Knock Intensity and Torque Control on an SVC Engine

Master's thesis
performed in **Vehicular Systems**

by

Klara Sinnerstad

Reg nr: LiTH-ISY-EX-3497-2003

12th December 2003

Knock Intensity and Torque Control on an SVC Engine

Master's thesis

performed in **Vehicular Systems**,
Dept. of Electrical Engineering
at **Linköpings universitet**

by **Klara Sinnerstad**

Reg nr: LiTH-ISY-EX-3497-2003

Supervisor: **Ph.D. Student Ylva Nilsson**
Linköpings Universitet
Research Engineer Martin Gunnarsson
Linköpings Universitet

Examiner: **Associate professor Lars Eriksson**
Linköpings Universitet

Linköping, 12th December 2003

LINKÖPINGS UNIVERSITET	**Avdelning, Institution** Division, Department Vehicular Systems, Dept. of Electrical Engineering 581 83 Linköping	**Datum** Date 12th December 2003

Språk Language ☐ Svenska/Swedish ☒ Engelska/English ☐ ______	**Rapporttyp** Report category ☐ Licentiatavhandling ☒ Examensarbete ☐ C-uppsats ☐ D-uppsats ☐ Övrig rapport ☐ ______	**ISBN** — **ISRN** LITH-ISY-EX-3497-2003 **Serietitel och serienummer** Title of series, numbering **ISSN** —

URL för elektronisk version

http://www.vehicular.isy.liu.se
http://www.ep.liu.se/exjobb/isy/2003/3497/

Titel	Reglering av Knackintensitet och Utmoment på en SVC Motor
Title	Knock Intensity and Torque Control on an SVC Engine
Författare Author	Klara Sinnerstad

Sammanfattning
Abstract

Knock is a phenomenon that limits how efficiently an engine can operate. Severe knock is harmful to the engine and must therefore be avoided. Controlling the knock intensity is complicated by a phenomenon called cycle to cycle variations. Because of these variations, the knock intensity must be considered a stochastic variable and the control is made on a mean value from a large number of cycles.

The SVC (Saab Variable Compression) concept adds the compression ratio as an extra degree of freedom. At Vehicular systems, research is done on how to put this additional variable to its best use.

A controller is developed that control the engine to a desired knock intensity and torque, using the ignition angle and the pedal position. The controller is implemented as two separate controllers in Matlab and Simulink. These are merged together with a Stateflow chart. A confidence interval calculation is implemented for the mean value of the knock intensity. A program is also developed to process a large number of operating points and make measurements in all of them.

The conclusion is that the basic construction of the controller and the script are filling their functions but that there are some improvements left to be done. The controller is rather slow and the calculations of the confidence interval needs further refinement.

Nyckelord
Keywords SVC, Knock control, Torque control, Variable compression, Stateflow

Abstract

Knock is a phenomenon that limits how efficiently an engine can operate. Severe knock is harmful to the engine and must therefore be avoided. Controlling the knock intensity is complicated by a phenomenon called cycle to cycle variations. Because of these variations, the knock intensity must be considered a stochastic variable and the control is made on a mean value from a large number of cycles.

The SVC (Saab Variable Compression) concept adds the compression ratio as an extra degree of freedom. At Vehicular systems, research is done on how to put this additional variable to its best use.

A controller is developed that control the engine to a desired knock intensity and torque, using the ignition angle and the pedal position. The controller is implemented as two separate controllers in Matlab and Simulink. These are merged together with a Stateflow chart. A confidence interval calculation is implemented for the mean value of the knock intensity. A program is also developed to process a large number of operating points and make measurements in all of them.

The conclusion is that the basic construction of the controller and the script are filling their functions but that there are some improvements left to be done. The controller is rather slow and the calculations of the confidence interval needs further refinement.

Keywords: SVC, Knock control, Torque control, Variable compression, Stateflow

Acknowledgment

First I would like to thank my supervisors Ylva Nilsson and Martin Gunnarsson who are not only very good at what they do but are also very nice and helpful people. Further, I would like to thank the entire staff of Vehicular Systems for taking taking such good care of me and making me feel like home and always answering any questions that i might ask. Per Gösta Andersson at MAI deserves to be thanked for putting time and effort into answering my questions concerning statistics. I also want to thank my opponent Linda Rattfält and my boyfriend Jonas Norberg for their support and creative feedback. Last but not least, I would like to thank my examiner Lars Eriksson.

Contents

Abstract **v**

Preface and Acknowledgment **vi**

1 Introduction **1**

1.1 Objectives 1
1.2 Problem Description 2
1.3 Method 2
1.4 Outline 3

2 SI engines **4**

2.1 Definitions of Cylinder Geometry 4
2.2 The Four Stroke Cycle 5
2.3 Knock 6
2.4 Output Torque 7
2.5 Cycle to Cycle Variations 8
2.6 Estimation of Resonance Frequency in a Cylinder 9
2.7 SVC Engine 10
2.7.1 Variable Compression 10
2.7.2 Down Sizing 11
2.7.3 Supercharging 12
2.8 Turbo Charged Engine 12
2.9 The Cylinder Pressure Sensors 13

3 Statistical Treatment of Knock Intensities **14**

3.1 Choice of Distribution 15
3.2 Gamma Distribution 16
3.3 Parameter Estimation 16
3.4 Confidence Interval for Arithmetic Mean 17
3.4.1 Reflections on the Confidence Interval 18

4 The Controller **20**
4.1 The Knock Intensity Controller 20
4.2 The Torque Controller 21
4.3 Merging the Controllers 23
4.3.1 Stateflow 23
4.3.2 The Knock Intensity Control Stateflow Chart . . 23
4.3.3 Reflections over the Knock Intensity Control . . 25
4.4 The Controller 26

5 Running the Controller **28**
5.1 GUI 28
5.1.1 GUIDE 30
5.2 Script 30
5.2.1 Defining the Operating Points 30
5.2.2 Iterating Through the Operating Points 30
5.2.3 Torque Ramping 32

6 Implementation in the Engine Laboratory **34**

7 Evaluation and Results **37**
7.1 Results from the Knock Controller 37
7.2 Results from the Torque Controller 39
7.3 Evaluation of the Controller 40
7.3.1 Case One 41
7.3.2 Case Two 44
7.3.3 Case Three 47
7.4 Evaluation of the Script 47
7.4.1 Evaluation Problems 50
7.4.2 Results from the script 50

8 Conclusion **53**
8.1 Summary 53
8.2 Future Work 54

References **57**

Notation **59**

A The Stateflow Chart **61**

B Matlab Functions **63**

C Manual for the program **66**

C.1 User Interface . . . 66

C.1.1 GUI . . . 66

C.1.2 Script . . . 67

C.1.3 Changing Fixed Values . . . 68

Chapter 1

Introduction

Knock is a phenomenon that limits how efficiently an engine can operate. Severe knock is harmful to the engine and must therefore be avoided. The engine control unit (ECU) tries to make the engine operate at the most fuel efficient operating point, but the demand not to let the knock get to high puts a limit to how the amount of charge in the cylinder and the ignition angle can bee chosen. As a consequence, it may not always be possible for the engine to operate at its most efficient operating point.

The SVC (Saab Variable Compression) concept adds the compression ratio as an extra degree of freedom. At Vehicular systems research is done on how to put this additional variable to its best use. In order to do this, a lot of measurements are needed from the engine to see how it behaves in different driving situations and to validate the models. Therefore there is a wish to automatize these measurements as far as possible.

1.1 Objectives

The purpose of this thesis is to simplify and automatize knock measuring on a Saab Variable Compression engine, mounted in a test-bench in Vehicular Systems engine laboratory. The goals are

- To build a controller that can control knock intensity and output torque for a requested compression, using ignition angle and air mass flow past the throttle as control signals.

- To use this controller to automatize measurements on an SVC engine, by running it from a script. In this script supervision is added so that the knock intensity does not get too high when the operating point is changed.

1.2 Problem Description

The reason why knock is so interesting is because it can be very harmful to the engine. Avoiding severe knock puts a limit on possible levels of charge and possible ignition angles. This can have the consequence that the engine is forced to operate in an operating point that does not have optimal fuel efficiency. Running an engine at its optimum has a lot of advantages. The most obvious being that the more efficient the engine runs, the more you can get out of it from the same input. The input in this case is gasoline, and therefore an efficiently running engine will reduce the amount of gasoline that it uses.

The knock intensity is mainly affected by four things; compression ratio, ignition angle, engine speed and amount of charge in the cylinder. Earlier ignition will cause a higher peak pressure and thus more knock. This is why the knock intensity can be controlled with the ignition angle.

Controlling knock intensity is however complicated by a phenomenon called cycle to cycle variations. This phenomenon, that will be explained in the next chapter section 2.5, causes the knock to vary even when the engine runs at steady state, which means that all controllable parameters are constant. Because of this, knock has to be treated as a stochastic variable, and the control has to be done on a mean value from a number of cycles. Since it takes comparatively long time to measure the cylinder pressure, filter out the knock and then calculate the mean value, the control can not be made in real time. In spite of this, the aim is of course to use the controller directly on the engine and this problem will have to be handled in some way.

Another problem is that most of the processes in the engine are interconnected. Changing one parameter may have other consequences than was originally intended. For example, changing the ignition angle affects the output torque. Therefore, the torque controller is implemented as a complement to the knock intensity controller.

1.3 Method

The controller is built in Matlab, using the toolboxes Simulink and Stateflow. It consist of two parts that are merged into one entity. The first part is the knock intensity controller. It is implemented as a number of Matlab functions that are managed by a Stateflow chart. The second part is the torque controller, which is built as a Simulink model. To connect the controllers, the Stateflow chart is incorporated into the Simulink model as a subsystem.

Since knock intensity is considered a stochastic variable, the knock intensity control is made on a mean value from a number of cycles. To know how many cycles to use, statistical calculations are implemented to ensure that the mean value is determined with desired accuracy.

A Graphical User Interface (GUI) is implemented from which the controller is called in one operating point at the time. A script is also implemented that calls the controller, makes measurements and stores data for any desired number of operating points.

1.4 Outline

In chapter 2, a short survey of the SI engine is presented. The chapter starts by telling a bit about how an SI engine works and then explains the problems that occur in an SI engine such as knock and cycle to cycle variations. It ends with a section about what makes the SVC engine special. Chapter 3 is about the statistical treatment of the knock. It is explained why the Gamma distribution is chosen to fit to the measured knock and how the distribution parameters and the sample size are estimated. In chapter 4 the controller is described and it is explained how it is constructed and why. Chapter 5 explains how the controller is intended to be used. Two methods to run the controller are presented; from a GUI or from a script. In chapter 6 a description is given of the laboratory environment in which the controller is implemented. Chapter 7 tells about how the controller is evaluated and what the results are. Finally, in chapter 8 a summary is made and some suggestions are given on what could be done in the future.

Chapter 2

SI engines

A spark ignited (SI) engine can also be referred to as an Otto, gasoline or petrol engine. This is the engine most commonly used for passenger cars. In this chapter some basic facts about the SI engine will be presented. For more information see [8].

2.1 Definitions of Cylinder Geometry

Figure 2.1 shows a sketch of the cylinder geometry. In this figure some important parameters are defined. Top Dead Center (TDC) is the highest piston position, where the piston turns and moves back down. Bottom Dead Center (BDC) is the turning point in the bottom. The picture also shows the definition of the crank angle, θ, the piston stroke

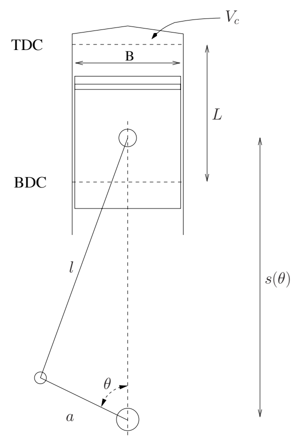


Figure 2.1: Some important parameters in the cylinder geometry.

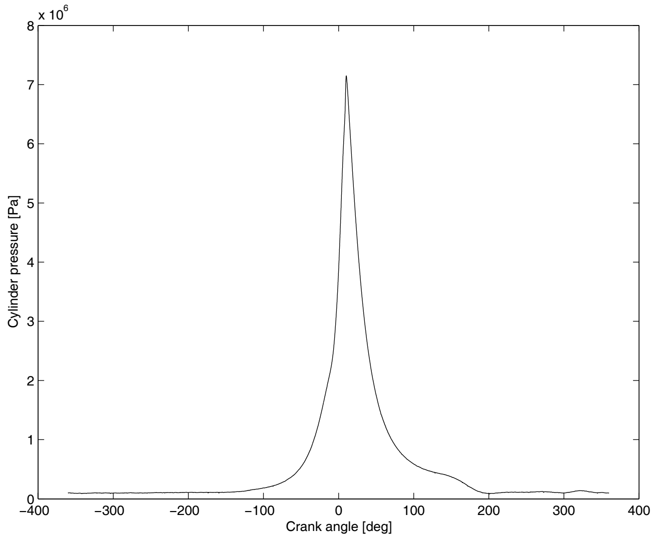


Figure 2.2: The cylinder pressure plotted against the crank angle.

L and the bore B. The clearance volume V_c is the cylinder volume that is left when the piston is at TDC. Another important parameter is the displaced volume V_d, which is the change of volume in the cylinder due to the piston movement. V_d is calculated as

$$V_d = \frac{\pi B^2 L}{4} \tag{2.1}$$

From V_c and V_d a parameter called compression ratio, r_c, is derived.

$$r_c = \frac{maximum\ cylinder\ volume}{minimum\ cylinder\ volume} = \frac{V_d + V_c}{V_c} \tag{2.2}$$

2.2 The Four Stroke Cycle

The four stroke cycle is a way of describing what happens in an SI engine. The four strokes intake, compression, expansion and exhaust are related to the piston movement. One cycle takes two full revolutions of the piston. A plot of the cylinder pressure against the crank angle can be seen in figure 2.2.

The first stroke is the intake, during which the piston moves from TDC to BDC. During this stroke the intake valve is open and the exhaust port is shut. As the piston moves down, the cylinder is filled with fresh air and fuel mixture from the intake manifold. This phase starts

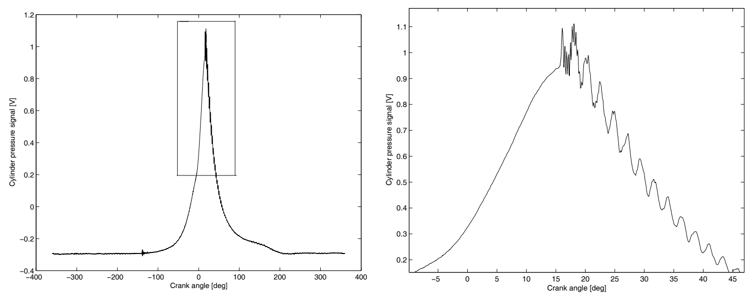


Figure 2.3: A cylinder pressure curve with knock. The left plot shows a whole cycle and the right plot shows an enlargement of the area of interest.

at approximately $-360\,^\circ$ and ends when the intake valves close about $-180\,^\circ$. During this phase, the cylinder pressure is close to constant.

The next stroke is compression and now the piston moves from BDC to TDC. During this stroke, both the intake valve and the exhaust port are shut. The air and fuel mixture is compressed and as a result the temperature and pressure rises. About $25\,^\circ$ before TDC a spark is ignited and the combustion starts. The angle where the spark is ignited is called ignition angle, θ_{ign}. The flame propagates through the cylinder and the combustion continues into the next phase which is the expansion phase.

The expansion stroke is where useful work is delivered from the engine. The high pressure in the cylinder pushes the piston down, and this creates the force that moves the vehicle forward. During the expansion phase the intake valve and the exhaust port are both still shut. The pressure has its peak shortly after TDC. The combustion finishes about $40\,^\circ$ after TDC.

Around $130\,^\circ$ after TDC, when the exhaust port opens and the exhaust stroke starts, the cylinder is almost emptied as the piston moves from BDC to TDC. The cycle ends at about $360\,^\circ$ and a new cycle can start.

2.3 Knock

Under normal conditions the combustion is ignited by a spark at the spark plug. The flame kernel grows and propagates through the combustion chamber until it reaches the cylinder walls where it extinguishes. The flame front propagates with a speed much less then the speed of sound and therefore the cylinder pressure can be considered constant in the cylinder. The unburned gas in front of the flame is called the end gas.

Knock is a phenomenon that occurs when high temperature and pressure causes the end gas to self ignite. This causes a very high local pressure and the propagation of pressure waves across the combustion chamber. These pressure waves excites the resonance modes of the cylinder. The frequency of the oscillations under knocking conditions depends on engine geometry, and is often in the range of 5 to 10 kHz. A plot of the cylinder pressure with a clearly visible knock can be seen in figure 2.3 [1].

There are several theories about what it is that causes the damage on the engine during knocking conditions. The most accepted is that it is caused by heat transfer [6]. When knocking conditions occur, the piston and the walls of the combustion chamber are exposed to a great deal of additional heat which results in overheating of these parts. As a result, the thermal boundary layer at the combustion chamber wall can be destroyed. This causes increased heat transfer which might lead to certain surfaces causing pre-ignition [11]. Substantial knock can damage the engine and is stressful to the driver and is therefore the most important limitation for SI engines. In order to control the knock it is sometimes necessary to regulate away from the most efficient operating point.

The most common way to control knock is to delay the ignition and thus prevent overheating. When changing the ignition angle it must be taken into consideration that this affects not only the knock but also the output torque.

2.4 Output Torque

The output torque is a function of several different variables, the most important of them being the air mass flow past the throttle, engine speed, compression rate and ignition angle. When the driver steps on the gas pedal in an ordinary car, the requested air mass flow will change. The engine control system regulates the air mass flow by changing the throttle angle. When the throttle angle is changed, the amount of air that will be used in the combustion is changed. The ECU corrects the amount of fuel injected, so that the air and fuel mixture is still stoichio-

[1]The intention was to present the cylinder pressure in Pa, but because of difficulties in transforming the measured signal it is presented in volt. This has no practical meaning since only the principal appearance of the curve is discussed.

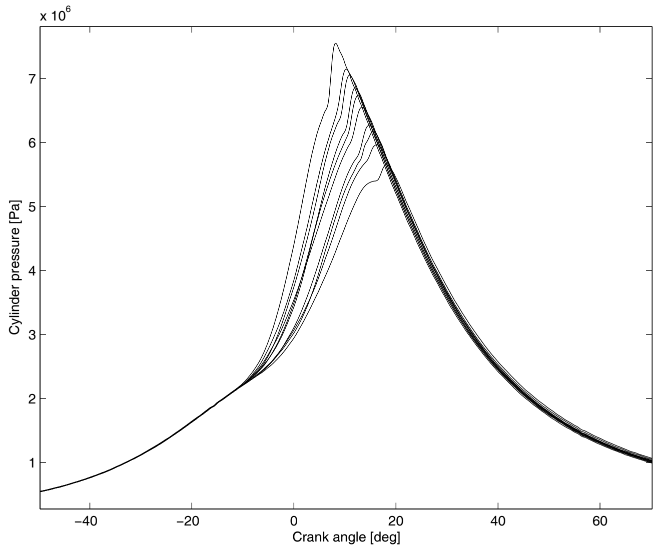


Figure 2.4: Cycle to cycle variations for ten consecutive cycles when all the controllable parameters are held constant.

metric. A stoichiometric air/fuel mixture is ideal, the fuel and oxygen will be completely consumed in the combustion. The only residues of this reaction are water and carbon monoxide. The more air and fuel mixture that is combusted, the higher the output torque.

2.5 Cycle to Cycle Variations

There are always cycle-to-cycle variations in SI engines. These variations occur even when all the controllable parameters, engine speed, air mass flow, fuel injection and all temperatures and pressures are held constant. In figure 2.4 ten consecutive measurements of the cylinder pressure can be seen, that clearly shows the variations. According to [8] three causes for the variations have been found.

- Variations in the gas motion in the cylinder.
- Variations in the amount of fuel, air and recycled gases causes the amount of energy in the cylinder to vary from one cycle to another.
- Spatial variations in the concentration of air, fuel and recycled gases. Especially the distribution close to the spark plug is important for early flame development and propagation.

Because of the cycle to cycle variations the knock intensity will also change even though all controllable parameters are constant. The knock intensity is therefore considered a stochastic variable.

2.6 Estimation of Resonance Frequency in a Cylinder

As stated in section 2.3, the theoretical value of the resonance frequency of a normal sized cylinder is 5 to 10 kHz. A theoretical calculation of this value was made for comparison. If the key tone is assumed to be in the tangent ($\hat{\varphi}$) direction of the cylinder, the resonance frequency of the cylinder can be estimated by the expression

$$f = \frac{c}{2\pi r_a} \tag{2.3}$$

Here c is the speed of sound and r_a is the average radius of the cylinder. The bore, B, of an SVC engine is 0.068 m and from this r_a can be calculated to 0.0227 m, using the expression [4]

$$r_a = \frac{\int_A r\,dA}{\int_A dA} = \frac{\int_0^{2\pi}\int_0^{\frac{B}{2}} r^2\,drd\phi}{\int_0^{2\pi}\int_0^{\frac{B}{2}} r\,drd\phi} \tag{2.4}$$

The speed of sound is given by

$$c = \sqrt{\frac{\gamma k_b T}{\mu}} \tag{2.5}$$

where γ is the ratio of specific heats which is approximately 1.3. k_b is the Boltzmann constant which equals $1.38066e^{-23}$ J/K and T is the average temperature of the gas near TDC [9]. T is approximately 2500 K. The choice of fuel is ISO-octane C_8H_{18} and the oxidizing reaction according to [8] gives

$$C_8H_{18}+12.5O_2+12.5\cdot3.773N_2 \rightarrow 8CO_2+9H_2O+12.5\cdot3.773N_2 \tag{2.6}$$

The molecular mass μ of the gas after the combustion is calculated to 28.6 u and u is $1.660540e^{-27}$ kg. From this, c can be estimated to about 970 m/s. It is now easy to calculate f, and the result is about 6.8 kHz.

A plot of the spectrum for a cylinder pressure curve for the SVC engine can be seen in figure 2.5. The second peak is the key tone of the knock which is approximately between 5 and 10 kHz. The following peaks are the overtones of the knock and after 20 kHz what can be seen is mainly noise.

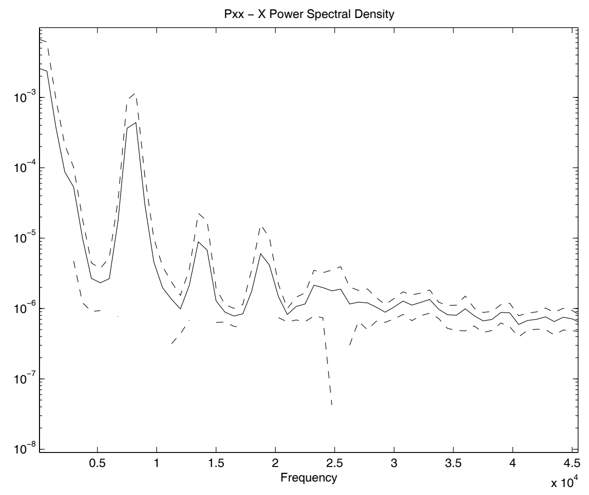


Figure 2.5: Spectrum for a cylinder pressure curve.

2.7 SVC Engine

Saab engine designers came up with the concept of a variable compression engine back in the early 1980s. The first patent application was lodged late in 1990 [2]. This engine is still under development and for the time being, only prototypes exist.

The SVC engine combines the new concept of variable compression with the existing ideas for down sizing and supercharging. These will be explained in sections 2.7.2 and 2.7.3.

2.7.1 Variable Compression

As a general rule the energy in the fuel will be used more efficiently the higher the compression ratio is. Figure 2.6 shows the efficiency as a function of the compression ratio in an ideal case when heat transfer is not taken into consideration. The efficiencies in the figure are calculated from the expression

$$\eta = 1 - \frac{1}{r_c^{\gamma-1}} \tag{2.7}$$

where γ is the ratio of specific heats and r_c is the compression ratio. For an SI engine operating at stoichiometric conditions, γ is approximately 1.3. [8]

In reality, if the compression ratio is to high, this will give rise to knock.

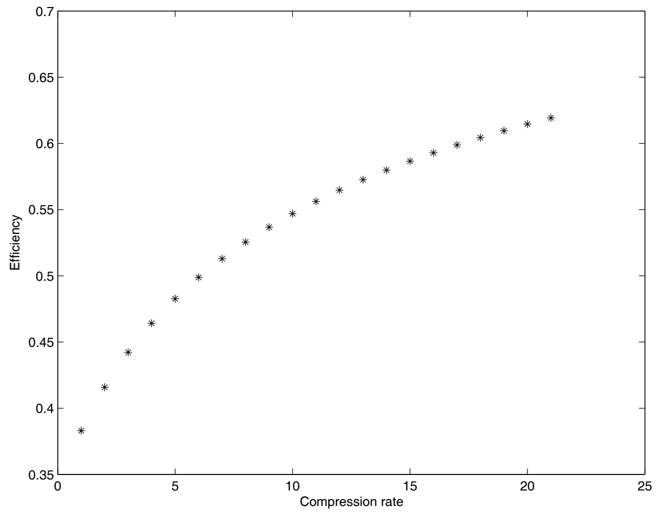


Figure 2.6: Engine efficiency as a function of the compression ratio. This shows the ideal case when heat transfer is not taken into consideration.

Increased compression will lead to increased temperature at the end of the compression stroke where the combustion starts. In conventional engines the highest knock will occur at high load. This sets a limit for the highest possible compression rate. On a SVC engine the compression ratio can be altered from 14:1, which puts the fuel to best use when the engine is running at low loads, to 8:1 at high loads to enable supercharging and prevent knock [2]. A schematic description of how the variable compression works can be seen in figure 2.7.

Although variable compression ratio is what makes the SVC unique, the fuel efficiency compared to a conventional, normally aspirated engine is only improved by 4-5 percent because of the variable compression system. To use the full potential of variable compression it is combined with a smaller engine design (down sizing) and high supercharging pressure. Doing this can improve the efficiency up to 30 percent. [2]

2.7.2 Down Sizing

An SI engine is most fuel efficient when it is running at high loads. This is because there is more energy loss per produced unit of useful energy at low loads. Also, when the load is increased, the friction does not increase accordingly. Therefore, a greater part of the produced energy is wasted on friction for low loads. When the engine is running

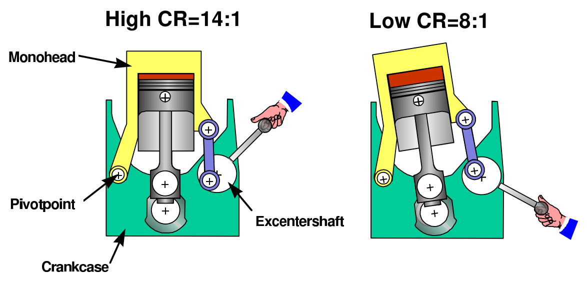


Figure 2.7: A schematic description of variable compression.

at low load it does not use so much fuel and therefore, to keep the air to fuel ratio right, the air supply must be restricted. This restriction in air supply can cause a slight vacuum in the cylinder that opposes the piston movement during the intake stroke. The pumping loss is the extra energy needed to get the piston down in the case of underpressure. Since a small engine needs a higher load than a bigger engine to produce the same amount of work, the small engine is more fuel efficient. A small engine is also lighter and has less friction. [2]

2.7.3 Supercharging

To make a small engine more powerful, supercharging can be used. The idea is that if more air can be forced into the engine, then more fuel can be injected and still keep the mixture stoichiometric. As a result, the engine will deliver more power for each piston stroke which will result in a higher output torque. Supercharging can be done in different ways, the most common being mechanical supercharging or turbocharging. In an SVC engine supercharging is done by a mechanical compressor, since it is faster and can deliver higher pressure than a turbo charger. [2]

2.8 Turbo Charged Engine

Due to mechanical problems with the SVC engine in Vehicular systems engine laboratory, some parts of this thesis work is made on a Turbo Charged (TC) engine. The TC engine is also mounted in the engine laboratory. Since the size and geometry of the cylinders in the SVC and the TC engines are different, the resonance frequencies will be slightly different. The TC is manufactured by Saab Automobil AB and is normally mounted in Saab 9-5 Aero. It is a 2.3 liter, four cylinder engine, equipped with a turbo charger. It is from year 2000.

2.9 The Cylinder Pressure Sensors

In the TC engine, the cylinder pressure sensor is mounted in the spark plug. It is a piezoelectric, high temperature, cylinder pressure sensor. The spark plug is mounted in the center of the cylinder head. Unfortunately the key tone of the knock frequency has a node here and can not be measured. Therefore, for the TC engine, the first overtone is investigated instead of the key tone. The frequency of this overtone has been found empirically and is approximately between 12 and 16 kHz.

In the SVC engine a hole is made through the cylinder head and the sensor is flush mounted in the hole. This sensor catches the key frequency of the knock. Hence, the investigations are made on the key tone (5-10 kHz).

Chapter 3

Statistical Treatment of Knock Intensities

For the continuation of this report, the term *knock intensity* will refer to a gathering of the peak knock amplitude from each of the measured cylinder pressure cycles. As stated in the previous chapter, section 2.5, the knock intensity can vary from cycle to cycle even when all controllable parameters are held constant. Because of these variations, the knock intensity should be regarded as a random variable and the control should be made on the intensity mean from a number of cycles. For the cycles without knock, the knock amplitude will be zero. These values are also included in the mean value calculation. This intensity mean will be referred to as *knock intensity mean*. In the expression below, BP stands for band pass filtration.

$$\text{knock intensity mean} = \frac{1}{n}\sum_{i=1}^{n} \max|p_{cyl(BP,i)}| \tag{3.1}$$

The term *sample size* will be used consistently in this report as referring to the number of measured cycles used in the calculations. The sample size needed to achieve a desired accuracy with adequate certainty depends on the size of the cycle to cycle variations.

The strategy is to find out what kind of distribution that can be fitted to the measured knock intensity data in most operating points. The considered distributions are Normal, Gamma, Chi-square and Weibull. For each set of measured data a new estimation will be made of the distribution parameters. From these parameter estimations and the confidence level it will be possible to estimate a confidence interval for the arithmetic mean of the knock intensity. If the length of the confidence interval is short enough then the sample size is adequate and if

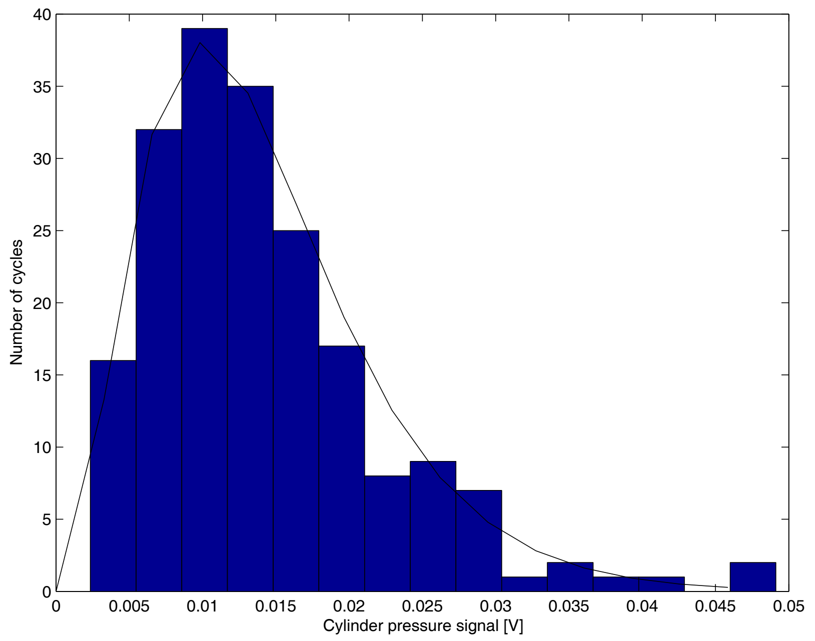


Figure 3.1: A histogram of the measured data from one operating point containing 200 cycles. Shown in the plot is also the Gamma distribution that gives the best fit.

the confidence interval is too long a larger sample size is required.

3.1 Choice of Distribution

Knock intensity data from approximately a hundred operating points was examined to get an idea of what kind of distribution to use. It was found that data from nearly every operating point had a skewed distribution. Because of the skewedness and since there are by definition no negative intensities, the Normal distribution was rejected. The knock intensities were plotted in histograms together with the fitted Chi-square, Weibull and Gamma distributions. From visual comparations of these plots it was decided that these distributions could be fitted approximately equally well to data. Since it proved to be difficult to make the desired calculations on the Chi-square and Weibull distributions, but was rather simple to make them on the Gamma distribution, the Gamma distribution was chosen. Figure 3.1 shows a histogram of the measured data and the Gamma distribution that gives the best fit.

3.2 Gamma Distribution

The formula for the probability density function (pdf) of the Gamma distribution X $\in \gamma(s,\lambda)$ is

$$f(x) = \begin{cases} \frac{\lambda^s}{\Gamma(s)} x^{s-1} e^{-x\lambda} & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{3.2}$$

where s is the shape parameter, λ is the scale parameter, and Γ is the Gamma function which has the formula [3]

$$\Gamma(s) = \int_0^\infty x^{s-1} e^{-x} dx \tag{3.3}$$

There are two theorems that will be helpful in the sample size estimation.

Theorem 1. *If X_1,...,X_n are independent random variables, with the same distribution $X_i \in \gamma(s,\lambda)$, then $\sum_{i=1}^n X_i \in \gamma(sn,\lambda)$ [3]*

Theorem 2. *If $X \in \gamma(s,\lambda)$, then $\lambda X \in \gamma(s,1)$ [10]*

3.3 Parameter Estimation

The method used to estimate the distribution parameters is Maximum Likelihood Estimation, MLE. The likelihood function of X is

$$L_X(s,\lambda) = \prod_{i=1}^n f_{X_i}(x_i,s,\lambda) = \frac{\lambda^{ns}}{\Gamma^n(s)} e^{-\lambda \sum_{i=1}^n x_i} \prod_{i=1}^n x_i^{s-1} \tag{3.4}$$

and the logarithm of the likelihood function is

$$\ln L_X(s,\lambda) = ns \ln \lambda - n \ln \Gamma(s) + (s-1) \sum_{i=1}^n \ln x_i - \lambda \sum_{i=1}^n x_i \tag{3.5}$$

The values of s and λ that maximize the likelihood function can be found by differentiating L_X with respect to each of these parameters and set the result equal to zero

$$\frac{\partial}{\partial s} \ln L_X(s,\lambda) = n \ln \lambda - n \frac{\Gamma'(s)}{\Gamma(s)} + \sum_{i=1}^n \ln x_i = 0 \tag{3.6}$$

$$\Rightarrow \frac{\Gamma'(s)}{\Gamma(s)} - \ln \lambda = \frac{1}{n} \sum_{i=1}^n \ln x_i \tag{3.7}$$

$$\frac{\partial}{\partial\lambda}\ln L_X(s,\lambda) = \frac{ns}{\lambda} - \sum_{i=1}^{n} x_i = 0 \tag{3.8}$$

$$\Rightarrow \frac{s}{\lambda} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{3.9}$$

Solving equation 3.9 for λ and substituting the result into equation 3.7 will give the equation

$$\frac{\Gamma'(s)}{\Gamma(s)} - \ln s = \frac{1}{n}\sum_{i=1}^{n}\ln x_i - \ln\sum_{i=1}^{n} x_i \tag{3.10}$$

This equation can be solved for s numerically, and then λ can be calculated from equation 3.9 [1]. In the controller this is done with the Matlab command *gamfit* that has the *knock intensity* as input and the ML-estimates of the parameters for the gamma distribution as output. After λ and s are estimated, the length of the confidence interval can be calculated.

3.4 Confidence Interval for Arithmetic Mean

By combining the theorems from section 3.2, the desired distribution can be derived. First theorem 1 is used to find the distribution of a sum of n variables, with the same distribution $X_i \in \gamma(s,\lambda)$.

$$\sum_{i=1}^{n} X_i \in \gamma(sn,\lambda) \tag{3.11}$$

Then theorem 2 is applied, λ is transfered out of the distribution and the result is

$$\lambda\sum_{i=1}^{n} X_i \in \gamma(sn,1) \tag{3.12}$$

If this equation is prolonged with n, it will give the desired distribution.

$$n\lambda\sum_{i=1}^{n}\frac{X_i}{n} \in \gamma(ns,1) \tag{3.13}$$

Based on this distribution, a confidence interval with level of confidence 1-α, is derived for the arithmetic mean.

$$P(b_1 < n\lambda\sum_{i=1}^{n}\frac{X_i}{n} < b_2) = 1-\alpha \tag{3.14}$$

Here b_1 and b_2 are the confidence limits from the $\gamma(ns{,}1)$ distribution and they depend on the wanted level of confidence. Solving expression 3.14 for the arithmetic mean gives

$$P\Big(\frac{b_1}{n\lambda} < \sum_{i=1}^{n} \frac{X_i}{n} < \frac{b_2}{n\lambda}\Big) = 1 - \alpha \tag{3.15}$$

where λ and n are known and b_1 and b_2 can be calculated. In this thesis, the length l, of the confidence interval is now calculated to be compared to a maximum allowed length.

$$\frac{1}{n\lambda}(b_2 - b_1) = l \tag{3.16}$$

In the formula above, n is the sample size, which will be given as an input argument, and λ is a Gamma distribution parameter that will be received from *gamfit*. The variables b_1 and b_2 are calculated with the Matlab command *gaminv* that takes in the Gamma parameters s and λ and a level of confidence and returns the inverse of the Gamma cumulative density function (cdf) at the given probabilities.

3.4.1 Reflections on the Confidence Interval

Since the distribution is skewed, and thus asymmetric, the length of the confidence interval that is calculated with this method must be divided to half, in order to guarantee the desired accuracy. A better way to ensure the desired accuracy would have been to compare the interval limits, $\frac{b_1}{n\lambda}$ and $\frac{b_2}{n\lambda}$ directly to reference values.

The way this is currently implemented the length l is estimated and then it is checked if $2*l$ is within the required range. If it is, the sample size is kept unchanged and if it is not, a proportional correction term is calculated and added to the current sample size.

It might be unsuitable to use s and λ in the estimation of the confidence interval. Both these parameters have been estimated based on the distribution of the measured knock intensities. Therefore, they contain some degree of uncertainty. When these estimations are used as if they were true values, the uncertainty is ignored and the sample size needed to achieve a certain accuracy is underestimated.

If too few knock intensity values are used in the estimation of the Gamma distribution, the shape of the estimated distribution might be fairly different from the true one. Hence, the estimated values of s and λ might be far from the true values. If this suspicion is true, this might cause large variations in the length of the confidence interval.

If possible, it would be better if the confidence interval could be calculated from a distribution that does not depend on s or λ but on estimations of them. More will be said about this in Future Work in section 8.2.

Chapter 4

The Controller

The objective is to build a controller that can control the knock intensity and the output torque towards their reference values. The controller will consist of two separate parts, one for the knock intensity control and one for the torque control. While running, it will iterate between them. When one part has reached its desired value the other one will take over and so on until one of the interruption criteria, that will be described in section 4.1, are reached.

Desired values for compression ratio and engine speed are given by the user and these are handled by the ECU. Both these values are kept constant during the knock intensity and torque control.

4.1 The Knock Intensity Controller

The purpose of the knock intensity controller is to regulate the *knock intensity mean* into a predefined allowed range, using the ignition angle as control signal. The range is set by the user in the GUI or the script. To get the *knock intensity mean*, the cylinder pressure is measured. The measurement is chopped up in cycles and each cycle is filtered through a band pass filter that is adjusted to catch the knock frequencies. The filtration is made with a Butterworth filter of order 6 and it is non-causal.

In section 2.3 it was stated that the key frequency of the knock is somewhere between 5 and 10 kHz for SVC. This was confirmed by the theoretical calculations in section 2.6. Therefore, the limits of the band pass filter are set to 5 and 10 kHz respectively for the SVC engine. For the TC engine, the key frequency can not be measured and therefore the filter limits are set to 12 and 16 kHz, to catch the first overtone. After the filtration, the highest knock value from each cycle is found and the *knock intensity mean* is calculated.

The *knock intensity mean* will be compared to a reference value and a proportional correction term will be calculated and added to the current ignition angle.

$$\theta_{ign} = \theta_{ign} + P_{reg}(\text{desired knock intensity} - \textit{knock intensity mean}) \quad (4.1)$$

In the equation above the value of P_{reg} is different for the different engine types.

The new ignition angle will be sent back to the engine. All of this is rather simple to do in Matlab m-files. Hence, the knock intensity controller is implemented as a number of Matlab functions calling each other. A list of the Matlab functions and a short description of each function is found in appendix B.

There are three interrupt criteria for the knock intensity controller.

- The control goal is achieved. That is, the knock intensity is within the given range and the confidence interval equals or is shorter than the desired value.
- The knock intensity is too low. To reach the requested value, the knock intensity controller tries to set θ_{ign} earlier than $35\,^\circ$ before TDC. If this happens, the operating point is not interesting since it will give a lower efficiency than other operating points that causes less knock.
- The knock intensity is too high. When the knock intensity controller wants to set θ_{ign} to after TDC and the knock intensity is still too high, the control is interrupted for safety reasons.

4.2 The Torque Controller

The torque controller can be implemented directly as a PI-controller in Simulink. It is seen in figure 4.1, where the PI-controller is to the left in the figure. The blocks to the right scales the pedal position and limits it to a range between 0 and 100, to prepare it for the ECU. The equation for the controller is

$$\text{Pedalpos} = K_P(M_{ref} - M) + K_I \int_0^t (M_{ref} - M)\, ds + \text{Initial pedalpos}$$

The K_P and K_I parameters have been found by using Ziegler-Nichols method [5]. The `Initial_pedalpos` block gets the initial pedal position from the ECU when the control starts and the controller uses this

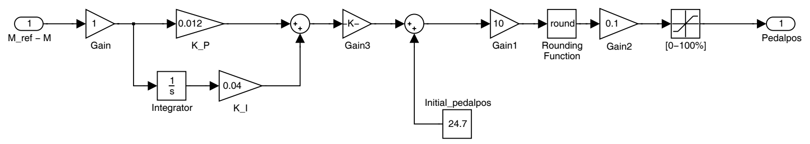


Figure 4.1: The torque control is done by a PI-controller that can be seen in the left part of the figure. The `Initial_Pedalpos` block receives the initial pedal position from the ECU each time the controller is called.

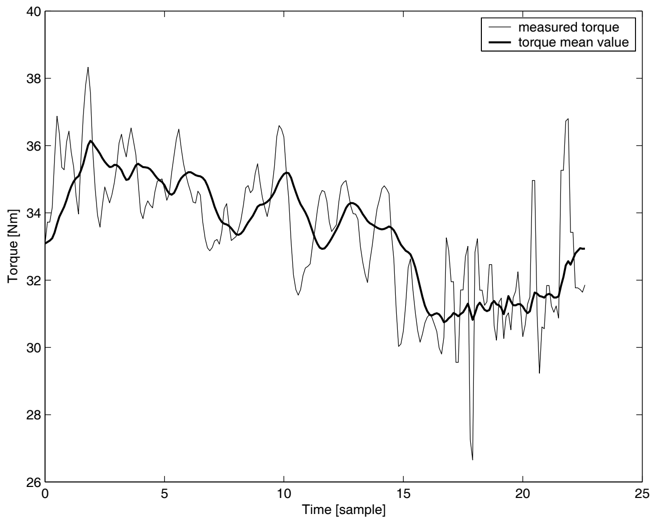


Figure 4.2: The torque value compared to the WLS of 15 samples.

value as a starting point. This decreases the risk of large control signals in the beginning of the control phase and is thus more gentle to the engine than it would be to always start from the same constant value. The risk of anti windup has not been taken into consideration. This is something that should be handled in the future.

There are pulsations in the measured torque. This is due to the combustion in the five cylinders. The pulsations can be considered as noise in the torque signal. To get rid of the noise, a new torque signal is calculated with a continuous mean value calculation of the last samples. This method is called Windowed Least Square (WLS) [7] and works as a filter to smooth the signal. Figure 4.2 shows the measured torque compared to the torque signal calculated with WLS. As can be seen in the plot, the WLS calculation causes a small delay in the torque signal. The interrupt criterion for the torque controller is that the the torque must be within a predefined range.

4.3 Merging the Controllers

When the two controllers have been build and tested separately the next step is to merge them into one controller unit. A convenient way to do this is to run the Matlab functions for knock intensity control from a Stateflow chart which is easily combined with a the Simulink model that handles the torque control.

4.3.1 Stateflow

Stateflow is a part of Simulink, and a Stateflow chart is implemented in a Simulink model in the same way as a subsystem. Inside the chart, Stateflow is a finite state machine, which means that it consists of a number of states and that the currently active state can be changed as a reaction to events. Another way to change the active state is to put conditions on the transactions between them, then the active state will change if the condition is fulfilled. Inside each state there are three 'levels', called *enter*, *during* and *exit*. Actions can be connected to each one of these. Stateflow also has something called junctions. A junction is a point from which there are alternative ways. Junctions must be combined with conditions, and it must be unambiguous which way is the right one.

One of the main benefits with Stateflow is that the chart can communicate easily with both Matlab workspace and Simulink model. To get hold of a workspace variable or a Matlab function from the Stateflow chart `ml.` is written in front of the function or variable name. To read a variable from, or write a variable to the Simulink model, something called a data parameter must be defined in Stateflow. When this is done, an input/output port is created on the Stateflow subsystem block in the Simulink model. This block can now be connected to other blocks just like normal.

4.3.2 The Knock Intensity Control Stateflow Chart

A stripped version of the knock intensity control Stateflow chart can be seen in figure 4.3. In this version, only the names of the states and some of the conditions for transitions are shown. This is to give a general idea of how the chart works. The full chart is found in appendix A.

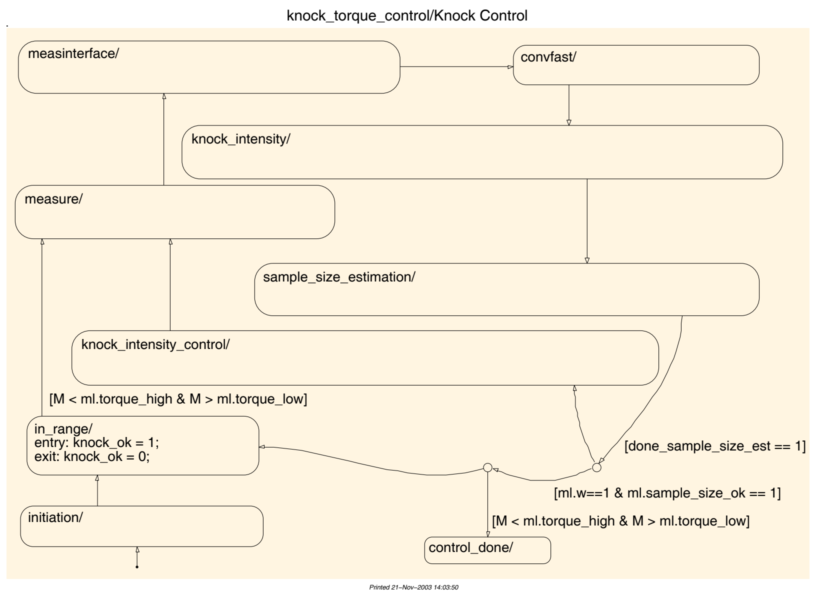

knock_torque_control/Knock Control
measinterface/
convfast/
knock_intensity/
measure/
sample_size_estimation/
knock_intensity_control/
[M < ml.torque_high & M > ml.torque_low]
in_range/
entry: knock_ok = 1;
exit: knock_ok = 0;
[done_sample_size_est == 1]
[ml.w==1 & ml.sample_size_ok == 1]
[M < ml.torque_high & M > ml.torque_low]
initiation/
control_done/
Printed 21-Nov-2003 14:03:50

Figure 4.3: A stripped version of the knock intensity control Stateflow chart.

For each of the Matlab functions in the knock intensity controller, a state is implemented in Stateflow. These states are given the same names as their respective function. When one of these states is activated the function with the same name is run. All functions are listed in appendix B, but the ones that appear in the chart will be shortly described here.

initiation When the controller is called, the chart starts in the state *initiation* where the measuring equipment is initiated. The next state is *in_range*, where the torque is controlled.

measure, measinterface, convfast These states handles the measurement of the cylinder pressure and some preparations of the measured data.

knock_intensity In this state the *knock intensity mean* is calculated and compared to a predefined interval. If the value is within the desired interval, the variable w is set to one and otherwise it is set to zero.

w This variable is set in *knock_intensity* and determines which way to choose at the junction that follows after the statistical calculations in *sample_size_estimation*. When w is zero the *knock in-*

tensity mean is outside the interval and therefore the next state is *knock_intensity_control*.

sample_size_estimation Even if the knock intensity is within the interval in *knock_intensity*, this is not enough to enable the other way at the first junction. The confidence interval calculated in *sample_size_estimation* must also be shorter than a predefined value. If however, both conditions are fulfilled the next junction is reached. This means that both the knock intensity and the confidence interval are ok, and thus it is time to check the torque. If the condition on the branch is fulfilled without the torque having to be adjusted the next state is *control_done*.

sample_size_ok This variable is set to one when the confidence interval is shorter than a predefined value.

knock_intensity_control In this state the new ignition angle is calculated and sent back to the engine. After this, the chart activates the measurement state again and starts over.

control_done This is where the controller is stopped.

in_range This is where the chart ends up if the torque needs to be adjusted. When this state is activated, a Stateflow parameter called *knock_ok* is set to one and when it is deactivated it is set to zero.

knock_ok This parameter is used to communicate with the `Controller Switch` in the Simulink model that can be seen in figure 4.4. When *knock_ok* is set to one the torque controller is active and when it is set to zero it is shut off. The condition for leaving *in_range* is that the torque controller, that is active while this state is active, has controlled the torque into the interval that has been set in the GUI. The consequence of this is an iteration between the knock intensity and the torque controllers.

4.3.3 Reflections over the Knock Intensity Control

One thing that needs to be brought to attention, is that the knock intensity will be controlled, regardless of the size of the confidence interval. This means that the control is sometimes made based on a value that has an uncertainty that is too large. The reason for doing this, is that since it takes more than a minute to make the cylinder pressure measurements, it would be a waste to throw away the new information about the knock intensity. Therefore the knock intensity and the sample size are controlled independently of each other. It is of course important that both values are correct before the algorithm ends.

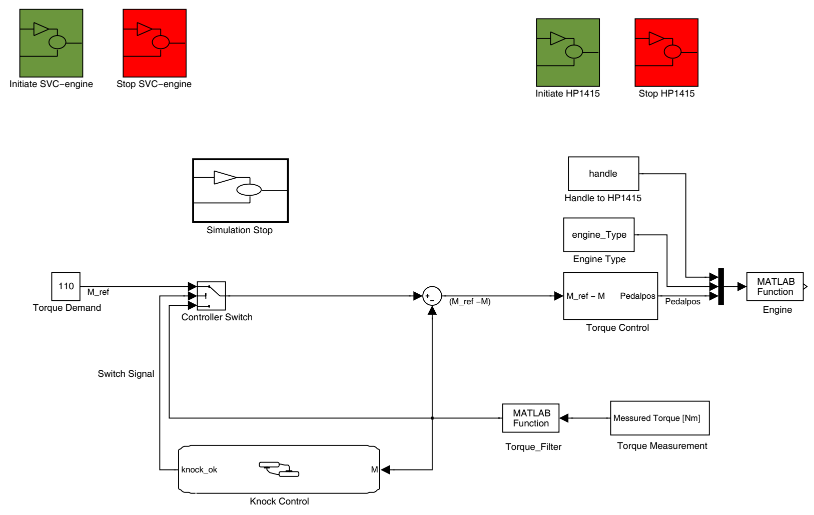


Figure 4.4: The Simulink model used for knock intensity and torque control.

Another thing that should be pointed out is that there are two kinds of intervals for the *knock intensity mean*. One is the confidence interval from section 3.4 and the other one is the allowed range that was introduced in section 4.1. A consequence of having these two different kinds of intervals is that a situation can occur where the *knock intensity mean* is in range, but the desired value is not covered by the confidence interval. The way the knock intensity controller is currently implemented, the control goal is considered achieved in this case. A possible solution would be to let the user set the limits of the confidence interval in the GUI and the script. With that, it would be possible to use only the confidence interval.

4.4 The Controller

The Simulink model for torque control and the Stateflow chart for the knock intensity control are put together to form the complete controller (figure 4.4). The torque controller is put into the subsystem `Torque control`. Blocks are added to handle the torque mean value calculation and the torque-related communication with the ECU.

The `Controller Switch` decides where the torque controller will get its reference value from. When *knock_ok* is set to one, which it according to subsection 4.3.2 will be when the knock controller is resting and the torque controller is running, the reference torque is taken from the box `Torque Demand`. When *knock_ok* equals zero, which it does when the knock intensity controller is running, the reference torque will be taken from the measured torque and hence `M_ref-M` will equal zero. As a result, the input to the torque controller will be zero, and thus the torque controller is shut off. That is, it will have a constant output, based on the sum of the value from the integrator and the value from the `Initial_pedalpos` block.

$$\text{Pedalpos} = K_P \underbrace{(M_{ref} - M)}_{0} + K_I \underbrace{\int_0^t (M_{ref} - M)\, ds}_{constant} + \text{Initial pedalpos}$$

This solution enables iteration between the controllers without having to worry about integrator windup.

The `Simulation Stop` subsystem can be called from different parts of the model when the simulation has to be stopped. The four blocks in the top section of the figure are separate subsystems. They handle the initiation of some parameters in the controller and the measuring equipment.

Chapter 5

Running the Controller

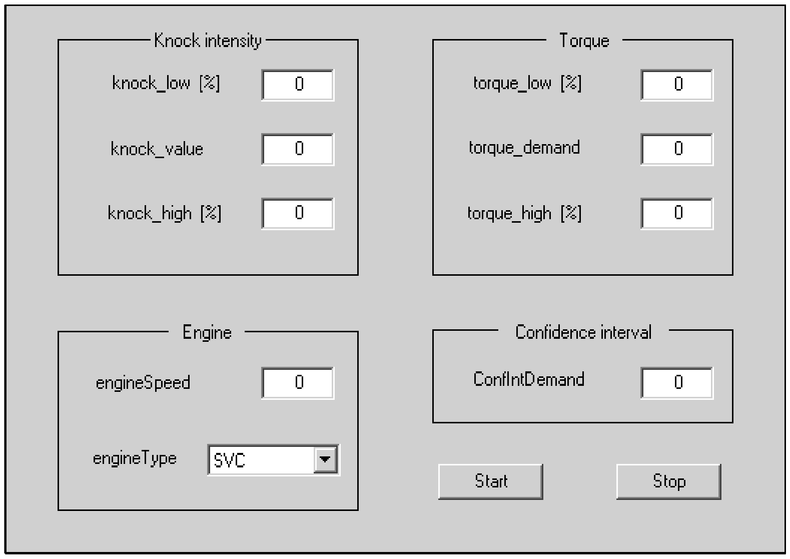
Two methods to run the knock intensity and torque controller have been implemented. The first is to run the controller from a Graphical User Interface (GUI) and the second is to run it from a script. The GUI is used when the purpose is to use the controller in one operating point at the time. In this case, measurements other that the ones internally used by the controller, must be made manually.

The script is used when measurements from several operating points are desired. The script calls the controller in each operating point, and when the control is done it makes the measurements and then stores the data. A manual that contains more information on how to run the controller from the GUI and the script is found in appendix C. The intention is that the manual should be possible to read detached from this report. Therefore, some of the information in this chapter will be repeated there.

5.1 GUI

In figure 5.1 the GUI is shown and as can be seen, it has a number of input boxes. In the boxes *knock_value* and *torque_demand* the desired values are set. These values are what the controller will regulate towards. The desired knock intensity is set in volt. It may be more suitable to have the knock intensity in [Pa], but due to translation problems the voltage from the cylinder pressure charge amplifier is not translated. Also the knock intensity will therefore be in volt.

Since it is difficult to obtain exactly the value that is desired, the control is considered successful when the value is within a given range. Inputs to the boxes *knock_low*, *knock_high*, *torque_low* and *torque_high* are per-



Knock intensity

knock_low [%] 0

knock_value 0

knock_high [%] 0

Torque

torque_low [%] 0

torque_demand 0

torque_high [%] 0

Engine

engineSpeed 0

engineType SVC

Confidence interval

ConfIntDemand 0

Figure 5.1: The GUI used to manage the knock intensity and torque controller.

centages of the respective desired value. For example, if the allowed interval for the knock intensity is from five percent under the desired value to five percent above the desired value, 0.95 and 1.05 should be written in the boxes. In *engineSpeed* the engine speed is inserted in [rpm] and in *engine_type* a choice can be made between SVC and TC engine. In *ConfIntDemand* the maximum acceptable length of the confidence interval is written. This value is also in volt since it is connected to the knock intensities. As a rule of thumb, the maximum acceptable length should be five percent of the largest knock value in an average operating point. The level of confidence is set permanently in the code. to 95 %. In appendix C it is explained where this value can be changed.

The GUI has two push buttons, *Start* and *Stop*. Pressing *Start* executes a callback function that checks if the Simulink model is open and opens it if it is not. It then initiates the Simulink parameters and starts the regulation. Pressing *Stop* stops the simulation and releases the ignition angle, θ_{ign}, so that the knock intensity control is handed over to the ECU. This is a safety precaution, in case the knock intensity gets too high which could seriously harm the engine. The reason why the reaction to a too high knock intensity is manual instead of coded is that this is more flexible, and since the GUI only can handle one operating point at the time, it needs constant supervision anyway.

5.1.1 GUIDE

GUIDE stands for the Matlab Graphical User Interface Development Environment. This program was used to create the GUI. When a GUI is opened in GUIDE it is displayed in a layout editor where it is easy to construct a GUI by dragging components into the layout area. From the layout, two files are automatically generated for each GUI. One `.fig` file that contains all information about the GUI layout and one `.m` file that contains the code that controls the GUI, including callbacks for the components.

5.2 Script

When measurements are desired from several operating points, the script is used. This script is a Matlab m-file that calls the controller and several other functions. The script consists of two parts. In the first part the measuring equipment is initiated and in the second part the operating points are stepped through. In this part, the controller is called and data storing is made after each operating point. Before the script is started, the first operating point must be reached manually.

5.2.1 Defining the Operating Points

Each operating point is defined by four parameters: demanded torque, demanded compression ratio, demanded knock intensity and demanded engine speed. In the script, the engine speed is constant, and the other three parameters are represented as vectors. The values of these three parameters should be implemented into the vectors in ascending order because the script is implemented to handle the operating points in a certain order. The torque and compression ratio vectors are read forward, starting with the lowest value and ascending. The knock intensity vector is read backwards, starting with the highest value and descending. The number of the current position in the knock vector is referred to as Index of Knock (I.K). The corresponding values for the torque and the compression ratio vectors are called Torque Index (T.I) and Compression Index (C.I).

5.2.2 Iterating Through the Operating Points

The iteration between operating points is handled by three nested loops. The outer loop steps through the torque vector, the middle loop steps through the compression ratio vector and the inner loop steps through the knock intensity vector.

Figure 5.2 shows a flow diagram of how the script iterates through the

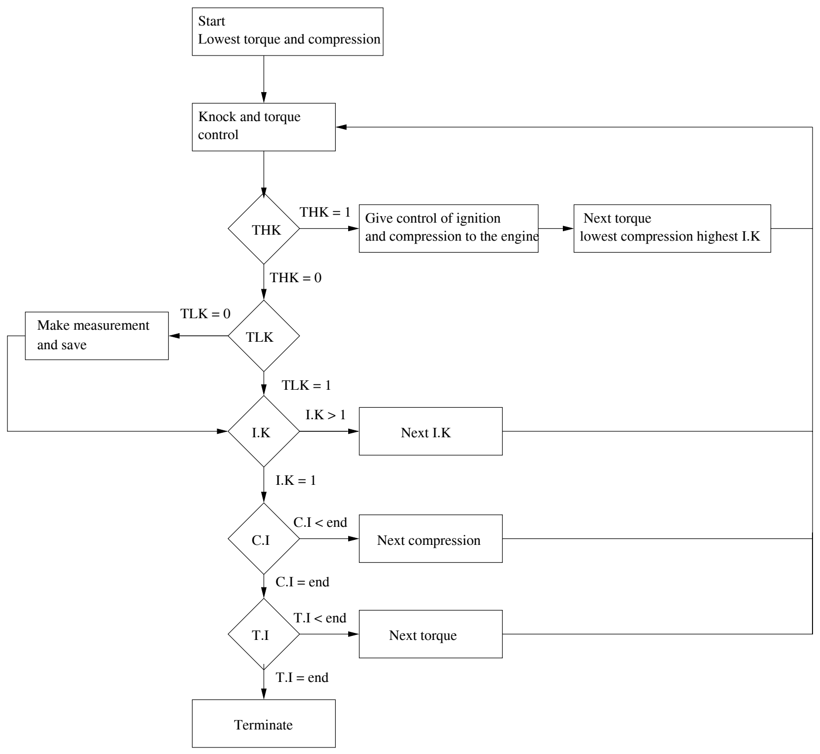


Figure 5.2: A flow diagram describing how the script iterates through all operating points and avoids those with to high knock. THK and TLK are short for To High Knock and To Low Knock respectively. I.K, C.I and TI are short for Index of Knock, Compression Index and Torque Index.

operating points. The first operating point, meaning the lowest torque and compression ratio and the highest knock intensity, is reached manually. This is represented by the block on top of the page. The next block shows how the controller is called and the knock intensity and torque are controlled. If the knock is too high even when the ignition is set to TDC, the control is interrupted and the flag *Too High Knock* (THK) is set to one. If the knock is too low, even when the ignition angle is set to 35 °, the control is also interrupted and the flag *Too Low Knock* (TLK) is set to one.

The block called *THK* represents a check of the value of the flag THK. If the flag is set to one, the script returns all control to the ECU. After that, the script steps up the torque vector and calls the controller again with the lowest compression ratio and highest knock intensity.

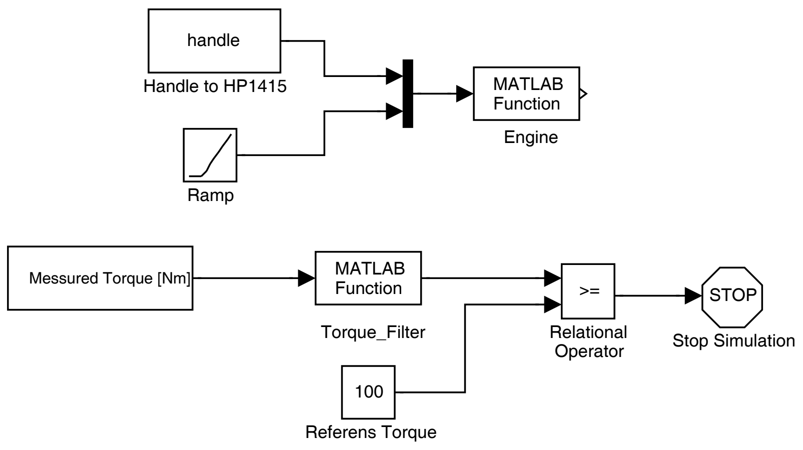


Figure 5.3: The Simulink model that handles the torque ramping.

If the flag THK is set to zero, the value of the flag TLK is checked. This is represented by the block called *TLK*. If TLK is also set to zero, the knock intensity value is within range and a measurement is made and stored. If TLK is set to one, no measurement is made. After the flag TLK has been checked, independently of if a measurement is made or not, the next thing that happens is a change of operating point. Since the knock intensities are implemented in the inner loop, the knock intensity vector is stepped through first. The block *I.K* represents a check of the variable I.K, to see where in the knock intensity vector the current value is placed. If I.K is greater then one, there are still more values left in the vector. In this case, the next knock intensity is chosen and the controller is called in this new operating point. If I.K equals one, the last value in the knock intensity vector has been reached. When this happens, the value of C.I is checked in the same way. This is seen as the block *C.I.* If the last value in the compression vector is reached, the value of T.I is checked. When the very last operating point has been reached the iteration terminates.

5.2.3 Torque Ramping

When going from one value to the next in the torque vector, it is important that the change is gentle for the engine. Therefore a Simulink model called `Torque Ramp` is built. `Torque Ramp` can be seen in figure 5.3. The model reads the next torque from the torque vector and writes it in the box called `Reference Torque`. It then sets the current throttle angle as start value for the ramp. The throttle angle and hence the torque is increased slowly by the ramp. The torque is measured from the engine and compared to the `Reference Torque` and as soon as the measured torque equals the `Reference Torque` the ramping stops.

Chapter 6

Implementation in the Engine Laboratory

The real challenge in this thesis work has been to implement the controller in the engine laboratory and to make it work together with the the measuring equipment and the computers that are needed to communicate with the ECU. The control room in the engine laboratory is

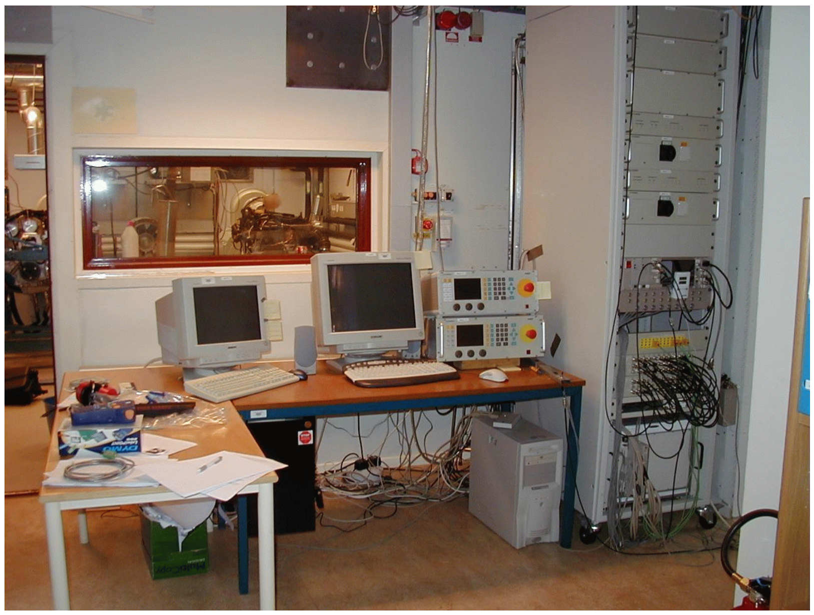


Figure 6.1: A view of the engine laboratory

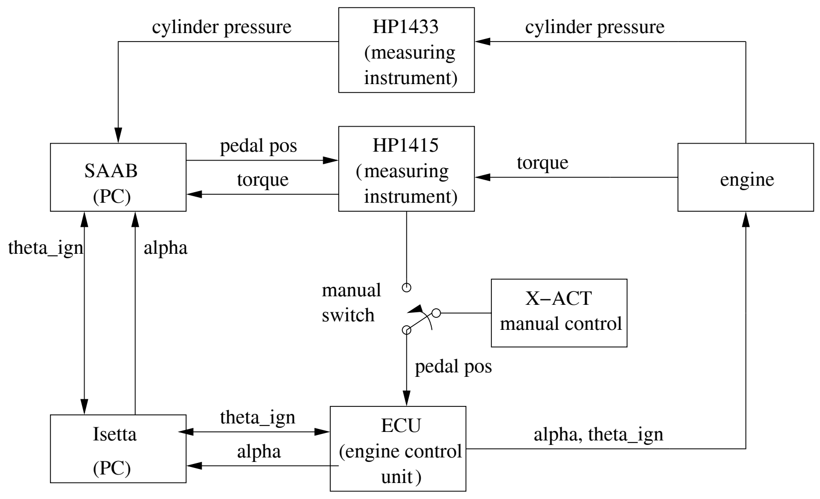


Figure 6.2: A schematic picture of the components that are used in the laboratory.

shown in figure 6.1, and a schematic picture of the components involved in figure 6.2. The intention here is to give an idea of the complexity of the environment in which the controller was implemented.

A short description of each component:

SAAB A PC on which the controller is implemented and run. SAAB communicates with the measuring equipment and Isetta.

Isetta A PC that communicates with the ECU using software from the engine supplier, FGP-S [1].

ECU The Engine Control Unit, ECU, controls all the parameters in the engine except for the ones that the knock intensity or torque controller temporarily takes control of.

HP1433 A fast measuring instrument. It is adjusted to measure the cylinder pressure with a sample frequency of 192000 Hz. It is necessary to sample fast in order to get the details of the cylinder pressure curve.

HP1415 A slow measuring instrument. It measures many different parameters but the one that is of interest here is the torque, which is sampled with a sample rate of 20 Hz. This instrument also has analogue voltage exits. One of these exits is used to send the pedal position to the ECU.

[1]Fiat GM Powertrain Sweden

Engine An SVC or a TC engine.

X-ACT This is a device for manual control of the engine break and the pedal position. When the switch is flipped, the control of the throttle angle is turned over to SAAB that uses HP1415 to communicate with the ECU.

Chapter 7

Evaluation and Results

In order to make sure that everything works the way it is supposed to, first the separate parts, and then the entire system was evaluated. The torque and knock controllers were both implemented directly in the laboratory and were thus tested during the construction. After they were merged together, the controller was tested again as a whole.

7.1 Results from the Knock Controller

The knock intensity controller has been tested continuously during the development phase. Therefore, the final test is made in only one operating point, which is chosen arbitrarily among those known by experience not to cause too high knock. The variable *knock_value* is set so that it will be reachable and the limits are set to values that have been tried out empirically. *ConfIntDemand* is set according to the rule of the thumb given in section 5.1. The values that were used in this run can be seen in table 7.1. The controller is now expected to get the knock intensity within the range and the length of the confidence interval shorter than the maximum value, and then stop.

torque	110 Nm
knock_value	0.0018 V
ConfIntDemand	0.0005
engineSpeed	2000 rpm
knock_high	1.10
knock_low	0.90
engine_Type	TC

Table 7.1: The values that were used in this run.

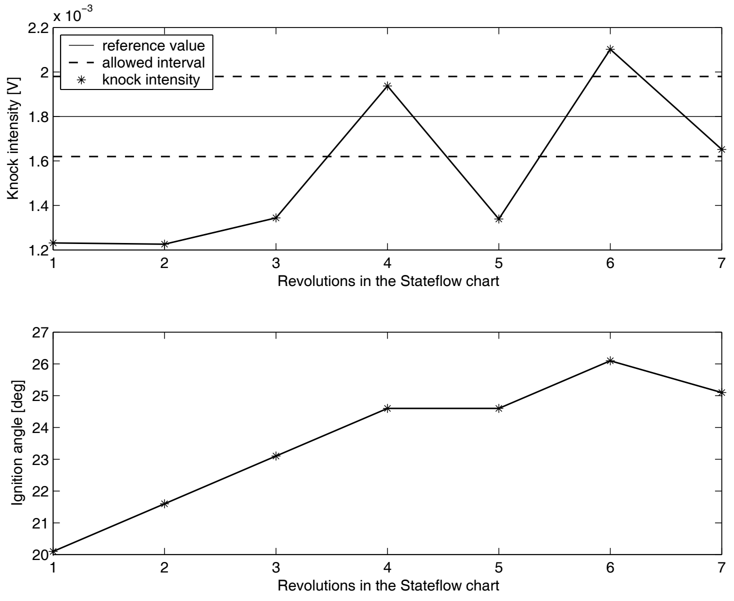


Figure 7.1: Knock intensity and ignition angle. The thin line is the goal value and the dashed lines is the accepted range.

In figure 7.1 plots of the knock intensity and the ignition angle are shown. It can be seen that the signals are discrete, which is natural since the values are only calculated once per revolution in the Stateflow chart.

The first ignition angle value in the plot is set by the ECU before the knock intensity controller takes over the control. The first knock intensity is the result of this original ignition angle. The second ignition angle is calculated by the knock intensity controller as a result of the first knock intensity. This continues until one of the interruption criteria from section 4.1 are reached. The thin line is the reference value of the knock intensity and the area between the dashed lines is the acceptable range.

In figure 7.1, it can also be seen that in revolutions 1, 2, 3 and 5 the knock intensity is too low and thus the ignition angles in revolutions 2, 3, 4 and 6 are each higher than the one prior to them. In revolution 4, the knock intensity is within the given range but the confidence interval in figure 7.2 is too long and therefore the control will not terminate here. In revolution 7 both the knock intensity and the confidence interval have reached satisfying values.

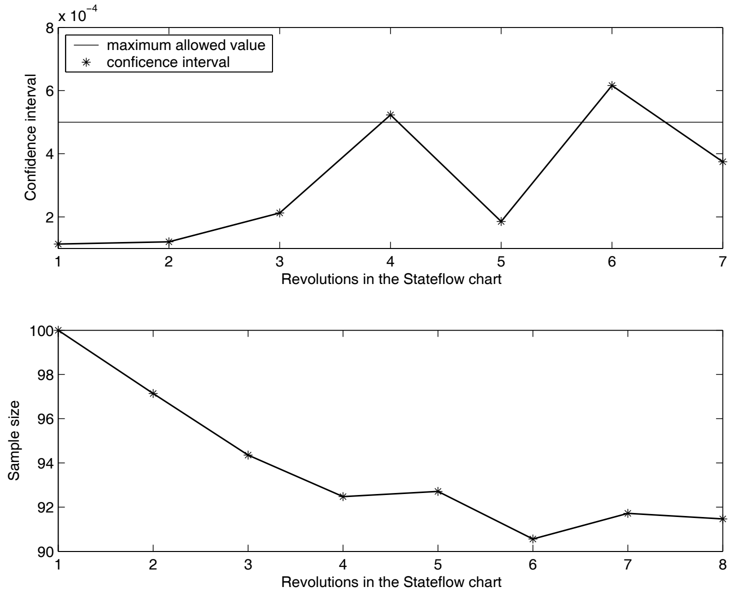


Figure 7.2: Confidence interval and sample size.

Figure 7.2 shows the confidence interval and the sample size. The variable *sample_size* is initiated to 100, and this gives the first confidence interval value from which a new sample size is calculated and so on. The goal of the knock intensity controller is not considered achieved unless the confidence interval is shorter than the maximum value that can be seen as a thin line in the plot.

A comparison of the figures 7.1 and 7.2 suggests that a very strong relation between the knock intensity and the confidence interval exists. It is also noticed that, as a percentage, the fluctuations in the length of the confidence interval is larger than the fluctuations in the knock intensity. This is an unexpected result and it strongly indicates that the suspicion from section 3.4 is correct. The estimations of the Gamma distribution is based on too few values. Hence, the estimated s and λ values contain a large uncertainty which will in turn affect the length and accuracy of a confidence interval based on these values.

7.2 Results from the Torque Controller

The torque controller has been tested in the same operating point as the knock controller. The allowed range for the torque is set as a smaller

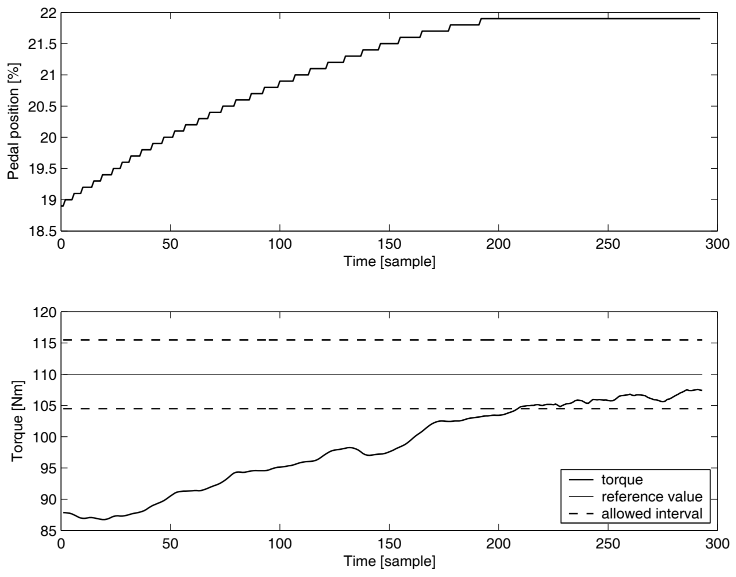


Figure 7.3: Pedal position and torque.

percentage, (0.95, 1.05), than for the knock intensity. This is because the torque is less stochastic.

The lower image in figure 7.3 shows the filtered torque. In the upper image the pedal position can be seen. The pedal position is the control signal that is sent from the torque controller to the ECU. The ECU interprets the pedal position as a measure of how much torque that is desired. It then changes the throttle angle to try to achieve this desired torque. At approximately sample 200 the torque is in range and the pedal position stops. In spite of this, the torque value keeps increasing. This is because of dynamics in the engine, for instance in the intake manifold. The measurements in the plot are not made in real time but in 'Simulink time' and therefore they are plotted against sample number instead of time. A fixed step size was used in Simulink.

7.3 Evaluation of the Controller

The complete controller was tested for three different cases that correspond to the three interruption criteria of the knock intensity controller that were given in section 4.1.

torque_demand	110 Nm
knock_value	0.0018 V
ConfIntDemand	0.0005
engineSpeed	2000 rpm
knock_high	1.10
knock_low	0.90
engine_Type	TC
torque_high	1.05
torque_low	0.95

Table 7.2: The values that were used in case one.

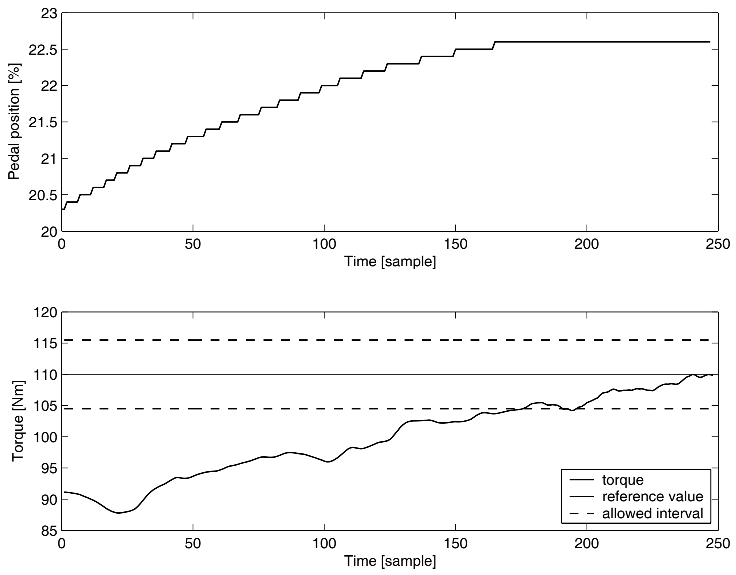


Figure 7.4: Case one. Torque and pedal position during the torque control.

7.3.1 Case One

In case one, the relation between the operating point and the desired knock intensity were such that the control goals could be reached. The values that were used in this run can be seen in table 7.2. The torque control and the pedal position are presented in the same way as before. They are plotted in figure 7.4. The torque starts at at a value that is too low. It is then controlled with the pedal position into the allowed interval. Around sample 160 the interval is reached and the pedal position stays constant. The torque keeps increasing for approximately 100 samples. This is due to engine dynamics.

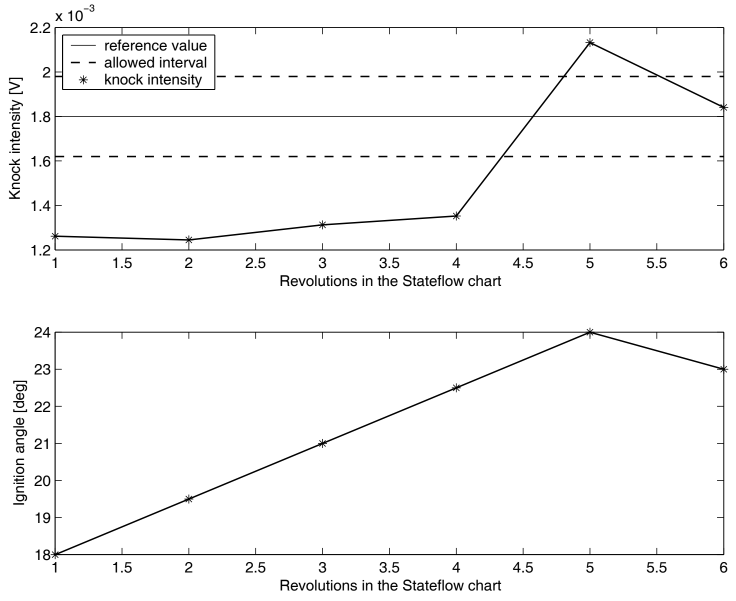


Figure 7.5: Case one. Knock intensity and ignition angle.

After the torque controller is done, the knock intensity controller takes over. A plot of the knock intensity and the ignition angle can be seen in figure 7.5. The knock intensity starts at a value that is too low, and the ignition angle is increased to get the knock intensity within the allowed interval. When this is achieved, the control is interrupted. Figure 7.6 shows the sample size and the length of the confidence interval. In the first revolutions, the confidence interval is much shorter than necessary. Therefore, the sample size is adjusted until the length of the confidence interval is just under the maximum allowed value.

In figure 7.7 the torque and throttle angle for the whole control sequence are shown. This torque measurement is not filtered with WLS and therefore, it is not as smooth as the plots from the torque control. Based on this figure some phenomena will be explained that will be the same for all plots of this kind. At approximately 22 seconds, a manual switch is flipped,which causes the torque control to switch over from X-ACT to SAAB as was explained in section 6. This can be seen as a discontinuity in the throttle angle and the torque. A few seconds later the controller is activated and the torque control starts. At around 40 seconds the desired value is reached and the knock controller takes over. The throttle angle and thus the torque are now still except for

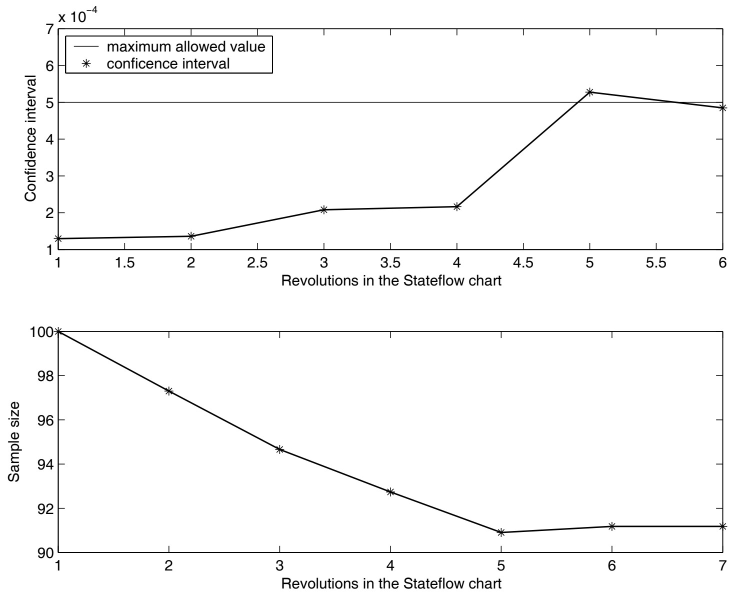


Figure 7.6: Case one. Confidence interval and sample size.

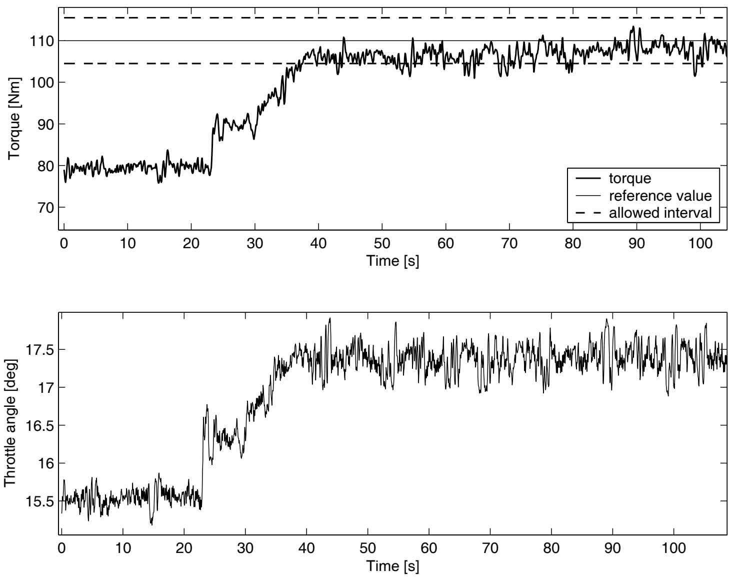


Figure 7.7: Case one. Torque and throttle angle during the whole control time.

torque_demand	110 Nm
knock_value	0.0001 V
ConfIntDemand	0.0005
engineSpeed	2000 rpm
knock_high	1.10
knock_low	0.90
engine_Type	TC
torque_high	1.03
torque_low	0.97

Table 7.3: The values that were used in case two.

some noise. When the knock intensity control is done, the torque is checked again. Since it is within the range, the control is finished.

The interruption criterion for the torque controller is to stop as soon as the torque is within range. The torque then stays constant on this value during the knock intensity control. There is a wish to make the interval as small as possible to make sure that the torque is close to the reference value. When data from case one was regarded it was established that the torque interval was wider than necessary so a shorter interval was tried for case two.

7.3.2 Case Two

In case two, the desired knock intensity is set very low compared to the knock intensities that can be expected in this operating point. Therefore the desired intensity will not be reached. The values that have been used in this run can be seen in table 7.3.

As in case one, the torque starts at a value that is to low, and is adjusted by an increase in the pedal position. This can be seen in figure 7.8. Figure 7.9 shows the knock intensity and the ignition angle. The knock intensity is much higher than the reference value. Therefore the ignition will be set later and later. When the ignition angle is set to zero, the knock intensity control is interrupted. The confidence interval and the sample size can be seen in figure 7.10. The confidence interval is much shorter than it needs to be and therefore the sample size is decreased. Figure 7.11, shows the torque and the throttle angle over the whole control sequence. Between approximately 25 and 140 seconds the knock intensity controller is running. Since the ignition angle is set back quite a lot, this will also affect the torque. At 140 seconds the interrupt criterion is reached and thus the ignition angle is released. As this happens, the torque value returns to what is was

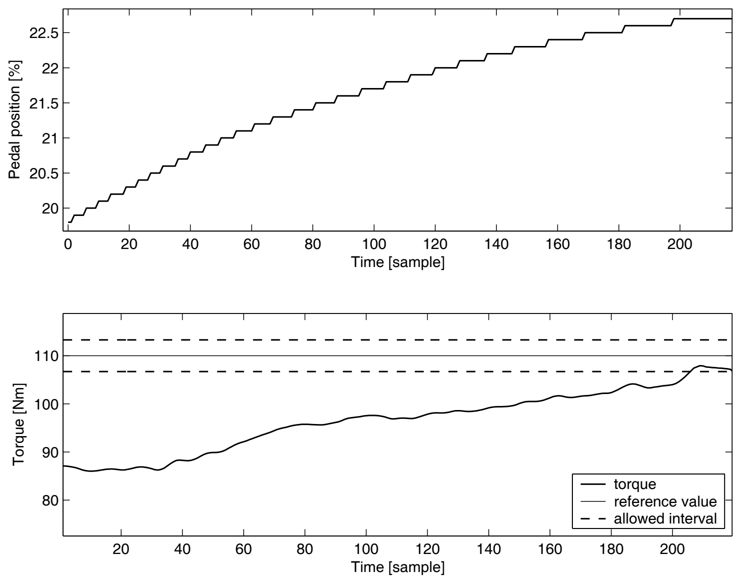


Figure 7.8: Case two. Torque and pedal position during the torque control.

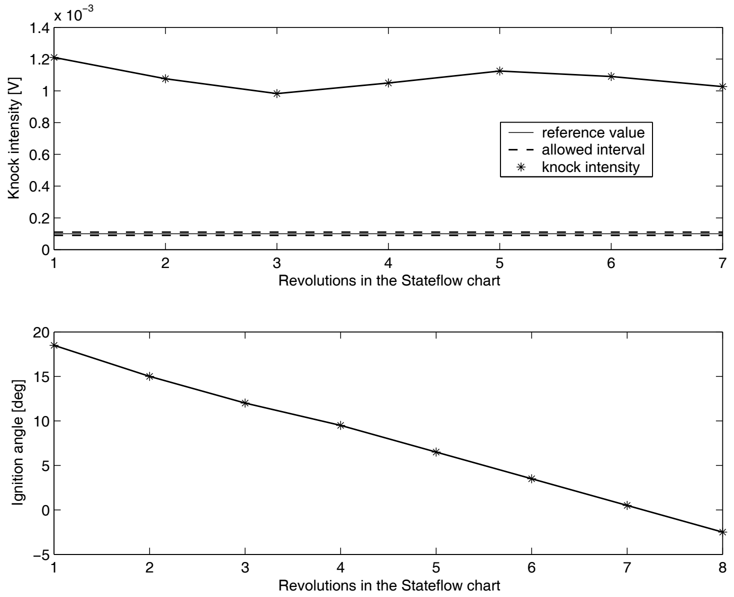


Figure 7.9: Case two. Knock intensity and ignition angle.

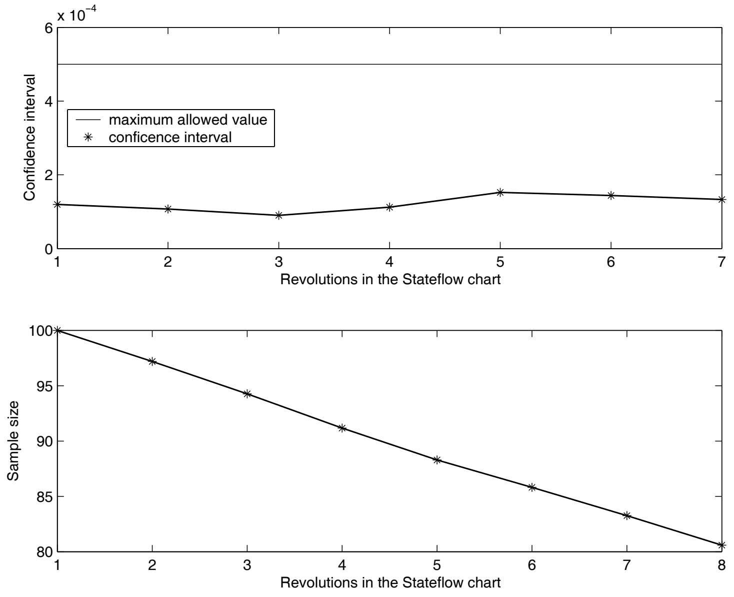


Figure 7.10: Case two. Confidence interval and sample size.

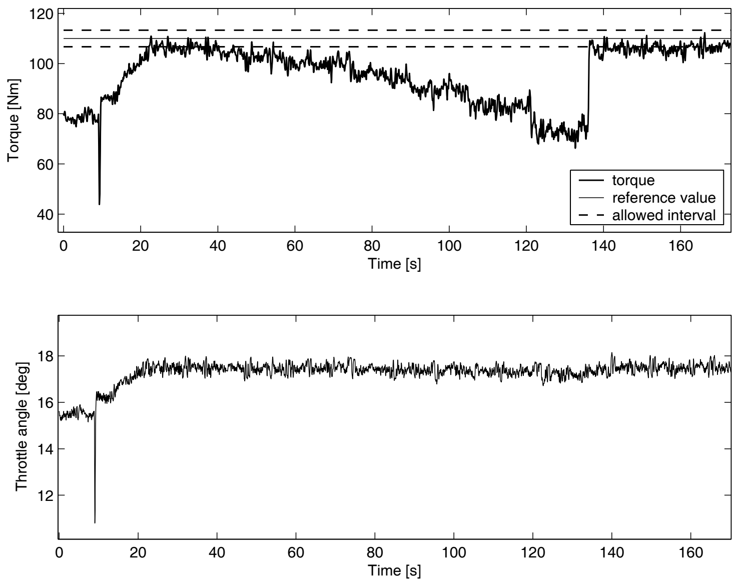


Figure 7.11: Case two. Torque and throttle angle during the whole control time.

torque_demand	30 Nm
knock_value	0.0012 V
ConfIntDemand	0.0005
engineSpeed	2000 rpm
knock_high	1.080
knock_low	0.92
engine_Type	TC
torque_high	1.03
torque_low	0.97

Table 7.4: The values that were used in case three.

before the control of the ignition angle was taken over. The big dip at approximately 10 seconds is caused by the same switch that was mentioned in case one.

7.3.3 Case Three

In the third case the desired knock intensity is set very high compared to the the knock intensities that can be expected in this operating point. Therefore, it will not be reached. In this case the control is interrupted when the ignition angle is set earlier than $35\,^{\circ}$ before TDC. The values that are used in this run can be seen in table 7.4. Figure 7.12 shows how the torque starts at a value that is too high and is adjusted by a decrease of the pedal position. The knock intensity and ignition angle are shown in figure 7.13. Here it is seen how the ignition angle is set earlier and earlier, to adjust the knock intensity towards the reference value. As in case two, the confidence interval is much shorter than it needs to be and therefore the sample size is reduced. This can be seen in figure 7.14. Figure 7.15 shows how the torque is first controlled down to the reference value. At approximately 30 seconds, the torque control stops and the knock intensity controller takes over. During the knock intensity control the torque increases slightly due to the changes in the ignition angle.

7.4 Evaluation of the Script

When the controller had proved to work when used on a single operating point at a time from the GUI, the script was tested on handling several operating points.

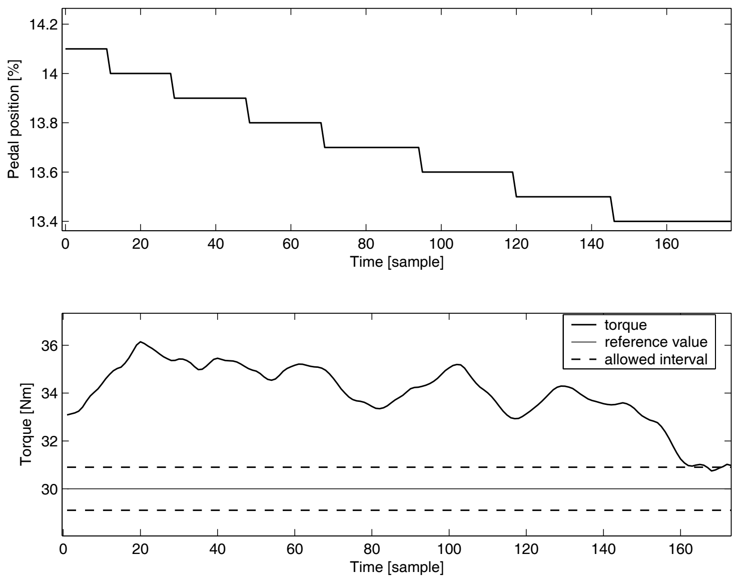


Figure 7.12: Case three. Torque and pedal position during the torque control.

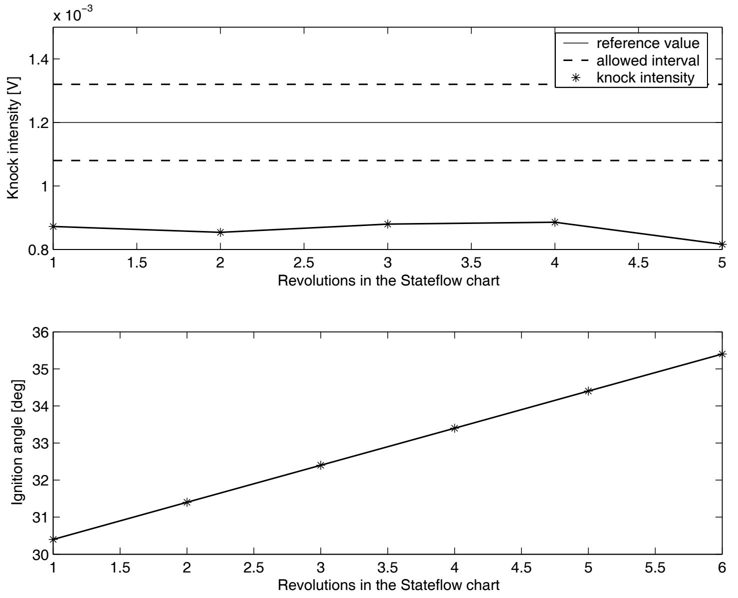


Figure 7.13: Case three. Knock intensity and ignition angle.

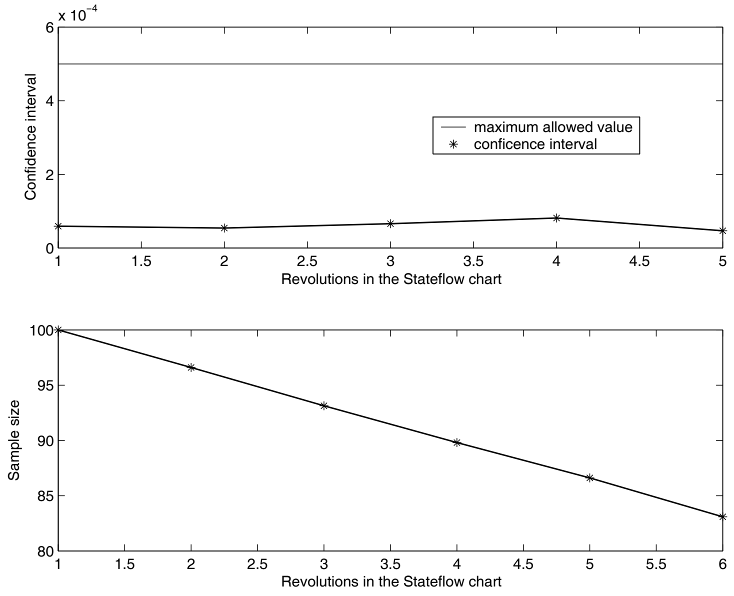


Figure 7.14: Case three. Confidence interval and sample size.

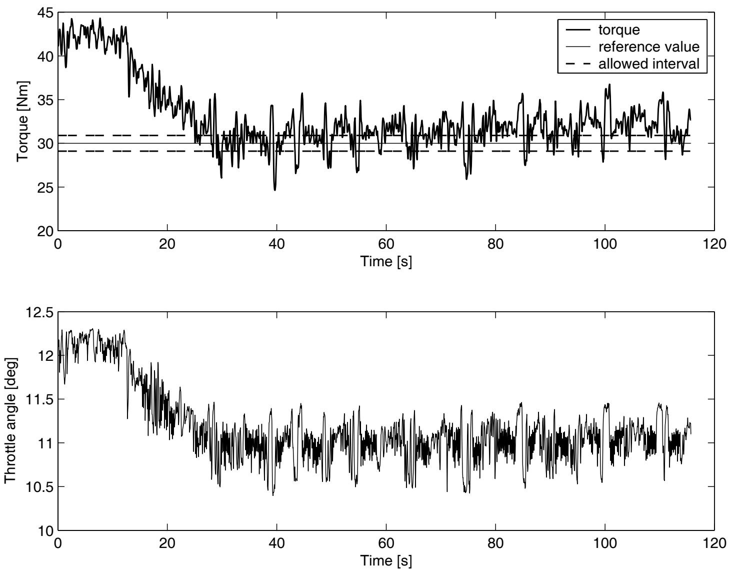


Figure 7.15: Case three. Torque and throttle angle for the whole control time.

7.4.1 Evaluation Problems

When the script was to be evaluated, the SVC engine had a breakdown and thus the tests had to be performed on a TC engine. The greatest difference between the engines is that the TC does not have variable compression ratio. Therefore, the segments of the script concerning the compression ratio could not be tested.

Unfortunately something went wrong with the torque measurement in this case. Therefore no plot of the torque can be shown. However, no problems were noted from the torque controller during this run.

7.4.2 Results from the script

In this run, three reference knock intensities and two reference torque values are used. Combining these gives six operating points. The first operating point constitutes of the lowest torque and the highest knock intensity. In the following operating points, the reference torque is kept and the reference knock intensities are stepped through in descending order. After the last knock intensity is reached, the torque is changed and the knock intensity vector starts over. The values that are used in this run can be seen in table 7.5.

Figure 7.16 shows the knock intensity and the ignition angle. As before, the thin line is the reference value for the knock intensity and the dashed lines are the allowed range. The changes in desired value for the knock intensity are seen as 'steps' in the dash- dotted line. During the first three steps the torque is set to the lower value and then it is changed. In the plot, this change occurs after revolution 23.

Each time the knock intensity has been successfully controlled, the torque is checked. If the torque is not in range, it is adjusted and after that the knock intensity controller is called again. This happens in rev-

torque_demand	90, 110 Nm
knock_value	0.0018, 0.0015 0.0012 V
ConfIntDemand	0.0005
engineSpeed	2000 rpm
knock_high	1.10
knock_low	0.90
engine_Type	TC
torque_high	1.05
torque_low	0.95

Table 7.5: The values that were used in the script evaluation.

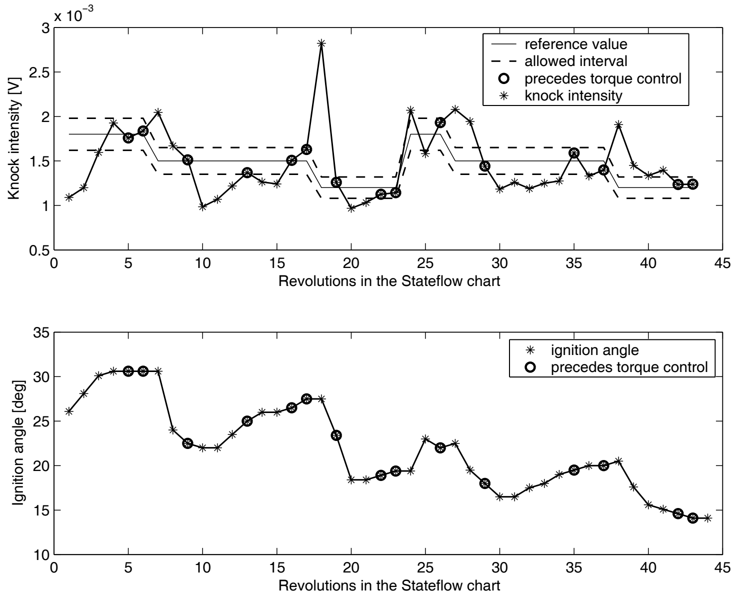


Figure 7.16: Knock intensity and ignition angle when the controller is run from the script. After revolution 23 the reference torque is changed from 90 to 110 Nm.

olutions 5, 9, 13, 16, 19, 22, 29, 35 and 42. Revolution 4 is not included here because the confidence interval is to large and therefore the torque controller is not called (figure 7.17). If the torque is in range, the script changes the operating point. This happens after revolutions 6, 17, 23, 26 and 37. All the knock intensities that are followed by torque control are marked with a circle in the plot. Controlling the torque means adjusting the air mass flow past the throttle, and this affects the knock intensity. Hence the unexpected values of the knock intensity in several of the revolutions that follow directly after the torque has been controlled.

When the operating point is changed, the ignition angle is released. Hence, the first ignition angle in each new operating point is set by the ECU. This explains why the ignition angle in revolution 27 is higher than the one in revolution 26 even though the knock intensity in revolution 26 is above the reference value. In revolutions 7, 18 and 24, the ignition angle is exactly the same as in the previous revolution. This could be because the ignition angle is not released, due to a communication problem between SAAB and Isetta.

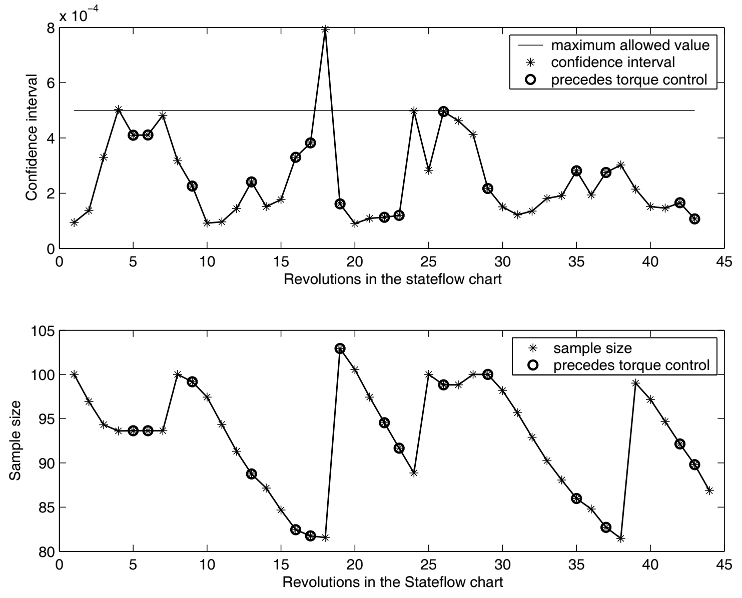


Figure 7.17: Confidence interval and sample size when the controller is run from the script.

When comparing figure 7.17 to figure 7.16 the reflections from section 7.1 are seen even more clearly. The correlation between the knock intensity and the confidence interval is much higher than expected and as a percentage, the fluctuations in the confidence interval are larger than for the knock intensity. A possible explanation for this is that the estimation of the parameters of the Gamma distribution are based on too few values. This would make the estimations of the parameters s and λ extremely sensitive to changes in the knock intensity. Hence, the strong correlation between the knock intensity and a confidence interval calculated from s and λ.

This problem can be handled by increasing the number of values that the estimation is based on, by including old measurements. After a small change in the ignition angle, the distribution is expected to have approximately the same appearance as before. Therefore, including measurements from previous ignition angles offers a possible solution.

Another possible strategy is to derive a distribution that is meant to use estimations of the parameters and is independent of the true values of s and λ. This distribution can then be used for a more accurate calculation of the confidence interval.

Chapter 8

Conclusion

8.1 Summary

The main part of the two goals that were set in the beginning of the thesis have been reached. A controller has been built that uses the ignition angle to control the knock intensity and the throttle angle to control the torque. The knock intensity controller is built in Matlab and consists of a number of m-files. The torque controller is built in Simulink. These two controllers are merged together by using Stateflow. A GUI has been implemented that calls the controller in one operating point at a time. A script has been written that can handle a large number of operating points. The script has also functions that supervise the knock intensity and makes sure that the knock intensity does not get too high.

As could be read in subsection 7.4.1 the script could not be evaluated on an SVC engine because of its valve brake-down. This is however not a big problem since the results from the SVC engine can be expected to be the same as those from the TC engine. There would be more operating points to step through, but this would not affect either one of the controllers.

The results from the confidence interval follows the knock intensity closely. It is also noted that, as a percentage, the fluctuations in the length of the confidence interval are larger than the fluctuations in the knock intensity. A possible explanation for this is that the estimations of the parameters of the Gamma distribution are based on too few values. This would make the estimated parameters s and λ extremely sensitive to changes in the knock intensity. Hence, the strong correlation between the knock intensity and a confidence interval calculated from s and λ. The fact that the variables s and λ are used as true variables even though they are estimated causes an underestimation of the number of cycles that are needed to get the desired certainty for the *knock intensity mean*.

In some of the result plots for the knock intensity controller there are big leaps in the knock intensity even though the ignition angle is kept constant. This is believed to be caused by the too large insecurity in the *knock intensity mean* caused by the problems with the confidence interval. Since too few values are used in the mean value calculations, the *knock intensity mean* is still stochastic.

The knock intensity controller is very slow, as it takes more than a minute to make a measurement and calculate the *knock intensity mean*.

8.2 Future Work

The first thing that should be done is to complete the evaluation of the script by running it on an SVC engine with the compression ratio included.

The delay in the torque control is caused by engine dynamics and the fact that the torque signal is filtered. This delay is currently compensated by the interruption criteria for the torque controller. Since the control is interrupted as soon as the allowed interval is reached the delay is used to get closer to the reference value. This is not the best way to handle the delay but it was still chosen because of its simplicity. In the future it is better to consider the delay in the torque controller.

When considering the results from the script it is noted that the iteration between the controllers is not working optimally. Since the changes in the control signal of one controller affects the output signal of the other controller, the knock intensity and torque controllers sometimes obstruct each other. This effect can be reduced if the torque controller is run each time a new ignition angle is set by the knock intensity controller.

The risk of anti windup for the torque controller is not taken into consideration. This has not caused any problems in the runs that have been made but should be handled in the future.

The conclusion is that the basic constructions of the controller and the script are filling their functions but that there are several improvements left to be done. These improvements are listed here:

- Making the controller faster. There are many different ways to speed up the controller, such as trimming the code or to use a faster computer. As it is now, the knock intensity control is disturbingly slow. It takes more than a minute for the knock intensity controller to make all the calculations that are needed in order to compute *knock intensity mean*.

- Increasing the intelligence. Perhaps there are other situations than the ones that are identified in the current version of the program that should be noticed and handled.
- A more continuous torque control. Instead of first running one controller until it reaches its goal value and then check the status of the other controller, the control would probably be faster and more stable if the torque was controlled after each time a new ignition angle is set.
- Increasing the number of values that the estimation of the Gamma distribution parameters are based on. This would hopefully make the length of the confidence interval less dependent of the current knock intensity.
- A continuous handling of the measured cylinder pressure and continuous calculation of the *knock intensity mean* would enable the knock intensity controller to run in real time.
- Changing the way that the confidence interval is compared to a reference value. As it is now the length of the confidence interval is calculated and compared to its desired length. Instead, it would be better to compare the interval limits to reference values.
- The level of confidence 1-α should be implemented as a variable in the GUI and the script. As it is now, the level of confidence is permanently set to 95 % and can only be changed by changing the code.
- The two different kinds of interval for the *knock intensity mean* should be replaced by only one kind of interval. This can be arranged by letting the user set the limits of the confidence interval in the GUI and the script.
- To derive a method for the calculation of the confidence interval that is not based on the true values of s and λ but on their estimations.
- To eliminate the risk of anti windup that currently exists for the torque controller.

References

[1] A Maximum Likelihood Estimator for the Gamma Distribution. Internet, June 2003. Address http://www-mtl.mit.edu/CIDM/memos/94-13/subsection3.4.1.html.

[2] Saab Variable Compression. Internet, June 2003. Address http://media.gm.com/events/00/geneva/e/saab.htm.

[3] M. Dwass. *Probability and Statistics*. W. A. Benjamin, Inc, New York, USA, 1970.

[4] L.-A. Engström. *Fysikaliska Principer*. Linus och Linnea AB, Linköpings universitet, Linköping, Sweden, fourth edition, 1994.

[5] T. Glad and L. Ljung. *Reglerteknik. Grundläggande teori*. Studentlitteratur, Lund, Sweden, 2nd edition, 1989. In Swedish.

[6] B Grandin. *Knock in Gasoline Engines*. Chalmers reproservice, Göteborg, Sweden, 2001.

[7] F Gustafsson. *Adaptive Filtering and Change Detection*. John Wiley & Sons, Ltd, Chichester, England, 2000.

[8] L. Nielsen and L. Eriksson. Course Material Vehicular Systems. Linköping, Sweden, 2002. Course material, Linköpings Universitet, Sweden.

[9] C. Nordling and J. Österman. *Physics Handbook*. Studentlitteratur, Lund, Sweden, fourth edition, 1980.

[10] L. Råde and M. Rudemo. *Sannolikhetslära och statistik för teknisk högskola*. Studentlitteratur, Lund, Sweden, 1992.

[11] R Stone. *Introduction to Internal Combustion Engines*. The Macmillan Press ltd, London, UK, 2nd edition, 1992.

Notation

Variables and parameters

V_d	Displacement volume
V_c	Clearance volume
B	Cylinder bore
θ	Crank angle
θ_{ign}	Ignition angle
r_c	Compression ratio
L	Piston stroke
f	Resonance frequency of the cylinder
γ	Ratio of specific heat
r_a	Average radius of cylinder
k_b	Boltzmann constant
T	Average temperature of the gas near TDC
μ	Molecular mass
b_1, b_2	confidence limits
M	Torque
c	Speed of sound
N	Engine speed
P	Probability
s	Shape parameter for Gamma distribution
λ	Scale parameter for Gamma distribution
l	Length of confidence interval

Abbreviations

TDC	Top Dead Center
BDC	Bottom Dead Center
SVC	Saab Variable Compression
pdf	Probability Density Function
cdf	Cumulative Density Function
SI	Spark Ignited
ML	Maximum Likelihood
MLE	Maximum Likelihood Estimation
WLS	Windowed Least Square

Appendix A

The Stateflow Chart

On the next page the complete Stateflow chart can be seen.

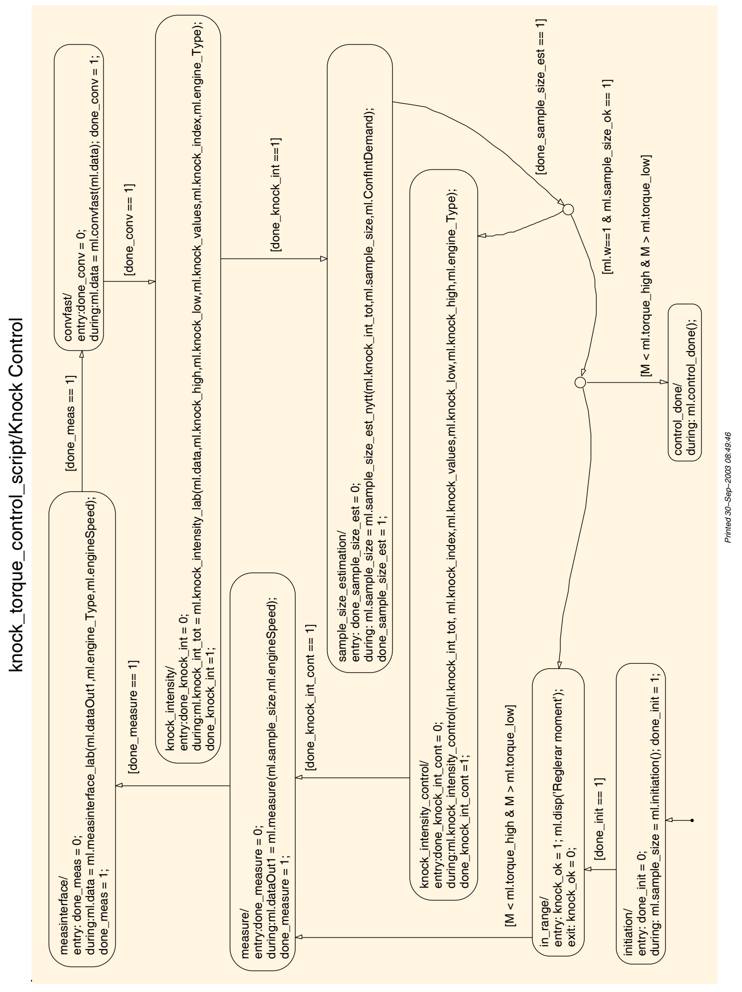

knock_torque_control_script/Knock Control
convfast/
entry:done_conv = 0;
during:ml.data = ml.convfast(ml.data); done_conv = 1;
[done_conv == 1]
knock_intensity/
entry:done_knock_int = 0;
during:ml.knock_int_tot = ml.knock_intensity_lab(ml.data,ml.knock_high,ml.knock_low,ml.knock_values,ml.knock_index,ml.engine_Type);
done_knock_int =1;
[done_knock_int ==1]
sample_size_estimation/
entry: done_sample_size_est = 0;
during: ml.sample_size = ml.sample_size_est_nytt(ml.knock_int_tot,ml.sample_size,ml.ConfIntDemand);
done_sample_size_est = 1;
[done_sample_size_est == 1]
[ml.w==1 & ml.sample_size_ok == 1]
[M < ml.torque_high & M > ml.torque_low]
control_done/
during: ml.control_done();
knock_intensity_control/
entry:done_knock_int_cont = 0;
during:ml.knock_intensity_control(ml.knock_int_tot, ml.knock_index,ml.knock_values,ml.knock_low,ml.knock_high,ml.engine_Type);
done_knock_int_cont =1;
[done_knock_int_cont == 1]
measure/
entry:done_measure = 0;
during:ml.dataOut1 = ml.measure(ml.sample_size,ml.engineSpeed);
done_measure = 1;
[done_measure == 1]
measinterface/
entry: done_meas = 0;
during:ml.data = ml.measinterface_lab(ml.dataOut1,ml.engine_Type,ml.engineSpeed);
done_meas = 1;
[done_meas == 1]
[M < ml.torque_high & M > ml.torque_low]
in_range/
entry: knock_ok = 1; ml.disp('Reglerar moment');
exit: knock_ok = 0;
[done_init == 1]
initiation/
entry: done_init = 0;
during: ml.sample_size = ml.initiation(); done_init = 1;
Printed 30-Sep-2003 08:49:46

Figure A.1: The knock intensity control stateflow chart

Appendix B

Matlab Functions

An alphabetic list of all the Matlab functions that appear in this thesis. All input and output variables are presented by their name. Explanation of what the variables contains are given when necessary.

alphaHP1415 This function transforms the input, *u*, that is given in percent, to a voltage that is sent to the ECU.
input: handle, engine_Type, u
output: none

control_done This function stops the simulation by setting the *Stop-Variable* in the Subsystem `Simulation Stop` to one.
input: none
output: none

convfast In *convfast* the long array of cylinder pressure data is cut up in cycles. A matching crank angle vector is also created .
input: data (measured cylinder pressure data)
output: data

initiation This function initiates the measuring equipment. This is where the simulation starts each time the Start button on the GUI is pressed or the controller is called from the script.
input: none
output: none

in_range The state *in_range* does not call any Matlab functions. This is the state that the Stateflow chart eventually ends up in when the knock intensity is successfully controlled. When this state is activated, a Stateflow parameter called *knock_ok* is set to one and when it is deactivated it is set to zero. This parameter is used to communicate with the switch in the Simulink model. When *knock_ok* equals one, the torque controller is active and when it

equals zero it is shut off. The condition for leaving this state is that the torque controller, that is active while this state is active, has controlled the torque into the interval that has been set in the GUI. The consequence of this is an iteration between the controllers.
input: none
output: none

knock_intensity The function *knock_intensity* filters out the knock from the cylinder pressure and finds the highest knock amplitude from each pressure cycle. This is defined as *knock intensity*. *knock_intensity* also calculates the *knock intensity mean* and determines whether it is in the range given by the values in the GUI/script or not. If the intensity is not in range, the next state that is activated will be *knock_intensity_control*, and otherwise it will be *in_range*.
input: data, knock_high, knock_low, engine_Type
output: knock_int_tot

knock_intensity_control This is where the actual control is done. The controller takes in the knock intensity mean and compares it to the reference value *knock_value*. The difference is multiplied by a constant and added to the angle of ignition, θ_{ign}. The new angle of ignition is sent back to the engine control system and when this is done the state *measurement* is reactivated and the Stateflow diagram starts over.
input: knock_int_tot, knock_index, knock_values, knock_low, knock_high, engine_Type
output: none

measureinterface In *measureinterface* the measured data is turned into a struct and completed with some information that is needed in the files that follow, such as sample frequency.
input: dataOut1 (measured cylinder pressure data), engineSpeed, engine_Type
output: data

measure This function handles the measure of cylinder pressure.
input: sample_size, engineSpeed
output: dataOut1 (measured cylinder pressure data)

model_open This function is only used when the regulator is run from the GUI. It checks if the model is open and opens it otherwise.
input: none
output: none

sample_size_estimation This file fits a Gamma distribution to the measured data and estimates a confidence interval for the arithmetic mean of the data. If the length of the confidence interval is short enough the sample size is kept. Otherwise the sample size is increased by calculating a new value of the parameter *sample_size*. Unfortunately, measurement of the cylinder pressure and the calculations of knock intensity, takes comparatively long time to make. For that reason, even if the sample size is too small, the *knock intensity mean* will still be used for control in the present revolution of the Stateflow chart. This can be modified only for the next revolution.
input: knock_int_tot, sample_size, ConfIntDemand
output: sample_size

torque_filter In *torque_filter* the mean value of the torque from the last ten samples is calculated. This is to even out some of the noise that occur in the torque signal.
input: none
output: none

Appendix C

Manual for the program

This manual can be read detached from the report.

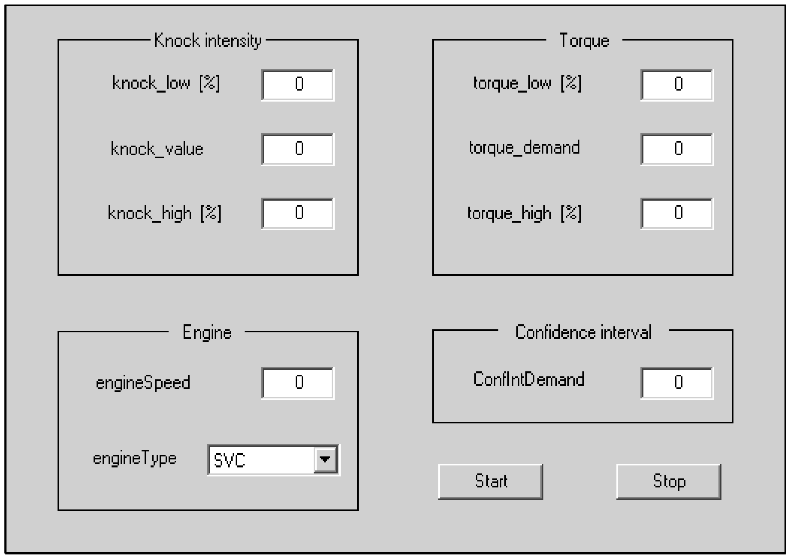
C.1 User Interface

The program can be run from one of two possible user interfaces. One is the GUI, which can handle only one operating point at a time. The other one is the script, which theoretically can handle an infinite number of operating points. The difference is that the GUI will only handle the control whereas the script will also make measurements and store the data.

C.1.1 GUI

In figure C.1 the GUI is shown. As can be seen, it has a number of input boxes. In the boxes *knock_value* and *torque_demand* the goal values are set. These values are what the controller will regulate towards. The desired knock intensity is set in volt. It may be more suitable to have the knock intensity in [Pa], but due to translation problems the voltage is not translated. Also the knock intensity will therefore be in volt.

Since it is not possible to obtain exactly the desired value, the control is considered successful when the value is within a given range. Inputs to the boxes *knock_low*, *knock_high*, *torque_low* and *torque_high* is a percentage of the respective goal value. For example, if the allowed interval for the knock intensity is from five percent under the goal value to five percent above the goal value, 0.95 and 1.05 should be written in the boxes. In *engineSpeed* the engine speed is inserted in rpm and in *engine_type* a choice can be made between SVC and TC



knock_low [%] 0
knock_value 0
knock_high [%] 0
torque_low [%] 0
torque_demand 0
torque_high [%] 0
engineSpeed 0
engineType SVC
ConfIntDemand 0

Figure C.1: The GUI used to manage the knock and torque regulator.

(Turbo Charged) engine. In *ConfIntDemand* the maximum acceptable range of the confidence interval is written. This value is also in volt since it is connected to the knock intensities. As a rule of thumb the maximum acceptable range should be five percent of the largest knock value in an average operating point.

The GUI has two push-buttons, *Start* and *Stop*. Pressing *Start* executes a callback function that checks if the Simulink model is open and opens it if it is not. It then initiates the Simulink parameters and starts the control. Pressing *Stop* stops the simulation and releases the ignition angle, θ_{ign}, so that the knock intensity control is handed over to the ECU. This is for safety, in case the knock intensity gets too high which could seriously harm the engine. The reason why the reaction to a too high knock intensity is manual instead of coded is that this is more flexible, and since the GUI only can handle one operating point at the time, it needs constant supervision anyway.

C.1.2 Script

In the script, the same variables are set as in the GUI. There are a few things that are different. The desired knock intensities and torque values are implemented as vectors here and there is also a vector for the wanted compression ratios. It is important to implement the values

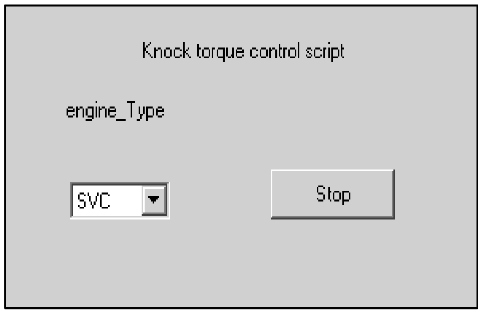


Figure C.2: The GUI that is used together with the script.

in the vectors in an ascending order. All these values are set in the first rows of the script.

There is a small GUI that is used together with the script. It is used to set the engine type and also has a stop button if the user wants to interrupt before all the operating points are run through (figure C.2).

When going from one value to the next in the torque vector, it is important that the change is gentle for the engine. Therefore a Simulink model called `Torque Ramp` is built. The model reads the the next torque from the torque vector and writes it in the box called `Reference Torque`. It then sets the current throttle angle as start value for the ramp. The throttle angle and hence the torque is increased slowly by the ramp. The torque is measured from the engine and compared to the `Reference Torque` and as soon as the measured torque equals the `Reference Torque` the ramping stops. The only parameter that can be set in `Torque Ramp` is the inclination of the ramp.

When the script version of the controller is opened, both this GUI and the `Torque Ramp` are opened automatically.

C.1.3 Changing Fixed Values

Apart from the GUI and the beginning of the script, there are a few other places in the code where parameters are also set. These values were initially not intended to be changed by the user of the program and hence they are coded directly as values instead of as variables. But situations may occur where these parameters also need to be changed, for instance if the program is run on another engine than a TC or an SVC. Therefore the following section will describe what these parameters do and where they can be found. Together with the comments in the code this should enable the user to make any changes he or she desires.

- In *knock_intensity_control* the control is interrupted if $\theta_{ign} > 35\,^\circ$. It is also interrupted if $\theta_{ign} < 0\,^\circ$.
- In *knock_intensity* the part of the cylinder curve that is investigated for the knock intensities is set to start at $\theta_{ign} + 3\,^\circ$.
- In *measinterface* the part of the cylinder curve that is investigated for knock intensities is set to end at a specific value. This value is implemented in degrees after TDC and it is different for the TC and the SVC engine.
- In *measinterface* there is also a variable called *thetaoffset*, that must be changed each time the position of the crank shaft incremental encoder is changed.
- In *initiation* the initial value of the variable *sample_size* is set to 100.
- In *knock_intensity_control* the value of K_P in the knock intensity controller is set. A scale factor that depends on the engine type is also set.
- In *torque_filter* the number of values that the WLS is calculated over, is set.
- In *sample_size* the level of confidence, 1-α for the confidence interval is set and also a scale factor that depends on the engine type.