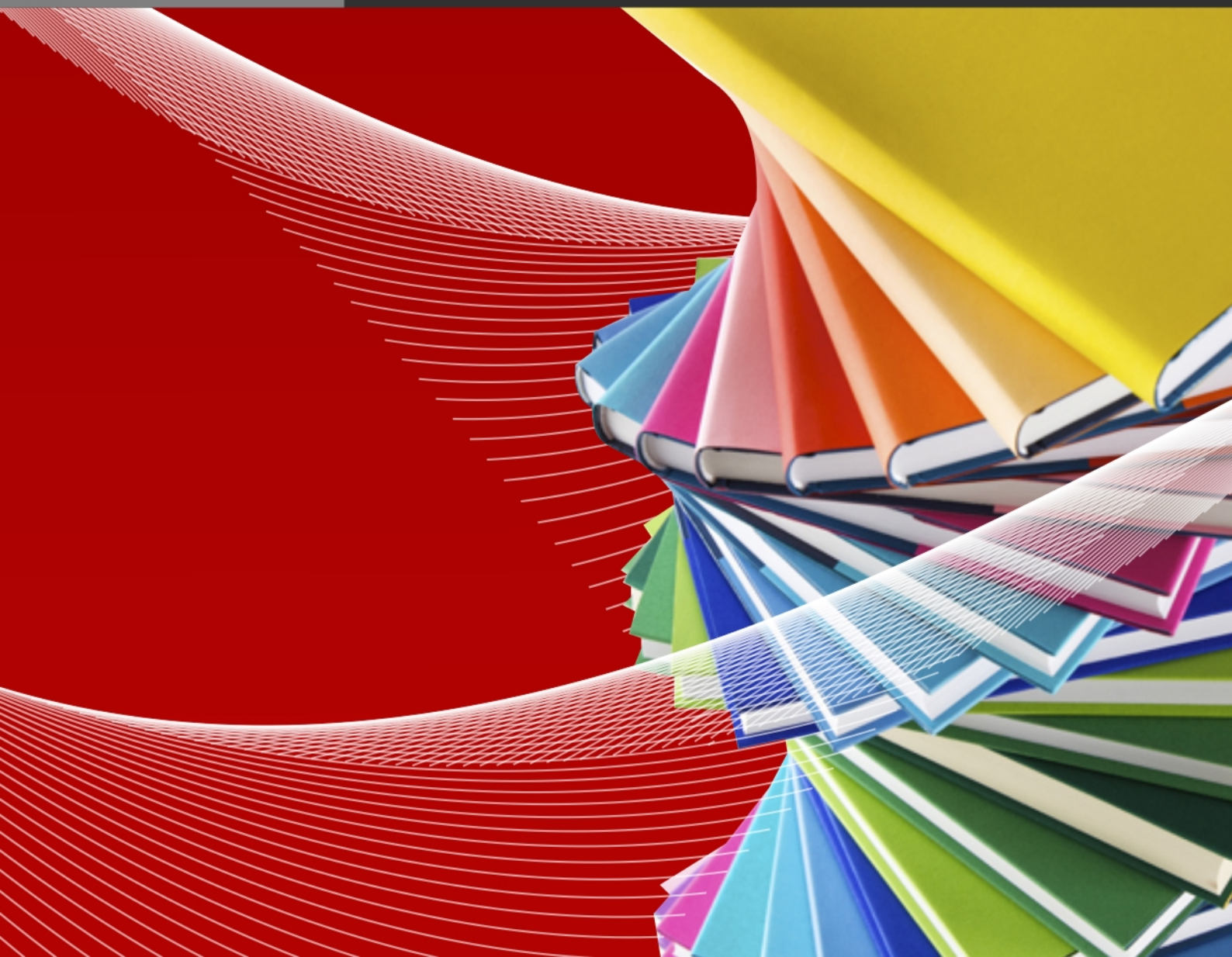
澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

Outstanding Academic Papers by Students
學生優秀作品



DESIGN AND IMPLEMENTATION OF AN INTELLIGENT ADAPTIVE CONTROLLER FOR AIR-FUEL RATIO CONTROL OF AN AUTOMOTIVE ENGINE SYSTEM

by

Xu Yuxiang

Luo Zichong

Cai Quan

B.Sc. in Electromechanical Engineering

2015

Faculty of Science and Technology
University of Macau

Design and Implementation of an Intelligent Adaptive Controller for Air-fuel Ratio Control of an Automotive Engine System

by

Xu Yuxiang (D-B1-2896-6)

Luo Zichong (D-B1-2768-2)

Cai Quan (D-B0-2337-4)

Final Year Project Report submitted in partial fulfillment of the requirements for the degree of

BSc. in Electromechanical Engineering

Faculty of Science and Technology
University of Macau

2015

University of Macau

Abstract

Design and Implementation of an Intelligent Adaptive Controller for Air-fuel Ratio Control of an Automotive Engine System

by

Xu Yuxiang (D-B1-2896-6)

Luo Zichong (D-B1-2768-2)

Cai Quan (D-B0-2337-4)

Project Supervisor: Prof. Wong Pak Kin

Department of Electromechanical Engineering, Faculty of Science and Technology

Air-fuel ratio (AFR) control is essential for maintaining the best engine performance. Practical technique for AFR control is the proportional-integral-derivative control in which the process in deriving the best controller parameters is very tedious and even a well-tuned controller still cannot guarantee long-term control performance. In the literature, various strategies have been developed for AFR control. These include the sliding mode control, fuzzy logic control and model predictive control based on intelligent techniques. However, all of these aforesaid methods require prior expert knowledge of the engine before the controller construction. If no prior knowledge is available or the available knowledge is not sufficient, it is practically impossible to

construct a reliable controller. To address this issue, this project aims to design an intelligent adaptive controller for AFR control, in which no prior knowledge, no pre-trained model and no optimizer are required. An intelligent method called fully online sequential extreme learning machine (FOS-ELM) is used to construct the controller. Simulations have been conducted to verify the designed controller. Moreover, most of studies in the literature only perform simulations for controller verification. In this project, to evaluate the effectiveness of the designed controller, experiments are further set up on a real test engine. The signals of the engine sensors were studied and analyzed so that the controller could be successfully implemented on the test engine. Both simulation and experimental results demonstrate that the designed intelligent adaptive AFR controller is effective for maintaining the AFR to a desired level. In addition, the designed controller has been compared with the engine built-in AFR controller. The comparison shows that the designed controller can achieve better tracking performance than the built-in one, indicating that the designed controller in this project is feasible and promising.

ACKNOWLEDGEMENTS

The authors wish to thank the supervisor, Prof. Wong Pak Kin, for his patient instruction, constructive guidance and valuable comments. The authors would also like to thank Mr. Wong Ka In and Mr. Wong Hang Cheong for their assistance with experiment setup and their precious help in the development of the intelligent controller. Without the support and encouragement from them, the authors cannot make the completion and success of this project.

TABLE OF CONTENT

LIST OF TABLES

Number *Page*

LIST OF FIGURES

Number *Page*

LIST OF ABBREVIATIONS

$\boldsymbol{\lambda}$	Lambda (normalized air-fuel ratio)
ANN	Artificial neural network
BP	Back propagation
CO	Carbon monoxide
CO_2	Carbon dioxide
ECU	Electronic control unit
ELM	Extreme learning machine
FOS-ELM	Fully online sequential extreme learning machine
HC	Hydrocarbon
IAE	Integral absolute error
LS	Least-squares
NO_x	Nitrogen oxides
OS-ELM	Online sequential extreme learning machine
PI	Proportional-integral
PID	Proportional-integral-derivative
RLS	Recursive least-squares
SMC	Sliding mode control

CHAPTER 1: INTRODUCTION

This chapter serves as an introductory overview of the project. The background of the project is briefly studied, followed by a literature review related to the project scope. The objective of the project is also given at the end of this chapter.

1.1 GENERAL BACKGROUND

Air pollution is a hot topic in Eastern society. Earlier in spring 2015, air pollution problem was bombed in mainland China because a documentary film concerning the air pollution in China, named “Under the Dome” (Chai, 2015), was posted on the Internet and spread quickly around the whole world. In the film, the director investigated the truth behind the air pollution problem in China through some factory visits and interviews with environmental experts, government officials and business owners. It has been mentioned in the film that, apart from industrial emissions, vehicular exhaust emissions are also one of the major contributors to the poor air quality in China. This is because there are too many on-road vehicles in China, and the using rate of them is especially high among all the countries in the world. This fact can be realized from Figures 1.1 and 1.2, which show the trend for the number of new passenger vehicles sold

and the total amount of vehicles in China from 2006 to 2013, and the trend of oil production and consumption in China from 2006 to 2015.

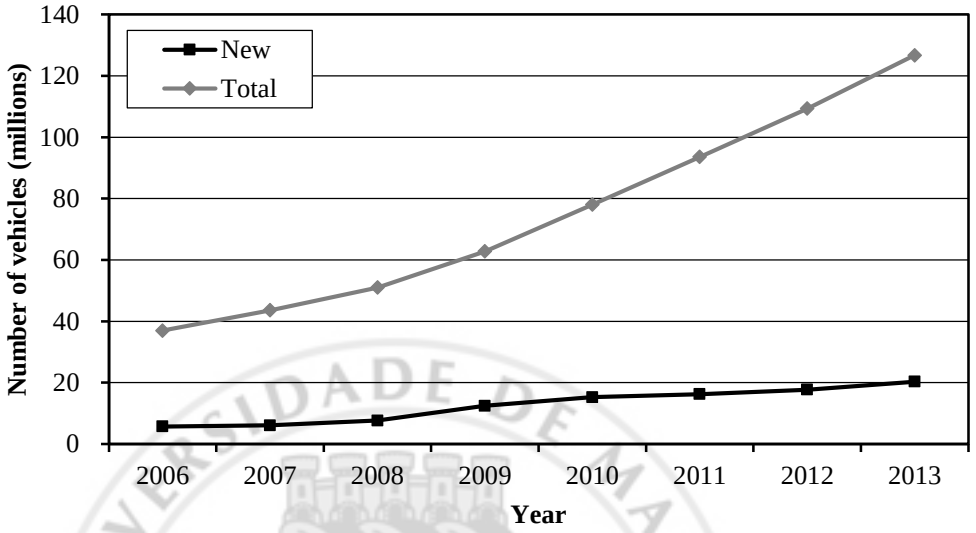


Figure 1.1 Number of vehicles in China from 2006 to 2013[1]

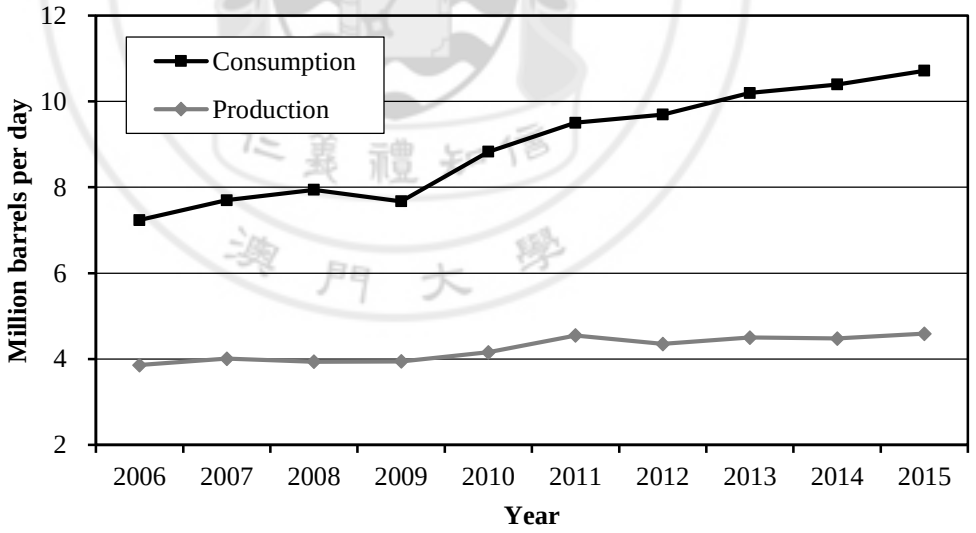


Figure 1.2 Oil production and consumption in China from 2006 to 2015[2]

[1] Data source: China Statistical Yearbook 2014, retrieved from http://www.stats.gov.cn/tjsj/ndsj/2014/indexeh.htm, on May 6 2015.
[2] Data source: EIA International Energy Statistics and Short-Term Energy Outlook April 2015, retrieved from http://www.eia.gov/forecasts/steo/outlook.cfm#issues2014, on May 6 2015.

Figures 1.1 and 1.2 reveal that the trend of increasing number of vehicles is still strong, as Chinese have great demand for vehicles due to the huge amount of population. It can also be predicted from the data that the number of vehicles will be continuously growing for future decades, in China as well as in the world. This is not a favorable phenomenon since too many vehicles on the road not only give rise to the intractable traffic congestion problems, but also result in serious environmental problems as excessive amount of exhaust gases are generated from running vehicles. The exhaust gases seriously affect the environmental surroundings and harm the residents in the air-polluted area (Kan, et al., 2012). For instance, carbon monoxide (CO), a product of incomplete combustion, is colorless but poisonous. Inhalation of high concentrations of CO can be fatal. Nitrogen oxides (NO_x), formed inside the engine under high temperature and pressure, can reduce the oxygen transport efficiency if they are breathed into lungs. Hydrocarbon (HC), which is odorant and irritant, has been considered as a factor that may cause cancer. On the other hand, NO_x and HC can react under sunlight to form photochemical smog, which is another pollution problem to the world (Pulkrabek, 2004). Moreover, carbon dioxide (CO_2), which is a greenhouse gas produced by vehicle engine, is the cause of global warming. Therefore, it has been an essential concern for the government to deal with the

problems of vehicular emissions, and increasingly stringent emission standards have been established.

In addition, since fossil fuels are unrenewable resources, they will eventually be depleted if they are still being wasted because of using vehicles in an unnecessary way. Thus, with the growing consumption of oil, price of fuel also maintains at a high position, which may also lead to global economic problems. As a result, vehicle manufacturers have been investigating different technologies to reduce the amount of emissions and fuel consumption while maintaining a high-level of engine performance.

1.2 SPECIFIC BACKGROUND

Currently, three-way catalytic converters have been widely adopted to reduce the exhaust pollutants from vehicles. They are installed on the exhaust pipe of vehicles to reduce HC and CO by oxidization and NO_x by reduction (Faiz, et al., 1996). It is important to note that, the conversion efficiency of the three-way catalytic converters is strongly affected by the mass ratio of air to fuel present in the engine (Heywood, 1988), or known as the engine air-fuel ratio (AFR). For example, rich or stoichiometric AFR values are more favorable for NO_x reduction, while slightly lean or stoichiometric AFR values can help lower the CO level. Figure 1.3 shows the effect of AFR on the conversion efficiency of

three-way catalyst converters.

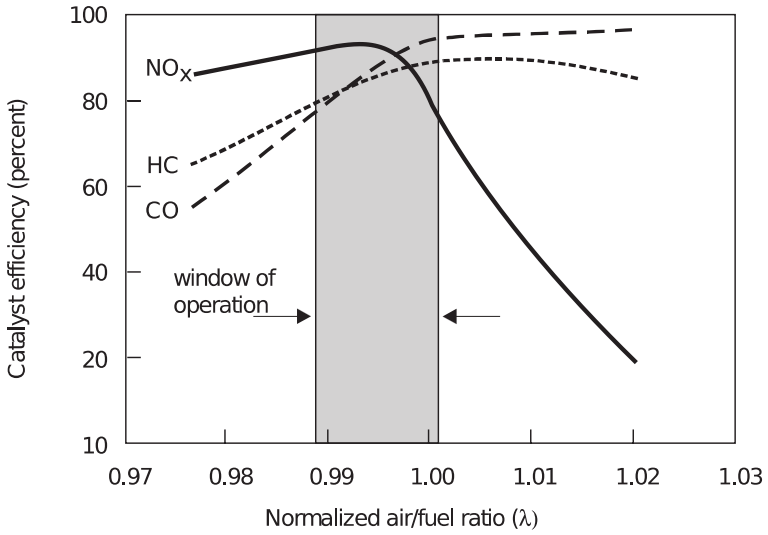


Figure 1.3 Effect of AFR on three-way catalyst efficiency (Faiz et al., 1996)

The λ value in Figure 1.3 is a normalized AFR value, which is defined as:

$$\lambda = \frac{\mathrm{AFR}_{\mathrm{actual}}}{\mathrm{AFR}_{\mathrm{stoichiometric}}} \tag{1.1}$$

where $\mathrm{AFR}_{\mathrm{actual}}$ is the actual AFR of the combustion mixture, and $\mathrm{AFR}_{\mathrm{stoichiometric}}$ is the stoichiometric AFR value of that mixture. It is obvious from Eq. (1.1) that the stoichiometric λ value is 1 no matter what mixture or fuel blend is burnt in the engine (but the stoichiometric AFR value is different for different fuel blend).

According to Figure 1.3, derivation of only 1% from the stoichiometric λ value can result in significant degradation on the conversion efficiency of the three-way catalytic converter. Thus, λ sensors (or also known as the oxygen

sensors) are usually installed on the three-way catalytic converter to monitor the status of the AFR performance and the functionality of the three-way catalytic converter, as shown in Figure 1.4.

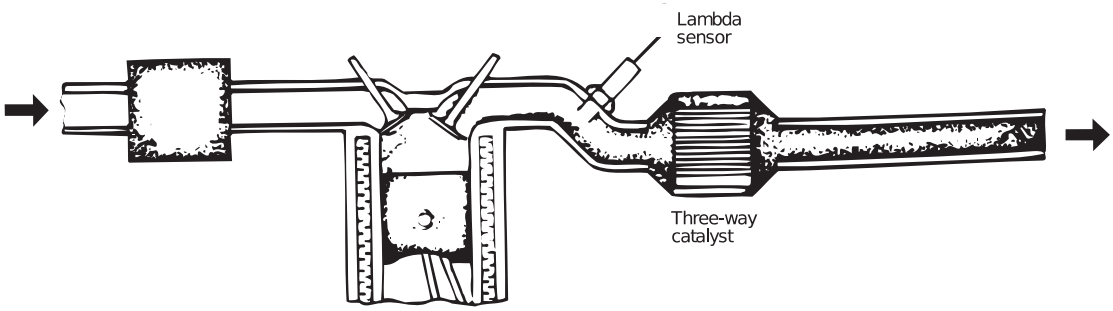


Figure 1.4 λ sensor and three-way catalyst on vehicle exhaust pipe (Faiz et al., 1996)

Practically, AFR control refers to the control of λ value. There are in general three target λ values for practical AFR control (Wong, et al., 2014):

(i) 1.00 for highest three-way catalytic conversion efficiency;

(ii) 0.95 for maximum engine power; and

(iii) 1.05 for best brake-specific fuel consumption (i.e., best economy).

Effective AFR control (i.e., λ control) is therefore essential for an automotive engine system to achieve different desired engine performance under various operating conditions, and it has been one of the most significant control problems for gasoline engines.

1.3 LITERATURE REVIEW

In the past decades, lots of researchers have focused on the feedback control of the AFR, and many strategies for AFR control have been developed. In most production engines, AFR control is done by using proportional-integral (PI) or proportional-integral-derivative (PID) controllers. PI and PID controllers are quite stable for maintaining normal engine operation in short-term, but the tuning process of the gains (e.g., gain P, gain I and gain D) is arduous and time-consuming, and the tuned constant gains cannot guarantee the control performance in long-term as the AFR dynamics may gradually change due to engine aging. It is interesting to notice that, in recent years, Franceschi, et al. (2007) and Ebrahimi, et al. (2012) respectively proposed an adaptive PID controller and a parameter-varying filtered PID strategy for AFR control, in which the gains of the controller can be varied in accordance with the engine conditions. However, noise disturbance and the highly nonlinear nature of AFR dynamics could still be some serious problems for the "linear-based" PID controllers.

To handle the nonlinearity of AFR, sliding-mode control (SMC) was extensively utilized (Puleston, et al., 2002, Won, et al., 1998). To name some, Choi, et al. (1998) developed an observer-based fuel injection control algorithm

based on SMC strategy and the results were shown to be better than engine factory controller. Yoon, et al. (2001) proposed an adaptive SMC strategy for AFR control, in which an adaptive update law was derived for the fuel parameters. The results in both simulations and experiments showed attractive control performance in transient state. Following the work of Yoon, et al. (2001), Souder, et al. (2004) designed an improved adaptive SMC algorithm for AFR control. Three adaptive laws were derived to allow the simultaneous update of three model parameters. Their study showed promising tracking performance in simulations. More recently, Pace, et al. (2012) also employed SMC for AFR control of a dual-fuel engine, and Ebrahimi, et al. (2014) proposed to use a second-order SMC strategy for AFR control of lean-burn engines.

Despite its effectiveness for AFR control, SMC requires an analytical dynamic model of the "to-be-controlled" engine. An exact dynamic model is almost impossible to derive in reality and excessive assumption has to be made in the model derivation. Thus, there always exist unmodeled dynamics in SMC based AFR controller, for example, neither the fuel film dynamics nor the fuel properties were well solved in (Choi, et al., 1998).

In view of the difficulties in deriving exact mathematical models for AFR dynamics, researchers have raised the idea of using artificial intelligence

approaches, such as fuzzy logic and machine learning techniques, for approximating the unknown model. Correspondingly, Al-Olimat, et al. (2000), Lauber, et al. (2011) and Morelos, et al. (2012) demonstrated how fuzzy logic can be used to estimate the engine model parameters, and the control performance of all these previous studies was verified via simulations, showing that this intelligent technique could be used for AFR control. Nevertheless, fuzzy logic requires expert knowledge to define the logic rules. Even an experienced engineer may not be able to determine an accurate sets of rules.

The machine learning approach, on the other hand, has drawn considerable attention in the literature. While SMC and fuzzy logic requires prior knowledge in model construction, machine learning techniques aim to "learn" the model directly from sample data. Owing to this advantage, machine learning techniques have been used by many researchers for building up prediction models for AFR control. For instance, Manzie, et al. (2002) used a radial basis function network, while Beltrami, et al. (2003) used a one hidden layer perceptron with linear output unit, to estimate the air mass flow into the engine cylinder. Arsie, et al. (2006), Zhai, et al. (2010), Wong, et al. (2012), Sardarmehni, et al. (2013), Wong, et al. (2014) and Wong, et al. (2015) respectively employed different kinds of machine learning techniques, including

multilayer perceptron, recurrent neural network, radial basis function network, least-squares support vector machines, online sequential extreme learning machine and sparse Bayesian extreme learning machine, for AFR predictions. The results from all these studies showed that the predictions from ANNs are much better than classic observers, leading to better AFR control performance than traditional approaches. However, in most of these researches, the controllers were only evaluated by simulations, but not experimentally. Although machine learning techniques provide promising predictions of AFR, the gradient information of these methods is generally not available. Thus, an iterative-based optimizer is usually used for determining the control signal from the AFR predictions, which may be computationally inefficient for real-time AFR control.

In order to overcome this issue while maintaining the use of the attractive machine learning approach, adaptive control may be the best choice (Spooner, et al., 2002). Instead of using an optimizer to determine the control signal, adaptive controller obtains the control law directly from the inverse of the approximation. Hence, no pre-defined model, no iterative-based optimizer and no prior knowledge are required in this control strategy. The only necessary thing for adaptive control using machine learning technique is the derivation of a

promising adaptive law, such that stable and accurate approximation of the AFR dynamics can be guaranteed.

The most famous type of machine learning techniques for adaptive control is the artificial neural network (ANN). The critical issue of the ANN is that the parameters of the networks are learned/adjusted using back-propagation (BP). This learning scheme can easily stuck in local minima and usually requires high computational complexity as large amount of variables need be tuned/determined (e.g., input weights and output weights). Hence, it may not be suitable for complicated adaptive situations. In a recent study by Wong, et al. (2014), a machine learning technique called fully online sequential extreme learning machine (FOS-ELM) was proposed. The limitations of local minima, slow converging speed and overfitting risk suffered in the BP learning algorithm were addressed by the proposed FOS-ELM. Some simulations were also conducted in (Wong, et al., 2014) showing that FOS-ELM should be more effective than BP in adaptive control applications. Therefore, the FOS-ELM approach may be suitable for the design of an adaptive AFR controller.

1.4 PROJECT OBJECTIVES

In view of the advantages of using adaptive control strategy for AFR control, the first project objective is to design a machine-learning-based adaptive AFR

controller. To achieve this, the AFR dynamics are studied in associated with some control theories. Some intelligent techniques, such as BP and FOS-ELM, are reviewed and the most suitable method will be carefully selected to implement the AFR controller.

The second objective is to verify the designed intelligent adaptive AFR controller through simulations. A simulated model is employed and the designed controller is set to control the simulated model and evaluate its control performance. To demonstrate the effectiveness of the designed controller, the results will be compared with other adaptive AFR control algorithms.

Finally, the third objective of this project is to implement the designed intelligent adaptive AFR controller on a real engine. In order to verify that the designed adaptive controller is feasible for real-time AFR control, it must be tested experimentally and compared with the engine built-in AFR controller. This is the most challenging task because automotive engine is a complicated system that consists of many sensors, wires and components. The key point for finishing this objective is to truly understand the build-up of the engine, including the signal types of the sensors, the properties of the engine components, and the connections between each engine component. Moreover, data acquisition system and data processing program, such as National

Instrument devices, LabVIEW program and MATLAB program, must be well studied before they are used. Once the whole data acquisition system is set up, the adaptive controller can be implemented and evaluated experimentally.

CHAPTER 2: REVIEW OF INTELLIGENT TECHNIQUES FOR ADAPTIVE CONTROL

This chapter briefly reviews some of the existing intelligent techniques suitable for the design of adaptive controller, and determines the most suitable one to be used in this project. The core idea of adaptive control is to adapt to the controlled system by self-adjusting the parameters of the controller online based on the feedback of the system so that good control can be achieved. In the field of artificial intelligence, there exist some techniques that can realize this core idea of adaptive control. These include (i) back-propagation (BP) method used in traditional ANN, which estimates that system parameters based on *gradient descent*, and (ii) online sequential extreme learning machine (OS-ELM) and its improved variant (iii) fully online sequential-extreme learning machine (FOS-ELM), which estimation is based on *recursive least-squares* (RLS).

2.1 BACK PROPAGATION

Back propagation (BP) is a technique proposed back to thirty years ago for learning ANN (Rumelhart, et al., 1986). The concept of this technique is to calculate the gradient of the error function with respect to the network parameters, and update the corresponding parameters in order to optimize the

error function. The error function is usually defined as the difference between the system output and the network output. Since the introduction of this powerful algorithm, BP has become the most common learning method for neural networks (Haykin, 1999) and today many researchers are still using this popular method for different intelligent applications.

For a network with one hidden layer and L hidden nodes, the output function is

$$y = f(\mathbf{x}) = \sum_{k=1}^{L} \beta_k \, h_k(\mathbf{x}) = \mathbf{h}(\mathbf{x})\boldsymbol{\beta} \tag{2.1}$$

where $\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_L(\mathbf{x})]$ is the output vector of the hidden layer feature mapping with respect to the input $\mathbf{x}$, and $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_L]^{\mathrm{T}}$ is the vector of output weights between the hidden layer and the output nodes.

Now, given a training dataset *S* with *N* samples, to approximate the data characteristics, BP determines the optimal weights in the network by first comparing the network output and the desired output (target output in *S*) with the following cost function (or error function):

$$\text{Minimize: } E = \frac{1}{2}\sum_{i}^{N} (y_i - t_i)^2 \tag{2.2}$$

where t_i is the *i*th desired output corresponding to the *i*th input-output pair.

The way how BP adjusts the network parameters is by evaluating the partial

derivative of the cost function with respect to the weights:

$$\nabla E = \left[\frac{\partial E}{\partial \beta_1}, \frac{\partial E}{\partial \beta_2}, \ldots, \frac{\partial E}{\partial \beta_k}\right]. \tag{2.3}$$

This derivate, which is also called the gradient of error, can then be used to update the weights, by the following delta rule as used in gradient descent learning:

$$\boldsymbol{\beta} = \boldsymbol{\beta} - \eta \nabla E, \tag{2.4}$$

where η is the user-specified learning rate which affects the speed and quality of the updating weights. If η is small enough, the learned network would be more accurate, but a longer training time is required. A larger η can significantly decrease the training time of the network, but the generalization performance would become worse.

Since the BP algorithm aims to minimize the difference between the desired output and the network output by adjusting the network parameters through an iterative manner, it becomes very suitable for adaptive control. Therefore, with the BP algorithm, an adaptive controller has been proposed by Chen (1990) twenty years ago. Although this method has been used for twenty years, and the neural controllers were shown to perform better than traditional proportional-integral-derivative controllers in some recent studies (Peng, et al., 2011, Yuan, et al., 2010), there is still one well-known critical drawback of this

algorithm. That is, BP is a gradient-decent based learning method which may easily converge to local minima (Rong, et al., 2013, Wong, et al., 2013). Therefore, it usually takes "more than required" steps for the controller to achieve satisfactorily performance. For instance, the simulation results by Chen (1990) showed that thousands of updating steps were needed before the controller could finally achieve the desired convergence. Moreover, in (Yuan, et al., 2010) and (Peng, et al., 2011), the controllers still take many steps to settle every time when the desired output is changed. These results indicate that the system dynamics cannot be globally approximated. Another disadvantageous property of BP is that, it updates the parameters in all the layers of the network, leading to a long processing time and hence a slow convergence speed.

2.2 ONLINE SEQUENTIAL EXTREME LEARNING MACHINE

Online sequential extreme learning machine (OS-ELM) is an online learning algorithm for single-hidden-layer feed-forward neural networks. It is originated from extreme learning machine (ELM), and can learn the network not only one-by-one but also chuck-by-chunk with fixed or varying chunk size (Liang, et al., 2006). It consists of two phases: initialization phase and sequential learning phase.

In the initialization phase, a base ELM network is trained using a small

chunk of initial training data. For the same network given in Eq. (2.1) and using the same training dataset *S* with *N* samples, ELM aims to minimize the norm of the output error:

$$\text{Minimize: } \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|^2 \tag{2.5}$$

where $\boldsymbol{T} = [t_1, t_2, \dots, t_N]^{\mathrm{T}}$ is the vector containing the desired output, and $\boldsymbol{H} = [\mathbf{h}(\mathbf{x}_1), \mathbf{h}(\mathbf{x}_2), \dots, \mathbf{h}(\mathbf{x}_N)]^{\mathrm{T}}$ is a $N \times L$ matrix used to present the hidden layer output and each row of $\boldsymbol{H}$ is a training sample after feature mapping.

Different from BP, ELM attempts to solve Eq. (2.5) using the least-squares (LS) method:

$$\boldsymbol{\beta} = \boldsymbol{H}^{\dagger}\boldsymbol{T} \tag{2.6}$$

where $\boldsymbol{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\boldsymbol{H}$. If $\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H}$ is non-singular, the orthogonal projection method can be used to calculate the pseudo-inverse of $\boldsymbol{H}$ and $\boldsymbol{\beta}$ can be re-written as:

$$\boldsymbol{\beta} = (\boldsymbol{H}^{\mathrm{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathrm{T}}\boldsymbol{T}. \tag{2.7}$$

Therefore, denoting a subscript of 0, the base ELM network trained in the initialization phase of OS-ELM can be represented by:

$$\boldsymbol{\beta}_0 = \boldsymbol{P}_0\boldsymbol{H}_0\boldsymbol{T}_0 \tag{2.8}$$

$$\boldsymbol{P}_0 = (\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0)^{-1}. \tag{2.9}$$

Then, in the sequential learning phase, when a new chunk of training data

arrives, the output weights are updated by:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \boldsymbol{P}_{k+1}\boldsymbol{H}_{k+1}^{\mathrm{T}}\left(\boldsymbol{T}_{k+1} - \boldsymbol{H}_{k+1}\boldsymbol{\beta}^{(k)}\right) \tag{2.10}$$

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_k - \boldsymbol{P}_k\boldsymbol{H}_{k+1}^{\mathrm{T}}(\boldsymbol{I} + \boldsymbol{H}_{k+1}\boldsymbol{P}_k\boldsymbol{H}_{k+1}^{\mathrm{T}})^{-1}\boldsymbol{H}_{k+1}\boldsymbol{P}_k \tag{2.11}$$

where $k+1$ indicates the $(k+1)$th arriving training data with k starting from zero, and $\boldsymbol{H}_{k+1}$ is the hidden layer output for the $(k+1)$th arriving training data.

The resulting parameter learning technique in OS-ELM is similar to recursive least-squares (RLS), which can solve the local minima problem and long training time issue of BP. However, there are still some factors limiting the direct application of OS-ELM to adaptive control. One major problem in OS-ELM is that, if the term $\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0$ is singular, then Eq. (2.9) is unsolvable. Therefore, to avoid the singular problem, OS-ELM restricts that the initial training dataset should have at least L (hidden node number) distinct samples (Liang, et al., 2006). Yet in many practical cases, a chunk of representative initial data is usually difficult to obtain in advance (Wong, et al., 2014). Moreover, the ill-posed problems of OS-ELM could also degrade the performance to an unacceptable level (Huynh, et al., 2011).

2.3 FULLY ONLINE SEQUENTIAL EXTREME LEARNING MACHINE

Fully online sequential extreme learning machine (FOS-ELM) is an

improved version of OS-ELM proposed by Wong, et al. (2014). It introduces a regularization factor to the network to overcome the singular problem, and derives a novel initialization way so that no initial training data is necessary for the construction of a base ELM network.

Firstly, in order to resolve the hidden-node-number-constraint of OS-ELM, a regularization term C can be added to Eq. (2.9):

$$\boldsymbol{P}_0 = (\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0 + C\boldsymbol{I})^{-1}. \tag{2.12}$$

According to the ridge regression theory, adding a small positive value into the diagonal of $\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0$ can avoid singular problem when the number of initial training data is less than the hidden nodes number. The main theory behind this is that, Eq. (2.9) is the solution to Eq. (2.5), which is only based on empirical risk minimization principle, while Eq. (2.12) is the solution of the following:

$$\text{Minimize: } \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|^2 \text{ and } \|\boldsymbol{\beta}\|^2 \tag{2.13}$$

in which the structural risk is also considered. The regularization term C in Eq. (2.12) mainly controls the trade-off of the minimization between the empirical risk and the structural risk in Eq. (2.13). Therefore, the resulting solution tends to have better and more stable generalization performance, as verified in (Bartlett, 1998, Deng, et al., 2009, Huang, et al., 2010, Huang, et al., 2012).

Now, considering an initial training dataset $S_0 = \{(x_i, t_i)|i = 1, \dots, N_0\}$ with a corresponding hidden layer output matrix $\boldsymbol{H}_0$, using Eqs. (2.8) and (2.12), the output weights $\boldsymbol{\beta}^0$ can be calculated as:

$$\boldsymbol{\beta}^0 = \boldsymbol{K}_0^{-1}\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{T}_0 \tag{2.14}$$

$$\boldsymbol{K}_0 = \boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0 + \lambda\boldsymbol{I} \tag{2.15}$$

where $\boldsymbol{T}_0 = [t_1 \dots t_{N_0}]^{\mathrm{T}}$ and $\boldsymbol{K}_0^{-1} = \boldsymbol{P}_0$.

Suppose now a new training dataset $S_1 = \{(x_j, t_j)|j = N_0 + 1, \dots, N_0 + N_1\}$ arrives with a corresponding hidden layer output matrix $\boldsymbol{H}_1$. By considering both training dataset S_0 and S_1, using Eqs. (2.8) and (2.12) again, the output weights $\boldsymbol{\beta}^1$ should be obtained as:

$$\boldsymbol{\beta}^1 = \boldsymbol{K}_1^{-1}\begin{bmatrix}\boldsymbol{H}_0\\ \boldsymbol{H}_1\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{T}_0\\ \boldsymbol{T}_1\end{bmatrix} \tag{2.16}$$

$$\boldsymbol{K}_1 = \boldsymbol{K}_0 + \boldsymbol{H}_1^{\mathrm{T}}\boldsymbol{H}_1 \tag{2.17}$$

where $\boldsymbol{T}_1 = [t_{N_0+1} \dots t_{N_0+N_1}]^{\mathrm{T}}$. Now expanding the last two terms on the right-hand side of Eq. (2.16):

$$\begin{bmatrix}\boldsymbol{H}_0\\ \boldsymbol{H}_1\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{T}_0\\ \boldsymbol{T}_1\end{bmatrix} = \boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{T}_0 + \boldsymbol{H}_1^{\mathrm{T}}\boldsymbol{T}_1 = \boldsymbol{K}_0\boldsymbol{\beta}^0 + \boldsymbol{H}_1^{\mathrm{T}}\boldsymbol{T}_1. \tag{2.18}$$

Then, combining Eqs. (2.16), (2.17) and (2.18), $\boldsymbol{\beta}^1$ is obtained as:

$$\boldsymbol{\beta}^1 = \boldsymbol{K}_1^{-1}\begin{bmatrix}\boldsymbol{H}_0\\ \boldsymbol{H}_1\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{T}_0\\ \boldsymbol{T}_1\end{bmatrix} = \boldsymbol{\beta}^0 + (\boldsymbol{K}_0 + \boldsymbol{H}_1^{\mathrm{T}}\boldsymbol{H}_1)^{-1}\boldsymbol{H}_1^{\mathrm{T}}(\boldsymbol{T}_1 - \boldsymbol{H}_1\boldsymbol{\beta}^0). \tag{2.19}$$

Now, considering only S_1, $\boldsymbol{\beta}^1$ can be obtained as:

$$\boldsymbol{\beta}^1 = (\boldsymbol{H}_1^{\mathrm{T}}\boldsymbol{H}_1 + C\boldsymbol{I})^{-1}\boldsymbol{H}_1^{\mathrm{T}}\boldsymbol{T}_1. \quad (2.20)$$

Comparing (2.19) and (2.20), it is obvious that (2.20) can be obtained from (2.19) if and only if $\boldsymbol{\beta}^0 = \mathbf{0}$ and $\boldsymbol{K}_0 = C\boldsymbol{I}$. Therefore, by initializing $\boldsymbol{\beta}^0 = \mathbf{0}$ and $\boldsymbol{K}_0 = C\boldsymbol{I}$, the initial training datasets S_0 can be omitted, while a model for S_1 can still be constructed. For the sake of better understanding, the algorithm of FOS-ELM is summarized as below (Wong, et al., 2014):

- **STEP 1:** Assign random values for input weights, and set $\boldsymbol{\beta}^0 = \mathbf{0}$ and $\boldsymbol{P}_0 = (\lambda\boldsymbol{I})^{-1}$.
- **STEP 2:** For the $(k+1)$th arriving training data, calculate the hidden layer output matrix $\boldsymbol{H}_{k+1}$, and then update the output weights $\boldsymbol{\beta}^{(k+1)}$ using (2.10) and (2.11).

Since the batch training in the initialization phase of OS-ELM is automatically integrated in FOS-ELM. FOS-ELM becomes a fully online sequential learning algorithm (Wong, et al., 2014), which is very suitable for adaptive control applications.

2.4 SUMMARY OF INTELLIGENT TECHNIQUES FOR ADAPTIVE CONTROL

In this chapter, three intelligent techniques suitable for adaptive control is reviewed. A comparison among the three techniques is provided in Table 2.1.

Table 2.1 Comparison among BP, OS-ELM and FOS-ELM

	BP	OS-ELM	FOS-ELM
Cost function	$\|\|\boldsymbol{H\beta}-\boldsymbol{T}\|\|^2$	$\|\|\boldsymbol{H\beta}-\boldsymbol{T}\|\|^2$	$\|\|\boldsymbol{H\beta}-\boldsymbol{T}\|\|^2$ and $\|\|\boldsymbol{\beta}\|\|^2$
Parameters	L and η	L	L and C
Training method	Gradient descent	Initial offline: LS Online: RLS	RLS
Initial sample number	Not necessary	$> L$	Not necessary
Overfitting risk	Yes	Yes	No
Learning speed	Slow	Fast	Fast

As shown in Table 2.1, FOS-ELM has overcome the problems suffered from BP and OS-ELM. For instance, no initial training data is required in FOS-ELM as compared to OS-ELM, and the singular problem is overcome. The overfitting risk and slow learning speed of BP are also resolved in FOS-ELM. In addition, the hidden node parameters in BP are updated iteratively, while those in FOS-ELM are initialized randomly and remain unchanged. Thus, the local

minima problem of BP is also alleviated in FOS-ELM. As a result, FOS-ELM is selected as the intelligent technique for the design of the adaptive air-fuel ratio controller in this project.

CHAPTER 3: DESIGN OF INTELLIGENT ADAPTIVE AIR-FUEL RATIO CONTROLLER

In Chapter 2, some techniques suitable for adaptive control is reviewed and compared. From the comparison, FOS-ELM is selected for the design of the intelligent adaptive controller. This chapter mainly discusses the design of the controller for air-fuel ratio control using FOS-ELM. The details of the adaptive AFR controller is provided, followed by some simulations of the FOS-ELM based adaptive control algorithm to verify the design of the intelligent adaptive controller. A comparison with the BP based control algorithm is also given to evaluate the effectiveness of the controller.

3.1 CONTROLLER DESIGN

Theoretically, the dynamics of AFR (or λ) can be presented by a discrete nonlinear function:

$$\lambda_{k+1} = f(\lambda_k, \lambda_{k-1}, \dots, \lambda_{k-n+1}, u_k, u_{k-1}, \dots, u_{k-n+1}) \tag{3.1}$$

where n is the order of the system, k is the time step, and u is the control signal.

Although AFR is governed by the amount of both the intake air and the injected fuel, usually only the amount of injected fuel is controlled as the intake

air is difficult to control precisely. Therefore, the control signal u in Eq. (3.1) is the amount of fuel injected to the engine. Then, to determine the control law for achieving the target λ, the inverse of Eq. (3.1) is required:

$$u_k = f^{-1}\left(\lambda_{d_{k+1}} | \lambda_k, \lambda_{k-1}, \dots, \lambda_{k-n+1}, u_{k-1}, \dots, u_{k-n+1}\right) \tag{3.2}$$

where λ_d is the desired AFR.

However, since the dynamics of AFR may involve many uncertainties due to the complex nature of engine (Wong, et al., 2013), it is practically difficult to derive the inverse of Eq. (3.1) explicitly. Therefore, in this project, the AFR dynamics is assumed to be a discrete approximated model in which the control appears linearly (Chen, 1990):

$$\begin{aligned} \lambda_{k+1} = {} & g(\lambda_k, \lambda_{k-1}, \dots, \lambda_{k-n+1}, u_{k-1}, \dots, u_{k-n+1}) \\ & + \varphi(\lambda_k, \lambda_{k-1}, \dots, \lambda_{k-n+1}, u_{k-1}, \dots, u_{k-n+1}) u_k \end{aligned} \tag{3.3}$$

where $\varphi(\cdot)$ must be a nonzero function.

Obviously, if both $g(\cdot)$ and $\varphi(\cdot)$ in Eq. (3.3) are known, the following control law can be used to exactly track the desired AFR:

$$u_k = \frac{\lambda_{d_{k+1}} - g(\cdot)}{\varphi(\cdot)}. \tag{3.4}$$

Different from PID control in which the control signal is linear to the error function, the functions $g(\cdot)$ and $\varphi(\cdot)$ here can be nonlinear. However, since

$g(\cdot)$ and $\varphi(\cdot)$ cannot be obtained exactly, approximations of these two functions are used, denoted as $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$, and the control law should become:

$$u_k = \frac{\lambda_{d_{k+1}} - \hat{g}(\cdot)}{\hat{\varphi}(\cdot)}. \tag{3.5}$$

Now, defining the tracking error between the target AFR and the actual AFR as:

$$e_{k+1} = \lambda_{k+1} - \lambda_{d_{k+1}}, \tag{3.6}$$

and substituting with Eqs. (3.3) and (3.5), it is obtained that:

$$e_{k+1} = g(\cdot) + \varphi(\cdot)\frac{\lambda_{d_{k+1}} - \hat{g}(\cdot)}{\hat{\varphi}(\cdot)} - \lambda_{d_{k+1}}. \tag{3.7}$$

It can be learnt from Eq. (3.7) that, if $g(\cdot) - \hat{g}(\cdot) \to 0$ and $\varphi(\cdot) - \hat{\varphi}(\cdot) \to 0$, then the tracking error $e_{k+1} \to 0$. Therefore, the purpose of the intelligent adaptive AFR controller in this project is to adjust the parameters of $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$ based on the actual AFR feedback from the controlled system so that the unknown $g(\cdot)$ and $\varphi(\cdot)$ can be approximated and the desired target can be tracked.

As discussed in Chapter 2, FOS-ELM has the ability to estimate the parameters online, it can be used to adaptively update the parameters of $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$ based on the different between the actual AFR feedback and the

approximated AFR:

$$\hat{e}_{k+1} = \lambda_{k+1} - \hat{\lambda}_{k+1}, \tag{3.8}$$

where $\hat{\lambda}_{k+1} = \hat{g}(\cdot) + \hat{\varphi}(\cdot)u_k$ is the approximated AFR.

However, the approximation error $\hat{e}_{k+1}$ in Eq. (3.8) is the error accumulated from both functions $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$, and it is again difficult to decompose the $\hat{e}_{k+1}$ for each function. To deal with this situation, the approximated AFR can be re-formulated as the following form:

$$\hat{\lambda}_{k+1} = \hat{g}(\cdot) + \hat{\varphi}(\cdot)u_k = \mathbf{h}_g(\cdot)\boldsymbol{\beta}_g + \mathbf{h}_\varphi(\cdot)\boldsymbol{\beta}_\varphi u_k = \begin{bmatrix} \mathbf{h}_g(\cdot) \\ \mathbf{h}_\varphi(\cdot)u_k \end{bmatrix} [\boldsymbol{\beta}_g \quad \boldsymbol{\beta}_\varphi]. \tag{3.9}$$

It is noticed that if denoting $\widehat{\boldsymbol{H}} = \begin{bmatrix} \mathbf{h}_g(\cdot) \\ \mathbf{h}_\varphi(\cdot)u_k \end{bmatrix}$ and $\widehat{\boldsymbol{\beta}} = [\boldsymbol{\beta}_g \quad \boldsymbol{\beta}_\varphi]$, the resulting form becomes only a single-hidden-layer network. Therefore, by updating the parameters $\widehat{\boldsymbol{\beta}}$ in this only one network, both $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$ can be updated simultaneously. Then, the adaptive law of the FOS-ELM controller becomes:

$$\widehat{\boldsymbol{\beta}}^{(k+1)} = \widehat{\boldsymbol{\beta}}^{(k)} + \widehat{\boldsymbol{P}}_{k+1}\widehat{\boldsymbol{H}}_{k+1}^{\mathrm{T}}\hat{e}_{k+1} \tag{3.10}$$

$$\widehat{\boldsymbol{P}}_{k+1} = \widehat{\boldsymbol{P}}_k - \widehat{\boldsymbol{P}}_k\widehat{\boldsymbol{H}}_{k+1}^{\mathrm{T}}\left(\boldsymbol{I} + \widehat{\boldsymbol{H}}_{k+1}\widehat{\boldsymbol{P}}_{\boldsymbol{k}}\widehat{\boldsymbol{H}}_{k+1}^{\mathrm{T}}\right)^{-1}\widehat{\boldsymbol{H}}_{k+1}\widehat{\boldsymbol{P}}_k. \tag{3.11}$$

With Eqs. (3.8), (3.10) and (3.11), $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$ are self-tuned until the target is reached. The control scheme is illustrated in Figure 3.1.

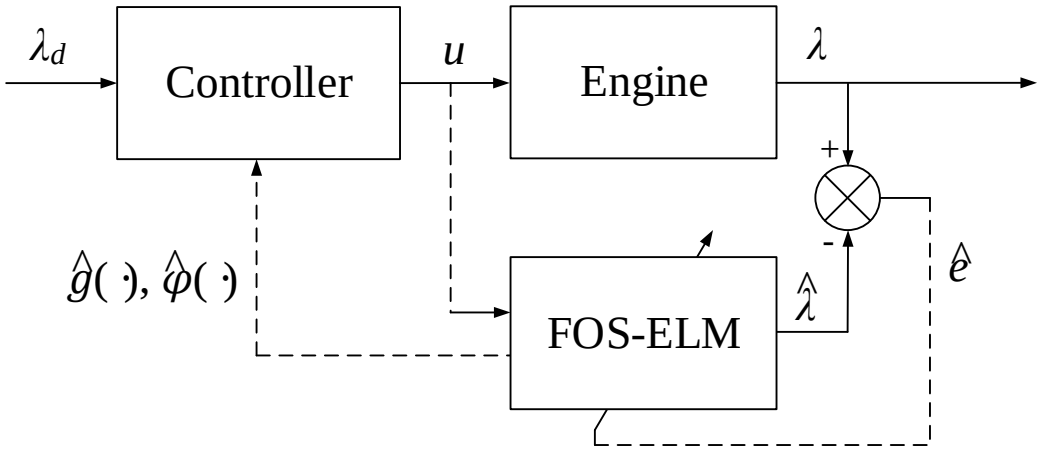


Figure 3.1 Adaptive AFR controller

3.2 SIMULATIONS ON ADAPTIVE AIR-FUEL RATIO CONTROL

In order to verify the designed adaptive air-fuel ratio controller, simulations on the AFR performance of an engine were conducted. A simulated engine model of a 465Q gasoline engine at 3500 rpm engine speed and 85 kPa manifold pressure was used, given as (Li, 2007):

$$\lambda_{k+1} = \frac{0.2 \sin \lambda_k + 3.5(9 - u_k)}{14.7}. \tag{3.12}$$

To demonstrate the effectiveness of FOS-ELM over BP for the adaptive air-fuel ratio controller, the simulations were also performed using BP and a comparison between FOS-ELM and BP was carried out. In the simulations, the hidden node number for both algorithms was set to 100, and the hidden layer activation function is radial basis function. The regularization factor C for FOS-ELM algorithm was set to 0.001, and the learning rate η for BP was set as

0.05. Both algorithms were implemented in MATLAB and all the simulations were executed in MATLAB on a PC with Intel Core i7-4790 CPU and 16GB RAM onboard.

3.2.1 Simulation I

In the first simulation, the step response of the controller was tested. The desired λ_d was set to drop from 1 to 0.95 at some point and then jump to 1.05 afterward, and finally return back to 1. The formulation of the reference output in this case is:

$$\lambda_d = \begin{cases} 0.95 & 50 \leq k < 150 \\ 1.05 & 150 \leq k < 250 \\ 1 & \text{otherwise.} \end{cases} \tag{3.13}$$

Based on this reference, the results of the two simulations of both algorithms are provided in Figures 3.2 & 3.3.

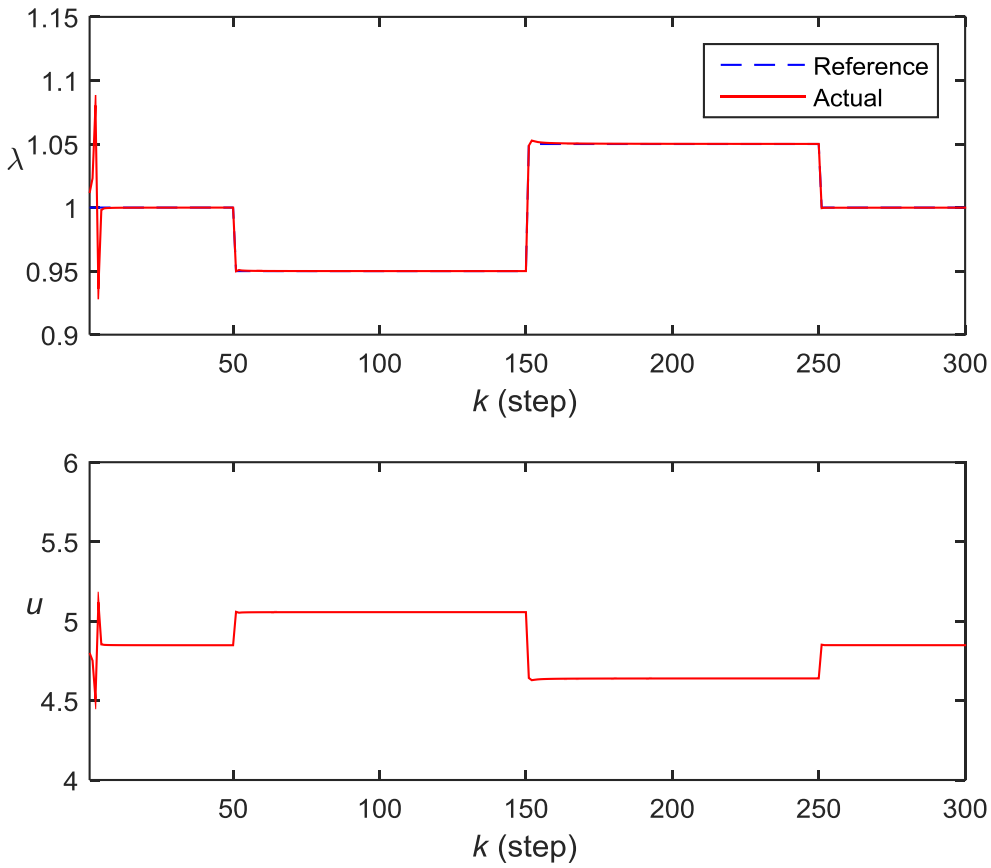


Figure 3.2 Control performance for FOS-ELM algorithm in Simulation I

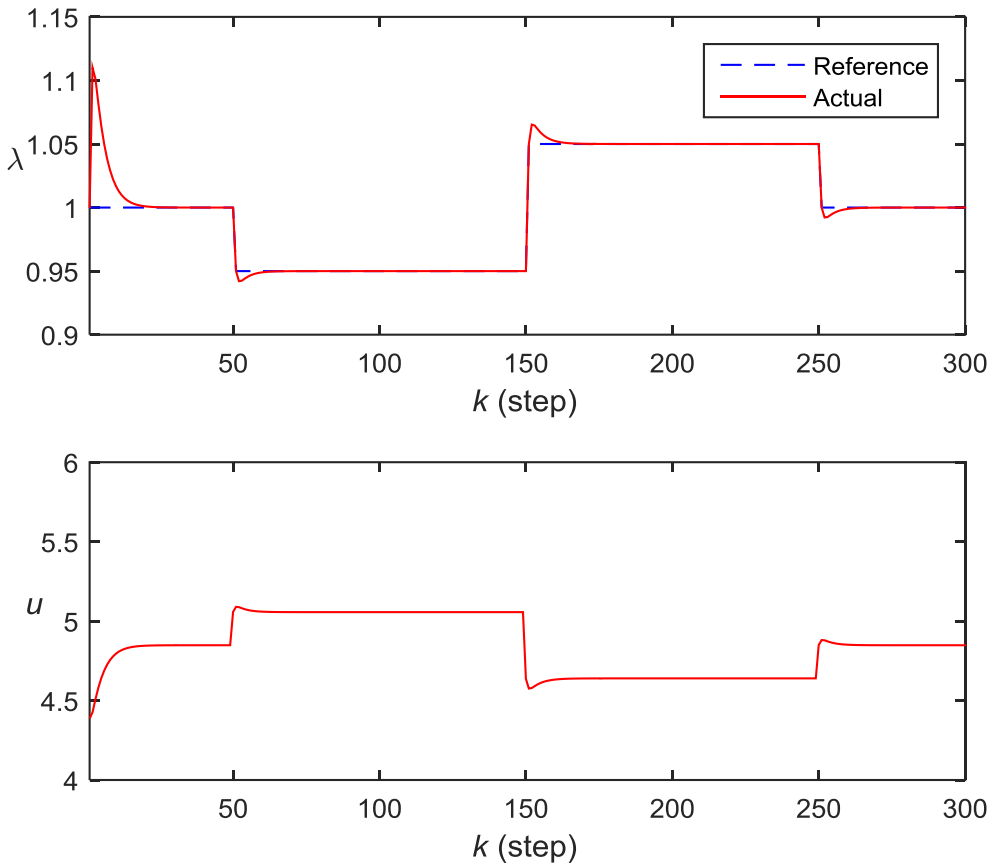


Figure 3.3 Control performance for BP algorithm in Simulation I

3.2.2 Simulation II

In the second simulation, the continuous tracking performance of the controller was evaluated. The desired λ_d was set to vary between 0.95 and 1.05, under a sine wave reference command. The formulation for in this case is:

$$\lambda_d = 0.03 \sin\left(\frac{2\pi k}{80}\right) + 0.02 \sin\left(\frac{2\pi k}{40}\right) + 1. \tag{3.14}$$

Based on this reference, the results of the two simulations of both algorithms are provided in Figures 3.4 & 3.5.

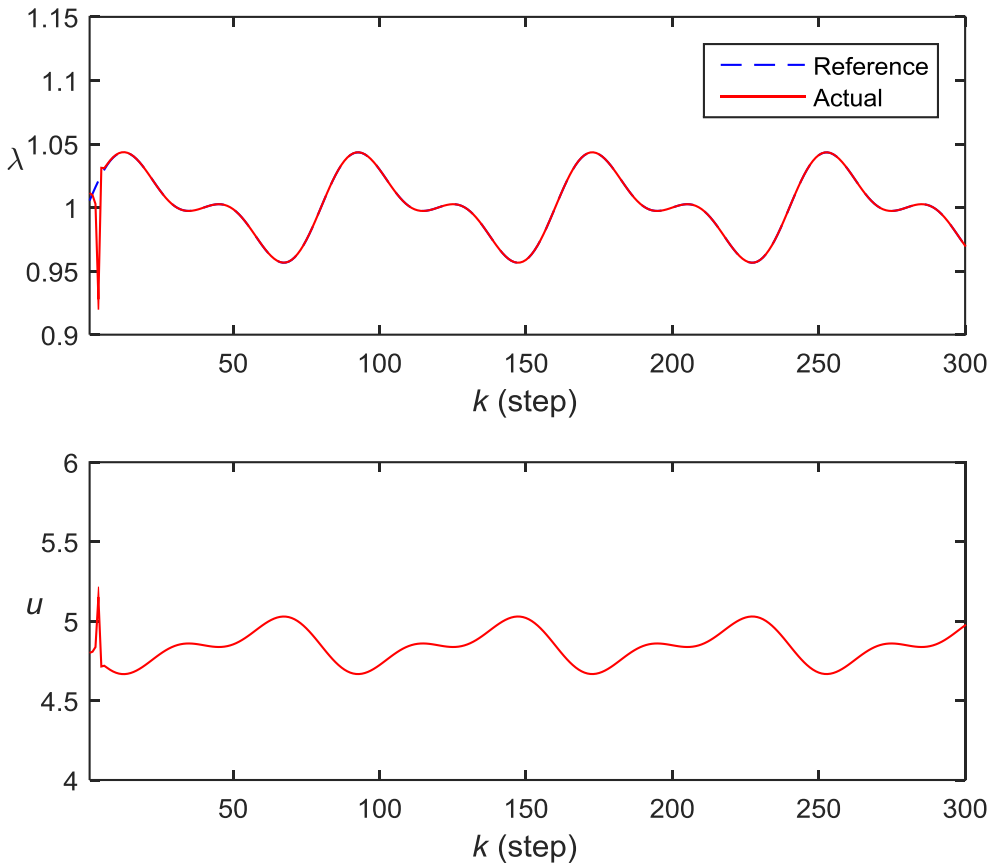


Figure 3.4 Control performance for FOS-ELM algorithm in Simulation II

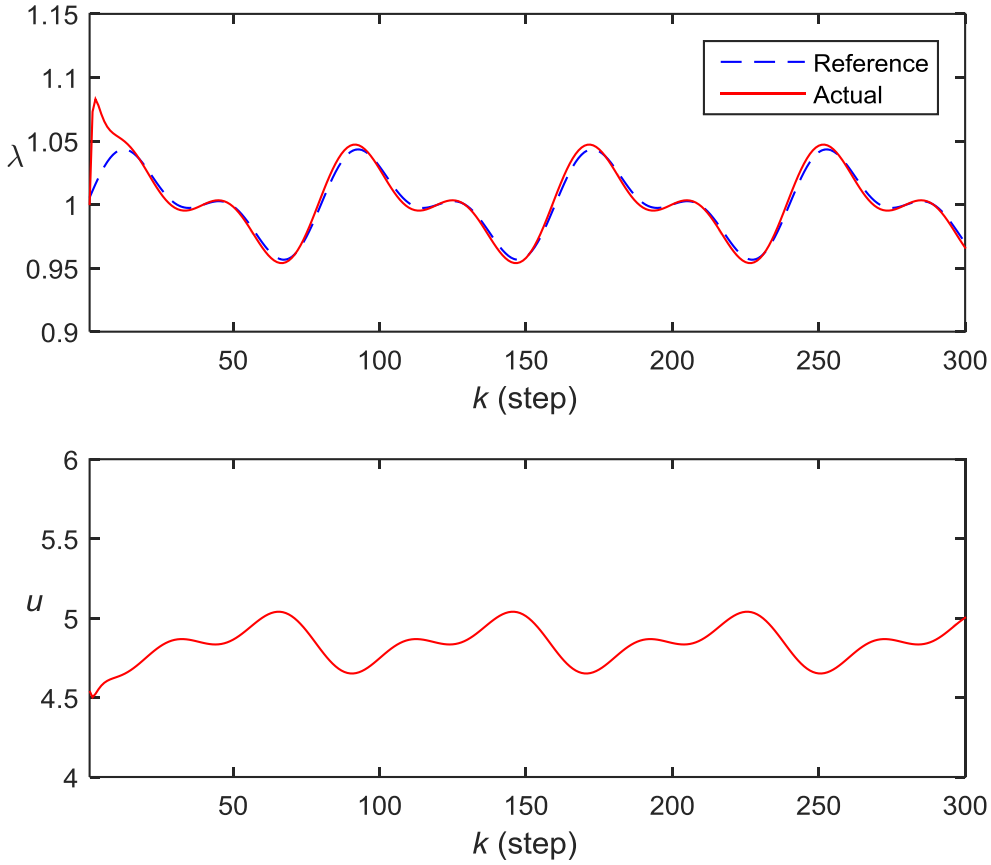


Figure 3.5 Control performance for BP algorithm in Simulation II

3.3 DISCUSSION OF SIMULATION RESULTS

The simulation results from Figures 3.2 to 3.5 show that, the control performance of FOS-ELM is much better than that of BP. Comparing Figures 3.2 and 3.3, it can be learnt that every time when the desired λ_d value changes, BP requires much more steps before reaching the equilibrium state. The overshoot of BP is also higher than that of FOS-ELM. This shows that BP always tends to 'forget' what it has learnt (Wong, et al., 2014). This is because BP updates all the parameters in both the hidden layer and output layer to

achieve the target output, which may easily suffer from local minima (i.e., optimal only for the most recent arrived data). Once it is stuck in local minima, the parameters need to be adjusted again when the target output changes (Wong, et al., 2014). In contrast, FOS-ELM tends to reach a global optimal (i.e., optimal for all the seen data) according to the ELM theory (Huang, et al., 2015). Hence, as shown in Figure 3.2, once the functions $\hat{g}(\cdot)$ and $\hat{\varphi}(\cdot)$ are globally learnt, the controller can directly adapt to the desired output no matter how it changes. The same results are also validated by Figures 3.4 and 3.5, in which FOS-ELM also achieves better tracking performance than BP.

From the simulations, the designed adaptive AFR controller using FOS-ELM is evaluated and the results verify that the designed controller is feasible and effective for AFR control. It not only has a very stable and fast learning and adapting speed, but also achieves a very attractive tracking performance. Therefore, the adaptive controller based on FOS-ELM was confidently selected for hardware implementation.

CHAPTER 4: IMPLEMENTATION OF INTELLIGENT ADAPTIVE AIR-FUEL RATIO CONTROLLER

In this chapter, details of the experimental implementation of the designed intelligent adaptive AFR controller are provided.

4.1 TEST BED

A Honda Integra DC5 car with a high performance electronic controlled, water-cooled, 4-cylinder gasoline engine was employed as the test car (Figure 4.1). Its specifications are provided in Table 4.1. A MoTeC M400 programmable electronic control unit (ECU) was used as the base controller to maintain the engine operation (Figure 4.2).

Table 4.1 Engine specifications

Base model	Honda K20A – Type R
Type	Water-cooled, four-stroke, DOHC i-VTEC
Cylinder arrangement	Inline four-cylinder, transverse
Bore and stroke	86 × 86 mm
Displacement	1998 cc
Compression ratio	11.5:1
Valve train	Chain drive, 16 valves
Maximum power	160 kW @ 8000 rpm
Maximum torque	196 N·m @ 7000 rpm

Figure 4.1 Honda K20A test engine

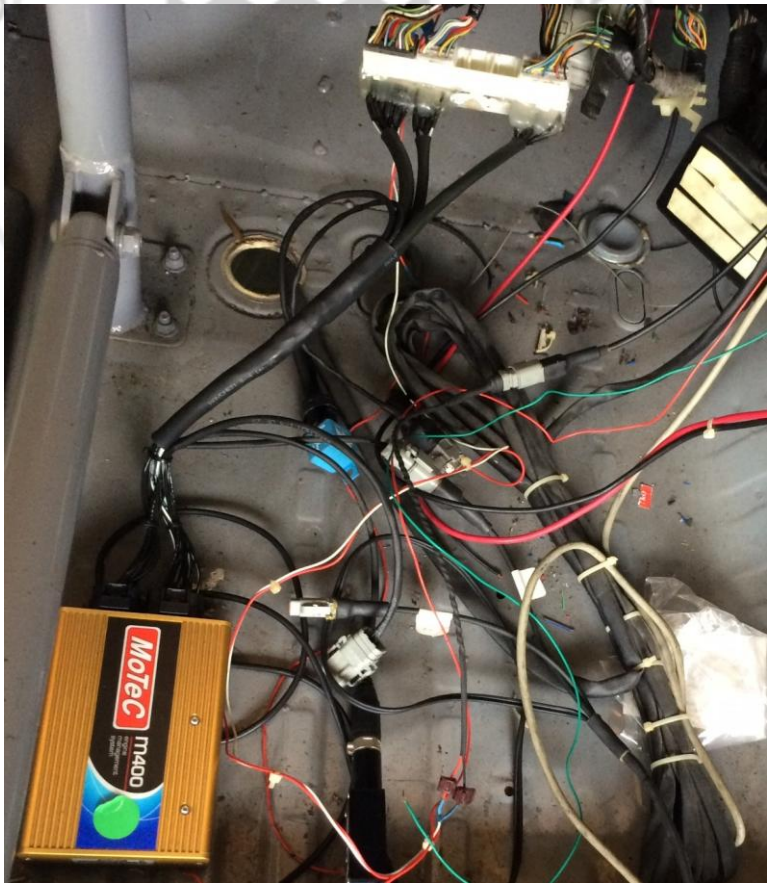


Figure 4.2 MoTeC M400 ECU for engine base control

4.2 CONTROL DEVICES & SOFTWARE

To implement the designed intelligent adaptive AFR controller, three National Instrument (NI) devices were used, as shown in Figure 4.3. An NI 9215 module was used for collecting analog signal, an NI 9263 module was used for sending analog signal, and an NI cDAQ-9178 chassis was used for signal transmission between the NI module and computer. The specifications of the two NI modules and the chassis are given in Tables 4.2 and 4.3.

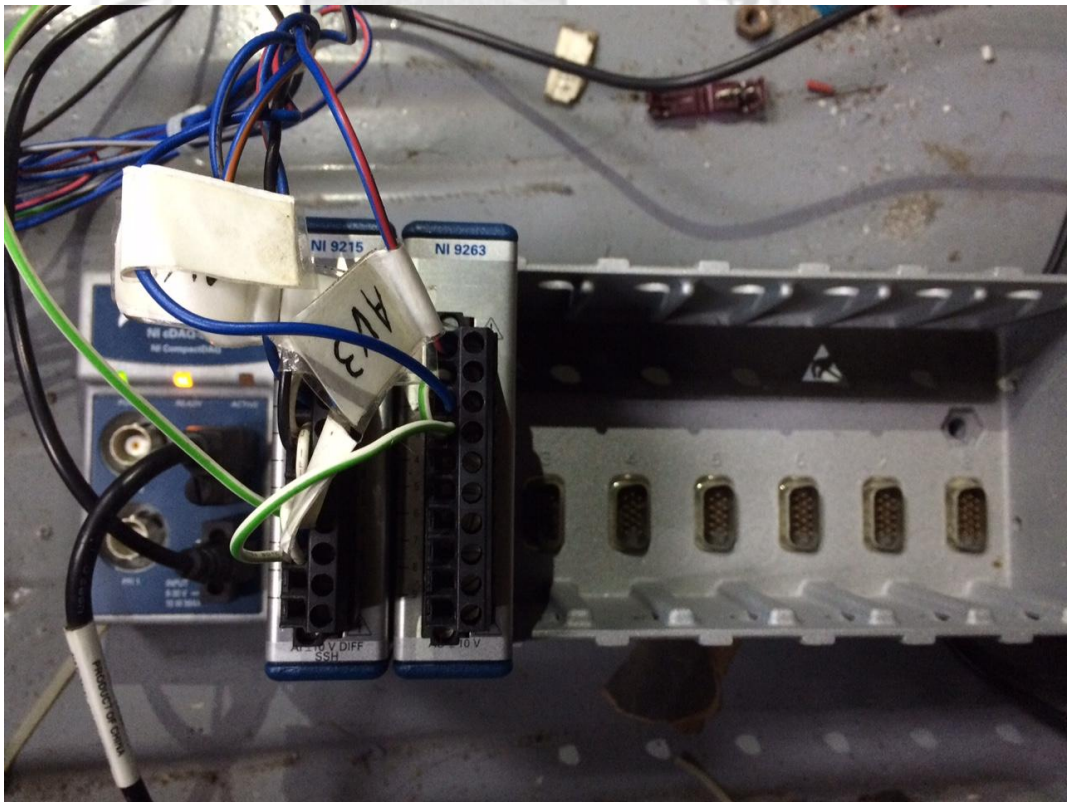


Figure 4.3 NI devices for controller implementation

Table 4.2 Specifications of NI modules

Model	NI 9215	NI 9263
Type	Simultaneous analog input	Analog output
Channels	4 differential	4
Signal range	±10 V	±10 V
Sample rate	100 kS/s/ch	100 kS/s/ch
Resolution	16-Bit	16-Bit

Table 4.3 Specifications of NI chassis

Model	NI cDAQ-9178
Slots	8
Counters	4
Number of simultaneous tasks	7
Number of AI timing engines	3
BNC triggers connections	Up to 1 MHz clocks and triggers

A LabVIEW 2012 software was installed on a computer, which was a graphical programming platform for the users to control the NI devices. In other words, the NI devices serve as the interface between the engine signal and the computer, and LabVIEW program serves as the interface between the computer and users. In LabVIEW, a MATLAB plugin is available such that the MATLAB script can be embedded directly. The connection between the devices is shown

in Figure 4.4. Details of the interface between the engine, NI devices, LabVIEW and MATLAB are provided in the following Section 4.3.

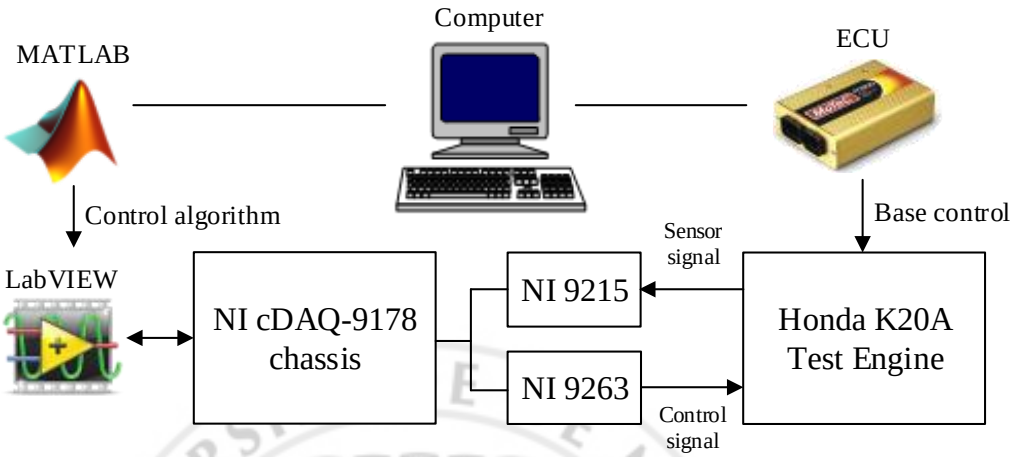


Figure 4.4 Connection between the devices and software

4.3 ENGINE – LABVIEW INTERFACE

There are many kinds of sensors installed at various part of engine to monitor the operation status of engine. In this project, the sensor signals from four engine parameters, namely (i) engine speed, (ii) throttle position, (iii) fuel injection time and (iv) λ value, were acquired for the controller. As it was difficult to find out all the corresponding sensors on the engines, the wiring between the ECU and the test engine was studied, as shown in Figure 4.5.

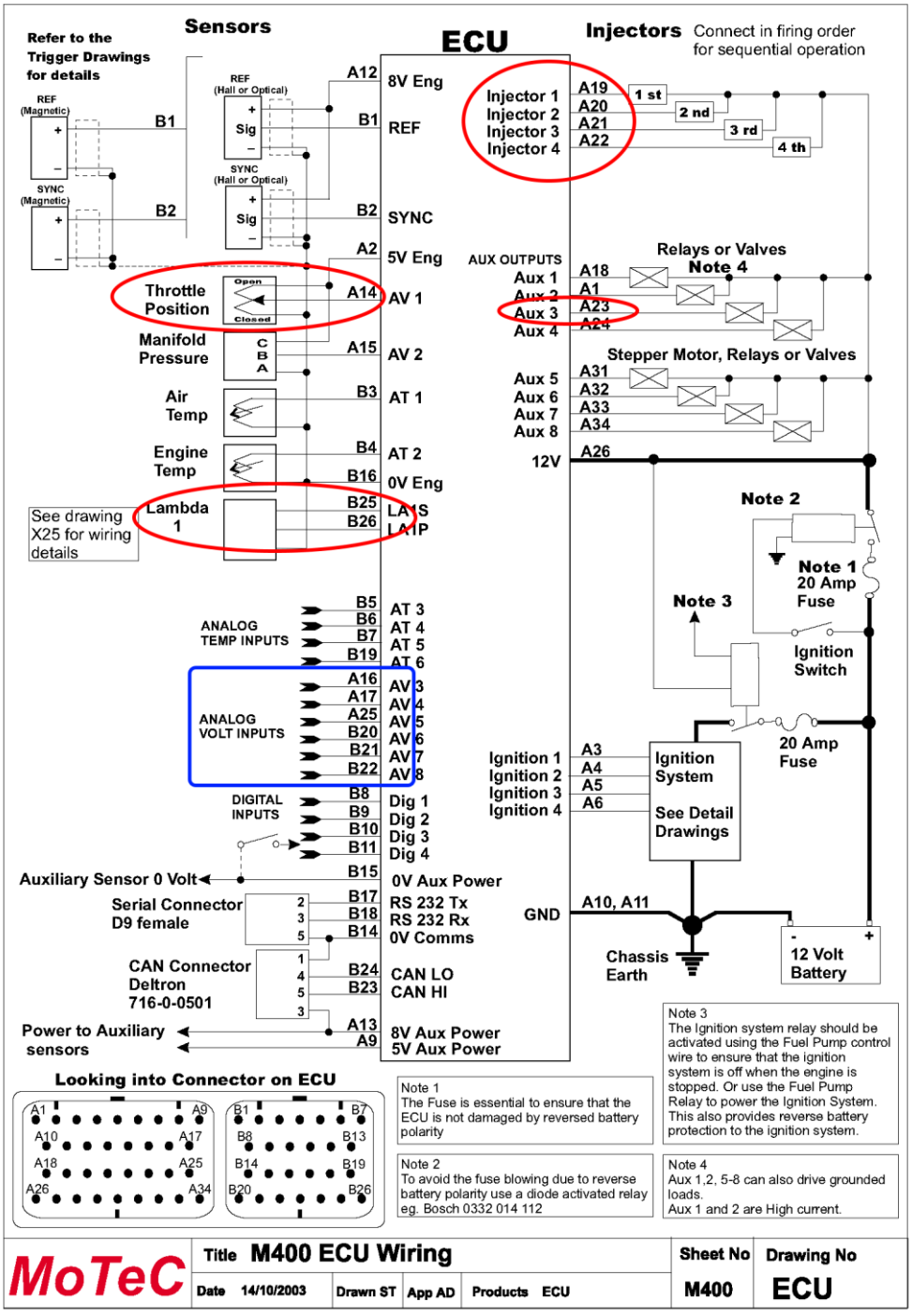


Figure 4.5 ECU wiring

It can be learnt from the ECU wiring that, the throttle position, fuel injection time and λ value can be acquired from pins A14, A19 to A22 and B25

respectively, while the engine speed can be acquired from pin A23 of the ECU. For controlling the fuel injection, instead of sending control signal directly to the engine fuel injectors, the control signal can be sent to the ECU and the ECU will convert the user given control signal into the control signal of the fuel injectors. This control signal could be sent to ECU using pin A16. As indicated in Figure 4.5, the connection pins in red circles are for collecting signal, and in blue rectangle are for sending signal. A summary of the connection setup is provided in Table 4.4.

Table 4.4 Specifications of NI modules

Parameter	Connection pin in ECU
Engine speed	A26
Throttle position	A14
fuel injection time	A19
λ	B25
Control signal	A16
Ground	B16

Before the implementation of the controller, the properties of the signals transmitted between the sensors and LabVIEW were studied, which is provided in the following sub-sections.

4.3.1 Engine speed

The sensor used for measuring engine speed of the test engine is a tachometer. The property of the signal is captured as shown in Figure 4.6. The raw signal shows that the tachometer generates pulses of which the frequencies are in accordance with the engine speed. Therefore, a "Pulse Measurement" module was used in the LabVIEW program to convert the signal into meaningful engine speed measurement.

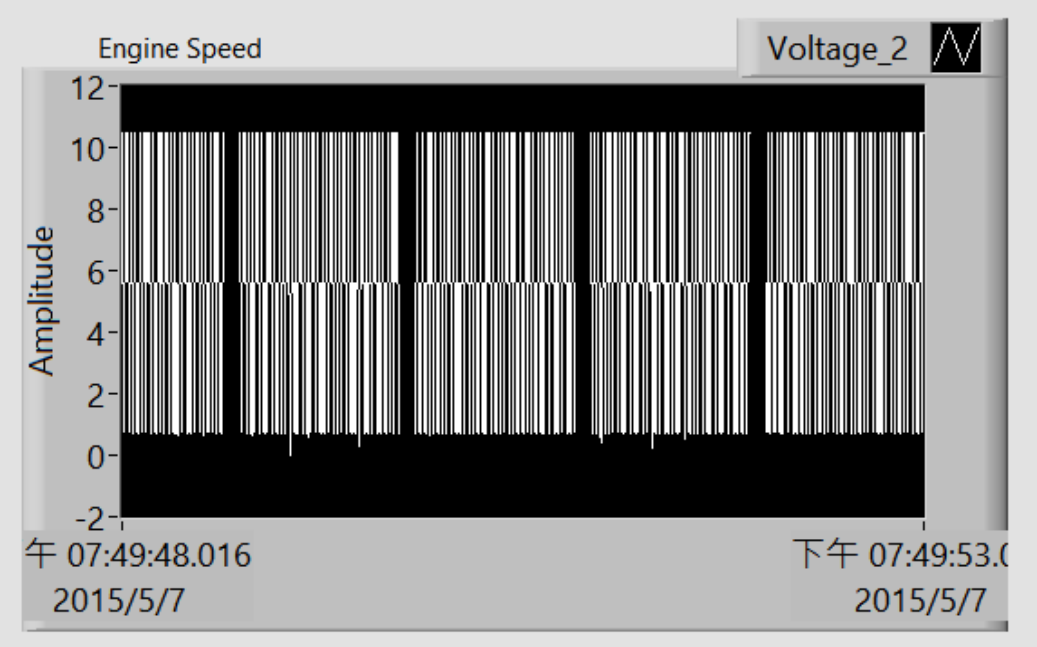


Figure 4.6 Raw signal of engine speed

4.3.2 Throttle position

The throttle position sensor of the test engine has a voltage range of 0.5V ~ 4.5V. The output voltage of the sensor is linearly proportional to the throttle

position in %. Thus, by scaling the analog voltage signal from the sensor, the throttle position can be acquired in LabVIEW.

4.3.3 Fuel injection time

The signal of the fuel injection time is the time within which the fuel injector is turned on, that is, the duration of fuel injection. To acquire this signal, a “Timing and Transition Measurement” module was used in LabVIEW.

4.3.4 λ value

The λ sensor sent out voltage signal that is same as the actual λ value, so it can be measured directly in LabVIEW.

4.3.5 Control signal

As mentioned, the signal for the control of fuel injection has to be sent to the ECU and let ECU convert the user provided signal into engine control signal. The user provided signal, which is the signal calculated by the controller, was sent to the ECU in analog with range of 1V ~ 9V. In the programmable ECU, a user channel was defined for the conversion of the given signal into engine control signal, as shown in Figure 4.7.

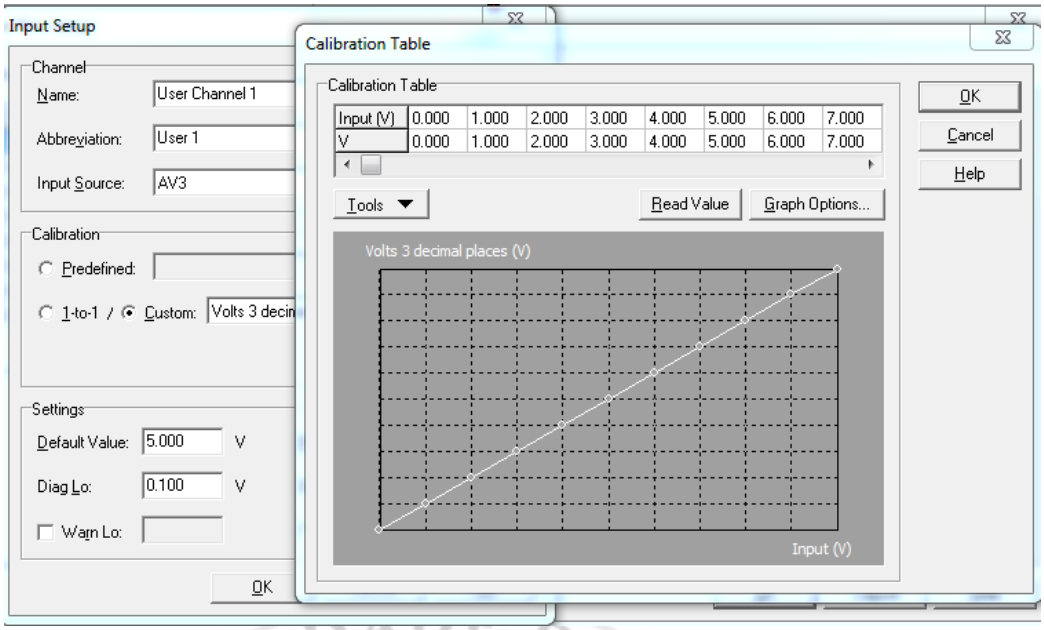


Figure 4.7 Defining channel in ECU for providing user control signal

4.4 LABVIEW – MATLAB INTERFACE

As the control algorithm has already been implemented in MATLAB, to allow the direct use of the MATLAB script, a "MATLAB Script Node" module was used. By embedding the algorithm in the LabVIEW program, the interface between the controller and the engine was completed. Figures 4.8 & 4.9 show the user interface (front panel) and the hardware interface (block diagram) of the LabVIEW program.

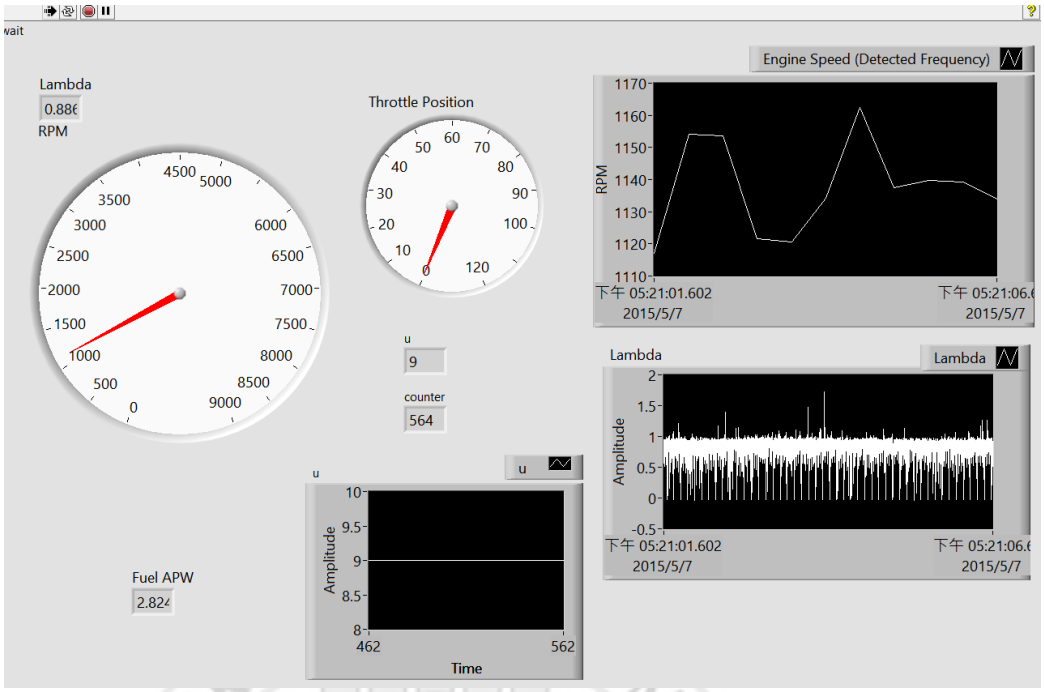


Figure 4.8 User interface of the LabVIEW program

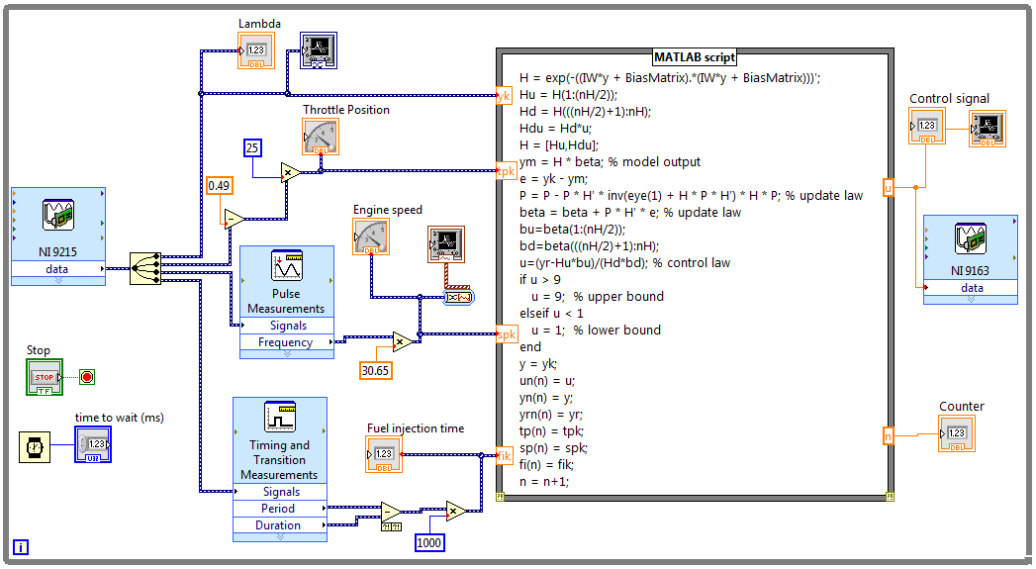


Figure 4.9 Interface between LabVIEW and NI devices

With this program and the NI devices, the designed controller was successfully implemented on the engine. Then, the effectiveness of the controller could be tested and evaluated through this hardware system.

CHAPTER 5: EXPERIMENTAL RESULTS

The experimental results of the designed intelligent adaptive AFR controller are presented in this chapter. Two tests were performed to evaluate the effectiveness of the controller. The parameters of the controller were same as those used in the simulation tests in Chapter 3. That is, the hidden node number is 100, and the regularization factor is 0.001. To demonstrate its effectiveness, the adaptive AFR controller was also compared with the engine built-in AFR controller.

5.1 TEST I – TRACKING ABILITY FOR CHANGE OF TARGET

Test I is designed to evaluate the tracking ability of the adaptive controller for change of target. In this test, the engine was started and warmed up until steady-state was reached, which was indicated by the intake air temperature and engine temperature. At the steady-state, the engine was kept at idle speed and the throttle position was fixed at 0%. The initial parameters of the controller, such as the number of hidden nodes and the regularization factor, were defined in the MATLAB workspace of the LabVIEW program first, and then the LabVIEW was run so that the intelligent adaptive AFR controller could be turned on. In the beginning, the desired λ value stayed at 1.00 (stoichiometric

value for maximum conversion efficiency of the three-way catalyst) for a short period of time. Then the desired λ value was changed to 1.05 (for best specific fuel consumption) for about 30*s* and finally changed back to 1.00 so that the tracking performance could be evaluated. The same procedure was then repeated again with the only difference that a desired λ value of 0.95 (for maximum engine power) was used instead of 1.05 in the 30*s* duration. For comparison purpose, both the designed AFR controller and the engine built-in AFR controller has undergone the same test procedure. The results of this test are given in Figures 5.1 to 5.4.

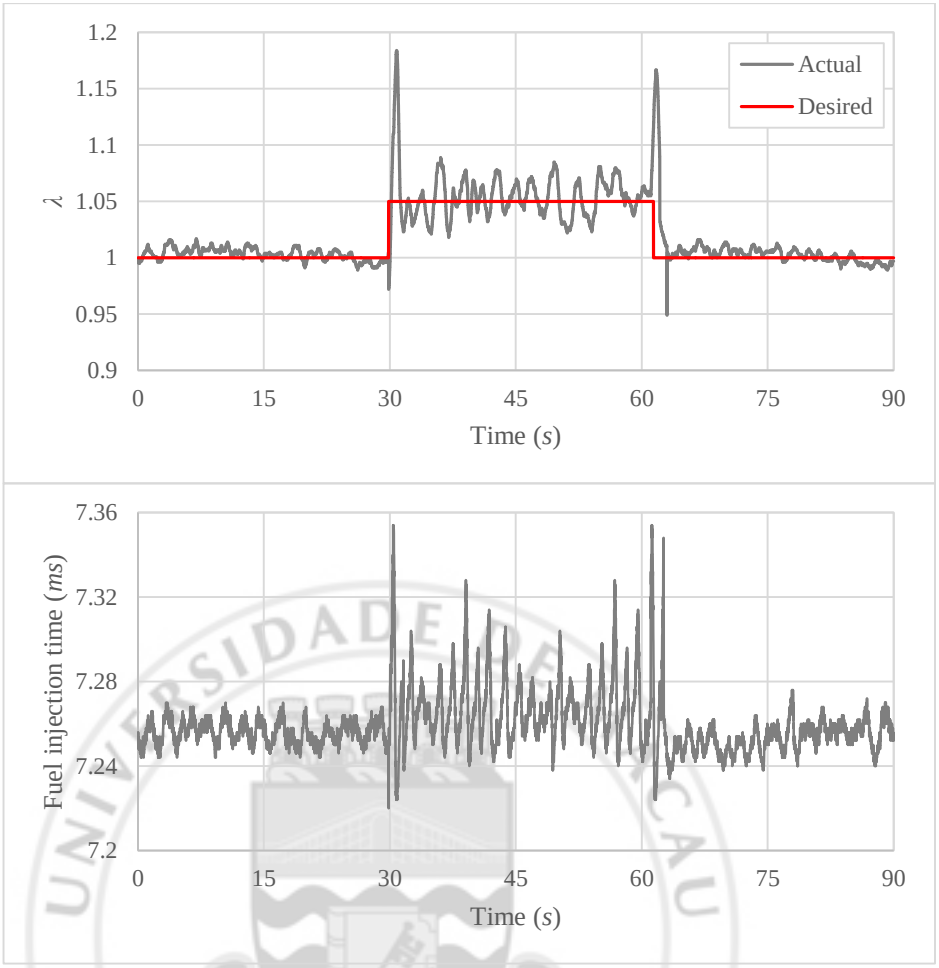


Figure 5.1 Performance of the designed controller for desired λ value varied among 1 and 1.05

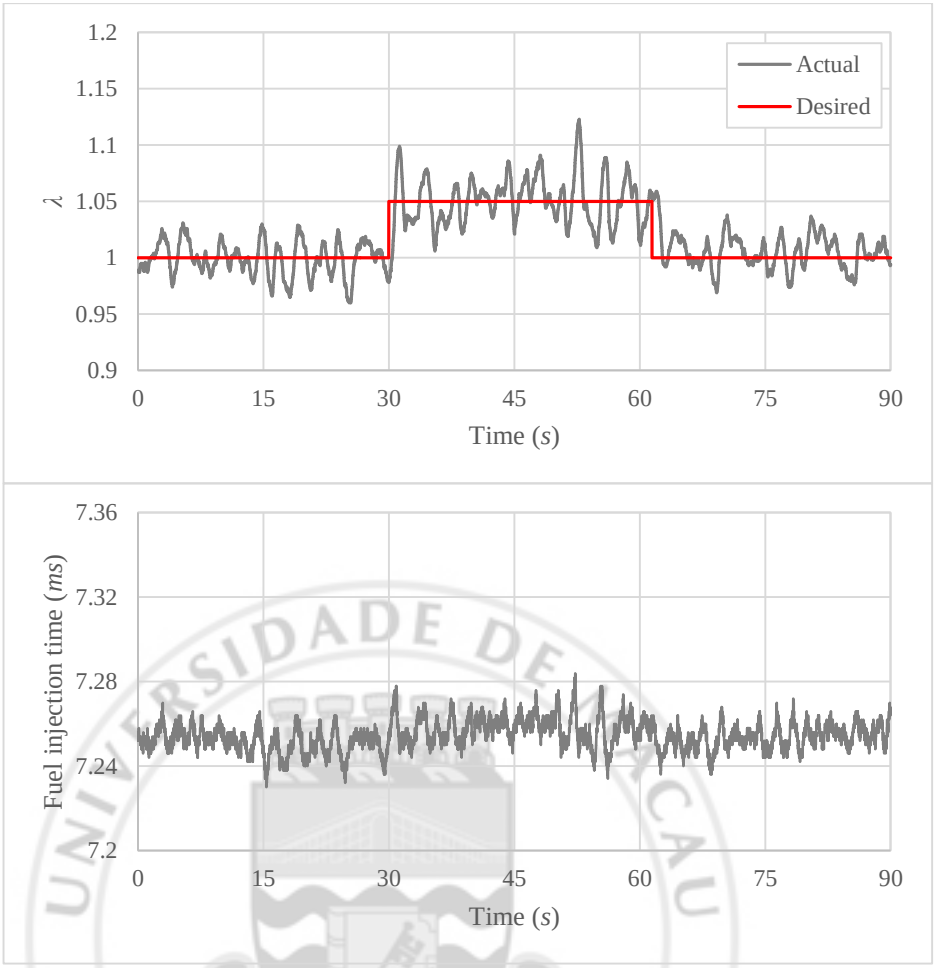


Figure 5.2 Performance of the engine built-in controller for desired λ value varied among 1 and 1.05

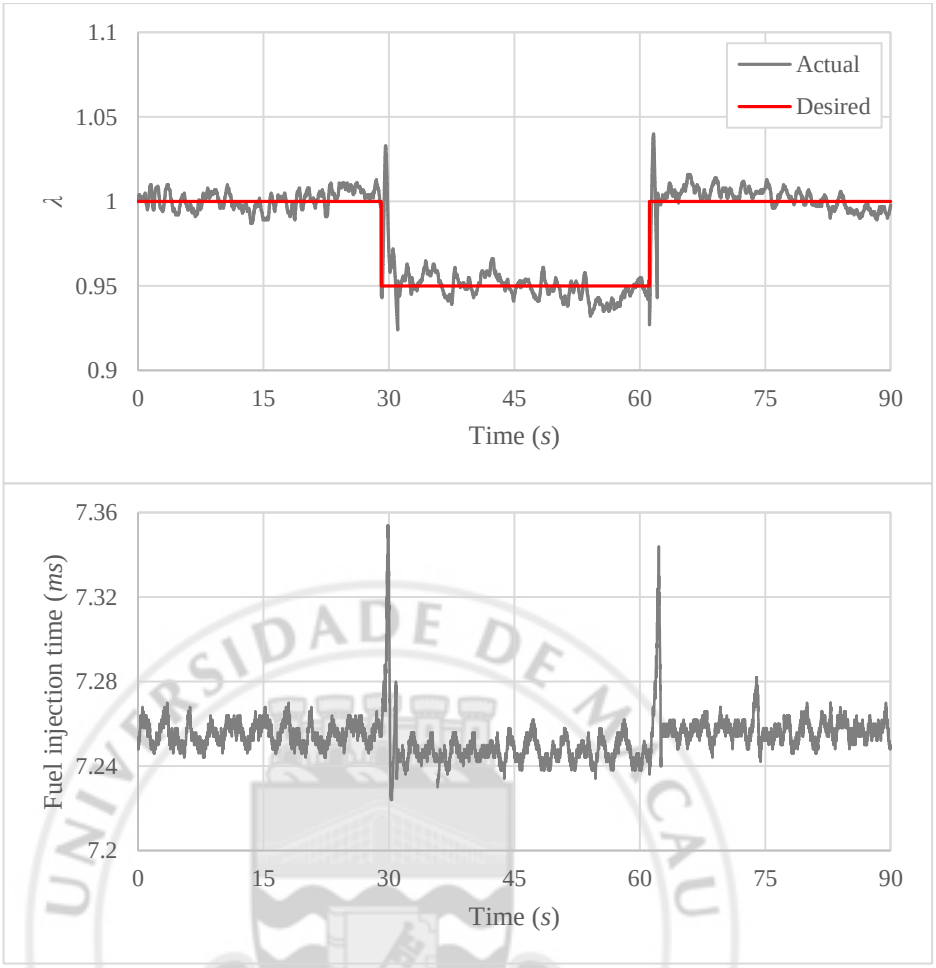


Figure 5.3 Performance of the designed controller for desired λ value varied among 1 and 0.95

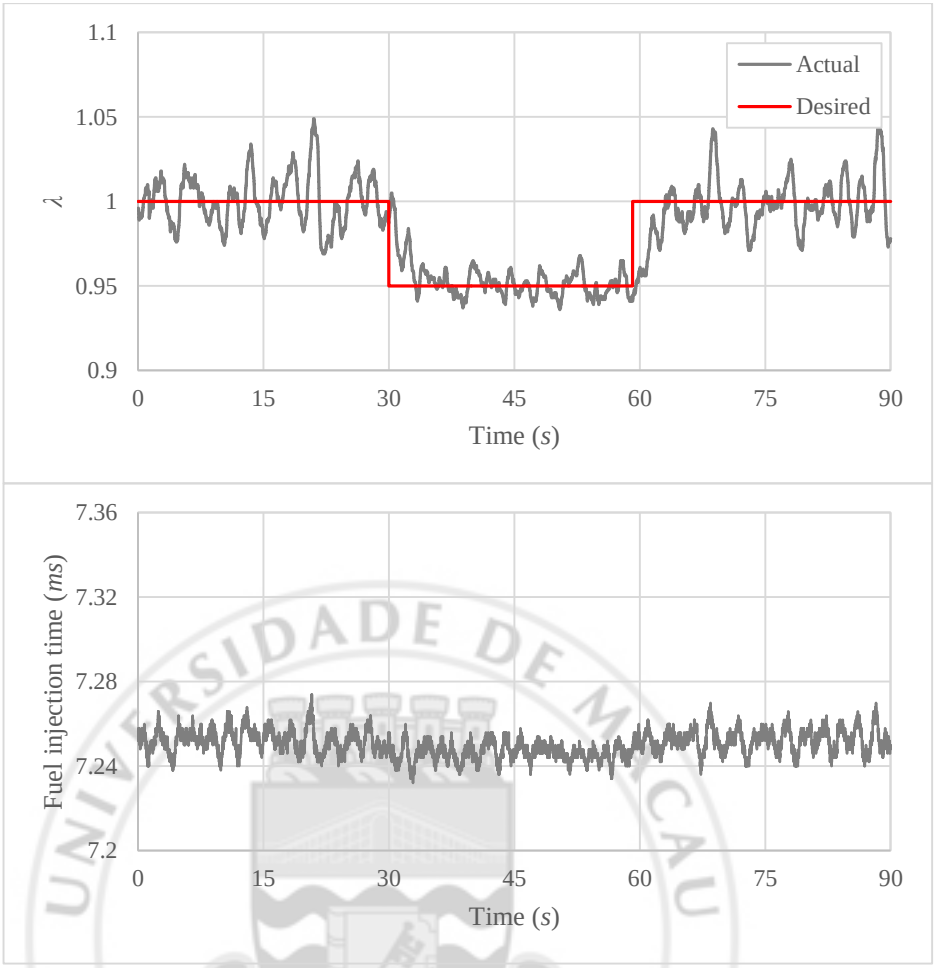


Figure 5.4 Performance of the engine built-in controller for desired λ value varied among 1 and 0.95

5.2 TEST II – TRACKING ABILITY FOR CHANGE OF OPERATING CONDITION

Test II is designed to evaluate the tracking ability of the adaptive controller for change of engine operating condition. Similar to Test I, the engine in this test was warmed up until steady-state was reached. The intelligent adaptive AFR controller was then turned on and the desired λ value was set to 1.00. The throttle position was then changed from 0% to 10% for around 45*s* and returned

back to 0%. The result of this test is given in Figures 5.5 and 5.6.

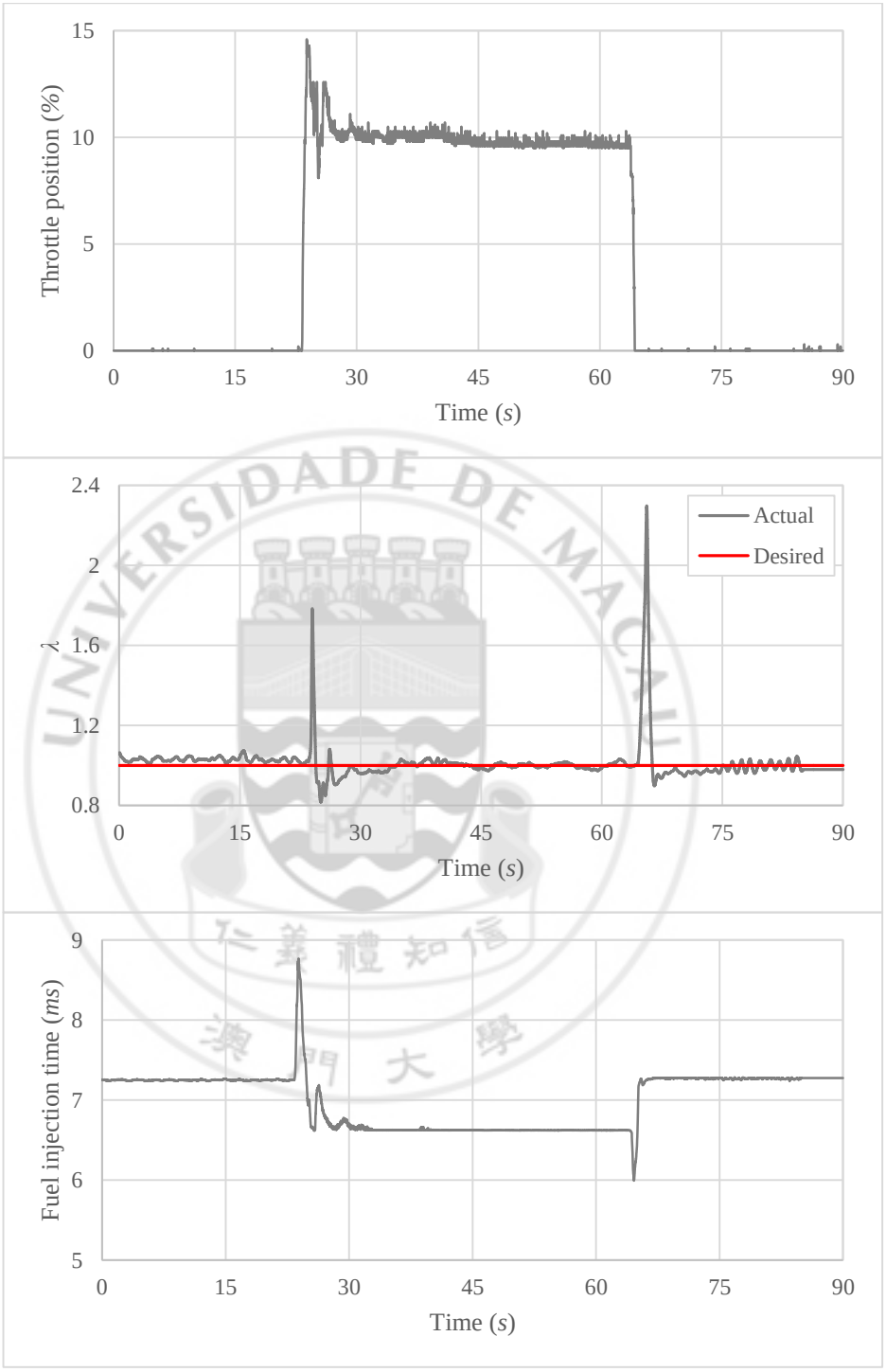


Figure 5.5 Performance of the designed controller for throttle position varied among 0% and 10%

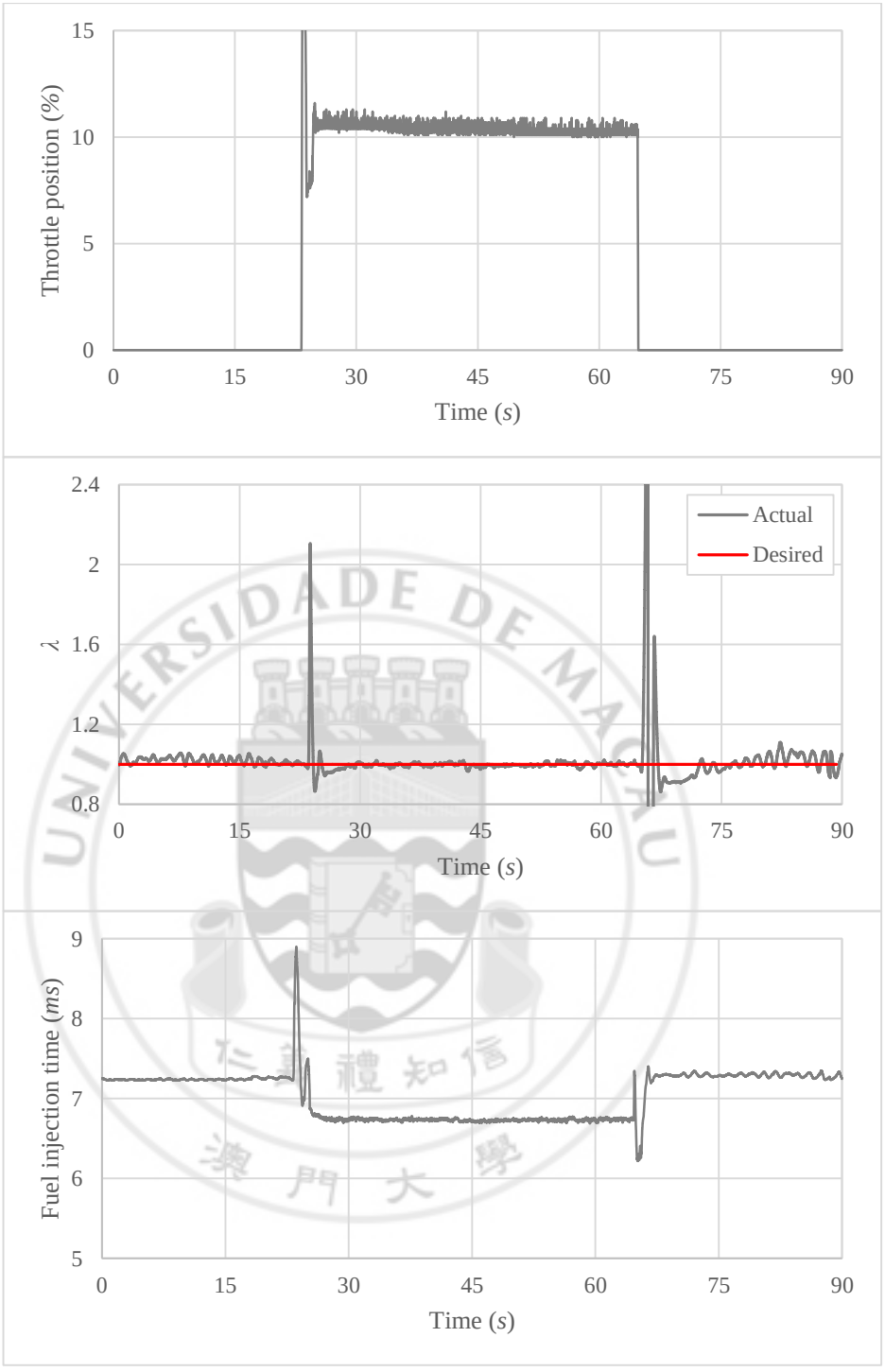


Figure 5.6 Performance of the engine built-in controller for throttle position varied among 0% and 10%

5.3 DISCUSSION OF EXPERIMENTAL RESULTS

From Figures 5.1 and 5.3, it can be seen that when the desired λ value was

changed either from 1.00 to 1.05 or from 1.00 to 0.95, the controller could immediately adjust the fuel injection time to achieve the desired λ value. From Figure 5.5, the design controller could successfully regulate the fuel injection time for the change of throttle position too. The response time and the tracking performance are all within an acceptable range. By comparing Figure 5.1 with 5.2, Figure 5.3 with 5.4, and Figure 5.5 with 5.6, it can also be learnt that the designed adaptive AFR controller outperforms the engine built-in AFR controller in both the response time and tracking performance. To quantify the control performance of the controllers, the integral absolute errors (IAE) of the control results were calculated using the following Eq. (5.1) and shown in Table 5.1:

$$IAE = \int |\lambda(t) - \lambda_d(t)| dt = \sum_{t=0}^{End\ of\ test} |\lambda(t) - \lambda_d(t)|. \quad (5.1)$$

Table 5.1 IAE results of Test I and Test II

	Test I (λ_d = 1.05)	Test I (λ_d = 0.95)	Test II
Designed controller	**92.12**	**52.77**	**333.47**
Engine built-in controller	132.07	101.94	465.41
Improvement	30.25%	48.24%	28.35%

It is clear from the comparison in Table 5.1 that the control performance of the designed controller is much better than that of the engine built-in controller. The AFR control performance has been improved by 28.35% up to 48.24%. This is a very significant result, especially for such tough control task, showing that the designed AFR controller is very useful and effective for AFR control.

Although the transient response of the designed controller for change of throttle may not be very appealing, this is still reasonable because when the throttle position was changed, the corresponding AFR dynamics was also altered, and hence it may take a short period of time for the adaptive controller to adjust its controller parameters in order to adapt to the new AFR dynamics. It has to be noted that, unlike other approaches where prior data and pre-trained were available, the designed adaptive controller does not use any pre-acquired data to determine the best controller parameters in advanced. The learning process (parameter adjusting process) is performed online together with the control process. Taking into account of this property, the experimental results for the designed controller are indeed quite acceptable. All in all, it has been verified by experiments that AFR control using intelligent adaptive approach is feasible and promising.

CHAPTER 6: CONCLUSIONS

6.1 SUMMARY

In this project, an intelligent adaptive controller for air-fuel ratio (AFR) control of an automotive engine system is developed. By studying some intelligent techniques suitable for adaptive control, a machine learning method called fully online sequential extreme learning machine (FOS-ELM) was selected. A controller was designed based on FOS-ELM, and some simulations were conducted to verify the designed controller. The simulation results showed that the designed controller could achieve better tracking performance than other adaptive method. Therefore, the controller was confidently implemented in a real test engine.

Experiments were then set-up in which NI devices, LabVIEW, MATLAB and MoTeC programmable ECU were utilized. The signals of the engine sensors were studied and analyzed so that the controller could be successfully implemented on the test engine. The experimental results demonstrated that the designed intelligent adaptive AFR controller could outperform the engine built-in AFR controller by about 28.35% to 48.24%, showing that the designed controller is effective and promising for maintaining the AFR to a desired level.

6.2 ORIGINALITIES

The originalities of this project include:

1) A newly designed intelligent adaptive controller for AFR control of an automotive engine system, in which no prior knowledge, no pre-trained model and no optimizer are required;
2) A comparison between two artificial intelligence techniques, namely FOS-ELM and BP, for adaptive control problem;
3) A first attempt to implement an intelligent adaptive AFR controller on real test engine;
4) A comparison between the designed intelligent adaptive AFR controller and engine built-in AFR controller.

6.3 RECOMMENDATION FOR FUTURE WORK

Based on the designed intelligent adaptive AFR controller, further researches on the AFR control under different fuel blends can be conducted in the future. The limitations in hardware implementation, such as the time delay problem and the signal noise problem, will be analyzed and hopefully be resolved in the following work. Moreover, more comparison on different control strategies, such as sliding mode control, model predictive control and fuzzy logic control, for AFR control can be carried out to further evaluate the

effectiveness of the adaptive controller. Furthermore, by studying the input-output relationships of other engine parameters, intelligent adaptive controllers for other engine control applications, such as idle speed control, can also be designed and implemented. In addition, it is recommended to apply the same strategy to other intelligent control applications too.

REFERENCE

Al-Olimat, K.S., Ghandakly, A.A. and Jamali, M.M., 2000. "Adaptive air-fuel ratio control of an SI engine using fuzzy logic parameters evaluation". SAE Technical Paper 2000-01-1246.

Arsie, I., Pianese, C. and Sorrentino, M., 2006. "A procedure to enhance identification of recurrent neural networks for simulating air-fuel ratio dynamics in SI engines". Engineering Applications of Artificial Intelligence, Vol. 19, No. 1, pp. 65-77.

Bartlett, P.L., 1998. "The sample complexity of pattern classification with neural networks: The size of the weights is more important than the size of the network". IEEE Transactions on Information Theory, Vol. 44, No. 2, pp. 525-536.

Beltrami, C., Chamaillard, Y., Millerioux, G., Higelin, P. and Bloch, G., 2003. "AFR control in SI engine with neural prediction of cylinder air mass". Proceedings of the 2003 American Control Conference, Vol. 2, pp. 1404-1409.

Chai, J., 2015. "Under the Dome".

Chen, F.-C., 1990. "Back-propagation neural networks for nonlinear self-tuning adaptive control". IEEE Control Systems Magazine, Vol. 10, No. 3, pp. 44-48.

Choi, S.B. and Hedrick, J.K., 1998. "An observer-based controller design method for improving air/fuel characteristics of spark ignition engines". IEEE Transactions on Control Systems Technology, Vol. 6, No. 3, pp. 325-334.

Deng, W., Zheng, Q. and Lin, C., 2009. "Regularized extreme learning machine". Proceedings of the 2009 IEEE Symposium on Computational Intelligence and Data Mining, Nashville, TN, pp. 389-395.

Ebrahimi, B., Tafreshi, R., Masudi, H., Franchek, M., Mohammadpour, J. and Grigoriadis, K., 2012. "A parameter-varying filtered PID strategy for air-fuel ratio control of spark ignition engines". Control Engineering Practice, Vol. 20, No. 8, pp. 805-815.

Ebrahimi, B., Tafreshi, R., Mohammadpour, J., Franchek, M., Grigoriadis, K. and Masudi, H., 2014. "Second-order sliding mode strategy for air-fuel ratio control of lean-burn SI engines". IEEE Transactions on Control Systems Technology, Vol. 22, No. 4, pp. 1374-1384.

Faiz, A., Weaver, C.S. and Walsh, M.P., 1996. Air Pollution from Motor Vehicles: Standards and Technologies for Controlling Emissions. The World Bank, Washington, D.C.

Franceschi, E.M., Muske, K.R., Jones, J.C.P. and Makki, I., 2007. "An adaptive delay-compensated PID air fuel ratio controller". SAE Technical Paper 2007-01-1342.

Haykin, S., 1999. Neural Networks: A Comprehensive Foundation. Prentice-Hall, New Jersey.

Heywood, J.B., 1988. Internal Combustion Engine Fundamentals. McGraw-Hill, New York.

Huang, G., Huang, G.B., Song, S.J. and You, K.Y., 2015. "Trends in extreme learning machines: A review". Neural Networks, Vol. 61, pp. 32-48.

Huang, G.B., Ding, X. and Zhou, H., 2010. "Optimization method based extreme learning machine for classification". Neurocomputing, Vol. 74, No. 1-3, pp. 155-163.

Huang, G.B., Zhou, H.M., Ding, X.J. and Zhang, R., 2012. "Extreme learning machine for regression and multiclass classification". IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, Vol. 42, No. 2, pp. 513-529.

Huynh, H.T. and Won, Y., 2011. "Regularized online sequential learning algorithm for single-hidden layer feedforward neural networks". Pattern Recognition Letters, Vol. 32, No. 14, pp. 1930-1935.

Kan, H.D., Chen, R.J. and Tong, S.L., 2012. "Ambient air pollution, climate change, and population health in China". Environment International, Vol. 42, pp. 10-19.

Lauber, J., Guerra, T.M. and Dambrine, M., 2011. "Air-fuel ratio control in a gasoline engine". International Journal of Systems Science, Vol. 42, No. 2, pp. 277-286.

Li, G.Y., 2007. Application of intelligent control and MATLAB to electronically controlled engines (in Chinese). 1st ed., Publishing House of Electronics Industry, Beijing.

Liang, N.Y., Huang, G.B., Saratchandran, P. and Sundararajan, N., 2006. "A fast and accurate online sequential learning algorithm for feedforward networks". IEEE Transactions on Neural Networks, Vol. 17, No. 6, pp. 1411-1423.

Manzie, C., Palaniswami, M., Ralph, D., Watson, H. and Yi, X., 2002. "Model predictive control of a fuel injection system with a radial basis function network observer". Journal of Dynamic Systems Measurement and Control-Transactions of the Asme, Vol. 124, No. 4, pp. 648-658.

Morelos, M.A.M. and Juan, A.M., 2012. "Fuzzy control strategy for stoichiometric air-fuel mixture in automotive systems". Proceedings of the 2012 World Automation Congress (WAC), pp. 1-6.

Pace, S. and Zhu, G.M.G., 2012. "Sliding mode control of both air-to-fuel and fuel ratios for a dual-fuel internal combustion engine". Journal of Dynamic Systems Measurement and Control-Transactions of the ASME, Vol. 134, No. 3.

Peng, J.Z. and Dubay, R., 2011. "Identification and adaptive neural network control of a DC motor system with dead-zone characteristics". ISA Transactions, Vol. 50, No. 4, pp. 588-598.

Puleston, P.F., Monsees, G. and Spurgeon, S.K., 2002. "Air-fuel ratio and speed control for low emission vehicles based on sliding mode techniques". Proceedings of the Institution of Mechanical Engineers Part I-Journal of Systems and Control Engineering, Vol. 216, No. I2, pp. 117-124.

Pulkrabek, W.W., 2004. Engineering Fundamentals of the Internal Combustion Engine. 2nd ed., Pearson Prentice Hall, New Jersey.

Rong, H.-J. and Zhao, G.-S., 2013. "Direct adaptive neural control of nonlinear systems with extreme learning machine". Neural Computing and Applications, Vol. 22, No. 3-4, pp. 577-586.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1986. "Learning representations by back-propagating errors". Nature, Vol. 323, No. 6088, pp. 533-536.

Sardarmehni, T., Keighobadi, J., Menhaj, M.B. and Rahmani, H., 2013. "Robust predictive control of lambda in internal combustion engines using neural networks". Archives of Civil and Mechanical Engineering, Vol. 13, No. 4, pp. 432-443.

Souder, J.S. and Hedrick, J.K., 2004. "Adaptive sliding mode control of air-fuel

ratio in internal combustion engines". International Journal of Robust and Nonlinear Control, Vol. 14, No. 6, pp. 525-541.

Spooner, J.T., Maggiore, M., Ordóñez, R. and Passino, K.M., 2002. Stable Adaptive Control and Estimation for Nonlinear Systems: Neural and Fuzzy Approximator Techniques. John Wiley & Sons, Inc., New York.

Won, M., Choi, S.B. and Hedrick, J.K., 1998. "Air-to-fuel ratio control of spark ignition engines using Gaussian network sliding control". IEEE Transactions on Control Systems Technology, Vol. 6, No. 5, pp. 678-687.

Wong, K.I., Vong, C.M., Wong, P.K. and Luo, J., 2015. "Sparse Bayesian extreme learning machine and its application to biofuel engine performance prediction". Neurocomputing, Vol. 149, pp. 397-404.

Wong, K.I., Wong, P.K., Cheung, C.S. and Vong, C.M., 2013. "Modelling of diesel engine performance using advanced machine learning methods under scarce and exponential data set". Applied Soft Computing Journal, Vol. 13, No. 11, pp. 4428-4441.

Wong, P.K., Vong, C.M., Gao, X.H. and Wong, K.I., 2014. "Adaptive control using fully online sequential-extreme learning machine and a case study on engine air-fuel ratio regulation". Mathematical Problems in Engineering, Vol. 2014, Article ID 246964, 11 pages.

Wong, P.K., Wong, H.C. and Vong, C.M., 2012. "Online time-sequence

incremental and decremental least squares support vector machines for engine air-ratio prediction". International Journal of Engine Research, Vol. 13, No. 1, pp. 28-40.

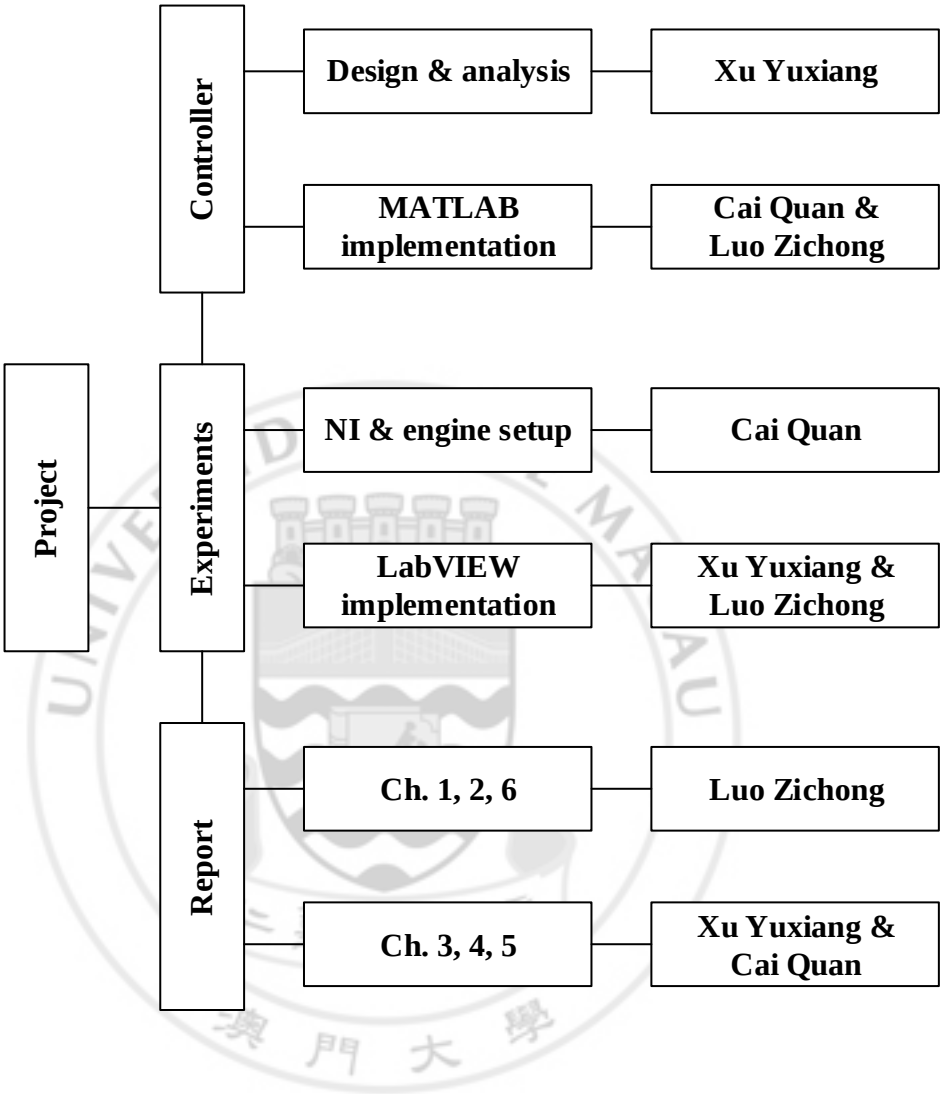
Wong, P.K., Wong, H.C., Vong, C.M., Xie, Z. and Huang, S., 2014. "Model predictive engine air-ratio control using online sequential extreme learning machine". Neural Computing and Applications, pp. 1-14.

Yoon, P. and Sunwoo, M., 2001. "An adaptive sliding mode controller for air-fuel ratio control of spark ignition engines". Proceedings of the Institution of Mechanical Engineers Part D-Journal of Automobile Engineering, Vol. 215, No. D2, pp. 305-315.

Yuan, X.F., Wang, Y.N., Sun, W. and Wu, L.H., 2010. "RBF networks-based adaptive inverse model control system for electronic throttle". IEEE Transactions on Control Systems Technology, Vol. 18, No. 3, pp. 750-756.

Zhai, Y.J., Yu, D.W., Guo, H.Y. and Yu, D.L., 2010. "Robust air/fuel ratio control with adaptive DRNN model and AD tuning". Engineering Applications of Artificial Intelligence, Vol. 23, No. 2, pp. 283-289.

APPENDIX I: WORK BREAKDOWN



APPENDIX II: MATLAB CODE

Simulation, FOS-ELM:

```
clear all;
close all;

nm = 300; %max iteration
nT = 1; % number of training data arrived each time
nI = 1; % number of input
nH = 100; % number of hidden nodes
nO = 1; % number of output
C = 0.001; % regularization factor
WR = 1; %weight for recursive least-squares [0.9 1]

IW = rand(nH,nI)*2-1; % initial input weight
Bias = rand(nH,1);  % initial bias of hidden node
ind = ones(1,nT);
BiasMatrix = Bias(:,ind); %create a matrix of bias for ease of
calculation

P = inv(C*eye(nH)); % initial P
beta = zeros(nH,nO); % initial beta

% generate reference output
% case I
yr = ones(301,1);
yr(51:150) = 0.95;
yr(151:250) = 1.05;
% case II
for k = 1:nm
    yr(k) = 0.03 * sin(2*pi*k/80) + 0.02 * sin(2*pi*k/40) + 1;
end
t(1:nm) = 1:1:nm;

y0 = 1; % initial y(k-1)
u0 = 4.85; % initial u(k-1)
```

```
for k = 1:nm
    if k == 1 % initial model
        y(k) = (0.2*sin(y0)+3.5*(9-u0))/14.7; % actual output from the
plant

        H = exp(-((IW*y0 + BiasMatrix).*(IW*y0 + BiasMatrix)))'; % rbf
        Hu = H(1:(nH/2));
        Hd = H(((nH/2)+1):nH);
        Hdu = Hd*u0;
        H = [Hu,Hdu];
        ym(k) = H * beta; % model output
        e(k) = y(k) - ym(k);
        P = P - WR * (P * H' * H * P) / (1 + WR * H * P * H'); % update
law
        beta = beta + WR * P * H' * e(k); % update law
        y0 = y(k);
        u(k) = u0;
    else
        H = exp(-((IW*y0 + BiasMatrix).*(IW*y0 + BiasMatrix)))'; % rbf
        %H = 1 ./ (1+exp(-(IW * y0 + BiasMatrix)))'; % sigmoid
        Hu = H(1:(nH/2));
        Hd = H(((nH/2)+1):nH);
        bu=beta(1:(nH/2));
        bd=beta(((nH/2)+1):nH);
        u0=(yr(k)-Hu*bu)/(Hd*bd); % control law

        y(k) = (0.2*sin(y0)+3.5*(9-u0))/14.7; % actual output from the
plant

        Hdu = Hd*u0;
        H = [Hu,Hdu];
        ym(k) = H * beta; % model output
        e(k) = y(k) - ym(k);
        P = P - WR * (P * H' * H * P) / (1 + WR * H * P * H'); % update
law
        beta = beta + WR * P * H' * e(k); % update law
        y0 = y(k);
        u(k) = u0;
```

```
    end
end
figure('Position',[300 200 540 450])
subplot(2,1,1);plot(t,yr(1:nm),'b--',t,y,'r');
xlabel('\itk \rm(step)');ylabel('\lambda','rot',0);axis([1 300 0.9
1.15]);legend('Reference','Actual');
subplot(2,1,2);plot(t,u,'r');
xlabel('\itk \rm(step)');ylabel('\it u','rot',0);axis([1 300 4 6]);
```

Simulation, BP:

```
clear all;
close all;

nm = 300; %max iteration
nT = 1; % number of training data arrived each time
nI = 1; % number of input
nH = 100; % number of hidden nodes
nO = 1; % number of output
lr = 0.05; % learning rate
Wij = rands(nH,nI); % initial input weight
Wki = rands(nO,nH); % initial output weight
Wij0 = zeros(size(Wij));
Wki0 = zeros(size(Wki));

% generate reference output
% case I
yr = ones(301,1);
yr(51:150) = 0.95;
yr(151:250) = 1.05;
% case II
for k = 1:nm
    yr(k) = 0.03 * sin(2*pi*k/80) + 0.02 * sin(2*pi*k/40) + 1;
end
t(1:nm) = 1:1:nm;

y0 = 1.0; % initial y(k-1)
u0 = 4.85; % initial u(k-1)
```

```
for k = 1:nm
    y(k) = (0.2*sin(y0)+3.5*(9-u0))/14.7; % actual output from the
plant

    neti = Wij * y(k); %input to hidden node
    for j = 1:nH
       Oi(j) = exp(-(neti(j)*neti(j)));
    end
    Ok = Wki * Oi'; %network output
    u(k) = 9 - (14.7*yr(k+1) - Ok) / 3.5;
    y1(k) =  (0.2*sin(y(k))+3.5*(9-u(k)))/14.7;
    u0 = u(k); %replace u(k-1) for next iteration
    y0 = y(k); %replace y(k-1) for next iteration
    e(k+1) = y1(k) - yr(k+1);
    for i = 1:nH
       dWki(i) = lr*e(k+1)*Oi(i); %bp output weight update
    end
    Wki1 = Wki+dWki; %compute output weight for k+1
    Wki0 = Wki; %replace Wki(k-1)
    Wki = Wki1; %replace Wki(k)
    for i = 1:nH
       dfi(i) = -2 * exp(-(neti(j)*neti(j))) * neti(j);
    end

    dWij = lr*e(k+1)*dfi.*Wki*y(k);  %bp input weight update
    Wij1 = Wij+dWij'; %compute input weight for k+1
    Wij0 = Wij; %replace Wij(k-1)
    Wij = Wij1; %replace Wij(k)
end
figure('Position',[300 200 540 450])
subplot(2,1,1);plot(t,yr(1:nm),'b--',t,y,'r');
xlabel('\itk \rm(step)');ylabel('\lambda','rot',0);axis([1 300 0.9
1.15]);legend('Reference','Actual');
subplot(2,1,2);plot(t,u,'r');
xlabel('\itk \rm(step)');ylabel('\it u','rot',0);axis([1 300 4 6]);
```