

Mechatronic Design for Component-Swapping Modularity using Bi-Directional Communications in Networked Control Systems

by
Melih Cakmakci

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Mechanical Engineering)
in The University of Michigan
2009

Doctoral Committee:

Professor A. Galip Ulsoy, Chair
Professor Dawn M. Tilbury
Professor Kang G. Shin
Ilya V. Kolmanovsky, Ford Motor Company

© Melih Çakmakcı 2009
All Rights Reserved

To my loving Family

ACKNOWLEDGEMENTS

I would like to thank first to my advisor Professor A. Galip Ulsoy. His guidance and constant encouragement, limitless patience and vision taught me how to be a better engineer and researcher. I also would like to thank him for allowing me to join his distinguished research group and giving me the opportunity to interact with many brilliant graduate students and researchers over the past several years.

I would like to thank my doctoral committee members Dr. Kolmanovsky, Professors Tilbury and Shin for their time and insightful comments during our meetings and to this dissertation.

I am grateful to my past and current colleagues and managers at Ford Research Laboratories for their constant support and encouragement which made my studies at University of Michigan feasible. I also gratefully acknowledge the financial support from Ford Motor Company through its tuition assistance program.

My time in Michigan would not feel this special if it weren't for my friends. For that, I am indebted to the *Willowtree Gang* and the blissful residents of SRL2626.

This thesis is dedicated to my parents and my brother. Their constant support and love gave me the strength to complete this long otherwise wonderful journey.

TABLE OF CONTENTS

DEDICATION	ii
ACKNOWLEDGEMENTS	iii
LIST OF FIGURES	vi
LIST OF TABLES	viii
LIST OF APPENDICES	ix
ABSTRACT	x
CHAPTER	
I. Introduction	1
1.1 Motivation	1
1.2 Background	3
1.2.1 Smart Control System Components	3
1.2.2 Networked Control Systems	4
1.2.3 Modularity and Modularity in Control Systems	5
1.3 Scope of this Dissertation	7
1.4 Original Contributions	11
II. Improving Component Swapping Modularity using Bi-directional Communication in Networked Control Systems	17
2.1 Introduction	18
2.1.1 Smart Control System Components	18
2.1.2 Networked Control Systems	19
2.1.3 State-of-the art in NCS and Smart Component Research	20
2.1.4 Bi-directional Communications among Smart Components in Networked Control Systems	21
2.1.5 Utilizing New Degree of Freedom to Improve Modularity of Control Systems	25
2.2 Problem Formulation to Maximize Component Swapping Modularity	28
2.2.1 Distribution Problem to Maximize Actuator Swapping Modularity	28
2.2.2 Components of the Distribution Problem	29
2.2.3 Extensions of the Maximizing Component Swapping Modularity Problem	31
2.3 Example	31
2.3.1 System Overview	32
2.3.2 Optimization Problem Components	33
2.3.3 Solution to the Optimization Problem	38

2.4	Summary and Conclusions	43
III.	Modular Discrete Optimal MIMO Controller for a VCT Engine	49
3.1	Introduction	49
3.2	Engine Modeling and MIMO Control Design	52
3.2.1	VCT Engine Model	52
3.2.2	Dynamics of the VCT Unit and EGO Sensor	53
3.2.3	Discrete-time MIMO Controller Design	54
3.3	Distribution Problem	56
3.3.1	Formulation of $\mathbf{C}_{des}$	59
3.3.2	Formulation of $\mathbf{C}_{dist}$	60
3.4	Conclusion	66
IV.	Combined Component Swapping Modularity for a VCT Engine Controller	71
4.1	Introduction	71
4.2	Combined Component-Swapping Modularity of a System	76
4.2.1	Simultaneous Approach	76
4.2.2	Sequential Approach	78
4.2.3	Optimization Problem to Maximize Combined Component Swapping Modularity	78
4.3	Example: Combined Component Swapping Modularity of the VCT Engine	79
4.3.1	Simultaneous Approach	81
4.3.2	Sequential Approach	84
4.4	Conclusion	88
V.	Summary and Conclusions	92
5.1	Summary	92
5.2	Engineering Relevance	94
5.3	Future Work	95
APPENDICES		97

LIST OF FIGURES

Figure

1.1	A SISO feedback control system.	1
1.2	A single loop traditional feedback control system with networked smart components	3
2.1	Evolution of Control System Components:(a) Traditional Feedback Control Systems.(b) Feedback Control Systems with Smart Components.	19
2.2	Networked Control System with Bi-Directional Communications among Smart Components.	22
2.3	Representation of the Closed Loop System with Centralized controller, C	23
2.4	Control System Component Physical and Functional Boundaries.	25
2.5	Physical Boundaries for Model Parameters	34
2.6	Closed loop system block diagram with controller.	34
2.7	Example Configuration: Base Controller, Sensor multicast	37
2.8	Summary of Results.	39
2.9	Effect of Changing Communication and Computation on Actuator Modularity . . .	42
3.1	Control System Component Physical and Functional Boundaries.	50
3.2	A Continuous VCT System.	52
3.3	Input Output relationship of the dynamic plant model for control development. . .	53
3.4	VCT Engine with Discrete MIMO Controller closed loop response	57
3.5	VCT Engine with Discrete MIMO Controller	58
3.6	VCT Engine with Distributed Discrete MIMO Controller	58
3.7	Polynomial Coefficients for $N_{11}(z)$ as τ_{vct} and τ_{ego} are varied.	61
3.8	Real-time schedule for distributed discrete MIMO controller.	61

3.9	Pole Zero Map of the $C_{des}(z)$ as VCT Component (i.e. $\tau_{vct,act}$ and $\tau_{vct,sen}$) changes. Pole or zero plots with the dashed boxes are assumed to vary as VCT component changes.	63
3.10	Candidate Distribution to Maximize VCT Modularity	65
3.11	Candidate Distribution to Maximize EGO Modularity	67
4.1	Bi-directional Communications and Overall Control and Plant Composition.	72
4.2	Illustration of Set Φ_X for a Two Parameter Component X , $\mathbf{p}_X = \{p_{X,1}, p_{X,2}\}$	77
4.3	VCT Engine with Discrete MIMO Controller	80
4.4	VCT Engine with Distributed Discrete MIMO Controller	81
4.5	Simultaneous and VCT Component first, EGO Sensor second Sequential Solution.	83
4.6	Distributed Controller in the Closed Loop	84
4.7	Comparison of Centralized vs. Distributed Controllers	85
4.8	EGO Sensor first, VCT Component second Sequential Solution.	87
B.1	Closed Loop System with MIMO Controller	111
B.2	Engine Plant and Feedforward Fuel Controller	111
B.3	Engine Plant	112
B.4	Regression Equations for Engine Performance Variables	113
B.5	Closed Loop System with Distributed Controllers	113
B.6	ECU Subsystem	114
B.7	VCT Subsystem	115
B.8	EGO Subsystem	115

LIST OF TABLES

Table

2.1	Nominal Plant Model Parameter Numerical Values	33
2.2	List of Parameters for Example Distribution Solution	38
2.3	List of Communication Configurations	41
2.4	Computation per components	42
3.1	Transfer Functions for the Modular VCT Controller Solution	66
3.2	Transfer Functions for the Modular EGO Controller Solution	66
4.1	Distribution Solution with Simultaneous Approach	82
4.2	Distribution Solution with Sequential Approach (EGO first, VCT second)	86

LIST OF APPENDICES

Appendix

A.	Matlab Scripts	98
B.	Simulink Models	110

ABSTRACT

Availability of low cost electronics has led to a new type of control system components, so called “smart” components, which can perform control actions in the actuator and sensor components as well as in the controller. The objective of our research is to improve component swapping modularity of feedback control system components by utilizing smart such components, and the bi-directional network communications among these components. Control systems that are designed to have modularly swappable components can be re-configured, by changing their components, and still can operate at their optimal closed loop performance by only re-designing the control algorithm that resides in the component that is being changed.

As a first step, a quantification measure for swapping-modularity of single component and an optimal design formulation that maximizes component swapping modularity are presented. The developed method is then demonstrated with a driveshaft speed control example where the component swapping modularity of the actuator (i.e. dc-motor and gearbox smart component) is maximized. The effects of communication and computation constraints on the resulting component-swapping modularity are also investigated.

The problem formulation originally developed for single input single output (SISO) controllers and continuous time systems is then extended to apply to multiple input multiple output (MIMO) controllers with discrete-time formulations. A control oriented *pre-optimization* technique, based on pole zero locations which simplifies

numerical solution of the optimization problem is developed. The method is then applied to the distribution of a discrete MIMO controller for a Variable Camshaft Timing (VCT) Engine to improve the component swapping modularity of the VCT and the Exhaust Gas Oxygen (EGO) Sensor components using network communications.

Development of measures for combined swapping modularity is important to be able to analyze more complex engineering cases. As the final research topic, two approaches (simultaneous and sequential) for combining component swapping modularity of two or more components are discussed. These combined modularity approaches are used to design controllers which maximize the component-swapping modularity of the VCT component and the EGO sensor for an internal combustion engine as an example.

CHAPTER I

Introduction

1.1 Motivation

Feedback control systems have four basic components: Controllers, actuators, sensors and the controlled system or plant. Traditionally, actuators and sensors are considered “brainless” devices which perform only actuation and sensing tasks respectively and these three control system components (i.e. controllers, actuators and sensors) sequentially communicate via a dedicated analog/digital connection. Figure 1.1 illustrates the feedback control system components with traditional roles and communication paths for a single input single output (SISO) system.

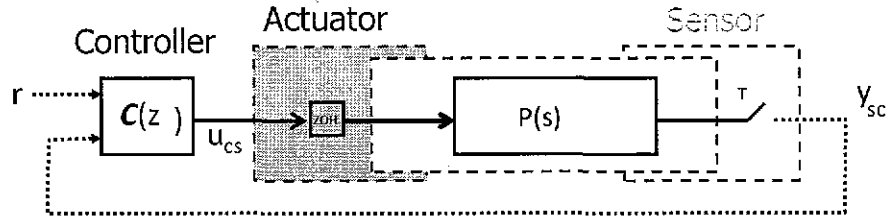


Figure 1.1: A SISO feedback control system.

With the introduction of “smart” components (i.e., components which have actuator and/or sensor responsibilities and on-board computing capabilities), increasingly more computer control algorithms are distributed into individual system components. The majority of these new smart components are also equipped with networking ca-

pabilities where information exchange among components is facilitated via a common communication bus.

In Figure 1.2, two smart components, component A and component S, performing actuation and sensing tasks respectively, are shown for a single loop feedback control system with base controller (where its transfer function is denoted with $C_{BC}(z)$) and plant (where its dynamics are represented with $P(s)$). The plant dynamics includes the controlled system dynamics as well as the dynamics of the components that physically interact with it to form the feedback control system. With the introduction of the smart components to the feedback control system, it is now possible to introduce four additional communication paths among the controller, the smart actuator and the smart sensor due to bi-directional (two-way) capabilities of components communicating on a network: (1) communication from the base controller to component S, (u_{cs}), (2) communication from component A to component S, (y_{as}), (3) communication from Component A to the base controller, (y_{ac}), and finally (4) communication from component S to component A, (y_{sa}). The use of smart components communicating via a network provides additional degree of freedom for designing distributed control systems which exhibit desirable engineering properties while maintaining the same closed loop performance as their centralized (single controller) equivalents.

The objective of the research presented in this dissertation is to improve component swapping modularity of feedback control system components by utilizing smart components and bi-directional network communications among these components. Control systems that are designed to have modularly swappable components can be re-configured, by way of changing their components, and still can operate at their optimal closed loop performance by only re-designing the control algorithm that resides in the component that is being changed.

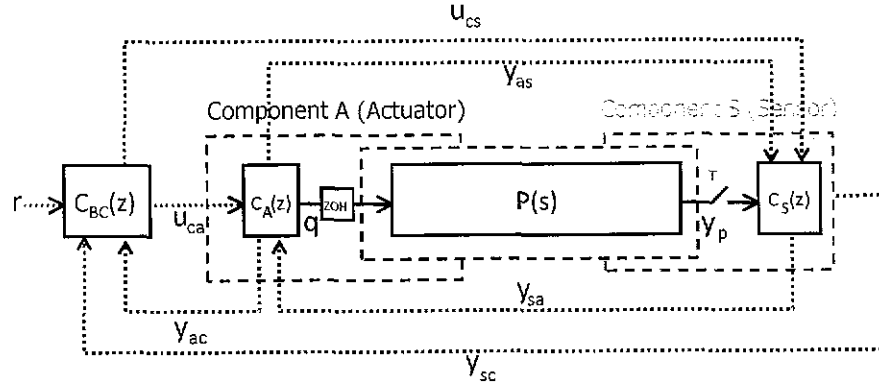


Figure 1.2: A single loop traditional feedback control system with networked smart components

1.2 Background

1.2.1 Smart Control System Components

Although in the early days of computer control, only one centralized computer was responsible for executing the algorithms for a particular system, increasingly, computer control algorithms reside inside individual system components in a distributed fashion. For example, there are up-to 80 micro-controllers in today's high-end vehicles and it is expected that by 2010, 90% of all computer code will reside in such embedded systems [2]. With the increased availability of electronics at much lower costs many commercially available sensors and actuators currently have on-board computers (i.e., CPU, memory, I/O interface) which enable them to perform diagnostics and component specific control functions such as mapping, filtering noise and saturation [18, 17, 29, 22].

Commercially, networked smart components have already been introduced by various companies [15],[12]. In [23], a network-based coordinated motion control scheme is successfully implemented for a complex multitasking mechatronic system by use of networked sensors, actuators and controllers. An overview of design considerations to integrate sensing, computing and communication with extremely limited packaging

constraints is given in [38].

1.2.2 Networked Control Systems

Introduction of networks for practical realtime control systems goes back as early as the 1980s when Bosch GmbH began “a feasibility of study of using networked devices to control different functions in passenger cars” [3]. In [19], controller design analysis for systems with random attributes was discussed. In [20], the authors investigate the stability of linear systems under random communication delays. Although using networks provides many benefits such as less wiring, better interfacing, lower costs [41, 37] and an open architecture, there are some disadvantages such as communication delays, bandwidth limitations and non-delivery of the message carrying the loop information from one component to the other [41, 37, 36, 40].

Control systems, whose feedback loops are closed via networks, are commonly called Networked Control Systems (NCSs) [41]. Current research on NCS has primarily focused on understanding the effects of network delays and bandwidth limitations [14]. Yook and coworkers present a mathematical framework to analyze the effect of time delays on the mechanical performance of distributed control systems [39]. This framework is then used to evaluate different control schemes and choose the best one. In [33], a gain scheduling approach, based on constant network delays, is used to improve overall NCS performance. Zhang and co-workers state that conventional control theories such as synchronized control, and non-delayed sensing and actuation must be reevaluated before they can be applied to NCSs [41]. A NCS model, with packet drop out and multiple packet transmission as an asynchronous system, is given and the system stability at different data loss rates is investigated.

Walsh and Ye demonstrate that improved performance is possible by dispensing with queues and dynamically scheduling the network traffic in networked control

systems [37]. Lian and co-workers provide design considerations related to control quality of performance as well as network quality of service [25]. Performance comparison of continuous control, digital control and networked control cases are given. It is also stated that although networked control systems are a class of digital control systems, by definition their performance degrades when the sampling rate is increased due to increased message traffic among components. Improvement opportunities for NCS performance are stated in two areas: (1) decrease delay due to hardware components and (2) design of the controller so that performance loss due to uncertainty is minimized. Walsh, Ye and Bushnell propose a new protocol, Try-Once-Discard, to improve NCS performance and provide a proof of stability for this protocol and the more common statically scheduled protocols [35].

Many industry oriented network protocols and solutions have emerged in the recent years such as CANbus [26], DeviceNet [1], ControlNet [11], ProfiNet [16], Fieldbus [13], besides analog signaling standards such as 4-20 mA. Lian, and co-workers present performance evaluation of three control networks: Ethernet, ControlNet and DeviceNet based on characteristics and requirements of control systems [24]. They conclude that different types of network protocols are suitable for different control applications. Understanding robustness of network hardware under specific environmental conditions is also important [26, 32].

1.2.3 Modularity and Modularity in Control Systems

A frequently cited, and widely accepted, definition of modularity is given by Ulrich and Tung in [34] and it depends on two characteristics of a design: (1) similarity between the physical and functional architecture of the design, and (2) minimization of incidental interactions between physical components. Three categories of modularity have been defined: *Component-swapping modularity* occurs when two or more

different basic components are paired with a module, thus creating different product variants belonging to the same product family. *Component-sharing modularity* is complementary to component-swapping modularity. Number of modules sharing the same basic component create different product variants belonging to different product families. *Bus modularity* occurs when a module can be matched with any number of basic components.

Based on the modularity definition given in [34], modularity can be improved by defining the component boundaries in the design stage such that the similarity between physical and functional architecture is increased as incidental interactions among components are minimized. In our research, we consider systems in which component boundaries are already defined (such as systems built from off the shelf components).

Increasing component-swapping modularity of a control systems component may actually enhance the performance of the system. By using a better performing component (such as faster responding actuator), we can obtain better a performing system. It is also conceivable that component-swapping modularity can be utilized to get the best performance out of a system. For example, an inferior alternative to the original component can be used due to cost limitations while the best possible closed loop performance from the system can still be achieved.

Usually modularity of hardware [21] and modularity of software [27] are considered separately. Considering both the hardware and software aspects of the system for modularity brings up the *co-design* concept which usually results in better performance and cost structure [10]. Our work aims to improve modularity by distributing the overall control activity of a system into smart component controllers such that only the relevant control module has to be designed in the event of a component

change; the rest of the system remains unchanged. We see our work as complementary to the notion of co-design mentioned in [28], where suppliers work together to deliver a larger product.

In [31], consequences of developing modular controllers for automotive powertrain control systems are discussed. Instead of tackling air fuel ratio and cam position problems separately as single loop (therefore modular) problems, a MIMO controller which compensate for the cross coupling plant dynamic effects is designed. Although the plant model used in this work and the work presented in Chapters 3-4 of this dissertation are originated from the same work (i.e. [30]), the MIMO controller used in this dissertation is based on the discrete LQG design approach and has a more general controller formulation (i.e. a transfer function matrix).

1.3 Scope of this Dissertation

This dissertation presents a method to improve component-swapping modularity of feedback control systems using smart components and bi-directional network communications. The aim of this presentation is to help build control systems that are designed to have modularly swappable components that can be re-configured, by way of changing their components, and still can operate at their optimal closed loop performance by only re-designing the control algorithm that resides in the component that is being changed. The dissertation is composed of three manuscripts [9, 8, 7] which are:

1. Chapter 2: “Improving Component Swapping Modularity using Bi-directional Communication in Networked Control Systems”, which is accepted for publication in the IEEE Transactions on Mechatronics [9] and earlier versions of this work [4, 5] were presented at the 2005 American Control Conference and the

2006 International Symposium on Flexible Automation, respectively. In this paper, the bi-directional communications in networked control systems with smart components are considered and the potential benefits of having such systems are discussed. A problem formulation and solution methodology to maximize component swapping modularity in control systems is presented. To illustrate this method, the feedback control driveshaft of speed with a tachometer example (where a base controller, a “smart” dc-motor with gearbox component and a “smart” tachometer component communicate over the network) is used. Networked controller system design utilizing bi-directional communication showed significant improvements in component swapping modularity compared to the traditional uni-directional feedback loop design. The effect of relaxing communication and computation constraints on the component-swapping modularity is investigated and it is concluded that although relaxing these constraints improves the swapping modularity of a component, there exists a saturation point where no further improvement is observed.

2. Chapter 3: “Modular Discrete Optimal MIMO Controller for a VCT Engine” [8], which has been accepted to the 2009 American Control Conference. In this paper, a method to distribute the discrete-time MIMO controller for the VCT engine, to improve the component swapping modularity of the VCT actuator and the EGO sensor components, is presented. This work is an extension of the method described in [9, 4, 5] to a discrete-time MIMO controller applied to a more complex system. A discrete-time LQG controller was designed based on work from [30]. Then, this controller was distributed to the ECU, a VCT controller and an EGO sensor controller in order to improve the component swapping modularity of the system. A control oriented *pre-optimization* technique

which simplifies the optimization problem by identifying and grouping together the controller poles/zeros that change, or remain unchanged as the VCT or EGO parameters vary is presented. Application of the pre-optimization technique results in a reduction of design variables and shorter computation times to obtain a solution. The performances of the resulting distributed controllers after the optimization problem is solved are identical to the performance of the original controller. These distributed controllers provide component-swapping modularity which does not exist if the entire algorithm was implemented using single centralized controller (i.e. using the ECU only).

3. Chapter 4: “Combined Component Swapping Modularity for a VCT Engine Controller” which will be submitted to the 2009 Dynamic Systems and Control Conference. In this paper, an application of the design method presented in [9] for combined swapping modularity of system component is discussed. Development of measures for combined swapping modularity is important to be able to analyze more realistic engineering cases. First use cases for applications for control systems modularly swappable components are reviewed. Two approaches (simultaneous and sequential) for combining component swapping modularity of two or more components are presented. A formulation to numerically calculate component swapping modularity in each approach, and the optimization problem to maximize the combined component swapping modularity of a system, are presented. Then these combined modularity approaches are used to design controllers which maximize the component-swapping modularity of the VCT Component (i.e. actuator and sensor) and EGO Sensor for the example given in [8].

Results of both approaches, simultaneous combined modularity of VCT Com-

ponent and EGO Sensor, sequential combined modularity for VCT component first, EGO component second and EGO sensor first and VCT component second, show combined swapping modularity of the system can be significantly improved by using bi-directional network communications and smart components. All the resulting systems would have the desired closed loop characteristics with optimal performance. Our results also show that the combined modularity problem is a more difficult problem than the individual component swapping modularity problem in Chapter 3 [8]. It is observed that the interval of solutions where optimal controllers can be designed is reduced from the results given in [8] with the increased use of communication paths and higher order transfer function solutions. We also observe that the solutions observed with the sequential approach depend on the sequence of configuration and are not necessarily the same as the simultaneous solution.

The optimal distributed controller solutions obtained utilize the bi-directional communications introduced earlier and have improved combined swapping modularity properties compared to the traditional centralized version of the controller.

To fully integrate this research into practical control system applications further research is planned. The possible application of the method presented in this dissertation to more general systems with networked smart components (with a simple sensor network example) was discussed in [6]. As the number of components or the order of the desired controller increase the numerical burden to solve the distribution problem with the method presented here also increases. In order to improve the research presented in this dissertation, improvements on the formulation of the problem, a combined method which enables the concurrent solution of the control

design and distribution problem application to more realistic examples are planned as discussed in Chapter 5.

1.4 Original Contributions

1. It is shown that the bi-directional communications among networked smart components provide an additional design degree of freedom in the distribution problem which can be exploited to improve modularity. (Chapter 2)
2. A design method for control systems utilizing bi-directional communications is developed for
 - (a) a simple low order SISO example. (Chapter 2)
 - (b) a more complex discrete-time MIMO example. (Chapter 3)
3. Pre-optimization analysis of controller pole-zeros based on identifying and grouping together the controller poles/zeros which change, or remain unchanged, as the component plant parameters vary, is given for improving efficiency of solution (Chapter 3)
4. Introduced Component Swapping Modularity for Control System Components
 - (a) Developed a quantification measure of component swapping modularity for a single component, (Chapter 2)
 - (b) Studied the effects of communication and computation, (Chapter 2)
 - (c) Defined two ways to measure combined component swapping modularity of components that are part of a control system. This enables the solution of the distributed control problem at the system level. (Chapter 4)

Bibliography

- [1] Open DeviceNet Vendors Association. Odva website. www.odva.org.
- [2] K. J. Astrom. Present developments in control applications. Keynote Paper at the IFAC 50th Anniversary Meeting, Heidelberg, 2006.
- [3] J. Baillieul and P.J. Antsaklis. Control and communication challenges in networked real-time systems. *Proceedings of the IEEE*, 95(1):9–28, Jan. 2007.
- [4] M. Cakmakci and A. G. Ulsoy. Networked control systems with bi-directional communication among "smart" components. *Proceedings of the American Control Conference*, pages 627–632, June 2005.
- [5] M. Cakmakci and A. G. Ulsoy. Improving component swapping modularity using bi-directional communication in networked control systems. *Proceedings of the International Symposium on Flexible Automation*, July 2006.
- [6] M. Cakmakci and A. G. Ulsoy. Design of modular controllers for systems with smart networked components. *Proceedings of the 4th Int. Conf. on Design and Production of Machines and Dies/Molds*, 2007.
- [7] M. Cakmakci and A. G. Ulsoy. Combined component swapping modularity for a vct engine controller. *Proceedings of the Dynamic Systems and Control Conference*, September 2009. (ready for submission).
- [8] M. Cakmakci and A. G. Ulsoy. Modular discrete optimal mimo controller for a vct engine. *Proceedings of the American Control Conference*, June 2009. (submitted).
- [9] M. Cakmakci and A.G. Ulsoy. Improving component swapping modularity using bi-directional communication in networked control systems. *IEEE Transactions on Mechatronics*. (in press).

- [10] M. Chiodo, P. Giusto, A. Jurecska, HC Hsieh, A. Sangiovanni-Vincentelli, L. Lavagno, and M. Marelli. Hardware-software codesign of embedded systems. *Micro, IEEE*, 14(4):26–36, 1994.
- [11] ControlNET.org. Controlnet protocol website. www.controlnet.org.
- [12] Flowserve Corporation. Flowserve website. www.flowserve.com.
- [13] Fieldbus Foundation. Fieldbus website. www.fieldbus.org.
- [14] J.P. Hespanha, P. Naghshtabrizi, and Yonggang Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162, Jan. 2007.
- [15] Dust Networks Inc. Dust networks website. www.dustnetworks.com.
- [16] Profibus International. Profinet website. www.profinet.com.
- [17] R. Isermann. On the design and control of mechatronic systems-a survey. *IEEE Transactions on Industrial Electronics*, 43(1):4–15, 1996.
- [18] R. Isermann and U. Raab. Intelligent actuators: ways to autonomous actuating systems. *Automatica (Journal of IFAC)*, 29(5):1315–1331, 1993.
- [19] N. N. Krasovskii and E. A. Lidskii. Analysis design of controller in systems with random attributes. *Automatic Remote Control*, 22:1021–1025, 1961.
- [20] R. Krtolica, U. Ozguner, H. Chan, H. Goktas, J. Winkelman, and M. Liubakka. Stability of linear feedback systems with random communication delays. *Proceedings of the American Control Conference*, 28, 1982.
- [21] A. Kusiak and C.C. Huang. Development of modular products. *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, 19(4):523–538, 1996.
- [22] N. Kyura and H. Oho. Mechatronics-an industrial perspective. *IEEE/ASME Transactions on Mechatronics*, 1(1):10–15, 1996.

- [23] Y. Li, L. Yang, and G. Yang. Network-based coordinated motion control of large-scale transportation vehicles. *IEEE/ASME Transactions on Mechatronics*, 12(2):208–215, 2007.
- [24] F. Lian, J. Moyne, and D. Tilbury. Performance evaluation of control networks: Ethernet, controlnet and devicenet. *IEEE Control Systems Magazine*, pages 66–83, February 2001.
- [25] F. Lian, J. Moyne, and D. Tilbury. Network design consideration for distributed control systems. *IEEE Transactions on Control Systems Technology*, 10(2):297–307, March 2002.
- [26] R.T. McLaughlin. The immunity to rf interference of a can system. *Proceedings of the IEE Colloquium on Integrity of Automotive Electronic Systems*, pages 4/1–4/8, 1993.
- [27] B. Meyer. *Object-oriented software construction Prentice Hall International series in computer science*. Prentice Hall, 1997.
- [28] M.S. Salerno and A.V.C. Dias. Product design modularity, modular production, modular organization: The evolution of modular concepts. *Actes du GERPISA no 33*, pages 61–73, 2002.
- [29] J.A. Sparks. Low cost technologies for aerospace applications. *Microprocessors and Microsystems*, 20(8):449–454, 1997.
- [30] A. G. Stefanopoulou. *Modeling and Control of Advanced Technology Engines*. PhD thesis, University of Michigan, 1996.
- [31] A.G. Stefanopoulou, K.R. Butts, J.A. Cook, J.S. Freudenberg, and J.W. Grizzle. Consequences of modular controller development for automotive powertrains: a case study. 1, 1995.

- [32] J.-P. Thomesse. Fieldbus technology in industrial automation. *Proceedings of the IEEE*, 93(6):1073–1101, June 2005.
- [33] Y. Tipsuwan and Mo-Yuen Chow. On the gain scheduling for networked pi controller over ip network. *IEEE/ASME Transactions on Mechatronics*, 9(3):491–498, 2004.
- [34] K. Ulrich and K. Tung. Fundamentals of product modularity. In A. Sharon, editor, *Issues in Mechanical Design International*, pages 73–79. ASME, New York, 1991.
- [35] G.C. Walsh, O. Beldiman, and L. Bushnell. Asymptotic behavior of networked control systems. *Proceedings of the IEEE International Conference on Control Applications*, (1448-1454), 1999.
- [36] G.C. Walsh, H. Ye, and L. Bushnell. Stability analysis of networked control systems. *IEEE Transactions on Control Systems Technology*, 10(3):438–446, May 2002.
- [37] G.C. Walsh and Hong Ye. Scheduling of networked control systems. *IEEE Control Systems Magazine*, pages 57–65, February 2001.
- [38] B. A. Warneke and K.S.J. Pister. Exploring the limits of system integration with smart dust. *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*, pages 17–22, November 2002.
- [39] J.K. Yook, D.M. Tilbury, and N.R. Soparkar. A design methodology for distributed control systems to optimize performance in the presence of time delays. *Proceedings of the American Control Conference*, pages 1959–1964, June 2000.
- [40] J.K. Yook, D.M. Tilbury, and N.R. Soparkar. Trading computation for bandwidth: Reducing communication in distributed control systems using state es-

timators. *IEEE Transactions on Control Systems Technology*, 10(4):503–518, 2002.

- [41] W. Zhang, M.S. Branicky, and S. M. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, pages 84–99, February 2001.

CHAPTER II

Improving Component Swapping Modularity using Bi-directional Communication in Networked Control Systems

Abstract: Availability of low cost electronics has led to a new breed of control system components, so called “smart” components, which can perform control actions in the actuator and sensor components as well as in the controller. “Smart” components can communicate bi-directionally in networked control systems (NCSs). One can improve control system performance and design due to the decentralized, yet more connected, nature of these systems. Current research on networked control systems primarily focuses on communication loss and delay. This paper investigates the potential benefits of bi-directional communication in a feedback control loop, which is at the heart of so many applications. First, a definition and quantification of component swapping modularity is presented. Next, an optimal design formulation is presented for an NCS that maximizes component swapping modularity. The approach is then demonstrated with an example of driveshaft speed control with a dc-motor.

Keywords: Networked Control Systems Design, Bi-directional Communications, Component-swapping Modularity

2.1 Introduction

In this section some background on smart control system components, networked control systems and the state-of-the-art in related research will be presented. Bi-directional communications among smart components in networked control systems will be defined. The additional design freedom due to bi-directional communication can be utilized to improve the component swapping modularity of control systems. The section will conclude with a discussion of the relevant literature, original contributions and contents of the paper.

2.1.1 Smart Control System Components

Components of a feedback control system, usually one per feedback loop, are controllers (i.e., the “brains”), actuators (i.e., the “brawn”), sensors (i.e., the “senses”) and the controlled system, or plant. Traditionally, the actuator and sensor are considered “brainless” devices which perform only actuation and sensing tasks, respectively.

Although in the early days of computer control, only one centralized computer was responsible for executing the algorithms for a particular system, increasingly, computer control algorithms reside inside individual system components in a distributed fashion. For example, there are up-to 80 micro-controllers in today’s high-end vehicles and it is expected that by 2010, 90% of all computer code will reside in such embedded systems [2]. With the increased availability of electronics at much lower costs many commercially available sensors and actuators currently have on-board computers (i.e., CPU, memory, I/O interface) which enable them to perform diagnostics and component specific control functions such as mapping, filtering noise and saturation. Fig. 2.1 compares the block diagram representations for traditional feedback control systems which have no control algorithm in the actuator and sensor

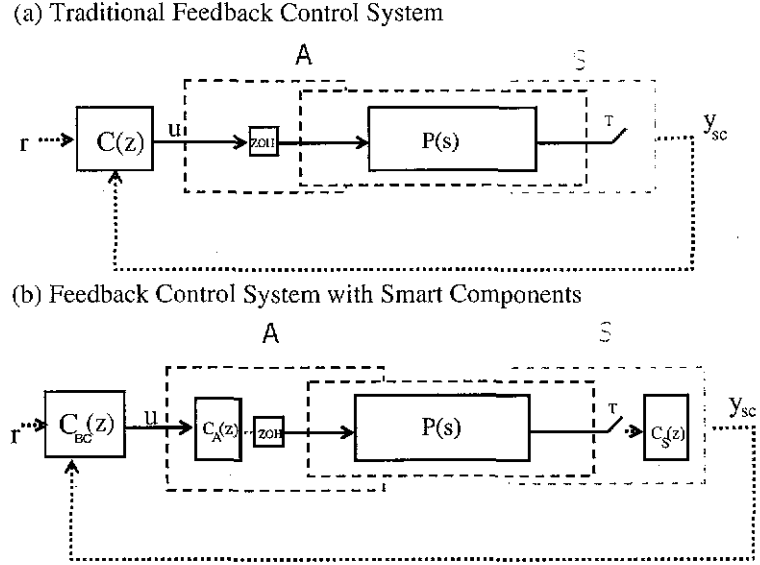


Figure 2.1: Evolution of Control System Components:(a) Traditional Feedback Control Systems.(b) Feedback Control Systems with Smart Components.

devices, to emerging systems with so called “smart” components which can perform control responsibilities in the actuator and sensor components as well as in the controller.

2.1.2 Networked Control Systems

Traditionally, the three control system components sequentially communicate via dedicated analog/digital connection. The controller sends a control signal to the actuator, the actuator acts on the “controlled system”, the sensor senses the results of the actuation and sends it to the controller for calculation of the new control signals. Both block diagrams shown in Fig. 2.1 are examples of such system communications with a single feedback loop.

Introduction of networks for real-time control systems goes back as early as 1983 when Bosch GmbH began “a feasibility of study of using networked devices to control different functions in passenger cars” [3]. Since then, application of networks to real-time control systems has become mainstream with the increasing availability of robust network hardware and software (i.e. protocols). Control systems, whose feedback loops are closed via networks, are commonly called Networked Control Systems (NCSs) [33].

Although using networks provides many benefits such as less wiring, better interfacing, lower costs [33, 28] and an open architecture, there are some disadvantages such as communication delays, bandwidth limitations and non-delivery of the message carrying the loop information from one component to the other [33, 28, 27, 32].

2.1.3 State-of-the art in NCS and Smart Component Research

The majority of the control related research in smart components are concerned with mapping of the control signals, on-board filtering of measured signals, on-board component diagnostics, and alleviating the adverse effects of network delays. An overview of design considerations to integrate sensing, computing and communication with extremely tight packaging constraints is given in [29].

Current research on NCS has primarily focused on understanding the effects of network delays [13]. For example in [25], a gain scheduling approach, based on constant network delays, is used to improve overall NCS performance. Many industry oriented network protocols and solutions have emerged in the recent years such as CANbus [19], DeviceNet [1], ControlNet [9], ProfiNet [15], Fieldbus [11], besides analog signaling standards such as 4-20 mA. Lian, and co-workers present performance evaluation of three control networks: Ethernet, ControlNet and DeviceNet based on characteristics and requirements of control systems [17]. They conclude

that different types of network protocols are suitable for different control applications. Understanding robustness of network hardware under specific environmental conditions is also important [19, 24]. Considering both the hardware and software aspects of the system for modularity brings up the *co-design* concept which usually result in a better performance and cost structure [8]. Our work aims to improve modularity by distributing the overall control activity of a system into component controllers such that only the relevant control module has to be designed in the event of a component change. We see our work as complementary to the notion of co-design mentioned in [22], where suppliers work together to deliver a larger product.

We believe that increased availability of embedded electronics and reliable high bandwidth networks will drastically increase the use of Networked Smart Components (NSCs). NSCs are components of a system that include actuation, sensing, or both actuation and sensing, responsibilities as well as on-board computing and networked capabilities. Commercially networked smart components have already been introduced by various companies [14],[10]. In [16], a network-based coordinated motion control scheme is successfully implemented for a complex multitasking mechatronic system by use of networked sensors, actuators and controllers.

2.1.4 Bi-directional Communications among Smart Components in Networked Control Systems

Fig. 2.1b includes a block diagram representation of systems which have control algorithm execution capability in actuators and sensors as well as the controller. Traditionally, a feedback control system has two types of communication: measured plant output from sensor(s) to the controller (y_{sc}) and a command signal issued by the controller to the actuator(s) (u_{ca}). With the bi-directional communication capability of networks, it is now possible to define four additional communication paths among

the controller, the "smart" actuator and the "smart" sensor: (1) communication from controller to sensor(s) (u_{cs}), (2) communication from actuator(s) to sensor(s) (y_{as}), (3) communication from actuator(s) to controller (y_{ac}), and finally (4) communication from sensor(s) to actuator(s) (y_{sa}).

In Fig. 2.2, we incorporate these new communication paths, arising from bi-directional communications in an NCS, into the original feedback controller block diagram with smart components. In Fig. 2.2, C_{BC} , C_A and C_S are transfer function

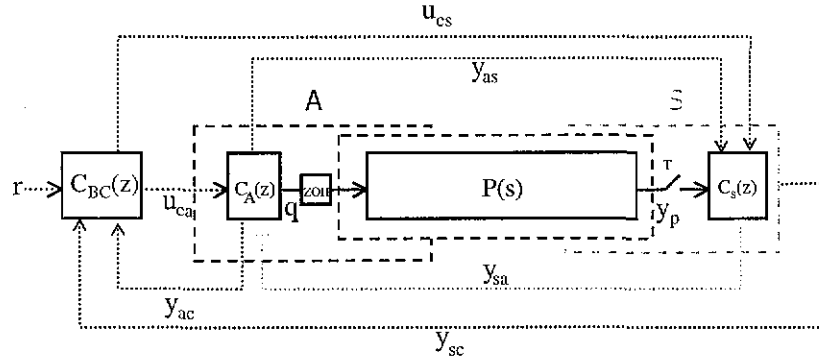


Figure 2.2: Networked Control System with Bi-Directional Communications among Smart Components.

matrices representing control algorithms in the base controller, the actuator and the sensor respectively. For example the base controller algorithm is represented with six transfer functions (i.e. 3 inputs, 2 outputs):

$$C_{BC} = \begin{bmatrix} C_{BC11} & C_{BC12} & C_{BC13} \\ C_{BC21} & C_{BC22} & C_{BC23} \end{bmatrix} \quad (2.1)$$

In this paper, we consider linear time invariant (LTI) single input single output (SISO) systems. Furthermore in our formulation, transfer function elements are represented with a parameter vector of gains. For example, the first row, first column

transfer function of the base controller transfer function matrix C_{BC} , C_{BC11} , which will be defined in terms of a vector of gains, $\mathbf{x}_{BC11}$, as:

$$C_{BC11}(\mathbf{x}_{BC11}, s) = \frac{x_{BC11,m}s^{m+1} + \dots + x_{BC11,n+2}s + x_{BC11,n+1}}{x_{BC11,n}s^{n+1} + \dots + x_{BC11,2}s + x_{BC11,1}} \quad (2.2)$$

From Fig. 2.2 and using the notation presented in (2.1) and (2.2), we can write the equations representing individual signals as shown in Eqns. (2.3)-(2.9):

$$u_{ca} = C_{BC11}r + C_{BC12}y_{sc} + C_{BC13}y_{sa} \quad (2.3)$$

$$u_{cs} = C_{BC21}r + C_{BC22}y_{sc} + C_{BC23}y_{sa} \quad (2.4)$$

$$y_{as} = C_{A21}u_{ca} + C_{A22}y_{sa} \quad (2.5)$$

$$y_{ac} = C_{A31}u_{ca} + C_{A32}y_{sa} \quad (2.6)$$

$$y_{sc} = C_{S11}y + C_{S12}y_{as} + C_{S13}u_{cs} \quad (2.7)$$

$$y_{sa} = C_{S21}y + C_{S22}y_{as} + C_{S23}u_{cs} \quad (2.8)$$

$$q = C_{A11}u_{ca} + C_{A12}y_{sa} \quad (2.9)$$

One can then calculate the overall centralized controller (see Figs. 2.1 and 2.3), C

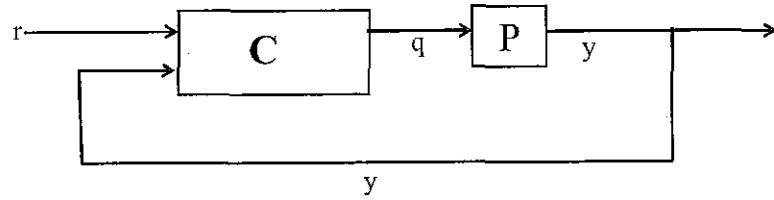


Figure 2.3: Representation of the Closed Loop System with Centralized controller, C

as:

$$\mathbf{C} = \begin{bmatrix} -C_{A11} & 0 & 0 & 0 & 0 & -C_{A12} \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & C_{BC13} & C_{BC12} & 0 \\ 0 & -1 & 0 & C_{BC23} & C_{BC22} & 0 \\ C_{A21} & 0 & -1 & 0 & 0 & C_{A22} \\ C_{A31} & 0 & 0 & -1 & 0 & C_{A32} \\ 0 & C_{S13} & C_{S12} & 0 & -1 & 0 \\ 0 & C_{S23} & C_{S22} & 0 & 0 & -1 \end{bmatrix}^{-1} \begin{bmatrix} C_{BC11} & 0 \\ C_{BC21} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & C_{S11} \\ 0 & C_{S21} \end{bmatrix} \quad (2.10)$$

where $\mathbf{C}$ is a transfer function matrix with the elements

$$\mathbf{C}(\mathbf{x}_1, \mathbf{x}_2) = \begin{bmatrix} C_1(\mathbf{x}_1) & C_2(\mathbf{x}_2) \end{bmatrix} \quad (2.11)$$

C_1 and C_2 are transfer functions with vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ of controller gains (Eqns. (2.12) and (2.13)) respectively,

$$C_1(\mathbf{x}_1, s) = \frac{x_{1,m}s^{m+1} + \dots + x_{1,n+2}s + x_{1,n+1}}{x_{1,n}s^{n+1} + \dots + x_{1,2}s + x_{1,1}} \quad (2.12)$$

$$C_2(\mathbf{x}_2, s) = \frac{x_{2,m}s^{m+1} + \dots + x_{2,n+2}s + x_{2,n+1}}{x_{2,n}s^{n+1} + \dots + x_{2,2}s + x_{2,1}} \quad (2.13)$$

and the control signal is given as $q = C_1r + C_2y$.

As shown in Eqn. (2.10), given component control transfer functions, it is always possible to calculate a unique overall centralized control $\mathbf{C}$. However, given

a centralized control there is more than one distributed control in terms of the C_{BC} , C_A and C_S , since the system of equations given in Eqns. (2.3)-(2.9) describe an underdetermined problem. The *underdetermined* nature of the distribution generates additional design freedom due to the new communication paths. One way of utilizing the new design freedom is to improve component swapping modularity in control systems which will be discussed in the next section.

2.1.5 Utilizing New Degree of Freedom to Improve Modularity of Control Systems

Ulrich and Tung [26] define modularity in terms of two characteristics of product design: (1) Similarity between the physical and functional architecture of the design and (2) Minimization of incidental interactions between physical components. They also state that *"component-swapping modularity occurs when two or more alternative basic components can be paired with the same modular components creating different product variants belonging to the same product family"*.

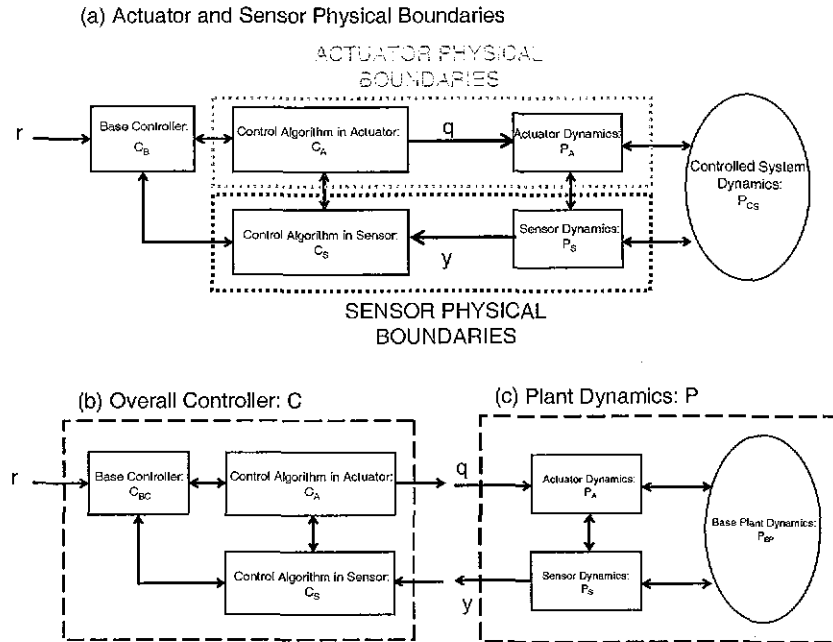


Figure 2.4: Control System Component Physical and Functional Boundaries.

Fig. 2.4 describes physical and functional boundaries for a networked control system with bi-directional communications and smart components. When a component change occurs (sensor or actuator shown in part a) both overall controller (part b) and overall plant dynamics (part c) are effected. Control systems with modularly swappable components can then be defined as systems in which the initial and final configurations due to a component change operate at their corresponding optimal performance. By using the additional design freedom, with networked control systems with bi-directional communications, it is possible to improve component swapping modularity of the system by containing the plant dynamics and corresponding control algorithm changes only in the affected components physical boundaries.

Since the definition of component swapping modularity is based on being able to use different components, the amount of modularity should be based on the size of the set of different component configurations. Also the default component should be a part of the set being considered. We present such a metric to quantify and maximize modularity in the next section.

There are many advantages of having a system with high component swapping modularity. Traditionally, a change in one of the components involves two sub-systems, the component and the base controller. By attaching the component related control to the component then re-work is done only in one sub-system. Minimizing rework can be vital for many companies: Butts and co-workers report that their systems engineering process involved up to as much as 60 % of re-engineering of requirements and defects during powertrain controller development process [4].

As discussed previously, uncertainty of information exchange (packet loss, delay, etc.) is the main focus of researchers interested in NCSs. It seems inevitable that with the improvements to network hardware and network protocols there will be increas-

ing utilization of networks for real-time applications. One good example of currently available hardware (at very high cost) is the SCRAMNet+ Network provided by Curtiss Wright Controls Inc. [30], which enables remotely located components to share a global memory. There has been research on sensor and actuator networks [23, 18] and networks of controllers [31]. However, there is very little research on the effects of information exchange between “smart” components in a networked feedback control system. Current research on modularity tackles hardware and software modularity separately. There is no interpretation and quantification of component swapping modularity for control system components in the previous literature.

The purpose of this paper is to outline the steps to investigate the potential benefits of *bi-directional communication* in a feedback control loop which is at the heart of so many control applications. In this section we have defined what we mean by bi-directional communication in networked control systems with smart components and the potential benefits of having such systems. Next, we focus on how to design such networked control systems with bi-directional communications and smart components. We then apply a problem solving methodology to maximize component swapping modularity in control systems implementation of which would benefit many economy of scale based industrial applications. We illustrate our ideas with an example: feedback control of driveshaft speed with a dc motor, tachometer and modularly swappable gearbox. Subsequently, we present our conclusions and research plan for future steps.

The original contributions of this paper include

- Definition and formulation of bi-directional communications in feedback control systems with distributed component controllers,
- Definition of component swapping modularity (i.e., concurrent hardware and

software modularity) for control system components,

- Problem formulation and example solution for maximizing actuator component modularity for single loop feedback systems.

2.2 Problem Formulation to Maximize Component Swapping Modularity

In this section the problem formulation to maximize actuator swapping modularity in a single input single output (SISO) linear time invariant (LTI) system will be discussed. First the optimization problem will be presented, then the formulation for major components of this optimization problem used to construct the cost function and constraints will be given. This section concludes with extensions of the formulation presented here, to other components and to more general networked systems.

2.2.1 Distribution Problem to Maximize Actuator Swapping Modularity

We define $\mathbf{p}_{CS}$, $\mathbf{p}_A$, $\mathbf{p}_S$ as parameter vectors representing the controlled system, actuator and sensor plant dynamics transfer functions, $P_{CS}(s)$, $P_A(s)$, $P_S(s)$, respectively as shown in Fig.2.4. Given nominal settings for these plant parameters denoted as $\mathbf{p}_{CS}^0$, $\mathbf{p}_A^0$, $\mathbf{p}_S^0$ for the controlled system, actuator and sensor respectively, we can formulate the distribution problem which maximizes actuator swapping modularity, M_A , to determine the parameters (or gains), $\mathbf{x}_{BC}$, $\mathbf{x}_A$, $\mathbf{x}_S$ of $\mathbf{C}_{dist}$ as:

$$\max_{\mathbf{x}_{BC}, \mathbf{x}_A, \mathbf{x}_S} M_A(\mathbf{p}_{CS}^0, \mathbf{p}_A^0, \mathbf{p}_S^0, \mathbf{x}_{BC}, \mathbf{x}_A, \mathbf{x}_S) \quad (2.14)$$

subject to

Distribution Constraint:

$$C_{dist}(\mathbf{x}_{BC}, \mathbf{x}_A, \mathbf{x}_S) = C_{des}(\mathbf{p}_{CS}^0, \mathbf{p}_A^0, \mathbf{p}_S^0) \quad (2.15)$$

Additional Constraints:

$$\mathbf{x}_{BC}^l \leq \mathbf{x}_{BC} \leq \mathbf{x}_{BC}^h \quad (2.16)$$

$$\mathbf{x}_A^l \leq \mathbf{x}_A \leq \mathbf{x}_A^h \quad (2.17)$$

$$\mathbf{x}_S^l \leq \mathbf{x}_S \leq \mathbf{x}_S^h \quad (2.18)$$

where definitions for C_{des} and C_{dist} and M_A will be given in the next section.

2.2.2 Components of the Distribution Problem

Desired Overall Controller Design Function, C_{des}

We present the design of the centralized controller $\mathbf{C}$ (see Eqns. (2.10)-(2.13)) as an optimization problem

$$\min_{\mathbf{x}_1, \mathbf{x}_2} J_C(P, \mathbf{C}) \quad (2.19)$$

subject to

$$\mathbf{g}(P, \mathbf{C}) \leq \sigma \quad (2.20)$$

where P and $\mathbf{C}$ are given in Fig. 2.3 and J_C , $\mathbf{g}$ are a control oriented cost function and set of constraints, such as settling time, control effort, etc. Then we can define

$C_{des}(\mathbf{p}_A, \mathbf{p}_S, \mathbf{p}_{CS})$ as

$$C_{des}(\mathbf{p}_A, \mathbf{p}_S, \mathbf{p}_{CS}) = C(\mathbf{x}_1^*, \mathbf{x}_2^*) \quad (2.21)$$

where $\mathbf{x}_1^*, \mathbf{x}_2^* = \operatorname{argmin} J_C(P, \mathbf{C})$ s.t. $\mathbf{g}(P, \mathbf{C}) \leq \sigma$. The $\mathbf{p}_{CS}, \mathbf{p}_A, \mathbf{p}_S$ are parameter vectors representing the controlled system, actuator and sensor parameters respectively.

Effective Centralized Controller Calculated from Component Controllers

C_{dist} is defined by C in Eqn. (2.10) in terms of the $C_{BC}(\mathbf{x}_{BC})$, $C_A(\mathbf{x}_A)$, $C_S(\mathbf{x}_S)$ are component controller transfer functions and $\mathbf{x}_{BC}$, $\mathbf{x}_A$, $\mathbf{x}_S$ are numerator and denominator polynomial coefficient vectors (e.g. see (2.2)) for these functions respectively, i.e.

$$\mathbf{x}_{BC} = \{\mathbf{x}_{BC11}, \mathbf{x}_{BC12}, \mathbf{x}_{BC13}, \mathbf{x}_{BC21}, \mathbf{x}_{BC22}, \mathbf{x}_{BC23}\} \quad (2.22)$$

$$\mathbf{x}_A = \{\mathbf{x}_{A11}, \mathbf{x}_{A12}, \mathbf{x}_{A21}, \mathbf{x}_{A22}, \mathbf{x}_{A31}, \mathbf{x}_{A32}\} \quad (2.23)$$

$$\mathbf{x}_S = \{\mathbf{x}_{S11}, \mathbf{x}_{S12}, \mathbf{x}_{S13}, \mathbf{x}_{S21}, \mathbf{x}_{S22}, \mathbf{x}_{S23}\} \quad (2.24)$$

$$\mathbf{x}_{dist} = \{\mathbf{x}_{BC}, \mathbf{x}_A, \mathbf{x}_S\} \quad (2.25)$$

Quantification of Actuator Swapping Modularity, M_A

Let Φ_A be a connected set of actuator plant parameter sets including the default parameter set, $\mathbf{p}_A^0$, that can be achieved by changing actuator control parameters within their limits given a distribution solution $\mathbf{x}_{dist}^0 = \{\mathbf{x}_{BC}^0, \mathbf{x}_A^0, \mathbf{x}_S^0\} \in [\mathbf{x}_{dist}^l, \mathbf{x}_{dist}^h]$, i.e.,

$$\begin{aligned} \Phi_A &= \{\forall \mathbf{p}_A \in \Phi_A \exists \mathbf{x}_A \in [\mathbf{x}_A^l, \mathbf{x}_A^h] \\ &\text{s.t. } C_{dist}(\mathbf{x}_{BC}^0, \mathbf{x}_A, \mathbf{x}_S^0) = C_{des}(\mathbf{p}_{CS}^0, \mathbf{p}_A, \mathbf{p}_S^0)\} \end{aligned} \quad (2.26)$$

where

- $\mathbf{p}_A^0 \in \mathbf{P}_A$,
- $\|\mathbf{p}_A^1 - \mathbf{p}_A^2\| < \epsilon \forall \mathbf{p}_A^1 \in \Phi_A \text{ and } \mathbf{p}_A^2 \in \Phi_A$

where ϵ is small. We then define the function M_A , the actuator modularity, as

$$M_A(\mathbf{p}_{CS}^0, \mathbf{p}_A^0, \mathbf{p}_S^0, \mathbf{x}_{BC}^0, \mathbf{x}_A^0, \mathbf{x}_S^0) = \int_{\Phi_A} d\mathbf{p}_A \quad (2.27)$$

The connected set assumption for set Φ_A provides convenience for numerically computing component swapping modularity. It allows us to calculate modularity using a directional search algorithm starting from the default configuration. Alternatively, component-swapping modularity can be calculated without this assumption (i.e., assume ϵ is ∞) for all feasible plant parameter ranges of component which can be cumbersome for some cases.

2.2.3 Extensions of the Maximizing Component Swapping Modularity Problem

It is possible to define similar sets (Φ_{CS} and Φ_S) for the other components in the system (i.e base controller and sensor). Therefore, it is possible to calculate sensor swapping modularity, M_S , and base controller swapping modularity, M_{BC} , accordingly. The distribution problem given in equations (2.14)-(2.18) can be solved to maximize the overall modularity, $M = M_{BC} + \rho_A M_A + \rho_S M_S$. By selecting different weights, ρ_A and ρ_S , designers can control the trade-off among modularity of components (i.e. base controller, actuator and sensor).

Generalizing the formulation of the optimization problem presented in Eqns. (2.14)-(2.18) to maximize component swapping modularity of general networked systems is possible by defining problem components C_{des} , C_{dist} and M_X .

2.3 Example

The purpose of this section is to present a simple, yet sufficiently complex, example that illustrates the potential benefits of bi-directional communication among “smart” components in a networked control system. Specifically, the feedback control of drive shaft speed for a system with a dc-motor and tachometer is considered. We first present the mathematical model of the plant, then define actuator, sensor and controlled system physical boundaries for the example. Next, we present an optimal

controller design method used to obtain optimal controller gain and investigate the effect of changing actuator plant parameters (potentially using better or worse actuators) and its effect on the optimal control. After presenting the controller design problem, we find the controller distribution solution for the optimal controller. The controller structure we will use for this example is also used in [6] to design ITAE optimal closed loop response for the feedback control system. Among many possible distribution solutions for the original plant and corresponding optimal controller, we pick the distribution which maximizes actuator component swapping modularity. Then, we give practical life-cycle implications of using a component swapping modular system as compared to the traditional one. Finally, we study the effect of assuming different communication and computational constraints on the system.

2.3.1 System Overview

A schematic of the drive shaft system is given in Fig. 2.5. It has three major plant components: dc-motor, driveshaft and tachometer. The gears and shafts are considered rigid and massless. Mathematical modeling of these components can be found in many textbooks such as [12] or [21].

A first order plant transfer function is obtained, when inductance in dc-motor armature is assumed small, with no measurement noise and disturbance torque:

$$\Omega_s(s) = \frac{K_s K_m / r_g}{J_{eq} R s + K_m^2 / r_g^2 + b_{eq} R} V(s) \quad (2.28)$$

where in the complex domain, the input voltage is denoted by $V(s)$ and the measured shaft rotational speed is $\Omega_s(s)$. Parameter values which will be used for this example are taken from experimentally evaluated values given in [20] except the gear reduction ratio, r_g . These values are shown in Table 2.1. The first order transfer function, $\Omega_s(s)/V(s)$ given in (2.28), can be defined in term of two parameters, i.e, the static

Parameter	Nom. Value in the Example
Motor torque constant, K_m	0.01 Nm/A
Vis. fric. coeff. of the eqv. system, b_{eq}	0.1 Nms/rad
Moment of inertia of the eqv. system, J_{eq}	0.1 kgm ²
Armature resistance, R	1 Ohm
Gearbox reduction ratio, r_g	1/10

Table 2.1: Nominal Plant Model Parameter Numerical Values

(dc) gain, K_{dc} and time constant, T :

$$K_{dc} = \frac{K_s K_m / r_g}{K_m^2 / r_g^2 + b_{eq} R} \quad (2.29)$$

$$T = \frac{J_{eq} R}{K_m^2 / r_g^2 + b_{eq} R} \quad (2.30)$$

It can be deduced by analyzing (2.29) and (2.30) that as $1/r_g$ increases in the range [5, 50] the bandwidth of the plant response increases.

Physical boundaries for the actuator, controlled system and sensor are defined as shown in Fig. 2.5. The dc-motor and the gearbox provide the actuation to the system. The tachometer is the sensor which measures the driveshaft speed, and the driveshaft with the drive pulley is considered to be the controlled system. According to the boundaries given in Fig. 2.5 using a different actuator would potentially change the values of R , K_m , r_g in Eqn. (2.28) as well as the equivalent inertia, J_{eq} , and equivalent viscous friction, b_{eq} , of the system.

2.3.2 Optimization Problem Components

Formulation for C_{des}

Design of a controller given the plant dynamics is a widely studied field and many controller design methods exist (for example [21]). Here, without loss of generality, we consider a particular controller structure as shown in Fig. 2.6. Given plant model parameters $\mathbf{P} = \{K_m, r_g, J_{eq}, b_{eq}, R\}$ and controller parameters $\mathbf{C} = \{c_1, c_2, c_3\}$ shown in Fig. 2.6, a control design problem can be formulated as given in (2.31)-(2.34).

$$\min_{\mathbf{C}} t_s(\mathbf{C}, \mathbf{P}) \quad (2.31)$$

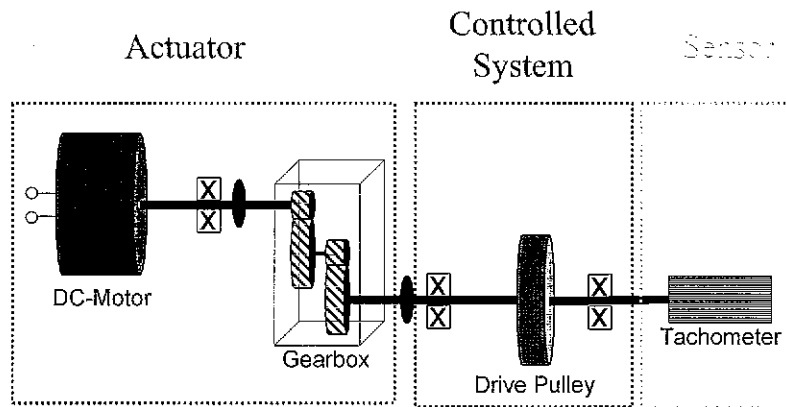


Figure 2.5: Physical Boundaries for Model Parameters

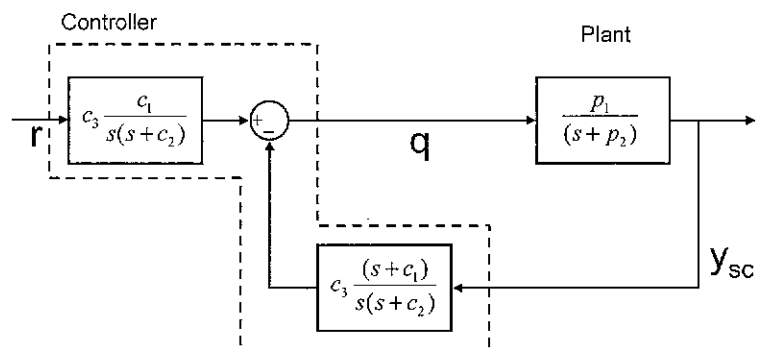


Figure 2.6: Closed loop system block diagram with controller.

subject to

$$M_p(\mathbf{C}, \mathbf{P}) \leq \sigma_{M_p} \% \quad (2.32)$$

$$u_{max}(\mathbf{C}, \mathbf{P}) \leq \sigma_{u_{max}} V \quad (2.33)$$

$$0 \leq c_i \leq 20000 \quad (i = 1, 2, 3) \quad (2.34)$$

where t_s is the 2% settling time, M_p is the maximum percent overshoot, u_{max} is the maximum control signal (i.e. $\max(q(t))$ over time, t) For the problem shown in (2.31)-(2.34) and using the nominal values for the example problem ($K_m = 0.01$, $r_g = 1/10$, $J = 0.1 \text{ kgm}^2$, $R = 1\Omega$ and $b = 0.1 \text{ Nms/rad}$) and setting $\sigma_{M_p} = 5\%$ and $\sigma_{u_{max}} = 250V$ optimal controller parameters are determined to be:

c1 = 3.6104
c2 = 1982.1474
c3 = 12707.8364

Parametric solutions for the optimal values of the controller gains, c_1 , c_2 and c_3 which describe the optimal controller, are found by solving the optimization problem given in (2.31)-(2.34) for different $1/r_g$ values in the range of $[5, 50]$.

$$c_1(r_g) = 0.35544(1/r_g) + 0.0934 \quad (2.35)$$

$$\begin{aligned} c_2(r_g) = & -1.4407 * 10^{-6}(1/r_g)^5 + 3.8305 * 10^{-4}(1/r_g)^4 \\ & -3.2074 * 10^{-2}(1/r_g)^3 + 1.1209(1/r_g)^2 \\ & -15.897(1/r_g) + 2054 \end{aligned} \quad (2.36)$$

$$\begin{aligned} c_3(r_g) = & 1.1983 * 10^{-3}(1/r_g)^5 - 1.8097 * 10^{-1}(1/r_g)^4 \\ & +10.447(1/r_g)^3 - 289.35(1/r_g)^2 \\ & +3.9092 * 10^3(1/r_g) - 6263.8 \end{aligned} \quad (2.37)$$

Formulation for C_{des} is given in (2.38)

$$\mathbf{C}_{des}(r_g) = \begin{bmatrix} \frac{c_3 c_1}{s^2 + c_2 s} & \frac{-c_3 s - c_3 c_1}{s^2 + c_2 s} \end{bmatrix} \quad (2.38)$$

where c_1, c_2, c_3 are given in (2.35)-(2.37).

Formulation of C_{dist}

In order to define $C_{dist}(s)$, and the distribution constraint in Eqn. (2.15), properly so that a solution algorithm can be implemented, we have to set component controller transfer function matrices $\mathbf{C}_{BC}(\mathbf{x}_{BC})$, $\mathbf{C}_A(\mathbf{x}_A)$, $\mathbf{C}_S(\mathbf{x}_S)$.

One example is given in Eqn. (2.39) to (2.40).

$$\mathbf{C}_A(\mathbf{x}_A) = \begin{bmatrix} \frac{x_{A11,2}}{x_{A11,1}} & \frac{x_{A12,2}}{x_{A12,1}} \\ \frac{x_{A21,2}}{x_{A21,1}} & \frac{x_{A22,2}}{x_{A22,1}} \\ \frac{x_{A31,2}}{x_{A31,1}} & \frac{x_{A32,2}}{x_{A32,1}} \end{bmatrix} \quad (2.39)$$

$$\mathbf{C}_S(\mathbf{x}_S) = \begin{bmatrix} \frac{x_{S11,2}}{x_{S11,1}} & \frac{x_{S12,2}}{x_{S12,1}} & \frac{x_{S13,2}}{x_{S13,1}} \\ \frac{x_{S21,2}}{x_{S21,1}} & \frac{x_{S22,2}}{x_{S22,1}} & \frac{x_{S23,2}}{x_{S23,1}} \end{bmatrix} \quad (2.40)$$

where $x_{A11,1} = x_{A12,1} = x_{A21,1} = x_{A22,1} = x_{A31,1} = x_{A32,1} = 1$ and $\mathbf{C}_S$ has identical rows (i.e. $x_{S1k,2} = x_{S2k,2}$ for $k = 1, 2, 3$ with $x_{S11,1} = x_{S12,1} = x_{S13,1} = x_{S21,1} = x_{S22,1} = x_{S23,1} = 1$). Also, the second order transfer functions in $\mathbf{C}_{BC}$ (not shown here due to lack of space) have identical rows (i.e. $x_{BC1k,i} = x_{BC1k,i}$ for $i = 4, 5, 6$ and $k = 1, 2, 3$) sharing a common denominator (i.e. $x_{BCj1,i} = x_{BCj2,i} = x_{BCj3,i}$ for $i = 1, 2, 3$ and $j = 1, 2$).

It is important to note that formulating $\mathbf{C}_{BC}(\mathbf{x}_{BC})$, $\mathbf{C}_A(\mathbf{x}_A)$, and $\mathbf{C}_S(\mathbf{x}_S)$ as given in (2.39) and (2.40) implies assumptions about the amount of computation and communication by each components. We assume the maximum order of transfer functions as two for the base controller, while the actuator and sensor controllers are just gains, i.e transfer functions of order 0. Also since the first and second rows of

C_{BC} and C_S are the same, we understand this definition represent a communication scheme where these components *multicast*. That is, each component is sent the same message as the other two. A block diagram for this configuration is shown in Fig. 2.7.

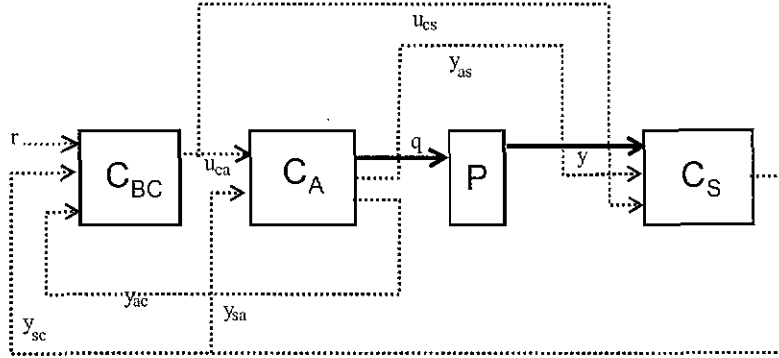


Figure 2.7: Example Configuration: Base Controller, Sensor multicast

Formulation of M_A

In order to keep the example problem simple we are going to use only the gearbox reduction ratio, r_g , as our actuator parameter while other actuator parameters are held constant at their nominal values (i.e. $\mathbf{p}_A = [R^0, K_m^0, r_g, J_m^0, b_{eq}^0]$). This represents a case where the only change in the actuator component comes from using a different gearbox. The list of parameters used in the general problem formulation for actuator modularity, M_A (i.e. Eqn. (4.1)) and distribution problem presented in Eqns. (2.14)-(2.18) and their definitions in the example problem is given in Table 2.2.

General Formulation	Example Formulation
$\mathbf{p}_{CS}^0$	$[J_d^0, b_d^0]$
$\mathbf{p}_A^0$	$[R^0, K_m^0, r_g^0, J_m^0, b_{eq}^0]$
$\mathbf{p}_S^0$	$[K_s^0]$
$\mathbf{p}_A$	$[R^0, K_m^0, r_g, J_m^0, b_{eq}^0]$

Table 2.2: List of Parameters for Example Distribution Solution

2.3.3 Solution to the Optimization Problem

The optimization problem given in Eqns. (2.14)-(2.18) and Table 2.2 was solved using the Matlab Optimization Toolbox. A traditional controller using unidirectional communications is taken as the initial condition with the nominal values presented in Table 2.1.

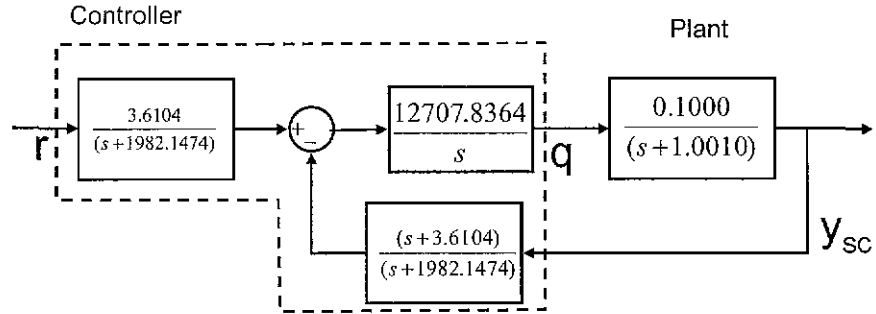
For the optimal distribution solution given in (2.35)-(2.37) the optimal actuator swapping modularity, M_A^* , turns out to be 27 and an optimal control can be obtained for all values of $5 \leq (1/r_g) \leq 32$.

A summary for the optimization problem is given in Fig. 2.8.

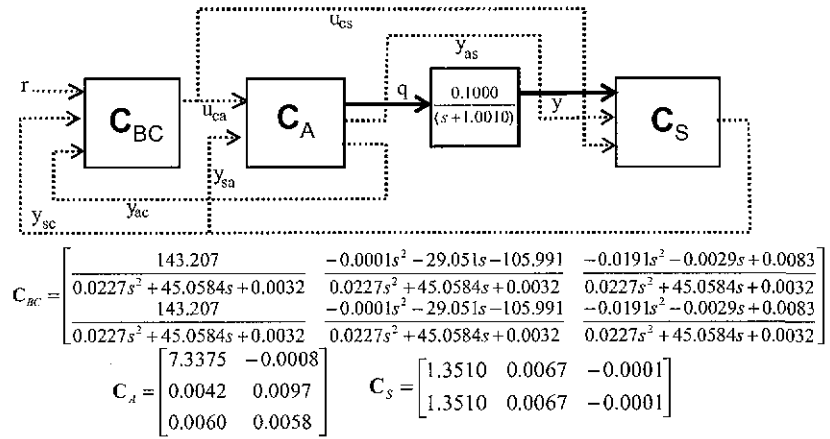
Engineering Analysis of Solutions

It can be calculated from (2.29) and (2.30) that, an increase in $1/r_g$ value results in faster responding dc-motor/gearbox component when other actuator plant parameters remains the same. Systems with faster actuators use less controller effort, q , in steady state conditions (Fig. 2.8).

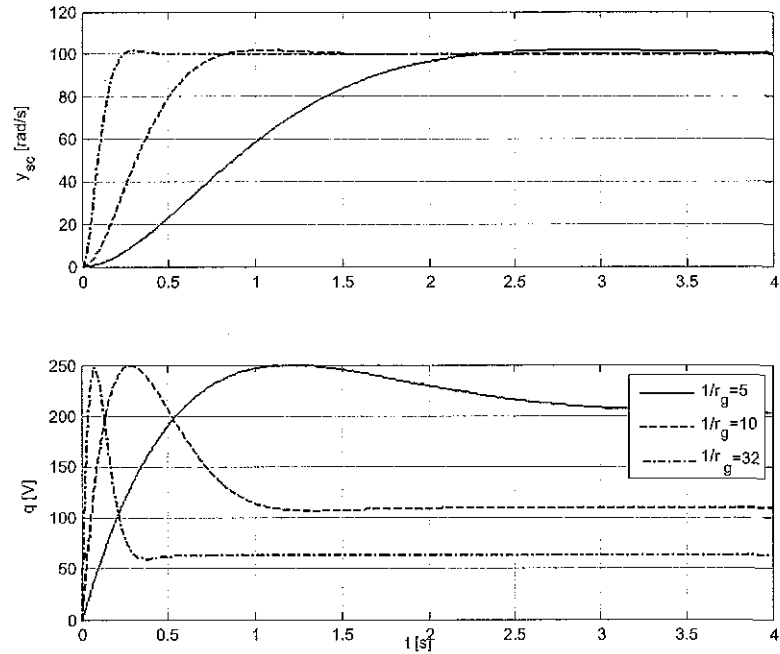
The solution of the distribution problem presented shows significant improvement in the actuator modularity of the control system: the original system configuration with uni-directional communications only provides the optimal controller for $1/r_g = 10$, whereas the communication configuration described in Fig. 2.7 can provide optimal solutions for all $1/r_g$ values in the range of $5 \leq 1/r_g \leq 32$ (i.e $M_A^* = 27$) by changing only the actuator controller, $C_A(s)$. The traditional controller, and the



(a) Centralized Controller using Traditional Feedback Communications



(b) Distributed Controller utilizing Bi-directional Communications in NCS



(c) Closed Loop System Response

Figure 2.8: Summary of Results.

controller designed for swapping modularity are compared in Fig. 2.8. We can see from part (c) that achievable optimal closed loop performance (i.e. smaller settling time) improves as $1/r_g$ value increases. This result shows that one can contain the desired optimal controller changes as $1/r_g$ values change from 5 to 32 by just modifying the gains $\mathbf{x}_A$ in the actuator controller $\mathbf{C}_A(s)$.

One way to investigate the practical implications of these results would be to assume we will be using the same dc-motor but different gearboxes. We can assume that these gearboxes will have the same acoustical and mechanical properties but different reduction ratios (for example $r_g = 1/5$, $r_g = 1/10$, $r_g = 1/25$ etc.). Having a gearbox with a larger ratio with the same mechanical loss and acoustical properties in the same packaging constraints will cost more since it involves better engineering design and materials to build the gearboxes from (i.e the higher gear reduction , the more expensive the actuator component). Furthermore, a higher gear reduction leads to better disturbance torque reduction. We can look at the results of our example problem in terms of the four expected impacts of improved component swapping modularity on industrial applications:

1. Sustainable maintenance and/or upgrade of an particular end product:

Designers make many decisions to build the best possible performing system within the allowed cost requirements, and cost of components change over time. Having a range of actuator alternatives available for use improves the life-cycle cost of the system.

2. Deploying controlled system platform based algorithms:

If we assume that this driveshaft system is part of a platform design, different gearbox designs will be selected for different end-products since they will have different total cost structures and quality requirements determined by the target

customers.

3. Deploying control algorithms for different builds of the same product:

When the driveshaft system is produced and sold across the globe in different market, it will be subjected to different performance standards and regulations (for example expected average disturbance torque is different for different locations). Cost of similar components from different suppliers will be different.

4. Reducing costs by developing highly customizable but less variant components (supplier point of view):

Since the suppliers buy their materials in bulk to be able to produce components in lower costs they can use studies such as presented in Section 2.3.3.

Effect of Communication and Computation on Distribution Problem Solution

In order to formulate C_{des} properly for numerical solution we have made assumptions about the elements of the transfer function matrices representing the control algorithms in component controllers in Section 2.3.2 of this paper. It can be seen from (2.39) and (2.40) that these equations actually imply assumptions related to amount of maximum computation (order of the transfer function polynomials) and communication (dependent or independent rows) capability of a particular component. In this section, we are going to present our results on using different computation and communication capabilities for the networked control system with smart components. Since introduction of bi-directional communications in networked control systems

Conf. Index (CI)	#of msg	Description
0	2	Traditional unidirectional comm.s
1	3	Base Cont., Act., Sens. multicast
2	4	Base Cont., Sens. multicast (Fig.2.7)
3	5	Only Sensor multicast
4	6	Full Bi-dir. Comm. (No Multicast)

Table 2.3: List of Communication Configurations

is our primary focus we are going to look at various configurations of communications including traditional uni-directional communications. These configurations are summarized in Table 2.3.

Also, for each communication configuration case we would like to see the effect of changing the computational assumptions for the component controllers. Table 2.4 describes two test cases we used for our analysis. Case ID NBC24NA0NS0 refers to the case where base controller matrix element are second order transfer functions maximum (i.e. $N_{BC} = 2$) and actuator and sensor controllers consist of gains (i.e. $N_A = N_S = 0$). The second case, NBC1NA1NS0, base controller and actuator controller elements are both first order transfer functions maximum (i.e. $N_{BC} = N_A = 1$) with sensor controller matrix consists of gains (i.e. $N_S = 0$).

Case ID	N_{BC}	N_A	N_S
NBC2NA0NS0	2	0	0
NBC1NA1NS0	1	1	0

Table 2.4: Computation per components

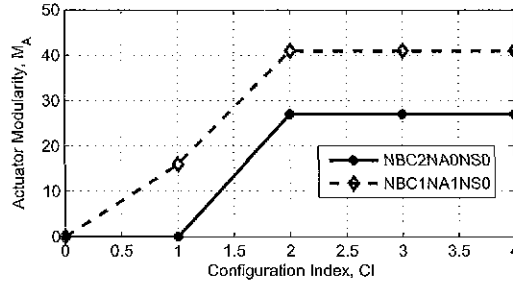


Figure 2.9: Effect of Changing Communication and Computation on Actuator Modularity

Results in Fig. 2.9 show the change in optimal distribution solutions by assuming different communication and computational constraints in distribution constraint given in Eqn. (2.15). When the communication and computation constraints are relaxed (i.e. higher order actuator control and more communication), modularity increases to $M_A^* = 41$. Increasing the amount of allowed computation increases the

modularity in the similar communication setups (i.e $CI = 1$ and beyond). Results also show that addition of communication does not always improve the modularity of the component. Therefore, an incremental solution process is beneficial to find the simplest communication and computation solution that fits the application needs.

2.4 Summary and Conclusions

Availability of low cost electronics has led to a new breed of control system components, so called “smart” components, which can perform control actions in the actuator and sensor components as well as in the controller. “Smart” components can communicate bi-directionally in networked control systems (NCSs). We observe opportunities in improving control system performance and design due to the decentralized yet more *connected* nature of these systems [5, 6].

In this paper we outlined the steps to investigate the potential benefits of bi-directional communication in a feedback control loop, which is at the heart of so many control applications. In the first section we defined bi-directional communications in networked control systems with smart components and the potential benefits of having such systems. Then, we focused on how to design such networked control systems with bi-directional communications and smart components and utilize the additional design freedom with the bi-directional communication capabilities. We then apply a problem solving methodology to maximize component swapping modularity in control systems implementation of which would benefit many economy of scale based industrial applications. We illustrated our ideas with feedback control of driveshaft speed with tachometer example in which networked controller system design with bi-directional communication alternative showed significant improvements in component swapping modularity compared to the traditional uni-directional feed-

back loop design. Specifically, the original system configuration with unidirectional communications only provides the optimal controller for $1/r_g = 10$, whereas the communication configuration described in Fig. 2.8 can provide optimal solutions for all $(1/r_g)$ values in the range of $5 \leq 1/r_g \leq 32$ (i.e. $M_A^* = 27$) by changing only the gains in the actuator controller, $C_A(s)$. When the communication and computation constraints are relaxed, modularity increases up to $M_A^* = 41$. We believe the simplest configuration (less communication, less transfer function order) within the desired actuator domain is the preferred overall solution to the distribution problem.

Original contributions of this paper include:

- Definition and formulation of bi-directional communications in feedback control systems with distributed component controllers,
- Definition of component swapping modularity (i.e concurrent hardware and software modularity) for control system components,
- Problem formulation and example solution for maximizing actuator component modularity for single loop feedback systems.

Our future research work will include adding sensor-swapping modularity and overall component swapping modularity to our formulation, including computational and communication effects in discrete-time and solving the component swapping modularity problem for multi-input, multi-output (MIMO) feedback control systems. We have already extended our method to handle more general cases, and applied it to a complex engine control problem, the results of which are partially presented in [7].

Acknowledgments

The authors would like to thank Ilya V. Kolmanovsky and Shifang Li for their valuable comments during the preparation of this paper.

Bibliography

- [1] Open DeviceNet Vendors Association. Odva website. www.odva.org.
- [2] K. J. Astrom. Present developments in control applications. Keynote Paper at the IFAC 50th Anniversary Meeting, Heidelberg, 2006.
- [3] J. Baillieul and P.J. Antsaklis. Control and communication challenges in networked real-time systems. *Proceedings of the IEEE*, 95(1):9–28, Jan. 2007.
- [4] K. Butts, J. Cook, C. Davey, J. Friedman, P. Menter, S. Raman, N. Sivashankar, P. Smith, and S. Toeppe. Automotive powertrain controller development using cacs. In Tariq Samad, editor, *Perspectives in Control: New Concepts and Applications*. IEEE Press, 2001.
- [5] M. Cakmakci and A. G. Ulsoy. Networked control systems with bi-directional communication among "smart" components. *Proceedings of the American Control Conference*, pages 627–632, June 2005.
- [6] M. Cakmakci and A. G. Ulsoy. Improving component swapping modularity using bi-directional communication in networked control systems. *Proceedings of the International Symposium on Flexible Automation*, July 2006.
- [7] M. Cakmakci and A. G. Ulsoy. Modular discrete optimal mimo controller for a vct engine. *Proceedings of the American Control Conference*, June 2009. (submitted).
- [8] M. Chiodo, P. Giusto, A. Jurecska, H.C. Hsieh, A. Sangiovanni-Vincentelli, L. Lavagno, and M. Marelli. Hardware-software codesign of embedded systems. *Micro, IEEE*, 14(4):26–36, 1994.
- [9] ControlNET.org. Controlnet protocol website. www.controlnet.org.
- [10] Flowserve Corporation. Flowserve website. www.flowserve.com.

- [11] Fieldbus Foundation. Fieldbus website. www.fieldbus.org.
- [12] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. *Feedback Control of Dynamic Systems*. Prentice Hall, 2002.
- [13] J.P. Hespanha, P. Naghshtabrizi, and Yonggang Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162, Jan. 2007.
- [14] Dust Networks Inc. Dust networks website. www.dustnetworks.com.
- [15] Profibus International. Profinet website. www.profinet.com.
- [16] Y. Li, L. Yang, and G. Yang. Network-based coordinated motion control of large-scale transportation vehicles. *IEEE/ASME Transactions on Mechatronics*, 12(2):208–215, 2007.
- [17] F. Lian, J. Moyne, and D. Tilbury. Performance evaluation of control networks: Ethernet, controlnet and devicenet. *IEEE Control Systems Magazine*, pages 66–83, February 2001.
- [18] J. Liu, M. Chu, J. J. Liu, J. E. Reich, and F. Zhao. State-centric programming for sensor and actuator network systems. *IEEE Pervasive Computing*, 2(4):50–62, October/December 2003.
- [19] R.T. McLaughlin. The immunity to rf interference of a can system. *Proceedings of the IEE Colloquium on Integrity of Automotive Electronic Systems*, pages 4/1–4/8, 1993.
- [20] University of Michigan and Carnegie Mellon University. Control tutorials for matlab. www.engin.umich.edu/group/ctm/.
- [21] Katsuhiko Ogata. *Modern Control Engineering*. Prentice Hall, 1990.

- [22] M.S. Salerno and A.V.C. Dias. Product design modularity, modular production, modular organization: The evolution of modular concepts. *Actes du GERPISA no 33*, pages 61–73, 2002.
- [23] B. Sinapoli, C. Sharp, L. Schenato, S. Schaffert, and S. Sastry. Distributed control applications within sensor networks. *Proceedings of the IEEE*, 91(8):1235–1246, August 2003.
- [24] J.-P. Thomesse. Fieldbus technology in industrial automation. *Proceedings of the IEEE*, 93(6):1073–1101, June 2005.
- [25] Y. Tipsuwan and Mo-Yuen Chow. On the gain scheduling for networked pi controller over ip network. *IEEE/ASME Transactions on Mechatronics*, 9(3):491–498, 2004.
- [26] K. Ulrich and K. Tung. Fundamentals of product modularity. In A. Sharon, editor, *Issues in Mechanical Design International*, pages 73–79. ASME, New York, 1991.
- [27] G.C. Walsh, H. Ye, and L. Bushnell. Stability analysis of networked control systems. *IEEE Transactions on Control Systems Technology*, 10(3):438–446, May 2002.
- [28] G.C. Walsh and Hong Ye. Scheduling of networked control systems. *IEEE Control Systems Magazine*, pages 57–65, February 2001.
- [29] B. A. Warneke and K.S.J. Pister. Exploring the limits of system integration with smart dust. *Proceedings of the ASME International Mechanical Engineering Congress and Exposition*, pages 17–22, November 2002.
- [30] Scramnet+ Website. Curtiss wright controls inc. www.systran.com/scmain.html.

- [31] K.C. Wong and J.H.van Schuppen. Decentralized supervisory control of discrete-event systems with communication. *Proceedings of the International Workshop on Discrete Event Systems*, pages 284–289, 1996.
- [32] J.K. Yook, D.M. Tilbury, and N.R. Soparkar. Trading computation for bandwidth: Reducing communication in distributed control systems using state estimators. *IEEE Transactions on Control Systems Technology*, 10(4):503–518, 2002.
- [33] W. Zhang, M.S. Branicky, and S. M. Phillips. Stability of networked control systems. *IEEE Control Systems Magazine*, pages 84–99, February 2001.

CHAPTER III

Modular Discrete Optimal MIMO Controller for a VCT Engine

Abstract: In early computer control, only one centralized computer was responsible for executing the algorithms for a particular system. Increasingly, computer control algorithms reside inside individual system components in a distributed fashion. Variable Camshaft Timing (VCT) is an appealing feature for automotive engines because it allows optimization of the cam timing over a wide range of operating conditions. In this paper, a method to distribute the discrete-time MIMO controller for the VCT Engine to improve the component swapping modularity of the VCT Actuator and the EGO Sensor components using network communications is presented. First a discrete-time LQG controller was designed then this controller was distributed to the ECU, the VCT controller and the EGO sensor controller in order to improve the component swapping modularity of the system. A control oriented *pre-optimization* technique, which simplifies the optimization problem and a candidate solution was devised to maximize component modularity.

3.1 Introduction

Although in the early days of computer control, only one centralized computer was responsible for executing the algorithms for a particular system, increasingly, com-

puter control algorithms reside inside individual system components in a distributed fashion. For example, there are up-to 80 micro-controllers in today's high-end vehicles and it is expected that by 2010, 90% of all computer code will reside in such embedded systems [1].

As control systems are implemented in an increasingly distributed fashion, modularity of the overall system becomes an important design decision. Ulrich and Tung [15] define modularity in terms of two characteristics of product design: (1) Similarity between the physical and functional architecture of the design and (2) Minimization of incidental interactions between physical components. They also state that “*component-swapping modularity occurs when two or more alternative basic components can be paired with the same modular components creating different product variants belonging to the same product family*”.

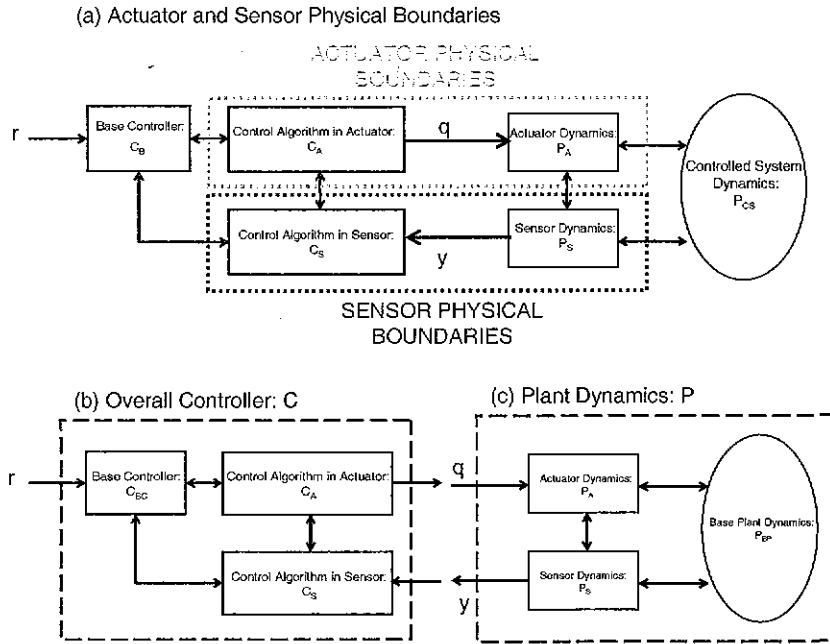


Figure 3.1: Control System Component Physical and Functional Boundaries.

Fig. 3.1 describes physical and functional boundaries for a networked control sys-

tem with bi-directional communications and smart components. When a component change occurs (sensor or actuator shown in Fig. 3.1-a) both overall controller (Fig. 3.1-b) and overall plant dynamics (Fig. 3.1-c) are affected. Control systems with modularly swappable components can then be defined as systems in which the initial and final configurations due to a component change operate at their corresponding optimal performance. By using the additional design freedom, with networked control systems with bi-directional communications, it is possible to improve component swapping modularity of the system by containing the plant dynamics and corresponding control algorithm changes only in the affected components physical boundaries [2, 3].

Variable Camshaft Timing (VCT) is an appealing feature for automotive engines because it allows optimization of the cam timing over a wide range of operating conditions [11]. VCT schemes not only improve fuel economy [4, 5, 7], but also reduce emissions [8, 13] while improving full load performance [6].

VCT schemes increase internal residual gas by affecting the intake, combustion and exhaust phases of the engine cycle. Increase in internal residual gas reduces the combustion temperature which decreases nitrogen oxide, NO_x , formation. The internally recirculated exhaust gas is rich in unburned hydrocarbons, HC , which can be burned in the next cycle. Application of VCT schemes, since they require higher manifold pressure, decrease pumping losses which results in improved fuel economy. However, dilution of the in-cylinder mixture adversely affects the engine torque response. These factors define the trade-off between good emissions and good drivability for VCT engines.

In [14], the detailed operation of a continuous variable cam timing component is described. This variable camshaft timing system works on the principle of sliding

helical gears controlled by a hydraulic piston. As the helical gear moves back and forth, the rotation of the camshaft relative to the timing pulley changes (Fig. 3.2 based on [14]).

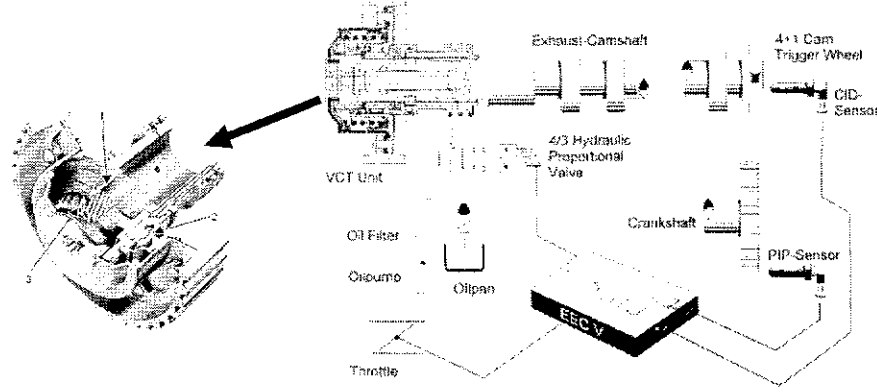


Figure 3.2: A Continuous VCT System.

In this paper, we present the design and distribution of a discrete MIMO controller for a variable camshaft timing (VCT) engine. The resulting distributed controllers maximize the component swapping modularity of the *smart* VCT Unit and *smart* EGO Sensor. In the next section, the VCT engine model used in the controller design and the discrete MIMO controller design will be presented. Next, distribution of the controller design to engine control unit (i.e. ECU), VCT Controller and EGO controller to maximize component-swapping modularity will be discussed. Subsequently, we present our conclusions and research plan for future steps in the last section.

3.2 Engine Modeling and MIMO Control Design

3.2.1 VCT Engine Model

The development of a continuous, non-linear, low-frequency, phenomenological and control oriented VCT engine model was discussed in [10] based on the model structure given in [9] and other references. The input/output relationship of the

plant model developed is given in Fig. 3.3.

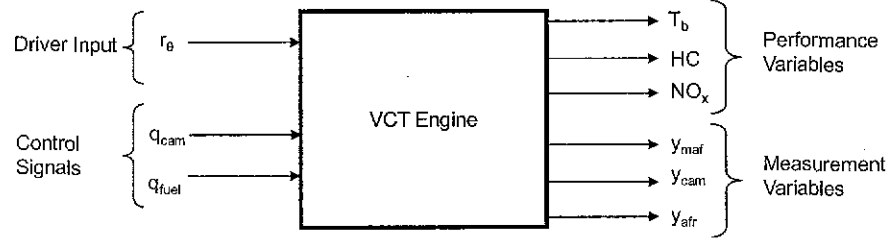


Figure 3.3: Input Output relationship of the dynamic plant model for control development.

In previous work an experimental setup was used to develop relationships for the engine breathing process, torque generation and feedgas HC and NO_x emissions were developed. Details of this work will not be discussed here, and the reader is referred to [10] and [11].

3.2.2 Dynamics of the VCT Unit and EGO Sensor

In order to model the VCT actuator dynamics, a first order transfer function (3.1) where $\tau_{vct,act} = 0.0371$ will be used:

$$Y_{cam,actual}(s) = \frac{-0.013s + \tau_{vct,act}}{s + \tau_{vct,act}} Q_{cam}(s) \quad (3.1)$$

where Q_{cam} is the commanded cam phase angle. Non-minimum phase zero observed in Eqn. (3.1) is an artifact of the identification process.

For the VCT sensor, a delay of two fundamental sampling periods was assumed, modeled as a first order Pade approximation with parameter $\tau_{vct,sen}$. For an n cylinder engine at a speed of N rpm the fundamental sampling rate is defined as [10]:

$$\Delta T = \frac{120}{Nn} \quad (3.2)$$

The dynamics of the EGO sensor is modeled as first order with a time constant $\tau_{ego} = 70\text{ms}$:

$$Y_{afr}(s) = \frac{1/\tau_{ego}}{s + 1/\tau_{ego}} Y_{afr,exhaust}(s) \quad (3.3)$$

The Matlab/Simulink plant model for the VCT engine was developed based on the information and regression data given in [10]. The response of the overall model is then compared to open loop model results in [10, 12] for validation.

3.2.3 Discrete-time MIMO Controller Design

The dynamic engine model described in the previous section is linearized around the nominal inputs, i.e CAM Angle = 10°, Fuel = 0 grams and Throttle Angle = 9.33°, and the corresponding steady state internal states using Matlab/Simulink.

The linearized model is then discretized with a sampling period ΔT to obtain

$$\mathbf{x}(k+1) = \mathbf{A}_d \mathbf{x}(k) + \mathbf{B}_d \mathbf{u}(k) + \mathbf{B}_{r1,d} r_\theta(k) \quad (3.4)$$

$$\mathbf{y}(k) = \mathbf{C}_d \mathbf{x}(k) + \mathbf{D}_d \mathbf{u}(k) + \mathbf{B}_{r1,d} r_\theta(k) \quad (3.5)$$

where

$$\mathbf{A}_d = \begin{bmatrix} \mathbf{A}_{d1} & \mathbf{A}_{d2} \end{bmatrix} \quad (3.6)$$

$$\mathbf{A}_{d1} = \begin{bmatrix} 0.8984 & -0.01638 & 0.02042 & 0.0025 & 0.0003 \\ 0 & 0.8169 & 0 & 0 & 0 \\ 0 & 0.1153 & 0.3679 & 0 & 0 \\ 0 & -0.00107 & 0.0307 & 0.9435 & 0.022 \\ 0 & 0 & 0 & 0 & 0.7575 \\ 0 & 0.0045 & 0 & 0 & 0 \\ 0 & 1.00E-06 & -7.00E-05 & 0.0002 & 1.00E-05 \\ 0 & -0.0238 & -0.0032 & 0.0456 & 0.0054 \\ 0 & -7.00E-05 & -0.0003 & 0.0005 & 5.00E-05 \end{bmatrix}$$

$$\mathbf{A}_{d2} = \begin{bmatrix} 0.1543 & -656.314 & -0.8944 & 10.1589 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.944 & 0 & 0 & 0 \\ 0 & 0.3679 & 0 & 0 \\ 0.2551 & -1240 & 0.466 & -14.0665 \\ 0.0011 & -6.3646 & 0.0053 & 0.9407 \end{bmatrix}$$

$$[B_{r1,d}, B_d] = \begin{bmatrix} 0.00038175 & -5.354e-005 & 0.14931 \\ 0 & 0.0067904 & 0 \\ 0 & 0.00046453 & 0 \\ 0.00024734 & -7.5544e-006 & 0 \\ 0.018606 & 0 & 0 \\ 0.0024429 & 1.5316e-005 & 0 \\ 2.2866e-007 & 2.2808e-008 & 0.0047409 \\ 0.00052672 & -6.6406e-005 & 1.4671 \\ 1.5265e-006 & -1.3251e-007 & 0.014705 \end{bmatrix} \quad (3.7)$$

$$C_d = \begin{bmatrix} 14.2857 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -27.3095 & 266.6667 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (3.8)$$

$$D_d = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0.013 & 0 \end{bmatrix} \quad (3.9)$$

Since maintaining the stoichiometric air fuel ratio, and zero steady state error in cam timing, is important during throttle angle changes, integral control of the plant outputs is implemented. This is done by augmenting the state vector with the integral of the output tracking errors:

$$\begin{aligned} \hat{\mathbf{x}}(k+1) = & \begin{bmatrix} \mathbf{A}_d & \mathbf{0} \\ \Delta T \mathbf{C}_d & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x}(k) \\ \mathbf{x}_I(k) \end{bmatrix} + \begin{bmatrix} \mathbf{B}_d \\ -\Delta T \mathbf{C}_d \end{bmatrix} \mathbf{u}(k) \\ & + \begin{bmatrix} \mathbf{B}_{r1,d} & \mathbf{0} \\ \mathbf{0} & -T_s \mathbf{I} \end{bmatrix} \begin{bmatrix} r_\theta(k) \\ r_{cam}(k) \\ r_{afr}(k) \end{bmatrix} \end{aligned} \quad (3.10)$$

By using the discrete-time linear system above an LQR controller with the state feedback gains, K_d was obtained:

$$\mathbf{K}_d^T = \begin{bmatrix} -0.00029813 & -0.013001 \\ 22.2911 & 0.0006195 \\ 126.643 & -0.0087846 \\ 2.84E-05 & 0.025793 \\ -2.79E-06 & 0.0030435 \\ -0.00010603 & -0.038505 \\ 1.6371 & 65.1827 \\ -0.00029203 & -0.0079145 \\ -0.065519 & -2.647 \\ 0.0001704 & 0.0085496 \\ -40.1194 & 0.00095088 \end{bmatrix} \quad (3.11)$$

Since the only measurements available are the cam timing measurement and the air-fuel measurement, a Kalman Filter is designed to estimate the remaining states with the gains:

$$\mathbf{L}_d = \begin{bmatrix} 0.095152 & 8.486e-005 \\ -9.9756e-011 & 0.00020019 \\ -1.8683e-011 & 0.0014151 \\ -2.6077e-008 & 0.00012182 \\ -1.0599e-010 & -4.7379e-015 \\ 2.8741e-008 & 1.495e-005 \\ -9.9113e-007 & -2.9073e-007 \\ -0.015256 & -6.9183e-006 \\ 0.00014419 & -1.3294e-006 \end{bmatrix} \quad (3.12)$$

The closed loop system with the discrete MIMO controller is given in Fig. 3.5. The resulting closed loop (i.e. discrete controller + nonlinear plant model) response to a throttle profile is shown in Fig. 3.4.

3.3 Distribution Problem

The block diagram representing the plant and controller relationship for the discrete-time MIMO controller in the previous section is given in Fig. 3.5. With the controller distribution problem our aim is to find component controllers, $\mathbf{C}_{ecu}$, $\mathbf{C}_{vct}$, $\mathbf{C}_{ego}$ which improve the component swapping modularity of the system by using the bi-directional network communications. The block diagram of the proposed distributed system with the proposed communication is given in Fig. 3.6.

A generic optimization problem formulation for maximizing the component swapping modularity of an actuator component was given in [3]. Given nominal settings for the plant parameters (denoted as $\mathbf{p}_{CS}^0$, $\mathbf{p}_{VCT}^0$, $\mathbf{p}_{EGO}^0$ for the controlled system i.e., rest of the engine, VCT component and EGO sensor, respectively), we can formulate the distribution problem which maximizes VCT component swapping modularity, M_{VCT} while the distribution constraint, desired overall controller must be equal to the overall effect of the distributed controller (i.e $C_{des} = C_{dis}$) holds.

In short, M_{VCT} represents the size of the region in the parameter space, which includes the nominal VCT parameters, for which we can ensure the distribution constraint, by changing only gains of the VCT controller.

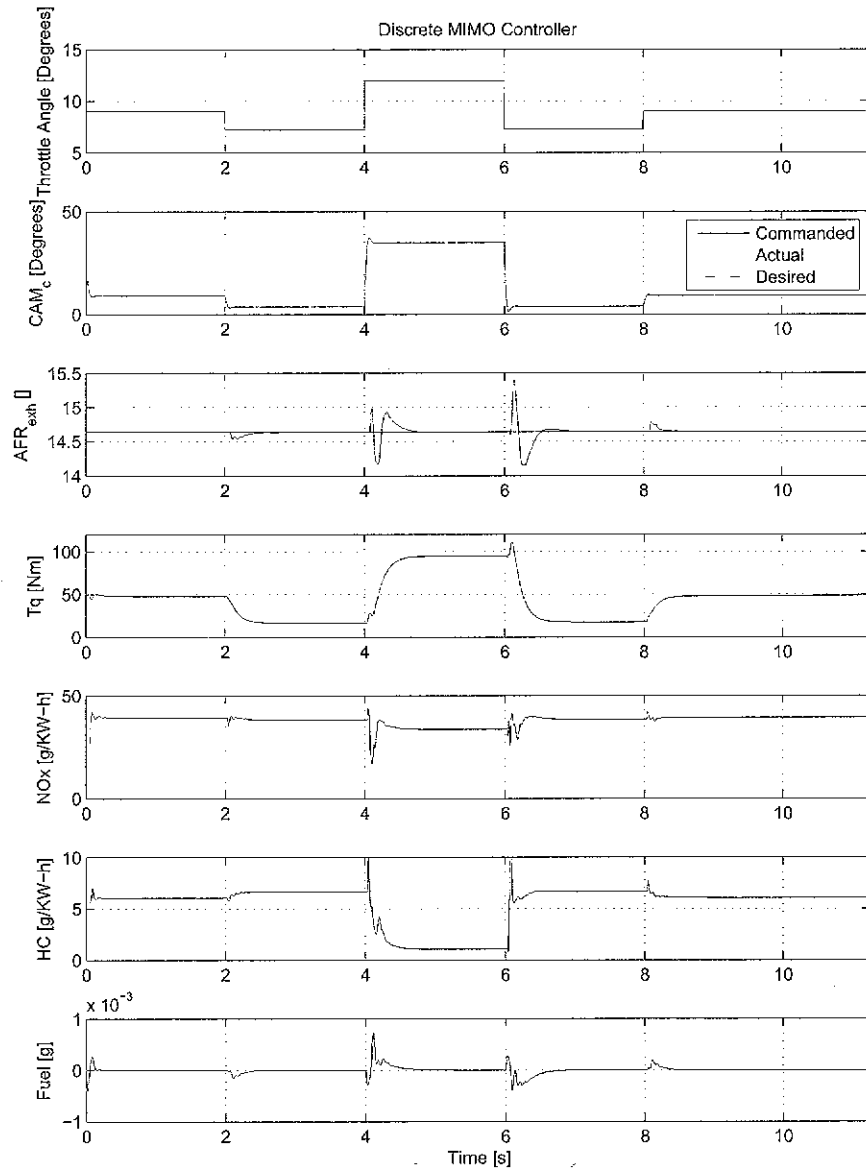


Figure 3.4: VCT Engine with Discrete MIMO Controller closed loop response

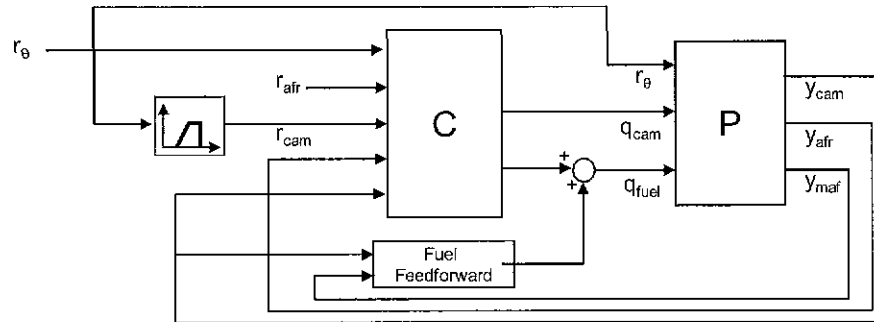


Figure 3.5: VCT Engine with Discrete MIMO Controller

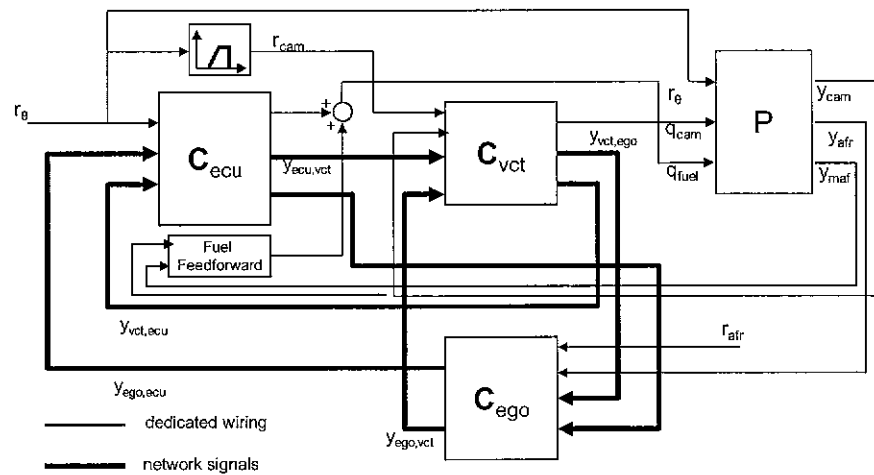


Figure 3.6: VCT Engine with Distributed Discrete MIMO Controller

3.3.1 Formulation of C_{des}

In this section, the formulation for the desired centralized controller, C_{des} , given plant parameters $\mathbf{p}_{CS}^0$, $\mathbf{p}_{VCT}^0$, $\mathbf{p}_{EGO}^0$ is presented. Given the optimal state feedback matrix $\mathbf{K}_d = [\mathbf{K1}_d, \mathbf{K2}_d]$ and linear observer gain $\mathbf{L}_d$, the state space representation of the discrete MIMO controller in Fig. 3.5 can be given as

$$\begin{aligned} \hat{\mathbf{x}}_c(k+1) = & \mathbf{Ac}_d \begin{bmatrix} \mathbf{x}_c \\ \mathbf{x}_{c,I} \end{bmatrix} (k) + \mathbf{Bc}_d \mathbf{u}(k) \\ & + \mathbf{Bc}_{d,r} \begin{bmatrix} r_\theta \\ r_{cam} \\ r_{afv} \end{bmatrix} (k) \end{aligned} \quad (3.13)$$

$$\mathbf{q}(k) = \mathbf{C}_d \begin{bmatrix} \mathbf{x}_c \\ \mathbf{x}_{c,I} \end{bmatrix} (k) \quad (3.14)$$

where

$$\mathbf{Ac}_d = \begin{bmatrix} \mathbf{A}_d - \mathbf{L}_d \mathbf{C}_d - \mathbf{B}_d \mathbf{K1}_d + \mathbf{L}_d \mathbf{D}_d \mathbf{K1}_d & \mathbf{L}_d \mathbf{D}_d \mathbf{K2}_d - \mathbf{B}_d \mathbf{K2}_d \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \quad (3.15)$$

$$\mathbf{Bc}_d = \begin{bmatrix} \mathbf{L}_d \\ -\mathbf{T}_s \mathbf{I} \end{bmatrix} \quad (3.16)$$

$$\mathbf{Bc}_{d,r} = \begin{bmatrix} \mathbf{B}_d - \mathbf{L}_d \mathbf{D}_d \mathbf{r}_d & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_s \mathbf{I} \end{bmatrix} \quad (3.17)$$

$$\mathbf{Cc}_d = \begin{bmatrix} -\mathbf{K1}_d & -\mathbf{K2}_d \end{bmatrix} \quad (3.18)$$

We compute the z-transform equivalent transfer function matrix for the discrete state space system given in Equations 3.13-3.14 in the form

$$\mathbf{C} = \begin{bmatrix} N_{11}(z)/D_{11}(z) & \cdots & N_{15}(z)/D_{15}(z) \\ N_{21}(z)/D_{21}(z) & \cdots & N_{25}(z)/D_{25}(z) \end{bmatrix} \quad (3.19)$$

N_{11} and D_{11} are polynomials with vectors $\mathbf{x}_{n11}$ and $\mathbf{x}_{d11}$ of controller gains (Eqn. (3.20)) respectively,

$$\begin{aligned} C_{11}(\mathbf{x}_{n11}, \mathbf{x}_{d11}, z) &= \frac{N_{11}(\mathbf{x}_{n11}, z)}{D_{11}(\mathbf{x}_{d11}, z)} = \\ &= \frac{x_{n11,m} z^{m+1} + \cdots + x_{n11,n+2} z + x_{n11,n+1}}{x_{d11,n} z^{n+1} + \cdots + x_{d11,2} z + x_{d11,1}} \end{aligned} \quad (3.20)$$

Then we can define $C_{des}(\mathbf{p}_{VCT}, \mathbf{p}_{EGO}, \mathbf{p}_{ECU})$ as

$$C_{des}(\mathbf{p}_{VCT}, \mathbf{p}_{EGO}, \mathbf{p}_{CS}) = C(\{\mathbf{x}_{nij}^*, \mathbf{x}_{dij}^* | i = 1, 2; j = 1, 2, \dots, 5\}) \quad (3.21)$$

where $\{\mathbf{x}_{nij}^*, \mathbf{x}_{dij}^* | i = 1, 2; j = 1, 2, \dots, 5\} = \argmin J_C(P, \mathbf{C}) = \Sigma_1^\infty (x(k)'Qx(k) + u(k)'Ru(k))$ s.t. $\mathbf{x}(k+1) = \mathbf{A}_d\mathbf{x}(k) + \mathbf{B}_d\mathbf{u}(k)$. The $\mathbf{p}_{CS}$, $\mathbf{p}_{VCT}$, $\mathbf{p}_{EGO}$ are parameter vectors representing the controlled system (rest of the engine), VCT component and EGO sensor parameters respectively.

These polynomial constants vary as VCT and EGO plant parameters change, as shown in Fig. 3.7 for $N_{II}(z)$, and can be implemented in the numerical solution phase of the problem as lookup tables or regression equations.

3.3.2 Formulation of C_{dist}

For the control system given in Fig. 3.6 and real-time distributed controller schedule given in Fig. 3.8, discrete MIMO equations to calculate the overall controller equation can be written to obtain the equation given in (3.22).

$$\begin{aligned} \mathbf{C}_{dist}(z) = & - \begin{bmatrix} C_{vct13} & 0 & 0 & 0 & 0 & C_{vct14} \\ 0 & 0 & 0 & C_{ecu13}z^{-1} & C_{ecu12} & 0 \end{bmatrix} \\ & \begin{bmatrix} -1 & 0 & 0 & C_{ecu23}z^{-1} & C_{ecu22} & 0 \\ 0 & -1 & 0 & C_{ecu33}z^{-1} & C_{ecu32} & 0 \\ C_{vct23} & 0 & -1 & 0 & 0 & C_{vct24} \\ C_{vct33} & 0 & 0 & -1 & 0 & C_{vct34} \\ 0 & C_{ego14}z^{-1} & C_{ego13}z^{-1} & 0 & -1 & 0 \\ 0 & C_{ego24}z^{-1} & C_{ego23}z^{-1} & 0 & 0 & -1 \end{bmatrix}^{-1} \\ & \begin{bmatrix} C_{ecu21} & 0 & 0 & 0 & 0 \\ C_{ecu31} & 0 & 0 & 0 & 0 \\ 0 & 0 & C_{vct21} & 0 & C_{vct22} \\ 0 & 0 & C_{vct31} & 0 & C_{vct32} \\ 0 & C_{ego11} & 0 & C_{ego13} & 0 \\ 0 & C_{ego21} & 0 & C_{ego23} & 0 \end{bmatrix} \\ & + \begin{bmatrix} 0 & 0 & C_{vct11} & 0 & C_{vct12} \\ C_{ecu11} & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \quad (3.22)$$

It is also important to note that Fig. 3.8 assumes perfect completion of the communication scheme and the effect of communication delays and bandwidth limitations

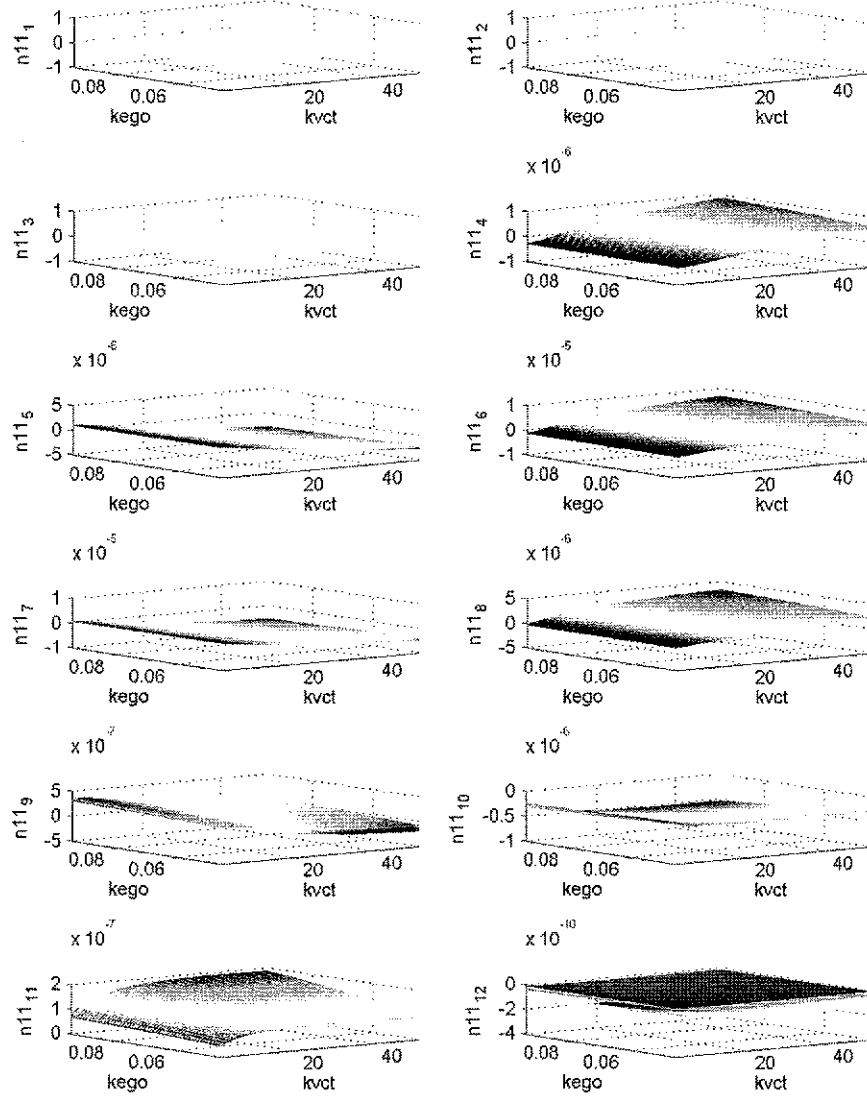


Figure 3.7: Polynomial Coefficients for $N_{11}(z)$ as τ_{vct} and τ_{ego} are varied.

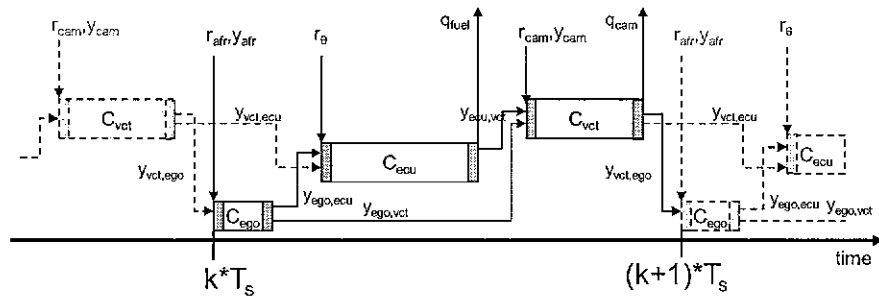


Figure 3.8: Real-time schedule for distributed discrete MIMO controller.

are not considered while finding the equivalent controller given in Eqn. (3.22). However, these effects could be investigated in future work by including delays on various communication paths when such information is available.

In order to obtain the *distribution constraint* needed to solve the distribution problem numerically, we need to make assumptions regarding the order of polynomial transfer function matrices $\mathbf{C}_{ecu}$, $\mathbf{C}_{vct}$, $\mathbf{C}_{ego}$. Although the most straightforward approach would be to assume all, $\mathbf{C}_{ecu}$, $\mathbf{C}_{vct}$, $\mathbf{C}_{ego}$ are composed of transfer functions of the same order of the centralized controller $\mathbf{C}_{des}$, this may not be preferable, since it increases the numerical burden of the problem substantially. For the application we are considering here our desired controller transfer matrix consists of polynomials of 11th order which means our nonlinear optimization problem would have almost 600 design variables and constraints to handle.

A better way to tackle the numerical burden of the distribution problem is to perform some pre-optimization analysis to better understand and simplify the problem. Fig. 3.9 shows the pole zero map of the desired optimal controller, $\mathbf{C}_{des}$, obtained in the previous section as the VCT Component parameters $\tau_{vct,act}$ and $\tau_{vct,sen}$ vary.

Analyzing these plots carefully we note that swapping the VCT component with another one would only effect the elements C_{11} , C_{12} , C_{13} , C_{14} , C_{15} , C_{23} , C_{25} (due to varying poles, zeros or both in Fig. 3.9) of our desired optimal controller, $\mathbf{C}_{des}$. Even in these elements one can observe some stationary poles and zeros as well as varying poles and zeros which can be exploited for the purposes of the improving modularity the system. The static pole and zero elements can be placed in the ECU and EGO sensor controller while varying poles can be placed in the VCT Controller.

In order to simplify the numerical optimization problem and obtain a good candidate distribution solution to start the search algorithm we used a two step configu-

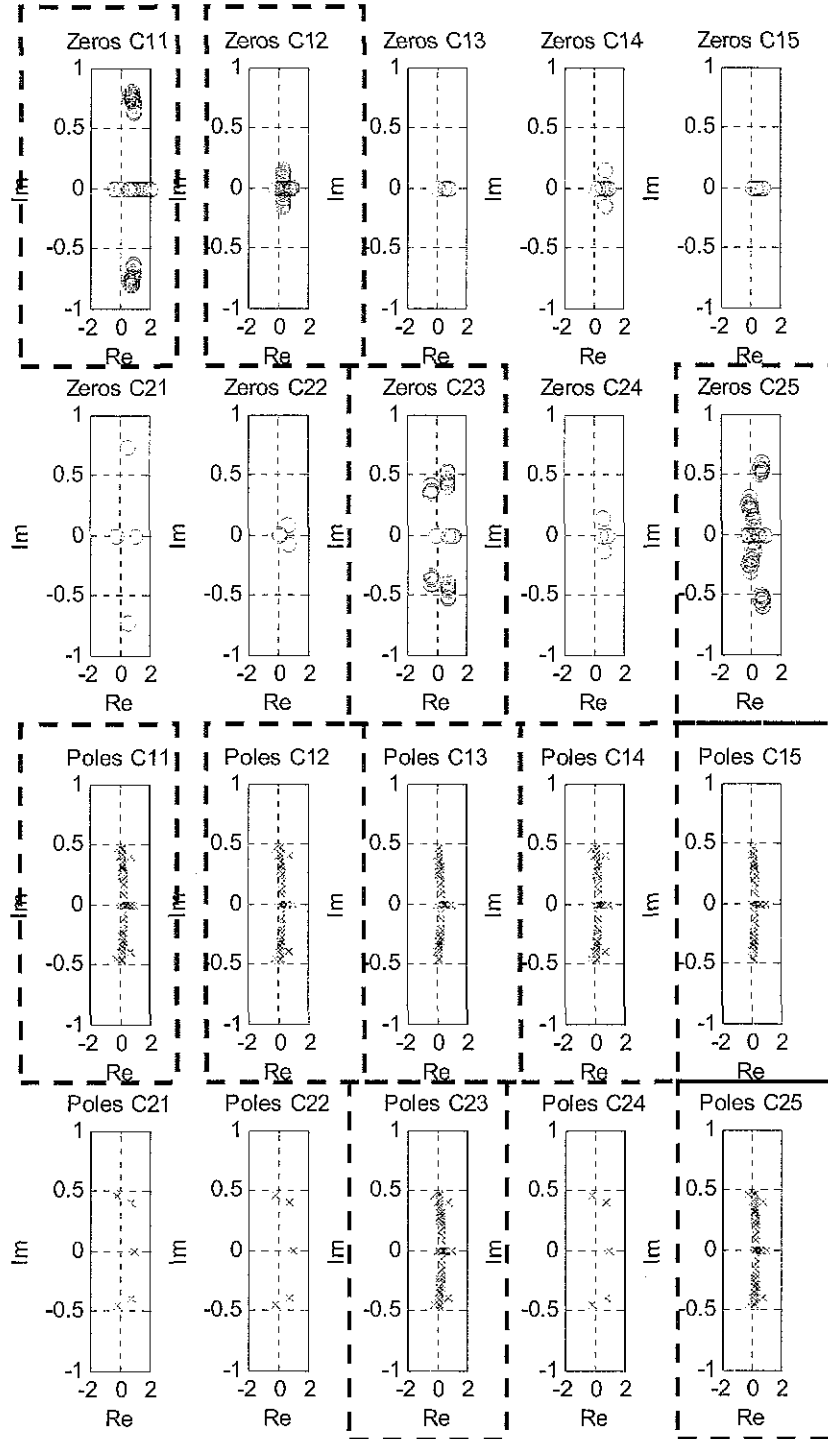


Figure 3.9: Pole Zero Map of the $C_{des}(z)$ as VCT Component (i.e. $\tau_{vct,act}$ and $\tau_{vct,sen}$) changes. Pole or zero plots with the dashed boxes are assumed to vary as VCT component changes.

ration design process once the Pole-Zero mapping of $C_{des}(\mathbf{p}_{cs}, \mathbf{p}_{vct}, \mathbf{p}_{ego}, z)$ elements versus the VCT component plant variables, $\mathbf{p}_{vct}$ is obtained:

1. Identify proper transfer functions $C_{des,ij}^s(z)$ and $C_{des,ij}^v(z)$ such that $C_{des,ij}(z) = C_{des,ij}^s(z)C_{des,ij}^v(z)$,
2. Find $\mathbf{C}_{ecu}(z)$, $\mathbf{C}_{vct}(z)$ (with k th row l th column element denoted as $C_{vct,kl}(z)$), $\mathbf{C}_{ego}(z)$ satisfying (a) and (b):
 - (a) $\mathbf{C}_{dist}(\mathbf{C}_{ecu}, \mathbf{C}_{vct}, \mathbf{C}_{ego}, z)$ is equal to $\mathbf{C}_{des}(\mathbf{p}_{cs}, \mathbf{p}_{vct}, \mathbf{p}_{ego}, z)$
 - (b) For all $C_{des,ij}^v(z)$ (identified in Step 2) there exists $C_{vct,kl}(z)$ such that $C_{dist,ij} = C_{des,ij}^s(z)C_{vct,kl}(z)$. Therefore $C_{vct,kl}(z)$ can be picked to be $C_{des,ij}^v(z)$ for different VCT components.

The superscript "s" denotes the stationary pole/zero pairs that do not change as $\tau_{vct,act}$ and/or $\tau_{vct,sen}$ change while the superscript "v" denotes those that do vary. It is important to note that the primary goal of this procedure is to place varying poles and zeros to the target component controller while satisfying the distribution constraint in Step 2(a). Therefore, it is still acceptable if there are cases where some or all of the stationary poles/zeros reside (e.g., EGO sensor controller as shown in Table 3.2) to satisfy the distribution constraint in Step 2(a).

The pre-optimization method described above is generic. However, the results (i.e., number of pole zero cancellations, identifying stationary transfer function matrix elements and/or poles and zeros in these elements) would change from one application to another.

Based on the procedure given above, we obtained a candidate solution that maximizes VCT component modularity as shown in Fig. 3.10. Implementation of this

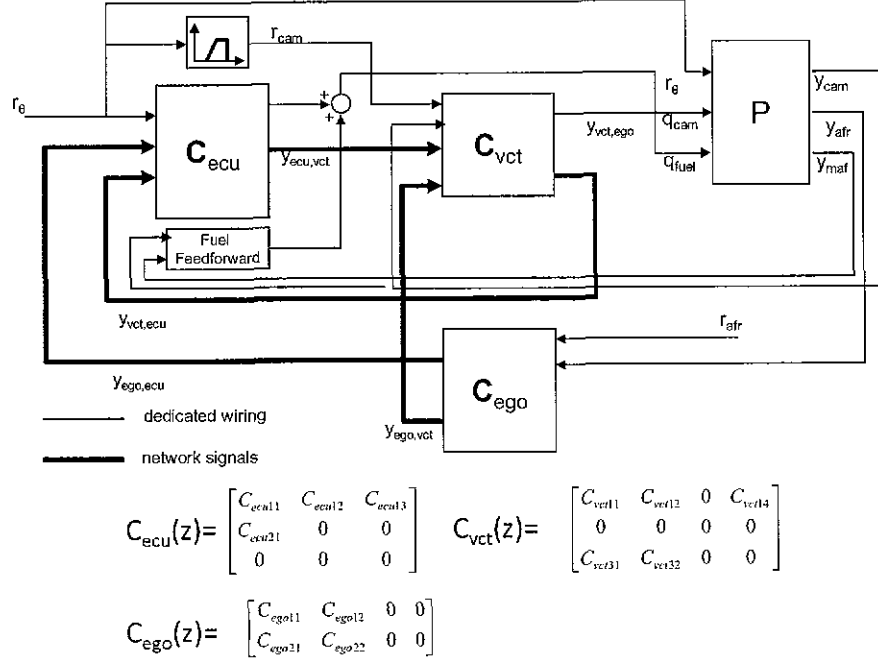


Figure 3.10: Candidate Distribution to Maximize VCT Modularity

distribution would result in an overall controller given, in (3.23).

$$C_{dist} = \begin{bmatrix} C_{vct13}C_{ecu21} & C_{vct14}C_{ego21} & C_{vct11} & C_{vct14}C_{ego22} & C_{vct12} \\ C_{ecu11} & C_{ecu12}C_{ego11} & C_{ecu13}C_{vct31}/z & C_{ecu12}C_{ego12} & C_{ecu13}C_{vct32}/z \end{bmatrix} \quad (3.23)$$

The closed-loop response of this solution is virtually indistinguishable from the overall controller response presented in Fig. 3.4. The solution for the optimization problem to maximize VCT component modularity (i.e parameters for C_{ecu} , C_{vct} , C_{ego} transfer function matrices) is the same as the candidate solution as presented in Table 3.1. The numerical value of M_{VCT}^* is $60 * 10 = 600(ms)^2$. A better explanation of this number can be given as follows: Assuming the default configuration for other components, optimal controllers can be obtained by only adjusting VCT component controller for VCT plant parameters changing in the ranges $\tau_{vct,act} = [7, 67]ms$ and $\tau_{vct,sen} = [10, 20]ms$.

The same steps can also be followed to obtain a candidate solution to maximize EGO sensor modularity: After a similar pole/zero analysis to the one shown in Fig.

Element	Solution	Element	Solution
C_{ecu11}	C_{21}	C_{ecu12}	1
C_{ecu13}	C_{23}^s/z	C_{ecu21}	C_{11}^s
C_{vct11}	C_{13}	C_{vct12}	C_{15}
C_{vct13}	C_{11}^v	C_{vct14}	C_{14}^v
C_{vct31}	C_{23}^v	C_{vct32}	C_{25}^v
C_{ego11}	C_{22}	C_{ego12}	C_{24}
C_{ego21}	C_{12}^s	C_{ego22}	C_{14}^s

Table 3.1: Transfer Functions for the Modular VCT Controller Solution

Element	Solution	Element	Solution
C_{ecu11}	C_{21}	C_{ecu12}	1
C_{ecu13}	1	C_{ecu31}	C_{11}^s
C_{vct11}	C_{13}	C_{vct12}	C_{15}
C_{vct14}	1	C_{vct31}	$C_{23}z$
C_{vct32}	$C_{25}z$	C_{ego11}	C_{22}
C_{ego13}	C_{24}	C_{ego21}	$C_{12}^s C_{12}^v$
C_{ego22}	C_{14}	C_{ego24}	$C_{11}^d z$

Table 3.2: Transfer Functions for the Modular EGO Controller Solution

3.9, we have identified only C_{11} and C_{12} elements have varying elements as the EGO sensor changes. By applying the pre-optimization procedure the candidate solution given in Fig. 3.11 is obtained. As in the VCT case, closed loop response of this solution is virtually indistinguishable from the overall controller response presented in Fig. 3.4. Since our static pole/zero numerical threshold is within the distribution constraint limits in the optimization problem our candidate solution (Fig. 3.11 and Table 3.2) is verified as the optimal distribution which results in $M_{EGO}^* = 60ms$ (i.e. given the default configuration, optimal controllers can be obtained by only changing EGO controller in the range $\tau_{EGO} = [40, 100]ms$).

3.4 Conclusion

In this paper, a method to distribute the discrete MIMO controller for the VCT engine to improve the component swapping modularity of the VCT actuator and the EGO sensor components is presented. This work is an extension of the method described in [2, 3] to a discrete MIMO controller applied to a more complex system.

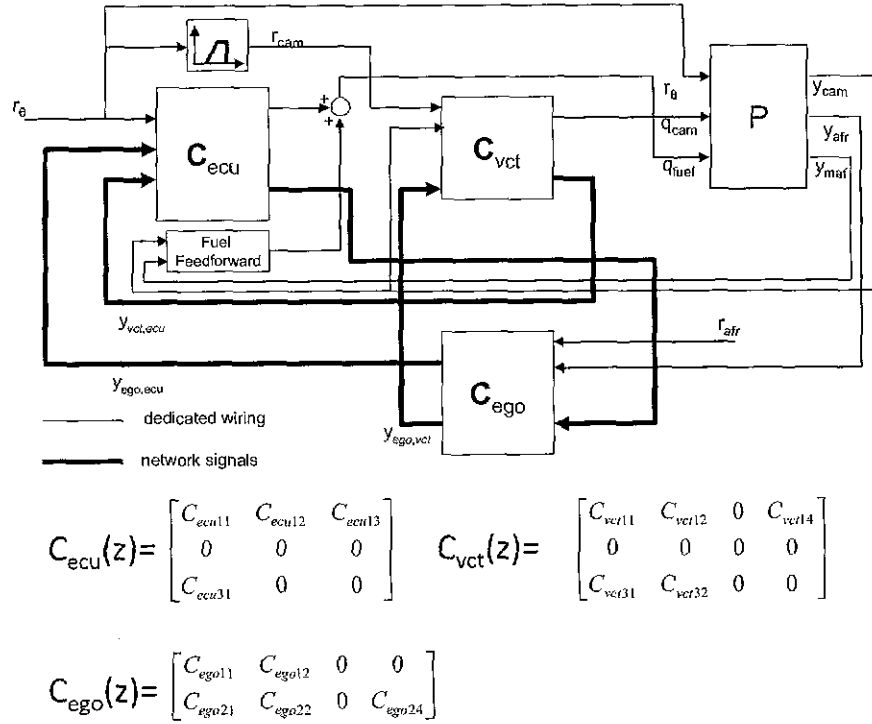


Figure 3.11: Candidate Distribution to Maximize EGO Modularity

In Section 3.2.3, a discrete LQG controller was designed based on the work from [10]. In Section 3.3.2, this controller was distributed to ECU, a VCT controller and an EGO sensor controller in order to improve the component swapping modularity of the system. In Section 3.3.2, we also present a control oriented *pre-optimization* technique which simplifies the optimization problem, which results in shorter computation times to obtain a solution. The pre-optimization method identifies and groups together controller poles/zeros which change, or remain unchanged as the VCT or EGO parameters vary. A candidate solution was devised and used in optimization problem as the initial solution. Resulting solutions maximizing VCT Component modularity and EGO sensor modularity are given in Figs. 3.10 and 3.11 respectively. The performances of these distributed controllers are virtually identical to the performance of the original controller as shown in Fig. 3.4. The range of the

obtained solutions depends on stability and properness of the overall controller and distributed controllers as well as the optimization toleration settings. These distributed controllers provide component-swapping modularity which does not exist in the case of implementing the whole algorithm using single centralized controller (i.e. using ECU only). The MIMO controller was designed using a linearized model around a specific operation point. It is expected that the practical implementation of this controller would be done by designing the MIMO controller for various operating points of the VCT engine and use gain scheduling to calibrate controller for different operating points. The same schedule oriented approach can be applied to the distributed controller by solving the distribution problem for all of the operating points considered. The minimum component swapping modularity obtained among operating points would then be used for optimization problem purposes.

Future work on this topic will include consideration of VCT and EGO modularity concurrently.

Bibliography

- [1] K. J. Astrom. Present developments in control applications. Keynote Paper at the IFAC 50th Anniversary Meeting, Heidelberg, 2006.
- [2] M. Cakmakci and A. G. Ulsoy. Networked control systems with bi-directional communication among "smart" components. *Proceedings of the American Control Conference*, pages 627–632, June 2005.
- [3] M. Cakmakci and A. G. Ulsoy. Improving component swapping modularity using bi-directional communication in networked control systems. *Proceedings of the International Symposium on Flexible Automation*, July 2006.
- [4] A. C. Elrod and M. T. Nelson. Development of variable valve timing engine to eliminate the pumping losses associated with throttled operation. *SAE Paper No. 860537*, 1986.
- [5] C. Gray. A review of variable engine valve timing. *SAE transactions*, 97:631–641, 1989.
- [6] H.P. Lenz and K. Wichart. Variable valve timing-a possibility to control engine load without throttle. 1988.
- [7] T.H. Ma. Effect of variable engine valve timing on fuel economy. *Society of Automotive Engineers international congress and exposition*, 29, 1988.
- [8] G.B.K. Meacham. Variable Cam Timing as An Emission Control Tool. 1970.
- [9] B.K. Powell and J.A. Cook. Nonlinear low frequency phenomenological engine modeling and analysis. *Proceedings of the American Control Conference*, 24, 1982.
- [10] A. G. Stefanopoulou. *Modeling and Control of Advanced Technology Engines*. PhD thesis, University of Michigan, 1996.

- [11] A.G. Stefanopoulou, J.A. Cook, J.W. Grizzle, and J.S. Freudenberg. Control-oriented model of a dual equal variable cam timing spark ignition engine. *ASME Journal of Dynamic Systems, Measurement and Control*, 120:257–266, 1998.
- [12] A.G. Stefanopoulou, J.S. Freudenberg, and J.W. Grizzle. Variable camshaft timing engine control. *IEEE Transactions on Control Systems Technology*, 8(1):23–34, 2000.
- [13] R.A. Stein, K.M. Galletti, and T.G. Leone. Dual Equal Vct-A Variable Camshaft Timing Strategy for Improved Fuel Economy and Emissions. 1995.
- [14] R. Steinberg, I. Lenz, G. Koehnlein, M.E. Scheidt, T. Saupe, and W. Buchinger. A Fully Continuous Variable Cam Timing Concept for Intake and Exhaust Phasing. 1998.
- [15] K. Ulrich and K. Tung. Fundamentals of product modularity. In A. Sharon, editor, *Issues in Mechanical Design International*, pages 73–79. ASME, New York, 1991.

CHAPTER IV

Combined Component Swapping Modularity for a VCT Engine Controller

Abstract: The use of bi-directional communications provide additional design freedom which can be used to maximize the swapping modularity of networked smart components. In this paper, application of a design method for combined swapping modularity of two or more system components is discussed. Development of measures for combined swapping modularity is important to be able to analyze more realistic engineering cases. The combined modularity problem is a more difficult problem compared to individual component swapping modularity problem. First, two approaches (simultaneous and sequential) for combining component swapping modularity of two or more components are presented. Then these combined modularity approaches are used to design controllers which maximize the component-swapping modularity of the Variable Camshaft Timing (VCT) component (i.e. actuator and sensor) and the Exhaust Gas Oxygen (EGO) sensor for an internal combustion engine.

4.1 Introduction

Availability of on-board electronics has increased the use of “smart” components in automatic control systems. Smart components with networking capabilities open up the possibility of using bi-directional communications among components

as shown in Fig. 4.1. The use of bi-directional communications provide additional design freedom which can be used to design control systems with better overall characteristics.

In [2], a method to design control systems where component swapping modularity is maximized using bi-directional communications was presented. Component-swapping modularity occurs when two or more alternative basic components can be paired with the same modular components creating different product variants belonging to the same product family [7]. Control systems with modularly swappable components can be defined as the systems in which the initial and final configurations due to a component change operate at their corresponding optimal performance.

As the first step of the method an overall controller, C , is designed using conventional design methods, then this controller is distributed to smart component controllers. The resulting distributed controller structure and optimal parameters maximize the component-swapping modularity while providing the same performance as the desired controller, C . The proposed method is then successfully applied to maximize the actuator swapping modularity of a simple continuous SISO problem [2].

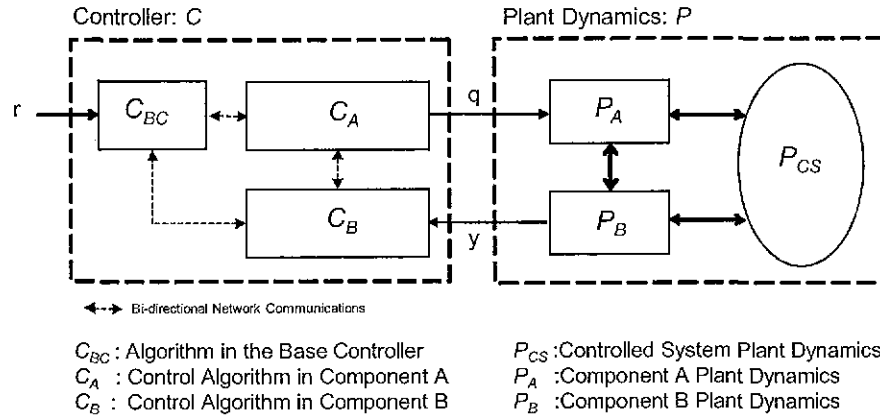


Figure 4.1: Bi-directional Communications and Overall Control and Plant Composition.

In [1], the design method presented in [2] is applied to a more complex problem of design and distribution of discrete MIMO control of a Variable Camshaft Timing (VCT) Engine. The engine plant model is developed based on the work presented in [5] and [4], and an overall discrete-time MIMO controller was designed based on [6]. After formulating the distribution problem and a pre-optimization analysis to simplify the numerical solution of the problem, two distinct optimal distribution solutions optimizing the component swapping modularity of the VCT component and Exhaust Gas Oxygen (EGO) sensor is given respectively.

In [2], four possible use cases for systems designed to have component-swapping modularity are outlined:

1. Sustainable maintenance and/or upgrade of a particular end product:

Increasing component swapping modularity shortens the engineering time and effort (i.e. cost) in the iterative phases of conceptual design, implementation and testing/validation after each maintenance and upgrade of the system.

Many manufacturing facilities house custom made machining systems that only produce a certain type of a product with high accuracy and precision. Building these systems with component swapping modularity in mind would pay off over time through their life cycle since, maintenance and upgrade of a swappable component (such as a “smart” electric motor for an axis manipulator) will not only be easier to perform, but also will result in the best performance possible from the overall system depending on the new component specifications.

2. Deploying controlled system platform based algorithms:

Use of platform engineering has been on the rise in recent years for companies which produce a variety of products. Product platforms require defining a common infrastructure with different component variants within a company’s

product line. Quality of control engineering can be increased drastically by focusing on designing control algorithms for product platforms (more engineering time, focus and experience) which will increase the overall performance of the end-product.

Today, many automotive companies develop their vehicles based on vehicle platforms (such as small, midsize, truck, etc.) but offer variants which appeal to different customer bases. For example many companies offer economy and luxury vehicles based on their small car platform. These two options would present two different cost structures due to the difference in the planned sale price. In parallel, this price difference also forces companies to use better performing alternatives of some components in the luxury option due to better performance expectancy from the vehicle sold in the luxury car segment. Designing these components to be swappable will decrease the engineering and development costs drastically with no impact of the best expected performance from both the economy and luxury version of the same platform design.

3. Deploying control algorithms for different builds of the same product:

For companies which use many different suppliers and operate in many different locations, different builds of the same product are needed because of subsystem variance and difference in system specifications due to regulations, local requirements, etc. For these type of global products, having component swapping modularity in control systems increases the overall efficiency of engineering by obtaining location specific optimal algorithms without redesigning and re-calibrating the whole system.

Ever increasing competition in the automotive industry forces many companies to launch so called “global” vehicles, i.e., very appealing vehicle designs

built in different parts of the world using the local automotive supply chain and conforming to local regulations. Global vehicles designed with swappable components minimize the re-design efforts to launch a vehicle locally and the need to maintain multiple engineering teams at different parts of the world to solve the same design problem with different constraints.

4. Reducing costs by developing highly customizable but less variant components:

Supplier companies which supply sub-systems to more than one company can develop control systems with component swapping modularity to focus on systems which optimally work on many customer end-products.

In the automotive industry, many components such as EGO Sensors are primarily provided by suppliers. Getting specifications for the next “smart” EGO sensor right (i.e. right amount of computing power with the right networking bus) would improve the competitiveness of the supplier considerably since these components are produced in bulk and sold to multiple auto manufacturers (OEMs).

The cases outlined above imply either design or deployment of systems belonging to the same or similar product families (Cases 2,3) or upgrade and/or maintenance of a single system (Case 1). In the studies discussed earlier (i.e [2] and [1]), component swapping modularity of only one component at a time is considered. Single component focused component swapping modularity studies can be useful for a scenario similar to Case 4. However, other realistic engineering scenarios require consideration of multiple components while obtaining an optimal distribution and correspond to cases 1-3 above.

In this paper, generalization of the design method presented in [2, 1] to the combined swapping modularity of system component will be presented. In the next

section, two approaches (i.e. simultaneous and sequential) for combining component swapping modularity of two or more components are presented. Next, these combined modularity approaches are used to design controllers which maximize the component-swapping modularity of the VCT Component (i.e. actuator and sensor) and the EGO Sensor for the example given in [1]. The designed distributed controllers utilize the bi-directional communications introduced earlier and exhibit improved combined swapping modularity than the traditional centralized version of the controller. This paper concludes with a summary of the results, conclusions and future work.

4.2 Combined Component-Swapping Modularity of a System

The mathematical formulation used to quantify the component swapping modularity for a single component was given in [2]. In summary, as illustrated in Fig. 4.2 for a component X whose dynamic equations can be represented in terms of two parameters (i.e. $\mathbf{p}_X = \{p_{X,1}, p_{X,2}\}$), Φ_X is a connected set of component plant parameters including the default parameter value, $\mathbf{p}_X^0$, that can be achieved by changing only the control parameters for component X within their limits given a distribution solution for the controller.

We then define the function M_X , the swapping modularity for component X as

$$M_X(\Phi_X, \mathbf{p}_X) = \int_{\Phi_X} d\mathbf{p}_X \quad (4.1)$$

4.2.1 Simultaneous Approach

One way of combining component swapping modularity is to add the component swapping modularity of components (i.e. M_X given in (4.1)) with appropriate weigh-

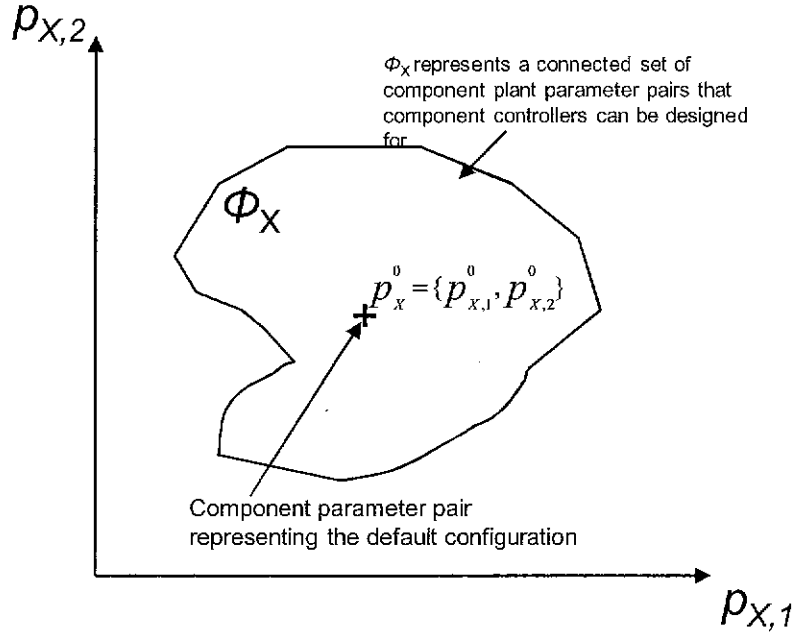


Figure 4.2: Illustration of Set Φ_X for a Two Parameter Component X , $\mathbf{p}_X = \{p_{X,1}, p_{X,2}\}$.

ing factors (i.e. ρ_X) as shown in (4.2) for a two component system with components A and B:

$$M_{sys}(\Phi_A, \Phi_B, \mathbf{p}_A, \mathbf{p}_B) = \rho_A M_A(\Phi_A, \mathbf{p}_A) + \rho_B M_B(\Phi_B, \mathbf{p}_B) \quad (4.2)$$

This way of simultaneously calculating the component swapping modularity of components for a particular distribution configuration is useful for cases when modularity of a baseline controller design (with the distributed structure) is evaluated. For example, for the cases described in 2 and 3 in the previous section, the baseline design can be used for different product platforms (i.e for example different controlled systems) using the same smart components in the product line (Case 2) or for different instrument configurations (i.e. for example smart components with different performance) due to local availability or requirements in a global deployment situation (Case 3).

4.2.2 Sequential Approach

When combined modularity of a specific system design rather than a baseline design is considered, adopting sequential a approach while calculating the combined modularity of the system could be more useful. For example, combined swapping modularity calculated by using the previously discussed simultaneous approach offers little information about the modularity of the system for sequential change of components over a time period since once the first component is changed, the original the configuration of the system is lost. This corresponds to Case 1 in the previous section when components of a particular end product are renewed due to failure and/or scheduled maintenance or upgrades.

In these problems a measure for combined modularity of components formulated based on their sequential update can be calculated. For example, if the order of the update is given as *update component A first, component B second*, then sequential combined modularity can be calculated for a two component system as shown in (4.3):

$$M_{sys}(\Phi_A, \Phi_B, \mathbf{p}_A, \mathbf{p}_B) = \int_{\phi_A} \int_{\phi_B(\rho_A)} dp_B dp_A \quad (4.3)$$

4.2.3 Optimization Problem to Maximize Combined Component Swapping Modularity

In Fig. 4.1, the block diagram representation for a system consists of a base controller, with transfer function, $\mathbf{C}_{BC}$, a controlled system, with plant dynamics $\mathbf{P}_{CS}$, equipped with two smart components, A and B is illustrated. We define $\mathbf{p}_X$, as a parameter vector representing the component X plant dynamics transfer function, P_X . We also denote nominal settings for these plant parameters as $\mathbf{p}_X^0$. We can formulate the distribution problem which maximizes swapping modularity of com-

ponent A, M_A , to determine the controller transfer function matrices, $\mathbf{C}_{BC}, \mathbf{C}_A, \mathbf{C}_B$ as:

$$\max_{\mathbf{C}_{BC}, \mathbf{C}_A, \mathbf{C}_B} M_A(\Phi_A, \mathbf{p}_A) \quad (4.4)$$

subject to

Distribution Constraint:

$$C_{dist}(\mathbf{C}_{BC}, \mathbf{C}_A, \mathbf{C}_B) = C_{des}(\mathbf{p}_{CS}^0, \mathbf{p}_A^0, \mathbf{p}_B^0) \quad (4.5)$$

Additional Constraints:

$$\mathbf{g}(\mathbf{p}_{CS}, \mathbf{p}_A, \mathbf{p}_B, \mathbf{C}_{BC}, \mathbf{C}_A, \mathbf{C}_B) \leq \mathbf{0} \quad (4.6)$$

where C_{des} is the desired centralized controller, determined by any traditional control design method given nominal plant parameters $\mathbf{p}_{CS}^0, \mathbf{p}_A^0, \mathbf{p}_B^0$. Then, $\mathbf{C}_{dist}$ is the effective centralized controller calculated from component controllers, $\mathbf{C}_{BC}, \mathbf{C}_A, \mathbf{C}_B$, and $\mathbf{g}(\mathbf{P}, \mathbf{C})$ in (4.6) refer to the additional problem specific constraints (e.g., limits on parameters or controller gains).

In order to formulate the combined swapping modularity problem we will use combined modularity, M_{sys} , given in (4.2) or (4.3) instead of individual component swapping modularity, M_A , given in (4.4).

4.3 Example: Combined Component Swapping Modularity of the VCT Engine

Variable Camshaft Timing (VCT) schemes increase internal residual gas by affecting the intake, combustion and exhaust phases of the engine cycle. Increase in internal residual gas reduces the combustion temperature which decreases nitrogen oxide, NO_x , formation. The internally recirculated exhaust gas is rich in unburned

hydrocarbons, HC , which can be burned in the next cycle. Application of VCT schemes, since they require higher manifold pressure, decrease pumping losses which results in improved fuel economy. However, dilution of the in-cylinder mixture adversely affects the engine torque response. These factors define the trade-off between good emissions and good drivability for VCT engines. Development of a continuous, non-linear, low-frequency, phenomenological and control oriented VCT engine model was given in [4] based on the model structure given in [3] and others.

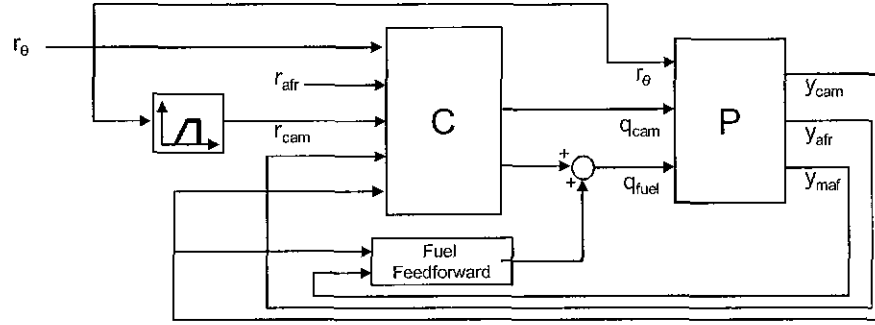


Figure 4.3: VCT Engine with Discrete MIMO Controller

In [1], the design and distribution of a discrete MIMO controller for a variable camshaft timing (VCT) engine is presented. The resulting distributed controllers maximize the component swapping modularity of the *smart* VCT component and *smart* EGO Sensor (i.e., these components have an on-board computing and network connectivity). The block diagram representing the plant and controller relationship for the discrete MIMO controller is given in Fig. 4.3. As described in [1], centralized controller, C , is a transfer function matrix obtained by solving the discrete LQG control design problem. Also, we model the VCT component plant dynamics with parameters $\tau_{vct,act}$ and $\tau_{vct,sen}$ (i.e. first order transfer function time constant parameter and a first order Pade approximation parameter respectively). For the EGO sensor plant dynamics, first order dynamics with time constant τ_{ego} is used.

The objective in solving the controller distribution problem is to find component controllers, C_{ecu} , C_{vct} , C_{ego} which improve the component swapping modularity of the system by using bi-directional network communications. Thus, in terms of our earlier notation in Section 4.2, $C_{BC} = C_{ecu}$ is the base controller located in the Engine Control Unit (ECU), $C_A = C_{vct}$ is the controller for component A located in the smart VCT component, and $C_B = C_{ego}$ is the controller for component B located in the smart EGO sensor. The block diagram of the proposed distributed system with the proposed communication is given in Fig. 4.4.

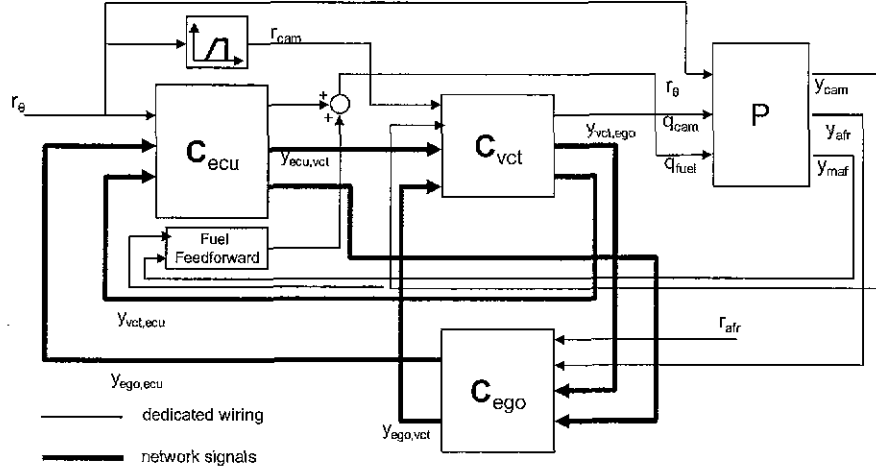


Figure 4.4: VCT Engine with Distributed Discrete MIMO Controller

By using the same discrete MIMO design methodology described in [1] to calculate C_{des} and C_{dist} , and using one of the combined modularity measures developed in the previous section, it is possible to formulate the design optimization problem that maximizes combined component swapping modularity of the VCT Component and the EGO Sensor.

4.3.1 Simultaneous Approach

Calculating the combined swapping modularity of the VCT component and EGO sensor, using the simultaneous approach previously discussed, applies to Cases 2

Element	Solution	Element	Solution	Element	Solution
C_{ecu11}	C_{21}	C_{vct11}	C_{13}	C_{ego11}	C_{22}
C_{ecu12}	1	C_{vct12}	C_{15}	C_{ego12}	C_{24}
C_{ecu13}	1	C_{vct13}	$C_{11} - C_{12}/z$	C_{ego13}	1
C_{ecu21}	1	C_{vct14}	C_{12}	C_{ego14}	0
C_{ecu22}	0	C_{vct21}	$C_{23}z$	C_{ego21}	1
C_{ecu23}	0	C_{vct22}	$C_{25}z$	C_{ego22}	-1
C_{ecu31}	1	C_{vct23}	0	C_{ego23}	0
C_{ecu32}	0	C_{vct24}	0	C_{ego24}	1
C_{ecu33}	0	C_{vct31}	0		
		C_{vct32}	1		
		C_{vct33}	0		
		C_{vct34}	0		

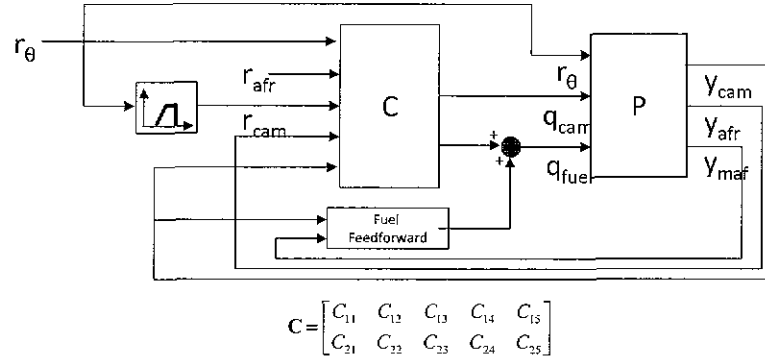
Table 4.1: Distribution Solution with Simultaneous Approach

and 3 in Section 4.1 where a baseline VCT Engine control design is developed and variants of this design (i.e. with different VCT component or EGO Sensor) will be used in different applications.

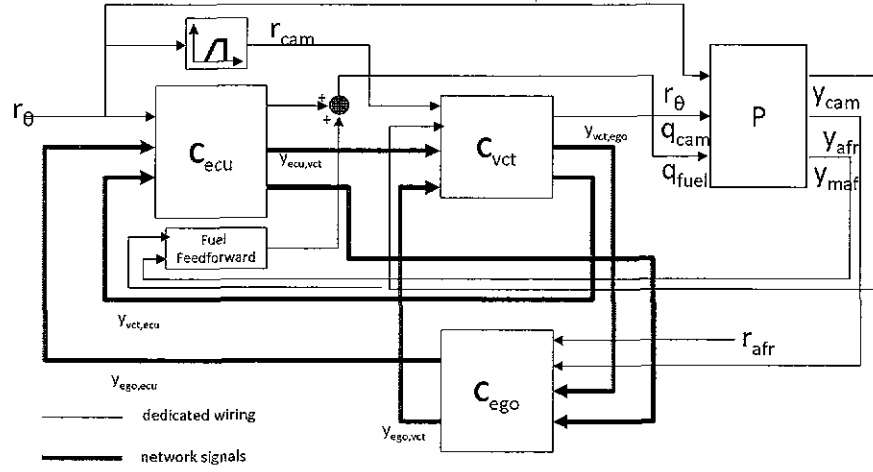
After the pre-optimization procedure described in [1] is used to obtain a candidate solution for the optimization problem, the distribution solution given in Table 4.1 is obtained. The solution presented in Table 4.1 and illustrated in Fig.4.5 has the optimal combined modularity $M_{sys}^* = 613$ based on the measure given in (4.2) where component modularities M_{vct} and M_{ego} have equal weights (i.e. $\rho_{vct} = \rho_{ego} = 1$). The distributed system shown in Fig. 4.5 can be reconfigured by only changing VCT controller, C_{vct} , for different VCT components with plant dynamic properties $\tau_{vct,act} = [7, 67]ms$ and $\tau_{vct,sen} = [10, 20]ms$. Also, the same distributed system can be reconfigured by only changing the EGO controller, C_{ego} , when different EGO sensors with plant dynamic properties $\tau_{ego} = [64, 77]ms$ are used. All the resulting systems would have the desired closed loop characteristics with optimal performance.

The distributed controller presented in Table 4.1 is run in closed loop with the Simulink model of the engine plant model developed as shown in Fig. 4.6. Results of the simulation is then compared to the original centralized MIMO controller and showed no distinguishable difference in response as shown in Fig. 4.7.

(a) Centralized Controller using Traditional Feedback Communications



(b) Distributed Controller utilizing Bi-directional Communications



$$C_{ecu} = \begin{bmatrix} C_{21} & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad C_{vct} = \begin{bmatrix} C_{13} & C_{15} & C_{11} - C_{12}/z & C_{12} \\ C_{23}z & C_{25}z & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad C_{ego} = \begin{bmatrix} C_{22} & C_{24} & 1 & 0 \\ 1 & -1 & 0 & 1 \end{bmatrix}$$

Figure 4.5: Simultaneous and VCT Component first, EGO Sensor second Sequential Solution.

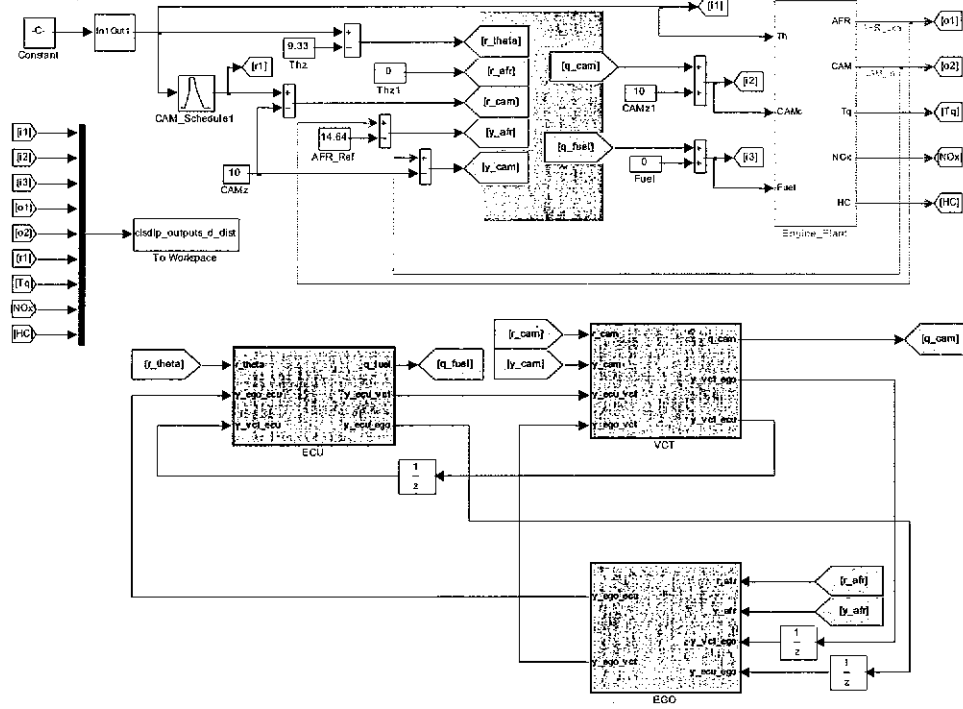


Figure 4.6: Distributed Controller in the Closed Loop

4.3.2 Sequential Approach

Calculating the combined swapping modularity of the VCT Component and EGO Sensor using the sequential approach previously discussed can be used for a case where upgrade and maintenance of an engine system is considered (i.e. Case 1 in Section 4.1). As time progresses, due to failure, or existence of better and cheaper alternatives, components can be changed one at a time and optimal performance is obtained at every step without re-configuring the whole system.

When combined modularity of the VCT Component and EGO Sensor is calculated sequentially two different orders of calculation are possible. If the order is *upgrade VCT Component first, EGO Sensor second* then, after the pre-optimization analysis, the optimal distribution solution obtained is the same as the simultaneous case presented in Table 4.1. This solution has the combined system modularity of

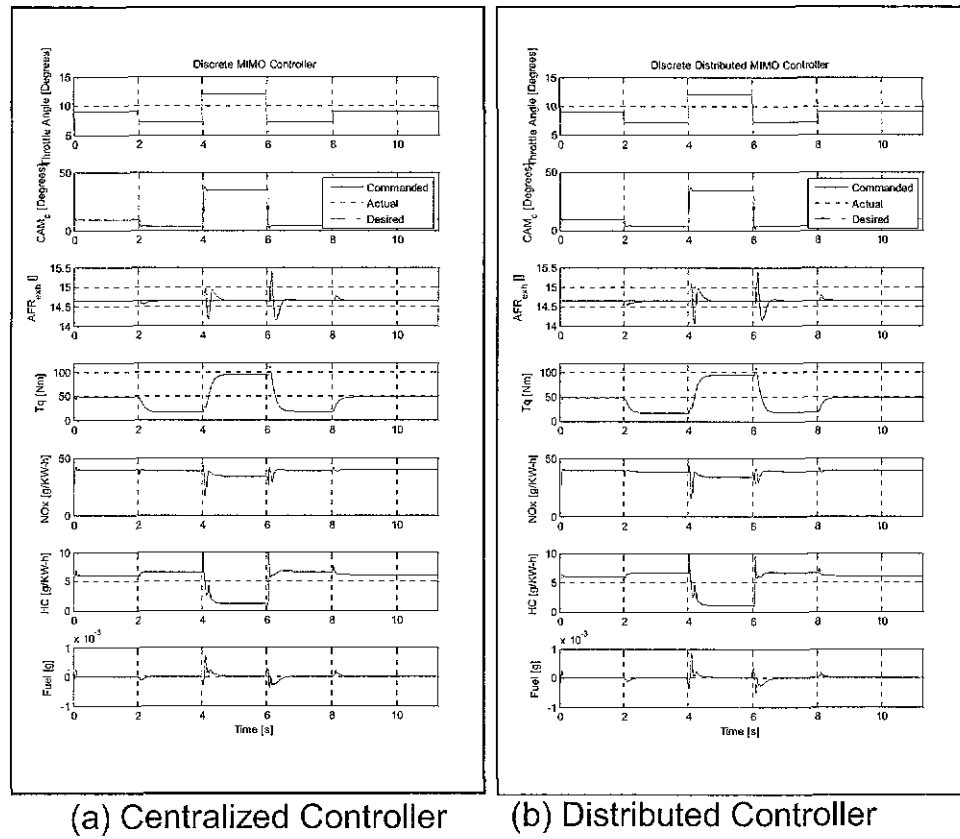


Figure 4.7: Comparison of Centralized vs. Distributed Controllers

Element	Solution	Element	Solution	Element	Solution
C_{ecu11}	C_{21}	C_{vct11}	C_{13}	C_{ego11}	C_{22}
C_{ecu12}	1	C_{vct12}	C_{15}	C_{ego12}	C_{24}
C_{ecu13}	1	C_{vct13}	0	C_{ego13}	0
C_{ecu21}	0	C_{vct14}	1	C_{ego14}	0
C_{ecu22}	0	C_{vct21}	0	C_{ego21}	C_{12}
C_{ecu23}	0	C_{vct22}	0	C_{ego22}	C_{14}
C_{ecu31}	1	C_{vct23}	0	C_{ego23}	0
C_{ecu32}	0	C_{vct24}	0	C_{ego24}	$C_{11}z$
C_{ecu33}	0	C_{vct31}	$C_{23}z$		
		C_{vct32}	$C_{25}z$		
		C_{vct33}	0		
		C_{vct34}	0		

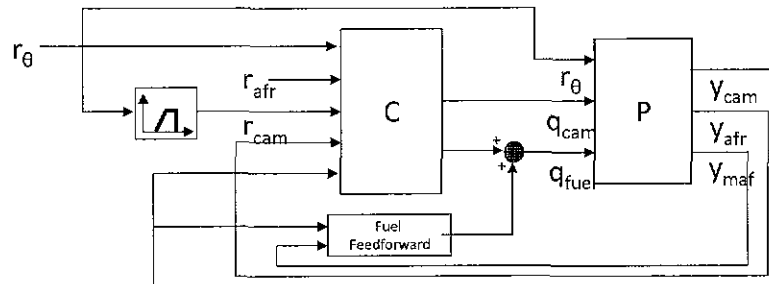
Table 4.2: Distribution Solution with Sequential Approach (EGO first, VCT second)

$M_{sys}^* = 23.2e3$ based on the sequential measure presented in (4.3). The modularity value represents a irregular shaped volume since it involves ranges from three plant parameters. But using a conservative estimate (i.e. giving the maximum dimensions of a rectangular prism that would fit inside this volume), it can be said that VCT components with plant dynamics $\tau_{vct,act} = [7, 67]ms$ and $\tau_{vct,sen} = [10, 20]ms$ can be modularly swapped first by only reconfiguring the VCT controller, C_{vct} , then for each configuration, EGO sensors with plant dynamics $\tau_{ego} = [70, 76]ms$ can be modularly swapped second by only reconfiguring the EGO controller, C_{ego} . All the resulting systems would have the desired closed loop characteristics with optimal performance.

When the assumed order is *upgrade EGO Sensor first, VCT Component second* then, after the pre-optimization analysis, the optimal distribution solution in Table 4.2 and Fig. 4.8 is obtained. The optimal combined system modularity is $M_{sys}^* = 8.1e3$ based on the sequential measure presented in (4.3). For this optimal distribution solution, using the same conservative estimate before, it can be said that EGO sensors with plant dynamics $\tau_{ego} = [64, 78]ms$ can be modularly swapped first by only reconfiguring the EGO controller, C_{ego} , then for each configuration, VCT components with plant dynamics $\tau_{vct,act} = [15, 48]ms$ and $\tau_{vct,sen} = [14, 20]ms$ can be modularly swapped second by only reconfiguring the VCT controller, C_{vct} . All

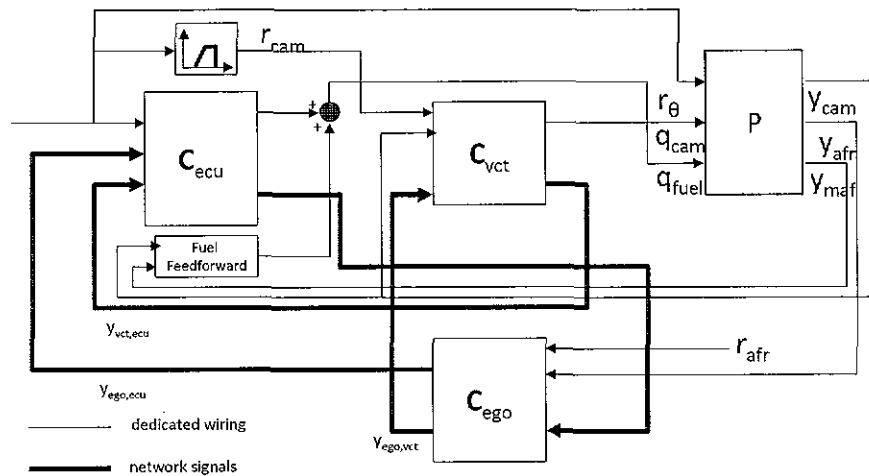
the resulting systems would have the desired closed loop characteristics with optimal performance.

(a) Centralized Controller using Traditional Feedback Communications



$$\mathbf{C} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} & C_{15} \\ C_{21} & C_{22} & C_{23} & C_{24} & C_{25} \end{bmatrix}$$

(b) Distributed Controller utilizing Bi-directional Communications



$$\mathbf{C}_{ecu} = \begin{bmatrix} C_{21} & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{C}_{vet} = \begin{bmatrix} C_{13} & C_{15} & 1 \\ C_{23Z} & C_{25Z} & 0 \end{bmatrix}$$

$$\mathbf{C}_{egn} = \begin{bmatrix} C_{22} & C_{24} & 0 & 0 \\ C_{12} & C_{14} & 0 & C_{11}z \end{bmatrix}$$

Figure 4.8: EGO Sensor first, VCT Component second Sequential Solution.

As in the simultaneous case, the distributed controller presented in Table 4.2 is run in closed loop with the simulink model of the engine plant model developed as shown in Fig. 4.6. Results of the simulation is then compared to the original centralized MIMO controller and showed no distinguishable difference.

4.4 Conclusion

Availability of on-board electronics has increased the use of “smart” components in automatic control systems. The use of bi-directional communications provide additional design freedom which can be used to maximize the swapping modularity of networked smart components. In this paper, application of the design method presented in [2] for combined swapping modularity of system component is discussed. Development of measures for combined swapping modularity is important to be able to analyze more realistic engineering cases. Two approaches (simultaneous and sequential) for combining component swapping modularity of two or more components are presented. These combined modularity approaches are used to design controller which maximizes the component-swapping modularity of the VCT Component (i.e. actuator and sensor) and EGO Sensor for the example given in [1].

Using the *simultaneous* combined modularity approach, we have found that the distributed system shown in Fig. 4.5 can be reconfigured to give optimal controller performance by only changing the VCT controller, C_{vct} , for different VCT components with time constants in the range $\tau_{vct,act} = [7, 67]ms$ and delay times in the range $\tau_{vct,sen} = [10, 20]ms$. The same distributed system can be reconfigured to give optimal controller performance by only changing the EGO controller, C_{ego} , when different EGO sensors with the delay times in the range $\tau_{ego} = [64, 77]ms$ are used.

Using the *sequential* combined modularity approach and assuming the order of upgrades as, VCT component first, EGO sensor second, we have found that with the distributed system shown in Fig. 4.5, VCT components with the time constants in the range $\tau_{vct,act} = [7, 67]ms$ and delay times in the range $\tau_{vct,sen} = [10, 20]ms$ can be modularly swapped first by only reconfiguring the VCT controller, C_{vct} .

Then for each of these configurations, EGO sensors with delay times in the range $\tau_{ego} = [70, 76]ms$ can be modularly swapped second by only reconfiguring the EGO controller, C_{ego} . When we assumed the order of upgrades as, EGO sensor first, VCT component second, EGO sensors with delay times in the range $\tau_{ego} = [64, 78]ms$ can be modularly swapped first by only reconfiguring the EGO controller, C_{ego} , with the optimal distribution solution given in Fig. 4.8. Then for each configuration, VCT components with time constants in the range $\tau_{vct,act} = [15, 48]ms$ and $\tau_{vct,sen} = [14, 20]ms$ can be modularly swapped second by only reconfiguring the VCT controller, C_{vct} . All the resulting systems would have the desired closed loop characteristics with optimal performance.

Our results here show that the combined modularity problem is a more difficult problem than the individual component swapping modularity problem in [1]. Due to their different units, a fair comparison among the different combined modularity measures (i.e. M_{sys} simultaneous and sequential) and component swapping modularity, M_X , is difficult. However, it is seen that the interval of solutions where optimal controllers can be designed is reduced from the results given in [1] (i.e. $\tau_{vct,act} = [7, 67]ms$, $\tau_{vct,sen} = [10, 20]ms$, $\tau_{ego} = [40, 100]ms$) with the increased use of communication paths and higher order transfer function solutions. We also observe that the solutions observed with the sequential approach depend on the sequence of configuration and is not the same as the simultaneous solution.

It is also important to note that distributed controller solutions presented here utilize the bi-directional communications introduced earlier and have improved combined swapping modularity properties than the traditional centralized version of the controller (see Fig. 4.3) if the VCT component or the EGO Sensor is changed the controller has to be redesigned to achieve desired optimal closed loop performance.

Future research on this topic will include, additional applications, as well as improvements to the formulation and solution of the distribution of control problem.

Bibliography

- [1] M. Cakmakci and A. G. Ulsoy. Modular discrete optimal mimo controller for a vct engine. *Proceedings of the American Control Conference*, June 2009. (submitted).
- [2] M. Cakmakci and A.G. Ulsoy. Improving component swapping modularity using bi-directional communication in networked control systems. *IEEE Transactions on Mechatronics*. (in press).
- [3] B.K. Powell and J.A. Cook. Nonlinear low frequency phenomenological engine modeling and analysis. *Proceedings of the American Control Conference*, 24, 1982.
- [4] A. G. Stefanopoulou. *Modeling and Control of Advanced Technology Engines*. PhD thesis, University of Michigan, 1996.
- [5] A.G. Stefanopoulou, J.A. Cook, J.W. Grizzle, and J.S. Freudenberg. Control-oriented model of a dual equal variable cam timing spark ignition engine. *ASME Journal of Dynamic Systems, Measurement and Control*, 120:257–266, 1998.
- [6] A.G. Stefanopoulou, J.S. Freudenberg, and J.W. Grizzle. Variable camshaft timing engine control. *IEEE Transactions on Control Systems Technology*, 8(1):23–34, 2000.
- [7] K. Ulrich and K. Tung. Fundamentals of product modularity. In A. Sharon, editor, *Issues in Mechanical Design International*, pages 73–79. ASME, New York, 1991.

CHAPTER V

Summary and Conclusions

5.1 Summary

The objective of the research described in this dissertation is to improve component swapping modularity of feedback control system components by utilizing smart components and bi-directional network communications among these components. Control systems that are designed to have modularly swappable components can be re-configured, by way of changing their components, and still can operate at their optimal closed loop performance by only re-designing the control algorithm that resides in the component that is being changed.

Previous work on smart components primarily focus on improving on-board diagnostics and performance, previous work on networked control systems focus on alleviating the adverse effects of the network delays and congestion. The previous work on modularity of control systems focus on modularity of hardware and software separately. Thus, this research makes original contributions, which are detailed in Chapters 2 to 4.

In Chapter 2, the bi-directional communications in networked control systems with smart components are defined and the potential benefits of having such systems is discussed. A methodology and problem formulation to maximize component

swapping modularity in SISO feedback control systems is presented and applied to a simple example. Results of the feedback control of driveshaft speed with a smart dc-motor and gearbox and smart tachometer example show significant improvement of the component swapping modularity of the dc-motor and gearbox component. The effect of communication and computation constraints on component-swapping modularity is also investigated.

In Chapter 3, the design methodology developed for a SISO feedback control system is extended to a discrete MIMO controller formulation and applied to a more complex system with higher order plant and controller and for higher number of component plant parameters in the modularity analysis. A discrete optimal MIMO controller for a VCT Engine was developed based on the previous work by the others. Then this MIMO controller was distributed to the engine control unit (ECU), variable camshaft timing (VCT) component controller and Exhaust Gas Oxygen (EGO) sensor controller to maximize the component swapping modularity of each component individually. In order to simplify the numerical solution, a pre-optimization analysis, based on identifying and grouping poles and zeros of the controller that vary as the plant parameters vary, is developed. Resulting distributed controllers showed no distinguishable difference compared to their single controller equivalents.

In Chapter 4, the formulation and the example presented in Chapter 3 is used to find the *combined* component modularity of VCT Engine smart components. Development of measures for combined swapping modularity is important to be able to analyze more realistic engineering cases. Two approaches (simultaneous and sequential) are developed to calculate combined component swapping modularity. Our results showed that the optimal distribution solution is different for each approach and for the order of change of the components in the sequential approach. It is also

shown that the combined modularity problem results showed higher order distributed controller solutions and a narrower range in which component plant parameters can be changed, therefore, suggesting a more difficult problem to solve.

5.2 Engineering Relevance

Traditionally, a change in one of the components of a feedback control system involves two sub-systems, the component and the base controller. By attaching the component related control to the component we can ensure that re-work is done only in one sub-system. Minimizing this rework can be vital for many companies. Today's competitive environment requires a one-hit product with many compelling options, different suppliers, standards, regulations, perceptions of performance, add-ons etc. By increasing the component swapping modularity, product design engineers focus on design for product platforms rather than an individual product. Even though today's software updates can be done in seconds, coming up with the update which will either work for all configurations or keeping track of different versions of the update for different configurations of the product is a major, and costly, effort for companies.

The developed methodology for improving component-swapping modularity of control systems, utilizing smart components and bi-directional network communication, can be used for several engineering scenarios:

1. Sustainable maintenance and/or upgrade of a particular end product
2. Deploying controlled system platform based algorithms
3. Deploying control algorithms for different builds of the same product
4. Reducing costs by developing highly customizable but less variant components

The design method to improve component swapping modularity can also be ap-

plied to non-smart (no on-board cpu) components driven by a centralized controller. In this case, the method would identify the portions of the control algorithm that need to be statically (i.e., calibration change) or dynamically (such as gain scheduling) reconfigured. Even though this kind of use of the method recovers some of the design benefits of using swappable components, the significant benefit of testing a simpler algorithm implemented on a component controller compared to a larger centralized algorithm implemented on a base controller module (such as engine control unit) during verification and validation stage will be lost.

5.3 Future Work

The future research on this topic can be outlined as follows:

1. Additional applications: The identified benefits of improving component swapping modularity drastically increase when the system to which the design method applied is a complex system composed of many components with complicated design requirements. In order to create a technological *pull*, rather than *push*, for this research to everyday industrial applications the complexity of examples it is applied to should be continuously evolving.
2. Improvements on the problem formulation and solution methods: The distribution problem presented here requires formulation of a non-linear optimization problem based on transfer function matrices of the desired and resulting controller structures. The order of the transfer functions and formulation increases as the problems become more complicated. Therefore, continuous improvements on the problem formulation and solution techniques will be needed to tackle more complicated problems. Based on our research group's initial work, formulations based on Linear Matrix Inequalities (LMIs) may provide a promising

path in this quest.

3. Concurrent solution of the control design and distribution problems: The methodology presented in this dissertation requires the solution of control design and distribution problems separately. In fact, the distribution problem is formulated based on assumptions (for example the maximum order of the distributed controller transfer functions) from the results of the control design problem. It is possible that these two steps can be combined into one single solution where the outcome of this new problem is the distributed controller parameters that maximize the component swapping modularity of the system or a particular component.

APPENDICES

APPENDIX A

Matlab Scripts

In this section, Matlab scripts used for numerical solution of the optimization problem are presented. These scripts as presented are configured for the optimization problem presented in Chapter 2 of this dissertation.

`single_max.m` initiates the solution of the optimization problem given a set of initial conditions and as well as accessory information (such as number of optimal solutions to obtain, radius of random perturbation etc.). During its execution, it calls the `fmincon.m` routine from Matlab Optimization Toolbox which is developed for solving constrained nonlinear optimization problems. The `fmincon.m` requires the optimization problem to be defined by developing separate m-scripts for the problem objective function and constraints.

Matlab scripts `objective.m` and `constraints.m` are developed to define the numerical representation of the component-swapping modularity maximization problem in Chapters 2 to 4 of this problem. Both `objective.m` and `constraints.m` call `cdes.m` and `cdist.m` to obtain numerical values of the polynomial constants (as well as some secondary information) of the C_{des} and C_{dist} functions as discussed in Chapters 2, 3, and 4 of this dissertation.

Matlab Function `single_max.m`:

```
function [Fs,Xs,rmagmax, sXkln, sXkun,Klbnd,Kubnd]=single_max(NV,X000,istart,iend
,rmagorig,scl,CompText,CommText,lb,ub,Astrt,Aend,Fsval,iteropt,iX0,Xkl0d,Xku0d,rmagtrsh)
warning off;
EXITFLAG=-1;
rmag=rmagorig;
dummpwd=pwd;
FVALt=1000;
FVALttresh=Fsval;
recFVALt(1)=0;
K=0.1;
sizeX000=size(X000);
% X00=X000(1,:);
cd([CompText '/' CommText])
recEXITFLAGt=0;
if isempty(rmagtrsh)
    rmagtrsh=0;
end
for i=istart:1:iend
    options=optimset('fmincon');
    options.TolFun=1e-4;
    options.TolCon=1e-4;
    options.Display=iteropt;
    param=[1e40,1e40,1e-4,options.TolX];
    EXITFLAGt=-1;
    FVALttresh=Fsval(iX0);
    if i<=iX0
        FVALttresh=Fsval(i);
    end
    rmag=rmagorig;
    Xkl0=[];
    Xku0=[];
    while ~(EXITFLAGt >= 1 && FVALt <= FVALttresh )

        [i rmag FVALttresh min(recFVALt)]
        itry=i;
        if i > sizeX000(1)
            itry=sizeX000(1);
        end
        X0=X000(itry,:)-rmag*rand(1,NV)+rmag*rand(1,NV);
        [C0 Ceq0]=constraints(X0,K,0,param);
        [C1 Ceq1]=constraints(X0,K,1,param);
        simp_conf=0;
        if abs(max(Ceq1)) < abs(max(Ceq0))
            simp_conf=1;
        end
        [Xt,FVALt,EXITFLAGt,OUTPUTt,LAMBDat]=fmincon(@(X) objective(X,K,Astrt,Aend,simp_conf
,options,param,X0,Xkl0,Xku0),X0,[],[],[],[],lb,ub,@(X) constraints(X,K,simp_conf,param)
,options);

        if (rmag ==rmagtrsh && i==1)
            rmag=rmagorig;
        end
        if (rmag ==rmagtrsh && ~(EXITFLAGt >= 1 && FVALt <= FVALttresh ))

            rmag=rmagorig;
        end
        rmag=rmag/scl(i);
    end
end
```

```

        if rmag < max(options.TolX, rmagtrsh)
            rmag = rmagtrsh;
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    options.Display = iteropt;
    recFVALt(i) = FVALt;
    recXt(i,:) = Xt;
    recEXITFLAGt(i) = EXITFLAGt;
    mkdir(['./optimresults/'])
    save(['./optimresults/t' datestr(now, 'ymmdd') '.mat'])
    FVALttresh = FVALt;
    [J dXkln dXkun lbnd ubnd Fvall Fvalu] = objective(Xt, K, Astrt, Aend, simp_conf, options, param,
    XO, Xkl0, Xku0);
    Xkln(:, :, i) = Inf * ones(1000, NV);
    Xkun(:, :, i) = Inf * ones(1000, NV);
    Xkln(1:size(dXkln, 1), :, i) = dXkln;
    Xkun(1:size(dXkun, 1), :, i) = dXkun;
    dKlbnd(i) = lbnd;
    dKubnd(i) = ubnd;
    d_rmag(i) = rmag;
    Fvall
    Fvalu
    options.Display = iteropt;
    FVALttresh = FVALt;
    rmag = rmag * scl(i);
    EXITFLAGt = -1;
    Xt1 = Xt;
end

ptrsh = 1/options.TolX;
[dummy, dummyI] = sort(recFVALt, 1, 'descend');
Xs = recXt(dummyI(iend-iX0+1:iend), :);
Fs = recFVALt(dummyI(iend-iX0+1:iend));
sXkln = Xkln(:, :, dummyI(iend-iX0+1:iend));
sXkun = Xkun(:, :, dummyI(iend-iX0+1:iend));
Klbnd = dKlbnd(dummyI(iend-iX0+1:iend));
Kubnd = dKubnd(dummyI(iend-iX0+1:iend));
rmagmax = max(d_rmag);

```

Matlab Function cdes.m:

```

function [cdn1 cdd1 cdn2 cdd2, clunstab, c1, c2, c3] = Cdes(J, R, b, K, par)

% J=0.1;
% R=1;
% b=0.1;
%Results =[1 2 3 4 5];
clunstab=0;
p1=K/J/R;
p2=(K^2+b*R)/J/R;
%Fitted Values
c1 = [35.544      0.093392]*[K 1]';
c2 = [-14407      38305      -32074      11209      -1589.7
      2054];
*[K^5 K^4 K^3 K^2 K 1]';
c3 = [1.1983e+007 -1.8097e+007  1.0447e+007 -2.8935e+006  3.9092e+005
      -6263.8]

```

```

*[K^5 K^4 K^3 K^2 K 1]';
cdn1=[c1*c3,0,0,0,0,0,0];
cdd1=[0,c2,1,0,0,0,0];
cdn2=-[c3*c1,c3,0,0,0,0,0];
cdd2=[0,c2,1,0,0,0,0];

clroots=roots([1 (p2+c2) (c2*p2+p1*c3) c3*p1*c1]);
if max(real(clroots)) > 0
    clunstab=1;
end

return

```

Matlab Function cdist.m:

```

function [n1 d1 n2 d2]=Cdist(X)

a = X(1);
b = X(2);
c = X(3);
d = X(4);
e = X(5);
f = X(6);
g = X(7);
h = X(8);
i = X(9);
j = X(1);
k = X(2);
l = X(3);
m = X(4);
n = X(5);
o = X(6);
p = X(7);
q = X(8);
r = X(9);
cs = X(10);
t = X(11);
u = X(12);
%
aa = X(13);
ab = X(14);
ac = X(15);
ad = X(16);
ae = X(17);
af = X(18);
%
sa = X(19);
sb = X(20);
sc = X(21);
sd = X(19);
se = X(20);
sf = X(21);

n1(1)=-c*aa*u+c*ab*ac*sb*sf*o-c*aa*sb*o*sf*ad+l*aa*sb*f*sf*ad-c*ab*ae*se*r
+c*aa*u*sf*ad-c*ab*ac*se*sc*o+c*aa*se*r*af-l*aa*se*i*af-c*ab*ac*sf*u
+c*aa*sb*o-l*aa*sb*f+c*aa*se*sc*o*ad-l*aa*se*sc*f*ad-l*ab*se*u
-l*ab*ac*f*sb*sf+l*ab*ac*f*se*sc+l*ab*ae*se*i;
n1(2)=(-(c*aa*sb*n+b*aa*sb*o)*sf*ad+(l*aa*sb*e+k*aa*sb*f)*sf*ad

```

```

+(c*aa*t+b*aa*u)*sf*ad+(l*ab*ac*e+k*ab*ac*f)*se*sc+c*aa*sb*n+b*aa*sb*o
-c*aa*t-b*aa*u+(c*aa*se*q+b*aa*se*r)*af+(c*aa*se*sc*n+b*aa*se*sc*o)*ad
-l*aa*sb*e-k*aa*sb*f-l*ab*se*t-k*ab*se*u-c*ab*ae*se*q-b*ab*ae*se*r
-c*ab*ac*sf*t-b*ab*ac*sf*u+c*ab*ac*sb*sf*n+b*ab*ac*sb*sf*o-c*ab*ac*se*sc*n
-b*ab*ac*se*sc*o-(l*aa*se*h+k*aa*se*i)*af-(l*aa*se*sc*e+k*aa*se*sc*f)*ad
+l*ab*ae*se*h+k*ab*ae*se*i-(l*ab*ac*e+k*ab*ac*f)*sb*sf);
n1(3)=(l*ab*ac*d+k*ab*ac*e+j*ab*ac*f)*se*sc-(l*ab*ac*d+k*ab*ac*e+j*ab*ac*f)*sb*sf
+(c*aa*cs+h*aa*t+a*aa*u)*sf*ad-a*ab*ac*sf*u-c*ab*ae*se*p+c*aa*sb*m+b*aa*sb*n
+a*aa*sb*o-c*aa*cs-b*aa*t-a*aa*u-(c*aa*sb*m+b*aa*sb*n+a*aa*sb*o)*sf*ad
+(c*aa*se*p+b*aa*se*q+a*aa*se*r)*af+(c*aa*se*sc*m+b*aa*se*sc*n
+a*aa*se*sc*o)*ad-l*aa*sb*d-k*aa*sb*e-j*aa*sb*f-l*ab*se*cs
-k*ab*se*t-j*ab*se*u-b*ab*ae*se*q-a*ab*ae*se*r-c*ab*ac*sf*cs-b*ab*ac*sf*t
+c*ab*ac*sb*sf*m+b*ab*ac*sb*sf*n+a*ab*ac*sb*sf*o-c*ab*ac*se*sc*m
-b*ab*ac*se*sc*n-a*ab*ac*se*sc*o+(l*aa*sb*d+k*aa*sb*e+j*aa*sb*f)*sf*ad
-(l*aa*se*g+k*aa*se*h+j*aa*se*i)*af-(l*aa*se*sc*d+k*aa*se*sc*e
+j*aa*se*sc*f)*ad+l*ab*ae*se*g+k*ab*ae*se*h+j*ab*ae*se*i);
n1(4)=-(b*aa*sb*m+a*aa*sb*n)*sf*ad+(b*aa*cs+a*aa*t)*sf*ad+(k*aa*sb*d
+j*aa*sb*e)*sf*ad+(k*ab*ac*d+j*ab*ac*e)*se*sc-(k*ab*ac*d+j*ab*ac*e)*sb*sf
+b*aa*sb*m+a*aa*sb*n-b*aa*cs-a*aa*t+(b*aa*se*p+a*aa*se*q)*af+(b*aa*se*sc*m
+a*aa*se*sc*n)*ad-k*aa*sb*d-j*aa*sb*e-k*ab*se*cs-j*ab*se*t-b*ab*ae*se*p
-a*ab*ae*se*q-b*ab*ac*sf*cs-a*ab*ac*sf*t+b*ab*ac*sb*sf*m+a*ab*ac*sb*sf*n
-b*ab*ac*se*sc*m-a*ab*ac*se*sc*n-(k*aa*se*g+j*aa*se*h)*af-(k*aa*se*sc*d
+j*aa*se*sc*e)*ad+k*ab*ae*se*g+j*ab*ae*se*h);
n1(5)=(a*aa*sb*m-a*aa*cs+a*aa*cs*sf*ad-a*aa*sb*m*sf*ad+a*aa*se*p*af
-j*aa*se*g*af-a*ab*ac*sf*cs+j*ab*ae*se*g-j*aa*sb*d+a*aa*se*sc*m*ad
-j*aa*se*sc*d*ad+a*ab*ac*sb*sf*m-j*ab*ac*d*sb*sf+j*aa*sb*d*sf*ad
-a*ab*ae*se*p-j*ab*se*cs-a*ab*ac*se*sc*m+j*ab*ac*d*se*sc);
n1(6)=0;
n1(7)=0;
%
n2(1)=-sa*aa*f*u-sd*aa*u*i*af+sa*aa*se*af*r*f-sd*aa*sb*af*r*f-sa*ab*ae*se*r*f
+sa*aa*f*u*sf*ad-sd*aa*u*sc*f*ad-sa*ab*se*o*u-sd*ab*u^2-sa*ab*ac*f*sf*u
-sa*aa*se*af*i*o+sd*aa*sb*af*i*o+sa*ab*ae*se*i*o+sd*ab*u*sb*o+sd*ab*ae*i*u
+sd*ab*ae*sb*r*f+sd*ab*ac*sc*f*u-sd*ab*ae*sb*i*o;
n2(2)=((sa*aa*f*t+sa*aa*e*u)*sf*ad+sd*ab*ae*sb*r*e-sa*aa*f*t-sa*aa*e*u
-2*sd*ab*u*t-sa*aa*se*af*i*n-sa*aa*se*af*h*o+sa*aa*se*af*r*o+sa*aa*se*af*q*f
-sa*ab*se*o*t-sa*ab*se*n*u-(sd*aa*u*h+sd*aa*t*i)*af+sa*ab*ae*se*i*n
+sa*ab*ae*se*h*o-sa*ab*ae*se*r*e-sa*ab*ae*se*q*f-sa*ab*ac*f*sf*t
-sa*ab*ac*e*sf*u-(sd*aa*u*sc*e+sd*aa*t*sc*f)*ad+sd*aa*sb*af*i*n
+sd*aa*sb*af*h*o-sd*aa*sb*af*r*e-sd*aa*sb*af*q*f+sd*ab*u*sb*n+sd*ab*t*sb*o
+sd*ab*ae*i*t+sd*ab*ae*h*u-sd*ab*ae*sb*i*n-sd*ab*ae*sb*h*o+sd*ab*ae*sb*q*f
+sd*ab*ac*sc*f*t+sd*ab*ac*sc*e*u);
n2(3)=(-sd*ab*(2*u*cs+t^2)+(sa*aa*f*cs+sa*aa*e*t+sa*aa*d*u)*sf*ad
+sd*ab*ae*sb*p*f+sa*aa*se*af*q*e-sa*aa*f*cs-sa*aa*e*t-sa*aa*d*u
-sa*aa*se*af*i*m-sa*aa*se*af*h*n-sa*aa*se*af*g*o+sa*aa*se*af*r*d
+sa*aa*se*af*p*f-sa*ab*se*o*cs-sa*ab*se*n*t-sa*ab*se*m*u-(sd*aa*u*g
+sd*aa*t*h+sd*aa*cs*i)*af+sa*ab*ae*se*i*m+sa*ab*ae*se*h*n+sa*ab*ae*se*g*o
-sa*ab*ae*se*r*d-sa*ab*ae*se*q*e-sa*ab*ae*se*p*f-sa*ab*ac*f*sf*cs
-sa*ab*ac*e*sf*t-sa*ab*ac*d*sf*u-(sd*aa*u*sc*d+sd*aa*t*sc*e
+sd*aa*cs*ac*f)*ad+sd*aa*sb*af*i*m+sd*aa*sb*af*h*n+sd*aa*sb*af*g*o
-sd*aa*sb*af*r*d-sd*aa*sb*af*q*e-sd*aa*sb*af*p*f+sd*ab*u*sb*m+sd*ab*t*sb*n
+sd*ab*cs*sb*o+sd*ab*ae*i*cs+sd*ab*ae*h*t+sd*ab*ae*g*u-sd*ab*ae*sb*i*m
-sd*ab*ae*sb*h*n-sd*ab*ae*sb*g*o+sd*ab*ae*sb*r*d+sd*ab*ae*sb*q*e
+sd*ab*ac*sc*f*cs+sd*ab*ac*sc*e*t+sd*ab*ac*sc*d*u);
n2(4)=(sd*ab*t*sb*m-sa*aa*se*cs-sa*aa*d*t+(sa*aa*se*cs+sa*aa*d*t)*sf*ad
-2*sd*ab*t*cs-sa*aa*se*af*h*m-sa*aa*se*af*g*n+sa*aa*se*af*q*d+sa*aa*se*af*p*e
-sa*ab*se*n*cs-sa*ab*se*m*t-(sd*aa*t*g+sd*aa*cs*h)*af+sa*ab*ae*se*h*m
+sa*ab*ae*se*g*n-sa*ab*ae*se*q*d-sa*ab*ae*se*p*e-sa*ab*ac*e*sf*cs
-sa*ab*ac*d*sf*t-(sd*aa*t*sc*d+sd*aa*cs*sc*e)*ad+sd*aa*sb*af*h*m

```

```

+sd*aa*sb*af*g*n-sd*aa*sb*af*q*d-sd*aa*sb*af*p*+sd*ab*cs*sb*n+sd*ab*ae*h*cs
+sd*ab*ae*g*t-sd*ab*ae*sb*h*m-sd*ab*ae*sb*g*n+sd*ab*ae*sb*q*d
+sd*ab*ae*sb*p*+sd*ab*ac*sc*e*cs+sd*ab*ac*sc*d*t);
n2(5)=(-sa*aa*se*af*g*m-sa*aa*d*cs+sa*aa*d*cs*sf*ad+sa*aa*se*af*p*d
-sa*ab*se*m*cs-sd*ab*cs^2-sa*ab*ac*d*sf*cs-sd*ab*ae*sb*g*m+sd*aa*sb*af*g*m
+sa*ab*ae*se*g*m+sd*ab*cs*sb*m-sd*aa*cs*g*af+sd*ab*ae*sb*p*d
-sd*aa*sb*af*p*d-sa*ab*ae*se*p*d+sd*ab*ae*g*cs-sd*aa*cs*sc*d*ad
+sd*ab*ac*sc*d*cs);
n2(6)=0;
n2(7)=0;
%
d1(1)=-u^2-ae*sb*i*o-u*sb*o*sf*ad-ae*sb*sf*ad*r*+ae*i*u+u^2*sf*ad
+ae*sb*r*+u*se*r*af-ae*se*sc*ad*i*o-ae*i*u*sf*ad+u*sb*o+ae*sb*sf*ad*i*o
+u*se*sc*o*ad+ae*se*sc*ad*r*+ac*u*i*sf*af-ac*sb*sf*af*i*o+ac*sb*sf*af*r*+f
-ac*se*sc*af*r*+ac*u*sc*f+ac*se*sc*af*i*o;
d1(2)=(t*sb*o-(ae*i*t+ae*h*u)*sf*ad+(ac*u*h+ac*t*i)*sf*af-ae*sb*sf*ad*r*+e
-2*u*t+2*u*t*sf*ad-ae*sb*i*n-ae*sb*h*o+ae*sb*r*+ae*sb*q*f+ac*u*sc*e+ac*t
*sc*f+u*sb*n+ae*i*t+ae*h*u-(u*sb*n+t*sb*o)*sf*ad+(u*se*q+t*se*r)*af
+(u*se*sc*n+t*se*sc*o)*ad+ae*sb*sf*ad*i*n+ae*sb*sf*ad*h*o-ae*sb*sf*ad*q*f
-ae*se*sc*ad*i*n-ae*se*sc*ad*h*o+ae*se*sc*ad*r*+ae*se*sc*ad*q*f
-ac*sb*sf*af*i*n-ac*sb*sf*af*h*o+ac*sb*sf*af*r*+ac*sb*sf*af*q*f
+ac*se*sc*af*i*n+ac*se*sc*af*h*o-ac*se*sc*af*r*+ac*se*sc*af*q*f);
d1(3)=(2*u*cs+t^2)*sf*ad-(ae*i*cs+ae*h*t+ae*g*u)*sf*ad+(ac*u*g+ac*t*h
+ac*cs*i)*sf*af+ae*se*sc*ad*p*f-ac*se*sc*af*r*d-2*u*cs-t^2-ae*sb*i*m
-ae*sb*h*n-ae*sb*g*o+ae*sb*r*d+ae*sb*q*+ae*sb*p*f+ac*u*sc*d+ac*t*sc*e
+ac*cs*sc*f+u*sb*m+t*sb*n+cs*sb*o-(u*sb*m+t*sb*n+cs*sb*o)*sf*ad+ae*i*cs
+ae*h*t+ae*g*u+(u*se*p+t*se*q+cs*se*r)*af+(u*se*sc*m+t*se*sc*n
+cs*se*sc*o)*ad+ae*sb*sf*ad*i*m+ae*sb*sf*ad*h*n+ae*sb*sf*ad*g*o
-ae*sb*sf*ad*r*d-ae*sb*sf*ad*q*+ae*sb*sf*ad*p*f-ae*se*sc*ad*i*m-ae*se
*sc*ad*h*n-ae*se*sc*ad*g*o+ae*se*sc*ad*r*d+ae*se*sc*ad*q*+ac*sb*sf*af*i*m
-ac*sb*sf*af*h*n-ac*sb*sf*af*g*o+ac*sb*sf*af*r*d+ac*sb*sf*af*q*+ac*sb
*sf*af*p*f+ac*se*sc*af*i*m+ac*se*sc*af*h*n+ac*se*sc*af*g*o-ac*se*sc*af*q*+e
-ac*se*sc*af*p*f);
d1(4)=(ac*se*sc*af*g*n-2*t*cs+2*t*cs*sf*ad-ae*sb*h*m-ae*sb*g*n+ae*sb*q*d
+ae*sb*p*+ac*t*sc*d+ac*cs*sc*e+t*sb*m+cs*sb*n-(t*sb*m+cs*sb*n)*sf*ad
+ae*h*cs+ae*g*t+(t*se*p+cs*se*q)*af+(t*se*sc*m+cs*se*sc*n)*ad-(ae*h*cs
+ae*g*t)*sf*ad+ae*sb*sf*ad*h*m+ae*sb*sf*ad*g*n-ae*sb*sf*ad*q*d-ae*sb*sf
*ad*p*+ae*se*sc*ad*h*m-ae*se*sc*ad*g*n+ae*se*sc*ad*q*d+ae*se*sc*ad*p*+e
+(ac*t*g+ac*cs*h)*sf*af-ac*sb*sf*af*h*m-ac*sb*sf*af*g*n+ac*sb*sf*af*q*d
+ac*sb*sf*af*p*+ac*se*sc*af*h*m-ac*se*sc*af*q*d-ac*se*sc*af*p*);
d1(5)=(cs^2*sf*ad+cs*se*p*af-cs^2+cs*sb*m-cs*sb*m*sf*ad+cs*se*sc*m*ad
+ae*g*cs+ac*cs*g*sf*af+ae*sb*p*d+ac*sb*sf*af*p*d-ae*se*sc*ad*g*m
-ae*g*cs*sf*ad+ac*cs*sc*d+ae*sb*sf*ad*g*m+ac*se*sc*af*g*m+ae*se*sc*ad*p*d
-ae*sb*g*m-ac*sb*sf*af*g*m-ae*sb*sf*ad*p*d-ac*se*sc*af*p*d);
d1(6)=0;
d1(7)=0;
%
d2(1)=-u^2-ae*sb*i*o-u*sb*o*sf*ad-ae*sb*sf*ad*r*+ae*i*u+u^2*sf*ad+ae
*sb*r*+u*se*r*af-ae*se*sc*ad*i*o-ae*i*u*sf*ad+u*sb*o+ae*sb*sf*ad*i*o
+u*se*sc*o*ad+ae*se*sc*ad*r*+ac*u*i*sf*af-ac*sb*sf*af*i*o+ac*sb*sf*af
*r*+ac*se*sc*af*r*+ac*u*sc*f+ac*se*sc*af*i*o;
d2(2)=(t*sb*o-(ae*i*t+ae*h*u)*sf*ad+(ac*u*h+ac*t*i)*sf*af-ae*sb*sf*ad*r*+e
-2*u*t+2*u*t*sf*ad-ae*sb*i*n-ae*sb*h*o+ae*sb*r*+ae*sb*q*f+ac*u*sc*e+ac*t
*sc*f+u*sb*n+ae*i*t+ae*h*u-(u*sb*n+t*sb*o)*sf*ad+(u*se*q+t*se*r)*af
+(u*se*sc*n+t*se*sc*o)*ad+ae*sb*sf*ad*i*n+ae*sb*sf*ad*h*o-ae*sb*sf*ad*q*f
-ae*se*sc*ad*i*n-ae*se*sc*ad*h*o+ae*se*sc*ad*r*+ae*se*sc*ad*q*f-ac*sb*sf
*af*i*n-ac*sb*sf*af*h*o+ac*sb*sf*af*r*+ac*sb*sf*af*q*f+ac*se*sc*af*i*n
+ac*se*sc*af*h*o-ac*se*sc*af*r*+ac*se*sc*af*q*f);
d2(3)=(2*u*cs+t^2)*sf*ad-(ae*i*cs+ae*h*t+ae*g*u)*sf*ad+(ac*u*g+ac*t*h

```

```

+ac*cs*i)*sf*af+ae*se*sc*ad*p*f-ac*se*sc*af*r*d-2*u*cs-t^2-ae*sb*i*m
-ae*sb*h*n-ae*sb*g*o+ae*sb*r*d+ae*sb*q*e+ae*sb*p*f+ac*u*sc*d+ac*t*sc*e
+ac*cs*sc*f+u*sb*m+t*sb*n+cs*sb*o-(u*sb*m+t*sb*n+cs*sb*o)*sf*ad+ae*i*cs
+ae*h*t+ae*g*ut+(u*se*p+t*se*q+cs*se*r)*af+(u*se*sc*m+t*se*sc*n+cs*se
*sc*o)*ad+ae*sb*sf*ad*i*m+ae*sb*sf*ad*h*n+ae*sb*sf*ad*g*o-ae*sb*sf*ad*r*d
-ae*sb*sf*ad*q*e-ae*sb*sf*ad*p*f-ae*se*sc*ad*i*m-ae*se*sc*ad*h*n-ae*se
*sc*ad*g*o+ae*se*sc*ad*r*d+ae*se*sc*ad*q*e-ac*sb*sf*af*i*m-ac*sb*sf*af*h*n
-ac*sb*sf*af*g*o+ac*sb*sf*af*r*d+ac*sb*sf*af*q*e+ac*sb*sf*af*p*f+ac*se
*sc*af*i*m+ac*se*sc*af*h*n+ac*se*sc*af*g*o-ac*se*sc*af*q*e-ac*se*sc*af*p*f);
d2(4)=(ac*se*sc*af*g*n-2*t*cs+2*t*cs*sf*ad-ae*sb*h*m-ae*sb*g*n+ae*sb*q*d
+ae*sb*p*e+ac*t*sc*d+ac*cs*sc*e+t*sb*m+cs*sb*n-(t*sb*m+cs*sb*n)*sf*ad
+ae*h*cs+ae*g*t+(t*se*p+cs*se*q)*af+(t*se*sc*m+cs*se*sc*n)*ad-(ae*h*cs
+ae*g*t)*sf*ad+ae*sb*sf*ad*h*m+ae*sb*sf*ad*g*n-ae*sb*sf*ad*q*d-ae*sb*sf
*ad*p*e-ae*se*sc*ad*h*m-ae*se*sc*ad*g*n+ae*se*sc*ad*q*d+ae*se*sc*ad*p*e
+(ac*t*g+ac*cs*h)*sf*af-ac*sb*sf*af*h*m-ac*sb*sf*af*g*n+ac*sb*sf*af*q*d
+ac*sb*sf*af*p*e+ac*se*sc*af*h*m-ac*se*sc*af*q*d-ac*se*sc*af*p*e);
d2(5)=(cs^2*sf*ad+cs*se*p*af-cs^2+cs*sb*m-cs*sb*m*sf*ad+cs*se*sc*m*ad
+ae*g*cs+ac*cs*g*sf*af+ae*sb*p*d+ac*sb*sf*af*p*d-ae*se*sc*ad*g*m-ae*g*cs
*sf*ad+ac*cs*sc*d+ae*sb*sf*ad*g*m+ac*se*sc*af*g*m+ae*se*sc*ad*p*d-ae*sb*g
*m-ac*sb*sf*af*g*m-ae*sb*sf*ad*p*d-ac*se*sc*af*p*d);
d2(6)=0;
d2(7)=0;

return

```

Matlab Function objective.m:

```

function [J Xkl Xku lbnd ubnd, Fvall Fvalu]=objective(X,K0,Astrt,
Aend,simp_conf,doptions,param,X0,Xkl0,Xku0)
%General Settings
Xbound=200000;
magorig=0;
dkmax=0.01;
dkmin=0.01;
trshflg=1;
magscl=10;
Xkl=X;
Xku=X;
Fvall=[0,0];
Fvalu=[0,0];
options=optimset('fsolve');
options.Display='off';
options.TolFun=doptions.TolCon;
options.NonlEqnAlgorithm='lm';

warning off
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Xp=X;
[C Ceq]=constraints(Xp,K0,simp_conf,param);
if (max(abs(Ceq)) > 100*doptions.TolFun)
    J=0;
    lbnd=K0;
    ubnd=K0;
    Xkl=X;
    Xku=X;
    Fvall=max(Ceq);
    Fvalu=max(Ceq);

```

```

    return
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mag=magorig;
k1=1;
dk=0;
dkprev=dk;
dkres=dkmax;
doneflag=-1;
X0ini=Xp(Astrt:Aend);
lbnd=K0;
lbndcan=K0;
while doneflag==-1

    X0=X0ini+mag*rand(1,Aend-Astrt+1)-mag*rand(1,Aend-Astrt+1);
    if (k1 <= size(Xk10,1) && Xk10(k1,1)~=Inf)
        X0=Xk10(k1,Astrt:Aend)+mag*rand(1,Aend-Astrt+1);
    end
    [X,FVAL,EXITFLAG1]=fsolve(@(X) controlwithA(X,lbndcan,Xp,Astrt,Aend,
    simp_conf,param),X0,options);
    if (max(abs(X))<Xbound) && (EXITFLAG1 >= trshflg)
        doneflag=-1;
        mag=magorig;
        X0ini=X;
        Fval1(k1,:)=max(FVAL) EXITFLAG1;
        Xk1(k1,:)=Xp;
        Xk1(k1,Astrt:Aend)=X;
        k1=k1+1;
        dkprev=dk;
        dkres=dkmax;
        dk=dk+dkres;
        lbnd=lbndcan;
        lbndcan=K0-dk;
    end

    if (EXITFLAG1 < trshflg)
        dkres=dkres/10;
        dk=dkprev+dkres;
        lbndcan=K0-dk;
    end
    if dkres < dkmin
        doneflag=1;
        break
    end
    if lbndcan < 0.05
        doneflag=1;
        break
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mag=magorig;
k2=1;
dk=0;
dkprev=dk;
dkres=dkmax;
doneflag=-1;
X0ini=Xp(Astrt:Aend);
ubnd=K0;
ubndcan=K0;

```

```

while doneflag==-1
    X0=X0ini+mag*rand(1,Aend-Astrt+1)-mag*rand(1,Aend-Astrt+1);
    if (k2 <= size(Xku0,1) && Xku0(k2,1)~=Inf)
        X0=Xku0(k2,Astrt:Aend)+mag*rand(1,Aend-Astrt+1)-mag*rand(1,Aend-Astrt+1);
    end
    [X,FVAL,EXITFLAG1]=fsolve(@(<X> controlwithA(X,ubndcan,Xp,Astrt,Aend
    ,simp_conf,param),X0,options);
    if (max(abs(X))<Xbound) && (EXITFLAG1 >= trshflg)
        mag=magorig;
        doneflag=-1;
        Fvalu(k2,:)= [max(FVAL) EXITFLAG1];
        X0ini=X;
        Xku(k2,:)= Xp;
        Xku(k2,Astrt:Aend)=X;
        k2=k2+1;
        dkprev=dk;
        dkres=dkmax;
        dk=dk+dkres;
        ubnd=ubndcan;
        ubndcan=K0+dk;
    end
    if (EXITFLAG1 < trshflg)
        dkres=dkres/10;
        dk=dkprev+dkres;
        ubndcan=K0+dk;
    end
    if dkres < dkmin
        doneflag=1;
        break
    end
    if ubndcan > 1
        doneflag=1;
        break
    end
end
%lbnd
%ubnd
J1=lbnd-ubnd;
Xk1=Xk1;
Xku1=Xku;
lbnd1=lbnd;
ubnd1=ubnd;
Fvall1=Fvall;
Fvalu1=Fvalu;
clear Xk1 Xku lbnd ubnd Fvall Fvalu;

%General Settings
%Xbound=200000;
magorig=0;
dkmax=0.01;
dkmin=0.01;
trshflg=1;
magscl=10;
Xk1=Xp;
Xku=Xp;
Fvall=[0,0];
Fvalu=[0,0];
options=optimset('fsolve');
options.Display='off';

```

```

options.TolFun=options.TolCon;
options.NonlEqnAlgorithm='lm';
options.LineSearchType='cubicpoly';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mag=magorig;
k1=1;
dk=0;
dkprev=dk;
dkres=dkmax;
doneflag=-1;
X0ini=Xp(Astrt:Aend);
lbnd=K0;
lbndcan=K0;
while doneflag==-1

    X0=X0ini+mag*rand(1,Aend-Astrt+1)-mag*rand(1,Aend-Astrt+1);
    if (k1 <= size(Xk10,1) && Xk10(k1,1)~=Inf)
        X0=Xk10(k1,Astrt:Aend)+mag*rand(1,Aend-Astrt+1);
    end
    [X,FVAL,EXITFLAG1]=fsolve(@(X) controlwithA(X,lbndcan,Xp,Astrt,Aend
    ,simp_conf,param),X0,options);
    if (max(abs(X))<Xbound) && (EXITFLAG1 >= trshflg)
        doneflag=-1;
        mag=magorig;
        X0ini=X;
        Fvall(k1,:)=max(FVAL) EXITFLAG1];
        Xk1(k1,:)=Xp;
        Xk1(k1,Astrt:Aend)=X;
        k1=k1+1;
        dkprev=dk;
        dkres=dkmax;
        dk=dk+dkres;
        lbnd=lbndcan;
        lbndcan=K0-dk;
    end

    if (EXITFLAG1 < trshflg)
        dkres=dkres/10;
        dk=dkprev+dkres;
        lbndcan=K0-dk;
    end

    if dkres < dkmin
        doneflag=1;
        break
    end

    if lbndcan < 0.05
        doneflag=1;
        break
    end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mag=magorig;
k2=1;
dk=0;
dkprev=dk;
dkres=dkmax;
doneflag=-1;

```

```

X0ini=Xp(Astrt:Aend);
ubnd=K0;
ubndcan=K0;
while doneflag==-1
    X0=X0ini+mag*rand(1,Aend-Astrt+1)-mag*rand(1,Aend-Astrt+1);
    if (k2 <= size(Xku0,1) && Xku0(k2,1)~=Inf)
        X0=Xku0(k2,Astrt:Aend)+mag*rand(1,Aend-Astrt+1)
        -mag*rand(1,Aend-Astrt+1);
    end
    [X,FVAL,EXITFLAG1]=fsolve(@(X) controlwithA(X,ubndcan,Xp,Astrt,Aend,
    simp_conf,param),X0,options);
    if (max(abs(X))<Xbound) && (EXITFLAG1 >= trshflg)
        mag=magorig;
        doneflag=-1;
        Fvalu(k2,:)= [max(FVAL) EXITFLAG1];
        X0ini=X;
        Xku(k2,:)= Xp;
        Xku(k2,Astrt:Aend)=X;
        k2=k2+1;
        dkprev=dk;
        dkres=dkmax;
        dk=dk+dkres;
        ubnd=ubndcan;
        ubndcan=K0+dk;
    end
    if (EXITFLAG1 < trshflg)
        dkres=dkres/10;
        dk=dkprev+dkres;
        ubndcan=K0+dk;
    end
    if dkres < dkmin
        doneflag=1;
        break
    end
    if ubndcan > 1
        doneflag=1;
        break
    end
end
%lbnd
%ubnd
J=lbnd-ubnd;

if J1<J
    Xk1=Xk11;
    Xku=Xku1;
    lbnd=lbnd1;
    ubnd=ubnd1;
    Fvall=Fvall1;
    Fvalu=Fvalu1;
    J=J1;
end

```

Matlab Function constraints.m:

```

function [C, Ceq]=constraints(Xp,K,simp_conf,param)

%%%Calculate Cdes %%%

```

```

J=0.1;
R=1;
b=0.1;
wn=1;

[cdn1 cdd1 cdn2 cdd2,clunstab,c1,c2,c3]=Cdes(J,R,b,K,wn);

%%%%0
[cdn1, cdd1, o]=stb_plzr_canc(cdn1,cdd1,param(1),param(2),param(3));
[cdn2, cdd2, o]=stb_plzr_canc(cdn2,cdd2,param(1),param(2),param(3));
[cdgain cdn1 cdd1 cdn2 cdd2]=simple_poly(cdn1, cdd1, cdn2, cdd2,0);

%%%Calculate Cdist %%%
[n1 d1 n2 d2]=Cdist(Xp);
[ gain n1 d1 n2 d2]=simple_poly(n1,d1,n2,d2,0);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Ceq1=[cdn1-n1];
Ceq2=[cdd1-d1];
Ceq3=[cdn2-n2];
Ceq4=[cdd2-d2];
Ceq5=cdgain-gain;
Ceq=[Ceq1,Ceq2,Ceq3,Ceq4,Ceq5];
C=[];

%%%1
[cdn1 cdd1 cdn2 cdd2,clunstab,c1,c2,c3]=Cdes(J,R,b,K,wn);
[cdgain1 cdn1 cdd1 cdn2 cdd2]=simple_poly(cdn1, cdd1, cdn2, cdd2,1);
%%%Calculate Cdist %%%
[n1 d1 n2 d2]=Cdist(Xp);
[n1, d1, o]=stb_plzr_canc(n1,d1,param(1),param(2),param(3));
[n2, d2, o]=stb_plzr_canc(n2,d2,param(1),param(2),param(3));
[ gain1 n1 d1 n2 d2]=simple_poly(n1,d1,n2,d2,1);
%%%Roots Stable%%%%
Ctfusb=clunstab;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ceq11=[cdn11-n11];
Ceq21=[cdd11-d11];
Ceq31=[cdn21-n21];
Ceq41=[cdd21-d21];
Ceq51=cdgain1-gain1;
Ceq1=[Ceq11,Ceq21,Ceq31,Ceq41,Ceq51];

if max(abs(Ceq1)) < max(abs(Ceq))
    Ceq=[Ceq1];
end
Ctf=proptf_cons(Xp,param(4));
Ceq=[Ceq, Ctf, Ctfusb];

```

APPENDIX B

Simulink Models

In this section, the Matlab/Simulink models which are used to simulate the closed loop response of the VCT Engine and MIMO controller are presented.

Specifically,

- Figure B.1 shows the top level diagram of the developed simulink model with discrete MIMO controller,
- Figure B.2 details the Engine subsystem block shown in Figure B.1,
- Figure B.3 shows the contents of the “Engine.Plant” subsystem block in Figure B.2,
- Figure B.4 shows the contents of the subsystem named “Regression.Functions” in Figure B.3,
- Figure B.5 shows the top level diagram of the developed simulink model with distributed discrete MIMO controllers,
- Figure B.6 shows contents of the subsystem named “ECU” in Figure B.5,
- Figure B.7 shows contents of the subsystem named “VCT” in Figure B.5,
- Figure B.8 shows contents of the subsystem named “EGO” in Figure B.5.

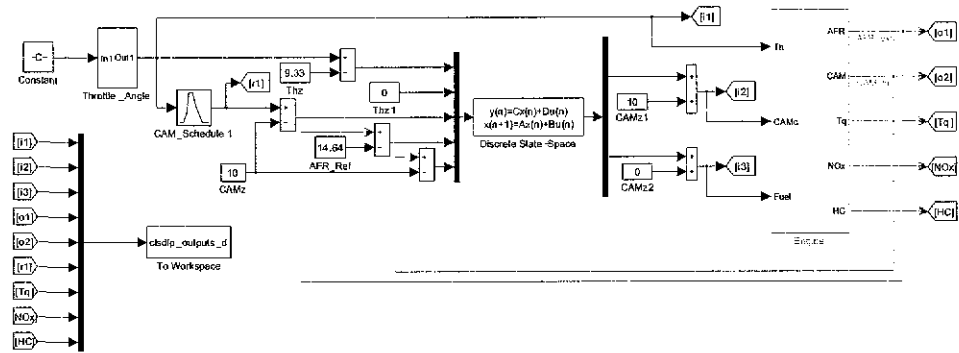


Figure B.1: Closed Loop System with MIMO Controller

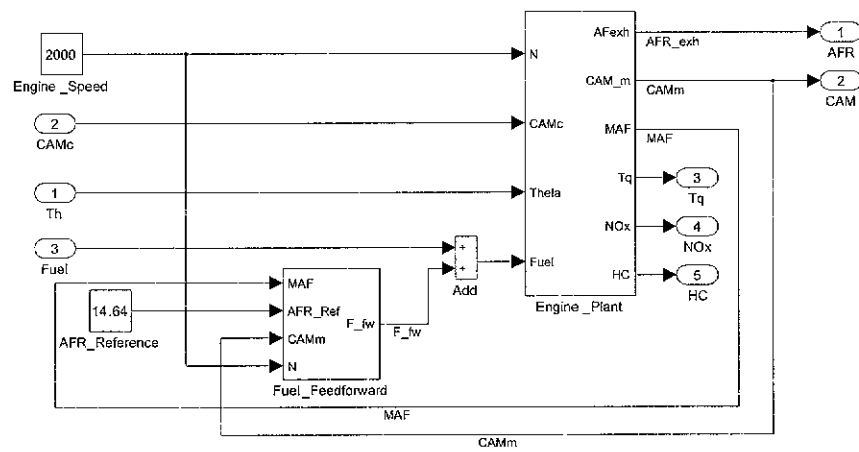


Figure B.2: Engine Plant and Feedforward Fuel Controller

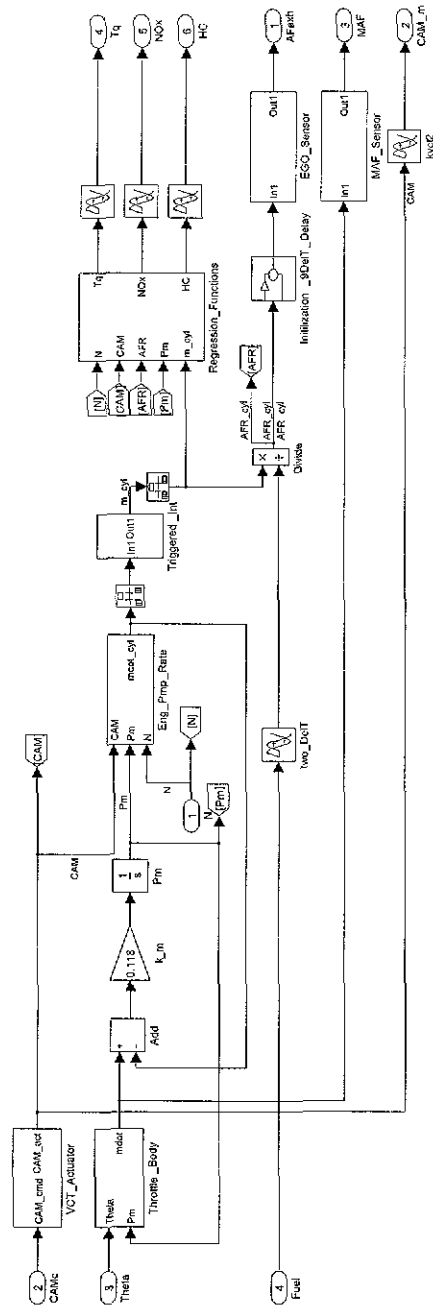


Figure B.3: Engine Plant

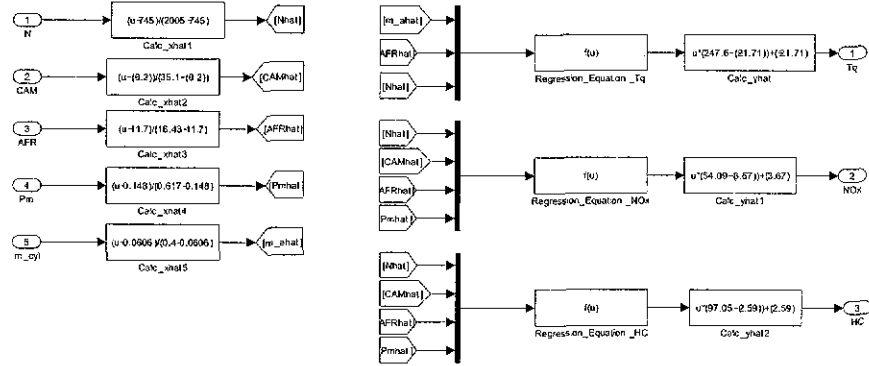


Figure B.4: Regression Equations for Engine Performance Variables

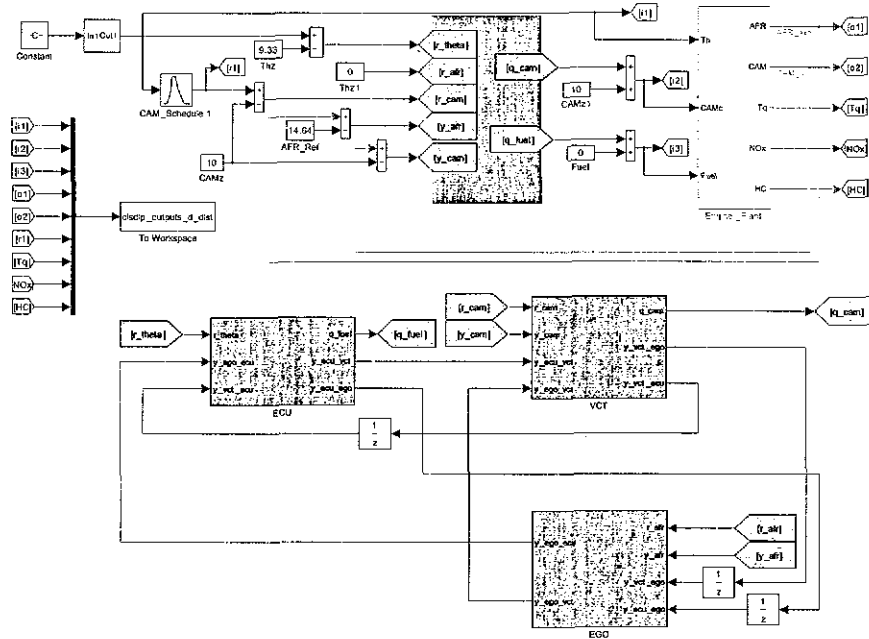


Figure B.5: Closed Loop System with Distributed Controllers

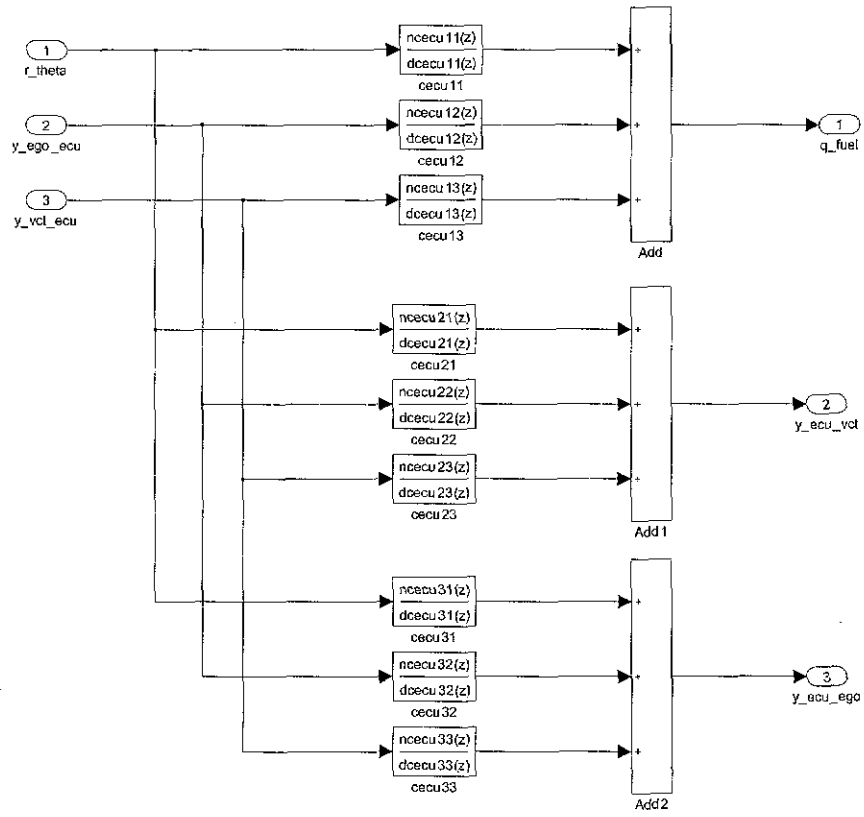


Figure B.6: ECU Subsystem

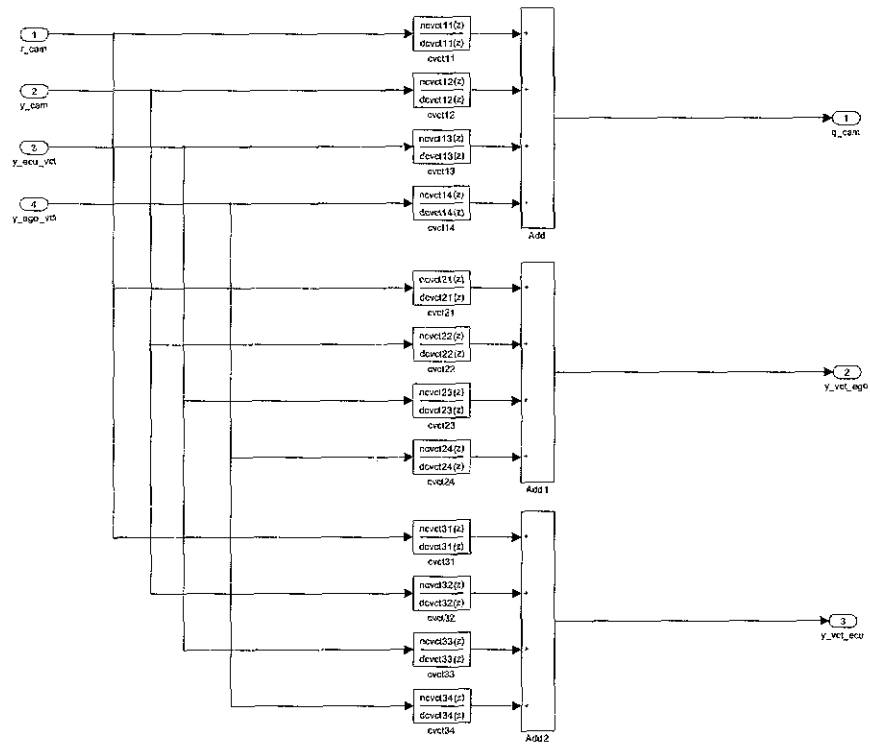


Figure B.7: VCT Subsystem

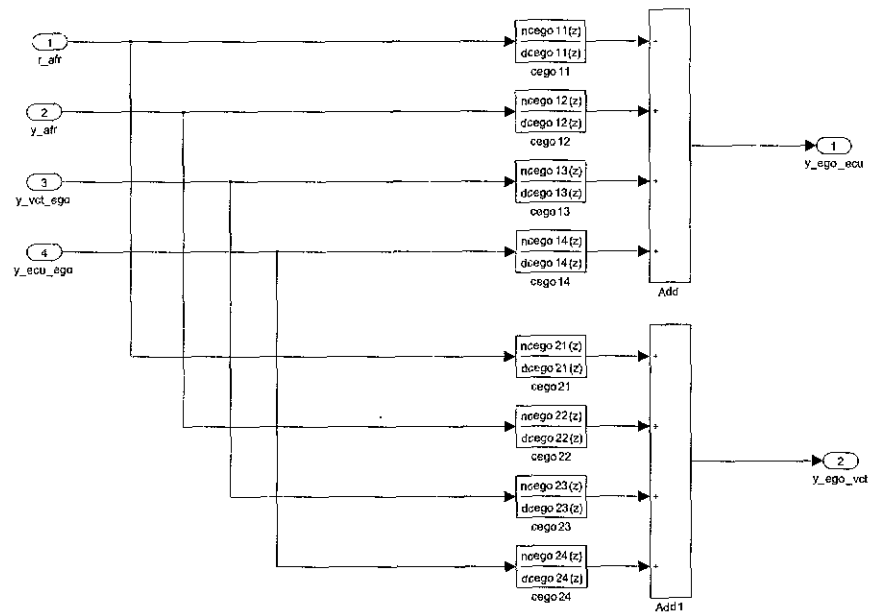


Figure B.8: EGO Subsystem

ABSTRACT

Mechatronic Design for Component-Swapping Modularity using Bi-Directional
Communications in Networked Control Systems

by

Melih Cakmakci

Chair: A. Galip Ulsoy

Availability of low cost electronics has led to a new type of control system components, so called “smart” components, which can perform control actions in the actuator and sensor components as well as in the controller. The objective of our research is to improve component swapping modularity of feedback control system components by utilizing smart such components, and the bi-directional network communications among these components. Control systems that are designed to have modularly swappable components can be re-configured, by changing their components, and still can operate at their optimal closed loop performance by only re-designing the control algorithm that resides in the component that is being changed.

As a first step, a quantification measure for swapping-modularity of single component and an optimal design formulation that maximizes component swapping modularity are presented. The developed method is then demonstrated with a driveshaft speed control example where the component swapping modularity of the actuator

(i.e. dc-motor and gearbox smart component) is maximized. The effects of communication and computation constraints on the resulting component-swapping modularity are also investigated.

The problem formulation originally developed for single input single output (SISO) controllers and continuous time systems is then extended to apply to multiple input multiple output (MIMO) controllers with discrete-time formulations. A control oriented *pre-optimization* technique, based on pole zero locations which simplifies numerical solution of the optimization problem is developed. The method is then applied to the distribution of a discrete MIMO controller for a Variable Camshaft Timing (VCT) Engine to improve the component swapping modularity of the VCT and the Exhaust Gas Oxygen (EGO) Sensor components using network communications.

Development of measures for combined swapping modularity is important to be able to analyze more complex engineering cases. As the final research topic, two approaches (simultaneous and sequential) for combining component swapping modularity of two or more components are discussed. These combined modularity approaches are used to design controllers which maximize the component-swapping modularity of the VCT component and the EGO sensor for an internal combustion engine as an example.