Proceedings of the 1998 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Victoria, B.C., Canada • October 1998

Mobile Robots Trajectories With Continuously Differentiable Curvature: An Optimal Control Approach

Johannes Reuter
TU-Berlin, Department of Electrical Engineering
Control Engineering and System Dynamics Group
Einsteinufer 17 10587 Berlin

Abstract

In this paper we consider the problem of generating smooth trajectories for a fast moving mobile robot in cluttered environment, when online path updating is performed. The well-known smoothing methods using clothoids, cubic spirals or polynominal curves are extended by taking into account the conditions for a continuously differentiable path update. We use the Hamilton-Jacobi framework to create a continuously differentiable interpolation of an existing path of straight lines and arcs of circles. The smoothing approach is combined with the generation of an appropriate velocity profile along the path. The results are compared to those obtained by more classical smoothing methods and to the polynominal approach implemented on the car-like robot MoViLaR [1] of our institute. Finally an embedding of this method into the robot's hierarchical control structure is presented.

1 Introduction

A wide set of path-planning methods for mobile robots results in a sequence of straight lines and arcs of circles [5], [10] [6]. For continuous movement without jerks the links between these primitives have to be interpolated be appropriate curves. Well-known smoothing methods utilize clothoids, cubic spirals or polynominal curves. Many researchers have considered the situation where an interpolation between straight lines is computed. In this case, symmetry of the boundary conditions, i.e. the curvature is zero at the boundary points of the interpolated curve, can be used to create a smooth curve between the lines. Others have utilized the interpolation between lines and arcs of circles. In all these cases, the curvature is zero or has a firmly established value at the boundary points. While moving in unknown, cluttered and dynamically changing environment, online path planning is mandatory, taking into account the dynamical changes of the environment. This requires path updating and smoothing during motion. Then, the assumption that the curvature κ and the derivative of the curvature $\dot{\kappa}$ are zero does not hold anymore. The new smoothed path has to be linked together with the current state of the robot which can obtain any value for κ and $\dot{\kappa}$ since a new path could be available while the robot drives through an interpolated segment of the path. This is the reason, why some of the classical smoothing methods cannot be used in these situations, since neither the curvature and its derivative are zero or fix at the endpoints of the curve, nor are they symmetric. For example, at the mobile robot MoViLaR a modified tangent graph path-planning method is used. Every 500ms a global path update is performed. The path planning is done in the moving local coordinate system (LCS) of the robot, so that the robot is located at the origin. To keep away from obstacles, two additional circles, C_1 and C_2, are put at the origin of the LCS (the dotted circles at the origin in fig.1). The goal is transformed into the local coordinate system and two additional circles are assumed at the goal position to determine the orientation of the robot at this location. To make it possible for the robot to follow such a circle, all radii of the circles have to be chosen with regard to the kinematic constraints $|\kappa| < \kappa_{max}$. Experimental results show that up to speeds of 1 m/s it is sufficent just to consider the segment from the origin to the first tangent point of one of the circles at the origin for smoothing, since in general a new path is available before the robot will have reached the end of the interpolated curve. For movement without slipping, the curvature should not only be continuous, but even continuously differentiable. This is essential with regard to jerks of the steering wheels, causing slippage. For this reason the current derivative of the curvature

[1] Mobile Vision and Laser based Robot

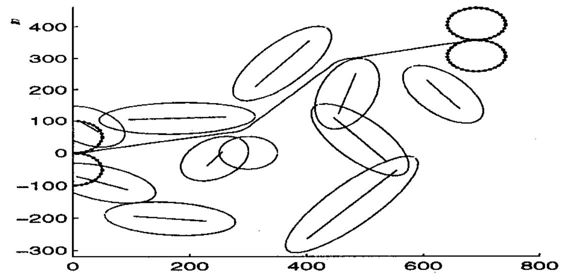

Figure 1: Smoothed tangent graph in a cluttered environment. Units in *cm*.

at the origin has to be considered.
The paper is structured as follows:
In the next section we discuss some of the well-known smoothing methods. In section 3 the problem is formulated as an optimal control problem. It is followed by the solution and the embedding into the hierarchical control structure. Simulation results are compared to those obtained by classical methods. After the conclusion, some detailed computations are presented which we have omitted before due to better readability .

2 Related Work

Clothoides or cornu spirals are well-known smoothing primitives for robot trajectories [1] [7]. Their outstanding property is that the curvature is an affine function of the arc length l.

$$\kappa(l) \quad = \quad k_c l + \kappa_0$$

They are often used as symmetric pairs where the curvature is zero at the boundary points. Their main drawback is that there is no closed solution for the position coordinates of the path since one has to solve the Fresnel integrals. However, approximative closed solutions and methods for fast computation of the Fresnel integrals based on Taylor series expansion are available [7].
Kanayama and Hartman [3] suggested so called cubic spirals to connect postures, i.e. the end-points of two lines of a given path. They used calculus of variation to obtain a path with minimum curvature for a given arc length L with cost functions

$$J_\kappa(L) = \int_0^L \kappa_{(l)}^2 \, dl; \quad J_{\dot\kappa}(L) = \int_0^L \dot\kappa_{(l)}^2 \, dl;$$

In some special cases, closed form solutions are achievable. They also assumed that the boundary conditions for the curvature are zero.
Vandorpe and van Brussel [10] used a pair of polynominal curves to smooth a given path. They utilized parameter optimization of an 4th order polynom with coefficients a, b, c, d, e and parameter s to minimize the criterion

$$\min_{a,b,c,d,e} \left[\max_s |\kappa(s)| \right]$$

subject to certain constraints. A minimum of curvature is obtained only for symmetric boundary conditions. No specified derivative of the curvature is considered.
Murray [8] used differential geometric methods and sinusoidal inputs to achieve smooth trajectories for a nonholonomic mobile robot. Emphasizing the more theoretical control engineering point of view, no optimality subject to the curvature was considered in his paper.
Scheuer and Fraichard [9] used clothoids as well to smooth the links between tangents and arcs of circles and vice versa. The starting-point of their approach is similar to ours, but no path-update in any situation and only continuous curvature is considered. Fig.2 shows different smoothing curves obtained by classical smoothing methods and by the approach described in this paper.

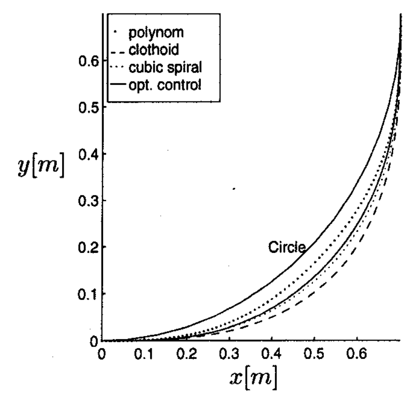

Figure 2: Smoothing curves for a 90 degree turn

3 Outline of the Problem

Given the nonholonomic model of a car-like robot [2]

$$\dot{x} = u\cos(\varphi), \;\; \dot{y} = u\sin(\varphi), \;\; \dot\varphi = u\frac{\tan\theta}{L_R} \qquad (1)$$

with configuration state (x, y, φ) generalized velocity u, steering angle θ and length of the robot L_R, we can state the problem of interpolation between two postures in the following way. Note that $u = \frac{dl}{dt}$ where l is the arc length and the oriented curvature $\kappa = \frac{\tan\theta}{L_R}$. A transformation from time to arc length can then easily be performed. Since we are not only interested in the initial and final values of (x, y, φ), we have to expand the model for two additional states κ and $\gamma = \kappa'$, where $'$ stands for the derivative subject to l, $\frac{d}{dl}$. We obtain an affine nonlinear 5th order model with new input v.

$$\underline{z}'_{(l)} = f(\underline{z}_{(l)}) + g(\underline{z}_{(l)})v_{(l)}, \tag{2}$$

with

$$\begin{aligned} \underline{z}^T_{(l)} &= (x_{(l)}, y_{(l)}, \varphi_{(l)}, \kappa_{(l)}, \gamma_{(l)}) \\ \underline{f}^T(\underline{z}_{(l)}) &= (\cos\varphi_{(l)}, \sin\varphi_{(l)}, \kappa_{(l)}, \gamma_{(l)}, 0) \\ \underline{g}^T(\underline{z}_{(l)}) &= (0,0,0,0,1) \end{aligned}$$

Finding a solution for the original interpolation problem is now stated as looking for an optimal feedforward control law for v which steers the system from the initial point $\underline{z}_0 = \underline{z}(l)|_{l=0}$ to the goal point $\underline{z}_1 = \underline{z}(l)|_{l=L}$. To guarantee minimum of curvature and to keep the input bounded

$$J(L, \underline{z}_0, \underline{z}_1) = \int_0^L (\alpha\kappa^2 + \beta v^2) dl \tag{3}$$

is established as a cost function. No explicit limitation for κ is taken into account. Thus the use of the Maximum Principle of Pontryagin, which could lead to noncontinuously differentiable curvature can be avoided. In section 6 we describe a possible way to although meet the boundaries on κ.

4 Solution of the Optimization Problem

The optimization problem derived in the last section can be solved by the Hamilton-Jacobi framework. Using the method of multiplier and omitting the argument we obtain the Hamilton function

$$H(\underline{z}, \underline{\Lambda}, v) = -(\alpha\kappa^2 + \beta v^2) + \underline{\Lambda}^T(\underline{f}(\underline{z}) + \underline{g}(\underline{z})v) \tag{4}$$

where $\underline{\Lambda}$ is the vector of Lagrangian multipliers which are functions of l. Using

$$\frac{\partial H(\underline{z}, \underline{\Lambda}, v)}{\partial v} = 0$$

and solving for v we obtain the canonical differential equations independent of v (please refer to the appendix for a detailed derivation).

$$\underline{\Lambda}' = -\frac{\partial H(\underline{z}, \underline{\Lambda})}{\partial \underline{z}} \tag{5}$$

$$\underline{z}' = -\frac{\partial H(\underline{z}, \underline{\Lambda})}{\partial \underline{\Lambda}} \tag{6}$$

Since the optimization problem has no specified value for L, one has to consider the transversality condition which leads to an additional equation $H(\underline{z}, \underline{\Lambda})|_L = 0$. The number of ODEs has to be the same as the number of boundary conditions, thus we need an additional differential equation. Since L has to be constant the 11th ODE is obtained by $L' = 0$. Finally a substitution is done, setting $l = Ls$ where $s \in [0 \dots 1]$. The canonical differential equations together with the points $\underline{z}_0$ and $\underline{z}_1$ describe an 11th order 2-point boundary-value problem. Actually, what the 2-point boundary-value problem solver has to do, is to determine the optimal inital values of the Lagrangian multipliers. We use a multiple shooting method [4] with modified Newton-Raphson algorithm to solve the problem.

4.1 Finding a Start Solution

A good start solution is substantial for fast convergence of the multiple shooting method. We use a 4th order polynom for y and a 6th order polynom for x to meet the boundary conditions, see appendix B. This choice seemed to work better instead of using one 10th order polynom, or two five order polynoms, since sensitivity is extremely rising by taking higher order polynoms. From these polynoms the values of the state vector $\underline{z}$ at 6 shooting nodes can be computed. The values for an initial guess of the multipliers are determined in the following way. λ_x and λ_y have to be constant and were chosen to -1000. A good initial guess for L can be made by a rather rough numerical integration of

$$\hat{L} = \int_0^1 \sqrt{\frac{dx}{ds}^2 + \frac{dy}{ds}^2}\, ds$$

Although direct computation of $\lambda_\gamma = \gamma' \frac{2\beta}{L}$ and $\lambda_\kappa = \frac{\gamma'}{L}$ by derivation seems to be possible, better convergence has been achieved by setting $\lambda_\gamma = 600$ and $\lambda_\kappa = 0$. With this start solutions, a computational time less than 300ms on a 200MHz Pentium Computer was achieved in all cases.

5 Feedforward Control

After solving the 2-point boundary-value problem we extract the optimal time dependent feed forward control input $\theta(t) = \arctan(\kappa(t)L_R)$ as follows. Given the optimal initial value of the canonical differential equations one can easily integrate the system subject to the parameter s from 0 to 1. Since $s(t) = \frac{1}{L}\frac{dl}{dt} = \frac{1}{L}u(t)$, for a given sample time T_s the step size at time instance t_k is simply given by $\Delta s_k = \frac{T_s}{L}u_{(k-1)}$. Sufficent accuracy can be obtained by using a 4th order Runge-Kutta method without any step size control. The advantage is that, given an appropriate velocity, the associated steering angle can be easily computed by integrating the canonical system with step sizes depending on the sample time and the velocity. Of course, the trajectory has to be stabilized. We use a flatness based controller as it can be found in [2]. As a reference input $\dot{x}, \ddot{x}$ and $\dot{y}, \ddot{y}$ are needed. These quantities can be easily computed from elements of $\underline{z}$ by using Fresnel's formula.

$$x'' = -y'\kappa; \quad y'' = x'\kappa.$$

The velocity during motion has to be adjusted by taking into account the curvature of the path. Path segments with small radii of turns have to be driven slower than those with small curvature. To this end a very natural choice for the velocity profile is

$$u_{(t)} = \frac{u_{max}}{1 + .5\kappa^2_{(t)}} \tag{7}$$

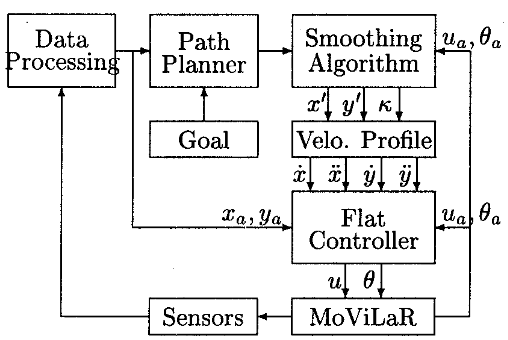

Figure 3: Hierarchical control structure of MoViLaR. The quantities with index $_a$ are the actual measurements.

6 Simulation Results

In the following figure, the shape of the smoothed connecting curve and the curvature and its derivative are compared to those obtained by other smoothing methods. In this simulation a 90 degree turn is performed, see Fig.2. As it is easy to see, using 4th order polynomes, clothoids or cubic spirals causes large jumps in κ' while running through the interpolated curve or at its start- and final-points.

The shape of κ and $\dot{\kappa}$ can be varied by adjusting the

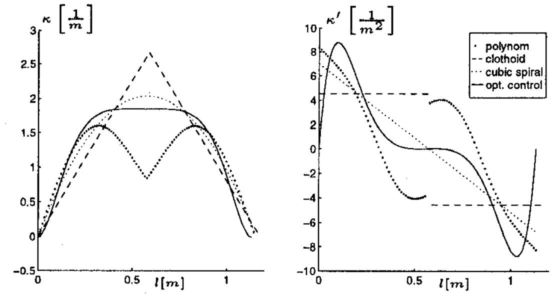

Figure 4: Comparison of κ and κ'

weighting factors. Fig. 5. shows how κ and κ' are influenced by choosing $\alpha = 4000$ and β varies from 0.01 to 1. As it is clear from the cost fnnction, the values of κ decrease as the ratio α/β of the weighting factors increases. This way, the curvature can obviously be reduced. The maximum curvature can be made as small as the curvature of the polynoms in [10]. Then, as a result, the derivative of the curvature increases, but stays continuous.

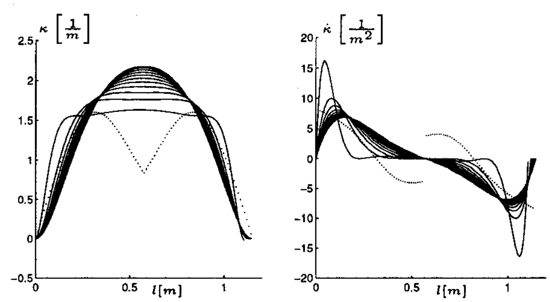

Figure 5: Shape variation by weighting factors adjustment. The dotted curves show the results, using the polynomes of [10].

For the 90 degree turn, in fig.2 the velocity profile, angular velocity and angular acceleration are shown utilizing eq.7 and different smoothing methods.

To demonstrate the improvement utilizing this approach, a path is driven through cluttererd environment using different smoothing methods. The sample time for the controller was chosen as 50ms, and everey 500ms a path update was computed. We use non-continuously (dashed)- and continuously differentiable (dotted) polynomial interpolation and compare it with the optimal control method proposed in this paper (solid). The velocity control is obtained by using eq. 7 with $u_{max} = 1m/s$. As it is easy to see in fig 7, the curvature is smaller and much smoother than in the non-continuously differentiable case, as expec-

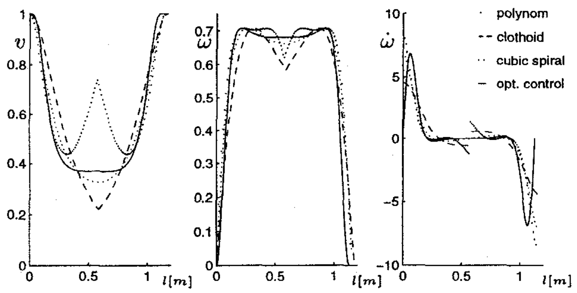

Figure 6: Velocity $v[\frac{m}{s}]$, angular velocity $\omega[\frac{rad}{s}]$ and angular acceleration $\dot{\omega}[\frac{rad}{s^2}]$ using different smoothing methods

ted. The dotted curve shows the results obtained by using higher order polynoms, to match the boundary conditions for κ'. Although no cusps at path update time instances occur, the curvature and its derivative are much higher than in the optimal control case.

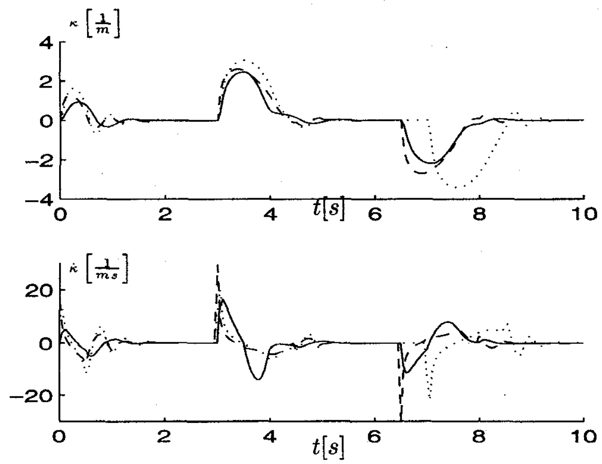

Figure 7: Curvature and its derivative during motion

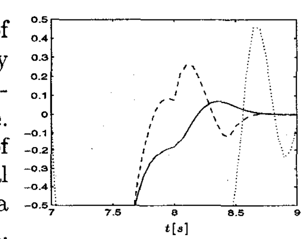

Fig.8: κ at path-update

The enlarged view of fig.8 demonstrates clearly the necessity of continously differentiable curvature. Cusps of the curvature of the 6th order polynomial curve occurs, whenever a path update is performed.

In cases where the first tangent touches one of the circles C_1, C_2 close to the origin and $|\kappa_0|$ is large, the boundaries on $|\kappa|$ are violated. Then for x(1), y(1) we use coordinates located on the tangent instead the tangent point itself. With this heuristical approach the curvature has matched the constraints $|\kappa| < \kappa_{max}$.

7 Conclusion

An algorithm which generates continuously differentiable trajectories inside an online path updating framework was proposed. It was shown that the use of optimal control provides smooth trajectories even in high speed situations. The computational effort is higher than for other smoothing methods, but becomes reachable for real-time applications. By adjustment of the weighting factors the curvature of the interpolated path segments can be made as small or even smaller compared to classical smoothing methods, but without the drawback of discontinuity. This makes the approach useful even for off-line path planning. Currently we are working on an implementation of this method on the real plant.

References

[1] S. Fleury, P. Souéres, J.-P. Laumond, and R. Chatila. "Primitives for Smoothing Mobile Robot Trajectories". In *Proceedings of the int. Conf. on Robotics and Automation*, 1993.

[2] M. Fliess, J. Lévine, P. Martin, and P. Rouchon. "Design of Trajectory Stabilizing Feedback for Driftless Flat Systems". In *Proceedings of the 3rd European Control Conference*, pages 1882–1887, 1995.

[3] Y. Kanayama and B.I. Hartman. "Smooth Local Path Planning for Autonomous Vehicles". In I.J. Cox and G.T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 62–67. Springer Verlag, 1990.

[4] H.B. Keller. *"Numerical Methods for Two Point Boundary-Value Problems"*. Blaisdell Publishing Company, München Wien, 1968.

[5] J.C. Latombe. *"Robot Motion Planning"*. Kluwer Academic Publisher, Norwell Dordrecht, "dritte" edition, 1993.

[6] J.-P. Laumond, Paul E. Jacobs, M. Taïx, and Richard M. Murray. "A Motion Planner for Nonholonomic Mobile Robots". *IEEE Transaction on Robotics and Automation*, 12(6):869–880, December 1996.

[7] H. Makino and H. Suda. "Continous Path Control Using Clothoidal Interpolation". In *Proceedings of the 20th int. Symp. on Industrial Robots*, 1989.

[8] R.M. Murray and S.S. Sastry. "Nonholonomic Motion Planning: Steering Using Sinusoids". *IEEE Transaction on Automatic Control*, AC-38(5), 1993.

[9] A. Scheuer and Th. Fraichard. Continous-Curvature Path Planning for Multiple Car-Like Vehicles. In *IEEE International Conference on Intelligent Robotis and System*, 1997.

[10] J. Vandorpe. *"Navigation Techniques For The Mobile Robot LiAS "*. PhD thesis, Katholieke Universiteit Leuven-Faculteit Toegepaste Wetenschappen, 1997.

A Canonical equations

Given a system

$$\underline{x}'_{(l)} = f(\underline{x}_{(l)}, \underline{u}_{(l)})$$

and a cost function

$$J(\underline{x}_{(l)}, \underline{u}_{(l)}) = \int_0^L h(\underline{x}_{(l)}, \underline{u}_{(l)}, l)dl$$

the Hamilton equation is obtained by

$$\begin{aligned} H(\underline{x}_{(l)}, \underline{\Lambda}_{(l)}, \underline{u}_{(l)}) &= -h(\underline{x}_{(l)}, \underline{u}_{(l)}, l)) + \\ &\quad \underline{\Lambda}^T_{(l)} f(\underline{x}_{(l)}, \underline{u}_{(l)}) \end{aligned}$$

The canonical equations are given as

$$\begin{aligned} \underline{\Lambda}'_{(l)} &= -\frac{\partial H_{(\underline{x}_{(l)}, \underline{\Lambda}_{(l)}, \underline{u}_{(l)})}}{\partial \underline{x}_{(l)}} \\ \underline{x}'_{(l)} &= -\frac{\partial H_{(\underline{x}_{(l)}, \underline{\Lambda}_{(l)}, \underline{u}_{(l)})}}{\partial \underline{\Lambda}_{(l)}} \\ 0 &= \frac{\partial H_{(\underline{x}_{(l)}, \underline{\Lambda}_{(l)}, \underline{u}_{(l)})}}{\partial \underline{u}_{(l)}} \end{aligned}$$

If no value for L is specified, the transversality condition has to be considered to get an equation for L. From the Hamilton Jacobian Framework, if no explicit dependency on l occurs, we have

$$H(\underline{x}_{(l)}, \underline{\Lambda}_{(l)}, \underline{u}_{(l)})|_{l=L} = 0.$$

In our case the Hamiltonian is by eq. (4). Solving the 3. canonical equation for v one gets $v = \frac{\Lambda_\gamma}{\beta}$. Replacing v in the other canonical equations, a 10th order ODE system, independend of v, is obtained. Using the transversality condition an additional boundary condition for L is given: $H(\underline{x}, \underline{\Lambda})_{l=L} = 0$ Thus, we need another differential equation for L. Since L is constant on gets

$$L' = 0.$$

After substituting l by Ls, $s \in [0 \dots 1]$ we obtain the canonical equations as

$$\begin{pmatrix} x' \\ y' \\ \varphi' \\ \kappa' \\ \gamma' \\ \lambda'_x \\ \lambda'_y \\ \lambda'_\varphi \\ \lambda'_\kappa \\ \lambda'_\gamma \\ L' \end{pmatrix} = \begin{pmatrix} \cos(\varphi)L \\ \sin(\varphi)L \\ \kappa L \\ \gamma L \\ \lambda_\gamma/(2\beta)L \\ 0 \\ 0 \\ (\lambda_x \sin(\varphi) - \lambda_y \cos(\varphi))L \\ (2\alpha\kappa - \lambda_\varphi)L \\ \lambda_\kappa L \\ 0 \end{pmatrix}$$

subject to the constraints

$$\begin{pmatrix} x(0) \\ y(0) \\ \varphi(0) \\ \kappa(0) - \kappa_0 \\ \gamma(0) - \gamma_0 \end{pmatrix} = \underline{0}; \quad \begin{pmatrix} x(1) - x_1 \\ y(1) - y_1 \\ \varphi(1) - \varphi_1 \\ \kappa(1) \\ \gamma(1) \\ H(1) \end{pmatrix} = 0$$

B Computation of a Start Solution

Since $x(t), y(t)$ is a flat output of the system eq.(1), [2] , all state variables can be obtained by differentation of x, y. Given the boundary conditions at the origin

$$x(0) = 0;\ y(0) = 0; \varphi(0) = 0\ \kappa(0) = \kappa_0; \gamma(0) = \gamma_0$$

and at the final point

$$x(1) = x_1;\ y(1) = y_1; \varphi(1) = \varphi_1;\ \kappa(1) = 0; \gamma(1) = 0$$

and assuming an 4th order polynom for y

$$y(s) = a_y + b_y s + c_y s^2 + d_y s^3 + e_y s^4 \quad s \in [0 \dots 1]$$

and a 6th order polynom for x

$$x(s) = a_x + b_x s + c_x s^2 + d_x s^3 + e_x s^4 f_x s^5 + g_x s^6$$

using

$$\begin{aligned} \varphi &= atan(\frac{y'}{x'}); \qquad \kappa = \frac{x'y'' - x''y'}{(x'^2 + y'^2)^{3/2}}; \\ \kappa' &= \frac{1}{(x'^2 + y'^2)^{5/2}} \{(x'y''' - x'''y')(x'^2 + y'^2)^2 - \\ &\qquad 3(x'y'' - x''y')(x'x'' + y'y'')\} \end{aligned}$$

as necessary conditions we have

$$\begin{aligned} a_y &= 0;\ b_y = 0;\ c_y = \kappa(0)x(0)'/2; \\ d_y &= x(0)'^2\kappa'(0) + 3\kappa(0)x'(0)x''(0);\ a_x = 0. \end{aligned}$$

$x'(0) = b_x$ and $x''(0) = 2c_x$ can be chosen. e_y is computed by

$$e_y = y(l) - c_y + d_y.$$

The boundary conditions $\kappa(1) = 0; \gamma(1) = 0$ are matched if

$$\frac{x'(1)}{y'(1)} = \frac{x''(1)}{y''(1)} = \frac{x'''(1)}{y'''(1)} = tan(\varphi(1)).$$

Given $y'(1), y''(1), y'''(1)$ from the first polynom, $x'(1)$, $x''(1)$, $x'''(1)$ can be computed and the coefficients of the polynom for x are obtained by solving the linear equation

$$\begin{pmatrix} d_x \\ e_x \\ f_x \\ g_x \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 3 & 4 & 5 & 6 \\ 6 & 12 & 20 & 30 \\ 6 & 24 & 60 & 120 \end{pmatrix} \begin{pmatrix} x(1) - b_x - c_x \\ x'(1) - b_x - 2c_x \\ x''(1) - 2c_x \\ x'''(1) \end{pmatrix}$$