Amaraporn Boonpratatong
Tawiwat Veeraklaew

Department of Mechanical Engineering, Faculty of Engineering, Srinakharinwirot University and Department of Mechanical Engineering, Chulachomkloa Royal Military Academy Jointed Post Graduated Program, Thailand.

THE DIRECT APPROACH OF GENERAL DYNAMIC OPTIMAL CONTROL: NONLINEAR PROGRAMMING AND RUNGE - KUTTA METHOD

Abstract:

In this paper we address issues in the numerical solution of differential - algebraic equations (DAEs) arising from the direct approach of General Dynamic optimal control. First the dynamic equation of motion is solved by using Runge - Kutta technique which is one of the method of the numerical integration. Next, resulting of state and costate variables arising from the numerical integration can be used to transform the direct optimal control problem into the nonlinear programming form. Then the algorithm used to solve the resulting parameterized optimization problem with an emphasis on its interaction to the collocation method is the nonlinear programming method. Finally, the general optimal control software based on this direct approach is developed in order to test problems and compare the result with a similar technique developed by K.E.BRENAN (Brenan, 1993).

Key words: Optimization, Numerical, Nonlinear Programming, Dynamics.

1. Introduction

In the problem of optimal control, the trajectory is determined which satisfies simultaneously equations of motion, boundary conditions, inequality constraints, equality constraints, and a performance index (or cost functional) must be minimized or maximized. There are many criteria in order to be used to solve the optimal control problems such as calculus of variations, minimum principle, matrix exponential, and Hamilton - Jacobi equations. However, these are considered as indirect procedures since the necessary and sufficient conditions must be derived and result in the differential - algebraic equations (DAEs). In this paper, we focus on a direct procedure which is known that the optimal control problems will be converted to a parameter optimization problems. In Section 2, the statement of the problem is described along with a technique that developed by K.E.BRENAN (Brenan, 1993) which is called Hermit - Simpson collocation method. We are proposing a similar technique in Section 3 namely Runge - Kutta collocation method. We believe that by using collocation method, the Hermit technique is not an only one that is necessary to be used for such an accuracy solution. In addition, using Runge - Kutta method that is a forward integrating tool can help in term of how to providing initial guesses. In Section 4, we describe how the general optimal control software is developed. Finally, an example is illustrated in Section 5.

2. Statement of the Problem

The statement of the problem is to find an optimal trajectory in both state and control variables to minimize the cost functional.

$$J = \phi(t_f, x(t_f), u(t_f)) + \int_{t_0}^{t_f} L(t, x(t), u(t))dt \quad (1)$$

such that

$$\dot{x}(t) = f(t, x(t), u(t)), \quad t \in [t_0, t_f] \quad (2)$$

$$\delta(x(t_0)) \leq 0 \quad (3)$$

$$\psi(x(t_f)) \leq 0 \quad (4)$$

$$c(x(t), u(t), t) \leq 0 \quad (5)$$

$$g(x(t), u(t), t) = 0 \quad (6)$$

where $x \in R^n$, $u \in R^m$, $\delta \in R^s$, $\psi \in R^q$, $c \in R^r$, and $g \in R^k$.

Methods for solving optimal control problems can be divided into two basic classes: indirect and direct methods. In the indirect approach, the optimal control problem is transformed into a boundary value problem by formulating the first order necessary conditions for optimality, thereby obtaining the Euler - Lagrange system (Bolza, 1931), (Bryson&Ho, 1975), (Kirk, 1970). In the direct approach, the optimal control problem is approximated by a parameter optimization problem in which the first order optimality conditions are not explicitly included. The Hermit - Simpson formula is used to discretize the optimal control problem to be a parameter optimal control problem, then the nonlinear programming algorithm is used to obtain solution (Brenan, 1993).

We now describe the HS (Hermit-Simpson) method as implement in (Brenan, 1993). Suppose the interval $[t_0, t_f]$ is partitioned into N subintervals such that $t_0 < t_1 < \cdots < t_N = t_f$. Define $h_i = t_i - t_{i-1}$ *for* $i = 1, \ldots, N$. *Let* x_i *and* u_i represent the approximate state and control values respectively at the nodes t_i. Introduce variables w_i to represent weighted derivatives of the controls at the nodes.

- Using Hermit cubic interpolation to represent the solution on each subinterval, and letting $f_p(t,x,u) = f(t,x,u,p)$, estimate the values of the states at the segment centers.

$$y_i = \frac{1}{2}(x_{i-1} + x_i) + \frac{h_i}{8}(f_p(t_{i-1}, x_{i-1}, u_{i-1}) - f_p(t_i, x_i, u_i)) \quad (7)$$

and the controls at the segment centers,

$$v_i = \frac{1}{2}(u_{i-1} + u_i) + \frac{1}{8}(w_{i-1} + w_i) \quad (8)$$

- Evaluate the differential equations at the center of each segment using the interpolated center values, $f_p(\hat{t}_i, y_i, v_i)$, where $\hat{t}_i = (t_{i-1} + t_i) / 2$.
- Integrate across the segment using Simpson's quadrature rule:

$$x_i = x_{i-1} + \frac{h_i}{6}(f_p(t_{i-1}, x_{i-1}, u_{i-1}) + 4f_p(\hat{t}_i, y_i, v_i) + f_p(t_i, x_i, u_i)) \quad (9)$$

- Evaluate the path constraints at the nodes and midpoints:

$$c(t_i, x_i, u_i) \leq 0 \quad (10)$$

$$c(\hat{t}_i, y_i, v_i) \leq 0 \quad (11)$$

$$g(t_i, x_i, u_i) = 0 \quad (12)$$

$$g(\hat{t}_i, y_i, v_i) = 0 \quad (13)$$

Equations (9) through (13) form a nonlinear system of equations for the unknown variables x_i, u_i, and w_i at the nodes as well as for the final time t_f. The parameter optimization problem can now be stated as nonlinear programming.

3. Runge - kutta Collocation Method

In this section, the Runge - Kutta collocation method is described in a similar procedure as Hermit - Simpson collocation technique since the Hermit - Simpson collocation technique and Runge - Kutta are known as the numerical integrating tools. Suppose the interval $[t_0, t_f]$ is partitioned into N subintervals such that $t_0 < t_1 < \cdots < t_N = t_f$. Define $h_i = t_i - t_{i-1}$ *for* $i = 1,\ldots, N$. *Let* x_i *and* u_i represent the approximate state and control values respectively at the nodes t_i . x from equation (2) in the neighborhood of x_i can be expressed in terms of the Taylor series. Letting the time increment be $h = \Delta t$, we have

$$x = x_i + \left(\frac{dx}{dt}\right)_i h + \left(\frac{d^2x}{dt^2}\right)\frac{h^2}{2} + \ldots$$

Instead of using these expressing, it is possible to replace the first derivative by an average slope and ignore higher - derivatives

$$x = x_i + \left(\frac{dx}{dt}\right)_{iav} h$$

If we used Simson's rule, the average slope in the interval h becomes, i.e.

$$\left(\frac{dx}{dt}\right)_{iav} = \frac{1}{6}\left[\left(\frac{dy}{dt}\right)_{t_i} + 4\left(\frac{dy}{dt}\right)_{t_i+h/2} + \left(\frac{dy}{dt}\right)_{t_i+h}\right]$$

The Runge - Kutta method is very similar to the preceding computations, except that the center term of the given equation is split into two terms and four values of t, x and f are computed for each point i as follows:

t	x	$f = \dot{x}$
$T_1 = t_i$	$X_1 = x_i$	$F_1 = f(T_1, X_1)$
$T_2 = t_i + \frac{h}{2}$	$X_2 = x_i + F_1\frac{h}{2}$	$F_2 = f(T_2, X_2)$
$T_3 = t_i + \frac{h}{2}$	$X_3 = x_i + F_2\frac{h}{2}$	$F_3 = f(T_3, X_3)$
$T_4 = t_i + h$	$X_4 = x_i + F_3 h$	$F_4 = f(T_4, X_4)$

These quantities are then used in the following recurrence formula:

$$x_{i+1} = x_i + \frac{h}{6}[F1 + 2F2 + 2F3 + F4] \quad (14)$$

where it is recognized that the four values of F divided by 6 results in an average of dy / dt as defined. At this step, we can evaluate all the path constraints i.e., equations (3), (4), (5), and (6) at the node points as a parameter p instead of x_i. These form a nonlinear system of equations as parameter optimization problems as

$$\min J(p) \quad (15)$$

subject to a set of equation (14) and

$$\delta(x(t_0)) \le 0 \quad (16)$$
$$\psi(x(t_f)) \le 0 \quad (17)$$
$$c(x(t), u(t), t) \le 0 \quad (18)$$
$$g(x(t), u(t), t) = 0 \quad (19)$$

The equations (15) through (19) are known as nonlinear programming problem that the parameter p must be determined in order to obtain the feasible solution to dynamic optimization problem.

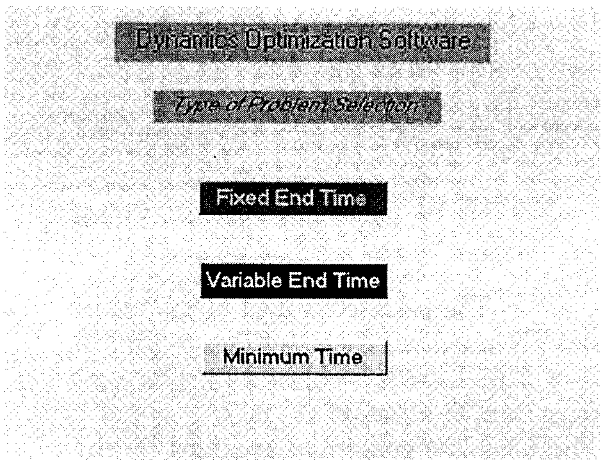
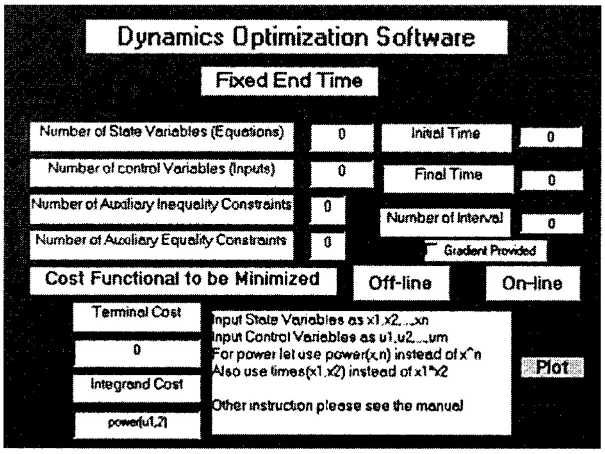


Fig. 1. The Front page

4. General Optimal Control Software

The general - purpose program has been developed using Matlab software. Furthermore, we design it as visual inputs and outputs for an easy implement and use. First, the problem has been divided into 3 categories as (i) Fixed end time, (ii) Variable end time, and (iii) Minimum time as shown in Figure 1.

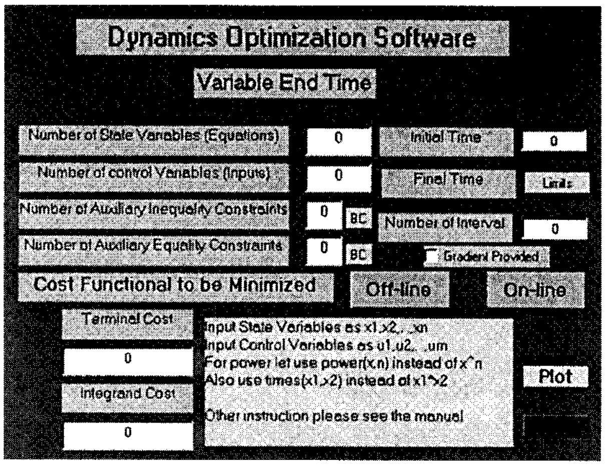
All three categories are designed in the front page as push button; therefore, whenever user pushes the button, the appropriate second window is opened as shown in Figure 2, 3, and 4.


Dynamics Optimization Software
Fixed End Time
Number of State Variables (Equations)
Number of control Variables (Inputs)
Number of Auxiliary Inequality Constraints
Number of Auxiliary Equality Constraints
Initial Time
Final Time
Number of Interval
Gradient Provided
Cost Functional to be Minimized
Off-line
On-line
Terminal Cost
Integrand Cost
Plot

0
0
0
0
0
0
0
0
power(u1,2)

Input State Variables as x1,x2,...xn
Input Control Variables as u1,u2,...um
For power let use power(x,n) instead of x^n
Also use times(x1,x2) instead of x1*x2
Other instruction please see the manual

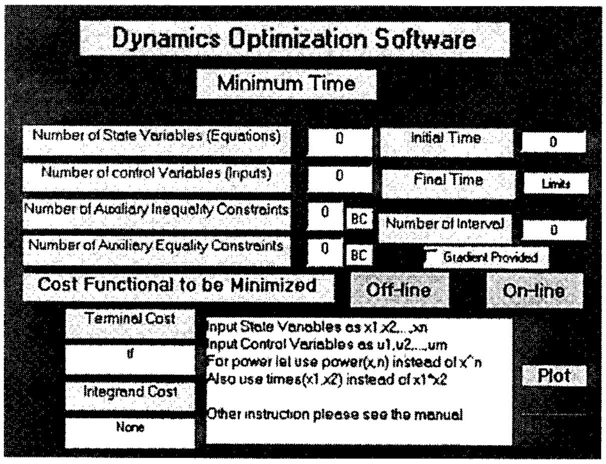
Fig. 2. The Second page (Fixed End Time)


Dynamics Optimization Software
Variable End Time
Number of State Variables (Equations)
Number of control Variables (Inputs)
Number of Auxiliary Inequality Constraints
Number of Auxiliary Equality Constraints
BC
BC
Initial Time
Final Time
Number of Interval
Gradient Provided
Cost Functional to be Minimized
Off-line
On-line
Terminal Cost
Integrand Cost
Plot

0
0
0
0
0
Limits
0
0
0

Input State Variables as x1,x2,..xn
Input Control Variables as u1,u2,..um
For power let use power(x,n) instead of x^n
Also use times(x1,x2) instead of x1*x2
Other instruction please see the manual

Fig. 3. The Second Page (Variable End Time)


Dynamics Optimization Software
Minimum Time
Number of State Variables (Equations)
Number of control Variables (Inputs)
Number of Auxiliary Inequality Constraints
Number of Auxiliary Equality Constraints
BC
BC
Initial Time
Final Time
Number of Interval
Gradient Provided
Cost Functional to be Minimized
Off-line
On-line
Terminal Cost
Integrand Cost
Plot

0
0
0
0
0
Limits
0
tf
None

Input State Variables as x1,x2,...xn
Input Control Variables as u1,u2,...um
For power let use power(x,n) instead of x^n
Also use times(x1,x2) instead of x1*x2
Other instruction please see the manual

Fig. 4. The Second Page (Minimum Time)

In each category, user must provide the considering problem. Also, this software is divided into two steps. First, the software parameterizes the input problem by the user into the form of nonlinear programming symbolically and stores all the algebraic equations as data files. Similarly for the cost functional (1) which represented in the integral form is parameterized by using Simpson rule. The second step is called on - line computation that solves the nonlinear programming problem described in Section 3. In the problem of nonlinear programming, it is very well known that the gradients of both the objective function and the constrained algebraic equation have the effective on how fast the solution could be obtained. Therefore, we propose this gradient as an option in each category as a simple click then the code provides all the gradients automatically. Finally, if the optimal control solutions are obtained, user can observe all the optimal trajectories by pushing the *plot* button. Note that the results plotted in this software are all the state and control variables respect to time.

5. Examples

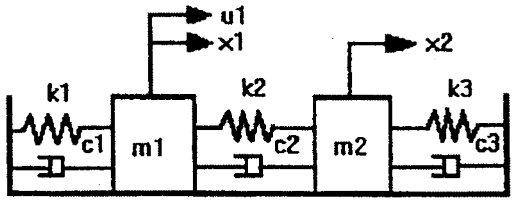


Fig. 5. Two degree - of - freedom of Spring - mass - damper System

5.1 Example 1: Spring - mass - damper System

The procedure outlined in this paper for dynamic optimization is illustrated with the following example of a two degree - of - freedom spring - mass - damper system sketched in equation as

$$A\dot{x} = Bu \qquad (20)$$

The matrices A and B for this system are as follows:

$$A = \begin{bmatrix} -M^{-1}C & -M^{-1}K \\ I_2 & 0 \end{bmatrix} \qquad (21)$$

$$B = \begin{bmatrix} \frac{1}{m_1} & 0 \\ 0 & \frac{1}{m_2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad (22)$$

where the matrices M, C, and K are:

$$M = \begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}, C = \begin{bmatrix} c_1 + c_2 & -c_2 \\ -c_2 & c_2 + c_3 \end{bmatrix} \qquad (23)$$

$$K = \begin{bmatrix} k_1 + k_2 & -k_2 \\ -k_2 & k_2 + k_3 \end{bmatrix} \qquad (24)$$

The parameters used in the model in MKS units are: $m_1 = m_2 = 1.0$, $c_1 = c_3 = 1.0$, $c_2 = 2.0$, $k_1 = k_2 = k_3 = 3.0$. The cost functional in equation (1) is $L = u_1 + u_2$. The boundary conditions specified for the problem are $X(t_0) = (5\ 10\ 0\ 0)^T$ *and* $X(t_f) = (0\ 0\ 0\ 0)^T$, *where* $t_0 = 0$ *and* $t_f = 2.0$. The state and control trajectories obtained from the optimization procedure described in this paper and the technique proposed in (Brenan, 1993) are overlap within the accuracy of the drawings shown in Figure 6 and 7.

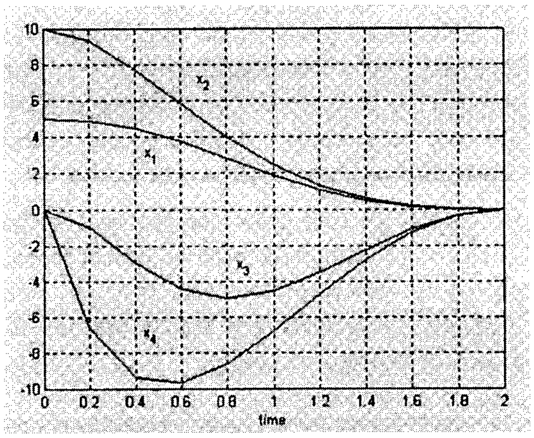


Fig. 6. The Optimal State Solution

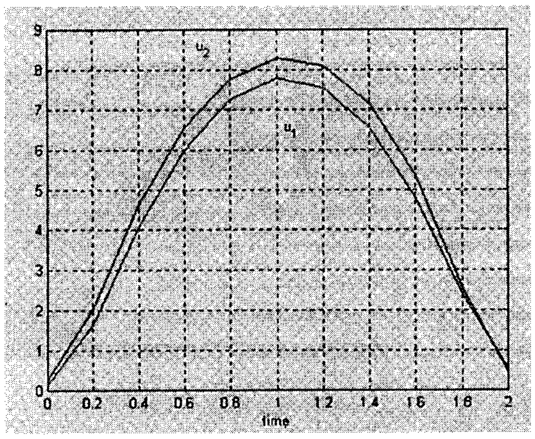


Fig. 7. The Optimal Control solution

5.2 Flexible Link Robot

In the problem of nonlinear systems, the example is of a single link manipulator rotating in a vertical plane driven through a flexible drive train (Spong & Vidyasaga, 1986) shown in Figure 8. The system has two degree - of - freedom and the equations of motion are

$$\begin{aligned} I\ddot{q}_1 + Mgl\sin q_1 + k(q_1 - q_2) &= 0 \\ J\ddot{q}_2 - k(q_1 - q_2) &= u_1 \end{aligned} \qquad (25)$$

where I and J is respectively the link and actuator moment of inertia, M is the mass of the link with mass center at a distance l from the joint, k is the stiffness of the drive train, g is the gravity constant, and u is the actuator torque. Let the objective be to steer the system from a given set of initial conditions on q_1, q_2, $\dot{q}_1$, and $\dot{q}_2$ at t_0 to a specified goal point at t_f while minimizing a cost .

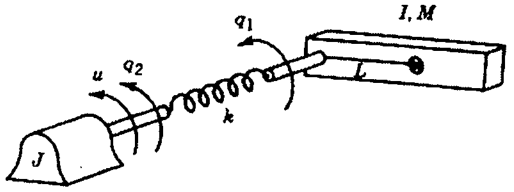


Fig. 8. Flexible Link Robot

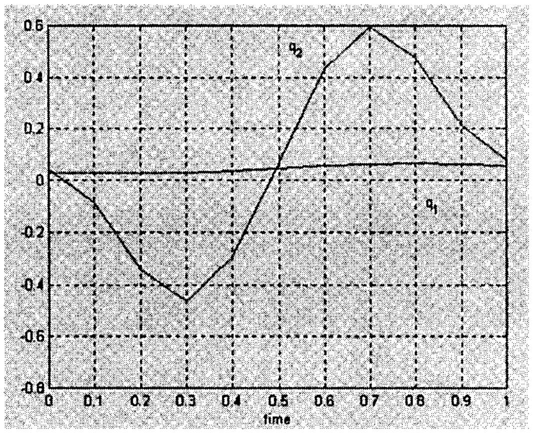


Fig. 9. The Optimal State Solution

$J = \int_{t_0}^{t_f} u^2 dt$. The trajectory must satisfy the constraint $-50 \leq u \leq 50$ during motion. The parameters used in the model (in MKS units) are: $I = J = 1.0$, $k = 1.0$, $g = 9.8$, $M = 1.0$, *and* $l = 0.5$.

The boundary conditions at both ends are: $q(t_0)$ = (0.03 0.04 - 0.0215 0.008)^T and $q(t_0)$ = (0.06 0.08 - 0.429 - 0.0639)^T. The state and control trajectories obtained from the optimization procedure described in this paper and the technique proposed in (Brenan, 1993) are overlap within the accuracy of the drawings shown in Figure 9 and 10.

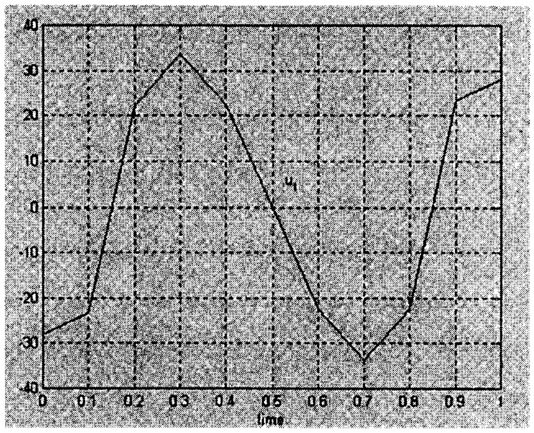


Fig. 10. The Optimal Control Solution

6. Conclusion

It is known from the literatures that there are many dynamic optimization theories for direct approach have been developed for the dynamic systems. This paper was aim to applied a simple technique for a general - purpose program; therefore, the Runge - Kutta collocation method has been used. Similarly in one literature developed by BRENAN, K.E., the approach is quite similar namely Hermit - Simpson collocation method. However, the solutions from both methods are compared within the accuracy of the drawing in this paper. As a result, the general - purpose program has been developed by using Runge - Kutta collocation method. Finally, the nonlinear algorithm is used since the dynamic optimization problems are converted to a parameter optimization problems when the Runge - Kutta collocation technique is applied.

Reference

Brenan, K.E.; "Differential-Algebraic Equations Issues in the Direct Transcription of Path Constrained Optimal"

Bryson, A.E. and Ho, Y.C., **Applied Optimal Control,** Hemisphere Publishing Company, 1975.

Bolza, O.; **Lectures on the Calculus of Variations,** G.E. Stechert and Company, 1931.

Control Problems, **Aerospace Report,** No. ATR-94 (8489)-1, December 1993.

Fliess, M.; Levine, J.; Martin, P.; Rouchon, P.; "Flatness and defect of Nonlinear Systems: Introductory Theory and Examples," **Journal of Control,** Vol. 61, No.2, 1995, 1327-1361.

Kirk, D.E.; **Optimal Control Theory: An Introduction, Prentice Hall Electrical Engineering Series,** 1970.

Spong, M.W.; Vidyasagar, M.; **Robot Dynamics and Control,** John Wiley and Sons, 1986.