INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Fuzzy Control for an Under-actuated Robotic Manipulator: Pendubot

Xiao Qing Ma

A Thesis

in

The Department

of

Mechanical Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Applied Science at

Concordia University

Montreal, Quebec, Canada

August 2001

Abstract

Fuzzy Control of an Under-actuated Robotic Manipulator: Pendubot

Xiao Qing Ma

Control of under-actuated mechanical systems (robots) represents an important class of control problem. This thesis studies several related control problems associated with an under-actuated robot, Pendubot, from the point view of fuzzy logic control. To swing up the Pendubot from a rest position to the upright configuration, a fuzzy algorithm is proposed from non-complete sets of linguistic rules that link some mechanism states to the sign of a single control action. Therein, a simplified Tsukamoto's reasoning method and quasi-linear-mean aggregating operators are used to derive and analyze the controller input-out mappings. In order to balance the Pendubot at the unstable upright top configuration after swinging up, another simple fuzzy controller is derived according to its joint states. This combining fuzzy algorithm for swinging-up and balancing is successfully applied to the Pendubot. This thesis also investigates the case that the Pendubot tracks a desired signal and a corresponding fuzzy scheme is proposed, which combines the linear regulator theory with the Takagi-Sugeno fuzzy methodology. The stability and stability conditions for this fuzzy scheme are analyzed. Numerical simulations for all the above controllers are carried out to validate the theoretical analysis by using SIMULINK. Finally, the hardware experiments in the Pendubot have successfully been conducted in the Robotics and Mechatronics Laboratory.

Acknowledgements

The author dedicates this thesis to the thesis supervisor Associate Professor Dr. Chun-Yi Su, for his invaluable guidance and consistent encouragement through out the course of this research and looks forward for his guidance in all future endeavors.

The author would like to express her appreciation to Mr. Z. Cai, whose enthusiasm to build and set up the whole system and help me perform the experiments played a significant role in the development of this project.

The author would like to express her sincere thanks to Mr. Z. L. Liu for the useful discussions with the author.

I would like to thank my entire family for their love and support throughout my graduate school days. They and especially my husband, KangZe Zheng, gave me the encouragement to complete this work.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

In recent years the control of under-actuated mechanical systems has attracted growing attention and is a topic of great interest [2][3][11][15][10]. As yet, there is no comprehensive theory for underactuated robot manipulators, i.e., manipulators having fewer actuators than degrees of freedom. Examples of such systems are illustrated, for example in [2][16][7][1][5][6]. Frequently cited applications include saving weight and energy by using fewer actuators and gaining fault tolerance to actuator failure.

The challenge of solving control problems associated with this class of systems will stimulate new results in robot control theory in the years to come. Interest in studying the under-actuated mechanical systems is also motivated by their role as a class of strongly nonlinear systems where complex internal dynamics, nonholonomic behavior, and lack of feedback linearizability are often exhibited, for which traditional nonlinear control methods are insufficient and new approaches must be developed.

Though dynamics of the under-actuated mechanical systems is well understood, the difficulty of the control problem for under-actuated mechanism is obviously due to the

reduced dimension of the input space. The literature on the control of under-actuated systems is mainly recent [2] [4] [5] [13] [12], and the discussion mainly focuses on two-degree-of-freedom examples [3] [15] [40] [59]. Earlier work that deals with control of under-actuated robotic systems is described in [55] [56]. Under-actuated mechanical systems have also been investigated from nonholonimc constraint point of view [13] [17], where, for instance, Oriolo and Nakamura [57] and Wichlund [58] established the conditions for partial integrability of second-order nonholonomic constraints and discussed control problems.

While some interesting techniques and results have been presented in the mentioned above publications, the control of such systems still remains an open problem. For example, most of the control schemes mentioned above either failed to provide a thorough analysis of the overall system stability or assumed that gravitation forces didn't act on the passive joints. Furthermore, the precise knowledge of dynamic model is generally required.

Fuzzy logic control techniques were originally advocated by Zadeh and Mamdani as a means of both collecting human knowledge and experience and dealing with uncertainties in the control process. It has become a very popular tool in control engineering [8][7][10][11]. Fuzzy control systems have the advantages that no formal mathematical models are needed and the system uncertainties can be coped with. Therefore, aiming at the mentioned above control problems for such under-actuated mechanical nonlinear systems, this thesis will investigate several related control

problems, associated with an under-actuated robot---Pendubot, from the point view of the fuzzy logic control.

The Pendubot (Pendulum robot) is a benchmark system for under-actuated robot manipulators, consisting of a double pendulum with an actuator at only the second joint. Using the Pendubot, one can mainly investigate the set-point regulation, including swinging up and balancing, and trajectory tracking.

The problem of swinging the Pendubot from the open loop stable configuration $q_1 = -\pi/2$, $q_2 = 0$ to any of the inverted equilibria is an interesting problem because of the strong nonlinearity and dynamic coupling between the links. Many different control algorithms could have been used to perform the swing up. In the classic control field, Spong [2] and Block [1] applied small partial feedback linearization to swing up the two links from hanging straight down to standing straight up. Yoshida [6] used energy-based methods to complete the swing-up control of an inverted pendulum. In addition, some researchers proposed the robust adaptive control algorithm or hybrid algorithm to control the two-link under-actuated robots [3] [5]. In the fuzzy control field, Sanchez [15] and Vidolov [10] designed a MIMO fuzzy PD controller to swing up an underactuated robot. Sousa and Madrid [11] proposed a control strategy based on genetic algorithm and Takagi-Sugeno fuzzy methodology to control the Pendubot. However, the genetic algorithm is not easy to implement because it depends on the number of rules. Some other fuzzy control schemes are just implemented in the simulation system, and they didn't give the experimental results.

Once both links are swung up, the problem of balancing the Pendubot about an open loop unstable equilibrium should be investigated. The Pendubot possesses infinitely many configurations at which it can be balanced. Spong [2] and Block [1] proposed a balancing controller, using linear quadratic optimal control theory to balance the Pendubot at top or mid configurations. Furthermore, the robust adaptive control algorithm or hybrid algorithm has also been proposed, but they are not fully general.

Besides the above-mentioned problem associated with the Pendubot, trajectory tracking for nonlinear systems also can be achieved in the Pendubot. Berkemeier [4] derived a surprising set of exact trajectories of the nonlinear equations of motion, which involve inverted periodic motions. Moreover, Begovich and Sanchez [14] combined linear regulator theory with the Takagi-Sugeno fuzzy methodology a new algorithm to get a novel approach to achieve trajectory tracking for the Pendubot. However, they didn't analyze the stability of the whole control system. In this thesis, nonlinear system trajectory tracking has been analyzed and implemented in real time in the Pendubot.

1.2 Objectives

The objective of this research is to develop a digital controller based on fuzzy logical, approximate reasoning, and quasi-linear-mean aggregating operators, to swing up and balance the under-actuated Robot — the Pendubot. Nonlinear system trajectory tracking will also be analyzed, based on the linear regulator theory and the Takagi-Sugeno (T-S) fuzzy methodology. Important issues for the under-actuated robotic controller design are

to develop the fuzzy logic algorithms and implement them. The motivation is to make an effort towards applying fuzzy controlling methods to the actual robot. I will build a completed and feasible simulation model and write a digital controller program.

The Pendubot is installed in the Robotics and Mechatronics Laboratory, led by Dr.Su. It is a typical under-actuated pendulum robot which is an electro-mechanical system consisting of two rigid links interconnected by revolute joints.

1.3 Thesis Organization

Chapter 2 mainly consists of three main parts. The first part is a brief introduction to robotic control. The second part is a description of fuzzy logic theory and fuzzy control. The third part is a description of the Pendubot.

Chapter 3 discusses the dynamics of the Pendubot, introduces the equilibrium manifold and identification of the link inertia parameters, and discusses the controllability of this system. In this chapter, I provide the mathematical model that can be used to derive various controllers and to simulate the response of the system. It serves as the base for the whole research.

Chapter 4 derives a two-rule-based fuzzy control scheme from a simplified Tsukamoto's reasoning method and quasi-linear-mean aggregating operators to swing-up the Pendubot from its rest position. The fuzzy scheme is derived from non-complete sets

of linguistic rules that link some mechanism states to the sign of a single control action. At the end of this chapter, I analyze the control law and give the principle of control parameters' tuning.

Chapter 5 displays a fuzzy control algorithm for balancing the Pendubot at unstable open-loop equilibrium states. This scheme is also obtained based on the fuzzy rule base, but contrary with the swing-up's rule base, because their control goals are different.

Chapter 6 analyzes the stability of a fuzzy scheme for trajectory tracking of the nonlinear system: Pendubot. There are two steps: the first one is the stability analysis of the fuzzy control system, consisting of a nonlinear plant and a fuzzy controller; the second one is to give the condition of applying linear regulation theory in the nonlinear system that consists of each local linear system by using the fuzzy methodology.

Chapter 7 displays the switching algorithm both in the simulation and in the experiment since the model of the Pendubot in the simulation is different from the actual model. These differences display not only in the parameter adjusting but also in the controller combining. This chapter will point out all these details.

Chapter 8 is the simulation of the whole control process using the SIMULINK. I start from the Pendubot model founding, and then build the swing-up controller and balancing controller respectively, finally connect them with the switching controller. In this chapter,

I also explain the tuning of controllers and give the simulation results for both continuous-time and discrete-time systems.

Chapter 9 presents the implemental method and results. I design a digital controller according to the fuzzy control scheme (given in the above chapters) using Microsoft C 7.0 language. I present the design clue through flow chart and give the implemental results using digital photographs and the response of joints.

Summary, conclusions, the recommendations for future work, and reference are presented in the next parts. Finally, Appendix A gives the linearization equations used in Chapter 6.

1.4 Contributions

The approach in this thesis is partly similar to the work of [1][16][14], but there are significant differences as well, which are exactly my contributions.

1) A simplified Tsukamoto's reasoning method and quasi-linear-mean aggregating operators are used to derive and analyze the swing-up control input-output mappings. I apply this fuzzy control scheme to swing up the Pendubot from the rest position $(q_1 = -\pi/2,\ q_2 = 0)$ to the unstable top position $(q_1 = \pi/2,\ q_2 = 0)$.

2) According to the Pendubot's joints states and fuzzy control rule, I design another fuzzy controller to balance the Pendubot at the top position. Through a logical switching algorithm, I combine the swing-up controller with this balancing controller.

3) Based on the developed T-S fuzzy trajectory tracking scheme on the balancing Pendubot, I give the stability analysis of this Takagi-Sugeno fuzzy scheme;

4) I have combined the fuzzy swing-up controller with the T-S fuzzy trajectory-tracking controller to generate a novel control scheme. I implemented it in the real time Pendubot;

5) All of the above are implemented in the Pendubot Model P-2 in our laboratory, and the simulations of 1) and 2) are achieved using SIMULINK.

Chapter 2

Basic Knowledge

2.1 Introduction of Robot

As many books related to robots are introduced, robots are basically positioning and handling devices [34]. A useful robot is one that is able to control its movement and the force it applies to its environment [35]. Presently different aspects of robotics research are carried out by experts in various fields that are mechanical manipulation, locomotion, computer control, and artificial intelligence. The major relevant fields are mechanics, control theory, and computer science. Omitting the accessorial parts, robot actually is the multi-link mechanics device. Thus, my research object is limited to this device and how to control it.

To control robot requires the knowledge of a mathematical model and of some sort of intelligence to act on the model. The mathematical model is obtained from the basic physical laws governing the robot's dynamics. Intelligence requires sensory capabilities and means for acting and reacting to the sensed variables. Therefore, there are two main steps for simple industrial robot: the first step is about the mechanics of mechanical manipulators and its dynamics; the second step is to apply the control theory to modify the actions and reactions of the robot to different stimuli. The particular controller used

will depend on the complexity of the mathematical model, the application at hand, the available resources, and a host of other criteria.

Step 1: Modeling

For servo-control design purposes, and to design better controllers, it is necessary to reveal the dynamic behavior of the robot via a mathematical model obtained from basic physical laws. I begin the development with the general Lagrange equations of motion [38]. Consider then Lagrange's equations for a conservative system as given by

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{q}}\right) - \frac{\delta L}{\delta q} = \tau \tag{2.1}$$

where q is an n-vector of generalized coordinates q_i, τ is an n-vector of generalized force τ_i, and the Lagrangian L is the difference between the kinetic and potential energies

$$L = K - P \tag{2.2}$$

It can then be shown [34] that the robot dynamics are given by

$$D(q)\ddot{q} + C(q,\dot{q}) + F(\dot{q}) + G(q) + \tau_d = \tau \tag{2.3}$$

where $D(q)$ is a symmetric, positive-definite inertia matrix, $C(q,\dot{q})$ is a vector containing the effects of the Coriolis and centripetal torque, $G(q)$ is an n-vector of gravity torques, $F(\dot{q})$ represents friction, and τ_d represents external disturbances.

Step 2: Control

A useful robot is one whose trajectory in its workspace may be specified, and the forces it exerts on its environment may be controlled. Looking back at equation (2.3), my control objectives will usually fall into one of the following categories:

1. *Motion Control.* Given a desired trajectory, specified by the vector time functions $q_d(t)$ and $\dot{q}_d(t)$, design and implement a controller whose output $\tau(t)$ will drive the actual trajectory $\{q(t), \dot{q}(t)\}$ to $\{q_d(t), \dot{q}_d(t)\}$ asymptotically. Note that the desired trajectory is usually specified in the task space and some preprocessing is required to obtain a desired joint space trajectory. The alternative would be to obtain the dynamics of the robot in the task space where the desired trajectory is specified. The fine motion control is then accomplished using precision movements of the end-effector.

2. *Force Control.* When the robot comes in contact with its environment, the contact forces and reactions of the robot need to be regulated. The force control requirements are specified by a desired force vector $f_d(t)$ in the task space. This force vector is the trajectory in the force space that the end-effector should follow, when a force controller is properly designed.

3. *Motion and Force Control.* In some cases, the robot is required to follow a desired motion trajectory while exerting a certain force on its environment. In this case, both previous control objectives are combined to design a suitable controller.

2.2 Review of Robotic Control

From equation (2.3), I know the dynamic model of a robot is described as a set of highly nonlinear and coupled differential equation. That means the robot system is a complicated nonlinear system. There are many control methods for such a nonlinear system. I will discuss them as following.

Firstly, a common approach is to linearize the dynamics and apply linear control theory. However, the operating region of a linearized design is limited, so the method of partial feedback linearization applying to the nonlinear systems is very limited [1][2][39]. Secondly, the developments in the theory of geometric nonlinear control provide powerful methods for controller design for a large class of nonlinear systems [29][40]. However, it requires the exact knowledge of the system, which is seldom met for robotic systems. Thirdly, to overcome this drawback, the adaptive control schemes have been used for the robotic systems [41][42]. The adaptive control approach is only applicable to the full-actuated robots, but for the under-actuated dynamic systems [13], it remains an open problem. Fourthly, with development of the fuzzy logic and fuzzy control, more and more researchers would like to apply this control method to deal with the uncertain nonlinear system. For example, Marcio developed a genetic algorithm to control the

under-actuated dynamic systems [11]; Edgar proposed a fuzzy PD scheme to swing-up the Pendubot [15]; Edgar and Ofelia also developed a nonlinear system trajectory tracking scheme [14]; Micheal [47] extended the classical Lyapunov synthesis method to the domain of computing with words to form a adaptive fuzzy controller.

From the above survey on the robotic control system, particularly for the under-actuated robot or dynamic systems, we can see that the fuzzy control approach may be more suitable for the complex uncertain nonlinear systems. Next, I will introduce the fuzzy logic and fuzzy control.

2.3 Fuzzy Logic Control and Reasoning

The ability to control a system in uncertainty or unknown environments is one of the most important characteristics of any intelligent control system. Fuzzy inference procedures are becoming, therefore, increasingly crucial to the process of managing uncertainty. Fuzzy sets theory provides a systematic framework for dealing with different types of uncertainty within a single conceptual framework [44].

A fuzzy controller works similar to conventional controllers [46]. It takes a set of input values, performs some calculations, and generates a set of output values. Figure 2.1 illustrates a typical fuzzy system. According to this figure, I will introduce the steps of designing a fuzzy controller.

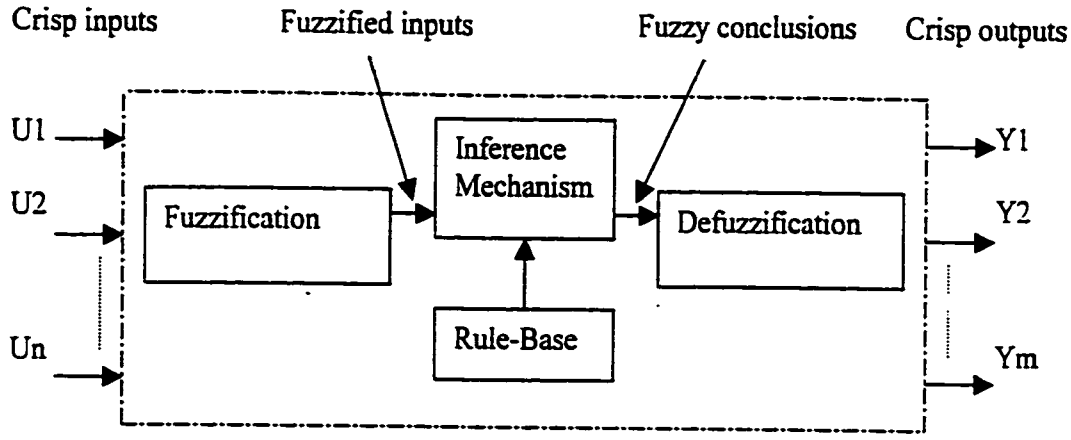


Figure 2.1 Fuzzy system

Step 1: Fuzzification

Fuzzification is a mapping from the observed input to the fuzzy sets defined in the corresponding universes of discourse [45]. Therefore, fuzzification is a mapping from $U \in R^n$ to the unit hypercube $[0, 1]^M$. Fuzzy sets were introduced by Zadeh to incorporate concepts such as vagueness into computer systems [48]. In conventional set theory an element either belongs to a set or does not, but in normal human decision making this sharp distinction does not normally exist.

A fuzzy set "A" is characterized by a membership function, whose elements x of X a real member $\mu_A(x)$ is in the interval [0, 1], and $\mu_A(x)$ representing the grade of membership of element x in the fuzzy set A. Here, the membership function can be chosen through many ways, including mathematic function, self-defined, or a simple reflection relation.

I should especially point out that, in our swing-up control algorithm, the fuzzy sets are given by:

$$XP(x) + XN(x) = 1 \quad \forall x \in X \tag{2.4}$$

$$XP(x) = (1 + \tanh(a_x x))/2 \quad a_x > 0 \tag{2.5}$$

where $\tanh(t) = \dfrac{e^{t} - e^{-t}}{e^{t} + e^{-t}}$, whose curve is illustrated in the Figure 2.2. Thus this definition is coincident with the definition of fuzzy set.

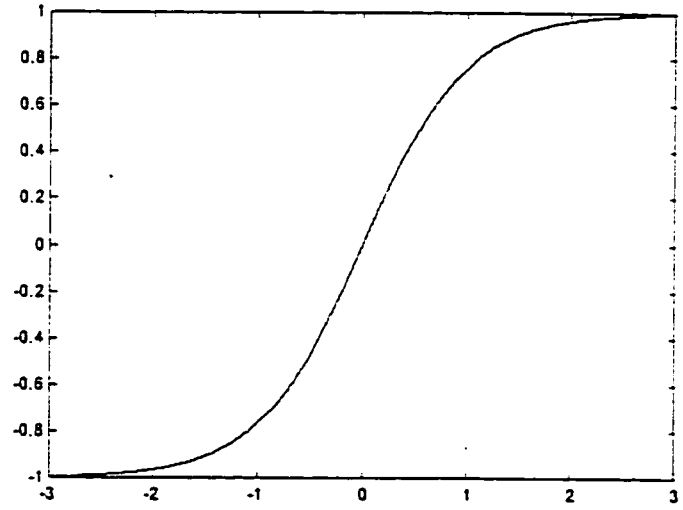


Figure 2.2 Function tanh(t)

Step 2: Rule-Base

A fuzzy rule is defined as a linguist logic relation between the input and output. It is simply expressed as:

IF ..... **THEN** .....

Rule base means all rules for the controlling system, which consists of each simple rule. The translation of this rule base system into a fuzzy relation is done by constructing the

fuzzy relation R_k for each rule r_k and combining these relations into a single fuzzy relation R. This combining of fuzzy rules into a fuzzy relation is called aggregation. Therefore, in order to express the rule base by mathematization, we should aggregate all rule bases (supposed number=q) by using mathematic method. In the process of aggregating, the weight $\lambda_i(z(t))$ $i = 1,2,...q$ of each firing rule is considered. This weight depends on the importance of that firing rule.

Step 3: Inference

Inference is also called fuzzy reasoning. Reasoning is the process by which we can conclude about new facts from already existing facts and causal links. Therefore, fuzzy reasoning refers to processes by which fuzzy conclusions about fuzzy facts are obtained from fuzzy rule base or fuzzy relations. In other words, inference process is a decision-making logic which determines fuzzy outputs corresponding to fuzzified inputs, with respect to the fuzzy rules. Thus, we should specify kinds of operators such as implication, conjunction, aggregation, and reasoning operator.

Step 4: Defuzzification

A conclusion is obtained from each rule for a certain observation, and then these conclusions are aggregated to obtain one conclusion. The resultant conclusion is a fuzzy set that can be defuzzified to obtain a crisp output. Hence, defuzzification produces a non-fuzzy output or a crisp output, using one of some usual methods [45] or particular

method [20]. The results of defuzzification is the final results of the whole fuzzy system that is composed of four main parts.

Fuzzy algorithm represents a powerful concept which can be used to provide fuzzy models and fuzzy controllers for control engineering. Such algorithms are essentially a collection of "IF" and "THEN" statements which use the input/output fuzzy sets, and relations defined, to deduce the control signal necessary to achieve the desired response.

2.4 Introduction of Digital Control

In a typical control system. several control loops are present. Each is designed to perform some particular task. The following procedure must take place in a digital control loop:

Step 1: Measure system output and compare with the desired value to give an error.

Step 2: Use the error, via a control law, to compute an actuating signal.

Step 3: Apply this corrective input to the system.

Step 4: Wait for the next sampling instant.

Step 5: Go to Step 1.

The loop structure which achieves this can take many forms, but two of the most common arrangements are shown in Figure 2.3. In my project, I utilized the second structure (b).

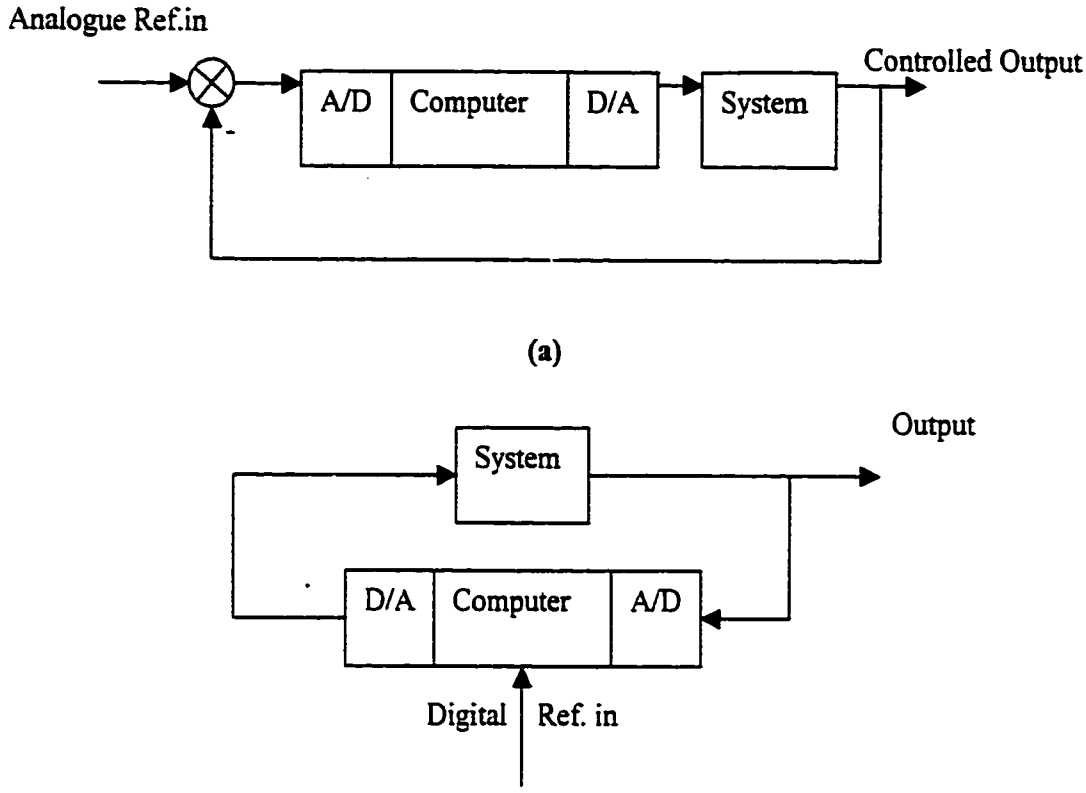


Figure 2.3 Structures for a digital control loop

2.5 Description of Pendubot

The Pendubot, short for **Pendu**lum Ro**bot**, is an electro-mechanical (or Mechatronic) system consisting of two rigid links interconnected by revolute joints. The first joint is actuated by a DC-motor and the second joint is unactuated. Thus the second link may be thought of as a simple pendulum whose motion can be controlled by actuation of the first link. The Pendubot is similar in spirit to the classical inverted pendulum on a cart. However, the nature of the dynamic coupling between the two links of the Pendubot results in some interesting properties not found in these other devices.

Therefore, the Pendubot is simple robot consisting of two rigid links. Because it also is an under-actuated mechanism, it is a nonlinear system. That means we can control it using various control methods suitable for nonlinear system. On the other hand, as a benchmark robot, now I just study its motion control, such as set-point regulation and trajectory tracking.

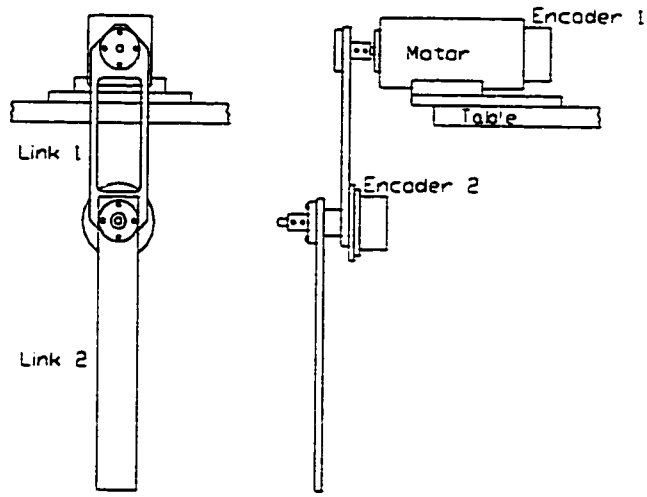


Figure 2.4 Front and side perspective drawings of the Pendubot.

The Pendubot is shown schematically in Figure 2.4. The actuated joint is driven by a high torque 90VDC permanent magnet motor without gear reduction. To give joint one direct drive control, we designed the Pendubot to hang off the side of a table coupling link one directly to the shaft of the motor. The mount and bearings of the motor are then the support for the entire system. Link one also includes the bearing housing which allows for the motion of joint two. Needle roller bearings riding on a ground shaft were used to construct this revolute joint for joint two. The shaft extends out both directions of the housing allowing coupling to both link two and an optical encoder mounted on link

one. This optical encoder produces the position feedback of link two. The design gives both links full 360° of motion. Link one, however, cannot continuously rotate due to the encoder cable for link two. Links have no constraint on continuous revolutions.

Two links are ¼ inch (0.635 cm) thick length aluminum. Link 1 is 6 inches (15.24 cm) long and is directly coupled to the shaft of a 90V permanent magnet DC motor mounted on the supporting base. Link 2 is 9 inches (22.86 cm) long and contains a coupling that attaches to the shaft of joint two.

The final component of the Pendubot's hardware is its controller. See Figure 2.5 for a pictorial description of the interface between the Pendubot and the controller. Two Dynamics Research Corporation 1250 counts/rev resolution optical encoders, one attached at the elbow joint and the other attached to the motor, and are used as the feedback mechanism for the joint angles. An Advanced Motion Control's 25A8 PWM servo amplifier is used to drive the motor. In the control algorithm this amplifier can be thought of as just a gain. In the case of the Pendubot we setup the amplifier in torque mode and adjusted it for a gain of 1V=1.2Amps.

In an attempt to simplify the controller for the Pendubot and minimize its cost, we implemented our control algorithm using only the microprocessor in our PC instead of purchasing an additional DSP card. We used a 486DX2/50 IBM compatible PC with a D/A card and an encoder interface card. The CIO-DAC-02 card from computer Boards, Inc. is used for digital to analog conversion and a Dynamics Research Corporation optical

encoder card is used to interface with the optical encoders. Controller timing is provided by a timer board of Computer Boards, Inc. that utilizes the 9513 timer chip. Using the standard software library routines supplied with the interface cards together with our own drivers we are able to program control algorithms directly in Microsoft C 7.0.

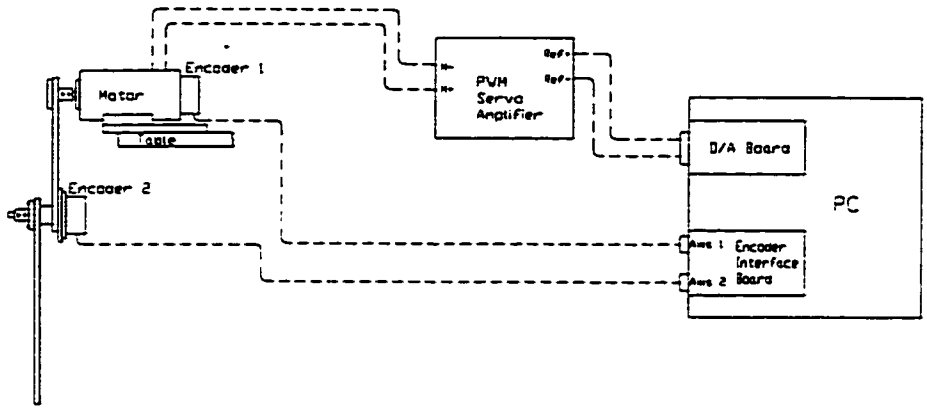


Figure 2.5 Pictorial of the Pendubot's interface with its controller

The range of sample period is from 0.001 second to 0.016 second. The sample period I choose is 0.005 second. This control design worked very well. As we know, the parameters of the controller are changing with different sample time, and the smaller sample period, the more similar as the continuous time controller. However, we should realize that we shouldn't only pursue the small sample period. A 5 ms sampling period was used in my controllers in order to allow computer to save response data while the controller was operating.

Chapter 3

Pendubot Model

In this chapter I discuss the dynamics of the Pendubot. The nonlinear equations of motion are described first. This provides a mathematical model that can be used to derive various controllers and to simulate the response. Since the device is a two links robot (with only one actuator), its dynamic equations can easily be derived using the so-called Euler-Lagrange equations and can be found in numerous robotics textbooks. In additional, the inertia parameters of the Pendubot have been provided by the Mechatronic Systems, Inc.

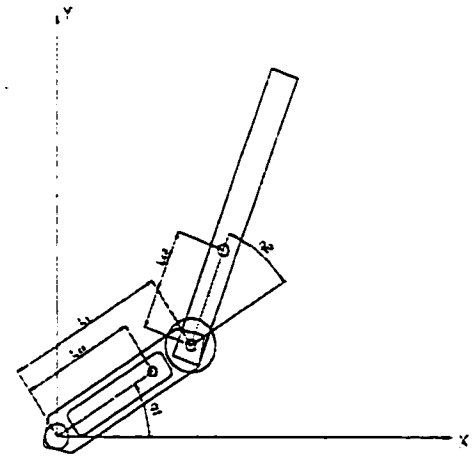


Figure 3.1 Coordinate description of the Pendubot. l_1 is the length of link one, l_{c1} and l_{c2} are the distances to the center of mass of the respective links and q_1 and q_2 are the joint angles of the respective links.

3.1 The Dynamics

Consider first a dynamic system defined on a configuration manifold Q. Let $(q,\dot{q}) = (q^1,...,q^n,\dot{q}^1,...,\dot{q}^n)$ denote local coordinates on the tangent bundle M=TQ. We refer to $q,\dot{q},$ *and* $\ddot{q}$ as the vectors of generalized coordinates, generalized velocities, and generalized acceleration, respectively. Let the system be under the action of m<n, m≥1, independent control forces and/or torques, i.e., there are fewer control inputs than degrees of freedom. Also let $\tau \in R^m$ denote the vector of control variables. Thus, the actuated degrees of freedom are m, and the unactuated degrees of freedom are $n-m$. In our device---Pendubot, there are just two links and one motor (see Figure 3.1). Derived from equation (2.3), we omit the friction $F(\dot{q})$, and the external disturbances τ_d, so the equations of motion for the Pendubot can be simply written in matrix:

$$D(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \begin{bmatrix} \tau \\ 0 \end{bmatrix} \tag{3.1}$$

where,

$D(q)$ is symmetric and positive definite, and

$$D(q) = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix}$$

$$d_{11} = m_1 l_{c1}^2 + m_2(l_1^2 + l_{c2}^2 + 2l_1 l_{c2}\cos q_2) + I_1 + I_2 \tag{3.2}$$

$$d_{12} = d_{21} = m_2(l_{c2}^2 + l_1 l_{c2}\cos q_2) + I_2$$

$$d_{22} = m_2 l_{c2}^2 + I_2$$

$C(q,\dot{q})$ has been chosen such that $\dot{D}-2C$ is skew symmetric.

$$C(q,\dot{q})=\begin{bmatrix} h\dot{q}_2 & h\dot{q}_2+h\dot{q}_1 \\ -h\dot{q}_1 & 0 \end{bmatrix}$$

$$h=-m_2 l_1 l_{c2}\sin q_2 \qquad (3.3)$$

Note the 0 in the vector on the right side of equation 3.1, indicating the absence of an actuator at the second joint. The symbol τ is the vector of torque applied to the links and **q** is the vector of joint angle positions.

Other,

$$g(q)=\begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix}$$

$$\phi_1=(m_1 l_{c1}+m_2 l_1)g\cos q_1+m_2 l_{c2} g\cos(q_1+q_2) \qquad (3.4)$$

$$\phi_2=m_2 g l_{c2}\cos(q_1+q_2)$$

m_1 : the total mass of link one.

l_1 : the length of link one (See Figure 3.1).

l_{c1} : the distance to the center of mass of link 1 (See Figure 3.1).

I_1 : the moment of inertia of link one about its centroid.

m_2 : the total mass of link two.

l_2 : the length of link two (See Figure 3.1).

l_{c2} : the distance to the center of mass of link 2 (See Figure 3.1).

I_2 : the moment of inertia of link one about its centroid.

g : the acceleration of gravity.

From the above equations it is observed that the seven dynamic parameters can be grouped into the following five parameters equations

$$\begin{aligned}
\theta_1 &= m_1 l_{c1}^2 + m_2 l_1^2 + I_1 \\
\theta_2 &= m_2 l_{c2}^2 + I_2 \\
\theta_3 &= m_2 l_1 l_{c2} \\
\theta_4 &= m_1 l_{c1} + m_2 l_1 \\
\theta_5 &= m_2 l_{c2}
\end{aligned} \tag{3.5}$$

Substituting these parameters (3.5) into the equations (3.2)--(3.4) leaves the following matrices

$$D(q) = \begin{bmatrix} \theta_1 + \theta_2 + 2\theta_3 \cos q_2 & \theta_2 + \theta_3 \cos q_2 \\ \theta_2 + \theta_3 \cos q_2 & \theta_2 \end{bmatrix}, \tag{3.6}$$

$$C(q,\dot{q}) = \begin{bmatrix} -\theta_3 \sin(q_2)\dot{q}_2 & -\theta_3 \sin(q_2)\dot{q}_2 - \theta_3 \sin(q_2)\dot{q}_1 \\ \theta_3 \sin(q_2)\dot{q}_1 & 0 \end{bmatrix}, \tag{3.7}$$

$$g(q) = \begin{bmatrix} \theta_4 g \cos q_1 + \theta_5 g \cos(q_1 + q_2) \\ \theta_5 g \cos(q_1 + q_2) \end{bmatrix}. \tag{3.8}$$

Finally, using the invertible property of the mass matrix $D(q)$, the state equations are given by

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = D(q)^{-1}\tau - D(q)^{-1}C(q,\dot{q})\dot{q} - D(q)^{-1}g(q) \tag{3.9}$$

$$x_1 = q_1, \quad x_2 = \dot{q}_1, \quad x_3 = q_2, \quad x_4 = \dot{q}_2$$

$$\dot{x}_1 = x_2 \tag{3.10}$$

$$\dot{x}_2 = \ddot{q}_1$$

$$\dot{x}_3 = x_4$$

$$\dot{x}_4 = \ddot{q}_2$$

3.2 Introduction of Identification

For a control design that neglects friction, these five parameters are all needed. There is no reason to go a step further and find the individual parameters since the control equations can be written with only the five parameters. Mechatronic Systems, Inc. performed on-line identification experiments using the Hamiltonian based energy equation method. The first step in this approach is to write the total energy as the sum of the kinetic energy and potential energy as

$$E = \frac{1}{2}\dot{q}^T D(q)\dot{q} + V(q) \tag{3.11}$$

where $D(q)$ is the inertia matrix defined in (3.1) and $V(q)$ represents the potential energy due to gravity. The total energy E is linear in the inertia parameters and so may be written as

$$E = W(q,\dot{q})\theta \tag{3.12}$$

where θ is the parametrization defined in (3.5). Using the well-known passivity or skew-symmetry property

$$\dot{E} = \dot{q}^T \tau \tag{3.13}$$

we have that, between any two times $t = T$ and $t = T + dT$

$$dE = E(T+dT) - E(T) = \int_{T}^{T+dT} \dot{q}^T(u)\tau(u)du = \{W(T+dT) - W(T)\}\theta \qquad (3.14)$$

Equation (3.14) can be used with a standard least squares algorithm to identify the parameter vector θ by applying an open loop signal $t \to \tau(t)$ and recording $\tau, q, \dot{q}$ over a given time interval. Carrying out this identification procedure using a step input to excite the Pendubot resulted in the following parameter values,

$$\theta_1 = 0.0308 Vs^2$$

$$\theta_2 = 0.0106 Vs^2$$

$$\theta_3 = 0.0095 Vs^2$$

$$\theta_4 = 0.2087 Vs^2 / m$$

$$\theta_5 = 0.0630 Vs^2 / m$$

3.3 The Equilibrium Manifold

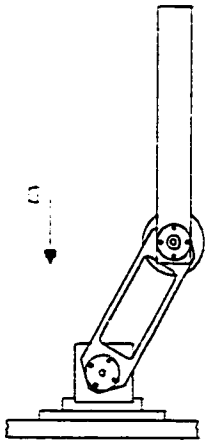


Figure 3.2 Equilibrium configuration

Under-actuated mechanical systems generally have equilibria that depend on both their kinematic and dynamic parameters; see, for example, the Pendubot. If the Pendubot is mounted so that the joint axes are perpendicular to gravity, then there will be a continuum of equilibrium configurations, as shown in Figure 3.2, each corresponding to a constant value, $\bar{\tau}$, of the input torque τ. Examining the equation (3.1) we see the equilibrium configurations are determined by

$$\theta_4 g\cos(q_1)+\theta_5 g\cos(q_1+q_2)=\bar{\tau} \tag{3.15}$$

$$\theta_5 g\cos(q_1+q_2)=0 \tag{3.16}$$

It is easily seen that, applying a constant torque $\bar{\tau}$, the Pendubot will balance at a configuration (q_1, q_2) such that

$$q_1=\cos^{-1}(\frac{\bar{\tau}}{\theta_4 g}) \tag{3.17}$$

$$q_2=n\frac{\pi}{2}-q_1;\quad n=1,3,5,... \tag{3.18}$$

provided $\frac{\bar{\tau}}{\theta_4 g}\le 1$.

3.4 Controllability

The nonlinear system Pendubot may be linearized about an operating point (expressed in 3.18). Although these configurations are unstable, the whole system is controllable by using kinds of control algorithms. The motion range of link one is 360°, and in the equilibrium manifold, there exists:

$$q_1 + q_2 = n\frac{\pi}{2}, \quad n = 1,3,5,...$$

However, we can also easily understand physically how the linearized system becomes uncontrollable at $q_1 = 0, \pm\pi$ as illustrated in Figure 3.3. Note that the zero or reference position for q_1 is horizontal. As the Pendubot approaches these unstable and uncontrollable configurations, the controllability matrix of the linearized approximation becomes increasingly ill-conditioned.

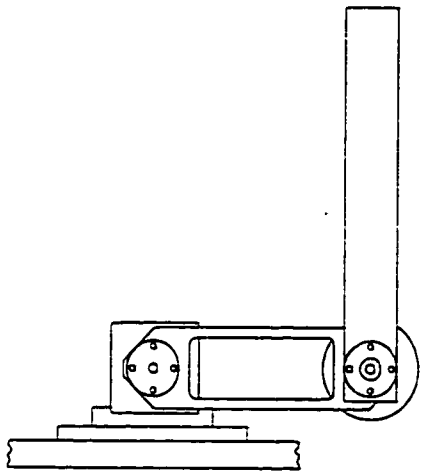


Figure 3.3 Uncontrollable configurations

Actually, from equation (3.17), $q_1 = \cos^{-1}(\frac{\bar{\tau}}{\theta_4 g})$. We can also find if and only if $\bar{\tau} = 0$, $q_1 = 0, \pm\pi$. However, a constant value, $\bar{\tau}$, of the input torque τ is impossible equal to zero to make the second link keep upward equilibrium. Thus, the configurations ($q_1 = 0, \pm\pi$), shown as Figure 3.3, are uncontrollable. In additional, the closer to these

uncontrollable configurations the Pendubot is, the poorer the controllability of the Pendubot is.

Chapter 4

Swing Up Control

As introduced by L.Zadeh in its seminal papers [22], fuzzy logic and approximate reasoning provide a framework for formalizing intuition and common sense knowledge. In the field of process control, these basic tools have been used to formalize experienced operator knowledge into fuzzy rule bases (FRB) giving rise to fuzzy logic controllers via the fuzzy machinery or the fuzzy associative memory model. The FRB proposed in the literature to describe a multiple-input-multiple-output (MIMO) plant controller are usually stated as a collection of If-Then rules such as:

If x_1 is F_1^j, x_2 is F_2^j, ..., and x_n is F_n^j,

Then v_1 is G_1^j, v_2 is G_2^j, ..., and v_q is G_q^j

with j=1,2, ..., M, the x_i and v_k denote the linguistic input and output variables respectively, while F_i^j and G_k^j stand for their linguistic qualifications which are associated with fuzzy subsets. In such a knowledge representation, each if-part of a rule can be viewed as a description of a particular process state to which a collection of linguistically specified control actions is associated. Therefore, the number of rules depends directly upon the number of couples {process state, control actions} an experienced operator can enumerate.

Different methods have been proposed to establish and justify such FRB: fuzzy-model-based control, verbalization of expert experience, observational control [23]. However, because they are based on expert knowledge and/or input-output records made on a plant, such methods are useless when the plant is not physically available nor manually controlled. On the other hand, even when a plant is manually controlled, the rule verbalization process itself can be a difficult task. This is due to the existence of cross-coupling effects between the different variables. Besides these facts, the implementation of a MIMO controller deriving from a standard FRB is much more complicated than the implementation of a MISO controller [24]. Ever though there are some theoretical results concerning the decomposition of multivariable control rules [25] or distributed fuzzy control of multivariable systems [26], they do not lead to a reduction of the size of the FRB.

In this thesis, I present how different basic principles of fuzzy logic can be used to cope with the cross-coupling effects which constructing small FRB for SIMO and MIMO plants. The approach I propose is a set in the context of feasibility studies of controllers when no experienced operator knowledge is available. Provided they satisfy completeness, small FRB are attractive because they lead to easy-to-implement controllers. To deal with the lack of experienced operator I propose to use qualitative common sense knowledge about the plant to be controlled. This qualitative knowledge corresponds to few particular process states for which the qualification of the control actions is easy to derive.

4.1 A Two-Rule-Based Fuzzy Controller

In this part, a fuzzy logic controller (FLC) was designed by the authors in [18] and [19] will be described. Under the assumption of a positive causality link between the control input and the plant output, a joint linguistic variable (plant-state) = (E, RE) is constructed first to represent the plant state [27], the atomic variables E and RE denoting the error and the rate of change of error respectively. Based on the qualitative knowledge of the cause-effect relationship, and because of the lack of the plant dynamics knowledge only two plant states are stated:

E is Positive And RE is Positive,

and

E is Negative And RE is Negative.

Using the notion of joint variable these two propositions translate to

plant-state is P,

and

plant-state is N,

where P and N are respectively defined from the Cartesian product of the fuzzy subsets of the atomic variable which makes up the joint variable. Then, using the available knowledge about the plant, two control rules are derived which correspond to the two situations depicted in figure 4.1:

If plant-state is P **Then** control action is P

If plant-state is N **Then** control action is N

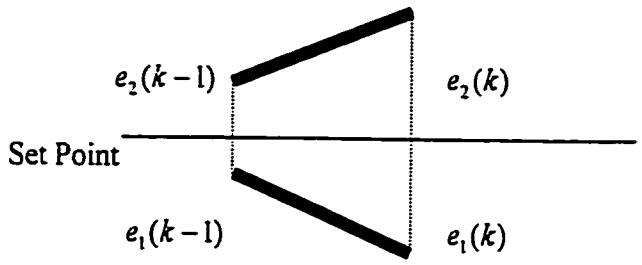


Figure 4.1 The two situations

In these rules, the variable "control action" may stand either for the control input of the plant or its rate of change. Note that in these two situations the sign of the control action can be defined without any ambiguity. The subsets associated with the conclusions are primarily defined over the unit interval such as:

$$VP(v) = \begin{cases} 0 & -1 \le v \le 0 \\ f(v) & 0 \le v \le 1 \end{cases} \tag{4.1}$$

$$VN(v) = \begin{cases} 0 & 0 \le v \le 1 \\ f(|v|) & -1 \le v \le 0 \end{cases} \tag{4.2}$$

where f is chosen to be an automorphism of the unit interval. To guaranty the completeness of the controller, it is sufficient that the support of each of the four input fuzzy subsets associated with the atomic variables cover the corresponding base set entirely. For instance, let X denotes either E or RE, these input subsets can be defined such as:

$$XP(x) + XN(x) = 1 \quad \forall x \in X \tag{4.3}$$

$$XP(x) = (1 + \tanh(a_x x))/2 \quad a_x > 0 \tag{4.4}$$

The degree of firing $a_x, i = 1,2$ of a rule is defined via the algebraic product, hence a_1 is a monotone increasing function of the crisp variables e and re, while a_2 is monotone decreasing. The fuzzy reasoning method is a simplified version of the Tsukamoto's method which leads to a crisp control action σ:

$$v = \left[f^{-1}(a_1) - f^{-1}(a_2)\right] \quad \sigma = Gv; \quad G > 0 \tag{4.5}$$

Since f is an automorphism, we conclude that for $\forall e < 0, \quad re < 0 \; (e > 0, re > 0)$ the value of v is negative $\{v < 0\}$, also $v = 0 \Leftrightarrow a_e e + a_{re} re = 0$. A relationship between this controller and classical PI or PD controllers can be derived, and results concerning the stability of closed loops systems have been proved for the class of LTI-SISO plant models and some non-linear Euler-Lagrange models.

4.2 A Two-Rule-Sign-Based Strategy for the Pendubot

From the above section 4.1, we can derive a novel two-rule-sign-based fuzzy control scheme for the Pendubot. In the Pendubot, the second link may be thought of as a simple pendulum whose motion can be controlled by the torque applied to the first joint. However, the nature of dynamic coupling between the two links results in properties not found in the cart-pole system. Specifically, in the Pendubot there is a continuum of equilibrium configurations, each corresponding to a constant value of the torque.

The control problem we consider is stated as follows: the Pendubot being in the downward stable configuration ($q_1 = -\pi/2;\quad q_2 = 0$), the control goal is to find a suitable control law that will swing up the Pendubot at the inverted unstable configuration ($q_1 = \pi/2;\quad q_2 = 0$) and successfully switch to the balance controller.

The first joint is called the shoulder (S)--- connect with link one, and the other the elbow (E)--- connect with link two. Now we separately discuss the joint variable of two joints.

SS represents the shoulder state that is made up with SE and RSE, that is

$$SS=\{SE, RSE\}. \tag{4.6}$$

The angular shoulder error (SE), is defined as

$$se = \pi/2 - q_1. \tag{4.7}$$

The rate of change of SE (RSE), at time t, is defined as

$$\begin{aligned} rse &= \frac{se(t) - se(t-h)}{h} = \frac{[\pi/2 - q_1(t)] - [\pi/2 - q_1(t-h)]}{h} \\ &= -\frac{q_1(t) - q_1(t-h)}{h} \\ &= -\dot{q}_1 \end{aligned} \tag{4.8}$$

where h is the sampling period (in our experiment, the sampling period used to approximate the rate of changes of the output variables is set to 0.005s).

ES represents the elbow state that is made up with EE and REE, that is

$$ES=\{EE, REE\}. \tag{4.9}$$

The angular elbow error (EE), is defined as:

$$ee = \pi/2 - (q_1 + q_2). \tag{4.10}$$

The rate of change of EE (REE), at time t, is defined as

$$\begin{aligned} ree &= \frac{ee(t) - ee(t-h)}{h} = \frac{[\pi/2 - q_1(t) - q_2(t)] - [\pi/2 - q_1(t-h) - q_2(t-h)]}{h} \\ &= -\frac{[q_1(t) + q_2(t)] - [q_1(t-h) + q_2(t-h)]}{h} \\ &= -\frac{q_1(t) - q_1(t-h)}{h} - \frac{q_2(t) - q_2(t-h)}{h} \\ &= -\dot{q}_1 - \dot{q}_2 \end{aligned} \tag{4.11}$$

Not only for the shoulder state (SS) but also for the elbow state (ES), they are separately made up with two atomic variables the angular error (*se* or *ee*) and its rate of change (*rse* or *ree*). Each of these atomic variables is associated with two fuzzy sets that are labeled ***positive*** and ***negative.***

Let Σ denotes one of these atomic variables, then, from the equations (4.3)-(4.4) in the section 4.1, for any $x \in R$, its related fuzzy sets are given by:

$$\Sigma P(x) + \Sigma N(x) = 1 \tag{4.12}$$

$$\Sigma P(x) = \frac{1 + \tanh(kx)}{2} \tag{4.13}$$

Now we consider the joint shoulder as an example to derive the fuzzy control law. Any real value of *se* (resp. *rse*) is given a non zero grade of membership to both input fuzzy sets. A control strategy for the highest priority goal is stated as:

Rule1: **IF** "SS is SSP" **THEN** force τ is Positive

Rule 2: IF "SS is SSN" THEN force τ is Negative

The fuzzy sets SSP and SSN of SS are defined using a quasi-linear-mean aggregating operator [28], that is the membership grade of any pair {se, rse} to SSP and SSN is given by:

$$SSP(se,rse) = \mu_s SEP(se) + (1-\mu_s) RSEP(rse) \tag{4.14}$$

$$SSN(se,rse) = \mu_s SEN(se) + (1-\mu_s) RSEN(rse) \tag{4.15}$$

where the weighting parameter $\mu_s \in [0,1]$ is drawn from the unit interval. It is used to give more or less importance to one of the two variables. These two rules, having preconditions that cover any range of sensor readings, are given the same meaning: the more true the precondition the more positive (resp. negative) the conclusion. Therefore, according to the equation (4.1)-(4.2) in section 4.1, we use the following output fuzzy sets as the system output:

$$FP(\zeta) = \begin{cases} \zeta & if \zeta \in [0,1] \\ 0 & if \zeta \in [-1,0[\end{cases} \tag{4.16}$$

$$FN(\zeta) = \begin{cases} |\zeta| & if \zeta \in [-1,0] \\ 0 & if \zeta \in]0,1] \end{cases} \tag{4.17}$$

where ζ is related to the applied force, τ, by a scaling factor G_1 such that $\tau = G_1\zeta$, and a slight modification of the reasoning method originally proposed by Tsukamoto [20] the defuzzified out put is simply given by:

$$\tau(se,rse) = G_1\zeta = G_1[SSP(se,rse) - SSN(se,rse)] \tag{4.18}$$

Using equations (4.12) - (4.15), we can get:

$$S(se, rse) = S(se_1, se_2)$$

$$= SSP(se, rse) - SSN(se, rse)$$

$$= \mu_s SEP(se) + (1 - \mu_s) RSEP(rse) - [\mu_s SEN(se) + (1 - \mu_s) RSEN(rse)]$$

$$= \mu_s [SEP(se) - SEN(se)] + (1 - \mu_s)[RSEP(rse) - RSEN(rse)]$$

$$= \mu_s [SEP(se) - (1 - SEP(se))] + (1 - \mu_s)[RSEP(rse) - (1 - RSEP(rse))]$$

$$= \mu_s [2SEP(se) - 1] + (1 - \mu_s)[2RSEP(rse) - 1]$$

$$= \mu_s [2 * \frac{1 + \tanh(k_1 se)}{2} - 1] + (1 - \mu_s)[2 * \frac{1 + \tanh(k_2 rse)}{2} - 1]$$

$$= \mu_s \tanh(k_1 se) + (1 - \mu_s) \tanh(k_2 rse)$$

$$= \mu_s \tanh(se_1) + (1 - \mu_s) \tanh(se_2) \qquad (4.19)$$

In the above equation (4.19), se_1, se_2 stand for $k_1 se, k_2 rse$. The joint variable SS is related to two fuzzy labels, SSP and SSN, from equations (4.12)-(4.15), we also can write as:

$$SSP(se, rse) + SSN(se, rse) = 1 \qquad \forall (se, rse) \in R^2 \qquad (4.20)$$

$$SSP(se, rse) = \mu_s SEP(se) + (1 - \mu_s) RSEP(rse) \qquad (4.21)$$

Similarly, the definition of the elbow error allows to define two categories of elbow states, ESP (positive) and ESN (negative), corresponding to states where the second link is falling to the right or to the left. Using the same basic membership functions, given by

equation (4.12)-(4.13), and the same kind of aggregation operators (4.14)-(4.15), the fuzzy sets related to ESP and ESN are such that:

$$ESP(ee,ree)+ESN(ee,ree)=1 \qquad \forall(ee,ree)\in R^2 \tag{4.22}$$

$$ESP(ee,ree)=\mu_e EEP(ee)+(1-\mu_e)REEP(ree) \tag{4.23}$$

Therefore, same as the derivation of (4.19), we can also get:

$$\begin{aligned} E(ee,ree) &= E(ee_1,ee_2) = ESP(ee,ree)-ESN(ee,ree) \\ &= \mu_e \tanh(k_3 ee)+(1-\mu_e)\tanh(k_4 ree) \\ &= \mu_e \tanh(ee_1)+(1-\mu_e)\tanh(ee_2) \end{aligned} \tag{4.24}$$

where ee_1, ee_2 stand for $k_3 ee$, $k_4 ree$.

In order to swing up the Pendubot and let the two links arrive at the goal position (top) at the same time, we should use the following swing-up control phase:

IF "SS is SSP" **ALSO IF** "ES is ESP" **THEN** torque is Positive;
IF "SS is SSN" **ALSO IF** "ES is ESN" **THEN** torque is Negative.

Where the logical connective "ALSO" is used to explicitly give more importance to the joint variable SS than to the joint variable ES in this swing-up controller. Here "ALSO" is again a quasi-linear-mean operator, with $\lambda > 0.5$ as its parameter. The rule firing strengths are thus given by:

$$OSP=\lambda SSP(se,rse)+(1-\lambda)ESP(ee,ree) \tag{4.25}$$

$$OSN=\lambda SSN(se,rse)+(1-\lambda)ESN(ee,ree) \tag{4.26}$$

where OS is overall joint variable used to describe the system state including the shoulder joint states and the elbow joint states.

Thus as the mentioned before around equation (4.18), using the already given reasoning method to combine the two joint states, and fuzzy sets for the torque variable, the control torque is given by:

$$\begin{aligned}\Gamma &= G[OSP - OSN] \\ &= G[\lambda(SSP(se,rse) - SSN(se,rse)) + (1-\lambda)(ESP(ee,ree) - ESN(ee,ree))] \\ &= G[\lambda S(se_1,se_2) + (1-\lambda)E(ee_1,ee_2)] \qquad (4.27)\end{aligned}$$

where $S(se_1,se_2)$, $E(ee_1,ee_2)$ are defined as equations (4.19) and (4.24).

4.3 Analysis of the Control Law

Both mappings $S(se_1,se_2)$, $E(ee_1,ee_2)$ are $R^2 \rightarrow (-1,1)$, they are continuous and non-decreasing with respect to their arguments. The control torque is thus continuous and bounded. Given (4.19), (4.24), and (4.25), it is easy to see that using $\{S(se_1,se_2),\ E(ee_1,ee_2)\}$ as a coordinate system, the control torque changes its sign on a straight line whose slope is $-(\lambda/(1-\lambda))$ where, in accordance with the chosen highest priority goal, $\lambda > 0.5$ should be chosen carefully.

It should be noted that this single fuzzy PD-scheme requires a very fine tuning to be successful in swinging up and let both links access to the top unstable configuration. As we see above, k_1,k_2,k_3,k_4 correspond the two atomic variables the angular error and its

rate of change in the internal field, which depend on the priority control goal. For instance, for the swing up controller, the priority control goal is to swing link one and link two to the upward inverted configuration, so k_1, k_3 related with the angular error should be set close to 1. On the other hand k_2, k_4 related with the angular velocity of two links should be set small, certainly, $k_1, k_2, k_3, k_4 \in [0,1]$.

The parameters μ_s, μ_e are used to give more or less importance to one of the two variables from the external field. In this swing-up controller, both of μ_s, μ_e are chosen 0.5. It means the angular error and its rate of change are considered as the equal importance.

Finally, let us study the scaling factor G in the equation (4.27), which is the same meaning as G_1 in equation (4.18). In this fuzzy control algorithm, the defuzzification is very simple, which is a slight modification of the reasoning method, shown in (4.27) and (4.18). From the equations (4.19) and (4.24), we can realize that:

$$S(se_1, se_2) \in [-1,1] \qquad \text{and} \qquad E(ee_1, ee_2) \in [-1,1]$$

Because the function "tanh" such as $\tanh(se_1)$, $\tanh(se_2)$, $\tanh(ee_1)$, $\tanh(ee_2)$ is drawn from the interval [-1,1], and the parameters μ_s, μ_e also belong to the unit interval [0,1]. Hence, the absolute value of part $[\lambda S(se_1, se_2) + (1-\lambda)E(ee_1, ee_2)]$ is less than or equal to 1. That means the scaling factor G should be chose relative bigger. During the process

of our implement. I find this factor plays an important pole in whole control algorithm. and its little changing also affects the output.

In the Chapters 8 and 9, when I simulate and implement our control algorithm, I can adjust the controller's parameters based on the above analysis. I can conclude that there exists difference between the continuous-time system and the discrete-time system, and between the simulation and the actual experiment. Although, the control algorithm is same, the tuning is important and different.

Chapter 5

Balancing Control

The control for balancing the Pendubot is very similar to the classical cart-pole inverted pendulum problem. A balancing configuration is defined to be any configuration where $|se| \cong 0$ and $|ee| \cong 0$. The highest priority goal is now to keep the second link balanced at the top position. Figure 5.1 illustrates the balancing configuration at the top position. In order to implement this goal, we design a PD-like fuzzy balancing controller. The rule base is almost contrary to the swing-up controller's rule base because the goals of both controllers are different.

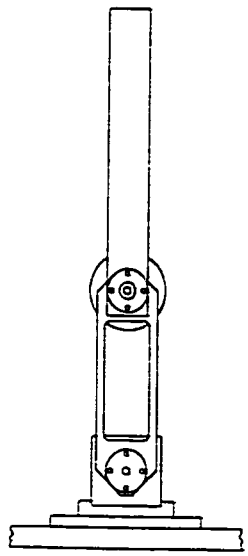
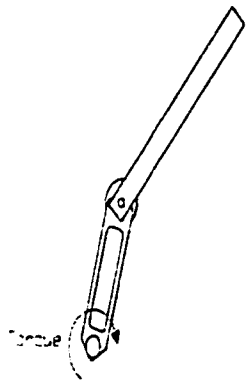


Figure 5.1 The balancing configuration at the top position

5.1 System Analysis

Balancing the Pendubot is similar to the action in acrobatics. In order to keep both links balanced at the unstable configuration, the motor should provided the appropriate torque with the correct direction and magnitude although this torque is too small to close to zero. At the top unstable position, there are four possible situations for the two links, illustrated in Figure 5.2 and Figure 5.3.

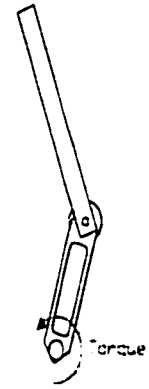


(a)

$q_1 < \pi/2, q_2 < 0$

$se > 0, ee > 0$

$\tau < 0$



(b)

$q_1 < \pi/2, q_2 > 0$

$se > 0, ee > 0$ *or* $ee < 0$

$\tau > 0$

Figure 5.2 The link one on the right side around the equilibrium point

In the Figure 5.2 (a),

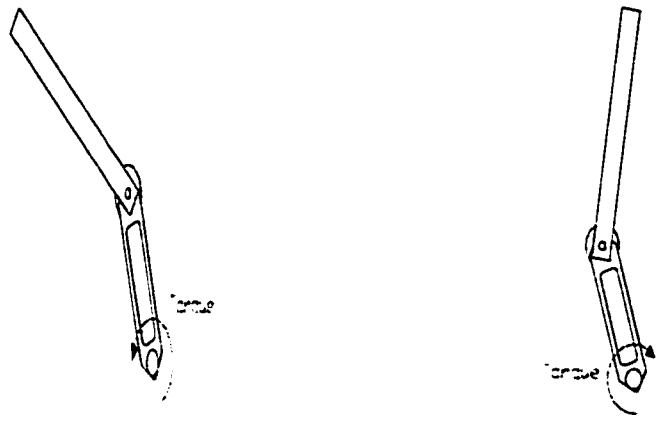
$$q_1 < \pi/2, \quad \Rightarrow \quad se = \pi/2 - q_1 > 0,$$

$$q_2 < 0, \quad \Rightarrow \quad ee = \pi/2 - (q_1 + q_2) > 0,$$

but the control torque has to be sufficiently negative so that the second link start to fall to the left, that is,

$$\tau < 0.$$

Next, in the Figure 5.2 (b), the subsequent attempt to keep a balancing configuration will require a positive torque to move both links toward the inverted upright configuration.



(a)

$q_1 > \pi/2, q_2 > 0$
$se < 0, ee < 0$
$\tau > 0$

(b)

$q_1 > \pi/2, q_2 < 0$
$se < 0, ee > 0$ *or* $ee < 0$
$\tau < 0$

Figure 5.3 The link one on the left side around the equilibrium point

Similarly, in the Figuration 5.3 (a),

$$q_1 > \pi/2, \quad \Rightarrow \quad se = \pi/2 - q_1 < 0,$$

$$q_2 > 0, \quad \Rightarrow \quad ee = \pi/2 - (q_1 + q_2) < 0,$$

but the control torque has to be sufficiently positive so that the second link start to fall to the right, that is,

$$\tau > 0.$$

Next, in the Figure 5.3 (b), the subsequent attempt to keep a balancing configuration will require a negative torque to move both links toward the inverted upright configuration.

5.2 Control Law

As we defined in the Chapter 4, the first joint is called the shoulder (S), and the other the elbow (E). The angular shoulder error (SE), is defined as: $se = \pi/2 - q_1$; the angular elbow error (EE), is defined as: $ee = \pi/2 - (q_1 + q_2)$. The joint variable ES, describing the elbow state is made up with EE and REE (rate of change of EE), whereas SS (shoulder state) is made up with SE and RSE. The definition of the elbow error allows to define two categories of elbow state, ESP (positive) and ESN (negative), corresponding to states where the second link is falling to the right or the left.

In the control system, there are two kinds of inputs: the error signals se, ee and their rate of change rse, ree; there is only one output, the torque τ actuated link one. In fact, from the Chapter 4, I can get:

$$rse = \frac{d(se)}{dt}$$

$$ree = \frac{d(ee)}{dt}$$

Hence, I just put two scaling factors to them, that is,

$$k_{p1}se, \quad k_{d1}rse \qquad \text{and} \qquad k_{p2}ee, \quad k_{d2}ree$$

which are like to the PD controller. After the error and the rate of change, such as se and rse, are correspond with each other by the scaling factors k_{p1}, k_{d1}, we can simply depict the "SSP" and "SSN" as:

IF "se is Positive" **AND IF** "rse is Positive" **THEN** "SS is SSP"

IF "se is Negative" **AND IF** "rse is Negative" **THEN** "SS is SSN"

Here, "AND" is the Zadeh's logical "AND". As the values of se and rse are real numbers, we can use mathematic form to express these rules, such as:

$$SS = k_{p1} se + k_{d1} rse ,$$

which has included both rules "SSP" and "SSN". Similarly, the elbow states can also expressed as the following:

$$ES = k_{p2} ee + k_{d2} ree$$

Because in the balancing control the highest priority goal is to keep the second link balanced, the elbow states are more important than the shoulder states. Thus, the following two rules are used to design a fuzzy PD-like balancing controller:

IF "ES is ESN" **ALSO IF** "SS is SSN" **THEN** torque is Positive

IF "ES is ESP" **ALSO IF** "SS is SSP" **THEN** torque is negative

where the logical connective "ALSO" is used to give more importance to the joint elbow variable "ES" than to the joint shoulder variable "SS". Here, "ALSO" is again a quasi-linear-mean operator, with $\lambda > 0.5$ as its parameter. As the discussion in Chapter 8, I chose 0.75 for the λ.

Therefore, the balancing control law is:

$$\Gamma = -[\lambda ES + (1-\lambda)SS]$$

$$= -\lambda(k_{p2}ee + k_{d2}ree) - (1-\lambda)(k_{p1}se + k_{d1}rse) \qquad (5.1)$$

Chapter 6

Stability Analysis of Fuzzy Trajectory Tracking Controller

The paper [14] displays a fuzzy trajectory tracking control scheme, which has been applied to the Pendubot in our laboratory by Zheng Cai. This algorithm is derived from combining linear regulator theory with the Takagi-Sugeno (T-S) fuzzy methodology. Zheng Cai illustrate the application of this algorithm, by forcing the Pendubot to track in real-time a sinusoidal signal of significative amplitude, around an unstable equilibrium point. Therefore, the implement shows the feasibility of this algorithm. However, the author of [14] failed to provide the stability analysis, which is very basic requirement for the control algorithm. In this chapter, I will analyze its stability.

6.1 Fuzzy Control Algorithm

In the chapter 3, the dynamic matrix equation of motion for the Pendubot is given by equation (3.1). If the state variables are defined as equation (3.9), the state equation of the Pendubot can be written as:

$$\dot{x}(t) = f(x) + g(x)u(t) \stackrel{\Delta}{=} F(x(t), u(t), t) \qquad (6.1)$$

where $x = [q_1, \dot{q}_1, q_2, \dot{q}_2]^T$ is the state vector and $u(t) = [\tau, 0]^T$ is the input vector. The output is $y(t) = q_2$.

A linear model for a specific equilibrium point can be obtaining using Taylor series. For the Pendubot every equilibrium point must satisfy: $q_1 + q_2 = \pi/2$. Thus linearization of equation (6.1) by Taylor series is given as:

$$\delta\dot{x} = F_x(t)\delta x + F_u(t)\delta u \tag{6.2}$$

where

$$F_x(t) := \frac{\partial F(x(t),u(t),t)}{\partial x}\Big|_{x_r,u_r} \tag{6.3}$$

$$F_u(t) := \frac{\partial F(x(t),u(t),t)}{\partial u}\Big|_{x_r,u_r} \tag{6.4}$$

x_r, u_r are the values of x and u for the specific equilibrium point. Refer to Appendix A for a full derivation of these partial derivation terms.

Defining $x := \delta x$; $A := F_x(t)$; $B := F_u(t)$, we have

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{6.5}$$

In the T-S fuzzy methodology [30] [31], nonlinear systems are approximated by a set of linear local models, which is described by a set of fuzzy "IF-THEN" rules, with fuzzy sets in the antecedents and local linear time invariant systems in the consequents. Every i-th rule of a T-S fuzzy model has the following form:

i^{th} Rule

IF $z_1(t)$ is $M_{i1},...,z_j(t)$ is $M_{ji},...,z_q(t)$ is M_{qi}

THEN $\begin{cases} \dot{x}(t) = A_i x(t) + B_i u(t) \\ y(t) = C_i x(t) \end{cases}$

Where $i = 1,...,r$ with r the number of rules; the $z_j (j = 1,...,q)$ are the premise variables, which may be functions of the states or another variables; the M_{ji} are the fuzzy sets; $x \in R^n$ is the state vector; $u \in R^m$ is the input vector; $y \in R^p$ is the output vector; A_i, B_i, and C_i are matrices of adequate dimension. Therefore, the final state and output of the fuzzy system is inferred as follows:

$$\dot{x}(t) = \frac{\sum_{i=1}^{r} \lambda_i(z(t))\{A_i x(t) + B_i u(t)\}}{\sum_{i=1}^{r} \lambda_i(z(t))} = \sum_{i=1}^{r} w_i(z(t))\{A_i x(t) + B_i u(t)\} \qquad (6.6)$$

$$y(t) = \frac{\sum_{i=1}^{r} \lambda_i(z(t)) C_i x(t)}{\sum_{i=1}^{r} \lambda_i(z(t))} = \sum_{i=1}^{r} w_i(z(t)) C_i x(t) \qquad (6.7)$$

where,

$\sum_{i=1}^{r} w_i(z(t)) = 1$, $w_i(z(t)) \in [0,1]$ for all i, is a known nonlinear function of $x(t)$ and

$$w_i(z(t)) = \frac{\lambda_i(z(t))}{\sum_{i=1}^{r} \lambda_i(z(t))} = \frac{\mu_{i1}(z_1(t)) * \mu_{i2}(z_2(t)) * ... * \mu_{iq}(z_q(t))}{\sum_{i=1}^{r} \{\mu_{i1}(z_1(t)) * \mu_{i2}(z_2(t)) * ... * \mu_{iq}(z_q(t))} \qquad (6.8)$$

μ_{ij} is the membership function of $z_j(t)$ in M_{ij}.

Now let's consider the linear regulator theory (LRT) [29] [21]. Given a linear system:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + P\omega(t) \\ \dot{\omega}(t) &= S\omega(t) \\ e(t) &= Cx(t) + Q\omega(t) \end{aligned} \qquad (6.9)$$

where ω is a vector containing external disturbances and/or references and e is the tracking error. Accordingly, state feedback and error feedback controllers can be desinged as

$$u(t) = Kx(t) + L\omega(t) \tag{6.10}$$

Although linear regulation theory comes from linear system, many researchers have successfully used it in the nonlinear system and verified its possibility and stability [29] [32]. Here we have linearized the nonlinear system (6.1) at each equilibrium point, so we can use LRT and T-S fuzzy methodology to control the nonlinear system.

For the nonlinear system (Pendubot), we can linearize it at each chosen equilibrium point, and then we design local control laws based on the LRT, for each local linear model. The fuzzy trajectory tracking control algorithm is described as the followings.

i^{th} Plant Rule

IF $z_1(t)$ is $M_{i1}, ..., z_j(t)$ is $M_{jt}, ..., z_q(t)$ is M_{qt}

$$\textbf{THEN} \begin{cases} \dot{x}(t) = A_i x(t) + B_i u(t) + P_i \omega(t) \\ \dot{\omega}(t) = S\omega(t) \\ e(t) = C_i x(t) + Q_i \omega(t) \end{cases} \tag{6.11}$$

i^{th} Controller Rule

IF $z_1(t)$ is $M_{i1}, ..., z_j(t)$ is $M_{jt}, ..., z_q(t)$ is M_{qt}

THEN $u = K_i x(t) + L_i \omega(t)$

Hence, the output of the fuzzy controller is given as:

$$u(t) = \frac{\sum_{i=1}^{r} \lambda_i(z(t))[K_i x(t) + L_i \omega(t)]}{\sum_{i=1}^{r} \lambda_i(z(t))} = \sum_{i=1}^{r} w_i(z(t))[K_i x(t) + L_i \omega(t)] \qquad (6.12)$$

6.2 Stability Analysis

Considering the equations (6.6), (6.11), and (6.12), the fuzzy control system has become as the following,

$$\dot{x}(t) = \sum_{i=1}^{r} w_i(z(t))\{A_i x(t) + B_i u(t) + P_i \omega(t)\}$$

$$= \sum_{i=1}^{r} w_i(z(t))\{A_i x(t) + P_i \omega(t)\} + \sum_{i=1}^{r} \{w_i(z(t)) B_i \sum_{i=1}^{r} w_i(z(t))[K_i x(t) + L_i \omega(t)]\} \qquad (6.13)$$

$$\dot{\omega}(t) = S\omega(t) \qquad (6.14)$$

Step1:

Let $\omega(t) = 0$, the fuzzy trajectory tracking control algorithm is simplified a fuzzy state feedback closed-loop control algorithm [33]. Accordingly, the fuzzy control system becomes,

$$\dot{x}(t) = \sum_{i=1}^{r} w_i(z(t)) A_i x(t) + \sum_{i=1}^{r} \{w_i(z(t)) B_i \sum_{i=1}^{r} w_i(z(t)) K_i x(t)\}$$

$$= \sum_{i=1}^{r} w_i(z(t)) A_i x(t) + \sum_{i=1}^{r} \sum_{j=1}^{r} \{w_i(z(t)) w_j(z(t)) B_i K_j x(t)\}$$

$$= \sum_{i=1}^{r} \sum_{j=1}^{r} \{w_i(z(t)) w_j(z(t))[A_i + B_i K_j] x(t)\}$$

$$= \sum_{i=1}^{r} \sum_{j=1}^{r} \{w_i w_j H_{ij} x(t)\} \qquad (6.15)$$

where $H_{ij} = A_i + B_i K_j$, $i, j = 1,2,...,r$, $w_i = w_i(z(t))$, and $w_j = w_j(z(t))$.

To investigate the stability of (6.15), I employ the following Lyapunov function in quadratic form [43],

$$V = \frac{1}{2} x(t)^T P x(t) \tag{6.16}$$

where $(.)^T$ denotes the transpose of a vector or matrix, $P \in R^{n \times n}$ is a symmetric positive definite matrix. Differentiating (6.16), we have,

$$\dot{V} = \frac{1}{2}(\dot{x}^T P x + x^T P \dot{x}) \tag{6.17}$$

From (6.15) and (6.17), we get,

$$\dot{V} = \frac{1}{2}\left[\left(\sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j H_{ij} x(t)\right)^T P x(t) + x(t)^T P\left(\sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j H_{ij} x(t)\right)\right]$$

$$= \frac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j x(t)^T (H_{ij}^T P + P H_{ij}) x(t)$$

$$= -\frac{1}{2}\sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j x(t)^T O_{ij} x(t)$$

$$= -\frac{1}{2}\sum_{j=1}^{r} w_j \left(x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)\right) \tag{6.18}$$

where $O_{ij} = -(H_{ij}^T P + P H_{ij})$

Recall (6.8), $w_i(z(t)) = \dfrac{\lambda_i(z(t))}{\sum_{i=1}^{r}\lambda_i(z(t))} = \dfrac{\mu_{i1}(z_1(t)) * \mu_{i2}(z_2(t)) * \ldots * \mu_{iq}(z_q(t))}{\sum_{i=1}^{r}\{\mu_{i1}(z_1(t)) * \mu_{i2}(z_2(t)) * \ldots * \mu_{iq}(z_q(t))}$. we design the membership functions $\mu_{ij}(z(t))$ in (6.8) as follows,

$$\mu_{i1} = \begin{cases} 1 - \dfrac{\sum_{j=2}^{r} x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|} & \text{if } \sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right| \neq 0 \\ \dfrac{1}{r} & \text{otherwise} \end{cases} \tag{6.19}$$

$$\mu_{ij} = \begin{cases} \dfrac{x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|} & \text{if } \sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right| \neq 0 \\ \dfrac{1}{r} & \text{otherwise} \end{cases} \quad j = 2,3,\ldots,r \tag{6.20}$$

If we assume $\sum_{j=1}^{r} w_j \left| \left(x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right) \right| \neq 0$, substitute the above two equation for m_j into (6.18), we have,

$$\dot{V} \leq -\frac{1}{2}\left[1 - \frac{\sum_{j=2}^{r} x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|} \right] x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) - \frac{1}{2} \sum_{j=2}^{r} \frac{\left(x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right)^2}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|}$$

$$\leq -\frac{1}{2}\left[1 - \frac{\sum_{j=2}^{r} x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|} \right] x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)$$

Then, $x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) > 0$, when $x(t) \neq 0$ and $x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) = 0$ when $x(t) = 0$. Hence,

$$\left[1 - \frac{\sum_{j=2}^{r} x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|} \right] > 0$$

We can conclude that,

$$\dot{V} \leq -\frac{1}{2}\left[1 - \frac{\sum_{j=2}^{r} x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t)}{\sum_{j=1}^{r} \left| x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \right|} \right] x(t)^T \sum_{i=1}^{r} w_i O_{ij} x(t) \leq 0 \tag{6.21}$$

Equalities hold when $x(t) = 0$.

Therefore, the above derivation has verified the stability of the fuzzy state feedback control algorithm.

Step2:

Let $\omega(t) \neq 0$, we should show the conditions for the solvability of this nonlinear system (the Pendubot).

For a linear system described in equation (6.9), the state feedback and error feedback controllers can be designed as equation (6.10). The linear regulator theory is based the following three assumptions (according to [21]):

A1: $\sigma(S) \subset clos(C^+) = \{\lambda \in C \,\mathrm{Re}[\lambda] \geq 0\}$;

A2: the pair (A, B) is stabilizable;

A3: the pairs $[C \quad Q]$, *and* $\begin{bmatrix} A & P \\ 0 & S \end{bmatrix}$ is detectable.

The first assumption does not involve a loss of generality because asymptotically stable modes in the exosystem do not affect the regulation of the output. The second assumption is indeed necessary for asymptotic stabilization of the closed loop via either state or error feedback. The third one is stronger than the assumption of detectability of the pair (C, A), which would be necessary for asymptotic stabilization of the closed loop via error feedback.

Suppose Assumptions A1, A2, and A3 hold, and then the linear state feedback regulator problem is solvable if and only if there exist matrices Π and Γ which solve the linear matrix equations

$$\Pi S = A\Pi + B\Gamma + P \tag{6.22}$$

$$C\Pi + Q = 0 \tag{6.23}$$

As a matter of fact, conditions (6.22) and (6.23) are exactly the conditions which the matrices Π and Γ should fulfill so that the following linear mappings hold

$$x(t) = \Pi\omega(t) \tag{6.24}$$

$$u(t) = \Gamma\omega(t) \tag{6.25}$$

If and only if the conditions (6.24) and (6.25) hold, the tracking error, will approach to zero, $e(t) \to 0$ as $t \to \infty$, which can be shown as the follows.

$$\begin{aligned} e(t) &= Cx(t) + Q\omega(t) \\ &= C\Pi\omega(t) + Q\omega(t) \\ &= (C\Pi + Q)\omega(t) = 0 \end{aligned} \tag{6.26}$$

In the fuzzy trajectory tracking control algorithm, the nonlinear system Pendubot is regarded as several local linear model around each equilibrium point. The rules for the plant and controller are expressed in equations (6.11)-(6.12). Then, we can write the tracking error of the whole system as:

$$e(t) = \sum_{i=1}^{r} w_i(z(t))\left[C_i x(t) + Q_i\omega(t)\right] \tag{6.27}$$

where $w_i(z(t))$ is defined in (6.18), which is related with the weight of i-th local linear model. As each local control law is designed based on the linear regulation theory, each

tracking error $e_i(t) \to 0$ if and only if there exist matrices Π_i and Γ_i (for i-th local linear system) which solve the linear matrix equations

$$\Pi_i S = A_i \Pi_i + B_i \Gamma_i + P_i \tag{6.28}$$

$$C_i \Pi_i + Q_i = 0 \tag{6.29}$$

That also can be written as:

$$x(t) = \Pi_i \omega(t) \tag{6.30}$$

$$u_i(t) = \Gamma_i \omega(t) \tag{6.31}$$

Substituting to (6.27), we can get the tracking error $e(t) = 0$.

Chapter 7

Combining the Controllers

The control scheme I proposed is to use a so-called logic based switching control: design a fuzzy PD-like controller I (Chapter 4) to swing up the link one and link two to the top configuration at the almost same time, then switch to another fuzzy PD-like controller II (Chapter . 5) to keep the links balanced at the unstable position ($q_1 = \pi/2, q_2 = 0$), or switch to another trajectory tracking controller III (Chapter 6, implemented by Zheng Cai) to handle the Pendubot to track the desire trajectory. This switching algorithm will connect each both controller together.

7.1 Combining in the Simulation

With both the swing up controller and the balancing controller complete, an algorithm is needed to connect them. At first, I separately designed the controllers using SIMULINK, and then designed a switching function to combine them together. In the whole research process, I finish the parameters adjusting in the simulation and apply its results to the Pendubot. Initially when working with only computer simulations of the Pendubot, the controllers were switched by different switching functions in the simulation of the discrete model and the continuous model. Let's show the two switching function as the followings.

In the simulation of continuous model, the switching function is same as the actual control algorithm, which consists of the error between the current states and the goal states (both links' position in the top configuration, that is, $q_1 = \pi/2,\ \ q_2 = 0$). Observing the output of the torque, we can find the controlling results of the simulation are very smooth and stable. There is no big jump for the torque during the whole simulation. After the switching function was determined, I use the multiport switching block to connect the two controllers and the switching function. The details about this part are illustrated in the Chapter 8. The switching function is chosen as:

$$\begin{aligned} &Switching\_Function \\ &= 1 + ((abs(u[1] - \frac{\pi}{2}) < 0.1) \,\&\, \&(abs(u[2]) < 0.2)) \end{aligned} \tag{7.1}$$

where

$u[1] = q_1$

$u[2] = q_2$

& & is the logical "AND" collation operation

abs is the function of absolute value.

In the simulation of the discrete model, the switching between the swing-up controller and the balancing was determined by the simulation time. When I change the model from the continuous model to the discrete model, the parameters of the controllers has been changed. However, this kind of controller is closed to the actual digital controller because I chose the same sampling time. That means the tuning of the parameters is suitable for the real digital controller whose original code is C Program. The switching time was determined by observing when the swing-up control had brought the links almost to rest

at the desired equilibrium position. Moreover, the current simulation time is calculated by the "clock" block that records the current simulation time. Because this switching method is sudden and compelling, the curve of the torque displays a little dithering in the two links. Although this method worked well for the simulation but behaved poorly when realized on the actual Pendubot. The reason is that the simulations are exactly repeatable but the actual runs are susceptible to different initial conditions and computational noise making them unable to repeat reliably. In addition, the simulation model doesn't consider the friction of the whole system and other actual factor. All in all, the switching time is set at 0.7 second, the switching function simply is:

$$Swithing\_Function = 1 + (u[1] > 0.7)$$

where $u[1]$ is the current_simulation time.

7.2 Combining in the Implementation

The following algorithm was used instead to give the Pendubot more intelligence and switch the control by watching the states of the system.

```
if (!catch) {  /* if links have not come in range to balance */
        If (fabs(x1k-HALFPI) < .10  {
                If (fabs(x3k) < .20)  {
                        u=.....          /* balancing control or trajectory tracking control */
                        if (fabs(u) < 6)  {  /* if balancing control check is not too large */
                                pd = 0;     /* switch from swing up control to balance */
                                catch = 1;
                        }       /* endif */
                }       /* endif */
        }       /* endif */
```

```
}       /* endif */
if (pd) {       /* if swing up control is still in use */
        ..... /* swing up controller */
        }
else {
        ..... /* balancing controller or trajectory tracking controller */
        }       /* endif */
```

This algorithm waits for link one to arrive within 0.1 radians of its equilibrium position and then checks link two. If link two is also within 0.20 radians of its equilibrium position, the balancing control is calculated. If the control output is less then 6 volts, 9.95 volts being the maximum DAC output for the Pendubot, the control is switched to the balancing mode. Otherwise the links are passing too quickly through the equilibrium point and the swing up control remains intact.

Another implementation issue that arises in the controller design of the Pendubot is the approximation of the joint velocities. There is only position feedback in the system so a finite approximation is used to estimate the velocity. To find the velocity I simply used the finite difference method,

```
x2k = (x1k-x1old) / SAMPLE;          /* x1=q1 , x2=q̇1 */

x4k = (x3k-x3old) / SAMPLE;          /* x3=q2 , x4=q̇2 */
```

This method creates numerical error or noise in the calculation of the control effort, though, due to the finite resolution of the optical encoders. Then I simply take the

average of the last three velocities helped to filter and decrease this noise, shown as the next program.

```
/* filter velocity with an average */
x2k = (x2k+x2old1+x2old2) / 3.0;      /* average last 3 velocities to get */
x4k = (x4k+x4old1+x4old2) / 3.0;      /* rid of quantization noise */
```

A dither signal was also needed to help balance the links in the top position. Due to friction and the increased effect of gravity on the links in the top position, the balancing control was not able to hold the links motionless.

Chapter 8

Simulation

This chapter displays the simulation results found when simulating the Pendubot with the SIMULINK 3.0 in MATLAB 6.0. I start to found the Pendubot model according to the dynamic equation. The second step is to build the swing-up controller based on the fuzzy swing-up algorithm (see chapter 4) and adjust the control parameters in order to reach the desired results. The third step is the same as the second step to get the simulation model of balancing controller. In succession, I combine the above two controllers together using the suitable switching function to complete the whole simulation system.

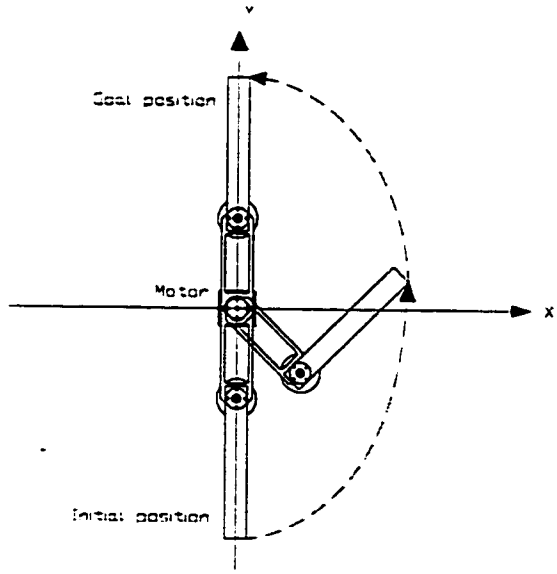


Figure 8.1 Swing-up and balancing control

As we know, the control task is to bring the Pendubot from a straight-down resting position to a balanced straight-up vertical position (as shown in Figure 8.1). Initial and goal positions are $(q_1 = -\pi/2, \quad q_2 = 0)$ and $(q_1 = \pi/2, \quad q_2 = 0)$, respectively.

8.1 Simulation Pendubot Model

As we know from chapter 3, the dynamic equations of Pendubot consist of the angles (q_1, q_2), angular velocities $(\dot{q}_1, \dot{q}_2)$, and angular acceleration $(\ddot{q}_1, \ddot{q}_2)$, so I should derive the relationship between them to build up the simulation model. We simulate the model (3.1) using SIMULINK, and the derivation is shown as the following part.

Recall the equation (3.9):

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = D(q)^{-1}\tau - D(q)^{-1}C(q,\dot{q})\dot{q} - D(q)^{-1}g(q)$$

$$\text{with } D(q) = \begin{bmatrix} \theta_1 + \theta_2 + 2\theta_3 \cos q_2 & \theta_2 + \theta_3 \cos q_2 \\ \theta_2 + \theta_3 \cos q_2 & \theta_2 \end{bmatrix},$$

$$C(q,\dot{q}) = \begin{bmatrix} -\theta_3 \sin(q_2)\dot{q}_2 & -\theta_3 \sin(q_2)\dot{q}_2 - \theta_3 \sin(q_2)\dot{q}_1 \\ \theta_3 \sin(q_2)\dot{q}_1 & 0 \end{bmatrix},$$

$$g(q) = \begin{bmatrix} \theta_4 g \cos q_1 + \theta_5 g \cos(q_1 + q_2) \\ \theta_5 g \cos(q_1 + q_2) \end{bmatrix},$$

$$\tau = \begin{bmatrix} \tau \\ 0 \end{bmatrix}$$

I can compute and get

$$D(q)^{-1} = \frac{1}{\theta_1\theta_2 - \theta_3^2\cos^2 q_2}\begin{bmatrix} \theta_2 & -\theta_2 - \theta_3\cos q_2 \\ -\theta_2 - \theta_3\cos q_2 & \theta_1 + \theta_2 + 2\theta_3\cos q_2 \end{bmatrix} \tag{8.1}$$

Thus,

$$\Rightarrow D(q)^{-1}\tau = \frac{\tau}{\theta_1\theta_2 - \theta_3^2\cos^2 q_2}\begin{bmatrix} \tau\theta_2 \\ \tau(-\theta_2 - \theta_3\cos q_2) \end{bmatrix} \tag{8.2}$$

$$\Rightarrow D(q)^{-1}C(q,\dot{q})\dot{q} = \frac{1}{\theta_1\theta_2 - \theta_3^2\cos^2 q_2}\begin{bmatrix} \theta_2 & -\theta_2 - \theta_3\cos q_2 \\ -\theta_2 - \theta_3\cos q_2 & \theta_1 + \theta_2 + 2\theta_3\cos q_2 \end{bmatrix}$$

$$\begin{bmatrix} -\theta_3\sin(q_2)\dot{q}_2 & -\theta_3\sin(q_2)\dot{q}_2 - \theta_3\sin(q_2)\dot{q}_1 \\ \theta_3\sin(q_2)\dot{q}_1 & 0 \end{bmatrix}\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$= \frac{1}{\theta_1\theta_2 - \theta_3^2\cos^2 q_2}\begin{bmatrix} -\theta_2\theta_3\sin q_2(\dot{q}_1 + \dot{q}_2)^2 - \theta_3^2(\sin q_2)(\cos q_2)\dot{q}_1^2 \\ \theta_3(\sin q_2)(\theta_2 + \theta_3\cos q_2)(\dot{q}_1 + \dot{q}_2)^2 + \theta_1\theta_3(\sin q_2)\dot{q}_1^2 \end{bmatrix}$$

(8.3)

$$\Rightarrow D(q)^{-1}g(q) = \frac{1}{\theta_1\theta_2 - \theta_3^2\cos^2 q_2}\begin{bmatrix} \theta_2 & -\theta_2 - \theta_3\cos q_2 \\ -\theta_2 - \theta_3\cos q_2 & \theta_1 + \theta_2 + 2\theta_3\cos q_2 \end{bmatrix}$$

$$\begin{bmatrix} \theta_4 g\cos q_1 + \theta_5 g\cos(q_1 + q_2) \\ \theta_5 g\cos(q_1 + q_2) \end{bmatrix}$$

$$= \frac{1}{\theta_1\theta_2 - \theta_3^2\cos^2 q_2}\begin{bmatrix} \theta_2\theta_4 g\cos q_1 - \theta_3\theta_5 g\cos q_2\cos(q_1 + q_2) \\ \theta_5 g(\theta_1 + \theta_3\cos q_2)\cos(q_1 + q_2) - \theta_4 g(\theta_2 + \theta_3\cos q_2)\cos q_1 \end{bmatrix}$$

(8.4)

Therefore, connecting the three part (8.2), (8.3), and (8.4), I have

$$\begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} = D(q)^{-1}\tau - D(q)^{-1}C(q,\dot{q})\dot{q} - D(q)^{-1}g(q) \quad \Rightarrow$$

$$\ddot{q}_1 = \tau\theta_2 - \theta_2\theta_3\sin q_2(\dot{q}_1 + \dot{q}_2)^2 - \theta_3^2(\sin q_2)(\cos q_2)\dot{q}_1^2 + \theta_2\theta_4 g\cos q_1$$

$$-\theta_3\theta_5 g(\cos q_2)\cos(q_1 + q_2) \tag{8.5}$$

$$\ddot{q}_2 = -\tau\theta_2 - \tau\theta_3 \cos q_2 + \theta_3(\sin q_2)(\theta_2 + \theta_3 \cos q_2)(\dot{q}_1 + \dot{q}_2)^2 + \theta_1\theta_3(\sin q_2)\dot{q}_1^2$$

$$+\theta_5 g\cos(q_1 + q_2)(\theta_1 + \theta_3 \cos q_2) - \theta_4 g(\cos q_1)(\theta_2 + \theta_3 \cos q_2) \qquad (8.6)$$

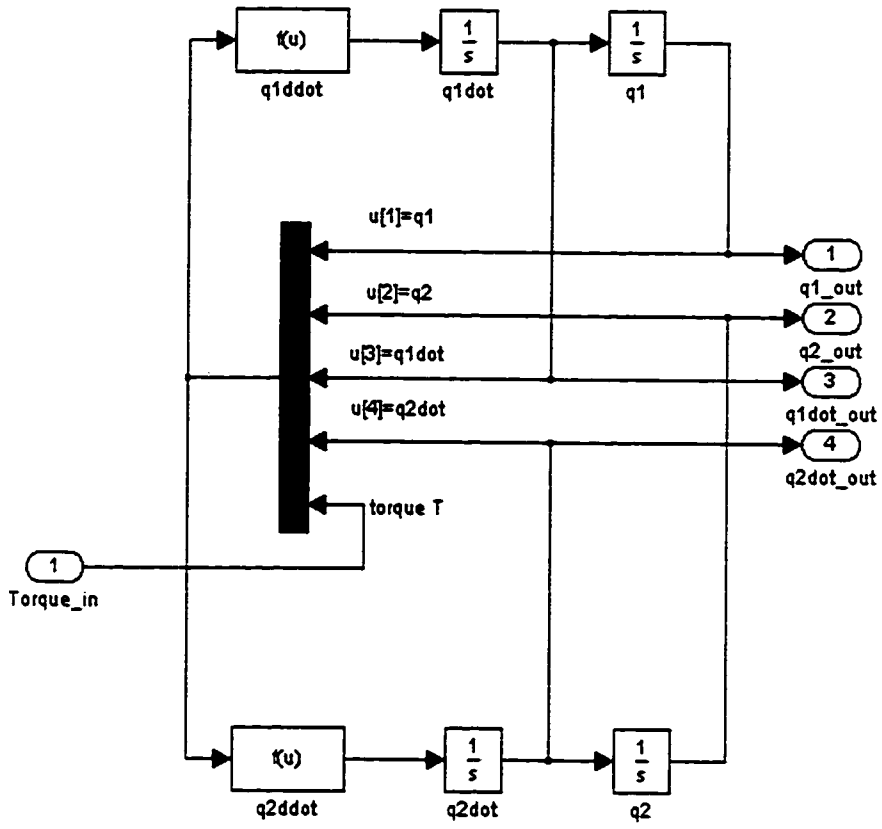


Figure 8.2 Subsystem 1: Pendubot model

In my simulation model, the function of "q1ddot" is expressed as equation (8.5), and the function of "q2ddot" is equation (8.6). As far the torque τ or T, it depends on the feedback value computed by the control law. The real simulation block about the Pendubot model is illustrated in Figure 8.2, which is defined as subsystem 1.

8.2 Swing-Up Simulation Model

At the beginning of the simulating swing-up controller, I just found it based on the swing-up control law, which is the equation (4.27),

$$\Gamma = G[\lambda S(se_1, se_2) + (1-\lambda)E(ee_1, ee_2)] \tag{8.7}$$

where

$$S(se_1, se_2) = \mu_s \tanh(se_1) + (1-\mu_s)\tanh(se_2) \tag{8.8}$$

$$E(ee_1, ee_2) = \mu_e \tanh(ee_1) + (1-\mu_e)\tanh(ee_2) \tag{8.9}$$

These functions were introduced in Chapter 4. Here, I should deal with the tunings such as the weight parameters μ_s, μ_e, λ, the defuzzification parameter G, and the scaling factors k_1, k_2, k_3, k_4, which are defined as:

$$se_1 = k_1 se, \quad se_2 = k_2 rse,$$

$$ee_1 = k_3 ee, \quad ee_2 = k_4 ree$$

Firstly, we chose the suitable scaling factors k_1, k_2, k_3, k_4 by trial and error based on the main goal is to swing up two links to the top position and let them arrive there at the same time as possible. The important problem that we should pay attention to is that the angular velocities $(\dot{q}_1, \dot{q}_2)$ can't be too high when both links are closed to the goal position. Because during the balancing the Pendubot, the angular velocities of links are almost equal to error, the switching between two controllers is not unstable if the angular velocities $(\dot{q}_1, \dot{q}_2)$ at the time of switching are very large. Therefore, I applied the above idea to choose the four factors, which are:

$$k_1 = 1.0, \quad k_2 = 0.167$$

$$k_3 = 1.0, \quad k_4 = 0.1$$

Secondly, I consider the weight parameters μ_s, μ_e, λ after the scaling factors were determined. I simply set $\mu_s, \mu_e = 0.5$, which means the importance of angles and angular velocities is same. Another parameter λ displays the weight of two links during the swing-up, and it depends on the main control goal: swing up the link one first, and make the link two arrive at the goal position at the same time. Thus $\lambda > 0.5$, we change it in the continuous time model simulation and the discrete model simulation.

Finally, G is related with the actual torque applied to the Pendubot. If it is too small, the link one will pass the tip position until the link two arrives there, or the accelerations at the switching time is too large. If it is too big, there is not enough energy to swing up two links. According to our trial and error, the range of G is about 31-36.

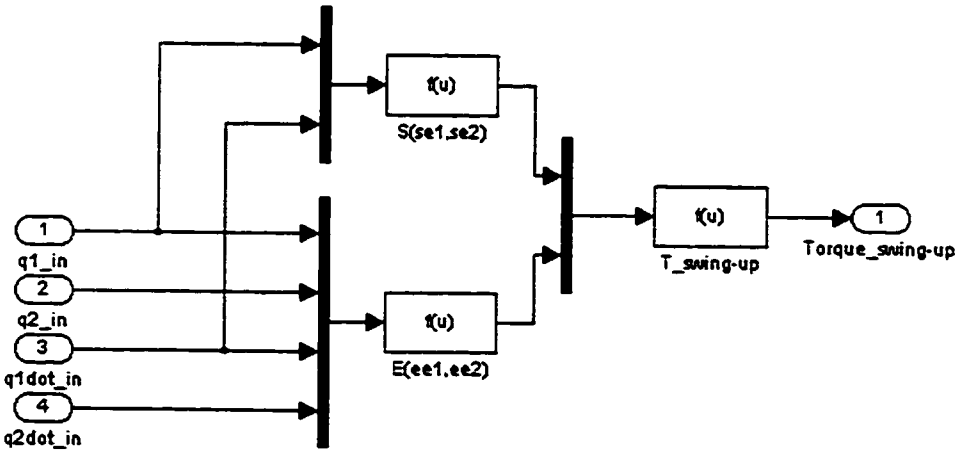


Figure 8.3 Subsystem 2: swing-up controller

Figure 8.3 shows the subsystem 2 in the simulation model, where the function $S(se_1, se_2)$ comes from equation (8.8), $E(ee_1, ee_2)$ from (8.9), and the function T_swing-up comes from (8.7).

8.3 The Balancing Simulation Model

Chapter 5 has given the balancing algorithm, and now we recall the equation (5.1)

$$\Gamma = -\lambda(k_{p2}ee + k_{d2}ree) - (1-\lambda)(k_{p1}se + k_{d1}rse). \qquad (8.10)$$

Where ee, ree, se, rse are defined in chapter 4 and in section 8.2 of this chapter. What we should do is to determine the control parameters $k_{p1}, k_{d1}, k_{p2}, k_{d2}$ and the weight factor λ. Because the priority control goal in the balancing is to keep the unactuated link two upright, the weight λ of elbow joint is large than 50% and we chose λ=0.75. As for the other four control parameters, we analyzed the data of controller designed by LQR method in order to decide our control parameters. Finally, I have

$$k_{p1} = 21.65, \quad k_{d1} = 2.75$$

$$k_{p2} = 0.88, \quad k_{d2} = 4.25$$

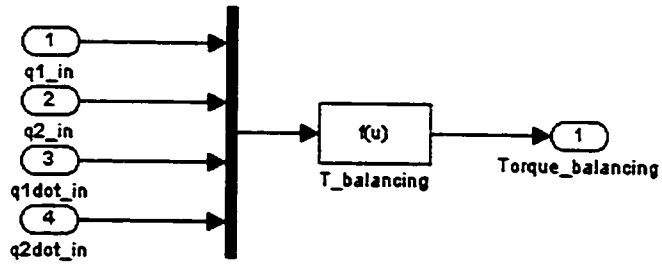


Figure 8.4 Subsystem 3: balancing controller

Figure 8.4 illustrate the simulation block of balancing controller, and the function T_balancing is come from equation (8.10).

8.4 The Final Simulation Model

Step 1:

After completing system model, swing-up model, and balancing model, I began with connecting system model with swing-up model and get the swinging-up simulation results (see Section 8.5). Figure 8.5 shows the swing-up simulation block.

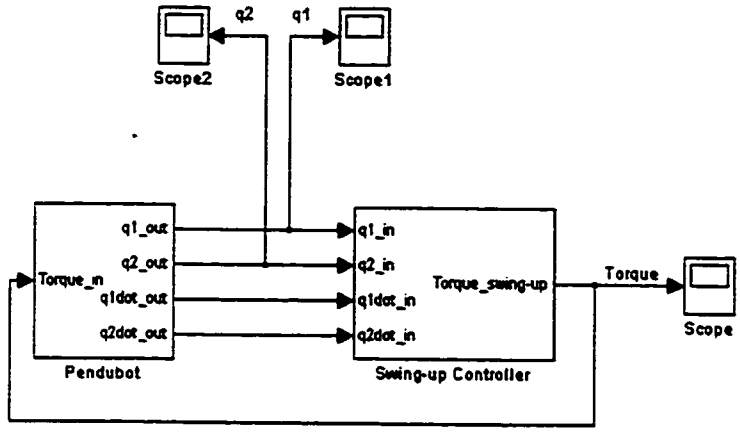


Figure 8.5 Swinging up simulation

Step 2:

I just adjust the balancing controller by connecting the system model and the balancing model, shown in Figure 8.6. Because the initial condition is set based on the analysis of the swing-up simulation results, the balancing simulation results are not very important. Thus we didn't show them separately.

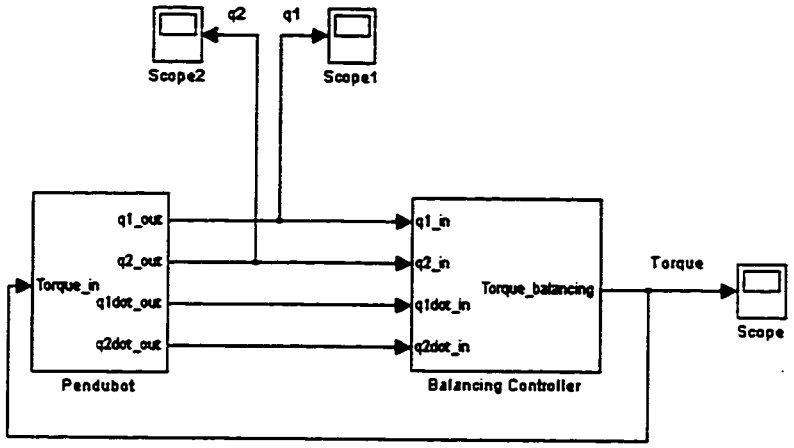
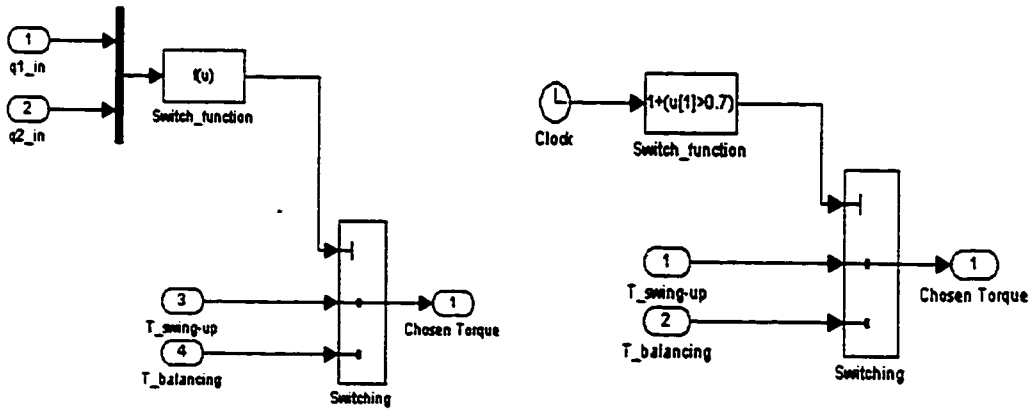


Figure 8.6 Balancing simulation

Step 3:

I design the switching function (see chapter 7) and build a switching subsystem that is illustrated in Figure 8.7.



(a) Using in the continuous time model **(b)** Using in the discrete time model

Figure 8.7 Subsystem 4: switching block

Step 4:

Connect all subsystems to simulate the swinging-up and balancing the Pendubot. Figure 8.8 shows the whole simulating block. The simulation results are in Section 8.5.

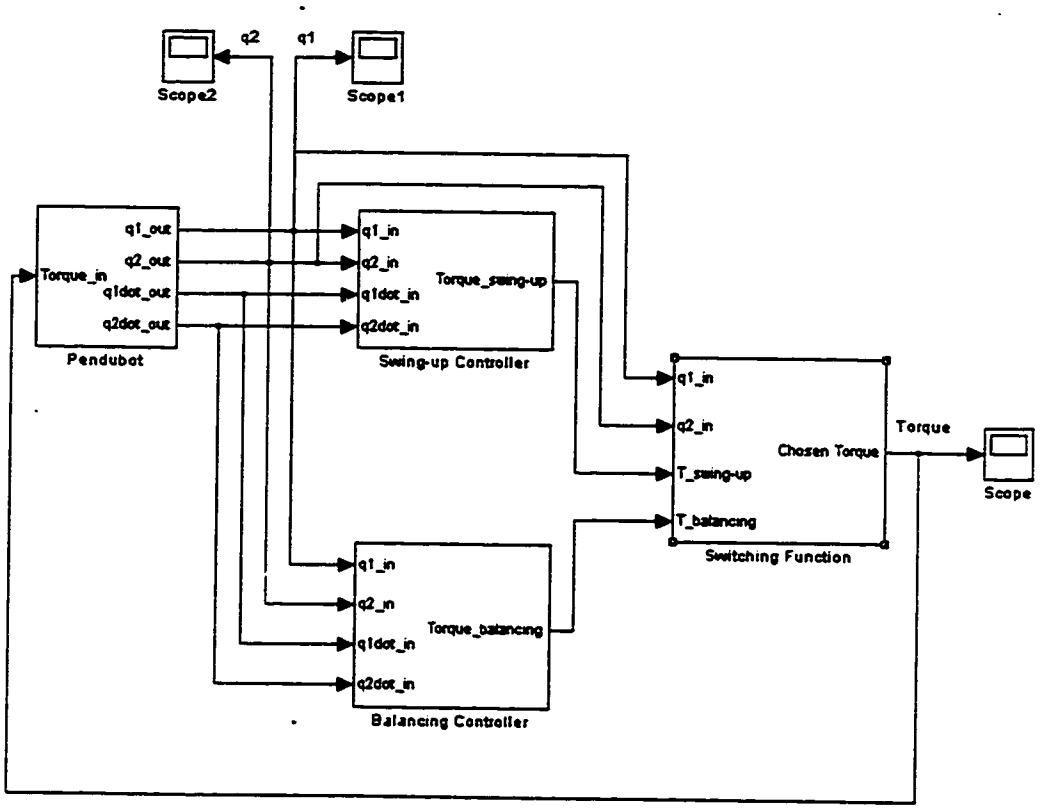


Figure 8.8 The whole simulation system

Step 5:

I first simulate the continuous time model, and the section 8.5.1 shows the simulation results. Then I replace the Integrator block of continuous system with the Discrete-Time Integrator and change the switching block. Figure 8.9 shows discrete-time system model, where the sampling time is equal to 0.005 seconds, same as the actual digital controller. Of course, we should adjust the control parameters such as λ, G, k etc. Finally, we simulate the discrete-time system and get the simulation results (see section 8.5.2).

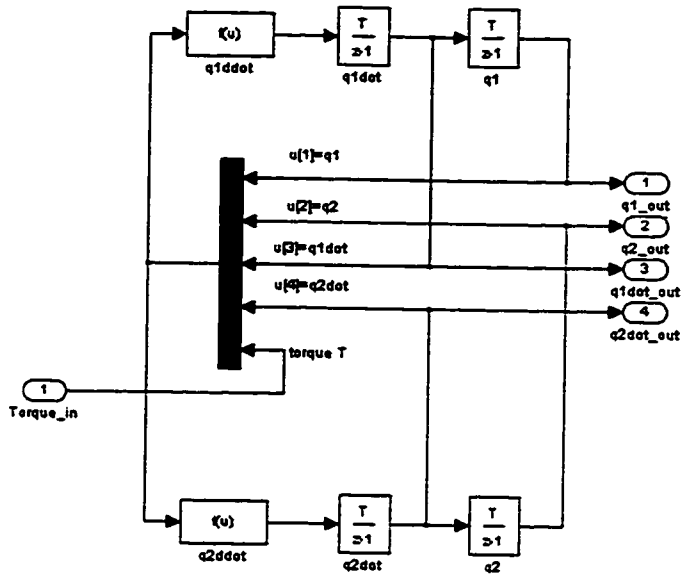


Figure 8.9 Discrete-time Pendubot model

8.5 Simulation Results

There are two kinds of simulation model: one is the continuous time model; the other is the discrete time model. Thus, we present the results of both of them. Since we use the digital controller in experiments, the sampling period is the same as that in the discrete time simulation model. Their control parameters are also approximate. Section 8.5.1 illustrates the simulation results for the continuous system. Section 8.5.2 illustrates the simulation results for the discrete system. All results include the positions of two links, their angular velocities, and the torque τ.

8.5.1 The Simulation Results for the Continuous System

Part 1: the swinging up and balancing the Pendubot

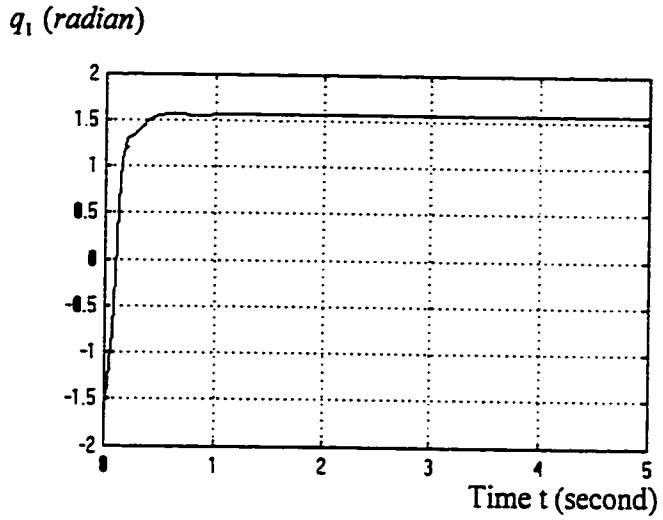


Figure 8.10 The output of q_1

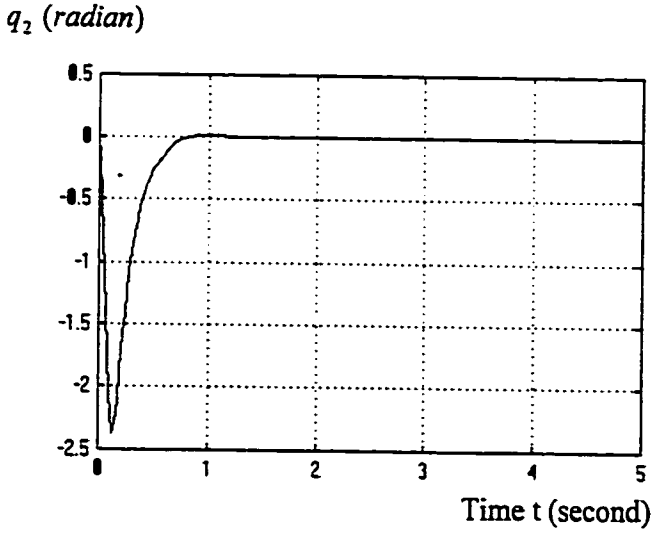


Figure 8.11 The output of q_2

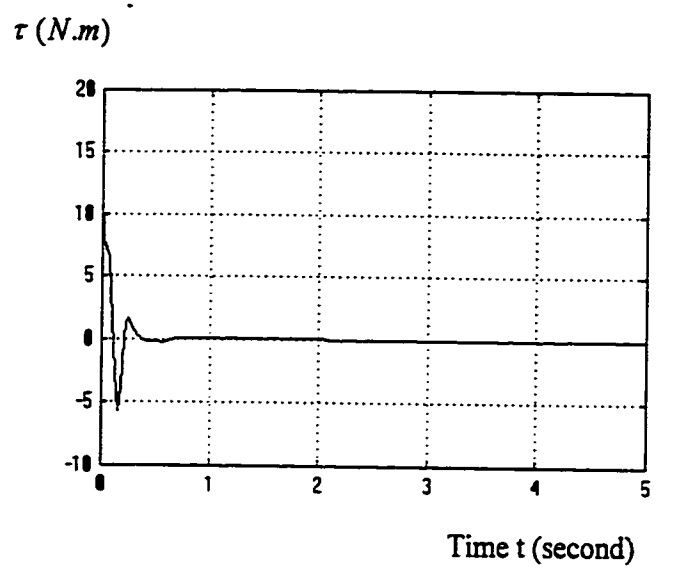


Figure 8.12 The output of torque τ

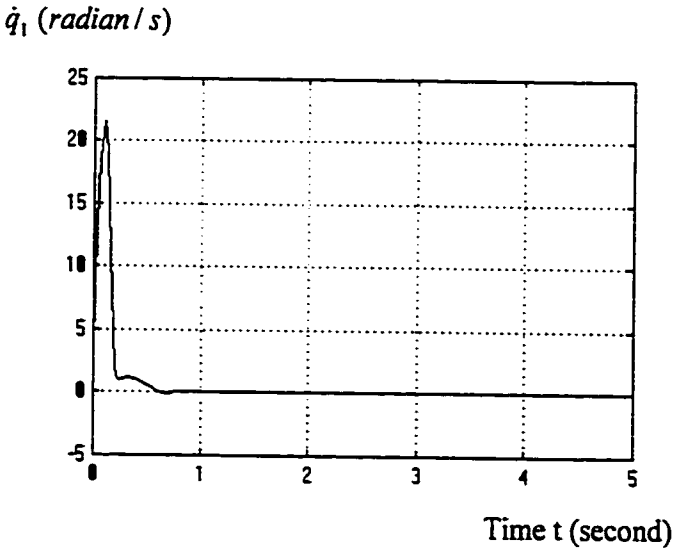


Figure 8.13 The angular velocity for link one

$\dot{q}_2$ $(radian/s)$

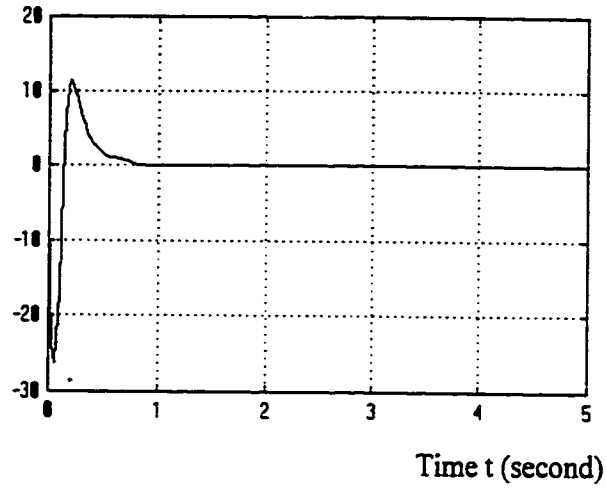


Figure 8.14 The angular velocity for link two

Part 2: Swinging up the Pendubot

τ $(N.m)$

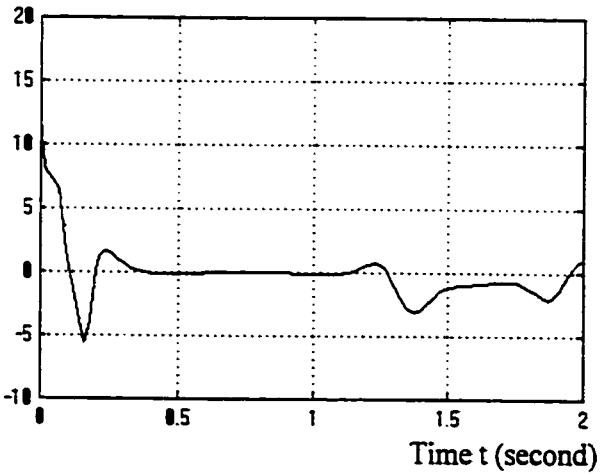


Figure 8.15 The output of τ (just swinging up)

q_1 (*radian*)

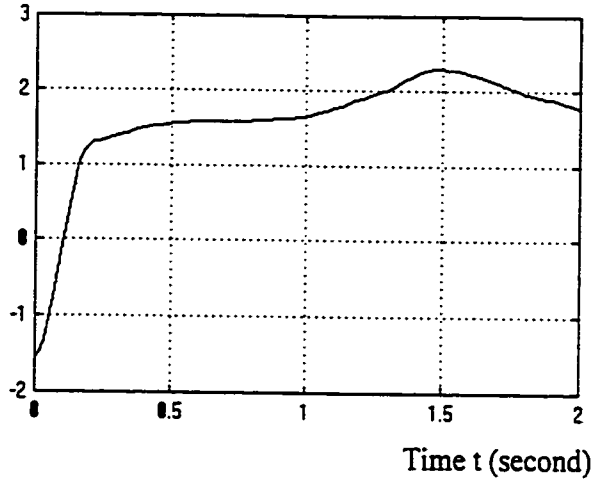


Figure 8.16 The output of q_1 (just swinging up)

q_2 (*radian*)

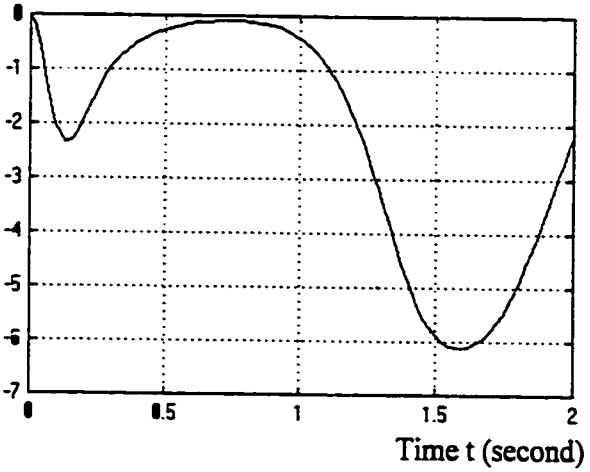


Figure 8.17 The output of q_2 (just swinging up)

8.5.2 The Simulation Results for the Discrete System

Part 1: the swinging up and balancing the Pendubot

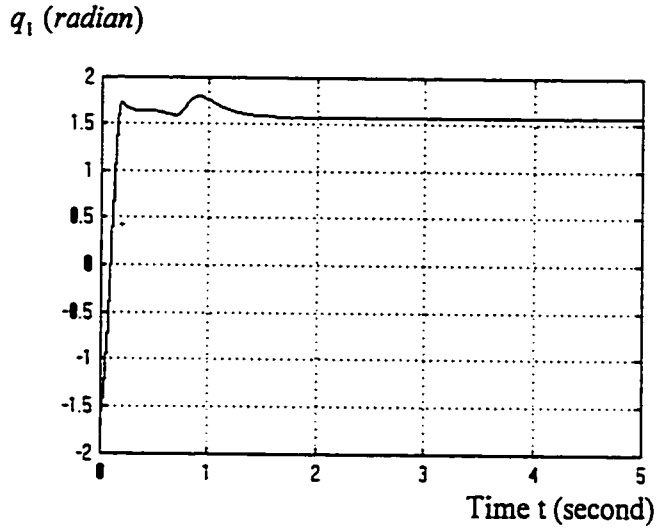


Figure 8.18 The output of q_1

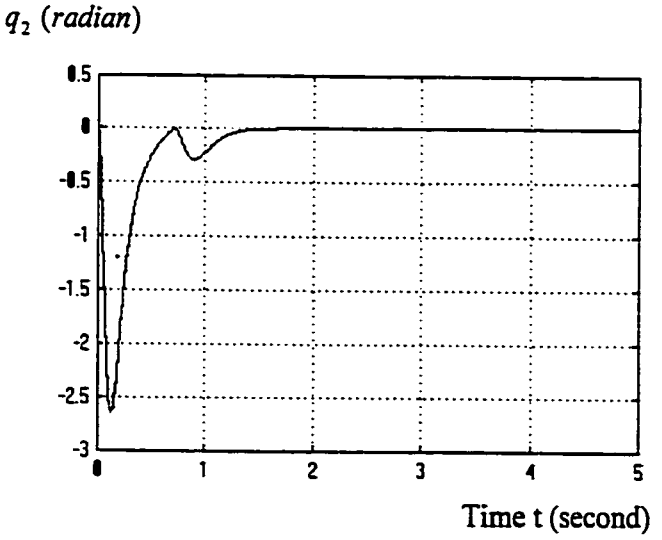


Figure 8.19 The output of q_2

τ $(N.m)$

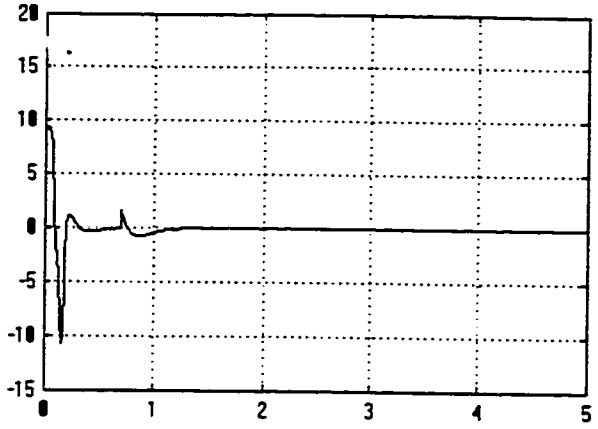


Figure 8.20 The output of torque τ

$\dot{q}_1$ $(radian/s)$

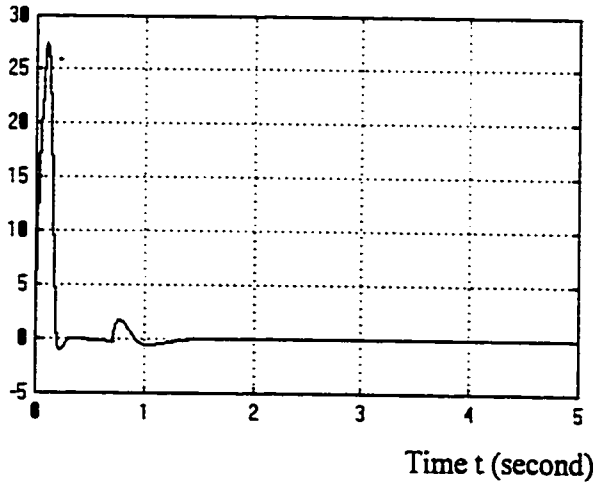


Figure 8.21 The angular velocity for link one

$\dot{q}_2$ $(radian/s)$

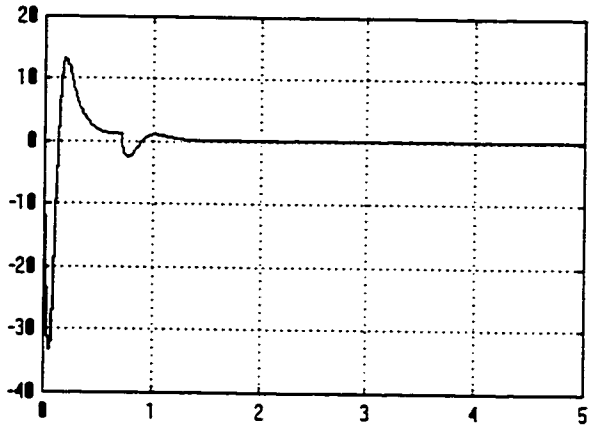


Figure 8.22 The angular velocity for link two

Part 2: Swinging up the pendubot

τ $(N.m)$

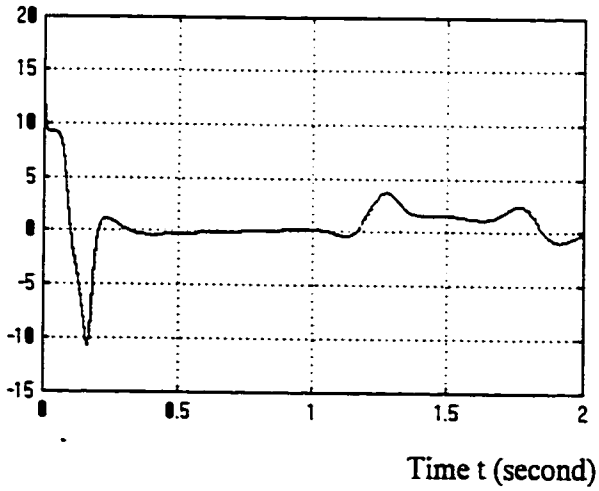


Figure 8.23 The output of τ (just swinging up)

q_1 (*radian*)

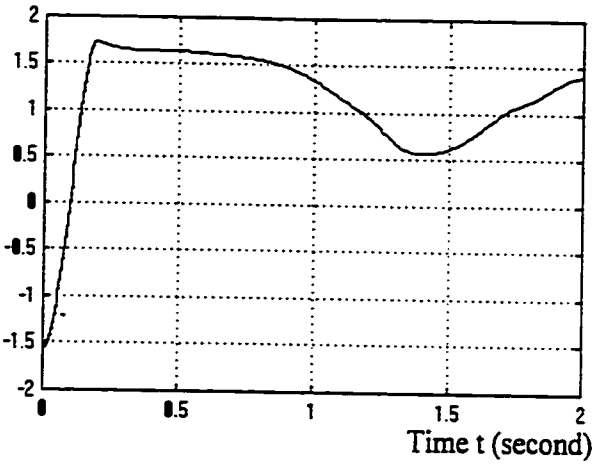


Figure 8.24 The output of q_1 (just swinging up)

q_2 (*radian*)

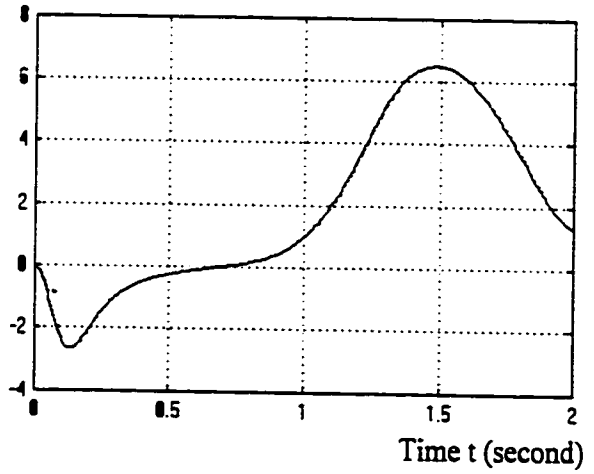


Figure 8.25 The output of q_2 (just swinging up)

Chapter 9

Real-time Experimental Tests

9.1 Introduction

This chapter presents the real-time experiment results of the fuzzy controllers, using the Pendubot Model P-2 as the test machine. The controller is the digital controller designed by the algorithms proposed in the above related chapters. In addition, we use a graphical user interface (GUI), named MSDEV.EXE, supplied by Mechatronic Systems, Inc., to simplify the task of running controllers, outputting the data file.

In our computer, we have installed Microsoft C++ 7.0 in the C:\MSI directory. I write the C program original code for our controller, and then we use command "msmake <filename>" to compile it. Consequently, I get the executive files named the same with "filename". Finally, I run real time controllers in the Windows 3.1 environments. In Windows 3.1 we create a PIF file for our controllers. Use the PIF file editor to create a pif file that

1. indicates the name of our executable,
2. runs in a full screen DOS box (Display Usage: Full Screen),
3. runs in exclusive mode (Execution: Exclusive) and
4. sets as the Start-Up Directory the directory where the executable is located.

In our experiment, I adjusted the controlling parameters in the controlling program. We find there is a little difference comparing with those in the simulation. However, this result show the difference between the actual Pendubot and the simulation model, and this is also discussed in the next chapter.

The experiment is divided to two parts. The first part is to design the swinging-up and balancing controller in order to swing up the Pendubot from the rest configuration ($q_1 = -\pi/2, \quad q_2 = 0$) and to balance it at the unstable position ($q_1 = \pi/2, \quad q_2 = 0$). The second part is to combine our swing up controller with the trajectory tracking controller (this controller is designed by Zheng Cai). The difference of switching algorithm between the two parts is just using the different limited value of ($q_1 - \pi/2$) and (q_2).

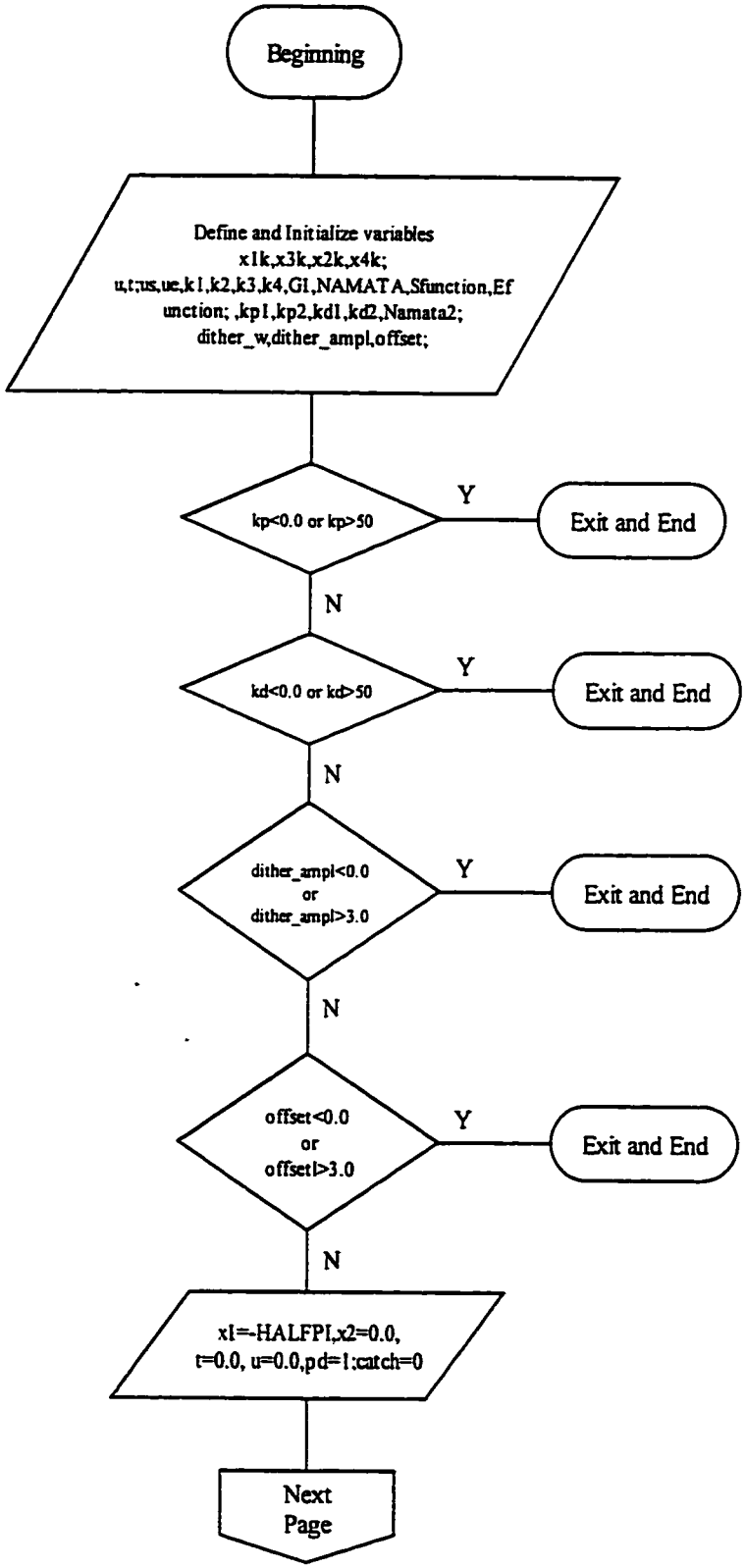
```
If (fabs(x1k-HALFPI)<0.1)
    If (fabs(x3k)<0.2)
    ...
```

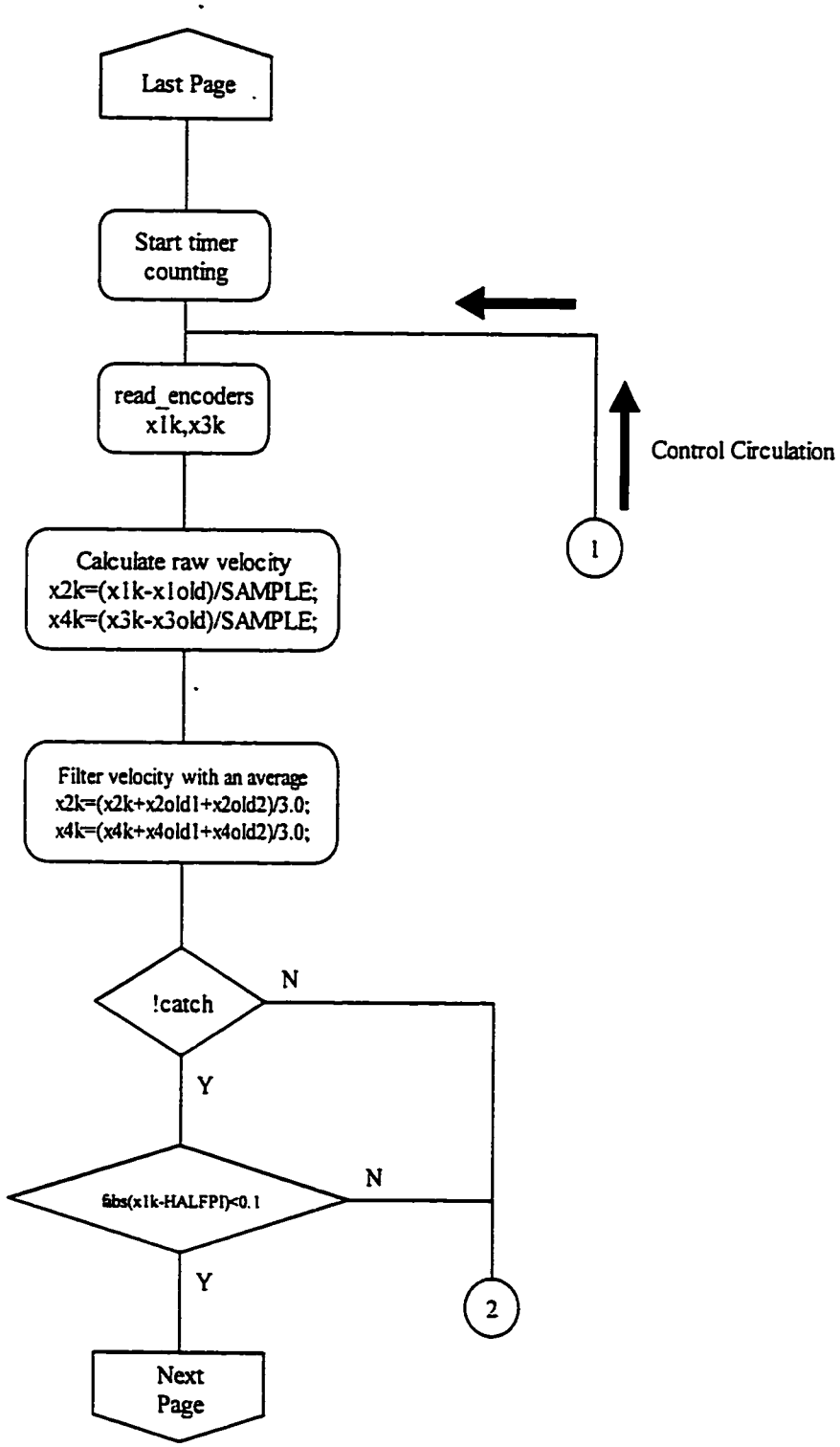
```
If (fabs(x1k-HALFPI)<0.7)
    If (fabs(x3k)<0.4)
    ...
```

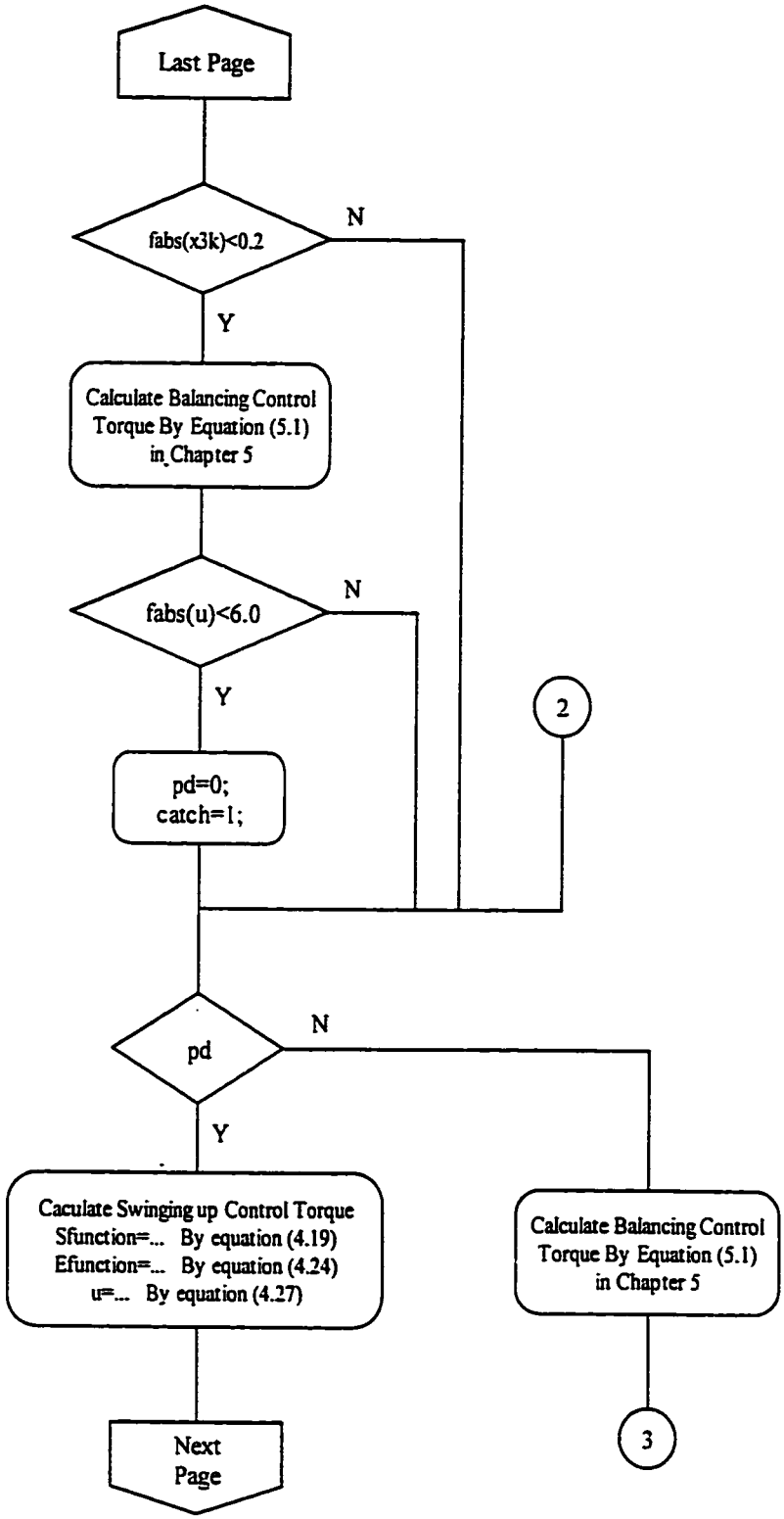
Section 9.2 is the flow chart of our controller. Section 9.3 describes the experimental results of the first part; plotted by M.file according to the output data. Section 9.4 describes the photographs of experiment for two parts.

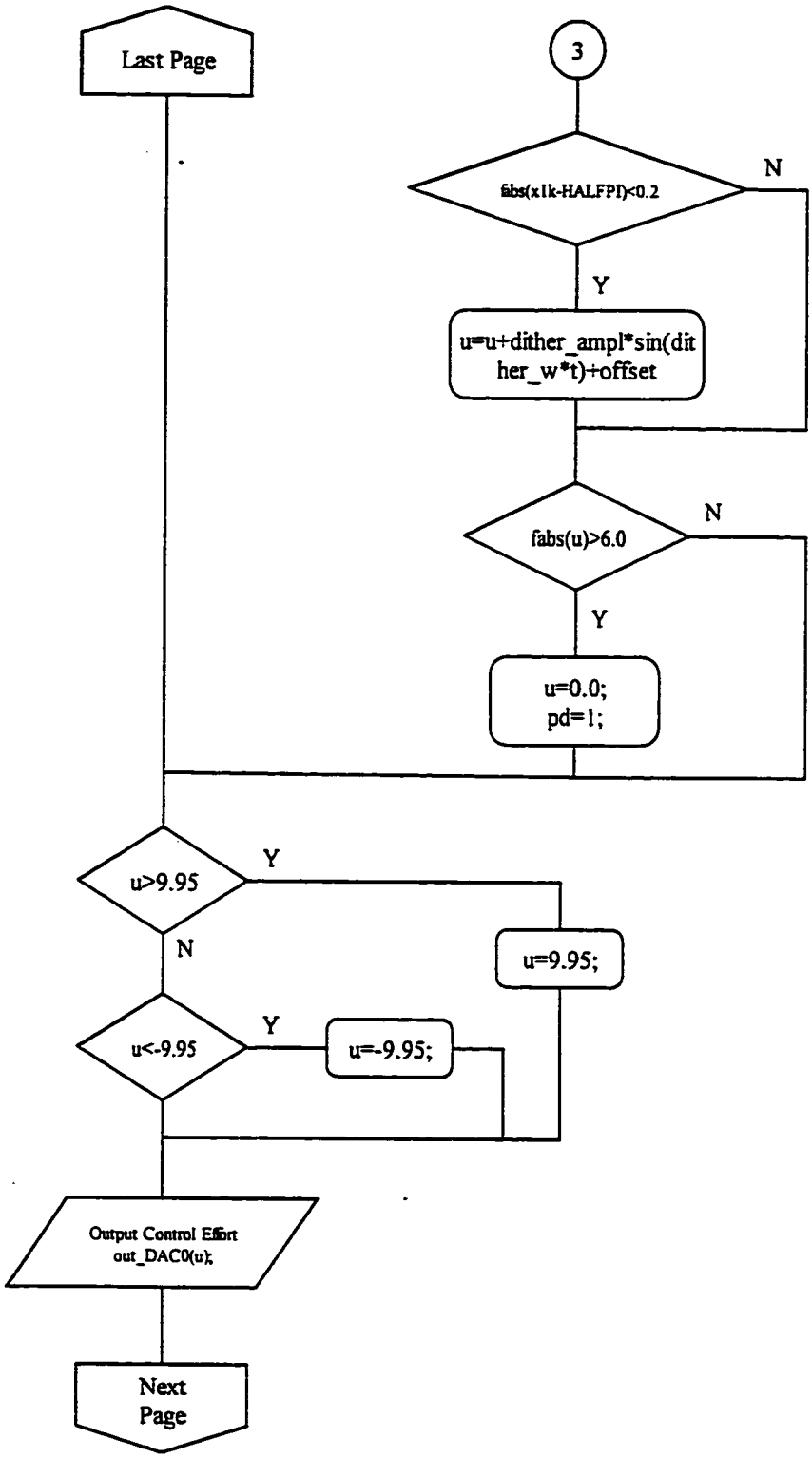
9.2 Flow Chart

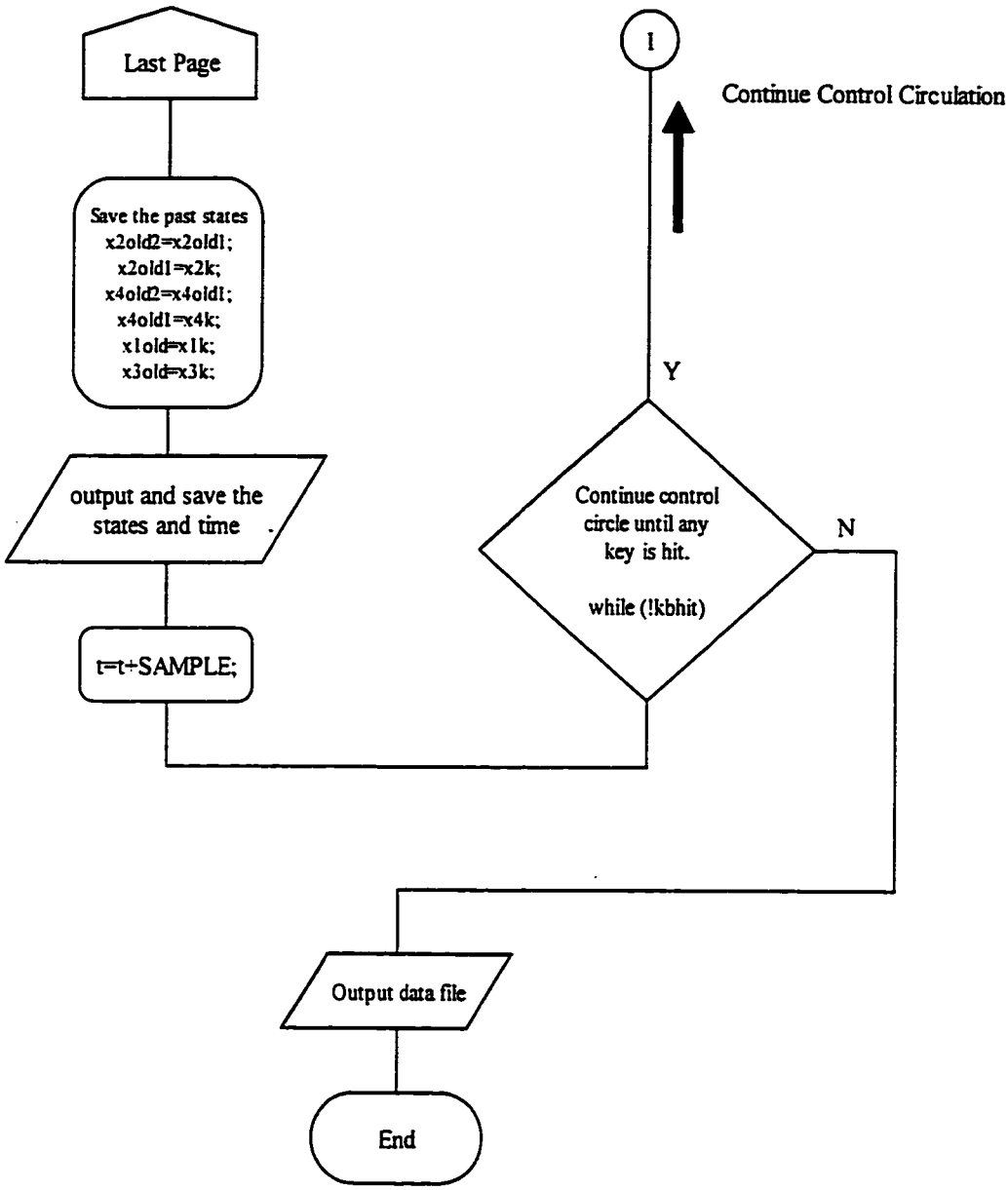
The next five pages is the flow chart of my controlling program.


Beginning
Define and Initialize variables
x1k,x3k,x2k,x4k;
u,t;us,ue,k1,k2,k3,k4,G1,NAMATA,Sfunction,Ef
unction; ,kp1,kp2,kd1,kd2,Namata2;
dither_w,dither_ampl,offset;
kp<0.0 or kp>50
Y
Exit and End
N
kd<0.0 or kd>50
Y
Exit and End
N
dither_ampl<0.0
or
dither_ampl>3.0
Y
Exit and End
N
offset<0.0
or
offset>3.0
Y
Exit and End
N
x1=-HALFPI,x2=0.0,
t=0.0, u=0.0,pd=1;catch=0
Next
Page


Last Page
Start timer counting
read_encoders x1k,x3k
Control Circulation
1
Calculate raw velocity
x2k=(x1k-x1old)/SAMPLE;
x4k=(x3k-x3old)/SAMPLE;
Filter velocity with an average
x2k=(x2k+x2old1+x2old2)/3.0;
x4k=(x4k+x4old1+x4old2)/3.0;
!catch
N
Y
fabs(x1k-HALFPI)<0.1
N
Y
2
Next Page


Last Page
fabs(x3k)<0.2
N
Y
Calculate Balancing Control Torque By Equation (5.1) in Chapter 5
fabs(u)<6.0
N
Y
2
pd=0;
catch=1;
pd
N
Y
Caculate Swinging up Control Torque
Sfunction=... By equation (4.19)
Efunction=... By equation (4.24)
u=... By equation (4.27)
Calculate Balancing Control Torque By Equation (5.1) in Chapter 5
Next Page
3


Last Page
3
fabs(x1k-HALFPI)<0.2
N
Y
u=u+dither_ampl*sin(dit
her_w*t)+offset
fabs(u)>6.0
N
Y
u=0.0;
pd=1;
u>9.95
Y
N
u=9.95;
u<-9.95
Y
u=-9.95;
Output Control Effort
out_DAC0(u);
Next
Page


Last Page
Save the past states
x2old2=x2old1;
x2old1=x2k;
x4old2=x4old1;
x4old1=x4k;
x1old=x1k;
x3old=x3k;
output and save the states and time
t=t+SAMPLE;
1
Continue Control Circulation
Y
Continue control circle until any key is hit.
while (!kbhit)
N
Output data file
End

9.3 Experimental Results for Swinging-up and Balancing

From the flow chart in the above section 9.2, we can see that, in each loop, the system saves the states of the Pendubot including angles (q_1, q_2), angular velocities ($\dot{q}_1, \dot{q}_2$), the torque (τ), and current time (t), and then output these states to the data file. When the digital controller stops, the system output the data file. Hereby, we can get one data file after we operate the controller one time. This data file is saved as M-file of MATLAB. Finally, we reprogram this data file and run it to plot the effect of each variable such as $\tau, q_1, q_2, \dot{q}_1, \dot{q}_2$.

Figure 9.1-9.5 shows the experimental results for swinging-up and balancing. Comparing with the simulation results of discrete system, shown in Figure 8.18-8.22, the torque of the system is close but not equal to zero; both links actually have a little dither, and the controller keeps them balancing at the unstable position.

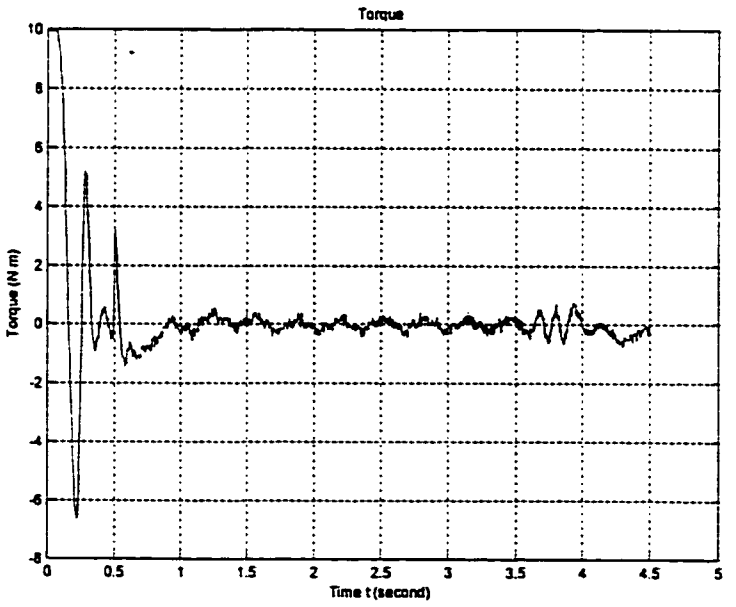


Figure 9.1 Effect of torque

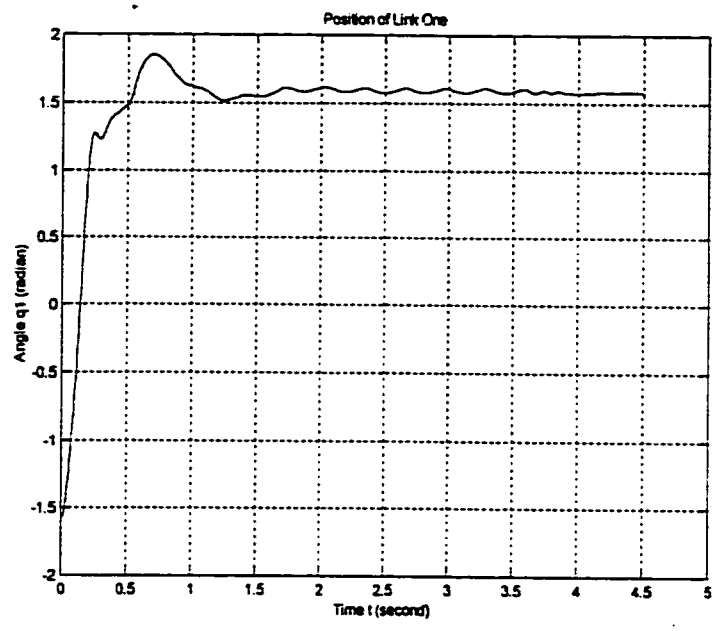


Figure 9.2 Effect of angle for link one

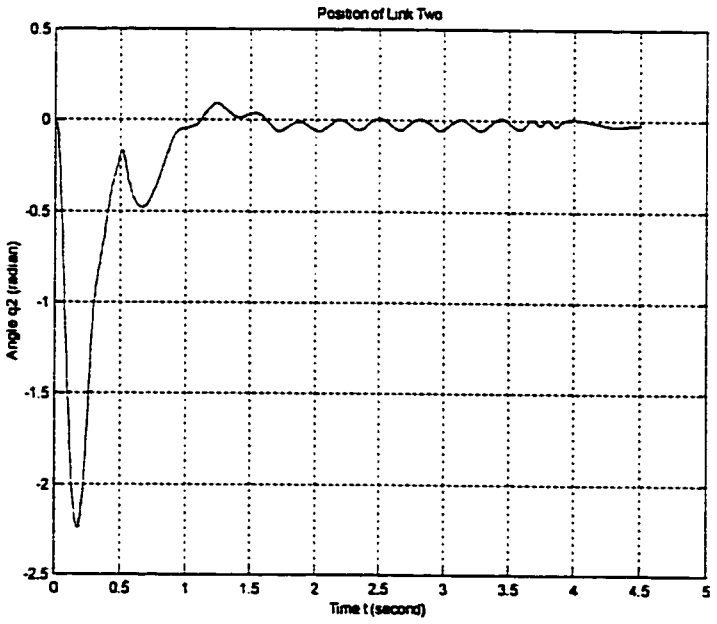


Figure 9.3 Effect of angle for link two

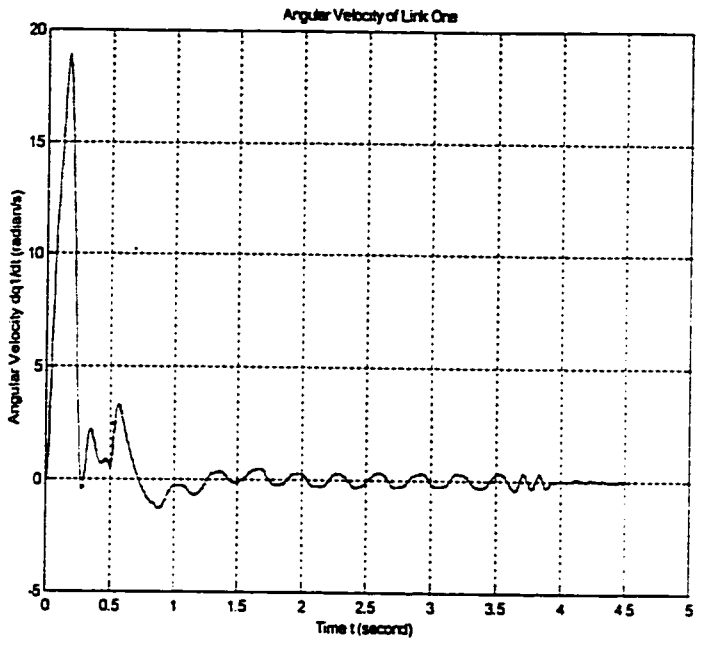


Figure 9.4 Effect of angular velocity for link one

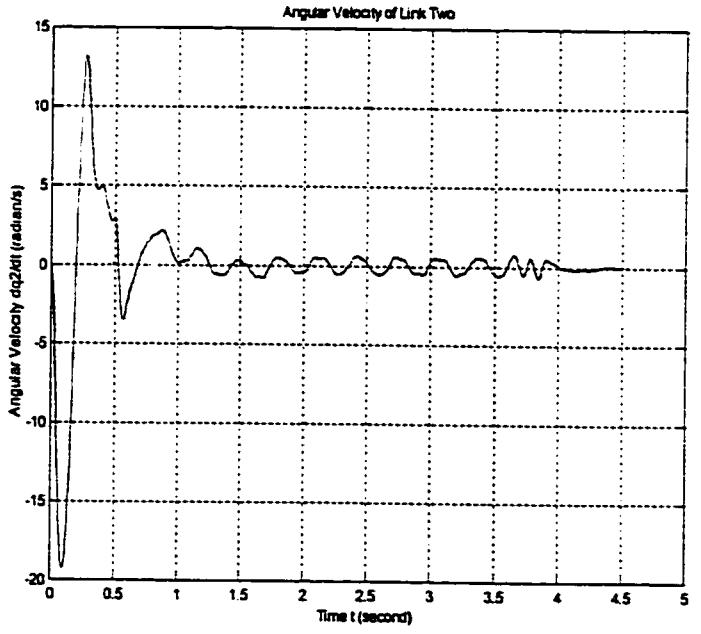


Figure 9.5 Effect of angular velocity for link two

9.4 Photographs of Experiments

Figure 9.6 illustrates the Pendubot at the rest configuration. Figure 9.7 illustrates the Pendubot balancing at the top unstable configuration. Figure 9.8 and 9.9 illustrate two positions while the Pendubot is tracking a desired trajectory.

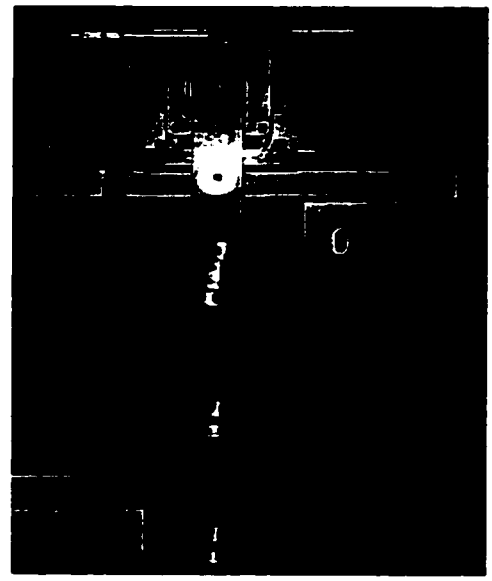


Figure 9.6 The Pendubot at the rest configuration

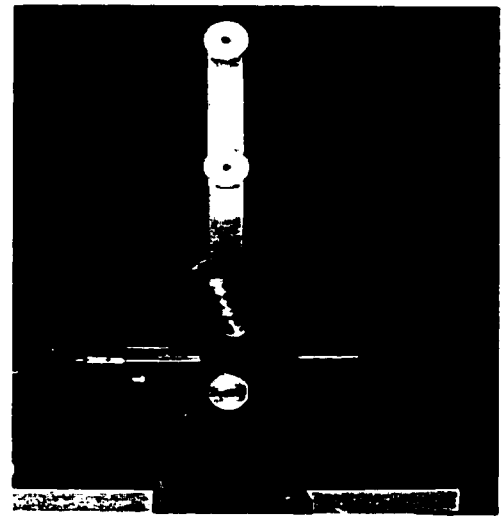


Figure 9.7 Balancing at the top configuration

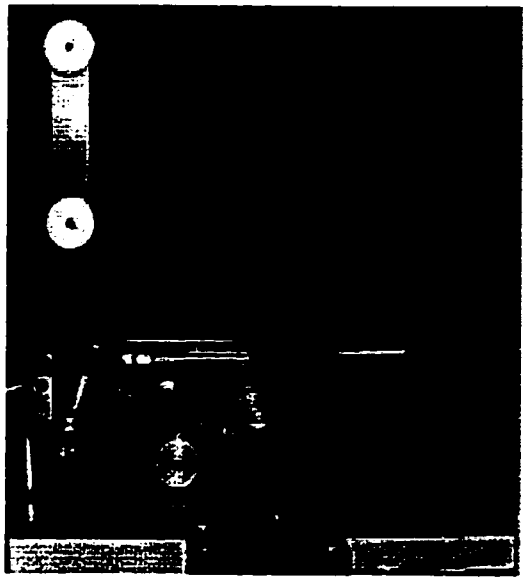


Figure 9.8 The Pendubot on the left side

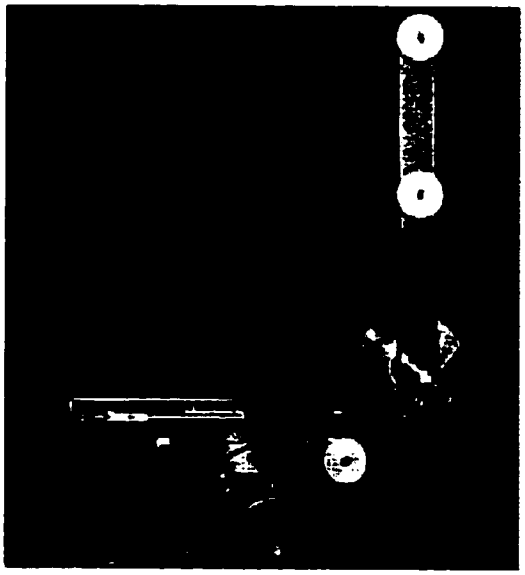


Figure 9.9 The Pendubot on the right side

Chapter 10

Conclusions and Recommendations for Future Work

10.1 Discussion

The study of under-actuated mechanisms is a topic of current research in the control community. These systems are characterized by the fact that they have fewer independent control actuators than degrees of freedom to be controlled. Different control schemes, derived from nonlinear systems theory [2] [49], classical control techniques [3] [5], or intelligent control methodologies [11] [50], have been proposed. From the recent research papers, we can find that the fuzzy control method attracts many researchers because of its robust characteristic.

In brief, the fuzzy control method used in the Pendubot, can be divided into several cases. First, the practical techniques based on genetic algorithms (GA), are introduce for the automatic tuning [50] [11]. The second case is to design a fuzzy PD controller for the under-actuated robot swing-up control [15] [51], and this method can automatically change the membership functions of the fuzzy PD scheme. The third case is to convert the nonlinear system to a set of linear local system model at each equilibrium point, based on the Takagi-Sugeno fuzzy methodology [14], and then they applied the linear control

method to control the system based on the fuzzy rule base. Comparing to the above methods, the thesis presents a two-rule-base fuzzy control scheme based on quasi-linear-mean aggregating operator and Tsukamoto's reasoning.

From another point of view, the research to the under-actuated mechanism includes the following aspects:

1. *Identification.* There are off-line or on-line, fuzzy method or classical method, energy method or optimization method.
2. *Swing up and Balance.* This is a set-point regulation, and I can design various controllers using various control methods, such as passivity or energy method, heuristic, fuzzy logic, partial feedback linearization, machine learning etc.
3. *Switching Control.* So-called logic based switching control is widely proposed: the logic condition may be the error of angle or the distance to the goal position [15]. I use the error of angle or use the switching time as the switching parameters.
4. *Trajectory Tracking.* This is a new study direct for the under-actuated mechanism. There is a little contribution besides the paper [14] [4].

10.2 Conclusions

In my study, a fuzzy scheme for swing-up control is proposed and implemented in the actual under-actuated mechanism --- the Pendubot. Comparing with the previous partial feedback linearization method, I find that, the swing-up speed is faster and the

stability is certainly good. This method is simpler than the classical control method; however, the tuning is a little difficult for the simulation or the experiment. The proposal method represents a new class of fuzzy controls. The real-time implementation also shows the success of the application and feasibility of the fuzzy control method to the nonlinear systems.

In my simulation, we start with the continuous time system, and then change it into the discrete time system. We conclude that the control parameters are different for these two control systems although the sampling time is 0.005. We also realize that the smaller the sampling time is, the more approximate the two different systems are. Comparing with the actual experimental model, the simulation model is only a mathematization model that ignores some existing factors such as friction and external disturbance.

Another importance part is the stability analysis of the fuzzy scheme for the Pendubot trajectory tracking. I verified the possibility and validity of a fuzzy state feedback control scheme and a fuzzy nonlinear regulation control method. The stability proof gives us the suggestion that we can apply linear control theory and fuzzy control method to study the nonlinear system.

Finally, in our experiment, digital computer plays the role of digital controller. From our experiment, I conclude that, for controlling systems with slow dynamics this will be indistinguishable from the performance give by a continuous controller,

but for systems that respond at higher frequencies I may encounter problems in achieving control. Therefore the cycle speed, or the sampling period, plays an important role in the design of the digital controller.

10.3 Recommendations for Future Work

1. Generalization

The fuzzy control scheme developed in this thesis has been successfully applied to the Pendubot. The Pendubot is an example of the under-actuated mechanics. Therefore, the future work is to generate this fuzzy control method to general under-actuated systems. Furthermore, in order to generalize this control method, we should design an automatic method to complete the controller's parameters tuning; for example, using computer program to search a best set of control parameters.

2. M.file Control

As we know, we start our research from the simulation using SIMULINK, and we can also program M_file code to simulate the control system. Furthermore, it is better for the discrete time system. According to the explanation of function of MATLAB [53] [54], M_files can be directly changed to the execute files. Therefore, we can use

the M_files to simulate the control system and directly to control the actual model: the Pendubot.

3. *Trajectory Tracking*

For the nonlinear system, trajectory tracking is a little more difficult than the set-point regulation. The research results [32] [29] for this aspect are limited, and the application to the Pendubot is spars [14]. Therefore, we are investigating to apply other modern control methods, except the linear regulation theory and T-S fuzzy methodology, to force the Pendubot to track a desired trajectory.

References

[1] Daniel Jerome Block, "Mechanical Design and Control of the Pendubot", M.S. Thesis, Department of General Engineering, University of Illinois at Urbana-Champaign, 1996.

[2] Mark W.Spong, "The Swing Up Control Problem For The Acrobot", *IEEE on Control Systems Magazine*, pp. 49-55, February 1995.

[3] Lsabelle Fantoni, Rogelio Lozano, and Mark W.Spong, "Passivity based control of the Pendubot", *Proceedings of the American Control Conference San Diego*, California, June 1999.

[4] Matthew D.Berkemeier and Ronald S.Fearing, "Tracking Fast Inverted Trajectories of the Underactuated Acrobot", *IEEE Transactions on Robotics and Automation*, vol.15, no.4, August 1999.

[5] Rafael Fierro, Frank L.Lewis, and Andy Lowe, "Hybrid Control for a Class of Underactuated Mechanical Systems", *IEEE Transaction on Systems*, Man and Cybernetics---Part A: Systems and Humans, vol.29, no.6. November 1999.

[6] Kazunobu Yoshida, "Swing-Up Control of an Inverted Pendulum by Energy-based Methods", *Proceedings of the American Control Conference*, San Diego, California, June 1999.

[7] Marcelo C.M., Teixeira, Stanislaw and H.Zak, "Stabilizing Controller Design for Uncertain Nonlinear Systems Using Fuzzy Models", *IEEE Transactions on Fuzzy Systems*, vol. 7, no.2, April 1999.

[8] H.K.Lam, F.H.F.Leung, and P.K.S.Tam, "Fuzzy State Feedback Controller for Nonlinear Systems: Stability Analysis and Design", *IEEE International Fuzzy Systems Conference Proceedings*, 2000.

[9] H.K.Lam, F.H.F.Leung, P.K.S.Tam, "An Improved Stability Analysis and Design of Fuzzy Control Systems", *IEEE International Fuzzy Systems Conference Proceedings*, 1999.

[10] Borislav Vidolov, Christian Melin, "An Approach to the design of MIMO Fuzzy Controllers in Cases of Incomplete Expert Knowledge", *IEEE International Fuzzy Systems Conference Proceedings*, 1998.

[11] Marcio A.T.de Sousa, and Marconi K.Madrid, "Optimization of Takagi-Sugeno Fuzzy Controllers Using a Genetic Algorithm", *IEEE International Fuzzy Systems Conference Proceedings*, 2000.

[12] Jianqiang Yi, Naoyoshi YUBAZAKI, and Kaoru HIROTA, "Systematically Constructing Stabilization Fuzzy Controllers for Single and Double Pendulum Systems", *IEEE International Fuzzy Systems Conference*, 2000.

[13] Ju-Jang Lee, "Robust Adaptive Control of an Underactuated Free-flying Space Robot under a Non-holonomic Structure in Joint Space". *International Journal of Systems Science*. vol. 27, no.11, pp 1113-1121, 1996.

[14] Ofelia Begovich, Edgar N.Sanchez and Marcos Maldonada, "Takagi-Sugeno Fuzzy Scheme for Real-Time Trajectory Tracking of an Underactuated Robot", *IEEE Transactions on Control Systems Technology*, November 1999.

[15] Edgar Sanchez, Luis A.Nuno, Ya-Chan Hsu, and Guanrong Chen, "Fuzzy PD Scheme for Underactuated Robot Swing-up Control", *IEEE International Fuzzy Systems Conference Proceedings*, 1998.

[16] Christian Melin, Borislav Vidolov, "A Fuzzy PD-like Scheme for Two Underactuated Planar Mechanisms", *IEEE International Fuzzy Systems Conference Proceedings*, 2000.

[17] Chun-Yi Su, and Yury Stepanenko, "Sliding Mode Control of Nonholonomic Mechanical Systems: Under-actuated Manipulator Case", *Nonlinear Control System Design Symposiums (NOLCOS)*, IFAC Prepringts, Tahac city, pp 609-613, 1995.

[18] B.Vidolov, C.Melin, "Fuzzy Control: Controllers Based on Two Rules", *International Fuzzy Systems and Intelligent Control Conference*, IFSICC –93, Louisville KY-USA, 1993.

[19] B.Vidolov, C.Melin, "Fuzzy Controllers Based on Two Rules – Study and Applications", *Proc. Journees Nationales sur les Applications des Ensembles Flous*, Nimes, 1993.

[20] Y.Tsukamoto, "An approach to fuzzy reasoning method", *Advances in Fuzzy Set Theory and Applications*, M/M Gupta, R.K.Ragade, and R.R.Yager Eds., Amsterdam, North-Hollamd, 1979.

[21] B.A.Francis, "The linear multivariable regulator problem", *SIAM.J.Control. Optimization*, vol.15, pp 486-505, 1977 .

[22] Zadeh L. A., " A theory of approximate reasoning", *Machine Intelligence Magazine*, vol. 9, Halstead Press, New York, pp 149-194, 1979.

[23] R.Kruse, J.Gebhardt, F.Klawonn, "Foundations of Fuzzy Systems", *John Wiley & Sons Ltd.*, 1993.

[24] M.M.Gupta, J.Bkiszka, G.M.Trojan, "Multivariable Structure of Fuzzy Control Systems", IEEE vol. SMC-16, no. 5, September/October 1986.

[25] C.W.de Silva, "A Criterion for Knowledge Base Decoupling in Fuzzy-Logic Control Systems", IEEE vol. SMC-24, no.10, October 1994.

[26] Gegov A., "Distributed Fuzzy Control of Multivariable Systems", *Kluwer Academic Publication*, 1996.

[27] R.R.Yager, and D.P.Filev "Essentials of Fuzzy Modeling and Control", *John Wiley & Sons, Inc.*, 1994.

[28] J.Fodor, and M/Roubens, "Fuzzy Preference Modeling and Multicriteria Decision Support", *Kluwer Academic Publishers*, 1994.

[29] A.Isidori, "Nonlinear Control Systems", *3rd ed. Springer-Verlag*, 1995.

[30] T.Takagi and M.Sugeno, "Fuzzy identification of systems and its applications to modeling and control", *IEEE Trans. Systems, Man and Cybernetics*, vol.15, no.1, pp 116-132, 1985.

[31] K.Tanaka and M.Sugeno, "Stability analysis and design of fuzzy control system", *Fuzzy Sets System Magazine*, vol.45, no.2, pp 133-156, 1992.

[32] A. Isidori, Christopher I. Byrnes, "Output Regulation of Nonlinear Systems", *IEEE Transactions On Automatic Control*, vol.35, no.2, February 1990.

[33] H.O.Wang, K. Tanaka, and M.F.Griffin, "An approach to fuzzy control of nonlinear systems: stability and the design issues", *IEEE Trans. Fuzzy System*, vol. 4, no. 4, pp 14-23, February 1996.

[34] John J.Craig, "Introduction to Robotics Mechanics and Control", *Second Edition, Copyright © 1989,1986 by Addison-Wesley Publishing Company, Inc.*, 1989.

[35] Mark W.Spong, F.L.Lewis, C.T.Abdallah, "Robot Control: Dynamics, Motion Planning and Analysis", *ISBN 0-7803-0404-7, © 1992 by the Institute of Electrical and Electronics Engineers, IEEE Order Number: PC0299-8*, 1992.

[36] M.W.Spong and M.Vidyasagar, "Robot Dynamics and Control", *New York: Wiley*, 1989.

[37] K.Kreutz, "On manipulator control by exact linearization", *IEEE Trans. Auto. Cont.*, vol.34, no.7, pp 763-767, July 1989.

[38] H. Goldstein, "Classical Dynamics", *MA: Addison-Wesley*, 1950.

[39] Nazareth S. Bedrossian, "Nonlinear Control of an Underactuated Two-Link Manipulator", *Proceedings of the American Control Conference*, San Francisco, California, June 1993.

[40] John Hauser, Richard M. Murray, "Nonlinear controllers for non-integrable systems: the Acrobot example", *IEEE Transactions on Automatic Control*, WP 65:00, 1993.

[41] You-Liang Gu, "A Direct Adaptive Control Scheme for Under-Actuated Dynamic Systems", *IEEE Transactions on Automatic Control, Proceedings of the 33rd Conference on Decision and Control*, December 1993.

[42] Y.L.Gu and Y.S.Xu, "A normal Form Augmentation Approach to Adaptive Control of Space Robot Systems", *Proc. 1993 IEEE International Conference on Robotics and Automation*, pp 731-737, Atlanta, GA, May 1993.

[43] Yann Blanco, Wilfrid Perruquetti, and Pierre Borne, "Relaxed stability conditions for Takagi-Sugeno's fuzzy models", *IEEE International Fuzzy Systems Conference Proceedings*, 2000.

[44] Lotfi A. Zadeh, Abraham Kandel, and Gideon Langholz, "Fuzzy Control Systems", *ISBN 0-8493-4496-4, ©1994 by CRC Press, Inc*, 1994.

[45] Witold Pedrycz, "Fuzzy Control and Fuzzy Systems", *Second Edition, ISBN 0-471-93475-5, ©1993 by Research Studies Press Ltd*, 1993.

[46] Kudret Demirli, "Notes of Fuzzy Sets & Fuzzy Logic", *Department of Mechanical Engineering, Concordia University*, 1999.

[47] Michael Margaliot and Gideon Langholz, "Adaptive Fuzzy Controller Design via Fuzzy Lyapunov Synthesis", *IEEE International Fuzzy Systems Conference Proceedings*, 1998.

[48] G.S.Virk, "Digital Computer Control Systems", *ISBN 0-07-067512-0, First published 1991 by Macmillan Education Ltd., ©1991 by G.S.Virk*, 1991.

[49] L.E.Ramos, B.Castillo-Toledom, and J. Alvarez, "Nonlinear Regulation of an underactuated robot", *Proc. 1997 IEEE Intl. Conference in Robotics and Automation*, pp 3288-3293, Albuquerque, New Mexico, USA, April 1997.

[50] M.A.Lee, and M.H.Smith, "Automatic Design and Tuning of a Fuzzy System for Controlling the Acrobot using Genetic Algorithms, DSFS, and Meta-Rule Techniques", *Proc. MAFIPS'94*, San Antonio, Texas, December 1994.

[51] Y. Hsu, G.Chen, and E.N.Sanchez, "A Fuzzy PD controller for multi-link robot control: Stability Analysis", *Proc. 1997 IEEE Intl. Conference in Robotics and Automation*, pp 1412-1417, Albuquerque, New Mexico, USA, April 1997.

[52] R.J.Schilling, "Fundamentals of Robotics: Analysis and Control", *Englewood Cliffs, NJ: Prentice-Hall*, 1990.

[53] WeiGou Cheng, "The Guide to the Application of MATLAB 5.3", *ISBN 7-115-08268-5/TP.1430*, 1999.

[54] Wei Gou Cheng, "Programming and Advanced Application of MATLAB 5.3", *ISBN 7-111-08012-2*, 2000.

[55] H.Arai and S.Tachi, "Position control of a manipulator with passive joints using dynamic coupling", *IEEE Trans. On Robotics and Automation*, vol.7, pp 528-534, 1991.

[56] H.Arai, "Controllability of a 3-DOF manipulator with a passive joint under a nonholonomic constraint", *Proc. of the 1996 IEEE Int. Conf. on Robotics and Automation*, pp 3707-3713, 1996.

[57] G.Oriolo and Y.Nakamura, "Control of Mechanical Systems with Second-order Nonholonomic Constraints: Under-actuated Manipulators", *Proc. of the 30th Conf. on Decision and Control*, pp 2398-2403, 1991.

[58] K.Y.Wichlund, O.J.Sordalen, and O.Egeland, "Control Properties of Under-actuated Vehicles", *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp 2009-2014, 1994.

[59] N.S.Bedrossian, "Nonlinear Control of an Under-actuated Two-link Manipulator", *Proc. of the 32nd Conference on Decision and Control*, pp 2234-2238, 1993.

Appendix A: Linearized equations

$$f_a(x,u) = f_a(x_r,u_r) + \frac{\partial f}{\partial x}|_{x_r,u_r}(x-x_r) + \frac{\partial f}{\partial u}|_{x_r,u_r}(u-u_r) \tag{A.1}$$

$$\begin{aligned} u_r &= \theta_4 g\cos(x_{r1}) \\ x_{r1} + x_{r3} &= \pi/2 \end{aligned} \tag{A.2}$$

$$\frac{\partial f}{\partial x} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{\partial f_2}{\partial x_1} & 0 & \frac{\partial f_2}{\partial x_3} & 0 \\ 0 & 0 & 0 & 1 \\ \frac{\partial f_4}{\partial x_1} & 0 & \frac{\partial f_4}{\partial x_3} & 0 \end{bmatrix} \tag{A.3}$$

with

$$\frac{\partial f_2}{\partial x_1} = \frac{g[2\theta_2\theta_4\sin(x_{r1}) - \theta_3\theta_5\sin(x_{r1}) - \theta_3\theta_5\sin(x_{r1}+2x_{r3})]}{2\theta_1\theta_2 - \theta_3^2 - \theta_3^2\cos(2x_{r3})} \tag{A.4}$$

$$\frac{\partial f_2}{\partial x_3} = \frac{-\theta_3^2\sin(2x_{r3})\left[\theta_2 u_r - g\theta_2\theta_4\cos(x_{r1}) + \frac{g\theta_3\theta_5\cos(x_{r1})}{2}\right] - \theta_3^2\sin(2x_{r3})\left[\frac{g\theta_3\theta_5\cos(x_{r1}+2x_{r3})}{2}\right]}{\left[\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}\right]^2}$$

$$+\frac{2g\theta_3\theta_5\sin(x_{r1}+2x_{r3})}{-2\theta_1\theta_2 + \theta_3^2 + \theta_3^2\cos(2x_{r3})} \tag{A.5}$$

$$\frac{\partial f_4}{\partial x_1} = \frac{g\theta_5\sin(x_{r1}+x_{r3})[\theta_1+\theta_2+2\theta_3\cos(x_{r3})] - g[\theta_2+\theta_3\cos(x_{r3})][\theta_4\sin(x_{r1})+\theta_5\sin(x_{r1}+x_{r3})]}{\left[\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}\right]} \tag{A.6}$$

$$\frac{\partial f_4}{\partial x_3} = \frac{\theta_3^2 \sin(2x_{r3})[g\theta_5 \cos(x_{r1}+x_{r3})[\theta_1+\theta_2+2\theta_3\cos(x_{r3})]]}{\left[\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}\right]^2}$$

$$- \frac{\theta_3^2 \sin(2x_{r3})[\theta_2+\theta_3\cos(x_{r3})][-u_r + g\theta_4\cos(x_{r1}) + g\theta_5\cos(x_{r1}+x_{r3})]}{\left[\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}\right]^2}$$

+

$$\frac{\left[\frac{g\theta_3\theta_4\sin(x_{r1}-x_{r3})}{2}\right] + \theta_3 u_r \sin(x_{r3}) - \left[\frac{g\theta_3\theta_4\sin(x_{r1}+x_{r3})}{2}\right] + g\theta_1\theta_5\sin(x_{r1}+x_{r3}) + g\theta_2\theta_5\sin(x_{r1}+2x_{r3})}{\left[\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}\right]} \qquad \text{(A.7)}$$

$$\frac{\partial f}{\partial u} = \begin{bmatrix} 0 \\ \frac{\partial f_2}{\partial u} \\ 0 \\ \frac{\partial f_4}{\partial u} \end{bmatrix} \qquad \text{(A.8)}$$

with

$$\frac{\partial f_2}{\partial u} = \frac{\theta_2}{\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}} \qquad \text{(A.9)}$$

$$\frac{\partial f_4}{\partial u} = \frac{-\theta_2 - \theta_3\cos(x_{r3})}{\theta_1\theta_2 - \frac{\theta_3^2}{2} - \frac{\theta_3^2\cos(2x_{r3})}{2}} \qquad \text{(A.10)}$$