IMPROVING PID CONTROLLER OF MOTOR SHAFT ANGULAR POSITION
BY USING GENETIC ALGORITHM

ARIF ABIDIN B MUHAMAD

A project report submitted in partial fulfillment of the requirement for the award of the Degree of Master of Electrical Engineering

Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn Malaysia

JUNE, 2015
ABSTRACT

This study represents Genetic Algorithm optimization of PID parameters gain in model reference robust control system structure for desired position of incremental servomotor. Experiments had been took out via Lab-Volt 8063 Digital Servo system equipment at Servo Control Laboratory. The key issue for PID controllers is the accurate and efficient tuning of parameters. The plant repeatedly has a problem in achieving the desire position control and system performance have an oscillatory response and gives a slightly steady state error. This problem among other is affected by existing the nonlinearities component in the system, the system communication noise, and not optimize PID parameter. The existing PID controller tuning with the help of the offline Genetic Algorithms approach comprises of automatically obtaining the best possible outcome for the three parameters gain (K_p, K_i, K_d) for improving the steady state characteristics and performance indices. Their step responses are then compared with a tuned conventional Ziegler-Nichols based PID controller. This paper explores the well established methodologies of the literature to realize the workability and applicability of Genetic Algorithms for process control applications. At last, a comparative study done between ZN-PID experiment and GA-PID experiment shows that the GA optimal controller is highly effective and outperforms the PID controller in achieving an enhancing the output transient response with improvement percentage of rise time is 91.83%, settling time is 89.36% and maximum overshoot is 82.24%. The robust and automatic gains parameter calculator; GA based PID technique also proven to be time savers as they are much faster to be conducted than ZN method which is basically based on trial-and-error in getting the best PID values before the system can be narrowed down in getting the closest to the optimized value.
Kajian ini mengenai kaedah mengoptimalkan parameter PID dalam kawalan posisi motor servo di model rujukan untuk sistem kawalan teguh menggunakan Algoritma Genetik. Eksperimen telah dijalankan menggunakan sistem peralatan Lab-Volt 8063 Digital Servo yang terletak di Makmal Kawalan Servo. Isu permasalahan utama bagi pengawal jenis PID adalah ketepatan dan keberkesanan talaan nilai parameter gandaan. Sistem Lab-Volt itu sering bermasalah dalam kawalan untuk mencapai posisi sasaran dengan prestasi sistem mempunyai sambutan ayunan dan memberikan ralat keadaan mantap yang kecil. Masalah ini antara lain dipengaruhi oleh parameter komponen tidak selari dalam sistem, gangguan dalam sistem komunikasi, dan parameter PID yang tidak optimum. Pengawal PID yang sedia ada dibantu oleh Algoritma Genetik secara berasingan untuk mendapatkan nilai optimum bagi ketiga-tiga parameter gandaan (Kp, Ki, Kd) untuk meningkatkan kestabilan dan prestasi tindak balas. Prestasi tindak balas itu kemudian dibandingkan dengan pengawal PID berasaskan penalaan kaedah konvensional iaitu Ziegler-Nichols. Kajian ini meneroka kaedah yang telah terbukti untuk merealisasikan keupayaan Algoritma Genetik dalam aplikasi pengawalan sesuatu proses. Di akhirnya, satu kajian kes perbandingan yang dilakukan di antara keputusan eksperimen ZN-PID dan GA-PID menunjukkan bahawa pengawal optimum GA adalah sangat berkesan dan lebih berkeupayaan terhadap pengawal PID dengan meningkatkan tindak balas sambutan dengan peratus peningkatan masa menaik adalah 91.83%, masa selesai adalah 89.36% dan peratus lajakan maksimum adalah 82.24%. Bertindak sebagai kalkulator parameter gandaan yang teguh dan automatik; operasi GA-PID juga terbukti menjimatkan masa berbanding ZN-PID yang pada asasnya berdasarkan kaedah cubajaya untuk mendapatkan nilai PID terbaik sebelum sistem boleh ditumpukan untuk mendapatkan parameter yang paling dekat dengan nilai dioptimumkan.
TABLE OF CONTENTS

TITTLE *v*

DECLARATION *ii*

DEDICATION *iii*

ACKNOWLEDGEMENT *iv*

ABSTRACT *v*

ABSTRAK *vi*

TABLE OF CONTENTS *vii*

LIST OF TABLES *xii*

LIST OF FIGURES *xiii*

LIST OF SYMBOLS AND ABBREVIATIONS *xvii*

LIST OF APPENDICES *xviii*

CHAPTER 1 INTRODUCTION *1*

1.1 DC Servomotor - An Overview *1*

1.2 Motor Shaft Angular Position Control - An Overview *1*

1.3 Proportional Integral Derivative Controller - An Overview *2*

1.4 Ziegler-Nichols Based PID Controller - An Overview *2*

1.5 Genetic Algorithm as Optimal Robust Controller –
Overview

1.6 Background of Project 3

1.7 Problem Statement 5

1.8 Project Objectives 5

1.9 Project Scope 6

1.10 Project Limitation 6

1.11 Significant of Study 7

CHAPTER 2 LITERATURE REVIEW 9

2.1 Introduction 9

2.2 Servo Motor Incremental Shaft Encoder 9

2.2.1 Position Sensing 9

2.2.2 Simplified Incremental Shaft Encoder 10

2.3 Motor Shaft Angular Position Control 12

2.3.1 Angular Position Control Block Diagram and Fundamentals 12

2.4 Servo Response Terminology 14

2.5 Standard PID control 15

2.6 ZN Based PID Terminology 16

2.7 GA Terminology 18

2.8 GA for Optimization 21

2.9 GA Software 23

2.10 Experimental Model: Lab-Volt Digital Servo Model 8063 24

2.10.1 Servo System Hardware 24

2.10.2 Servo System Software 28
2.11 Case Studies on the Previous Researches

2.11.1 Title: Nonstationary Optimal Positioning Controller Design Using GA Meta-Optimization

2.11.2 Title: Tuning PID Controller Using Adaptive Genetic Algorithms

2.11.3 Title: Dc Motor Control Using Ziegler Nichols and Genetic Algorithm Technique

2.11.4 Title: Modeling and Control of Three Phase BLDC Motor using PID with Genetic Algorithm

2.11.5 Title: Position Control of DC Motor Using Genetic Algorithm Based PID Controller

2.11.6 Robust Genetic Algorithm Approach to Modelling and Control

2.11.7 Parameter Optimization of PID Controllers Based on Genetic Algorithm

2.11.8 Analysis and Summary of Case Studies on the Previous Researches

2.11.9 Project Hypothesis based on Previous Researches Result

CHAPTER 3 METHODOLOGY

3.1 Introduction

3.2 Project Development

3.3 Project Concept

3.4 Lab-Volt Motor Characteristics
CHAPTER 4 RESULT AND ANALYSIS

4.1 Introduction
4.2 Results Experiment for the ZN Based PID Controller
4.3 Results Simulation for the ZN Based PID Controller
4.4 Result Simulation for the GA Based PID Controller
4.5 Result Experiment for the GA Based PID Controller
4.6 Comparative Study between ZN and GA Performances

CHAPTER 5 CONCLUSION AND RECOMMENDATION

5.1 Introduction
5.2 Conclusion 79
5.3 Recommendation for Future Works 80

REFERENCES 82

APPENDICES 85
APPENDIX A: MATLAB Code for GAOT 86
APPENDIX B: Gantt Chart for Master Project 1 Planning 101
APPENDIX C: Gantt Chart for Master Project 2 Planning 102

VITA
### LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Effects of PID parameters variations on a step response</td>
<td>17</td>
</tr>
<tr>
<td>2.2</td>
<td>Specifications of the Lab-Volt Digital Servo Model 8063</td>
<td>25</td>
</tr>
<tr>
<td>2.3</td>
<td>Summary of case studies</td>
<td>32</td>
</tr>
<tr>
<td>3.1</td>
<td>Servo motor characteristics for a brush-type permanent-magnet dc motor</td>
<td>38</td>
</tr>
<tr>
<td>3.2</td>
<td>Initialization Parameters</td>
<td>53</td>
</tr>
<tr>
<td>4.1</td>
<td>Analysis of GA based PID simulation</td>
<td>72</td>
</tr>
<tr>
<td>4.2</td>
<td>Comparative study between ZN based PID by experiment and simulation</td>
<td>74</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparative study between GA based PID by experiment and simulation</td>
<td>75</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparative study between ZN-PID simulation and GA-PID simulation</td>
<td>76</td>
</tr>
<tr>
<td>4.5</td>
<td>Comparative study between ZN-PID experiment and GA-PID experiment</td>
<td>77</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Direct Digital Control (DDC) schematic</td>
<td>8</td>
</tr>
<tr>
<td>2.1</td>
<td>Simplified incremental encoder diagram</td>
<td>11</td>
</tr>
<tr>
<td>2.2</td>
<td>Quadrature signals A and B</td>
<td>11</td>
</tr>
<tr>
<td>2.3</td>
<td>Block diagram of a position-control servo system with motor shaft incremental encoder</td>
<td>13</td>
</tr>
<tr>
<td>2.4</td>
<td>Simplified position control servo system block diagram with overall scaling factor</td>
<td>13</td>
</tr>
<tr>
<td>2.5</td>
<td>Simplified position-control servo system block diagram</td>
<td>13</td>
</tr>
<tr>
<td>2.6</td>
<td>A typical stable position response</td>
<td>15</td>
</tr>
<tr>
<td>2.7</td>
<td>Expended PID controller block diagram.</td>
<td>16</td>
</tr>
<tr>
<td>2.8</td>
<td>Flow chart of GA functions and process</td>
<td>19</td>
</tr>
<tr>
<td>2.9</td>
<td>Representation of the executed operations during a generation</td>
<td>21</td>
</tr>
<tr>
<td>2.10</td>
<td>GA for optimization</td>
<td>22</td>
</tr>
<tr>
<td>2.11</td>
<td>Lab-Volt Digital Servo Model 8063</td>
<td>24</td>
</tr>
<tr>
<td>2.12</td>
<td>The mechanical unit</td>
<td>26</td>
</tr>
<tr>
<td>2.13</td>
<td>The servo controller</td>
<td>27</td>
</tr>
<tr>
<td>2.14</td>
<td>Flywheel coupled to motor shaft, platform unloaded</td>
<td>27</td>
</tr>
<tr>
<td>2.15</td>
<td>Default-position control screen</td>
<td>28</td>
</tr>
<tr>
<td>3.1</td>
<td>Flowchart of project development</td>
<td>36</td>
</tr>
<tr>
<td>3.2</td>
<td>Block diagram of project system design</td>
<td>37</td>
</tr>
<tr>
<td>3.3</td>
<td>Quarter amplitude decay response</td>
<td>40</td>
</tr>
<tr>
<td>3.4</td>
<td>Chromosome representation</td>
<td>42</td>
</tr>
<tr>
<td>3.5</td>
<td>Simulation flow chart for the computation of GA-PID controller parameters</td>
<td>43</td>
</tr>
<tr>
<td>3.6</td>
<td>Defining system transfer function</td>
<td>44</td>
</tr>
<tr>
<td>3.7</td>
<td>Code for initialize the GA</td>
<td>45</td>
</tr>
<tr>
<td>3.8</td>
<td>Code for set the parameters</td>
<td>46</td>
</tr>
<tr>
<td>3.9</td>
<td>Code for performing the GA</td>
<td>47</td>
</tr>
</tbody>
</table>
3.10 Code to perform the objective function 48
3.11 Code for ITAE implementation 49
3.12 Code to ensure stabilization of the controlled system 49
3.13 Code for initialization function used by ga.m 50
3.14 Code for termination function used by ga.m 50
3.15 Code for selection function used by ga.m 51
3.16 Code for crossover operator used by ga.m 51
3.17 Code for mutation operators used by ga.m 52
3.18 Code for utility function 52
4.1 Illustration of exported data file 55
4.2 The pulses generated by position sensing incremental encoder for Step 1 55
4.3 Effect of the proportional gain on the Step response 56
4.4 The pulses generated by position sensing incremental encoder for Step 2 57
4.5 Tuning the controller with the ZN method step response 57
4.6 The pulses generated by position sensing incremental encoder for Step 3 (t_d = 70ms) 58
4.7 Quarter amplitude decay Step response with t_d = 70ms step response 58
4.8 The pulses generated by position sensing incremental encoder for Step 4 59
4.9 Significantly damped step response 60
4.10 M-file for the plant system with PID controller 61
4.11 Performance response for ZN based PID controller simulation 62
4.12 Convergence of the generations with population size 20 64
4.13 Optimum solution of the PID chromosomes with population size 20 64
4.14 Performance of the optimized PID chromosomes with population size 20 64
4.15 Convergence of the generations with population size 40 65
4.16 Optimum solution of the PID chromosomes with
4.17 Performance of the optimized PID chromosomes with population size 40 65
4.18 Convergence of the generations with population size 50 66
4.19 Optimum solution of the PID chromosomes with population size 50 66
4.20 Performance of the optimized PID chromosomes with population size 50 66
4.21 Convergence of the generations with population size 60 67
4.22 Optimum solution of the PID chromosomes with population size 60 67
4.23 Performance of the optimized PID chromosomes with population size 60 67
4.24 Convergence of the generations with population size 70 68
4.25 Optimum solution of the PID chromosomes with population size 70 68
4.26 Performance of the optimized PID chromosomes with population size 70 68
4.27 Convergence of the generations with population size 80 69
4.28 Optimum solution of the PID chromosomes with population size 80 69
4.29 Performance of the optimized PID chromosomes with population size 80 69
4.30 Convergence of the generations with population size 90 70
4.31 Optimum solution of the PID chromosomes with population size 90 70
4.32 Performance of the optimized PID chromosomes with population size 90 70
4.33 Convergence of the generations with population size 100 71
4.34 Optimum solution of the PID chromosomes with population size 100 71
4.35 Performance of the optimized PID chromosomes with population size 100 71
4.36 The pulses generated by a position sensing incremental encoder for $K_p 97.33601$, $K_i 99.6854$, $K_d 1.81466$ gain by offline GA  

4.37 The step response of $K_p 97.33601$, $K_i 99.6854$, $K_d 1.81466$ gain by offline GA  

5.1 The block diagram of proposed real time position control based on LabView program
# LIST OF SYMBOLS AND ABBREVIATIONS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGA</td>
<td>adaptive genetic algorithms</td>
</tr>
<tr>
<td>DAQ</td>
<td>Data acquisition</td>
</tr>
<tr>
<td>DAS</td>
<td>Data Acquisition System</td>
</tr>
<tr>
<td>DC</td>
<td>Direct Current</td>
</tr>
<tr>
<td>DDC</td>
<td>Direct Digital Control</td>
</tr>
<tr>
<td>DSP</td>
<td>Digital Signal Processing</td>
</tr>
<tr>
<td>F(x)</td>
<td>fitness function</td>
</tr>
<tr>
<td>GA</td>
<td>Genetic Algorithm</td>
</tr>
<tr>
<td>GAOT</td>
<td>Genetic Algorithm Optimization Toolbox</td>
</tr>
<tr>
<td>$\mu$</td>
<td>population size</td>
</tr>
<tr>
<td>$g_{\text{max}}$</td>
<td>maximum number of generations</td>
</tr>
<tr>
<td>$K_d$, $T_d$</td>
<td>Derivative Gain</td>
</tr>
<tr>
<td>$K_i$, $T_i$</td>
<td>Integral Gain</td>
</tr>
<tr>
<td>$K_p$</td>
<td>Proportional Gain</td>
</tr>
<tr>
<td>$K_U$</td>
<td>ultimate gain</td>
</tr>
<tr>
<td>NOR</td>
<td>nonstationary optimal regulator</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PID</td>
<td>Proportional integral derivative</td>
</tr>
<tr>
<td>$p_c$, $p_m$</td>
<td>probabilities</td>
</tr>
<tr>
<td>$P_k$</td>
<td>$k$-th population</td>
</tr>
<tr>
<td>$P_{k+1}$</td>
<td>new population</td>
</tr>
<tr>
<td>$P_g$</td>
<td>population</td>
</tr>
<tr>
<td>$P_{g+1}$</td>
<td>next population</td>
</tr>
<tr>
<td>$P_0$</td>
<td>first population</td>
</tr>
<tr>
<td>ZN</td>
<td>Ziegler-Nichols</td>
</tr>
</tbody>
</table>
# LIST OF APPENDICES

<table>
<thead>
<tr>
<th>APPENDIX</th>
<th>TITLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>MATLAB Code for GAOT</td>
<td>86</td>
</tr>
<tr>
<td>B</td>
<td>Gantt Chart for Master Project 1 Planning</td>
<td>101</td>
</tr>
<tr>
<td>C</td>
<td>Gantt Chart for Master Project 2 Planning</td>
<td>102</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

This chapter introduces the project that has been working out. The important overview of the project background and description including the problem statements, project objectives, project scopes and significant of this project have emphasized in this chapter.

1.1 DC Servomotor– An Overview

A servomechanism is an automatic device that uses error-sensing feedback to correct the error in the mechanism. A servo motor, which is a type of servomechanism, is provided with a sensor (e.g., an incremental encoder, a position potentiometer, a speed sensor) that compares the command (e.g., the applied voltage) with the actual movement (e.g., the motor position) (Ruano, 2005). Using a controller and appropriate control strategies, the error existing between the command and the actual movement can be determined, analysed, and then corrected. Servo motors are used more and more because they give much more precision and/or rapidity to the movements of a mechanical system. A robot manipulator, for example, usually contains many servo motors.

1.2 Motor Shaft Angular Position Control – An Overview

Position control system is a system that converts a position input command to a position output response besides it extensively usage in industrial application such as robotics and drive control. Modern position control systems are achieved using optical incremental encoder sensors (Ruano, 2005). This is mainly due to high
reliability (no moving parts contact each other) and low cost. It is essentially a position transducer that reports the angle of shaft displacement in discrete steps.

1.3 Proportional Integral Derivative Controller- An Overview

Proportional integral derivative (PID) control schemes are widely utilized in most of control system for a long time for its easy in structure, reliable operating and robust in performances function. One key issue for their success is that they act within the processes in a manner closely like human’s natural responses to outside stimuli, that is the combined effects of naturalness (proportional action), post training (integral action) and projection into future (derivative action) (Stephen, 2011). However, it is still a awfully necessary downside a way to determine or tune the PID parameters, as a result of these parameters have an excellent influence on the stability and therefore the performance of the control system. Yet, the plants with high nonlinearity, high time-delay and high order can’t be controlled effectively employing a simple PID controller.

1.4 Ziegler-Nichols Based PID Controller – An Overview

Ziegler-Nichols (ZN) is one of the most widely used method for the tuning of the PID controller gains, yet this method is limited for application till the ratio of 4:1 for the first and the second peaks in the closed loop response, thus leading towards an oscillatory response (Astrom, Hagglund, 1995). Initially, the unit step function i.e. transfer function of the plant is obtained, and by the ZN rule base, the parameters required can easily be estimated, but far from optimal.

1.5 Genetic Algorithm as Optimal Robust Controller - An Overview

Genetic Algorithm (GA) is optimization methods, which operate on a population of points, designated as individuals. Each individual of the population represents a attainable solution of the optimization problem. Individual are evaluated relying
upon their fitness. The fitness indicates how well an individual of the population solves the optimization problem. One of the popular approaches to the mathematics-based approach to optimal design of a control system has been optimal robust control, in which an objective function, often based on a norm of a functional, is optimized, whereas a controller (dynamic or static) is obtained that may tolerate variation of plant parameters and unordered dynamics (Stephen, 2011).

1.6 Project Background

Digital servomotor has been widely known to be employ in associate automation and industrial because of its excellent speed control characteristics. Servo systems contain error-driven control loops. Servo tuning is an associate integral a part of any motion system and directly impacts the accuracy and performance, which driven a properly tuned system to give higher precision and additional stability. A system is considered stable if the particular position is finite once the commanded position is finite. By mean, a system is stable if a commanded position results in the motor coming to rest at a single position. In another ways, system is considered unstable when any commanded position typically results in an exponential increase in position error. By mean a system is unstable when the attempts to achieve a position result in oscillations that never dampen. Hence, the whole system performance strongly depends on the controller efficiency and will cause the tuning process plays a key role in the system behaviour. In this work, the servo systems will be analysed, specifically the positioning control systems.

Despite a huge advances in the field of control systems engineering, the fact that the algorithm provides an adequate performance in the vast majority of applications has helped PID to be still remains the foremost common control algorithm in industrial use nowadays. PID controllers are wide used for speed and position control of servo motor. Due to some constraints in the step response of the PID tuning, a lot of strategies regarding PID controller parameter tuning have been proposed; basically on (1) Empirical methods, such as Ziegler-Nichols methods, (2) Analytical methods, for instance, the root locus based techniques, (3) Methods based on optimization, such as GA methods.
In this project, Lab-Volt 8063 Digital Servo system equipment had been used as a platform to create an optimal robust controller strategy for the linear control systems. GA during this project is used to seek out an optimal gain automatically for robust controller for linear control systems. This project had been done on this technique to examine the implication of exploitation GA in a system and to obtain best result in tuning position control of a servo motor. GA will be applied to the area of PID optimisation in an off-line tuning environment. Tuning a system off-line means that the PID parameters of the controller are updated when the system has been taken off-line. The PID values are updated using the systems input and output data after the system has been placed offline. These updated PID values are used in place of the old PID values. This process continues till the optimum PID coefficients of the system in question have been obtained.

The first important issue in designing a control system is the consideration of stability. A control system is stable if and only if all roots of the characteristic equation are placed in the left half of the s-plane. If its real parts are negative, it displays absolute stability. According to the Hurwitz test, the absolute stability of a control system can be tested by means of the coefficients of the characteristic equation, without calculation of the exact position of the roots of the characteristic equation. The goal of the control, despite disturbance \( \delta(t) \) acting on the plant, is to keep the value of the controlled variable (the output variable) \( y(t) \) within tolerance of the value given by the reference variable (set-point) \( r(t) \) (Lab-Volt Ltd., 2010). Classical strategies for controller design use a nominal model of the plant. The robustness of the control loop is indicated by the parameters: phase margin and gain margin. The determination of appropriate controller parameters depends on the requirements of the control system. Typical requirements are: short settling time, small overshoot, good damping or small value of the squared error surface.

End of the GA optimization process, the rise and the settling times and the overshoot are compared with classically tuned system ZN method corresponding system performance. Simulation results should show the effectiveness on damping and robustness of proposed GA controller to provide the angular position control of servomotor incremental encoder Lab-Volt 8063 Digital Servo system equipment.
1.7 Problem Statements

The positioning systems are normally unstable when they are implemented in a closed loop configuration, so when the controller is introduced into the closed loop it needs to be effectively tuned. The key issue for PID controllers is the accurate and efficient tuning of parameters. Whether the user is a relative novice, or an experienced hand to handle the parameters set up is a stressful job especially for some serious uncertain systems. Typically, the adjusting of the controller parameters is carried out using trial-and-error formulas to provide a performance, which, although not deficient, is far from optimal. In this study case of motor shaft angular position control on the Lab-Volt 8063 Digital Servo plant, the servomotor controller repeatedly has a problem in achieving the desired position control and system performance have an oscillatory response, damped and gives a slightly steady state error. In this report GA are proposed as a method for PID optimisation of nonlinear systems and compared with those of traditional ZN tuning method.

1.8 Project Objectives

The main objective of this study is to run motor operation of the Lab-Volt 8063 Digital Servo plant to ideal conditions. Hence, the specific objectives of this study are:

(i) To observe motor shaft angular position control in digital servo tuning.
(ii) To compute the existing plant controller stability.
(iii) To analyse the effect of the ZN tuning method and GA tuning method on the transient operation and damping on the step response of a servo positioning system used for linear position control.
(iv) To minimize the following error of a servo system by analyse of proportional, integral and derivative tuning action on the linear position control systems.
(v) To achieve the indicator performance specifications for optimize controller by; (i) Steady-state error <1% , (ii) Overshoots <1% , (iii) Settling-time < 2 seconds , (iv) Rise time < 2 seconds.
1.9 **Project Scope**

By using the Lab-Volt 8063 Digital Servo System as platform for this study, the scopes of this project are as follows:

(i) Run test on the equipment to carry out an experiment of angular position control by using the classical ZN method tuning.

(ii) Simulate and the plant system with PID controller using MATLAB Simulink.

(iii) Develop GA algorithm file and simulate the plant system optimization tuning control using MATLAB Simulink.

(iv) Analyse the effect of the ZN tuning method and GA tuning method on the transient operation, damping on the step response and following error of a servo positioning system used for linear position control.

1.10 **Project Limitation**

At the beginning of this project, the plan of this study was to prepare an online optimization PID controller by help of GA that will control the system plant in an independent control strategies by modifying the existence ones. Unfortunately, several problems had arise; (i) The processes done were unable to validate the interface of LabView and MATLAB Simulink, (ii) Due to inadequate expertise in understanding on how to interface personal computer (PC) with position control system through Data Acquisition System (DAS) where experimental on an actual system could not been carried out, (iii) Some of the plant technical specifications are not available for the public study, hence this create some limitation of references while investigating problems on the DAQ, (iv) Researcher in control discipline in real industry field still waiting for high computer processing capability, high Digital Signal Processing (DSP) communicator, and applications software fast computation for accurate response and quick solutions for real time optimization control system developing which is still in research and development stages.

Thus, this project is completed by resolving using offline simulation method only. The result of this study may not reflect the actual performance of the positional of DC motor system. Thus this study may not represent the robust controller and the
plant itself as it is only a simulation in MATLAB software using mathematical modelling prediction.

1.11 Significant of Study

Eventough the existing PID controller at the plant have a simple structure, it is quite challenging to find the optimized PID gains. Developing optimal control system will monitor the input and output signal of the plant and then will generate control signal so as to minimise the effect of parameter variation and at the same time to track the reference input (to avoid a following error). In other word, the optimal control system will automatically change its behaviour to accommodate the changes in the dynamics of the process and disturbances.

Since this earlier aim of this project was to embed a self-online optimal GA based PID controller into the plant are put off due to the limitations, but this project still give a important stage as part of the project development. Today industrial users are looking for a controller technology with flexibility to tailor near any system to specific needs using convenient software based control, as they call Direct Digital Control (DDC). DDC is basically a microprocessor based technology in which a controller performs closed loop functions via sophisticated algorithms and strategies. The controller function handles inputs and outputs electronically while the software provides the logic. Figure 1.1 shows the schematic of DDC.
Figure 1.1: Direct Digital Control (DDC) schematic

DDC control brings speed, precision, flexibility to control functions at low cost. It can range from simple control of a single loop to the application of larger systems controlling a multitude of loops. Controllers may function as stand-alone units or be networked into systems using a PC as a host to provide additional functions.
CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter will discuss on system model of servo motor incremental encoder, angular position control, robustness, model uncertainty and genetic optimal control to realize designed control. The equipment specification that used in this study will enclose. It also highlights some previous case studies that are related to this project.

2.2 Servo Motor Incremental Shaft Encoder

The most common implementation of shaft encoders today in position servos is the optical incremental type (Stephen, 2011). This is mainly due to high reliability (no moving parts contact each other) and low cost. It is essentially a position transducer that reports the angle of shaft displacement in discrete steps. These encoders give only relative position information, so absolute references are needed in any real machine. These are known usually as “home” or “limit” sensors.

2.2.1 Position Sensing

The incremental decoders generate pulses as the encoders internal shaft-mounted disc rotates. The encoder can be coupled to a servo motor or to a pulley that is part of a belt-driven conveyor. Two signals in quadrature (90 degrees out of phase) from the encoder, generate four transition signals that are totaled to determine the position as the motor shaft (Lab-Volt Ltd, 2003).
The movement direction is determined using the phase relationship generated by the 90-degrees (in quadrature) out-of-phase signal pairs. If the first signal pulse leads in relation to the second, then movement is determined to be in a given direction. If the first signals pulse lags behind the second, movement occurs in the opposite direction.

When the direction is determined, the LVServo software can calculate the platform position by decoding the quadrature pulses to either add or to subtract (depending on the direction) from the pulse total. To calculate the position in this way, however, a reference point must first be set. This can be done by means of a third reference that is generated once per encoder revolution. In the Digital Servo system, a command button from the computer interface resets the incremental encoder to 0. Because of this need reference point, robots that make use of incremental encoders for sensing joint position need to be set high an initial reference point for each joint. This process is called homing the robot.

2.2.2 Simplified Incremental Shaft Encoder

The incremental encoder described below is a simplification of actual incremental encoders. This simplification is useful to understand the basic concept underlying the operation of any incremental encoders.

Imagine a disc with 16 holes penetrating the circumference as shown in Figure 2.1 (Lab-Volt Ltd, 2003). Located at the outside edge of the disc is an assembly with a pair of photo transmitting devices such as leds, and photo receiving devices such as photo transistors. The device pairs will be labeled A and B.
Figure 2.2 shows what happens as the disc rotates. When the rotation is in a counter-clockwise direction, photo receiver A initially senses the light signal, then both A and B, followed by B only, and finally neither A nor B. In a clockwise direction, the sequence is reversed. Both signals A and B generate four-edge transitions that can be totaled to indicate the disc position. Thus, one revolution of the disc produces 64 pulses (16 x 4).
2.3 **Motor Shaft Angular Position Control**

A DC motor position control finds wide applications in servo systems, especially in aerospace, automotive and mechatronics applications. The position of a DC motor can be controlled by controlling the Armature Voltage or Field Voltage. The following are few of the simple methods of controlling a DC motor shaft position (Thomas, Poongodi, 2009):

(i) Open loop - without feedback of current position, method is ineffective and inaccurate in the presence of load disturbance.

(ii) On-off controller - motor is turned on with maximum torque till it reaches set point and switches off, this may result in overshoots and oscillations.

Single directional control reaches set point in a single direction only, the angular position of a DC motor can also be controlled by varying the torque generated by varying the armature voltage or field voltage.

2.3.1 **Angular Position Control Block Diagram and Fundamentals**

The motor shaft incremental encoder generates 4000 counts per revolution (Lab-Volt Ltd, 2003). The digital servo system default range for position sensing measurement is ±5000 counts, which is the ±100% position. A 100% position travel is thus equivalent to 1.25 motor shaft revolution (5000/4000). An angular position of 90 degrees, for example, is equivalent to 1000 counts ((90/360)x4000). The resulting position is percentage is 20% ((1000/5000)x100).

Figure 2.3 shows the Digital Servo positioning system first order block diagram. The controller is set to use proportional action only, which means that the controller gain $K_c$ is equivalent to the proportional gain $K_p$. 
A simplified block diagram for the proportional only position control servo system with the motor shaft incremental encoder is shown at Figure 2.4 below:

For analysis purposes, a further simplification can be made by combining the scaling factor 5.56, controller gain $K_C$, and the speed constant $K$ into the term $K'$. Figure 2.5 shows the resulting diagram:
2.4 Servo Response Terminology

The first objective of tuning is to stabilize the system. The formal definition of system stability is that when a bounded input is introduced to the system, the output of the system is also bounded. Meaning that to a motion control system is that; if the system is stable, then when the position setpoint is a finite value, the final actual position of the system is also a finite value. On the other hand, if the system is unstable, then no matter how small the position setpoint or how little a disturbance (motor torque variation, load change, noise from the feedback device, etc.) the system receives, the position error will increase continuously, and exponentially in almost all cases. In practice, when the system experiences instability, the actual position will oscillate in an exponentially diverging fashion. One common perception shared by many is that whenever there is oscillation, the system is unstable. However, if the oscillation finally diminishes (damps out), even if it takes a long time, the system is still considered stable.

While investigating the plot of the position response versus time, there are a few measurements that can be considered as the quantitatively assess the performance of the servo:

(i) Overshoot - the measurement of the maximum magnitude that the actual position exceeds the position setpoint. It is usually measured in terms of the percentage of the setpoint value.

(ii) Rise Time - the time it takes of the actual position to pass the setpoint.

(iii) Settling Time - the time between when the commanded position reaches the setpoint and the actual position settles within a certain percentage of the position setpoint.

Figure 2.6 shows the response of servo position.
2.5 Standard PID Control

The typical PID control law in its standard form is

\[
u(t) = K_p [e(t) + T_d \frac{de(t)}{dt} + \frac{1}{T_i} \int_0^t e(\tau)d\tau]\]

(2.1)

where \(e(t) = Y_{sp}(t) - y(t)\) is the system error (difference between the reference input and system output), \(u(t)\) the control variable, \(K_p\) the proportional gain, \(T_d\) the derivative time constant and \(T_i\) the integral time constant.

Equation (2.1) can also be written as

\[
u(t) = K_pe(t) + K_d \frac{de(t)}{dt} + K_i \int_0^t e(\tau)d\tau\]

(2.2)

where \(K_d = K_pT_d\) and \(K_i = K_p/T_i\). The tuning problem consists of determining the values of these three parameters with the aim of satisfying different control specifications such as set-point following, load disturbance attenuation, robustness to model uncertainties and rejection of measurement of noise.

A proportional controller \((K_p)\) has the ability to reduce the rise time but cannot eliminate the steady-state error. An integral control \((K_i)\) has the capability of
removing the steady-state error, but may worsen the transient response. A derivative control \( K_d \) has the power to increase the stability of the system, reduce the overshoots and improve the transient responses.

### 2.6 ZN Based PID Terminology

In order to manage the proportional, integral and derivative terms, the Digital Servo system uses a PID controller. A PID controller is a control loop feedback controller that attempts to correct the error between a measured process variable and a desired set point by calculating and then instigating a corrective action that can adjust the process to keep the error minimal. In the case of the Digital Servo system, the measured variable is the position loop. The Digital Servo controller thus measures the error between the position reference and the actual platform position and attempts to correct it.

The PID controller used in the Digital Servo system is represented in Figure 2.7 (Lab-Volt Ltd, 2003). As illustrated, derivative action can be performed either on the error value or on the process negative. Performing derivative on the process negative eliminates the impulse associated with derivative of error during position reference step transitions.

![Figure 2.7: Expended PID controller block diagram.](image-url)
The PID controller calculates three separate parameters in order to correct the error: proportional, integral, and derivative action. Proportional action determines the controller reaction to the current error. Integral action determines the controller reaction based on the sum of the last measured error value.

Derivative action determines the controller reaction based on the rate at which the error changes. These actions are then added together to correct the system output (platform position) in order to reduce the difference between the reference value and the actual value, i.e., the system error.

When using a PID controller, it is necessary for the proportional, integral and derivative terms to be set properly, otherwise, the controlled process can become unstable. There are usually four main points to an optimal response:

(i) A minimal overshoot or no overshoot at all,
(ii) A quick rise time,
(iii) A quick settling time,
(iv) A low steady state error.

The most basic tuning method is the manual tuning. This type of tuning can only be performed by people who are experienced with the process type of the application. There are three main tuning parameters: the proportional gain $K_p$, the integral time $t_i$, and the derivative time $t_d$. Each tuning parameters has different effects on the response characteristics and so it is important to know when to use one instead of the other. Table 2.1 summarizes the effect of $K_p$, $t_i$, or $t_d$ variations on rise time, overshoot, settling time and steady-state error.

Table 2.1: Effects of PID parameters variations on a step response.

<table>
<thead>
<tr>
<th>PID parameters</th>
<th>Rise time</th>
<th>Overshoot</th>
<th>Settling time</th>
<th>Steady state error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increasing $K_p$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Negligible change</td>
<td>Decrease</td>
</tr>
<tr>
<td>Decreasing $t_i$</td>
<td>Decrease</td>
<td>Increase</td>
<td>Increase</td>
<td>Eliminated</td>
</tr>
<tr>
<td>Increasing $t_d$</td>
<td>Minor decrease</td>
<td>Minor decrease</td>
<td>Minor decrease</td>
<td>No change</td>
</tr>
</tbody>
</table>
A quick and easy alternative to the manual tuning method is the ZN method. It requires first to deactivate the integral and derivative terms. The proportional gain $K_p$ is then increased (starting from 0) until the system reaches a constant oscillation with a constant amplitude and period. This gain value is referenced as the ultimate gain $K_U$. The oscillation period is also measured and is called the oscillation period $t_U$. Using these two measured values, the three tuning parameters are then calculated using the equations, depending on the controller type: (P, PI, PID). This type of tuning produces a quarter amplitude decay response which is acceptable, but not optimal.

To summarize the difference effects of proportional, integral and derivative action on a servo system in angular position control, we can say that:

(i) Increasing the proportional gain $K_p$ causes the position step response of an overdamped system to become critically damped. If the proportional gain is increased further, the critically damped system will become underdamped. As the gain increases, an oscillatory component that increases in frequency and amplitude will appear and eventually cause the system to become unstable.

(ii) Adding integral action eliminates static friction error, but also increases the tendency of the step response to oscillate and can cause the system to become unstable.

(iii) Adding derivative action can dampen or suppress an oscillating step response. Derivative action, however, is very sensitive to noise and can cause erratic behaviours when set at high values.

2.7 GA Terminology

Generally, GA consist of three fundamental operators: reproduction, crossover and mutation. Given an optimization problem, simple GA encode the parameters concerned into finite bit strings, and then run iteratively using the three operators in a random way but based on the fitness function evolution to perform the basic tasks of copying strings, exchanging portions of strings as well as changing some bits of strings, and finally find and decode the solutions to the problem from the last pool of mature strings (Jamshidi, Leandro, Peter, 2003).
GA terminology basically similar to that use in Biological principle:

(i) Gene – the basic unit of the string (chromosome)
(ii) Chromosome – consisting of genes, its the searched optimal solution (string)
(iii) Generation – it is a population in a certain phase of GA
(iv) Population – consists of chromosomes, changes from generation to generation, its size can be changed during GA but is usually constant
(v) Fitness - quantifies the optimality of a chromosome, usually it is a positive scalar, it is a result of an appropriate transform of the objective function

The flow chart of GA works is shown by Figure 2.8.

Figure 2.8: Flow chart of GA functions and process
The GA has the following steps:

(i) The first population $P_0$ is generated. Strings are randomly generated from the defined space or initialized by some appropriate rule.

(ii) The objective function is computed and the fitness of each string in the $k$-th population $P_k$ is evaluated.

(iii) Ending conditions are tested, if they are satisfied, the best string is selected.

(iv) The first population $P_0$ is generated. Strings are randomly generated from the defined space or initialized by some appropriate rule.

(v) The objective function is computed and the fitness of each string in the $k$-th population $P_k$ is evaluated.

(vi) Ending conditions are tested, if they are satisfied, the best string is selected.

(vii) If the ending conditions are not satisfied, the best strings are selected (group A) and are transferred to the next population. The other group C includes strings on which the crossover and mutation operations will be applied. The last group B continues directly to the new population.

(viii) By crossing, some parts of parent strings are randomly combined and changed during mutation and thus the new combined strings - genes are created. The result of these operations is a new group $C'$.

(ix) The new population $P_{k+1}$ consists of groups A, B, $C'$.

(x) Go to the step (ii).

The ending conditions are met when the above defined conditions for the string are satisfied and the gradient of the purpose function is very low or unchanged. The commonly used GA stop condition is achieving of a pre-specified number of generations.

A GA differs from other common optimizing methods in some aspects:

(i) it can escape from the local extremes and bring closer to the global extreme.

(ii) it can search in parallel, simultaneously in several directions.

(iii) it does not need partial information about the solution like gradient of objective function.

(iv) it can solve optimization problems with a many variables.

(v) it is easy to apply it in the large space of optimized parameters.

(vi) it is very time-consuming.
2.8 GA for Optimization

Normally, GA begin the process of optimization with a randomly selected population of individuals. Then, the fitness for each individual is calculated. Next comes the application of the genetic operators: selection, crossover, and mutation. Thus, new individuals are produced from this process, which then form the next population. The transition of a population $P_g$ to the next population $P_{g+1}$ is called generation, where $g$ designates the generation number (Ruano, 2005). In Figure 2.9, the operations executed during a generation are schematically represented. The evolution of the population continues through several generations, until the problem solved, which in most cases, ends in a maximum number of generations $g_{\text{max}}$.

![Figure 2.9: Representation of the executed operations during a generation](image)

Figure 2.9: Representation of the executed operations during a generation

Figure 2.10 shows a GA to solve an unconstrained optimization problem. Initially, the fitness function $F(x)$ is defined, based on the problem to be solved. The probabilities $p_c$, $p_m$, and the population size $\mu$ are chosen. The individuals of the initial population $P_0$ are randomly initialized. So begins the first generation through the fitness calculation $F(c_i)$ with $i = 1, \ldots, \mu$ for each individual of the population. By applying selection to the individuals of the population $P_g$, a transition population $P$ would result. From the application of crossover with the probability $p_c$, a further
transition from P’ to population P” results. From the application of mutation operator, with the probability \( p_m \), to the individuals of the population P” a new population results, which is designated \( P_{g+1} \). If the maximum generation number \( g_{\text{max}} \) is not achieved, then the fitness is calculated, and the genetic operators are applied. If \( g = g_{\text{max}} \), then the optimization is terminated, and the fittest individual represents the solution of the optimization problem. By repeated application of the genetic operators, its possible that the fittest individual of a generation was not selected or destroyed by crossover or mutation. Thus, the best individual would be no more contained in the next population. This problem can be avoided by ensuring that the best individual of the previous population goes into the next generation, if the best individual of the current population has a lower fitness. The best individual is replaced only by a still better individual.

\[
\begin{align*}
\text{Input:} & \quad F(x), p_c, p_m, \text{ and } \mu \\
\text{Output:} & \quad c_i \\
\text{Auxiliary variable:} & \quad g \text{ and } g_{\text{max}} \\
\text{Begin} & \\
& g = 0 \\
& \text{initialize: } P_g = \{c_1, \ldots, c_i, \ldots, c_\mu\} \\
& \text{while } (g \leq g_{\text{max}}) \text{ do} \\
& \quad \text{fitness calculate: } F(c_i) \\
& \quad \text{selection: } P_g \rightarrow P' \\
& \quad \text{crossover: } P' \rightarrow P'' \\
& \quad \text{mutation: } P'' \rightarrow P_{g+1} \\
& \text{end while} \\
& \text{return } c_i \\
\text{end}
\end{align*}
\]

Figure 2.10 : GA for optimization
2.9 GA Software

Since GA has attracted vast number of considerable research, there are several established module available in market ready for user such as:

(i) GENOCOP III – Genetic Algorithm for Constrained Problems in C (by Zbigniew Michalewicz)

(ii) DE – Differential Evolution Genetic Algorithm in C and Matlab (by Rainer Storn).

(iii) PGAPack – Parallel Genetic Algorithm in Fortran and C (from Argonne National Laboratory)

(iv) PIKAIA – Genetic algorithm in Fortran 77/90 (by Charbonneau, Knapp and Miller)

(v) GAGA – Genetic Algorithm for General Application in C (by Ian Poole)

(vi) GAS – Genetic Algorithm in C++ (by Jelasity and Dombi)

(vii) GALib – C++ Genetic Algorithm Library (by Matthew Wall)

(viii) Genetic Algorithm in Matlab (by Michael B. Gordy)

(ix) GADS – Genetic Algorithm and Direct Search Toolbox in Matlab (from MathWorks)

(x) GEATbx – Genetic and Evolutionary Algorithm Toolbox for Matlab (by Hartmut Pohlheim)

(xi) GAOT – Genetic Algorithms Optimization Toolbox in Matlab (by Jeffrey Joines)

The algorithm that will be implemented in this project is GAOT, GA Optimization Toolbox. Each module of the algorithm is implemented using a MATLAB function. This provides for easy extensibility, as well as modularity. The GAOT is suit for real, binary, and order-based presentations. The basic function is ‘ga’ function, which runs the simulated evolution. The ‘ga’ function perform the simulated evolution using ‘evalFN’ to determine the fitness of the solution strings. The ‘ga’ function uses operators ‘zOverFN’ and ‘mutFN’ to alter the solution strings during the search. The system maintains a high degree of modularity and flexibility as a result of the decision to pass the selection, evaluation, termination functions to the ‘ga’ as well as a list of genetic operators. Thus, the base GA is able to perform evolution using any combination of selection, crossover, mutation, evaluation and
termination functions that conform to the functional specifications. The GAOT module can be downloaded for free at: http://www.isc.ncsu.edu/mirage/GAToolBox/gaot/.

2.10 Experimental Model: Lab-Volt Digital Servo Model 8063

The Lab-Volt Digital Servo training system as shown in Figure 2.11 is a compact trainer designed to familiarize students with the fundamentals of digital servo control. The system features a single-axis belt-driven positioning system, a digital servo controller, and powerful software tools. The motor control can be achieved in several ways: by using the included hardware controller, LABVIEW or MATLAB/SIMULINK, or an optional analog controller.

![Figure 2.11: Lab-Volt Digital Servo Model 8063](image)

The features & benefits of the Lab-Volt Digital Servo Model 8063 as below:

(i) Compact system that can be used on a table or bench
(ii) Servo controller and linear axis
(iii) Position and speed control, friction break, belt tensioning and backsplash, dual encoders, transferable inertia load
(iv) Safe and robust
(v) High-speed communication through a USB connection
(vi) Easy connection to mechanical devices
(vii) Observation and control can be performed simultaneously
REFERENCES


Hong Zhang, Yuanhu Cai, Yuchun Chen (2010). Parameter Optimization of PID Controllers Based on Genetic Algorithm. School of Power and Energy Northwest Polytechnical University Xi’an, China, pp. 47-49.


Huang Yourui, Chen Xiuqiao and Hu Yihua. Optimization for Parameter of PID Based on DNA Genetic Algorithm. PLA Electronic Engineering, Institute, pp.859-860.