Submitted to the 11th International Conference on Advanced Robotics, June 30 - July 3, 2003, University of Coimbra, Portugal

Efficient Parametric Synthesis of Optimal Mobile Robot Trajectories

Alonzo Kelly

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pa
alonzo@ri.cmu.edu

Abstract

The problem of computing a single continuous trajectory which connects an arbitrary initial posture (position, heading, curvature) with a final posture remains unsolved. Previous approaches have either failed to achieve curvature continuity for the general case or generated sampled solutions which have not been efficient enough for real time use. The polynomial spiral is a generalization of the clothoid which is general enough to represent any feasible vehicle motion with a few coefficients. Starting from an optimal control formulation, the problem of generating a polynomial spiral between two arbitrary postures is transformed naturally into a nonlinear programming problem. This formulation yields to standard numerical approaches which produce trajectories throughout the set of feasible motions in real time.

1. Introduction

The problem of generating mobile robot trajectories is more difficult than it may at first appear. One expression of vehicle dynamics (and of dead reckoning) for a vehicle actuated in curvature-speed and moving in the plane is the system of four nonlinear, coupled, underdetermined differential equations:

$$\begin{aligned} \dot{x}(t) &= V(t)\cos\theta(t) & \dot{\theta}(t) &= \kappa(t)V(t) \\ \dot{y}(t) &= V(t)\sin\theta(t) & \dot{\kappa}(t) &= u(t) \end{aligned} \qquad (1)$$

where the vehicle state vector (also called a posture in this context) consists of the position coordinates (x, y), heading θ, and curvature κ:

$$\underline{x} = (x, y, \theta, \kappa)^T \qquad (2)$$

The input or control vector consists of speed V and desired curvature u:

$$\underline{u} = (V, u)^T \qquad (3)$$

It is straightforward to effect a change of variable from time to distance and express the solution in terms of integrals:

$$\begin{aligned} x(s) &= \int_0^s \cos\theta(s)ds & \theta(s) &= \int_0^s \kappa(s)ds \\ y(s) &= \int_0^s \sin\theta(s)ds & \kappa(s) &= u(s) \end{aligned} \qquad (4)$$

The forward problem is that of determining the state space trajectory given the input trajectory. This is equivalent to dead reckoning and, it can be solved by evaluating the above integrals numerically.

The inverse problem is the trajectory generation problem. It is one of inverting these equations - of finding the inputs $\underline{u}$ which generate the desired output $\underline{x}$. This statement clearly identifies the problem as one of control. When smoothness, curvature limits, and boundary conditions such as terminal posture are considered, it becomes one of optimal control. A related and important view is that of the two point boundary value problem in differential equations. Once the problem is classified in this manner, it becomes natural to assess the degree to which standard approaches are relevant to this case.

1. 1. Previous Work

Given that this problem is one of the fundamental problems in mobile robot control, the relatively little attention it has received must perhaps be attributed to its difficulty and/or the fact that the more general case is dealt with elsewhere. Earlier approaches in [3] and [5] for example are characterized by a curve-fitting formulation where the parameter space of a family of one or more curves of some assumed general form is searched for a solution which satisfies constraints on terminal posture.

While this earlier literature ignores how the curves are computed, the parameter space is of low enough dimension, and computational power was sufficiently limited, that lookup tables would have been a preferred solution. The curve-fitting formulation has over time been refined repeatedly in order to meet more terminal constraints but

it has fallen far short of a general solution.

More recently, variational approaches to the problem have been investigated [2] [8]. In this approach, a continuously deformable representation is adjusted in order to optimize a functional which might consider such factors as smoothness in addition to initial and terminal constraints. This approach can solve a broader class of problems at the expense of increased run time. Of course, optimal curve generation forms the first example application of variational methods in many of the textbooks.

Beyond robotics, the literatures of optimal control, boundary value problems, and curve fitting clearly address related and even identical problems. For example, [1] appeals in its references to early work on the mathematics of shortest paths of bounded curvature and also serves as an example of much work on the related problem of trajectory planning - where sequences of primitives are assembled to achieve some goal.

The relationship between curve fitting an optimal control is expressed in [4] and its references. Here, the use of linear system dynamics to fit curves is first attributed to practitioners in flight control. Of course, the boundary value problem of differential equations is a closely related topic but the deliberate introduction of a dynamic system to define behavior between points of interpolation is attributed here to research writings as recent as 1991.

There is also a growing body of literature which addresses the difficult and specialized topic of nonholonomic motion planning. Recent work in [6] for example, broadens earlier work due to Brockett that applies variational principles to nonholonomic systems. Brockett's work in this area establishes the conditions for the existence of steering solutions between arbitrary initial and final configurations.

1. 2. Approach

The approach presented in this paper is one which combines the strengths of earlier techniques in order to achieve both a highly general formulation and a real-time solution. First, the clothoid and related curves of earlier approaches are generalized to a curvature polynomial of arbitrary order which becomes the assumed form of the solution. Second, the very general optimal control formulation is applied and, based on the assumed form of solution, converted into one of nonlinear programming. Initial steps in this direction were presented in [7].

While the approach is in some sense new, it is also ancient. The method of substituting a family of curves into a differential equation and solving for the parameters is, of course, classical "variation of parameters". Likewise, the use of power series in order to solve differential equations has been employed for centuries.

1. 3. Motivation

While it is typical to approach trajectory generation as an offline problem - to store the representations of every case - this is only possible for curves with few parameters. The satisfaction of constraints on final position, heading and curvature, as well as initial curvature is the minimal requirement for curvature continuity. The storage of the associated five-parameter lookup tables with perhaps 100 samples per degree of freedom requires five arrays of 10 billion floating point numbers.

While interpolation could potentially be used to reduce storage, the solutions must still be computed for the tables. A table of forward solutions could in principle be inverted but the nonlinearity of the equations requires the parameter space to be oversampled significantly leading to unrealistic computing times. Hence, the obvious brute force approaches do not seem to scale well enough even to achieve minimal curvature continuity. This work addresses the computation of the inverse solution which could be either tabulated in inverse form or computed online.

2. Polynomial Spirals

The *polynomial spiral* is an obvious generalization of the clothoids and related curves which have been used historically. An nth order spiral is simply an nth order polynomial expressing curvature in terms of arc length:

$$\kappa(s) = a + bs + cs^2 + ds^3 + \ldots \quad (5)$$

and the clothoid is the special case where the series terminates at the second term. In the complex plane, these curves will be referred to as the *generalized Cornu Spiral*. A representative spiral of cubic order is shown in figure 1.

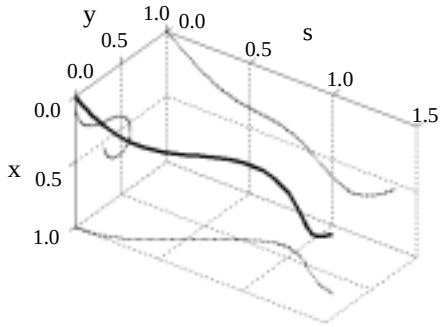

Figure 1: Generalized Cornu Spiral. This curve is generated by a cubic curvature polynomial of the form $\kappa(s) = a + bs + cs^2 + ds^3$. The coefficients are a = 0.0, b = 33, c = -82, d = 41.5. In this example, the primitive reverses curvature and terminates pointing back at the origin.

This new primitive possesses many advantages that can be briefly summarized as the ability to represent any fea-

sible vehicle motion using a small number of parameters. Such a bold statement is easy to justify by noting that all feasible motions have an associated control and the primitive is merely the Taylor series of the control. The Taylor remainder theorem then supplies the basis of the claim that all controls can be represented.

Given that the control in this case represents the actual motion of the steering actuator, it can also be concluded that higher order terms in the series will vanish due to the impossibly high frequencies they imply. A small number of parameters is also valuable from the perspective of representing and communicating the results, but most importantly, it dramatically reduces the dimensionality of the search space.

3. Optimal Control Formulation

This section will develop the conversion of the formulation from optimal control to nonlinear programming.

3. 1. Variational Calculus Formulation

For our problem, there are known nonlinear system dynamics:

$$\dot{\underline{x}} = f(\underline{x}, \underline{u}, t) \tag{6}$$

Let there be a performance index to be optimized which is expressed as some functional evaluated over the trajectory:

$$J = \phi[x(t_f)] + \int_{t_0}^{t_f} L(\underline{x}, \underline{u}, t)dt \qquad t_f \text{ free} \tag{7}$$

Let there be initial and terminal constraints on the states:

$$\underline{h}(\underline{x}, \underline{u}, t_0, t_f) = \underline{x}_b \tag{8}$$

There may also be additional inequality constraints (reflecting such concerns as limited actuator power) on the inputs. For example:

$$\left|\underline{u}(t)\right| \le \underline{u}_{max}(t) \qquad \left|\dot{\underline{u}}(t)\right| \le \dot{\underline{u}}_{max}(t) \tag{9}$$

This is the classical formulation of an optimal control problem in the Bolza form. The principle of optimality yields the solution in classical theory. A vector of Lagrange multiplier functions is used to adjoin the constraints to the performance index to form the scalar function of time known as the Hamiltonian:

$$H(\underline{x}, \underline{\lambda}, \underline{u}, t_0, t_f) = J(\underline{x}, \underline{u}, t_0, t_f) + \underline{\lambda}^T \underline{h}(\underline{x}, \underline{u}, t_0, t_f) \tag{10}$$

First order necessary conditions for a local optimum are the Euler-Lagrange equations:

Recall that the second equation is the transversality condition on final time and the last equation is simply the boundary conditions. This result is repeated for comparison and to introduce the notation of the next section.

$$\begin{aligned}
\frac{\partial}{\partial \underline{x}} H(\underline{x}, \underline{\lambda}, \underline{u}, t_0, t_f) &= \frac{\partial}{\partial \underline{p}} J(\underline{x}, \underline{u}, t_0, t_f) + \underline{\lambda}^T \frac{\partial}{\partial \underline{p}} \underline{h}(\underline{x}, \underline{u}, t_0, t_f) = \underline{0}^T \\
\frac{\partial}{\partial t_f} H(\underline{x}, \underline{\lambda}, \underline{u}, t_0, t_f) &= \frac{\partial}{\partial t_f} J(\underline{x}, \underline{u}, t_0, t_f) + \underline{\lambda}^T \frac{\partial}{\partial t_f} \underline{h}(\underline{x}, \underline{u}, t_0, t_f) = 0 \\
\frac{\partial}{\partial \underline{\lambda}} H(\underline{x}, \underline{\lambda}, \underline{u}, t_0, t_f) &= \underline{h}(\underline{x}, \underline{u}, t_0, t_f) - \underline{x}_b = \underline{0}
\end{aligned} \tag{11}$$

3. 2. Nonlinear Program Formulation

The conversion to a nonlinear program is made as follows:

• assume that the input $\underline{u}$ can be expressed in terms of a parameter vector $\underline{p}$ thus: $\underline{u}(t) = \underline{u}(\underline{p}, t)$.

• note that the input completely determines the state and the parameters completely determine the input so the dependence on both state and input is just a dependence on the parameters.

• transform the independent variable from distance to time.

• set the initial distance to zero and absorb the free terminal distance s_f into an adjoined parameter vector $\underline{q}$ thus $\underline{q} = [\underline{p}^T, s_f]^T$

Under these transformations, the problem statement becomes:

$$\begin{aligned}
&\text{minimize:} \quad J(\underline{q}) = \phi(\underline{q}) + \int_0^{s_f} L(\underline{q})ds \\
&\text{subject to:} \quad \underline{f}(\underline{q}) = \underline{0}
\end{aligned} \tag{12}$$

From the theory of constrained optimization, the solution is obtained by defining the Hamiltonian (often called the Lagrangian in the constrained optimization context):

$$H(\underline{q}, \underline{\lambda}) = J(\underline{q}) + \underline{\lambda}^T \underline{f}(\underline{q}) \tag{13}$$

The first order necessary conditions are:

$$\begin{aligned}
&\frac{\partial}{\partial \underline{q}} H(\underline{q}, \underline{\lambda}) = \frac{\partial}{\partial \underline{q}} J(\underline{q}) + \underline{\lambda}^T \frac{\partial}{\partial \underline{q}} \underline{f}(\underline{q}) = \underline{0}^T \quad \text{(p eqns)} \\
&\frac{\partial}{\partial \underline{\lambda}} H(\underline{q}) = \underline{f}(\underline{q}) = \underline{0} \quad \text{(n eqns)}
\end{aligned} \tag{14}$$

Here, the symbol $\underline{f}(\ldots)$ refers to $\underline{h}(\ldots) - \underline{x}_b$ whereas it was the system dynamics function above. The symbol overloading is an attempt to be consistent with standard notation in both theories. It was considered necessary to derive this result from optimal control theory because classical constrained optimization does not obviously treat the case where the performance index, though admittedly a function of the parameters, is in fact defined by an integral. Nor does it admit any equivalent of the transversality condition necessary for the present purpose.

A major difficulty is hidden beneath the notation. The

boundary condition expression $\underline{f}(\underline{q}) = \underline{0}$ elaborates into the coupled system of integrals:

$$\begin{aligned} x(\underline{q}, s) &= \int_0^s \cos[\theta(\underline{q}, s)]ds \quad \kappa(\underline{q}, s) = u(\underline{q}, s) \\ y(\underline{q}, s) &= \int_0^s \sin[\theta(\underline{q}, s)]ds \quad \theta(\underline{q}, s) = \int_0^s \kappa(\underline{q}, s)ds \end{aligned} \tag{15}$$

Which means that $\frac{\partial}{\partial \underline{q}}\underline{f}(\underline{q})$, being the partial derivative of an integral (for x or y) which depends on the partial derivative of another integral (for θ), is far from straightforward to compute. Although this general case is treatable numerically, the introduction of the polynomial spiral simplifies the situation.

3. 3. Polynomial Spiral Formulation

The polynomial spiral is integrable in closed form to generate an explicit expression for heading as well as curvature:

$$\begin{aligned} \kappa(s) &= a + bs + cs^2 + ds^3 + \dots \\ \theta(s) &= as + \frac{bs^2}{2} + \frac{cs^3}{3} + \frac{ds^4}{4} + \dots \end{aligned} \tag{16}$$

The heading expression can then be substituted into the position integrals and the whole system becomes decoupled:

$$\begin{aligned} x(s) &= \int_0^s \cos\left[as + \frac{bs^2}{2} + \frac{cs^3}{3} + \frac{ds^4}{4} + \dots\right]ds \\ y(s) &= \int_0^s \sin\left[as + \frac{bs^2}{2} + \frac{cs^3}{3} + \frac{ds^4}{4} + \dots\right]ds \end{aligned} \tag{17}$$

Whereas in the general case, the solution integral for heading would have to be evaluated before or in parallel with the position integrals, now all four equations are independent. Each is a nonlinear function of the parameters (when terminal arc length is considered a parameter).

Clearly, any technique for solving the first order conditions will require an understanding of first order behavior. The partials of the heading and curvature are immediate. In general Leibnitz rule supplies the mechanism for computing the partial derivatives of integrals. Defining:

$$C^n(s) = \int_0^s s^n \cos\theta(s)ds \quad S^n(s) = \int_0^s s^n \sin\theta(s)ds \tag{18}$$

Then, the position integrals are:

$$x(s) = C^0(s) \qquad y(s) = S^0(s) \tag{19}$$

The partial derivative with respect to the parameters satisfy useful recurrences:

$$\begin{aligned} \frac{\partial}{\partial s}C^n(s) &= s^n\cos\theta(s) \quad \frac{\partial}{\partial s}S^n(s) = s^n\sin\theta(s) \\ \frac{\partial}{\partial a}C^n(s) &= -S^{n+1}(s) \quad \frac{\partial}{\partial a}S^n(s) = C^{n+1}(s) \\ \frac{\partial}{\partial b}C^n(s) &= -\frac{1}{2}S^{n+2}(s) \quad \frac{\partial}{\partial b}S^n(s) = \frac{1}{2}C^{n+2}(s) \\ &\dots \qquad\qquad\qquad \dots \end{aligned} \tag{20}$$

4. Implementation and Results

Given the nonlinearity of the equations to be solved, the mechanism for numerical solution is linearization. Newton's method as it applies to constrained optimization is used. The derivation follows but first let the parameter vector for the general case be denoted:

$$\underline{q} = \begin{bmatrix} a & b & c & d & \dots & s_f \end{bmatrix}^T \tag{21}$$

Let there be p parameters and n boundary conditions.

4. 1. Linearization

Consider the first order conditions. Transpose the first set of equations, linearize about a point where all equations are not satisfied, and insist that they become satisfied to first order after perturbation. This gives:

$$\begin{aligned} &\frac{\partial^2 H}{\partial \underline{q}^2}(\underline{q}, \underline{\lambda})\Delta\underline{q} + \frac{\partial \underline{f}}{\partial \underline{q}}(\underline{q})^T\Delta\underline{\lambda} = -\frac{\partial}{\partial \underline{q}}H(\underline{q}, \underline{\lambda})^T \quad \text{(p eqns)} \\ &\frac{\partial \underline{f}}{\partial \underline{q}}(\underline{q})\Delta\underline{q} = -\underline{f}(\underline{q}) \quad \text{(n eqns)} \end{aligned} \tag{22}$$

Notation for the Hessian of the Hamiltonian (with respect to $\underline{q}$) was used:

$$\frac{\partial^2 H}{\partial \underline{q}^2}(\underline{q}, \underline{\lambda}) = \frac{\partial^2 J}{\partial \underline{q}^2}(\underline{q}) + \underline{\lambda}\frac{\partial^2}{\partial \underline{q}^2}\underline{f}(\underline{q}) \tag{23}$$

The last term involves a third order tensor. It can be interpreted as a multiplier weighted sum of the n Hessians of each of the individual constraint equations:

$$\underline{\lambda}\frac{\partial^2}{\partial \underline{q}^2}\underline{f}(\underline{q}) = \sum_i \lambda_i \frac{\partial^2}{\partial \underline{q}^2}\underline{f}_i(\underline{q}) \tag{24}$$

In matrix form, this is now of the form:

$$\begin{bmatrix} \frac{\partial^2 H}{\partial \underline{q}^2}(\underline{q}, \underline{\lambda}) & \frac{\partial \underline{f}}{\partial \underline{q}}(\underline{q})^T \\ \frac{\partial \underline{f}}{\partial \underline{q}}(\underline{q}) & 0 \end{bmatrix} \begin{bmatrix} \Delta\underline{q} \\ \Delta\underline{\lambda} \end{bmatrix} = \begin{bmatrix} -\frac{\partial}{\partial \underline{q}}H(\underline{q}, \underline{\lambda})^T \\ -\underline{f}(\underline{q}) \end{bmatrix} \tag{25}$$

This is finally in a form suitable for computer implementation. The constraint satisfaction case (where there is

no performance index) reduces to:

$$\left[\frac{\partial f}{\partial \underline{q}}(\underline{q})\right]\Delta\underline{q} = -f(\underline{q}) \tag{26}$$

This case was discussed earlier in [6].

4. 2. Numerical Solution

In the present implementation, Simpson's rule is used to perform all of the integrations numerically. In the worst cases, the integrands are sinusoids. These are usually integrated over less than one revolution. Hence the integrands are quite smooth and can be estimated well numerically in as little as 10 integrand evaluations.

The nonlinear nature of the problem means that implementations are exposed to the failure mode of falling into a local rather than the desired global minimum. Some relief is available through the use of the Levenberg-Marquardt modification to Newton's method which widens the radius of convergence and permits the solution to be completely determined by the initial estimate. A good initial estimate is therefore critical to a robust solution.

4. 3. Performance Index

Experimental validation was performed by defining a performance index of the form:

$$J_\kappa(\underline{q}) = \frac{1}{2}\int_0^{s_f}[\kappa(\underline{q})]^2 ds \tag{27}$$

The intention is to discourage curves of high curvature and to use any additional parametric degrees of freedom (above what is necessary to satisfy the boundary conditions) to reduce the curvatures required to achieve the goal posture.

In order to express the parameter gradients, it is useful to define the following general gradient integral:

$$K^n(s) = \int_0^s s^n \kappa(s) ds \tag{28}$$

For polynomial spirals, these integrals can be evaluated in closed form:

$$\begin{aligned} K^0(s) &= \theta(s) = as + \frac{b}{2}s^2 + \frac{c}{3}s^3 + \dots \\ K^1(s) &= a\frac{s^2}{2} + b\frac{s^3}{3} + c\frac{s^4}{4} + \dots \\ &\dots \\ K^n(s) &= a\frac{s^{n+1}}{n+1} + b\frac{s^{n+2}}{n+2} + \dots \end{aligned} \tag{29}$$

The gradients of the smoothness performance index can now be written in terms of these gradient integrals:

$$\frac{\partial J_\kappa}{\partial \underline{p}} = \int_0^{s_f} \kappa \begin{bmatrix} 1 \\ s \\ s^2 \\ \dots \end{bmatrix}^T ds = \begin{bmatrix} K^0(s_f) \\ K^1(s_f) \\ K^2(s_f) \\ \dots \end{bmatrix}^T \qquad \frac{\partial J_\kappa}{\partial s_f} = \frac{1}{2}\kappa(\underline{p}, s_f)^2 \tag{30}$$

The gradients of these integrals satisfy:

$$\begin{aligned} \frac{\partial}{\partial a}K^n(s) &= \frac{s^{n+1}}{n+1} \qquad & \frac{\partial}{\partial s}K^n(s) &= s^n\kappa(s) \\ \frac{\partial}{\partial b}K^n(s) &= \frac{s^{n+2}}{n+2} & & \\ &\dots & & \end{aligned} \tag{31}$$

The Hessian matrix for the performance index is:

$$\frac{\partial^2 J_\kappa}{\partial \underline{q}^2} = \begin{bmatrix} s_f & s_f^2/2 & \dots & \kappa_f \\ s_f^2/2 & s_f^3/3 & \dots & s_f\kappa_f \\ \dots & \dots & \dots & \dots \\ \kappa_f & s_f\kappa_f & \dots & \kappa_f\kappa'_f \end{bmatrix} \tag{32}$$

4. 4. Results

Figures 2 and 3 present results on a curvaceous trajectory chosen so as to illustrate the operation of the constrained optimization formulation. The 5 parameter case

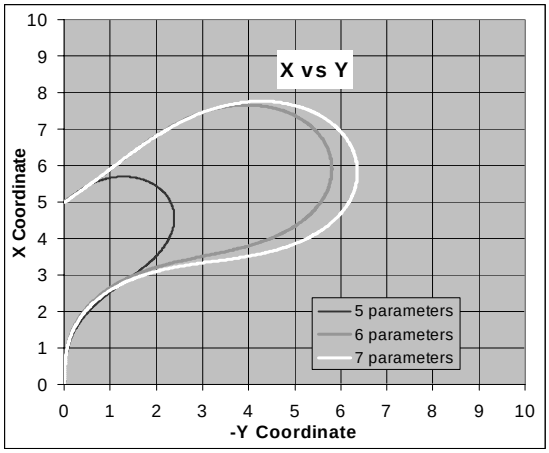

Figure 2: Trajectories for Varying Number of Parameters. These three curves all terminate at the same posture but smoothness increases with the number of parameters used.

is computed in 3 milliseconds and the other two cases in 20 milliseconds on a 1 GHz Pentium 4 processor.

The trajectory has no initial curvature and is required to terminate at the posture:

$$[x, y, \theta, \kappa] = \left[5, 0, \frac{3\pi}{4}, 0\right]$$

At least five parameters are required to satisfy the

boundary conditions. The 5 parameter solution therefore retains no free parameters that can be used to improve the performance index. When free parameters become available (in the 6 and 7 parameter solutions), there is an initial dramatic effect on the curvatures used but the effect tapers off quickly.

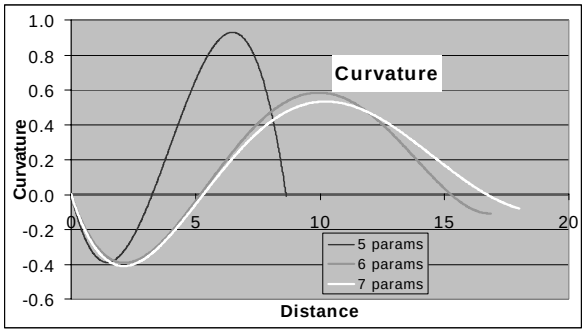

Figure 3: Curvature Profiles for Varying Number of Parameters. These three curves all terminate at the same posture but smoothness increases with the number of parameters used.

The performance index used is essentially the continuous inner product of curvature with itself. This index will favor longer paths over shorter ones to the degree that the area under the squared curvature function is reduced by doing so. Of course lengthening itself has an impact on the performance index so a best compromise is sought. The curvature profiles in Figure 3 show how longer paths can achieve lower overall curvature.

In practice, different performance indices may be more appropriate. Given the need for a numerical implementation, however, any index at all could be accommodated instead of the example used.

4. 5. Pragmas

Two other pragmatic issues are worth mentioning briefly. In addition to the fundamental sensitivity of odometry to small changes in the inputs, the choice of polynomial spiral introduces scaling issues which aggravate matters. For a curve of length of 10 units. A change of unity in the third significant figure of the coefficient of s^3 causes a change of unity in the curvature - which is then integrated twice to determine its effect on the terminal position. At the cost of scaling the roundoff error in the answer, the present implementation scales all problems to place the terminal position on the unit circle and then rescales the answer in order to mitigate this problem.

Secondly, it is very important to realize that all but two of the constraint equations are linear in all parameters but arc length. As a result, it is possible to fix one parameter and the arc length and satisfy all but the position boundary conditions exactly in each iteration. This device has the effect of reducing the search to a relatively well-behaved subspace of parameter space.

5. Summary and Conclusions

The polynomial spiral has been introduced as a natural primitive to express arbitrary steering functions. It possesses the significant benefit of being integrable in closed form for heading, linear in all but one parameter, and of minimum degrees of freedom. These properties are responsible for its ability to convert a general optimal control problem into a derived nonlinear programming problem which finds the parameters of the solution curve. Numerical solution of the associated first order necessary conditions requires the partial differentiation of quadratures, but otherwise, classical numerical approaches apply directly.

This paper has extended the initial work in [7], which produced a real-time algorithm to join arbitrary postures, to include the case where a performance index is also to be optimized.

References

[1] J Boisonnat, A Cérézo, J Leblond, “Shortest Paths of Bounded Curvature in the Plane”, Proc ICRA, pp. 2315-2320, Nice, France 1992

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

[3] Y. Kanayama, N.Miyake, "Trajectory Generation for Mobile Robots", Robotics Research, MIT Press, Cambridge, 1985.

[4] H. Kano, M. Egerstedt, H. Nakata, and C.F. Martin. B-Splines and Control Theory. To appear in Applied Mathematics and Computation, 2003.

[5] K. Komoriya, S. Tachi, and K. Tanie, “A Method for Autonomous Locomotion of Mobile Robots” Journal of the Robotics Society of Japan, vol 2, pp 222-231, 1984.

[6] R. Murray and S. Sastry, “Nonholonomic Motion Planning: Steering Using Sinusoids”, IEEE Trans on Automatic Control, Vol 38, No. 5, May 1993.

[7] B. Nagy and A. Kelly, “Trajectory Generation for Car-Like Robots Using Cubic Curvature Polynomials”, in Field and Service Robots, June 11, 2001, Helsinki, Finland.

[8] Johannes Reuter, “Mobile Robot Trajectories With Continuously Differentiable Curvature: An Optimal Control Approach”, Proceedings of the 1998 IEEE/ RSJ Conference on Intelligent Robots,\