行政院國家科學委員會專題研究計畫　成果報告

動態系統最佳設計數值方法及應用(2/2)

計畫類別：個別型計畫
計畫編號：NSC94-2212-E-009-011-
執行期間：94 年 08 月 01 日至 95 年 07 月 31 日
執行單位：國立交通大學機械工程學系(所)

計畫主持人：洪景華

計畫參與人員：黃智宏，曾錦煥

報告類型：完整報告
報告附件：出席國際會議研究心得報告及發表論文
處理方式：本計畫可公開查詢

中　華　民　國 95 年 8 月 4 日

行政院國家科學委員會補助專題研究計畫 ■ 成 果 報 告 □期中進度報告

動態系統最佳設計數值方法及應用

計畫類別：■ 個別型計畫　　□ 整合型計畫
計畫編號：NSC93－2212－E009－024
執行期間：　93 年　8 月　1 日至　95 年　7 月　31　日

計畫主持人：洪景華
共同主持人：
計畫參與人員：

成果報告類型(依經費核定清單規定繳交)：□精簡報告　■完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立交通大學機械工程研究所

中　華　民　國　九十五 年　八　月　　日

摘 要

動態系統所引發的特性一直困擾著工程設計人員，而只在靜態系統模式下，採用最佳化設計方法所求得的設計，則往往在實際的應用上有所不足。本研究計畫主要依據最佳設計與最佳控制理論基礎，結合動態分析與數值分析求解技巧，發展一套通用之動態系統最佳設計方法與軟體。

一般動態系統之最佳化問題可以轉換成標準的最佳控制問題,再透過離散技術轉換成非線性規劃問題，如此便可利用現有之最佳化軟體進行求解。在本研究計畫中，首先將動態系統的解題方法與流程發展為最佳控制分析模組，再將該模組與最佳化分析軟體(MOST) 整合得到整合最佳控制軟體，可以用來解決各種類型的最佳控制問題。為驗證軟體的效能與準確性，利用本研究計畫所發展之整合最佳控制軟體求解文獻資料中所提出之各類型最佳控制問題。藉由分析結果之數值與控制軌跡曲線的比對，整合最佳控制軟體所求出之數值解，在效能與準確性上都能與文獻資料所獲得的最佳解吻合，確認該整合最佳控制軟體的確可以用來解決我們工程應用上的最佳控制問題。

另外，針對工程設計中存在的離散（整數）最佳控制問題，本研究計畫依據混合整數非線性規劃法(mixed integer nonlinear programming) 做進一步的研究。猛撞型控制(bang-bang type control) 是常見的離散最佳控制問題，其複雜與難解的特性更是吸引諸多文獻探討的主因。許多文獻針對此一問題所提出的方法多在控制函數的切換點數量為已知的假設條件下所推導，但這並不符合實際工程上的應用需求，因為控制函數的切換點數量大多在求解完成後才會得知。因此，本研究計畫針對此類型問題發展出兩階段求解的方法，第一階段先粗略求解該問題在連續空間下的解，並藉此求得控制函數可能的切換點資訊，第二階段再利用混合整數非線性規劃法求解該問題的真實解。發展過程中，加強型的分支界定演算法 (enhanced branch-and-bound method)被實際應用並且納入

前一階段所開發的整合最佳控制軟體中，這也使得這個軟體可以同時處理實際動態系統中最常見的連續及離散最佳控制問題。

最後，本研究計畫將所發展的整合最佳控制軟體用來求解兩個實際的工程應用問題：飛航高度控制問題與車輛避震系統設計問題。兩個問題都屬於高階非線性控制問題，首先利用本研究計畫中所建議的解題步驟建構完成這兩個問題的數學模型，接著直接利用本研究所發展的軟體求解符合問題要求的最佳解。經由這些實際應用案例的驗證，顯示本研究計畫所發展的方法與軟體的確可以提供工程師、學者與學生一個便利可靠的動態系統設計工具。

ABSTRACT

The nonlinear behaviors of dynamic system have been of continual concern to both engineers and system designers. In most applications, the designs – based on a static model and obtained by traditional optimization methods – can never work perfectly in dynamic cases. Therefore, researchers have devoted themselves to find an optimal design that is able to meet dynamic requirements. This project focuses on developing a general-purpose optimization method, based on optimization and optimal control theory, one that integrates dynamic system analysis with numerical technology to deal with dynamic system design problems.

A dynamic system optimal design problem can be transformed into an optimal control problem (OCP). Many scholars have proposed methods to solve optimal control problems and have outlined discretization techniques to convert the optimal control problem into a nonlinear programming problem that can then be solved using extant optimization solvers. This project applies this method to develop a direct optimal control analysis module that is then integrated into the optimization solver, MOST. The numerical results of the study indicate that the solver produces quite accurate results and performs even better than those reported in the earlier literatures. Therefore, the capability and accuracy of the optimal control problem solver is indisputable, as is its suitability for engineering applications.

A second theme of this project is the development of a novel method for solving discrete-valued optimal control problems arisen in many practical designs; for example, the bang-bang type control that is a common problem in time-optimal control problems. Mixed-integer nonlinear programming methods are applied to deal with those problems in this project. When the controls are assumed to be of the bang-bang type, the time-optimal control problem becomes one of determining the switching times. Whereas several methods for determining the time-optimal control problem (TOCP) switching times have been studied extensively in the literature, these methods require that the number of switching times be

known before their algorithms can be applied. Thus, they cannot meet practical demands because the number of switching times is usually unknown before the control problems are solved. To address this weakness, this project focuses on developing a computational method to solve discrete-valued optimal control problems that consists of two computational phases: first, switching times are calculated using existing continuous optimal control methods; and second, the information obtained in the first phase is used to compute the discrete-valued control strategy. The proposed algorithm combines the proposed OCP solver with an enhanced branch-and-bound method and hence can deal with both continuous and discrete optimal control problems.

Finally, two highly nonlinear engineering problems – the flight level control problem and the vehicle suspension design problem – are used to demonstrate the capability and accuracy of the proposed solver. The mathematical models for these two problems can be successfully established and solved by using the procedure suggested in this project. The results show that the proposed solver allows engineers to solve their control problems in a systematic and efficient manner.

目　　錄

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

Two typical methods are usually used to solve optimal control problems: the indirect and direct approaches. The indirect approach bases on the solution of the first order necessary conditions for optimality. Pontryagin Minimum Principle (Pontryagin *et al.* 1962) and the dynamic programming method (Bellman 1957) are two common methods for indirect approach. The direct method (Jaddu and Shimemura 1999, Hu *et al.* 2002, Huang and Tseng 2004) based on nonlinear programming (NLP) approaches that transcribe optimal control problems into NLP problems and apply existed NLP techniques to solve them. In most of practical applications, the control problems are described by strongly nonlinear differential equations that the solutions is hard to be solved by indirect methods. For those cases, the direct methods can provide another choice to find the solutions.

In spite of extensive use of direct and indirect methods to solve optimal control problems, engineers still spend much effort on reformulating problems and implementing corresponding programs for different control problems. For engineers, this routine job will be tedious and time-consuming. Therefore, a systematic computational procedure for various optimal control problems has become an imperative for engineers, particularly for those who are inexperienced in optimal control theory or numerical techniques. Hence, the purpose of this project is to apply NLP techniques to implement an OCP solver that facilitates engineers in solving optimal control problems with a systematic and efficient procedure. To illustrate the practicability and convenience of propose solver, a flight control problem with two different cases is chosen to illustrate the capability for solving optimal control problem of proposed solver. The results demonstrate the proposed solver can get the solution correctly and the procedure suggested in this project can facilitate engineers to deal with their problems.

In many practical engineering applications, the control action is restricted to a set of discrete values that forms a discrete-valued control problem. These systems can be classified as switched systems consisting of several subsystems and switching laws that orchestrate the active subsystem at each time instant. Optimal control problems (OCPs) for switched systems, which require solution of both the optimal switching sequences and the optimal continuous inputs, have recently drawn the attention of many researchers. The primary difficulty with these switched systems is that the range set of the control is discrete and hence not convex. Moreover, choosing the appropriate elements from the control set in an appropriate order is a nonlinear combinatorial optimization problem. In the context of time optimal control problems, as pointed out by Lee *et al.* (1997), serious numerical difficulties may arise in the process of identifying the exact switching points. Therefore, an efficient numerical method is still needed to determine the exact control switching times in many practical engineering problems.

This study focuses on developing a numerical method to solve discrete-valued optimal control problems and the time-optimal control problem that is one of their special cases. The proposed algorithm, which integrates the admissible optimal control problem formulation (AOCP) with an enhanced branch-and-bound method (Tseng *et al*., 1995), is implemented and applied to some example systems.

2. Literature Review and Objectives

Methods for Optimal Control Problems

Optimal control problems can be solved by a variational method (Pontryagin *et al.*, 1962) or by nonlinear programming approaches (Huang and Tseng, 2003, 2004; Hu *et al.*, 2002; Jaddu and Shimemura, 1999). The variational or indirect method is based on the solution of first-order necessary conditions for optimality obtained from Pontryagin's maximum principle (Pontryagin *et al.*, 1962). For problems without inequality constraints, the optimality conditions can be formulated as a set of differential-algebraic equations, often in the form of a two-point boundary value problem (TPBVP). The TPBVP can be addressed using many approaches, including single shooting, multiple shooting, invariant embedding, or a discretization method such as collocation on finite elements. On the other hand, if the problem requires that active inequality constraints be handled, finding the correct switching structure, as well as suitable initial guesses for the state and costate variables, is often very difficult.

Much attention has been paid in the literature to the development of numerical methods for solving optimal control problems (Hu *et al.*, 2002; Pytlak, 1999; Jaddu and Shimemura, 1999; Teo, and Wu, 1984; Polak, 1971), the most popular approach in this field is the reduction of the original problem to a NLP problem. Nevertheless, in spite of extensive use of nonlinear programming methods to solve optimal control problems, engineers still spend much effort reformulating nonlinear programming problems for different control problems. Moreover, implementing the corresponding programs for the nonlinear programming problem is tedious and time consuming. Therefore, a general OCP solver coupled with a systematic computational procedure for various optimal control problems has become an imperative for engineers, particularly for those who are inexperienced in optimal control theory or numerical techniques.

Additionally, in many practical engineering applications, the control action is restricted to a set of discrete values. These systems can be classified as switched systems consisting of several subsystems and switching laws that orchestrate the active subsystem at each time instant. Optimal control problems for switched systems, which require solution of both the optimal switching sequences and the optimal continuous inputs, have recently drawn the attention of many researchers. The primary difficulty with these switched systems is that the range set of the control is discrete and hence not convex. Moreover, choosing the appropriate elements from the control set in an appropriate order is a nonlinear combinatorial optimization problem. In the context of time optimal control problems, as pointed out by Lee *et al*. (1997), serious numerical difficulties may arise in the process of identifying the exact switching points. Therefore, an efficient numerical method is still needed to determine the exact control switching times in many practical engineering problems.

Time-Optimal Control Problems

The TOCP is one of most common types of OCP, one in which only time is minimized and the control is bounded. In a TOCP, a TPBVP is usually derived by applying Pontryagin's maximum principle (PMP). In general, time-optimal control solutions are difficult to obtain (Pinch, 1993) because, unless the system is of low order and is time invariant and linear, there is little hope of solving the TPBVP analytically (Kirk, 1970). Therefore, in recent research, many numerical techniques have been developed and adopted to solve time-optimal control problems.

One of the most common types of control function in time-optimal control problems is the piecewise-constant function by which a sequence of constant inputs is used to control a given system with suitable switching times. Additionally, when the control is bounded, a very commonly encountered type of piecewise-constant control is the bang-bang type, which switches between the upper and lower bounds of the control input. When the controls are assumed to be of the bang-bang type, the time-optimal control problem becomes one of determining the switching times, several methods for which have been studied extensively in the literature (see, *e.g*., Kaya and Noakes, 1996; Bertrand and Epenoy, 2002; Simakov *et al*., 2002). However, as already mentioned, in contrast to practical reality, these methods require that the number of switching times be known before their algorithms can be applied. To overcome the numerical difficulties arising during the process of finding the exact switching points, Lee *et al*. (1997) proposed the control parameterization enhancing transform (CPET), which they also extended to handle the optimal discrete-valued control problems (Lee *et al*., 1999) and applied to solve the sensor-scheduling problem (Lee *et al*., 2001).

In similar manner, this project focuses on developing a numerical method to solve time-optimal control problems. This method consists of the two-phase scheme: first, switching times are calculated using existing optimal control methods; and second, the resulting information is used to compute the discrete-valued control strategy. The proposed algorithm, which integrates the admissible optimal control problem formulation with an enhanced branch-and-bound method (Tseng *et al.*, 1995), is then implemented and applied to some examples.

Objectives

The major purpose of this project is to develop a computational method to solve the time-optimal control problems and find the corresponding discrete-valued optimal control laws. The other purpose of this project is to implement a general OCP solver and provide a systematic procedure for solving OCPs that provides engineers with a systematic and efficient procedure to solve their optimal control problems.

3. METHODS

3.1 Developing Process of an Multi-Function OCP Solver

The developing processes of a general purpose solver for dynamical optimization can be described as follows.

3.1.1 Problem formulation

A dynamical optimization problem can be described by a generalized Bolza problem formulation: Find the design variables b, the control functions u(t) and terminal time tf which minimize the object function

$$J_0 = \psi_0(\mathbf{b}, \mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} F_0(\mathbf{b}, \mathbf{u}(t), \mathbf{x}(t), t)dt \qquad (1)$$

subject to the system equations

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{b}, \mathbf{u}(t), \mathbf{x}(t), t), \quad t_0 \le t \le t_f \tag{2}$$

with initial conditions

$$\mathbf{x}(t_0) = \mathbf{x}_0(\mathbf{b}), \tag{3}$$

functional constraints

$$J_i = \psi_i(\mathbf{b}, \mathbf{x}(t_f), t_f) + \int_{t_0}^{t_f} F_i(\mathbf{b}, \mathbf{u}(t), \mathbf{x}(t), t)\,dt \begin{cases} = 0; i = 1, \ldots\ldots\ldots, r' \\ \le 0; i = r'+1, \ldots., r \end{cases} \tag{4}$$

and dynamic point-wise constraints

$$\phi_j(\mathbf{b}, \mathbf{u}(t), \mathbf{x}(t), t) \le 0; \; j = 1, \ldots., q \tag{5}$$

where $\mathbf{b} \in \mathbf{R}^k$ is a vector of the design variables, $\mathbf{u}(t) \in \mathbf{R}^m$ is a vector of the control functions, and $\mathbf{x}(t) \in \mathbf{R}^n$ is a vector of the state variables. The functions $\boldsymbol{f}$, Ψ_0, F_0, Ψ_i, F_i *and* ϕ_j are assumed to be at least twice differentiable.

3.1.2 NLP Methods for dynamical optimization

By applying modeling and optimization technologies, a dynamic system optimization problem can be re-formulated as an optimal control problem (OCP). Hence, many approaches used to deal with the OCPs can be also applied to solve the dynamical optimization problems. Most popular approach in this field turned to be reduction of the original problem to a NLP.

Sequential Quadratic Programming (SQP), one of the best NLP methods for solving large-scale nonlinear optimization, is applied to solve optimal control problems (see, e.g., Gill *et al*. 2002, Betts 2000). Before applying the SQP methods, optimal control problems in which the dynamics are determined by a system of ordinary differential equations (ODEs) are usually transcribed into nonlinear programming (NLP) problems by discretization strategies. Due to the consideration of efficiency, the sequential discretization strategy which only the control variables are discretized is applied. The resulting formulation is then called the admissible optimal control problem (AOCP) formulation (Huang and Tseng 2003).

3.1.3 Computational Algorithm

The computational algorithm of the OCP solver which integrates AOCP with SQP is illustrated in Figure 1 and can be described as the following steps:

Given: Initial values of the design variables vector $\mathbf{P}^{(0)} = [\mathbf{b}^{(0)}, \mathbf{U}^{(0)}, \mathbf{T}^{(0)}]$ and Number of time intervals, **N**.

Initialize iteration counter k :=0 and Hessian Matrix $\mathbf{H}^{(0)}$:= Identity **I**.

1. Current design variable vector, $\mathbf{P}^{(k)}$, is passed to **CTRLMF** module of AOCP.
2. Evaluate the values of state variable, $\boldsymbol{x}(k)$, by solving the IVP by substituting $\mathbf{P}^{(k)}$ into the system equation.

$$\begin{aligned} &\dot{\mathbf{x}}^{(k)} = \mathbf{f}(\mathbf{b}^{(k)}, \mathbf{u}^{(k)}, \mathbf{x}^{(k)}, t), \\ &\mathbf{x}(t_0) = \mathbf{x}_0(\mathbf{b}^{(k)}) \end{aligned} \tag{6}$$

3. Compute the values of performance indexes, $\boldsymbol{J}_0^{(k)}$.

$$\begin{aligned} &J_0^{(k)} = \psi_0(\mathbf{b}^{(k)}, \mathbf{x}(\mathbf{b}^{(k)}, \mathbf{U}^{(k)}, \mathbf{T}^{(k)}, t_f), t_f) \\ &+\int_{t_0}^{t_f} F_0(\mathbf{b}^{(k)}, \mathbf{U}^{(k)}, \mathbf{x}^{(k)}, t)\,dt \end{aligned} \tag{7}$$

4. Substitute $\mathbf{x}(k)$ into Eqs. (4) and (5) to evaluate the values of functional and dynamic constraints.
5. Evaluate $\nabla \boldsymbol{J}_0^{(k)}$, $\nabla \boldsymbol{J}_i^{(k)}$, and $\nabla \boldsymbol{\phi}_j^{(k)}$ by using the finite difference method.
6. Find the descent direction, $\mathbf{d}^{(k)}$, by solving the QP subproblem.
7. Check convergence criteria, $\mathbf{d}^{(k)} \leq \varepsilon$. If satisfied, stop and show the results.
8. Compute the step size, $\alpha^{(k)}$.
9. Update Hessian Matrix $\mathbf{H}^{(k)}$ by applying BFGS method.
10. Update design variables

$$\mathbf{P}^{(k+1)} = \mathbf{P}^{(k)} + \alpha \cdot \mathbf{d}^{(k)} \tag{8}$$

11. Increase iteration counter, $k \leftarrow k+1$, go back to step 1.

3.1.4 Systematic Procedure for Solving OCP

The following steps describe a systematic procedure for solving the OCP with the proposed OCP solver:

1. Program formulation: The original optimal control problem must be formulated according to the extended Bolza formulation.
2. Preparing two parameter files: One of the parameter files describes the numerical schemes used to solve the OCP and also the relationships between performance index, constraint functions, dynamic functions, state variables and control variables. The other parameter file includes the information on SQP parameters, such as convergence parameter, upper/lower bound and initial guess of design variables, *etc*.
3. Implementing user-defined subroutines.
4. Execute the optimization: The user defined subroutines are compiled and then linked with the SQP solver, MOST (Tseng *et al*., 1996).
5. Execute the optimization.

With the proposed OCP solver, engineers can focus their efforts on formulating their problems and then follow an efficient and systematic procedure to solve their optimal control problems.

3.2 Mixed-Integer NLP Algorithm for Solving Discrete-valued OCPs

The algorithm developed in this study consists of three major processes: branching, the AOCP, and bounding. Initially, all discrete-valued restrictions are relaxed and the resulting continuous NLP problem is solved using the AOCP. If the solution of continuous optimum design problem occurs when all discrete-valued variable values are in the discrete set $\boldsymbol{U}_{\mathrm{d}}$, which is preset by the user to meet practical requirements, then an optimal solution is determined and the procedure ends. Otherwise, the algorithm selects one of the discrete-valued variables whose value is not in the discrete set $\boldsymbol{U}_d$ – for example, the *j*-th design variable, P_j, with value $\hat{P}_j$ – and branches on it.

Branching process: In the branching process, the original design domain is divided into three subdomains by two allowable discrete values, $\bar{u}_i$ and $\bar{u}_{i+1}$, that are nearest to the continuous optimum, as shown in Figure 2. Among the three subdomains, subdomain II, included in the continuous solution but not in the feasible discontinuous solution, is dropped. In the other two subdomains, called nodes, two new NLP problems are formed by adding simple bounds, $\hat{P}_j \le \bar{u}_i$ and $\hat{P}_j \ge \bar{u}_{i+1}$, respectively, to the continuous NLP problems. One of the two new NLP problems is selected and solved next. Many search methods based on tree searching – including depth-first search, breadth-first search and best-first search – can be applied to choose the next branching node. The branching process is repeated in each of the subdomains until the feasible optimal solution is found in which all the discrete variables have allowable discrete values. Obviously, the number of subdomains may grow exponentially so that a great deal of computing time is required. Thus in the enhanced branch-and-bound method (Tseng *et al.*, 1995), multiple branching and unbalanced branching strategies have been developed to improve the efficiency of the method.

Bounding process: In discrete optimization, the minimum cost is always greater than or equal to the cost of the original regular optimal design that was originally branched. This fact provides a guideline for when branching should be stopped. If the branching process yields a feasible discontinuous solution, then the corresponding cost value can be considered a bound. Any other subdomain that imposes a continuous minimum cost larger than this bound need not be branched further. This bounding strategy can be used to select the branching route intelligently and avoid the need for a complete search over all the branches.

3.3 Algorithm for Solving Discrete-valued OCPs

In this study, the AOCP algorithm is used as the core iterative routine of the enhanced branch-and-bound method. All candidates will be evaluated and finally an optimal solution can be found. Here, symbol $\boldsymbol{S}$ is used to represent the discretized control variable set and the $\boldsymbol{P}$ is the design variable vector. Assuming that the problem at least has one feasible solution, it can then be proven that an optimal solution exists and can be found by the proposed method. The details of the proposed algorithm are as follows and Figure 3 presents a schematic flow chart of the algorithm for solving discrete-valued optimal control problems.

Initialization:

Relax all discrete-valued restrictions and then place the resulting continuous NLP problem on the branching tree.

Set the cost bound $J_{max} = \infty$.

while (there are pending nodes in the branching tree) **do**

1. Select an unexplored node from the branching tree.
2. Control discretization.
3. **Repeat** (for k-th AOCP iteration)
   (1). Solve the initial value problem for state variable $\boldsymbol{x}^{(k)}$ of AOCP.
   (2). Calculate the values of the cost function, J_0, and the constraints.
   (3). Solve the $QP^{(k)}$ problem by applying the BFGS method to obtain the descent direction $\boldsymbol{d}^{(k)}$.
   (4). **if** ($QP^{(k)}$ is feasible and convergent) **then exit** AOCP.
   (5). Find the step size $\alpha^{(k)}$ of the SQP method by using the line search method.
   (6). Update the design variable vector: $\mathbf{P}^{(k+1)} = \mathbf{P}^{(k)} + \alpha^{(k)}\, \boldsymbol{d}^{(k)}$.
4. **if** (NLP is optimal) and ($J_0 < J_{max}$) **then**

if ($\mathbf{S}^{(k+1)}$ is feasible) **then**

Update the current best point by setting the cost bound $J_{max} = J_0$.

Add this node to the feasible node matrix.

else

Evaluate the values of criteria for selecting the branch node.

Choose a discrete-valued variable $\mathbf{S}_l^{(k+1)} \notin \mathrm{M}$ and branch it.

Add two new NLP problems into the branching tree.

Drop this node.

endif

else

Stop branching on this node.

endif

end while.

3.4 Two-Phase Scheme for Solving TOCP

The mixed integer NLP algorithm developed in this dissertation is one type of switching time computation (STC) method. Most switching time computation methods (see, *e.g.*, Kaya and Noakes, 1996; Lucas and Kaya, 2001; Simakov *et al.*, 2002) assume that the structure of the control is bang-bang and the number of switching times is known. Unfortunately, the information on the switchings of several practical time-optimal control problems is unknown and hard to compute using analytical methods. Hence, to overcome this difficulty, this dissertation proposes a two-phase Scheme that consists of the AOCP plus the mixed-integer NLP method. In Phase I, the AOCP is used to calculate the information on switching times with rough time grids so that the information can be used in Phase II as the feasible initial design of the mixed integer NLP method. This scheme is described briefly below.

Phase I: Find the information about the switching times and terminal time.

1. Solve the time-optimal control problem using continuous controls by following the steps of the AOCP method.
2. Based on the numerical results, extract information about the switching times and terminal time, t_f.

Phase II: Calculate the exact solutions

3. Based on the information about switching times obtained in Phase I, treat the switchings as design variables and add them into the time grid vector $\boldsymbol{T}$. It should be noted that each interval between the upper and lower bounds on each of those design variables must include one switching.
4. Insert the terminal time, t_{f}, into the design variable vector $\boldsymbol{P}$.
5. Discretize each control variable into the number of switchings plus one. Then the discrete control vector, $\boldsymbol{S}$, can be added to the design variable vector $\boldsymbol{P}$ and the corresponding upper and lower bounds be limited by the original bounds of the controls.
6. Solve the problem by applying the mixed integer NLP method, and then find the optimal discrete-type control trajectories.

A third-order system shown in following section is used to demonstrate the processes of this numerical scheme.

4. Illustrative Examples

The numerical results for the following examples are obtained on an Intel Celeron 1.2 GHz computer with 512 MB of RAM memory. The AOCP is coded in FORTRAN, and C language is used to implement the enhanced branch-and-bound method. The Visual C++ 5.0 and Visual FORTRAN 5.0 installed in a Windows 2000 operating system are adopted to compile the corresponding programs. The total CPU times for solving the F-8 fighter craft problem in Phase I and Phase II are 3.605 and 1.782 seconds, respectively.

4.1 Third-Order System

The following system of differential equations is a model of the third-order system dynamics taken from Wu (1999).

$$\dot{x}_1 = x_2, \tag{9}$$

$$\dot{x}_2 = x_3, \tag{10}$$

$$\dot{x}_3 = -10x_3 + 10u. \tag{11}$$

The problem here is to find the control $|u| \le 10$ in order to bring the system from the initial state $[-10, 0, 0]^T$ to the final state $[0, 0, 0]^T$ in minimum time.

First, this problem is solved directly by the mixed integer NLP method. Assuming four switching times (T_1, T_2, T_3, T_4) and five control arcs have values in the discrete set, $\mathbf{U_d}$: {-10, 10}, the terminal time, t_f, is treated as a design variable, so the design variable vector $\boldsymbol{P}$ can be expressed as $[T_1, T_2, T_3, T_4, t_f, U_{d1}, U_{d2}, U_{d3}, U_{d4}, U_{d5}]^T$. Most notably, the final conditions of the state variables are transferred to the equality constraints. Thus, the TOCP problem becomes one of determining the switching times. Figure 4(a) presents the continuous solution obtained by using the AOCP and the discrete solution determined by applying the mixed integer NLP method proposed herein. The results indicate that the control trajectory determined by the mixed integer NLP method is of the bang-bang type and the solution consistent with the results obtained by Wu (1999).

As stated in previous section, several assumptions must be made when the mixed integer NLP method is applied to solving TOCP directly. Unfortunately, these assumptions cannot be guaranteed to hold in practical cases. Consequently, the two-phase scheme proposed in this project is needed. For illustration, the third-order system is again solved using this two-phase scheme. In Phase I, the two switching times are found to be $[0.330, 0.725]^T$ and the terminal time t_f is 0.7864. In the first phase, these switching data need not be accurate because they are only used to help users decide on the number of switching times, the control arcs and their corresponding boundaries. Thus, in Phase II, the design variable vector $\boldsymbol{P}$ is re-formed as $[T_1, T_2, t_f, U_{d1}, U_{d2}, U_{d3}]^T$; the numerical result obtained by applying the mixed integer NLP method is as presented in Figure 4(b). In Phase II, the switching times of the discrete control input are $[0.323, 0.713]^T$, and the terminal time t_f is 0.7813 seconds. The control trajectory also agrees with that obtained by Wu (1999).

4.2 F-8 Fighter Aircraft

The F-8 fighter aircraft has been considered in several pioneering studies (*e.g.*, Kaya and Noakes, 1996; Banks and Mhana, 1992; Simakov *et al.*, 2002) and has become a standard for testing various optimal control strategies. A nonlinear dynamic model of the F-8 fighter aircraft is considered below. The model is represented in state space by the following differential equations:

$$\begin{aligned}\dot{x}_1 = &-0.877x_1 + x_3 - 0.088x_1x_3 + 0.47x_1^2 - 0.019x_2^2 - x_1^2x_3 + 3.846x_1^3 \\ &-0.215u + 0.28x_1^2u - 0.47x_1u^2 + 0.63u^3\,,\end{aligned} \tag{15}$$

$$\dot{x}_2 = x_3\,, \tag{16}$$

$$\dot{x}_3 = -4.208x_1 - 0.396x_3 - 0.47x_1^2 - 3.564x_1^3 - 20.967u \tag{17}$$

$$+6.265x_1^2u+46x_1u^2+61.4u^3,$$

where x_1 is the angle of attack in radians, x_2 is the pitch angle, x_3 is the pitch rate and the control input u represents the tail deflection angle. For convenience of comparison, the standard settings (Kaya and Noakes, 1996; Lee *et al.*, 1997) are used. A control $|u| \le 0.05236$ must be found that brings the system from its initial state $\left[26.7\pi/180, \quad 0, \quad 0\right]^T$ to the final state $\left[0, \quad 0, \quad 0\right]^T$ in minimum time.

When the two-phase scheme is applied, as described in Section 5.4, the switching times computed in Phase I are 0.115, 2.067, 2.239, 4.995, and 5.282, and the terminal time is t_f = 5.7417. These switching data are used to set the design variables and their corresponding bounds, and then the problem is solved by the mixed integer NLP method. Finally, the switching times for the discrete control input are 0.098, 2.027, 2.199, 4.944, and 5.265, and the terminal time t_f is 5.74216. Figure 5 shows the comparison of the controls between Phase I and Phase II, while Figure 6 shows the trajectories of the states and the control of Phase I and Phase II. This example is also solved by Kaya and Noakes (1996) using the switching time computation method and by Lee *et al.* (1997) using the Control Parameterization Enhancing Transform (CPET) method. Table 1 shows the terminal time t_f, switching times and the accuracy of terminal constraints computed by various methods for this problem. According to the numerical results, the two-phase scheme provides a better solution, and the accuracy of the terminal constraints is acceptable.

5. Conclusions

In this project an optimal control problem solver, the OCP solver, based on the Sequential Quadratic Programming (SQP) method and integrated with many well-developed numerical routines is implemented. A systematic procedure for solving optimal control problems is also offered in this project.

This project also presents a novel method for solving discrete-valued optimal control problems. Most traditional methods focus on the continuous optimal control problems and fail when applied to a discrete-valued optimal control problem. One common type of such problems is the bang-bang type control problem arising from time-optimal control problems. When the controls are assumed to be of the bang-bang type, the time-optimal control problem becomes one of determining the TOCP switching times. Several methods for such determination have been studied extensively in the literature; however, these methods require that the number of switching times be known before their algorithms can be applied. As a result, they cannot meet practical situations in which the number of switching times is usually unknown before the control problem is solved. Therefore, to solve discrete-valued optimal control problems, this dissertation has focused on developing a computational method consisting of two phases: (a) the calculation of switching times using existing optimal control methods and (b) the use of the information obtained in the first phase to compute the discrete-valued control strategy.

The proposed algorithm combines the proposed OCP solver with an enhanced branch-and-bound method. To demonstrate the proposed computational scheme, the study applied third-order systems and an F-8 fighter aircraft control problem considered in several pioneering studies. Comparing the results of this study with the results from the literature indicates that the proposed method provides a better solution and the accuracy of the terminal constraints is acceptable.

REFERENCES

Banks, S.P., and Mhana, K.J., “Optimal Control and Stabilization of Nonlinear systems,” IMA Journal of Mathematical Control and Information, Vol. 9, pp. 179-196, 1992.

Bellman, R., 1957, Dynamic Programming, Princeton University Press, Princeton, NJ, USA.

Betts, J. T., 2000, “Very Low-Thrust Trajectory Optimization Using a Direct SQP Method,” Journal of Computational and Applied Mathematics, Vol. 120, pp. 27-40.

C.H. Huang, and C.H. Tseng, 2003, “Computational Algorithm for Solving A Class of Optimal Control Problems,” IASTED International Conference on Modelling, Identification, and Control (MIC’2003), Innsbruck, Austria, pp. 118-123.

C.H. Huang, and C.H. Tseng, 2004, “Numerical Approaches for Solving Dynamic System Design Problems: An Application to Flight Level Control Problem,” Proceedings of the Fourth IASTED International Conference on Modelling, Simulation, and Optimization (MSO’2004), Kauai, Hawaii, USA, pp. 49-54.

C.H. Tseng, W.C. Liao, and T.C. Yang, 1996, MOST 1.1 User's Manual, Technical Report No. AODL-93-01, Department of Mechanical Engineering, National Chiao Tung Univ., Taiwan, R.O.C.

Gill, P. E., Murray, W., and Saunders, M. A., 2002, “SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization,” SIAM Journal on Optimization, Vol. 12, No. 4, pp. 979-1006.

G.S. Hu, C.J. ONG, and C.L. Teo, 2002, “An Enhanced Transcribing Scheme for The Numerical Solution of A Class of Optimal Control Problems,” Engineering Optimization, Vol. 34, No. 2, pp. 155-173.

Jaddu, H., and Shimemura, E., 1999, “Computational Method Based on State Parameterization for Solving Constrained Nonlinear Optimal Control Problems,” International Journal of Systems Science, Vol. 30, No. 3, pp. 275-282.

Kaya, C.Y., and Noakes, J.L., “Computations and Time-Optimal Controls,” Optimal Control Applications and Methods, Vol. 17, pp. 171-185, 1996.

Lee, H.W., Jennings, L.S., Teo, K.L., and Rehbock, V., “Control Parameterization Enhancing Technique for Time Optimal Control Problems,” Dynamic Systems and Applications, Vol. 6, pp. 243-262, 1997.

Lucas, S.K., and Kaya, C.Y., “Switching-Time Computation for Bang-Bang Control Laws,” Proceedings of the 2001 American Control Conference, pp. 176-181, 2001.

Simakov, S.T., Kaya, C.Y., and Lucas, S.K., “Computations for Time-Optimal Bang-Bang Control Using A Lagrangian Formulation,” 15th Triennial World Congress, Barcelona, Spain, 2002.

Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V., and Mischenko, E. F., 1962, The Mathematical Theory of Optimal Processes, Wiley.

Tseng, C.H., Wang, L.W., and Ling, S.F., “Enhancing Branch-And-Bound Method for Structural Optimization,” Journal of Structural Engineering, Vol. 121, No. 5, pp. 831-837, 1995.

Wu, S.T., “Time-Optimal Control and High-Gain Linear State Feedback,” International Journal of Control, Vol. 72, No.9, pp. 764-772, 1999.

Table 1 Results of various methods for the F-8 fight aircraft problem.

Method	t_f	Switching Times	Accuracy of Terminal Constraints
STC (Kaya and Noakes, 1996)	6.3867	0.0761, 5.4672, 5.8241, 6.3867	$\leq 10^{-5}$
CPET (Lee *et al.*, 1997)	6.0350	2.188, 2.352, 5.233, 5.563	$\leq 10^{-10}$
Two-phase scheme	5.7422	0.098, 2.027, 2.199, 4.944, 5.265	$\leq 10^{-10}$

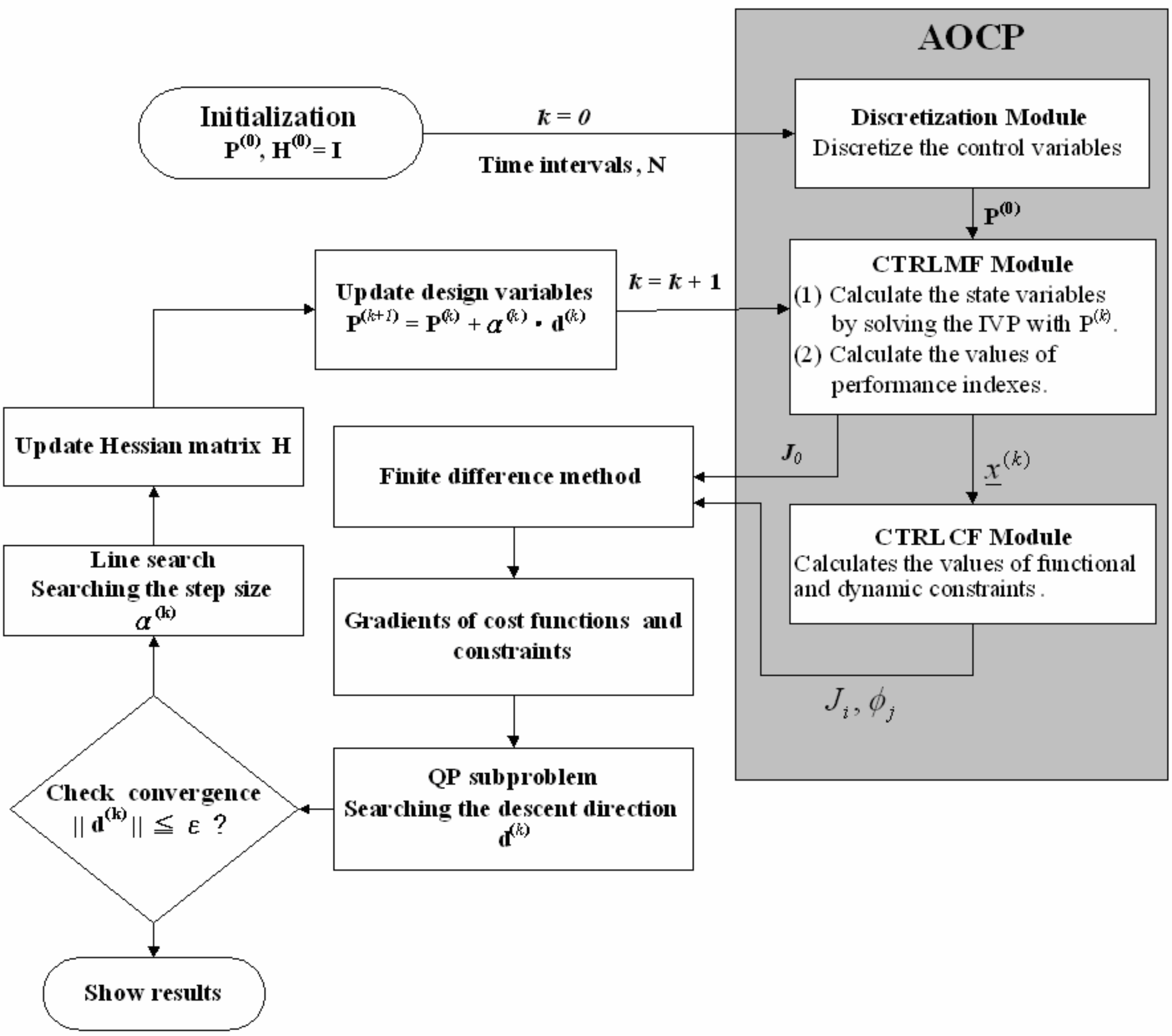


Figure 1 flow chart of the NLP method for solving OCP

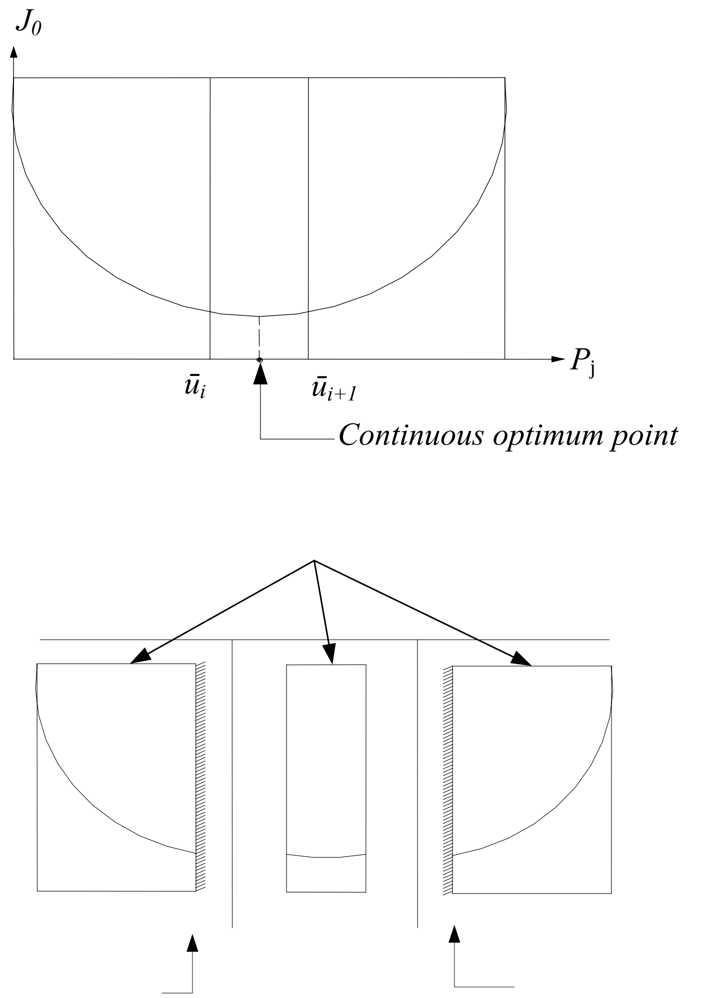


Figure 2 Conceptual layout of the branching process.

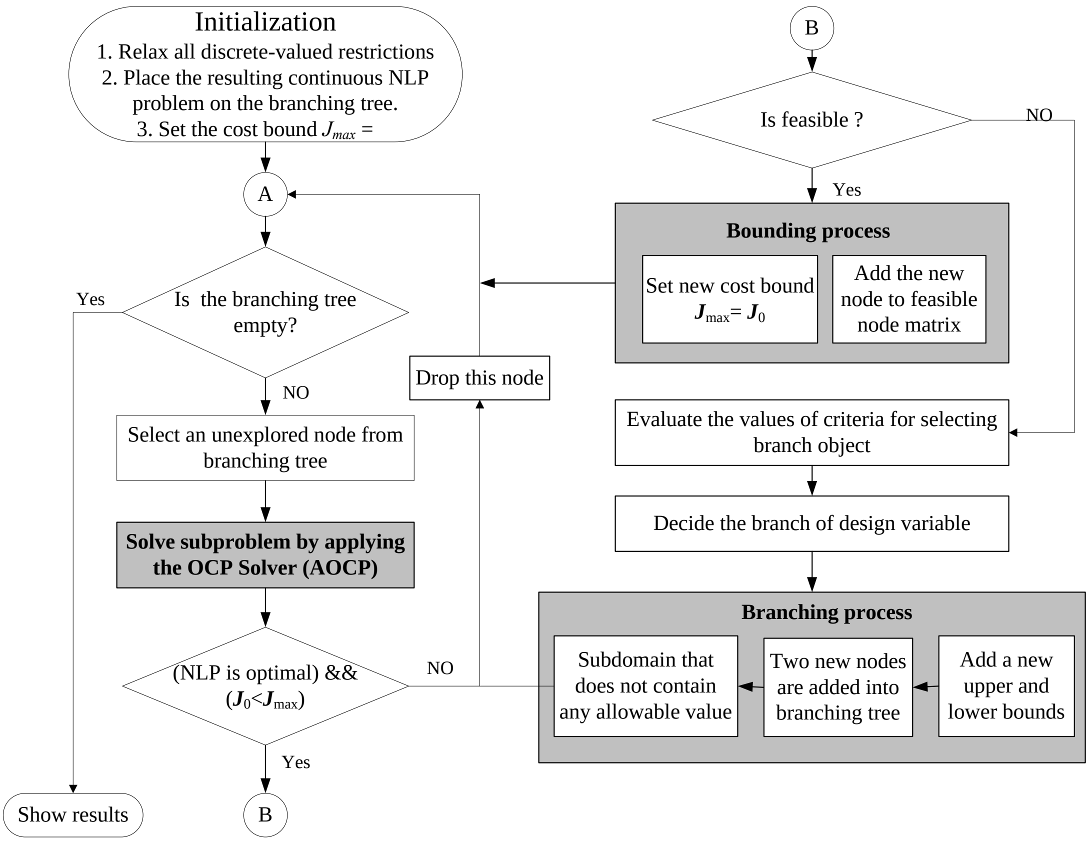
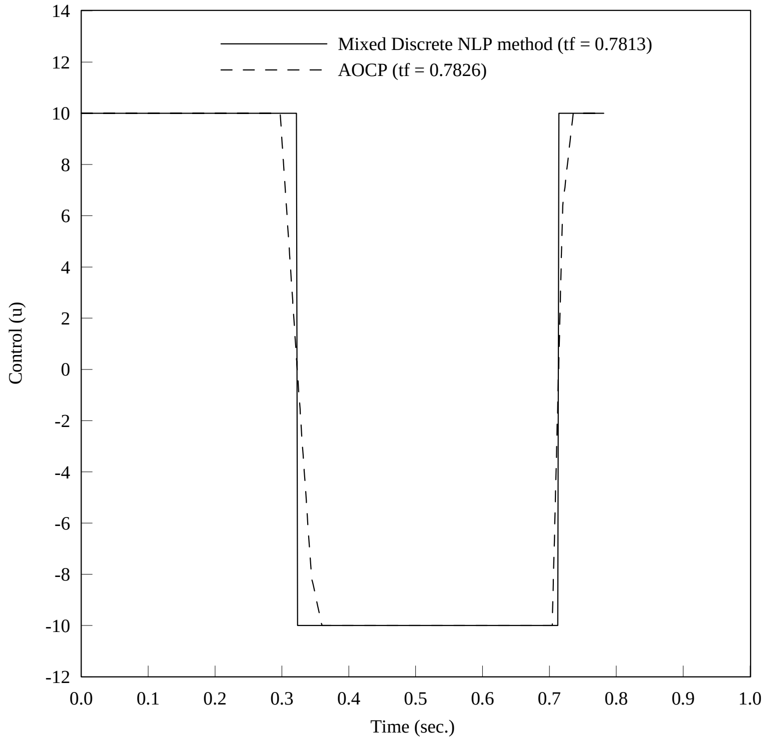
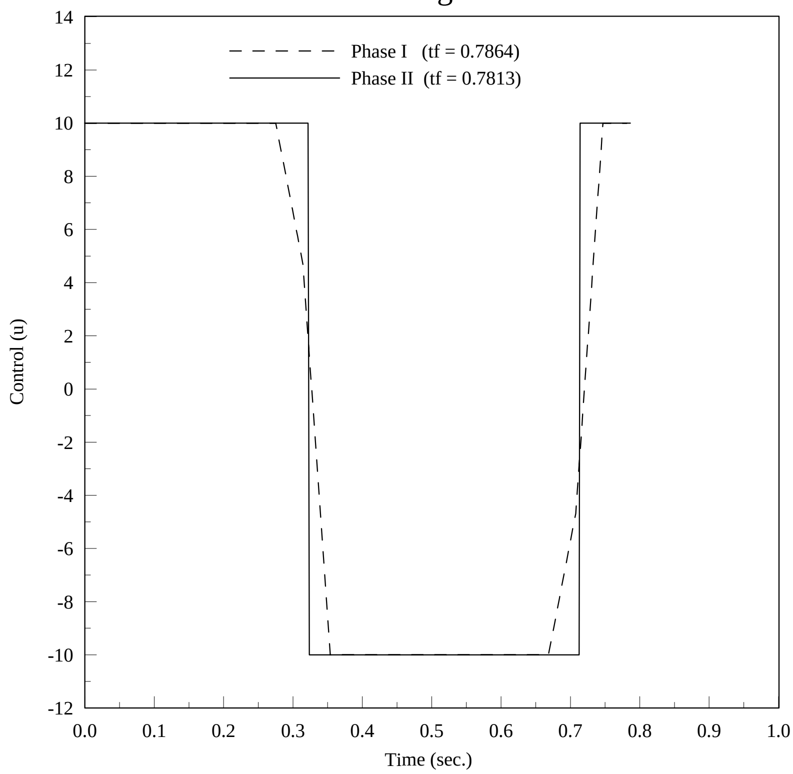


Figure 3 Flow chart of the algorithm for solving discrete-valued optimal control problems.



AOCP vs. a mixed-integer NLP method.



Phase I vs. Phase II.

Figure 4 Control trajectories for the third-order system.

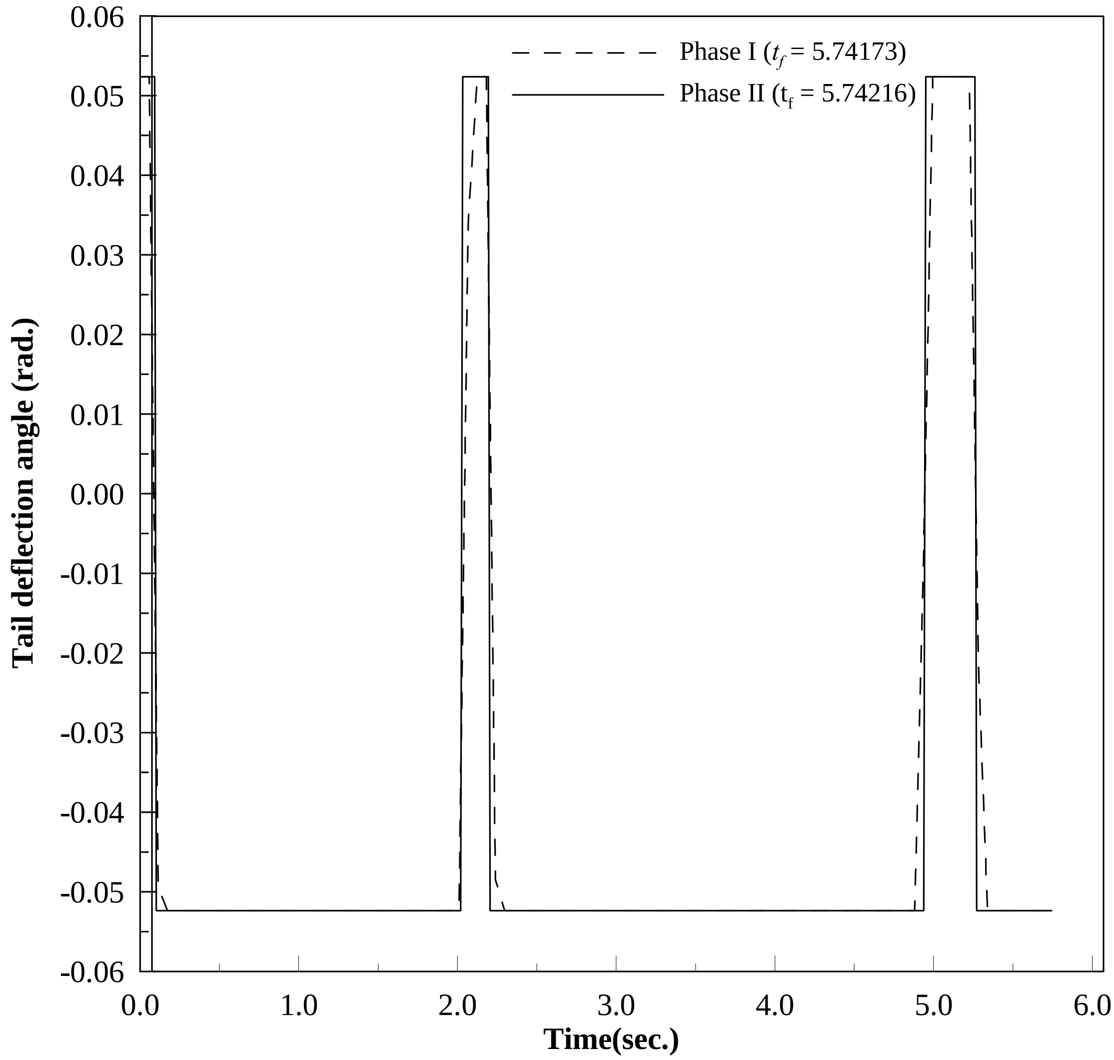
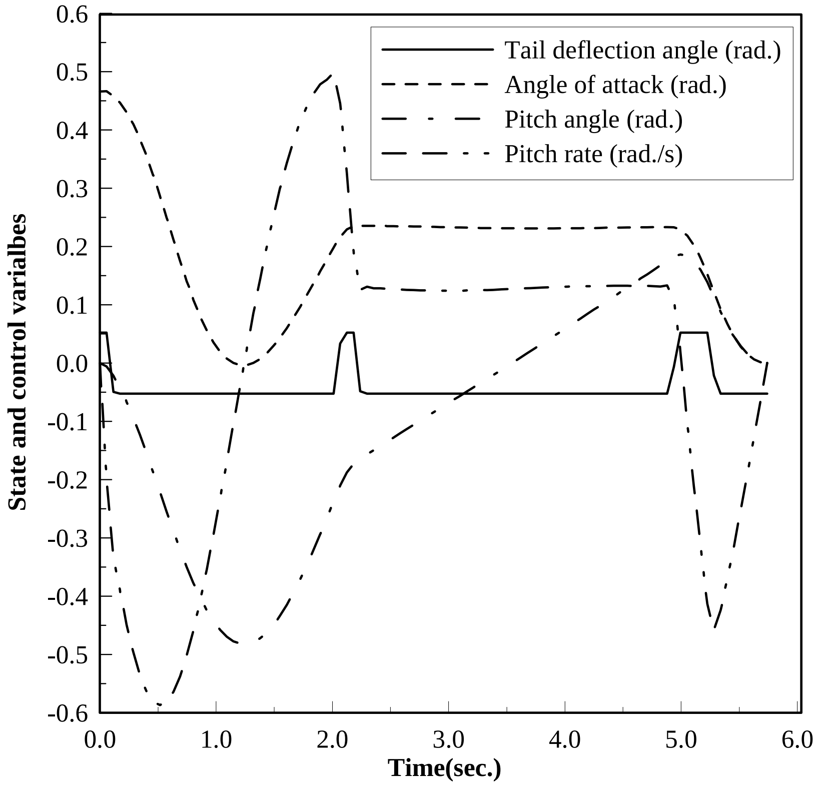
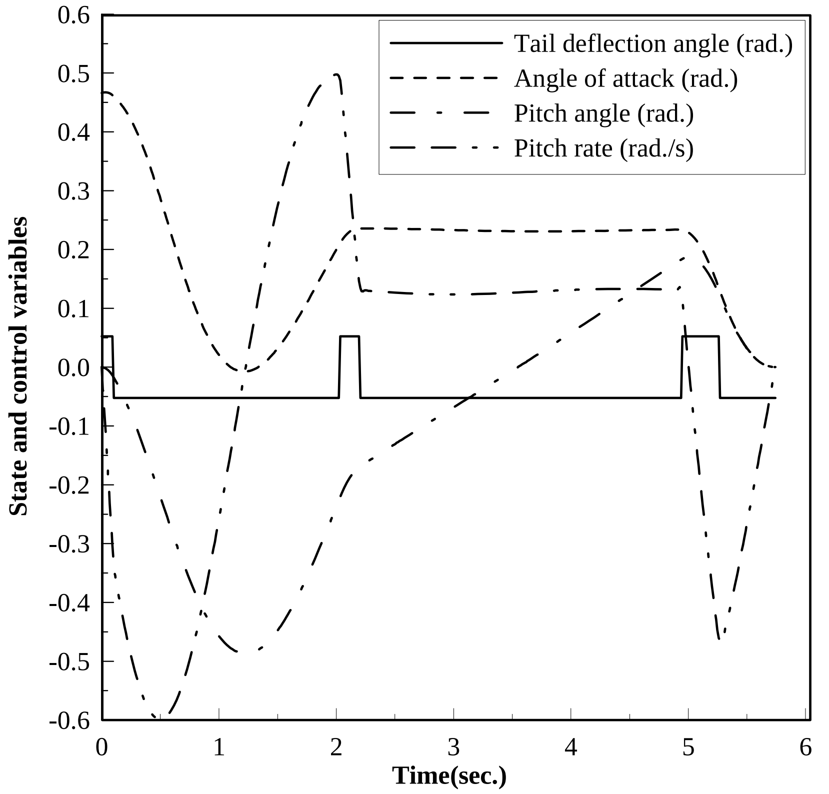


Figure 5 Control trajectories for the F-8 fighter aircraft.



(a) Phase I.



(b) Phase II.

Figure 6 Trajectories of the states and control input for the F-8 fighter aircraft.

計畫成果自評

傳統最佳化設計方法單靠靜態分析所得到產品在實際動態工作環境下往往表現不佳，甚至有時會面臨無法正常動作的窘境。因此，本研究計畫主要發展目標是依據動態系統分析與最佳化理論，整合數值分析技巧，發展一套有效率的方法與軟體來協助設計者處理動態系統最佳設計問題，其中必須將非線性與離散參數設計問題皆需納入考量，以因應實際工程需求。

本研究計畫分兩年實施，第一年研究重點在於發展出一套整合動態分析與最佳化技巧的方法，並將之實作成一套泛用型之動態最佳化軟體，此軟體的發展將有助於動態系統設計者縮短分析及設計的時程，並對產品的更新與市場佔有率的提升提供顯著的幫助。而要發展一套整合動態分析與最佳化技巧的方法，首先必須解決工程系統非線性的問題，系統動態特性分析時所解得的系統方程式常常是高度非線性的微分方程組，此時通常無法求得解析解，而藉助電腦數值分析技巧來求得收斂解是必要的。 第二個困難點是在於許多動態分析必須藉助專業的分析軟體來進行，此時如何整合最佳化與分析軟體便成為另一項挑戰。

本計畫利用數值分析方法與程式設計技巧，順利完成預期目標中所要發展的泛用型之動態最佳化軟體，並利用許多文獻上著名的動態設計與控制問題來進行驗證，從軟體求得的數值解與文獻的結果作比對，發現本計畫所發展之軟體所得的結果與文獻的結果是吻合的，甚至有些問題利用本計畫所發展出來的軟體求得的解優於文獻上的結果，由這些結果我們得到的初步的驗證，也順利完成本計畫中第一年所預期想要達到的目標。

本研究計畫第二年除了延續第一年的主題外，更將工程設計常會遭遇到某些設計尺

寸是有限離散數值（或整數）的情況納入考量。這個看似簡單的問題，其實是讓原本單純的連續變數最佳化設計問題，轉換成複雜難解的混合型整數最佳化設計問題，但這類型的問題卻是實際工程上所常常會遭遇的，如果能進一步提供解決這類型的動態最佳化問題，將可大幅提升研發設計的能力。本研究計畫，採納第一年所發展的解題軟體做為核心，並將加強型的分支界定演算法 (enhanced branch-and-bound method)納入前一階段所開發的整合最佳控制軟體中，這也使得這個軟體可以同時處理實際動態系統中最常見的連續及離散最佳控制問題。

本研究計畫的成果除了實作成功能強大的泛用型動態最佳化分析軟體外，其方法與應用也發表於國際期刊與會議中。由整個計畫執行過程中，於每個計畫執行階段，計畫執行所預期之目標均已完成，而整個研究過程中所執行之目標與成果敘述如下：

1. 發展一系統化解決動態最佳化問題的方法與流程，使用者只要依循所建議的方法將問題定義成標準形式，便可以利用本研究計畫所發展的軟體求得其最佳解。
2. 發展一新穎的方法來求解離散數值最佳控制問題(混合整數之離散數值動態設計問題)，此方法可以讓使用者在對於原來連續最佳控制問題上對於某些設計變數作些微的設定修改，就可以求解複雜的離散數值最佳控制問題。
3. 發展一新穎的方法來求解猛撞型的最佳控制問題(Bang-bang Control problems)使用者無須事先知道控制變數的切換時間點數量，即可求解出最佳的Bang-bang control law，對於非線性的問題以往要求得其Bang-bang control law是相當困難的，但利用本研究計畫所發展的方法，搭配數值計算技巧，可以順利求得符合拘束條件的控制法則。

本研究目前已發表兩篇國際期期刊及兩篇研討會論文

期刊論文

1. C.H. Huang and C.H. Tseng, "An Integrated Two-Phase Scheme for Solving Bang-Bang Control Problems," Accepted for publication in Optimization and Engineering (SCI Expended/ISI).
2. C.H. Huang and C.H. Tseng, "A Convenient Solver for Solving Optimal Control Problems," Journal of the Chinese Institute of Engineers, Vol. 28, pp. 727-733, 2005 (Ei/SCI).

研討會論文

1. C.H. Huang, and Tseng, C.H., "Numerical Approaches for Solving Dynamic System Design Problems: An Application to Flight Level Control Problem," Proceedings of the Fourth IASTED International Conference on Modelling, Simulation, and Optimization (MSO2004), Kauai, Hawaii, USA, 2004, pp. 49-54.
2. C.H. Huang, and Tseng, C.H., "Computational Algorithm for Solving A Class of Optimal Control Problems," IASTED International Conference on Modelling, Identification, and Control (MIC2003), Innsbruck, Austria, 2003, pp. 118-123.

附錄

1. (期刊論文) C.H. Huang and C.H. Tseng, “A Convenient Solver for Solving Optimal Control Problems,” Journal of the Chinese Institute of Engineers, Vol. 28, pp. 727-733, 2005 (Ei/SCI).
2. (會議論文) C.H. Huang, and Tseng, C.H., “Numerical Approaches for Solving Dynamic System Design Problems: An Application to Flight Level Control Problem,” Proceedings of the Fourth IASTED International Conference on Modelling, Simulation, and Optimization (MSO2004), Kauai, Hawaii, USA, 2004, pp. 49-54.
3. (會議論文) C.H. Huang, and Tseng, C.H., “Computational Algorithm for Solving A Class of Optimal Control Problems,” IASTED International Conference on Modelling, Identification, and Control (MIC2003), Innsbruck, Austria, 2003, pp. 118-123.

Short Paper

A CONVENIENT SOLVER FOR SOLVING OPTIMAL CONTROL PROBLEMS

Chih-Hung Huang and Ching-Huan Tseng*

ABSTRACT

This paper focuses on the development of a solver for solving optimal control problems. A developed numerical optimal control module integrated with the Sequential Quadratic Programming method is introduced. An optimal control problem solver based on the proposed method is implemented to solve optimal control problems efficiently in engineering applications. In addition, a systematic procedure for solving optimal control problems by using the optimal control problem solver is also proposed. A time-optimal benchmark problem presented in the literature is used to illustrate for the capability and facility of solving optimal control problems. The numerical results demonstrate the proposed method and the procedure suggested in this paper are helpful to engineers in solving optimal control problems in a systematic and efficient manner.

Key Words: nonlinear programming (NLP), optimal control problem (OCP), sequential quadratic programming (SQP).

I. INTRODUCTION

Two typical methods are usually used to solve optimal control problems: the indirect and direct approaches. The indirect approach is based on the solution of the first order necessary conditions for optimality. Pontryagin Minimum Principle (Pontryagin *et al.*, 1962) and the dynamic programming method (Bellman 1957) are two common methods utilizing the indirect approach. The direct method (Jaddu and Shimemura 1999; Hu *et al.*, 2002) is based on nonlinear programming (NLP) approaches that transcribe optimal control problems into NLP problems and apply existing NLP techniques to solve them. In most of practical applications, the control problems are described by strongly nonlinear differential equations hard to be solved by indirect methods. For those cases, direct methods can provide another choice to find the solutions.

In spite of extensive use of direct and indirect methods to solve optimal control problems, engineers still spend much effort on reformulating problems and implementing corresponding programs for different control problems. For engineers, this routine job will be tedious and time-consuming. Therefore, a systematic computational procedure for various optimal control problems has become an imperative for engineers, particularly for those who are inexperienced in optimal control theory or numerical techniques. Hence, the purpose of this paper is to apply NLP techniques to implement an OCP solver that assists engineers in solving optimal control problems with a systematic and efficient procedure. To illustrate the practicality and convenience of the proposed solver, a benchmark problem presented in the literature is chosen to illustrate the capability for solving optimal control problems. The results demonstrate the proposed solver can get the solution correctly and the procedure suggested in this paper can help engineers to deal with their problems.

The paper is organized as follows. In Section II, a general formulation of optimal control problems is given. The proposed NLP method and computational architecture for solving OCP are discussed in

*Corresponding author. (Tel: 886-3-5726111 ext. 55155; Fax: 886-3-5717243; Email: chtseng@cc.nctu.edu.tw)

The authors are with the Department of Mechanical Engineering, National Chiao Tung University, Hsinchu, Taiwan 300, R.O.C.

Section III. The systematic procedure by applying proposed solver to solve the OCP is described in Section IV. A benchmark problem presented in the literature is described and the numerical results obtained by applying the OCP solver are also demonstrated in Section V. Conclusions are drawn in Section VI.

II. GENERAL FORMULATION OF OPTIMAL CONTROL PROBLEMS

The generalized Bolza problem formulation for optimal control problems can be defined as follows: Find the design variables $\boldsymbol{b}$, the control functions $\boldsymbol{u}(t)$ and terminal time t_f which minimize the performance index

$$J_0 = \psi_0(\boldsymbol{b}, \boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} F_0(\boldsymbol{b}, \boldsymbol{u}(t), \boldsymbol{x}(t), t)dt \qquad (1)$$

subject to the state (or system) equations

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{b}, \boldsymbol{u}(t), \boldsymbol{x}(t), t),\ t_0 \le t \le t_f \qquad (2)$$

with initial conditions

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0(\boldsymbol{b}) \qquad (3)$$

functional constraints

$$J_i = \psi_i(\boldsymbol{b}, \boldsymbol{x}(t_f), t_f) + \int_{t_0}^{t_f} F_i(\boldsymbol{b}, \boldsymbol{u}(t), \boldsymbol{x}(t), t)dt \begin{cases} = 0;\ i = 1, \cdots, r' \\ \le 0;\ i = r' + 1, \cdots r \end{cases} \qquad (4)$$

and dynamic point-wise constraints

$$\phi_j(\boldsymbol{b}, \boldsymbol{u}(t), \boldsymbol{x}(t), t) \le 0;\ j = 1, \cdots, q \qquad (5)$$

where $\boldsymbol{b} \in \boldsymbol{R}^k$ is a vector of the design variables, $\boldsymbol{u}(t) \in \boldsymbol{R}^m$ is a vector of the control functions, and $\boldsymbol{x}(t) \in \boldsymbol{R}^n$ is a vector of the state variables. The functions $\boldsymbol{f}$, $\boldsymbol{\Psi}_0$, F_0, $\boldsymbol{\Psi}_i$, F_i and ϕ_j are assumed to be at least twice differentiable.

The preceding definition extends the original Bolza problem to account for inequality constraints, as the original Bolza formulation containing only equality constraints is not general for the OCP. It also does not treat the design variables $\boldsymbol{b}$, which may serve a variety of useful purposes apart from obvious design parameters; e.g., weight and velocity of a vehicle. Also, when the terminal time t_f is unconstrained (for optimization), a free time problem is obtained. Otherwise a fixed time problem is given. In addition, the initial conditions are separated from the functional constraints in Eq. (4) for practical considerations and the terminal conditions are treated as equality constraints in the first term of Eq. (4). The differential equations for the system in Eq. (2) are written in general first-order form. Eq. (5) represents the mixed state and control inequality dynamic constraints.

III. NLP METHODS FOR SOLVING OCP

As mentioned in Section I, two common methods, the indirect and direct approaches, used to solve optimal control problems can be found in the literature. Each method has its fitness and difficulties for solving OCP. In this paper, a direct approach based on nonlinear programming (NLP) is adopted to develop an OCP solver. According to the strategies of discretization, NLP methods for solving OCP can be separated into two groups: the simultaneous and sequential strategies. In the simultaneous methods, the state and control variables are fully discretized and led to large-scale NLP problems that usually require special solution strategies (Cervantes and Biegler 2000) to obtain the solutions. In sequential NLP methods, only the control variables are discretized. Obviously, the sequential NLP method has smaller design spaces and is more efficient than simultaneous NLP methods. Therefore, this paper is focused on the sequential NLP method and applies it to develop the OCP solver.

Sequential Quadratic Programming (SQP) is one of the best NLP methods for solving large-scale nonlinear optimization and is frequently applied to solve optimal control problems (see, e.g., Gill *et al.*, 2002, Betts 2000). Before applying the SQP methods, optimal control problems in which the dynamics are determined by a system of ordinary differential equations (ODEs) are usually transcribed into nonlinear programming (NLP) problems by discretization strategies.

1. Discretizing the Control Functions

The entire time interval $[t_0, t_f]$ is subdivided into N general unequal time intervals and the grid is designated as

$$t_0, t_1, t_2, \cdots, t_{N-1}, t_N = t_f \qquad (6)$$

The time intervals between the grid points are defined in a vector form as

$$\boldsymbol{T} = [T_1, T_2, \cdots, T_N]^T \qquad (7)$$

where $T_i = t_i - t_{i-1}$ and $\sum_{i=1}^{N} T_i = t_f - t_0$ which generate the parameter set

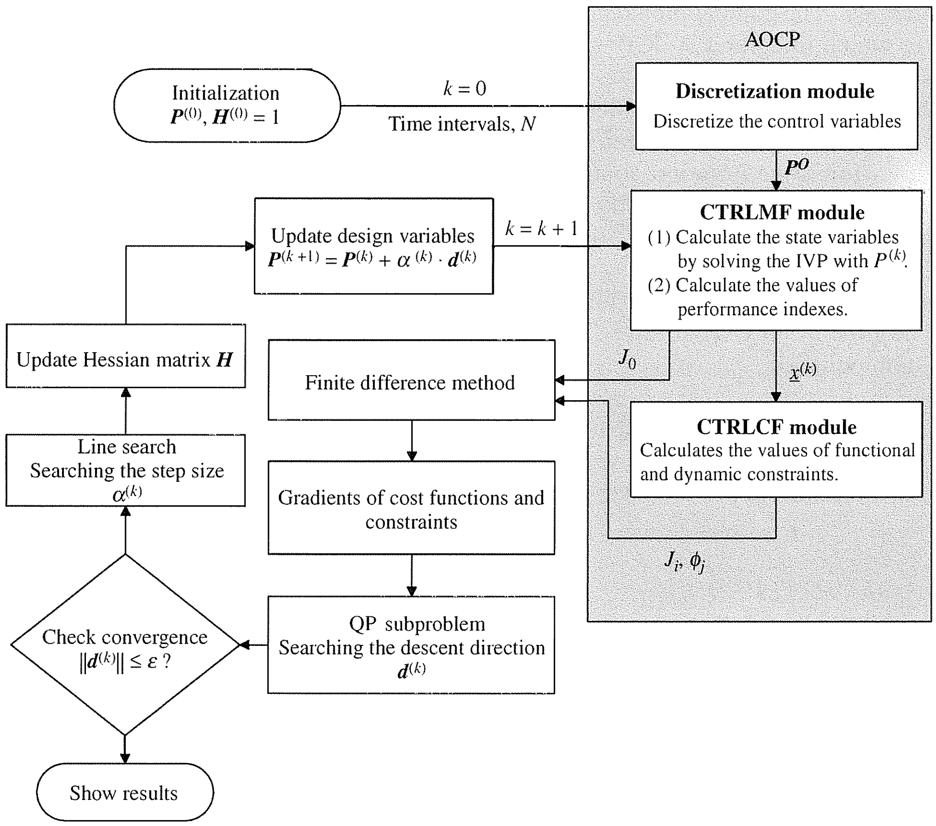


Fig. 1 Conceptual flow chart of the SQP method for solving OCP

$$
\begin{aligned}
U &= [u^{(1)}, u^{(2)}, \cdots, u^{(N)}]^T \\
&= [u_1(t_0), \cdots, u_m(t_0), u_1(t_1), \cdots, u_m(t_1), \cdots, \\
&\quad u_1(t_{N-1}), \cdots, u_m(t_{N-1})]^T \\
&= [U_1, \cdots, U_m, U_{m+1}, \cdots, U_{2m}, U_{2m+1}, \cdots, \\
&\quad U_{(N-1)m+1}, \cdots, U_{mN}]^T \qquad (8)
\end{aligned}
$$

where $u^{(k)} \in R^m$ is the vector of control variables at the k-th time grid point. The continuity of the $u^{(k)}$ and their derivatives at the time grids are enforced by means of appropriate linear equality constraints. Any bounds on the $u^{(k)}$ at the nodes imply additional linear inequalities on the coefficients of the polynomial.

2. Admissible Optimal Control Problem Formulation

In this paper, the Admissible Optimal Control Problem (AOCP) formulation, which is based on sequential NLP methods, is developed and implemented. With AOCP, the system equation in Eq. (2) with initial condition in Eq. (3) is formed as an initial value problem (IVP) and the corresponding values of state variables can be calculated by solving the problem with the initial conditions x_0 and the values of design variables in each iteration. As mentioned before, the values of control can be approximated by a piecewise polynomial function, in which the coefficients are treated as design variables and determined in each iteration of SQP. Hence, Eqs. (2) and (3) form an IVP of state variables. Some good first order differential equation methods having variable step size and error control are available to solve the IVP, e.g. Adam's method and the Runge-Kutta-Fehlberg method. These solvers can give accurate results with user-defined error control. The state trajectories are internally approximated using interpolation functions in the differential equation solvers. Values of the state and control variables between the grid points can be also obtained with different kinds of interpolation schemes.

3. Computational Algorithm of AOCP

The architectural framework of the OCP solver, illustrated in Fig. 1, is composed of SQP and AOCP algorithms. The AOCP algorithm contains three major modules: discretization, CTRLMF and CTRLCF. The discretization module, which is mentioned in Section III.1, discretizes the control inputs according to specified time intervals. The computational algorithm of the OCP solver which integrates AOCP with SQP can be described as the following steps:

Given: Initial values of the design variables vector $\boldsymbol{P}^{(0)} = [\boldsymbol{b}^{(0)}, \boldsymbol{U}^{(0)}, \boldsymbol{T}^{(0)}]$ and Number of time intervals, N. Initialize iteration counter k: = 0 and Hessian Matrix $\boldsymbol{H}^{(0)}$: = Identity $\boldsymbol{I}$.

1. Current design variable vector, $\boldsymbol{P}^{(k)}$, is passed to **CTRLMF** module of AOCP.
2. Evaluate the values of state variable, $\boldsymbol{x}^{(k)}$, by solving the IVP by substituting $\boldsymbol{P}^{(k)}$ into the system equation.

$$\dot{\boldsymbol{x}}^{(k)} = \boldsymbol{f}(\boldsymbol{b}^{(k)}, \boldsymbol{u}^{(k)}, \boldsymbol{x}^{(k)}, t),\ \boldsymbol{x}(t_0) = \boldsymbol{x}_0(\boldsymbol{b}^{(k)}) \tag{9}$$

3. Compute the values of performance indexes, $\boldsymbol{J}_0{}^{(k)}$.

$$J_0^{(k)} = \psi_0(\boldsymbol{b}^{(k)}, x(\boldsymbol{b}^{(k)}, \boldsymbol{U}^{(k)}, \boldsymbol{T}^{(k)}, t_f), t_f) + \int_{t_0}^{t_f} F_0(\boldsymbol{b}^{(k)}, \boldsymbol{U}^{(k)}, \boldsymbol{x}^{(k)}, t)dt \tag{10}$$

4. Substitute $\boldsymbol{x}^{(k)}$ into Eqs. (4) and (5) to evaluate the values of functional and dynamic constraints.
5. Evaluate $\nabla \boldsymbol{J}_0{}^{(k)}$, $\nabla \boldsymbol{J}_j{}^{(k)}$, and $\nabla \phi_j{}^{(k)}$ by using the finite difference method.
6. Find the descent direction, $\boldsymbol{d}^{(k)}$, by solving the QP subproblem.
7. Check convergence criteria, $\boldsymbol{d}^{(k)} \leq \varepsilon$. If satisfied, stop and show the results.
8. Compute the step size, $\alpha^{(k)}$.
9. Update Hessian Matrix $\boldsymbol{H}^{(k)}$ by applying BFGS method.
10. Update design variables

$$\boldsymbol{P}^{(k+1)} = \boldsymbol{P}^{(k)} + \alpha \cdot \boldsymbol{d}^{(k)} \tag{11}$$

11. Increase iteration counter, $k \leftarrow k + 1$, go back to step 1.

IV. SYSTEMATIC PROCEDURE FOR OCP

In this paper, the OCP is converted into an NLP problem by a discretization process and an admissible optimal control formulation mentioned in Section III. Then the optimizer based on the SQP method is used to solve the NLP problem numerically. In this paper the discretization process and the numerical schemes discussed in the previous section are implemented in the OCP solver. All of the complicated details of the transformation and numerical algorithms have been implemented in the OCP solver. The optimal control and state trajectories will be obtained and recorded in the output files. With the proposed OCP solver, engineers can focus their efforts on formulating their problems and then follow an efficient and systematic procedure to solve their optimal control problems. The following steps describe a systematic procedure for solving the OCP with the proposed OCP solver:

1. Program formulation: The original optimal control problem must be formulated according to the extended Bolza formulation.
2. Preparing two parameter files: One of the parameter files describes the numerical schemes used to solve the OCP and also the relationships between performance index, constraint functions, dynamic functions, state variables and control variables. The other parameter file includes the information on SQP parameters, such as convergence parameter, upper/lower bound and initial guess of design variables, etc.
3. Implementing user-defined subroutines.
4. Execute the optimization: The user-defined subroutines are compiled and then linked with the SQP solver, MOST (Tseng *et al.*, 1996). Then, execute the optimization.

Obviously, the proposed OCP solver simplifies the computational procedure for solving OCP and aids engineers and students in solving optimal control problems.

V. NUMERICAL EXAMPLES

Time-Optimal Rest-to-Rest Maneuvering Problem

A single-axis, rest-to-rest maneuvering problem of flexible spacecraft used as a benchmark problem in many studies (Driessen 2000, Pao 1996, Liu and Wie 1992, Wie *et al.*, 1993) is chosen as an example of the time-optimal control problem in this section. The system model, shown in Fig. 2(a), only with a scalar control input $u_1(t)$ is considered here. Following the NLP formulation described in Section III, the optimal control can be defined as follows.

$$\text{Minimize } J_0 = \int_0^{t_f} dt = t_f \tag{12}$$

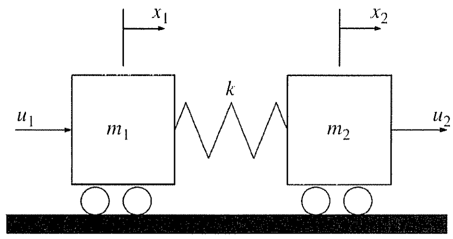
Subject to

$$\begin{aligned} \dot{x}_1 &= x_3 \\ \dot{x}_2 &= x_4 \\ \dot{x}_3 &= \frac{u_1}{m_1} - \frac{k}{m_1}(x_1 - x_2) \\ \dot{x}_4 &= \frac{k}{m_2}(x_1 - x_2) \end{aligned} \tag{13}$$

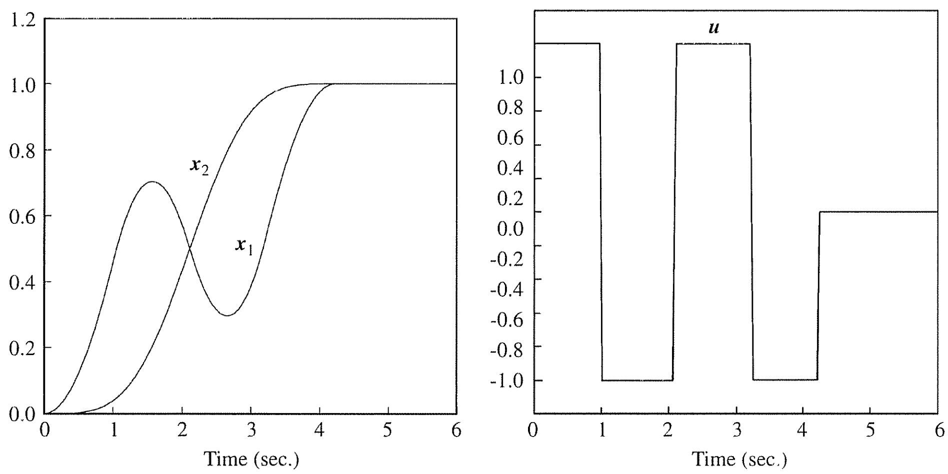
with initial states

$$\boldsymbol{x}^T(0) = [0, 0, 0, 0]^T \tag{14}$$

where x_1 and x_2 are the positions of body 1 and body



(a) Two-mass-spring system model (Liu and Wie 1992)



(i) State trajectories

(ii) Control trajectories

(b) Numerical results

Fig. 2 Time-optimal rest-to-rest maneuvering problem

2, respectively, the nominal parameters are $m_1 = m_2 = k = 1$ with appropriate units, and time is in seconds. The terminal state constraints and saturation constraints on control are described as:

$$\psi_1 = x_1(t_f) - 1 = 0 \tag{15}$$

$$\psi_2 = x_2(t_f) - 1 = 0 \tag{16}$$

$$\psi_3 = x_3(t_f) = 0 \tag{17}$$

$$\psi_4 = x_4(t_f) = 0 \tag{18}$$

$$\psi_5 = |u_1| - 1 \leq 0 \tag{19}$$

Time-optimal control problems often occur in many practical control problems. In this case, the derivation of the PMP is complex and thus the details are skipped. Following the procedure described in Section IV, users only need prepare two parameter files and user routines. Table 1 shows the user routines of this problem. By applying the proposed method and suggested procedure, a solution as $t_f = 4.2178746$ is obtained and the trajectories of the states and control input are shown in Fig. 2(b). In this problem, the proposed solver also obtains three switching times of input control as 1.00266823, 2.10892571 and 3.21518969. Those results agree with the results obtained by Liu and Wie (1992).

As the numerical results show, OCP is successfully converted into an NLP problem with the admissible control formulation and solved with the proposed method. The results show that the proposed method is applicable. According to the procedure suggested in this paper, users need not spend a vast amount of effort on programming in order to obtain solutions to problems. After formulating the problems and writing the user-defined routines, the proposed solver can solve the problems easily.

VI. CONCLUSIONS

An optimal control problem solver, the OCP solver, based on the Sequential Quadratic Programming (SQP) method and integrated with many well-developed numerical routines is implemented in this paper. A systematic procedure for solving optimal control problems is also offered in this paper. A high-order nonlinear time-optimal control problem is used to demonstrate the capability of the OCP solver. The results show that the OCP solver can help engineers in solving optimal control problems with a systematic and efficient procedure.

Table 1 User routines for solving the Benchmark problem

```
// Parameters for numerical examples
#define m1 1.0
#define m2 1.0
#define k 1.0
//Routine to calculate the integral term of the performance index or functional constraint.
void ffn(double *B, double *U, double *Z, double *T, double *F, int NV, int NU, int NEQ, int N, int NBJ)
{
   *F = 0.0;
}
// Routine to calculate the first term of the performance index or functional constraint
// or dynamic constraint.
void gfn(double *B, double *U, double *Z, double *T, double *G, int NV, int NU, int NEQ, int N, int NBJ)
{
   switch (N) {
      case 0:
         *G = B[3];  /* B[3]:terminate time tf */
          break;
      case 1:
         *G = Z[0] - 1.0;  /* terminal constraints */
          break;
      case 2:
         *G = Z[1] – 1.0;
          break;
      case 3:
         *G = Z[2] ;
          break;
      case 4:
         *G = Z[3] ;
          break;
   };
}
// Routine to calculate the state trajectory.
void hfn(double *B, double *U, double *Z, double *DZ, double *T, int NV, int NU, int NEQ)
{
      DZ[0] = Z[2];
      DZ[1] = Z[3];
      DZ[2] = (U[0]/m1)-(k/m1)*(Z[0]-Z[1]);
      DZ[3] = (k/m2)*(Z[0]-Z[1]);
}
```


ACKNOWLEDGMENTS

The research reported in this paper, was supported by a the National Science Council Grant, Taiwan, R.O.C., NSC90-2212-E009-039, which is greatly appreciated.

NOMENCLATURE

$\boldsymbol{b}$	design variables
$\boldsymbol{d}^{(k)}$	descent direction defined in SQP algorithm
t_0	start time
t_f	terminal time
$\boldsymbol{u}$	control variable vector
$\boldsymbol{x}$	state variable vector
$\boldsymbol{H}$	Hessian Matrix
N	number of time intervals
$\boldsymbol{P}$	extended design variable vector
T_i	the i-th time grid point
α	step size of SQP algorithm
ε	convergence parameter of SQP algorithm

REFERENCES

Bellman, R., 1957, *Dynamic Programming*, Princeton University Press, Princeton, NJ, USA.

Betts, J. T., 2000, "Very Low-Thrust Trajectory Optimization Using a Direct SQP Method," *Journal of Computational and Applied Mathematics*, Vol. 120, pp. 27-40.

Cervantes, A., and Biegler, L. T., 2000, "A Stable Elemental Decomposition for Dynamic Process Optimization," *Journal of Computational and Applied Mathematics*, Vol. 120, pp. 41-57.

Driessen, B. J., 2000, "On-Off Minimum-Time Control with Limited Fuel Usage: Near Global Optima via Linear Programming," *Proceedings of the American Control Conference*, Chicago, IL, USA, pp. 3875-3877.

Gill, P. E., Murray, W., and Saunders, M. A., 2002, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," *SIAM Journal on Optimization*, Vol. 12, No. 4, pp. 979-1006.

Hu, G. S., ONG, C. J., and Teo, C. L., 2002, "An Enhanced Transcribing Scheme for The Numerical Solution of a Class of Optimal Control Problems," *Engineering Optimization*, Vol. 34, No. 2, pp. 155-173.

Jaddu, H., and Shimemura, E., 1999, "Computational Method Based on State Parameterization for Solving Constrained Nonlinear Optimal Control Problems," *International Journal of Systems Science*, Vol. 30, No. 3, pp. 275-282.

Liu, Q., and Wie, B., 1992, "Robust Time-Optimal Control of Uncertain Flexible Spacecraft," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 3, pp. 597-604.

Pao, L. Y., 1996, "Minimum-Time Control Characteristics of Flexible Structures," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 1, pp. 123-129.

Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V., and Mischenko, E. F., 1962, *The Mathematical Theory of Optimal Processes*, Wiley, New York, USA.

Tseng, C. H., Liao, W. C., and Yang, T. C., 1996, "MOST 1.1 User's Manual," *Technical Report No. AODL-96-01*, Department of Mechanical Engineering, National Chiao Tung University, Taiwan, R.O.C..

Wie, B., Sinha, R., and Liu, Q., 1993, "Robust Time-Optimal Control of Uncertain Structural Dynamic Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No.5, pp. 980-983.

Manuscript Received: Jun. 07, 2004
Revision Received: Sep. 12, 2004
and Accepted: Oct. 20, 2004

Proceedings of the Fourth IASTED International Conference
MODELLING, SIMULATION, AND OPTIMIZATION
August 17-19, 2004, Kauai, Hawaii, USA

Numerical Approaches for Solving Dynamic System Design Problems: An Application to Flight Level Control Problem

C.H. Huang* and C.H. Tseng**
Department of Mechanical Engineering, National Chiao Tung University
Hsinchu 30056, Taiwan, R. O. C. E-mail: chtseng@mail.nctu.edu.tw
TEL: 886-3-5726111 Ext. 55155 FAX: 886-3-5717243

(*Research Assistant, **Professor)

ABSTRACT

The optimal control theory can be applied to solve the optimization problems of dynamic system. Two major approaches which are used commonly to solve optimal control problems (OCP) are discussed in this paper. A numerical method based on discretization and nonlinear programming techniques is proposed and implemented an OCP solver. In addition, a systematic procedure for solving optimal control problems by using the OCP solver is suggested. Two various types of OCP, A flight level tracking problem and minimum time problem, are modeled according the proposed NLP formulation and solved by applying the OCP solver. The results reveal that the proposed method constitutes a viable method for solving optimal control problems.

KEY WORDS

optimal control problem, nonlinear programming, flight level tracking problem, minimum time problem, SQP, AOCP.

1. INTRODUCTION

Over the past decade, applications in dynamic system have increased significantly in the engineering. Most of the engineering applications are modeled dynamically using differential-algebraic equations (DAEs). The DAE formulation consists of differential equations that describe the dynamic behavior of the system, such as mass and energy balances, and algebraic equations that ensure physical and dynamic relations. By applying modeling and optimization technologies, a dynamic system optimization problem can be re-formulated as an optimal control problem (OCP). There are many approaches can be used to deal with these OCPs. In particular, OCPs can be solved by a variational method [1, 2] or by Nonlinear Programming (NLP) approaches [3-5].

The indirect or variational method is based on the solution of the first order necessary conditions for optimality that are obtained from Pontryagin's Maximum Principle (PMP) [1]. For problems without inequality constraints, the optimality conditions can be formulated as a set of differential-algebraic equations which is often in the form of two-point boundary value problem (TPBVP). The TPBVP can be addressed with many approaches, including single shooting, multiple shooting, invariant embedding, or some discretization method such as collocation on finite elements. On the other hand, if the problem requires the handling of active inequality constraints, finding the correct switching structure as well as suitable initial guesses for state and co-state variable is often very difficult.

Much attention has been paid to the development of numerical methods for solving optimal control problems [6, 7]. Most popular approach in this field turned to be reduction of the original problem to a NLP. A NLP consists of a multivariable function subject to multiple inequality and equality constraints. The solution of the nonlinear programming problem is to find the Kuhn-Tucker points of equalities by the first-order necessary conditions. This is the conceptual analogy in solving the optimal control problem by the PMP. NLP approaches for OCPs can be classified into two groups: the sequential and the simultaneous strategies. In simultaneous strategy the state and control variable are fully discretized, but in the sequential strategy only discretizes the control variables. The simultaneous strategy often leads the optimization problems to large-scale NLP problems which usually require special strategies to solve them efficiently. On the other hand, instability questions will arise if the discretizations of control and state profiles are applied inappropriately. Comparing to the simultaneous NLP, the sequential NLP is more efficient and robust when the system contains stable modes. Therefore, the admissible optimal control problems which bases on the sequential NLP strategy is propose to solve the dynamic optimization problems in this paper. To facilitate engineers to solve their optimal control problems, a general optimal control problem solver which integrates proposed method with SQP algorithm is developed.

In spite of extensive use of nonlinear programming methods to solve optimal control problems, engineers still spend much effort reformulating nonlinear programming problems for different control problems. Moreover, implementing the corresponding programs of the nonlinear programming problem is tedious and time-consuming. Therefore, a systematic computational procedure for various optimal control problems has become an imperative for engineers, particularly for those who are inexperienced in optimal control theory or numerical techniques. Hence, the other purpose of this paper is to apply nonlinear mathematical programming techniques to implement a general optimal control problem solver that facilitates engineers in solving optimal control problems with a systematic and efficient procedure.

Flight level tracking plays an important role in autopilot systems receives considerable attentions in many researches [8-12]. For a commercial aircraft, its cruising altitude is typically assigned a flight level by Air Traffic Control (ATC). To ensure aircraft separation, each aircraft has its own flight level and the flight level is separated by a few hundred feet. Changes in the flight level happen occasionally and have to be cleared by ATC. At all other times the aircraft have to ensure that they remain within allow bounds of their assigned level. At the same time, they also have to maintain limits on their speed, flight path angle, acceleration, etc. imposed by limitations of the airframe and engine, passenger comfort requirements, or to avoid dangerous situations such as aerodynamic stall. In this paper, the flight level tracking problem is formulated into an optimal control problem. For safety reasons, the speed of the aircraft and the flight path angle has to be kept in a safe "aerodynamic envelope" [9] and the envelope can be translated into the dynamic constraints of the optimal control problem. A flight level tracking problem and a minimum time problem are shown in Section 5 and then solved by the proposed method.

2. NLP FORMULATION

The formulation of admissible optimal control problems (AOCP) which bases on the sequential strategy is derived by Huang and Tseng [3]. Various types of OCPs are solved successfully by applying AOCP and the formulation is melded with SQP algorithm to develop a general optimal control solver, the OCP solver. Because the NLP formulation based on the AOCP will be applied to solve aircraft flight control problems, a brief description of the NLP formulation and AOCP algorithm is helpful to understand.

Find the design variables $\mathbf{P} = [\mathbf{b}^{\mathrm{T}}, \mathbf{T}^{\mathrm{T}}, \mathbf{U}^{\mathrm{T}}]^{\mathrm{T}}$ to minimize a performance index

$$J_0 = \psi_0[\mathbf{b}, \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t_f), t_f] + \int_{t_0}^{t_f} F_0[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt \tag{1}$$

subject to state equations

$$\dot{\mathbf{x}} = \mathbf{f}[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), x(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt, \; t_0 \le t \le t_f \tag{2}$$

with initial conditions

$$\mathbf{x}(t_0) = \mathbf{h}(\mathbf{b}) \tag{3}$$

functional constraints as

$$J_i = \psi_i[\mathbf{b}, \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t_f), t_f] + \int_{t_0}^{t_f} F_i[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt \begin{Bmatrix} = 0; i = 1, \ldots\ldots\ldots, r' \\ \le 0; i = r'+1, \ldots., r \end{Bmatrix} \tag{4}$$

and dynamic constraints as

$$\phi_j[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t] \begin{Bmatrix} = 0; j = 1, \ldots\ldots\ldots, q' \\ \le 0; j = q'+1, \ldots., q \end{Bmatrix} \tag{5}$$

This NLP formulation presents a general form that includes equality/inequality, functional and dynamic constraints and can be applied to a variety of control problems of engineering applications.

AOCP ALGORITHM

The architectural framework of the OCP solver illustrated in Fig. 1 is composed of SQP and AOCP algorithms. The AOCP algorithm contains three major modules: discretization, CTRLMF and CTRLCF. Each SQP iteration the values of design variable vector $\mathbf{P}^{(k)}$ is passed into the CTRLMF module to compute the values of state variables by solving the initial value problem and then the values of performance indexes can be evaluated. After the CTRLMF module, the CTRLCF module uses the values of state variables calculated by CTRLMF module to compute the values of constraints. The values of the performance indexes and constraints are also passed back to the SQP algorithm and used to calculate the gradient information. In SQP algorithm, the gradient information will be used to evaluate the convergence and update the design variable vector $\mathbf{P}^{(k+1)}$. If the convergence criteria are satisfied, the algorithm be stopped and shows the results. SQP is a robust and popular optimization solver and the details can be found in many literatures. Because SQP is the computational foundation of proposed method and hence the convergence and sensitivity of proposed method is same as the convergence and sensitivity of SQP algorithm. The convergence of SQP algorithm has been proposed in many literatures (e.g. [15]). Büskens and Maurer [16] provide a detail description of the sensitivity analysis of SQP method for solving OCP. In this paper, a general optimization solver, MOST [13], which bases on SQP is chosen to develop a general OCP solver.

With admissible optimal control, some good first order differential equation methods having variable step size and error control are available to solve the DAE which is composed of Eqs. (12) and (13), e.g. Adams method and Runge-Kutta-Fehlberg method [14]. These solvers can give accurate results with user desired error control. The state trajectories are internally approximated using interpolation functions in the differential equation solvers. Values of the state and control variables between the grid points can be also obtained with different kinds of interpolation schemes. These numerical schemes are also included and implemented in the proposed OCP solver.

SYSTEMATIC PROCEDURE FOR SOLVING OCP

In this paper, the OCP is converted into NLP problem with admissible optimal control formulation and then the optimizer based on SQP method is used to solve the NLP problem numerically. Most of the numerical schemes are implemented in the OCP solver and the complicated details of the transformation and programming will be completed in the OCP solver automatically. The optimal control and state trajectories will be obtained and recorded in the output files. Therefore, engineers can follow an efficient and systematic procedure to solve various optimal control problems. The procedure for solving the OCP with the OCP solver is described as follows.

1. Program formulation: The original optimal control problem must be formulated according to the extended Bolza formulation.
2. Preparing two parameter files: One of the parameter files describes the numerical schemes used to solve the OCP and also the relationships between performance index, constraint functions, dynamic functions, state variables and control variables. The other parameter file includes the information on SQP parameters, such as convergence parameter, upper/lower bound and initial guess of design variables, etc.
3. Implementing two user-defined subroutines.
4. Execute the optimization: The user-defined subroutines are compiled and then linked with the SQP solver, MOST. Then, execute the optimization.

Obviously, the proposed OCP solver simplifies the computational procedure for solving OCP and facilitates engineers and students in solving optimal control problems.

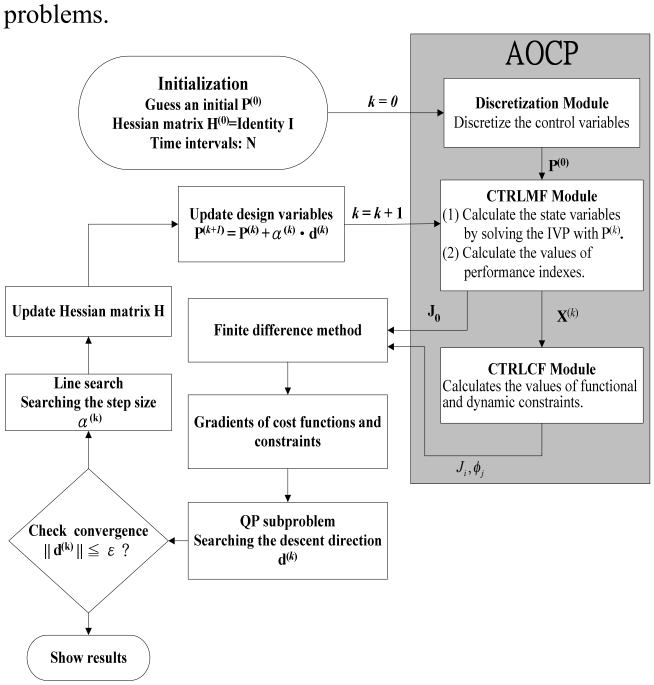


Fig.1 Conceptual flow chart of the AOCP method

3. AIRCRAFT MODEL

Many ATC researches [11, 12] apply a point mass model to describe the aircraft motion and only the movement of the aircraft in the lateral-directional is considered. In Figure 2, three coordinate frames are used to describe the motion of the aircraft: X_g-Y_g denotes the ground frame, the body frame denoted by X_b-Y_b and the X_w-Y_w denotes the wind frame. Besides, θ, γ, and α denote the rotation angle between the frames. $V \in \mathbb{R}$ represents the speed of the aircraft which is aligned with the positive X_w direction and h is the altitude of the aircraft.

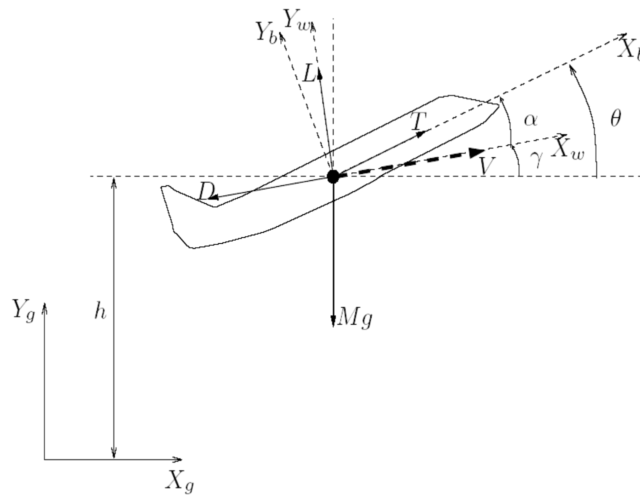


Fig. 2 The aircraft model (Lygeros 2003).

The equations of the motion can be derived from force balance relationships:

$$\begin{aligned} m\dot{V} &= T\cos\alpha - D - mg\sin\gamma \\ mV\dot{\gamma} &= L + T\sin\alpha - mg\cos\gamma \end{aligned} \tag{16}$$

Herein, T is the thrust exerted by the engine, D is the aerodynamic drag, and L is the aerodynamic lift. By applying basic aerodynamics, the lift (L) and drag (D) can be approximated by

$$\begin{aligned} L &= \frac{C_L S \rho V^2}{2}(1 + c\alpha) = a_L V^2 (1 + c\alpha) \\ D &= \frac{C_D S \rho V^2}{2} = a_D V^2 \end{aligned} \tag{17}$$

where C_L, C_D, and c are dimension-less lift and drag coefficients, S is the wing surface area, ρ is the air density. According to the admissible optimal control formulation described in Section 3, the air model can be formulated by a three state model with state variable vector $\mathbf{x}(t) = [x_1, x_2, x_3]^T = [V, \gamma, h]^T$ and control input vector $\mathbf{u}(t) = [u_1, u_2]^T = [T, \theta]^T$. By approximating α with a small angle, the equations of the motion (system equations) can be written as

$$\dot{\mathbf{x}} = \begin{bmatrix} -\dfrac{a_D}{m}x_1 - g\sin x_2 \\ \dfrac{a_L}{m}x_1(1-cx_2) - g\dfrac{\cos x_2}{x_1} \\ x_1 \sin x_2 \end{bmatrix} + \begin{bmatrix} \dfrac{1}{m} & 0 \\ 0 & \dfrac{a_L}{m}x_1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \quad (18)$$

This model proposed by Lygeros [10] which extends the three dimensions of an aerodynamic envelope protection problem is adopted. Considering the safety conditions, speed of aircraft and flight path angle are bounded in a rectangular limitations called "safe aerodynamic envelop". Lygeros [8] proposed a simplified aerodynamic envelope which is referred to Tomlin et al. [9] is adopted in this paper and translated into the following dynamic constraints.

$$\begin{aligned} &V_{\min} \le x_1 \le V_{\max} \\ &\gamma_{\min} \le x_2 \le \gamma_{\max} \\ &h_{\min} \le x_3 \le h_{\max} \end{aligned} \quad (19)$$

Following the NLP formulation described in Section 2, those constraints can be treated as dynamic constraints and rewritten as follows.

$$\begin{aligned} &\phi_1 : -x_1 + V_{\min} \le 0, \\ &\phi_2 : x_1 - V_{\max} \le 0, \\ &\phi_3 : -x_2 + \gamma_{\min} \le 0, \\ &\phi_4 : x_2 - \gamma_{\max} \le 0, \\ &\phi_5 : -x_3 + h_{\min} \le 0, \\ &\phi_6 : x_3 - h_{\max} \le 0, \end{aligned} \quad (20)$$

To illustrate the capabilities of the proposed method, the flight level tracking problem and minimum time problem are chosen in this paper.

Case I: Flight level tracking problem

A tracking problem is to find the controls to maintain the system state **x**(t) as close as possible to desired state **r**(t) in the interval $[t_0, t_f]$. The performance index for tracking problem can be written as

$$J = \int_{t_0}^{t_f} \|\mathbf{x}(t) - \mathbf{r}(t)\|^2_{\mathbf{Q}(t)} dt \quad (21)$$

where **Q**(t) is a real symmetric n × n matrix that is positive semi-definite for all $t \in [t_0,t_f]$. A flight level tracking problem is to keep the aircraft as possible to desire level and aircraft speed. Therefore, the performance index can be represented as

$$J = \frac{1}{2}\int_{t_0}^{t_f} \left[(x_1 - x_{1d})^2 + (x_2 - x_{2d})^2 + (x_3 - x_{3d})^2 \right] dt \quad (22)$$

where x_{1d} is the desired aircraft speed, x_{2d} is desired flight path angle and x_{3d} is the assigned altitude.

Case II: Minimum time problem

A minimum time problem is to transfer a system from an arbitrary initial state $\mathbf{x}(t_0) = \mathbf{x}_0$ to a specified target set **S** in minimum time. The performance index for minimum time problem can be written as

$$J = t_0 - t_f = \int_{t_0}^{t_f} dt \quad (23)$$

where t_f is the first instant of time when **x**(t) and **S** intersect. In some emergencies, the aircraft is asked to change their level as soon as possible.

4. NUMERICAL EXAMPLES

The following parameters are used in both cases:

a_L = 65.3 Kg/m,	a_D = 3.18 Kg/m,	m = 160×10^3 Kg,
g = 9.81 m/s^2,	θ_{min} = -20°,	γ_{min} = -20,
c = 6,	θ_{max} = 25°,	γ_{max} = 25,
T_{min} = 60×10^3 N,	T_{min} = 120×10^3 N,	V_{min} = 92 m/s,
V_{max} = 170 m/s,	h_{min} = -150 m,	h_{max} = 150 m

Case I: Flight level tracking problem

The initial values of state variables are

$\mathbf{x}_0 = [100, 20, -120]^T$ (24)

and the purpose of this problem is to find a suitable control to maintain the flight level and keep the aircraft altitude in assign altitude. Thus, the desired states are set with following values.

$\mathbf{r}(t) = [150, 0, 00]^T$ (25)

In addition to the dynamic constraints proposed in Eq. (20), the control inputs are also limited as following bounds:

$$\begin{aligned} &T_{\min} \le u_1 \le T_{\max}, \\ &\theta_{\min} \le u_2 \le \theta_{\max} \end{aligned} \quad (26)$$

Substitute parameters into Eqs. (18) and (22), the flight tracking problem is solved by the proposed OCP solver. The numerical results are shown in Figure 3. From Figure 3(a), all of the states meet the constraints and the flight level and aircraft speed return to the desired states. Table 1 shows the user subroutines of this case. Obviously, the proposed OCP solver provides an easier tool to solve dynamic optimization problems.

Case II: Minimum time problem

The aircraft is asked to increase their altitude in a minimum time. The initial and final altitude are h_0 = 0 m and h_f = 500 m respectively. All of the constraints imposed on Case I are also imposed on this case. The initial state $\mathbf{x}_0 = [100, 0, 0]^T$. The final time, t_f, obtained by using the AOCP is 73.98 seconds and the final altitude is 499.928 m. The control histories are shown in the Fig. 4(a) and Fig. 4(b) represent the state trajectories. From Fig.4, all of the trajectories meet the safe "aerodynamic envelope" (dynamic constraints).

5. CONCLUSIONS

In this study an optimal control problem solver, the OCP solver, based on the Sequential Quadratic Programming (SQP) method and integrated with many well-developed numerical routines is implemented. A systematic procedure for solving optimal control problems is also offered in this paper. Two common types of optimal control problems for flight level control are presented and solved by proposed method successfully. Numerical results show the proposed method can facilitate engineers in solving optimal control problems with a systematic and efficient procedure.

6. ACKNOWLEDGEMENTS

The research reported in this paper was supported under a project sponsored by the National Science Council Grant, Taiwan, R.O.C., NSC90-2212-E009-039, is greatly appreciated.

REFERENCES

[1] L.S. Pontryagin, V.G. Boltyanskii, R.V Gamkrelidze, and E.F. Mischenko, *The Mathematical Theory of Optimal Processes* (Wiley, 1962).

[2] D.E. Kirk, *Optimal Control Theory: An Introduction* (Prentice-Hall, 1970).

[3] C.H. Huang, and C.H. Tseng, Computational Algorithm for Solving A Class of Optimal Control Problems, *IASTED International Conference on Modelling, Identification, and Control (MIC2003)*, Innsbruck, Austria, 2003, pp. 118-123.

[4] G.S. Hu, C.J. ONG, and C.L. Teo, An Enhanced Transcribing Scheme for The Numerical Solution of A Class of Optimal Control Problems, *Engineering Optimization*, 34(2), 2002, 155-173.

[5] H. Jaddu, and E. Shimemura, Computational Method Based on State Parameterization for Solving Constrained Nonlinear Optimal Control Problems, *International Journal of Systems Science*, 30(3), 1999, 275-282.

[6] Polak E. *Computation Method in Optimization* (New York and London: Academic Press, 1971).

[7] K.L. Teo, and Z.S. Wu, Computation Methods for Optimizing Distributed Systems (Academic Press, Orlando, 1984).

[8] J. Lygeros, Minimum Cost Optimal Control: An Application to Flight Level Tracking, *IEEE 11th Mediterranean Conference on Control and Automation (MED'03)*, Rhodes, Greece, June 18-20, 2003.

[9] C. Tomlin, J. Lygeros, and S. Sastry, Aerodynamic envelope protection using hybrid control, *Computer Vision, and Materials Science* (Cambridge University Press, New York, 1996).

[10] J. Lygeros, C. Tomlin, and S. Sastry, Controllers for Reachability Specifications for Hybrid Systems, *Automatica*, 1999, 349-370.

[11] B. Etkin, and L.D. Redi, *Dynamics of Flight: Stability and Control*, 3rd Ed. (New York: Wiley, 1996).

[12] M.V. Cook, *Flight Dynamics Principles* (New York: Wiley, 1997).

[13] C.H. Tseng, W.C. Liao, and T.C. Yang, *MOST 1.1 User's Manual*, Technical Report No. AODL-93-01, Department of Mechanical Engineering, National Chiao Tung Univ., Taiwan, R.O.C., 1996.

[14] W. Press, S.A. Teukolsky, W.T. Vetterling, and B. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd Edition (New York: Cambridge, 1992).

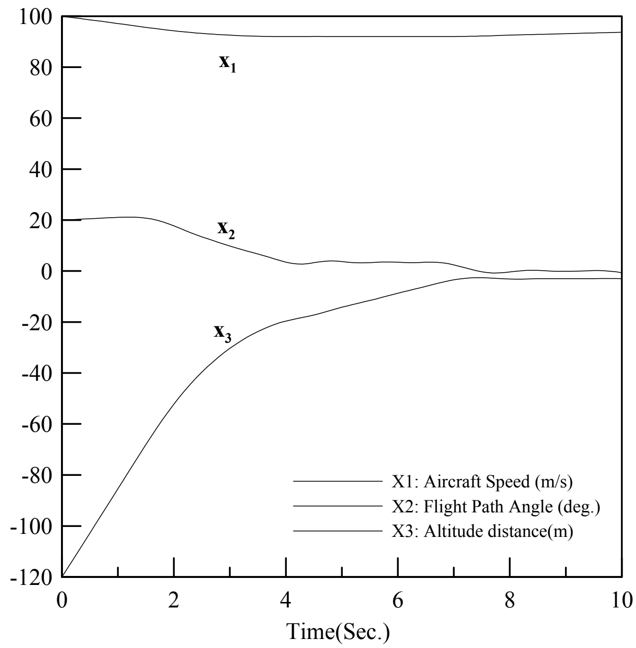
[15] P. T. Boggs and J. W. Tolle, Sequential Quadratic Programming, *Acta Numerica*, 1996, 1-52.

[16] C. Büskens, H. Maurer, SQP-methods for solving optimal control problems with control and state constraints: adjoint variables, sensitivity analysis and real-time control, *Journal of Computational and Applied Mathematics*, 120, 2000, 85-108.

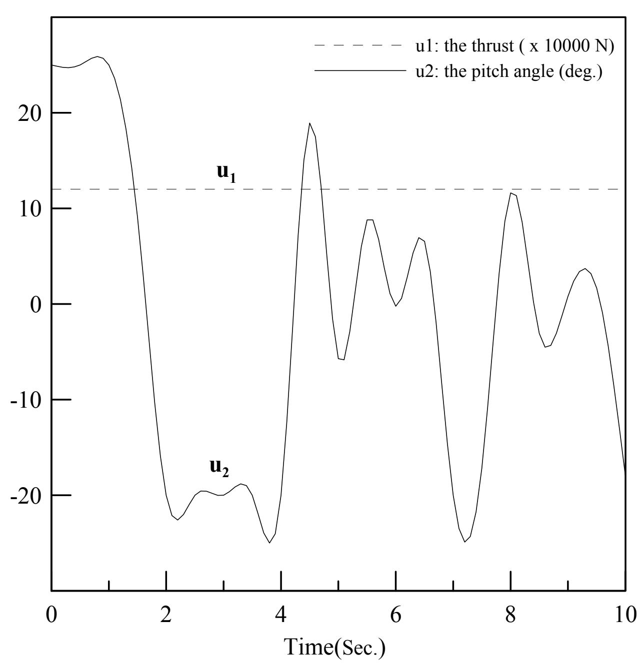
Table 1. User subroutines for flight level tracking problem

```
//-------Program parameters ---------------------------------------------
 //B: Discrete design parameters of design variable vector. (INPUT)
// U: Admissible control function vector. (INPUT)
// Z: State variable vector. (INPUT)
// T: Given time grid point. (INPUT)
// G: First term of performance index or functional constraint or
//     dynamic constraint. (OUTPUT)
// NV: Number of design variables for optimizer (INPUT)
// NU: Number of control functions. (INPUT)
// NEQ: Number of state equations (INPUT)
// N: Index of current number of function evaluation. (INPUT)
//
// -------------------------------FFN()---------------------------------------
// Routine to calculate the integral term of the performance index
// or functional constraint
void ffn(double *B, double *U, double *Z, double *T, double *F,
           int NV, int NU, int NEQ, int N, int NBJ)
{
  if (N==0)
         *F = 0.5*((Z[0]-150.0)*(Z[0]-150.0)) + (Z[1]*PI/180.0) *
               ( Z[1] * PI/180.0 ) + (Z[2]*Z[2]);
  else
          *F = 0.0;
}
//----------------------- GFN() -------------------------------------------------
// Routine to calculate the first term of the performance index or
// functional constraint or dynamic constraint
void gfn(double *B, double *U, double *Z, double *T, double *G,
            int NV, int NU, int NEQ, int N, int NBJ)
{
  switch (N)
  {
    case 0:
           *G = 0.0;   break;
    case 1:
          *G = -1 * Z[0] + 92.0; break;
    case 2:
          *G = Z[0] – 170.0; break;
    case 3:
          *G = -1 * Z[1] -20.0; break;
    case 4:
      *G = Z[1] – 25.0; break;
    case 5:
      *G = -1 * Z[2] -150.0; break;
    case 6:
      *G = Z[2] – 150.0;  break;
  };
}
//--------------------- HFN() --------------------------------------------------
//Routine to calculate the state trajectory.
void hfn(double *B, double *U, double *Z, double *DZ, double *T,
            int NV, int NU, int NEQ )
{
   DZ[0] = -1*((aD*Z[0]*Z[0]/m) + (g*sin(Z[1]*PI/180.0))) + U[0] *
              10000 / m;
     DZ[1] = (aL*Z[0]*(1-c*Z[1])/m) - (g*cos(Z[1]*PI/180.0)/Z[0]) +
                aL*c*Z[0]*U[1]/m;
     DZ[2] = Z[0]*sin(Z[1]*PI/180.0);
```

```
}
//--------------------- Z0FN() ----------------------------------------------------
// Routine to calculate the initial state vector.
void z0fn(double *B, double *ZINT, int NV, int NEQ )
{
   ZINT[0] = 100.0;
   ZINT[1] = 20.0;
   ZINT[2] = -120.0;
}
```

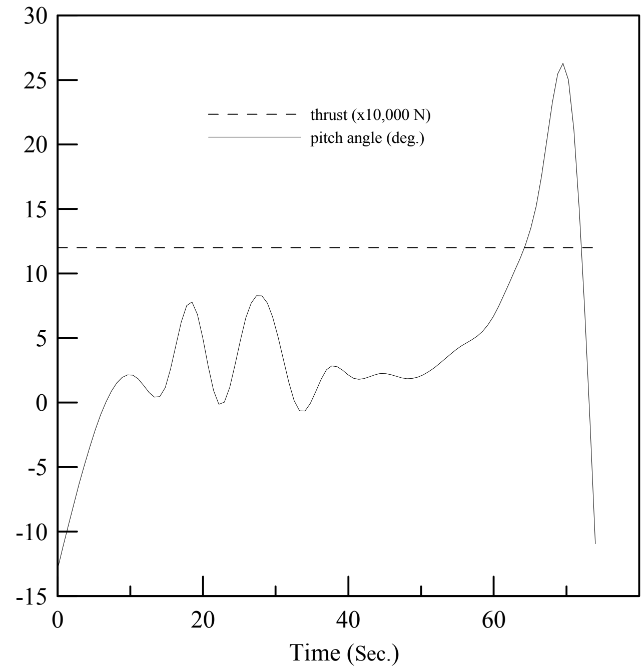


(a) State trajectories



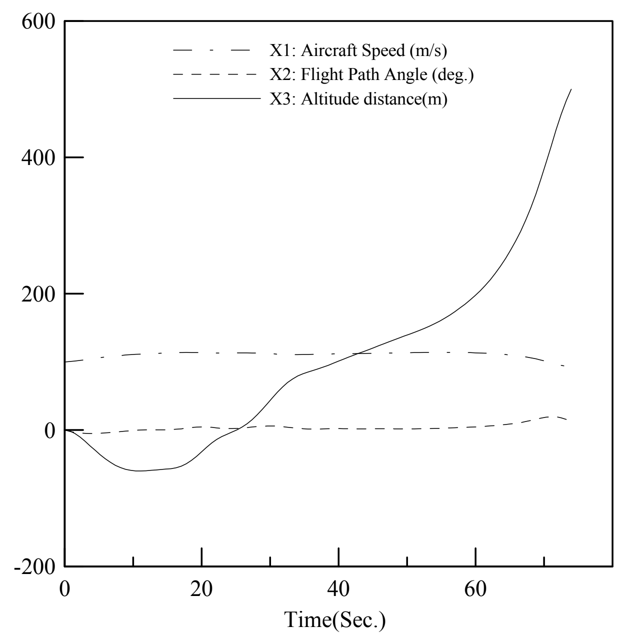
(b) The thrust trajectory

Fig. 3 Numerical results of the tracking problem.



(a) Control input trajectories

Fig. 4 Trajectories of the minimum time problem



(b) State trajectories

Fig. 4 Trajectories of the minimum time problem.

Proceedings of the 22nd IASTED International Conference
MODELLING, IDENTIFICATION, AND CONTROL
February 10-13, 2003, Innsbruck, Austria

COMPUTATIONAL ALGORITHM FOR SOLVING A CLASS OF OPTIMAL CONTROL PROBLEMS

C.H. Huang* and C.H. Tseng**
Department of Mechanical Engineering, National Chiao Tung University
Hsinchu 30056, Taiwan, R. O. C. E-mail: chtseng@mail.nctu.edu.tw
TEL: 886-3-5726111 Ext. 55155 FAX: 886-3-5717243

(*Graduate Student, **Professor)

ABSTRACT
This paper focuses on the numerical methods for solving the optimal control problems. An optimal control problem (OCP) solver is proposed to solve efficiently the optimal control problems in the engineering applications. In last two decades, discretization and parameterization techniques had been used to convert the optimal control problem into the nonlinear programming problem and then it can be solved it by using the standard optimization software. In this paper, a developed optimal control analytical module, an OCP solver, is integrated with the optimization software, MOST. A systematic procedure for solving optimal control problem by using the OCP solver is also proposed. Two various types of the optimal control problems presented in the literatures are chosen to verify the accuracy and efficiently for the OCP solver. From the numerical results in this paper, the trajectories for state and control variables agree with literatures' results. Therefore, the capability and accuracy of the OCP solver is demonstrated and it can facilitate engineers to solve various optimal control problems with a systematic and efficient procedure.

KEY WORDS
optimal control problem, nonlinear programming, discretization and parameterization techniques, admissible optimal control formulation.

1. INTRODUCTION

Optimal control problems (OCP) have a large number of applications in many different fields, e.g. mechanical system [1], automotive vehicle design [2,3] and manufacturing process [4]. However, finding the solution is often a difficult and time-consuming task, particularly for nonlinear and state constrained problems. The optimal control problems can be solved by the classical calculus of variations approach or by the dynamic programming method. On the other hand, many practical problems are described by strongly nonlinear differential equations and cannot easily get the analytical solutions. Hence, many approximation methods [5-8] based on nonlinear programming method are used to solve the nonlinear optimal control problem.

A nonlinear programming problem consists of a multivariable function subject to multiple inequality and equality constraints. The solution of the nonlinear programming problem is to find the Kuhn-Tucker points of equalities by the first-order necessary conditions. This is the conceptual analogy in solving the optimal control problem by the Pontryagin maximum principle [18]. Various discretization and parameterization techniques for state and control variables allow having an optimal solution for the OCP via the nonlinear programming. Jaddu and Shimemura [7] used the quasilinearization and state parameterization by using Chebyshev polynomials to solve constrained nonlinear optimal control problems. Hu et al. [8] applied an enhanced scheme based on the Direct Collocation and Nonlinear Programming Problem (DCNLP) to transform the system dynamics into constraints of the nonlinear programming. Many numerical examples are solved successfully by those methods.

In this paper, a different approach for solving the OCP is proposed and integrated with various numerical schemes to implement a general optimal control problem solver that facilitate engineers to solve their optimal control problems with a systematic and efficient procedure. With the solver, the OCP can be converted into the nonlinear programming problem automatically and then be solved by a general constrained optimization solver-MOST. The OCP solver allows the designer to control both of the optimization process and the analysis as well as design sensitivity analysis using his owning experience and intuition. Two numerical examples in the recent literatures will be used to demonstrate and verify the OCP solver in this paper. The numerical results show the OCP solver is effective and accurate for solving the optimal control problem.

2. PROBLEM FORMULATION

A general problem of continuous time optimal control system can be defined as follows:

Find the design variables **b**, the control functions **u** and terminal time t_f which minimize performance index

$$J_0 = \psi_0[\mathbf{b}, x(t_f), t_f] + \int_{t_0}^{t_f} F_0[\mathbf{b}, \mathbf{u}(t), x(t), t]dt \qquad (1)$$

subject to the state (or dynamic) equations

$$\dot{x} = \mathbf{f}[\mathbf{b}, \mathbf{u}(t), x(t), t]dt, \;\; t_0 \le t \le t_f \qquad (2)$$

with the initial conditions

$$x(t_0) = \mathbf{h}(\mathbf{b}) \qquad (3)$$

functional constraints

$$J_i = \psi_i[\mathbf{b}, x(t_f), t_f] + \int_{t_0}^{t_f} F_i[\mathbf{b}, \mathbf{u}(t), x(t), t]dt \begin{Bmatrix} = 0; i = 1, \ldots\ldots\ldots, r' \\ \le 0; i = r'+1, \ldots., r \end{Bmatrix} \qquad (4)$$

and dynamic point-wise constraints

$$\phi_j[\mathbf{b}, \mathbf{u}(t), x(t), t] \begin{Bmatrix} = 0; j = 1, \ldots\ldots\ldots, q' \\ \le 0; j = q'+1, \ldots., q \end{Bmatrix} \qquad (5)$$

where $\mathbf{b} \in R^k$ is a vector of design variables, $\mathbf{u} \in R^m$ is a vector of control functions, and $\mathbf{x} \in R^n$ is a vector of state variables. The functions ψ_0, F_0, ψ_i, f, F_i and ϕ_j are assumed two at least twice differentiable. The preceding definition extends the original Bolza problem to account for inequality constraints. The original Bolza formulation containing only equality constraints is not general for the OCP. It also does not treat design variables **b** which may serve a variety of useful purposes apart from obvious design parameters; e.g., weight, area, velocity. Also, when the terminal time t_f allows be free (for optimization), the free time problem is obtained, otherwise the fixed time problem is given. In addition, the initial conditions are separated from the functional constraints in Eq. (4) for practical considerations. The differential equations for the system in Eq. (2) are written in general first-order form.

3. DISCRETIZATION MODEL

The general continuous time optimal control problem can be transferred into NLP problem by introducing a parametric control and state model. To treat this continuous model as the discrete model, it must represent the infinite-dimensional control function **u** and state variable **x** by a finite set of parameters. At first discretization of the control function is considered. The entire time interval $[t_0, t_f]$ is subdivided into N general unequal time intervals and the grid is designated as

$$t_0, t_1, t_2, \ldots, t_{N-1}, t_N = t_f \qquad (6)$$

The time intervals between the grid points are defined in a vector form as

$$\mathbf{T} = [\, T_1, T_2, \ldots, T_N \,]^T \qquad (7)$$

Where $T_i = t_i - t_{i-1}$ and $\sum_{i=1}^{N} T_i = t_f - t_0$ which generate the parameter set

$$\mathbf{U} = [u_1(t_0), \ldots, u_1(t_N), u_2(t_0), \ldots, u_m(t_0), \ldots, u_m(t_N)]^T \qquad (8)$$
$$= [U_1, \ldots, U_{N+1}, U_{N+2}, \ldots, U_{mN+1}, \ldots U_{mN+m}]^T \qquad (9)$$

This can be treated as design variable vector. This results in a total number of k+N+mN+m design variables as

$$\mathbf{P} = [b_1, \ldots, b_k, T_1, \ldots, T_N, \ldots, U_1, \ldots, U_{mN+m}]^T \qquad (10)$$

The coefficients of the interpolation function for the control function **u** may be considered as design variables instead of Ti and U_i in Eq. (10). For example, if time grid is not considered as design variable, the interpolation function based on third-order polynomial, $u_1(t) = U_1 + U_2 \times t + U_3 \times t^2 + U_4 \times t^3$ can be used to represent the first component of the control forces in **u**; and U_1, U_2, U_3, U_4 can be treated as design variables. Besides, in some control problems, the initial conditions on the state variables are unknown. For example, the number of the initial conditions is not enough for integrating or the initial conditions cannot be separated from the functional constraints in Eq. (4). The artificial variables are added as design variables to treat the problem that expand the size of design variable vector **P**. Two discretization techniques for the state variables in Eq. (2) can be used to evaluate the state trajectories. The first method discretizes both the control and the state variables and enforces the compliance of the state equations by introducing them as constraint equations in the nonlinear programming (NLP) problem. This method approximates both the states and the controls in discrete intervals and uses them as variables of the resulting NLP problem. This method is known as the Direct Collocation and Nonlinear Programming Problem (DCNLP) [8]. The next method is a straightforward approach that the control variables u(t) are approximated by some interpolation functions **I**(t) in each time interval, where **I**(t): $[t_0, t_f] \in R^m$. The approximate trajectories of state variables **x** are generated by solving the initial value problem defined in Eqs. (2) and (3). This method is commonly called the direct shooting method [9]. With the direct shooting method, the problem is called an admissible optimal problem [10].

In the DCNLP, the states and controls are discretized and used as the design variables of the NLP problem. In practical problems, a fine interval of discretization is necessary for an accurate solution. On the other hand, a huge amount of state and control variables are needed to describe the dynamic behavior in a large-scale problem, e.g. structure control problems. In those situations, the number of design variables of the NLP problem will be increase rapidly. The influence of the number of design variables on the performance of the NLP problem is self-evident. Furthermore, many well-developed subroutines exist for numerical integration that is needed in the admissible optimal control problems. Therefore, the admissible optimal control problem is more efficient and easy implementation than DCNLP and it has been adopted in this paper.

4. ADMISSIBLE OPTIMAL CONTROL FORMULATION

Consider now a typical functional that may be the performance index or a functional constraint:

$$J = \psi[\mathbf{b}, \mathbf{x}(t_f), t_f] + \int_{t_0}^{t_f} F[\mathbf{b}, \mathbf{u}(t), \mathbf{x}(t), t]dt \tag{11}$$

where the first term involves only design variables and the state of the system at terminal time and the second term contains mean behavior over entire interval of motion. As noted earlier, the control functions **u** are treated as a subset of the vector P. The terminal time t_f can be treated as one of the design variables in **T**, for example, $t_f = \sum_{i=1}^{N} T_i + t_0$. The dependence on design variable in Eq. (11) arises both explicitly and implicitly through the control and state variables. The admissible control function is represented in the form **u**(t) = **I**(**U**, **T**, t) and the state variable is written the form **x**(**b**, **U**, **T**, t) to emphasize that it is a function of design variables **P**. Therefore, equation (11) can be rewritten as follows:

$$\begin{aligned} J &= \psi[\mathbf{b}, x(\mathbf{b}, \mathbf{U}, \mathbf{T}, t_f), t_f] \\ &\quad + \int_{t_0}^{t_f} F[\mathbf{b}, I(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt \end{aligned} \tag{12}$$

The state equation in Eq. (2) is transformed as

$$\dot{\mathbf{x}} = \mathbf{f}[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), x(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt, \quad t_0 \le t \le t_f \tag{13}$$

and a typical dynamic constraint function in Eq. (5) as

$$\phi[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t] \tag{14}$$

Now, it can express the admissible optimal control problem in an NLP formulation as follows:
Find the design variables $\mathbf{P} = [\ \mathbf{b}^T, \mathbf{T}^T, \mathbf{U}^T]^T$ to minimize a performance index

$$\begin{aligned} J_0 &= \psi_0[\mathbf{b}, \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t_f), t_f] \\ &\quad + \int_{t_0}^{t_f} F_0[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt \end{aligned} \tag{15}$$

subject to state equations

$$\dot{\mathbf{x}} = \mathbf{f}[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), x(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt, \quad t_0 \le t \le t_f \tag{16}$$

with initial conditions

$$\mathbf{x}(t_0) = \mathbf{h}(\mathbf{b}) \tag{17}$$

functional constraints as

$$\begin{aligned} J_i &= \psi_i[\mathbf{b}, \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t_f), t_f] \\ &\quad + \int_{t_0}^{t_f} F_i[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t]dt \begin{Bmatrix} = 0; i = 1, ..., r' \\ \le 0; i = r'+1, ..., r \end{Bmatrix} \end{aligned} \tag{18}$$

and dynamic constraints as

$$\phi_j[\mathbf{b}, \mathbf{I}(\mathbf{U}, \mathbf{T}, t), \mathbf{x}(\mathbf{b}, \mathbf{U}, \mathbf{T}, t), t] \begin{Bmatrix} = 0; j = 1, ..., q' \\ \le 0; j = q'+1, ..., q \end{Bmatrix} \tag{19}$$

With admissible optimal control, some good first order differential equation methods having variable step size and error control are available to integrate the state equations in Eqs. (16) and (17), e.g. Adams method and Runge-Kutta-Fehlberg method [11]. These solvers can give accurate results with user desired error control. The state trajectories are internally approximated using interpolation functions in the differential equation solvers. Values of the state and control variables between the grid points can be also obtained with different kinds of interpolation schemes.

5. NUMERICAL SCHEMES

Ordinary Differential equation (ODE) solver: With the admissible optimal control formulation, the state equations (13) and design sensitivity equations need to be integrated by the ODE solver. In this paper, DDERKF, DDEABM and DDEBDF that developed by Sandia Laboratory are selected to integrate state or design sensitivity equations [12]. DDEBDF is based on the variable-order (1-5) backward-differention formula. DDERKF is a fifth-order Runge-Kutta code and DDEABM is a variable-order (1-12) Adams-Bashforth code. Those equation solvers use variable-step-size algorithms and have good error control.

Numerical Integration Scheme: Two integration schemes, Simpson's rule and Gaussian quadrature formula, are adopted in this paper. Those schemes are used to integrate the integral part of the functional constraints.

Interpolation Schemes: For the admissible optimal control formulation, interpolation schemes are needed at several places. The zero-order, first-order and piecewise cubic-spline interpolation functions can be used in this paper.

Optimization Solver: In the admissible optimal control formulation, the optimal control problem is converted into a NLP program and then a standard optimization solver can solve the NLP problem numerically. A great deal of attention has been paid to the NLP problems by using the Sequential Quadratic Programming (SQP) method [7, 13]. In this paper, an optimization solver-MOST [14] based on the SQP method is chosen to solve the NLP problem.

Design sensitivity analysis: Base on SQP method used in numerical methods of optimization, one must perform design sensitivity analysis; i.e., calculate design gradients of problem functions. In general, two approaches are applied to calculate these gradients. A first approach is to use a finite-difference approximation. The other method is to differentiate implicit functions analytically. There are two methods for calculating analytical derivatives of a constraint with respect to the design variables **P**: the direct differentiation method (DDM) and the adjoint variable method (AVM) [15].

6. SYSTEMATIC PROCEDURE FOR SOLVING THE OCP PROBLEMS

In this paper, the OCP is converted into NLP problem with admissible optimal control formulation and then the optimizer based on SQP method is used to solve the NLP problem numerically. Now, those procedures are implemented in the OCP solver and the complicated details of the transformation and programming will be completed in the OCP solver automatically. The optimal control and state trajectories will be obtained and recorded in the output files. Therefore, engineers can follow an efficient and systematic procedure to solve various optimal control problems. The procedure for solving the OCP with the OCP solver is described as follows.

1) Define the OCP problem following the formulation defined in Section 2.
2) According to the formulation, prepare the parameter files and user-defined subroutines.
3) Compile the user's subroutines and link with OCP solver.
4) Execute the OCP solver and get the optimal results.

7. NUMERICAL EXAMPLES

Two various types of optimal control problems have been used in literatures as test problems to evaluate the performance of the proposed method. Both the acceptable violation of constraints for feasible designs and acceptable tolerance for the convergence parameter are 1.0E-3 in SQP method.

Case 1: The van der Pol oscillator problem

The van der Pol oscillator problem was given and solved by Bullock and Franklin [16] with second variation method. Also, Jaddu and Shimemura [7] used this problem to verify their computational method. In this work, it is also used to evaluate the performance and capabilities of the OCP solver. The van der Pol oscillator problem can be formulated as follows:

Minimize

$$J = \frac{1}{2}\int_0^5 (x_1^2 + x_2^2 + u^2)\,dt \tag{20}$$

Subject to

$$\dot{x}_1 = x_2 \tag{21}$$

$$\dot{x}_2 = -x_1 + (1 - x_1^2)x_2 + u \tag{22}$$

with initial states $\mathbf{x}^T(0) = [1, 0]^T$.

Table 1 shows the performance comparison of various numerical schemes for the OCP solver. The finite-difference method (DSA=FDM) and direct differentiation method (DDM) for sensitivity analysis are selected to evaluate their performance. The Simpson's rule (INTG=SIMPSN) and Gaussian quadrature formula (INTG=GAUSS) are used to carry out the numerical integration over the time interval. DDERKF and DDEABM with an option to switch to DDEBDF are selected for solving first-order differential equations. The number of time grid points for the control function is selected as 21. The numerical results include the number of iterations (NIT) for the SQP method, optimal value of performance index (J*), CPU time for the entire iterative process. The numerical results for all example problems were obtained on an Intel Celeron 1.2 GHz computer with 384 MB of RAM memory.

This optimal value of performance index for this problem was found by Bullock and Franklin [16] to be J* = 1.433508 using second variation method. Also Jaddu and Shimemura [7] found the optimal value J* = 1.433487 using a Chebyshev series of ninth order to approximate $x_1(t)$. In this case, the numerical results for various numerical schemes of the OCP solver are listed in Table 1 and the optimal control and state trajectories is shown in Figs. 1. As the numerical results show, the performance with all of the numerical schemes of the OCP solver is quite accurate. Owing to the efficiency the finite-difference approach for sensitivity analysis and Gaussian quadrature formula for the numerical integration are selected to solving the following problem.

Case 2: Fourth-Order Systems: A Flexible Mechanism

A flexible mechanism was proposed and solved by Wu [17]. The OCP formulation of this problem can be described as follows.

Minimize

$$J = t_f \tag{23}$$

Subject to

$$\begin{aligned} \dot{x}_1(t) &= x_2 \\ \dot{x}_2 &= -\frac{k}{m_1}(x_1 - x_3) + \frac{u}{m_1} \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= \frac{k}{m_2}(x_1 - x_3) \end{aligned} \tag{24}$$

Control constraints

$$\phi_1 = |u(t)| \le M \tag{25}$$

with boundary conditions $\mathbf{x}^T(0) = [0,\ 0,\ 0,\ 0]^T$ and $\mathbf{x}^T(t_f) = [1,\ 0,\ 1,\ 0]^T$.

With admissible control formulation, the control variables are converted into the design variables and the control constraints are treated as the dynamic constraint. In this work, the system is solved by the OCP solver with following parameters: k = 1N-m-rad^{-1}, $m_1 = m_2 =$ 1kg-m^2, and M = 1N-m. The numbers of time grid points for the control function (NGP) are selected as 5, 11 and 51 to study the effect of coarser or finer mesh. Two initial guesses, u (t) = 0.0 and u (t) = 1.0, for the control function with three piecewise interpolation schemes – zero order, first order, and cubic spline are used in this problem. The hybrid method that combines the DDM and AVM for design sensitivity analysis is used to calculate the design sensitivity coefficients.

The optimal solution for this problem is given in Table 2 and the trajectories of state variables are shown in Fig.2. All the 18 test runs are successfully solved with the proposed method, but the runs with small number of control grid points (NGP) give higher optimum values and less CPU time. The terminal time, t_f, and the trajectories obtained in this work are agreed with the results, $t_f \cong 4.3$, obtained by Wu [17]. The numerical results also show that the proposed method has the capability to deal with the high order time-optimal control problem.

8. CONCLUSIONS

The admissible control formulation combines with SQP method used to solve the optimal control problem is presented. The theoretical basis of nonlinear programming approach for the optimal control problem is also described. An optimal control problem solver, the OCP solver, based on the nonlinear mathematical programming techniques and integrated with many well-developed numerical routines is implemented. A systematic procedure for solving optimal control problem is also proposed. Two various types of optimal control problems are used to evaluate the capability and accuracy of the OCP solver. The results show that the OCP solver can facilitate engineers to solve the optimal control problems with a systematic and efficient procedure.

9. ACKNOWLEDGEMENTS

The research reported in this paper was supported under a project sponsored by the National Science Council Grant, Taiwan, R.O.C., NSC90-2212-E009-039, is greatly appreciated.

REFERENCES

[1] Kim, T.H. and Ha, I.J., Time-Optimal Control of a Single-DOF Mechanical System with Friction, *IEEE Trans. Automat. Contr., 46(5)*, 2001, 751-755.

[2] Tsiotras, P., On the Optimal Braking of Wheeled Vehicles, *Proc. of the American Control Conference*, Chicago, 2000, 569-573.

[3] Jalili, N. and Esmailzadeh, E., Optimum Active Vehicle Suspensions with Actuator Time Delay, *Transcations of the ASME. Journal of Dynamic Systems, Measurement and Control, 123*, 2001, 54-61.

[4] Samaras, N.S. and Simaan, M.A., Optimized Trajectory Tracking Control of Multistage Dynamic Metal-Cooling Processes, *IEEE Tran. Ind. Applicat., 27(3)*, 2001, 920-927.

[5] Lin, S.Y., Complete Decomposition Algorithm for Nonconvex Separable Optimization Problems and Applications, *Automatica, 28*, 1992, 249-254.

[6] Pytlak, R., *Numerical Methods for Optimal Control Problems with State Constraints* (Lecture Notes in Mathematics 1707, Springer-Verlag, 1999).

[7] Jaddu, H. and Shimemura, E., Computational Method Based on State Parameterization for Solving Constrained Nonlinear Optimal Control Problems, *International Journal of Systems Science, 30(3)*, 1999, 275-282.

[8] Hu, G.S., ONG, C.J. and Teo, C.L., An Enhanced Transcribing Scheme for The Numerical Solution of A Class of Optimal Control Problems, *Engineering Optimization, 34(2)*, 2002, 155-173.

[9] Bock, H.G. and Plitt, K.J., A Multiple Shooting Algorithm for Direct Solution of Optimal Control Problems, *Proceedings of the 9th IFAC World Congress, Budapest*, 1984.

[10] Sage, A.P. and White, C.C. III, *Optimum systems Control, 2nd Edition* (Prentic Hall, 1977).

[11] Press, W., Teukolsky, S.A., Vetterling, W.T., and Flannery, B., *Numerical Recipes in C: The Art of Scientific Computing*, 2nd Edition (Cambridge, New York, 1992).

[12] Shampine, L.F. and Watts, H.A., DEPAC-Design of User Oriented Package of ODE Solvers, SAND 79-2347, Sandia Laboratory, 1979.

[13] Tseng, C.H., *Optimal Design for Dynamics and Control Using a Sequential Quadratic Programming Algorithm* (PhD dissertation, Department of Mechanical Engineering, Iowa University, 1987).

[14] Tseng, C. H., Liao, W. C. and Yang, T. C., MOST 1.1 User's Manual. Technical Report No. AODL-93-01, Department of Mechanical Engineering, National Chiao-Tung Univ., Taiwan, R.O.C., 1993.

[15] Hsieh, C.C. and Arora, J.S., Design Sensitivity Analysis and Optimization of Dynamic Response, *Computer Methods in Applied Mechanics and Engineering, 43*, 1984, 195-219.

[16] Bullock, T.E. and Franklin, G.F., A Second-order Feedback Method for Optimal Control computations, *IEEE Transactions on Automatic Control, 12*, 1967,666-673.

[17] Wu, S.T., Time-Optimal Control and High-Gain Linear State Feedback, *International Journal of Control, 72(9)*, 1999, 764-772.

[18] Pontryagin, L.S., Boltyanskii, V.G., Gamkrelidze, R.V. and Mischenko, E.F., *The Mathematical Theory of Optimal Processes* (Wiley, 1962).

Table 1 Performance comparison of various numerical schemes for the oscillator problem.

DSA	INTG	DIFF	INTP	NIT	J*	Conv.Par.	CPU
DDM	SIMPSN	DDERKF	Zero	11	1.4530	4.9577e-4	19.639
			First	21	1.4333	7.6480e-4	6.069
			Cubic	20	1.4334	5.4521e-4	18.146
		DDEABM	Zero	11	1.4530	4.9577e-4	18.455
			First	21	1.4333	7.6480e-4	19.147
			Cubic	20	1.4334	5.4521e-4	15.202
	GAUSS	DDERKF	Zero	12	1.4422	9.7245e-4	20.499
			First	21	1.4328	8.4954e-4	4.665
			Cubic	20	1.4334	5.4616e-4	14.392
		DDEABM	Zero	12	1.4422	9.7245e-4	19.078
			First	21	1.4328	8.4954e-4	17.686
			Cubic	20	1.4334	5.4616e-4	11.776
FDM	SIMPSN	DDERKF	Zero	13	1.4530	3.6313e-4	3.615
			First	21	1.4333	7.6261e-4	1.382
			Cubic	20	1.4334	5.4124e-4	1.542
		DDEABM	Zero	13	1.4530	3.6306e-4	3.615
			First	21	1.4333	7.5959e-4	3.244
			Cubic	20	1.4334	5.4139e-4	2.333
	GAUSS	DDERKF	Zero	14	1.4422	5.3821e-4	3.445
			First	21	1.4328	8.4715e-4	0.752
			Cubic	20	1.4334	5.4218e-4	0.871
		DDEABM	Zero	14	1.4422	5.3905e-4	3.555
			First	21	1.4328	8.3944e-4	2.754
			Cubic	20	1.4334	5.4269e-4	1.603
Bullock and Franklin (1967)					J* = 1.433508		
Jaddu and Shimemura (1999)					J* = 1.433487		

Table 2 Optimal results for the fourth-order system.

$u(t_0)$	NGP	INTP	NIT	J*	Conv.Par.	CPU
0.0	5	Zero	5	4.33196	1.04E-05	0.131
		First	5	4.86764	5.69E-06	0.07
		Cubic	5	4.90565	4.67E-07	0.06
	11	Zero	15	4.30699	5.71E-09	1.382
		First	7	4.28066	5.65E-06	0.35
		Cubic	10	4.30041	1.52E-08	0.34
	51	Zero	50	4.26239	1.38E-07	43.803
		First	44	4.22087	3.50E-07	18.596
		Cubic	40	4.22187	1.10E-08	12.659
1.0	5	Zero	6	4.33197	6.16E-07	0.12
		First	7	4.86765	2.25E-07	0.091
		Cubic	9	4.90560	3.72E-05	0.12
	11	Zero	7	4.36249	3.51E-08	0.651
		First	8	4.28064	1.09E-05	0.43
		Cubic	10	4.30041	3.50E-07	0.39
	51	Zero	49	4.26229	7.21E-08	42.872
		First	42	4.22087	2.20E-06	17.315
		Cubic	38	4.22187	2.88E-06	11.847

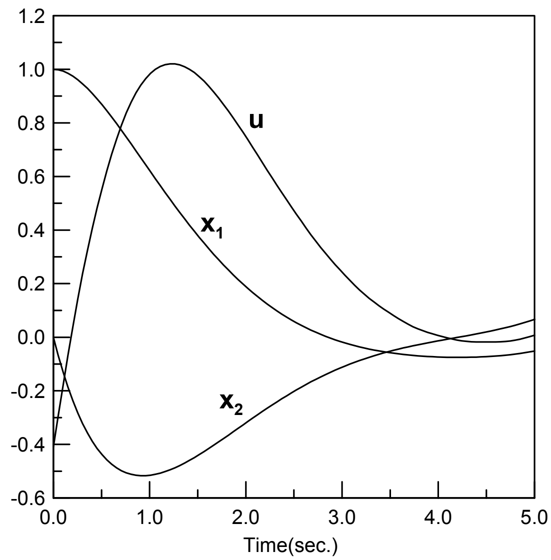
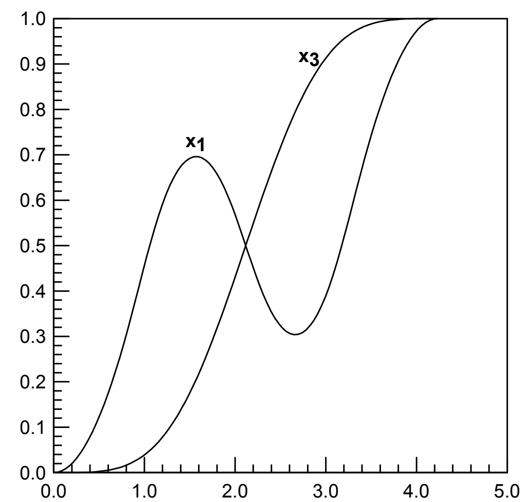
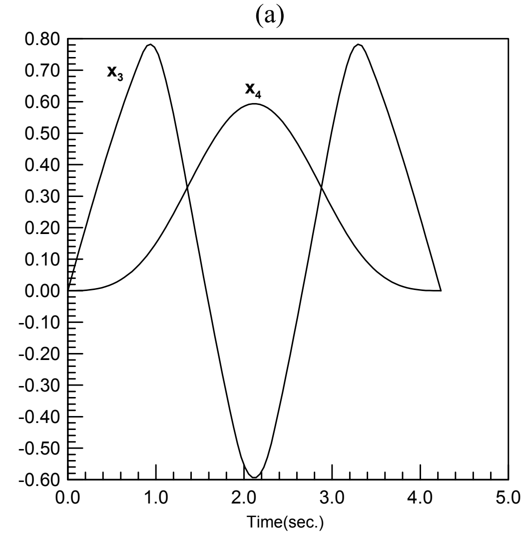


Fig. 1 State trajectories for the van der Pol oscillator problem.



(a)



(b)

Fig. 2 State trajectories for the fourth-order system.