Aircraft trajectory optimization with direct collocation using movable gridpoints

Ch. Jänsch, M. Paus
German Aerospace Research Establishment
Institute for Flight Systems Dynamics, 8031 Oberpfaffenhofen, FRG

Abstract

One of the shortcomings of the direct collocation method, as presented in the recent literature, will be addressed in this paper. The major problem area concerns the node placement, i.e. the discretization. A practical approach to improve the quality of the optimal solution with only a slight increase in computing time is to introduce movable nodes that are optimally placed by the algorithm. The principle capabilities of this method will be demonstrated on some general aircraft trajectory optimization problems. In addition, solutions to some undesirable side effects that occur with a pointwise evaluation of path constraints will be presented.

Introduction

Advances in solving constrained nonlinear programming problems have stimulated the development of trajectory optimization algorithms based on parameterizing the control functions since the early 70's. Early publications in this area are [19,4,15]. Newer approaches have been described by [1,7] where sequential quadratic programming (SQP) is applied in conjunction with an indirect method to solve space trajectory optimization problems. Other approaches where SQP methods are used with direct methods are presented in [16,11,3]. Another new approach which parameterizes the state and control functions by piecewise polynomials and employs collocation to satisfy the state differential equations is described in [10,17].

Indirect methods where the optimal control problem is reduced to a boundary value problem by applying the maximum principle have also been available for a long time. An indirect collocation method is described in [6]. The successful and widely used multiple shooting algorithm BNDSCO was first presented in [5] and extended in [2].

The reason why indirect methods are from an engineering point of view less useful than direct methods is that they require for realistic trajectory optimization problems a considerable amount of preliminary analysis to convert an optimal control problem to a boundary value problem. Moreover, subtle changes in the problem statement usually require a significant amount of additional analysis. Unfortunately, indirect methods are needed if solutions of high accuracy are required or the structure of the optimal solution is of primary importance.

Therefore, some refinements of the direct methods have recently been addressed in [12] in an attempt to combine the advantages of direct methods - ease of use, short problem setup time - with some of the features of the indirect methods.

In this paper, extensions of the direct collocation method presented in [10] are described. Then several aircraft trajectory optimization problems will be presented to show some of the advantages of the modifications.

Optimal Control Problem

In order to simplify the presentation, the class of optimal control problems considered in this paper is only based on one-stage systems. The optimal control problem (OCP) can then be described as follows: Given are the equations of motion in the form

$$\dot{x} = f(t, x(t), u(t), \rho) \quad , \quad t_1 \le t \le t_2 \tag{1}$$

where $x \in R^n$ is the state vector, $u \in R^m$ is the control vector and $\rho \in R^p$ is the vector of design parameters.

The objective is to determine the optimal control u(t), the boundary times t_1, t_2 and the design parameters ρ such that the functional

$$I(u(t), t_1, t_2, \rho) = \sum_{j=1}^{2} F^j(x(t_j), t_j, \rho) \tag{2}$$

is minimized. At the initial and final time, boundary conditions of the form

$$\psi^j_{eq}(t_j, x(t_j), \rho) = 0\,,\ \psi^j_{iq}(t_j, x(t_j), \rho) \geq 0 \qquad (3)$$

for $j = 1, 2$ must be satisfied, and path constraints of the form

$$h_{eq}(t, x(t), u(t), \rho) = 0\,,\ h_{iq}(t, x(t), u(t), \rho) \geq 0 \quad (4)$$

for $t_1 \leq t \leq t_2$ may be required. Note that ψ^j and h may be vector valued functions.

As for all direct methods, the above given OCP is reduced to a nonlinear programming (NLP) problem, i.e. a finite dimensional optimization problem. The various methods only differ in the way they transform the given OCP to a NLP problem. The transformation approach used in the direct collocation approach is described in the following section.

Direct Collocation

In direct collocation the OCP (1) - (4) is transformed to a NLP problem by approximating the state $x(t)$ and the control $u(t)$ by piecewise polynomials and introducing collocation conditions to satisfy the ODE system (1). The polynomials are constructed by partitioning the interval $[t_1, t_2]$ into $k-1$ not necessarily equidistant subintervals $t_1 = \tau_1 < \tau_2 < \cdots < \tau_k = t_2$.

In each subinterval the control $u(t)$ is approximated by a linear or piecewise constant function and each component of the state $x(t)$ is approximated by piecewise cubic hermite polynomials that satisfy the following conditions:

$$p_i(t) = a_i + b_i(t-\tau_i) + c_i(t-\tau_i)^2 + d_i(t-\tau_i)^3 \quad (5)$$

$$\begin{aligned} p_i(\tau_i) &= x(\tau_i) \qquad & p_i(\tau_{i+1}) &= x(\tau_{i+1}) \\ \dot{p}_i(\tau_i) &= \dot{x}(\tau_i) \qquad & \dot{p}_i(\tau_{i+1}) &= \dot{x}(\tau_{i+1}) \end{aligned} \qquad (6)$$

where $\tau_i \leq t \leq \tau_{i+1}$, $\dim(p_i) = n$, $i = 1, \ldots, k-1$ and the time derivative of the state evaluated at τ_i is determined by evaluating the right hand side of (1)

$$\dot{x}(\tau_i) = f(\tau_i, x(\tau_i), u(\tau_i), \rho) \qquad i = 1, \ldots, k\,. \qquad (7)$$

From (5) and (6) one obtains the polynomial coefficients by solving a linear system of equations. The resulting polynomials are C^1 throughout $[t_1, t_2]$ and satisfy the differential equations at all nodes τ_i. As collocation checkpoints the center of each interval is used, that is at $\tau^c_i = (\tau_{i+1} - \tau_i)/2$ the polynomials (5) $p^c_i = p(\tau^c_i)$ and the time derivative of the polynomials $\dot{p}^c_i = \dot{p}(\tau^c_i)$ are evaluated to obtain the interpolated value of the state and state time derivative.

As a measure how well the polynomials satisfy the differential equation one defines in each subinterval the collocation error as the deviation of the right hand side of (1) and the time derivative of the polynomials at the collocation checkpoint

$$K_i(x_i, u_i, x_{i+1}, u_{i+1}, \rho) = \dot{p}^c_i - f(\tau^c_i, p^c_i, u(\tau^c_i), \rho) \quad (8)$$

where the subscript i in x_i, u_i, etc. designates that the quantities are evaluated at the i^{th} collocation node τ_i.

The nonlinear programming problem resulting from the direct collocation transformation can be stated as follows. Define the optimization parameter vector $P = \left[x_1^T, u_1^T, x_2^T, u_2^T, \cdots, x_k^T, u_k^T, t_1, t_2, \rho\right]^T$, collect all collocation constraints and all boundary and path constraints of equality type in a single constraint vector C_e

$$C_e(P) = \begin{bmatrix} K_1(x_1, u_1, x_2, u_2, \rho) \\ K_2(x_2, u_2, x_3, u_3, \rho) \\ \vdots \\ K_{k-1}(x_{k-1}, u_{k-1}, x_k, u_k, \rho) \\ \psi^1_{eq}(t_1, x(t_1), \rho) \\ \psi^2_{eq}(t_2, x(t_2), \rho) \\ h_{eq}(P) \end{bmatrix} = 0$$

and all boundary and path constraints of inequality type in a single constraint vector C_i

$$C_i(P) = \begin{bmatrix} \psi^1_{iq}(t_1, x(t_1), \rho) \\ \psi^2_{iq}(t_2, x(t_2), \rho) \\ h_{iq}(P) \end{bmatrix} \geq 0\,,$$

where it is to be noted that all path constraints are only evaluated pointwise at the nodes τ_i instead of transforming them into the usual integral form. The present approach has shown to be numerically better behaved.

In this way the OCP can be transformed into the following NLP:

$$\min_{P \in R^{n_p}} \bar{I}(P) \quad \text{with} \quad C_e(P) = 0\,,\ C_i(P) \geq 0 \qquad (9)$$

where the number of parameters $n_p = k(n+m)+p+2$. The performance of the direct collocation method depends crucially on the performance of the NLP code used for solving (9). Several codes have been checked in [14,13] for realistic trajectory problems and SLSQP [18] as well as NPSOL [9] have shown consistently better performance than others.

Implementation Considerations

Experience during the development of the software has shown that two items strongly influence the numerical behavior of the method. The first is concerned with scaling. This is partly handled by an automatic scaling algorithm that maps all optimization parameters into the range $[-1, 1]$. A function scaling method has not been implemented yet, but can easily

be performed by the user for the particular problem. A good description of useful scaling heuristics is given in [8].

The second and more critical item is concerned with the discretization, i.e. noding. The location as well as the number of nodes are crucial in satisfying a given collocation error tolerance. A static stepsize control algorithm has been added which determines for given initial estimates, i.e. only at the beginning, the location and the number of nodes required to satisfy a given collocation error tolerance. However, this feature does not allow the nodes to be moved dynamically from one iteration to the next and therefore improves the quality of the solution only marginally if the initial trajectory (starting estimate) is far from the optimal solution.

Hence a dynamic noding feature has recently been added that provides a dynamic optimal placement of the nodes. A simple but effective way to implement this feature is to extend the optimization parameter vector with the collocation nodes as follows:

Normalize the time interval $[t_1, t_2]$ to $[0, 1]$ using the transformation $s = (t - t_1)/(t_2 - t_1)$ to obtain fixed boundary times. This allows the algorithm to treat the movable collocation nodes independent from the free boundary times. The normalized time interval is partitioned into $k - 1$ subintervals $0 = s_1 < s_2 < \ldots < s_k = 1$. However, since the nodes $s_2 \cdots s_{k-1}$ are now considered to be optimization parameters, they influence in addition to the states, controls and design parameters the collocation error.

This is a special point that has to be considered in the computation of the constraint vector Jacobian matrix needed by the SQP method, because that matrix has for the collocation constraint gradient part a block band structure that allows an efficient computation of those gradients even when using central differences to approximate the derivates. One can maintain this block band structure by extending the parameter vector as follows:

$$P = \left[x_1^T, u_1^T, s_2, x_2^T, u_2^T, \cdots, s_{k-1}, x_{k-1}^T, u_{k-1}^T, \cdots\right]^T$$

with the rest of the NLP as defined in (9).

That this simple way of creating a dynamic noding feature works well to produce high quality solutions even for a few number of nodes will be shown in the following examples.

Example 1

The first example shown in Fig. 1-2 is a two dimensional minimum time high performance aircraft intercept problem with load factor n and power setting ξ as controls. A low altitude target has to be intercepted in minimum time from a medium altitude cruise condition. Along the entire flight path, several constraints have to be met (n_{max}, $C_{L,max}$, q_{max}). The main problem areas that the optimization has to cope with in this example are the touching point of the dynamic pressure constraint and the associated thrust reduction.

Both the fixed (F) and movable (V) node test cases were started with 13 nodes and identical initial starting estimates. The results shown are the ones achieved after the first regular stop of the algorithm. In addition, for both test cases the stepsize control algorithm incorporated in the computer code was deactivated to avoid the insertion of any additional nodes so that one can only observe the effects of fixed versus movable nodes. The reference solution (R) was produced with the same method, but with higher accuracy requirements, an activated stepsize control algorithm, 45 movable gridpoints, and several continuation runs.

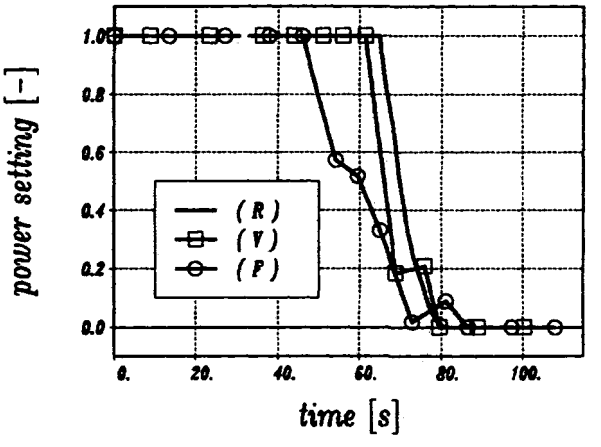

Fig. 1 Time history of power setting. (Ex. 1)

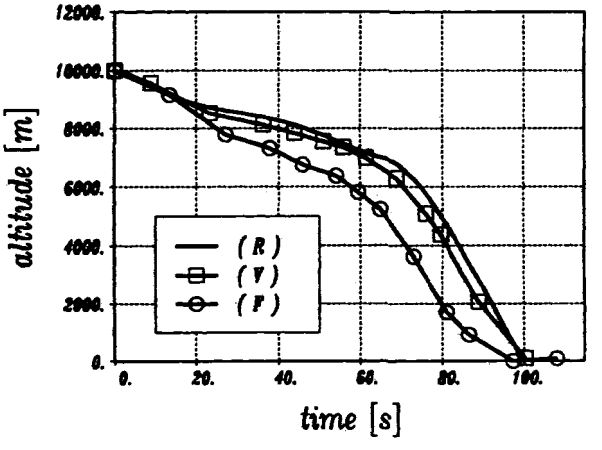

Fig. 2 Time history of altitude. (Ex. 1)

The minimum final time achieved by the reference solution is 99.27 seconds, whereas the fixed node solution only achieved 108.14 seconds and the movable

node solution 100.38 seconds. Fig. 1-2 clearly show that the movable node solution solution is much closer to the reference solution than the fixed node solution. This example is well suited to illustrate that the solution may critically depend on the node placement. In our example, the optimal solution depends critically on the touching point of the dynamic pressure boundary and the associated thrust reduction. The underlying structure of the optimal control can also be better recognized in the movable node solution, since the node location can be adjusted by algorithm so that the thrust reduction occurs at the right moment.

This example shows that an optimal node placement can be achieved to some extent with movable gridpoints. However, this effect can also accentuate an incorrect behavior, when the path constraints are evaluated pointwise. The next example shows that the pointwise evaluation of path constraints is ill-suited for certain type of constraints, such as space-localized path constraints, and can lead to intolerable violation of the constraints between the evaluation nodes.

Example 2

This example shown in Fig. 3-4 is a three dimensional minimum time intercept problem with the horizontal n_h and vertical component n_v of the load factor as controls. The thrust was set constant at its maximum value. The path constraints are, in addition to the ones used in the previous example, two space-localized obstacles with infinite altitude.

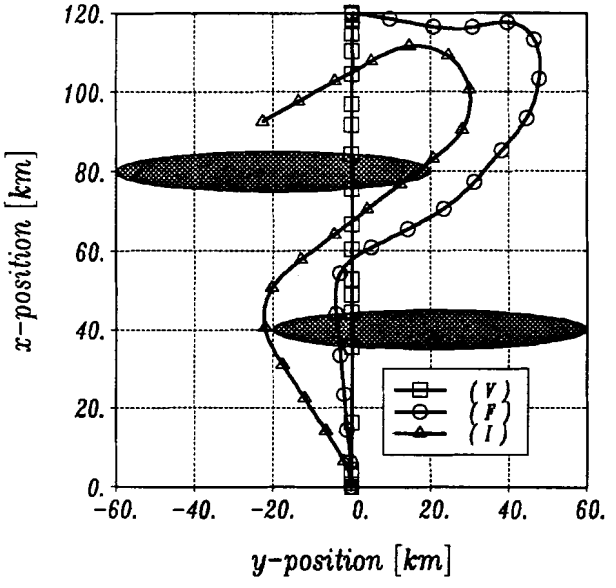

Fig. 3 Ground track of flight path with conventional treatment of path constraints. (Ex. 2)

Fig. 3 shows the ground tracks of the results obtained with 20 fixed and movable gridpoints together with the initial starting estimate (I). The fixed node solution ended at a final time of 360.5 seconds and violated the first obstacle constraint even though from the optimization output data the violation could not be detected because the path constraints are evaluated only at the gridpoints and all gridpoints are outside the restricted area. The reason why the final time is not reduced any further is due to the fact that the nodes are placed relatively to the initial and final time, so that a change in the final time moves the absolute position of all nodes. In our example, a reduction of the final time would lead with the pointwise path constraint evaluation to a constraint violation that the NLP method is unable to resolve.

The ground track obtained with movable gridpoints shows a totally different behavior. It is a straight line through both of the obstacles ending at a final time of 221.5 seconds. Although this result is also not the desired one it clearly shows that the movable gridpoints are much more flexible in adapting themselves to the incorrect problem formulation. The enormous reduction in time is possible because the critical gridpoints could be shifted, circumventing the need to move them through the restricted area.

With fixed gridpoints the problem with the path constraints could in principle be alleviated by introducing additional gridpoints at the critical places. With movable gridpoints that does not help either because they would simply be shifted in such way away that the path constraints are satisfied from a pointwise point of view.

Improved path constraint evaluation

To solve the problem arising from the pointwise evaluation of the path constraint function one can extend the state approximation approach to the constraints. If one approximates the continuous path constraint function in each collocation interval as a function of time by a cubic polynomial, one can analytically determine the maximum constraint violation within this interval. Using this maximum instead of the pointwise evaluated constraint value one obtains much better results.

The way the approximation is done best depends on the constraint function. For our example we used hermite polynomials because it was easy to specify the time derivative of the constraint function. With the value and the time derivative at each interval boundary a hermite polynomial is fully defined.

Fig. 4 shows the results with the improved path constraint evaluation for fixed (FC) and movable (VC) gridpoints. The tracks and final times (about 270 seconds) are almost identical for both cases and the constraints are met perfectly.

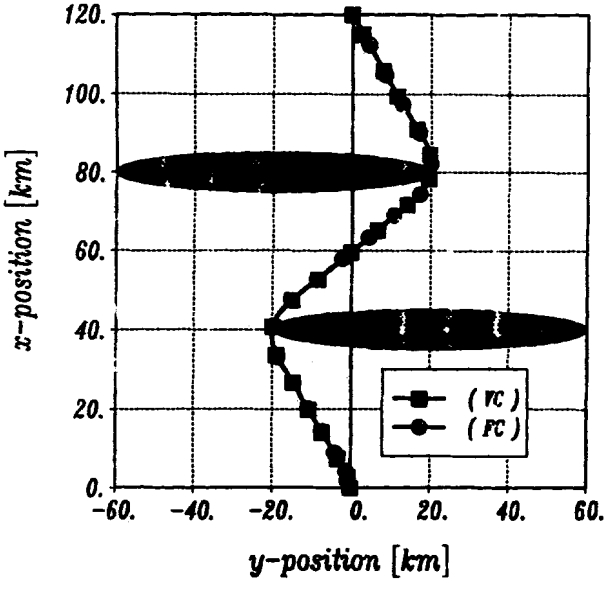

Fig. 4 Ground track of flight path with improved treatment of path constraints. (Ex. 2)

Example 3

Since part of the motivation for introducing movable gridpoints is to use only a few number of nodes in order to save computer time, we will now consider the same scenario with only 13 nodes. The overall results are basically the same as with 20 nodes but the solution with movable gridpoints is over 2 seconds faster than the one with fixed gridpoints and the movable nodes are concentrated at the critical places (the edges of the obstacles). This is even better reflected in the time history of the load factor component n_h, because one can obviously recognize the structure of the optimal control. This example clearly shows that movable gridpoints can yield very good results even if only few gridpoints are used.

Conclusions

Extensions of the direct collocation method concerning the node placement have been presented in conjunction with aircraft trajectory optimization problems. It has been manifested that movable nodes improve the quality of the solution without an noticeable increase in computing time. In addition, it has been demonstrated that features previously reserved to indirect methods, such as the recognition of the optimal solution structure, can in a somewhat cursory way be achieved with movable gridpoints. It has also been shown that the pointwise path constraint evaluation can lead to incorrect results in the sense that the path constraints are significantly violated between the nodes. Satisfactory corrections to this problem were presented in the examples. We believe to have shown that the direct collocation method with the appropriate extensions can be a valuable tool for trajectory optimization. However, there are still many untapped approaches to improve the computational efficiency of the method. Most notably, the direct collocation transformation produces a very special structure in the resulting NLP problem (block band structure of the constraint jacobian) which could be exploited with the appropriate NLP solver to produce an extremely efficient method.

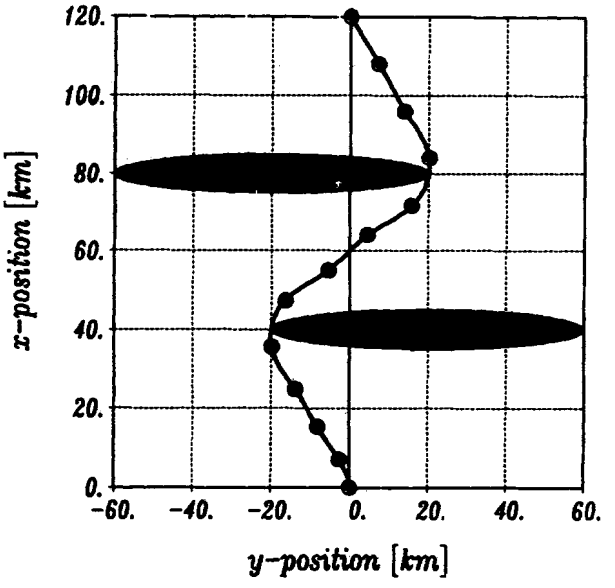

Fig. 5 Ground track with fixed gridpoints (Ex. 3)

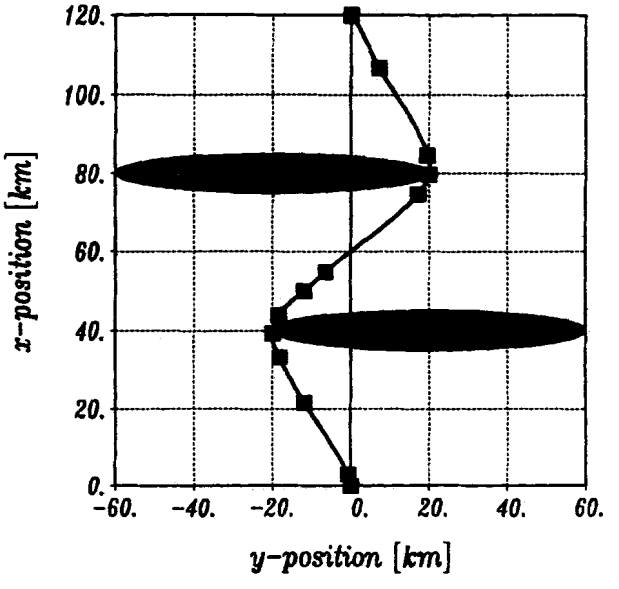

Fig. 6 Ground track with movable gridpoints (Ex. 3)

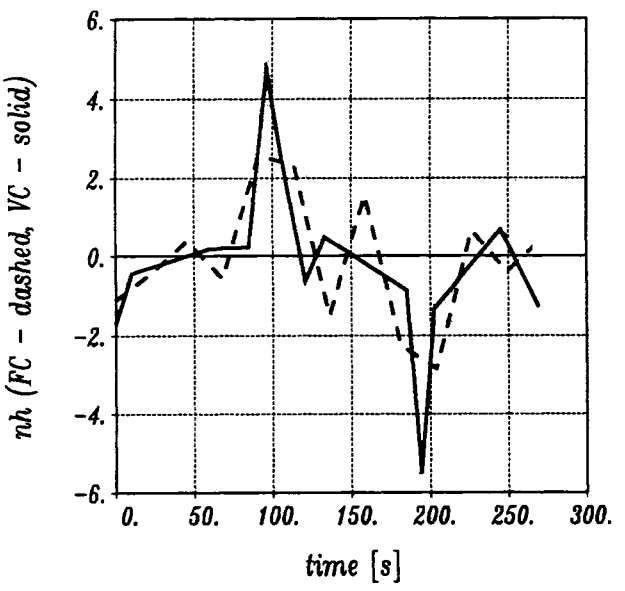

Fig. 7 Time histories of horizontal load factor. (Ex. 3)

References

[1] M. C. Bartholomew-Biggs. Recursive quadratic programming based on penalty functions for constrained minimization. In L. C. W. Dixon, G. P. Szeg, and E. Spedicato, editors, *Nonlinear Optimization Theory and Applications*, Birkhauser, 1980.

[2] H. G. Bock. Numerical solution of nonlinear multipoint boundary value problems with applications to optimal control. *Zeitschrift für Angewandte Mathematik und Mechanik*, 58:407–409, 1978.

[3] H. G. Bock and K. J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *Preprints of IFAC International Federation of Automatic Control*, pages 243–247, Budapest, Hungary, 9th World Congress, July 2-6 1984.

[4] R. G. Brusch. A nonlinear programming approach to space shuttle trajectory optimization. *Journal of Optimization Theory and Applications*, 13:94–118, 1974.

[5] R. Bulirsch. *Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung.* Lehrgang über Flugbahnoptimierung, Carl-Cranz-Gesellschaft, Heidelberg, 1971. Nachdruck: TU München, Math.Institut, 1985.

[6] E. D. Dickmanns and K. H. Well. Approximate solution of optimal control problems using third order hermite polynomial functions. *Springer Lecture Notes*, Com. Sci. 27:158–166, 1975.

[7] L. C. W. Dixon and M. C. Bartholomew-Biggs. Adjoint control transformations for solving practical optimal control problems. *Journal of Optimal Control Applications and Methods*, 2:365–381, 1982.

[8] P.E. Gill, W. Murray, and M.H. Wright. *Practical Optimization.* Academic Press, New York, 1981.

[9] P.E. Gill, W. Murray, and M.H. Wright. *User's Guide for NPSOL (Version 2.1): A FORTRAN package for Nonlinear Programming.* Technical Report SOL 84-7, Systems Optimization Lab., Stanford Univ., Stanford, CA., 1984.

[10] C.R. Hargraves and Paris S.W. Direct trajectory optimization using nonlinear programming and collocation. *J. Guidance and Control*, 10(4), 1987.

[11] M. K. Horn. *A FORTRAN Program for Solving State/Control- Constrained Optimal Control Problems with System Equations Having Expressions Involving Tabular Data.* Internal Report 515-83/1, DLR, Oberpfaffenhofen, FRG, 1983.

[12] M. K. Horn. Solution of the Optimal Control Problem Using Software Package STOMP. In *Preprints of IFAC International Federation of Automatic Control*, Paris, France, 8th Workshop on Control Applications of Nonlinear Programming and Optimization, June 7-9 1989.

[13] C. Jänsch, D. Kraft, K. Schnepper, and K.H. Well. *Survey on Trajectory Optimization Methods.* Technical Report TR1, ESA/ESTEC, 1989. Contract No. A0/1-2161/88/NL/MAC.

[14] C. Jänsch, D. Kraft, and K.H. Well. *Comparative Study of Nonlinear Programming Codes for Trajectory Optimization.* Technical Note TN1, ESA/ESTEC, 1989. Contract No. A0/1-2161/88/NL/MAC.

[15] I. L. Johnson. Optimization of the solid-rocket assisted space shuttle ascent trajectory optimization. *Journal of Spacecraft and Rockets*, 12:765–769, 1975.

[16] D. Kraft. *FORTRAN-Programme zur numerischen Lösung optimaler Steuerungsprobleme.* Internal Report 515-80/3, DLR, Oberpfaffenhofen, FRG, 1980.

[17] D. Kraft. Nonlinear system analysis by direct collocation. In *Lecture Notes in Control and Information Sciences, Optimal Control, Vol. 95*, Springer, 1987.

[18] D. Kraft. *A Software Package for Sequential Quadratic Programming.* Forschungsbericht DFVLR-FB 88-28, DFVLR, Oberpfaffenhofen, 1988.

[19] J. L. Speyer, H. J. Kelley, N. Levine, and F. Denham. *Accelerated Gradient Projection Technique with Application to Rocket Trajectory Optimization.* 1971.