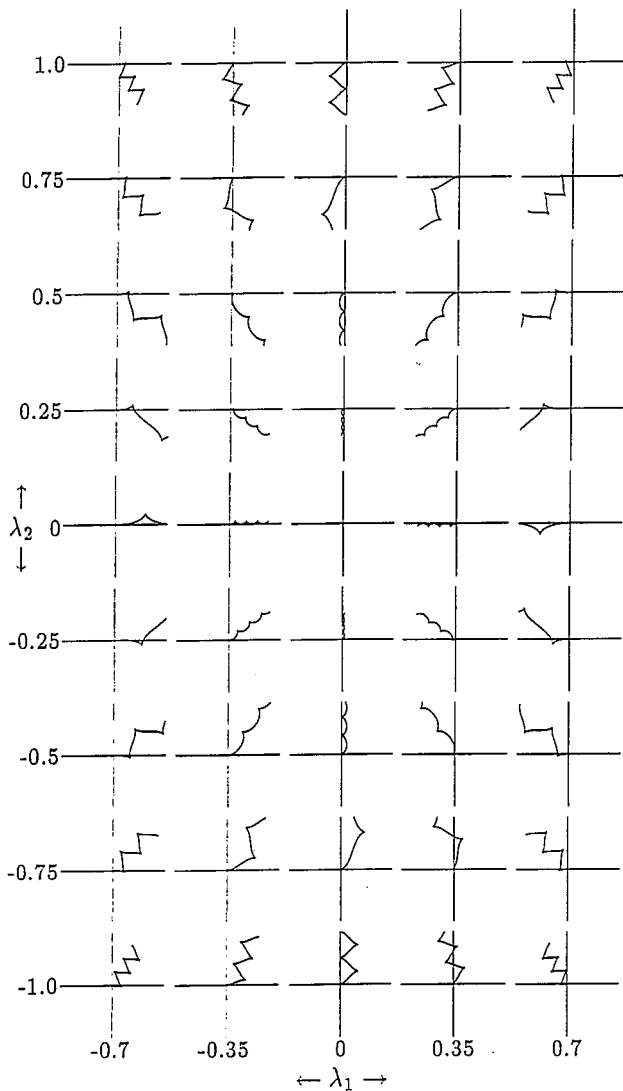


Figure 2: Example optimal paths of a wheel for various λ_1 and λ_2 (with $t_f = 8$ seconds, $a = 1/2$, $b_1 = 1$, and $b_2 = 1$).

third and fourth quadrants can then be found by symmetry.

- Some paths involve backing up.
- The paths are not simple functions (such as circular arcs).
- The general direction of each path of the varies strongly with λ_1 and λ_2 .
- The basic shape of the path varies strongly with λ_1 and λ_2 . For instance, compare the path for $\lambda_1 = 0.35$ and $\lambda_2 = -0.5$ with the path for $\lambda_1 = 0.35$ and $\lambda_2 = -0.75$. The basic direction in which the two paths move are not significantly

different, but the shape of the paths are dramatically different (cycloidal versus zig-zag). So a 50% change in λ_2 can significantly affect the basic shape of the path.

3.5 Search surfaces

One way to consider the problem is to think of it as a search problem in the unknown parameters λ_1 , λ_2 , and t_f . To get a feel for what kind of search is involved, it is helpful to consider a specific case. For instance, suppose we want to find a path to $x_g = 2$, $y_g = 2$, and $\theta_g = 0$ (for which we will use the notation $(2,2,0)$). Fix the final time to t_f^* (the optimum final time), and construct a 3-dimensional surface of the total configuration error. This surface is shown in Figure 3. The search is actually 3-dimensional but

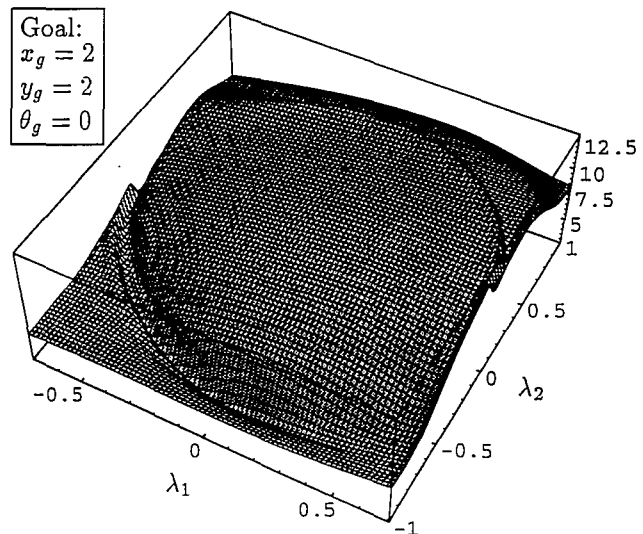
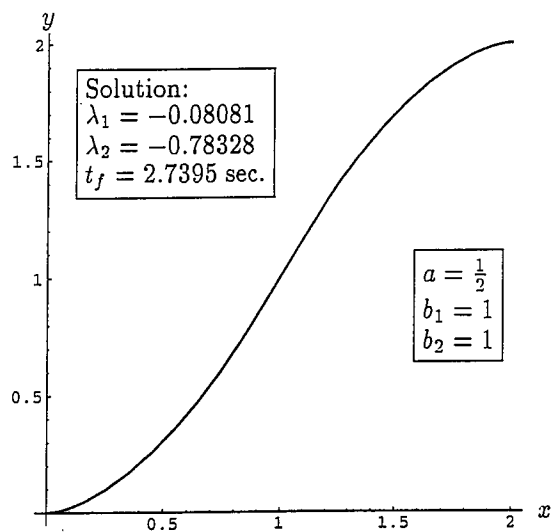


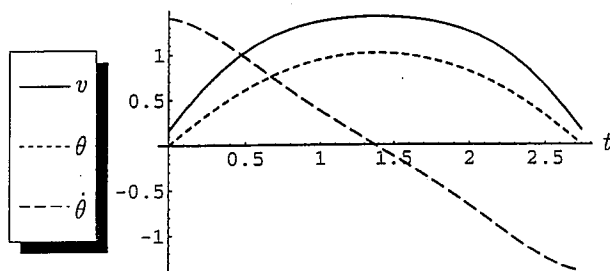
Figure 3: Final configuration error surface ($|x_f - x_g| + |y_f - y_g| + |\theta_f - \theta_g|$) for a path to $(2,2,0)$ with $t_f = 2.7395$ seconds, $a = 1/2$, $b_1 = 1$, and $b_2 = 1$. The solution is located in the groove where $\lambda_1 = -0.08081$ and $\lambda_2 = -0.78328$.

this simplification allows us to get a feel for the nature of the search space in two dimensions.

Since the height of surface represents the final configuration error, the solution will occur only where it goes to zero. The solution for the path to $(2,2,0)$ is located in the groove where $\lambda_1 = -0.08081$ and $\lambda_2 = -0.78328$. Notice that the solution lies in a deep groove. The shape of the path generated using this solution is shown in Figure 4. Figure 4a shows the optimal path to $(2,2,0)$. Figure 4b shows the other variables (v, θ, θ) during the motion.



(a) Optimal path to (2,2,0)



(b) Other variables during the motion

Figure 4: Optimal solution for a path to (2,2,0)

Finding the solution gets harder as the goal gets farther away. For example, if the goal is located at (6,6,0), the cost surface is similar to the one in Figure 3 but the groove is extremely narrow and difficult to find. As is illustrated in Section 3.4, the nature of the path is highly dependent on the costate values. This is because the costates determine the control inputs and because the costates determine the initial conditions. If the goal is farther away, the integration necessary to determine the optimal path gets longer and longer. Integration errors build up and the final configuration becomes more and more sensitive to the initial conditions—which are determined by the costate values, λ_1 and λ_2 .

4 Solution techniques

Several approaches were tried to solve the problem of finding the values of λ_1 , λ_2 , and t_f necessary to get the wheel to a specified goal configuration.

4.1 Shooting

This optimal path planning problem can be treated as a zero-finding problem. We want to solve the set of equations $f_i(\lambda_1, \lambda_2, t_f) = 0$ where:

$$f_1(\lambda_1, \lambda_2, t_f) = x_f - x_g \quad (20)$$

$$f_2(\lambda_1, \lambda_2, t_f) = y_f - y_g \quad (21)$$

$$f_3(\lambda_1, \lambda_2, t_f) = \theta_f - \theta_g \quad (22)$$

where x_f , y_f , and θ_f are determined by integrating the equations of motion (Equations (1), (2), (14), and (13)) from $t = 0$ to t_f with the initial conditions specified by λ_1 and λ_2 .

Notice that it is possible to converge to suboptimal answers using this approach.

A typical approach to this kind of problem is basically multi-dimensional Newton-Raphson and is often called “shooting”. (It is outlined in [10].) For this problem, the necessary derivative information is determined by direct numerical differentiation. This technique converges quite well if the initial guess for λ_1 , λ_2 , and t_f are close enough to the correct solution. For example, consider the example run shown in Table 1. The initial guess is shown on the first line.

t_f	x_f	y_f	θ_f	λ_1	λ_2
3.00000	1.7109	2.0635	-0.3171	0.00000	-0.80000
2.71456	1.8734	2.0187	0.0457	-0.03870	-0.79390
2.73662	1.9993	1.9847	-0.0130	-0.08172	-0.78429
2.73948	1.9999	2.0001	0.0001	-0.08077	-0.78329
2.73949	2.0000	2.0000	-0.0000	-0.08081	-0.78328

Table 1: Output of sample shooting run

In this case, shooting converges cleanly in four shots. If the initial guess is farther afield, the shots jump all around and can get quite wild. For instance, shooting does not converge for the same problem with an initial starting guess of $\lambda_1 = 0$, $\lambda_2 = -1/2$, and $t_f = 2$. But if a particular shot happens to land “close enough” to the solution, it will then converge.

Shooting does not work adequately when the problem gets more difficult. For instance, finding the path to (6,6,0) requires starting extremely close to the correct answer. This is an inherent limitation to the approach of using costates and is not a flaw in the shooting approach itself. (See [2, p. 214] for details.)

4.2 Genetic algorithms

Another popular approach to solving minimization problems is by using “genetic algorithms”. For background information, see Goldberg [4]. This technique uses a population (or set) of guesses which “evolve”

from generation to generation. In each generation, the population is generated, evaluated, and used to generate the population of guesses for the next generation. The fittest guesses of each generation are used to generate the guesses of the next generation. How this is done depends on the type genetic algorithms implemented. We used Grefenstette's public-domain genetic algorithm package called GENESIS 5.0 [5]. The "genes" for the problem are the costates and the final time. The fitness function must return a single value for each set of genes. Our fitness function integrates the system from the initial time to the final time and returns the total cost ($J + |x_f - x_g| + |y_f - y_g| + |\theta_f - \theta_g|$). This approach generally gets moderately close to the globally optimum solution, but is slow.

4.3 Iterative dynamic programming

Dynamic programming is a method for optimization which determines the optimal path through a grid. If the grid is coarse, dynamic programming works reasonably quickly. If the grid is fine enough to get useful, the computation and storage required (even for small problems) quickly gets out of hand due to the "curse of dimensionality." Luus has developed a technique to circumvent this problem to some degree [9]. Initially, the grid is coarse. The optimal path is found in the grid, and then the coarseness of the grid is systematically reduced and the new grid is centered about the previous optimal path. This cycle is repeated until the path is adequately refined. We implemented this iterative dynamic programming technique for the motion of the wheel. It converges reasonably close to the optimal solution although it is slow. Although this approach does not directly yield the initial values of the costates, it is possible to estimate them from the shape of the optimal path near $t = 0$. Unfortunately, the estimates of the costates prove to be very poor. Also, finding this technique does not provide a direct way to find the optimal final time.

4.4 Continuation

Finally, we investigated a continuation technique for solving a set of nonlinear algebraic equations outlined in Kane and Levinson [7]. Bless also uses continuation to find an initial guess for finding the solution to an optimal motion problem using Hamilton's weak principle [1, p. 154]

This technique introduces a continuation variable τ and proposes the following relationship ($i = 1, 2, 3$):

$$f_i(\lambda_1, \lambda_2, t_f) = f_i(k_1, k_2, k_3)(1 - \tau) \quad (23)$$

where the f_i are the same functions as are defined in Equations (20)–(21) and k_i are a set of initial guesses for λ_1 , λ_2 , and t_f . These equations are true for $\tau = 0$ since initially $\lambda_1 = k_1$, $\lambda_2 = k_2$, and $t_f = k_3$. Consider λ_1 , λ_2 , and t_f to be functions of τ . If we could evolve them properly as τ varies from 0 to 1, then the final values of λ_1 , λ_2 , and t_f would be the solution to the original problem since the right hand side of each equation would be zero when $\tau = 1$. Differentiate Equations (23) with respect to τ to obtain the following set of first-order differential equations:

$$\frac{\partial f_1}{\partial \lambda_1} \frac{d\lambda_1}{d\tau} + \frac{\partial f_1}{\partial \lambda_2} \frac{d\lambda_2}{d\tau} + \frac{\partial f_1}{\partial t_f} \frac{dt_f}{d\tau} = -f_1(k_1, k_2, k_3) \quad (24)$$

$$\frac{\partial f_2}{\partial \lambda_1} \frac{d\lambda_1}{d\tau} + \frac{\partial f_2}{\partial \lambda_2} \frac{d\lambda_2}{d\tau} + \frac{\partial f_2}{\partial t_f} \frac{dt_f}{d\tau} = -f_2(k_1, k_2, k_3) \quad (25)$$

$$\frac{\partial f_3}{\partial \lambda_1} \frac{d\lambda_1}{d\tau} + \frac{\partial f_3}{\partial \lambda_2} \frac{d\lambda_2}{d\tau} + \frac{\partial f_3}{\partial t_f} \frac{dt_f}{d\tau} = -f_3(k_1, k_2, k_3) \quad (26)$$

Integrating this set of ordinary differential equations from $\tau = 0$ to 1 will evolve λ_1 , λ_2 , and t_f from the initial guesses to a solution. In order to apply this to the problem at hand, the partial derivatives are computed by varying each parameter slightly and doing direct numerical differentiation. We implemented this technique and it works quickly and converges to answers near the correct solution. It does not generally converge to exactly the correct solution because of approximations in the integration. Figure 5 shows the

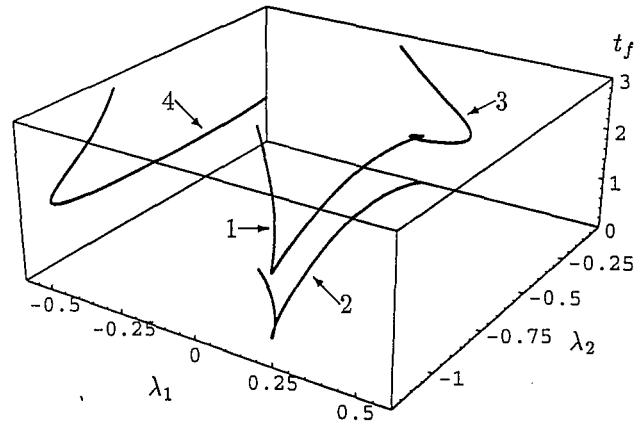


Figure 5: Continuation track for a path to (2,2,0).

The final values are: $\lambda_1 = -0.086068$, $\lambda_2 = -0.783663$, and $t_f = 2.75787$.

track of the variables λ_1 , λ_2 , and t_f as τ varies from 0 to 1 during an example continuation run for finding a path to (2,2,0) with $a = 1/2$, $b_1 = 1$, and $b_2 = 1$. Line 1 is the continuation track in 3 dimensions (λ_1 , λ_2 , t_f). Line 2 is the projection of the continuation

track onto the λ_1 — λ_2 plane, line 3 is the projection onto the λ_1 — t_f plane, and line 4 is the projection onto the λ_2 — t_f plane. The figure shows the convergence process starting from a poor initial guess ($\lambda_1 = 0$, $\lambda_2 = 0$, and $t_f = 1$ second). The final result of the continuation is $\lambda_1 = -0.086068$, $\lambda_2 = -0.783663$, and $t_f = 2.75787$ which is close to the final solution. When these values are used as the starting guess, shooting finds the correct solution in four shots.

If the problem is difficult, continuation does not do very well without careful and lengthy integration. It can also get run into numerical difficulties where the system of Equations (24)—(26) become singular when solving for $d\lambda_1/d\tau$, $d\lambda_2/d\tau$, and $dt_f/d\tau$. This can sometimes be overcome but choosing another starting point (so the continuation track doesn't go to the same problematical place).

The techniques of continuation and shooting were combined to provide a method for finding solutions to optimal path planning for the wheel. The algorithm is to start with continuation from a computer-generated starting guess. Use the result of the continuation as the initial guess for shooting. If the shooting doesn't converge in a reasonable number of shots, use the result of the previous continuation as the starting guess for another round of continuation. Repeat this cycle until a solution is found or a maximum number of cycles are exceeded.

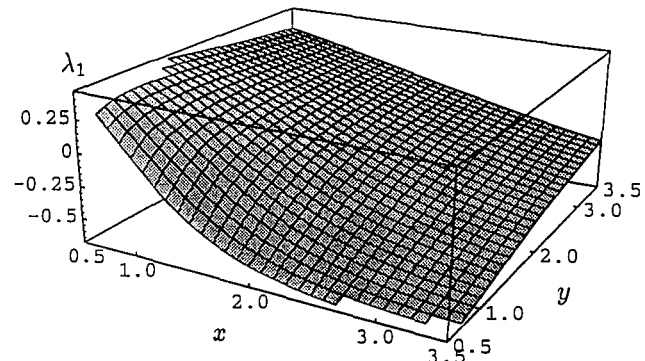
The combination of continuation and shooting work well for planning optimal motions for a wheel within certain ranges of goal distance.

5 The nature of the solutions

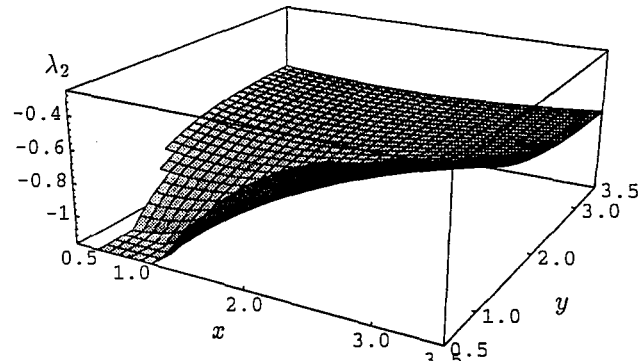
The combined method of continuation and shooting was used to compute the solution to a set of problems for $\theta_g = 0$, $-3.5 \leq x_g \leq 3.5$, and $0 \leq y_g \leq 3.5$. The resulting solutions for λ_1 , λ_2 , and t_g are shown as 3-dimensional surfaces in Figure 6. An important thing to note about these surfaces is that they are relatively smooth. The surfaces are not complex and could be stored compactly.

6 Conclusions

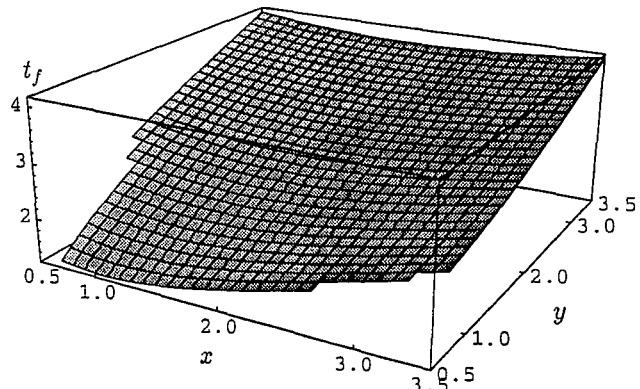
Finding optimal motions for a wheel is difficult but the combination of continuation and shooting works well (within limits). The technique explored in this paper is applicable to other types of motion planning problems, such as planning the motion of robot arms, other types of vehicles, and arms on vehicles.



(a) Surface of λ_1 solutions



(b) Surface of λ_2 solutions



(c) Surface of t_f solutions

Figure 6: Optimal solutions for paths to $(x, y, 0)$

Acknowledgements

This research is supported by the National Science Foundation under grant IRI-9113747.

References

- [1] R.R. Bless, *Time-Domain Finite Elements in Optimal Control with Application to Launch-Vehicle Guidance*, Ph.D. Thesis, Georgia Institute of Technology, January, 1991.
- [2] A.E. Bryson and Y.-C. Ho, *Applied Optimal Control*, John Wiley and Sons, New York, 1975.
- [3] 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*, Vol. 79, 1957, pp. 497-516.
- [4] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publishing Company, Reading, Mass., 1989.
- [5] J.J. Grefenstette, *A User's Guide to GENESIS: Version 5.0*, October 1990. (Available via Internet: <ftp:aic.nrl.navy.mil:/pub/galist/source-code/ga/genesis.tar.Z>)
- [6] Y. Kanayama and B.I. Hartman, *Smooth Local Path Planning for Autonomous Vehicles*, Technical Report, Department of Computer Science, University of California at Santa Barbara, July 1988.
- [7] T.R. Kane and D.A. Levinson, *DYNAMICS: Theory and Applications*, McGraw-Hill Book Company, New York, 1985.
- [8] J.A. Reeds and L.A. Shepp, "Optimal Paths for a Car that Goes Both Forwards and Backwards," *Pacific Journal of Mathematics*, Vol. 145, 1990, pp. 367-393.
- [9] R. Luus, "Optimal control by dynamic programming using systematic reduction in grid size," *International Journal of Control*, Vol. 51, No. 5, 1990, pp. 995-1013.
- [10] W.H. Press, B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C: The art of Scientific Computing*, Cambridge University Press, Cambridge, 1988.