# SPEED CONTROL OF ELECTRIC DRIVES IN THE PRESENCE OF LOAD DISTURBANCES

Wander Gonçalves da Silva

A thesis submitted for the degree of
Doctor of Philosophy

March, 1999

Department of Electrical and
Electronic Engineering

University of Newcastle upon Tyne

# ABSTRACT

The speed control of a Brushless DC Motor Drive in the presence of load disturbance is investigated. Firstly some practical results are presented where a simple proportional-integral speed controller is used in the presence of a large step input speed demand as well as load disturbance. The *wind-up* problem caused by the saturation of the controller is discussed.

In order to improve the performance of the proportional-integral speed controller in the presence of load variation, a load estimator is used with torque feedforward control. The results presented show the speed holding capability in the presence of load variation is significantly improved.

A genetic algorithm is used on line to optimise the controller for different conditions such as large and small step input speed demand and load disturbance. The results presented show that a genetic algorithm is capable of finding the tuning of the controller for optimal performance.

Single-input single-output and two-input two-output fuzzy speed controllers are also used and the results compared to a proportional-integral controller. Results are presented showing that a single-input single-output fuzzy controller works as a proportional controller with variable gain whereas the two-input two-output fuzzy controller is capable of driving the motor at variable speed and load torque with excellent performance. The robustness of the fuzzy controllers is compared to the proportional-integral controller and the results presented show that the fuzzy one is more robust then the proportional-integral.

A genetic algorithm is also used on line for the optimisation of the two-input two-output fuzzy speed controller and the results show that despite the large number of parameters to be optimised, the tuning for optimal performance is also possible.

# ACKNOWLEDGMENTS

The pursuer of a Ph.D. degree has all the responsibility for the works necessary to obtain it but many people are directly or indirectly involved. These people are the family of the candidate, sponsors, supervisor, technicians and other members of staff as well as friends. Their work and support are essential for the success of the candidate nevertheless they are not awarded any degree. However, recognition for what they have done is the least they should obtain. No matter how much each one has done, their works were equally important.

Because of this I would like to say thank you to everyone in the Electrical and Electronic Engineering Department at the University of Newcastle upon Tyne. However, a special thanks goes to Prof. P. Acarnley for the competent supervision received. Thanks also go to the friends from Fundação Educacional de Ituiutaba and Instituto Superior de Ensino e Pesquisa de Ituiutaba for the incentive and support; thanks to Fundação Coordenação de Aperfeçoamento de Pessoal de Nível Superior - CAPES and Fundação Educacional de Ituiutaba - FEIT for the financial support.

Thanks to the relatives who came from Brazil to visit me and my family bringing us some special moments of joy.

Thanks to a special friend, Alan dos Reis, who gave the best of himself to represent me in Brazil during my absence.

A very special thanks goes to my dear wife, Rubiane, for the support, care and love received but above all, thanks God for enabling me to do this work and trusting to me and my wife another lovely son.

To

Rubiane, Álvaro and Arthur.

# Table of Contents

**CHAPTER 1**: INTRODUCTION TO THE SPEED CONTROL OF ELECTRIC DRIVES **1.1**

**CHAPTER 2:** LITERATURE REVIEW **2.1**

2.1 - INTRODUCTION 2.1

2.2 - THE USE OF LOAD DISTURBANCE OBSERVER 2.3

2.3 - THE USE OF ALTERNATIVE CONTROL APPROACHES 2.6

2.4 - CONCLUSION 2.8

**CHAPTER 3**: EXPERIMENTAL SET UP **3.1**

3.1 - INTRODUCTION 3.1

3.2 - DESCRIPTION OF THE EXPERIMENTAL SETUP 3.2

**CHAPTER 4**: SOME PRACTICAL RESULTS WITH STANDARD PI SPEED CONTROLLER **4.1**

4.1 - INTRODUCTION 4.1

4.1.1 - 1100 rpm step input speed demand with 11% of the full load 4.3

4.1.2 - 1100 rpm step input speed demand in the presence of load disturbance 4.3

4.1.3 - 1500 rpm step input speed demand in the presence of load disturbance 4.4

4.2 - ANTI-WINDUP CIRCUIT 4.5

**CHAPTER 5**: A GENETIC ALGORITHM APPLIED TO THE ON LINE OPTIMIZATION OF A STANDARD PROPORTIONAL-INTEGRAL SPEED CONTROLLER **5.1**

5.1 - INTRODUCTION 5.1

5.2 - INTRODUCTION TO GENETIC ALGORITHMS 5.2

5.3 - APPLICATION OF GENETIC ALGORITHMS TO THE TUNING OF A PI SPEED CONTROLLER 5.3
5.4 - EXPERIMENTAL RESULTS 5.4
5.4.1 - Optimisation for a large step input speed demand (0 to 1100 rpm) 5.5
5.4.2 - Optimisation for a small step input speed demand (1000 to 1100 rpm) 5.9
5.4.3 - Optimisation for load disturbance 5.12
5.4.4 - Optimisation for variable speed and load 5.15
5.4.5 - Tuning of the PI speed controller by using classical control theory 5.18
5.4.5 - Optimisation for load disturbance with integrator anti-windup 5.21
5.5 - INVESTIGATION ABOUT THE EXISTENCE OF LOCAL MINIMA 5.24
5.6 - ANALYSIS OF RESULTS 5.27
5.7 - CONCLUSION 5.28

**CHAPTER 6**: THE USE OF LOAD ESTIMATOR **6.1**
6.1 - INTRODUCTION 6.1
6.2 - EFFECT OF LOAD ESTIMATOR 6.3
6.2.1 - 1100 rpm step input speed demand with 62% of the full load 6.3
6.2.2 - 1100 rpm step input speed demand with 11% of the full load 6.4
6.2.3 - 1100 rpm step input speed demand in the presence of load disturbance 6.6
6.2.4 - 1500 rpm step input speed demand in the presence of load disturbance 6.6

6.2.5 - 1100 rpm constant speed in the presence of the load disturbance 6.6
6.3 - OPTIMIZATION OF THE SPEED CONTROLLER BY GENETIC ALGORITHM 6.8
6.4 - INVESTIGATION INTO THE EXISTENCE OF LOCAL MINIMA 6.11
6.5 - CONCLUSION 6.13

**CHAPTER 7**: ROBUSTNESS OF THE PI SPEED CONTROLLER AGAINST LOAD VARIATION AND SPEED DEMAND **7.1**
7.1 - INTRODUCTION 7.1
7.2 - EXPERIMENTAL RESULTS 7.2
7.2.1 - 1100 rpm with 11% load torque 7.2
7.2.2 - 1100 rpm with 62% load torque 7.2
7.2.3 - 1100 rpm in the presence of load disturbance 7.2
7.2.4 - 1500 rpm with 11% of maximum load torque 7.4
7.2.5 - 1500 rpm with 75% load torque 7.4
7.2.6 - 1500 rpm in the presence of load disturbance 7.6
7.2.7 - 700 rpm with 11% load torque 7.7
7.2.8 - 700 rpm with 55% load torque 7.7
7.2.9 - 700 rpm in the presence of load disturbance 7.9
7.2.10 - Summary of the results on robustness against load variation 7.9
7.3 - CONCLUSION 7.10

**CHAPTER 8**: SINGLE INPUT SINGLE OUTPUT - SISO FUZZY CONTROLLER **8.1**
8.1 - INTRODUCTION 8.1
8.2 - PRINCIPLES OF FUZZY LOGIC 8.1
8.2.1 - Introduction to fuzzy logic 8.1
8.2.2 - Membership functions, rules and transfer functions 8.2

8.2.3 - Influence of the input membership functions
on transfer functions 8.4
8.2.4 - Influence of the output membership functions
on transfer functions 8.6
8.2.5 - Influence of increased number of membership functions 8.9
8.3 - EXPERIMENTAL RESULTS 8.11
8.3.1 - Fuzzy controller with two membership functions
per variable and low gain 8.12
8.3.2 - Fuzzy controller with four membership
functions per variable 8.13
8.3.3 Fuzzy controller with two membership functions
per variable and high gain 8.14
8.3.4 - Proportional speed controller 8.16
8.3.5 - Proportional-Intregral speed controller 8.17
8.4 - COMPARISON OF PERFORMANCE IN RESPONSE
TO A STEP INPUT OF SPEED DEMAND 8.18
8.5 - ROBUSTNESS AGAINST LOAD VARIATION 8.19
8.5.1 - Fuzzy controller with two membership functions
per variable and low gain 8.19
8.5.2 - Fuzzy controller with four membership functions
per variable 8.20
8.5.3 - Fuzzy controller with two membership functions
per variable and high gain 8.20
8.5.4 - Proportional speed controller 8.22
8.5.5 - Proportional-Integral speed controller 8.22
8.6 - CONCLUSION 8.25

**CHAPTER 9**: MULTIPLE INPUT MULTIPLE OUTPUT - MIMO
FUZZY CONTROLLER **9.1**
9.1 - INTRODUCTION 9.1
9.2 - THE FUZZY SPEED CONTROLLER 9.2

9.2.1 - The membership functions and rules 9.3
9.3 - EXPERIMENTAL RESULTS 9.4
9.4 - ROBUSTNESS AGAINST LOAD VARIATION AND SPEED DEMAND 9.6
9.4.1 - 1100 rpm with 11% load torque 9.7
9.4.2 - 1100 rpm with 62% load torque 9.7
9.4.3 - 1100 rpm in the presence of load disturbance 9.7
9.4.4 - 1500 rpm with 11% of maximum load torque 9.9
9.4.5 - 1500 rpm with 75% of maximum load torque 9.10
9.4.6 - 1500 rpm in the presence of load disturbance 9.10
9.4.7 - 700 rpm with 11% load torque 9.11
9.4.8 - 700 rpm in the presence of 55% load torque 9.12
9.4.9 - 700 rpm in the presence of load disturbance 9.12
9.4.10 - Summary of the results on robustness against load variation 9.13
9.6 - CONCLUSION 9.14

**CHAPTER 10**: GENETIC ALGORITHM APPLIED TO FUZZY CONTROLLER **10.1**
10.1 - INTRODUCTION 10.1
10.2 – THE USE OF A GENETIC ALGORITHM FOR A FUZZY CONTROLLER 10.1
10.3 - DEFINITION OF THE RANGE OF EACH MEMBERSHIP FUNCTION 10.3
10.4 - EXPERIMENTAL RESULTS 10.4
10.4.1 - Optimisation for step input speed demand in the presence of load variation 10.4
10.4.2 - Optimisation for 1100 rpm constant speed in the presence of load disturbance 10.6
10.5 - INCREASING THE POPULATION SIZE AND RESTRICTING THE SEARCHING SPACE 10.8

10.5.1 - Optimisation for step input speed demand in the presence of load variation 10.9
10.5.2 - Optimisation for 1100 rpm constant speed in the presence of load disturbance 10.10
10.6 - CONCLUSION 10.13

**CHAPTER 11**: GENERAL CONCLUSIONS **11.1**
FUTURE WORKS 11.4

**REFERENCES** **R.1**

**APPENDIX A**: ANALYSIS OF THE LOAD ESTIMATOR **A.1**

**APPENDIX B:** INTRODUCTION TO FUZZY LOGIC AND FUZZY CONTROL **B.1**
B.1 - INTRODUCTION B.1
B.2 - FUZZY SETS B.2
B.3 - MEMBERSHIP FUNCTIONS B.4
B.4 - LOGICAL OPERATORS B.4
B.5 - IF-THEN RULES B.5
B.6 - FUZZY INFERENCE ENGINE B.6
B.6.1 - Making the inputs fuzzy B.8
B.6.2 - Application of the fuzzy operators B.9
B.6.3 - Implication B.9
B.6.4 - Aggregation B.10
B.6.5 - Taking the output B.11
B.6.7 - Another simple example focusing on the influence of the shape of the membership functions B.14

**APPENDIX C**: AN OVERVIEW ON GENETIC ALGORITHM **C.1**
C.1 - INTRODUCTION C.1

C.2 - GENETIC ALGORITHM OPERATIONS C.3
C.2.1 - Crossover operation C.3
C.2.2 - Mutation operation C.3
C.2.3 - Selection operation C.4
C.2.4 - Stochastic universal sampling C.5

**APPENDIX D**: ANALYSIS OF THE CASTING DRUM DC MOTOR DRIVE ON THE LINE 51 OF DuPONT **D.1**
D.1 - INTRODUCTION D.1
D.2 - SIMULATION RESULTS D.2
D.3 – THE IMPACT OF LOAD DISTURBANCE D.4
D.4 – THE LOAD ESTIMATOR D.6
D.5 – THE EFFECT OF RETUNING THE SPEED CONTROLLER D.8
D.6 – THE IMPACT OF THE PRESENCE OF NOISE ON THE SPEED SIGNAL D.12
D.7 - CONCLUSION D.14
D.8 PARAMETERS OF THE DC MOTOR DRIVE AS DETERMINED BY THE DuPONT ENGINEER D.15
D.8.1 – Components value D.15
D.8.2 – Mechanical Parameters D.15
D.8.3 – Parameters used along the block diagram of the dc motor drive system D.15
D.8.3.1 - Tacho generator D.16
D.8.3.2 - Speed demand filter D.16
D.8.3.3 - Speed loop D.16
D.8.3.4 - Current loop D.17
D.8.3.5 - Motor constants D.17
D.8.3.6 - Casting drum constants D.18

# CHAPTER 1

# INTRODUCTION TO THE SPEED CONTROL OF ELECTRIC DRIVES

There are many industrial applications of motor drives where precise speed control is essential to ensure final product quality. In the film, paper or steel making process for instance, accurate speed holding capability in the presence of load variation is another required characteristic of the motor drive. The reason is very simple: In such production processes, the profile of the final product deteriorates if the speed of the motor is not kept rigorously constant. However, many load disturbances are experienced in the production line. They happen for many different reasons and from different sources.

Ho *et al* [HO *et al*, 1994] has reported that in a typical film production plant there are approximately 30 variable speed drives, most of them based on the DC motor. In such manufacturing process, the drives have to operate together at constant speed. Because of this, if the speed of any of them changes, the disturbance can be transmitted to others via the web as they are mechanically linked. Then, in order to ensure that the speed of the drives does not change regardless of load variation, the controller has to respond fast to any load change.

Most of the speed controllers found in motor drives used in industry are based on the classical Proportional-Integral (PI) control. According to the paper mentioned above, the tuning of the controller is another key point. Different control tuning techniques have emerged over the past few decades, each one claiming advantage over the others, and there have been significant improvements in the tuning process. However, according to Ho *et al*, in many cases variable speed drives have the speed controller tuned based on the step input to achieve optimum setting for both the current and

speed loops. This classical tuning method may be satisfactory for many applications. Nonetheless, for the applications in the film, paper or steel making process, it has not yet proved efficient, as speed-holding of the motor drives is the key point.

It has also been report by Ho *et al* that there is not sufficient time in the production schedules to tune the controller. As a consequence, in many cases, the tuning done in the drive maker's factory is kept unchanged. However, because of non-linearities always present in the process, even a well tuned controller loses performance when the speed demand or load torque changes.

In the literature there are many different control approaches, different types of motor drive controls, as well as tuning techniques. It means that even today, researchers are carrying out investigations in order to provide the best possible speed controller. In most of the cases, robustness against parameter variation is also claimed.

There are today many different control strategies available. Some of them are based on the classical linear PI controller and others on non-linear approach sometimes using artificial intelligence such as expert systems, fuzzy logic and neural network. Other control techniques found in the literature are, for instance, sliding mode control, adaptive control and bang-bang control. Each one has its peculiarities and the authors claim performance advantage as well as robustness against parameter variation. However, what has also been seen in the literature is that, an even better controller has to incorporate the best characteristics of each control approach. As a consequence, there can be found Fuzzy-PI Controller, Neuro-Fuzzy Controller, Adaptive Fuzzy Sliding Mode Control and others. The controllers become so complex that another problem arises: the best tuning. Then comes, the use of Genetic Algorithms, Evolutionary Design and other optimisation techniques. It implies that finding the best speed controller for electric drives is not an easy task.

Obviously, the investigation of all the possible control techniques would take years of study. Even then, what is good for one particular case, may not be for another. The work presented here concentrates on the study of what can be obtained by using the conventional Proportional-Integral speed controller to hold the speed of a Brushless DC Motor Drive, in the presence of load disturbance. Investigation of fuzzy

controllers has also been carried out and the quality of the speed responses obtained was compared to those obtained by using Proportional-Integral.

This work is presented as follows. In Chapter 1, the introduction, as above. In Chapter 2, a literature overview is presented as an attempt to show what has already been done on the speed control of electric drives by other researchers. In Chapter 3, a description of the experimental set up is given. In Chapter 4, some experimental results obtained by using the standard conventional PI speed controller for different speed demand and load torque are presented. In addition to this, one example of an *anti-windup* circuit used to improve the performance of the controller due to large step input speed demand is presented. In Chapter 5 a Genetic Algorithm (GA) is used on-line to optimise the tuning of the PI speed controller. The motor is driven at different conditions when the optimisation process is carried out. It is shown that GA is capable of finding the best tuning for a particular situation. It is also shown that the best tuning for one condition is not suitable for other different load torques and step speed demands. Also, an example where a Genetic Algorithm is used to optimise the controller and the *anti-windup* circuit setting at the same time, for a large step input speed demand in the presence of load variation, is presented. In Chapter 6 a load estimator is added in order to improve the speed holding capability of the motor drive against load variation. Explanation of how it works is given and on-line optimisation of the controller setting by using Genetic Algorithm is introduced. In Chapter 7, the robustness of the PI speed controller against step input speed demand and load variation, after the controller has been tuned for one specific condition, is investigated. In Chapter 8 a Single-Input Single-Output (SISO) fuzzy speed controller is introduced. It is shown that this controller has some features, which can not be matched by the conventional PI controller. However, it is also shown that it does not guarantee zero steady state speed error. In Chapter 9, a Multiple Input Multiple Output (MIMO) fuzzy speed controller with load estimator is presented. Experimental results show that this controller can provide good control capability with different speed and load demand, with some advantages over the PI regarding robustness against load variation and large step input speed demand. In Chapter 10, a GA is applied to the on-line optimisation of the fuzzy controller. Finally, in Chapter 11, the conclusion, followed by the list of References and the Appendices.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 - INTRODUCTION

The speed control of electric drives is a subject that seems quite simple in the first place, as there is plenty of literature available on this topic within the classical control theory. It is possible to find information about the speed control of a DC motor drive in nearly every single book on control systems. However, due to the inaccuracy of mathematical models and the practical difficulties mentioned in chapter 1, the speed control of electric drives is not as simple as it may appear to be. In this chapter, some of the work that has already been carried out by researchers on the speed control of electric drives are discussed. The starting point has to be the classical and well-understood Proportional-Integral (PI) speed controlled DC motor, still largely used in industry.

The most commonly used speed controlled DC motor drive has a block diagram as presented in Fig. 2.1.

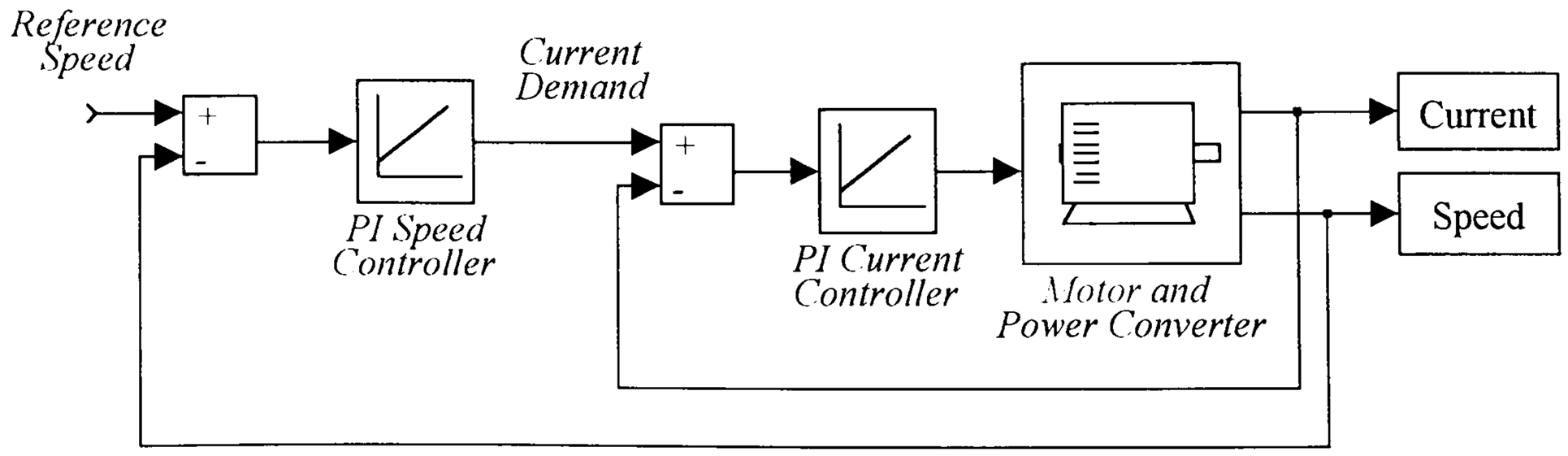



Fig. 2.1 - Classical block diagram of the speed control DC motor drive.

In this configuration, two Proportional-Integral controllers are used in cascade configuration. The inner one does the current control whereas the outer one does the speed control. The actual speed of the motor is compared to a demand value, generating a speed error. The speed error goes into the speed controller that gives appropriate current demand for the current controller. The actual current of the motor is compared to the demand value given by the speed controller, generating a current error. The current error then reaches the Proportional-Integral current controller that determines the appropriate current output and consequently, the torque produced by the motor. By controlling the torque it is possible to regulate the speed.

The response time of the armature current controller is usually much shorter than the motor/load mechanical time constant so transients in the electrical circuit are neglected. This makes the analysis of the control system much simpler, as shown in Fig. 2.2

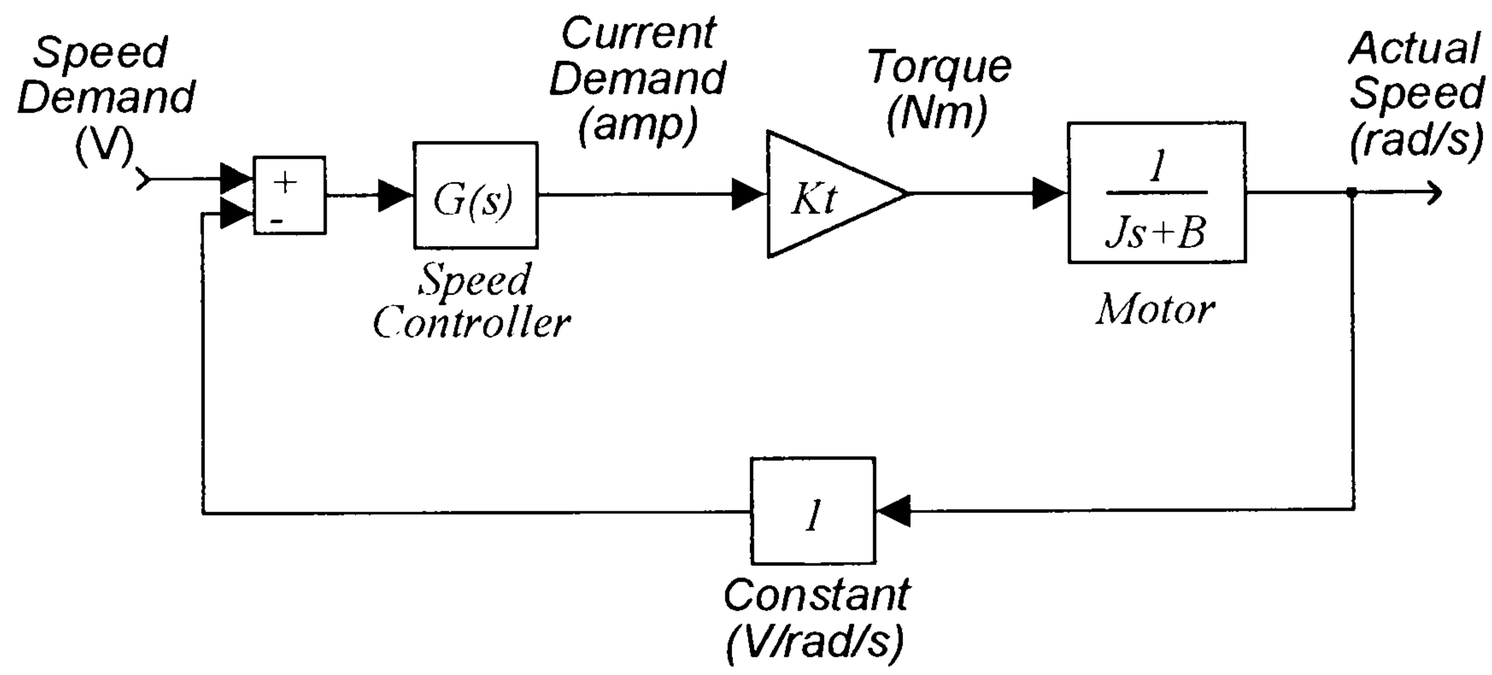



Fig. 2.2 - Simplified block diagram of a DC motor drive with speed controller.

Should a proportional-integral speed controller be used, a second order control system is obtained and the design of the speed controller is very simple. There are many different methods available found in most linear control books, such as Root-Locus, Nyquist or Frequency Response methods [OGATA, 1997 and SHINNERS, 1998]. In addition to this, as long as the gain of the proportional and integral portions of the speed controller is positive, the system is always stable. So, when the speed holding capability against load variation is the major concern, the higher the gains of the controller the faster it responds to speed changes. So, why not use high proportional and integral gain for the controller? The answer is simple. For an absolutely linear

system, high gains impose high current values through the windings of the motor and power converter devices during transients. However, in a practical situation, the current capabilities of the converter switches and the motor are limited as well as the output of the controllers. In this case, high gains can be used but the system becomes non-linear.

As a result of the current limitation of the motor drives, another problem arises known as integrator *windup* [WIEN, 1987, VRANCIC *et al*, 1996, ALBERTOS *et al*, IEE 1997 and SHIN, 1998]. The reason is that the integrator can not "see" when the output of the controller has reached saturation and keeps on integrating the speed error. As a consequence, depending on the setting of the controller, a very large speed overshoot may take place. The saturation of the controller constitutes a serious non-linearity in the control system.

Attempting to solve this problem, several different configurations of *anti-windup* have been created and are available in the literature. However, what was a linear system in the first place, became non-linear. The right setting of the *anti-windup* circuit is as important as the setting of the speed controller itself. Nevertheless, the setting of the speed controller which is good for one condition, may not be for another, so an alternative strategy for improving the performance of the speed control of electric drives is necessary.

Within linear control systems, state feedback control based on state-space representation of the plant and controllers can be used [BROGAN, 1991, SHINNERS, 1998 and WILLIAMSON, 1991]. In this case, the drive is modeled in terms of a set of first order linear differential equation where the states of the drive are related to the inputs. However, the non-linearity imposed by the current limitation of the drive in a practical situation still applies. In addition to this, in many cases some of the states of the plant are inaccessible. As a consequence, this approach is often found to be unsatisfactory.

## 2.2 - THE USE OF LOAD DISTURBANCE OBSERVER

In order to improve the performance of electric drives in the presence of load disturbance several researchers have proposed the use of load torque observers. Ko *et*

*al* [KO *et al*, 1994] has proposed an adaptive load observer for use on a position control of a Brushless DC Motor. Nishii [NISHII *et al*, 1992] has used a disturbance observer for the vibration control of a wheeled mobile robot. Some other papers have been published where load disturbance observers were used on electric drives [BUJA *et al*, 1995, KO *et al*, 1993, VOLCANJK *et al*, 1994, SENJYU *et al*, 1995 and KAWAJI *et al*, 1995].

Due to the simplicity, special attention has been given to Iwasaki [IWASAKI *et al*, IECON'91]. In his paper a load observer together with a Proportional-Integral speed controller for a vector controlled induction motor has been proposed: The block diagram is shown in Fig. 2.3.

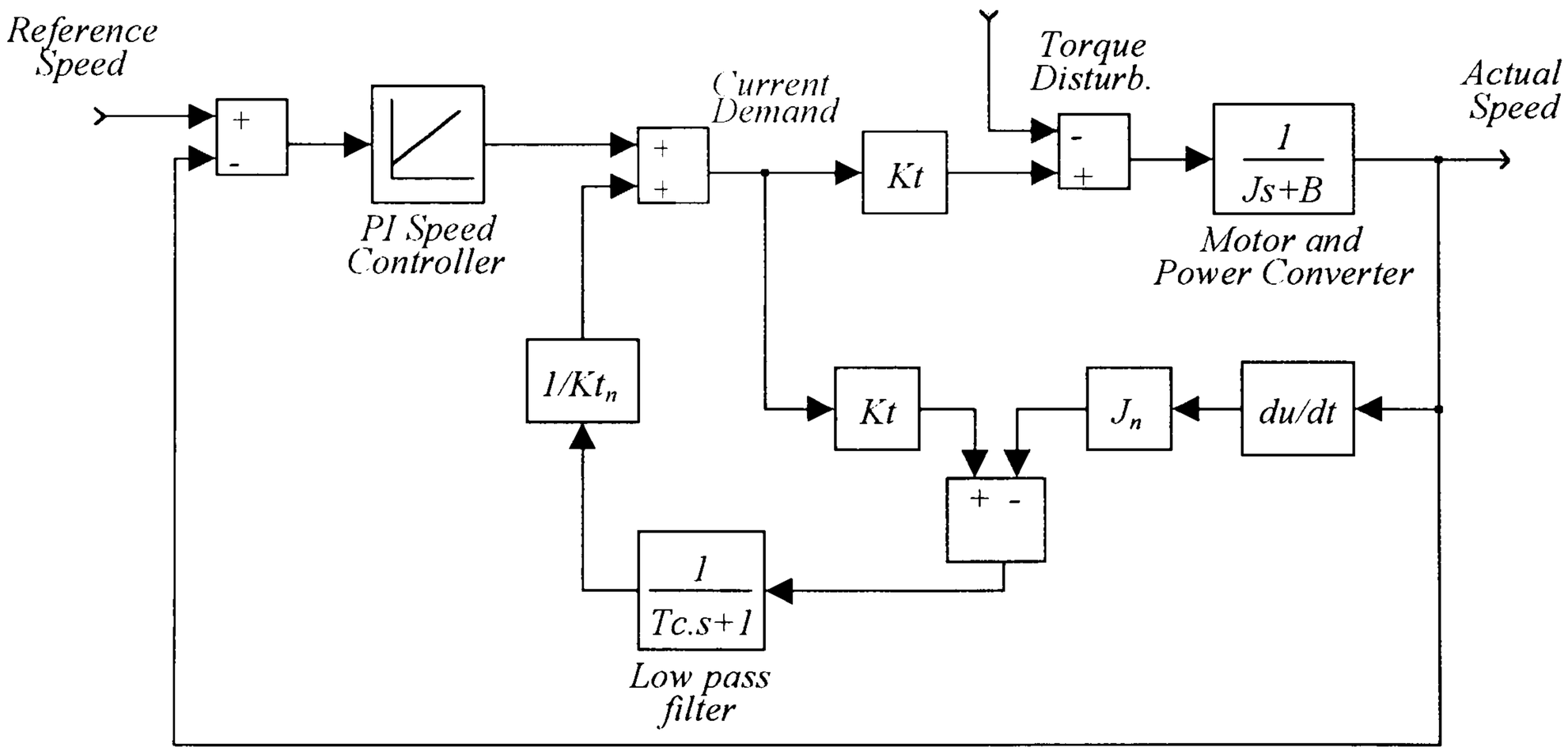



Fig. 2.3 - DC motor drive, with load observer and torque feedforward control as proposed by Iwasaki.

Where:

$J$ = Motor/load inertia.

$J_n$ = Nominal inertia.

$Kt$ = Motor torque constant.

$Kt_n$ = Nominal torque constant.

Tc = Low pass filter time constant.

$B$ = Friction coefficient.

Should a Proportional-Integral speed controller be used where $Kp$ is the proportional gain and $Ki$ the integral one, $J = J_n$ and $Kt = Kt_n$, the closed loop transfer function of system shown in Fig. 2.3 becomes.

$$T.F. = \frac{Kt(TcS+1)(KpS+Ki)}{JTcS^3 + (J+BTc+KtKpTc)S^2 + (B+KtKp+KtKiTc)S + KtKi} \qquad (2.1)$$

The authors demonstrated the stability and robustness of the system against parameter variation. Experimental results were shown in order to demonstrate the effectiveness of the control approach against load torque variation.

Kang [KANG et al, IECON '91] has proposed a similar load estimator for use on a field-oriented induction motor with a "pseudo derivative feedback" speed controller, which has the block diagram shown below:

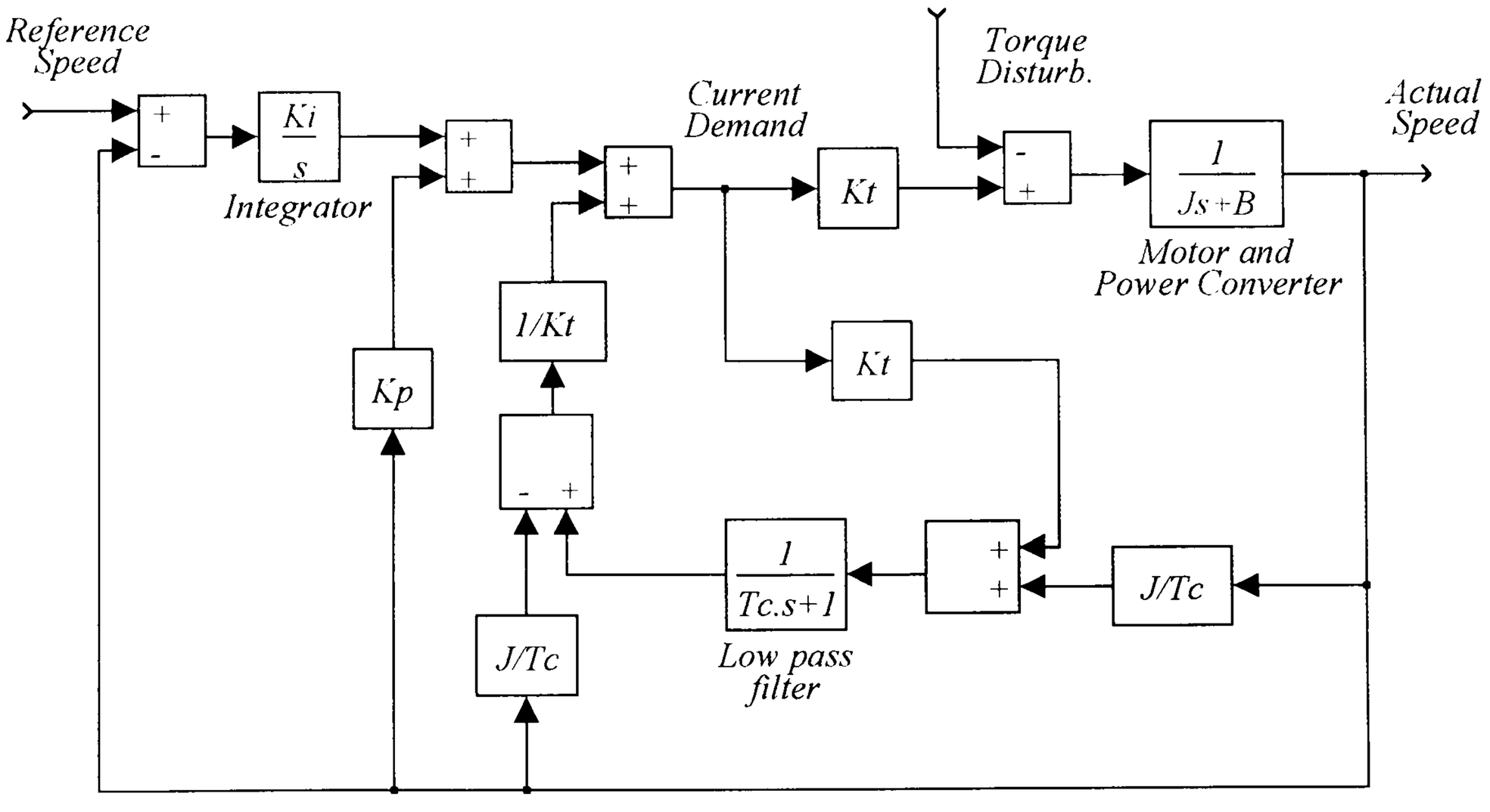



Fig. 2.4 - DC motor drive with load observer and torque feedforward control as proposed by Kang.

The transfer function of the system shown in Fig. 2.4 is as in equation 2.2.

$$T.F. = \frac{KtKi(TcS+1)}{JTcS^3 + (J+BTc+KtKpTc)S^2 + (KtKp+KtKiTc)S + KtKi} \qquad (2.2)$$

Should a Proportional-Integral speed controller be utilized in Fig. 2.4 as it was in Fig. 2.3, the transfer function between actual and reference speed becomes as follow.

$$T.F. = \frac{Kt(TcS+1)(KpS+Ki)}{JTcS^3 + (J+BTc+KtKpTc)S^2 + (KtKp+KtKiTc)S + KtKi} \qquad (2.3)$$

Comparing equations 2.1 and 2.3, shows that the only difference between them is the friction coefficient "*B*" multiplying "*S*" in the denominator of equation 2.1 that does not appear in equation 2.3. The major difference in equation 2.2, compared to equations 2.1 and 2.3, is the absence of one *zero* in the transfer function. However, as the characteristic equation (denominator) is nearly the same, the system has approximately identical disturbance transient behavior. The simulation and experimental results presented have shown that the load estimators working together with the speed controller provide better speed holding capability in the presence of load torque disturbance. However, the problems caused by the non-linearities of the motor drive system would not be solved as a Proportional-Integral speed controller is still being used in a non-linear system.

## 2.3 - THE USE OF ALTERNATIVE CONTROL APPROACHES

Researchers round the world realized that alternative control approaches for the speed control of electric drives should be studied. Several different non-linear control approaches have been utilized in electric drives, such as Adaptive Control, Sliding Mode Control, Bang-Bang Control and those based on artificial intelligence like Fuzzy Logic and Neural Networks. Each one has particular characteristics, advantages and disadvantages when compared to the others. As a consequence, researchers started to exploit the qualities of each one by putting them together in different drive systems and so many other possible control approaches have emerged. Among them can be found fuzzy bang-bang controllers [CHANG *et al*, 1995 and HWANG et al, 1995], adaptive and sliding mode [BAIK et al, 1998], adaptive fuzzy sliding-mode [LIN et al, 1998], fuzzy logic tunable speed controller [SOLIMAN et al, 1994], fuzzy-neural controller [BIERKE et al, 1997].

So, what was supposed to be a simple problem becomes quite complex. As a consequence, the tuning of the speed controller became another key point, as human experts may not be capable of finding the optimal tuning for the controller. The method based in linear control theory mentioned above no longer applies. As a consequence, researchers have tried alternative methods. Park [PARK *et al*, 1995] has utilized what he called a "Auto-Regressive Moving Average Model" whereas Kim [KIM *et al*, 1998] has proposed a "well known" Iteractive Learning Control (ILC) approach. Cotta [COTTA *et al*, 1996] came up with "Evolutionary Design" applied on fuzzy controller. Also applied to a fuzzy controller, Inoue [INOUE *et al*, 1998] has developed an "Advanced Autotuning Approach" whereas Soliman [SOLIMAN *et al*, 1995] proposed a tunable fuzzy logic control scheme for the speed control of DC series motor drives.

Fuzzy controllers have gained widespread application in power electronics and drives over the past few years, being used by many researchers round the world [SOUZA *et al*, 1995, LEE *et al*, 1994, VAS *et al*, 1994, FODOR, 1996, CATALIOTI *et al*, 1997, GUILLEMIN, 1996, LIN, 1994, BIRD *et al*, 1997 and GOVIND *et al*, 1995], although many other could be mentioned. While some investigators have found benefit in using fuzzy control, others are still reluctant or not yet convinced of its advantages. According to Souza [SOUZA and BOSE, 1995], there are control theorists who criticize the lack of formal design and analysis methodology in the fuzzy approach. Seeking better design methodology for fuzzy controllers, many researchers have used genetic algorithm for the design of the membership functions and definition of the rule base in fuzzy controllers [LISKA *et al*, 1994, HOMAIFAR *et al*, 1992, HOMAIFAR *et al*, 1995, LEE *et al*, 1993, MEREDITH *et al*, 1993, CHANG *et al*, 1995, TANG *et al*, 1994, KARR, 1991 and TANG *et al*, 1998]. However, there are those who do not think fuzzy logic could bring any advantage over the classical controller. Michels [MICHELS, 1997] said that any response behavior obtained by fuzzy logic could be achieved with "classical methods". Abramovitch [ABRAMOVITCH, 1994] said that, in many cases, what has made fuzzy control better was the fact that the proponents have used extra sensors, which could be incorporated into a standard control algorithm. Among the critics is also

Chen [CHEN, 1993] who came up with a list of applications where fuzzy controller could be used o not. However what is evident by the number of papers already published in this subject is that there is still room for improvements as no agreement has been reached on the benefits each control approach could bring.

Another common claim regarding most of the control approaches, is robustness against parameter variation. However according to Michels [MICHELS, 1997], the degree of robustness has not yet been measured and any surface derived from fuzzy control is as robust as any other derived from classical methods. Still according to Michels, some researchers mix up robustness with stability. Despite controversies, an increase in the number of drives employing artificial intelligence such as fuzzy logic and neural networks is expected in the future [VAS, 1997].

## 2.4 - CONCLUSION

From the investigations already carried out by researchers round the world on the speed control of electric drives, comes the realization that the problem is quite complex. The non-linearities of the drive systems create different problems in different applications. Due to the advance in terms of processing power of microcontrollers and digital signal processor, the use of many alternative control strategies became possible. Most of them are based on non-linear approaches and artificial intelligence. However, the controller became complex and other problems have emerged as a result. Several test performance comparisons have been done but disagreements still exist between researchers. It means that there is still a lot to be done in terms of speed control of electric drives.

# CHAPTER 3

# EXPERIMENTAL SET UP

## 3.1 - INTRODUCTION

The implementation of a simple Proportional-Integral speed controller for a DC motor drive can be done either using analogue devices based on operational amplifiers or digitally on a Microcontroller or Digital Signal Processor (DSP). However, when it comes to alternative control approaches such as those based on Artificial Intelligence like Fuzzy Logic or Neural Networks, the implementation using analogue devices is impossible. In today's world, computer speed and processing power is sufficiently high at a reasonable price. As a consequence, the use of discrete time control systems has become largely used by researchers round the world. In addition to this, industry has already started to use discrete time control systems on many commercial drives. Its use brings many advantages compared to continuous time [Kuo, 1992]:

1 - Flexibility, as it is possible to change controller gains, parameters and even the algorithm by means of software;

2 - Digital components are less affected by environmental changes than analogue ones.

3 - Digital control is more reliable.

4 - Discrete time control systems are less sensitive to variations on the controller.

On the other hand, some disadvantages can be highlighted:

1 - Limitations on the speed of computers and signal resolution can be a problem in certain applications.

2 - The delays in the control loop caused by the computer speed can cause instability in some closed loop systems.

3 - The control system does not "see" what is happening between signal samples.

However, in the case of speed control of electric drives, the microprocessors and DSPs available today can be easily used, providing excellent flexibility and performance. Because of this and due to the use of fuzzy controllers later on, a discrete time control system was chosen.

## 3.2 - DESCRIPTION OF THE EXPERIMENTAL SETUP

The motor drive system used was initially based on the Proportional-Integral (PI) speed control of a Brushless DC Motor, in cascade configuration with a PI speed controller, as shown in Fig. 3.1.

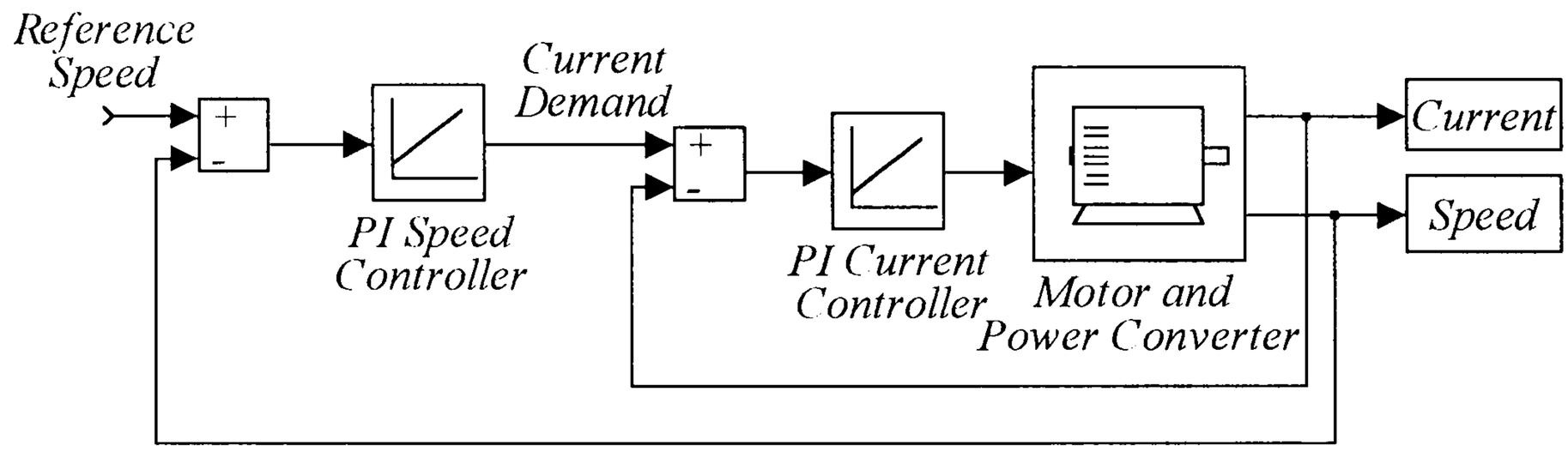



Fig. 3.1 - PI speed controlled Brushless DC Motor block diagram.

The experimental apparatus consisted of a standard commercial Brushless DC Drive System as follow.

Three-phase 415V AC - 50 Hz was applied straight into the supply module, which was a Bosch Servodyn VM60-T full bridge thyristor rectifier. The DC link voltage was supported by a Bosch Servodyn Capacitor Module, KM 1100T 1100 μF/840V. The output of the supply module was regulated at 600V DC.

A three-phase inverter was used to supply the Brushless DC Motor. The inverter was a Bosch Servodyn SM50/100, with a built in Proportional-Integral current controller.

The motor itself was a Bosch Servodyn Servo Motor SE-B4, with the following parameters:

Nominal Current: 9A

Nominal Speed: 3000 rpm

Torque Constant: 0.91 Nm/A

Winding Inductance: 9.25 mH

Winding Resistance: 0.74 Ohms

Emf constant: 95.7 V/1000 rpm.

Tachogenerator emf constant: 2.7V/1000rpm ± 5% at $t_{amb} = 20^{o}C$

Motor/Load Inertia: 0.016 $Nm.sec^2/rad$

Viscous Friction Coefficient: 0.0092 Nm.sec/rad.

For loading purposes, the Brushless DC Motor was mechanically coupled to a generator, which supplied a resistive load bank. The three-phase voltage generated was rectified by using a full bridge diode rectifier SEMIKRON SKD 30/04 A1. At the output of the rectifier an electronic switch, SEMIKRON IGBT SKM 75GARO63D, was used to be able to apply load steps to the motor, by switching on the resistor bank. The IGBT was triggered on and off by an analogue input drive card SEMIKRON SKHI10. The input signal for the IGBT drive card was obtained through a 12 bits resolution D/A converter, used as the interface with a Digital Signal Processor (DSP).

The speed controllers were implemented on a Texas TMS320C31 – 40 MHz DSP. A 10 bit resolution A/D converter was used to interface the speed signal produced by the tachogenerator. Another 10 bits resolution A/D card was used for the current. A 12 bit resolution D/A converter provided the interface between the processor and the drive, supplying current demand within the range 0 A to 9 A. This range was initially chosen as the motor would be driven mostly at constant speed, in the presence of load variation. As a consequence, there would be no need of operating in 4 quadrants, with negative current. The current controller used was an analogue Proportional-Integral, built in the drive. Matlab/Simulink was used to generated the "C" code programme for the controller and this programme was compiled before being downloaded into the processor. By using the Matlab/Simulink package, together with a software interface

developed in the Electrical and Electronic Engineering Department of the Universityof Newcastle upon Tyne, it was possible to communicate directly from the host PC to the DSP, in the Matlab environment. The sampling time for input and output signals was 1.25 ms.

# CHAPTER 4

# SOME PRACTICAL RESULTS WITH STANDARD PI SPEED CONTROLLER

## 4.1 - INTRODUCTION

In this chapter, the use of a classical Proportional-Integral speed controller, for the standard commercial Brushless DC Motor Drive described in the previous chapter, is investigated. When the main concern is the speed holding capability in the presence of load variation, the performance of the motor drive due to step changes on the speed demand is not the key issue. However, in order to provide better understanding of how the controller works, it has been initially set up for the following condition.

The tuning of the controller was done directly on the drive for the following test condition. An 1100 rpm step input reference speed was applied. The resistor bank was set to produce a constant load torque equal to 62% of the maximum rated electrical torque produced by the motor (8.2 Nm), which corresponds to nominal armature current (9 A) times the torque constant (0.91 Nm/A), at 1100 rpm. The gain of the proportional portion of the speed controller was adjusted so the current limit of the motor was just hit when an 1100 rpm step input speed demand was applied. The integral gain was then adjusted to produce a slightly underdamped response with 9% speed overshoot. The speed and current response is shown in Fig. 4.1. The proportional gain was 0.08 A/rad/s whereas the integral one was 0.32 A/rad.

As can be seen in Fig. 4.1, the maximum current (9 A) was reached when the step input demand was applied. As the motor speeds towards the reference value, the current reduces at the same rate, before stabilising at the final demand value. However, the speed response changes according to the load or reference speed. Some examples are shown below.

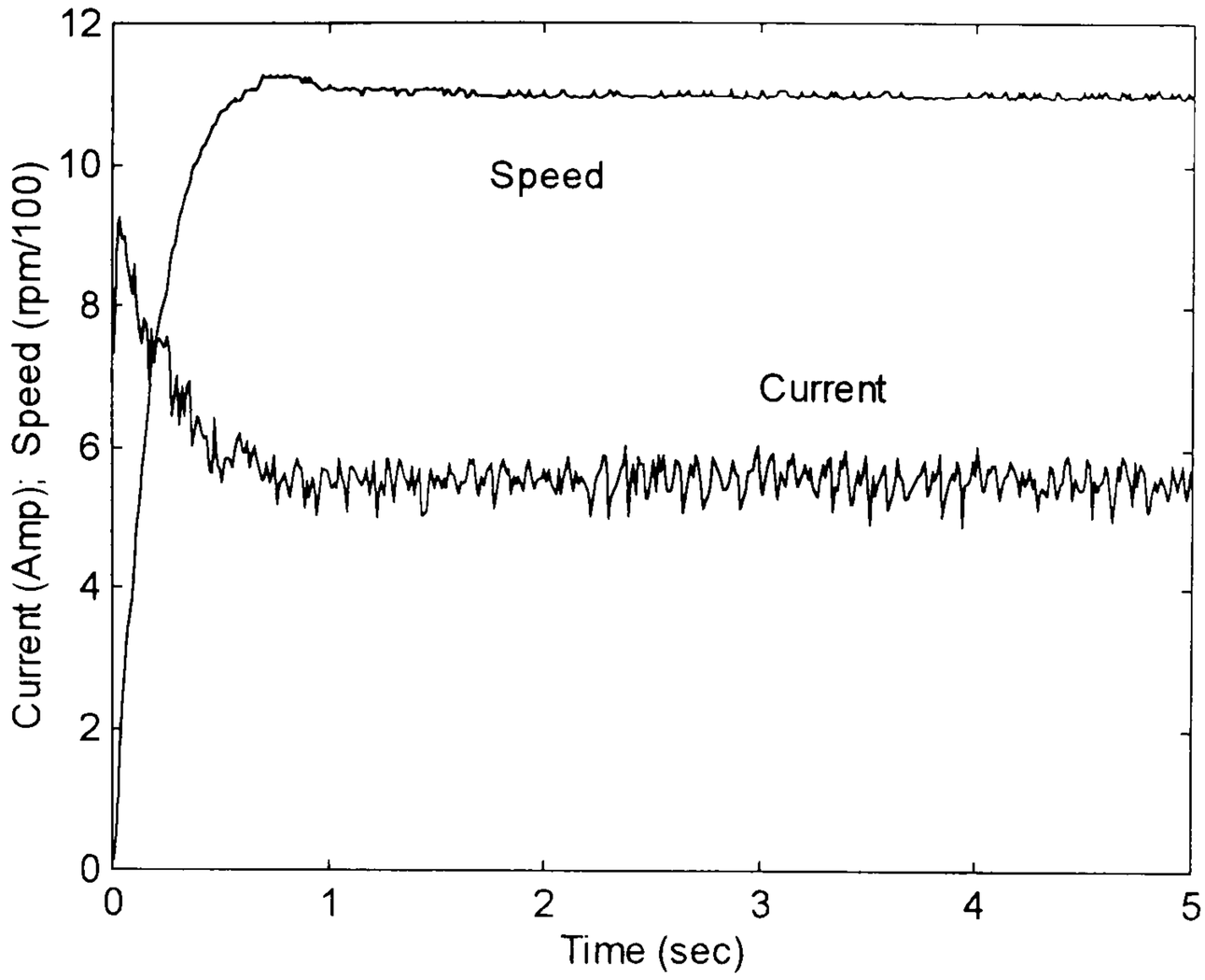



Fig. 4.1 - Speed and current response of the Brushless DC Motor Drive with standard PI speed controller.




Fig. 4.2 - Speed and current response of the motor drive due to a 1100 rpm step input speed demand in the presence of 11% of load torque.

### 4.1.1 - 1100 rpm step input speed demand with 11% of the full load

Without changing the setting of the speed controller as used in the previous section, the motor was now run at the same speed demand, 1100 rpm but with different load torque. The load resistor bank was set to produce only 11% of the maximum load torque at that speed. The speed and current response is shown in Fig. 4.2.

Because of this new condition, the speed controller applies a current demand signal bigger then the maximum accepted by the analogue output card, which is equivalent to 9A. This current demand signal is shown in Fig. 4.3. It means that there was torque saturation. As a consequence, the effect known as integrator *windup* [WIEN, 1987, VRANCIC et al, 1996, ALBERTOS, 1997 and SHIN, 1998] starts to take place resulting in 24.5% speed overshoot.

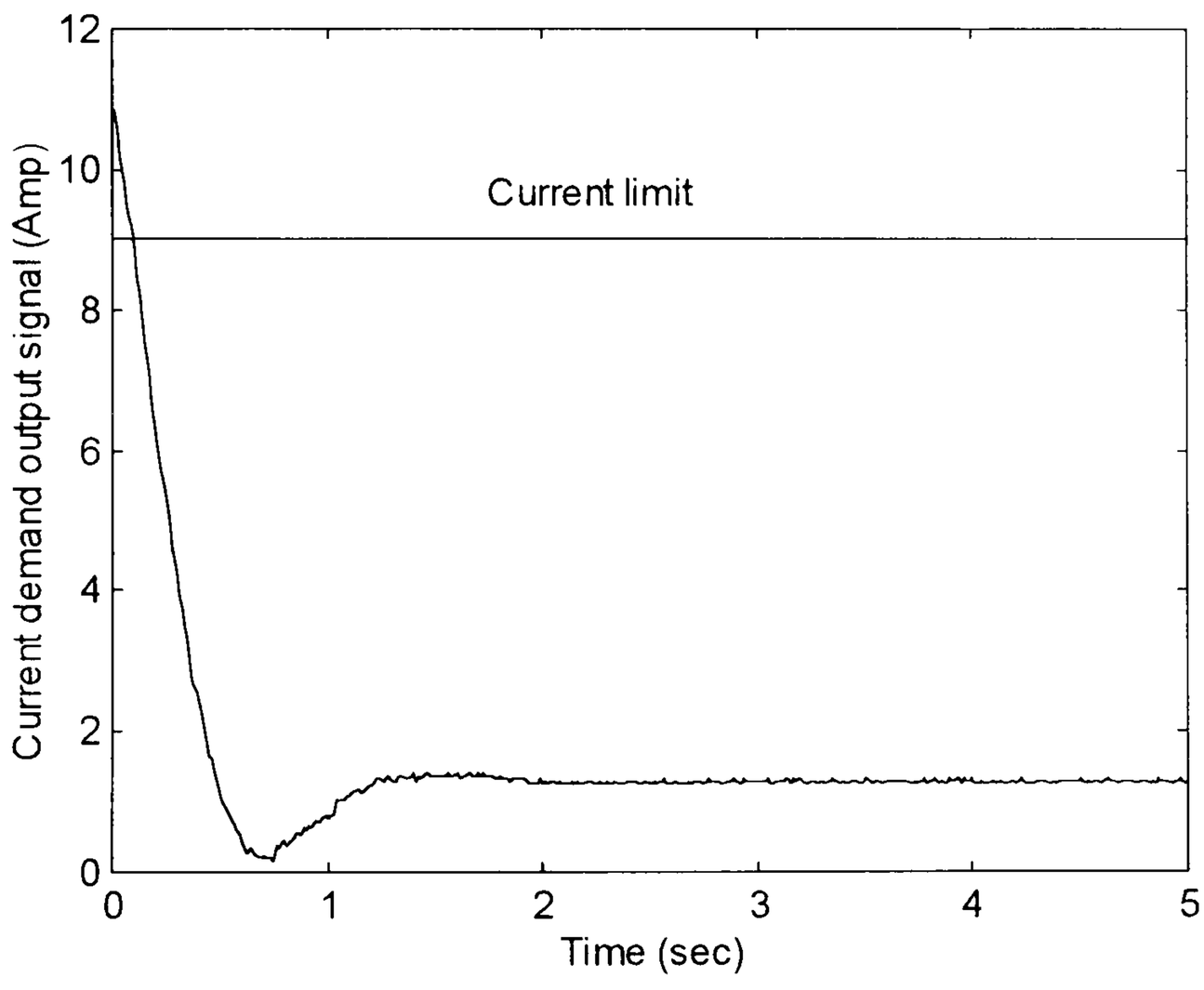



Fig. 4.3 - PI speed controller output current demand signal due to 1100 rpm step input speed demand with 11% of load torque, with settings as in section 4.1.

### 4.1.2 - 1100 rpm step input speed demand in the presence of load disturbance

In this test condition, the same step input speed demand (1100 rpm) was applied, with 11% of the load torque. However, at *time* = 1 s, a step input load disturbance was applied, remaining until the end of 5 sec time window. The speed and current

response is shown in Fig. 4.4. As can be seen, the controller could not keep the speed in the presence of the disturbance. There was about 24% speed drop, and the speed took over 1.5 s to return to its steady state value.

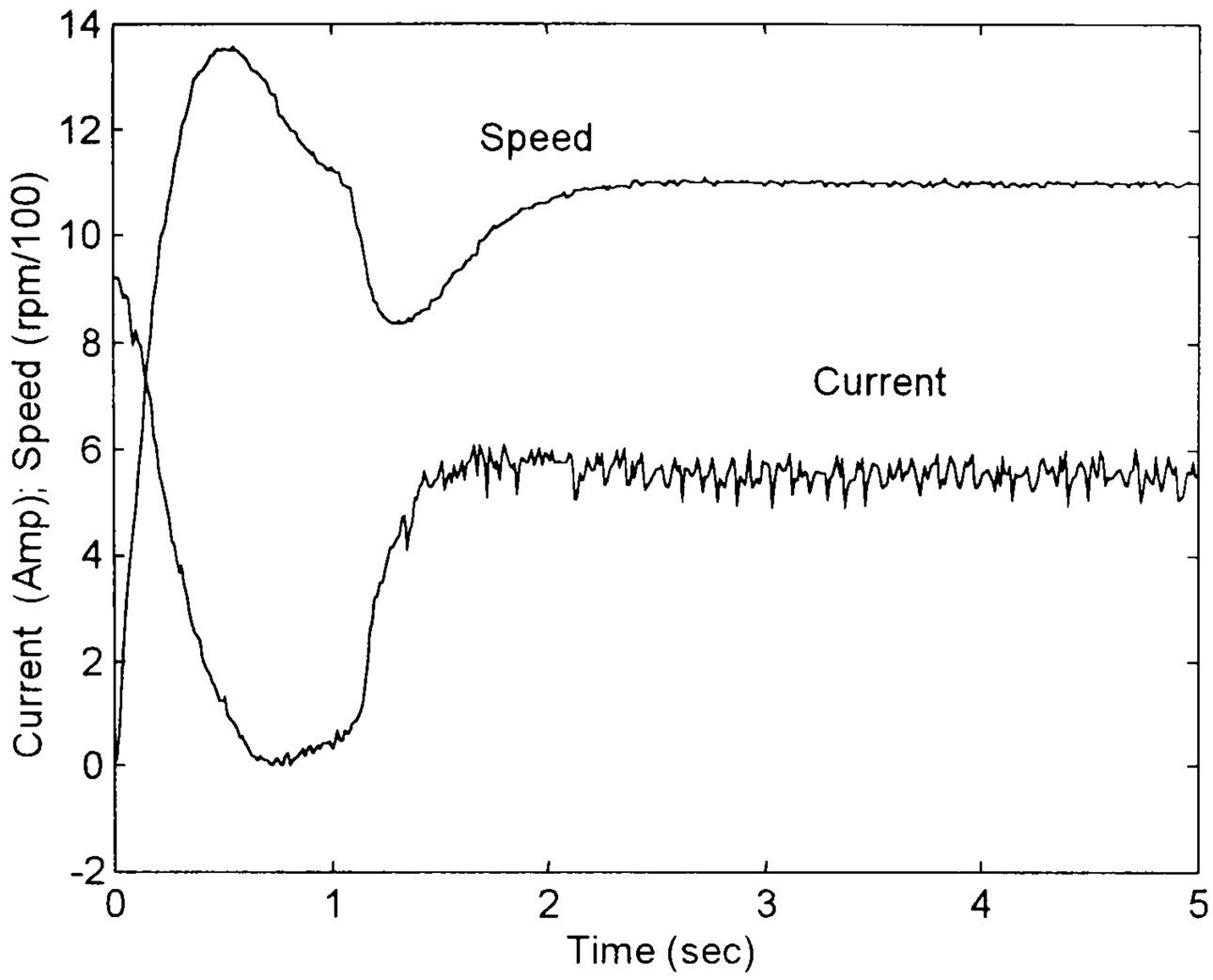



Fig. 4.4 - Speed and current response of the motor drive in the presence of load disturbance.

### 4.1.3 - 1500 rpm step input speed demand in the presence of load disturbance

This test was carried out at higher step input speed demand, 1500 rpm. The initial load was 11% of the maximum torque the motor can produce. At *time* = 1 s, a step input load equivalent to 75% of the maximum torque produced by the motor was applied, lasting for the rest of the time window. The speed overshoot was equal to 35% and the speed dropped about 28% due to the load increase, as shown in Fig. 4.5. This means that the performance of the speed response became worse when compared to that of section 4.1.1. The larger speed overshoot has happened because of the non-linearity caused by the torque saturation and the integrator's *windup*. The current demand output of the PI speed controller for this particular case is shown in Fig. 4.6. As can be seen, the output of the controller is even larger than that shown previously, lying well above the 9A current demand limit.

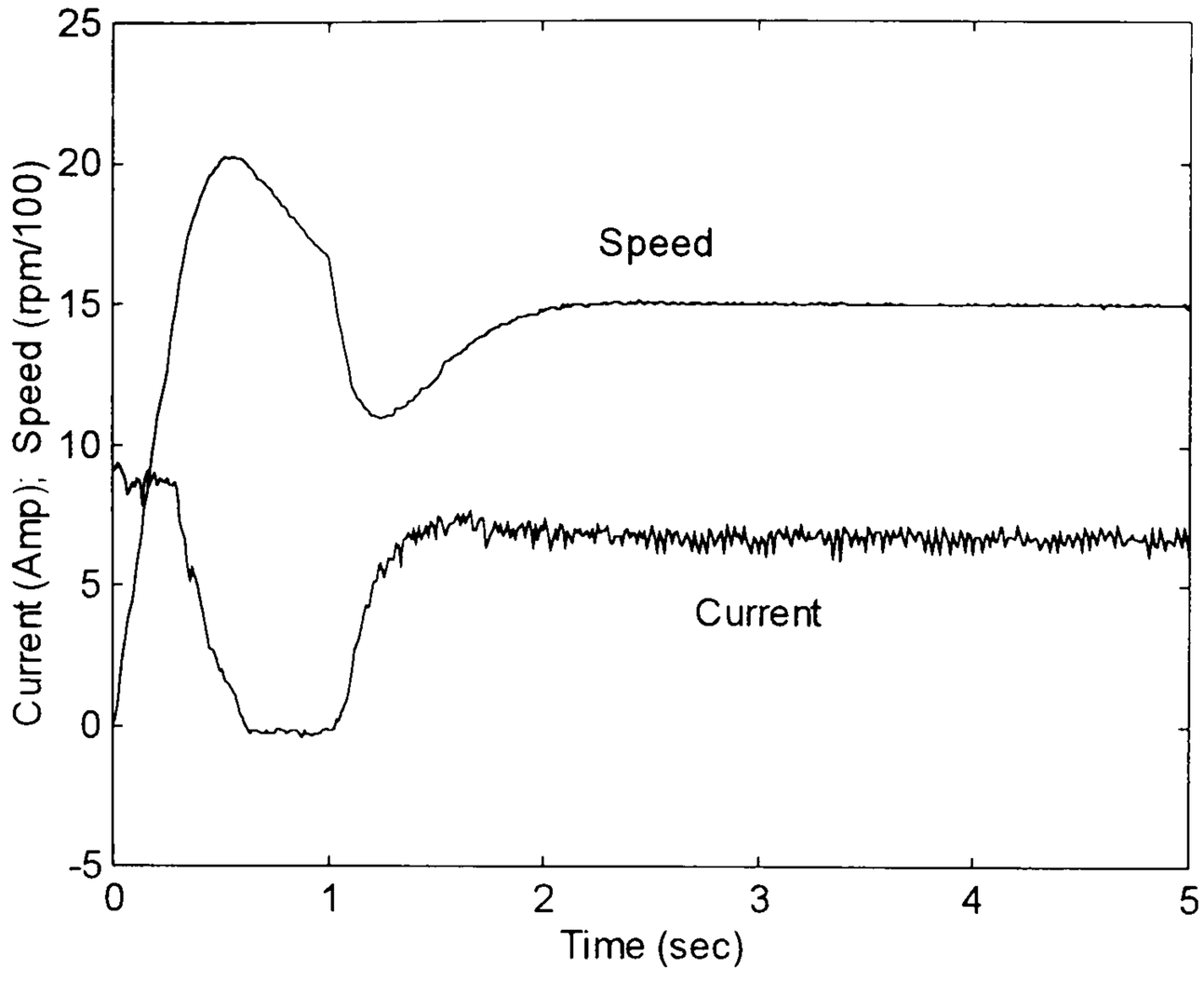



Fig. 4.5 - Speed response of the motor due to 1500 rpm step input speed demand in the presence of 75% load disturbance.

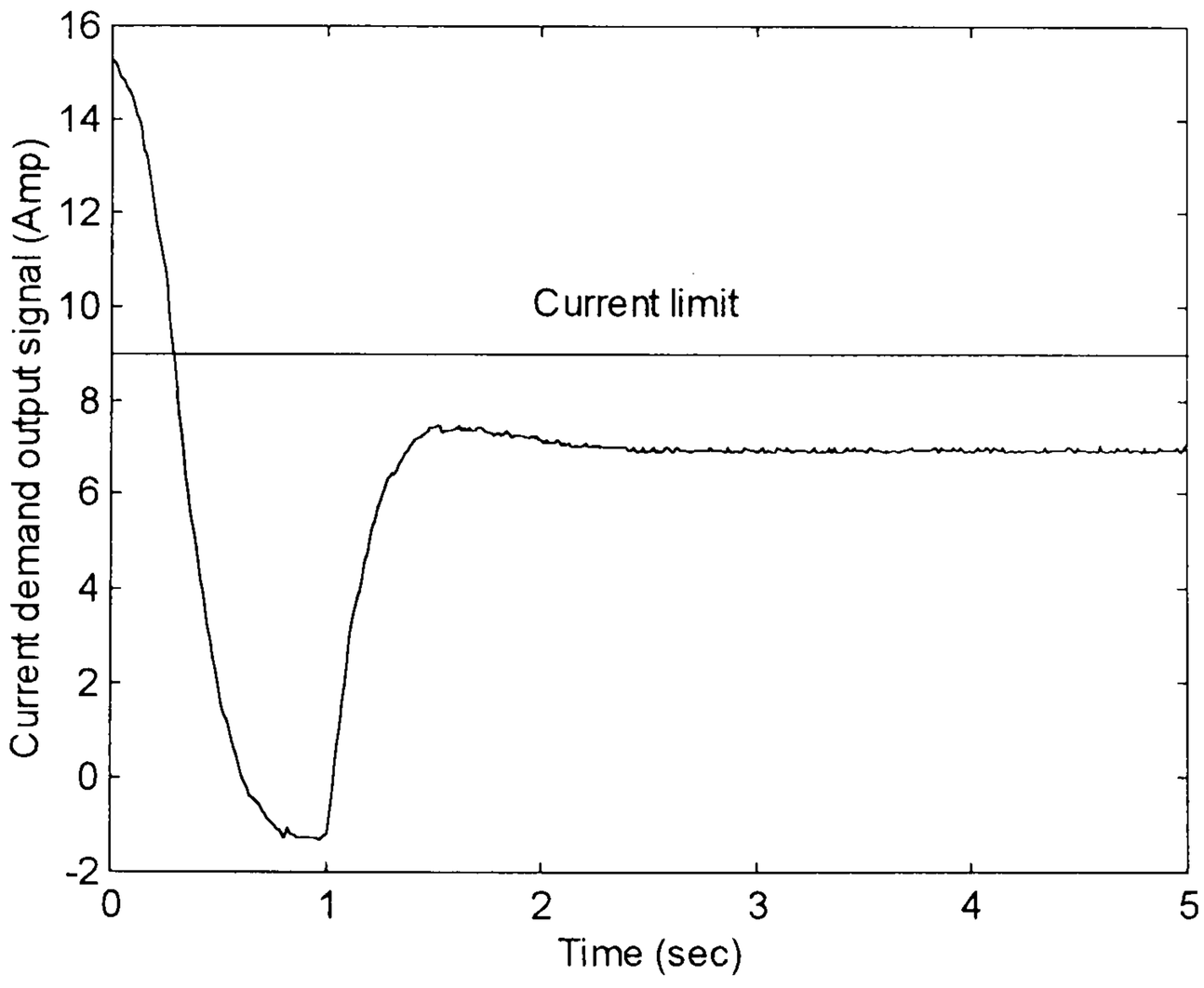



Fig. 4.6 - PI speed controller's current demand output signal.

## 4.2 - ANTI-WINDUP CIRCUIT

In order to avoid the *windup* of the integrator of the PI controller, different *anti-windup* circuits have already been used in industry. Depending on the application,

they have different performance. Concerned to the speed holding capability, the best *anti-windup* circuit is that which allows the use of a high proportional and integral gain. The reason is the following: 1 - the higher the proportional gain the smaller the speed drop due to the disturbance and 2 - the higher the integral gain, the faster the speed recovery.

An example has been chosen to illustrate how this problem can be solved. The *anti-windup* circuit presented here is the one that allows the use of high gain on the controller setting while keeping the integrator active all the time. This circuit is shown in Fig. 4.7.

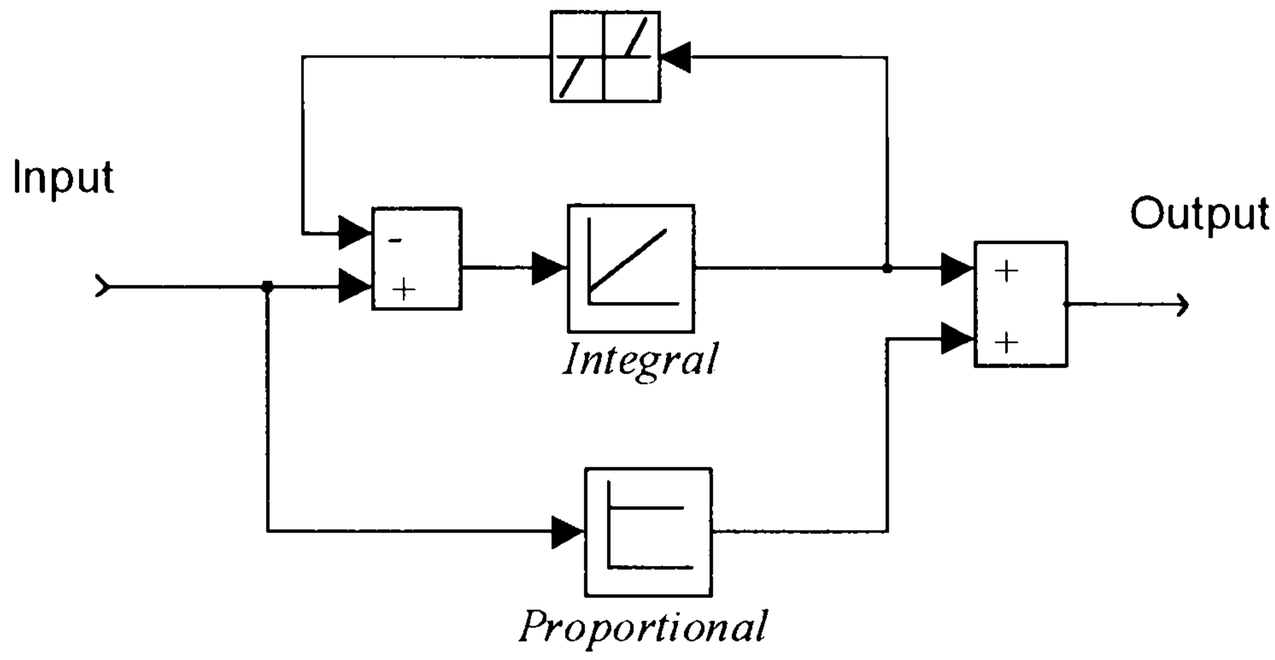



Fig. 4.7 - *Anti-windup* circuit based on the use of a dead-zone.

In this circuit, the input of the integrator is limited in accordance with the following: the dead-zone window is defined within the range (*Min* - *Max*), as shown in Fig. 4.8. It means that:

1 - When the output of the integrator, which is the *input* of the dead-zone, becomes greater then *Max*, the output of the dead-zone block becomes *Output* = *Input* - *Max*;

2 - When the output of the integrator becomes smaller then *Min*, the output of the dead-zone becomes *Output* = *Input* - *Min*.

3 - For any *Input* of the dead-zone between the range (*Min* - *Max*), the *Output* is zero.

It means that the input of the integrator is limited, however, it is active all the time whereas some other approaches usually turn off the integrator input by setting it to zero.

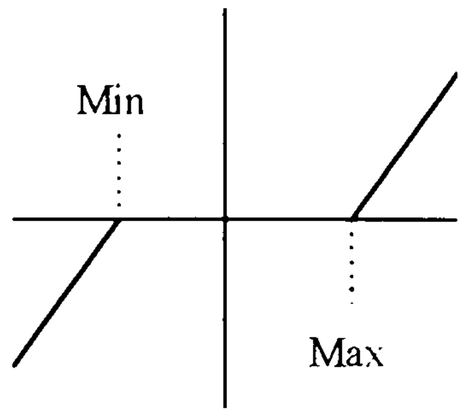



Fig. 4.8 - Dead-zone block showing the dead-zone window (*Min* - *Max*).

Thus, by adjusting the dead-zone window, the amount of speed error that goes into the integrator is limited.

The following examples illustrate how it works. The proportional and integral gain of the PI speed controller was adjusted to 0.5 Amp/rad/s and 1 A/rad respectively. The dead-zone window was set to be very high so its window limit (*Min* - *Max*) would never be reached. By doing this, the speed error could go into the integrator without suffering any reduction. As a consequence, there was no limitation on the input of the integrator and no non-linearity other then controller limits was introduced on the drive system.

The motor had to respond to an 1100 rpm step input speed demand with 62% of the maximum load torque. This setting is high enough to cause integrator *windup*. The speed and current response of the motor within this condition is shown in Fig. 4.9. The saturation is apparent as the maximum current (9 A), was applied during all the acceleration time. Due to the integrator *windup*, there was a speed overshoot of 17%. However, if the dead-zone window is set to (-10 A to 10 A), the speed response becomes as shown in Fig. 4.10.

The speed overshoot is smaller, only 6.4%. It is apparent that the current remains at its maximum value for shorter period of time, as illustrated in fig. 4.11

Should *Max* be set exactly to the current demand at steady state, there is no overshoot at all. Such a response can be seen in Fig. 4.12.

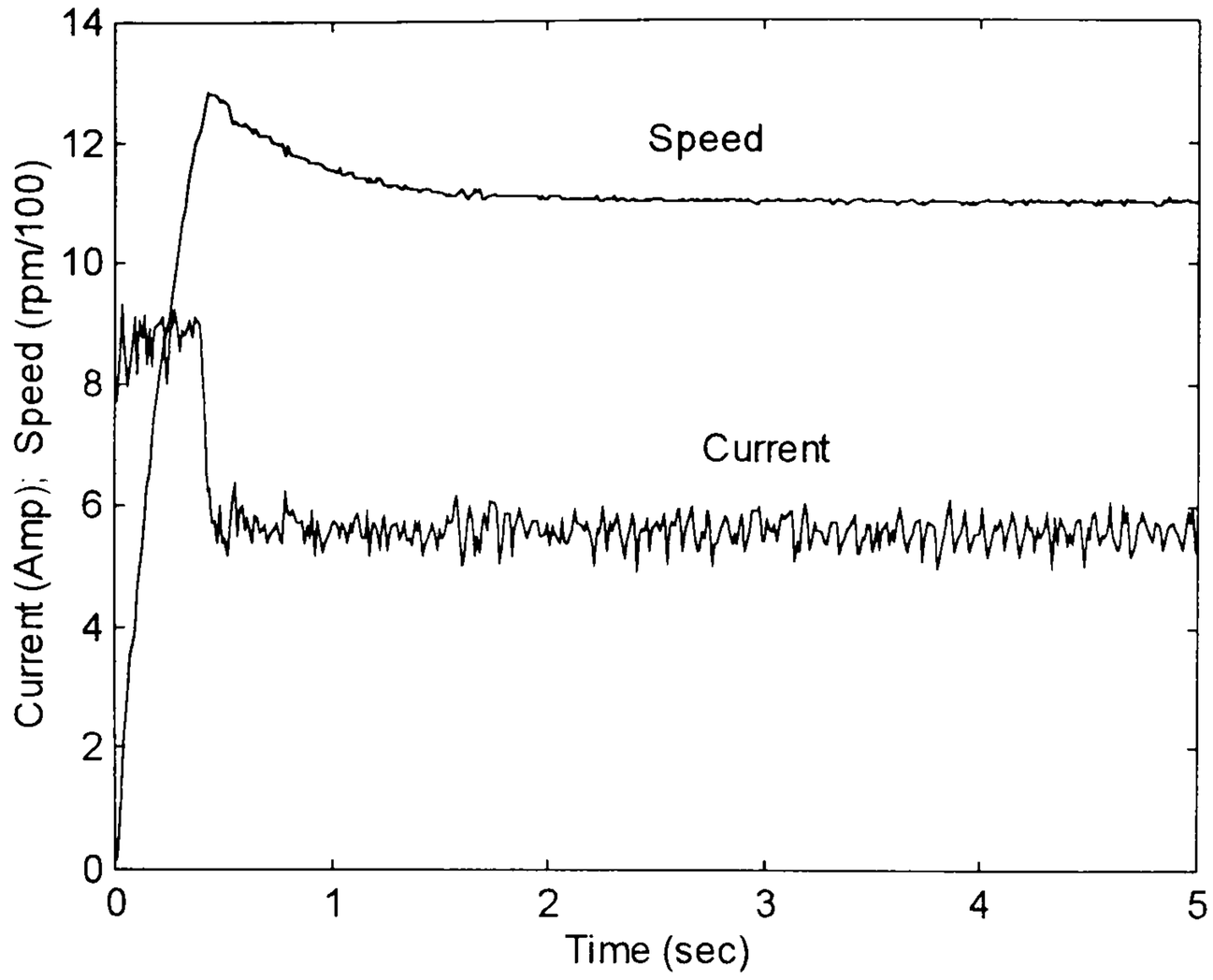



Fig. 4.9 - 1100 rpm - 62% of load step input speed and current response of the brushless with proportional and integral gain of the speed controller set to 0.5 A/rad/s and 1 A/rad respectively and no dead-zone anti-windup.

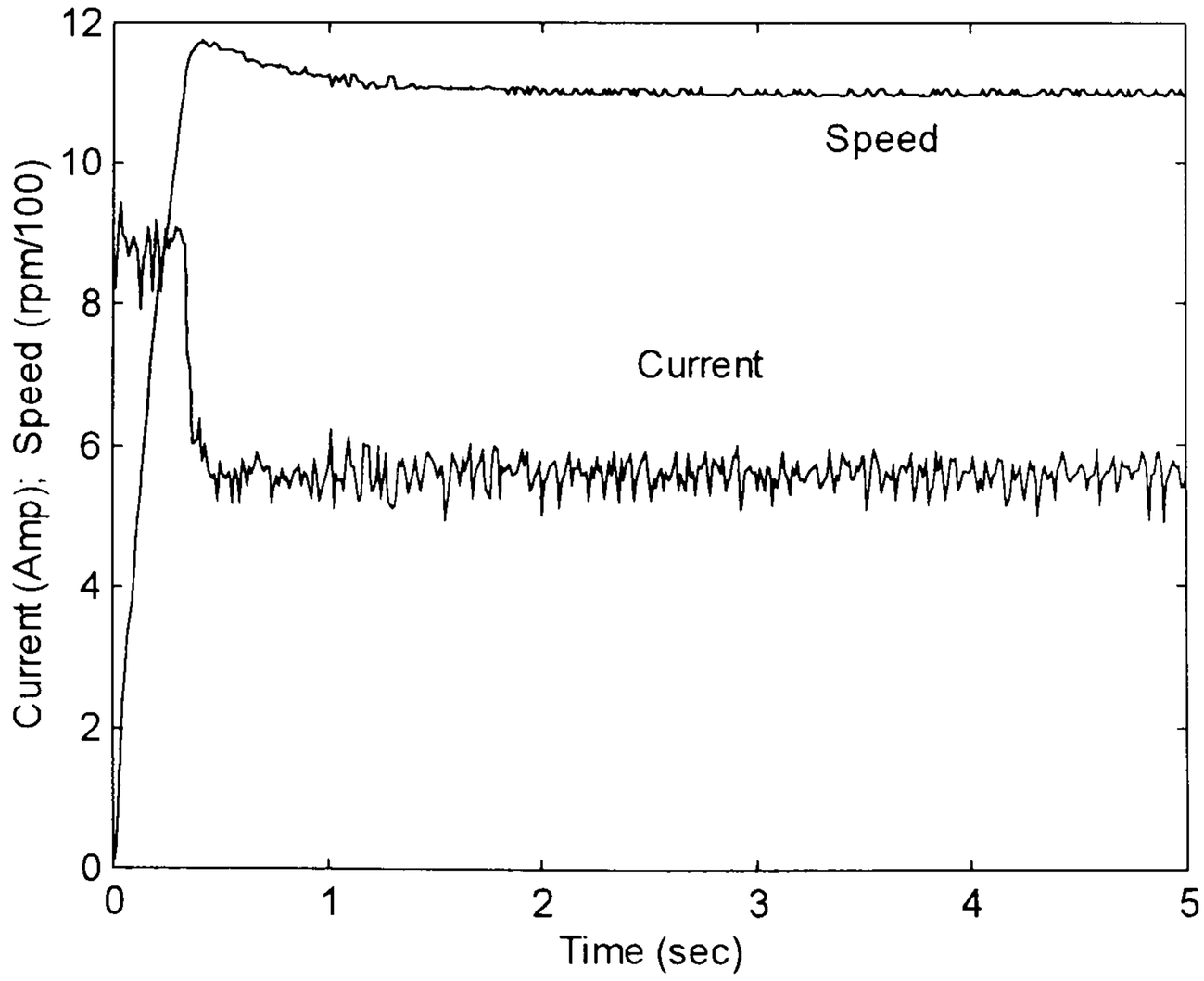



Fig. 4.10 - 1100 rpm - 62% of load step input speed and current response of the brushless with proportional and integral gain of the speed controller set to 0.5 A/rad/s and 1 A/rad respectively with (-10 A to 10 A) dead-zone window.

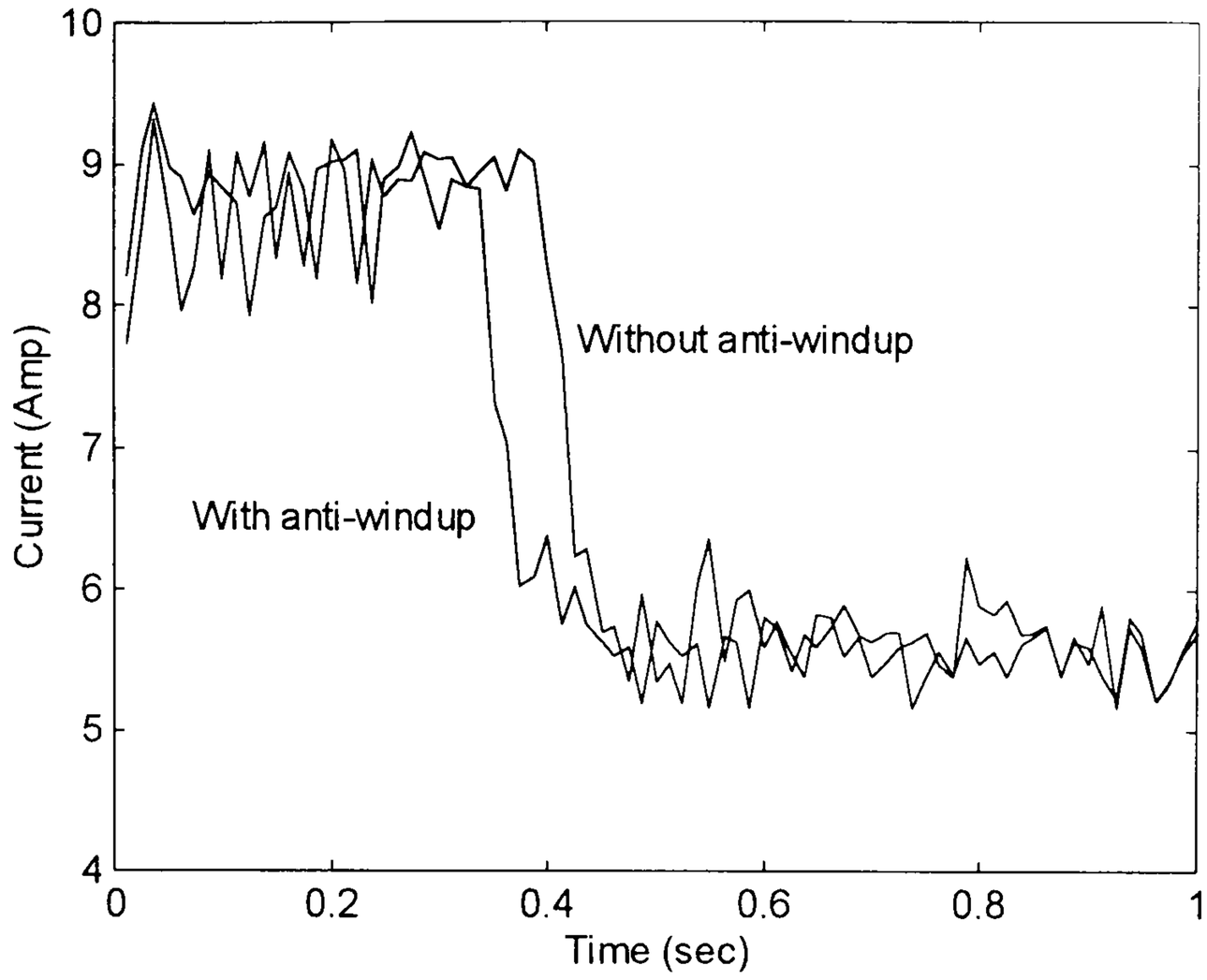



Fig. 4.11 - Current response of the motor as shown in Figs. 4.9 and 4.10, in enlarged scale, highlighting the time it remains at its maximum with and without *anti-windup*.

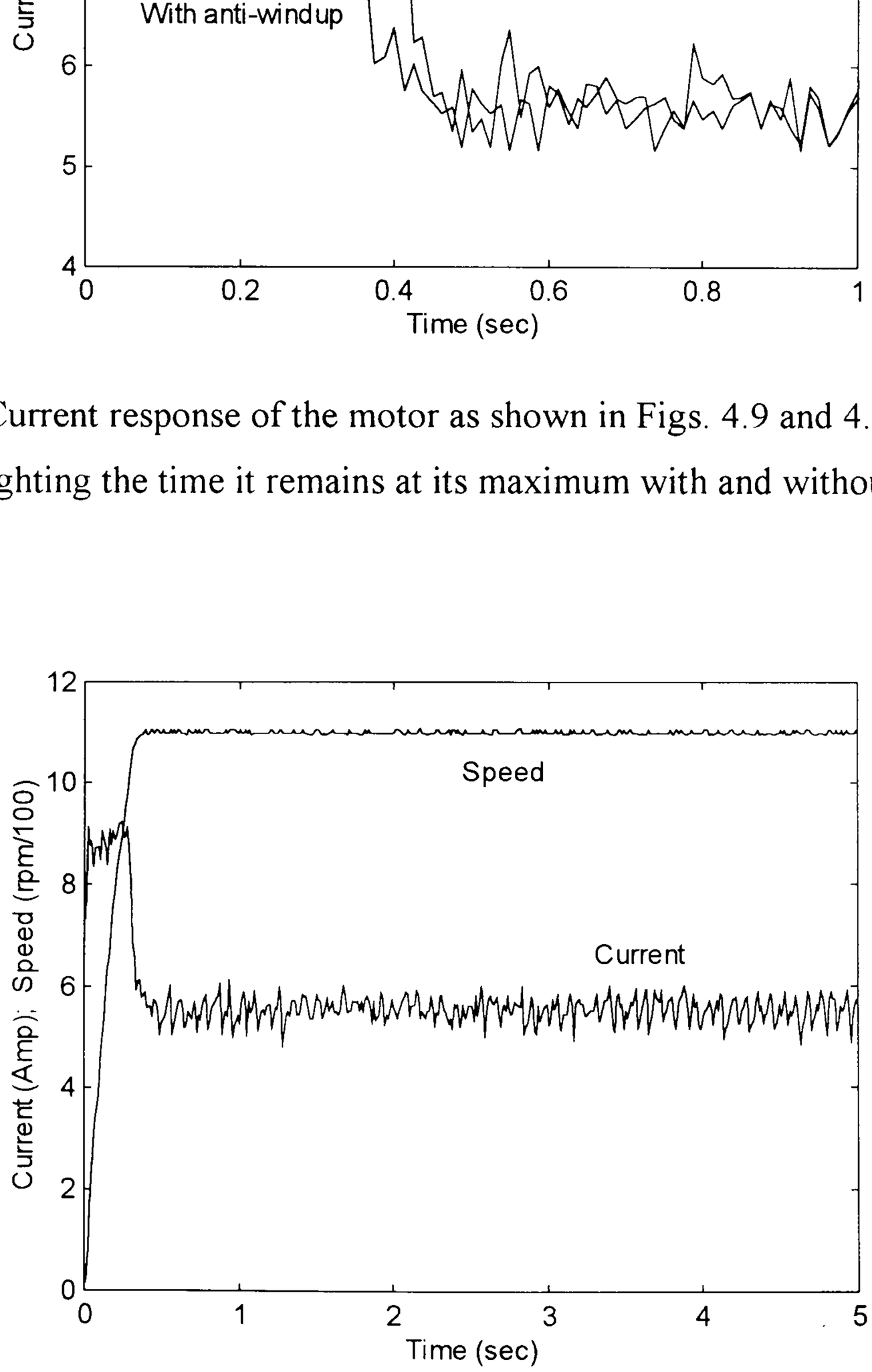



Fig. 4.12 - 1100 rpm - 62% of load step input speed and current response of the brushless with proportional and integral gain of the speed controller set to 0.5 A/rad/s and 1 A/rad respectively together with (-5.83 A to 5.83 A) dead-zone window.

It is important to highlight that the dead-zone window is exactly the same as the current demand the motor needs to produce torque capable of counterbalancing the load torque. The motor accelerates with its maximum torque, without speed overshoot. However, because of the *anti-windup* circuit the control becomes non-linear and its tuning is good only for one specific condition. Should the load torque or speed demand change, the controller has to be re-tuned otherwise the speed performance deteriorates.

In order to illustrate this problem, the drive was run at two different conditions although, the controller tuning was unaltered. Firstly, with only 11% of the full load and secondly, at 1500 rpm with 75% load torque. The responses are shown in Figs. 4.13 and 4.14.

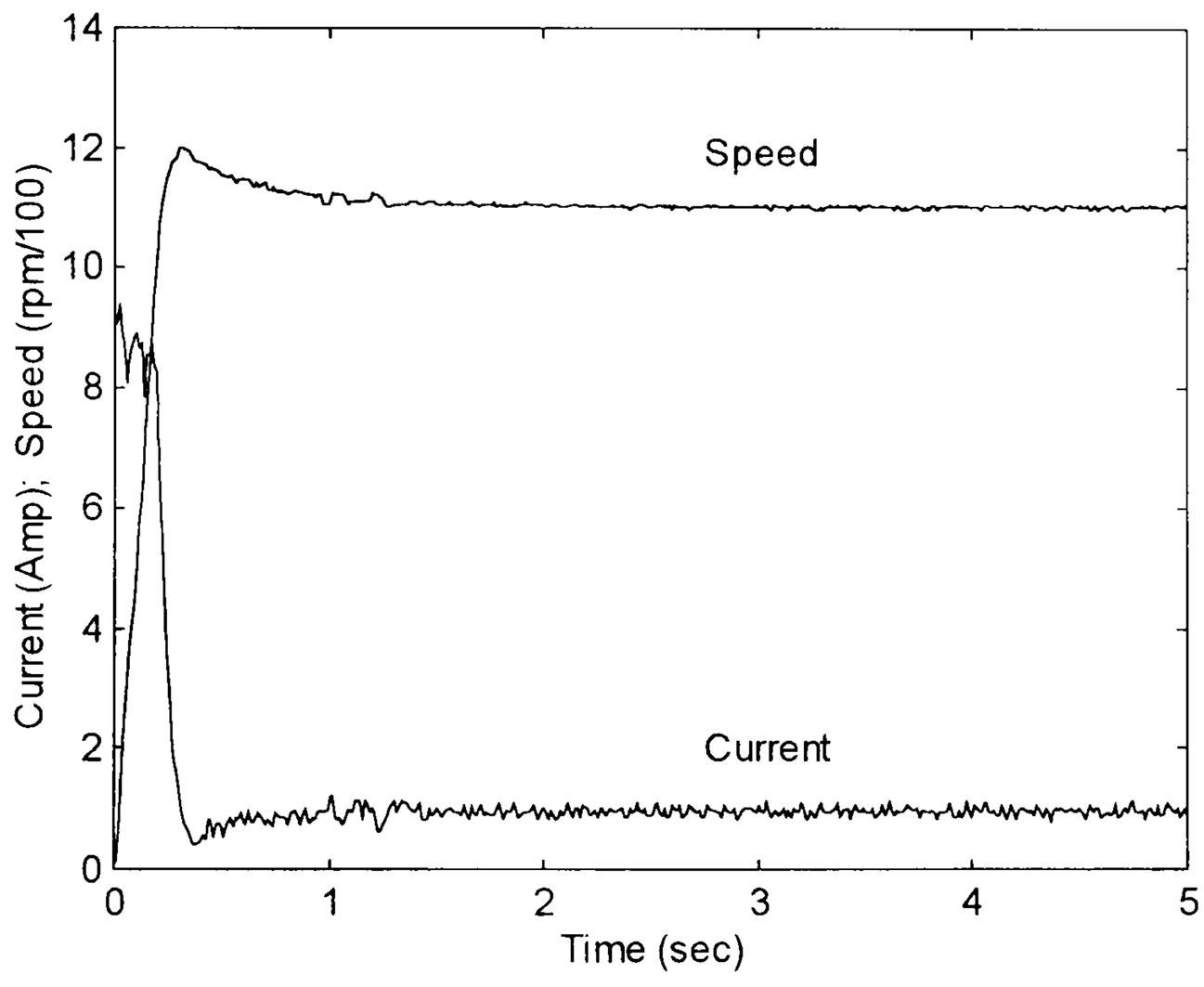



Fig. 4.13 - 1100 rpm - 11% of load step input speed and current response of the brushless with proportional and integral gain of the speed controller set to 0.5 A/rad/s and 1 A/rad respectively, together with (-5.83 A to 5.83 A) dead-zone window.

In Fig. 4.13 there was a speed overshoot because the integrator went past the current demand at steady state. On the contrary, in Fig. 4.14, there is a steady state speed error, as the *anti-windup* circuit does not let the integrator go up to the current demand.

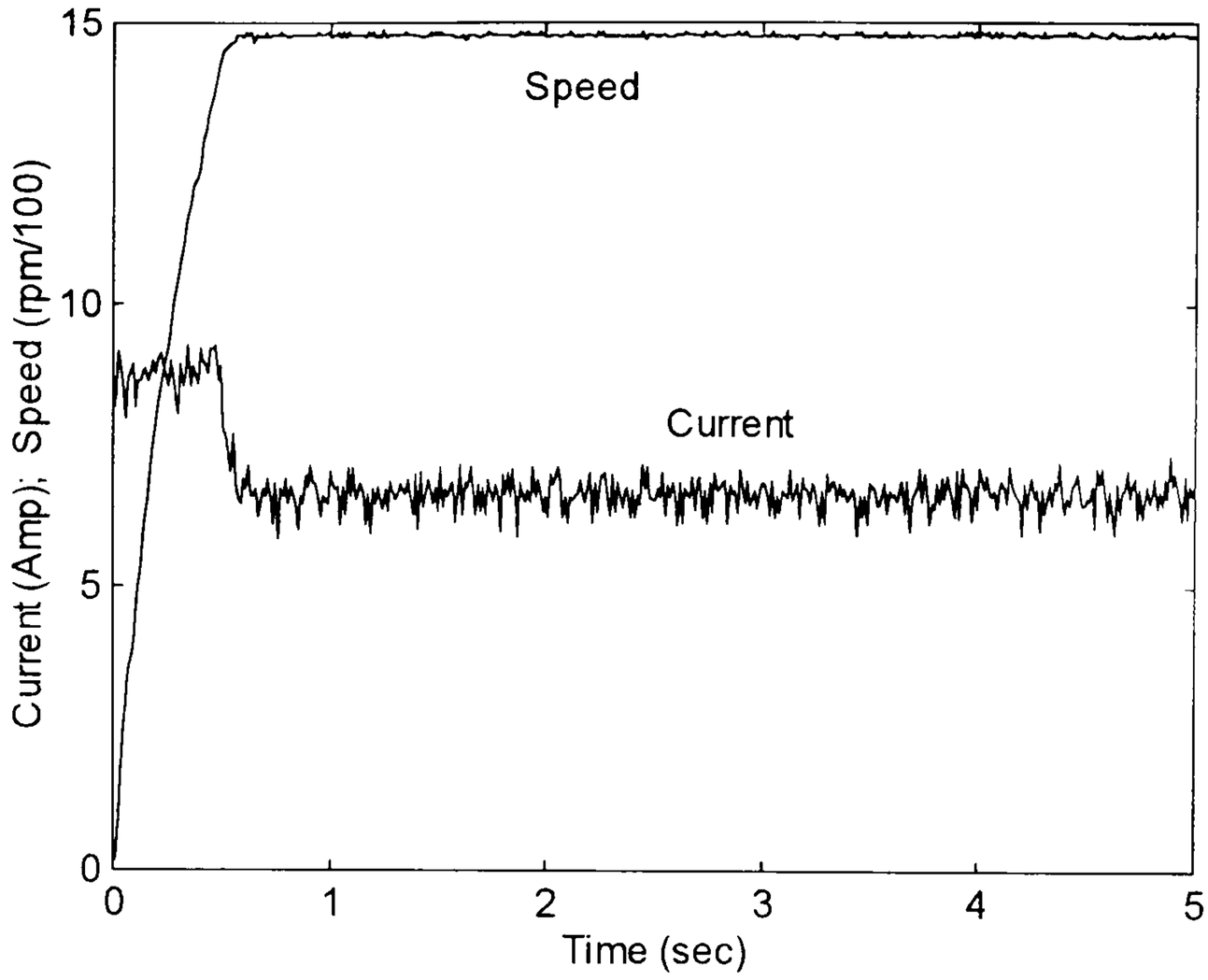



Fig. 4.14 - 1500 rpm - 75% of load step input speed and current response of the brushless with proportional and integral gain of the speed controller set to 0.5 A/rad/s and 1 A/rad respectively, together with (-5.83 A to 5.83 A) dead-zone window.

## 4.3 - CONCLUSION

In this Chapter, the speed response of the brushless DC motor drive with a well-known PI speed controller, in the classical cascade configuration with a PI current controller, was investigated. It has been shown that non-linearity such as torque saturation imposed by the maximum current that can be driven into the motor or through the drive devices, can cause integrator *windup*. Depending on the controller settings, this problem becomes more apparent. Concerning to the speed control in the presence of load disturbance, the higher the proportional gain of the PI speed controller, the smaller the speed drop. The higher the integral gain, the faster the speed returns to its steady state value. However, depending on the step input reference speed and load applied to the motor, a high controller gain setting can lead to the *windup* of the integrator. *Anti-windup* circuits can be used in order to place a limitation on the integrator. On the other hand, should the conditions in which the

motor is working change, another setting is necessary, not only with but also without the *anti-windup* circuit. The use of limitations on the integrator allows us to use higher gains, and increase the dynamic performance of the controller.

# CHAPTER 5

# A GENETIC ALGORITHM APPLIED TO THE ON LINE OPTIMIZATION OF A STANDARD PROPORTIONAL-INTEGRAL SPEED CONTROLLER

## 5.1 - INTRODUCTION

Genetic algorithms are finding widespread applications in system optimisation problems, including the design of electrical machines [BIANCHI, et al, 1998]. In comparison to other optimisation methods, such as steepest descent, genetic algorithms are especially successful at avoiding local minima, which is a particular feature of non-linear systems. However genetic algorithms have the reputation of being slow to arrive at an optimum solution in well conditioned problems, because the entire range of potential solutions is investigated.

The tuning of electric drive controllers is recognised as a complex problem due to the many non-linearities of the machine, power converter and controller. Even in the relatively simple example of the cascade control of a Brushless DC Drive as shown in Fig. 3.1 and reprinted in Fig. 5.1, non-linearities arise in the motor from magnetic saturation and commutation effects as dc current is switched between pairs of motor windings. The power converter imposes a limit on output current due to the device ratings and has a non-linear transfer function characteristic because of dead times introduced into the device switching to protect from shoot-through. Meanwhile the current controller exhibits speed-dependent behaviour as it uses a fixed dc link voltage to force current into the motor against a motional emf. Therefore the problem of drive controller tuning is appropriate for the application of genetic algorithms.

If full account is to be taken of all sources of non-linearity, the most effective approach is to work directly on the on-line tuning of the drive. For the purposes of

illustration this chapter concentrates on finding the optimum settings for the parameters of a Proportional-Integral speed controller for a Brushless DC Motor Drive with the cascade control structure shown in Fig. 5.1. Although this is a relatively simple problem, because values for just two parameters (proportional gain and integral gain) are being sought, it is complicated by the presence of a significant non-linearity caused by saturation of the current controller. The non-linearity ensures that optimum controller settings depend on the form of the input stimulus, for example the magnitude of a step change in speed demand.

After describing the basic principles of genetic algorithms, in this chapter, experimental results about the evolution of optimum controller settings during on-line testing are presented. The dependence of the optimum controller settings on the size of the step input is explored and comparisons are made with controller settings derived using the assumption that non-linearities can be neglected.

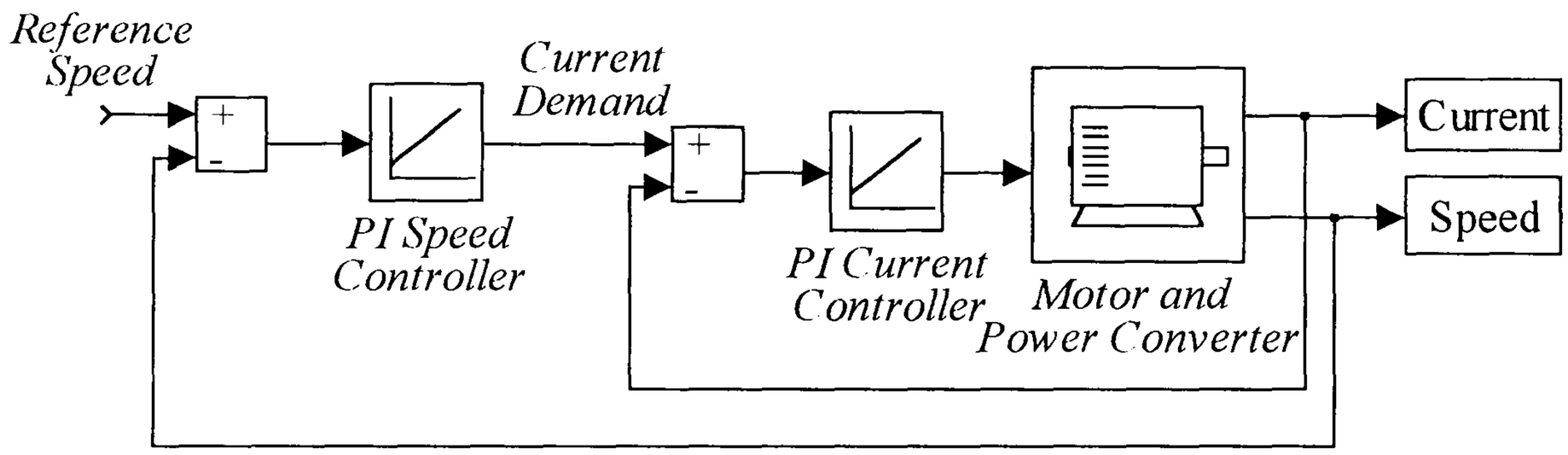



Fig. 5.1 - Proporional-Integral speed controlled Brushless DC Motor block diagram.

## 5.2 - INTRODUCTION TO GENETIC ALGORITHMS

Gen and Cheng [GEN et al, 1997] report that there has been an increasing interest since the 1960s in attempts to solve complex problems by imitating the process of evolution, where the fittest individuals are likely to survive in a competing environment. A genetic algorithm [MAN, K. et al, 1996 and CHIPPERFIELD, University of Sheffield] starts with a random population of potential individuals, or chromosomes, each representing one possible solution to a problem. A chromosome is formed by a number of genes, usually, but not necessarily, coded by using binary numbers. The chromosomes are then evolved through successive generations.

During each generation, all the chromosomes are evaluated, according to a defined fitness criterion, and the best chromosomes are selected to mate and generate offspring. The least fit chromosomes of each population are then replaced by the offspring so that the population size remains constant. After several generations, the algorithm converges to the best chromosome which represents an optimal solution to the problem.

A further refinement of the evolution process, again mirroring nature, is that any chromosome in any generation has a finite probability of suffering mutation, in which some of the genes are randomly perturbed. It is this process which ensures that the genetic algorithm does not converge to a local minimum when searching for an optimum problem solution.

## 5.3 - APPLICATION OF GENETIC ALGORITHMS TO THE TUNING OF A PI SPEED CONTROLLER

For the solution of any particular problem using a genetic algorithm, there are two important aspects: i) coding the chromosomes and ii) defining the evaluation criteria.

When tuning the PI speed controller, two parameters have to be adjusted, the proportional and integral gains. Hence, each chromosome or individual has to include values for each of these parameters and therefore the genes of the chromosomes are possible gain values for the controller:

Chromosome = $[K_p\ K_i]$

where: $K_p$ = proportional gain; $K_i$ = integral gain.

The chromosomes can be coded by using binary or real numbers. When using binary strings, the length of the string depends on the precision required and range of the variables. As a consequence, several digits may be needed to represent one single variable and the chromosome may become long. However, by using real numbers, the chromosome length is reduced and becomes easier to understand, as each gene of the chromosome is expressed by its real value, in decimal form.

During the evolution process the fitness of each chromosome must be evaluated using an appropriate quantitative criterion. In this study the drive's response to a step

change in speed demand is being assessed and therefore an appropriate criterion is the Integral with respect to Time of the Absolute speed Error (ITAE) [DaSilva *et al*, EPE'97]. The chromosome representing the best tuning is associated with the smallest ITAE.

## 5.4 - EXPERIMENTAL RESULTS

The efficiency of genetic algorithm tuning of a Brushless DC Drive's PI speed controller was tested using a commercial drive as described in Chapter 3.

The genetic algorithm itself was implemented in MATLAB on the host PC. For each chromosome in each generation the relevant proportional and integral gains were downloaded to the speed controller resident in the DSP. The appropriate step response was initiated automatically and the speed and current signals as well as the ITAE for the duration of an appropriate time window were passed back to the host. The host PC runs only the Genetic Algorithm, based on the information passed back from the DSP. Therefore the entire tuning process is completely automatic: there is no need for manual intervention either between successive speed responses or between successive generations.

In order to start, definition of the limits of the controller's gains is necessary, so that the chromosomes can be initialised over an appropriate range. In a practical situation these limits could be defined by previous experience, from the results of off-line simulation studies or by the application of classical control theory neglecting non-linearities. For the purposes of this study, the ranges of the proportional (0-10.0 A/rad/s) and integral (0-20.0 A/rad) gains have been set exceptionally wide to illustrate the genetic algorithm's operation to maximum effect.

The parameters of Genetic algorithm such population size, generation gap and mutation rate varies from problem to problem. There is no rule in the literature to be followed although some guidelines can be found. From the experience, for the problem of optimising the PI controller, a population size of 20 chromosomes has shown satisfactory although it is considered small in standard use of Genetic Algorithm [Man et al, 1996]. The generation gap, which defines the number of individuals selected to mate and generate offspring, is 0.9, indicating that 90% of the

chromosomes are mated in each generation. Intermediate recombination [CHIPPERFIELD, University of Sheffield] is the method used to exchange genes when mating the chromosomes.

The mutation rate is another parameter that has to be defined. Although there is no specific rule for defining this parameter, Man et al [Man et al, 1996] has suggested a mutation probability between 0.001, for a large population (100 individuals) and 0.01 for a small population (30 individuals), when the population is represented by using binary numbers. Chipperfield [CHIPPERFIELD, University of Sheffield] has suggested a mutation probability equal to the inverse of the number of variables of each chromosome or individual, for a real value representation. In this particular case, since the individuals are composed of two variables (the proportional and the integral gain of the controller), the probability is as high as 50%. This value has been used. The speed response is evaluated over a 5 s time window, the duration selected so that the transient is completed and the motor speed settled to the steady-state. Thus each chromosome is evaluated on-line in approximately 8.5 s and the entire tuning process takes place in around 50 min.

The following sections present results from two examples of on-line tuning using the genetic algorithm.

### 5.4.1 - Optimisation for a large step input speed demand (0 to 1100 rpm)

In the first example, the genetic algorithm was used on-line to derive the tuning of the controller for large step input of speed demand (0-1100 rpm). The major non-linearity of the drive is a current limit of 9.0 A, which translates into a torque limit of 8.2 Nm. The resistive load bank was set to a value which imposed a load torque of 67% of the torque limit at 1100 rpm.

The chromosomes of the first generation are created by evenly distributing 17 chromosomes within the search space, and placing a further 3 chromosomes at random, as shown in Fig. 5.2. Fig. 5.3 shows the speed responses obtained with some of these chromosomes, together with the best speed response obtained in the first population

As the genetic algorithm progresses, the chromosomes evolve towards the chromosome that produces the best speed response, as judged by minimum ITAE. The evolution process is shown in Fig. 5.4, where the ITAE of the best chromosome of each generation is plotted against generation number. The minimum ITAE of 17.02 rad is produced by a chromosome, which evolves after only 11 generations and no further improvement in ITAE, occurs between generations 11 and 20. However during these generations more chromosomes cluster around the best chromosome, giving reassurance that the optimum has been clearly identified.

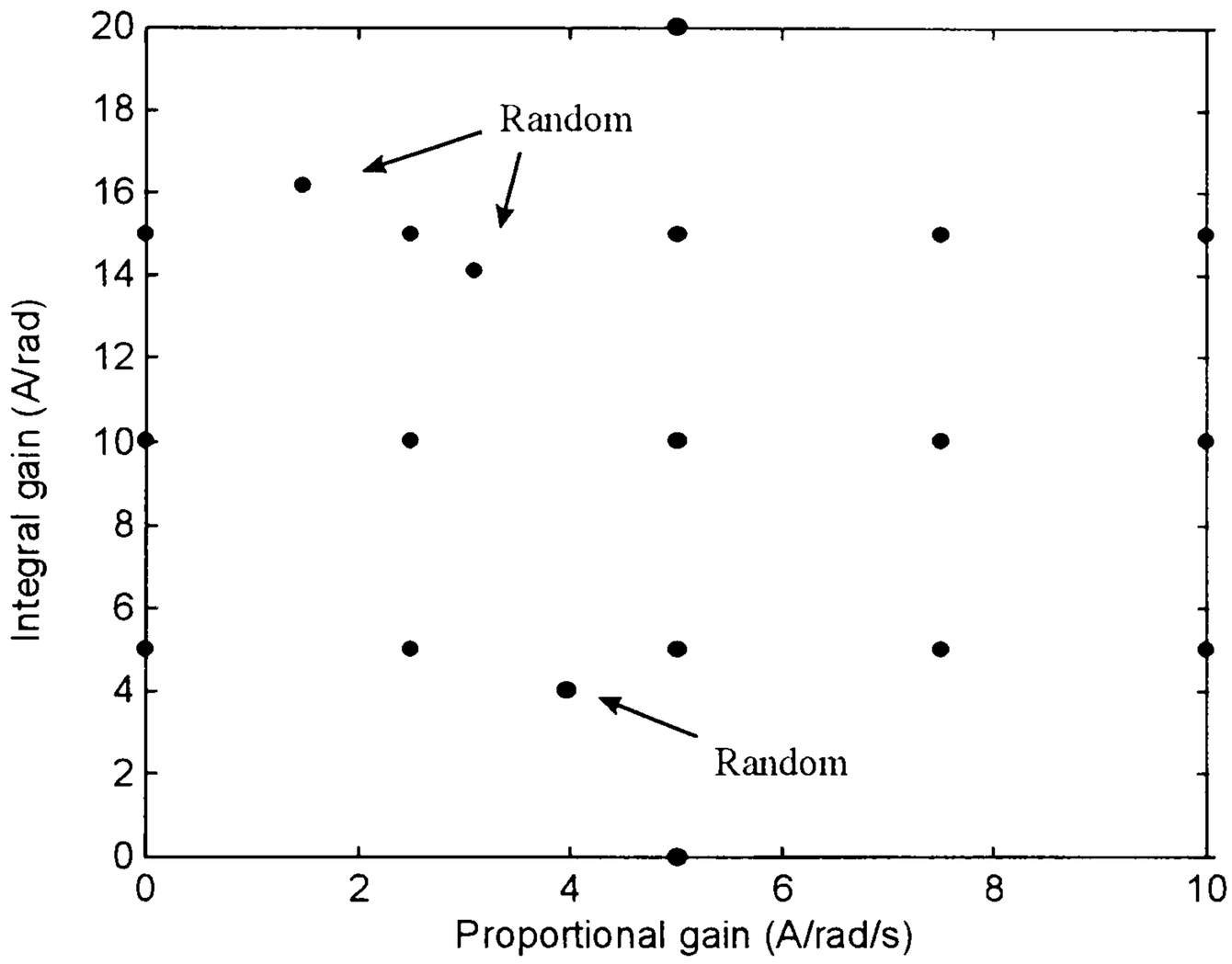



Fig. 5.2 Initial population of chromosomes.

The best speed response within the first population was obtained with the chromosome [5 0], which represents a simple proportional controller with a gain of 5 A/rad/s. There was no speed overshoot but a steady state speed error.

Fig. 5.5 illustrates the evolution process by plotting the position of chromosomes from the first and last generations in the proportional-integral plane. Initially the chromosomes are randomly scattered across the plane, but after 20 generations they are clustered around the fittest individual, which is the chromosome [5.2 0.4]. The speed and current responses obtained with these controller settings are shown in

Fig. 5.6, where the influence of current limiting during the first 0.4 s of the speed transient is apparent.

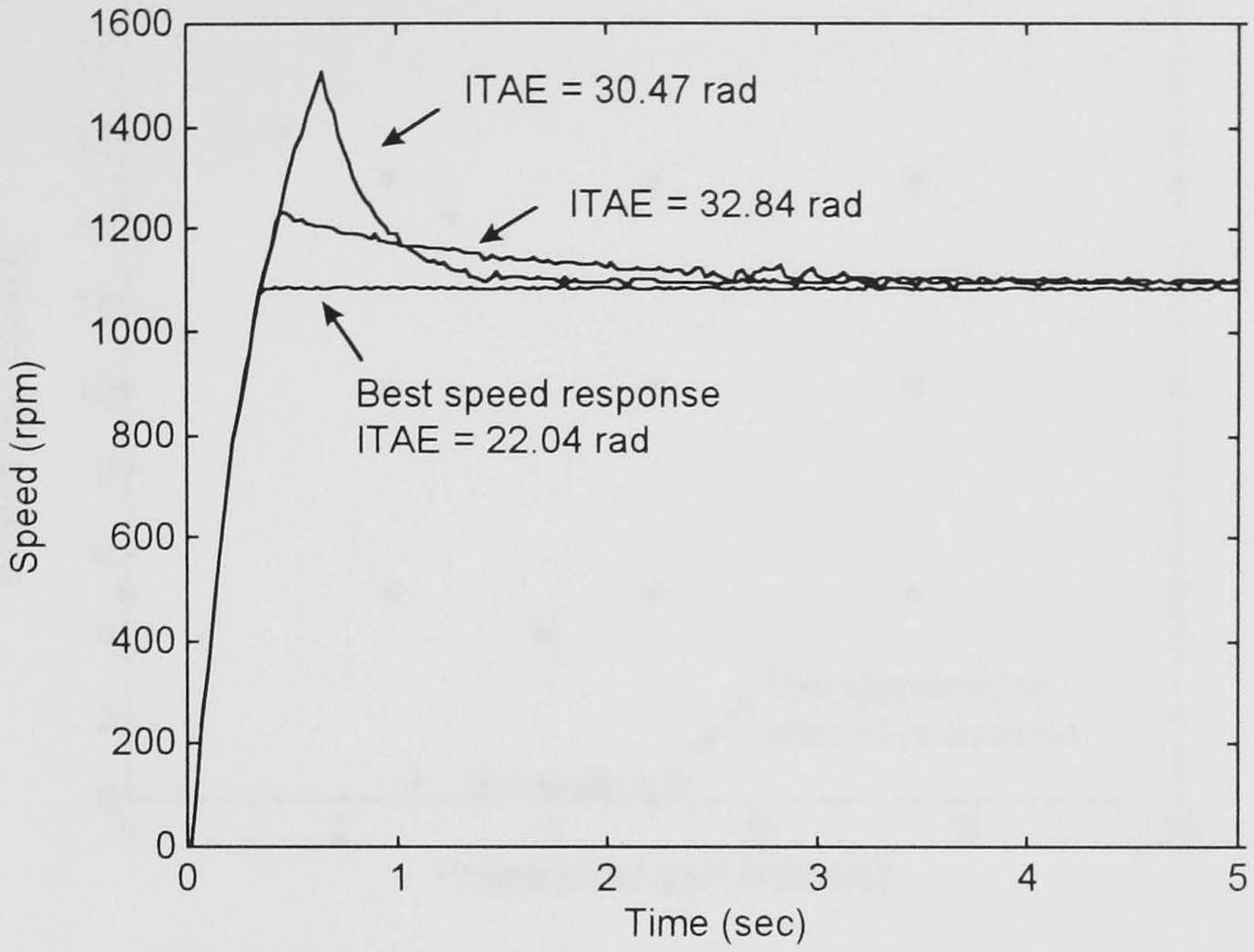



Fig. 5.3 - Some speed responses obtained within the initial population of chromosomes.

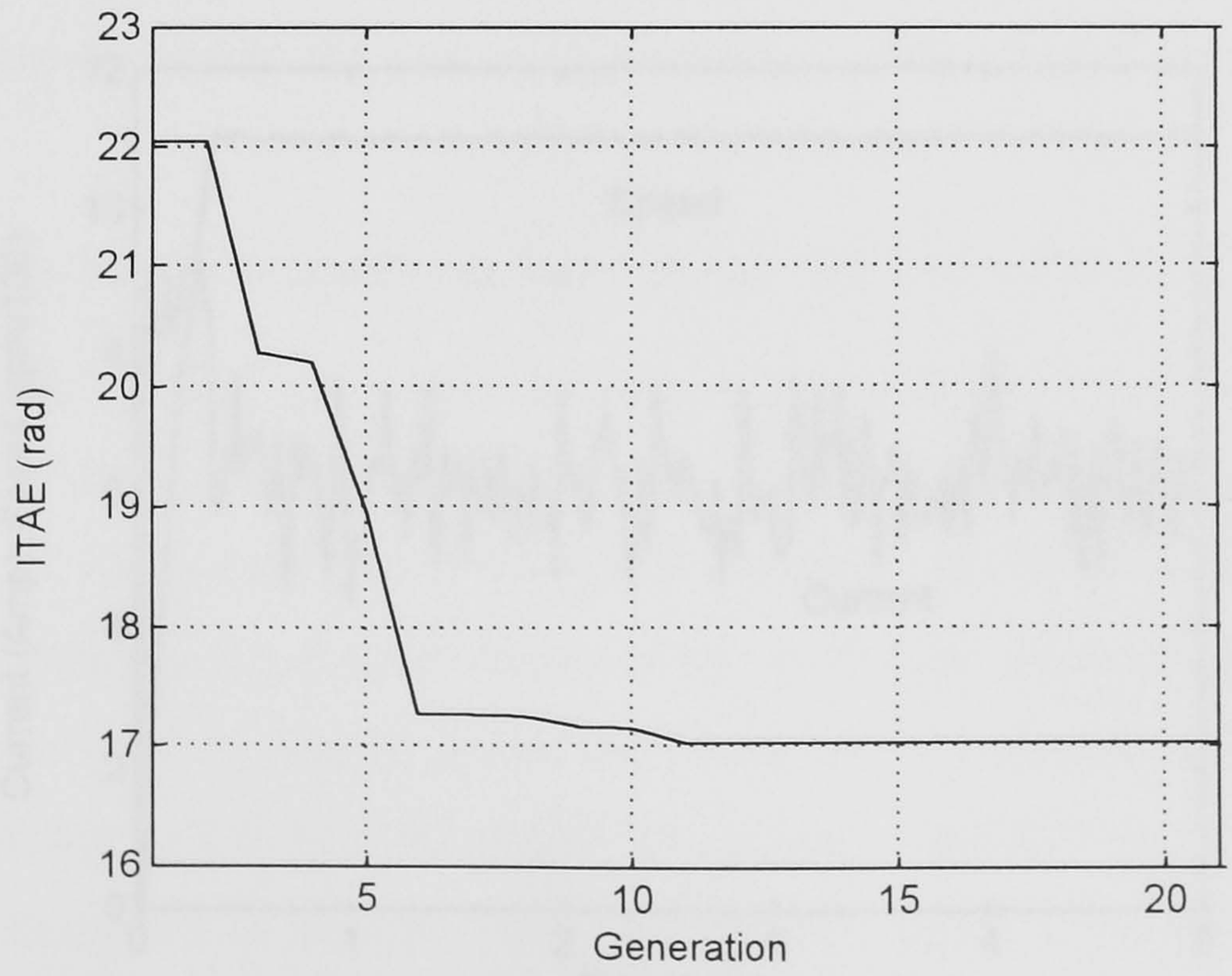
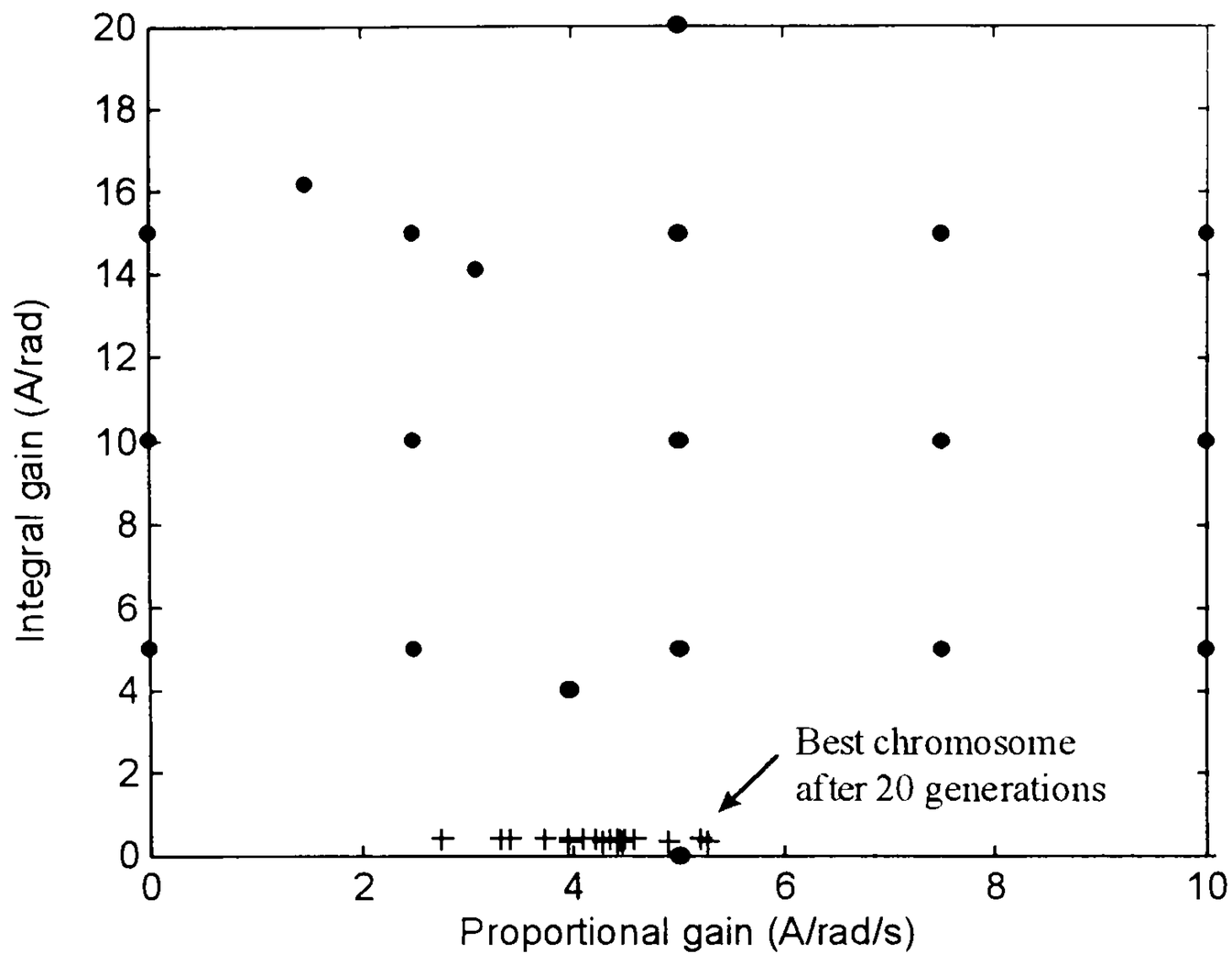



Fig. 5.4 - ITAE for the best chromosome of each generation as a function of generation number.




"•" – initial population; "+" – final population.

Fig. 5.5 - Initial and final population of chromosomes, showing the convergence towards the best individual.

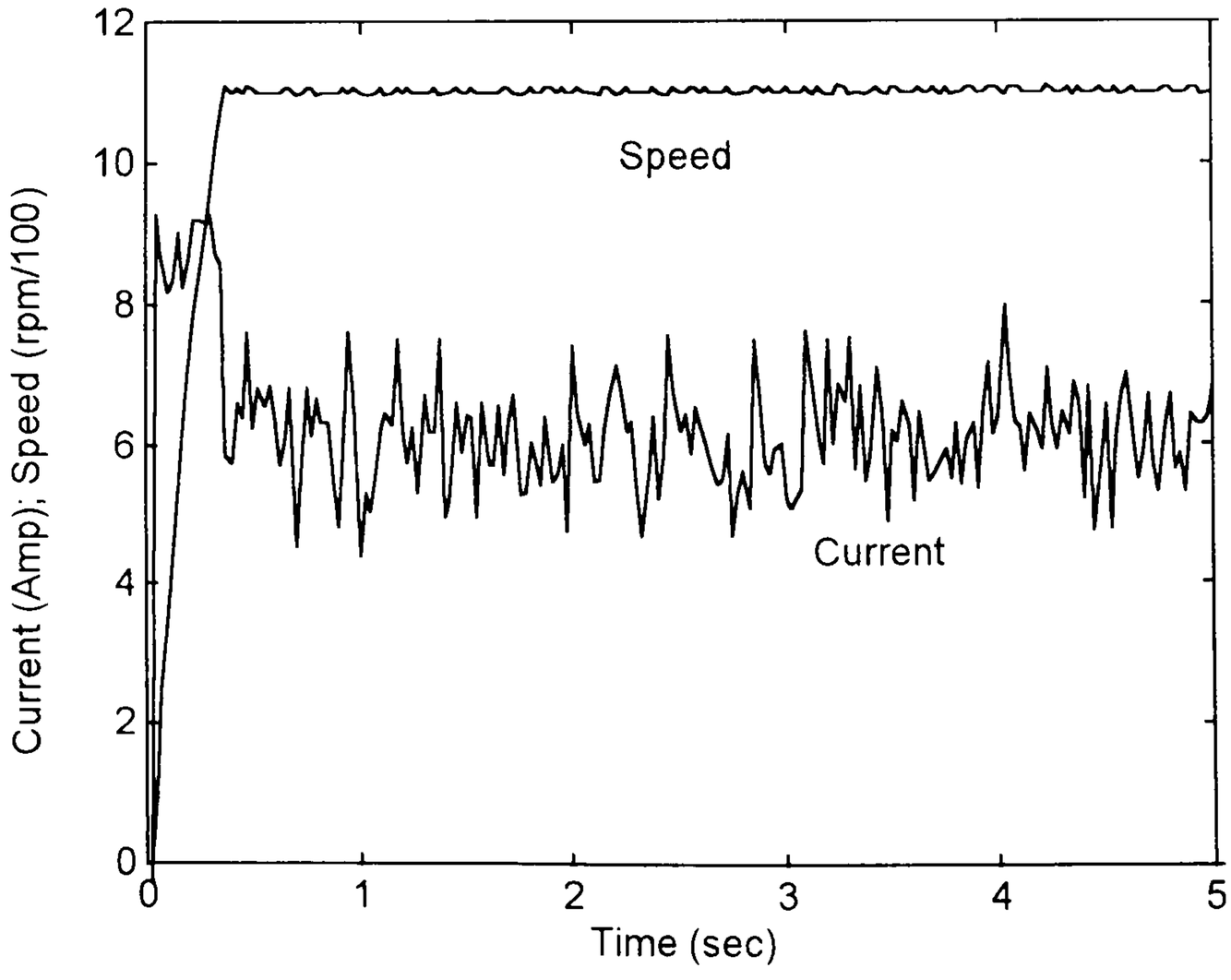



Fig. 5.6 - Best speed and current response derived by the genetic algorithm for a large step in speed demand.

The impact of the current / torque limit non-linearity on tuning becomes clear if the controller settings derived above are used for a smaller step of speed demand (1000-1100 rpm). The speed response is shown in Fig. 5.7, where the current limit is active for only 0.05 s. The ITAE for this speed response is 0.60 rad, a figure which may be compared with the results of the following section where the controller is optimised for this smaller step input.

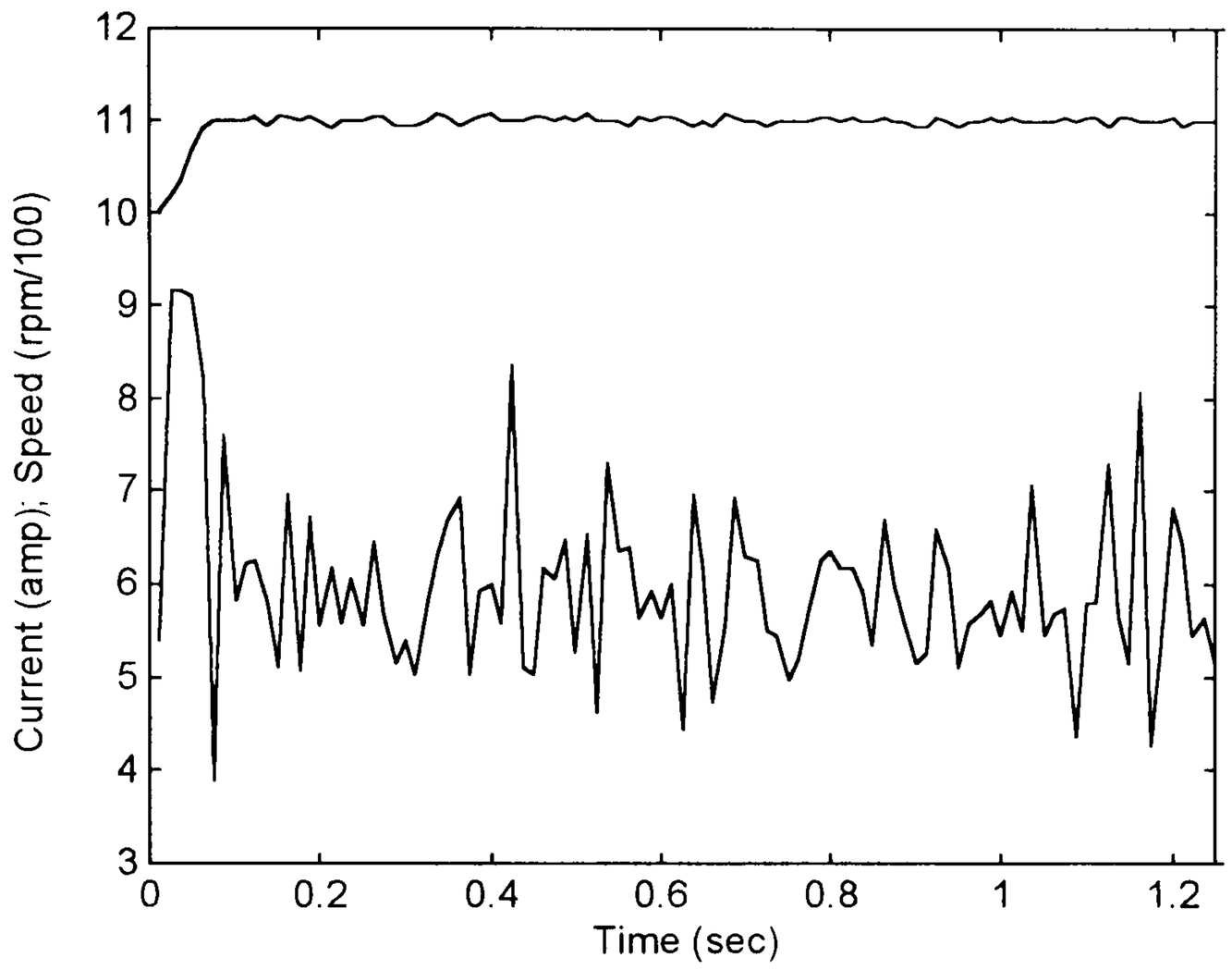



Fig. 5.7 - Speed and current response due to a small step in speed demand, with the controller tuning derived by the genetic algorithm for a large step in speed demand.

### 5.4.2 - Optimisation for a small step input speed demand (1000 to 1100 rpm)

The non-linearity caused by torque saturation has a major effect on the drive's response to large step changes in speed demand: in Fig. 5.6, for example, the current limit is active for the greater part of the speed transient. To illustrate the consequent effect of step change magnitude on optimum controller parameters, the genetic algorithm is now used on line to derive the tuning of the controller for a small step input speed change (1000 to 1100 rpm). The initial population was created in the same way as described in section 5.4.1. In Fig. 5.8, some of the speed responses found with the chromosomes in the initial population are presented. The evolution of the optimisation process is shown in Figs. 5.9 and 5.10. Fig. 5.11 represents the best

speed and current response of the Brushless DC Motor Drive, after the 20th generation of optimisation process.

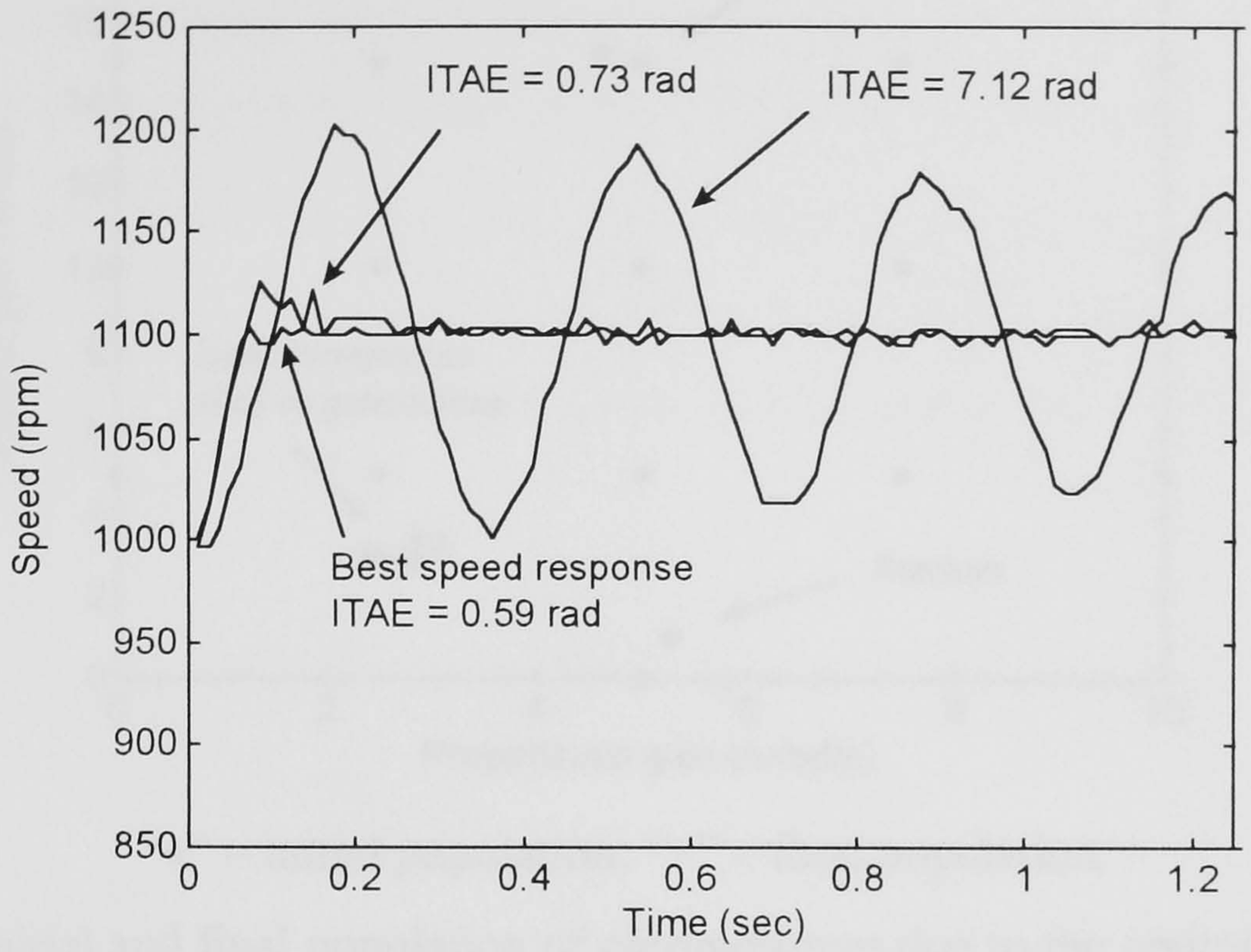



Fig. 5.8 - Speed responses following a small step input obtained with some chromosomes of the initial population and also the best of the first population.

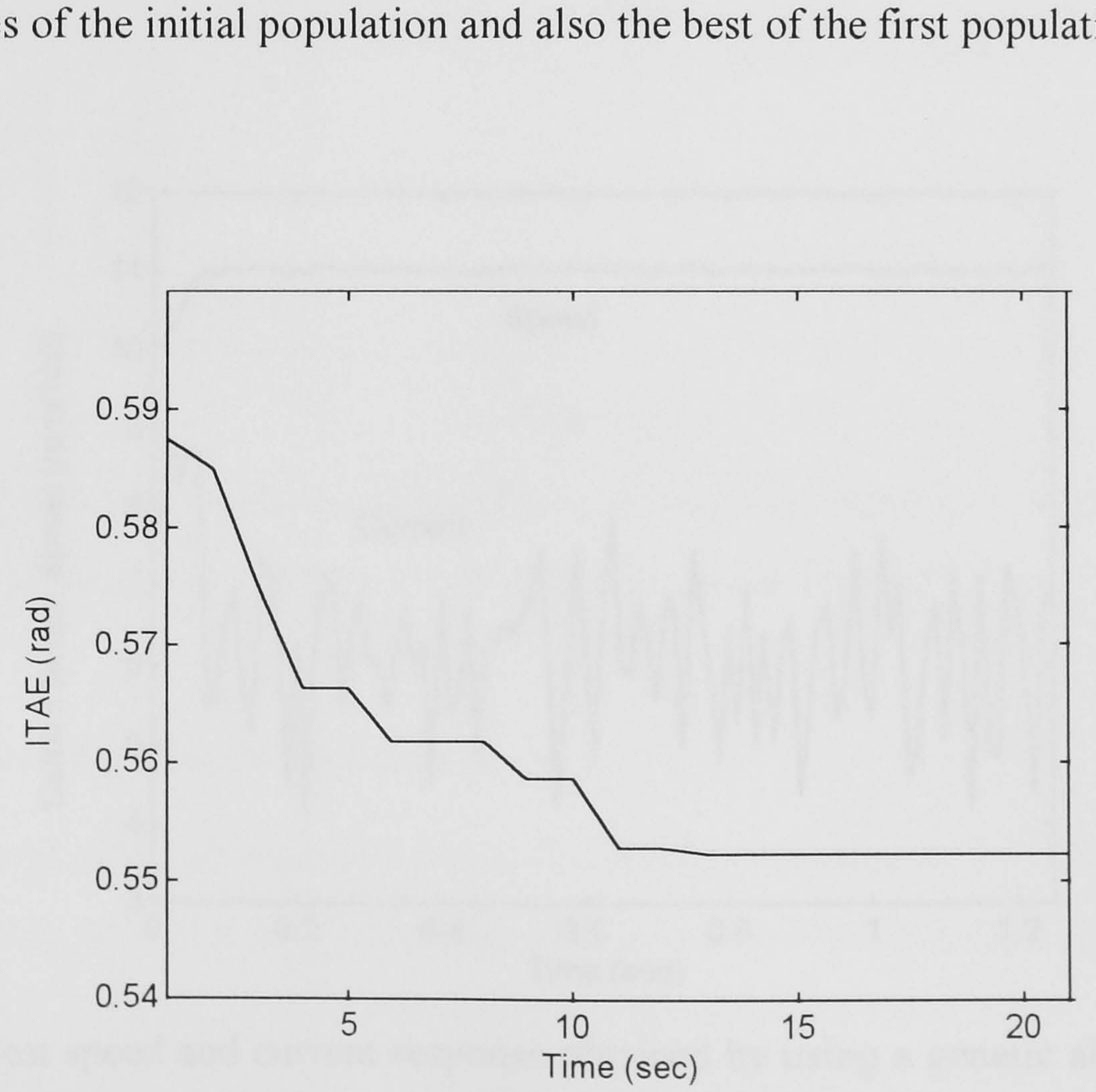
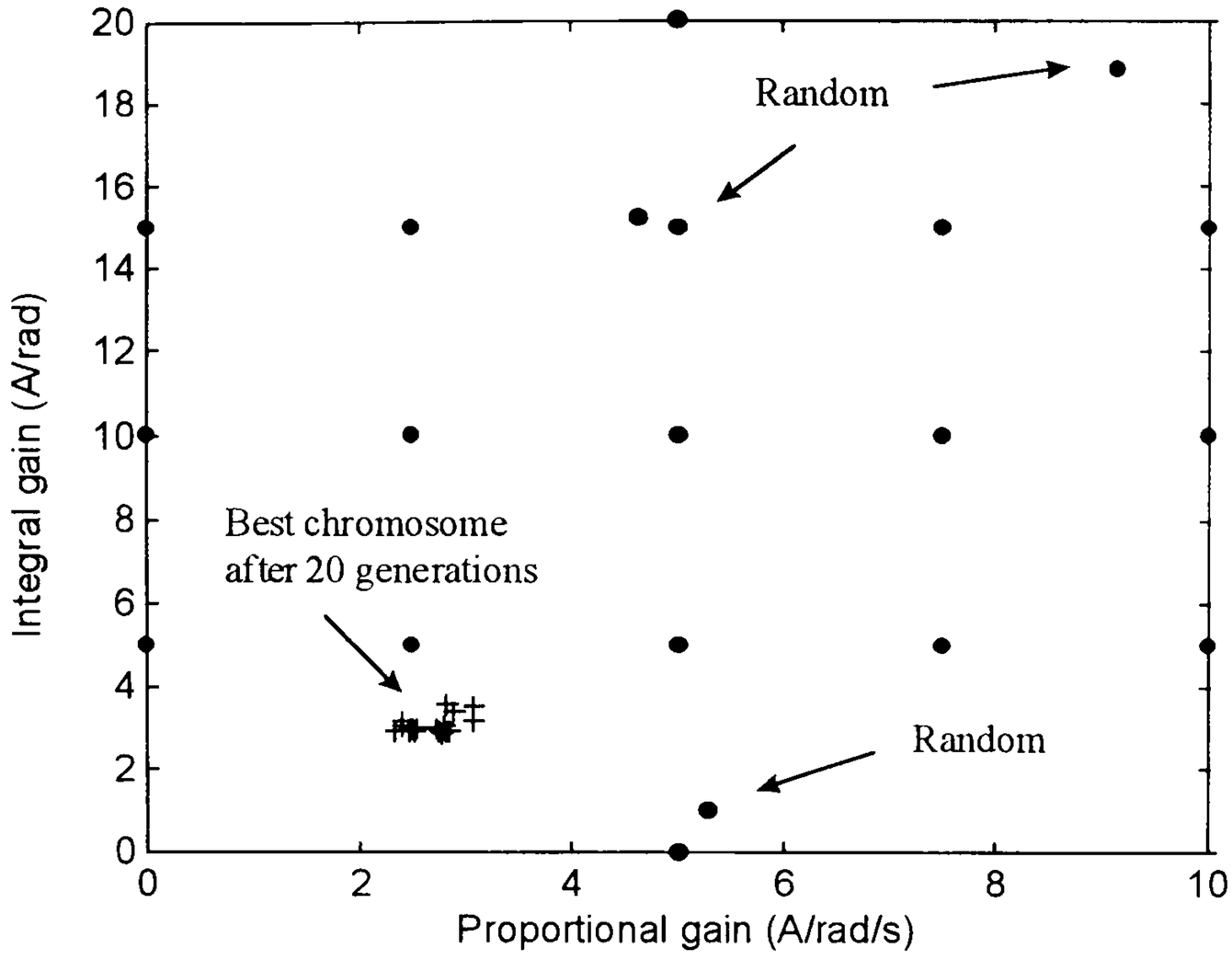



Fig. 5.9 - Genetic algorithm evolution of the on-line tuning of the PI speed controller for a small step input.




"•" – initial population; "+" – final population.

Fig. 5.10 - Initial and final population of chromosomes due to the optimisation of the PI speed controller for a small step input (1000 – 1100 rpm).

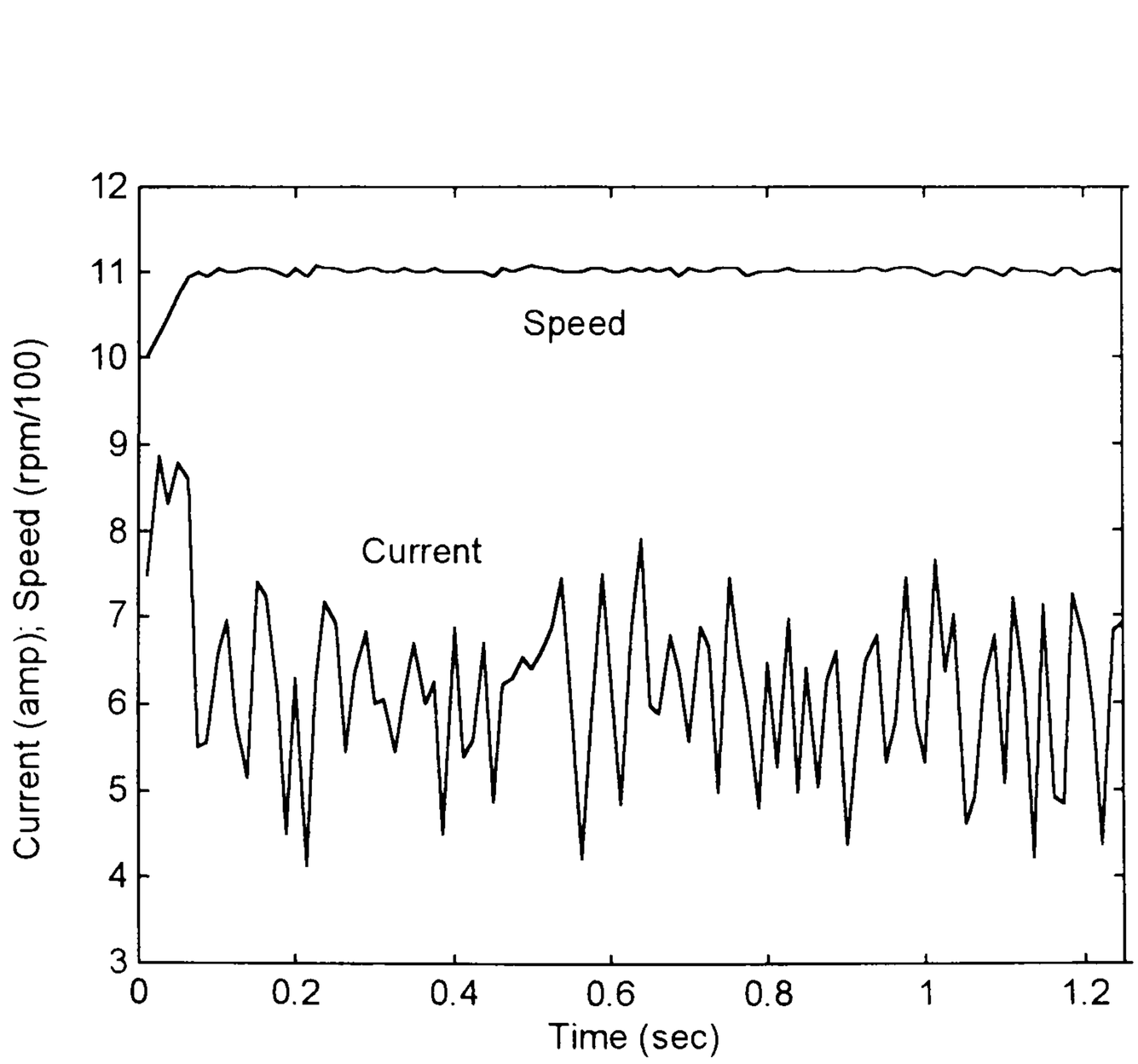



Fig. 5.11 - Best speed and current response obtained by using a genetic algorithm on-line for the optimisation of the PI speed controller due to a small step input (1000 – 1100 rpm).

The best speed response derived by the genetic algorithm has an ITAE of 0.55 rad with the PI speed controller having a proportional gain of 2.7 A/rad/s and integral gain of 2.9 A/rads$^2$. In comparison to the controller settings for large step input, the integral gain is increased substantially, because integrator *windup* has a smaller impact for the small step input and consequent lower values of speed error.

If the tuning above is kept the same for a large step input speed demand, the performance of the drive deteriorates, as shown in Fig. 5.12. The torque saturation and the large integrator gain causes integrator *windup*, and speed overshoot, resulting in an ITAE of 30.4 rad.

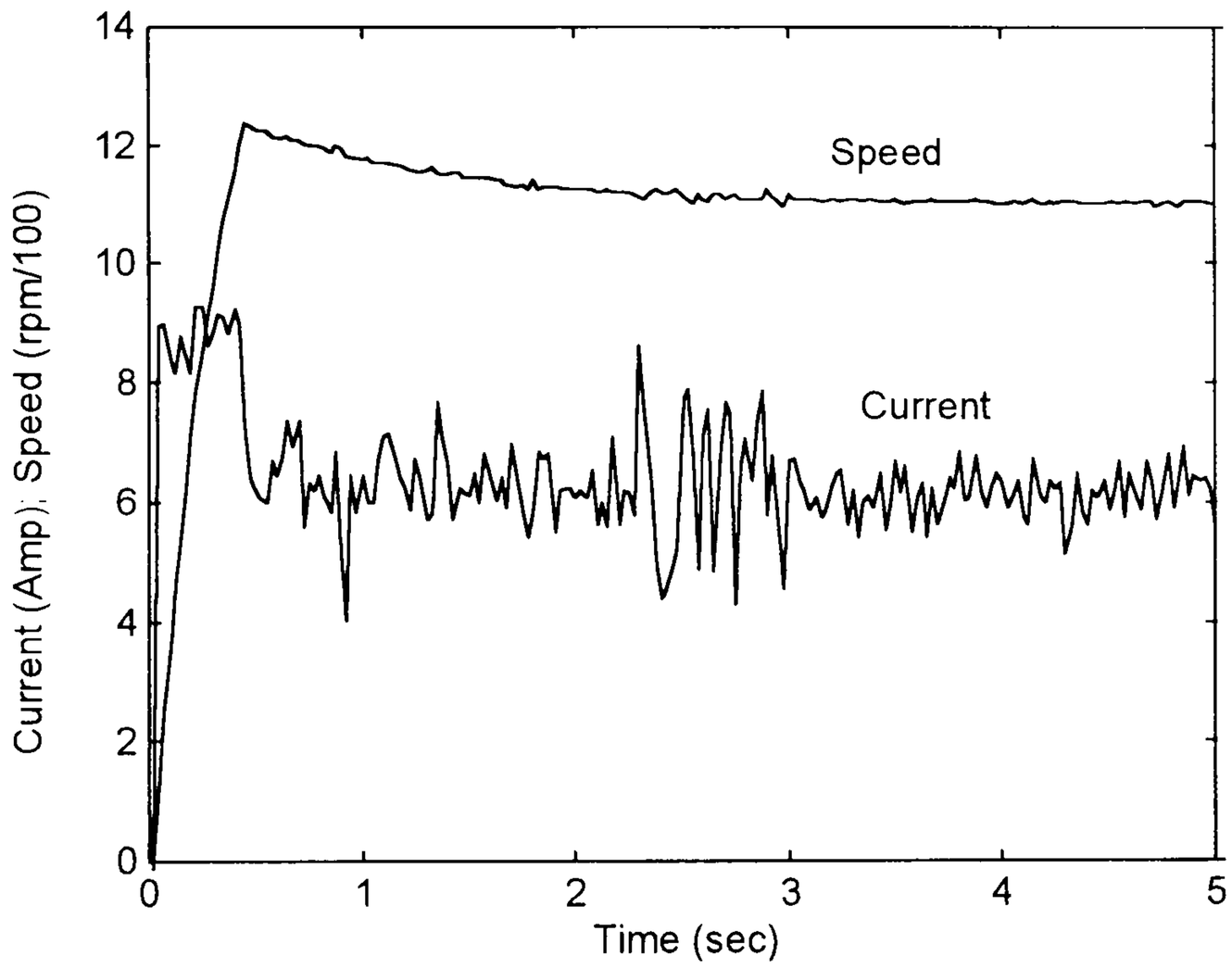
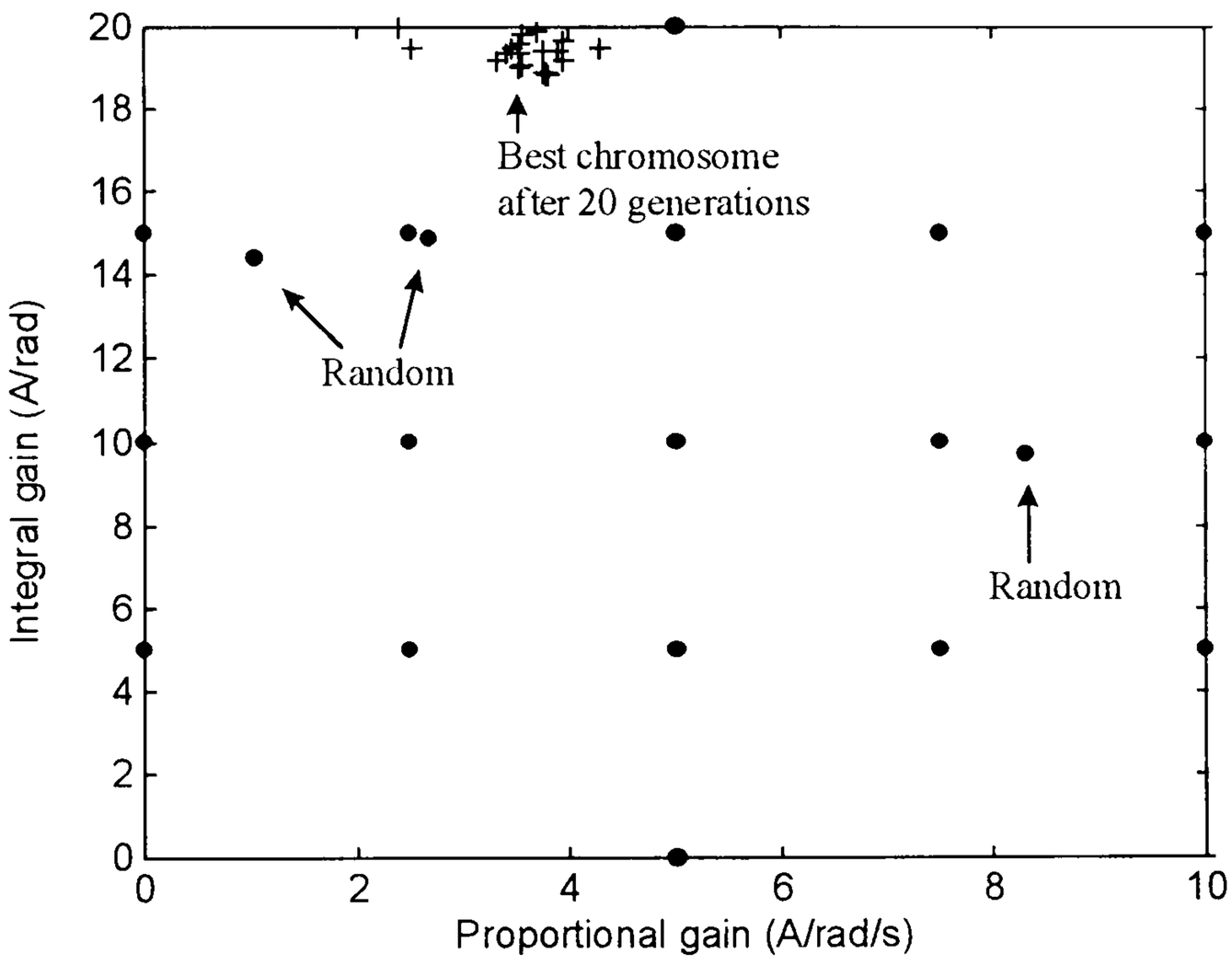



Fig. 5.12 - Speed and current response due to a large step input (0 – 1100 rpm), with the PI speed controller tuned for a small step input speed demand.

### 5.4.3 - Optimisation for load disturbance

There are applications of electrical motor drives where change in the reference speed is not important, such as the film making process as discussed in chapter 1 and also by Ho *et al* [HO, S. et al, 1994]. Then, in order to illustrate once more the Genetic Algorithm's potential, it has also been used for optimising the controller to keep the speed in the presence load torque disturbance. The optimisation process was done

within the following condition: the motor was initially running at 1100 rpm constant speed with 11% of the maximum load. At *time* = 1s, 62% of the maximum load torque step input load disturbance was applied, lasting for 3 s. The performance of the controller was assessed during the disturbance only. The Genetic Algorithm was set up with identical conditions as used previously in this chapter. The evolution of the optimisation process is shown in Figs. 5.13 and 5.14. As can be seen in Fig. 5.13, the chromosomes have migrated towards even higher gains as there is no impact of the integrator *windup*. The best chromosome has a proportional and integral gains equal to 3.42 A/rad/s and 19.4 A/rad, respectively.




"•" – initial population; "+" – final population.

Fig. 5.13 - Initial and final population of chromosomes due to the optimisation of the PI speed controller in the presence of load disturbance.

The best speed response obtained within the first population has an ITAE of 0.87 rad. However, at the end of the optimisation process, a better one was found, which produced an ITAE was equal to 0.81 rad.

Fig. 5.15 shows the best speed and current response after the optimization process whereas Fig. 5.16, in enlarged scale, demonstrates the speed variation due to the disturbance. The ITAE determined during the disturbance is 0.81 rad.

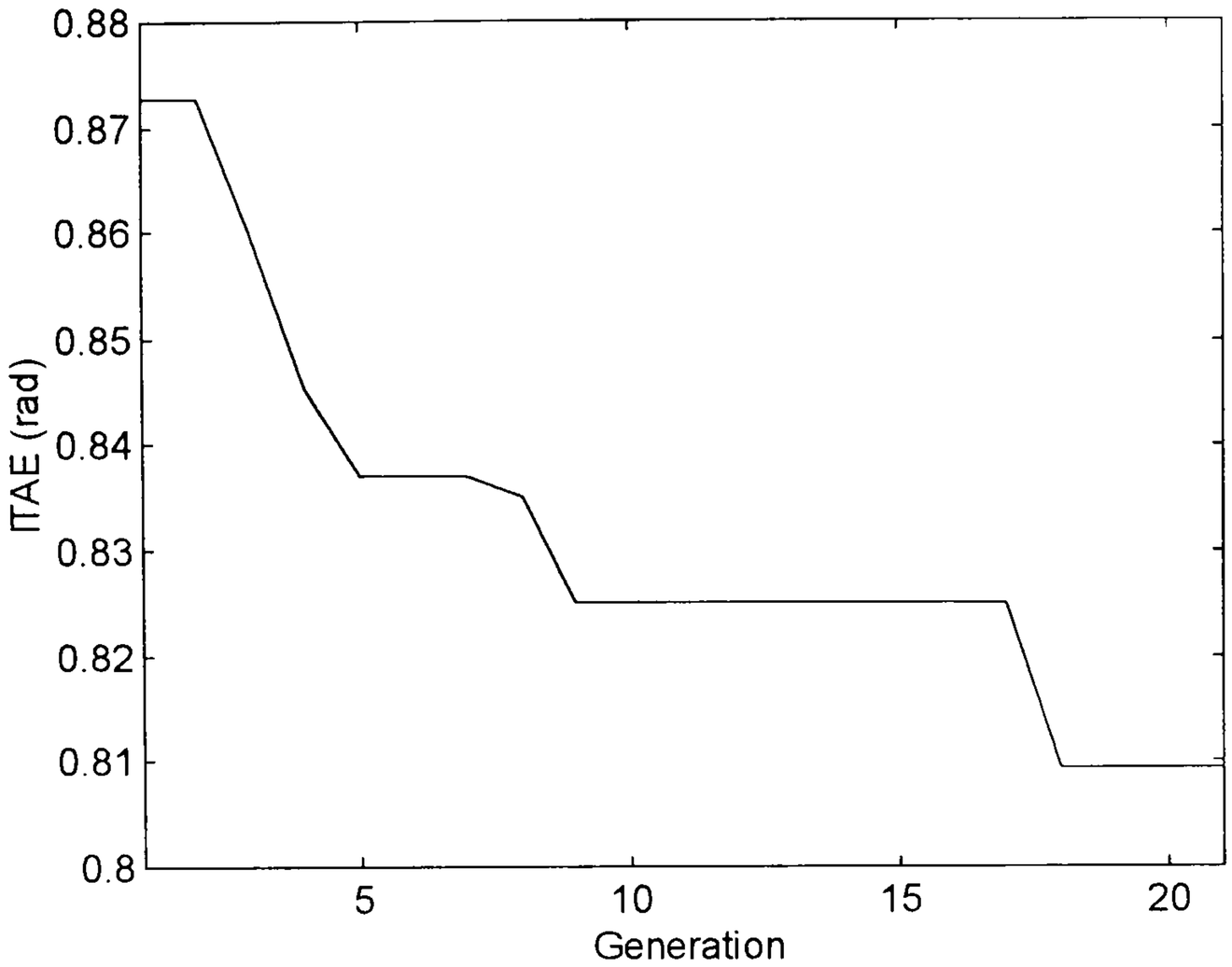



Fig. 5.14 - Genetic algorithm evolution of the on-line tuning of the PI speed controller in the presence of load disturbance.

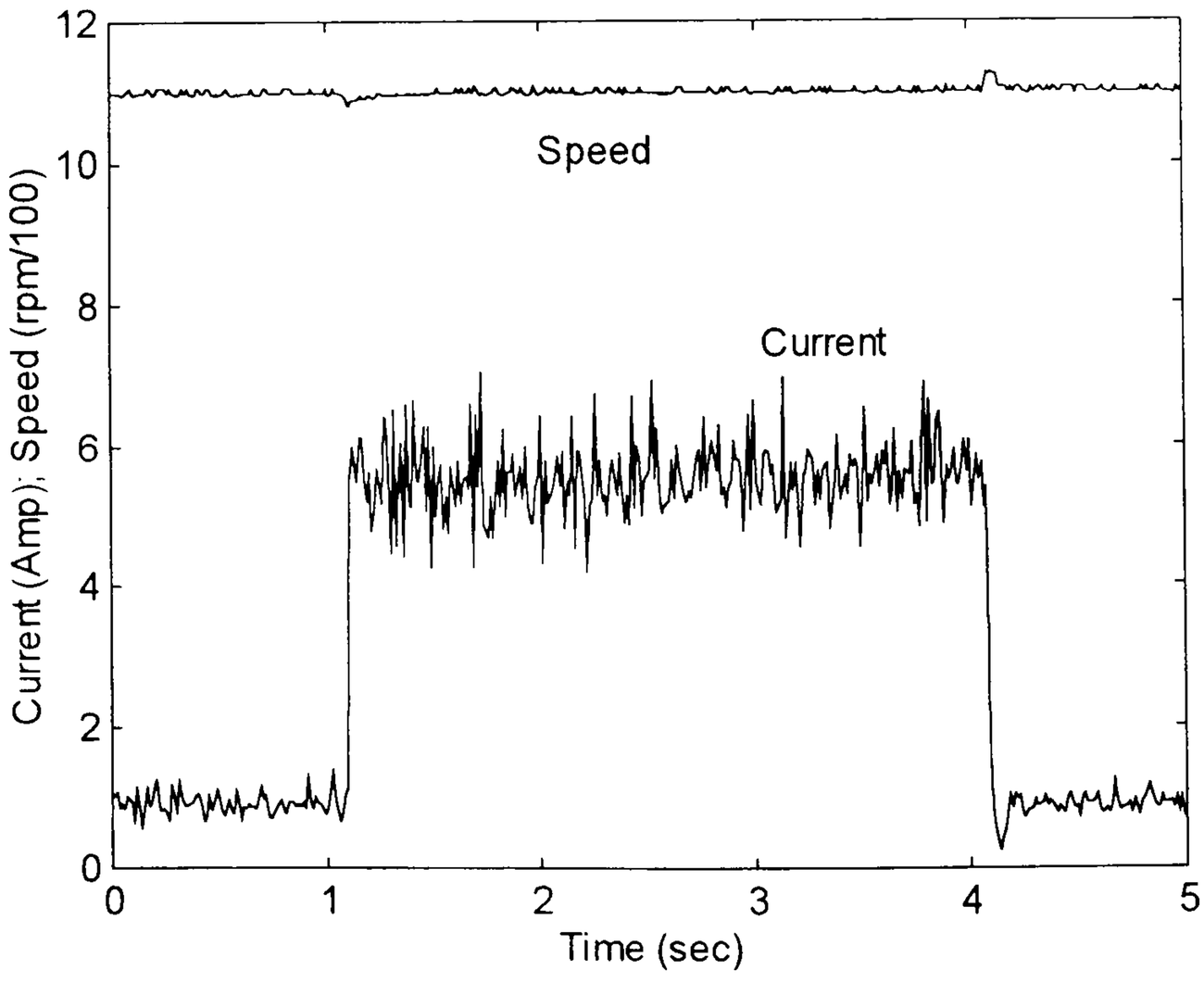



Fig. 5.15 - Best speed and current response obtained by using a genetic algorithm on-line for the optimisation of the PI speed controller for load torque disturbance.

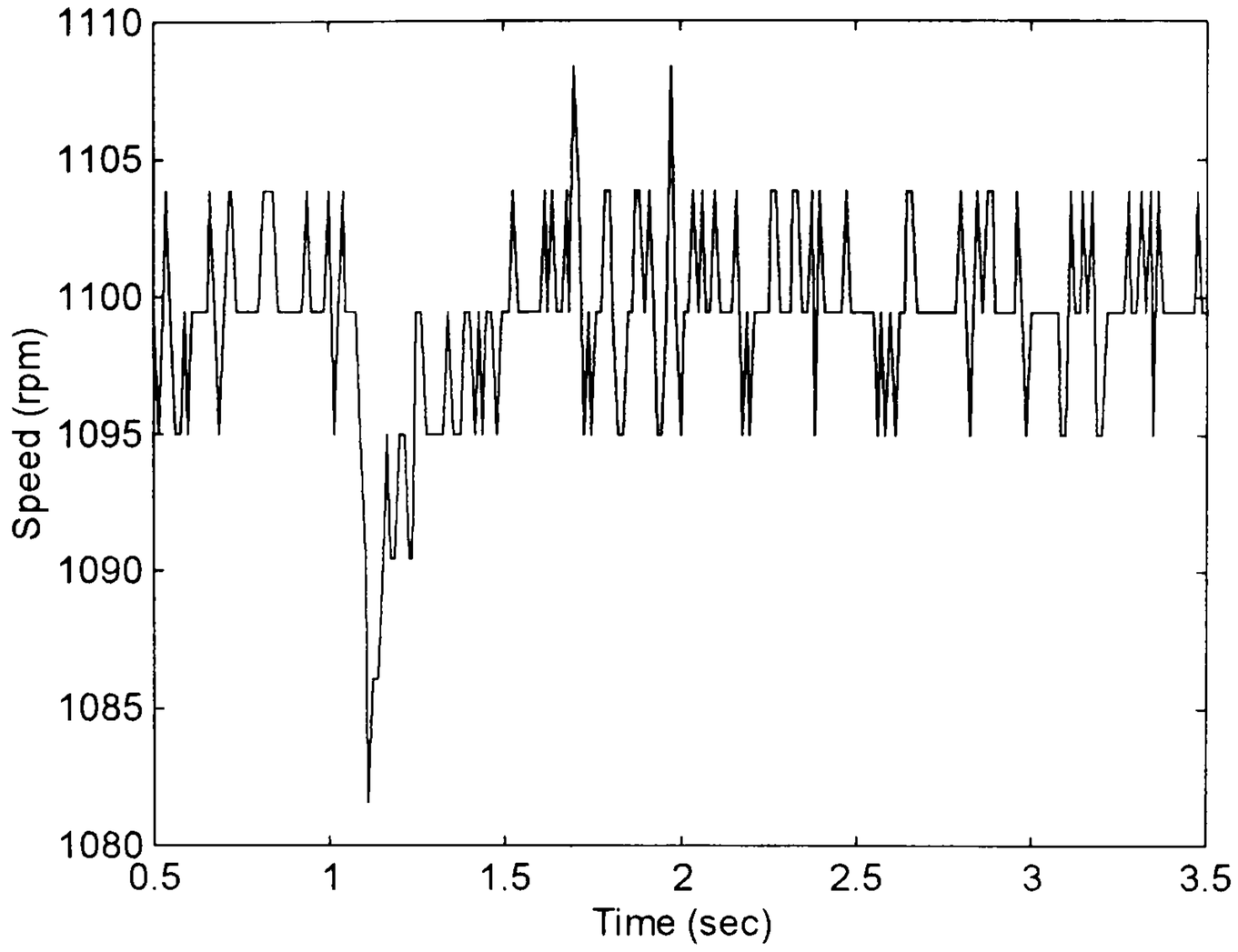



Fig. 5.16 - Speed response, in enlarged scale, highlighting the speed variation due to the load torque disturbance.

### 5.4.4 - Optimisation for variable speed and load

In this particular test, the optimisation took place within a condition where the speed demand changed after the starting transient as well as the load. The motor drive was initially at rest when a 1000 rpm step input speed demand was applied, with 11% load torque. 1.05 s later (*time* = 1.05 s), a small step input speed demand was applied, making the motor speed demand jump from 1000 rpm to 1100 rpm. Another 1.05 s elapsed (*time* = 2.05 s) and the load was suddenly changed from 11% to 62% of the maximum torque produced by the motor. A Genetic Algorithm with identical setting as used in the previous section, in terms of population size, generation number and mutation probability, was used to find the optimal tuning for the PI speed controller within a 5 s time window. Fig. 5.17 illustrates some speed responses given by some chromosomes of the first population, together with the best speed response obtained by the Genetic Algorithm, at the end of the optimisation process. Fig. 5.18 depicts the evolution of the chromosomes with respect to the best ITAE obtained within each generation. Fig.5.19 shows the initial and final population of chromosomes. In Fig. 5.20 is presented the best speed and current response obtained after 20 generations.

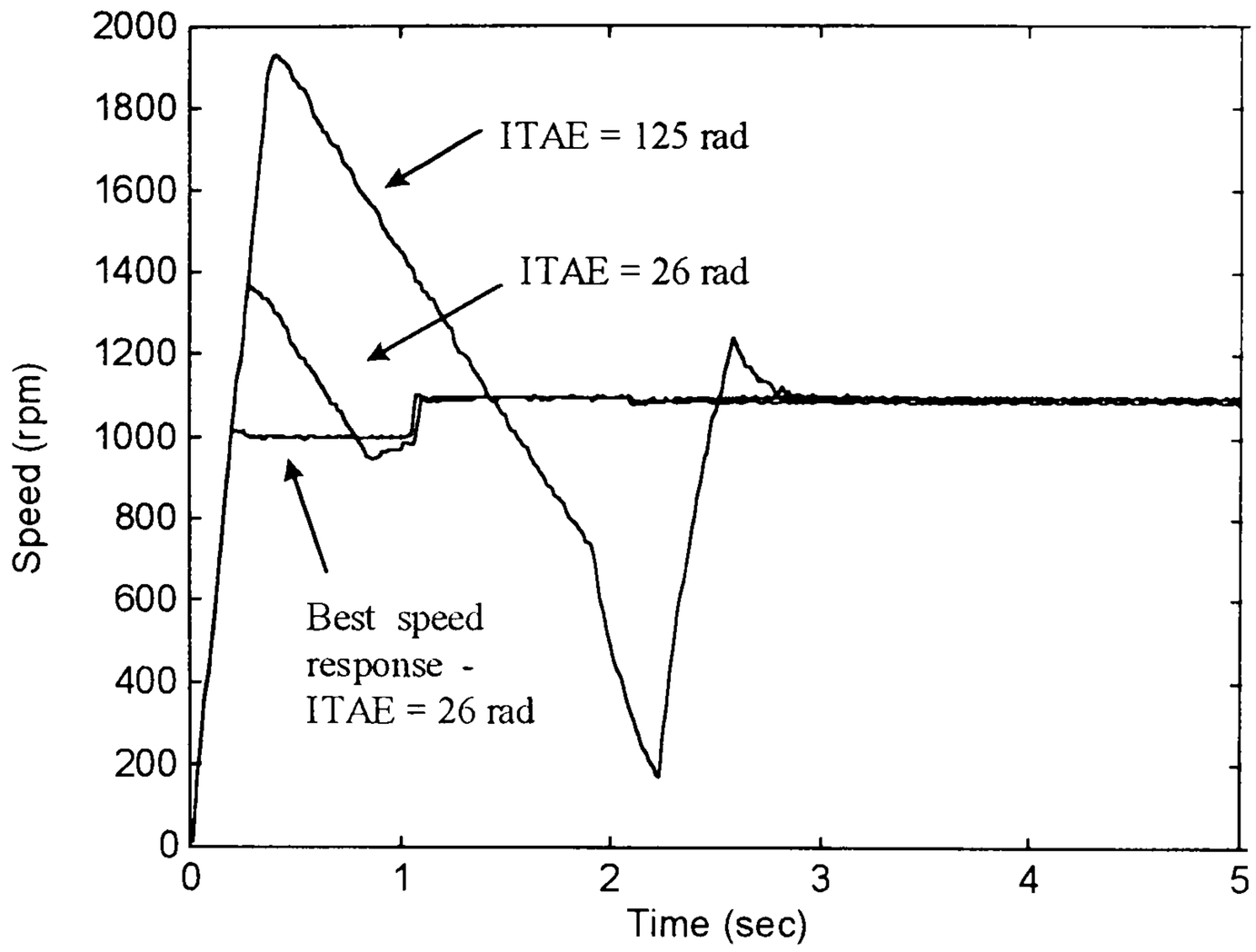



Fig. 5.17 - Speed responses following a large and small step input speed demand as well as load variation, obtained with some chromosomes of the initial population and also the best after the optimisation process.

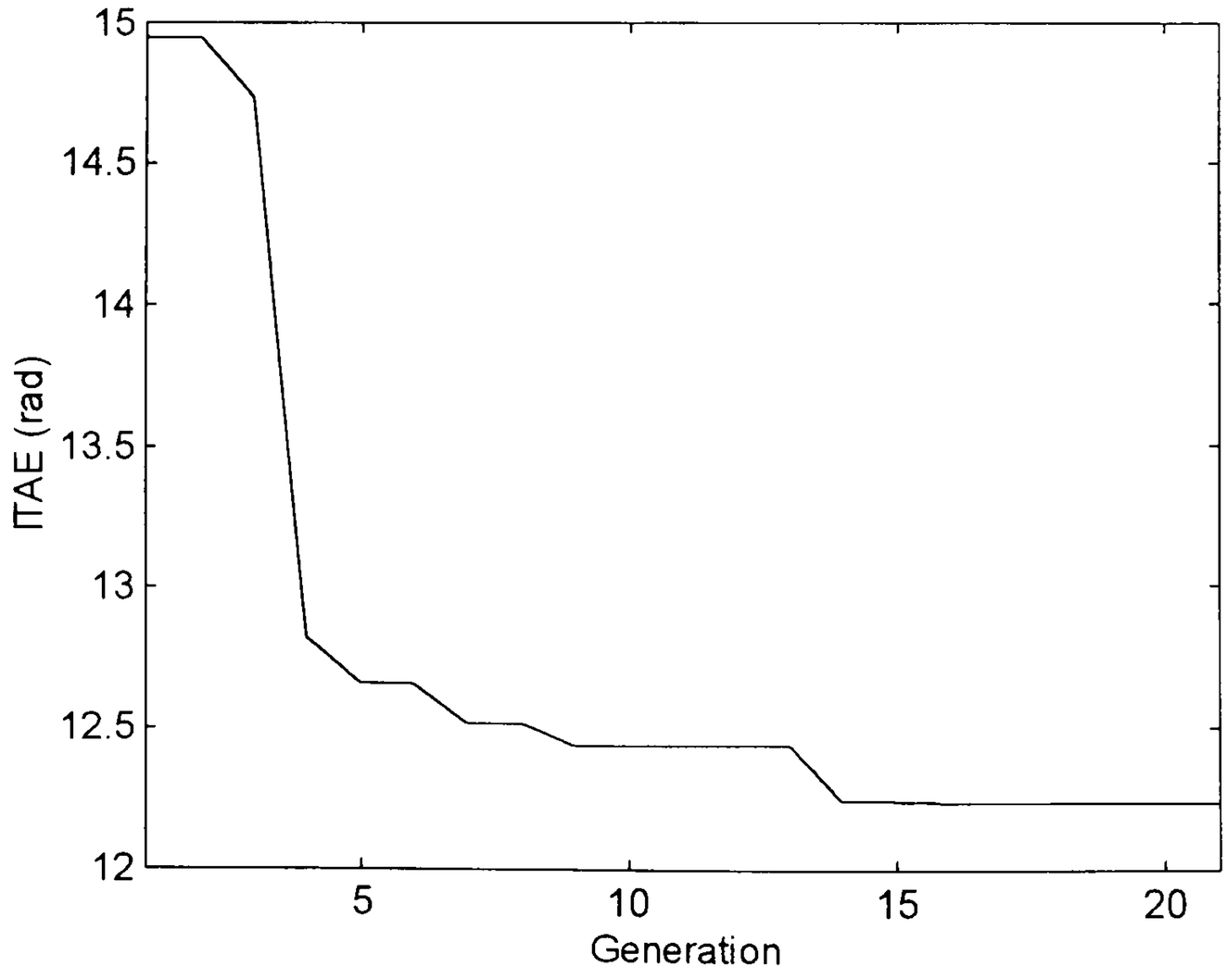
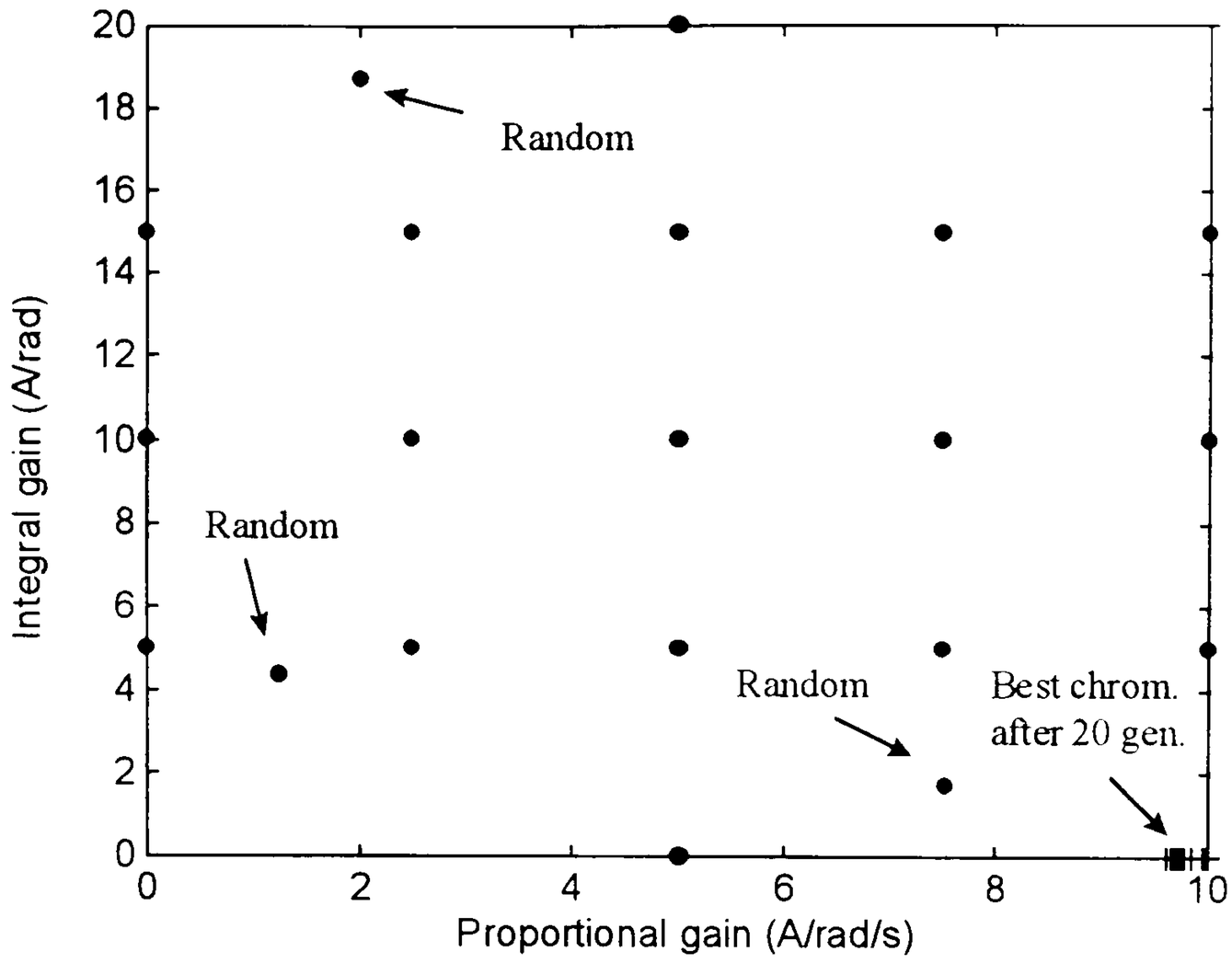



Fig. 5.18 - Genetic Algorithm evolution for the optimisation of the speed of the motor drive for speed demand and load changing.




"•" – initial population; "+" – final population.

Fig. 5.19 - Initial and final population of chromosomes for the optimisation of the PI speed controller for speed demand and load changing.

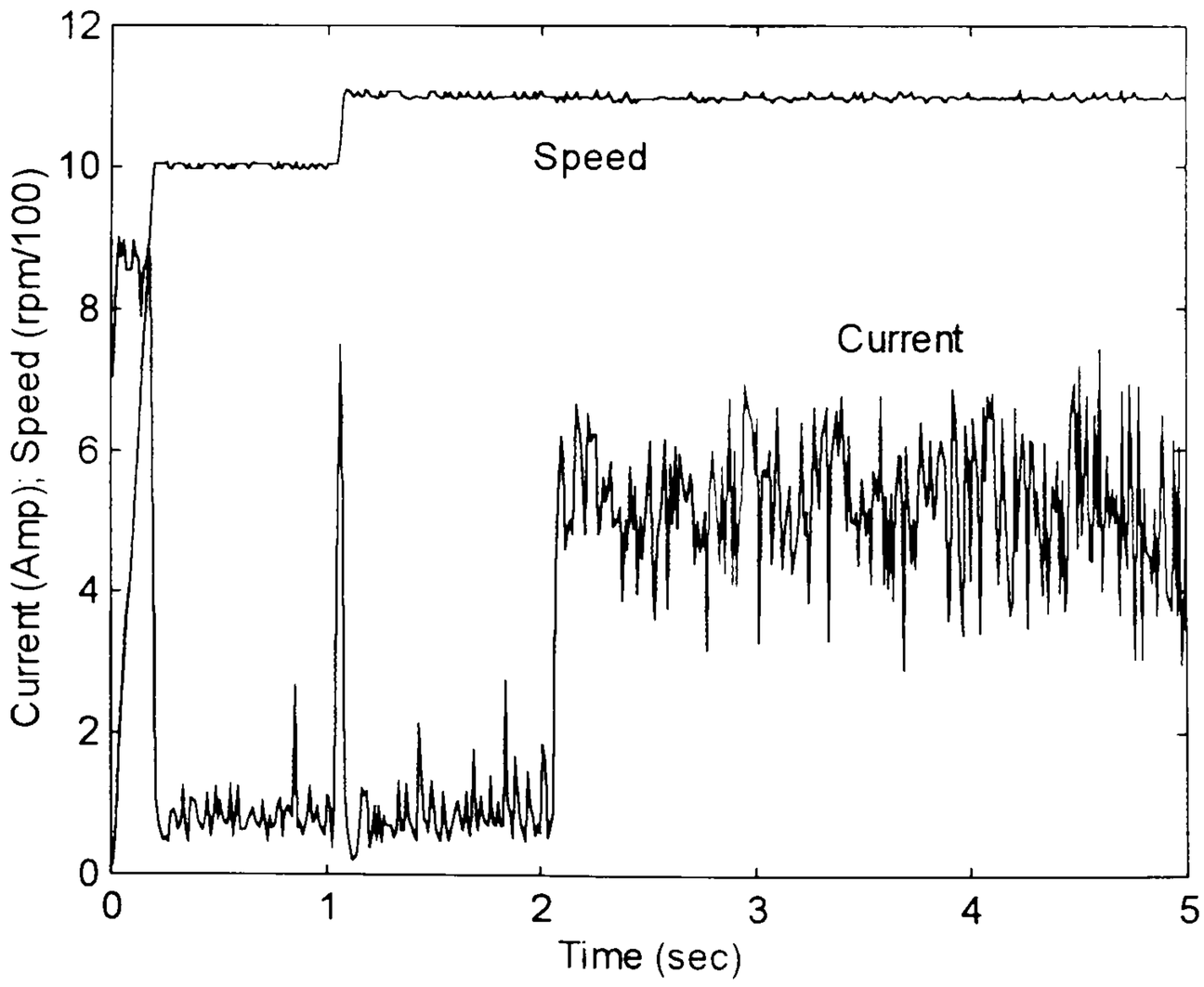



Fig. 5.20 - Best speed and current response after the optimisation of the PI speed controller for speed demand and load changing.

At the end of the optimisation process, the best setting for the PI speed controller was 9.6 A/rad/s for the proportional gain and 0 A/rad, for the integral one. It means that the best controller was not a PI but a simple proportional one with very high gain. The reason is because the ITAE of the speed response in this particular condition is dominated by the acceleration of the motor due to a step change in the speed demand, rather then in the load torque, within the 5 s time window. Due to the large proportional gain, the rate of switching of the electronic switches of the driver is higher then those shown in sections 5.4.1-2. As a consequence, the speed and current signals are noisier.

Zero steady state speed error cannot be obtained. However, it is very small for the ITAE within the 5 s time window. Certainly, a for larger time window, it would be otherwise.

### 5.4.5 - Tuning of the PI speed controller by using classical control theory

The tuning of the speed controller manually and directly on the motor drive has been done before in chapter 3, for a large step input speed demand (1100 rpm), in the presence of 62% of load torque. In order to avoid torque saturation and integrator *windup*, the proportional gain was adjusted so that the maximum current demand was hit for the speed demand. The integral gain was set up to give a slightly underdamped speed response.

In this section, another example is shown where the speed controller was tuned by using the classical control theory, for a small step input speed response (1000 rpm to 1100 rpm), in the presence of the same load torque as used previously. The aim of doing this is to show the difference between the off-line tuning using the classical control theory and the on-line one, given by the Genetic Algorithm. In order to assure conformity of the results, tuning was carried out for identical load conditions and speed demand as that discussed previously in section 5.4.2.

A schematic representation of the speed controller and motor / load dynamics is shown in Fig. 5.21, where the current controller is assumed to be fast-acting relative to the speed controller.

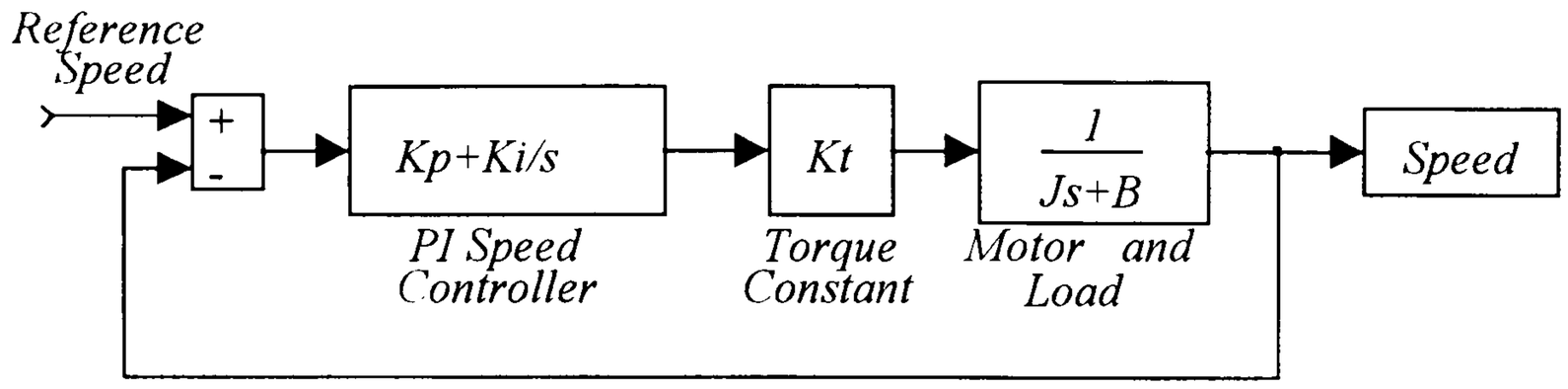



Fig. 5.21 - Simplified block diagram of the speed controlled brushless dc drive.

The transfer function equation of the closed loop system is:

$$\frac{W(s)}{W_{ref}(s)} = \frac{K_t.(K_p s + K_i)}{J.s^2 + (B + K_t K_p).s + K_t K_i} \tag{5.1}$$

and the closed-loop poles are the roots of the characteristic equation:

$$J.s^2 + (B + K_t K_p).s + K_t K_i = 0 \tag{5.2}$$

Responses obtained using on-line genetic algorithm tuning exhibit little or no overshoot, so it is appropriate to aim for critical damping, with equal roots for the characteristic equation. From Equation 2, the condition for critical damping is that:

$$(B + K_t K_p)^2 = 4JK_t K_i \tag{5.3}$$

However, there are many different values of $K_i$ and $K_p$ that satisfy the condition. It is known that a higher proportional gain leads to a faster response, but increases the effect of the current limitation. Nonetheless, for small step speed changes, a larger gain values can be used without reaching the current limit.

In this case the current is limited to 9.0 A, but the steady-state current when operating at 1100 rpm is approximately 6.0 A. Therefore a current of 3.0A is available to

accelerate the motor and load and the proportional gain is chosen so that this limit is just reached for a 100 rpm (10.5 rad/s) step change in speed demand. Hence:

$$K_p = 3.0 / 10.5 \text{ A/rad/s} = 0.28 \text{ A/rad/s}$$

Substituting this value of $K_p$ into the condition for critical damping (equation 5.3) gives a value for the integral gain:

$$K_i = 1.2 \text{ A/rad}$$

The performance of the drive with these controller settings for a small step input (1000 - 1100 rpm) is illustrated in Fig. 5.22 and in Fig. 5.23 for a large step input of 0 – 1100 rpm. It is apparent in Fig. 5.22 that, the current limit is nearly reached during the early part of the speed transient and the speed transient exhibits no overshoot, as expected for a critically damped response. The ITAE is equal to 0.81 rad. On the other hand, it is clear in Fig. 5.23 that the controller was saturated as the transient current stays at its maximum value during the acceleration time.

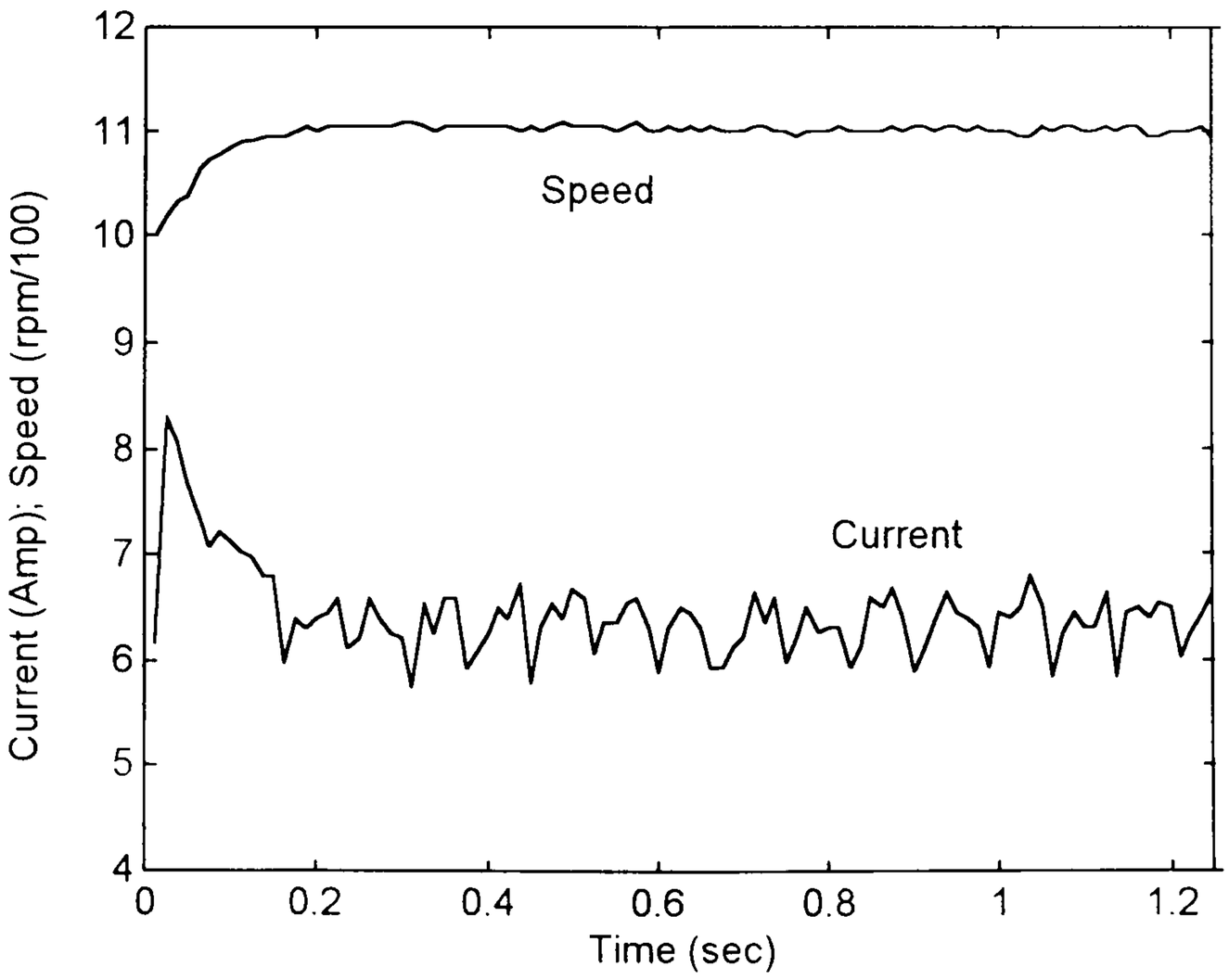



Fig. 5.22 - Speed and current for small step input (1000 – 1100 rpm) with the tuning of the PI speed controller derived by using classical control theory.

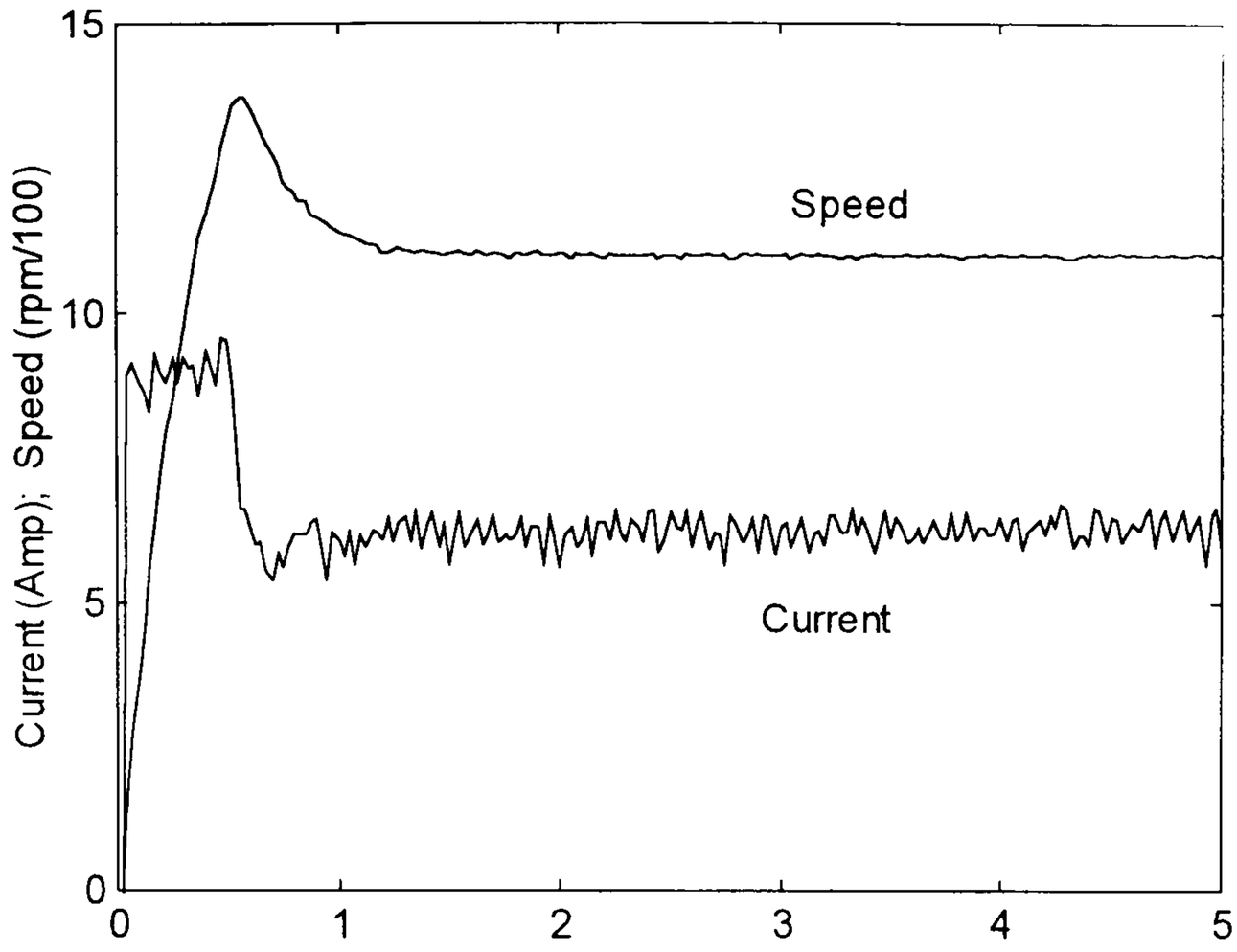
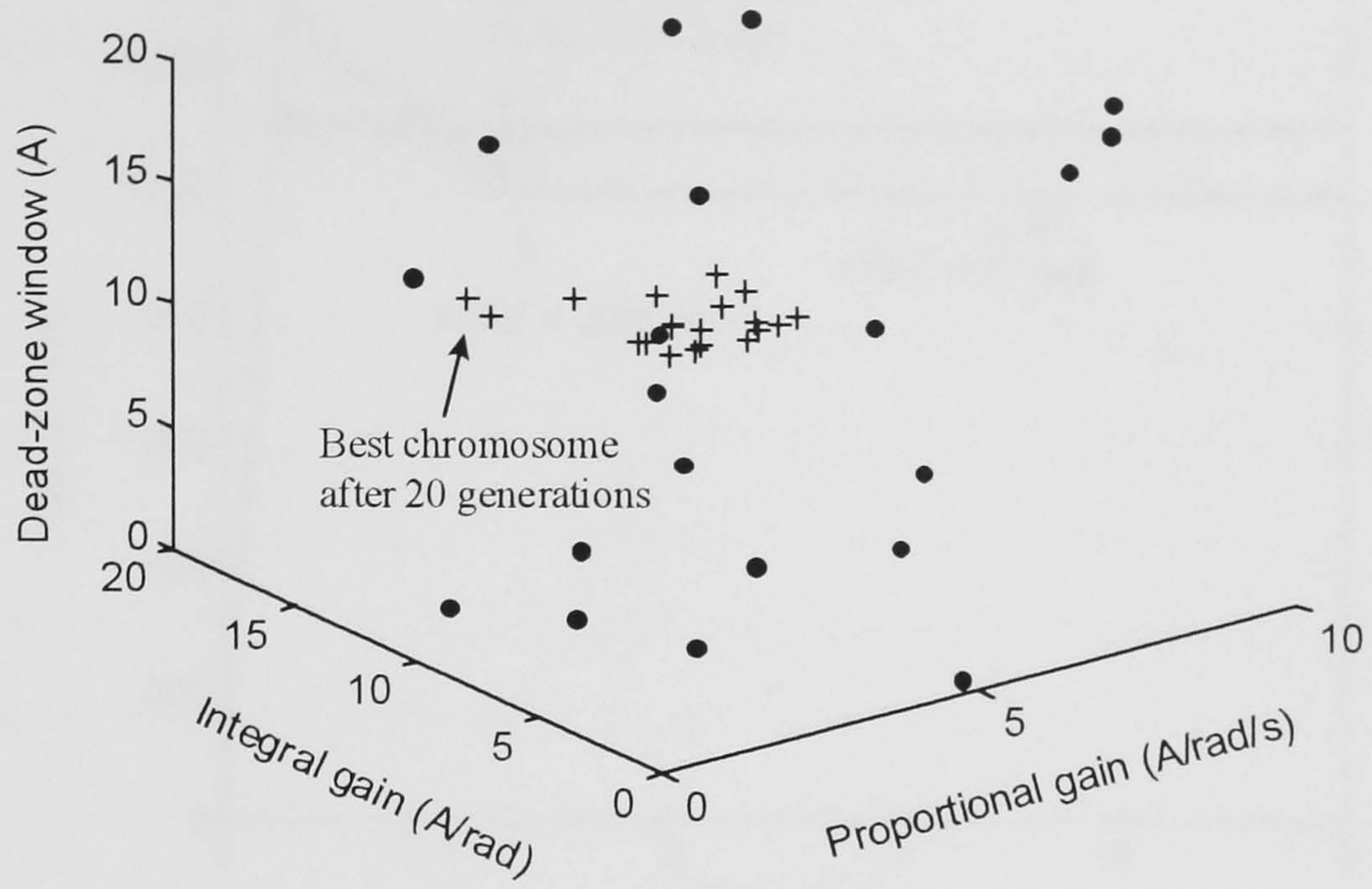



Fig. 5.23 - Speed and current for large step input (0 – 1100 rpm) with the tuning of the PI speed controller derived by using the classical control theory for small speed change (1000 to 1100 rpm).

### 5.4.5 - Optimisation for load disturbance with integrator anti-windup

In this example, a genetic algorithm is used to derive the tuning of the controller as well as the setting of the *anti-windup* circuit, for a large step input of speed demand (0-1100 rpm). The *anti-windup* circuit is that mentioned in chapter 4. However, in this particular case, the motor started from standstill with 11% load torque. At *time* = 1 s, additional step input load was applied, remaining until the end of the time window. By doing so, the optimal condition for the starting transient together with a load disturbance is found, exploiting the non-linearity of the drive system. Different from what was done so far, the initial population was created entirely at random, however, within the same range used previously. The generation number as well as the rest of the parameters of the genetic algorithm was kept the same as previously used. The results obtained are shown as follows. The 3 dimensional Fig 5.24 shows the initial and final population while Fig. 5.25 presents the evolution of the optimisation process.




"•" – initial population; "+" – final population.

Fig. 5.24 - Initial and final population of chromosomes due to the optimisation of the PI speed controller and the setting of the *anti-windup* circuit, in the presence of load disturbance.

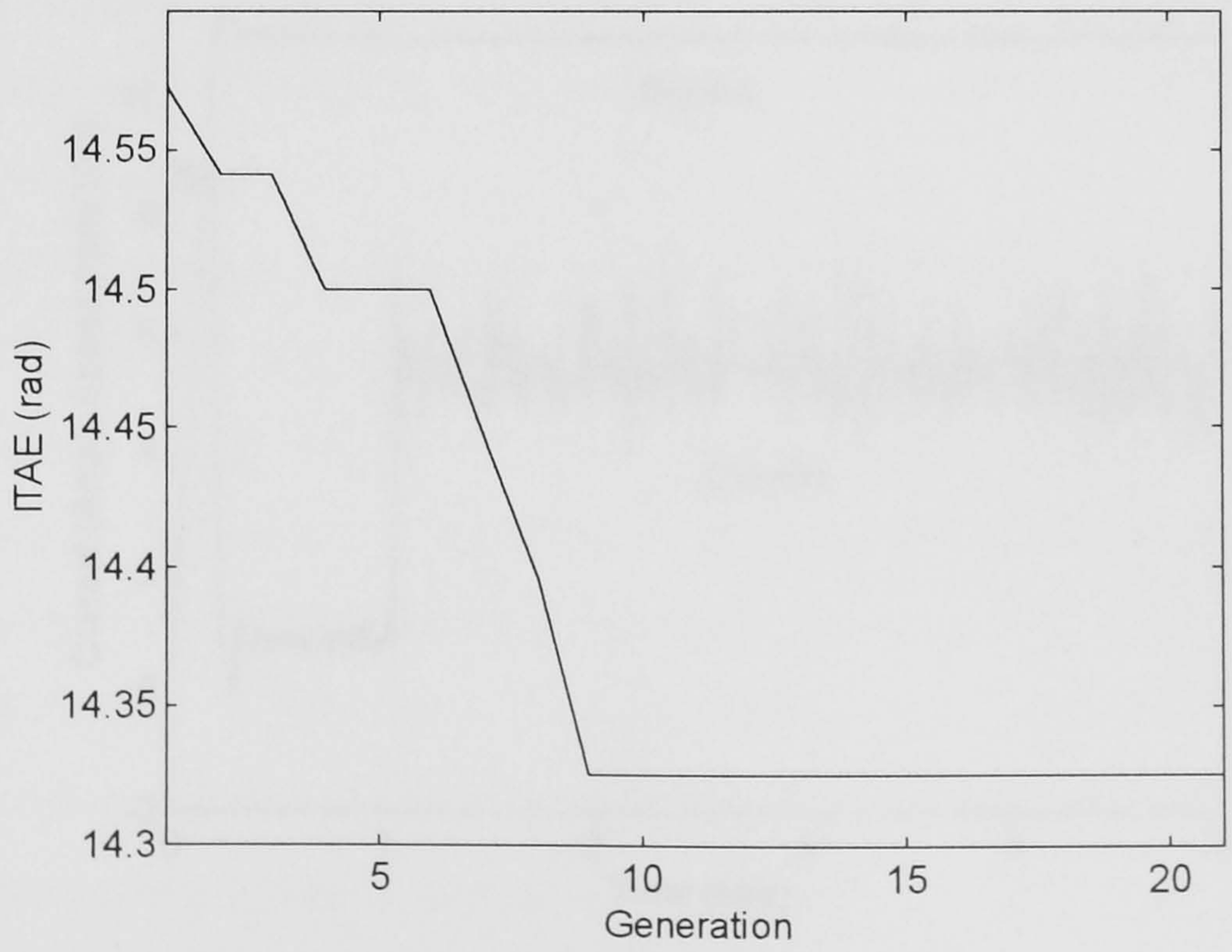



Fig. 5.25 - Genetic algorithm evolution of the on-line tuning of the PI speed controller together with the setting of a *anti-windup* circuit, in the presence of load disturbance.

Fig. 5.26 shows some of the speed response found within the first population of chromosomes and the best one obtained after the evolution process.

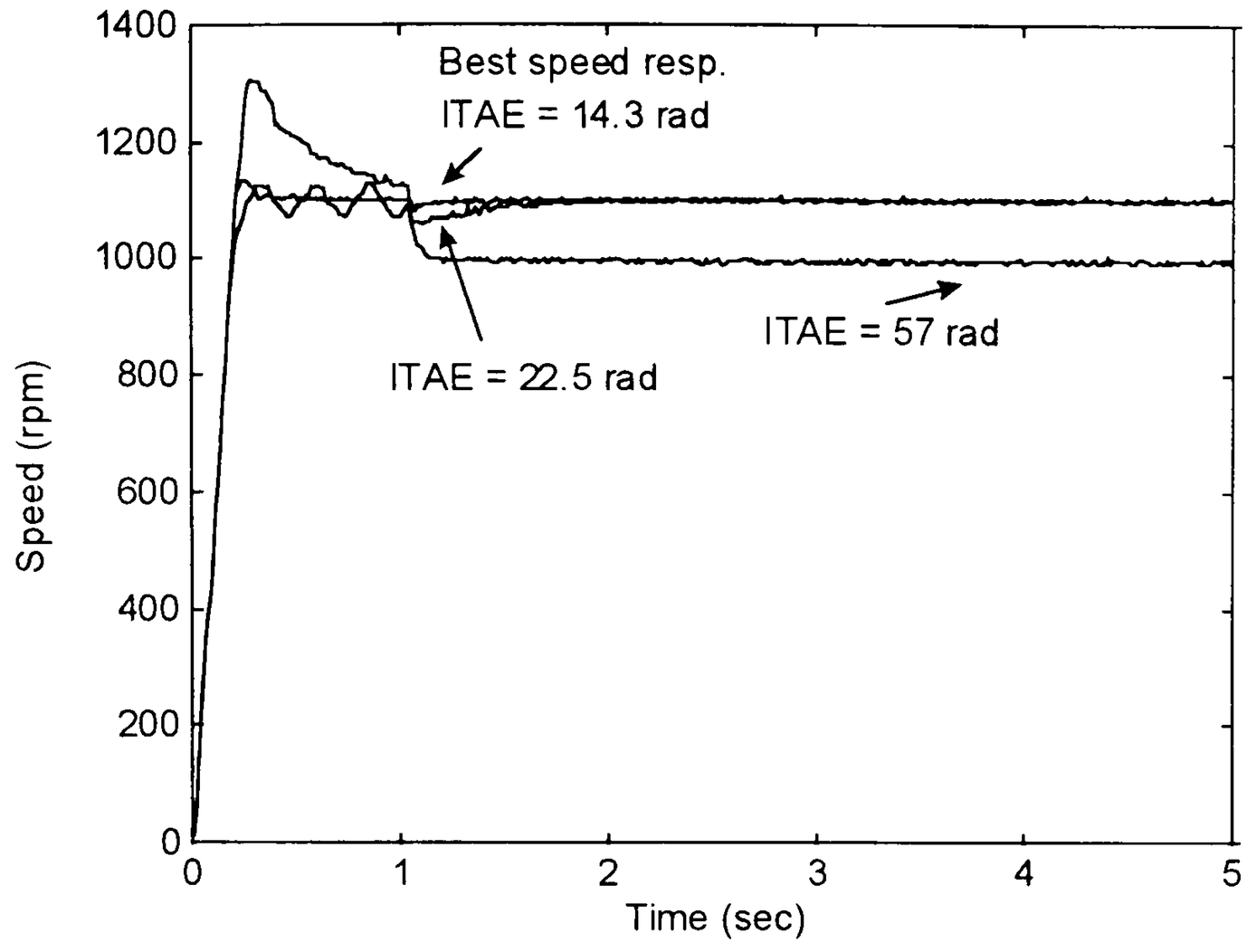



Fig. 5.26 - Speed responses following a large step input obtained with some chromosomes of the initial population and also the best after 20 generations for the optimisation of the PI speed controller and the setting of the *anti-windup* circuit.

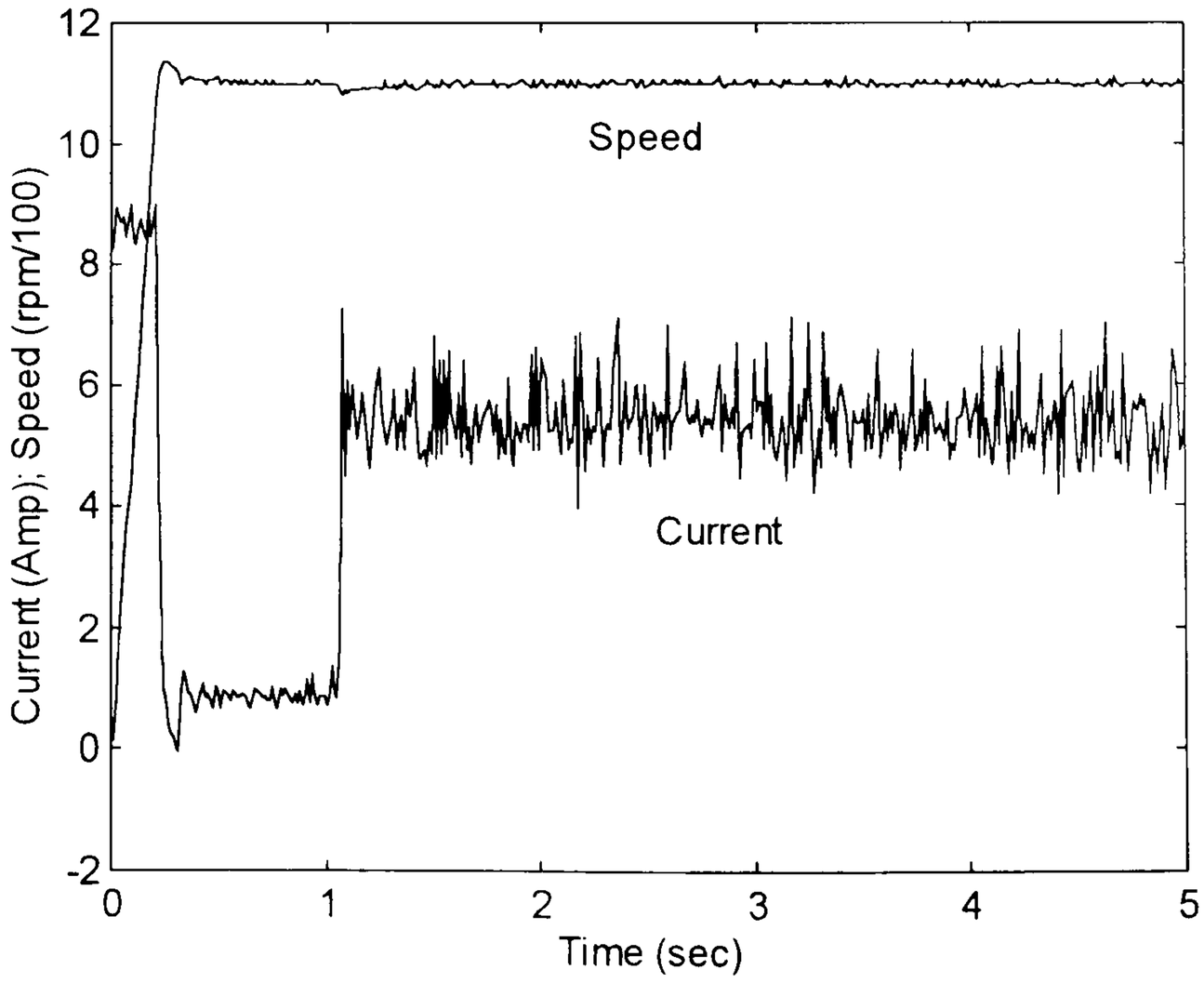



Fig. 5.27 - Best speed and current response after the optimization process for the PI speed controller with *anti-windup* circuit.

In Fig. 5.27 is shown the best speed and current response after the optimization process. The best chromosome found after 21 generation brought up a proportional gain of 3.7 A/rad/s, an integral gain of 16.7 A/rad and a dead-zone window of 8.4 A. The ITAE of this speed response was equal to 14.3 rad only. The optimal tuning is virtually impossible to be found manually as the load torque with which the motor started from standstill was different at the end of the time window.

## 5.5 - INVESTIGATION ABOUT THE EXISTENCE OF LOCAL MINIMA

A Genetic Algorithm for the optimisation of the setting of a PI speed controller of the Brushless DC Motor Drive has been used and results presented. Despite the non-linearities of the motor drive, a simple PI speed controller seems easy to tune as only two parameters determine the performance of the controller. Because of this, if the tuning of the parameters of the PI speed controller only are to be optimised, another method could possibly be utilised.

It has been said by researchers [MAN *et al*, 1996 and GEN *et al*, 1997] that one of the most important characteristic of Genetic Algorithms is the avoidance of local minima. However, if a particular problem is *well behaved*, local minima may not exist. If the the tuning of the PI speed controller as discussed in this chapter is one such case, then another method could possibly do the task in a shorter time then the Genetic Algorithm. In order to find out whether there are or are not local minima in the tuning the PI controller for the fastest speed response of the Brushless DC Motor Drive, the following test was done.

A large set of possible tuning for the PI speed controller was created and tested regarding the quality of speed response obtained through the ITAE for each run. The set of different tuning for the PI speed controller can be seen as a population of chromosomes of the Genetic Algorithm. A hundred individuals or chromosomes have been distributed within the searching space as shown in Fig. 5.28. The proportional gain ranges from 1 A/rad/s to 10 A/rad/s whereas the integral from 0 A/rad to 10 A/rad. The integral gain was concentrated close to the setting (0.4 A/rad) given by the on-line optimisation using a Genetic Algorithm for a 1000 rpm step input speed demand with a load torque of 67% of the maximum torque produced by the motor. The ITAE of the speed response was assessed during a 5 s time window for each

chromosome, in identical way as in section 5.4.1. The surface picture is shown in Fig. 5.29. The presence of local minima can be seen by looking at the contour map of the surface of Fig. 5.29, shown in Fig. 5.30. The local minima are also shown in Fig. 5.31, obtained by rotating Fig. 5.29 by -90° and enlarging the *Z* axis.

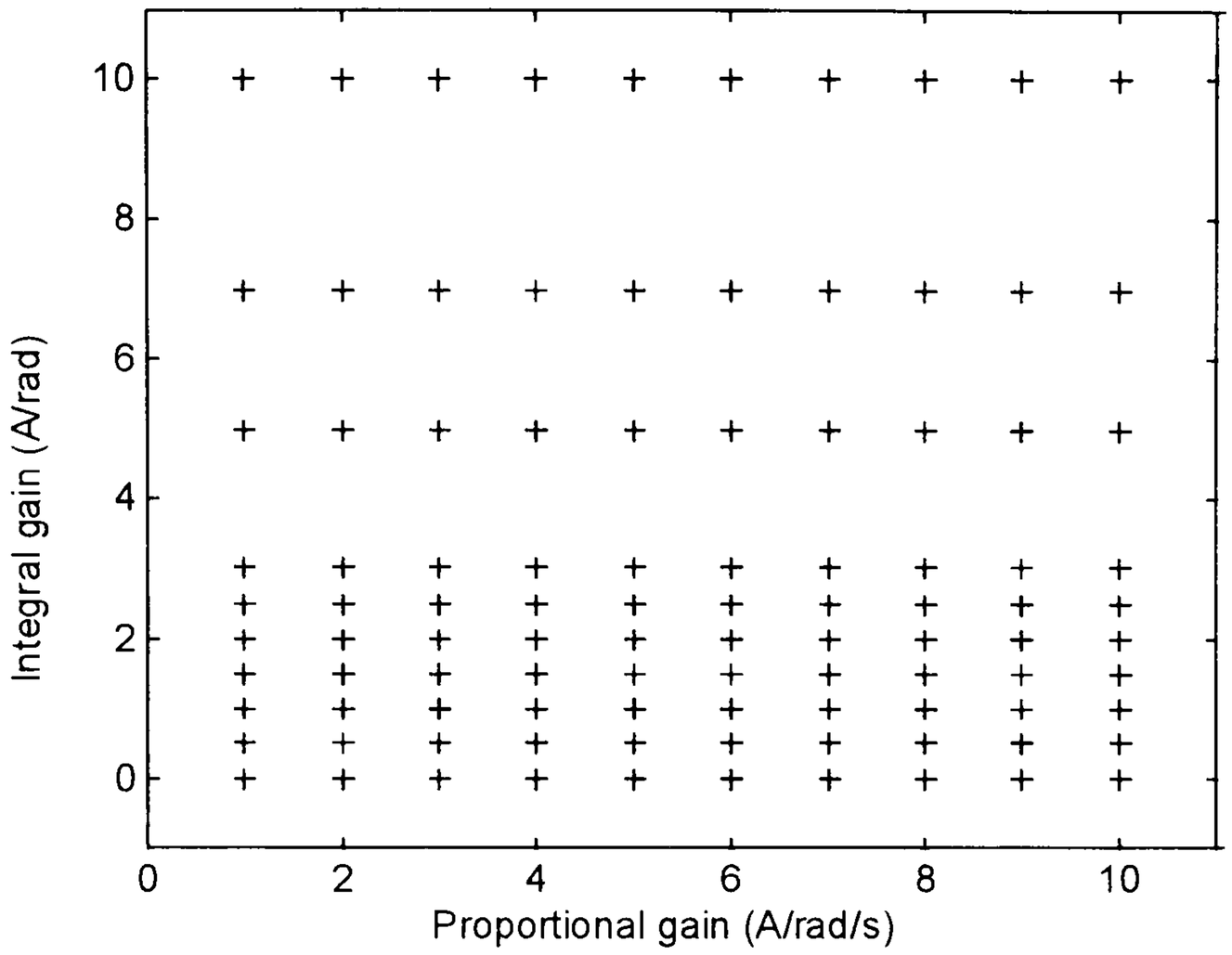



Fig. 5.28 - Population of chromosomes selected for the test of local minima.

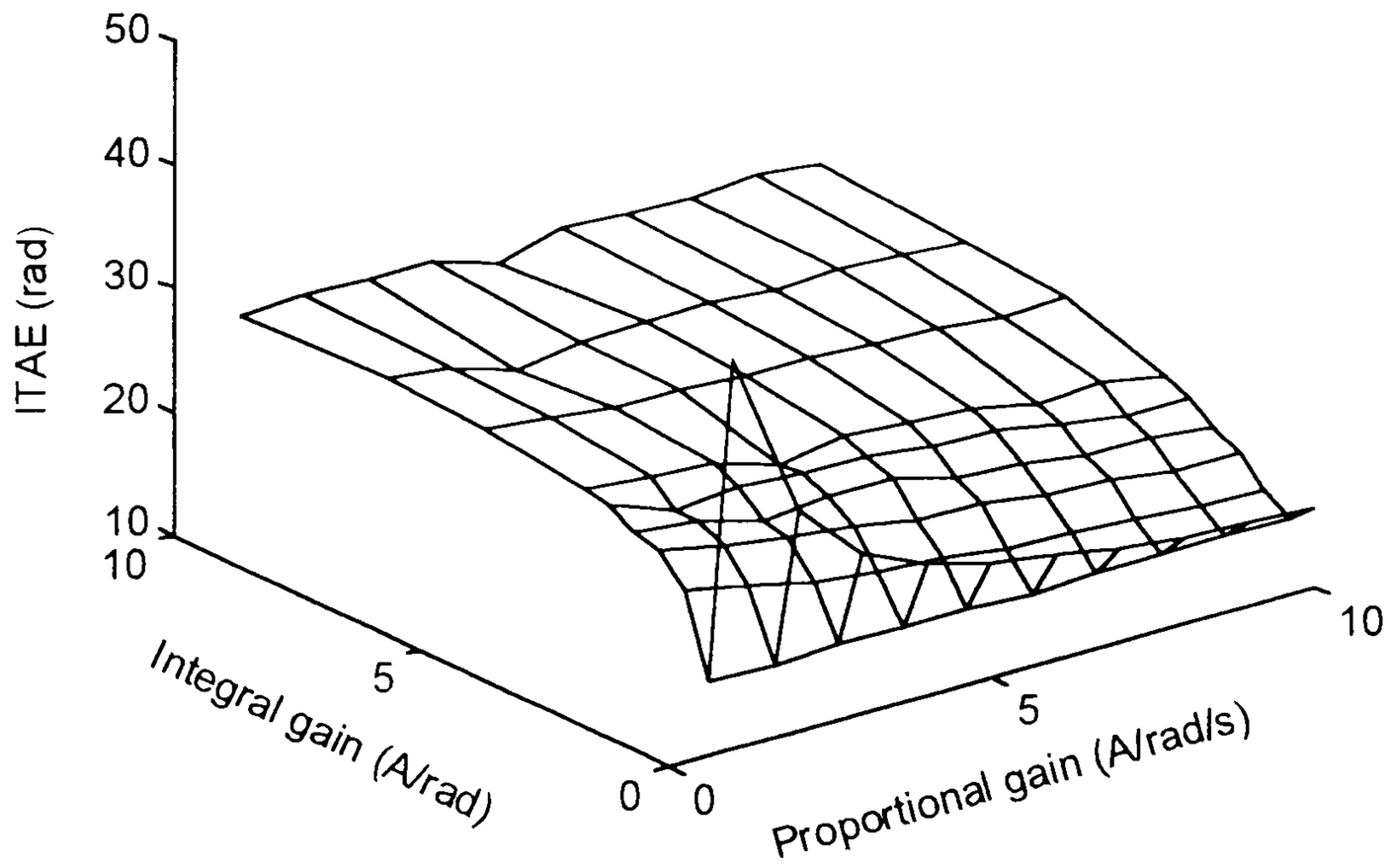



Fig. 5.29 - Surface given by the speed response of the Brushless DC Motor Drive as a function of the setting of the PI speed controller.

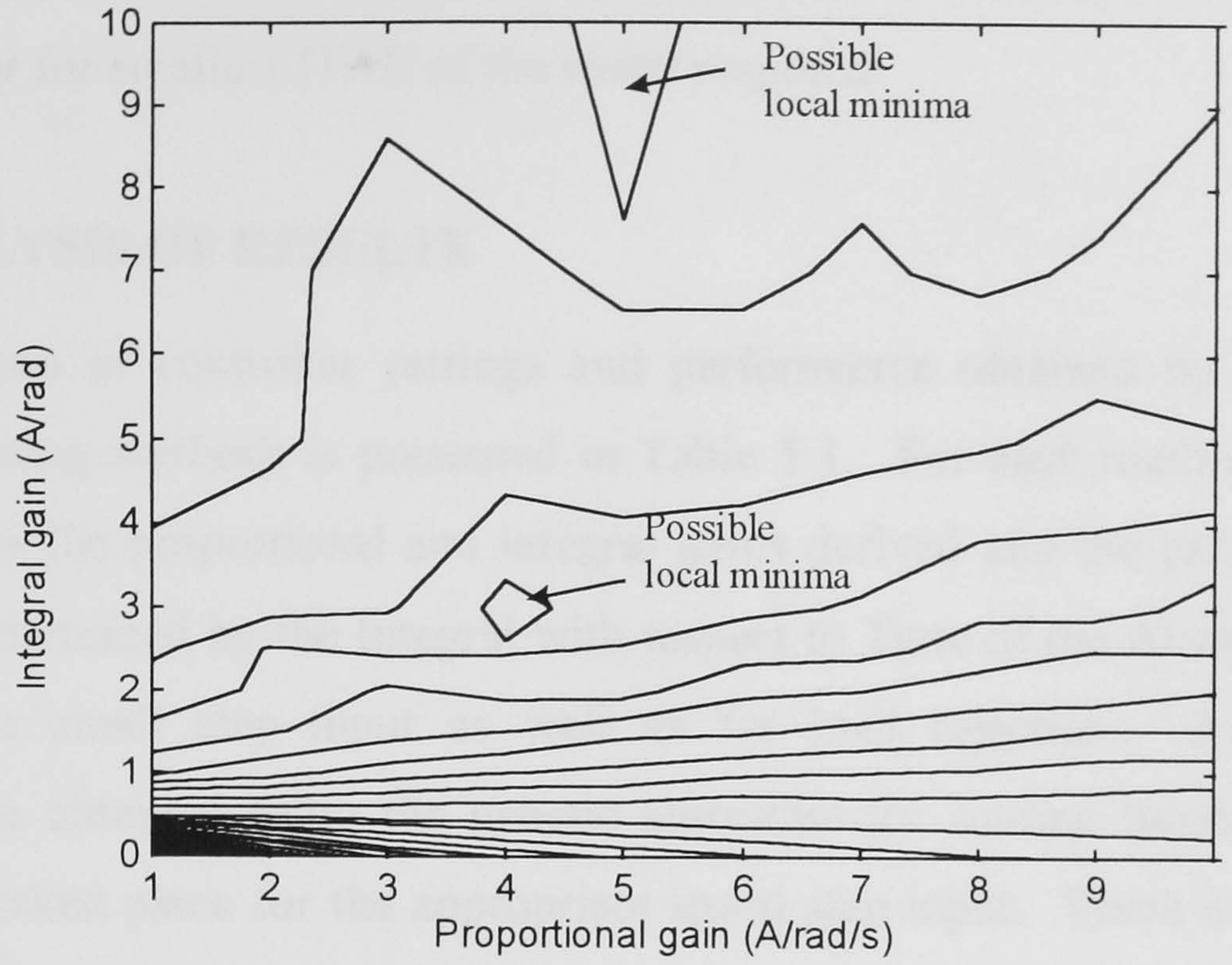



Fig. 5.30 - Contour map showing the existence of points of local minima.

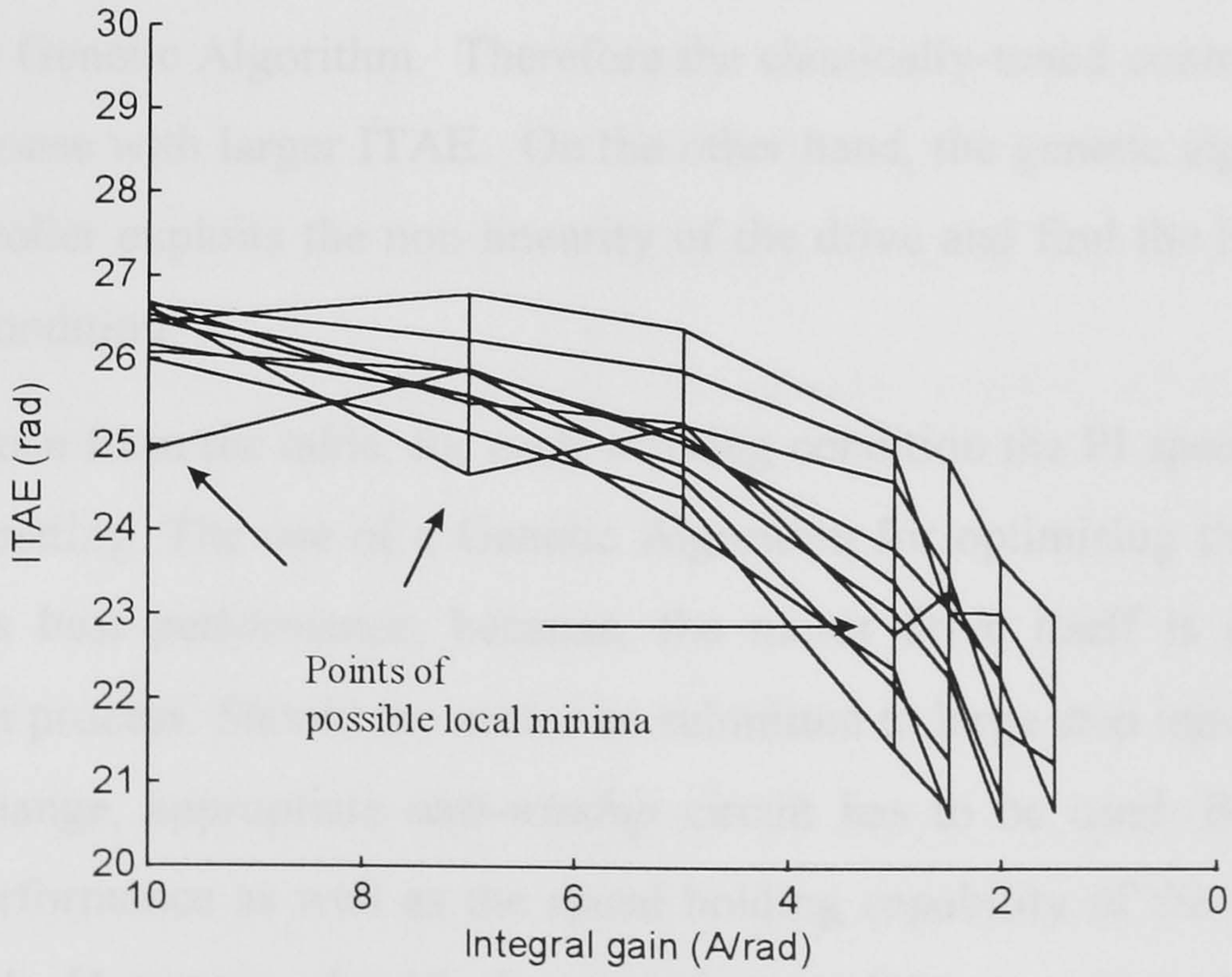



Fig. 5.31 - Surface picture rotated by -90° in enlarged $Z$ axis, highlighting the existence of points of local minima.

Because the existence of local minima, other methods could fail to find the global minima which represents the solution to the problem of finding the best tuning of the PI controller for smallest ITAE of the speed response.

## 5.6 - ANALYSIS OF RESULTS

A comparison of controller settings and performance obtained by using the three different tuning methods is presented in Table 5.1. For each method of tuning, the Table shows the proportional and integral gains derived and the performance of the drive (as determined by the Integral with respect to Time of the Absolute Error) for a large and a small step input as well as for load rejection. As expected, the performance obtained using the genetic algorithm for on-line tuning is best where tuning has taken place for the appropriate speed step input. There is a general trend towards higher values of both proportional and integral gain, when the speed controller is tuned for a small step input and for load disturbance, because the effect of the current limit and consequent integrator *windup* is less pronounced.

The tuning by using the classical approach, for a small step input speed demand, gives smaller proportional and integral gain for the speed controller, compared to the one obtained by Genetic Algorithm. Therefore the classically-tuned controller produces a slower response with larger ITAE. On the other hand, the genetic algorithm's tuning of the controller exploits the non-linearity of the drive and find the best tuning for a particular condition.

As can be seen from the table, for each working condition the PI speed controller had a different setting. The use of a Genetic Algorithm for optimising the controller on-line ensures best performance, because, the motor drive itself is used during the optimisation process. Should the motor be submitted to large step input speed demand and load change, appropriate *anti-windup* circuit has to be used. By doing so, the transient performance as well as the speed holding capability of the motor drive can be improved. However, should the operating condition to which the controller has been tuned change, the quality of the speed response deteriorates. In order to ensure best performance, another tuning has to be found.

Table 5.1 Controller settings and ITAE obtained using different approaches to tuning.

| Approach used for tuning | Section | Tuning of the PI speed controller | | Experimental ITAE (rads) | |
|---|---|---|---|---|---|
| | | P gain, A/rad/s | I gain, A/rad | input 0-1100 rpm 5s window | input 1000-1100 rpm 1.25 s window |
| Gen. Algorithm (0-1100 rpm) | 5.4.1 | 5.2 | 0.4 | 17.02 | 0.60 |
| Gen. Algorithm (1000-1100 rpm) | 5.4.2 | 2.7 | 2.9 | 30.5 | 0.56 |
| Classical Control Theory (1000-1100 rpm) | 5.4.5 | 0.28 | 1.2 | 26.75 | 0.81 |
| | | | | | |
| Gen. Algorithm in the presence of Load Dist. | 5.4.3 | 3.42 | 19.4 | 0.81* | |
| Gen. Algorithm in the presence of speed and load changing | 5.4.4 | 9.6 | 0.0 | 12.2 | |

* Reading taken over 3 s time window and double number of sampled points of sections 5.4.1-5.

## 5.7 - CONCLUSION

This chapter has illustrated the capability of genetic algorithms for obtaining optimised tuning of electric drive speed controllers in the presence of major non-linearities. The technique has been applied to the specific problem of optimising PI speed controller parameters in a Brushless DC Motor Drive subject to a hard current

limit and with the aim of minimising the ITAE occurring after a step change in speed demand. It has also been applied to the optimisation of performance in the presence of load changes. However, it can be applied for other operating conditions, e.g. maximum overshoot or mimimum response time. Due to the existence of local minima, other methods could fail to find the optimum solution for such a problem.

# CHAPTER 6

# THE USE OF LOAD ESTIMATOR

## 6.1 - INTRODUCTION

The speed response of the Brushless DC Motor with PI speed controller was investigated in chapters 4 and 5. It has been shown that the performance of the controller is dependent on the condition in which the motor is working. Should the load torque or speed demand change, a different setting for the controller has to be found in order to ensure best performance. In addition to this, it has also been shown that non-linearities play an important role in the way the controller is tuned. However, there are many different applications for motor drives and, in this chapter, special attention is given to the application where the speed of the motor has to be kept constant despite load torque changes. Nevertheless, the dynamic of the motor drive due to a large step input speed demand is also investigated here as a load estimator is used in conjunction with the PI speed controller.

In order to improve the performance of the Brushless DC Motor Drive in use, as described in chapter 3, the load estimator proposed by Iwasaki [IWASAKI *et al*, 1991] was used, together with a Proportional-Integral speed controller. The proposed load estimator is based on the assessment of the load torque, as follows. In the first place, the load torque is determined by multiplying the motor speed differentiation by the motor/load inertia constant. The load torque is compared to the electric torque produced by the motor and the difference is then fed back into the control system through a low-pass filter, providing additional current demand for the motor. The block diagram of the Brushless DC Motor Drive in use, as shown in Fig. 3.1, with the proposed load torque estimator is presented in Fig. 6.1. The points A, B, C, D and E pointed out in the picture are references to help the reader identify where the control

signals are taken when the pictures they relate to are displayed.

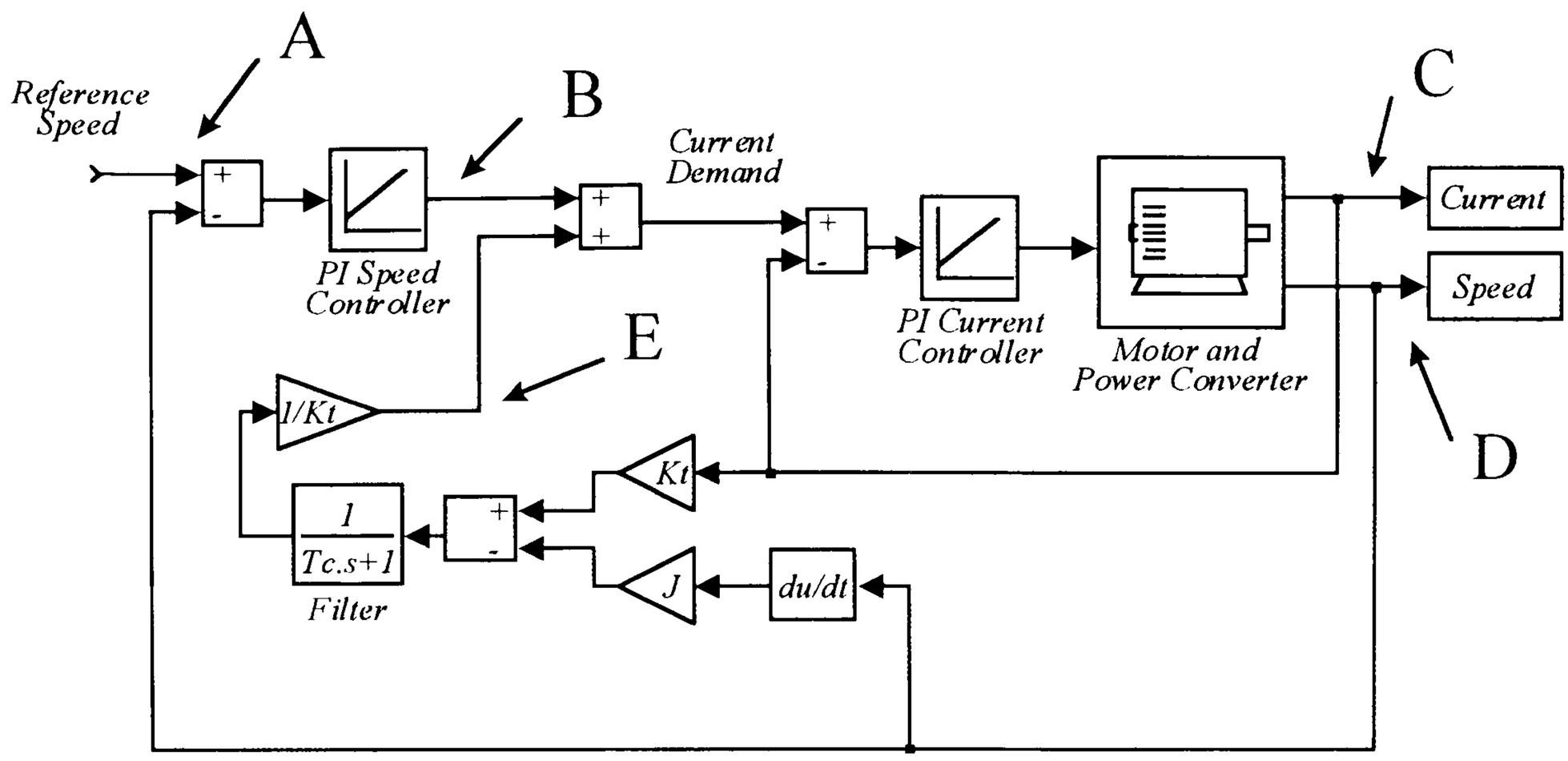



Fig. 6.1 - Speed control brushless dc motor block diagram with load estimator.

Where,

*J* - Motor and Load inertia;

*Kt* - Brushless dc motor torque constant;

*Tc* - Low pass filter time constant.

The aim of the load estimator is to provide additional current demand to the PI current controller. As this signal comes straight from the load estimator into the current controller, it provides fast action due to the load change.

The performance of the motor drive after the inclusion of the load observer, as shown in Fig. 6.1, was investigated. The conditions in which the motor was tested are the same as those used in chapter 4. Firstly (section 6.2), the motor is run within identical speed controller setting as used in chapter 4. Thus, the proportional gain was set to 0.08 A/rad/s whereas the integral one was set to 0.32 A/rad. It means that the only difference between the tests shown here and those presented in chapter 4, comes from the presence of the load estimator, which was added to the block diagram shown in Fig. 3.1. In section 6.3, a Genetic Algorithm is used to optimize the controller when the motor is driven at constant speed, in the presence of load disturbance. Finally (section 6.4), the robustness against load variation is investigated by driving the motor at different load condition and speed demand.

## 6.2 - EFFECT OF LOAD ESTIMATOR

In order to help understand the improvement the load estimator can bring to the performance of the speed response of the Brushless DC Motor Drive, the following tests have been done. The speed and load conditions are identical to that presented in chapter 4 so comparison can be made. The proportional gain is 0.08 A/rad/s and the integral one, 0.32 A/rad. The low pass filter time constant, *Tc* was set to 62.5 ms.

### 6.2.1 - 1100 rpm step input speed demand with 62% of the full load

In this test, the motor was at rest when an 1100 rpm step input speed demand was applied. The resistive load bank was set to produce 62% of the maximum load torque at that speed. The speed and current response is shown in Fig. 6.2. The reference speed corresponds to the input signal applied at the point "A" shown in Fig. 6.1. The speed curve corresponds to the signal obtained at point "D" and the current, to the point "C".

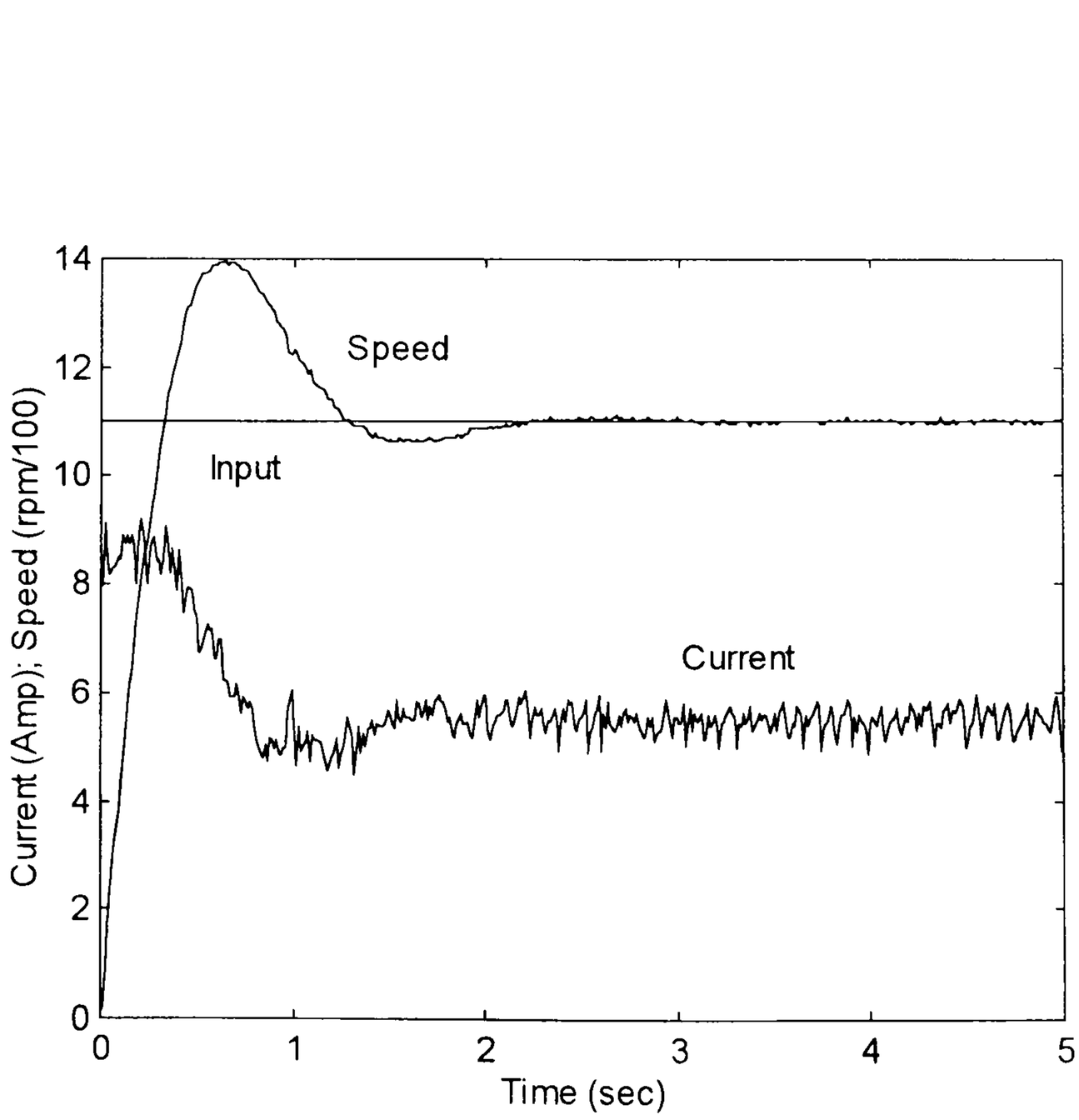



Fig. 6.2 - Speed and current response of the brushless dc motor drive for 1100 rpm step input speed demand with 62% of the maximum load torque.

The response is underdamped with 27% speed overshoot. It means that, for this condition, the integrator gain is excessive, causing the overshoot. As the block diagram that represents the drive system has changed, in order to avoid this overshoot, another setting for the speed controller should be found.

Fig. 6.3 presents the output of the PI speed controller and the estimated load. At steady state the estimated load holds the same value as the current of the motor. This was expected as, at steady state, since the speed is constant, its differentiation is zero. As a result, the current of the motor is fed back into the control system. The sum of both signals determines the current demand supplied to the current controller, as shown in Fig. 6.2. The "Estimated Load" curve corresponds to the signal obtained from point "E" in Fig. 6.1 whereas, "Speed Cont. Output" corresponds to the signal obtained at point "B".

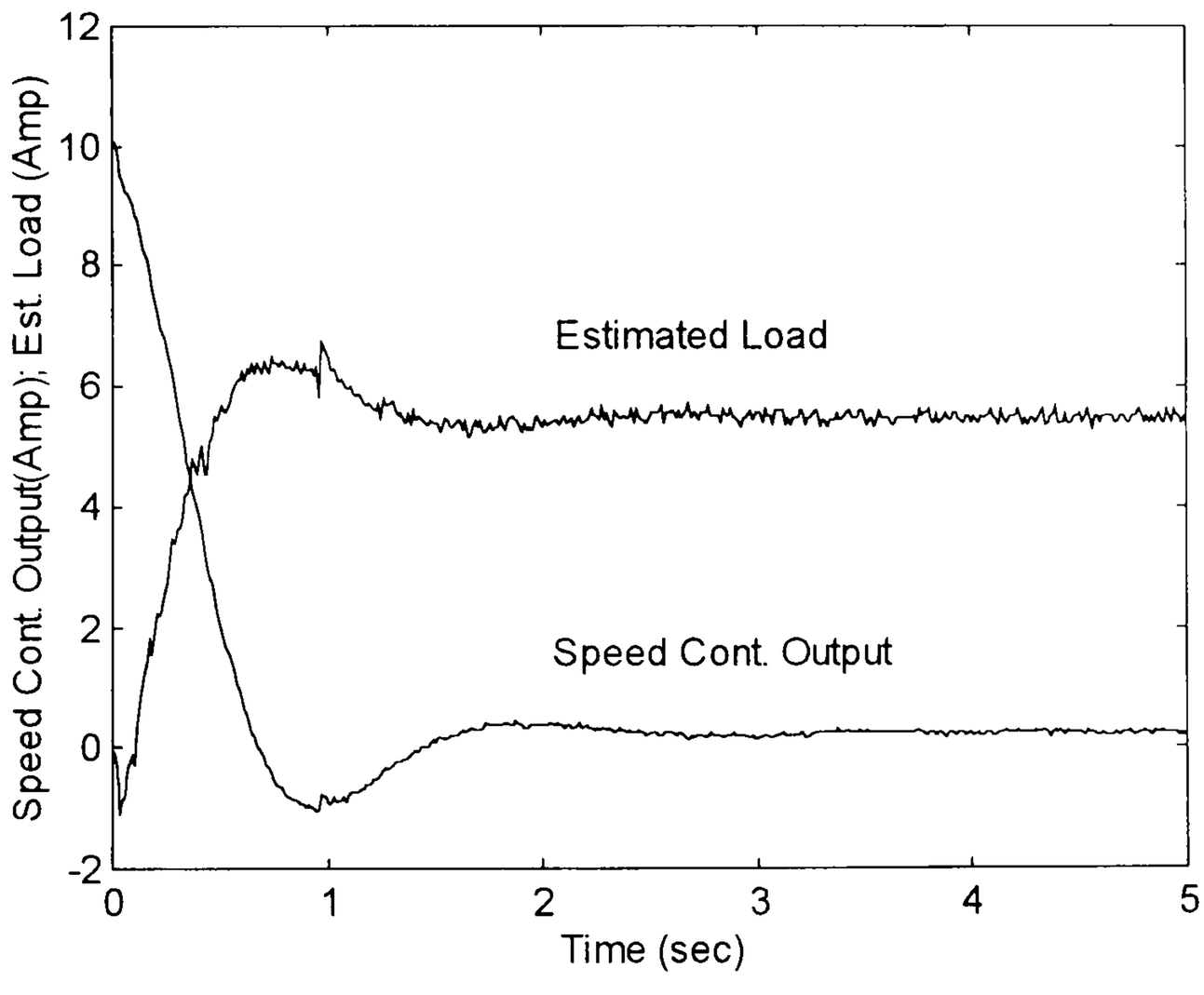



Fig. 6.3 - Output of the PI speed controller and Estimated Load due to a 1100 rpm step input speed demand and 62% of the full load torque.

### 6.2.2 - 1100 rpm step input speed demand with 11% of the full load

In this condition, an 1100 rpm step input reference speed was applied from standstill with the load bank set to produce only 11% of the maximum load torque at full speed.

The speed and current response is shown in Fig. 6.4. In this case, the speed overshoot was little larger then before (31%).

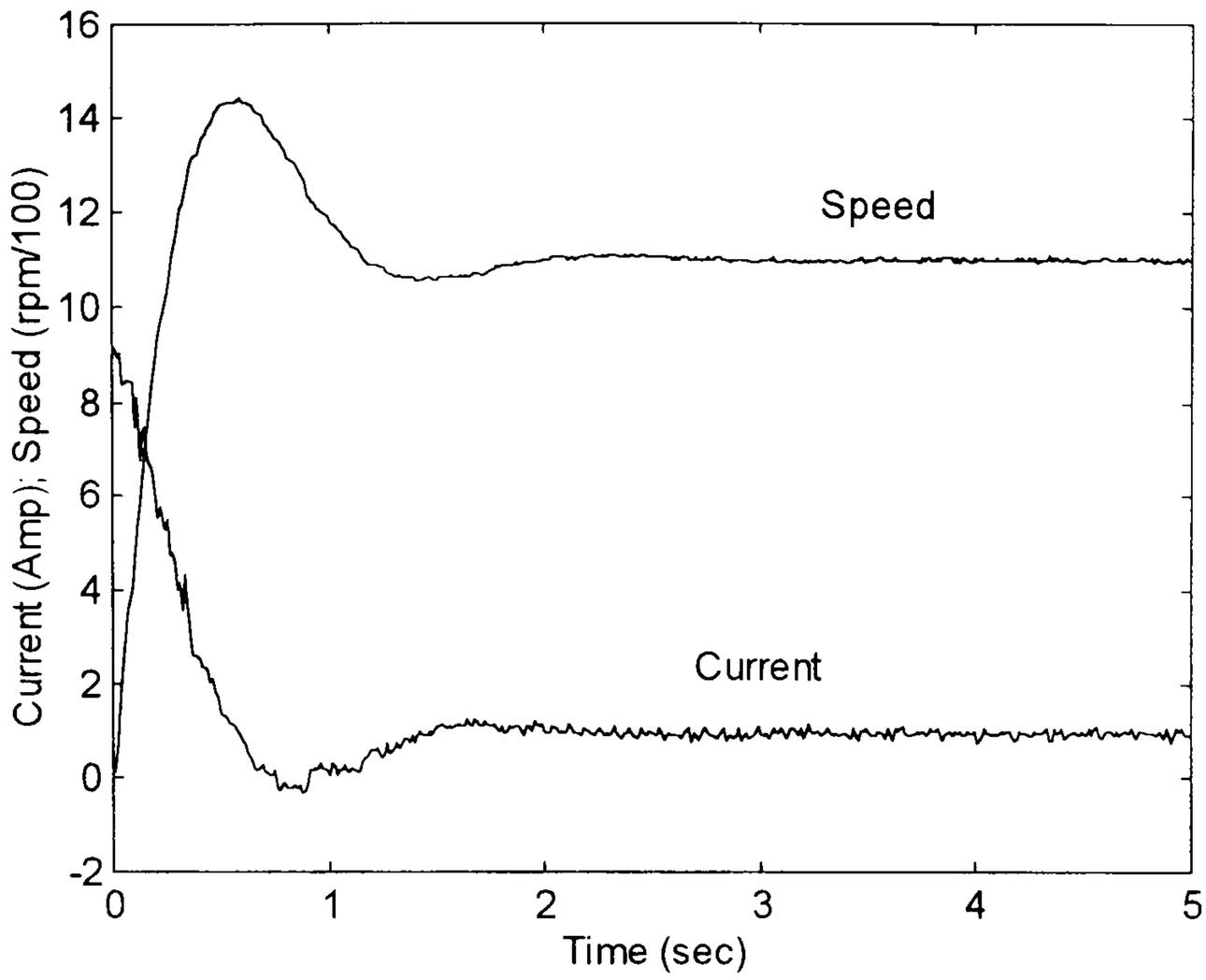



Fig. 6.4 - Speed and current response of the brushless dc motor drive due to 1100 rpm step input speed demand with 11% of the maximum load torque.

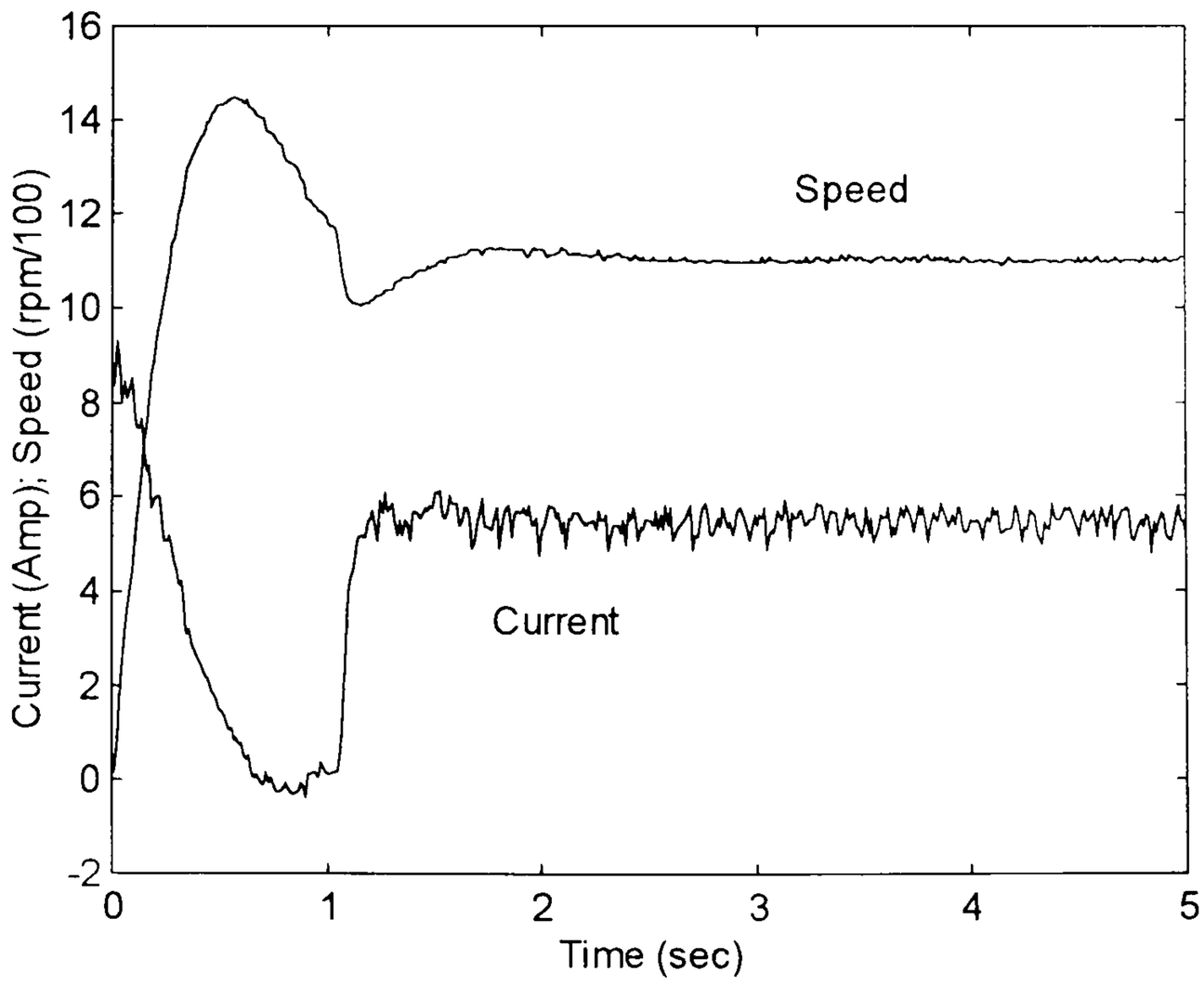



Fig. 6.5 - Speed and current response of the motor drive in the presence of load disturbance.

### 6.2.3 - 1100 rpm step input speed demand in the presence of load disturbance

For this test, an 1100 rpm step input reference speed was applied to the motor with 11% of the maximum load torque. At *time* = 1 s, a step input load disturbance was applied, bringing the load torque to 62% of the full load for the remaining 4 s of the time window. The speed and current response is presented in Fig. 6.5. It is apparent that, because of the load estimator, the current rises faster when the disturbance is applied, compared to that of Fig. 4.5, without the load estimator. It means that the speed recovery due to the disturbance takes place in a shorter time.

### 6.2.4 - 1500 rpm step input speed demand in the presence of load disturbance

Similarly to what has been done before, a 1500 rpm step input speed demand was applied to the motor from standstill, with 11% of load torque. Again, at *time* = 1 s, a step input load disturbance was applied, bringing the load torque to 75% of it maximum during the rest of the time window, as shown in Fig. 6.6. At that time (*time* = 1 s), the speed was not yet settled, from the step input. There was a temporary speed drop because of the disturbance. However, it is again apparent that the current rose faster due to the load estimator, when compared to that of Fig. 4.6, where only the PI speed controller was in use. Nevertheless, it is clearly seen that there was saturation of the speed controller, as the current stayed at its maximum value during all the acceleration time. The motor speeded up towards the speed demand with the maximum torque.

### 6.2.5 - 1100 rpm constant speed in the presence of the load disturbance

As the aim of the load estimator is to help the controller to keep the speed of the motor constant despite load variation, the performance of the motor drive against load variation only was also investigated. At this time, the motor was running at constant speed with 11% of load torque. At *time* = 1 s, the load was suddenly increased to 62% of the full load torque. The speed holding capability with and without the load estimator is shown in Fig. 6.7 where the presence of the load estimator has improved significantly the speed holding capability. It is clearly seen that the current level with

and without load is the same in both cases. It means that the same initial load as well as the disturbance was applied to the motor. It is also seen that there was a speed overshoot when recovering from the disturbance. The reason is that the gain of the integrator is too high and the controller needs to be re-tuned.

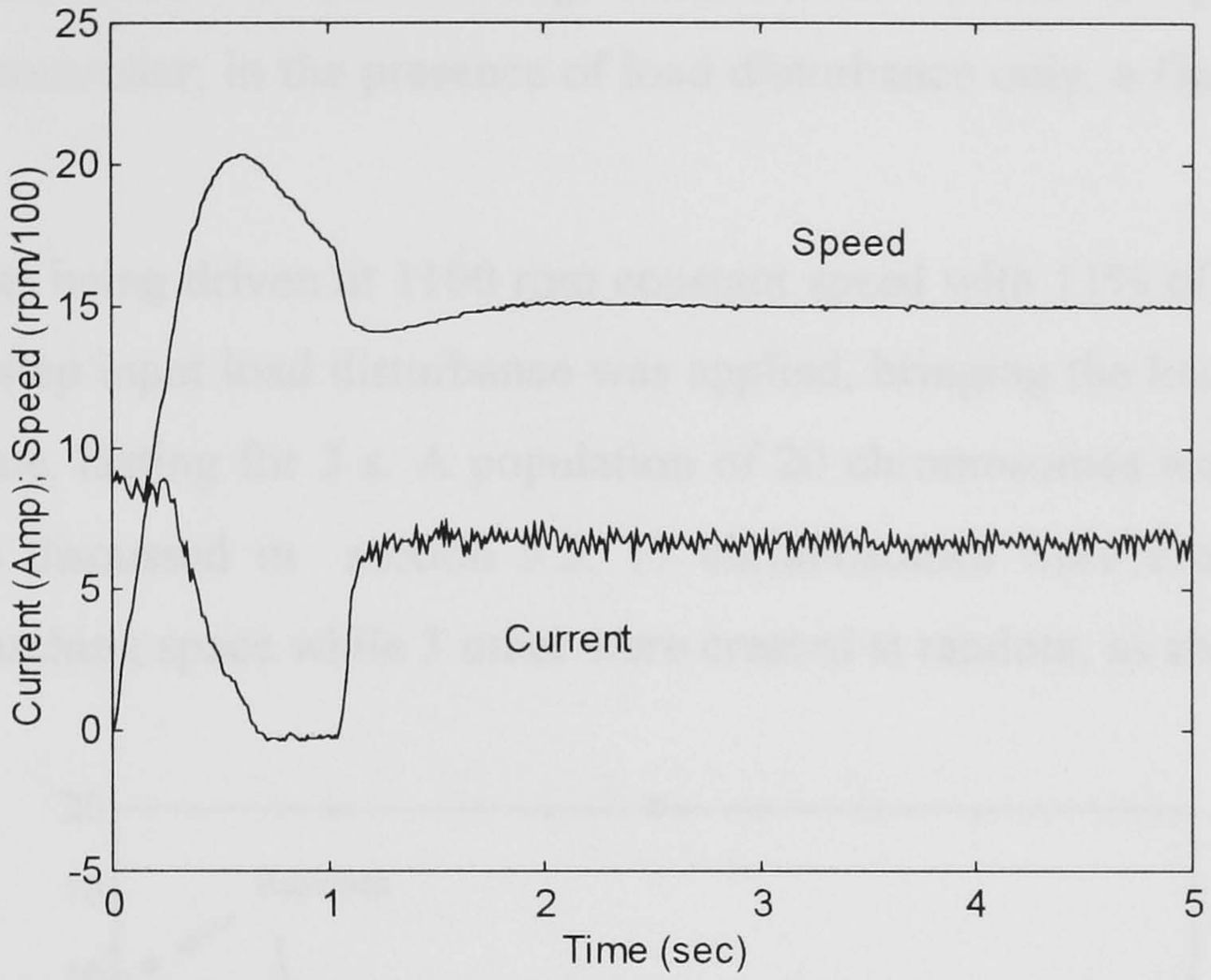



Fig. 6.6 - Speed and current response due to 1500 rpm step input speed demand in the presence of load disturbance.

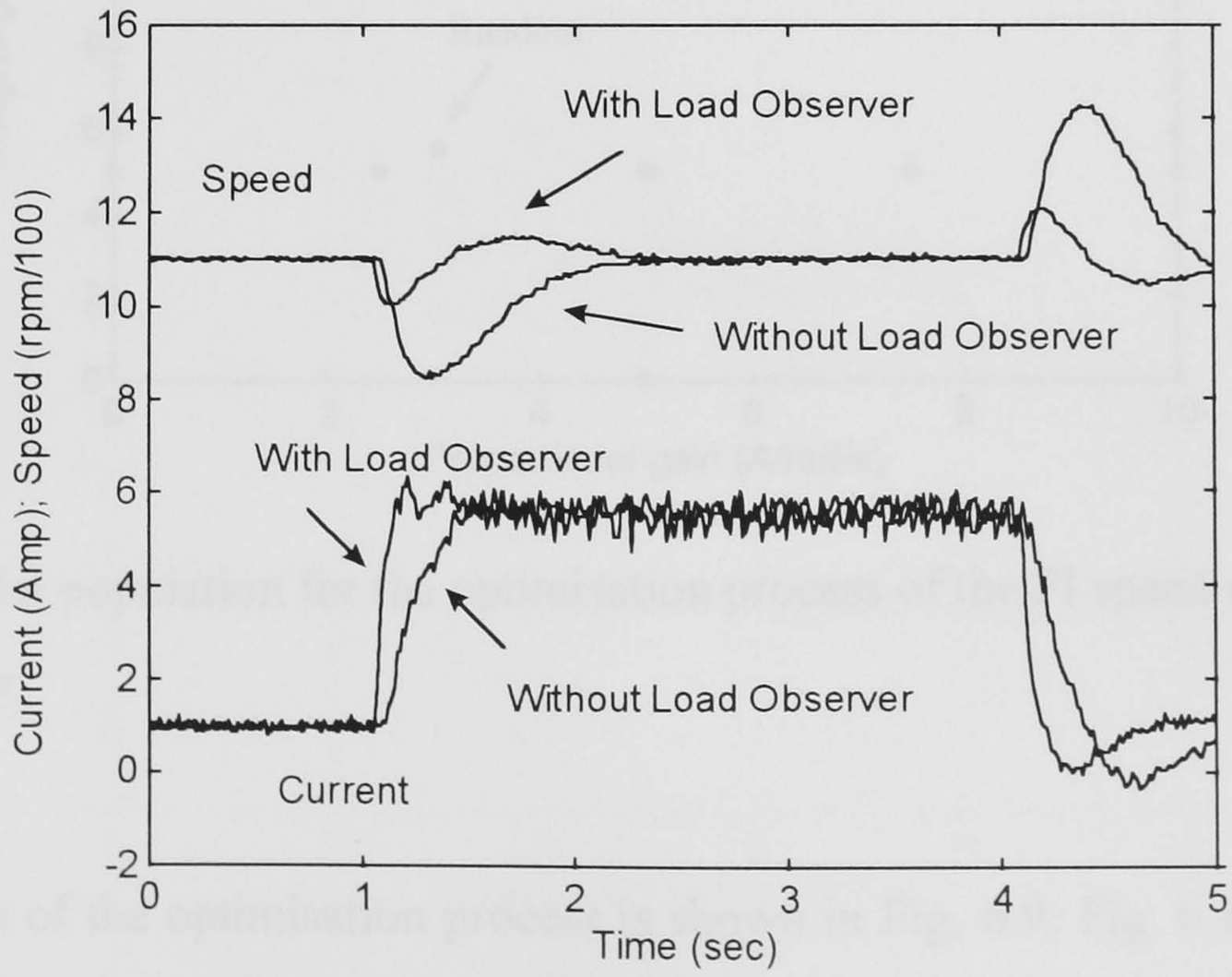



Fig. 6.7 - Speed holding capability of the PI speed controlled Brushless DC Motor Drive with and without load estimator.

## 6.3 - OPTIMIZATION OF THE SPEED CONTROLLER BY GENETIC ALGORITHM

As shown previously, the inclusion of the load estimator has improved the speed holding capability of the motor drive. However, it has been realised that the PI speed controller should have a different setting. Thus, in order to find the optimal tuning for the PI speed controller, in the presence of load disturbance only, a Genetic Algorithm was used.

The motor was being driven at 1100 rpm constant speed with 11% of load torque. At *time* = 1 s, a step input load disturbance was applied, bringing the load to 62% of the full load torque, lasting for 3 s. A population of 20 chromosomes was created in the same way as discussed in section 5.3. 17 chromosomes were evenly distribuited within the searching space while 3 other were created at random, as shown in Fig. 6.8.

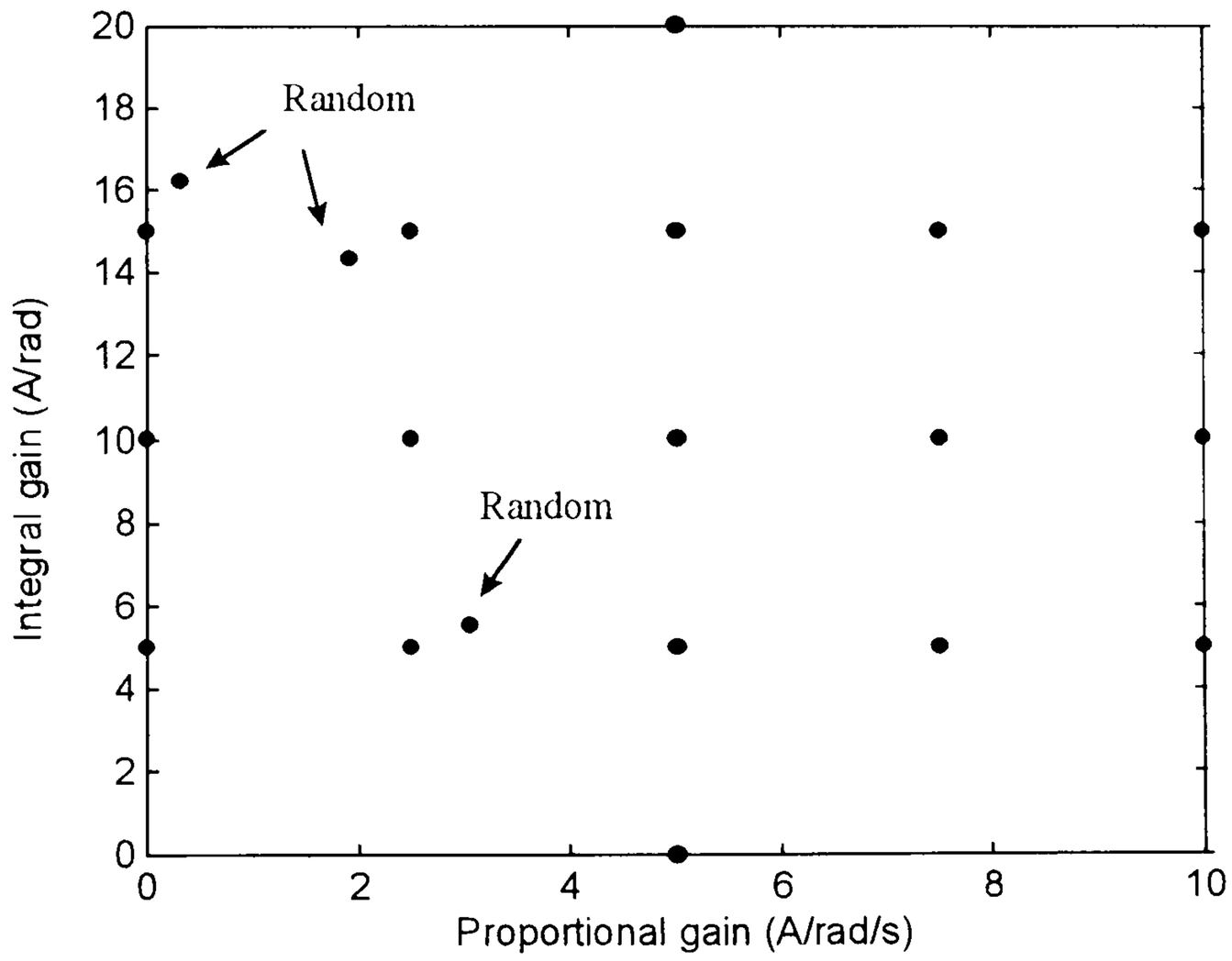



Fig. 6.8 - Initial population for the optimisation process of the PI speed controller with load estimator.

The evolution of the optimisation process is shown in Fig. 6.9. Fig. 6.10 presents the final population of chromosomes whereas Fig. 6.11 illustrates the best speed and current response obtained by the genetic algorithm after 21 generation. Fig. 6.12

presents the speed in enlarged scale, highlighting the speed variation during the disturbance. The ITAE of the best speed response, measured during the disturbance only, was 0.73 rad against 0.81 rad shown in section 5.4.3, without the load estimator.

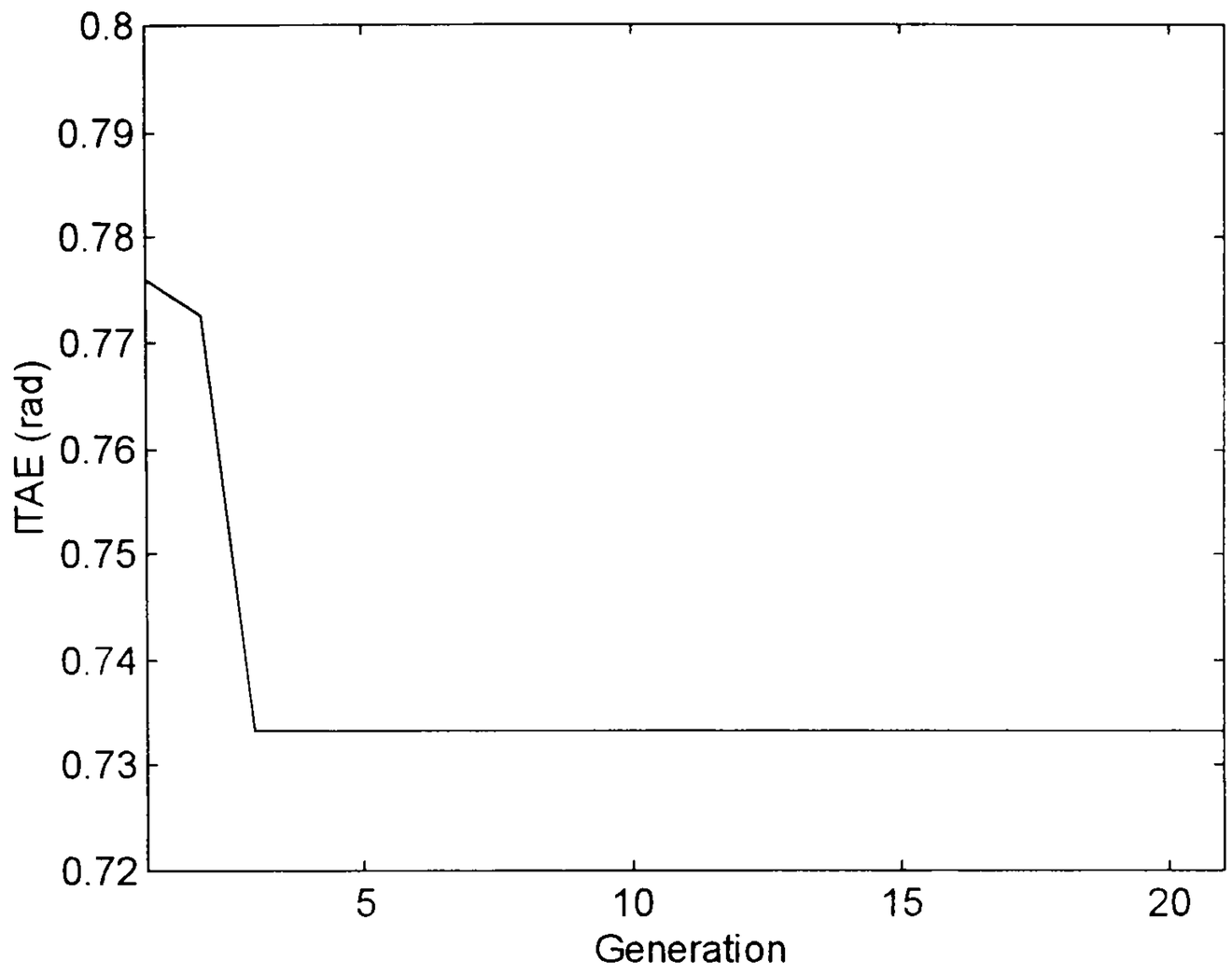



Fig. 6.9 - Genetic Algorithm evolution for the optimisation of the PI speed controller in the presence of load disturbance.

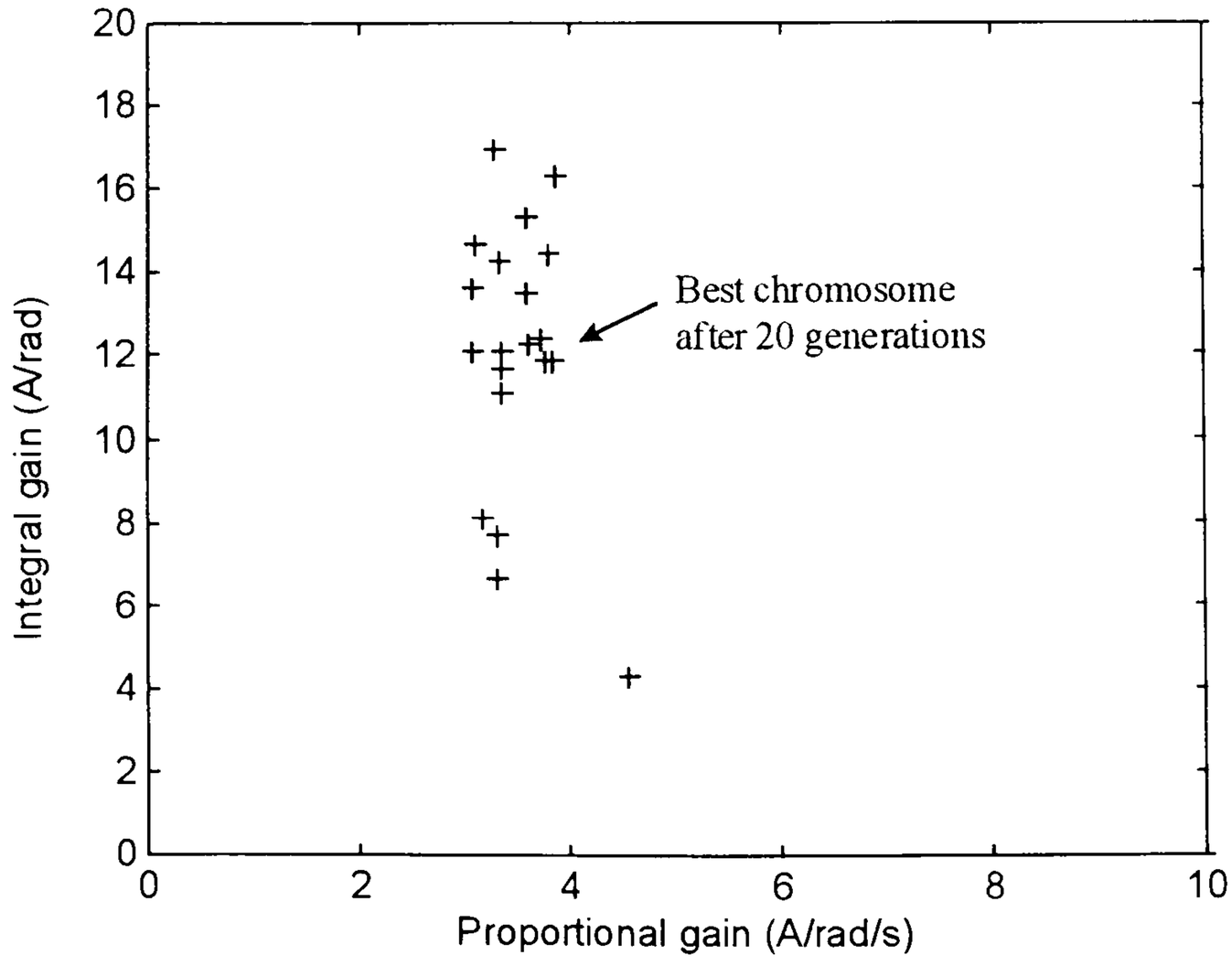



Fig. 6.10 - Final population after the optimisation process for the tuning of the speed controller with load estimator.

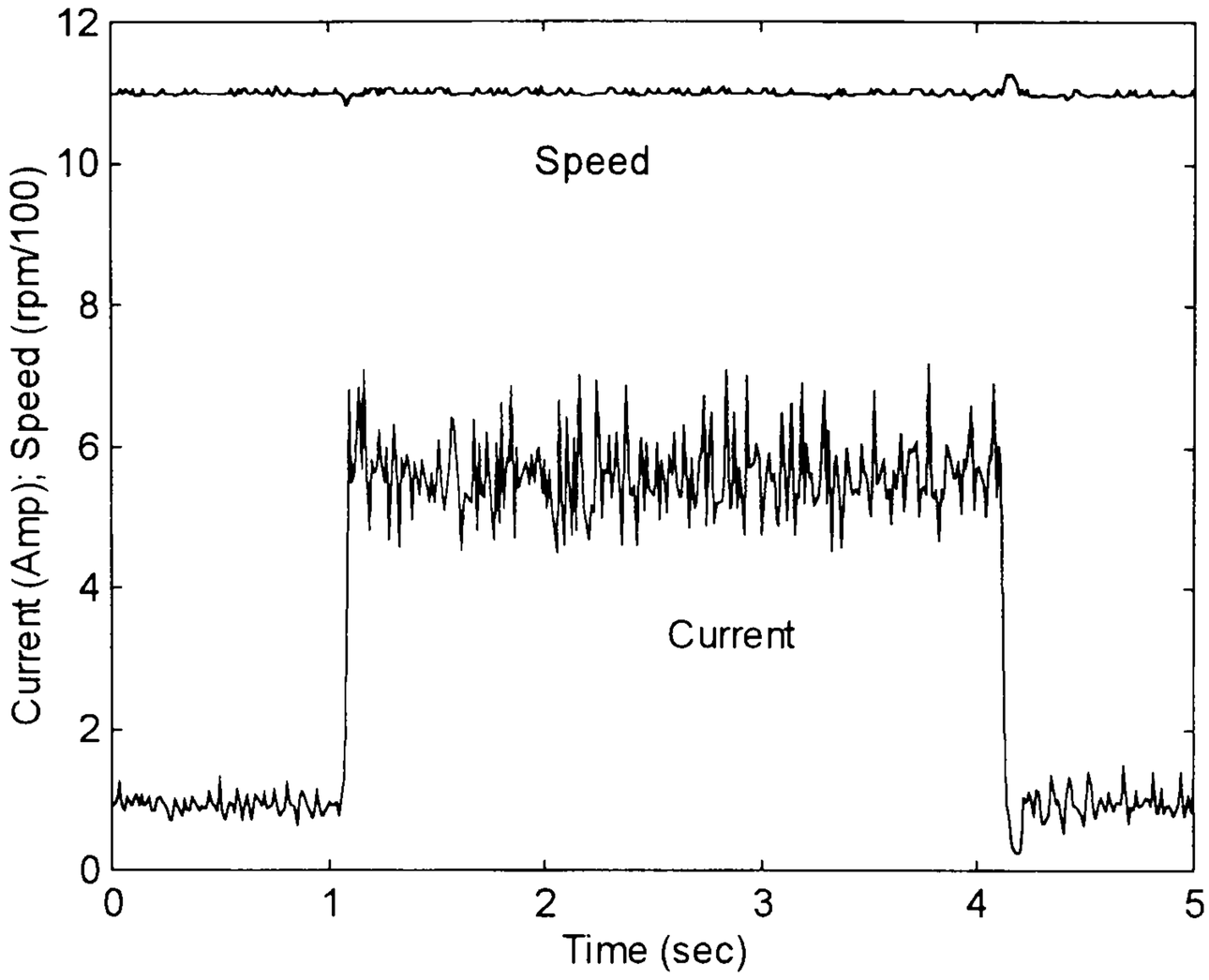



Fig. 6.11 - Best speed and current response obtained after the optimisation of the PI speed controller with load estimator, by Genetic Algorithm.

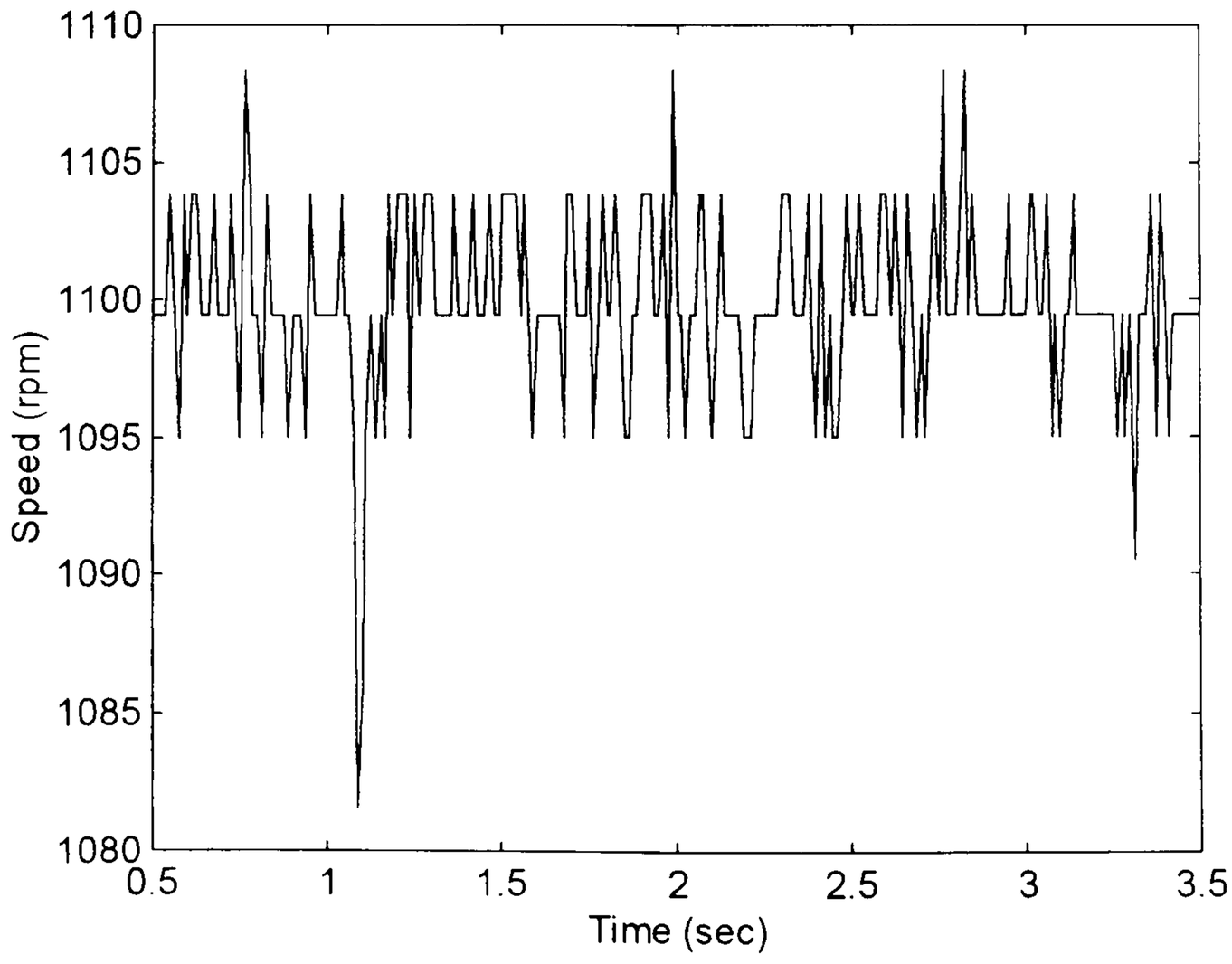



Fig. 6.12 - Best speed response in enlarged scale highlighting the speed variation due to the disturbance.

As can be seen in Fig. 6.12, the speed variation is very small despite the large load torque disturbance. In addition to this, the recovery time is very short, lasting fractions of seconds.

## 6.4 - INVESTIGATION INTO THE EXISTENCE OF LOCAL MINIMA

Another investigation into the existence of local minima around the setting given by the Genetic Algorithm was done. The motor was running in identical condition as in section 6.3. The search was concentrated close to the setting of the PI speed controller given by the Genetic Algorithm in section 6.3, as shown in Fig. 6.13.

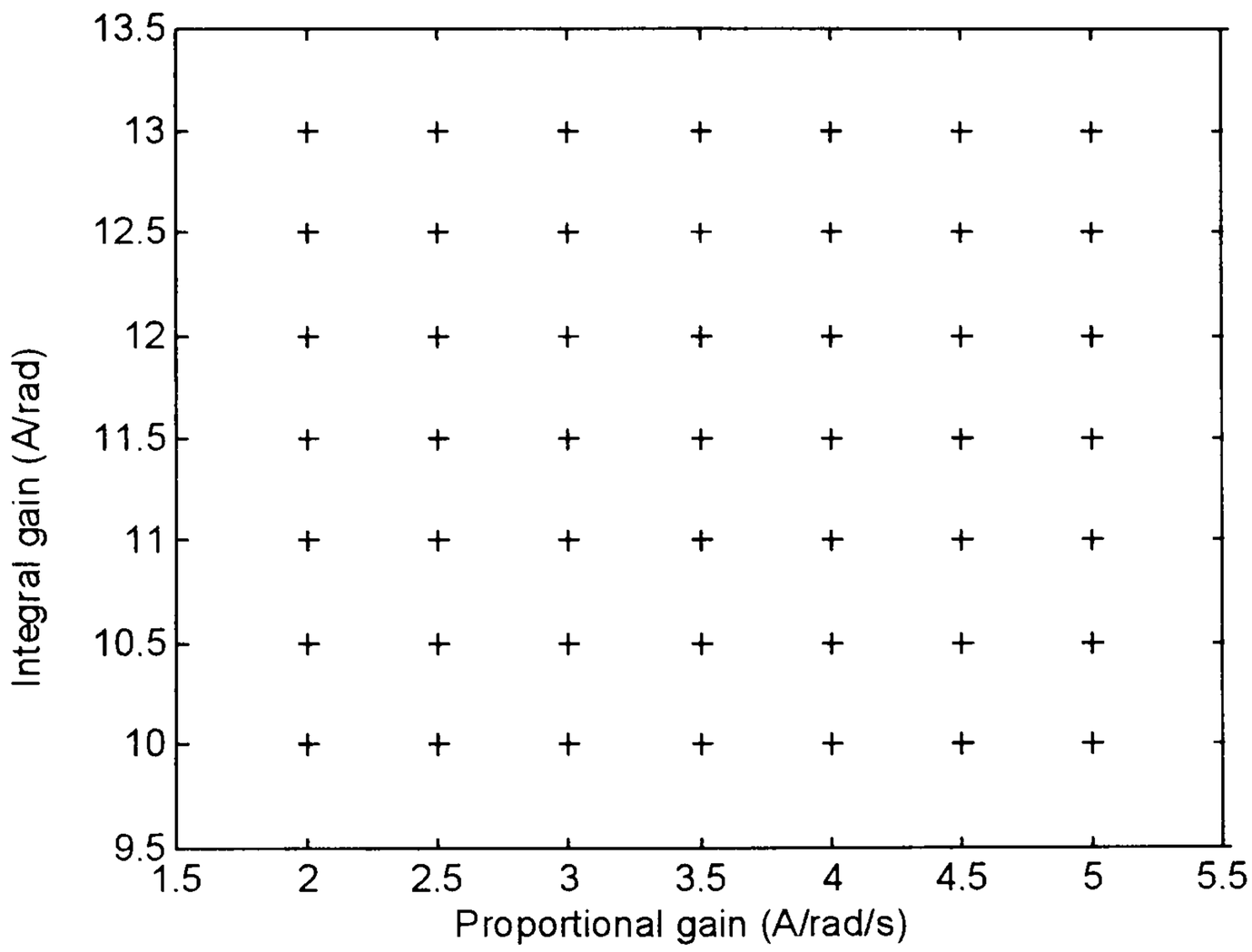



Fig. 6.13 - Population of chromosomes selected for the test on the existence of local minima.

The surface given by the ITAE of the speed response obtained with each setting of the speed controller shown in Fig. 6.13 is presented in Fig. 6.14 whereas the contour map plot is shown in Fig. 6.15. It can be seen again that there are several possible points of local minima, close to the solution given by the Genetic Algorithm. However, it is important to highlight the following: due to the sampled speed data within a small time window, the ITAE obtained in one run differs a little from the one before even if the setting of the controller does not change. As a consequence, many different

settings around the solution found by the Genetic Algorithm can give similar ITAE and are close to the best tuning for the speed controller.

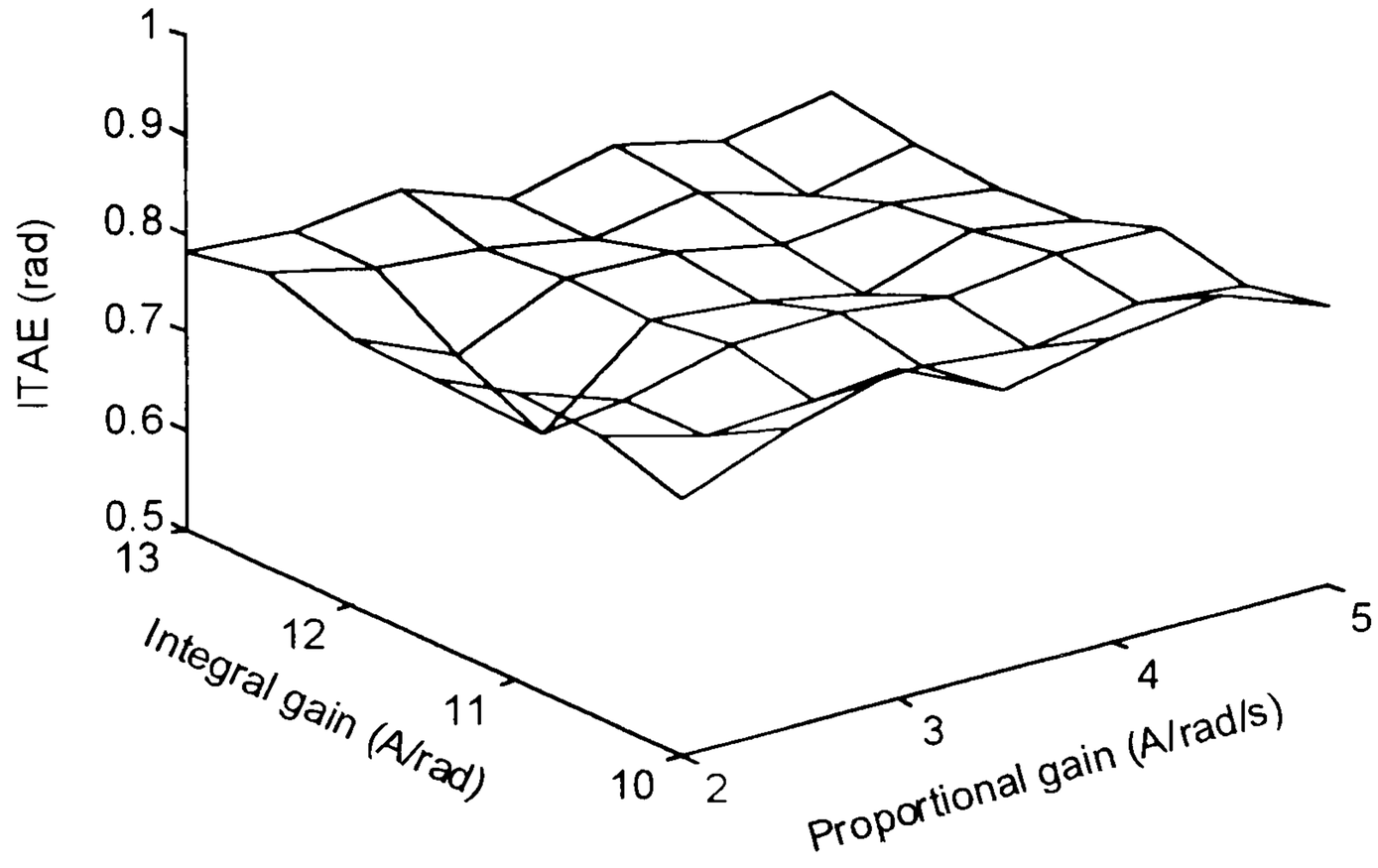



Fig. 6.14 - Surface given by the ITAE of the speed response of the Brushless DC Motor Drive with the tuning of the PI speed controller as shown Fig. 6.13.

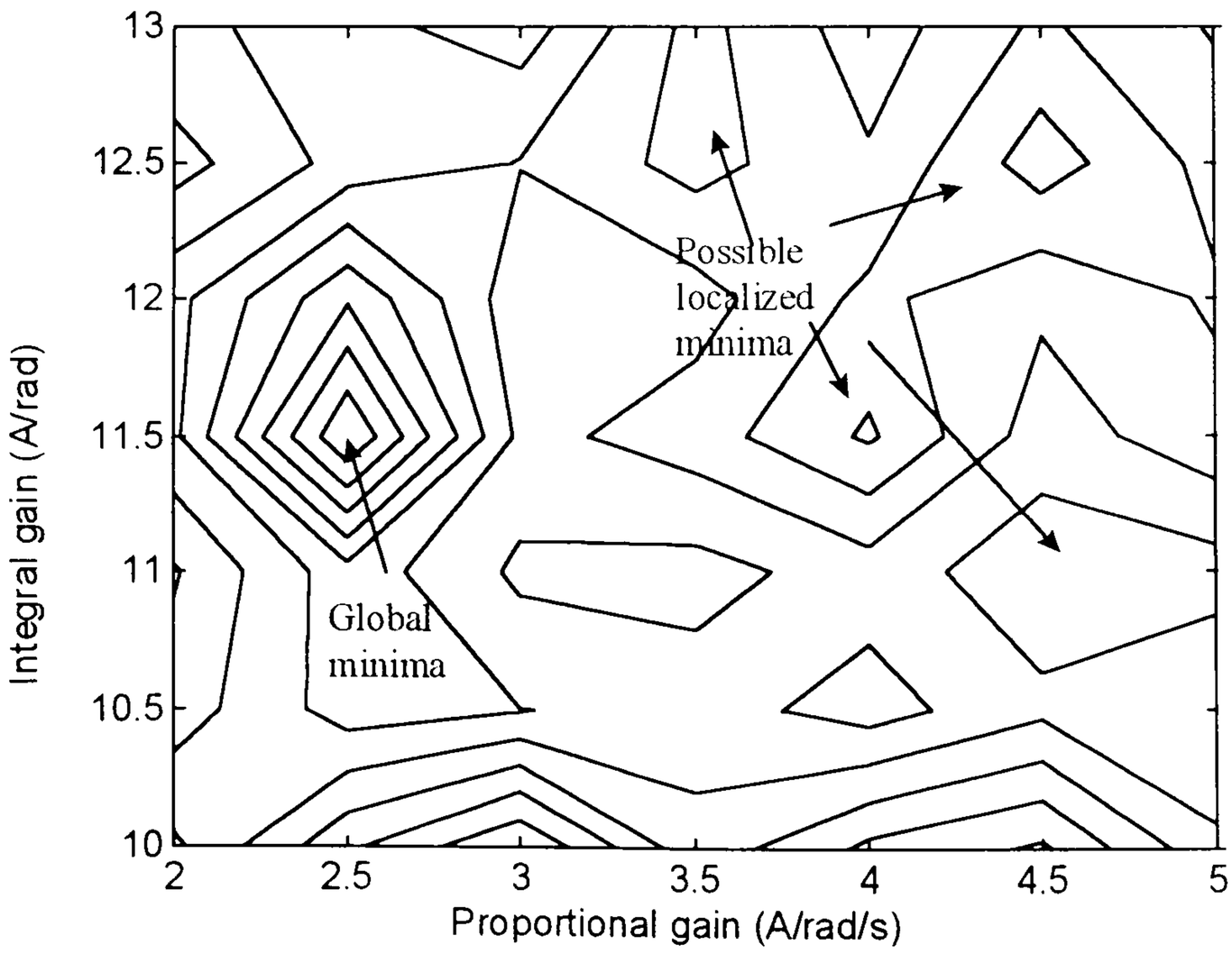



Fig. 6.15 - Contour map of Fig. 6.14, showing the existence of possible local minima.

When the proportional gain is too high, the speed signal becomes noisier and the ITAE of the speed response tends to increase. As the Genetic Algorithm looks for the best solution through all the searching space, it can find a better speed response whereas other methods could get stuck at any point in the searching space. Fig. 6.16 presents the profile of the surface obtained by rotating Fig. 6.14 by -90°. Again can be seen possible points of local minima in the searching space.

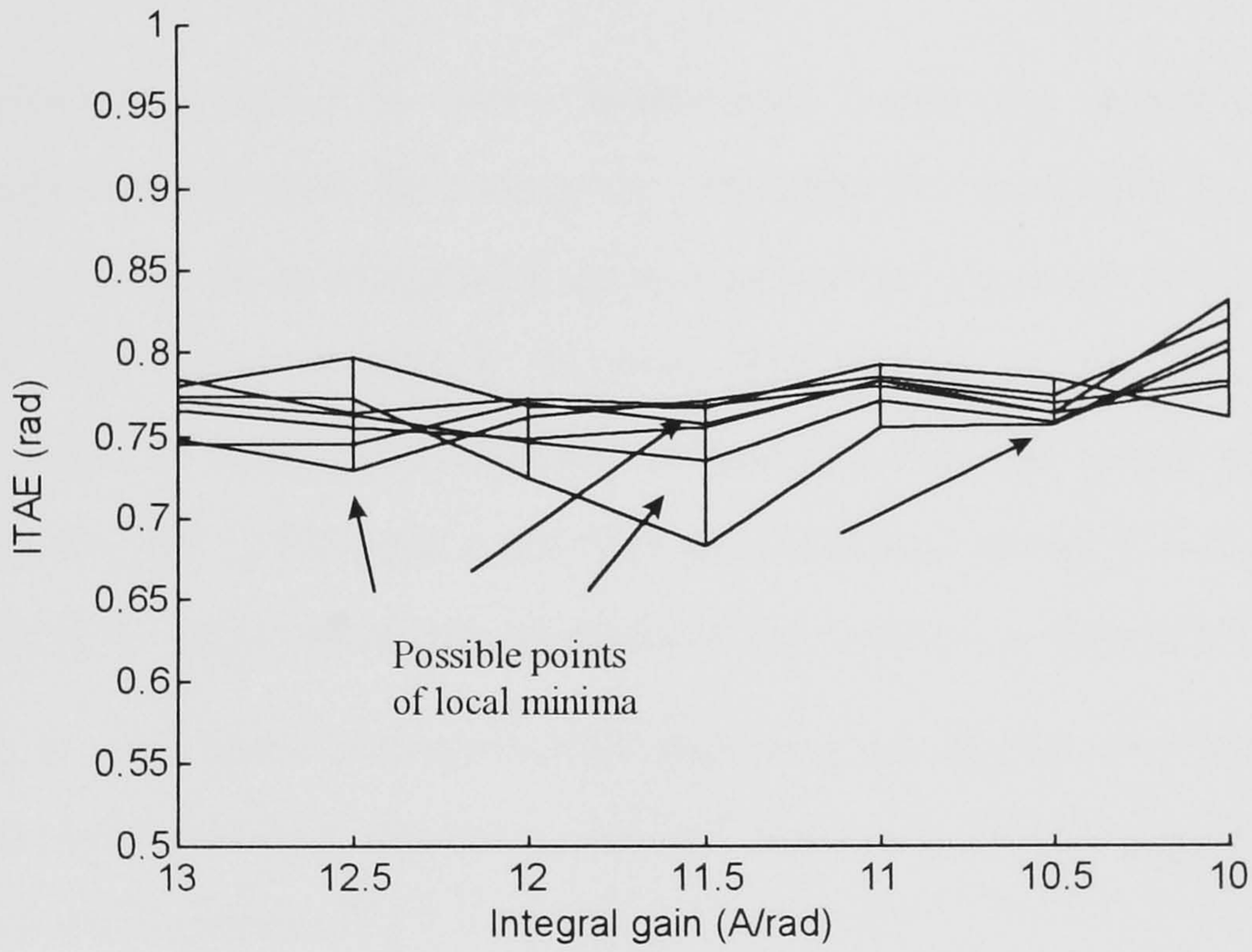



Fig. 6.16 - Profile of the surface shown in Fig. 14, showing possible points for local minima.

## 6.5 - CONCLUSION

In this chapter, the use of a load estimator in order to improve the speed holding capability of the Brushless DC Motor Drive has been investigated. It has been shown that the load estimator brings significant improvement to the speed response of the motor drive in the presence of load disturbance. The existence of possible points of local minima around the final setting given by the Genetic Algorithm has also been

investigated. The results have again shown that the problem of optimising the setting of a simple PI speed controller is not very simple. The existence of possible points of local minima in the searching space means that other optimisation methods could easily fail to find the best solution for the optimisation of the tuning of the PI speed controller whereas the Genetic Algorithm can be sucessful.

# CHAPTER 7

# ROBUSTNESS OF THE PI SPEED CONTROLLER AGAINST LOAD VARIATION AND SPEED DEMAND

## 7.1 - INTRODUCTION

As mentioned in chapter 2, many researchers round the world have developed different approach towards the best speed controller for motor drives. In many of the papers, the authors claim robustness against parameter variation for the speed control approaches they have proposed. Iwasaki [IWASAKI *et al*, 1991] has claimed robustness against variation of system parameters for the speed control approach discussed in chapter 6 (PI speed controller plus load estimator). However, some others are still asking for quantitative measurements for robustness degree of the controllers.

Attempting to give some measurements regarding the degree of robustness of the PI speed controller with load estimator, several tests have been done at different speed demand and load condition.

The motor was driven, using the tuning given by the Genetic Algorithm for the motor running at constant speed, in the presence of load disturbance, as discussed in section 6.3. Three different step input speed demands were applied to the motor, with different load torque, as follows: 700 rpm, 1100 rpm and 1500 rpm step input speed demand. For each speed demand, two different load conditions were used. In addition to this, for each reference speed, a test was done where the load torque changed during the running time.

The integral of the absolute value of the speed error (ITAE) mentioned earlier in chapter 5, is the parameter used to assess the quality of the speed response. The same treatment is given to the fuzzy speed controller, presented later in chapters 8 and 9.

## 7.2 - EXPERIMENTAL RESULTS

The following tests have been done to the Brushless DC Motor Drive with PI speed controller and load estimator. The proportional gain of the controller was 3.7 A/rad/s whereas the integral one was 12.37 A/rad. This setting was given by Genetic Algorithm during on-line optimisation process for the motor running at 1100 rpm constant speed when a load disturbance was applied, as discussed in section 6.3.

### 7.2.1 - 1100 rpm with 11% load torque

Fig. 7.1 presents the speed and current response of the motor drive for 1100 rpm step input reference speed, with 11% of the full load. There was a large speed overshoot due to the integrator *windup*, resulting from the torque saturation of the motor drive. When the motor speed goes past the reference value, negative current demand should be applied to the motor in order to bring the speed back towards the reference value. However, it is limited to the range between 0 A and 9 A for the reasons explained in chapter 3. As a result, the current drops from 9 A applied during the acceleration, to 0A during the braking. Nonetheless, when the speed of the motor settles at the demanded value, the current returns to its steady state value, which is approximately 1A. The maximum speed overshoot happens at approximately 0.3 s. The ITAE measured for this speed response within 5 s time window was 29.2 rad.

### 7.2.2 - 1100 rpm with 62% load torque

Fig. 7.2 shows the speed and current response of the motor for 1100 rpm, with 62% of the maximum torque the motor can produce. The speed overshoot is nearly the same as the one before, about 36%. Nevertheless, the acceleration until the maximum overshoot lasts a bit longer because of the larger load torque. The deceleration time is also a bit longer as the current does not drop to zero to pull down the speed. As a consequence, the ITAE within this condition was larger, 31.8 rad.

### 7.2.3 - 1100 rpm in the presence of load disturbance

In this condition, an 1100 rpm step input speed demand was applied to the motor with

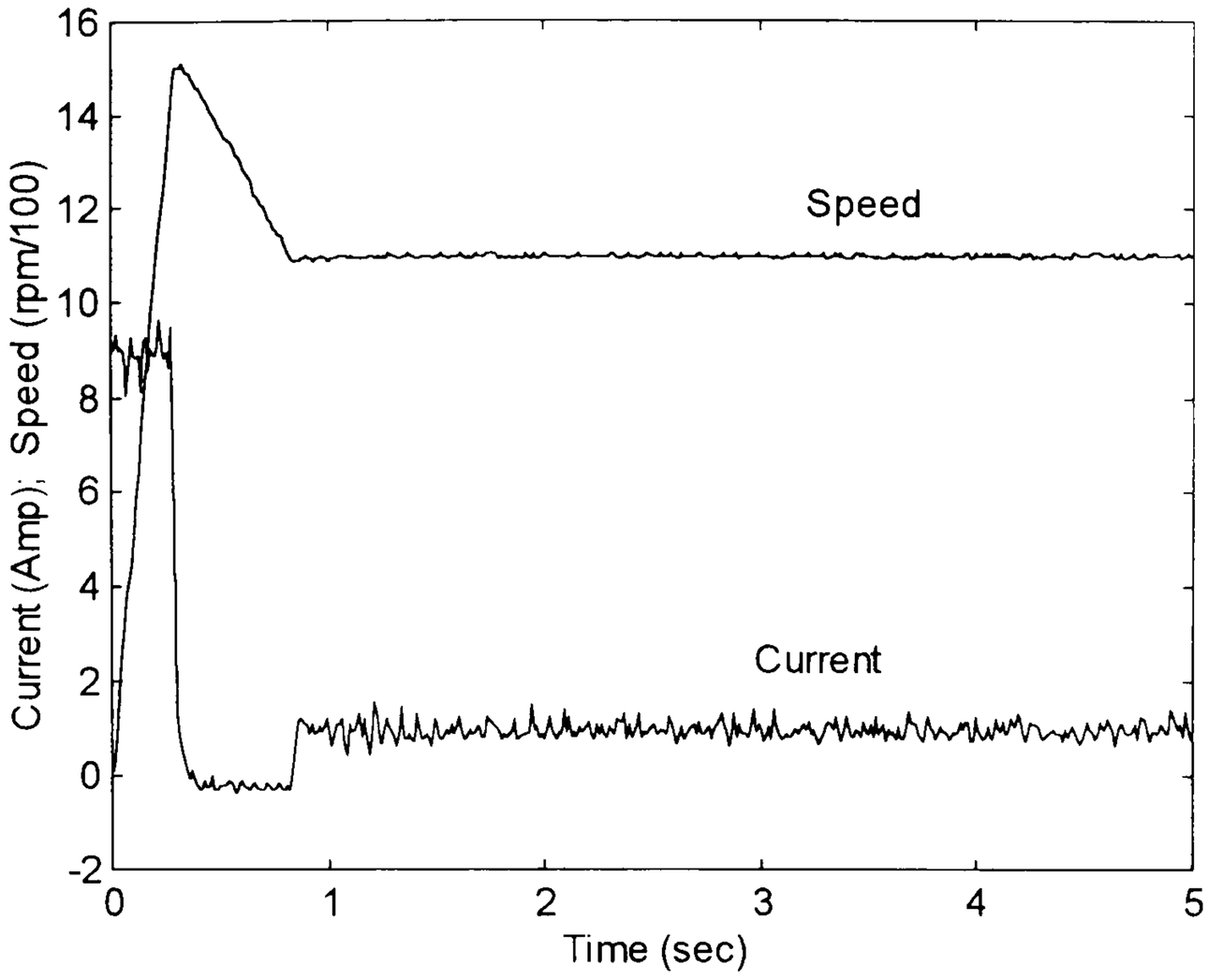



Fig. 7.1 - Speed and current response of the motor for 1100 rpm step input speed demand with load torque equal to 11% of the maximum torque produced by the motor.

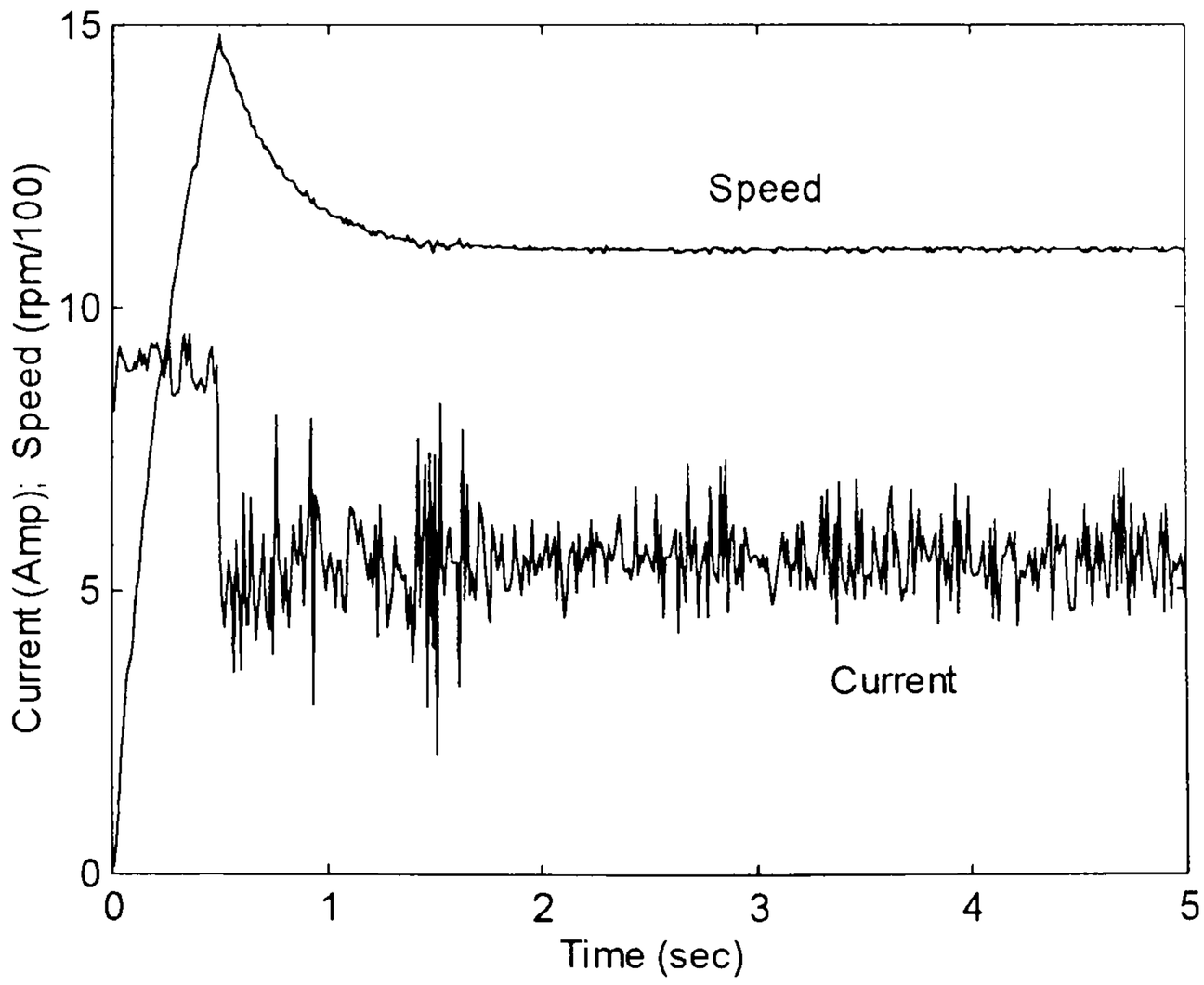



Fig. 7.2 - Speed and current response of the motor for 1100 rpm step input speed demand with load torque equal to 62% of the maximum torque produced by the motor.

11% of load torque. At *time* = 1 s, a step input load disturbance equal to 62% of the maximum load was applied, lasting for the rest of the 5 s time window. The speed and current response can be seen in Fig. 7.3. The ITAE was equal to 28.8 rad. The starting transient is the same as section 7.4.1, as the motor starts with the same load torque. The performance against the disturbance itself is the same as shown in Fig. 6.11-12, as the motor was already at steady state when the load step was applied.

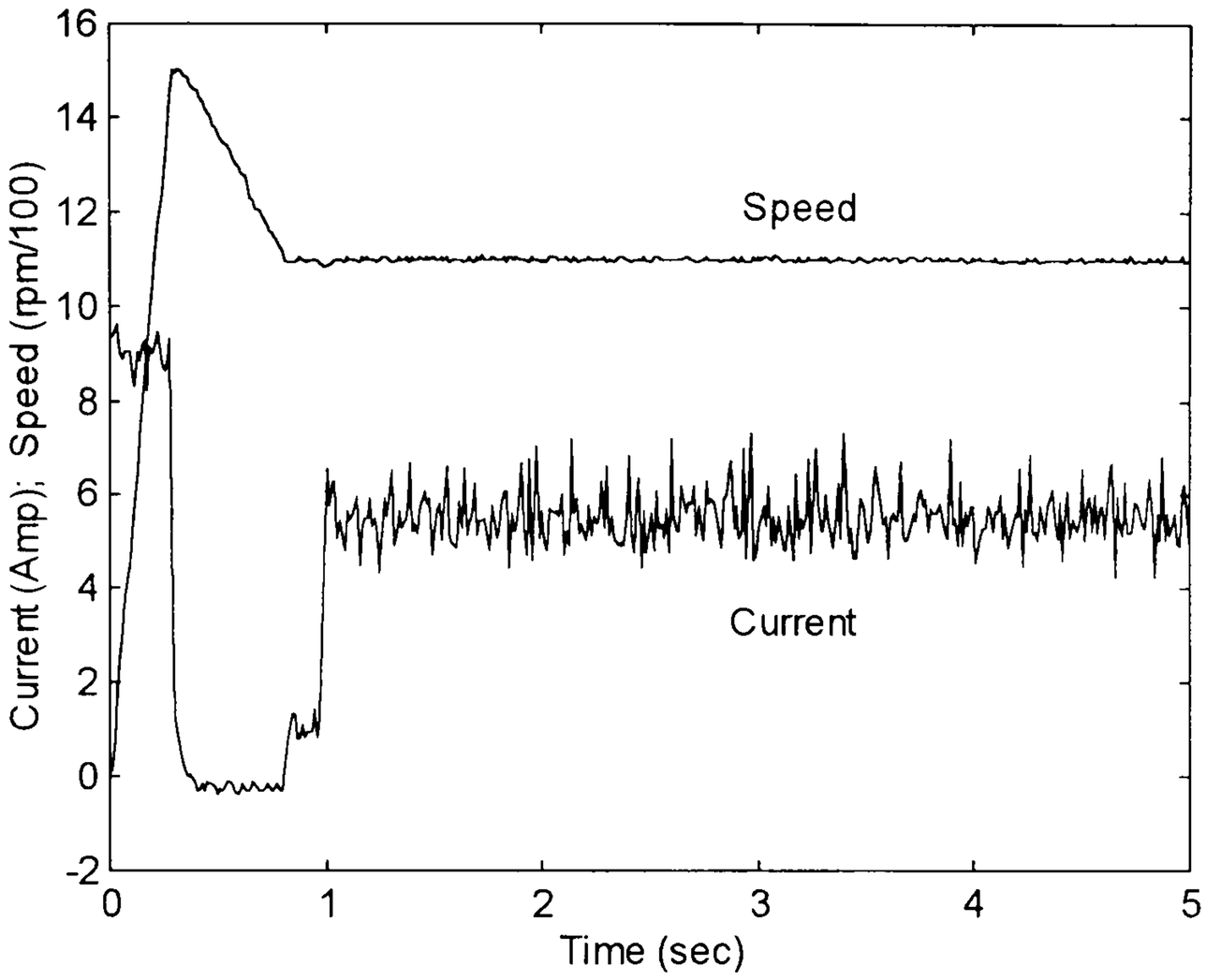



Fig. 7.3 - Speed and current response of the motor for 1100 rpm step input speed demand load as step input load disturbance equivalent to 62% of the full load, applied at $t$ = 1 s.

### 7.2.4 - 1500 rpm with 11% of maximum load torque

Fig. 7.4 depicts the speed and current response of the motor drive due to 1500 rpm step input speed demand, with 11% of the maximum load torque. The ITAE measured for this speed response within 5 s time window was 68.14 rad. The *windup* problem in this case is even more apparent as the step input speed demand is larger. The speed overshoot is 46.7%. As a consequence, it takes longer to settle.

### 7.2.5 - 1500 rpm with 75% load torque

Fig. 7.5 illustrates the speed and current response of the motor due to 1500 rpm step

input speed demand, with 75% of the maximum load torque. The shape of the speed response resembles that as in Fig. 7.2. However, because of the even larger load torque at 1500 rpm, the ITAE during 5 s time window, was 64.25 rad.

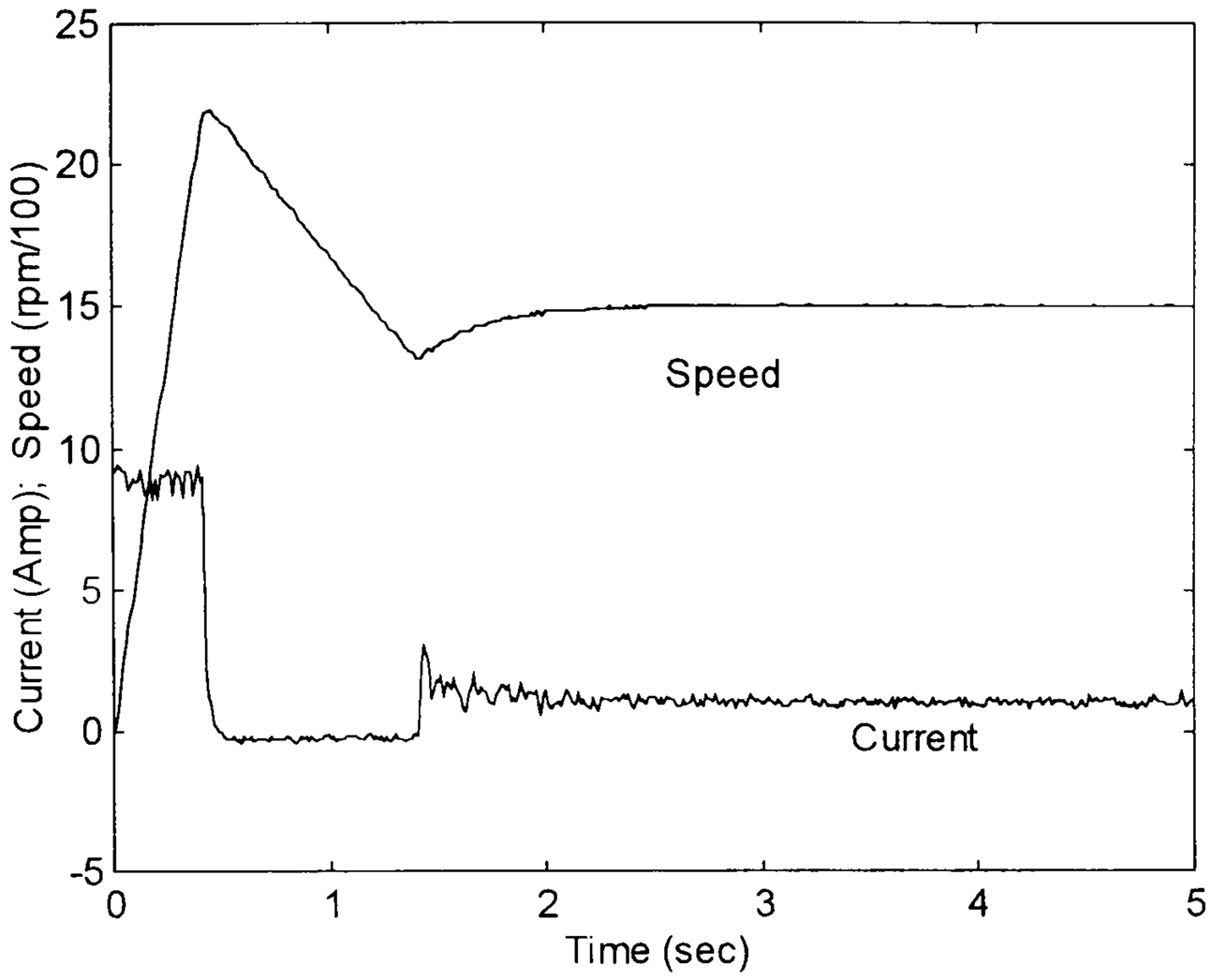



Fig. 7.4 - Speed and current response due to 1500 rpm step input speed demand with 11% of the full load torque.

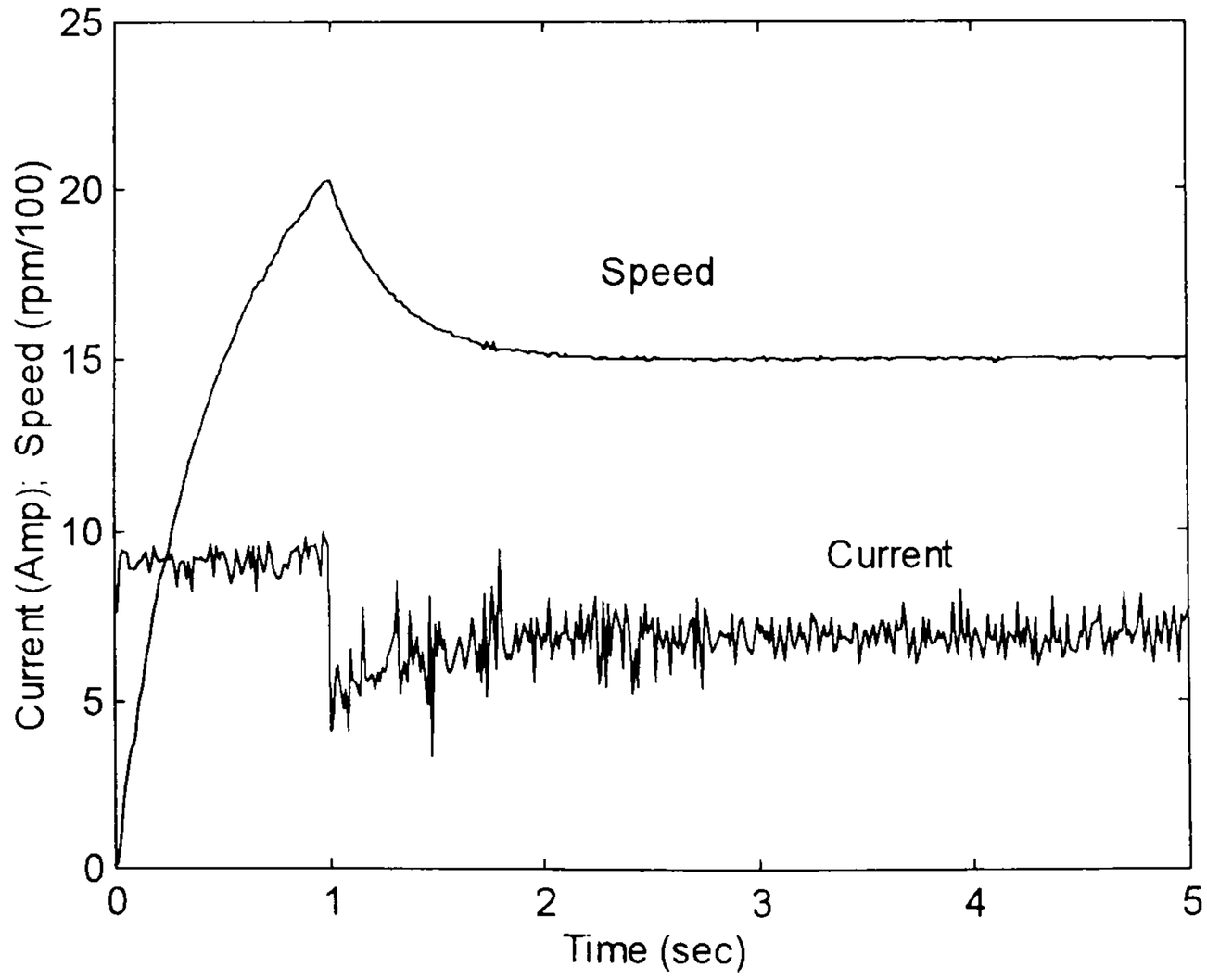



Fig. 7.5 - Speed and current response due to 1500 rpm step input speed demand with 75% of the full load torque.

### 7.2.6 - 1500 rpm in the presence of load disturbance

In this particular condition, a 1500 rpm step input speed demand was applied to the motor with 11% of load torque. At *time* = 1 s, a step input load disturbance equal to 75% of the maximum load was applied, lasting for the rest of the 5 s time window. The speed and current response can be seen in Fig. 7.6. The transient during the deceleration is different from that seen in Fig. 7.4. The effect of the load change during the braking of the speed is apparent. At *time* = 1 s, when a large load step was applied, the motor speed was still being pulled down with the aid of the initial load. As a consequence, there was an increase on the deceleration rate, forcing the speed go below the reference value. The ITAE within 5 s time window was 72.35 rad.

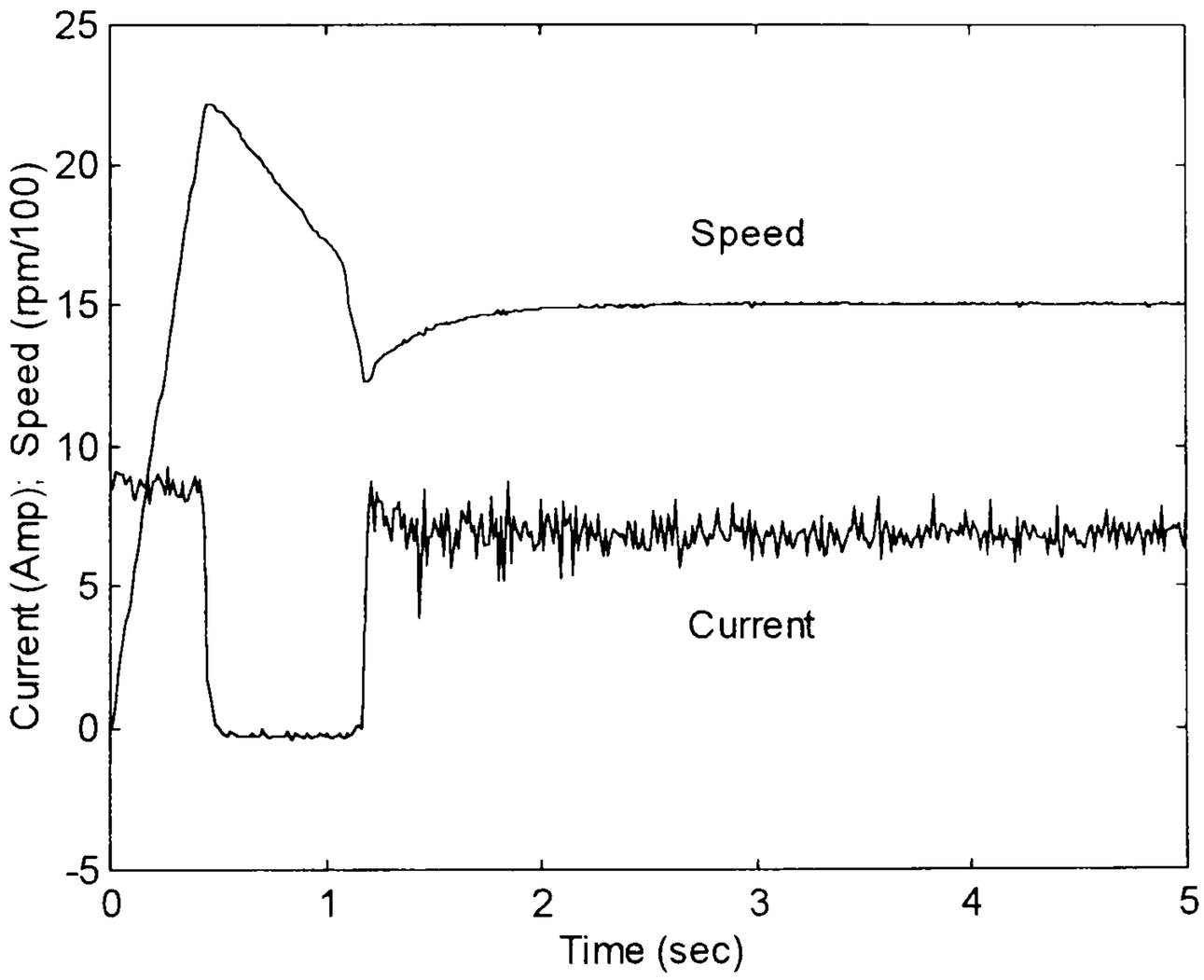



Fig. 7.6 - Speed and current response of the motor for 1500 rpm step input speed demand load as step input load disturbance equivalent to 75% of the full load, applied at $t$ = 1 s.

Following, the same experiment previously done for 1100 rpm and 1500 rpm step input speed demand, is done for lower reference speed, 700 rpm. The *windup* problem is expected to be less significant because of the smaller step input speed demand. The results are shown as follow.

### 7.2.7 - 700 rpm with 11% load torque

Fig. 7.7 presents the speed and current response of the motor due to 700 rpm step input speed demand, with of 11% of the full load torque. The speed overshoot is now approximately 32%, against 36% for 1100 rpm and 46.7% for 1500 rpm. This is in accordance to what was expected, resulting from less integrator *windup* as the speed controller stays saturated for shorter period of time. The ITAE for this speed response is 14.93 rad.

### 7.2.8 - 700 rpm with 55% load torque

Fig. 7.8 demonstrates the speed and current response of the motor due to 700 rpm step input speed demand, with 55% of the maximum load torque. The ITAE within this condition, during 5 s time window, was 12.70 rad, slightly smaller then in section 7.2.7 (14.93rad). This has happened mostly because of the smaller speed overshoot, 24% against 32% shown in the previous section.

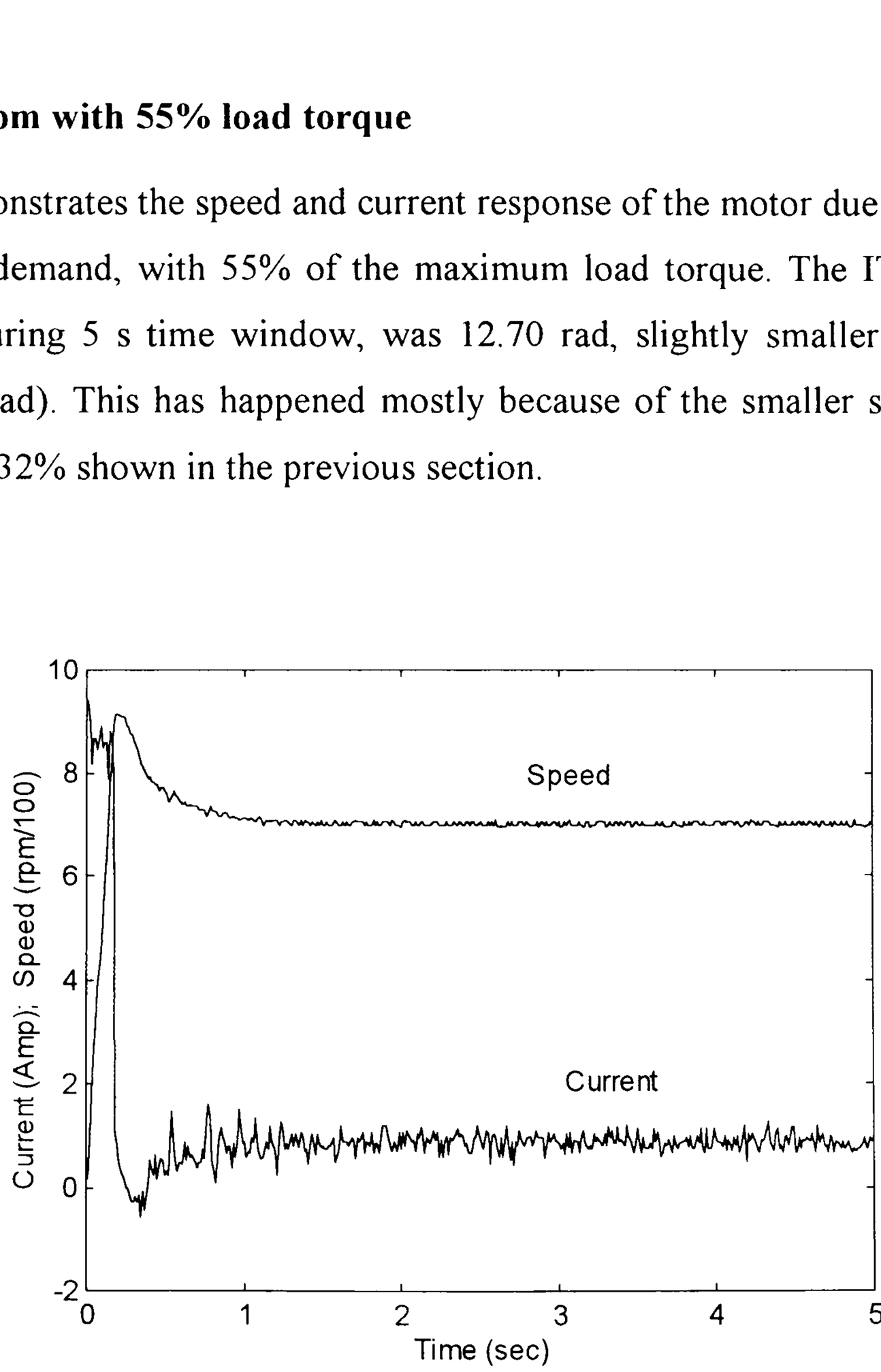



Fig. 7.7 - Speed and current response due to 700 rpm step input speed demand with 11% of the full load torque.

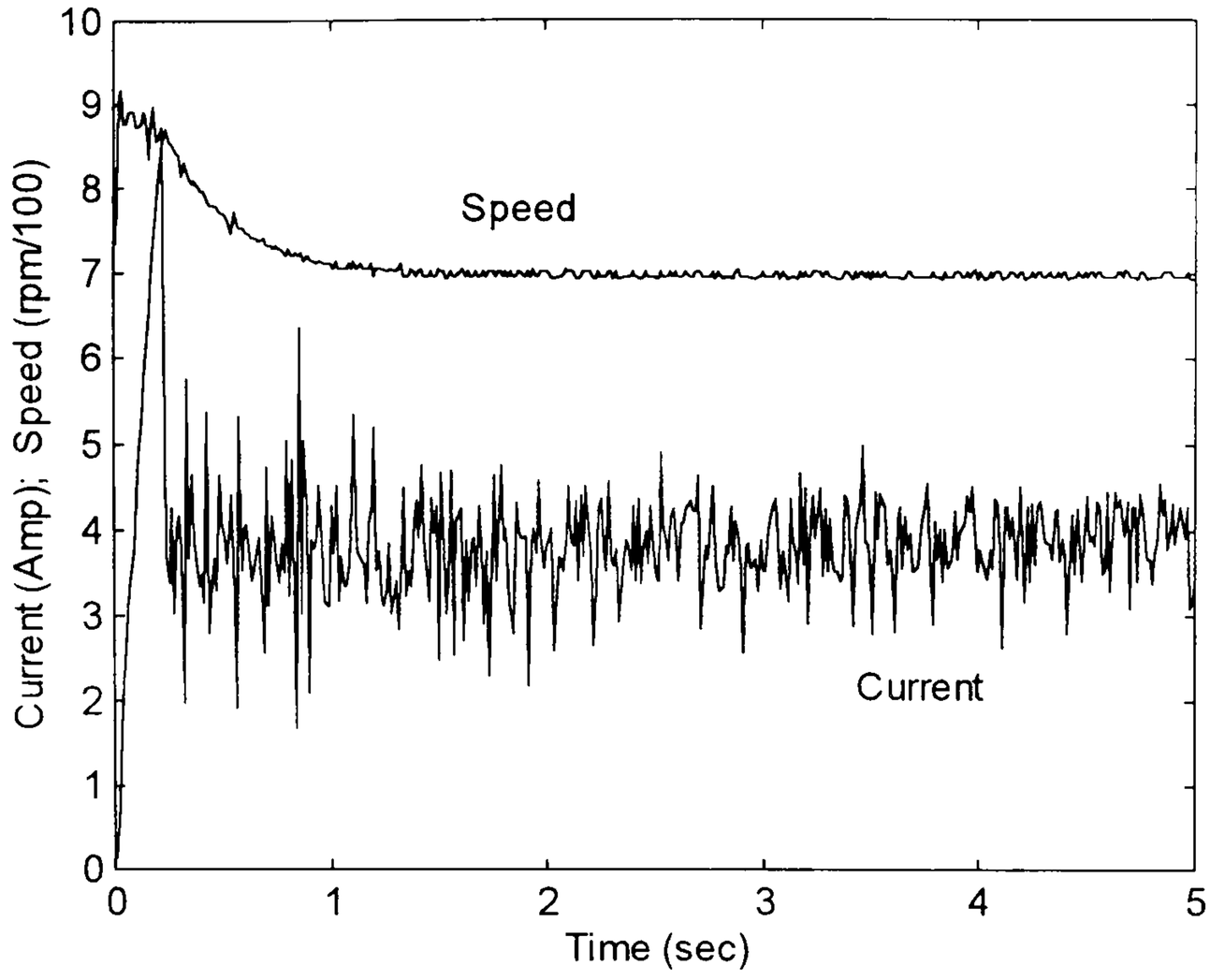



Fig. 7.8 - Speed and current response due to 700 rpm step input speed demand with 55% of the full load torque.

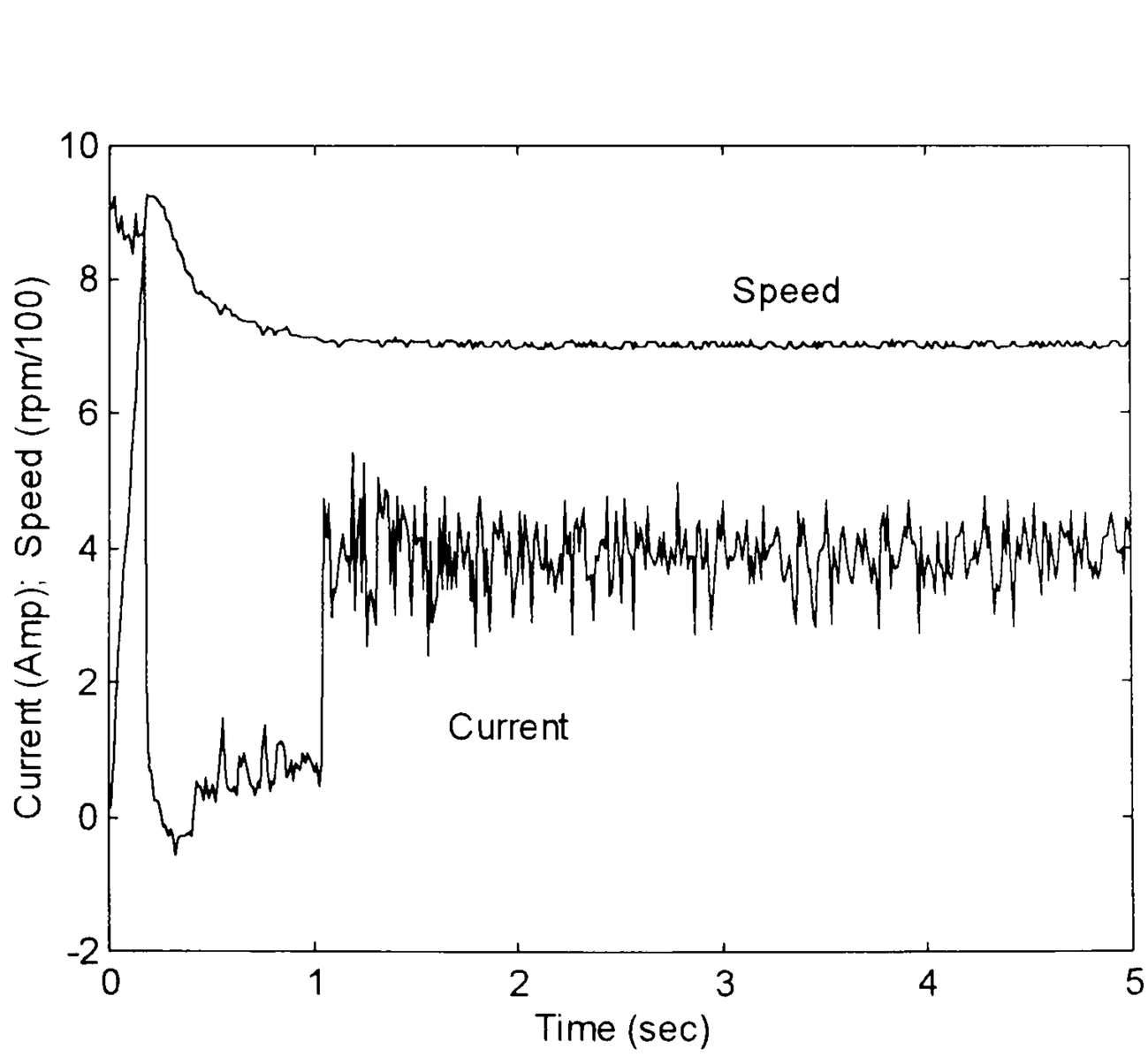



Fig. 7.9 - Speed and current response of the motor for 700 rpm step input speed demand load as step input load disturbance equivalent to 55% of the full load, applied at $t$ = 1 s.

### 7.2.9 - 700 rpm in the presence of load disturbance

In the last test, a 700 rpm step input speed demand was applied to the motor with 11% of load torque. At *time* = 1 s, a step input load disturbance equal to 55% of the maximum load was applied, lasting for the rest of the 5 s time window. The speed and current response can be seen in Fig. 7.9. The speed response is nearly identical to that shown in Fig. 7.7, as most of the speed transient takes place with the same load torque. When the disturbance was applied, the motor speed was nearly settled to its steady state value. The ITAE within 5 s time window was 15.57 rad.

### 7.2.10 - Summary of the results on robustness against load variation

In order to give a clear understanding of robustness of the PI speed controlled Brushless DC Motor Drive with load estimator, several tests have been done for different speed demand and load torque.

| STEP INPUT SPEED DEMAND | LOAD CONDITION (% of Full Load) | ITAE OF THE SPEED RESPONSE (rad) |
|---|---|---|
| **700 rpm** | 11% | 14.9 |
| | 55% | 12.7 |
| | Load change to 55% | 15.6 |
| **1100 rpm** | 11% | 29.2 |
| | 62% | 31.8 |
| | Load change to 62% | 28.8 |
| **1500 rpm** | 11% | 46.7 |
| | 75% | 64.2 |
| | Load change to 62% | 72.3 |

Table 7.1 - Summary of the quality of the speed response at different condition.

The tuning of the controller was given by the Genetic Algorithm in the on-line optimisation for best performance against load variation, at 1100 rpm, as discussed in section 6.3. Speed demand above (1500 rpm) and below (700 rpm) that to which the drive has been tuned, have been used with different load torque. A summary of these results is displayed in Table 7.1. It is clear that, because of the high current limitation, the controller loses performance because of the torque saturation. These results are compared to those obtained by using a fuzzy controller, presented later in Chapter 9.

Looking at the table above, the quality of the speed responses for 700 rpm in terms of ITAE has varied between 12.7 rad and 15.5 rad. For 1100 rpm, between 28.8 and 31.8 rad whereas for 1500 rpm, between 46.7 rad and 72.3 rad. These numbers mean that the quality of the speed response deteriorates more for higher values of step change in speed. It is understandable as, the higher the step in speed demand the larger the impact of the torque saturation on the controller, resulting in larger integrator *windup*.

## 7.3 - CONCLUSION

The robustness against load variation and changing in step input speed demand of the PI speed controlled Brushless DC Motor Drive has been investigated in this chapter. After setting the parameters of the speed controller for one condition (as in section 6.3), they were kept unaltered. Several tests have been done at different reference speeds and load torques. It has been shown that in practical application, the controller performance degrades significantly due to non-linearities of the drive system, specially concerning to the torque saturation. The integrator *windup* problem is relevant and happens whenever the controller saturates. For a given setting of the speed controller, the higher the step input speed demand the bigger the *windup* and the worse the quality of the speed response.

# CHAPTER 8

# SINGLE INPUT SINGLE OUTPUT - SISO FUZZY CONTROLLER

## 8.1 - INTRODUCTION

In this chapter, a Single-Input Single-Output (SISO) fuzzy speed controller is investigated. The controller was created in order to replace the classical Proportional Integral (PI) speed controller used previously. Similarly to the PI, the fuzzy controller discussed here has a single input, which is the speed error and a single output which is the current demand for the PI current controller.

## 8.2 - PRINCIPLES OF FUZZY LOGIC

### 8.2.1 - Introduction to fuzzy logic

Fuzzy logic [GULLEY and ROGER, The Math Works Inc., 1995 BOSE, 1997] is an area of artificial intelligence that deals with uncertainty and imprecision, where a statement or object does not need necessarily to be exclusively true or false. In fuzzy logic a particular statement or object has a degree of membership in a given set, between zero (completely excluded) and one (completely included). There is no need of mathematical models to deal with a problem, though skill is needed to create a fuzzy logic controller. Input and output variables of the controller have to be created and their range of possible value (known as the 'universe of discourse') divided into fuzzy sets by using membership functions. Rules are then used to map the input into the output. Hence, depending on the input, the fuzzy controller gives an output according to both the membership functions and rules. It is utilised here on the speed control of a Brushless DC Motor Drive.

The input of the controller is the speed error, obtained by subtracting the actual speed from the reference value. The output is current demand for a PI current controller used in cascade configuration, as shown in Fig. 8.1.

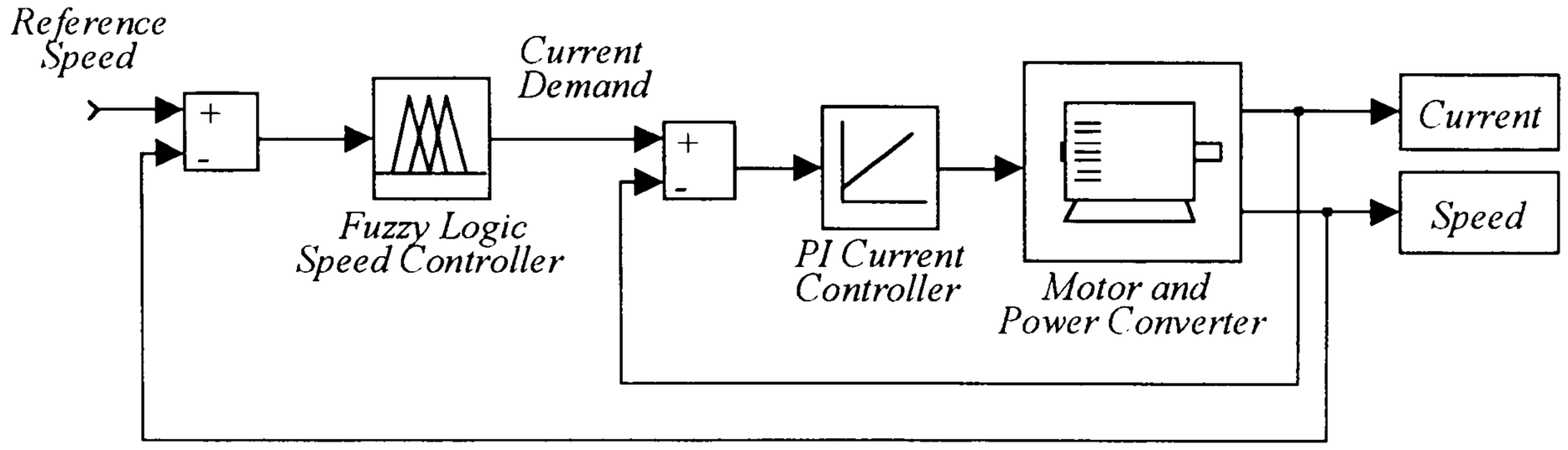



Fig. 8.1 – Cascade control of an electric drive (Fuzzy / PI).

### 8.2.2 - Membership functions, rules and transfer functions

The first task is to define the input membership functions. For initial simplicity just two trapezoidal membership functions are used, as illustrated in Fig. 8.2. A similar set of two trapezoidal membership functions is used for the output variable, which in this case is the current demand for the inner control loop. In order to highlight the influence of the membership functions in this fuzzy controller, the ranges of the input and output variables have been normalised to -100% to +100%.

Two rules are used to map the input membership functions into the output membership functions as follows:

RULE 1 - **If** input is positive **then** output is positive;

RULE 2 - **If** input is negative **then** output is negative.

By creating the fuzzy controller as described above and using the centroid defuzzification method [GULLEY and ROGER, The Math Works Inc., 1995 BOSE,1997] to derive the unique output value according to the degree of membership of its two output functions, the transfer function shown in Fig. 8.3 is obtained.

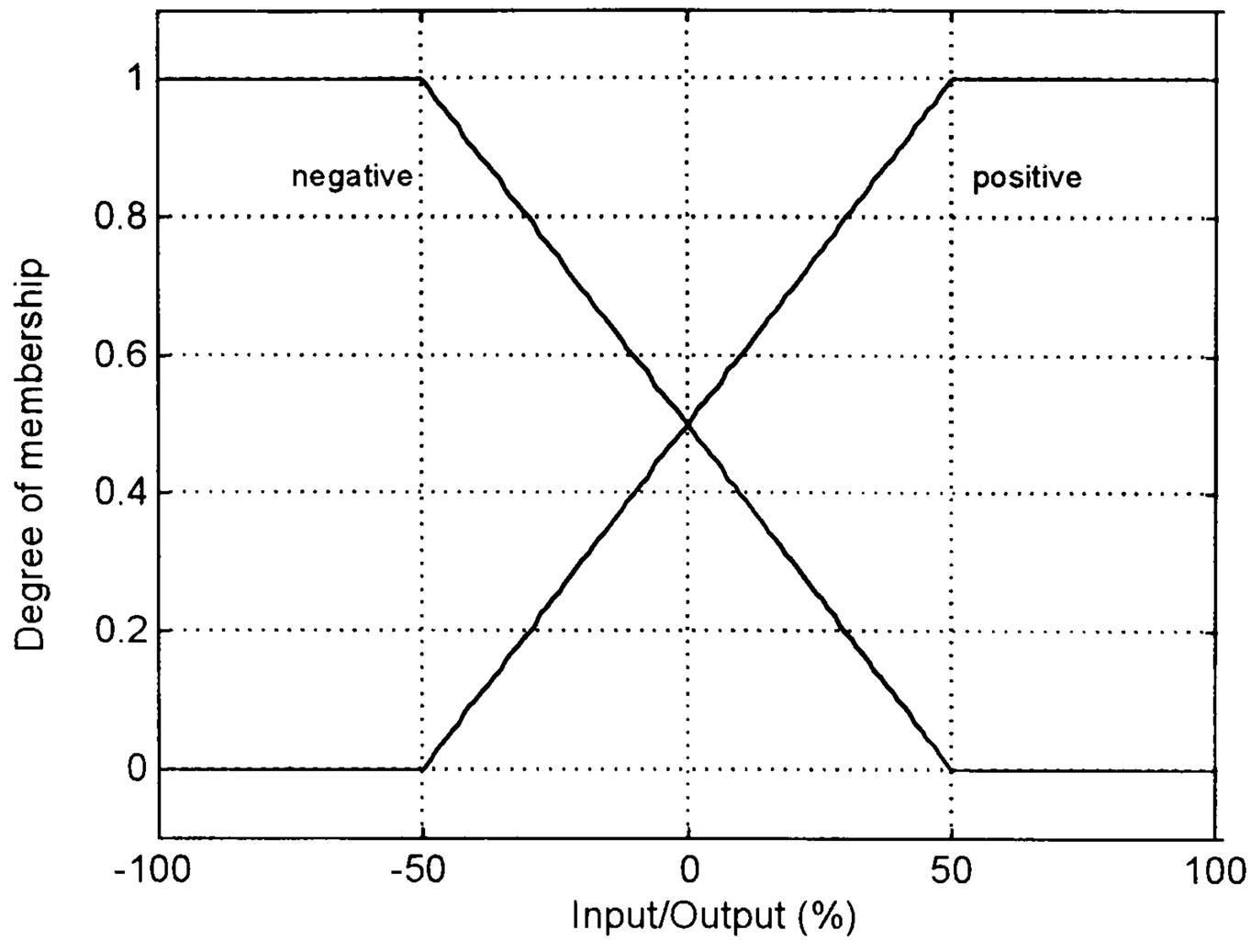



Fig. 8.2 - Input and output trapezoidal membership functions for the fuzzy speed controller.

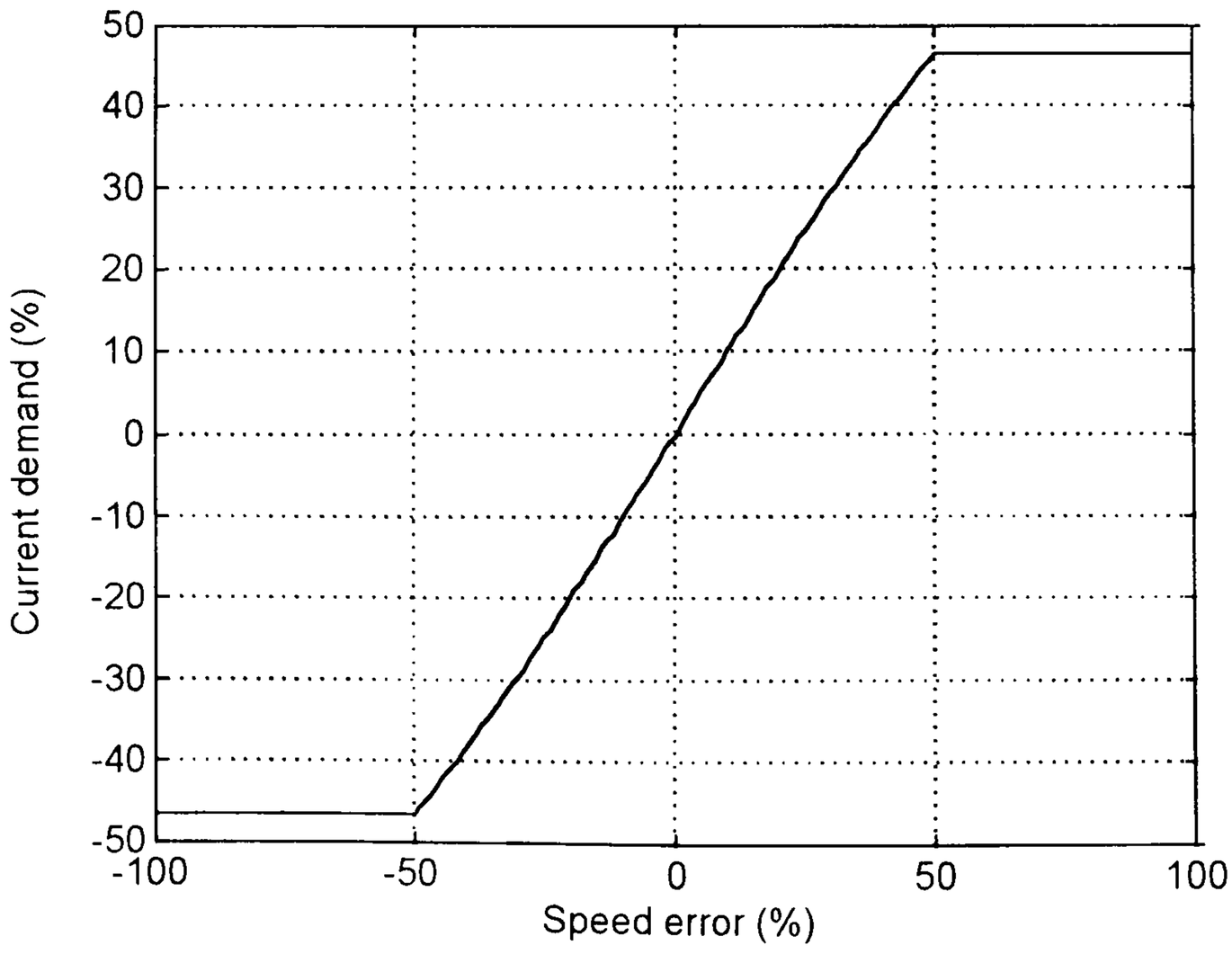



Fig. 8.3 - Transfer function of the single-input, single-output fuzzy speed controller with two membership functions for the input and output variables.

The transfer function is non-linear and its shape and range can be easily changed by relocating the membership functions, modifying their shape or adding membership functions and rules to the fuzzy controller, as illustrated in the following Section.

### 8.2.3 - Influence of the input membership functions on transfer functions

The effect of changing the shape of the input membership function is illustrated in Figs. 8.4, 8.5, 8.6 and 8.7. In each Figure the two input membership functions (Figs. 8.4 and 8.6) have the same trapezoidal shape as the original input membership functions shown in Fig. 8.2. However the sloping section of the two transfer functions is made steeper, so that the range of speed error corresponding to partial membership is reduced. As a consequence, the corresponding transfer function of the fuzzy logic controller changes as shown in Figs. 8.5 and 8.7.

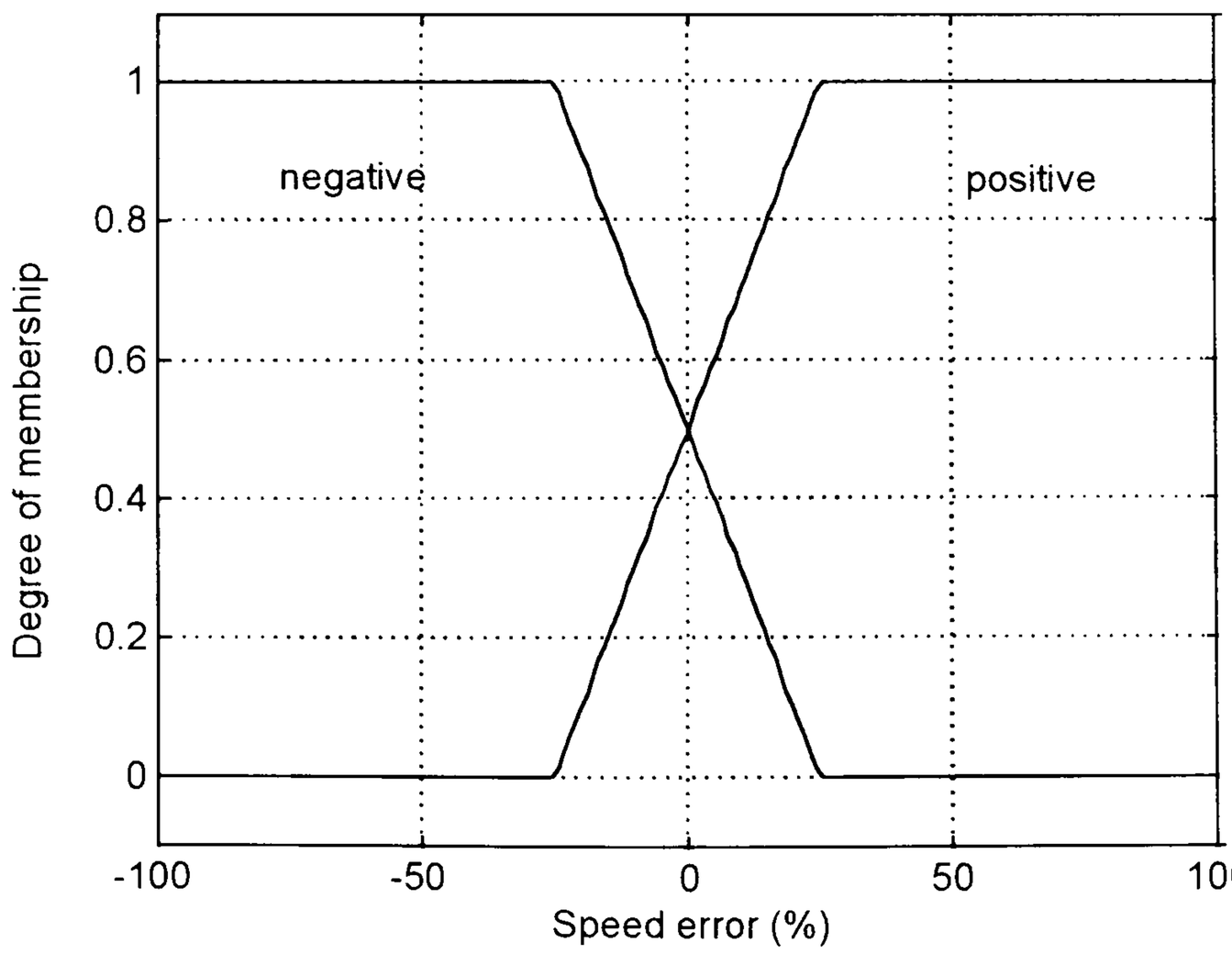



Fig. 8.4 - Speed error input membership functions.

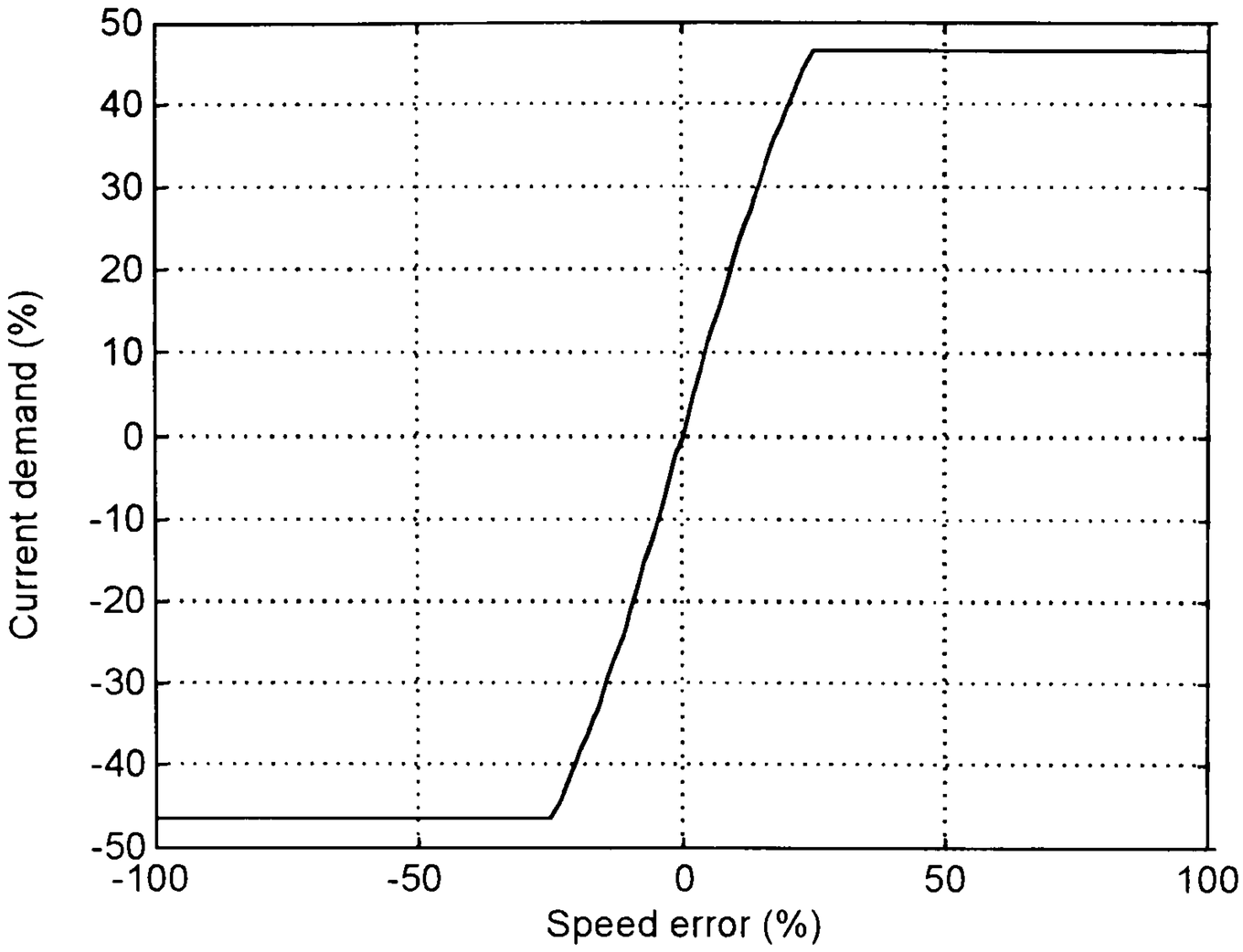



Fig. 8.5 - Transfer function of the single-input single-output fuzzy speed controller with the input membership functions shown in Fig. 8.4.

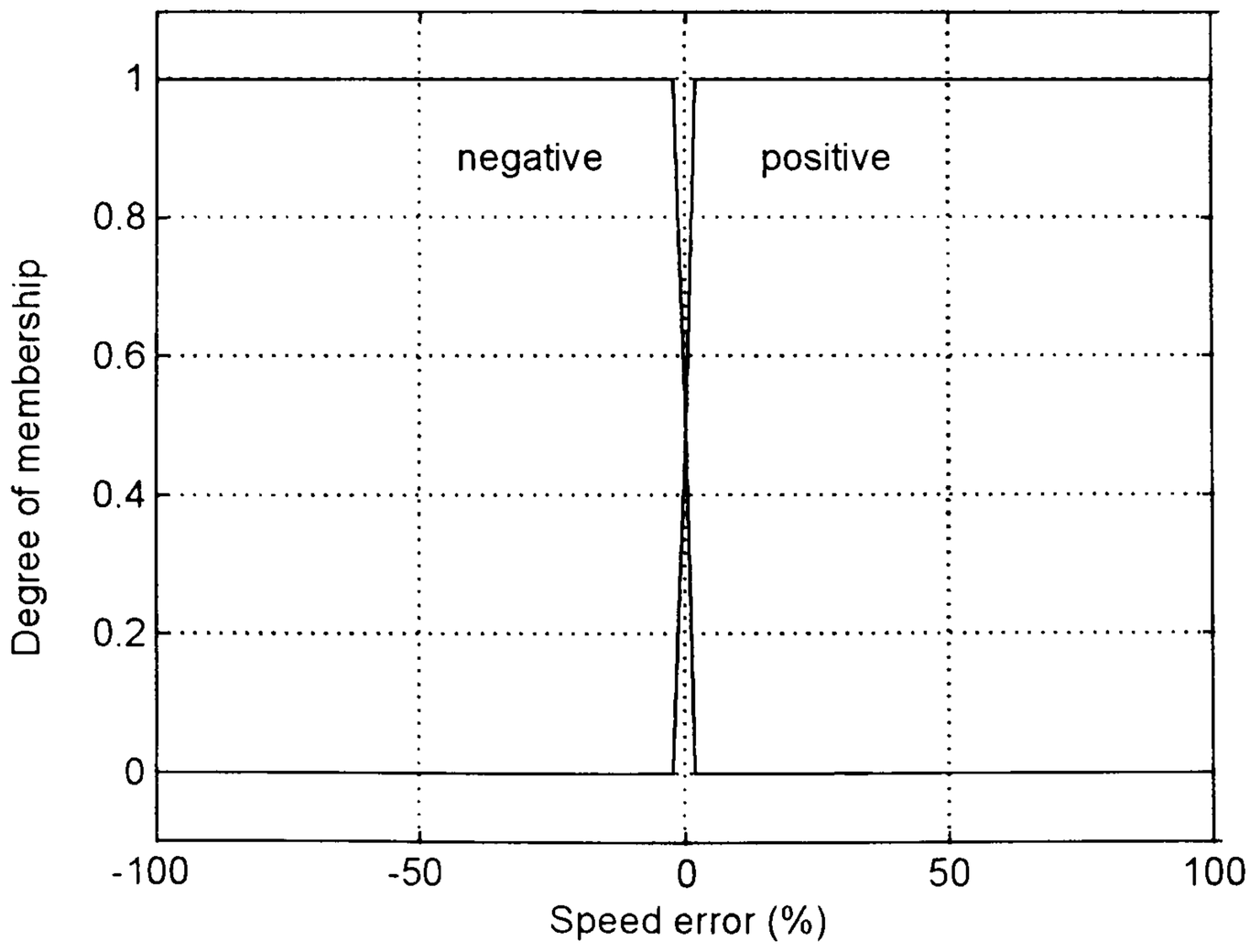



Fig. 8.6 - Speed error input membership functions.

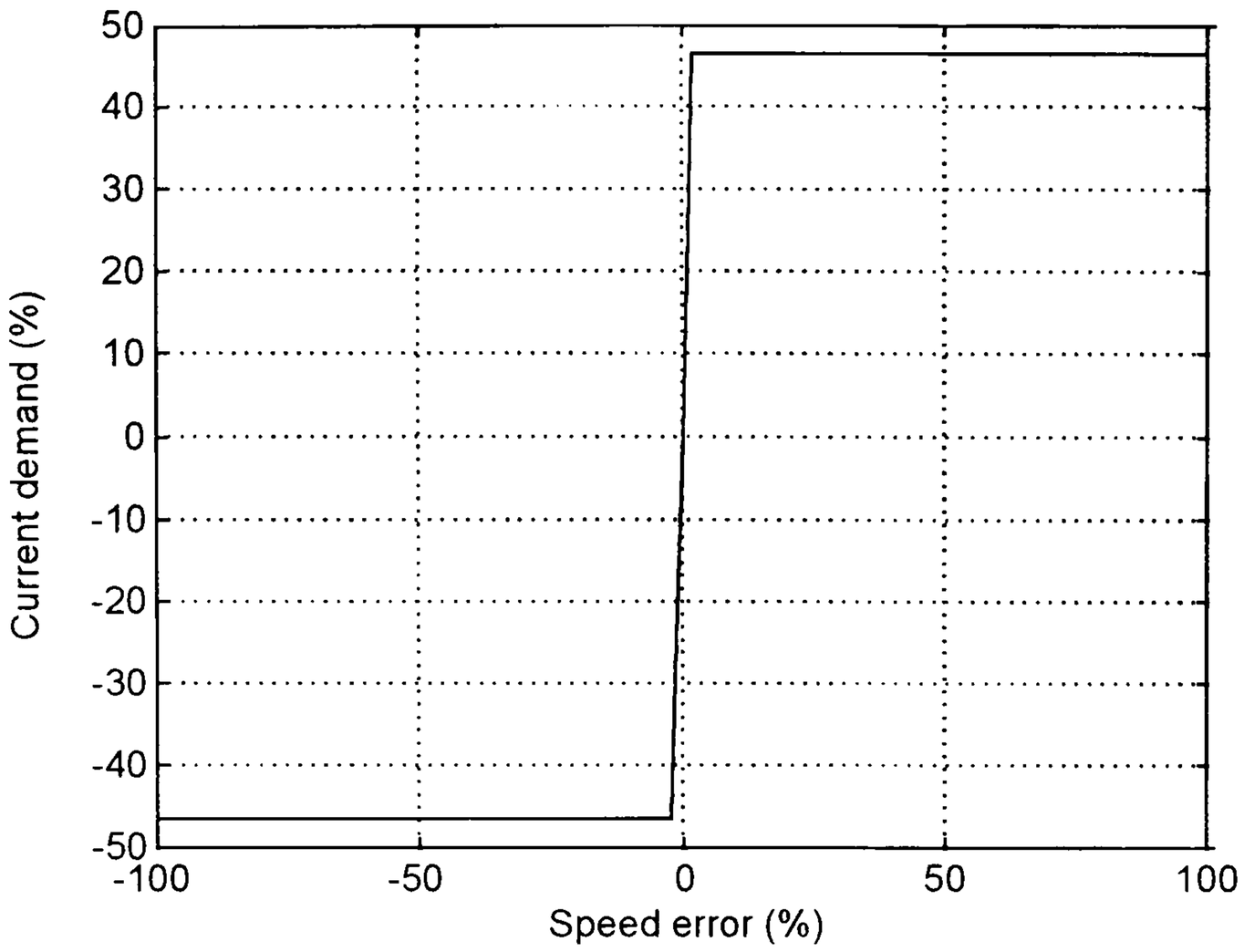



Fig. 8.7 - Transfer function of the single-input single-output fuzzy speed controller with the input membership functions shown in Fig. 8.6.

A comparison of the transfer functions in Figs. 8.3, 8.5 and 8.7 reveals that the output (current demand) saturation is the same in all three cases, but the gain for speed errors around zero becomes progressively larger as the sloping section of each membership function is made steeper.

### 8.2.4 - Influence of the output membership functions on transfer functions

The effect on the transfer function of changing the output membership functions is illustrated in Figs. 8.8, 8.9, 8.10 and 8.11. In both cases the input membership function has the form shown in Fig. 8.2. From a comparison of Figs. 8.9 and 8.11, it is apparent that the only change in the fuzzy speed controller transfer function relates to the output (current demand) level, which approaches closer to its limiting value of 50%, as the output membership function slope is increased.

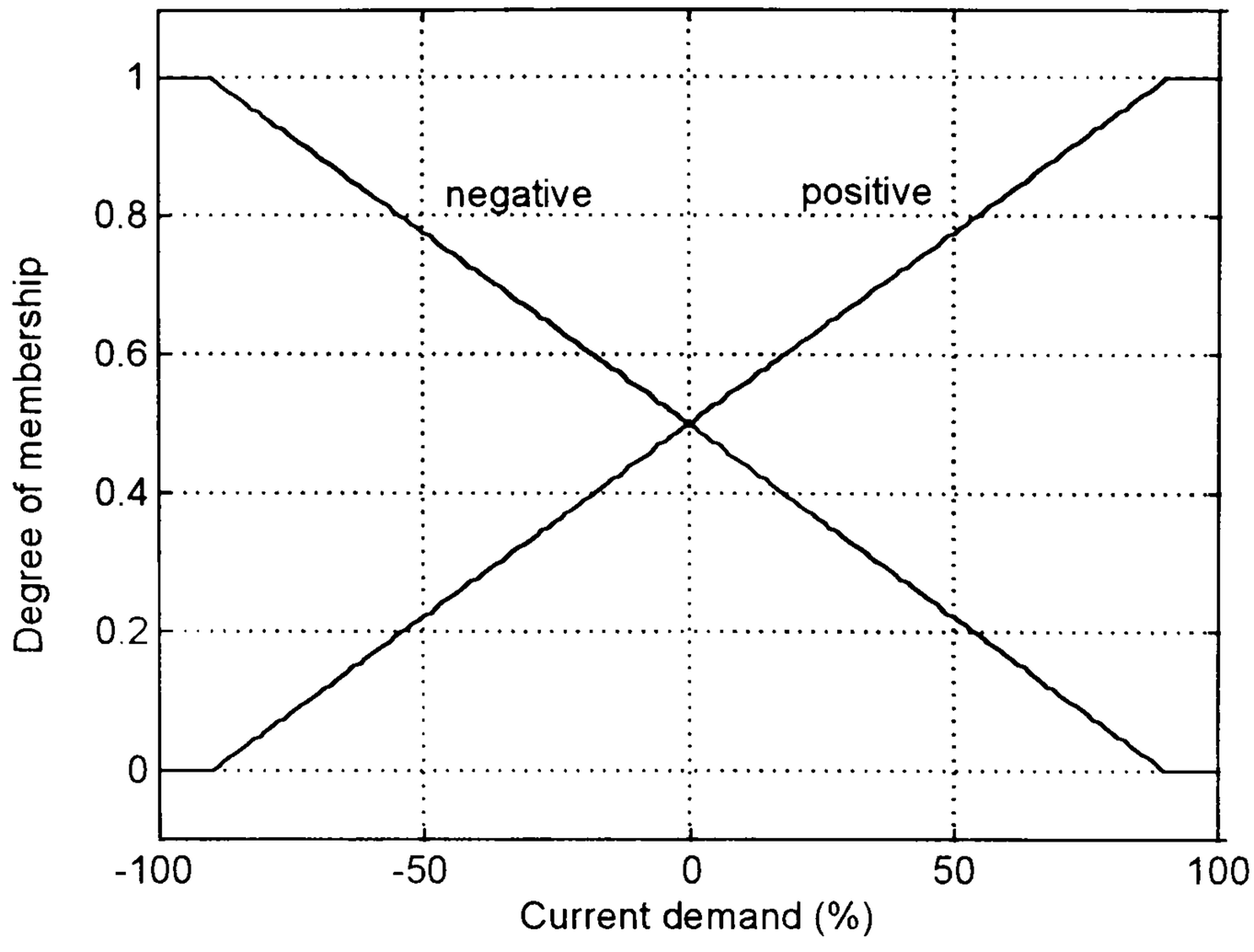



Fig. 8.8 - Output membership functions.

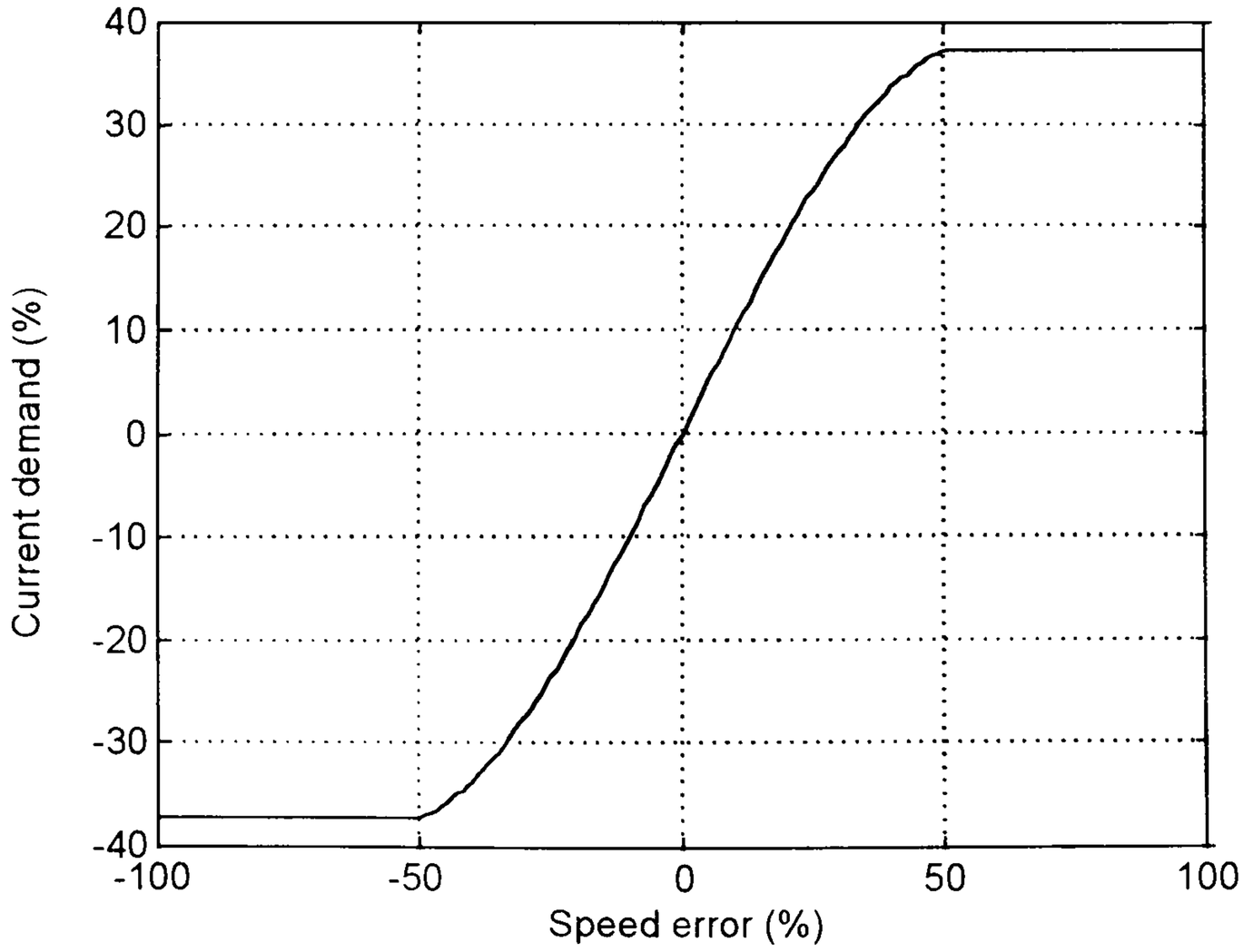



Fig. 8.9 - Transfer function of the single-input single-output fuzzy speed controller with the output membership functions shown in Fig. 8.8.

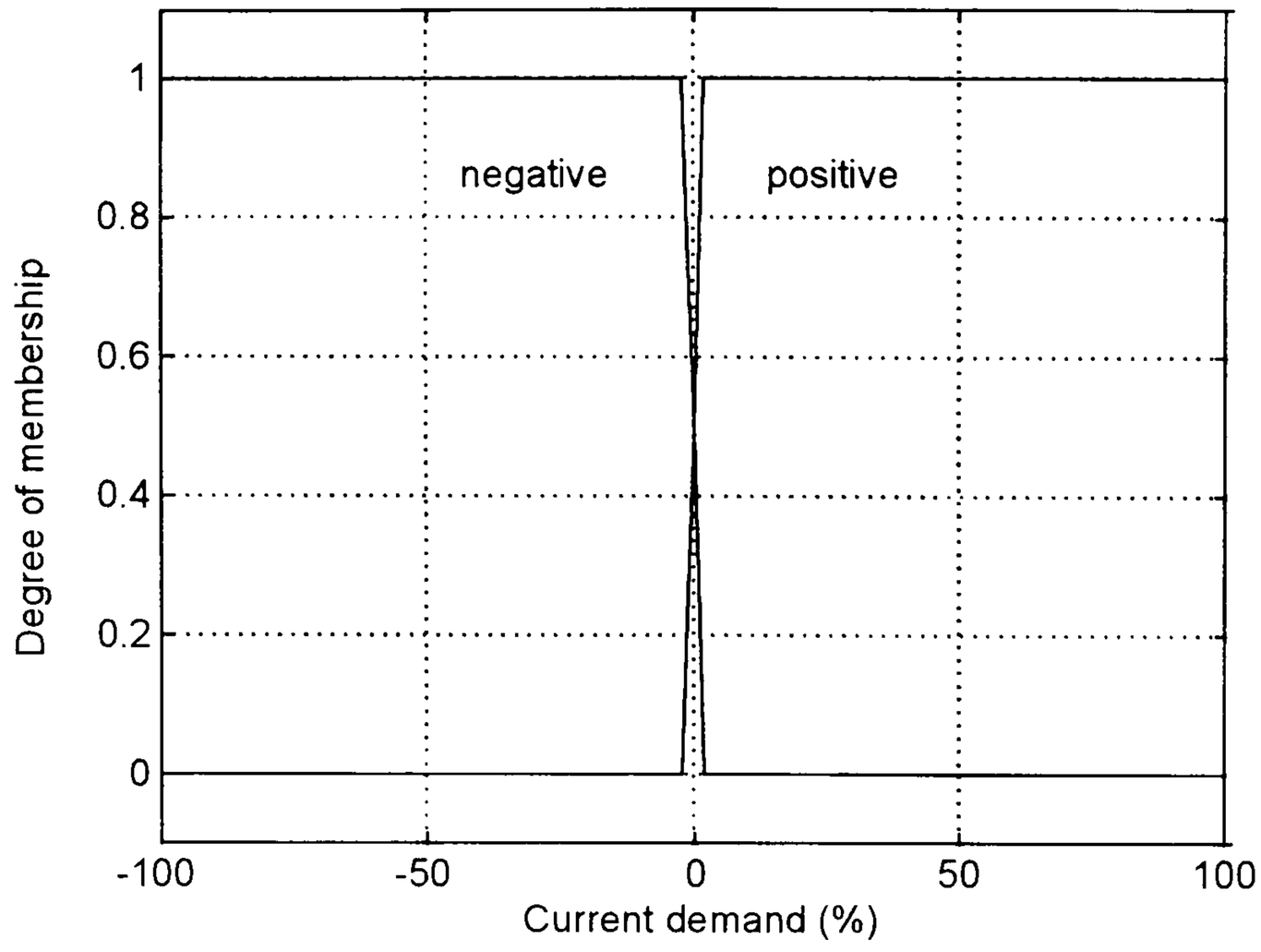



Fig. 8.10 - Output membership functions.

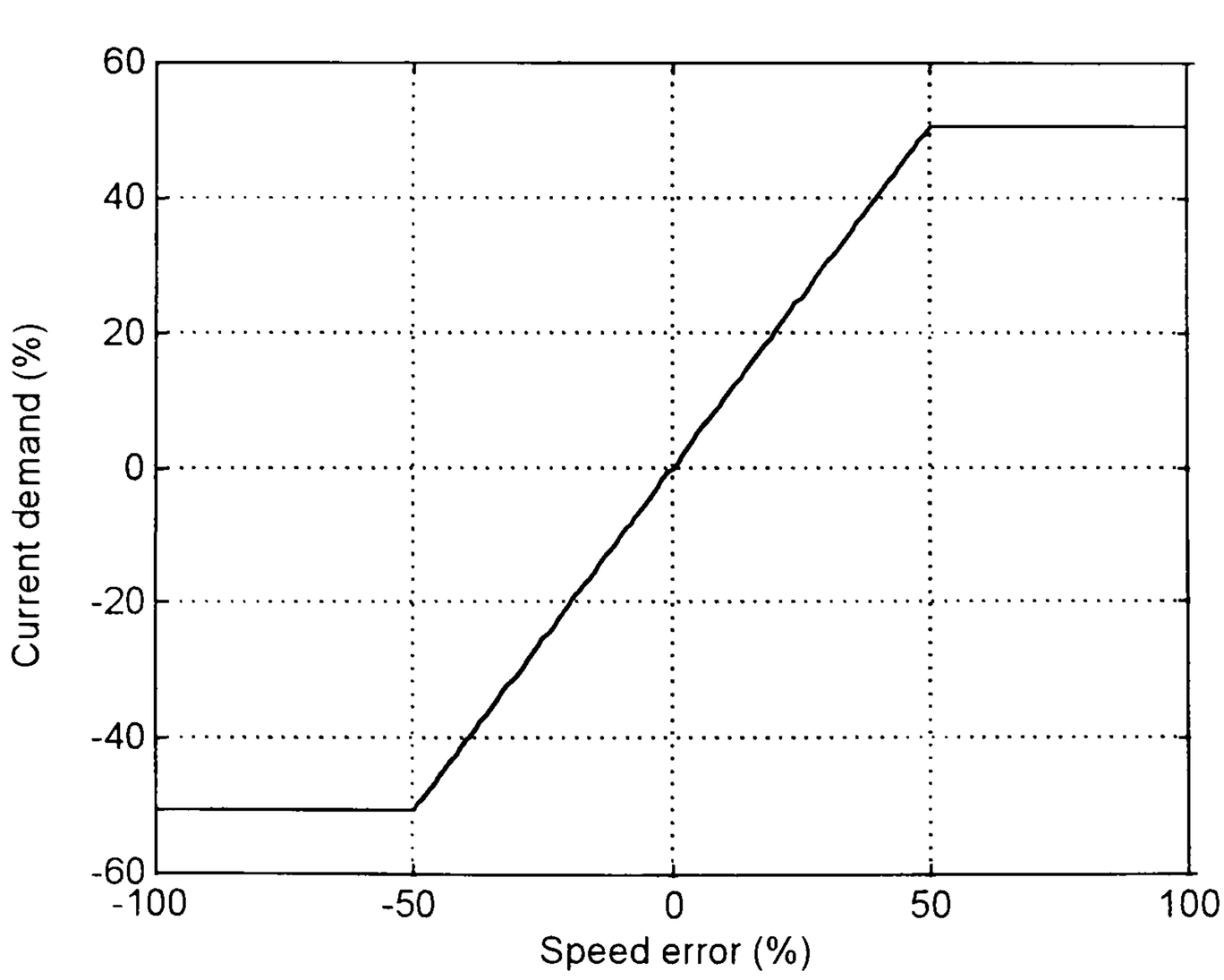



Fig. 8.11 - Transfer function of the single-input single-output fuzzy speed controller with the output membership functions shown in Fig. 8.10.

### 8.2.5 - Influence of increased number of membership functions

Fuzzy controllers offer the possibility of shaping the transfer function such that the effective gain varies over the speed error range. To change the gradient of the transfer function so that higher gain is applied close to zero speed error, two extra membership functions are used (Figs. 8.12 and 8.14) and two more rules are needed to map the extra input membership functions into the output:

RULE 3 – **If** input is small negative **then** output is small negative

RULE 4 - **If** input is small positive **then** output is small positive

A comparison of the transfer functions in Figs. 8.3, 8.13 and 8.15 shows that the effect of the extra membership functions is to change the gradient of the transfer function around zero speed error. By reducing the crossover range of the two input membership functions which are closer to zero, the gradient of the transfer function is increased and as a consequence, the speed error is minimised.

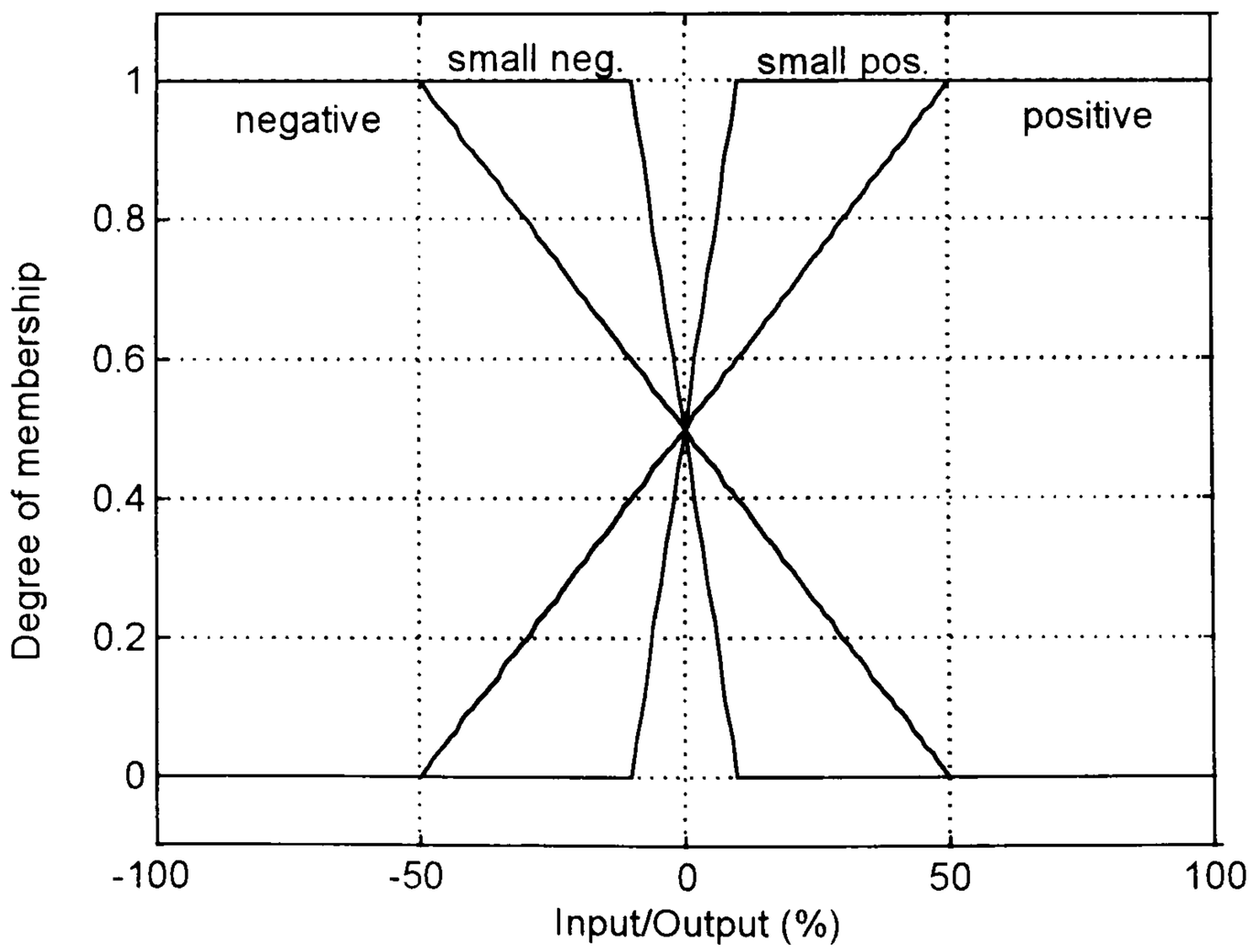



Fig. 8.12 - Input and output membership functions of the fuzzy speed controller with four membership functions for each variable.

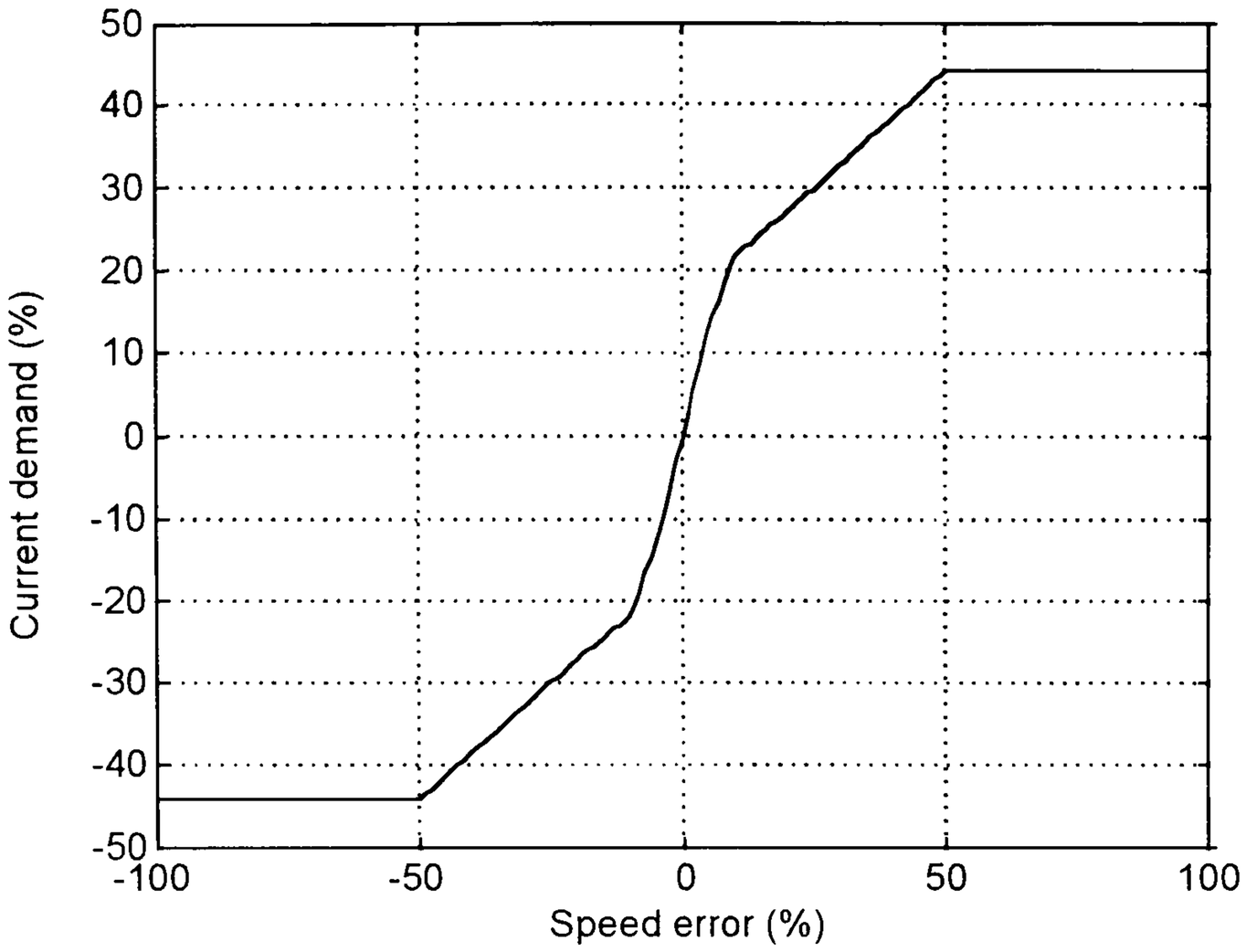



Fig. 8.13 - Transfer function of the fuzzy speed controller with membership functions shown in Fig. 8.12.

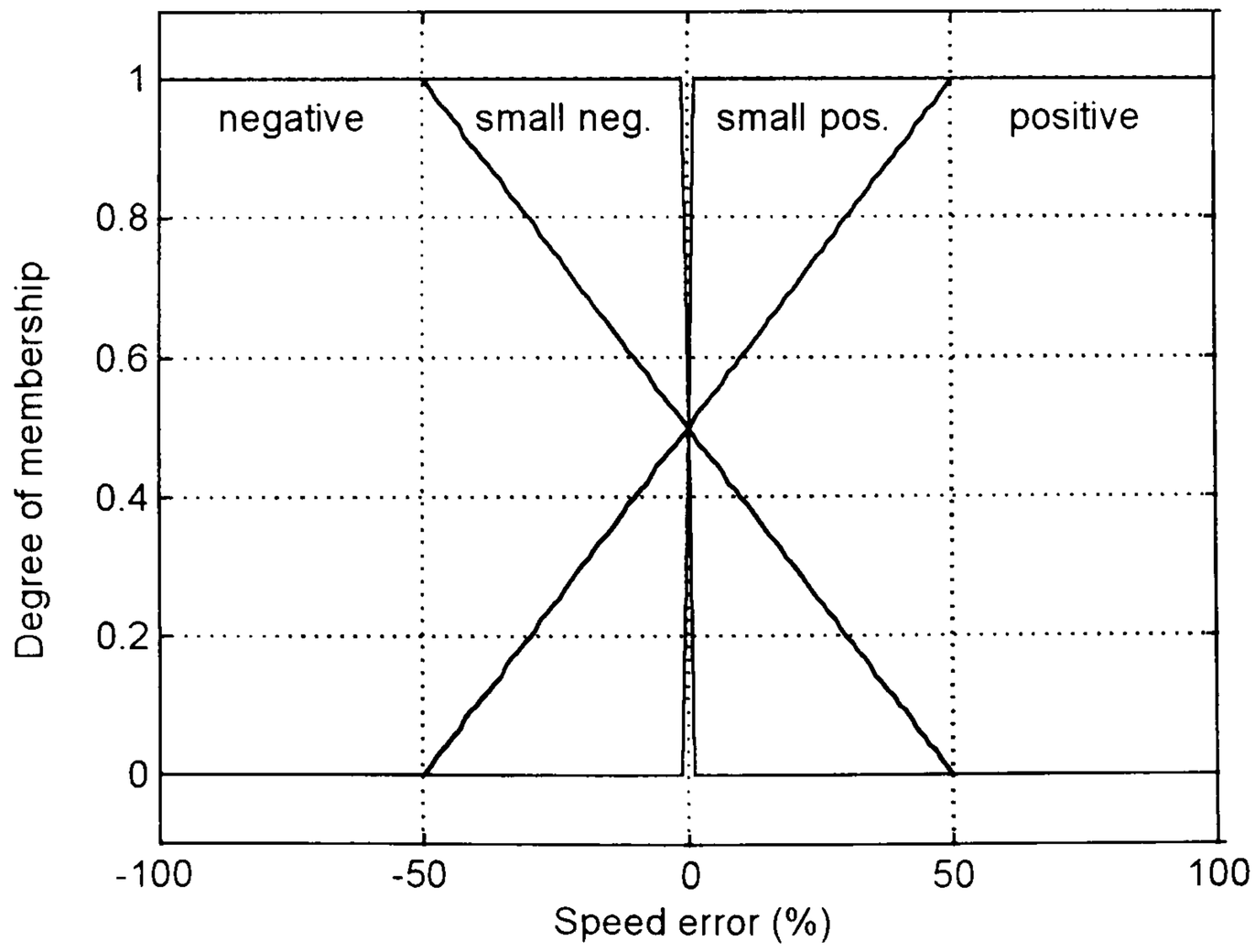



Fig. 8.14 Input and output membership functions of the fuzzy speed controller with four membership functions for each variable.

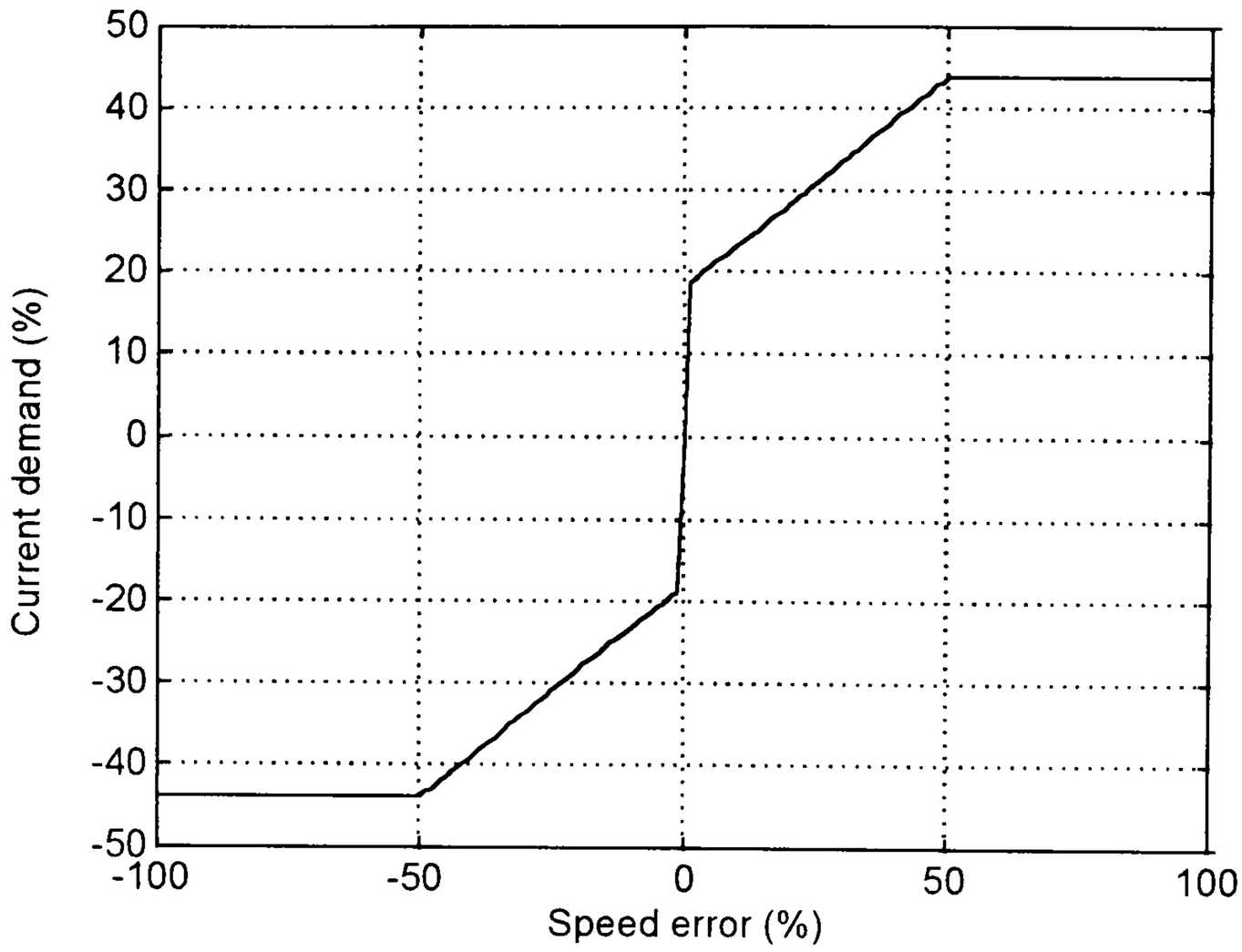



Fig. 8.15 - Transfer function of the fuzzy speed controller with membership functions shown in Fig. 8.14.

The results of this Section have demonstrated that the single-input single-output fuzzy controller has the characteristics of a proportional controller with variable gain. The gain characteristic can be modified by adjusting the parameters of the input and output membership functions. A large number of membership functions and rules can be useful in the control of a highly non-linear system, but in speed control of electric drives, satisfactory performance is possible with a very modest number of membership functions and rules.

## 8.3 - EXPERIMENTAL RESULTS

The experimental set up was as explained previously in chapter 3. In order to demonstrate experimentally the general comments in section 8.2, three single-input single-output fuzzy controllers were used. Firstly, a fuzzy controller with two membership functions per variable and low gain. Secondly, a controller with four membership functions per variable and higher gain around zero speed error. Lastly, a fuzzy controller with two membership functions per variable and high gain. For illustration purpose, a comparison with a simple proportional and proportional-integral is made.

In order to avoid interference of non-linearity such as torque / current limit, the controller settings used here have been carefully chosen to minimise the impact of torque saturation on the results. In order to reinforce this condition, the step input speed has been chosen to be sufficiently small (1000 rpm to 1100 rpm). Within this condition, the controller tuning has been made so saturation is avoided.

For all the tests in this chapter, the resistor load bank was set to produce a viscous load corresponding to 67% of the full load torque at 1000 rpm. The method of quantifying differences in speed response is the Integral with respect to Time of Absolute speed Error (ITAE) [DaSilva *et al*, EPE'97], as revealed earlier in chapter 5.

### 8.3.1 - Fuzzy controller with two membership functions per variable and low gain

In this fuzzy controller, the membership functions are chosen so that the current limit of 9A is reached for a 100 rpm speed error. The transfer function is shown in Fig. 8.16. In the linear part of the transfer function characteristic the average gain is 0.09 A/rpm. However the fuzzy controller is capable of limiting the output current demand and there is a gradual transition from the linear to the limiting part of the characteristic.

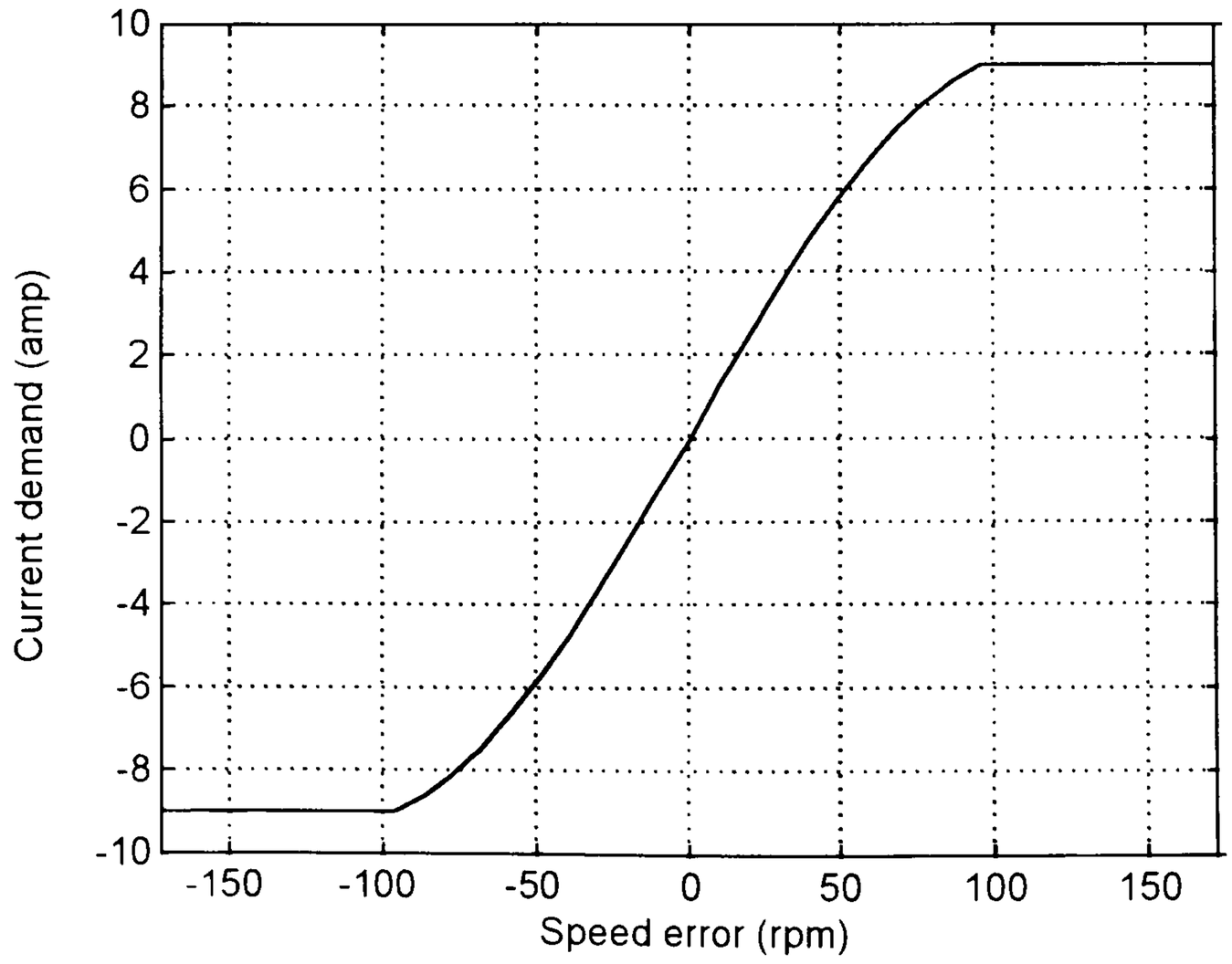



Fig. 8.16 - Transfer function of the fuzzy controller with two membership functions per variable and low gain.

Fig. 8.17 shows the speed and current response with this controller for a step increase in speed demand from 1000 to 1100 rpm. For the 1000 rpm initial speed demand, the load torque reduces the actual speed to 957rpm, giving a steady state speed error of 4.3%. At $t = 0$, a 1100 step input speed demand was applied. The current just hit its limit and the motor accelerated towards the speed demand. However, the maximum speed reached was 1055 rpm, giving a steady state speed error of 4.1% in the presence of 67% load. The ITAE was 6.1 rad.

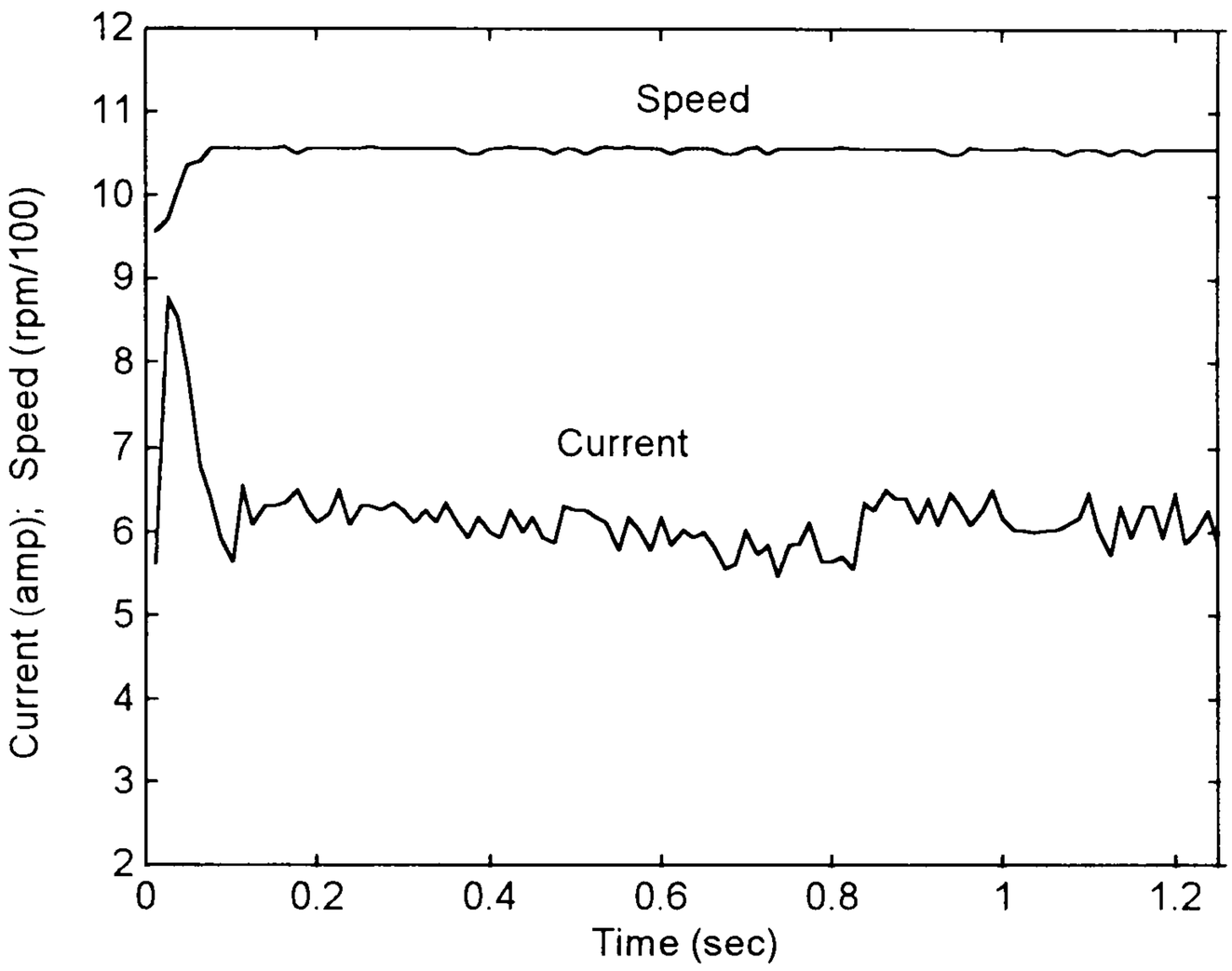



Fig. 8.17 Speed and current response with the fuzzy speed controller with two membership functions per variable and low gain.

### 8.3.2 - Fuzzy loic controller with four membership functions per variable

The fuzzy controller described in Section 8.3.1 gives large steady-state speed errors due to the low gain around zero speed error. As demonstrated in Section 8.2, it is possible to shape the transfer function of the fuzzy controller by using more membership functions. The second fuzzy controller has four membership functions per input and output variable, so that the gain can be made a function of speed error. The transfer function of the controller is shown in Fig. 8.18. Note that the maximum current demand is again limited to 9 A for speed errors of 100 rpm and above. However the gain around zero error is larger (0.34 A/rpm).

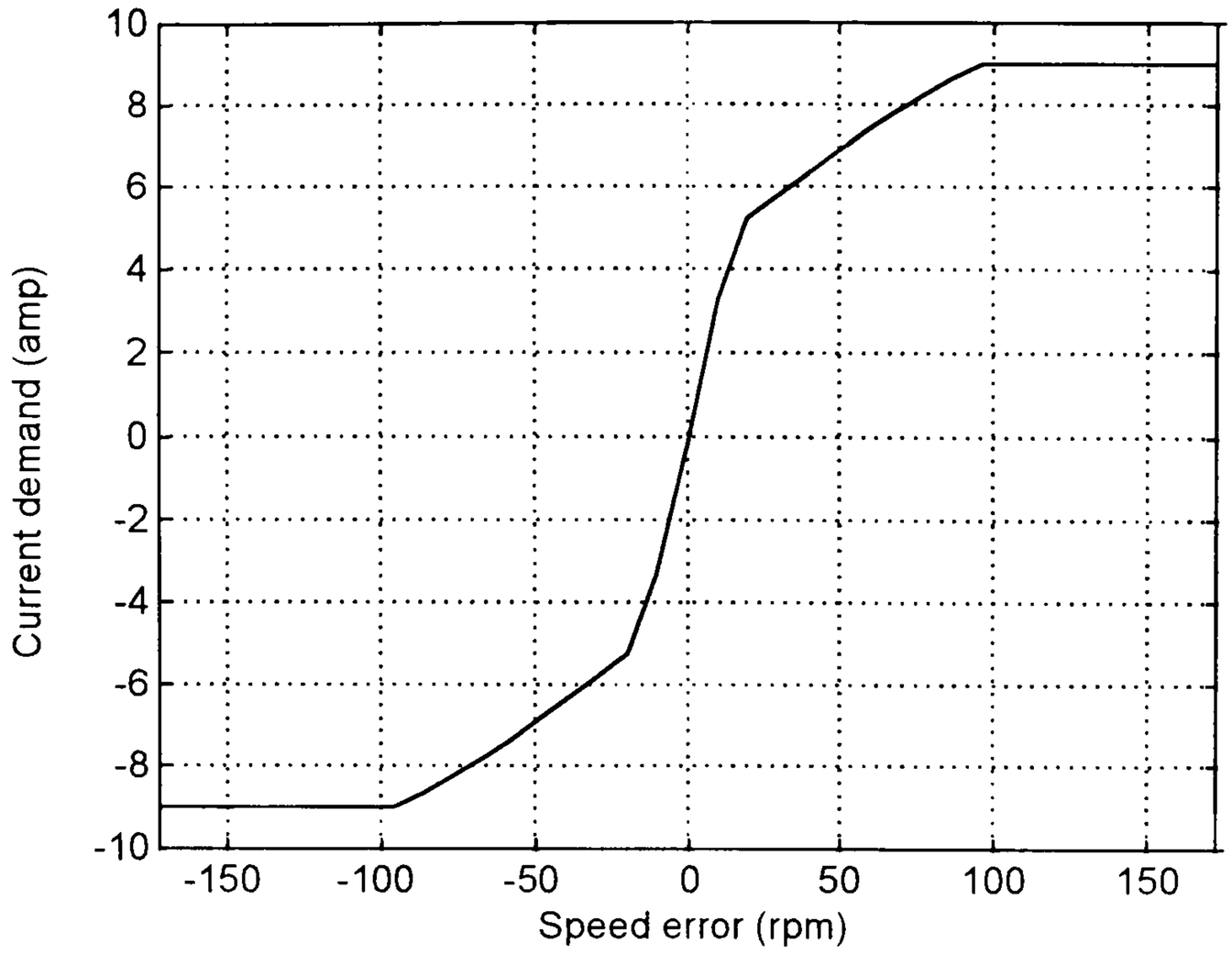



Fig. 8.18 - Transfer function of the fuzzy controller with four membership functions per variable.

The speed response for the four membership function fuzzy controller appears in Fig. 8.19. In the initial steady-state with a 1000 rpm speed demand, the motor speed was 979 rpm: a steady-state speed error of 2.1%. The final steady-state speed for the 1100 rpm speed demand is 1072 rpm, corresponding to 2.5% steady state speed error. The load for the motor at this speed is 67%. The ITAE over the 1.25 s window was 3.8 rad.

### 8.3.3 Fuzzy controller with two membership functions per variable and high gain

The third fuzzy controller has only two membership functions per input and output variable, but is tuned so that the gain around zero steady-state speed error is high (0.75 A/rpm), while preserving the maximum output current demand of 9 A, as shown in Fig. 8.20.

Fig. 8.21 depicts the corresponding speed and current response for the step increase in speed demand. The speed response with this controller surpassed that of the other controllers because the current remained at its limit during the acceleration phase. The high gain reduced steady-state speed errors: at 1000 rpm speed demand, the actual

speed was 993 rpm (steady-state speed error = 0.7%); at 1100 rpm speed demand, the actual speed was 1092 rpm (steady-state speed error = 0.72%). The ITAE from the step input of speed demand was 1.3 rad.

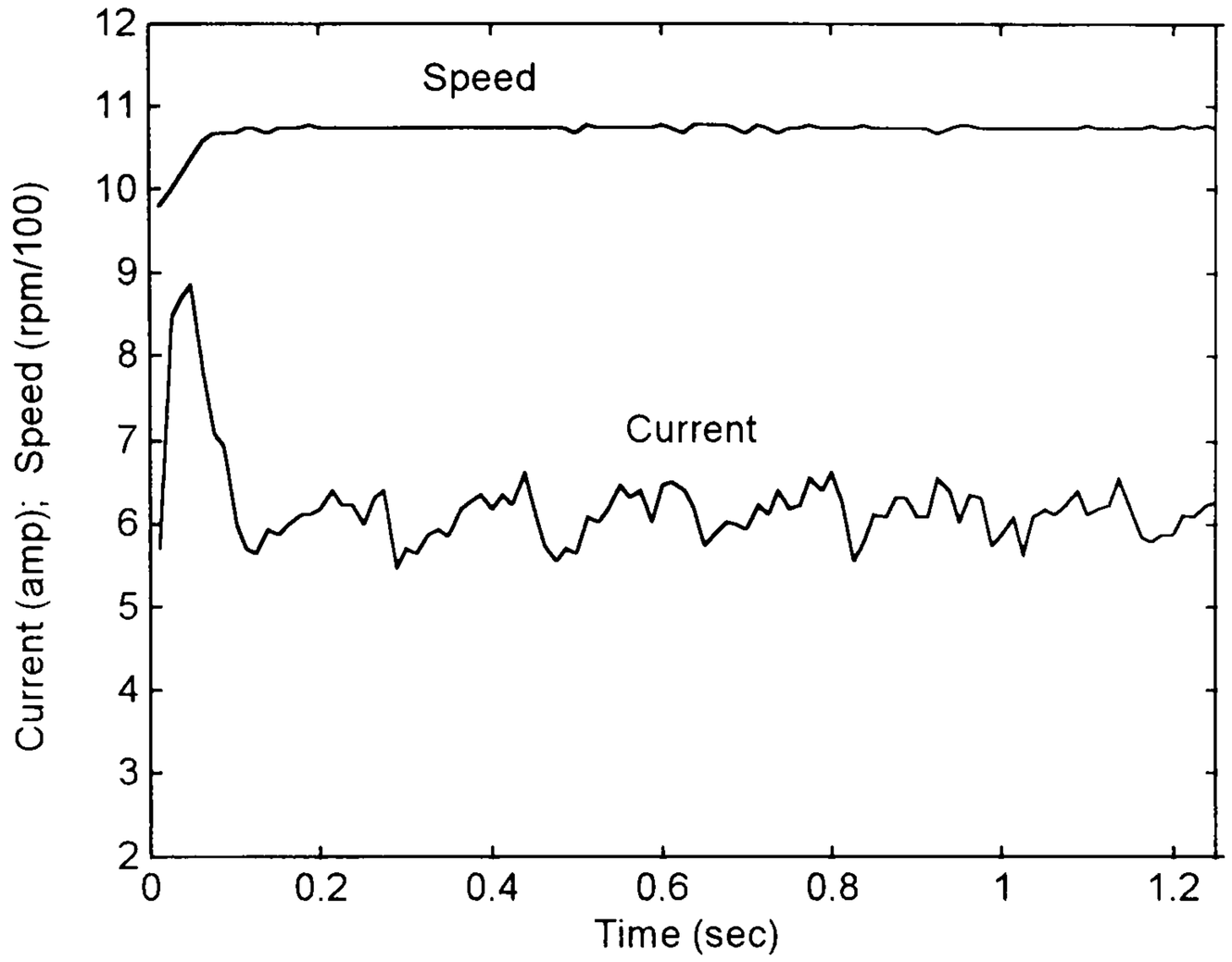



Fig. 8.19 - Speed and current response for the fuzzy controller with four membership functions per variable.

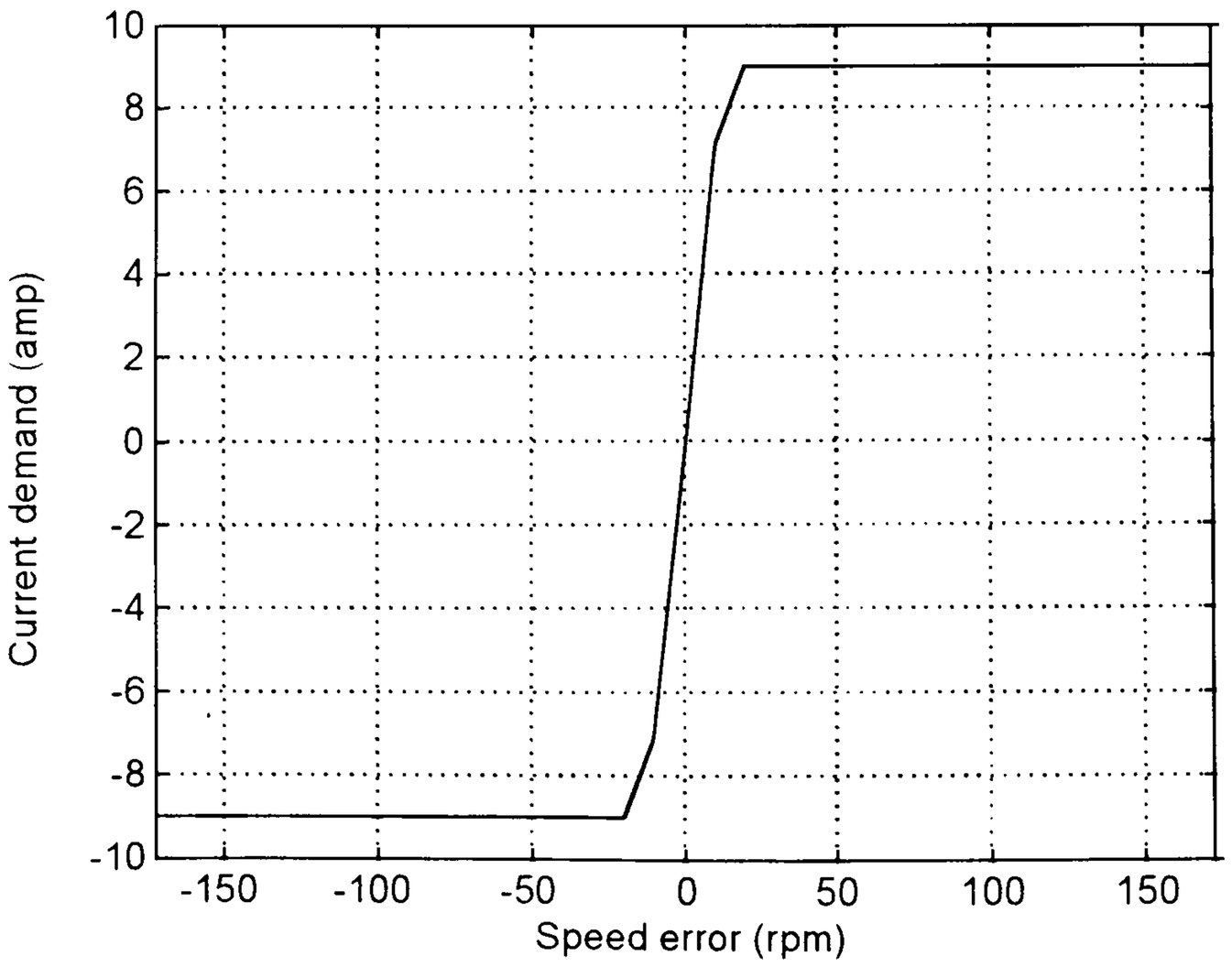



Fig. 8.20 - Transfer function of the fuzzy controller with two membership functions per variable and high gain.

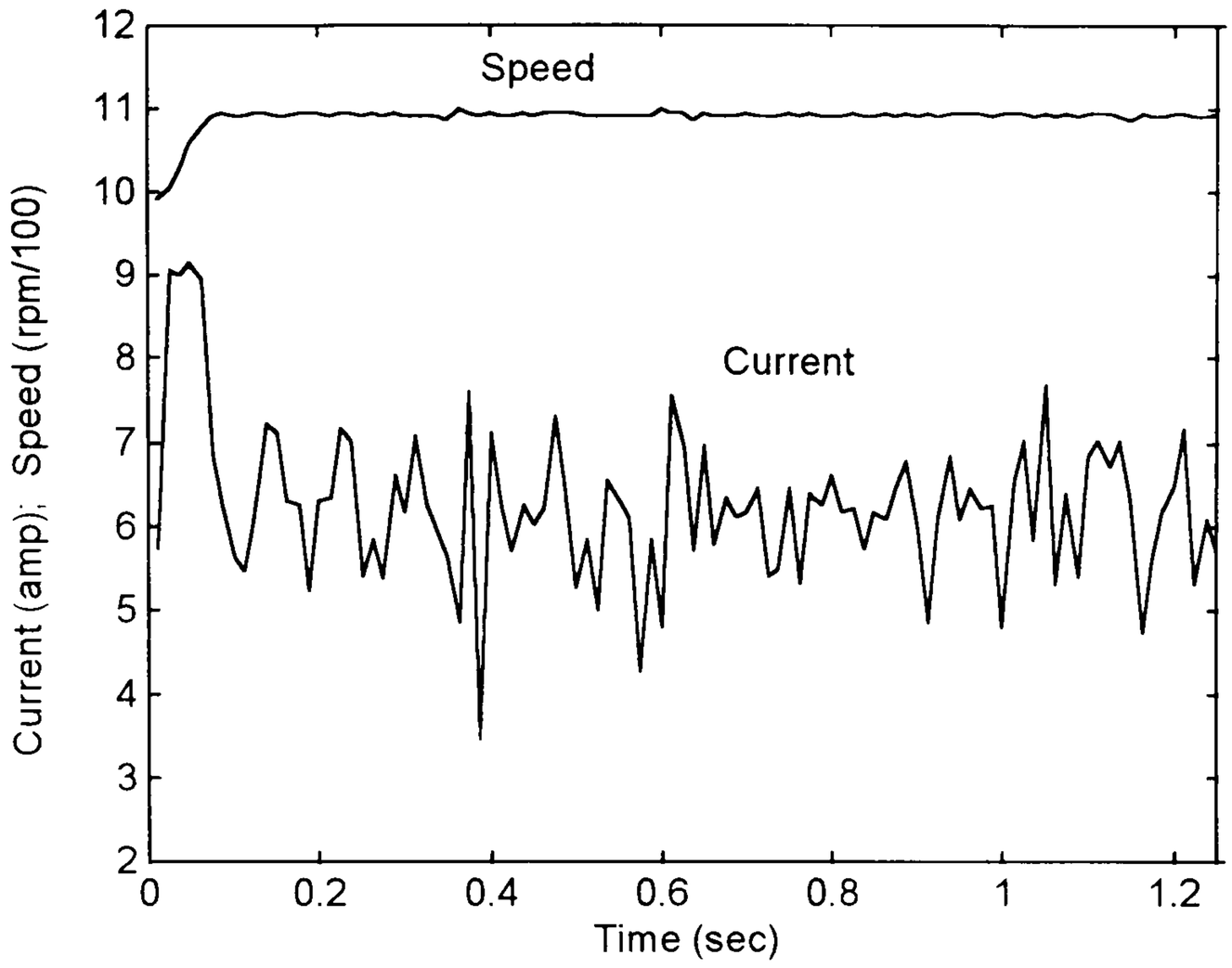



Fig. 8.21 - Speed and current response for the fuzzy controller with two membership functions per variable and high gain.

### 8.3.4 - Proportional speed controller

The procedure for selection of the chosen proportional gain is as follows: The gain of the controller was adjusted so that the maximum current demand signal (9 A) was applied, without saturating the controller, to speed up the motor from 1000 to 1100 rpm. Since saturation was avoided, the control system was linear. The maximum speed error is 100 rpm and therefore the controller gain is set to 9/100 = 0.09 A/rpm, identical to those used on the fuzzy controllers. The speed and current responses for the proportional controller are shown in Fig. 8.22. Despite the 1000 rpm initial speed demand, the motor's speed was 940 rpm, giving a steady state speed error of 6%. At $t = 0$, the speed demand was increased to 1100rpm. The current just hit its limit and the motor accelerated towards the speed demand. However, the maximum speed it reached was 1035 rpm, (steady state speed error = 5.9%) in the presence of 67% load. The ITAE within the 1.25 s time window was 8.7 rad.

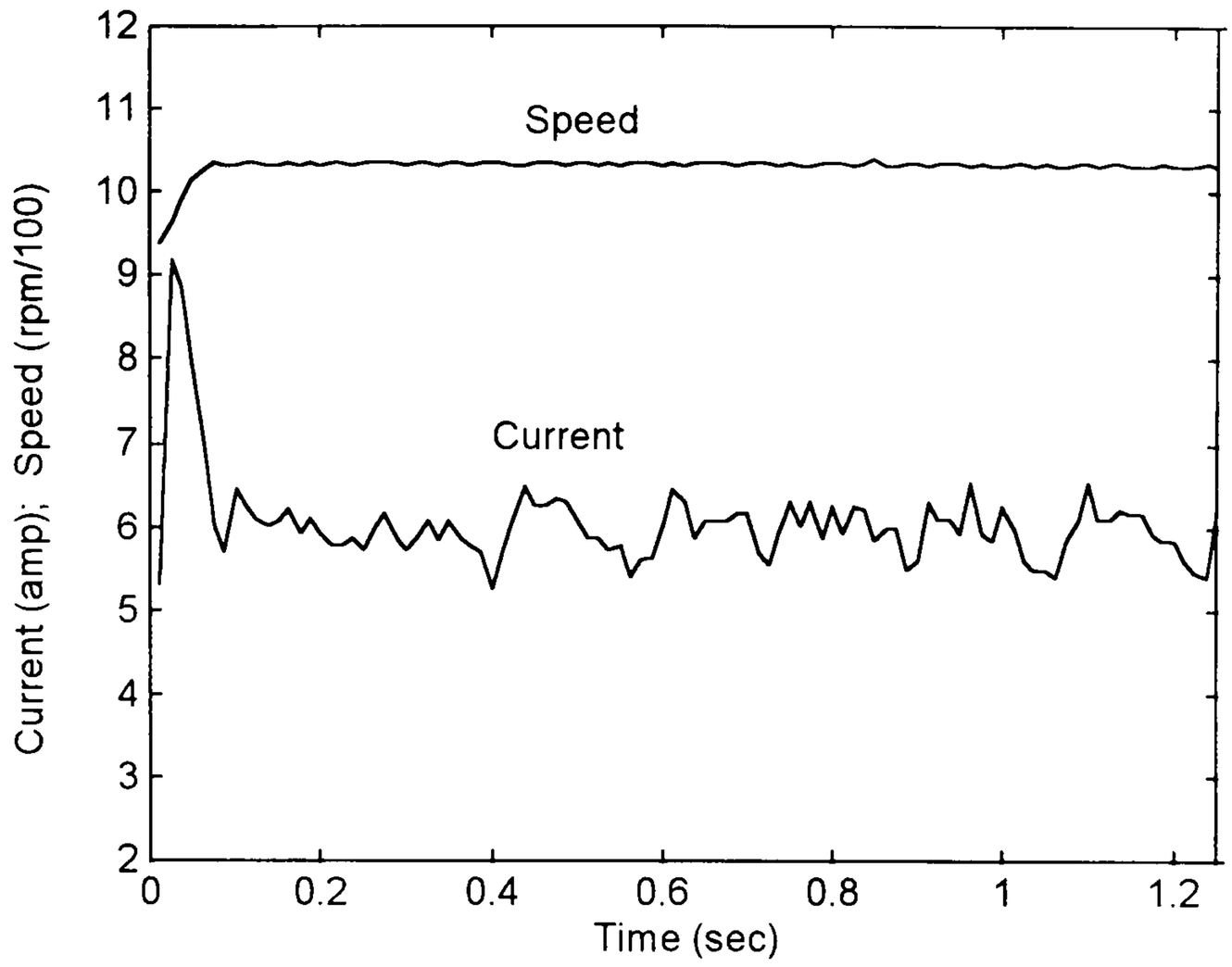



Fig. 8.22 - Speed and current response with proportional speed controller due to 1000 to 1100 rpm step of speed demand.

### 8.3.5 - Proportional-Intregral speed controller

The steady-state speed errors in the proportional controller are counteracted by introducing an integral term in the controller. The integral gain was adjusted to a value of (0.11 A.s/rpm), so that a critically damped speed response was produced for a 100 rpm step input speed demand. By doing this, the fastest acceleration was obtained without speed overshoot and the control system was linear. Fig. 8.22a represents the current and speed response obtained by using the PI speed controller. In this case there is no steady state speed error and the ITAE is equal to 0.6 rad, mostly due to the speed error during the acceleration time. The load at 1100 rpm corresponds to 70%. of full load.

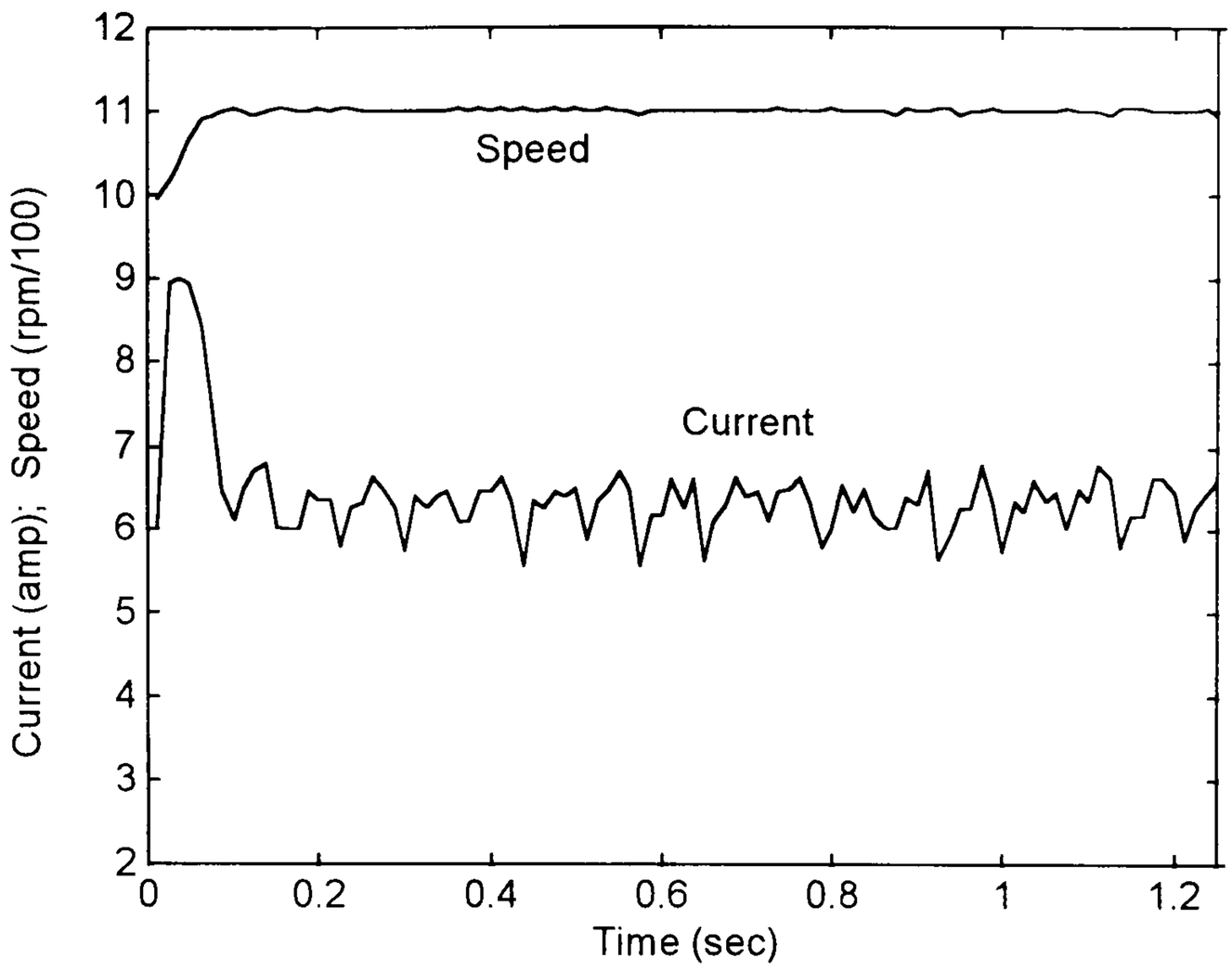



Fig. 8.22a - Speed and current response with PI speed controller due to 1000 to 1100rpm step of speed demand.

## 8.4 - COMPARISON OF PERFORMANCE IN RESPONSE TO A STEP INPUT OF SPEED DEMAND

The results for the controllers described in this Section are summarised in Table 8.1. The results demonstrate that the performance of the fuzzy controllers is better than that of a simple proportional controller, because of the shaped transfer function characteristic of the fuzzy controllers. However there will always be a non zero steady state speed error, which is proportional to the gain imposed by the fuzzy controller and the load torque of the motor. However none of the fuzzy controllers can match the performance of the proportional-integral controller with respect to steady-state speed error or the ITAE over the 1.25 s window used for the comparisons. The high-gain, two-membership function fuzzy controller (Section 8.3.3) would be able to outperform the proportional-integral controller over a shorter time window, which would emphasise the faster initial response.

| Controller | Section No. | Actual speed for 1100rpm demand (rpm) | Speed Error (%) | ITAE (rad) |
|---|---|---|---|---|
| **Fuzzy logic** | | | | |
| Two membership function, low gain | 8.3.1 | 1055 | 4.1 | 6.1 |
| Four membership function | 8.3.2 | 1072 | 2.5 | 3.8 |
| Two membership function, high gain | 8.3.3 | 1092 | 0.7 | 1.3 |
| **Conventional** | | | | |
| Proportional | 8.3.4 | 1035 | 5.9 | 8.7 |
| Prop-Integral | 8.3.5 | 1100 | 0.0 | 0.6 |

Table 8.1: Summary of results for step input speed demand

## 8.5 - ROBUSTNESS AGAINST LOAD VARIATION

The robustness of the fuzzy controller described in sections 8.3.1, 8.3.2 and 8.3.3 has been examined for a step change in load torque. The drive was operated with a constant speed demand of 1000 rpm and an initial load torque of 11% of rated torque. The load torque was suddenly increased to 78% of rated torque for an interval of 1.5s and the speed transient recorded. In all cases the controller tuning was the same as described in section 8.3. The same load torque change has been applied to the proportional and proportional-integral speed controllers. The results are as follows.

### 8.5.1 - Fuzzy controller with two membership functions per variable and low gain

Fig. 8.23 illustrates the speed and current response for the fuzzy controller defined in Section 8.3.1, which transfer function is shown Fig. 8.16. The initial speed was 990

rpm for the 1000 rpm speed demand (1.0% steady state speed error). After the disturbance was applied, the speed came down to 946 rpm, which corresponds to 5.4% steady state speed error.

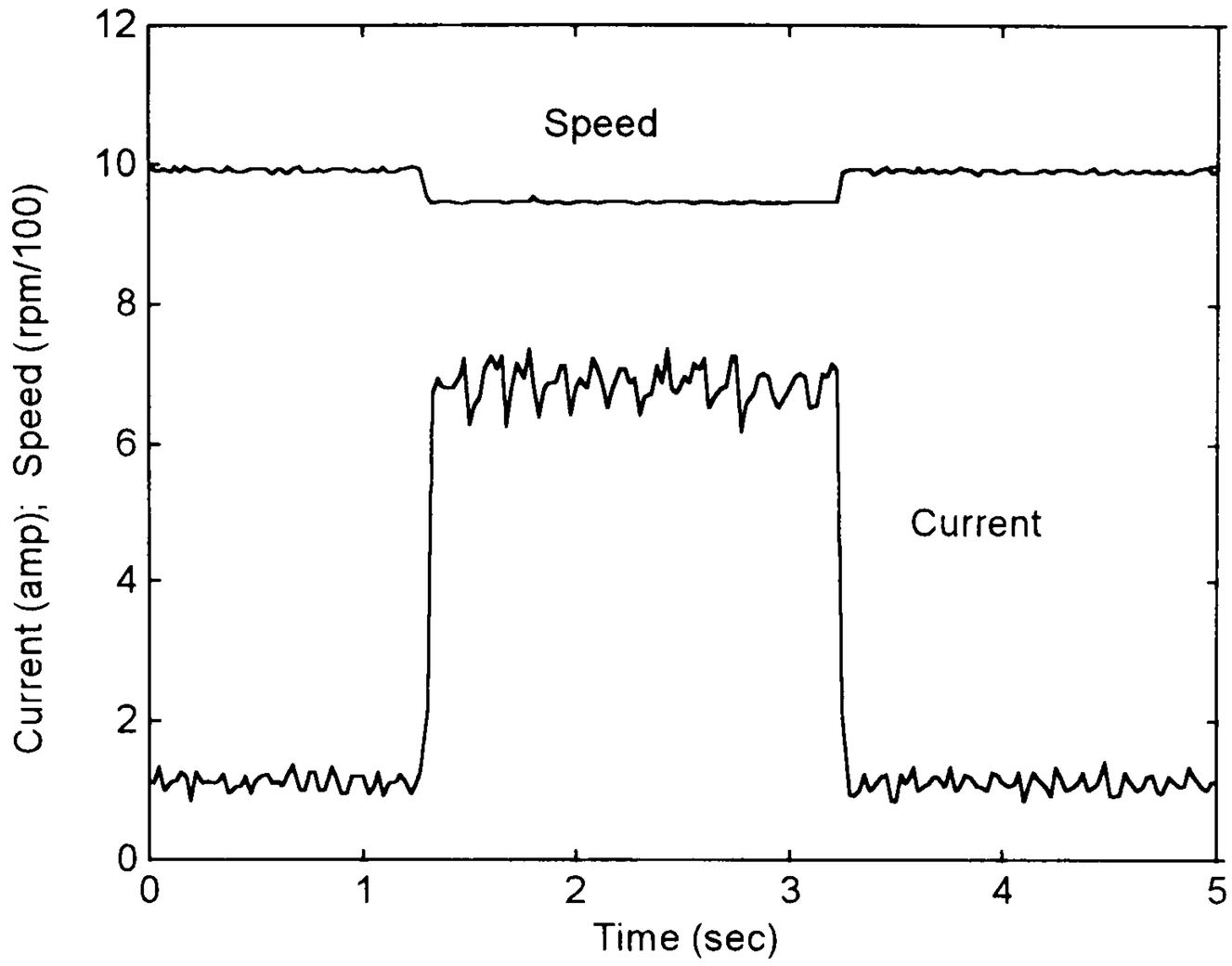



Fig. 8.23 - Speed and current response for the fuzzy controller with two membership functions per variable and low gain due to 78% load torque.

### 8.5.2 - Fuzzy controller with four membership functions per variable

Fig. 8.24 depicts the speed and current response for the fuzzy controller having four membership functions (Section 8.3.2) and the transfer characteristic shown in Fig. 8.3.18. As this controller has a higher gain around zero speed error, steady-state speed errors were smaller than for the previous fuzzy controller. The motor ran initially at 997 rpm for a 1000 rpm speed demand: a 0.3% steady state speed error. Following the step input load disturbance, the speed reduced to 960 rpm, corresponding to 4.0% steady state speed error.

### 8.5.3 - Fuzzy controller with two membership functions per variable and high gain

The speed and current responses obtained with the fuzzy controller of Section 8.3.3, with transfer function as shown in Fig. 8.20. This fuzzy controller has the highest

gain of all controllers around zero speed error and therefore the steady state speed errors are minimised. The motor ran initially at 999.5 rpm, a 0.05% steady state speed error.

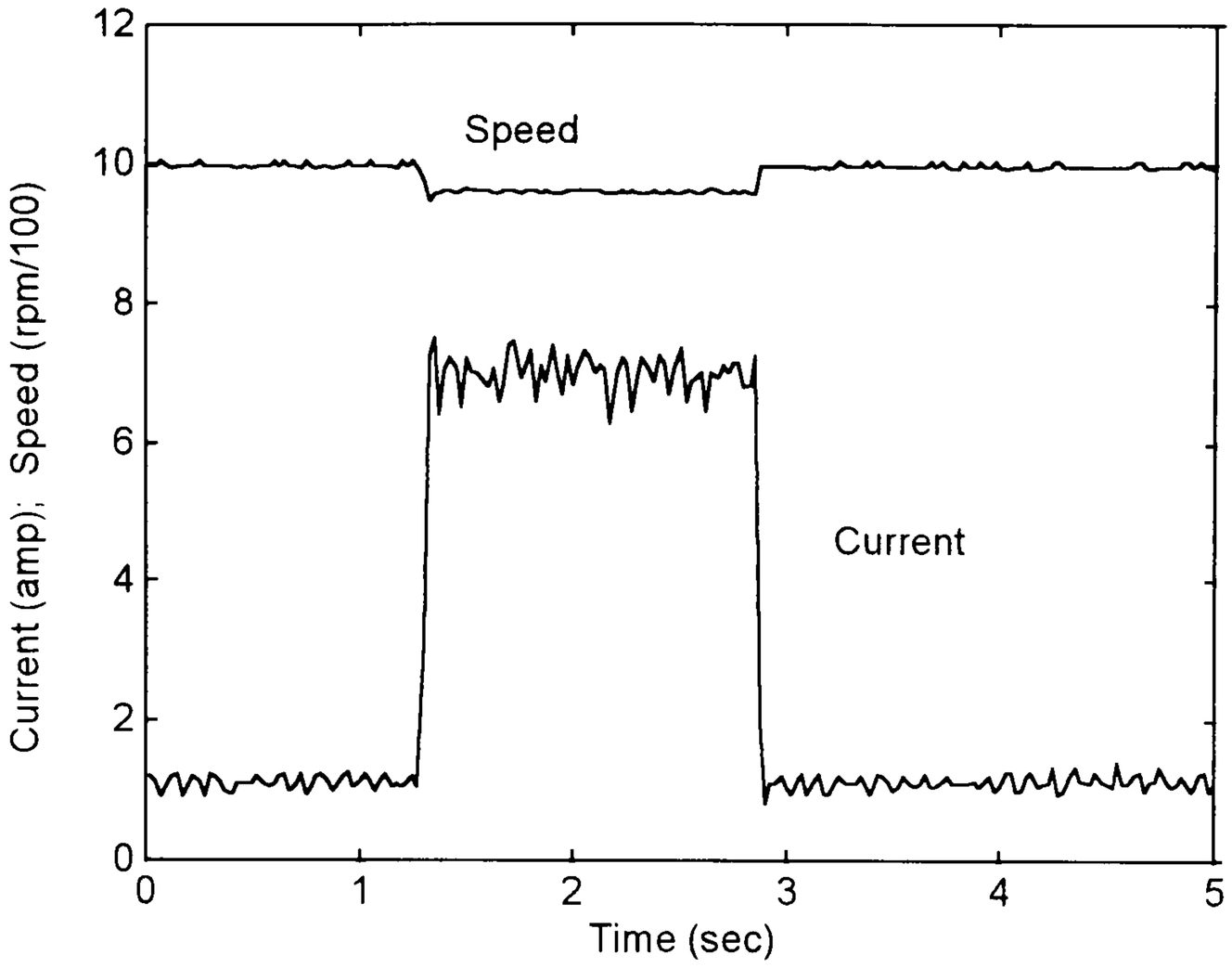



Fig. 8.24 - Speed and current response for the fuzzy controller with four membership functions per variable due to 78% load torque.

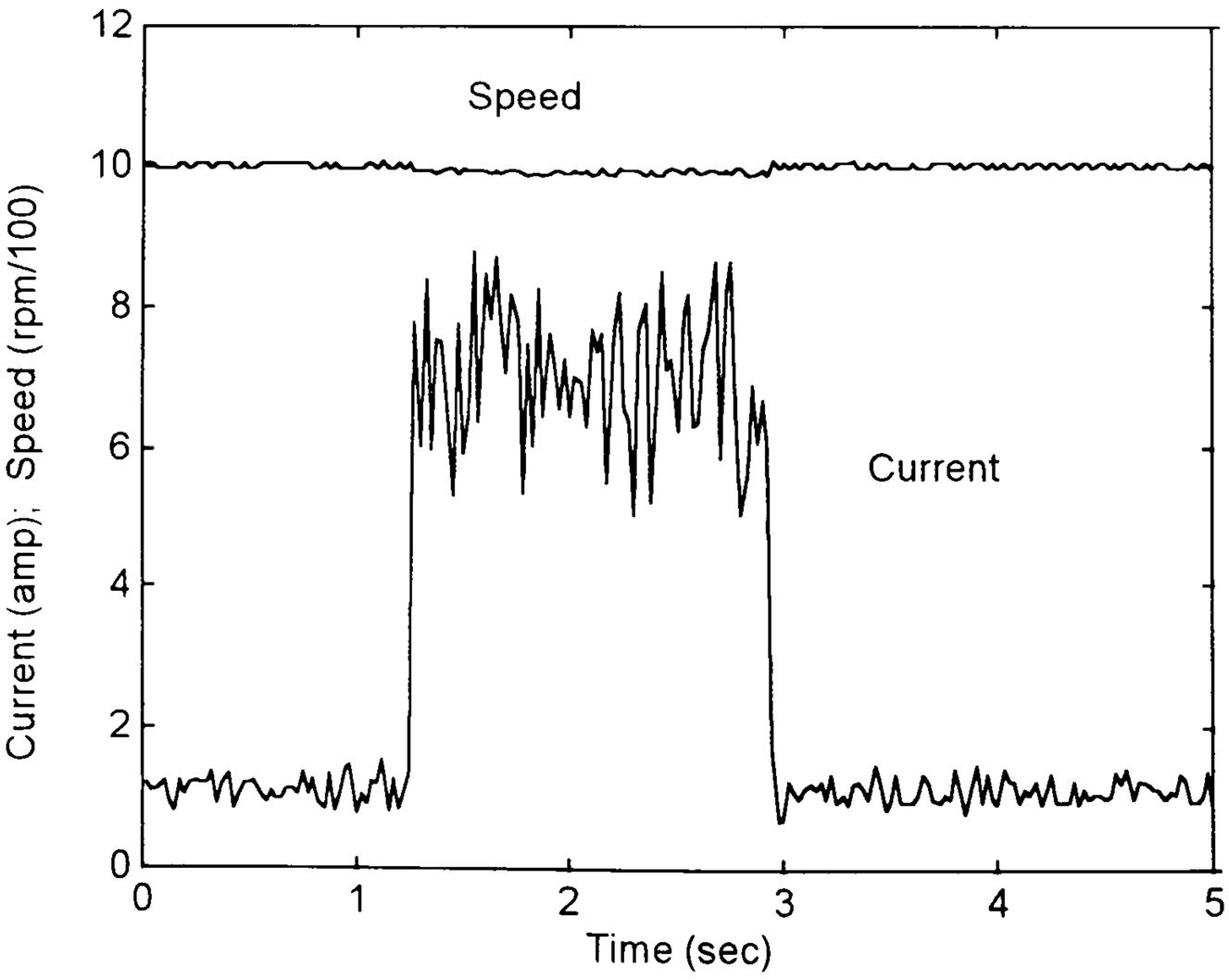



Fig. 8.25 - Speed and current response with high gain fuzzy speed due to 78% load torque.

Due to the step input load increase, the speed came down to 990 rpm, corresponding to 1.0% steady state speed error, as shown in Fig. 8.25 and, in enlarged scale, in Fig. 8.26.

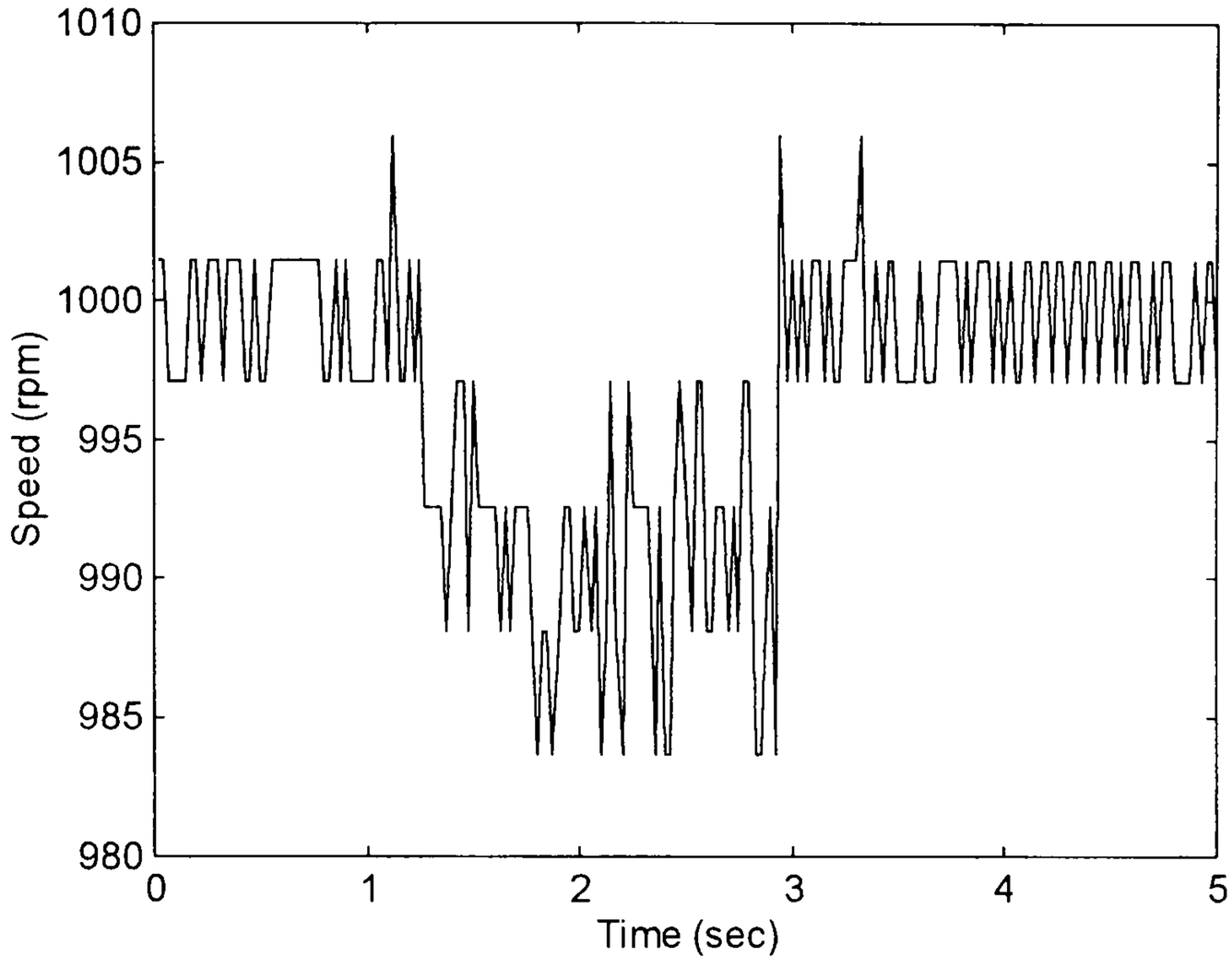



Fig. 8.26 - Speed and current response with high gain fuzzy speed due to 78% load torque in enlarged vertical scale.

### 8.5.4 - Proportional speed controller

Fig. 8.27 shows the speed and current response with the proportional speed controller having a gain of 0.09 A/rpm. With a load of 11% the actual motor speed was 986 rpm (steady-state speed error = 1.4%). After applying the step input load disturbance, the speed reduced to 926 rpm, corresponding to a steady state speed error of 7.4%.

### 8.5.5 - Proportional-Integral speed controller

In Fig. 8.28 are shown the speed and current responses obtained with the PI controller described in section 8.3.5. The steady state speed errors before and after the disturbance were 0% because of the integrator action. The speed initially dropped to 940 rpm due to the disturbance, corresponding to 6.0% of the demanded speed.

However, the integrator brought the speed back to its original value with 0% steady state speed error, as shown in enlarged scale in Fig. 8.29.

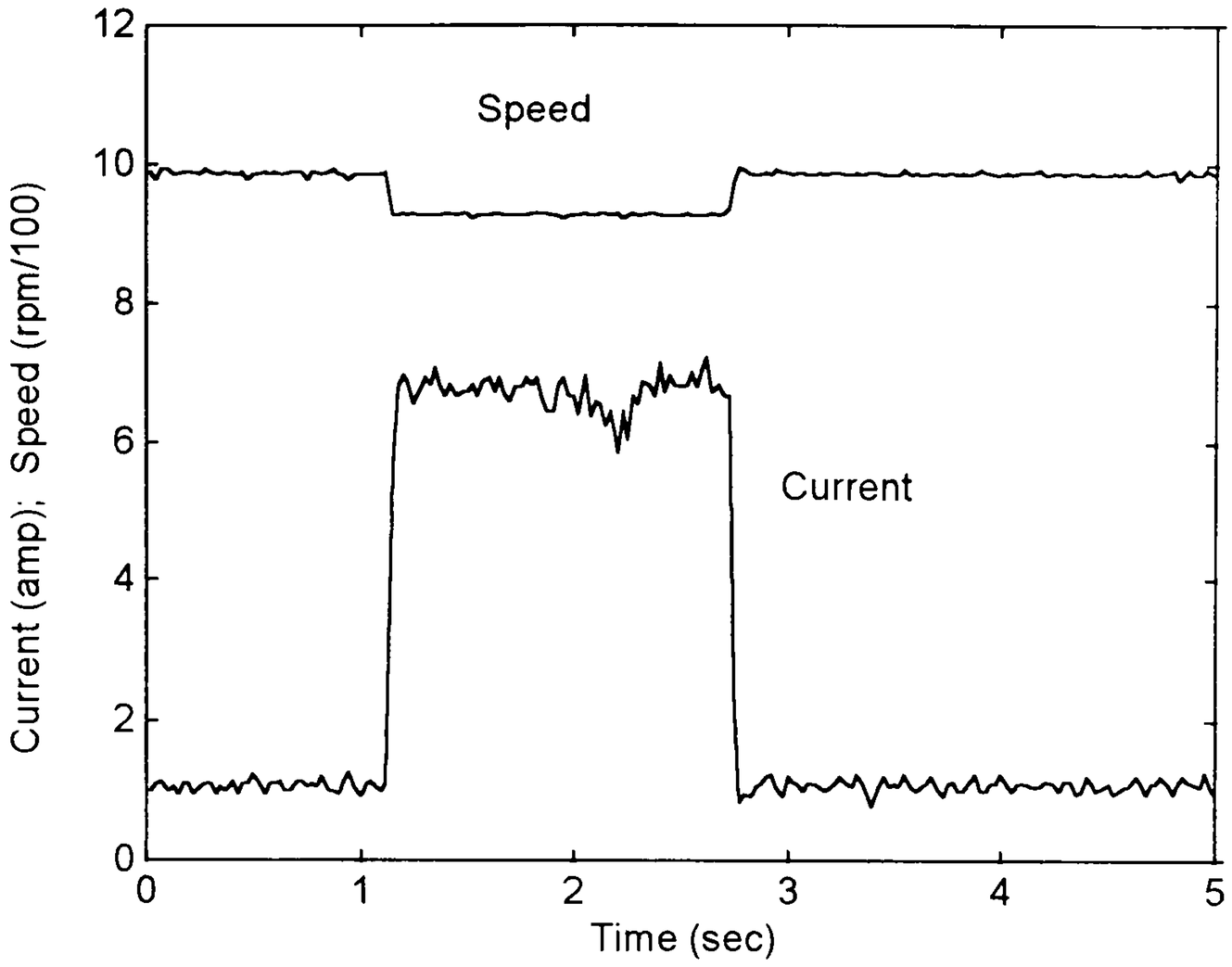



Fig. 8.27 - Speed and current response with proportional speed controller due to 78% load torque.

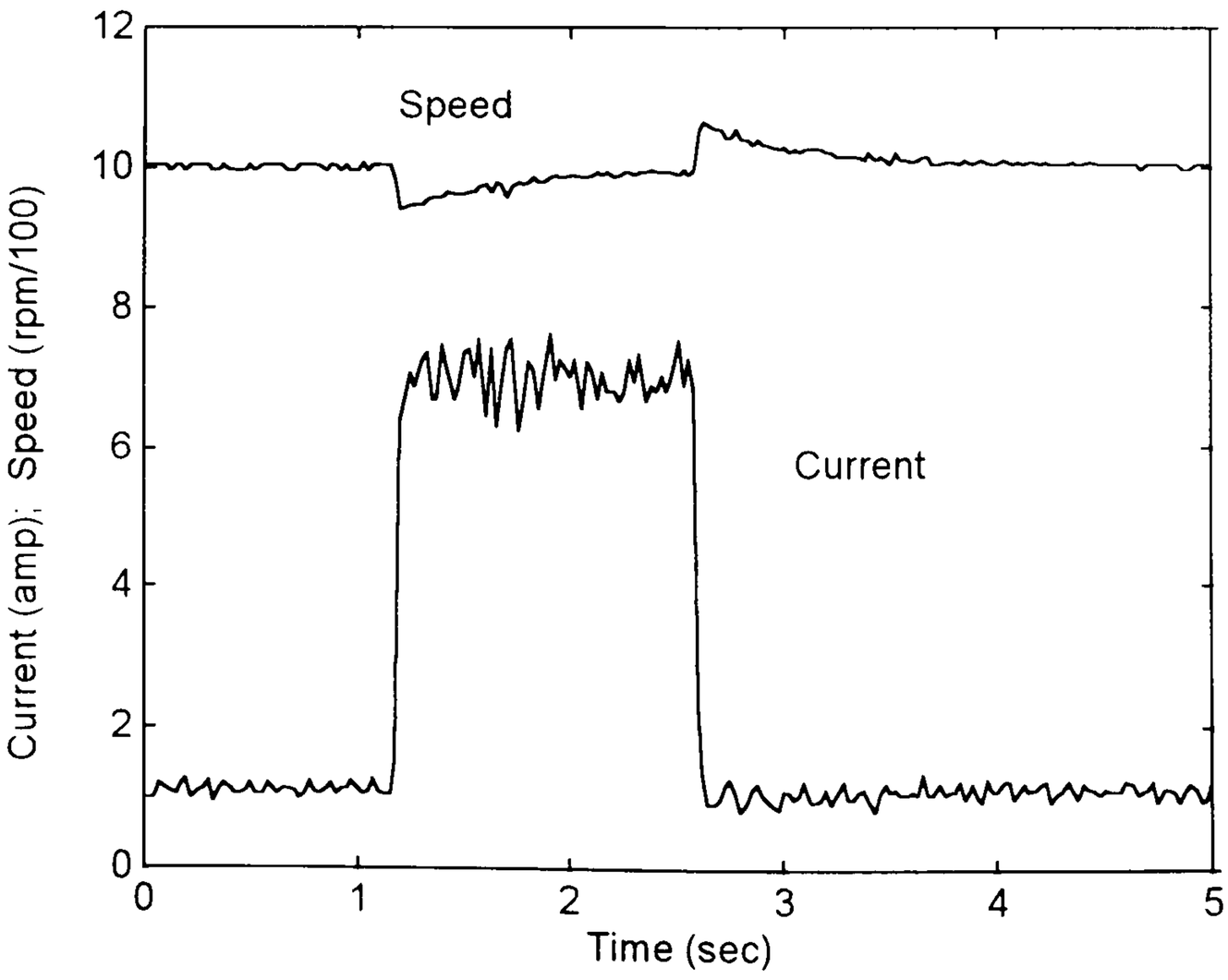



Fig. 8.28 - Speed and current response.

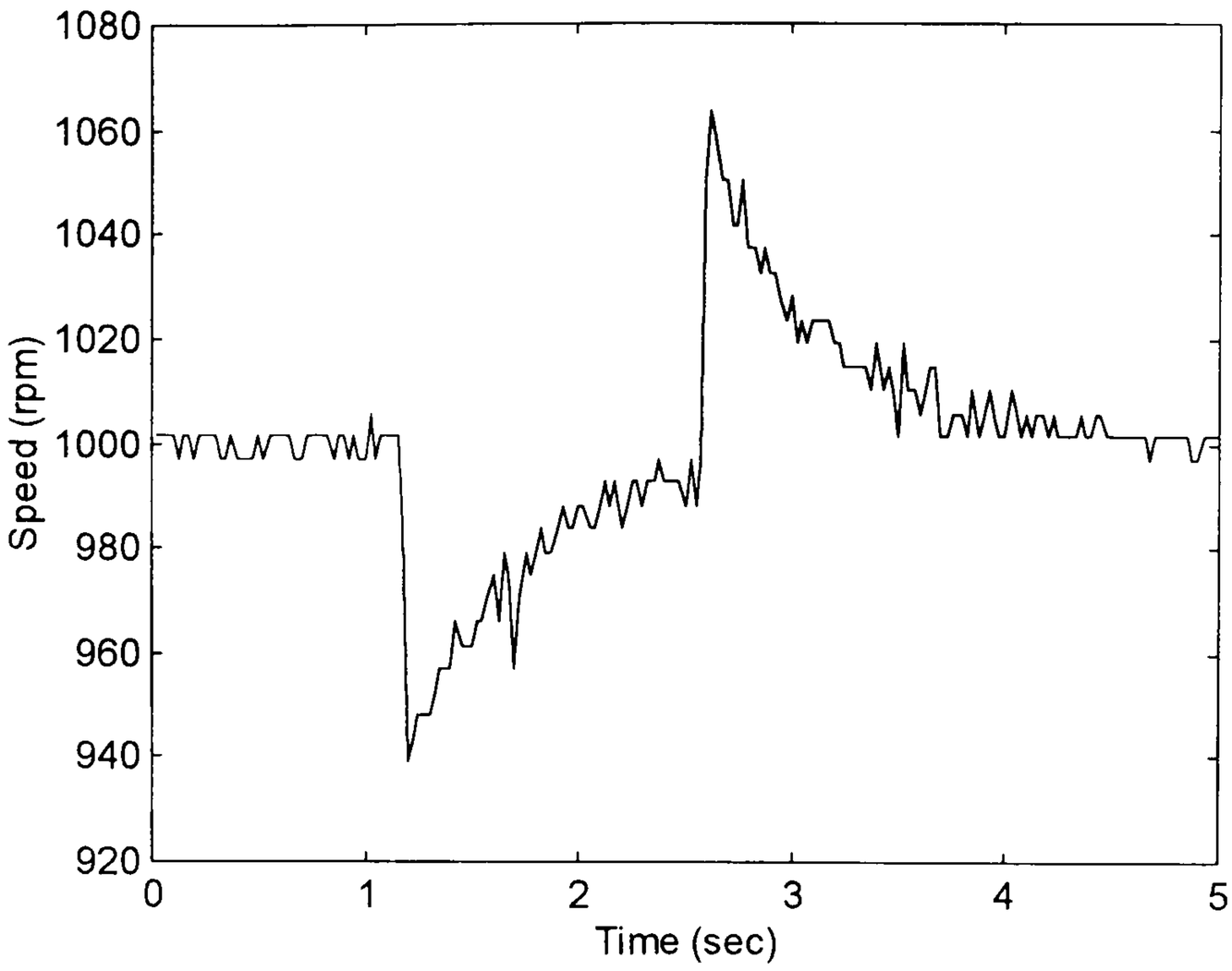



Fig. 8.29 – Speed response in enlarged vertical scale.

Table 8.2 summarises the performance of the five controllers during a large load disturbance.

| Controller | Section No. | Actual speed for 11% load (rpm) | Speed error for 11% load (%) | Minimum speed for 78% load (rpm) | Speed error for 78% load (%) |
|---|---|---|---|---|---|
| **Fuzzy logic** | | | | | |
| Two membership function, low gain | 8.3.1 | 990 | 1.0 | 946 | 5.4 |
| Four membership function | 8.3.2 | 997 | 0.3 | 960 | 4.0 |
| Two membership function, high gain | 8.3.3 | 999.5 | 0.05 | 990 | 1.0 |
| **Conventional** | | | | | |
| Proportional | 8.3.4 | 986 | 1.4 | 926 | 7.4 |
| Prop-Integral | 8.3.5 | 1000 | 0.0 | 940 | 6.0* |

* *This figure refers to the largest speed error during the transient arising from the load change. The integrator action eventually reduces the speed error to zero.*

Table 8.2: Robustness against load variation

The results in Table 8.2 demonstrate that fuzzy controllers can have advantages over even a PI controller when speed of response to load disturbances is considered.

## 8.6 - CONCLUSION

In this chapter, a single-input single-output fuzzy controller has been investigated and compared to the conventional proportional and proportional-integral controllers. It has been shown that fuzzy controllers offer the possibility of shaping the transfer function characteristic to obtain, for example, high gain at low values of speed error. However, no matter the number of membership functions and rules, a single-input single-output fuzzy controller works as a proportional controller with variable gain.

Because the possibility of increasing the gain around zero speed error, fuzzy controller are able to outperform proportional controllers with regard to speed response and speed holding in the presence of load torque change. The fuzzy controller with two membership functions per variable and high gain produced a better transient response to a large load disturbance than even the proportional-integral controller. However the integrator action of the proportional-integral controller was always capable of reducing steady-state speed errors to zero and this performance feature could not be replicated by a single-input single-output fuzzy controller as considered in this chapter.

# CHAPTER 9

# MULTIPLE INPUT MULTIPLE OUTPUT - MIMO FUZZY CONTROLLER

## 9.1 - INTRODUCTION

In many publications, the fuzzy speed controller has up to three inputs but a single output [BAGHLI *et al*, EPE'97, GUILLEMIN, P., 1996, TANG, Y *et al*, 1995, QUIN et al, EPE'97]. Probably, there was no need of more then one output in the fuzzy controllers as they were used in different motor drives. However, in some cases, the authors have tried to imitate proportional integral speed controller by using fuzzy logic in which there were two inputs, usually the speed error and rate of speed error.

It has been shown in chapter 8 that replacing the classical Proportional-Integral speed controller by the single-input single-output fuzzy controller for the speed control of DC motor drive, does not bring improvements at steady state operation because of the steady state speed error.

As discussed in chapter 1, in an industrial process such as the film making, the controller has to be capable of holding the speed of the motor in the presence of load disturbance. Should the load torque increase, the torque produced by the motor has also to increase in order to keep the speed at the reference value. Information about the speed error only is not enough for the fuzzy controller to hold the speed when the load changes. An additional input, representing the load variation, is necessary. However, in order to make this input effective, it has to be associated to an output. In the case of the Brushless DC Motor Drive, should the load torque increases, the armature current of the motor has to be increased so more torque can be produced.

A convenient and simple way of doing this in fuzzy controller is by adding an extra output. In this case, this additional output works as a component of current demand

for the PI current controller. In this chapter, a two-inputs two-outputs as proposed by DaSilva *et al* [DaSilva *et al*, EPE'97] is discussed. In this fuzzy speed controller, which replaces the classical Proportional-Integral, the inputs are 1 - Speed Error and 2 - Estimated Load.

## 9.2 - THE FUZZY SPEED CONTROLLER

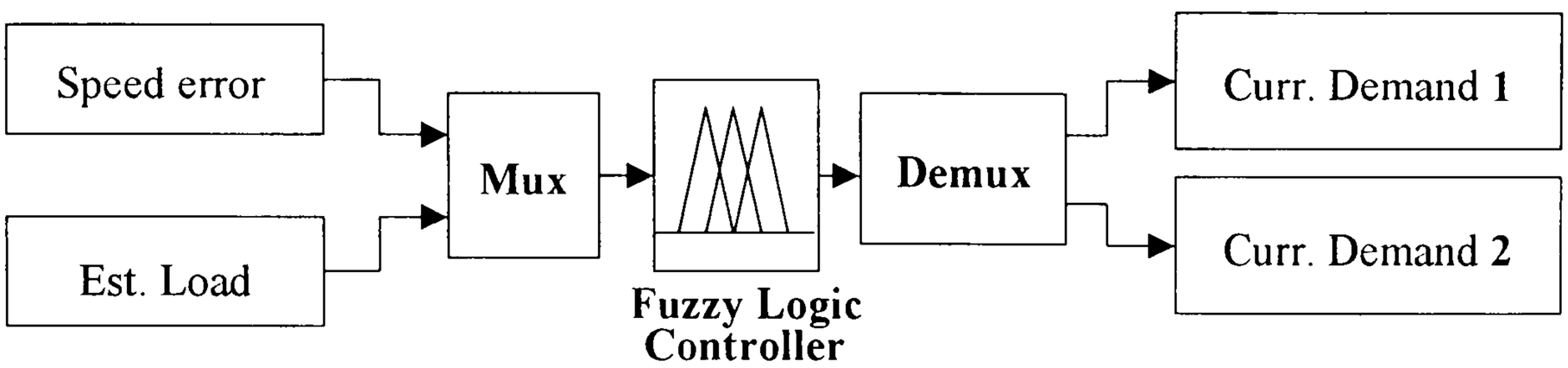



Fig. 9.1 - Two inputs two outputs fuzzy speed controller.

In the fuzzy speed controller (Fig. 9.1), the "Speed error" input is obtained simply by subtracting the actual speed value from the demanded one. The "Est. Load" input is obtained from the load estimator, as proposed by Iwasaki [IWASASKI, M and MATSUI, N., IECON'91] and discussed in chapter 5. The output, Current Demand 1 is responsible mainly for dealing with the Speed Error, whereas Current Demand 2 is responsible for dealing with the load. Both are then added together to give the total current demand for the Proportional-Integral current controller. The fuzzy speed controlled dc motor block diagram is shown in Fig. 9.2.

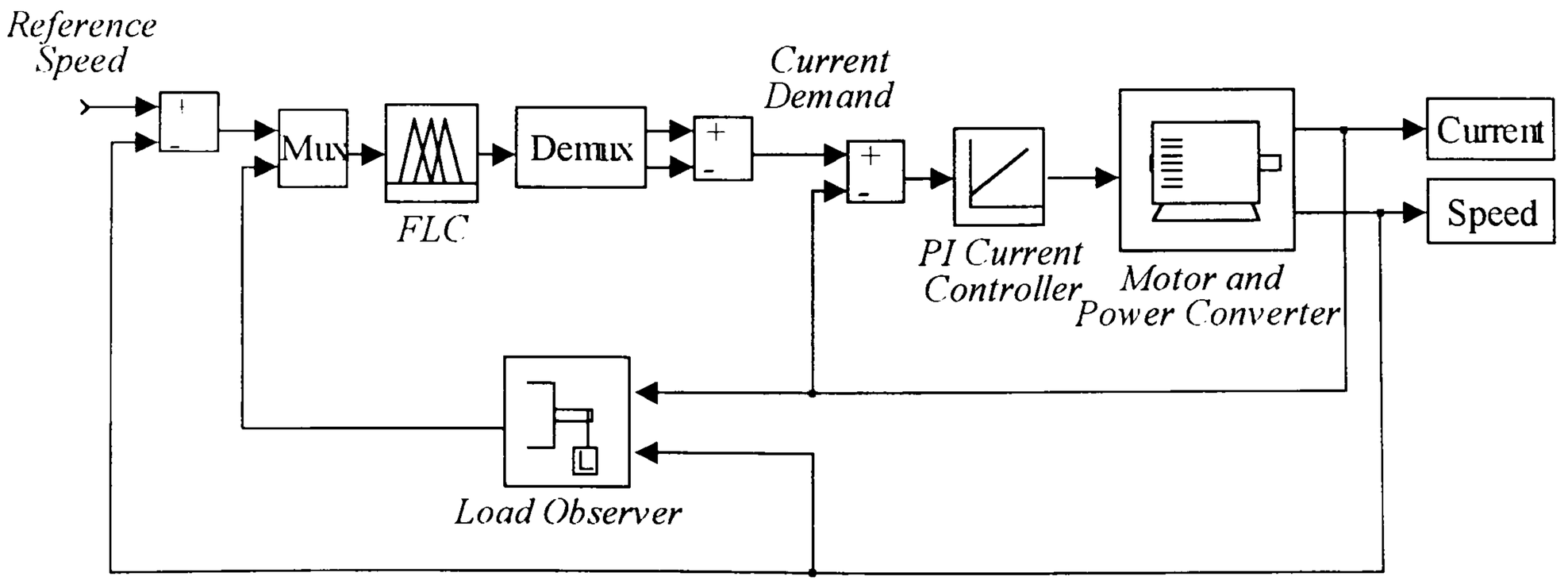



Fig. 9.2 - Fuzzy speed controlled dc motor with load estimator.

### 9.2.1 - The membership functions and rules

Each input and output of the fuzzy controller has trapezoidal shape membership functions as shown in Fig. 9.3. Any other form of membership functions could be chosen however, the trapezoidal one is simpler and requires less computational effort during the defuzzification process.

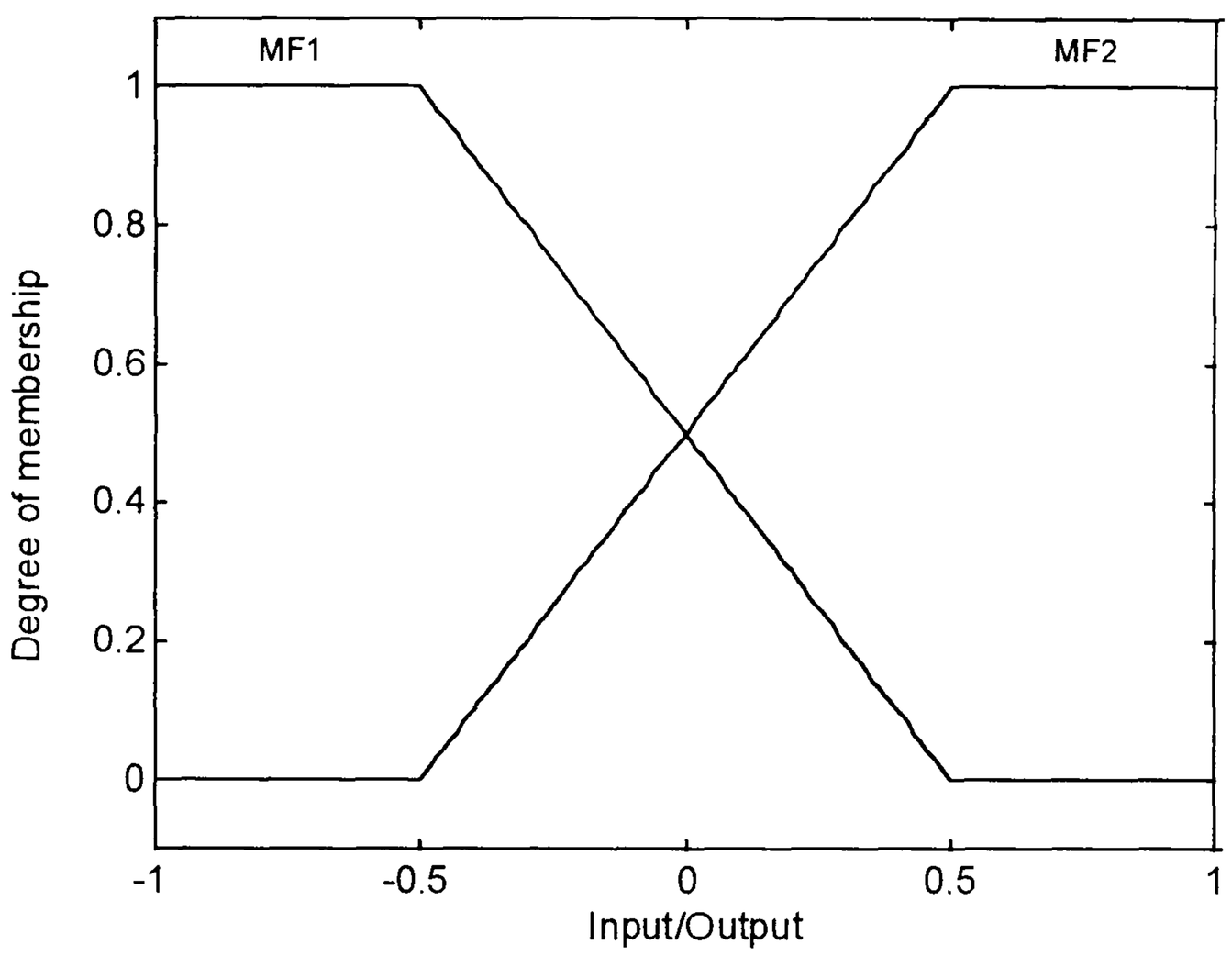



Fig. 9.3 - Fuzzy controller's input and output membership functions.

As there are 4 variables with two membership functions each, the total number of membership functions is 8. Four rules are used to map the inputs to the outputs as follow:

**IF** input(i) is MF(i) **then** Output(j) is MF (j)

For i, j = 1,2.

This fuzzy controller can supply current demand for any reference speed, regardless of load variation. The accuracy of the controller depends on the tuning of the membership functions within their universe of discourse.

## 9.3 - EXPERIMENTAL RESULTS

As the goal is to hold the speed regardless of load torque change, the fuzzy speed controller was tuned manually to control the speed at 1100 rpm in the presence of 62% load disturbance. The transfer functions are shown in Figs. 9.4 and 9.5.

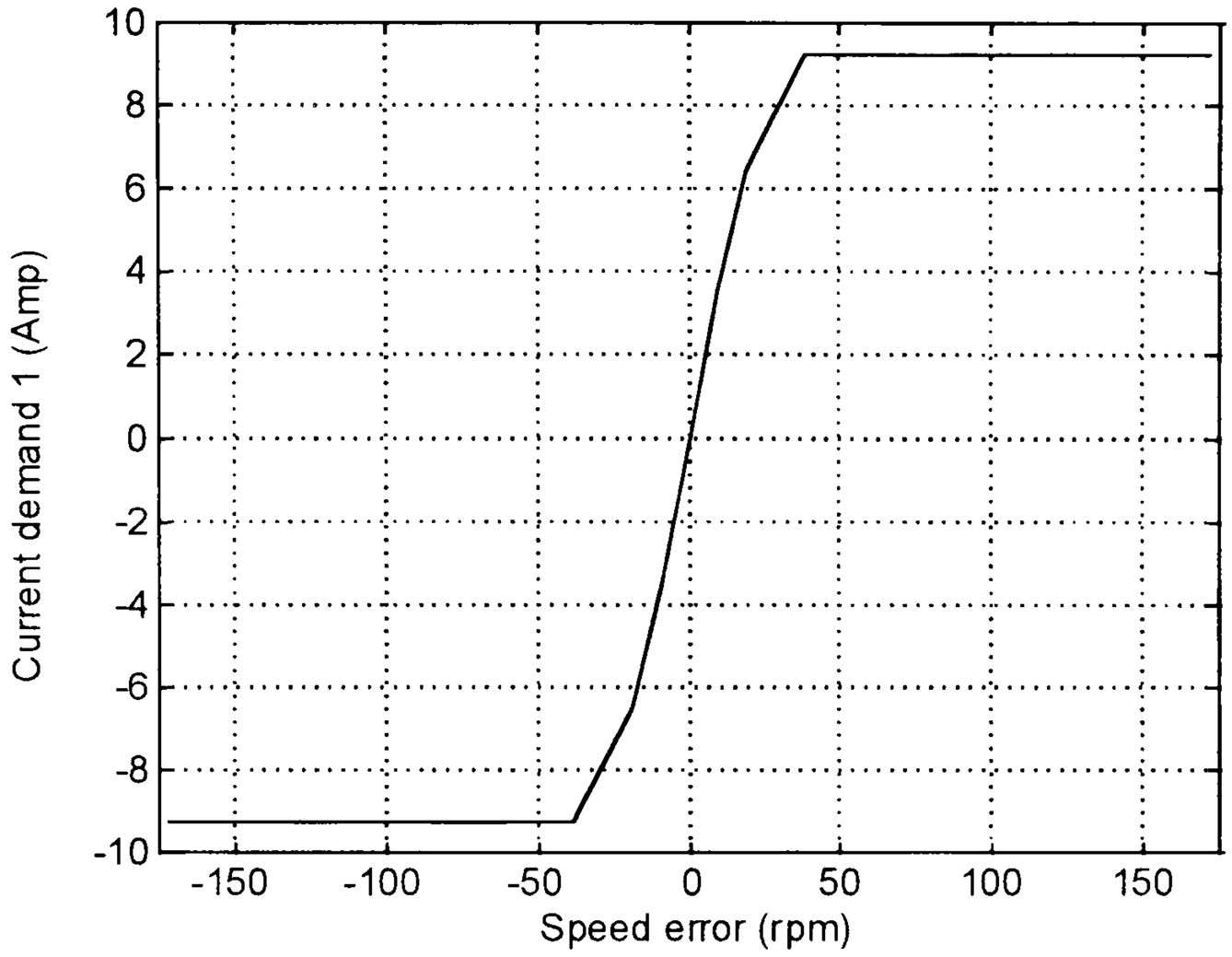



Fig. 9.4 - Transfer function between Input 1 (Speed Error) and Output 1 (Current Demand 1) of the fuzzy controller.

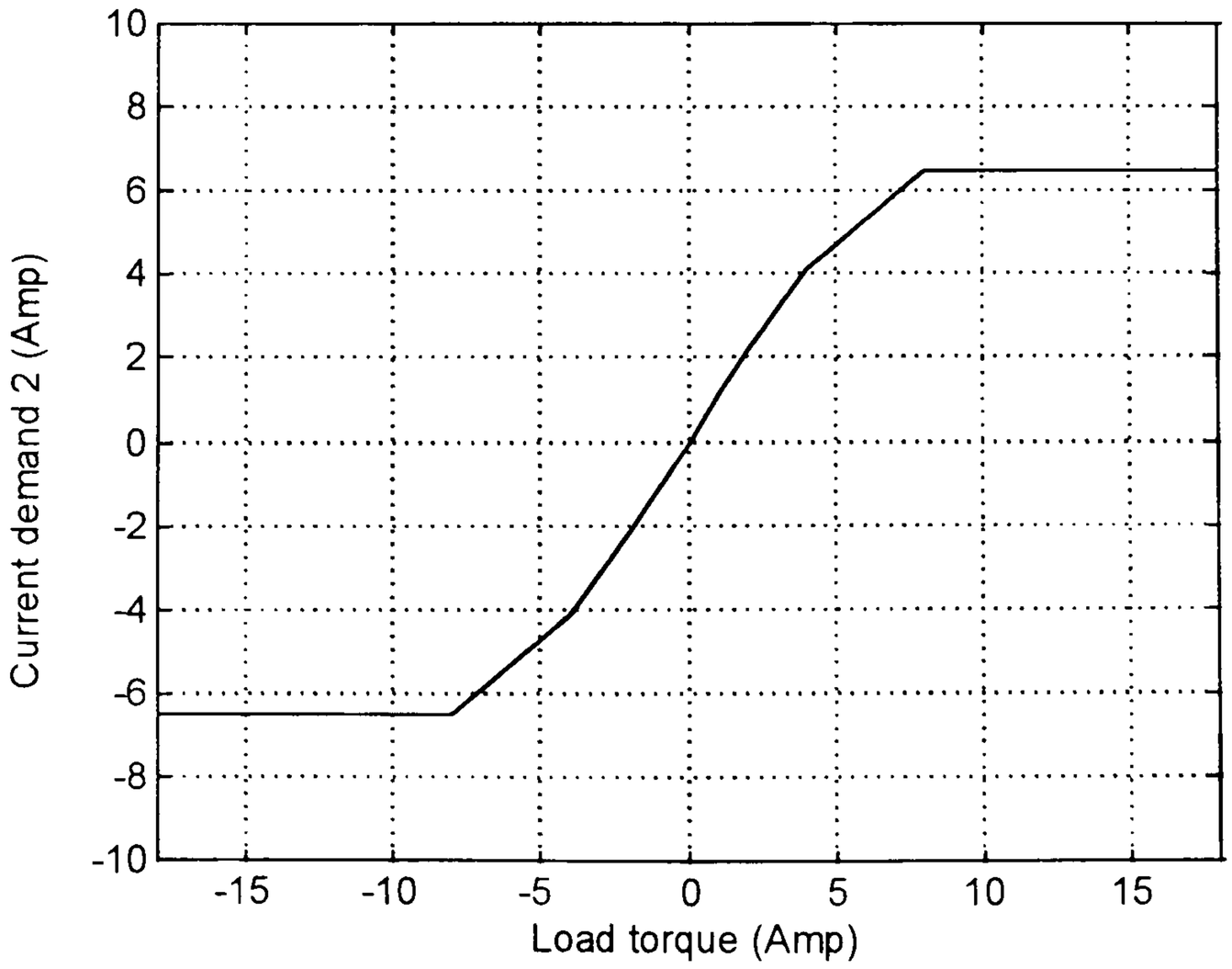



Fig. 9.5 - Transfer function between Input 2 (Load torque) and Output 2 (Current Demand 2) of the fuzzy controller.

In Fig. 9.4, the gain imposed by the fuzzy controller is equivalent to 0.36 A/rpm or 3.5 A/rad/s around zero steady state speed error. However, the maximum Current Demand 1 output is 9 A. This gain value is very close to that given by the Genetic Algorithm in section 5.4.3 for the proportional gain of the PI speed controller without load estimator (3.42 A/rad/s) and in section 6.3 with the load estimator, where the gain was (3.72 A/rad/s).

In Fig. 9.5, the gain imposed by the fuzzy controller is about 1 A/A which means that, for the estimated load equivalent to 1A, the same current is supplied by the fuzzy controller to the PI current controller.

For comparison purpose, the motor was driven under identical condition as discussed in section 6.3, where a PI speed controller was used together with the same load disturbance observer and optimised by using Genetic Algorithm. The motor was at steady state, running at 1100 rpm with 11% of the maximum load torque. At *time* = 1.1 s, a step input load disturbance equal to 62% of the maximum load torque was applied. The current increased from 1 to nearly 6A, increasing the torque produced by the motor in order to hold the speed. The speed and current response of the motor drive with the proposed fuzzy speed controller is shown in Fig. 9.6 and, in enlarged scale, in Fig. 9.7.

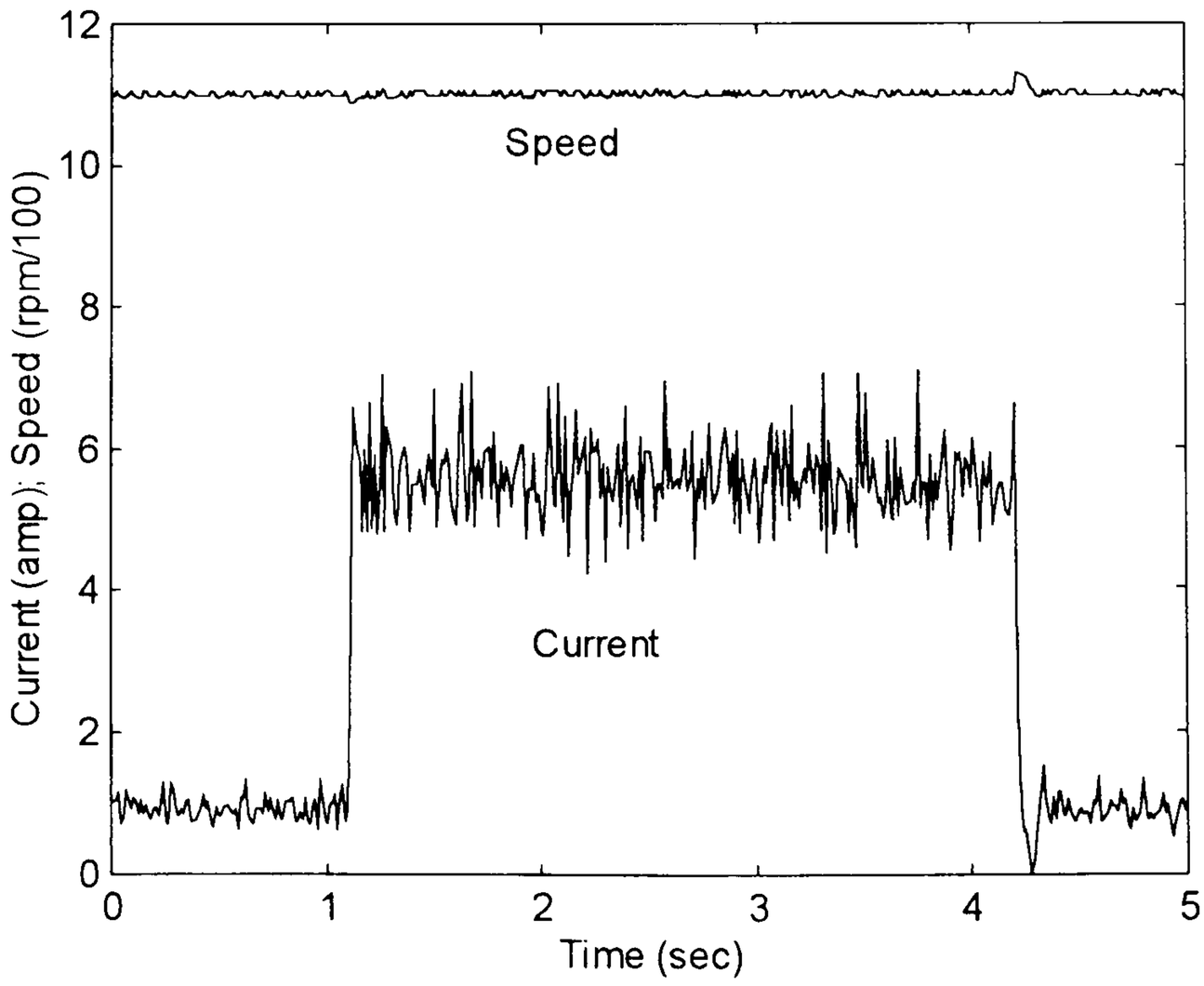



Fig. 9.6 - Speed and current response of the motor drive due to a 62% of the maximum load torque applied as a step input load disturbance.

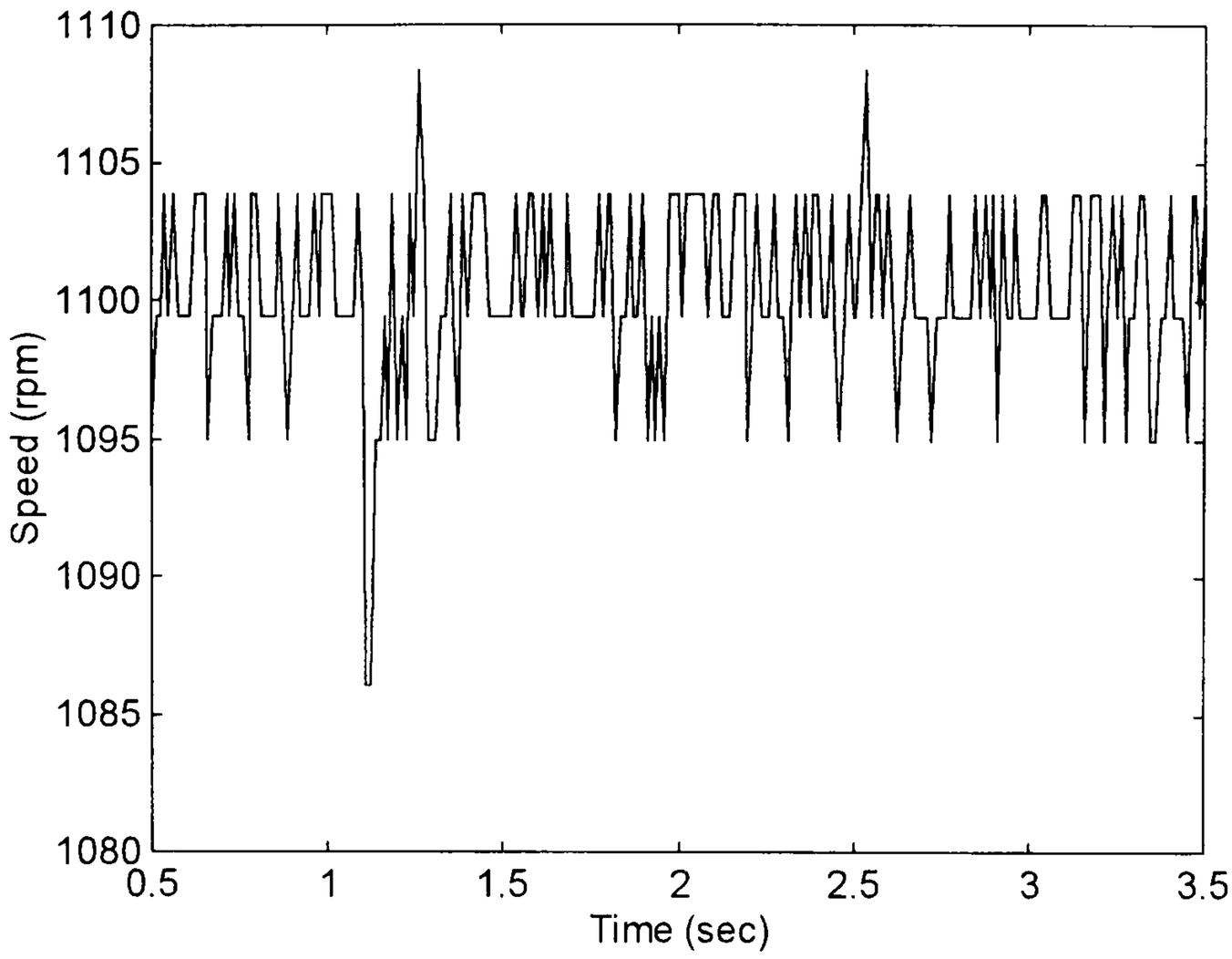



Fig. 9.7 - Speed response of the motor drive in enlarged scale showing the speed variation due to the load torque disturbance.

The ITAE measured during the disturbance only is equal to 0.74 rad. Despite the manual tuning, the performance obtained by using this fuzzy is as good as the best one obtained by using PI speed controller, together with the load estimator and optimised by GA as shown in section 6.3 - Fig. 6.11, for which the ITAE was 0.73 rad.

## 9.4 - ROBUSTNESS AGAINST LOAD VARIATION AND SPEED DEMAND

In order to test the robustness of the fuzzy speed controller due to change in the load as well as speed demand, it has been tested at different working condition. However, in order to be able to compare the performance of the fuzzy controller to the PI speed controller, identical conditions in terms of step input speed demand and load, as used for the PI controller in chapter 7, has to be ensured. It is important to draw attention to the fact that both controllers, the fuzzy as well as the PI, have been tuned for the best performance at constant speed (1100 rpm) in the presence of load variation.

In order to guarantee conformity of the test conditions and base for comparisons, identical step input speed demand and load variation as done in chapter 7 is used here.

### 9.4.1 - 1100 rpm with 11% load torque

Fig. 9.8 shows the speed and current response of the motor drive with the fuzzy controller due to 1100 rpm step input speed demand, with 11% of the maximum load torque. The ITAE measured for this speed response within 5 s time window was 25.68 rad, against 29.2 rad for the PI speed controller within identical condition.

As shown in Fig. 9.4, which represents the transfer function between the Speed Error and Current Demand 1, the controller would never go over the current limit, 9A. However, because the Estimated Load input and Current Demand 2 output have the transfer function shown in Fig. 9.5, the output of the controller may become saturated during the starting transient, because of the presence of the load observer. However, it does not cause the *windup* problem because there is no integrator at all. It can be seen that, as the controller was not tuned for the fastest speed transient response due to a step change in speed, the current does not increase instantaneously. The current increases slower then in the cases shown in chapter 7, representing in this case, slower acceleration. As the motor speeds up, the current reaches the limit. Nonetheless, it stays there for a shorter period of time, when compared to the speed responses shown in sections 7.2.1-9.

### 9.4.2 - 1100 rpm with 62% load torque

Fig. 9.9 shows the speed and current response of the motor drive with fuzzy controller at 1100 rpm, with 62% of the maximum load torque. The ITAE within this condition was 25.80 rad, against 31.8 rad for the PI speed controller.

### 9.4.3 - 1100 rpm in the presence of load disturbance

In this test, 1100 rpm step input speed demand was applied to the motor with 11% of load torque. At *time* = 1 s, a step input load disturbance equal to 62% of the maximum load was applied, lasting for the rest of the 5 s time window. The speed

and current response can be seen in Fig. 9.10. The ITAE was equal to 25.09 rad. For the PI speed controller within identical condition, the ITAE was 28.8 rad.

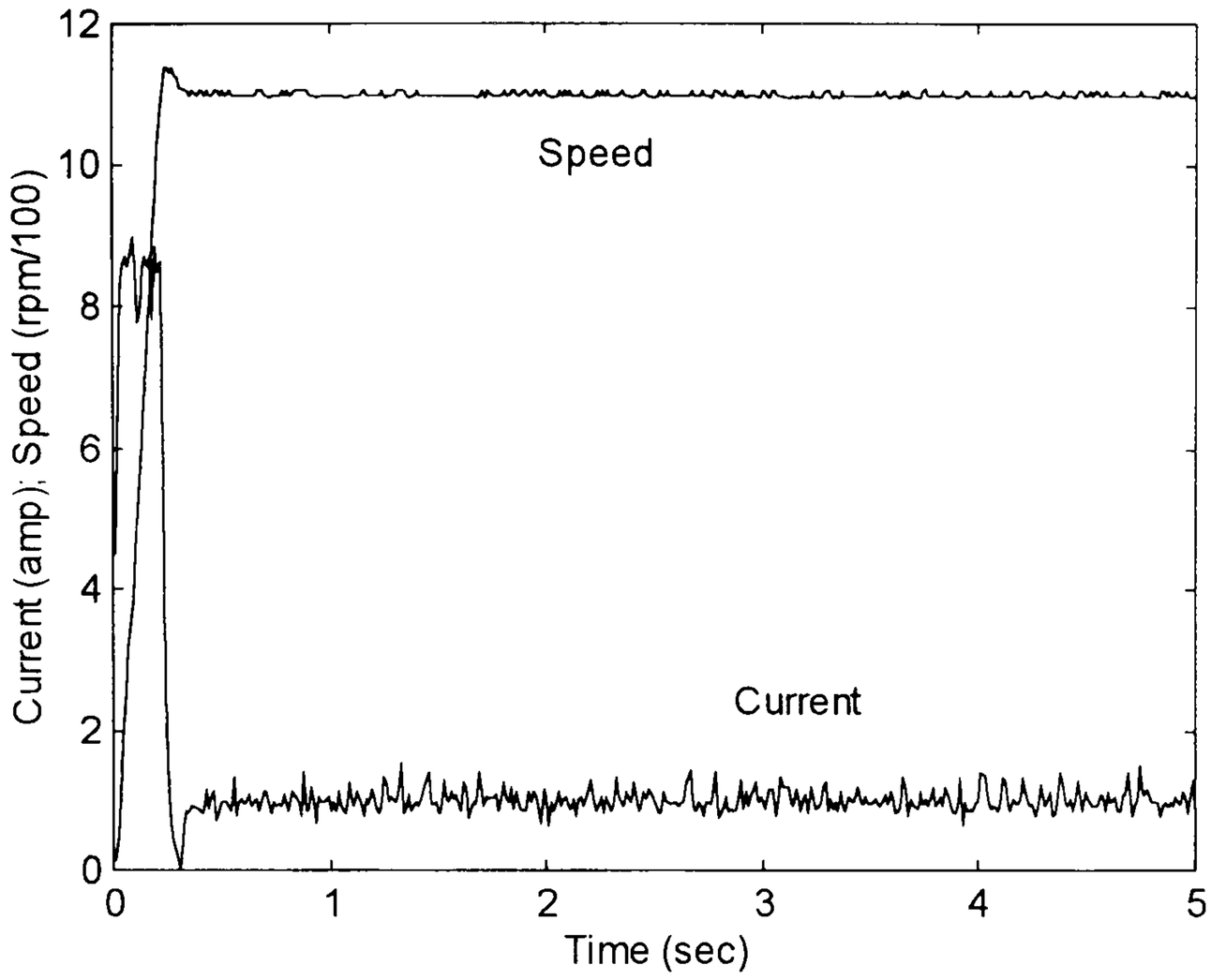



Fig. 9.8 - Speed and current response of the motor for 1100 rpm step input speed demand and 11% of the maximum torque produced by the motor.

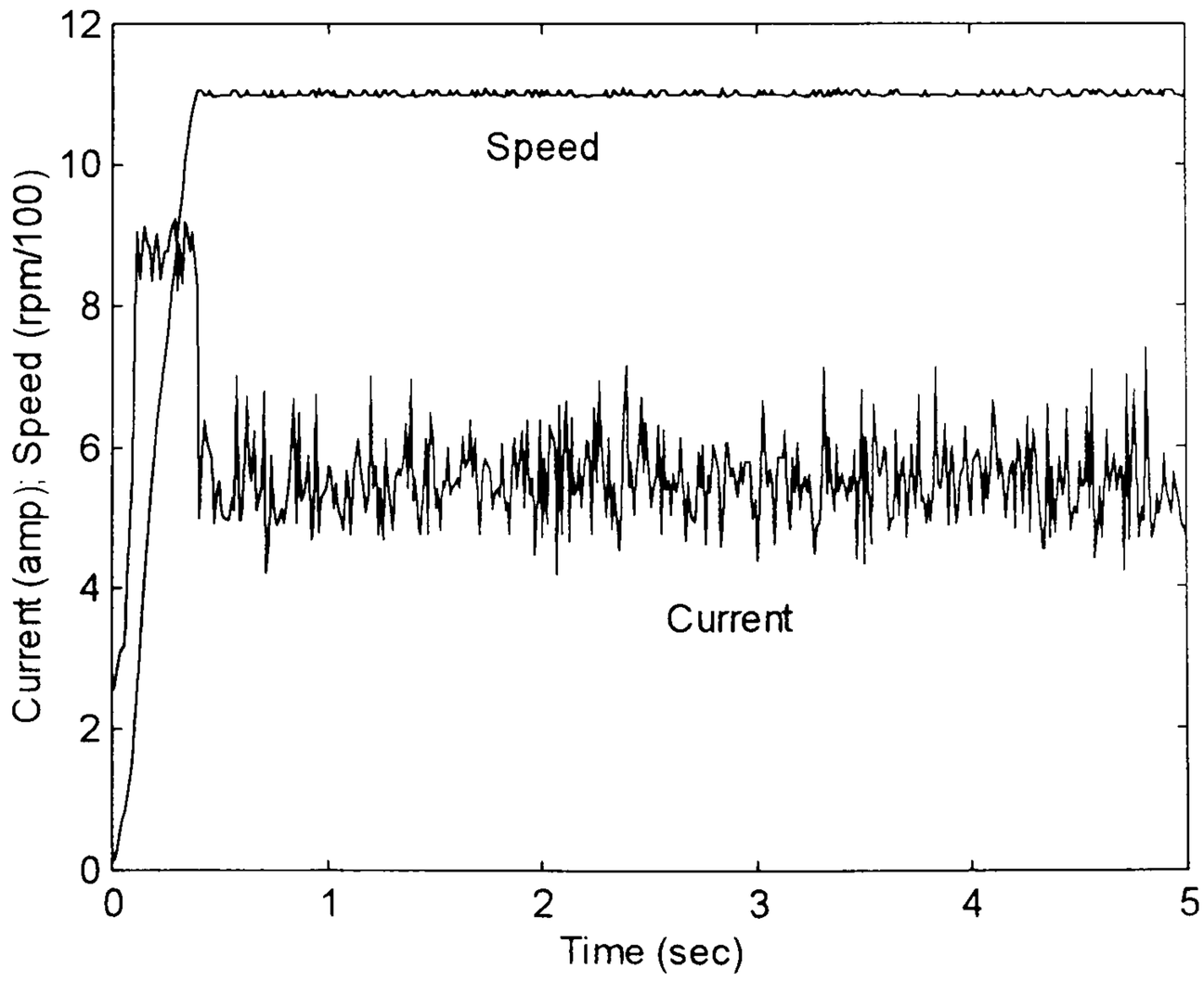



Fig. 9.9 - Speed and current response of the motor for 1100 rpm step input speed demand and 62% of the maximum torque the motor can produce.

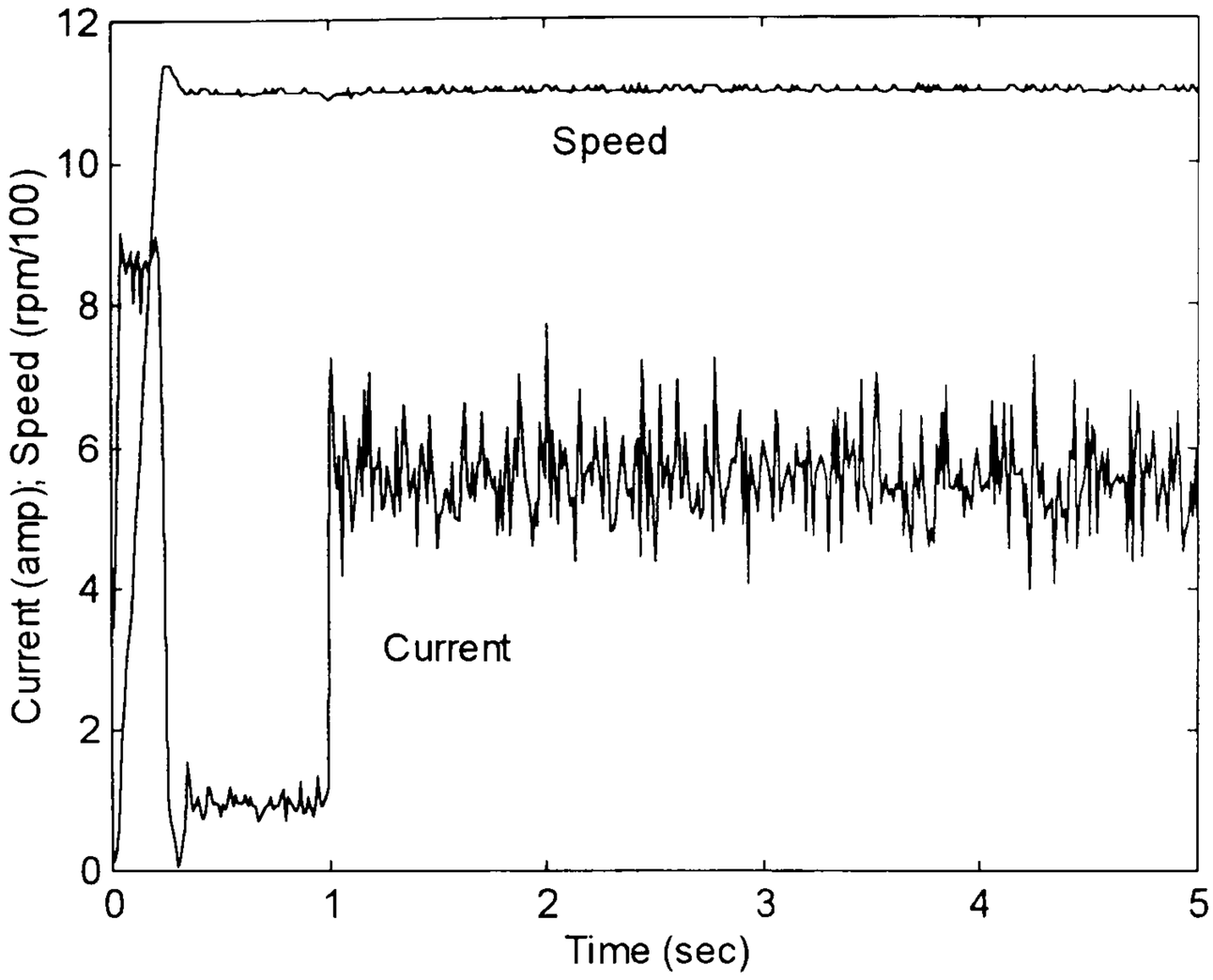



Fig. 9.10 - Speed and current response of the motor for 1100 rpm step input speed demand and 62% step input load increase applied at $t = 1$ s.

### 9.4.4 - 1500 rpm with 11% of maximum load torque

Fig. 9.11 depicts the speed and current response of the due to 1500 rpm step input speed demand, with 11% of the maximum load torque.

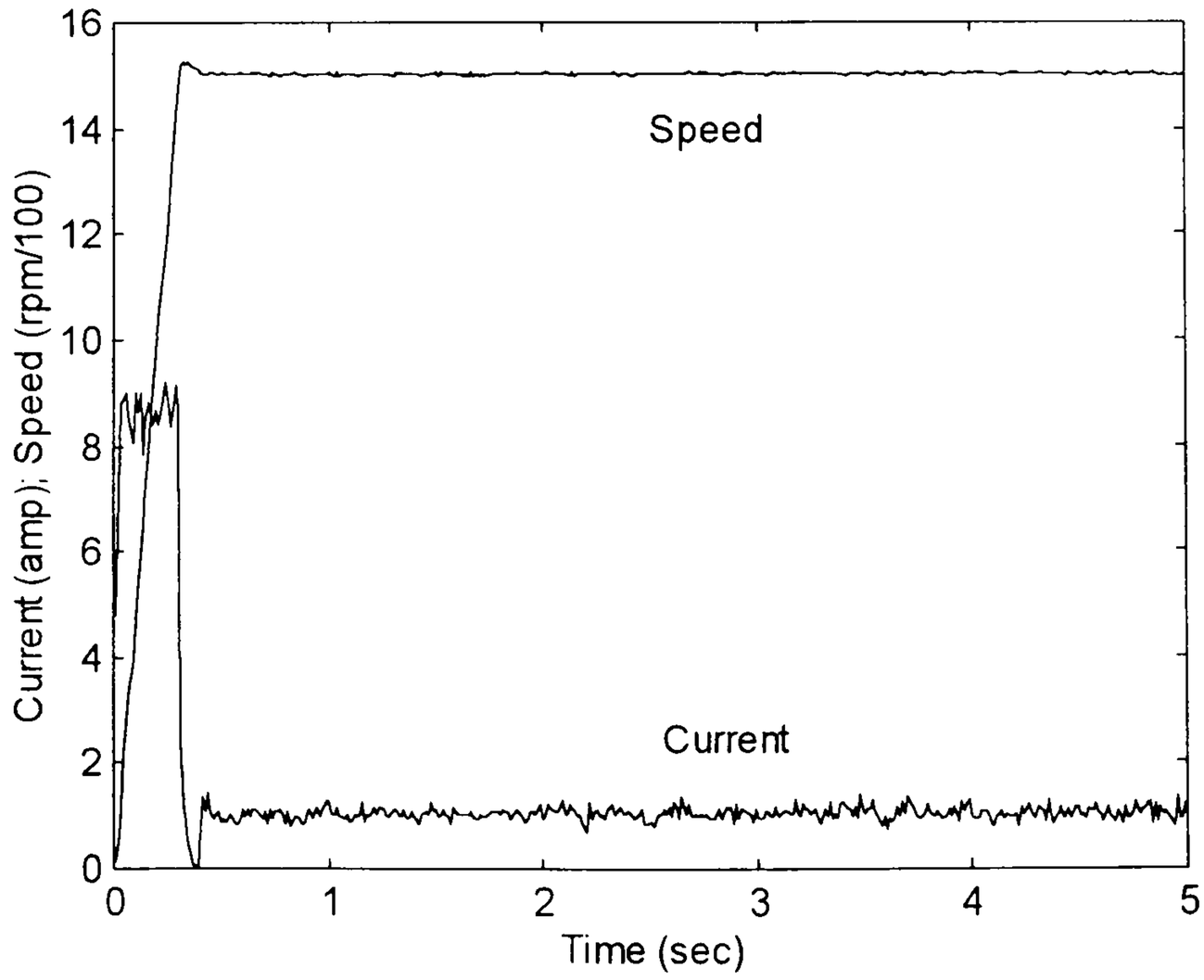



Fig. 9.11 - Speed and current response of the motor for 1500 rpm step input speed demand and 11% of the torque produced by the motor.

The ITAE measured for this speed response within 5 s time window was 43.23 rad, against 46.7 for the PI speed controller.

### 9.4.5 - 1500 rpm with 75% of maximum load torque

Fig. 9.12 shows the speed and current response of the motor drive sigh fuzzy controller at 1500 rpm, with 75% of the maximum torque the motor can produce. The ITAE for this condition within 5 s time window was 47.97 rad, against 64.2 rad for the PI speed controller.

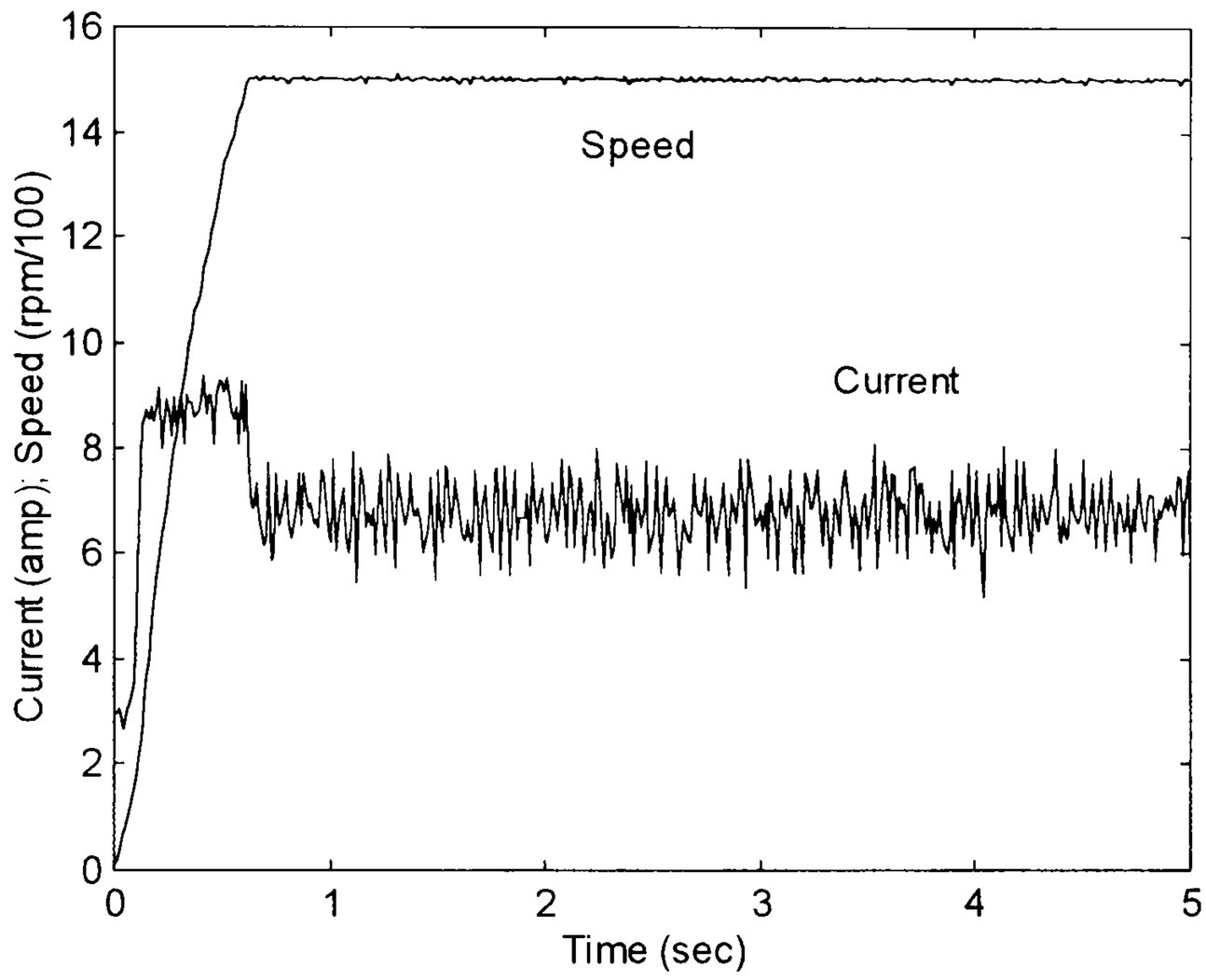



Fig. 9.12 - Speed and current response of the motor for 1100 rpm step input speed demand and 75% of the maximum torque.

### 9.4.6 - 1500 rpm in the presence of load disturbance

As done previously with different reference speed, a 1500 rpm step input speed demand was then applied to the motor with 11% of load torque. At time equal to 1 s, a step input load disturbance equal to 75% of the maximum load was applied, lasting for the rest of the 5 s time window. The speed and current response can be seen in Fig. 9.13. The ITAE within 5 s time window was 41.95 rad, against 72.3 rad for the PI speed controller.

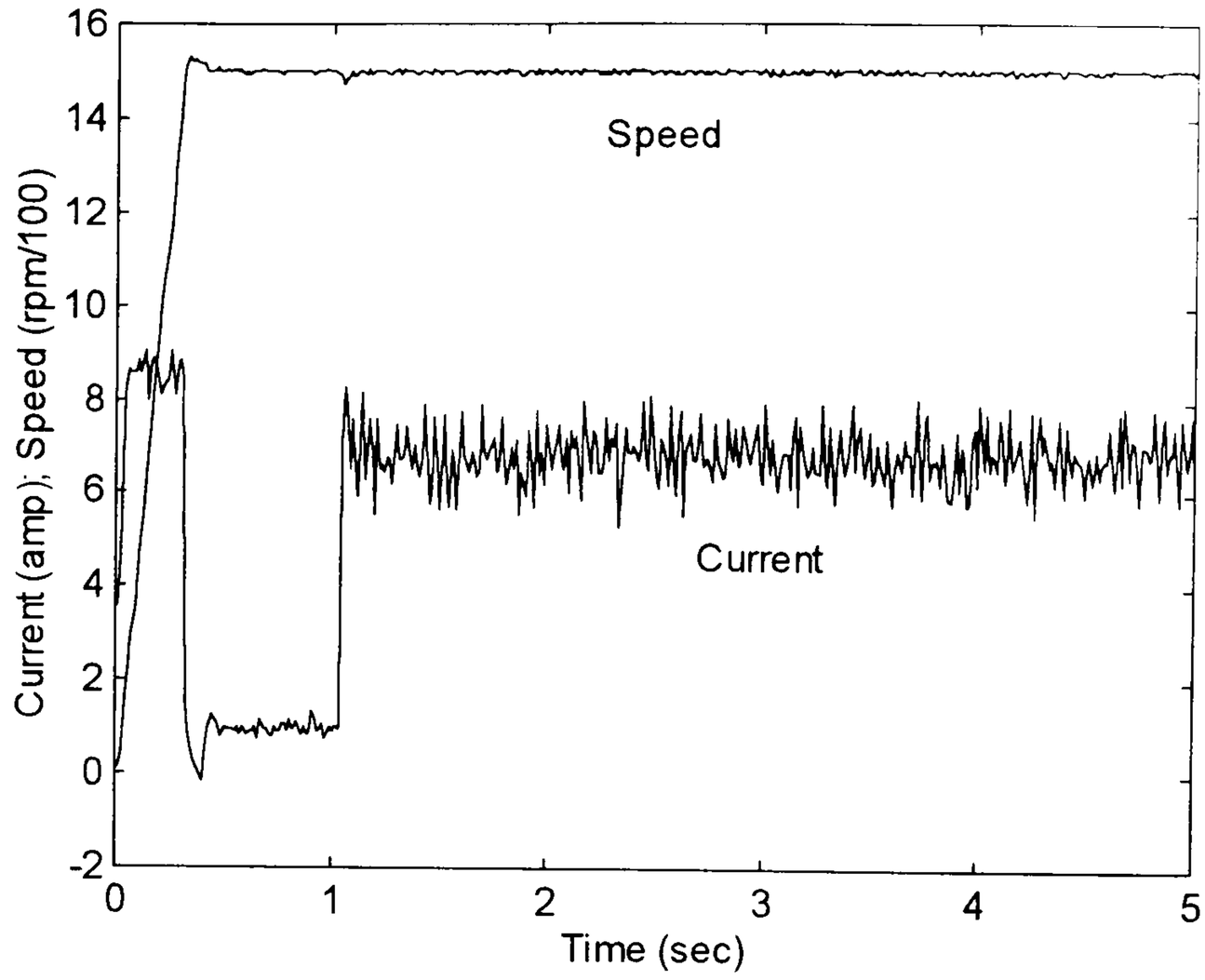



Fig. 9.13 - Speed and current response of the motor for 1500 rpm step input speed demand and 75% step input load disturbance applied at *time* = 1 s.

### 9.4.7 - 700 rpm with 11% load torque

Fig. 9.14 shows the speed and current response of the motor drive with fuzzy speed controller at 1100 rpm, with 9% of the maximum load torque.

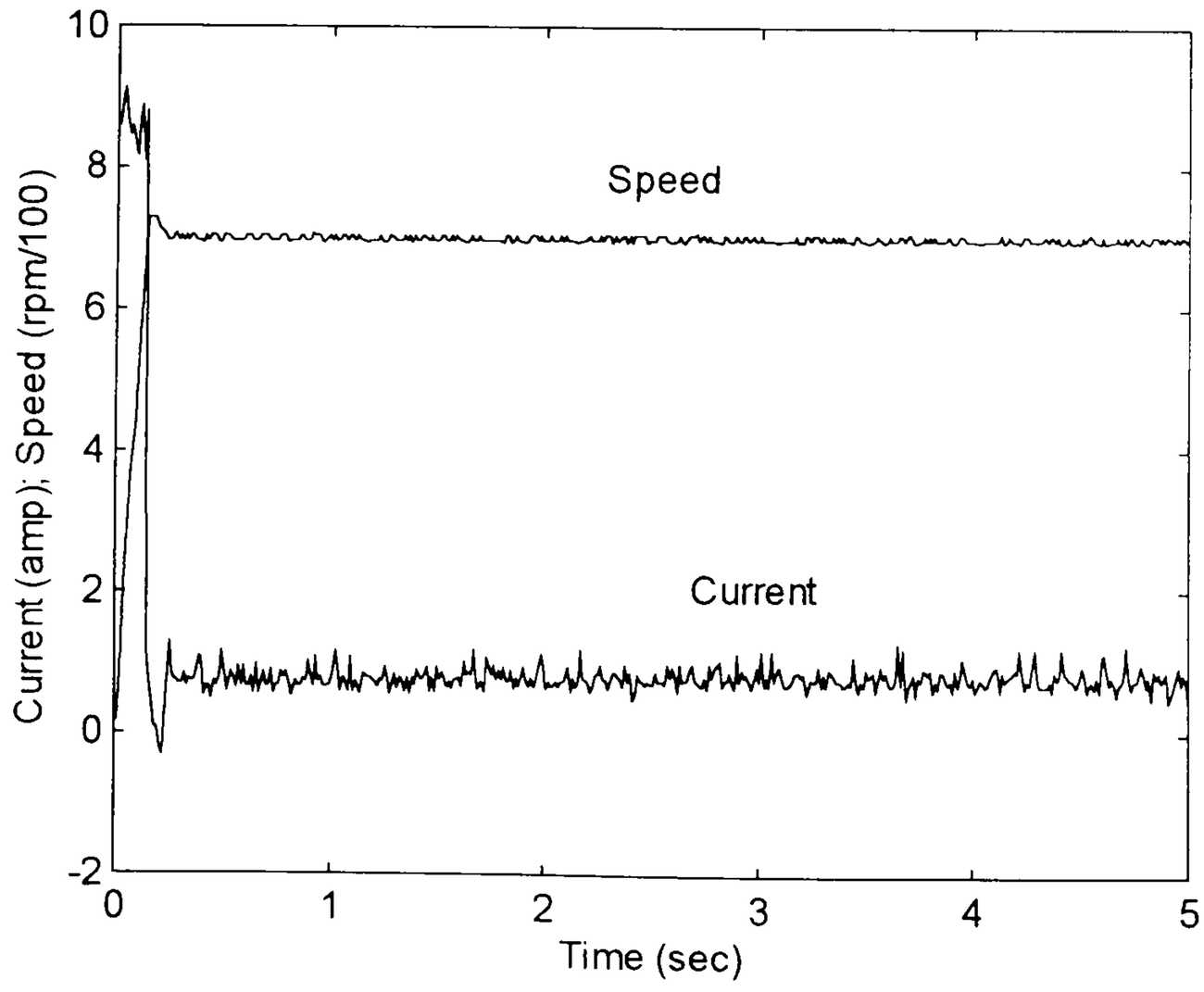



Fig. 9.14 - Speed and current response of the motor for 700 rpm step input speed demand and 9% of the maximum load torque.

The ITAE measured for this speed response within 5 s time window was 14.55 rad. Within the same condition, the ITAE for the motor drive with PI speed controller was 14.9 rad.

### 9.4.8 - 700 rpm in the presence of 55% load torque

Fig. 9.15 shows the speed and current response of the motor drive with fuzzy controller at 700 rpm, with 55% of the maximum load torque. The ITAE was 11.32 rad, against 12.7 for the PI speed controller.

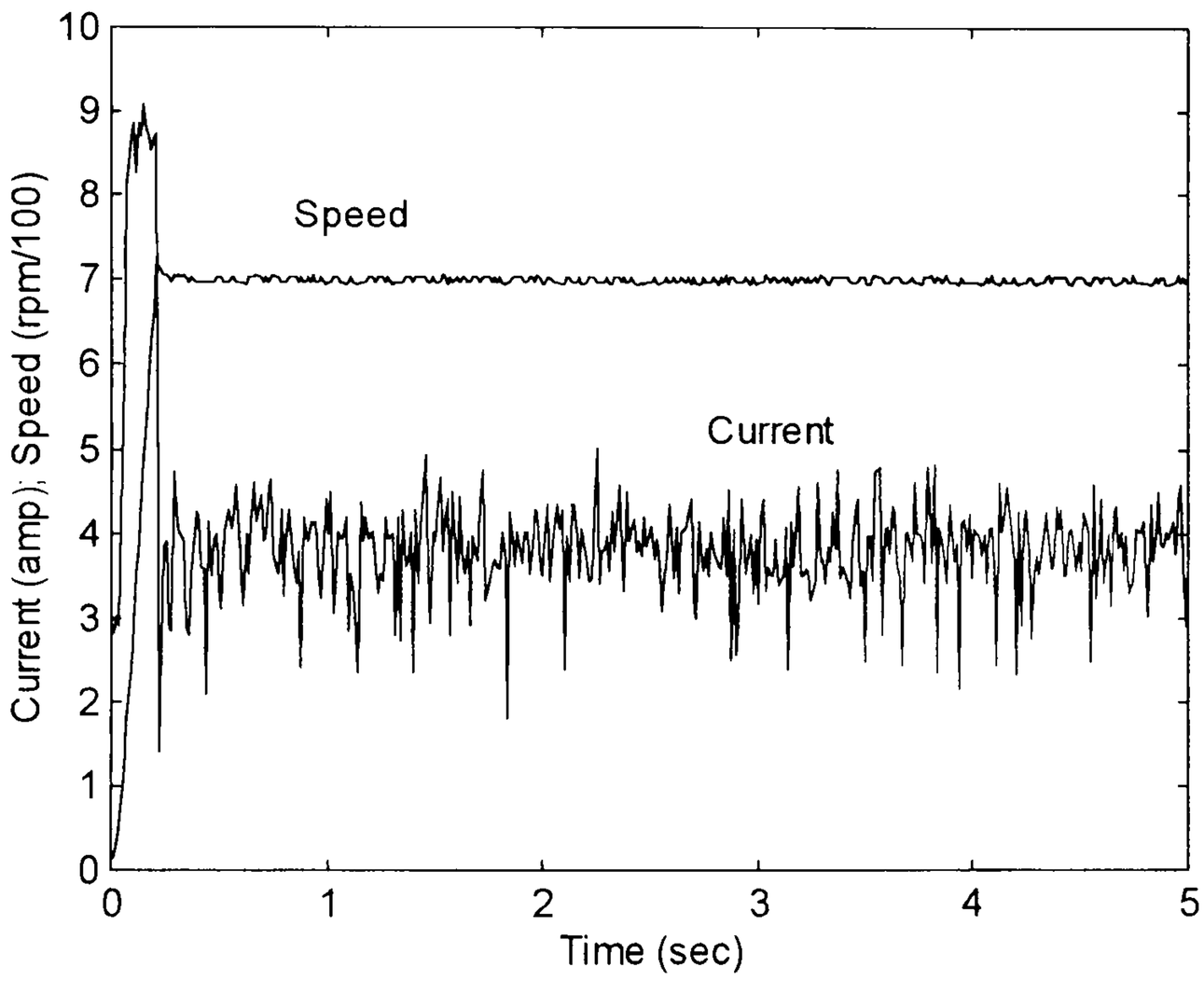



Fig. 9.15 - Speed and current response of the motor for 700 rpm step input speed demand and 55% of the maximum load torque.

### 9.4.9 - 700 rpm in the presence of load disturbance

Similarly to what was shown previously in section 7.2.9, a 700 rpm step input speed demand was applied to the motor with 9% of load torque. At time equal to 1 s, a step input load disturbance of 55% of the maximum load was applied, lasting 4 s. The speed and current response can be seen in Fig. 9.16. The ITAE within 5 s time window was 13.49 rad, against 15.6 for the PI speed controller.

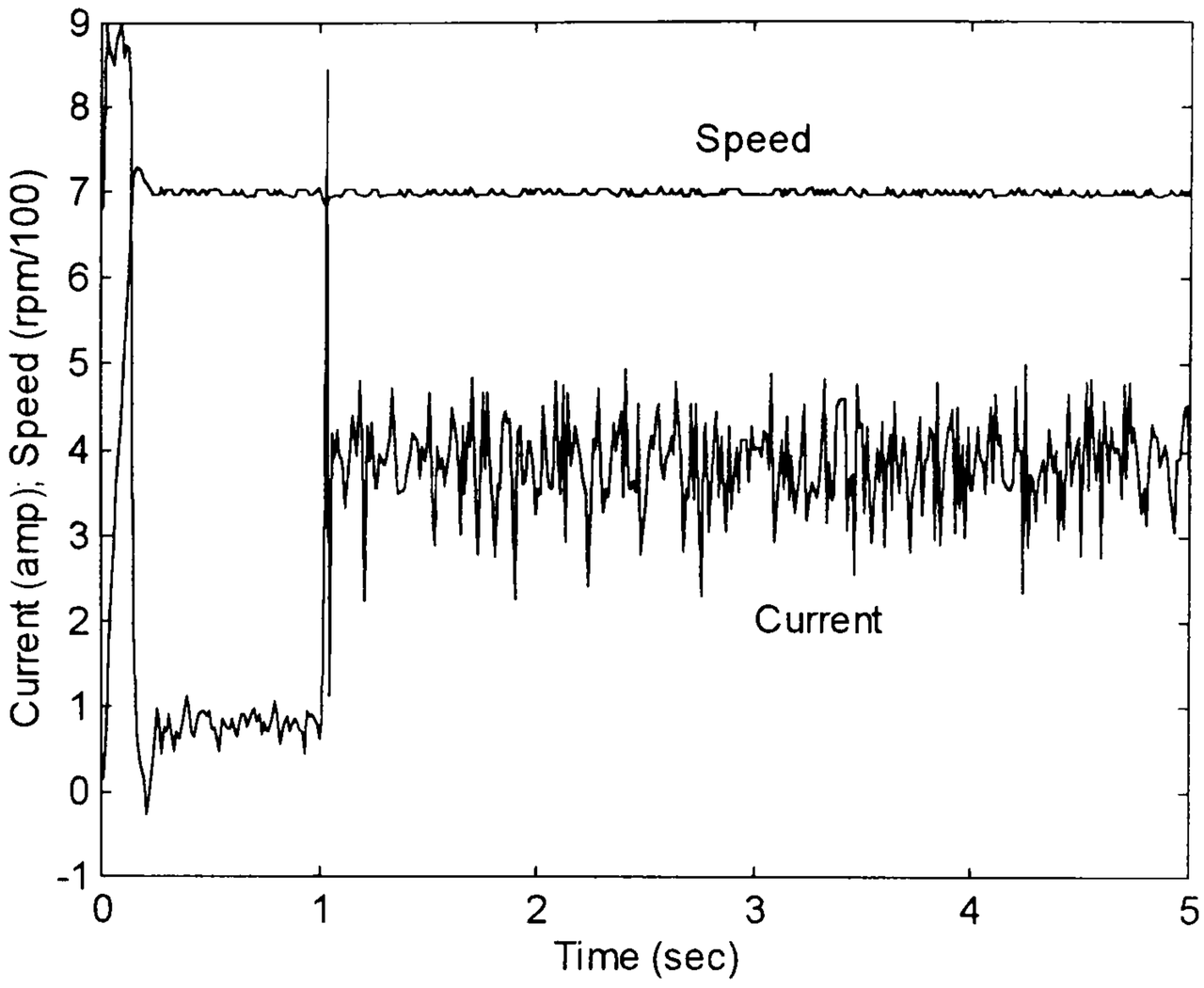



Fig. 9.16 - Speed and current response of the motor for 700 rpm step input speed demand and 55% step input load disturbance applied at *time* = 1 s.

### 9.4.10 - Summary of the results on robustness against load variation

In the same way as previously done in chapter 7, a table summarising the results is presented here. Table 9.1 exposes the results of the tests done within section 9.4.1-9. At the same time, it displays the results of the ITAE obtained in chapter 7, for the PI speed controller, so comparisons can be made.

Despite all the changes in speed demand and load, the responses obtained by using the fuzzy speed controller do not change significantly. For the same step input speed demand, the presence of load torque can only affect the acceleration of the motor drive towards the reference speed.

The quality of each speed response, obtained by using fuzzy controller, has also been assessed by the ITAE of each run. These results are compared to those obtained by using PI speed controller together with load disturbance observer, as discussed earlier in chapter 7. In all the tests, the ITAE of each speed response with the fuzzy controller is smaller then those obtained with the PI. It means that the fuzzy controller is more

robust against load variation than the PI one. When running at constant speed, the high gain setting of the PI speed controller given by on-line tuning by using Genetic Algorithm, together with the load estimator, makes it respond fast against load disturbance. The fuzzy controller tuned manually could match the performance of the PI speed controller. However, in the next chapter, a fuzzy speed controller optimised on-line by Genetic Algorithm is presented and its performance evaluated.

| STEP INPUT SPEED DEMAND | LOAD CONDITION (% of max torque produce by the motor) | ITAE WITH FUZZY SPEED CONTROLLER (rad) | ITAE WITH PI SPEED CONTROLLER (rad) |
|---|---|---|---|
| **700 rpm** | 11% | 14.5 | 14.9 |
| | 55% | 11.3 | 12.7 |
| | Load change to 55% | 14.5 | 15.6 |
| **1100 rpm** | 11% | 25.7 | 29.2 |
| | 62% | 25.8 | 31.8 |
| | Load change to 62% | 25.1 | 28.8 |
| **1500 rpm** | 11% | 43.2 | 46.7 |
| | 75% | 48.0 | 64.2 |
| | Load change to 62% | 41.9 | 72.3 |

Table 9.1 - Summary of the quality of the speed response obtained by using the fuzzy speed controller compared to the PI one, at different condition, using load estimator.

## 9.6 - CONCLUSION

In this chapter, a two-input two-output fuzzy speed controller has been discussed. It has been shown that this fuzzy controller is capable of controlling motor at different speed demand and load.

The main difference between the PI and fuzzy speed controller comes to the integrator in the PI one. Unlike the PI, fuzzy controller works by looking at its input and applying appropriate output. The integrator of the PI controller is essential for zero steady state speed error. On a simple PI speed control Brushless DC Motor Drive, it holds a "history" of the speed response to be able to give the appropriate current demand for the current controller. However, the integrator can not "see" when the output of the controller reaches saturation and keeps on integrating the speed error, causing the integrator *windup*.

Fuzzy control is different, as there is no integrator at all. As a consequence, there is no "history" whatsoever about the speed response. It simply applies the appropriate output value for each input. One important feature of a fuzzy controller is that, it is possible to shape its transfer function to accommodate any non-linearity of the system under control. This feature is not found in a PI controller.

# CHAPTER 10

# GENETIC ALGORITHM APPLIED TO FUZZY CONTROLLER

## 10.1 - INTRODUCTION

It has been shown in chapter 9 that the use of a fuzzy controller can bring improvements to the speed response performance of the Brushless DC Motor Drive, when compared to the PI one. However, in order to obtain best performance, good tuning is essential, as it is for all types of controllers for drive application. However, in a fuzzy controller the designer can define the number of inputs, outputs, membership functions and rules, so there are multiple degrees of freedom. As a consequence, human experts are not expected to be capable of finding the best tuning for the controller.

In this chapter, a Genetic Algorithm is used for the on-line optimisation of the fuzzy speed controller as defined in chapter 9. Tests have been made to illustrate the capability of the GA in finding the best setting for one specific condition.

## 10.2 – THE USE OF A GENETIC ALGORITHM FOR A FUZZY CONTROLLER

The starting point in creating a fuzzy controller is to define the input and output variables. After creating the controller variables, comes the partition of the range of each variable in terms of fuzzy sets by using membership functions. Next comes the definition of the rules used to map the input(s) into the output(s). In many cases, the optimisation of the membership functions of the controller only is sufficient to obtain

satisfactory performance, as done by Karr [KAR, 1991], Tsang [TANG, *et al*, 1994], Chang [CHANG et al, 1995] and Meredith [MERIDITH *et al*, 1992]. However, other researchers have gone further by optimising the membership functions and rules used to associate them to one another [LEE et al, 1993 and HOMAIFAR et al, 1992]. There were even cases where another step forward has been given and the researchers used a GA to find also the minimum number of membership functions and rules [TANG *et al*, 1998].

In the fuzzy speed controller as discussed in chapters 8 and 9 the number of membership functions per variable is the minimum (2 per variable). Furthermore, the number of rules necessary to map the inputs to the outputs of the fuzzy controller shown in chapter 9 is the minimum. Then, the membership functions are the only parameters left to be optimised, as it has been shown that, despite the modest number of membership functions and rules, the controller could provide good performance.

An example of the utilisation of the GA on a fuzzy controller is given by applying it to the optimisation of the two-input two-output fuzzy speed controller discussed previously in chapter 9. The fuzzy controller has two membership functions per variable, as shown in Fig. 9.3. The performance of the controller is related to the location of the marked corners of the membership functions, as shown in Fig. 10.1. It has to be ensured that a crossover point exist between the two membership functions otherwise, there could be a region of the fuzzy set where a certain input or output would not belong to either of the membership functions. As a consequence, the negative side corner of each membership function can only vary between 0% and -100% whereas the positive one varies from 0% to 100%.

There are two inputs and two output variables in the fuzzy controller, each one with two membership functions. For each membership function, two key points (two corners for each membership function) have to be properly located within the universe of discourse, so there are sixteen points to be optimised by Genetic Algorithm. By using real value chromosomes, as in chapter 5, each chromosome or individual has to be sixteen characters long. It means that sixteen parameters have to be optimised in this fuzzy controller, against only two for the PI. As a consequence, the degree of

difficulty for the Genetic Algorithm is a lot higher for the fuzzy controller than it was for the PI.

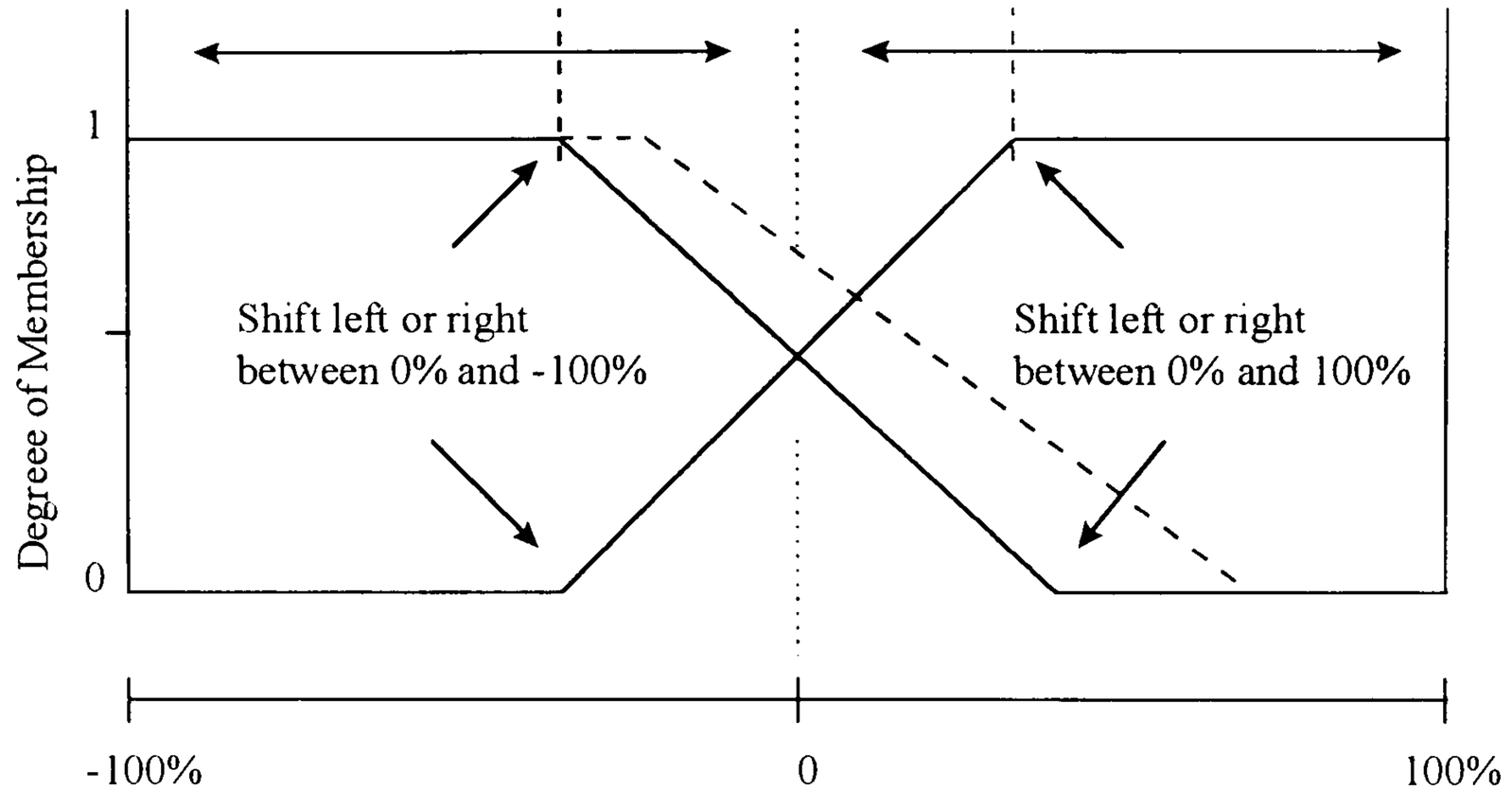



Fig. 10.1 - Illustration of the way the membership functions of the fuzzy speed controller have to be adjusted.

## 10.3 - DEFINITION OF THE RANGE OF EACH MEMBERSHIP FUNCTION

The inputs of the above fuzzy speed controller are 'Speed error' and 'Est. Load', as presented in chapter 9. Although the motor was to be run only from 0 rad/s to 200 rad/s, the range of 'Speed error' was initially set to vary from -200 rad/s to 200 rad/s, which represents a range from -100% to 100%, as explained in chapter 6. The same range (-100% to 100%) was set for all other variables. This particular range corresponds to -22A to 22A for the 'Est. Load' input and 'Curr. Demand 1' variables and from -15A to 15A for the 'Curr. Demand 2' output. An illustration of what the chromosomes look like is given below:

$$\text{Chrom} = [\underbrace{a_1 \ldots a_4}_{\text{Input 1}} \quad \underbrace{b_1 \ldots b_4}_{\text{Input 2}} \quad \underbrace{c_1 \ldots c_4}_{\text{Output 1}} \quad \underbrace{d_1 \ldots d_4}_{\text{Output 2}}]$$

The Integral with respect to Time of Absolute speed Error - ITAE mentioned in chapters 5 and 6 was used to evaluate the quality of the speed response of the

Brushless DC Motor Drive obtained by the fuzzy speed controller given by each chromosome.

## 10.4 - EXPERIMENTAL RESULTS

In order to be able to compare the evolution of the Genetic Algorithm for the optimisation of the fuzzy speed controller, identical parameters in terms of population size, generation number, mutation probability and generation gap, used in chapter 5 for the PI controller, were used here. The results are as follows.

### 10.4.1 - Optimisation for step input speed demand in the presence of load variation

An 1100 rpm step input speed demand was applied to the Brushless DC Motor Drivee from rest, with a load torque equal to 11% of the maximum torque the motor can produce. At *time* = 1 s, the load torque was suddenly increased to 62% at 100 rpm. The Genetic Algorithm should find the optimal tuning to speed up the motor as fast as possible while holding the steady state speed at the demand value. The evolution of Genetic Algorithm through 20 generations is shown in Fig. 10.2, which took place in about 3 hours and 40 min. Fig. 2a presents some speed responses obtained within the initial population. The ITAE of the speed response with the best chromosome of the first generation was 40.6 rad. At the end of the optimisation process, the best chromosome produced an ITAE of 26.9 rad. It is apparent that the evolution was considerable from the first to the last generation. However, there was still occurring evolution of the chromosomes when the algorithm was stopped. It means that the size of the population or the generation number was too small for such large number of parameters being optimised. In addition to this, there was an extra difficulty in the problem caused by the load torque. Because of the way fuzzy logic works, the actual speed could vary significantly from the reference value during the optimisation process and consequently, the load torque. The reason is that the load torque is dependent upon the voltage generated by the generator coupled to the Brushless DC Motor Drive. As the actual speed of the motor in each run can differ significantly

from the run before, the load torque was also different. Despite this effect, the Genetic Algorithm has proved capable of learning from previous generations and converges towards the best solution to the problem.

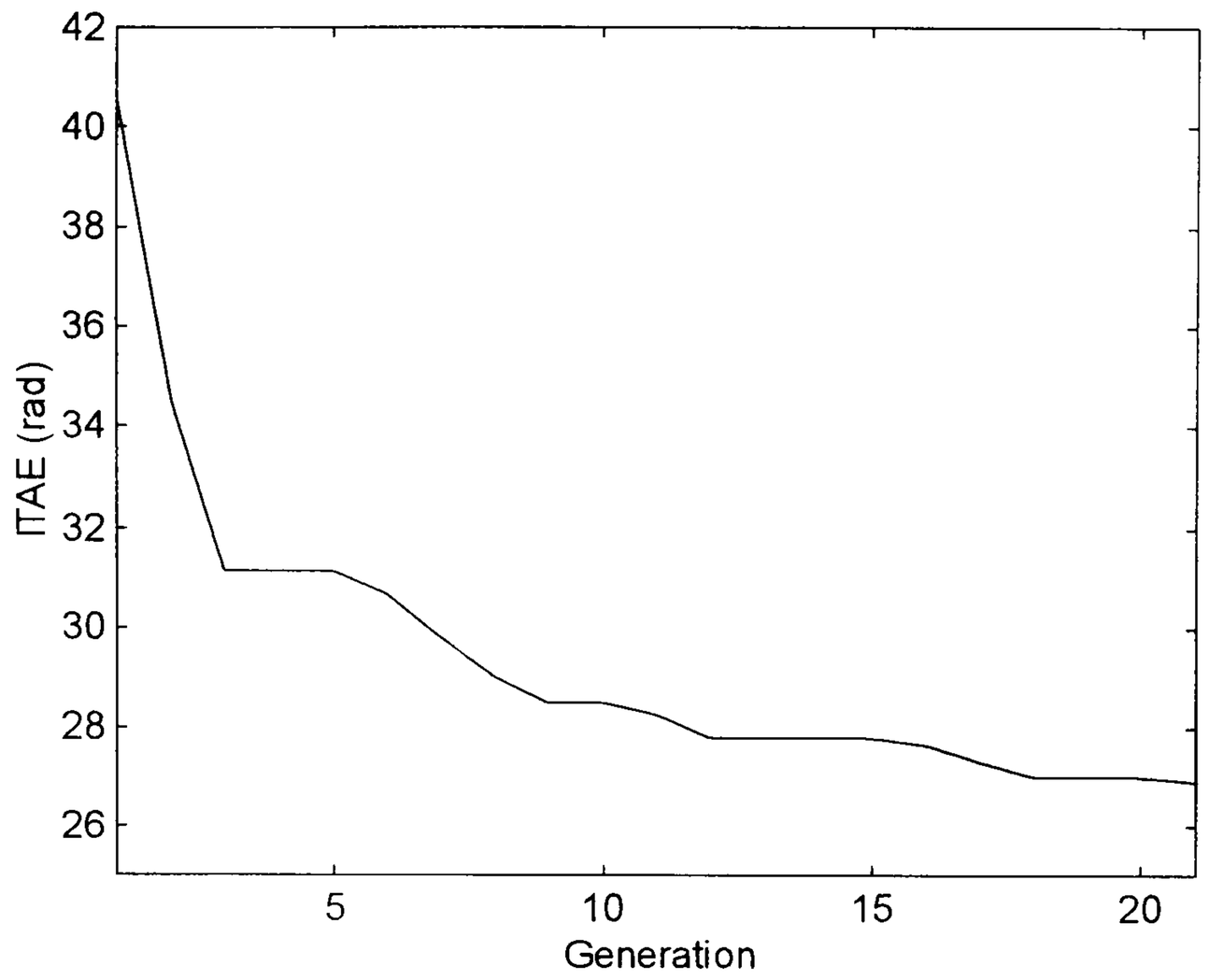



Fig. 10.2 - Evolution of the Genetic Algorithm optimisation process for the fuzzy speed controller.

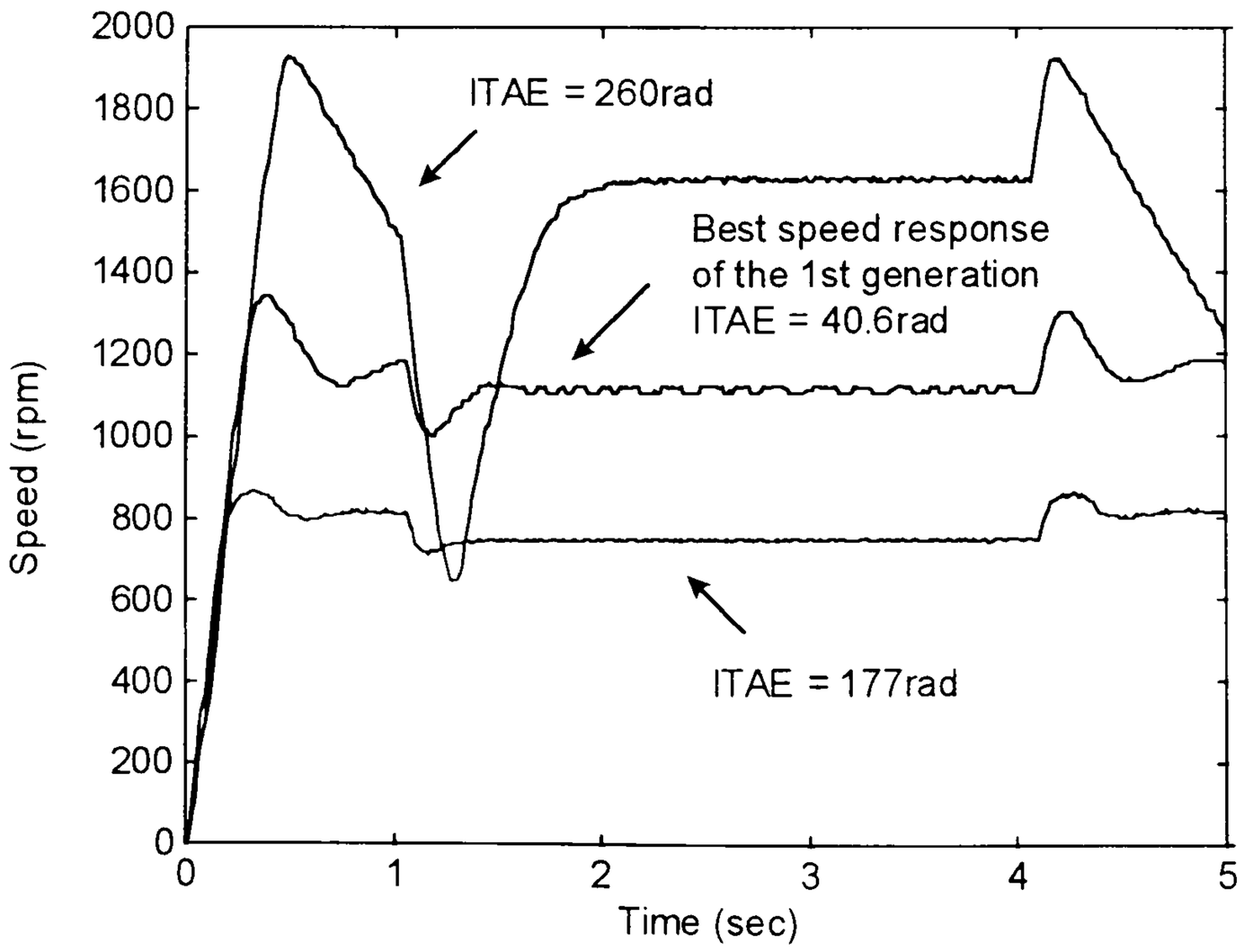



Fig. 10.2a - A few speed responses obtained with the fuzzy controller given by some chromosomes of the first population.

Fig. 10.3 presents the best speed and current response of the motor drive obtained at the end of the optimisation process. There can be seen a large steady state speed error during the time the load torque was only 11% of the maximum torque produced by the motor, although the speed error was "zero" between 1 s and 3 s, when the load torque was 62% of the maximum torque produced by the motor.

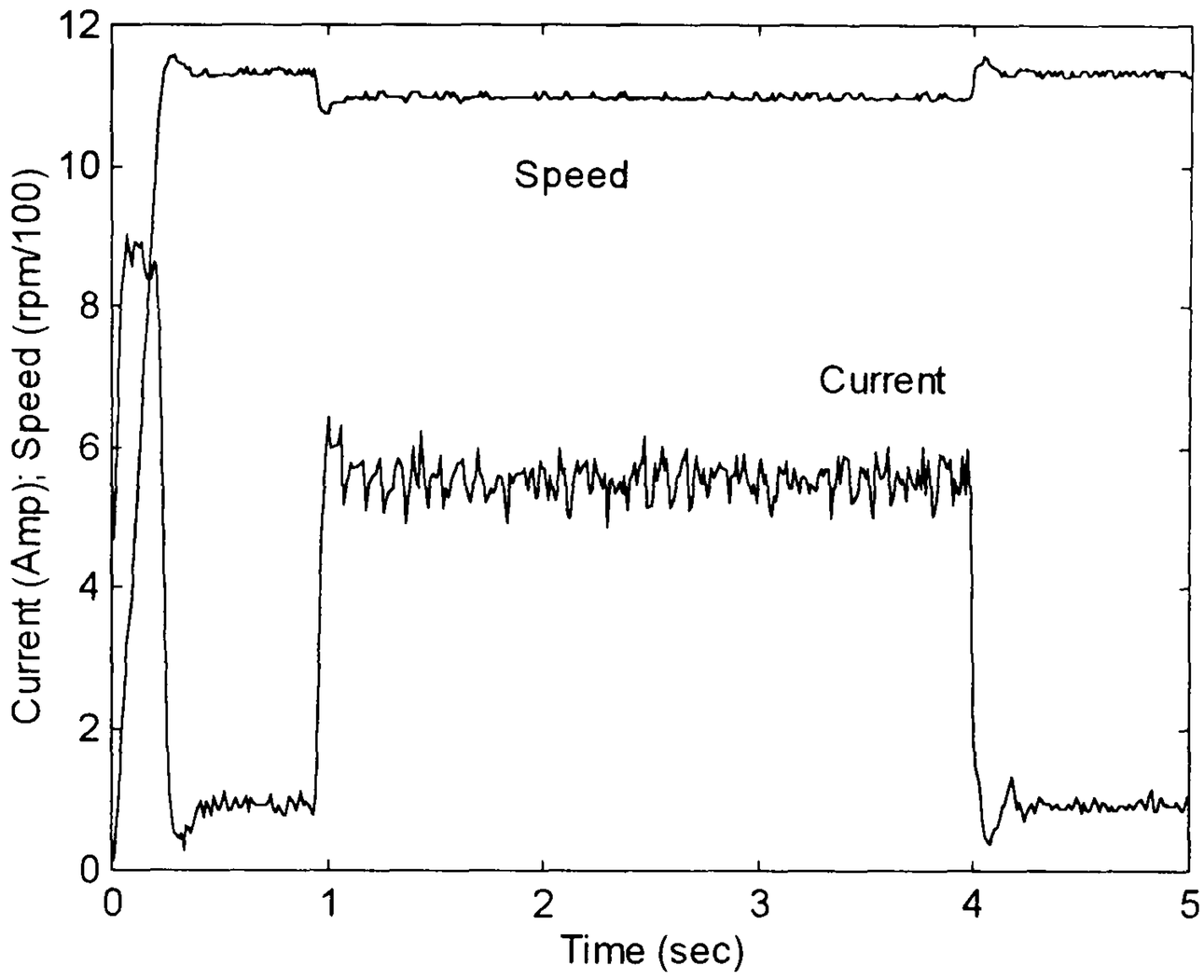



Fig. 10.3 - Best speed and current response after the optimisatin process where the 20 chromosomes were evolved through 20 generation, for 1100 rpm step input speed demand and load torque variation.

### 10.4.2 - Optimisation for 1100 rpm constant speed in the presence of load disturbance

In this example, the Genetic Algorithm was used to optimise the fuzzy speed controller for driving the motor at constant 1100 rpm speed, in the presence of load disturbance. The controller should be capable of running the motor at 1100 rpm constant speed with 11% of full load torque and in the presence of load torque variation. A load torque equal to 62% of the maximum torque produced by the motor was applied, lasting for 3 s. Twenty chromosomes were created at random and evolved through 20 generations, with identical mutation probability and generation gap as used in chapter 5 and above, in section 10.4.1. The evolution of the

optimisation process, which took place in 2 hours, is shown in Fig. 10.4. The ITAE of the speed response was assessed during the 5 s time window.

Fig. 10.5 presents the best speed and current response found by the Genetic Algorithm after 20 generations. It is clear that the Genetic Algorithm was converging towards the best solution. However, once more the population size and generation number was not large enough for such a complex problem. Nevertheless, the improvement from the first to the last generation was huge, from about 32 rad to 7 rad. The steady-state speed was held precisely at 1100 rpm but during load transients it dropped by 10% and took around 0.5 s to regain the steady-state.

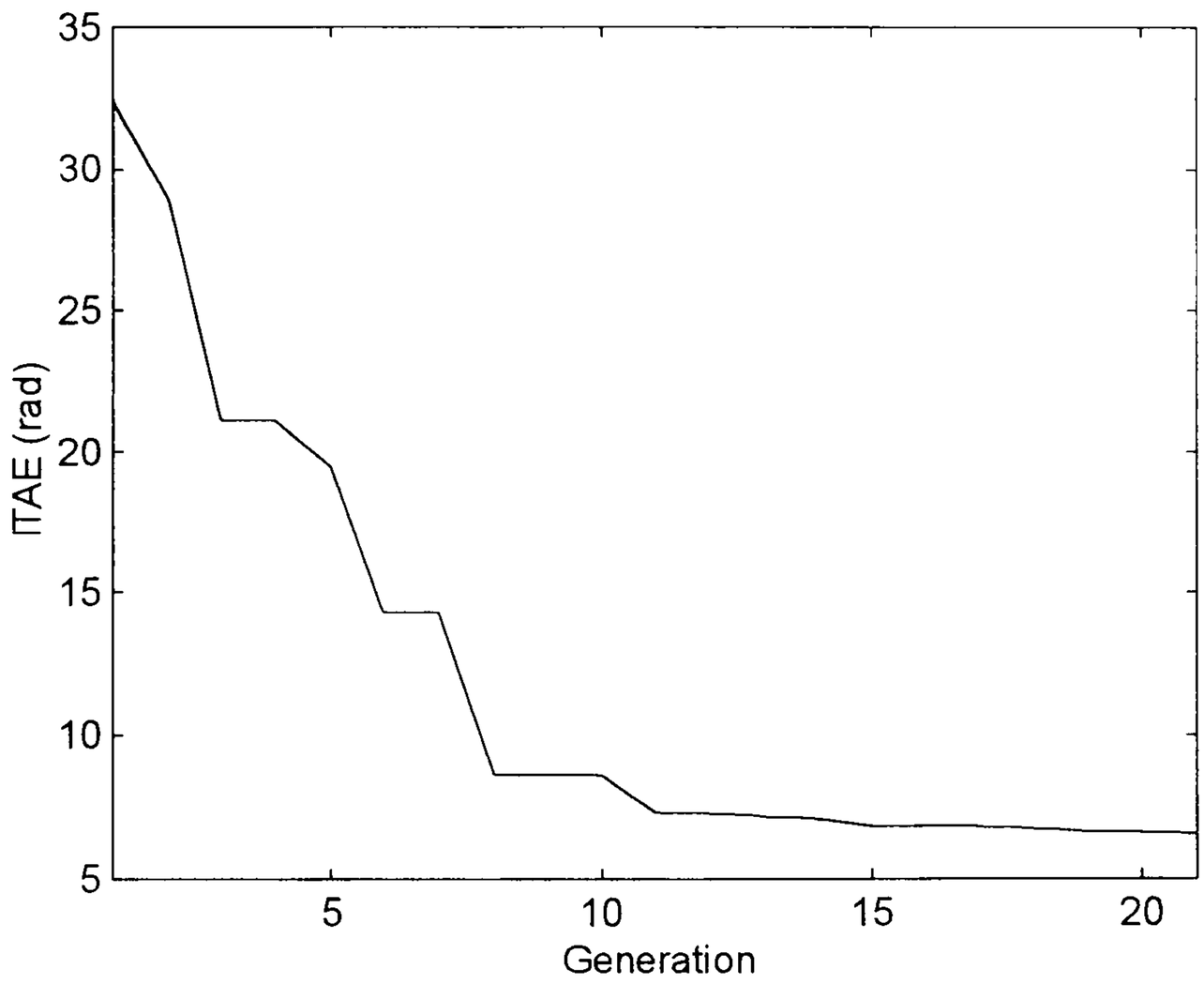



Fig. 10.4 - Evolution of the optimisation process for 1100 rpm constant speed, in the presence of load disturbance.

In order to find even better chromosomes, that represent a better tuning for the fuzzy speed controller, two actions can be taken: i) increase the population size and generation number ii) restrict the searching space to an area close to the final solution, obtained beforehand by using simulation. The first one means even longer evolution time whereas the second one, depending on the searching space, may reduce evolution time. Should for instance, a population of 40 chromosomes be evolved through 40

generations, the optimisation process may last up to 8 hours. In same cases this may be acceptable whereas in others it may be impracticable.

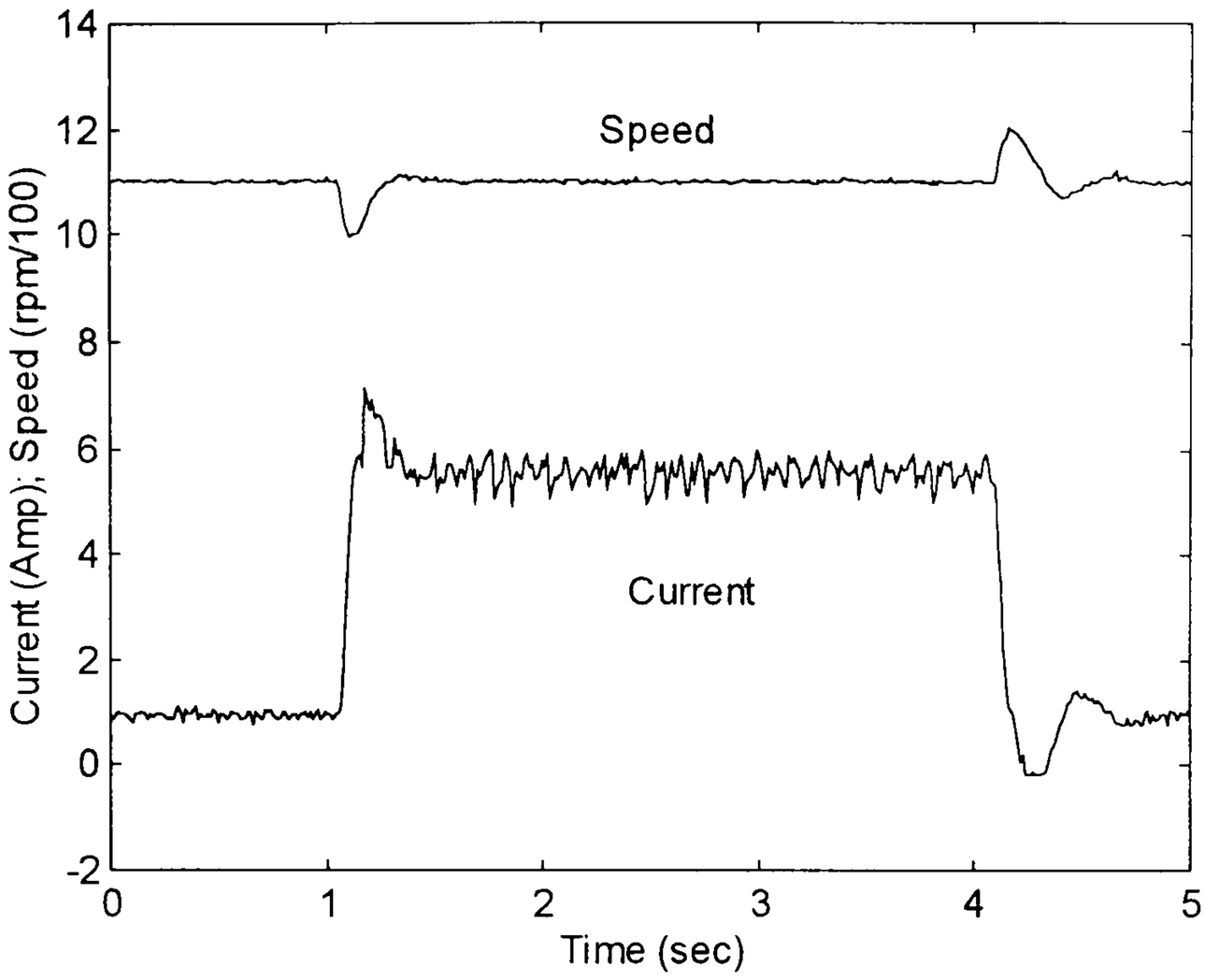



Fig. 10.5 - Best speed and current response of the motor drive obtained after 20 generations, for 1100 rpm constant speed in the presence of 62% of the full load as load torque disturbance.

## 10.5 - INCREASING THE POPULATION SIZE AND RESTRICTING THE SEARCHING SPACE

In the next examples, the motor was driven under identical conditions as in sections 10.4.1 and 10.4.2. However, in order to speed up the evolution process, a simulation was run using identical software and parameters as on the test rig. A searching space closer to the solution given from the simulation was used on the real drive system as follows. The searching space of the 'Speed error' input variable was reduced by 80% and the 'Est. Load' variable by 60%. The 'Curr. Demand 1' and 'Curr. Demand 2' output searching space was reduced by 60%. Despite all the reductions on the searching space, it was still large for the number of variables that should be optimised. Because of this, the population size was made double the one used in the previous

section. The generation number was kept the same, 20 generations. The results are as follows.

### 10.5.1 - Optimisation for step input speed demand in the presence of load variation

The motor was running with identical conditions in terms of speed demand and load variation, as section 10.4.1. The evolution process is shown in Fig. 10.6. Within the restricted searching space, the best chromosome of the first population could provide a speed response better then the best found in section 10.4.1. In addition to this, the performance of the best speed response obtained after 20 generation, was better then the one given by the manual tuning with similar conditions, discussed in section 9.4.2 - Fig. 9.10 where the ITAE was 25.09 rad. The improvement from the first population of chromosomes to the last one was still significant, despite the restricted searching space. It means that the searching space was still quite large and further improvement would be made if it was left to run longer.

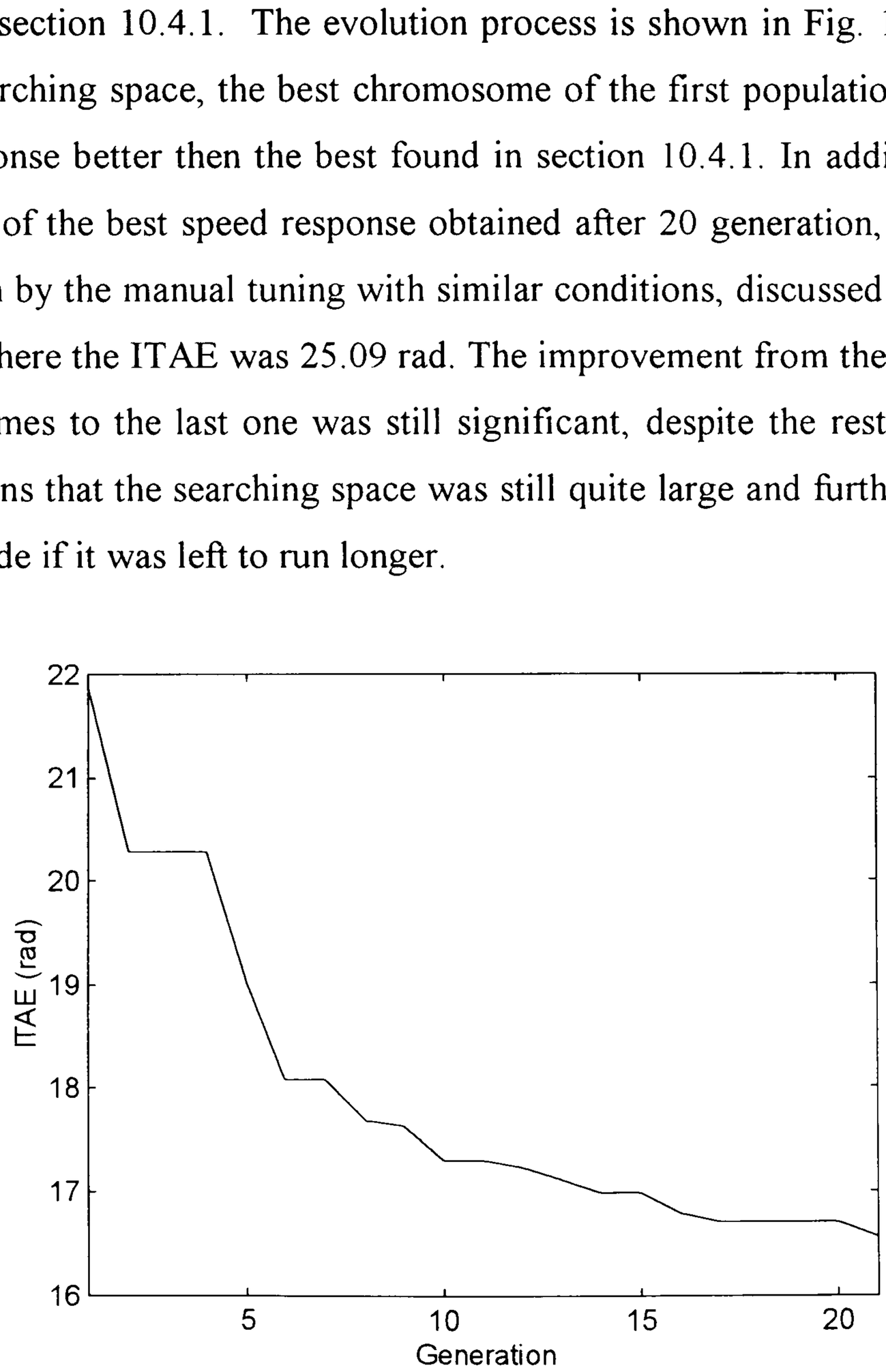



Fig. 10.6 - Evolution of the optimisation process for 1100 rpm step input speed demand in the presence of load disturbance, where the GA evolved a population of 40 chromosomes over 20 generations, in a restricted searching space.

Fig. 10.7 shows the best speed and current response obtained at the end of the optimisation process. The form of the speed response is very similar to the one shown in Fig. 9.10. However, the ITAE of the response given by GA is 16.5 rad against 25.09 rad of that shown in section 9.4.3. The major difference is concerned to the acceleration during the starting transient, which is a bit faster here, evidenced by the current response in the beginning of the time window. It is bigger here compared to that shown in Fig. 9.10. The optimisation process lasted 3 hours 46 min.

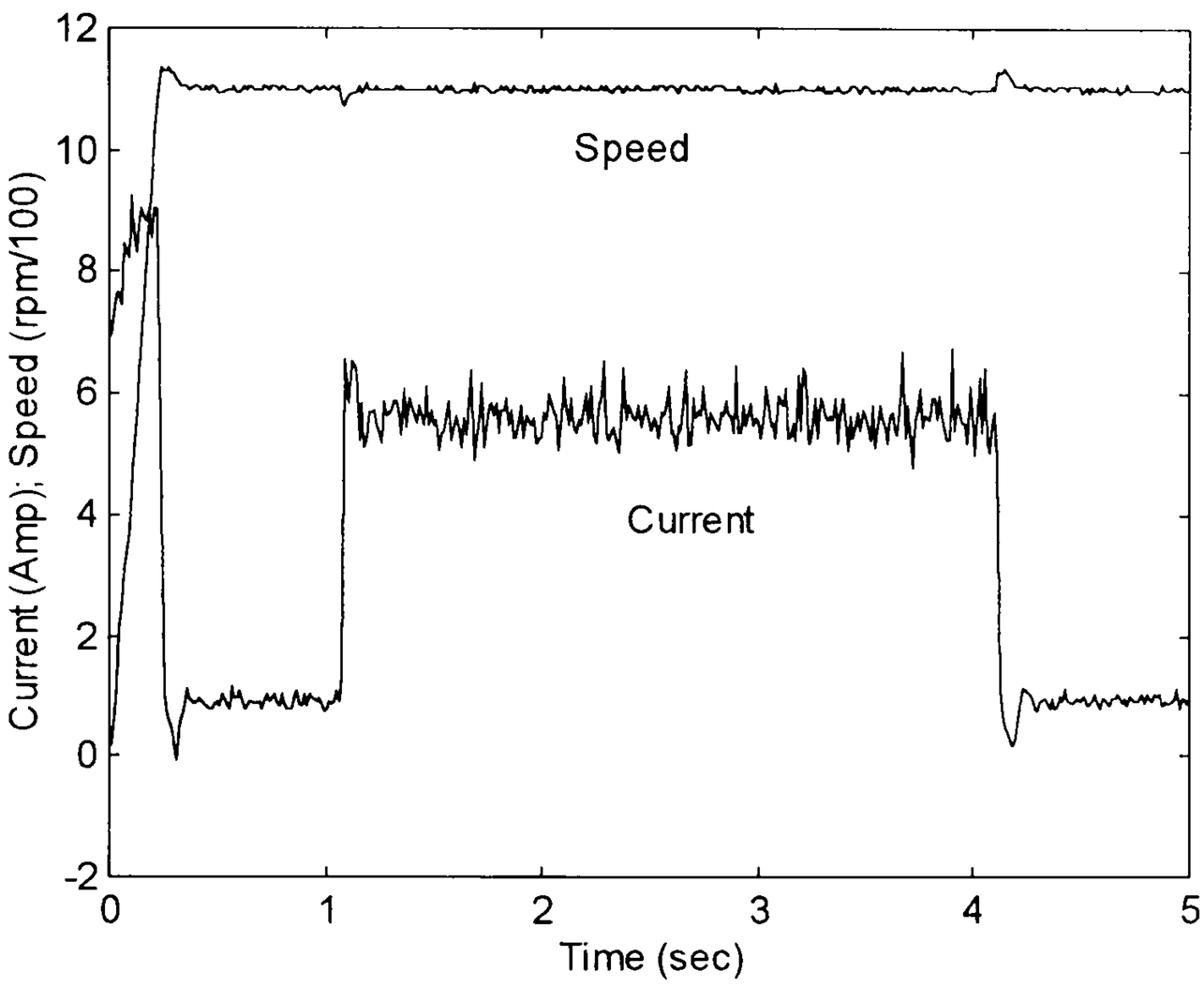



Fig. 10.7 - Best speed and current response after the optimisatin process where the 40 chromosomes were evolved through 20 generations, for 1100 rpm step input speed demand and load torque variation.

### 10.5.2 - Optimisation for 1100 rpm constant speed in the presence of load disturbance

The load condition here here is identical to that of section 10.4.2. The fuzzy controller should be able to drive the motor at 1100 rpm constant speed with load torque 11% of the full torque produced by the motor and hold the speed in the presence of a 62% load torque disturbance.

A population of 40 chromosomes was created at random, within the same searching space as described in section 10.4 and used in section 10.4.1. The results are as follows. In Fig. 10.8 is shown the evolution of the chromosomes throughout the generations, where a speed response to which an ITAE of 1.81 rad over a 5 s time window, was found at the end of the optimisation process which lasted 3 hours and 55 min.

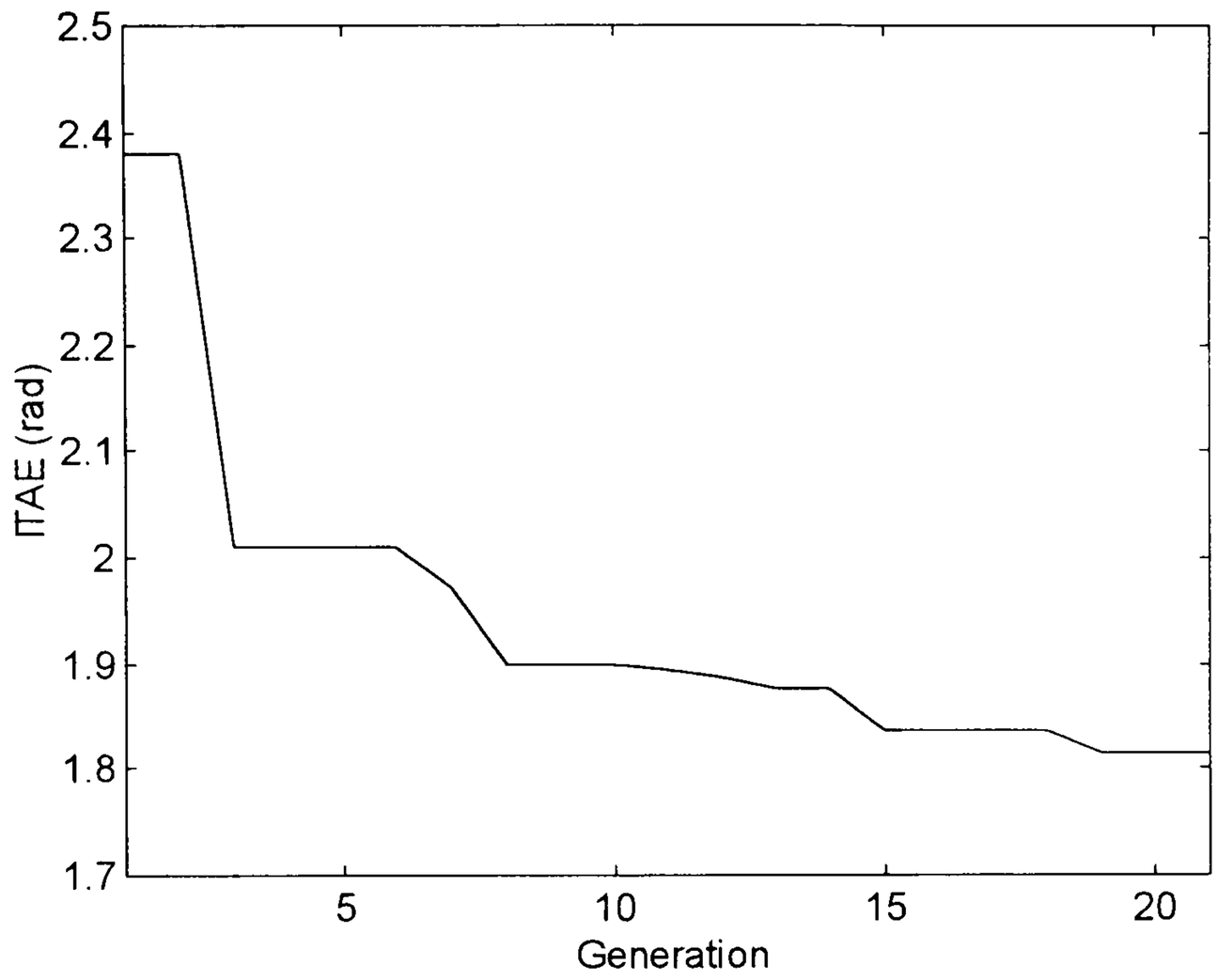



Fig. 10.8 - Evolution of the optimisation process for 1100 rpm constant speed in the presence of load disturbance, where the GA evolved a population of 40 chromosomes over 20 generations, in a restricted searching space.

The best speed and current response found after 20 generations is shown in Fig. 10.9. It is nearly identical to that shown in Fig. 9.6 where the fuzzy controller was tuned manually. The motor was driven afterwards in identical condition however, the ITAE was measured during the duration of the disturbance only and an ITAE of 0.74 rad was found. This value is the same as the one found in section 9.3 with a manual tuning. It means that the same performance against load variation can be obtained by using either the PI or fuzzy speed controller with load estimator. It is important to emphasise that the comparison was made within identical conditions and that the best result for each one was obtained using the on-line optimisation by Genetic Algorithm.

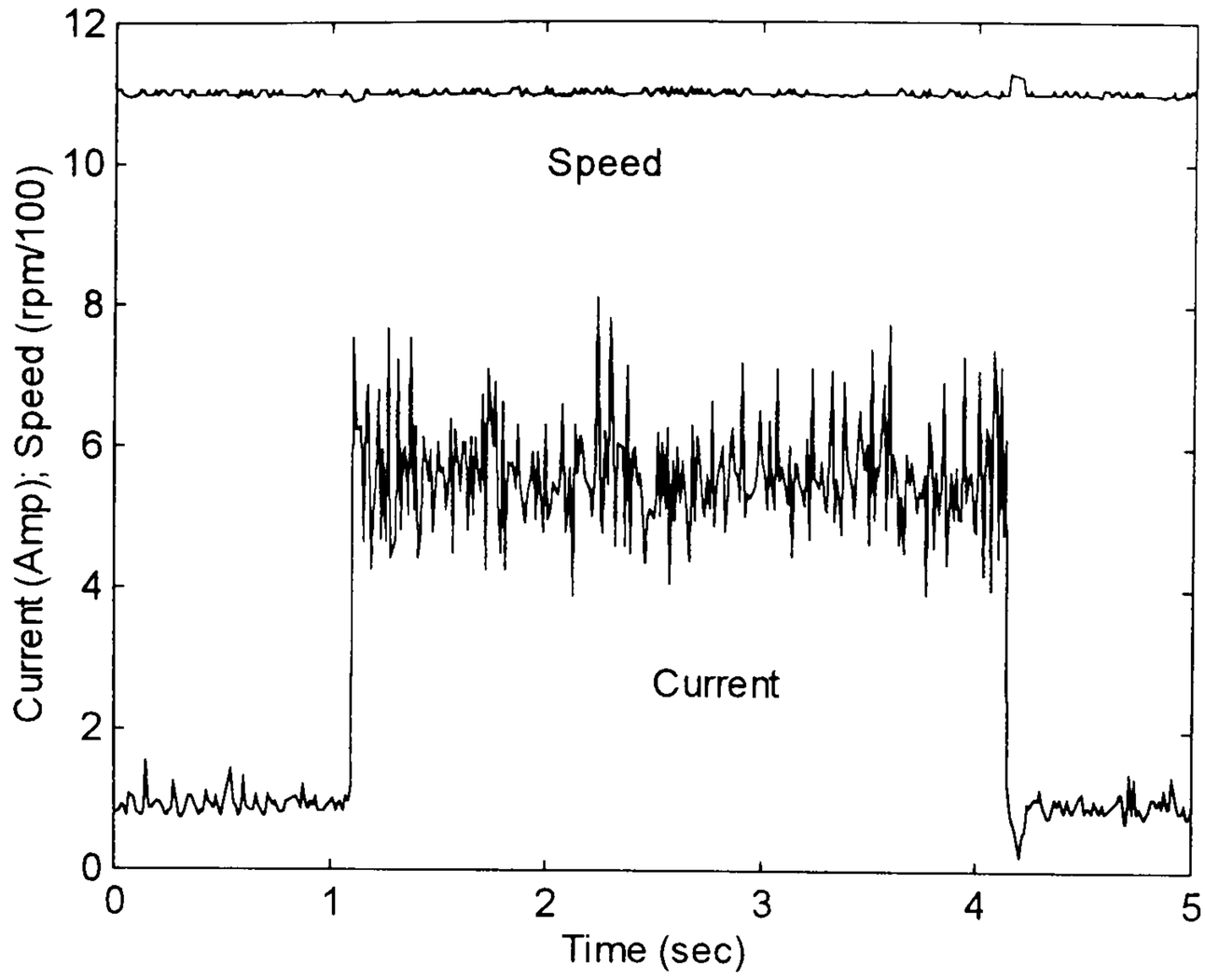



Fig. 10.9 - Best speed and current response of the motor drive obtained after evolving 40 chromosomes over 20 generation within restricted searching space, for 1100 rpm constant speed in the presence of 62% of the full load as load torque disturbance.

Fig. 10.10 shows the speed response only, in enlarged scale highlighting the speed variation during the disturbance.

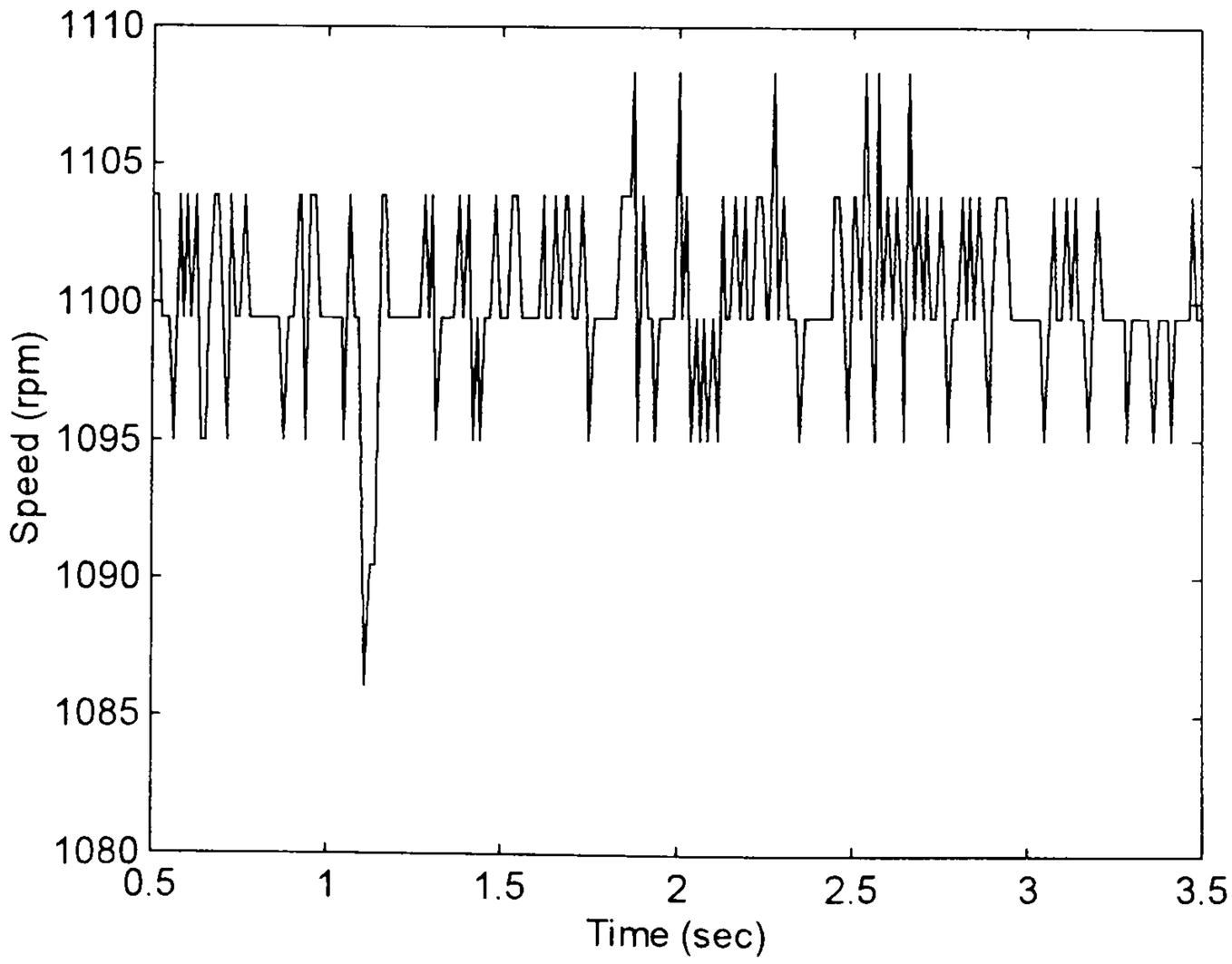



Fig. 10.10 - Speed response in enlarged scale highlighting the speed variation during the load disturbance.

## 10.6 - CONCLUSION

In this chapter the use of Genetic Algorithms for the tuning of a fuzzy speed controller has been investigated. Due to the large number of parameters to be optimised, the optimisation process for the fuzzy controller requires a bigger population and evolution through more generations than for a simple PI controller. As a consequence, the Genetic Algorithm may take several hours to complete the tuning process. However, simulation can be used to find the optimal tuning off-line. By doing this, it is possible to identify smaller area that could be used on-line on the Brushless DC Motor Drive, where final adjustments for best performance can be made. Regarding the performance in terms of speed holding in the presence of load disturbance, the best fuzzy logic controller tuning found by Genetic Algorithm after evolving a population of 40 chromosomes over 20 generations could only match the performance of the PI with load estimator, optimised with identical speed and load condition. Nevertheless, optimising a PI controller is a much easier task for the Genetic Algorithm as only two parameters have to be tuned. On the other hand, for the fuzzy controller discussed in chapter 9 and 10, there are 16 parameters to be optimised.

# CHAPTER 11

## GENERAL CONCLUSIONS

In the first place, the aim of this work was to investigate the speed control of electric drives working at constant speed, in the presence of load disturbance. However, there are applications for electric drives in industry where the motor has to run at variable speed. Because of this, the performance of the speed controllers has been tested for different speed demand and load torque. The motor drive used was a Brushless DC one, although the theory discussed here applies to any other that can be modelled as a simple DC motor such as the DC motor itself or the vector controlled induction motor.

Two different control approaches have been studied: i) - the speed control of the motor drive using the classical proportional-integral controller and ii) - the use of fuzzy speed controller.

In order to improve the performance of the controllers in the presence of load torque variation, a load estimator was added to both speed controllers.

Since the performance of the controllers is directly related to their tuning, a genetic algorithm for the on-line optimisation of the speed controllers was used.

In order to investigate the robustness of the speed controller, several comparisons have been done within different conditions, after ensuring that the basis for comparison was identical.

The performance of a simple proportional-integral speed controller was discussed in chapter 4. It has been shown that, depending on the step input speed demand and the settings of the controller, its performance is poor. For the controller to be able to hold the speed in the presence of load disturbance, large proportional and integral gain

has to be used. However, if the controller setting is high and a large step input speed demand is applied, the problem known as *windup* may occur due to the current limitation of the motor drive.

In order to improve the performance for a step input speed demand and load variation, an appropriate *anti-windup* circuit has to be used. Nevertheless, should the step input reference speed and/or load torque change, the performance deteriorates.

Aiming to improve the speed holding capability of the Brushless DC Motor Drive with proportional-integral speed controller, a load estimator and torque feedforward control was utilised. Experimental results presented in chapter 6 demonstrates the improvement in the performance of the speed response in the presence of load disturbance, especially when the tuning was done on-line by using a genetic algorithm. The benefit of the load estimator is apparent and does help the speed controller hold the speed when the load torque changes. The speed drop due to a load torque disturbance is reduced and its recovery time is shorter.

The tuning of the proportional-integral speed controller off-line by using the classical tuning approach found in the literature can not guarantee best performance because of the non-linearities of the drive system. Genetic algorithms have proved to be a powerful tool that can be used for the optimisation of the tuning of the speed controller for best performance, for any particular condition. The on-line tuning can guarantee optimum performance because the optimisation process takes place on the motor drive itself and all the non-linearities are taken into account.

In chapter 8 a single-input single-output fuzzy speed controller was created to replace the classical proportional-integral one. It has been shown that this controller works similarly to a simple proportional one and can not guarantee zero steady state speed error. However, in the fuzzy controller it is possible to shape the transfer function. This characteristic is not found in the proportional or proportional-integral. Nevertheless, it is important to highlight that it is also possible to adapt the gains of the PI controller so its transfer function can be adjusted to meet requirements. Although this has found favour in some circumstances, it introduces considerable extra complication into the controller.

A two-input two-output fuzzy speed controller was discussed in chapter 9. It has been shown that the fuzzy controller was capable of driving the motor at different speed demand and in the presence of load variation with good performance. On the other hand, as the degree of freedom in designing a fuzzy controller is high, the optimum tuning may not be easy to find.

A genetic algorithm has been used for the on-line optimisation of the membership functions of the fuzzy speed controller discussed in chapter 9. Because of the large number of parameters to be optimised, the population size as well as the generation number has to be larger than that needed for the proportional-integral. As a consequence, if the searching space used in the on-line optimisation process is very wide, it takes very long to complete. However, the off-line optimisation can be done by using computer simulation and the result used as a guide to set the searching space for the on-line. By doing this the genetic algorithm converges faster towards the optimum solution.

The robustness of the proportional-integral and the fuzzy speed controller with load estimator has been investigated. After tuning the controllers for best performance at constant speed in the presence of load variation, the motor was driven at different conditions as discussed in chapter 7. It has been shown that the performance of the proportional-integral deteriorates significantly. The reason is the integrator *windup*. Although it is essential for ensuring zero steady state speed error, it keeps a "history" of the speed response. However, it is not capable of "seeing" when the controller becomes saturated and stop integrating the speed error. The fuzzy controller is different because there is no integrator at all. The controller just applies appropriate output(s) by looking at its input(s). When the motor is driven at constant speed, very high gain can be set to the proportional-integral speed controller, without worrying about integrator *windup*. By using high gains, the controller is capable of responding fast to speed changes caused by the load torque disturbances. This is essential for holding the speed in the presence of load variation. However, the load estimator with feedforward control can bring extra help to the speed controller and the quality of the speed response is enhanced. Nonetheless, the tuning obtained through the on-line optimisation by genetic algorithm ensures optimal performance with further

improvement to the speed response. By doing this, the best possible speed controller within the condition to which it has been optimised was obtained. Even then, the two-input two output fuzzy speed controller with the same load estimator could only match the performance obtained by using the proportional-integral controller.

## FUTURE WORKS

An investigation on the speed control of a Brushless DC Motor Drive has been done where the classical Proportional-Integral and fuzzy controller were used for controlling the speed only. However, an integrated fuzzy speed and current controller has also been developed and tested in simulation although results have not been presented. It differs from those presented here as the output of the fuzzy controller is armature voltage applied to the motor rather than current demand given to the PI current controller. This fuzzy controller has not yet been tested experimentally.

Investigation on the parameter settings of the genetic algorithm for the optimisation of the PI and fuzzy controllers has not been done and is an investigation that could well be carried out.

# REFERENCES:

ABRAMOVITCH, D.: 'Some Crisp Thoughts on Fuzzy Logic', Proceedings of the American Control Conference, 1994, Vol. 1, pp. 168-172.

ALBERTOS, P., STRIETZEL, R. and MORT, N.: 'Control Engineering Solucions - a practical approach', The Institution of Electrical Engineers, London, UK, 1997.

BAGHLI, L., RAZIK, H. and REZZOUG, A.: 'Comparison Between Fuzzy and Classical Speed Control within a Field Oriented Method for Induction Motors', EPE'97, Vol. 2, pp. 2444 - 2448.

BAIK, I.-C., KIM, K.-H. and YOUN, M.-J.: 'Robust nonlinear speed control of PM synchronous motor using adaptive and sliding mode control techniques', IEE Proc. Electr. Power Appl., Vol. 145, no. 4, July 1998, pp. 369 - 376.

BEIERKE, S., VAS, P. and STRONACK, A. F.: 'TMS320 DSP Implementation of Fuzzy-Neural Controlled Induction Motor Drives', EPE'97, Vol. 2, pp. 2.449 - 2.453.

BIANCHI, N., and BOLOGNANI, S.: 'Design optimisation of electric motors by genetic algorithms' Proc. IEE, Electrical Power Applications, Vol 145, No 5, pp. 475-483, September 1998.

BIRD, I. G. and De La PARRA, H. Z.: 'DSP Implementation of a Fuzzy Based Direct Flux and Torque Controller', EPE'97, Vol. 2, pp. 2.415 - 2.419.

BOSE, B. K.: 'Power electronics and variable frequency drives: technology and applications', IEEE Press, 1997 - New York - US.

BROGAN, W. L.: 'Modern Control Theory', Prentice-Hall, Inc., Third Edition, 1991.

BUJA, G. S., MENIS, R. and VALLA, M. I.: 'Disturbance Torque Estimation on a Sensorless DC Drive', IEEE Trans. on Industrial Electronics, Vol. 42, August 1995, pp. 351 - 357.

CATALIOTTI, A. and POMA, G.: 'A Fuzzy-Logic Approach for Easy and Robust Control of an Induction Motor', EPE'97, Vol. 2, pp. 2.421-2.425.

CHANG, C. H. and WU, Y. C.: 'The Genetic Algorithm Based Tuning Method for Symmetric Membership Functions of Fuzzy Logic Control Systems', Proceeding of the 1995 International IEEE/IAS Conference on Industrial, Automation and Control: Emerging Technologies, Taipei - Taiwan, pp. 421 - 428.

CHANG, S. Y., HUANG, T. L. and HUANG, T. Y.: 'Fuzzy Bang-Bang Controller For Servo Systems VIA Optimal Path Estimation Method', Proc. International Conference on Power Electronics and Drive Systems, 1995, IEEE USA, Vol. 1, pp. 208 - 211.

CHEN, G.; 'When and When Not Use Fuzzy Logic in Industrial Control', Proc. of the 29th Annual Meeting of the IEEE Industry Application Conference, 1993, Toronto, pp. 2035 - 2040.

CHIPPERFIELD, A., FLEMING, P., POHLHEIM, H., and FONESCA, C.: 'Genetic algorithm toolbox for use with MATLAB', Version 1.2., Dept. of Automatic Control and Systems Engineering, University of Sheffield, England.

COTTA, CARLOS, ALBA, E. and TROYA, J. M.: 'Evolutionary Design of Fuzzy Logic Controllers', Proceedings of the 1996 IEEE Internatioinal Symposium on Intelligent Control, Dearborn, MI, September 1996, pp. 127 - 132.

DA SILVA, W. G. and ACARNLEY, P. P.: 'Fuzzy logic controlled DC motor drive in the presence of load disturbance', EPE'97 – Vol. 2, pp. 2386–2391.

FODOR, D., KATONA, Z. and VASS, J.: 'On Fuzzy Logic Speed Control for Vector Controlled AC Motors', Proceedings of 4th International Workshop on Advanced Motion Control - AMC'96-MIE, Vol. 1, pp 186-191.

GEN, M., and CHENG, R.: 'Genetic algorithms and engineering design', John Wiley & Sons, Inc., 1997.

GOLDBERG, D.: 'Genetic Algorithm in Search, Optimization and Machine Learning', Addison-Wesley Pub. Co.,1989.

GOVIND, N. and HASAN, A. R.: 'Real Time Fuzzy Logic Speed Control Using Conventional, Assembly and Simulation Methods for Industrial Motors', Proceedings of the IEEE/IAS International Conference on Industrial Automation & Control, 1995, Hyderabab, India, pp. 203 - 208.

GUILLEMIN, P.: 'Fuzzy Logic applied to Motor Control', IEEE Trans. on Industry Applications, Vol. 32, no. 1, January/Febryart 1996, pp. 51 - 56.

GULLEY, N. and ROGER JANG, J. S.: 'Fuzzy logic toolbox for use with MATLAB', The Math Works Inc., 1995.

HO, S., JONES, A. and COX, C.S.:'Torque Disturbance and their Effect on Process Control', Power Electronics and Variable Speed Drives, October 1994, Conf. Publicatiion no. 399, IEE, pp. 602 - 607.

HOMMAIFAR, A. and McCORMICK, E.: 'Full design of fuzzy controllers using genetic algorithm', Conference on Neural Stochastic Methods Image Signal Processing, 1992, Vol 1776, pp. 393 - 404.

HOMMAIFAR, A. and McCORMICK, E.: 'Simultaneous Design of Membership Functions and Rule Sets for Fuzzy Controllers Using Genetic Algorithms, IEEE Transactions on Fuzzy Systems, Vol. 3, no. 2, May 1995, pp. 129 - 139.

HWANG, T.-Y., YEN, J.-Y. and LU, S.-S.: 'Bang-Bang Based Fuzzy Controller for Time Optimal and Minimum Chattering Servo Systems', Electric Machines and Power Systems, 1995, Vol. 23, pp. 25 - 35.

INOUE, K. and NAKAOKA, M.: 'Autotuning gain parameter implementation with fuzzy learning control scheme for DC brushless servo system:, IEE Proc. Control Theory Appl., Vol. 145, no. 5, September 1998, pp. 419 - 427.

IWASAKI, M and MATSUY, N.: Robust Speed Control of IM with Torque Feedforward Control, IECON'91, pp. 627 - 632.

KANG, J. K., LEE, J. T., KIM, Y. M., KWON, B. H. and CHOI, K. S.: 'Speed Controller Design for Induction Motor Drives using a PDF Control and Load Disturbance Observer', IECON'91, pp. 799 - 803.

KAR, CH. L.: 'Design of an Adaptive Fuzzy Logic Controller Using a Genetic Algorithm', Proceedings of 4th International Conference on Genetic Algorithm, 1991, pp. 450 - 457.

KAWAJI, S., SUENAGA, Y., MAEDA, T., MATSUNAGA, N. and SASAOKA, T.: 'Control of Cutting Torque in the Drilling Process using Disturbance Observer', Proc. of the American Control Conference - Seatle, Washington, 1995, Vol. 1, pp. 723 - 728.

KIM, Y. H., HAN, S. H., CHO, S. I. and HA, I. J.: 'Learning approach to control of servomotors under disturbance torque dependent on time and states', IEE Proc. Control Theory Appl., Vol 145, no. 3, May 1998, pp. 251 - 258.

KO, J. S., LEE, J. H. and YOUN, M. J.: 'Robust digital position control of brushless DC motor with adaptive load torque observer', IEE Proc. Electr. Power Appl., Vol 141, no. 2, March 1994, pp. 63 - 70.

KO, J. S., LEE, J. H., CHUNG, S. K. and YOUN, M. J.: 'A Robust Digital Position Control of Brushless DC Motor with Dead Beat Load Torque Observer', IEEE Trans. on Industrial Electronics, Vol 40, no. 5, October 1993, pp. 512 - 520.

KUO, B.: 'Digital Control Systems', Second Edition, Saunders College Publishing, 1992.

LEE, C. K. and PANG, W. H.: 'A brushless DC motor speed control system using fuzzy rules', IEE Conference on Power Electronics and Variable Speed Drives, no. 399 - 26 – 28 October 1994, pp. 101–106.

LEE, M. A. and TAKAGI, H.: 'Integrating Design Stages of Fuzzy Systems using Genetic Algorithms', Proceeding of the IEEE International Conference on Fuzzy Systems, 1993, pp. 612 - 617.

LIN, F. -J. and CHIU, S.-L.: 'Adaptive fuzzy sliding-mode control for PM synchronous servo motor drives', IEE Proc. Control System Theory, Vol. 145, no. 1, January 1998, pp. 63 - 72.

LIN, Y.-C.: 'The Application of Fuzzy Logic Control to Speed Control of a DC Servo Motor Drive', Proceedings of the American Control Conference, Baltimore, Maryland, June 1994, pp. 590 - 594.

LISKA, J. and MELSHEIMER, S. S.: 'Design of Fuzzy Logic Systems for Nonlinear Process Identification', Proceeding of the American Control Conference, Baltimore, Maryland, June 1994, pp. 976 - 980.

MAN, K.F., TANG, K.S., and KWONG, S.: 'Genetic algorithms: concepts and applications', IEEE Transactions on Industrial Electronics, Vol. 43, No. 5, pp. 519 - 533, October 1996.

MEREDITH, D. L., KARR, C. L. and KUMAR, K. K.: 'The Use of Genetic Algorithms in Design of Fuzzy Logic Controllers', Proceeding of 3rd Workshop on Neural Networks Academic/Industrial/NASA/Defense, 1992 Alabama, Proc. SPIE, 1993, Vol. 1721, pp. 549 - 555.

MICHELS, KAI: 'Fuzzy logic for electrical drives?', 7th European Conference on Power Electronics and Applications - EPE'97 – Vol. 1, pp. 1.102 - 1.105.

NISHII, H., SHIBATANI, K. and SUZUKI, K.: 'Vibration Control of a Wheeled Mobile Robot based on Disturbance Observer', Proc. of the 1992 Japan USA Symposium on Flexible Automation, Part 1, pp. 1365 - 1369.

OGATA, K.: 'Modern Control Engineering', Prentice-Hall, Inc., Third Edition, 1997

PARK, YOUNG-MOON, MOON, UN-CHUL and LEE, K. Y.: 'A Self-Organizing Fuzzy Logic Controller for Dynamic Systems Using a Fuzzy Auto-Regressive Moving Average (FARMA) Model', IEEE Transactions on Fuzzy Systems, Vol. 3, no. 1, February 1995, pp. 75 - 82.

QIN, YU and DU, S.: 'A Fuzzy PID Controller Optimized by Genetic Algorithms used for a Single Phase Power Factor Pre-Regulator', EPE'97, Vol. 2, pp. 2392 - 2396.

SENJYU, T., TOMITA, M., KOKI, S. and OKUMA, S.: 'Sensorless Vector Control of Brushless DC Motors Using Disturbance Observer', PESC Record IEEE Power Electronic Specialists Conference, 1995, Vol. 2, pp. 772 - 777.

SHIN, HWI-BEOM: 'New Antiwindup PI Controller for Variable-Speed Drives', Trans. on Industrial Electronics, Vol. 45, No. 3, June 1998, pp. 445 - 450.

SHINNERS, S. M.: 'Modern Control System Theory and Design', John Wiley & Sons, Inc., Second Edition, 1998.

SOLIMAN, H. F. and SHARAF, A. M.: 'A Fuzzy Logic Tunable Speed Controller for A Rectifier Fed PMDC Motor Drives', Proceeding of the 26th Southeastern Symposium on System Theory, Ohio - USA, 1994, pp. 22 - 26.

SOLIMAN, H. F., MANSOUR, M. M. and KANDIL, S. A.: 'A Robust Tunable Fuzzy Logic Control Scheme for Speed Regulation of DC Series Motor Drives', CCECE/CCGEI'95, pp. 296 - 299.

SOUZA, G. D. D. and BOSE, B. K.: 'Fuzzy logic applications to power electronics and drives - an overview', Proceedings 1995 IEEE 21st International Conference on Industrial Electronics Control and Instrumentation, 1995, Vol. 1 PT1, pp 57-62.

TANG, KIT-SANG, MAN, KIM-FUNG, LIU, ZHI-FENG and KWONG, S.: 'Minimal Fuzzy Membership and Rules Using Hierarchical Genetic algorithm', IEEE Transactions on Industrial Electronics, Vol. 45, no. 1, February 1998, pp. 162 - 169.

TANG, Y. and XU, L.: 'Vector Control and Fuzzy Logic Control of Doubly Fed Variable Speed Drives with DSP Implementation', IEEE Trans. on Energy Conversion, Vol. 10, no. 4, December 1995, pp. 661 - 668.

TANG, K. S. MAN, K. F. and CHAN, C. Y.: 'Fuzzy Control of Water Pressure Using Genetic Algorithm', IFAC Workshop on Safety Reliability and Applications of Emerging Intelligent Control Technologies, 1994, Hong Kong, pp. 15 - 20.

VAS, P., CHEN, J. and STRONACH, A. F.: 'Application of improved fuzzy-tuned and fuzzy controllers in variable-speed drives', CONTROL'94, 21 - 24 March 1994, Conference Publication No. 389, pp. 1501-1506.

VAS, P., DRURY, W. and STRONACH, A.F.: 'Present and Future of Drives, Sensorless and Artificial Intelligence Application', EPE'97 - Vol. 2, pp. 4.573-4.578.

VOLCANJK, V. and JEZERNIK, K.: 'Induction Motor Control with PI-Load estimator', Mediterranean Eletrotechnical Conference - MELECON'94, Vol 3, pp. 1097 - 1100.

VRANCIC, D. and PENG, Y.: 'Some Practical Recommendations in Anti-windup Design', UKACC International Conference on CONTROL'96, 1996, pp. 108 - 113.

WIEN, R. N.: 'Anti-Reset-Windup-Maßnahmen bei Eingößernregelungen', Automatisierungstechnik at, 35 Jahrgang, Heft 1,1987, pp. 32 - 39.

WILLIAMSON, D.: 'Digital Control and Implementation – Finite Wordlength Consideration', Prentice Hall of Australia Pty Ltd, 1991.

YING-CHI LIN: 'The Application of fuzzy Logic control to Speed Control of a DC Servo Motor System', Proceedings of the American Control Conference, 1994, Vol. 1, pp. 590-594.

# APPENDIX A

# ANALYSIS OF THE LOAD ESTIMATOR

In this appendix the mathematical analysis of the load torque estimator proposed by Iwasaki [IWASAKI *et al*, IECON'91] is presented. Since it is clearly explained in the paper, showing it again would not be necessary. However, it is helpful for those who do not yet hold a copy of the mentioned paper. The block diagram of the load torque estimator only is shown in Fig. A.1.

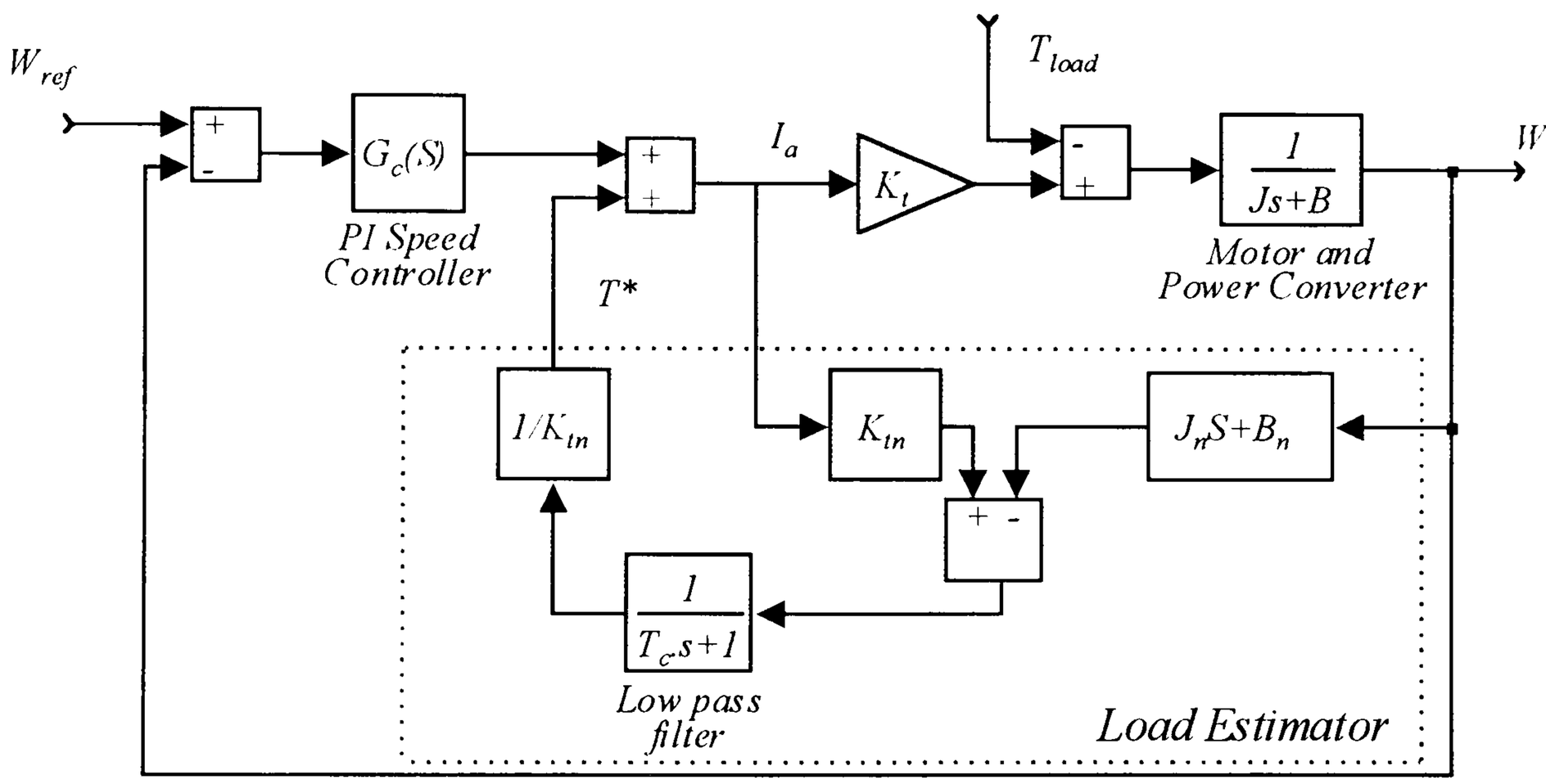



Fig. A.1 - Speed controlled Brushless DC Motor with load estimator.

Where:

$W_{ref}$ - Speed demand.

$W$ - Actual speed of the motor.

$I$ - Armature current.

$T_c$ - Low pass filter time constant

$T_{load}$ - Load torque

$T^*$ - Observed torque

$K_t = K_{tn}$ - Torque constant.

$J = J_n$ - Motor/load inertia.

$B = B_n$ - Friction coefficient.

In a DC motor drive system, as shown in Fig. A.1, the torque balance is given as in equation A.1.

$$T_{elec} = T_{load} + J\frac{dW}{dt} + BW \qquad (A.1)$$

Usually, the friction coefficient is very small and can be neglected. By doing so, equation A.1 reduces to:

$$T_{elec} = T_{load} + J\frac{dW}{dt} \qquad (A.2)$$

The electrical torque produced by the motor is given by:

$$T_{elec} = K_t I_a \qquad (A.3)$$

After substituting equation A.3 into A.2 we find that:

$$J\frac{dW}{dt} = K_t I_a - T_{load} \qquad (A.4)$$

Since the load torque in an unknown variable, the load estimator should be introduced assuming that it is constant within the sample time. This assumption is valid as the sampling period of the observer is usually very short compared to the variations of the load torque. Then,

$$\frac{dT_{load}}{dt} = 0 \qquad (A.5)$$

Based on equations A.4 and A.5, the following one can be obtained:

$$\begin{bmatrix} dW/dt \\ dT_{load}/dt \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{J} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} W \\ T_{load} \end{bmatrix} + \begin{bmatrix} K_t/J \\ 0 \end{bmatrix} Ia \qquad (A.6)$$

Based on the state equation A.6, $T^*$ can be estimated by the minimal order state observer using Gopinath's design method given in equation A.5 [IWASAKI et al, IECON'91].

$$T^* = \frac{1}{T_c S + 1}\left(K_t I_a - SJW\right) = \frac{1}{T_c S + 1} T_{load} \qquad (A.7)$$

Where $T_c$ is the time constant of the observer.

Should a conventional DC motor control system without the load estimator and the feedforward loop be considered, the transfer functions $W/W_{ref}$ and $W/T_{load}$ are represented as in equations A.8 and A.9 respectively.

$$\frac{W}{W_{ref}} = \frac{K_t G_c(S)}{SJ + K_t G_c(S)} \qquad (A.8)$$

$$\frac{W}{T_{load}} = \frac{1}{SJ + K_t G_c(S)} \qquad (A.9)$$

It can be seen that the denominator of equations A.8 and A.9 are the same. As a consequence, the speed response for changes in speed demand and load rejection response can not be designed separately. With the load estimator, the transfer function $W/W_{ref}$ is the same as in equation A.8. However, regarding the $W/T_{load}$, the transfer function changes to that of equation A.10.

$$\frac{W}{T_{load}} = -\frac{T_c S}{T_c S + 1} \frac{1}{JS + K_t G_c(s)} \qquad (A.10)$$

It means that the speed response for change in demand is independent of the load estimator and can be determined by the speed controller only. The load disturbance rejection can be determined by the speed controller and the load estimator. Then each transfer function can be designed separately. As a consequence, the sytem can have better response against load torque variation while the speed response to changes in speed demand is dictated by the tuning of the speed controller, $G_c(S)$.

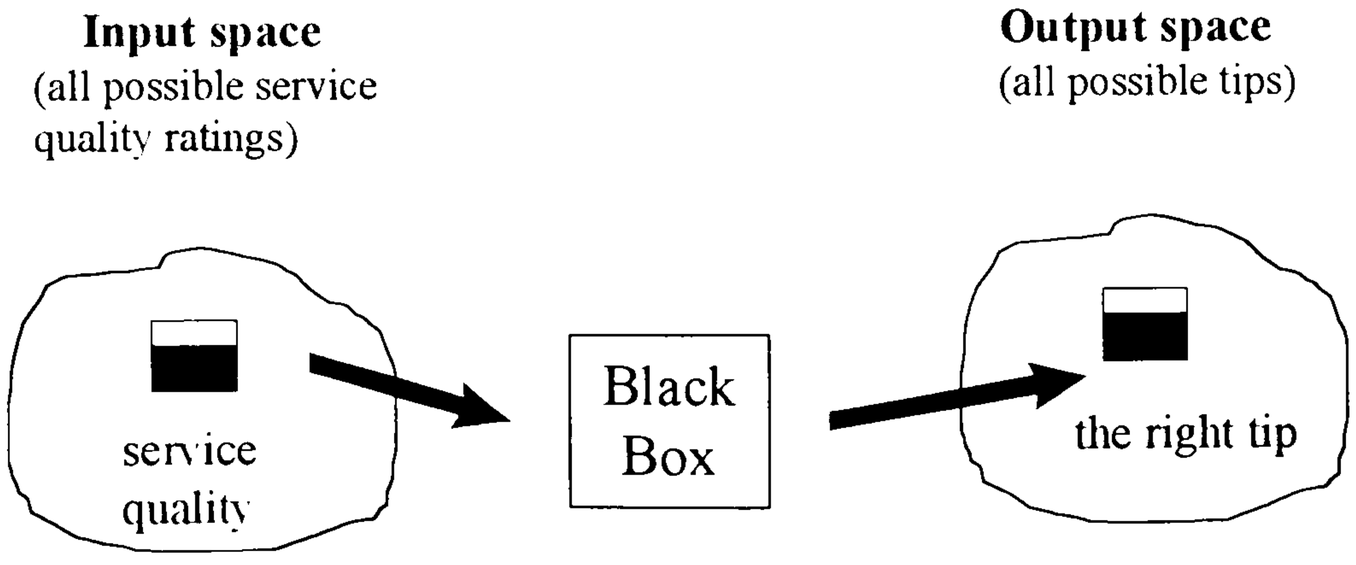
# APPENDIX B

# INTRODUCTION TO FUZZY LOGIC AND FUZZY CONTROL

## B.1 - INTRODUCTION

Fuzzy Control is one among the many possible applications of Fuzzy Systems Theory, developed in the 1920s and 1930s to solve some paradoxes that could not be clearly explained by conventional methods. However, the past few years have witnessed a significant growth in the use of fuzzy theory for control applications.

The key idea of fuzzy system is to develop a framework to deal with imprecision. Instead of using the ordinary concept of set inclusion, fuzzy sets allows the use of functions that express the degree of membership to a given set between zero and one. Then, fuzzy logic can be synonymous of fuzzy sets, a theory that relates to classes of objects with non-sharp boundaries in which membership is a matter of degree. As a consequence, fuzzy logic can be easily used to deal with non-linearities. After all, fuzzy logic can be defined as a convenient way to map an input space to an output space. The tip problem in a restaurant [GULLEY *et al*, 1995] is used as an example. How good the service in a hypothetical restaurant is defines what the tip shall be. The graph below gives a simple idea of how fuzzy logic works.



Then the problem is just a matter of mapping inputs to the appropriate outputs. The black box represents that and inside of it is what can be called Fuzzy Inference System (FIS). Actually, inside of that black box could be any other way to map the inputs to the output like for example, linear system, expert systems, neural network, differential equations and so on.

There are several important characteristics that make fuzzy logic to be very promisingly [GULLEY *et al*, 1995]:

- Conceptually easy to understand;

- Flexible;

- Tolerant to imprecise data;

- Can model non-linear functions of arbitrary complexity;

- Can be built on top of the experience of experts;

- Can be blended with conventional control techniques.

- Based on natural language.

All the above statements constitute important characteristics and show that fuzzy logic works like *human reasoning*. Because of this, some people call it a sort of intelligent control.

## B.2 - FUZZY SETS

Differently from a classical set where an element is completely included or completely excluded, in fuzzy sets the elements can be included with partial degree of membership. The example below illustrates this [GULLEY *et al*, 1995]. In a classical set of days of the week shown in Fig. B.1, Monday, Wednesday and Saturday are completely included. However in a set of the weekends, Friday may appear partially included as in Fig. B.2.

Most people would agree that Saturday a Sunday belongs to the weekend. On the other hand many others may say that Friday is almost weekend but should be technically excluded. However, in fuzzy logic the truth of any statement becomes a

matter of degree. Any statement can be fuzzy and its reasoning gives the ability to reply to a yes-no question like humans do. It works by generalising the familiar yes-no (Boolean) logic. Should "true" be assigned equal to 1 and "false" equal to 0 as it is

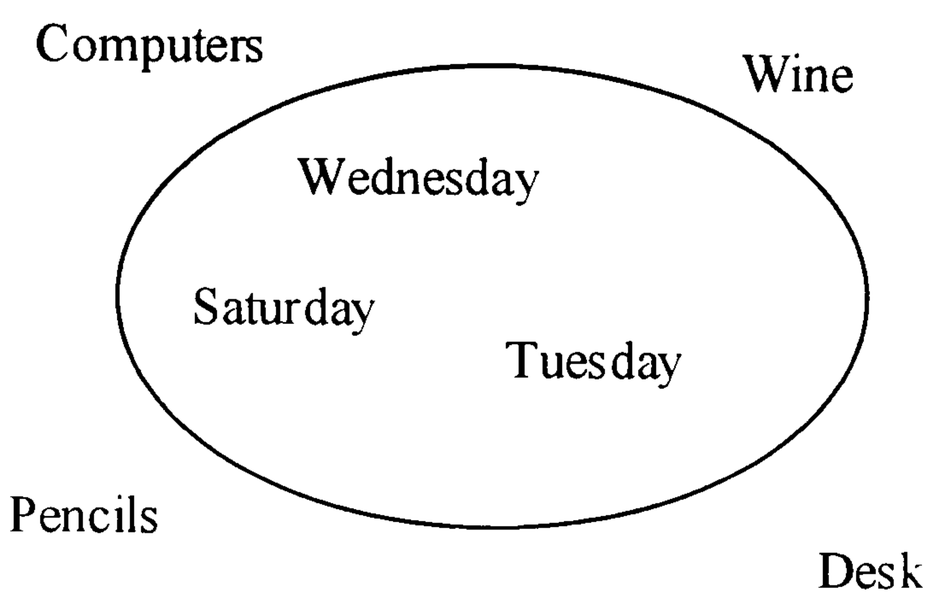



Fig. B.1 - Classical set

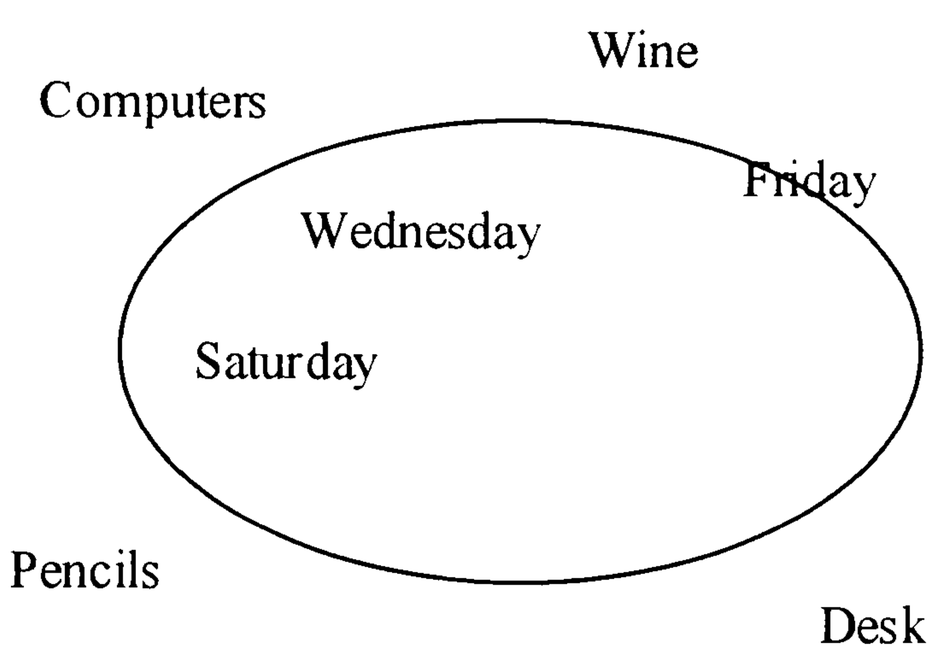



Fig. B.2 - Fuzzy set.

usually done, fuzzy reasoning permits values between 0 and 1. Then, for instance, should anyone be asked if Friday is a weekend day, the reply could be "not completely" or "0.8 yes", or even "yes but not as Saturday". An example of its degree of membership in a classical and in a fuzzy set has been given by Gulley and is shown in Fig. B.3.

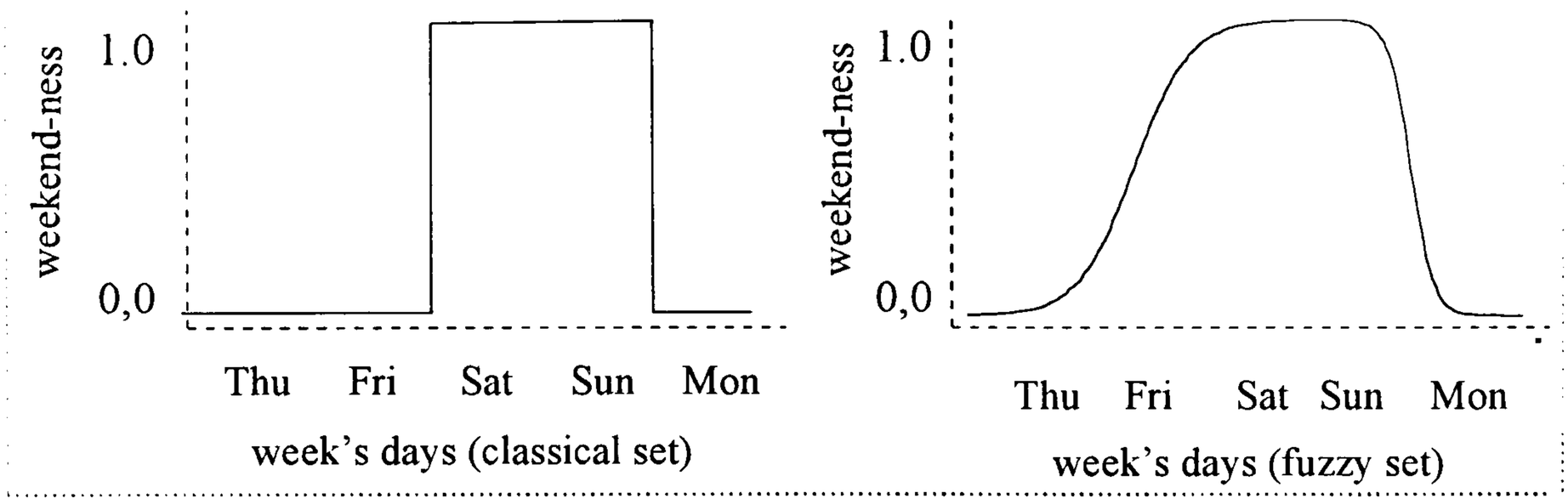



Fig. B.3 - Representation of the degree of membership of an element in a classical and in a fuzzy set.

## B.3 - MEMBERSHIP FUNCTIONS

According go Gulley, "A membership function (MF) is the curve that defines how each point in the input space is mapped to a membership value (or degree of membership) between 0 and 1". The shape of the membership function can be anyone although *triangular* and/or *trapezoidal* are most commonly used due to implementation facility. Then in a fuzzy set, should X be the universe of discourse and its elements denoted by x, a fuzzy set A in X is written as:

$$A = \{x, \mu_A(x) \mid x \in X\}.$$

Where $\mu_A(x)$ is the membership function of x in A. Then, a membership function maps each element of X to a membership value between 0 and 1 or 0% and 100%.

## B.4 - LOGICAL OPERATORS

The fuzzy part of fuzzy logic has been explained. However, the most important thing to be understood is concerned with its reasoning which is based upon three standard Boolean logic operators: AND, OR and NOT. Should for instance, the fuzzy values be kept to the extremes of 1 (completely true) and 0 (completely false), those standard logical operator will be found as shown below:

| A | B | A and B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**AND**

| A | B | A or B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**OR**

| A | not A |
|---|---|
| 0 | 1 |
| 1 | 0 |

**NOT**

However, in fuzzy logic the truth of any statement is a matter of degree where the input can be any real number between 0 and 1. Then, functions like ***min*** and ***max*** operations are used to perform respectively AND and OR logical operators as follows.

| A | B | **min**(A,B) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

**AND**

| A | B | **max**(A,B) |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

**OR**

| A | **1-A** |
|---|---|
| 0 | 1 |
| 1 | 0 |

**NOT**

In that case, since A and B are limited to the range (0,1), min(A,B) is equivalent to the statement A AND B as well as max(A,B) is equivalent to A OR B. NOT A can easily found by using the operation 1-A.

## B.5 - IF-THEN RULES

Fuzzy sets and fuzzy operators constitute the subjects and verbs of fuzzy logic. Nevertheless, complete sentences are needed in order to make fuzzy logic useful. Conditional statements of the type *if-then* are then used. A single fuzzy *if-then* rule takes the form of "*if x is A then y is B*" where A and B are linguistics values defined by fuzzy sets on the ranges X and Y respectively. Gulley has called the if-part as antecedent and the then-part the consequence.

The interpretation of an if-then rule involves two parts: the evaluation of the antecedent (making the input fuzzy and applying the fuzzy operators when necessary) and the application of the result to the consequence (also known as implication). In the case of binary logic, if the premise is true then the conclusion is true. Nevertheless, in the case of fuzzy logic, if the antecedent is 0.5 true then the consequence is also 0.5 true. In other words, partial antecedents imply partial consequences.

One important characteristic of fuzzy logic is the fact that either the premise or the implication can have multiple parts. Example:

if ***x is A*** and ***y is B*** and ***z is C*** then ***r is D*** and ***s is E*** and ***t is F***.

In fuzzy logic, one rule itself does not help so much. Two or more rules that work together are necessary. However, the output of each rule is a fuzzy set and what is needed is the output for an entire collection of rules to be a single number. The process of how it is made is now going to be explained.

## B.6 - FUZZY INFERENCE ENGINE

Fuzzy inference is the process of mapping a given input to an output using fuzzy logic. The process is comprised of 5 parts. Firstly, the inputs are made fuzzy. Secondly, the fuzzy operators are applied. In the next step, the Implication is assigned. In the fourth step, the outputs are aggregated and finally, the output is taken. The classical "tip problem" as used by Gulley is presented to illustrate how the whole process works. In this problem, the "right" tip for a dinner at a restaurant will be determined by using the two-input one-output fuzzy inference system defined in Fig. B.4. The inputs are "service" and "food" whereas the output is "tip".

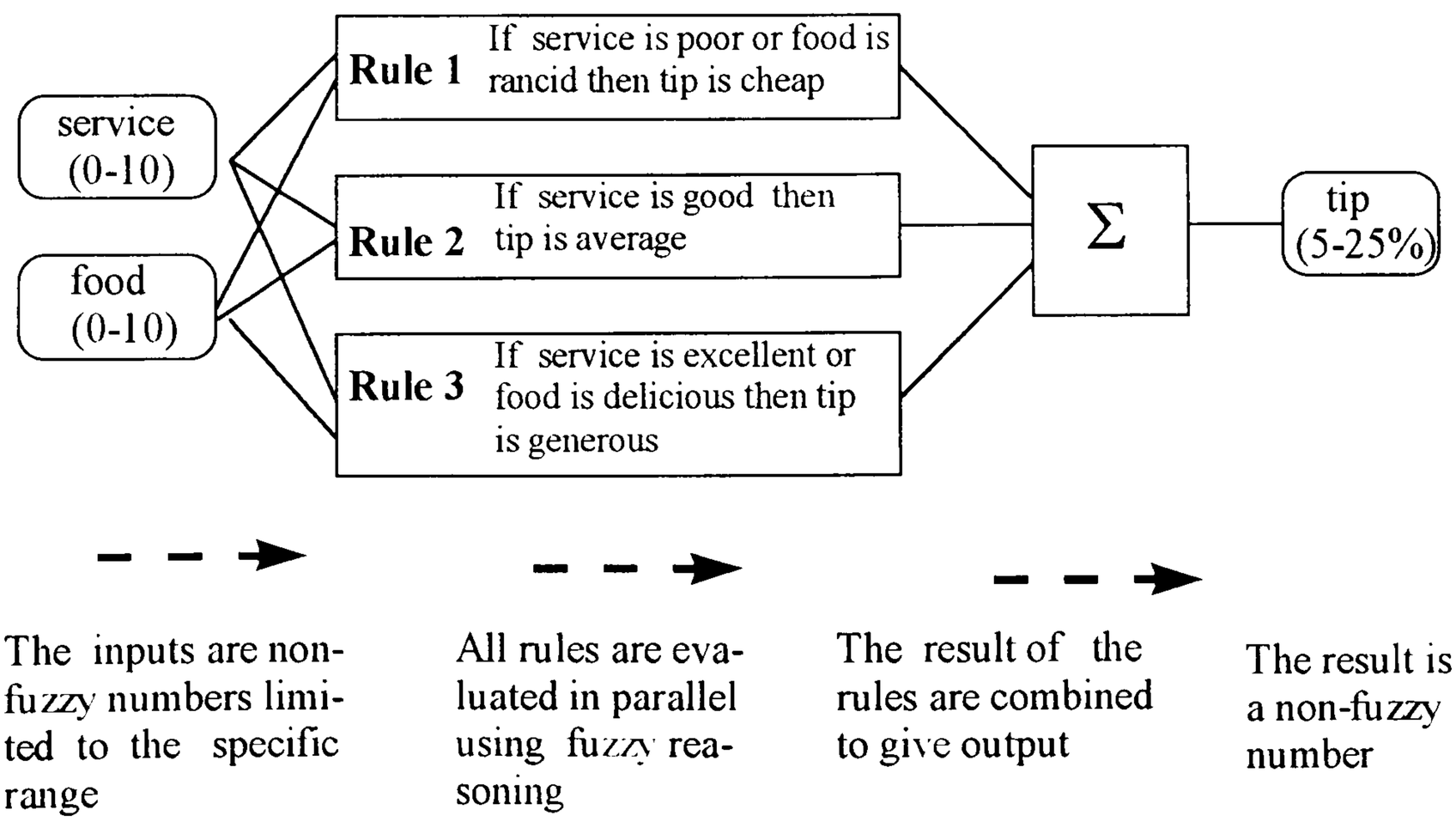



Fig. B.4 - Tipping problem as used by Gulley.

Each variable, "service", "food" and "tip" is linguistic and has its own membership functions shown in Figs. B.5, B.6 and B.7. The shape of the membership functions could be anyone since the degree of membership varies between 0 and 1. The range is defined according to the necessities. In this particular problem, "service" and "food" was assigned by Gulley to range from 0 to 10. The tip may vary from 5% to 30%. According to the "food" variable membership function shown in Fig. B.7, its influence will be taken into account only if "food" is rancid or delicious.

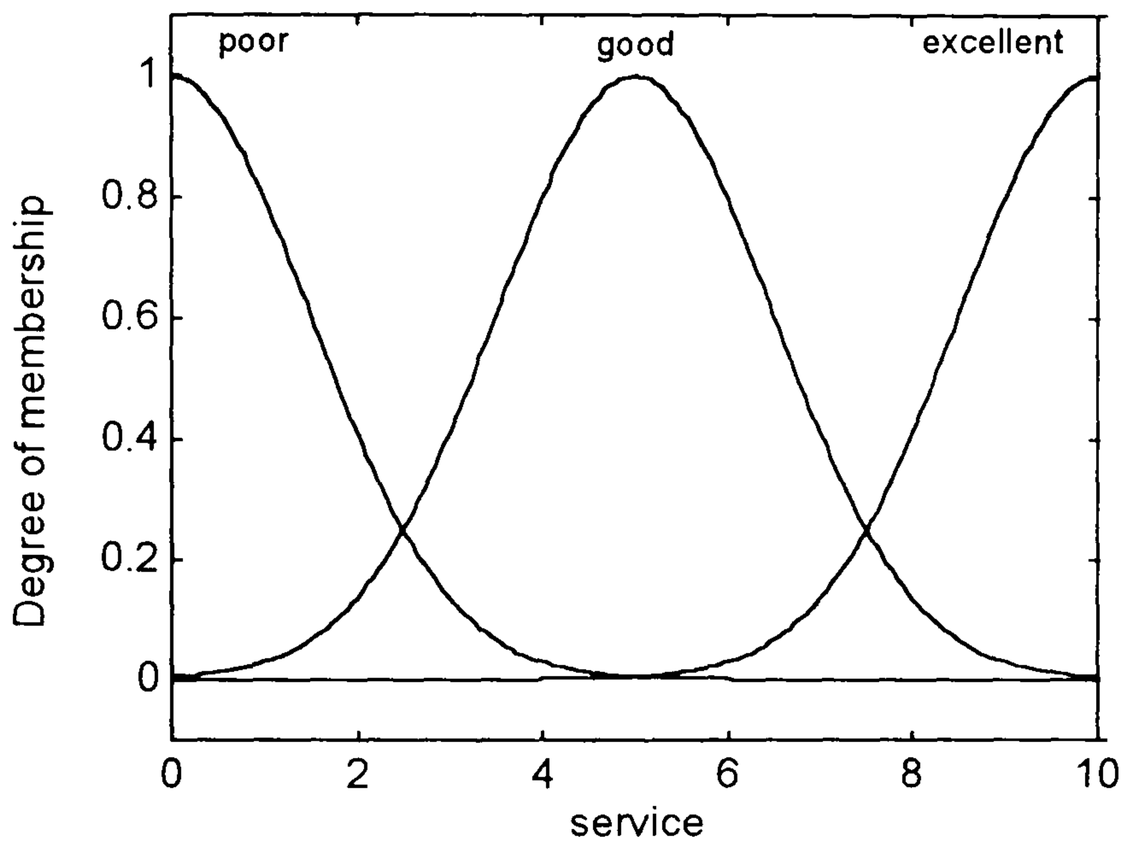



Fig. B.5 - "Service" input membership functions.

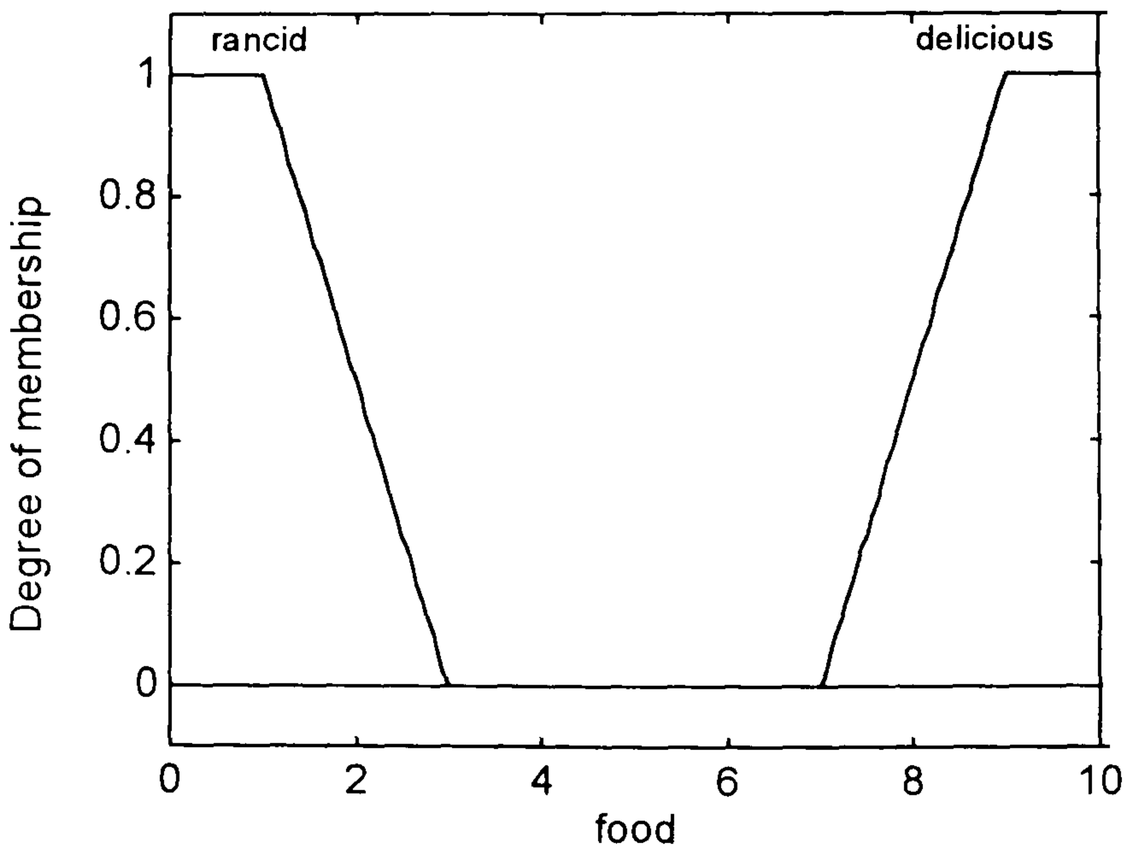



Fig. B.6 - "Food" input membership functions.

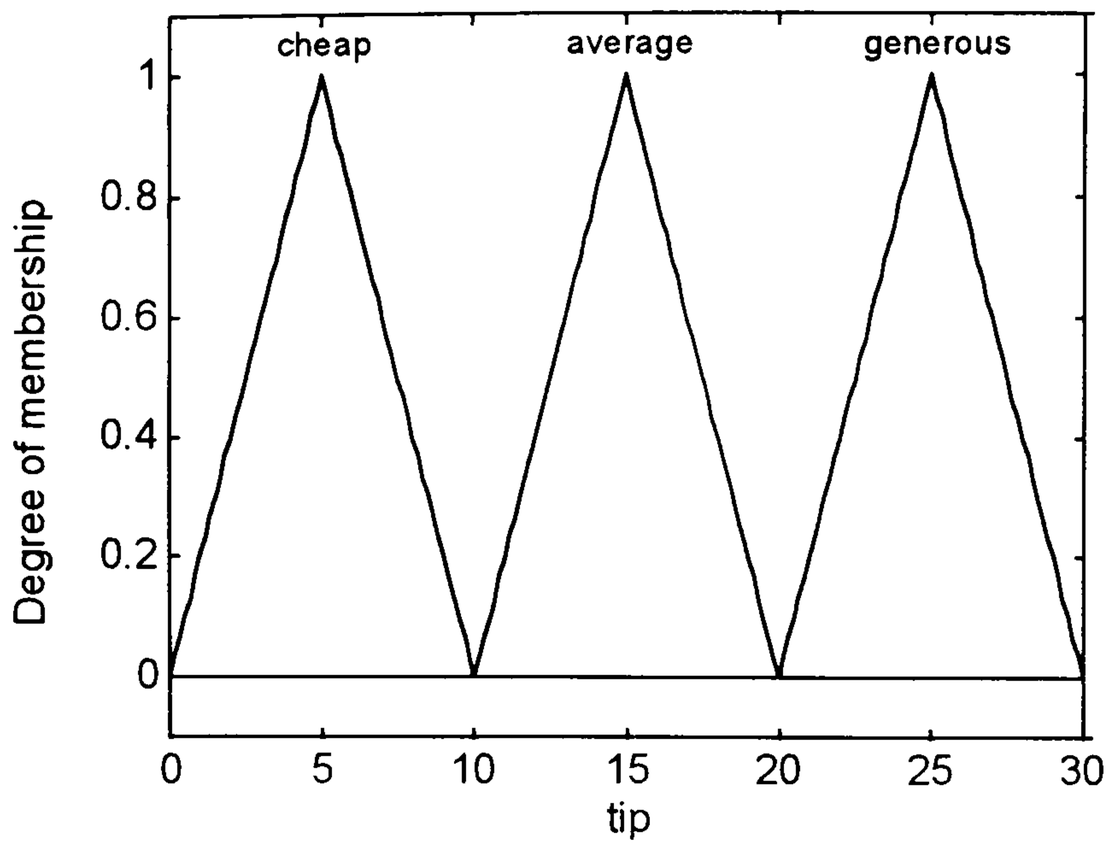



Fig. B.7 - "Tip" output membership functions.

The rules that map the inputs to the output are shown in Fig. B.4 above. How the fuzzy reasoning works is presented as follows.

### B.6.1 - Making the inputs fuzzy

In this step the inputs are taken in order to determine to which degree they belong to each of the appropriate fuzzy sets via MF. The inputs are always non-fuzzy number limited to the universe of discourse or range defined (0 to 10 in this particular case). The output is a fuzzy degree of membership (μ), between 0 and 1. If the food is rated 8 for instance, it means that it corresponds to 0.5 to the degree of delicious previously defined, as seen in Fig. B.8.

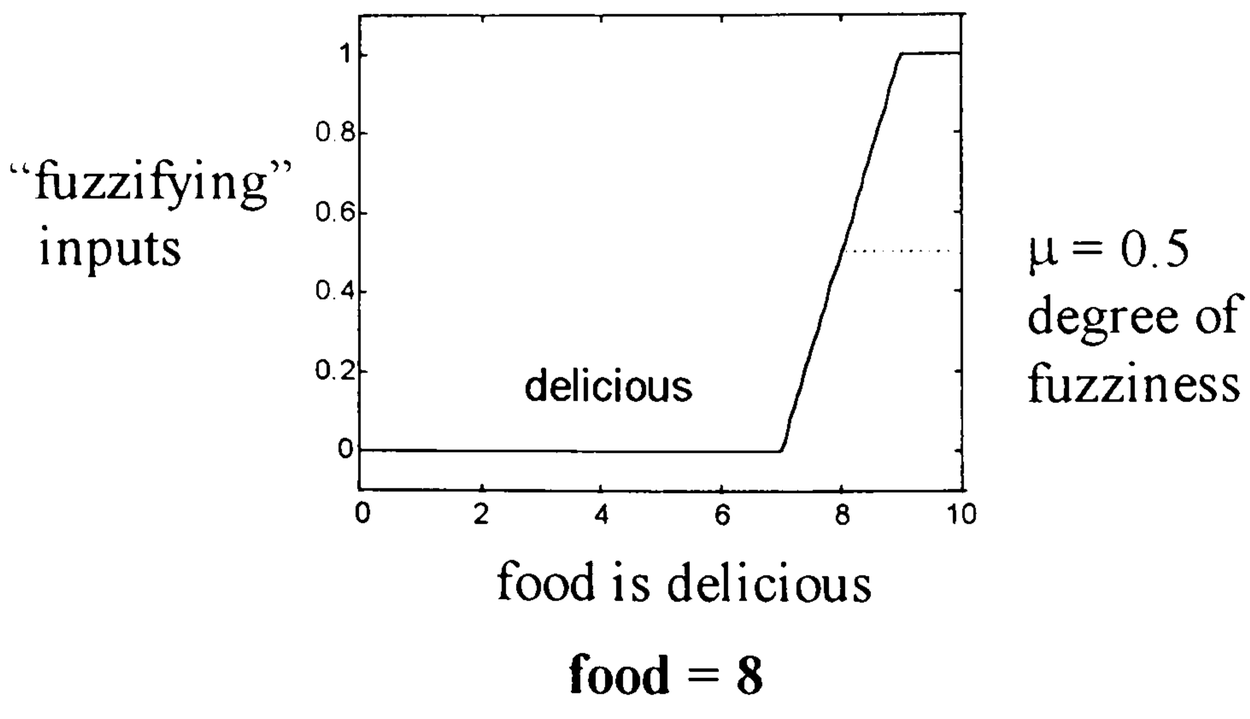



Fig. B.8 - Making the "food" input fuzzy.

### B.6.2 - Application of the fuzzy operators

After making the inputs fuzzy, the degree to which each part of the antecedent has been satisfied for each rule, is known. In the case the antecedent has more then one part, the fuzzy operator (AND, OR or NOT) is used to obtain one number that represents the result for that rule. This number is then applied to the output of the fuzzy operator. The graph of Fig. B.9 illustrates the process.

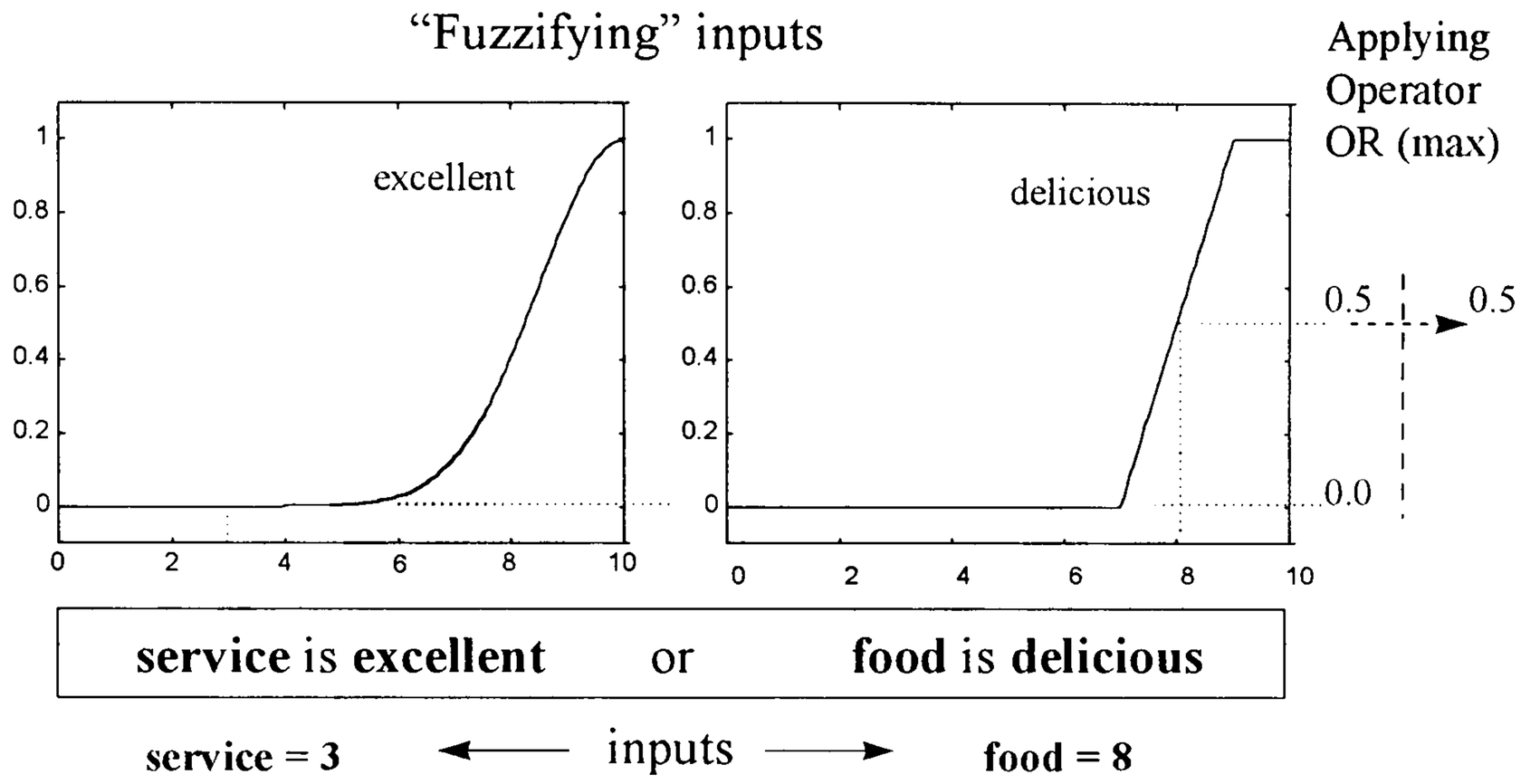



Fig. B.9 - Application of the fuzzy operator.

The fuzzy output due to the "service" input equal to 3 is $\mu = 0.0$ whereas for the "food" input equal to 8 is $\mu = 0.5$. After applying the fuzzy operator OR, the result is max(0.0, 0.5) = 0.5. Thereupon, the fuzzy value for the antecedent part of that rule is 0.5. This value will be applied to the consequence.

### B.6.3 - Implication

The implication is the result of the application of the antecedent result to the consequence. The result of the antecedent is a single number, resulted from the application of the fuzzy operator to the fuzzy inputs for each rule. The result of the implication is a fuzzy set. The method used for the implication is the min (minimum) which truncates the fuzzy output set, as shown in Fig. B.10.

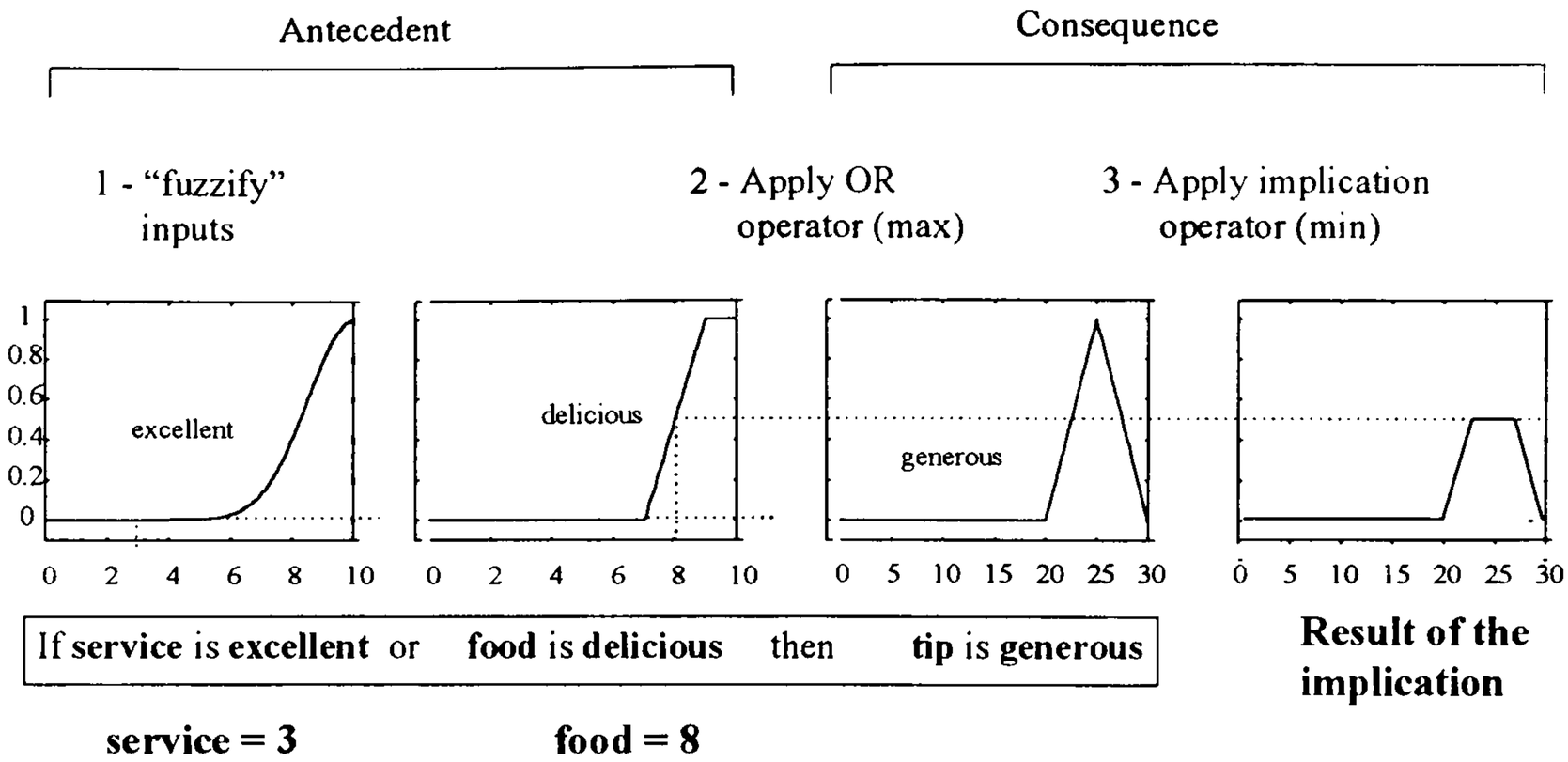



Fig. B.10 - Use of the implication operator.

It can be seen that the antecedent has been partially satisfied and so the consequence. This is applied for each rule.

### B.6.4 - Aggregation

Aggregation is the step where the outputs of each rule are put together to form a single fuzzy set. It means that the contribution for each rule is taken into account even if it is zero. Fig. B.11 illustrates how it is made in fuzzy logic from the beginning.

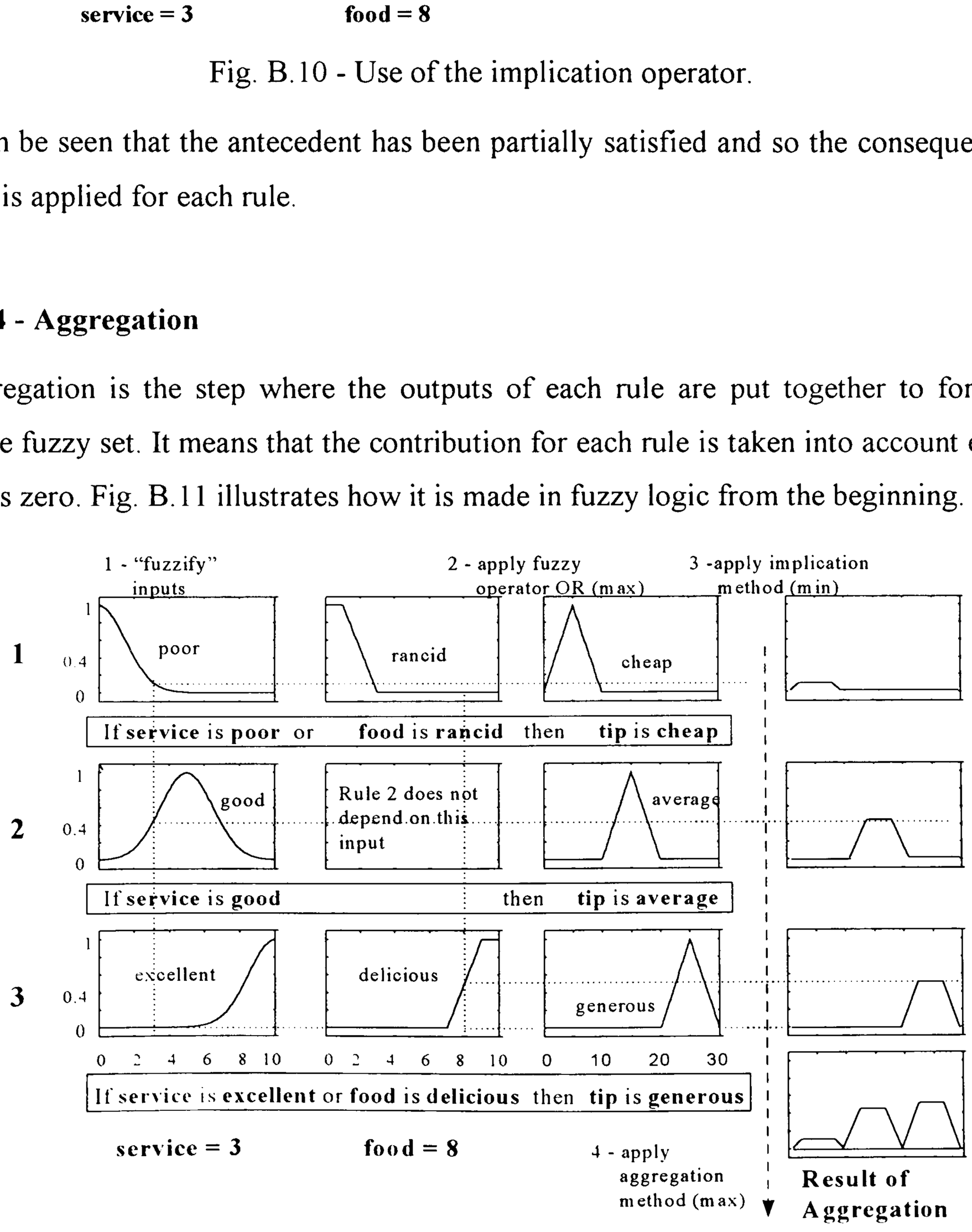



Fig. B.11 - Process until aggregation step.

The fuzzy set seen in Fig. B.11 as a result of the aggregation process shows how each rule is satisfied. That picture will then be used to determine the single numerical output value, in the final step.

### B.6.5 - Taking the output

Usually called "defuzzification", this step is where the fuzzy set resulted from the aggregation process is used to determine the single numerical output value. There are several methods available to do so. However, the most common is the *centroid* calculation. This method simply determines the center of the area under the curve that represents the fuzzy set after the aggregation. In the example shown above, the figure resulted from the previous step is shown as follows as well as the result of the calculation of the centre of the area.

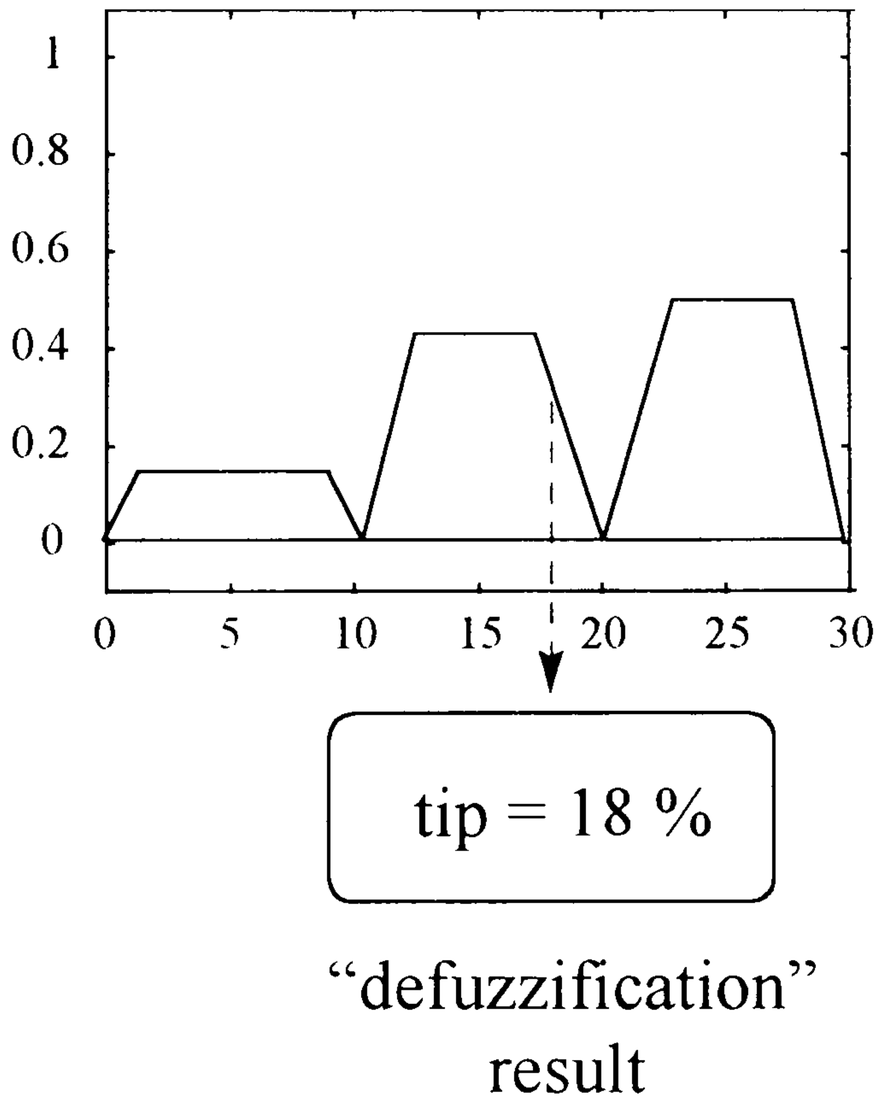


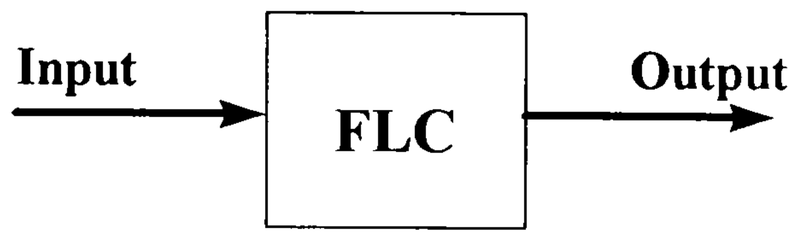

Fig. B.12 - Taking the output value from fuzzy set.

Then, the single numerical output value for the inputs "service = 3" and "food = 8" is 18 %. Thereupon, the reasoning of fuzzy logic can be used in many different problems. What is needed is to determine what the inputs and outputs variables are, their ranges and the rules to map the inputs to the outputs.

This is the end of the example given by Gulley. Next another example is given focusing on the influence of the shape of the membership functions.

## B.6.7 - Another simple example focusing on the influence of the shape of the membership functions

Consider the following single-input single-output fuzzy logic controller:




The input is any variable that can vary from 0 to 100%. The output range is defined to be between 0 and 100%. Their membership functions are defined as shown in Fig. B.13.

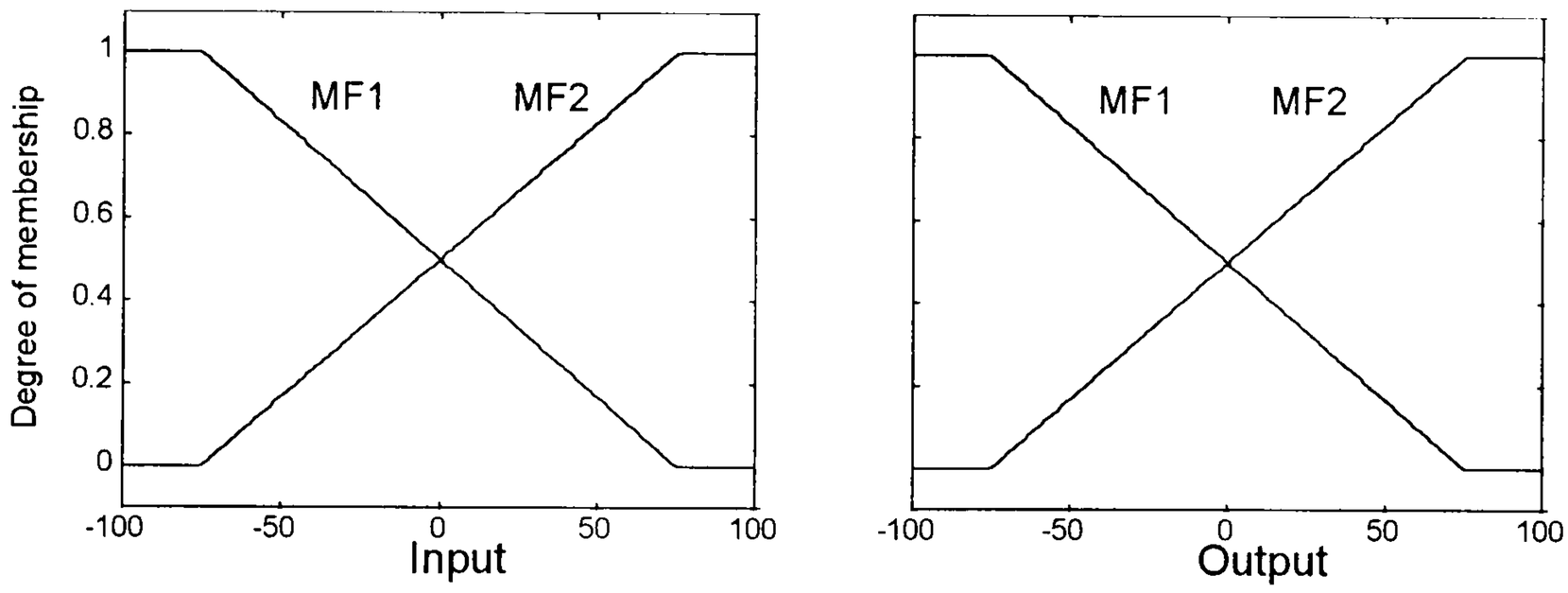



Fig. B.13 - Input and output membership functions.

Two rules are used to map the input to the output as follows:

1 - If *input* is MF1 then *output* is MF1.

2 - If *input* is MF2 then *output* is MF2.

Assuming an input 0.8 or 80%, the output will be equal to 41.2% as illustrated by the map shown in Fig. B.14.

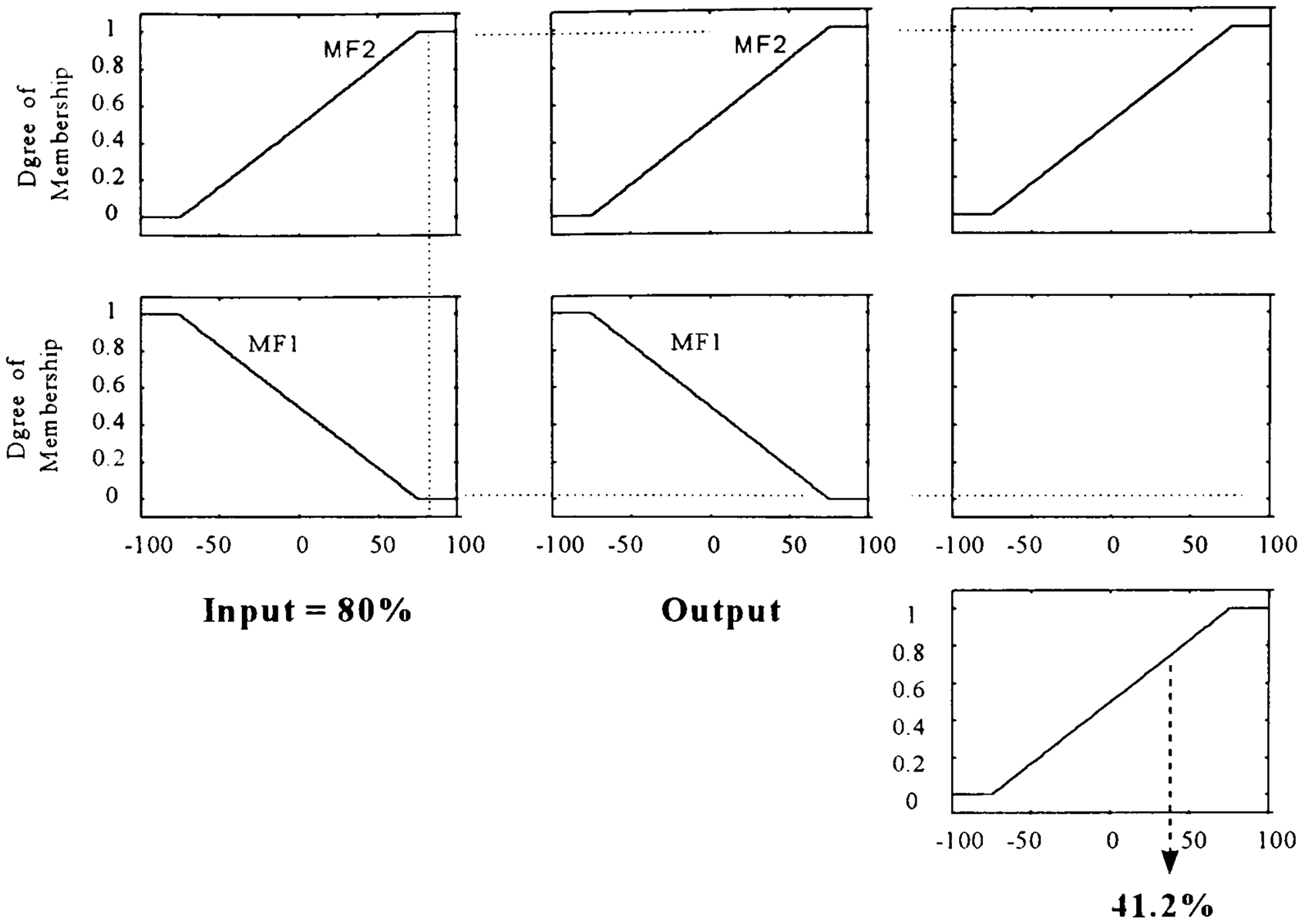



Fig. B.14 - Another example - input equal to 80%.

Now, instead of input equal to 80%, an input of 20% is considered. The output is shown in Fig. B.15.

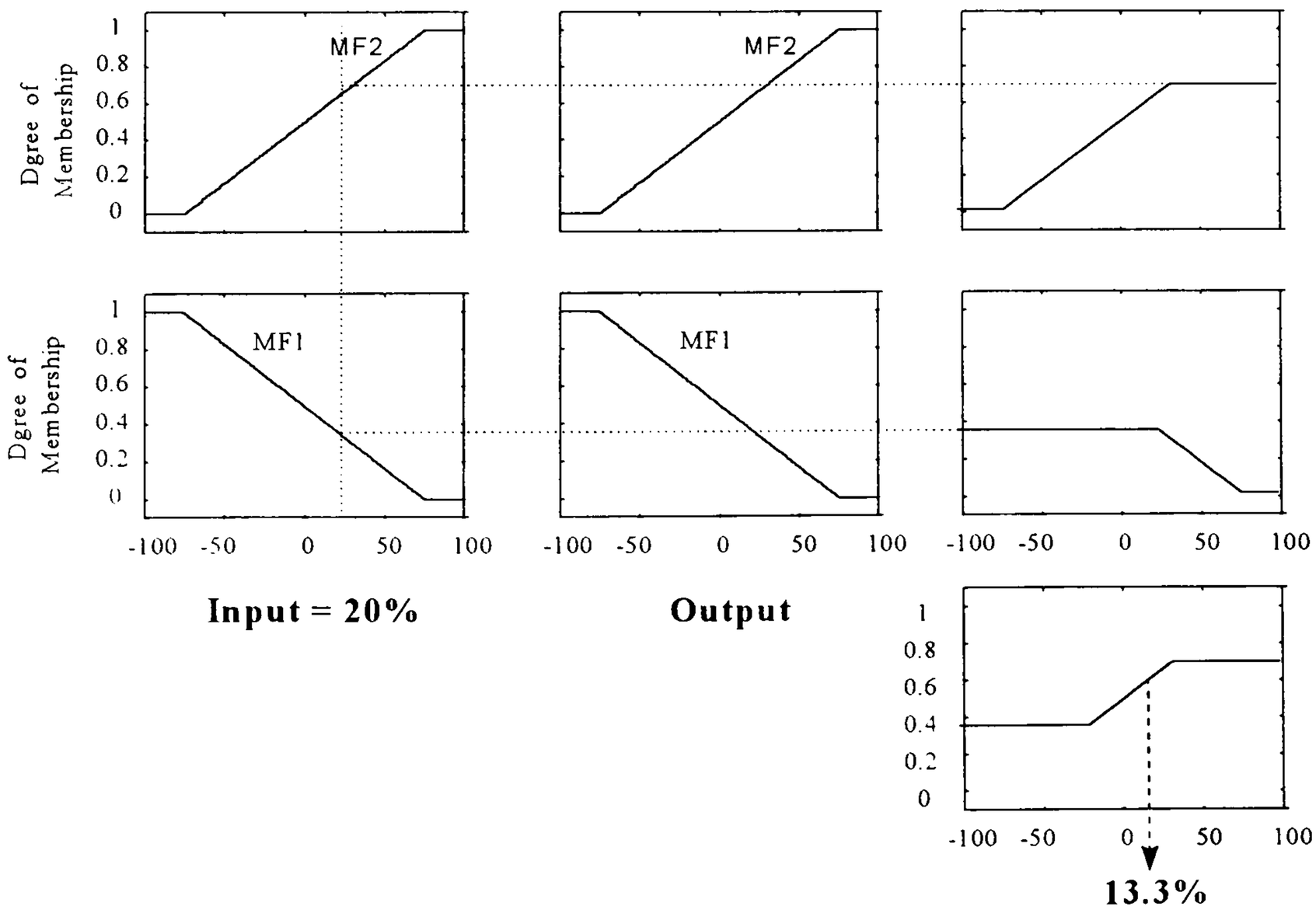



Fig. B.15 - Another example - input equal to 20%.

Then, from varying the input within its range, the output changes according to the following curve which represents the transfer function between input and output.

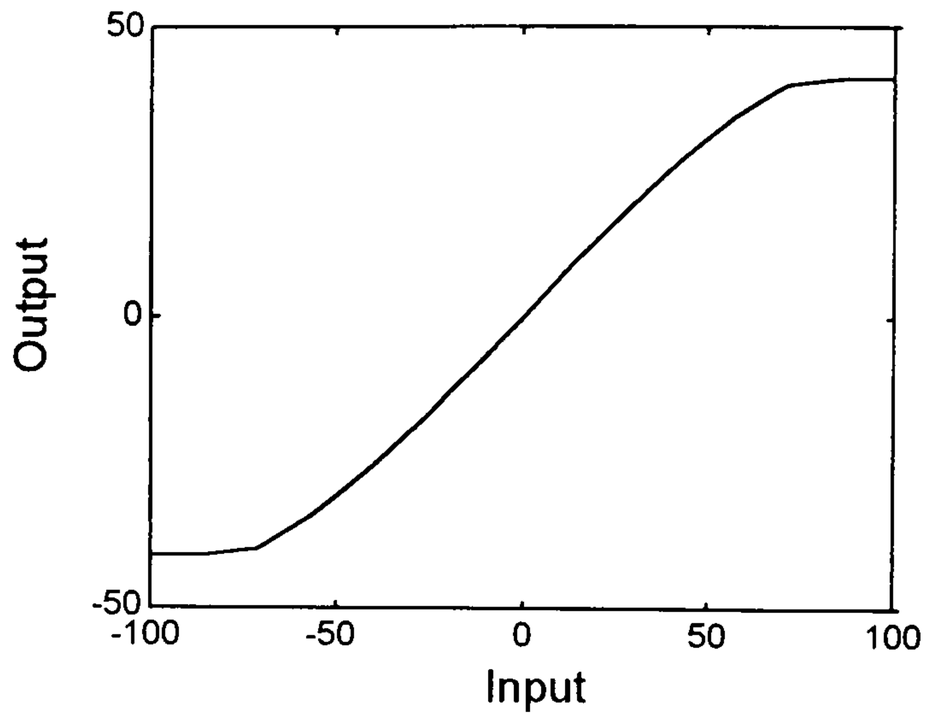



Fig. B.16 - Transfer function of the fuzzy system of the example above.

This relationship can be completely changed if the membership functions and/or the rules are different. Thereupon it is possible to work with fuzzy logic in order to achieve input(s)/output(s) relationship that suits each desired purpose. This is an important characteristic of fuzzy logic controllers.

# APPENDIX C

# AN OVERVIEW ON GENETIC ALGORITHM

## C.1 - INTRODUCTION

The use of Genetic Algorithm for solving problems comes from 1970's when J. H. Holland's work proved to be a significant improvement for scientific and engineering application [MAN *et al*, 1996]. However, its application in engineering started to take of late in 1980's when Goldberg [GOLDBERG, 1989] published his book on this subject and computer facilities were no longer a problem.

Gen [GEN *et al*, 1997] has defined the Genetic Algorithm as a stochastic search technique based on the mechanism of natural selection and natural genetics. It presumes that a potential solution for a problem is an individual represented by a set of parameters regarded as genes of a chromosome. The chromosomes can be structured by a string of values, usually but not necessarily, in binary form. The chromosomes or individuals are evolved through successive generations. During each generation the chromosomes are evaluated according to a *fitness* criterion, the term usually used in genetic algorithms. The best chromosomes of each generation are then selected to mate and generate offspring. A new generation is then created and the best individuals are selected to replace the least fit chromosomes of the generation before so the population size is kept constant. In such mating and selection process, the fittest individuals have higher probability of being selected to generate offspring for the following generation. As a consequence, the offspring are likely to have better characteristics then their parents. After a number of generations the algorithm converges to the best chromosome which represents the optimum or near optimum solution for the problem. A simple structure of the Genetic Algorithm as given by Man *et al* is shown in Fig. C.1.

```
// Start with an initial generation
g =  0;
// initiate the population
initialise P(g);
// evaluate fitness of all individuals of the population
evaluate P(g);
// test for termination criteria (generation, fitness, etc.)
while not done do
        // select a sub-population for offspring reproduction
        P'(g) = select parents from P(g);
        // recombine the "genes" of selected parents
       recombine P'(g);
       // perturb the mated population stochastically
       mutate P'(g);
       // evaluate the fitness of the new individuals
       evaluate P'(g);
       // select the survivals from actual fitness
       P(g+1) = survival P(g), P'(g)
       // increase the generation counter
       g = g+1
end
```

Fig. C.1 - Simple structure of a Genetic Algorithm as given by Man *et al.*

The Genetic Algorithms is different from traditional search and optimisation methods such as hill-climbing and random search in several fundamental ways [GEN *et al*, 1997]:

1 - Genetic algorithms work with a coding of solution set, not the solution themselves.

2 - It searches from a population of solutions, not a single solution.

3 - It uses payoff information (fitness function), not derivatives or other auxiliary knowledge.

4 - Genetic algorithms use probabilistic transition rules, not deterministic rules.

## C.2 - GENETIC ALGORITHM OPERATIONS

In Genetic Algorithms only two kinds of operations are executed: the genetic operation and the evolution operation. The genetic operations are crossover and mutation whereas the evolution operation is the selection one.

### C.2.1 - Crossover operation

In this operation, two chromosomes are selected to generate offspring by randomly choosing a cut point and combining the segment to the left of the cut point of one parent with the segment to the right of the cut point of the other parent. Consider for instance, the chromosomes formed by binary numbers as follows.

Chrom 1: 1 1 1 0 0 1 0 0 0 1

Chrom 2: 1 0 0 0 0 0 0 0 1 1

A cut point is randomly selected after the fourth digit from the right:

Chrom 1: 1 1 1 0 0 1 0 0 0 1

Chrom 2: 1 0 0 0 0 0 0 0 1 1

The offspring resulted from the mating of the chromosomes above becomes:

Offspring: 1 1 1 0 0 1 0 0 1 1

The performance or fitness of the offspring depends largely on the performance of the crossover operator used. A crossover rate is defined as the ratio of number of offspring produced in each generation to the population size [GEN *et al*, 1997]. Then in Genetic Algorithms, this ratio controls the number of chromosomes selected to undergo the crossover operation.

### C.2.2 - Mutation operation

As in natural selection any individual may suffer mutation along the generations, in genetic algorithm a mutation operation is defined where one or more genes of a

chromosome is randomly changed according to a mutation probability. According to Gen, it is important to replace genes lost from the population during selection or provide genes that were not present in the first population. The mutation helps the genetic algorithm to investigate a possible solution for the problem over the entire searching space. Mutation rate has been defined as a percentage of the total number of genes in the population [GEN, 1997]. It controls the rate at which new genes are introduced into the population for test.

### C.2.3 - Selection operation

The selection operation is the process of selecting individuals for generating offspring. The two most commonly ways of doing this is either using the roulette wheel selection (RWS) or the stochastic universal sampling (SUS) [CHIPPERFIELD *et al*, University of Sheffield and MAN *et al*, 1996]. In the roulette wheel selection, the individuals are probabilistically selected based on their "goodness" for solving the problem. Chipperfield [CHIPPERFIELD *et al*, University of Sheffield] has explained the way it works, as follows.

A real-valued interval *Sum* is determined either by the sum of the individual's expected selection probability or the sum of the fitness values of all the individuals in the current population. Then, the individuals are mapped one-to-one in adjacent intervals in the range [0, Sum], as shown in Fig. C.2. The size of each individual interval corresponds to the "goodness" of the individual: the better the individual the larger the space it takes in the wheel. In Fig. C.2 the individual number "6" is the fittest whereas the individual number "1" is the least fit one.

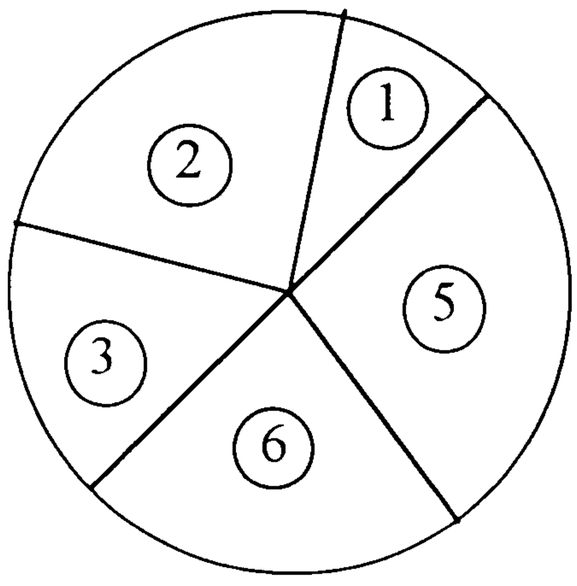



Fig. C.2 - Roullete wheel selection.

In order to select an individual, a number is generated randomly within the range [0, Sum]. The individual whose segment of the wheel spans the number is selected. This process is repeated several times until the wanted number of individuals have been selected. The individual that takes larger part of the roulette wheel is more likely to be selected.

### C.2.4 - Stochastic universal sampling

Chipperfield has defined this selection method as single-phase sampling algorithm with minimum spread and zero bias. It means that differently from the roulette wheel where a single selection pointer is employed, in stochastic universal sampling $N$ equally spaced pointers is used where $N$ represents the number of selection required. The order of the population is then changed at random and a single number, *ptr*, in the range [0, *Sum*/$N$] is generated [CHIPPERFIELD *et al*, University of Sheffield]. Then, $N$ individuals are chosen by generating N pointers spaced by 1, [*ptr*, *ptr*+1,...,*ptr*+$N$-1] and selecting the individual whose fitnesses span the position of the pointers.

A graphical example of the mechanics of the Genetic Algorithm has been given by Gen and is shown in Fig. C.3. However, in a more elaborated genetic algorithm the degree of complexity increases although the structure is kept the same as above. Many different parameters may be set up such as the form of the crossover, mutation probability, population size, number of individuals of each population selected for mating and the number of new individuals of the new population that shall be reinserted into the old population after being evaluated. In addition to this, should a real value population be used rather than binary numbers, some other changes and parameters have to be adjusted. The parameters settings change from problem to problem and there is no rule to be followed although some guidance may be found in the literature.

The genetic algorithm toolbox for use with MATLAB, as created by Chipperfield is quite flexible and offers a wide range of different possibilities of setting up a genetic algorithm problem.

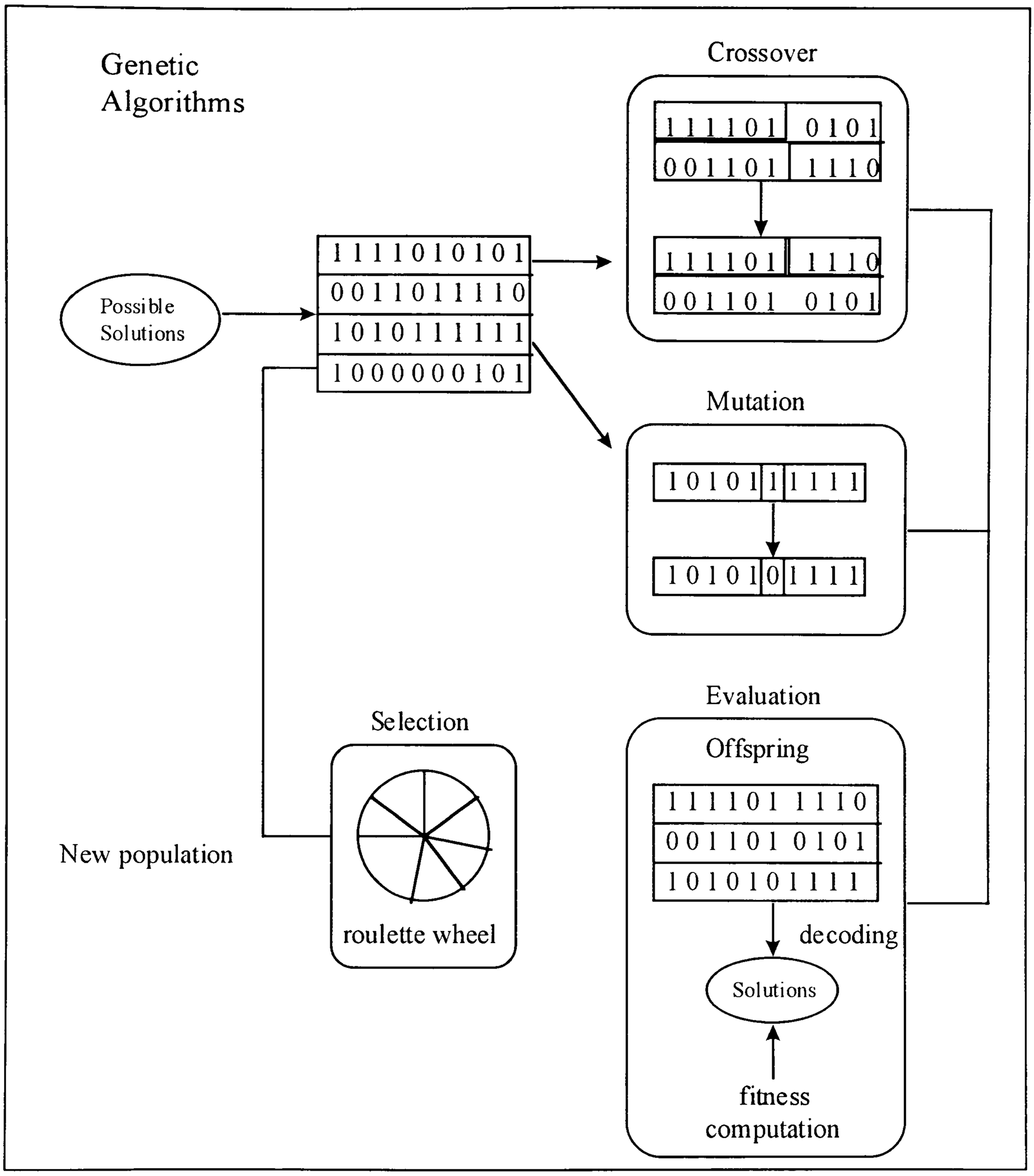



Fig. C.3 - Graphical demonstration of a genetic algorithm.

Also available in the toolbox are some examples of its use. Since the theory is a lot more extensive then what has been shown here, it is left for those interested in deeper understanding to look for. However, the references given are strongly recommended.

# APPENDIX D

# ANALYSIS OF THE CASTING DRUM DC MOTOR DRIVE ON THE LINE 51 OF DuPONT

## D.1 - INTRODUCTION

The analysis of the motor drive is based on the block diagram model supplied by DuPont, according to the document "Paper Model of Casting Drum Simulation", written by B W Rutter in October/93. The block diagram is shown in shown Fig. D.a, which is the simplified model that can be used for analysis purposes.

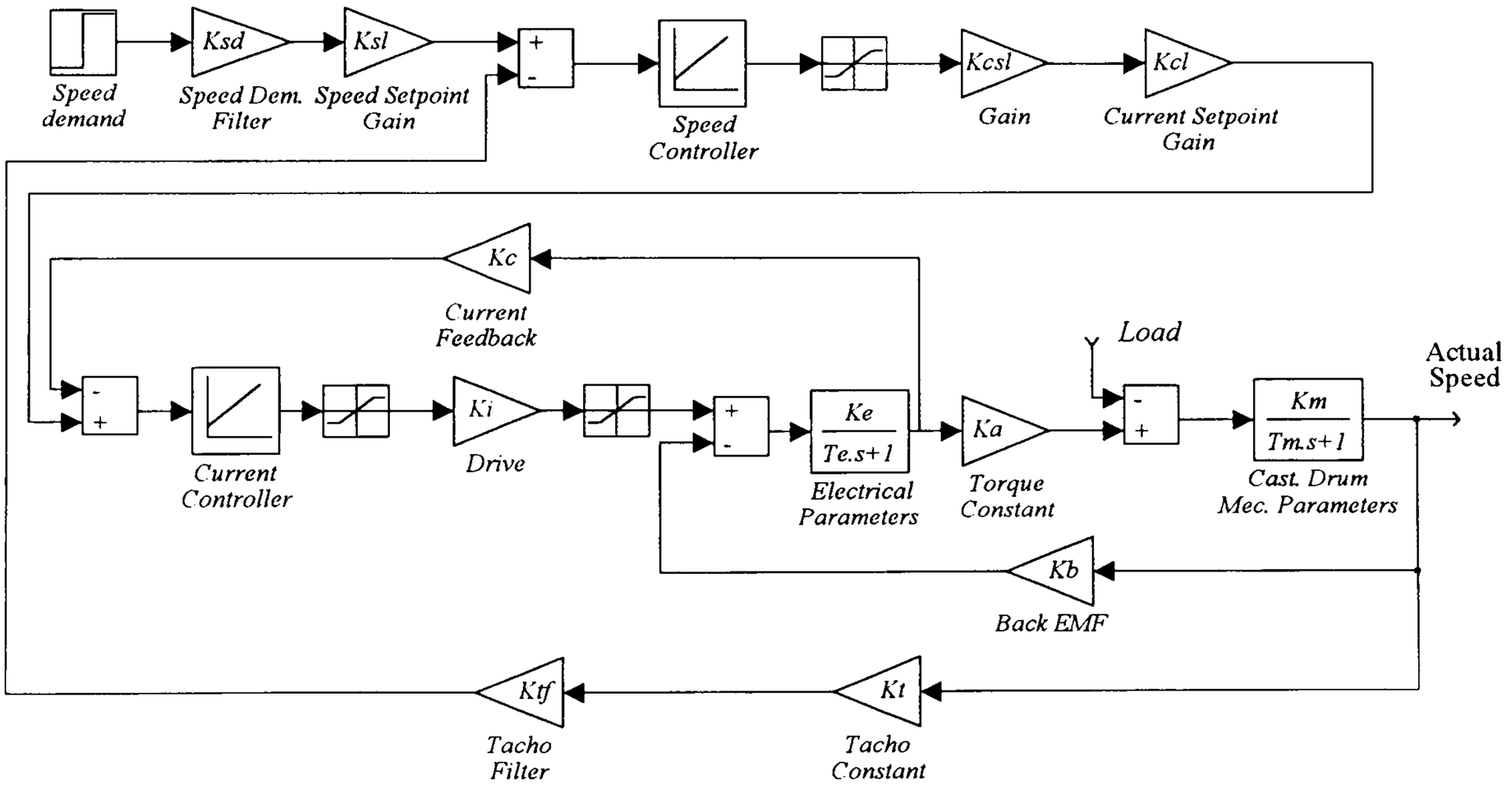



Fig. D.a – SIMULINK block diagram of the casting drum dc motor drive of Line 51.

As the current and speed control are based on analogue operational amplifiers, saturation blocks have been placed at the output of the controllers, representing the maximum output value they can reach. In addition to this, another saturation block with 200 V limit has been placed immediately after the block which represents the converter, as this is the maximum voltage which can be applied to the motor.

The parameters of the dc motor as well as the speed and current controller together with other gains across the block diagram, have been derived according to the equations shown later in this appendix, in section Di. Simulation using MATLAB/SIMULINK has been run for the speed demand of 2 rpm.

No information is given regarded the maximum torque the motor can produce. In that case, 150 Nm constant load torque is used. This gives a steady state armature current around 6 A, which is close to the current level experienced in Line 51.

## D.2 - SIMULATION RESULTS

The dc motor drive has been simulated within the conditions explained above. The results are as follows. Fig. D.1 shows the speed and current response of the motor.

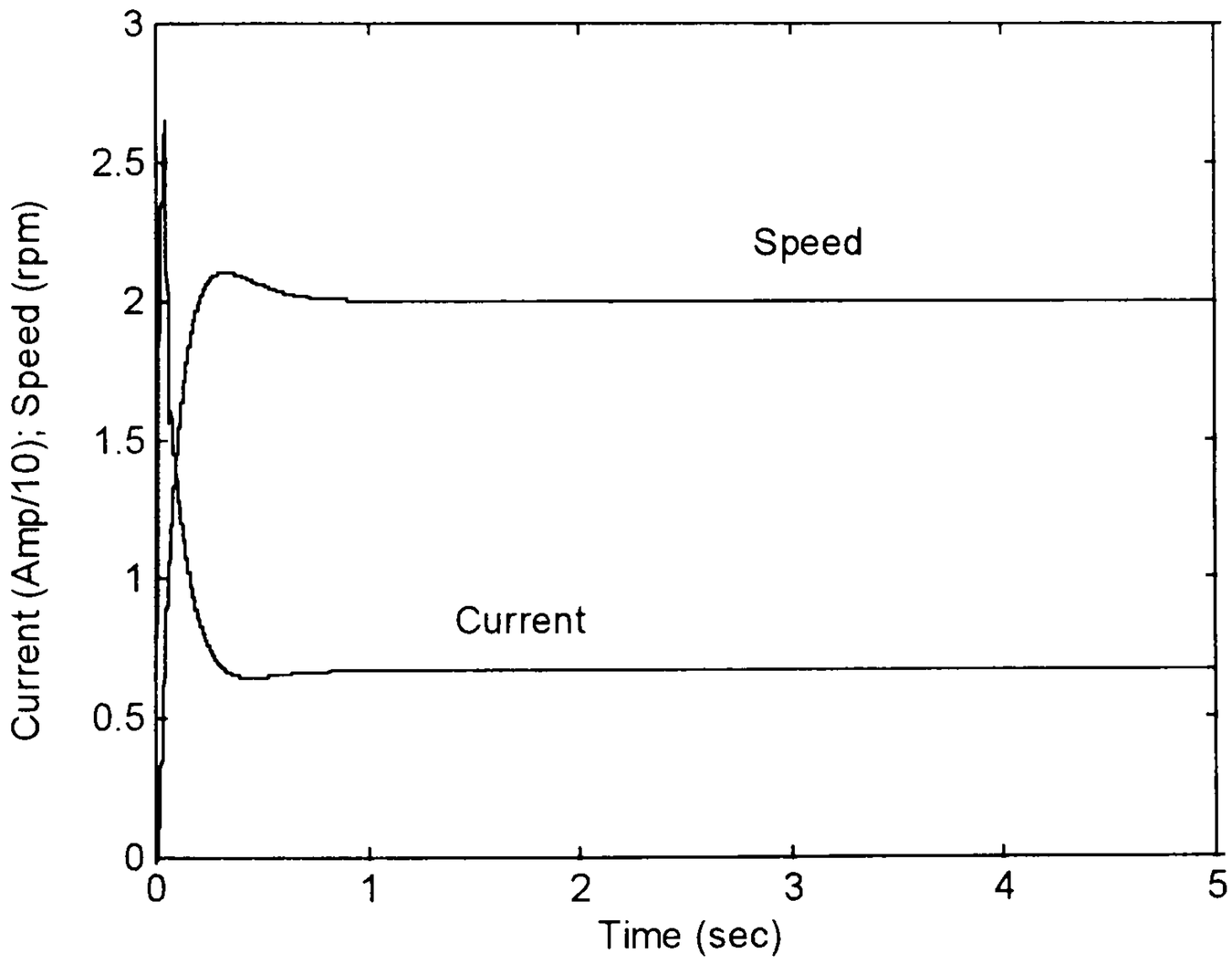



Fig. D.1 – 2 rpm step input speed and current response of the dc motor.

Figs. D.2 and D.3 depicts the speed and current controller output signals. Due to the gains imposed (proportional gain), the output of the controllers saturates at 15 V during the starting transient but stay within the linear range at steady state.

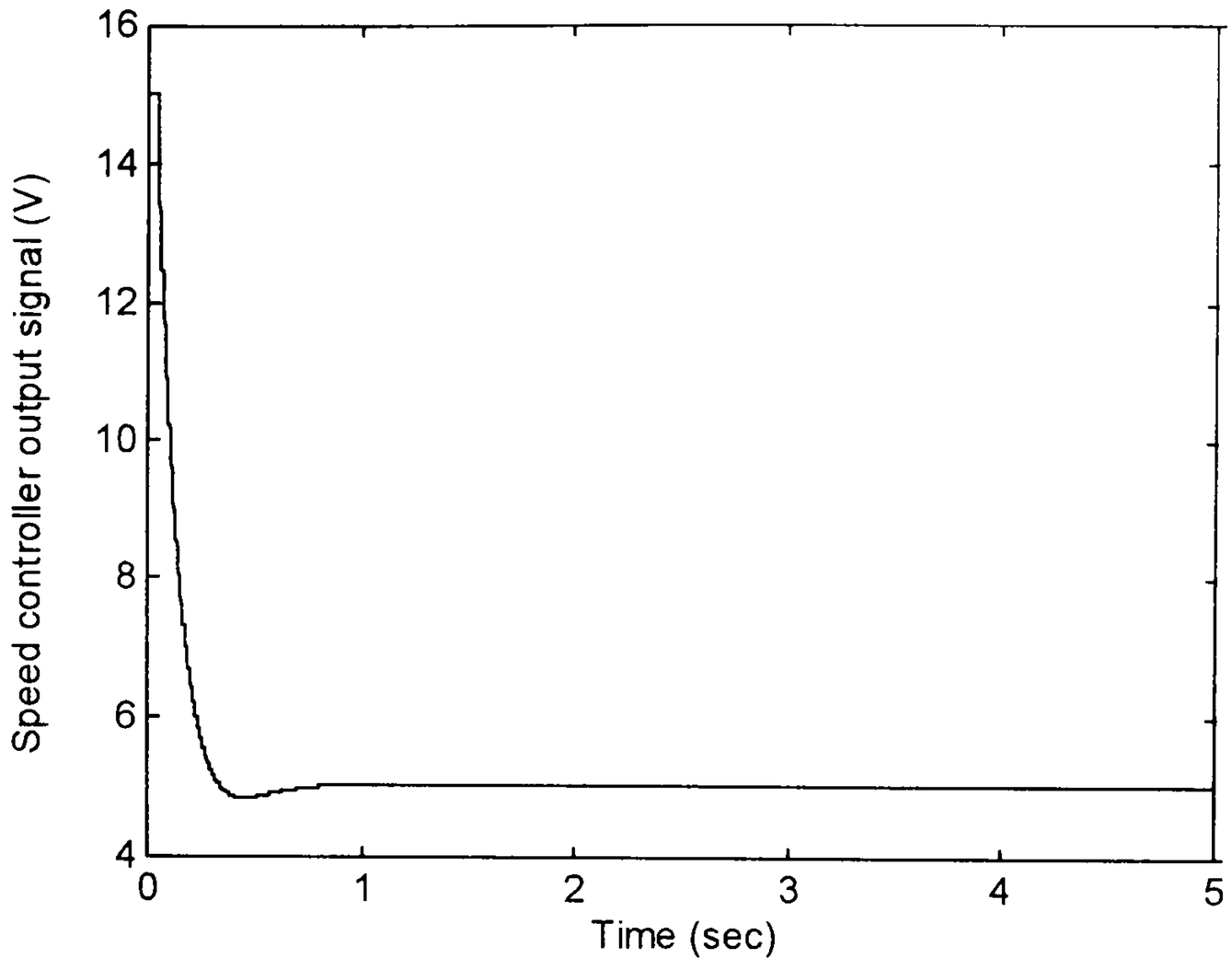



Fig. D.2 – Speed controller output signal.

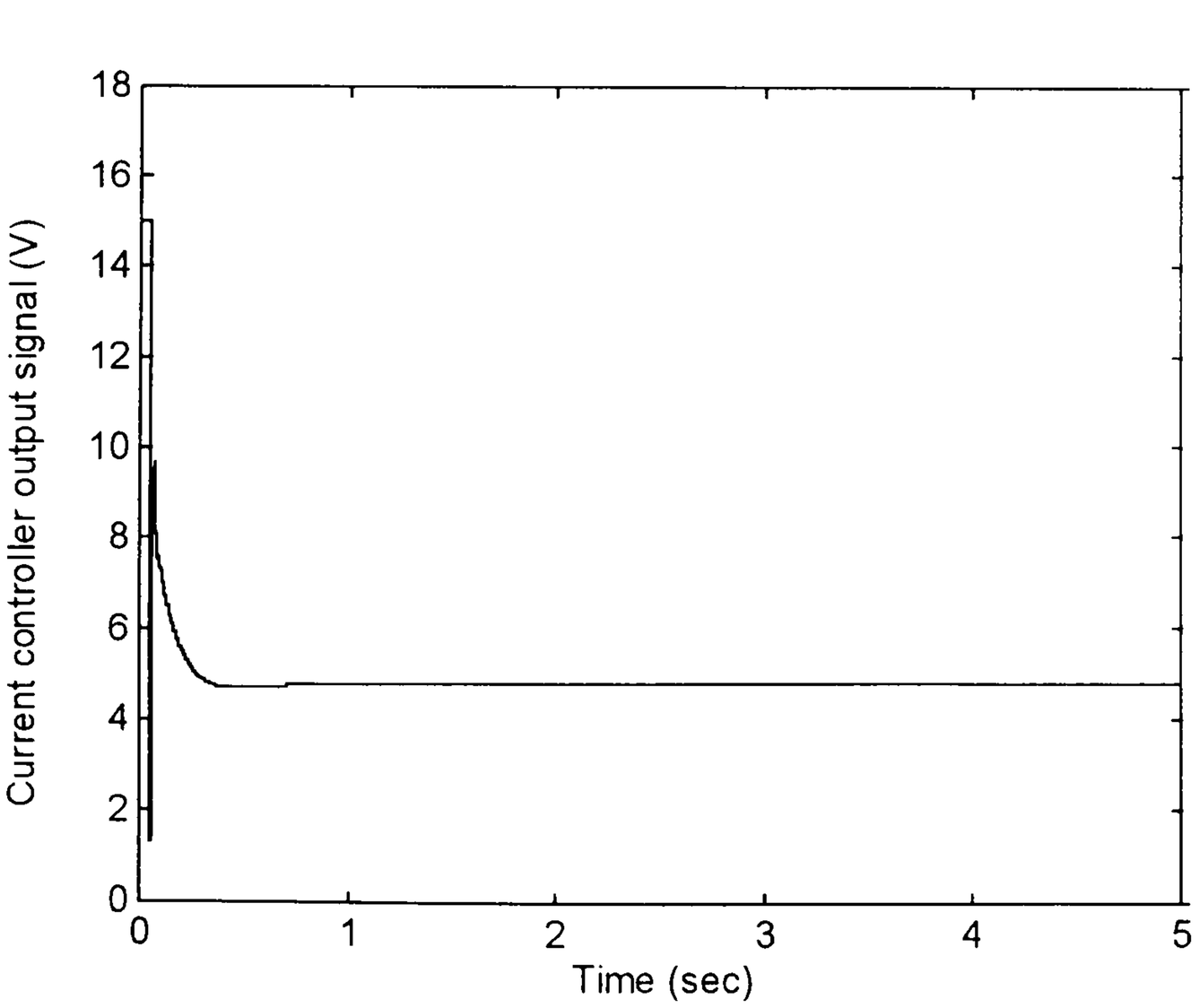



Fig. D.3 – Current controller output signal.

Fig. D.4 shows the armature voltage applied to the motor by the drive that supplies it, which limit was set to 200 V. However, since the drive is modelled by a constant

gain, $Ki$ = 4 V/V, that limit will never be reached as the maximum value is 15V (the maximum output of the Operational Amplifier) times $Ki$ (the gain of the drive), giving 60V.

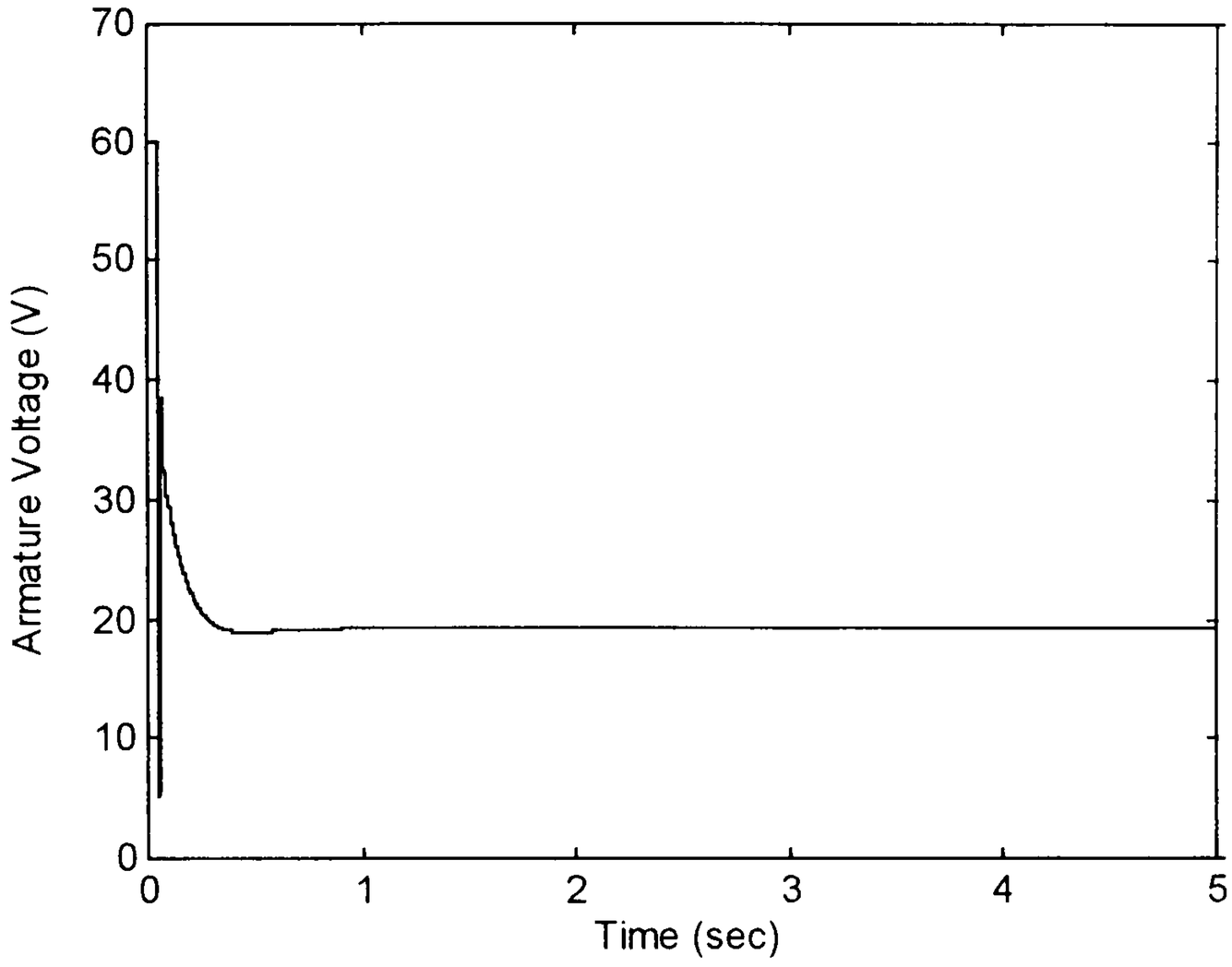



Fig. D.4 – Armature voltage of the dc motor due to the 2 rpm step input speed demand.

The results have shown that controller is adjusted so that a little underdamped speed response is obtained, with 5% overshoot.

## D.3 – THE IMPACT OF LOAD DISTURBANCE

The dc motor drive has been simulated within the conditions explained in the introduction, however, with the tuning of the controllers as used by DuPont, for a 2 rpm step input reference speed. In order to analyze the speed holding capability of the motor drive, a 10 Nm step input load disturbance is applied at *time* = 3s with the motor at steady state speed operation, lasting for 4s. The results are shown as follows. Fig. D.5 shows the actual speed and current response of the motor. The speed variation due to the disturbance is shown in enlarged scale in Fig. D.6.

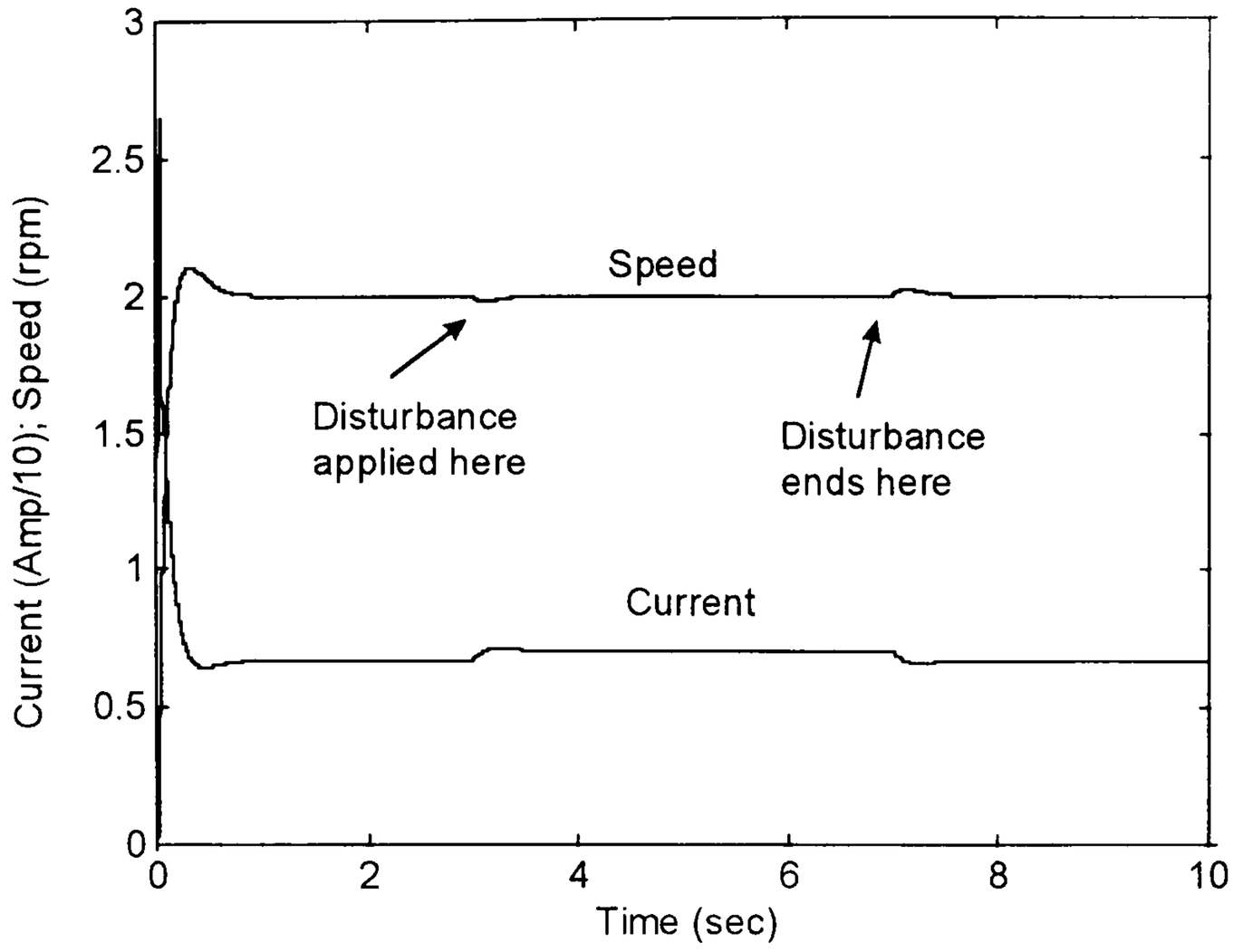



Fig. D.5 – Speed and current response of the dc motor drive in the presence of load disturbance.

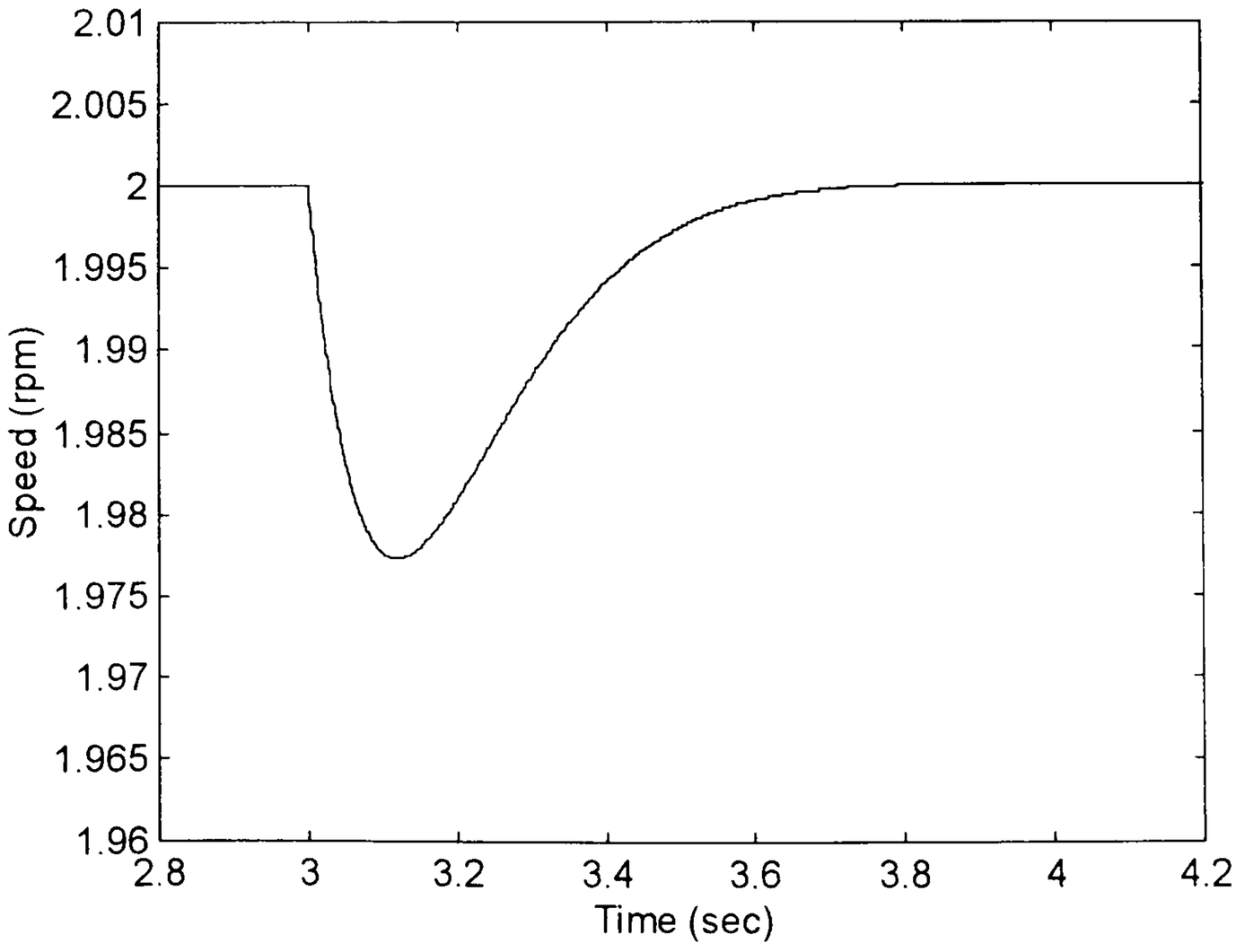



Fig. D.6 – Speed response in enlarged scale showing the impact of 10 Nm step input load disturbance.

## D.4 – THE LOAD ESTIMATOR

In order to improve the speed holding capability against load disturbance, a load estimator with torque feedforward, as proposed by Iwasaki [IWASAKI et al, 1991] is introduced, as shown in Fig. D.7 and in Fig. Aii (at the end of this appendix), which is the SIMULINK model used for simulation.

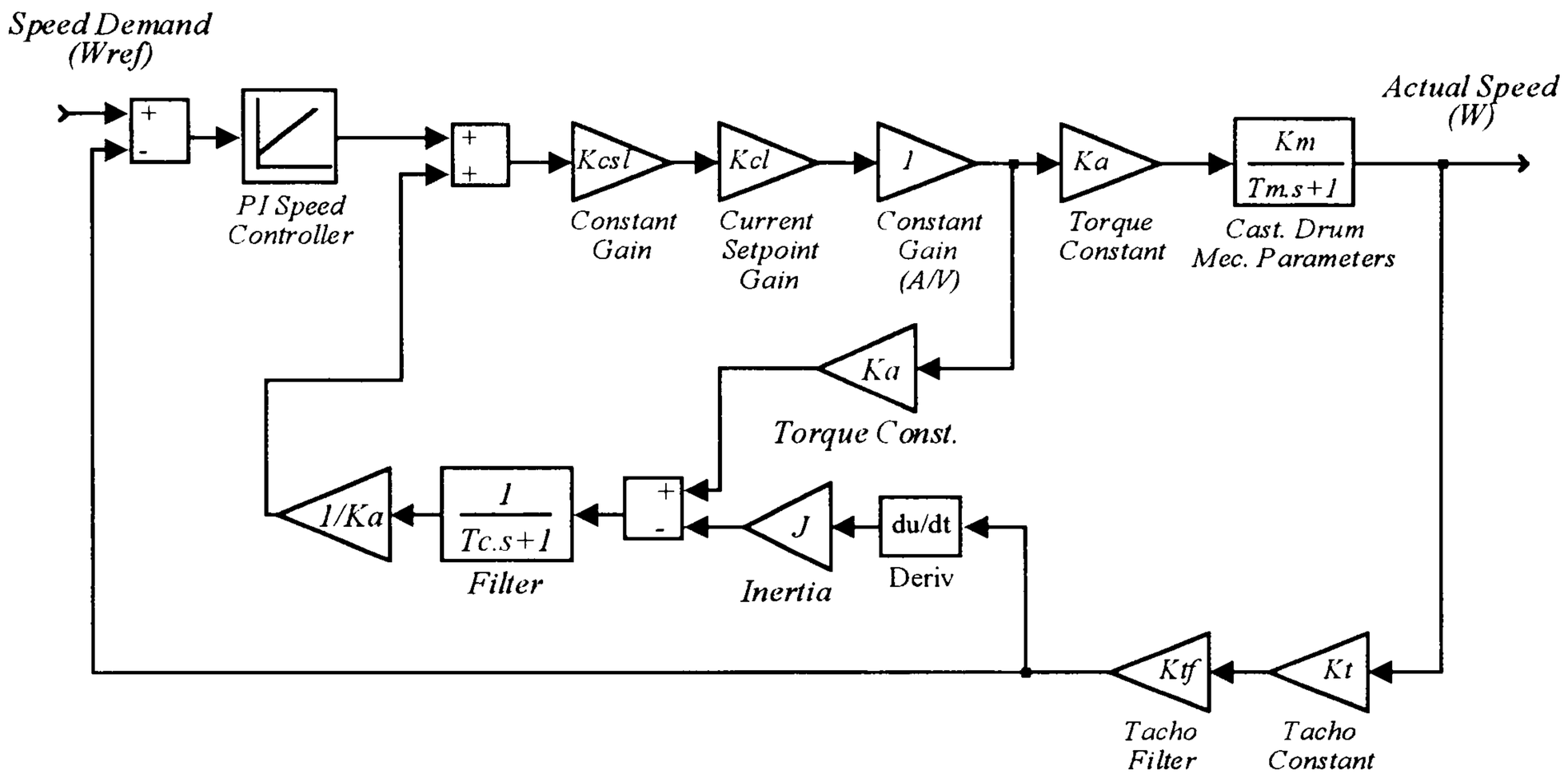



Fig. D.7 – DC motor block diagram with load disturbance observer and torque feedforward control.

Where,

$Tc$ – low pass filter time constant.

Simulation results show the effect the torque feedforward control on the speed holding capability against load disturbance. Fig. 8 shows the speed and current response due to 2 rpm step input speed demand, within the load conditions described in the previous section. Because of the load disturbance observer with torque feedforward control, a larger speed overshoot takes place during the starting transient. However, the speed drop due to the disturbance is smaller, although a little oscillatory.

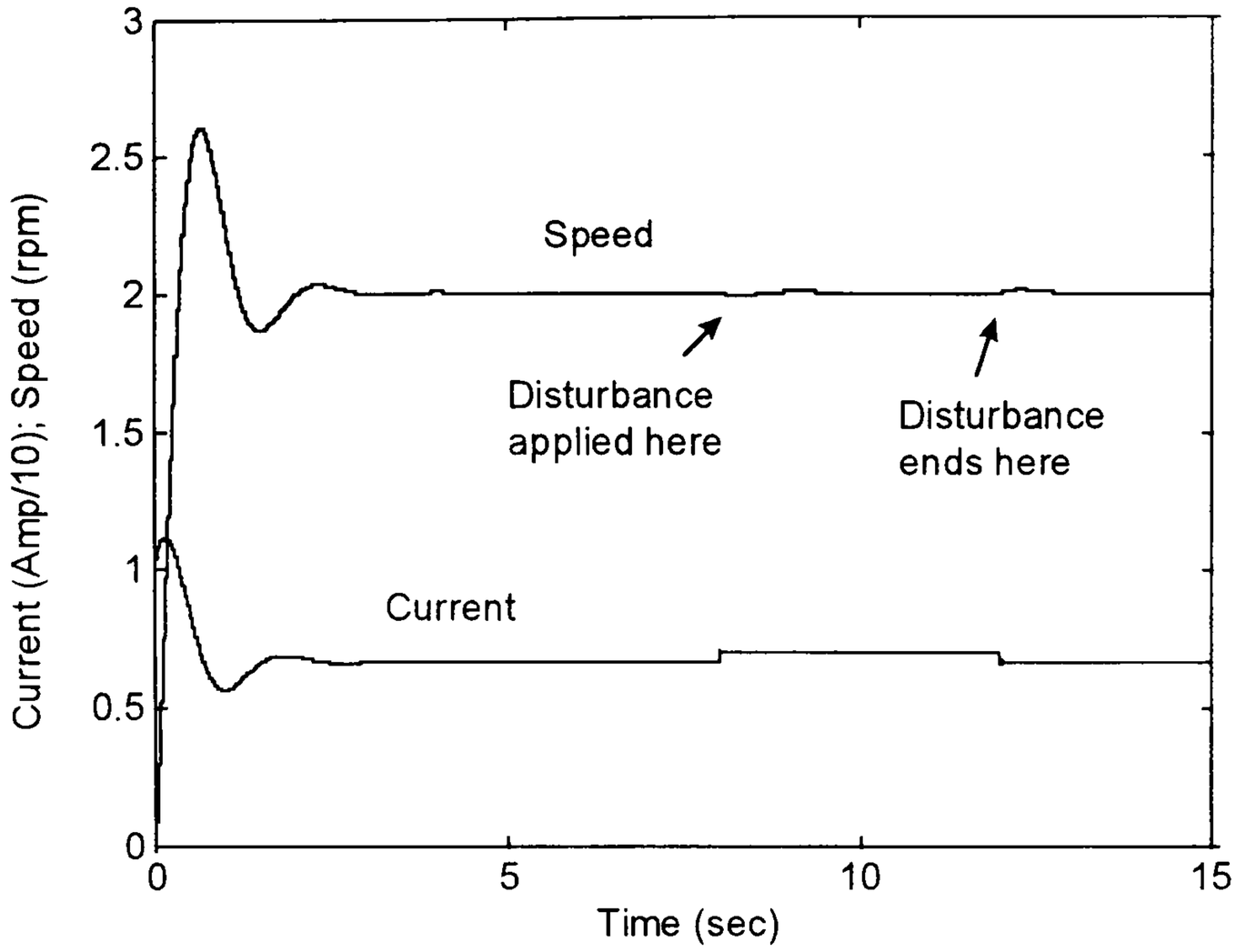
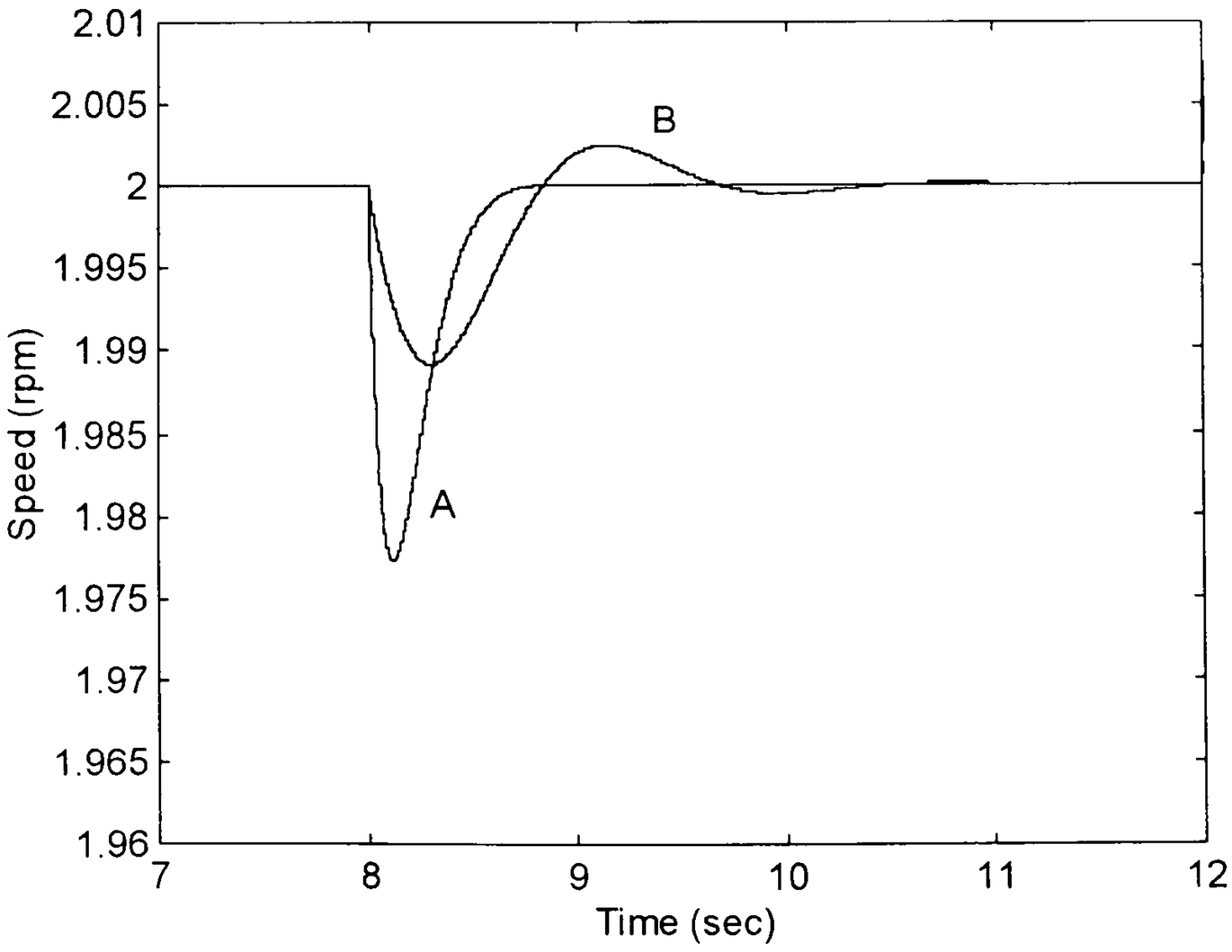



Fig. D.8 – Speed and current response of the dc motor drive with load disturbance observer and torque feedforward control.




"A" – without torque feedforward control; "B" – with torque feedforward control.

Fig. D.9 – Comparison of the speed holding capability of the dc motor drive with and without torque feedforward control.

The speed holding capability of the motor drive against load disturbance has improved because of the load disturbance observer and torque feedforward control. However, the tuning of speed controller plays an important role on the drive performance. In order to avoid the oscillation mentioned above, retuning it is necessary. To reduce the speed overshoot and oscillation due to the load disturbance, decreasing the integral gain or increasing the proportional one is needed.

## D.5 – THE EFFECT OF RETUNING THE SPEED CONTROLLER

In the case where the motor runs always at constant speed, the starting transient due to a step input speed demand is not important. Even if saturation of the controllers happens during the starting transient, due to a large step input speed change, the output of the controllers will stay within the linear region at steady state. Because of this, the gains of the controllers can be increased. The higher the proportional gain of the speed controller, the smaller the speed drop due to a disturbance. The higher the integral gain, the faster the speed recovery.

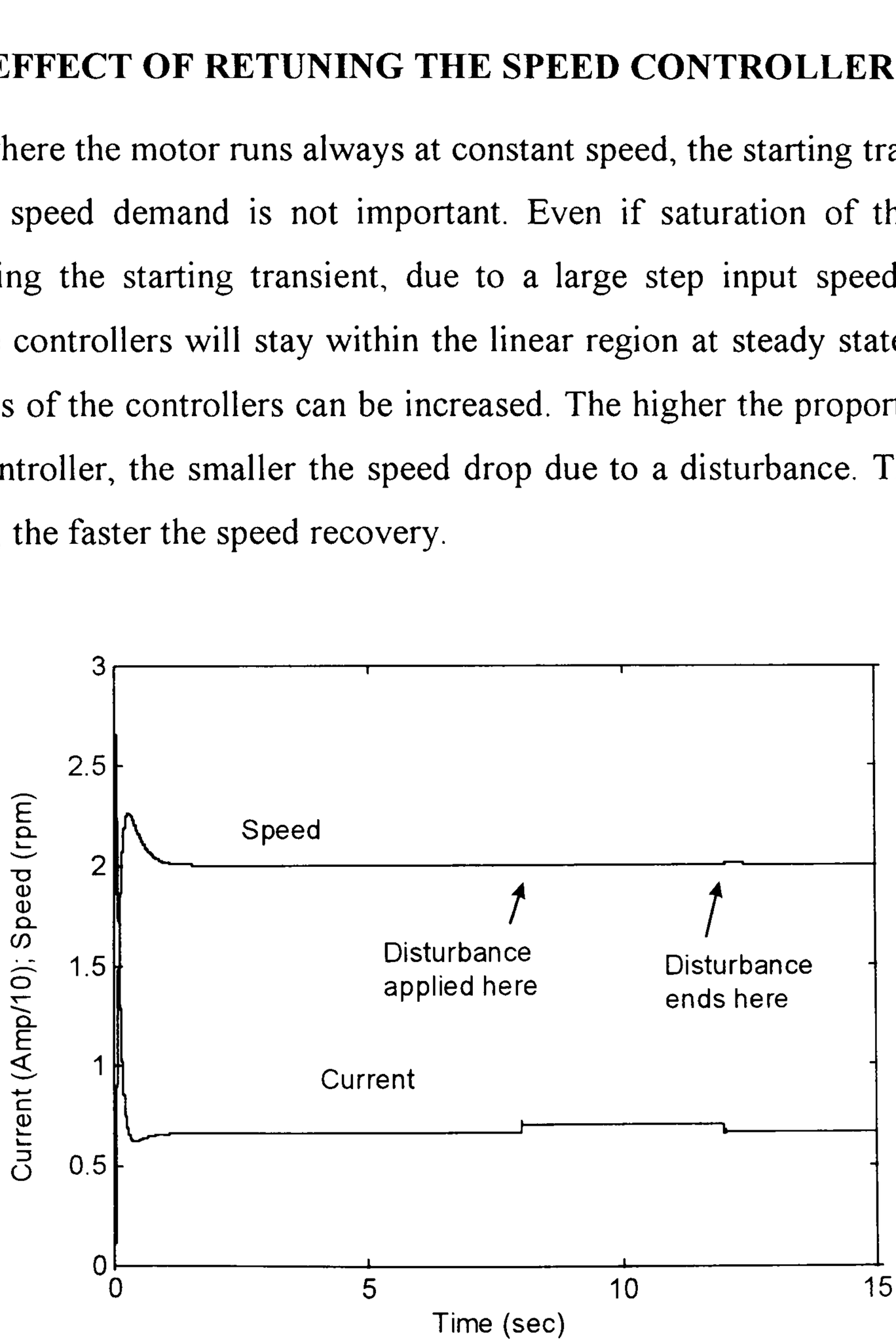



Fig. D.10 – Speed and current response of the dc motor drive with load estimator and torque feedforward control, in the presence of load disturbance, with $Ksf$ = 150 V/rad/s and $1/Tsf$ = 500 V/rad.

In order to improve the speed holding capability, the speed controller proportional gain, *Ksf*, was increases by a factor of approximately 5, becoming **150** V/rad/s. The integral (1/*Tsf*) gain was also increased but, aiming avoid speed oscillation after the disturbance, the rate was different and was made equal to **500** V/rad. The speed response becomes as shown in Fig. D.10. Fig. D.11 presents the speed response, in enlarged scale, highlighting the speed variation due to the load disturbance.

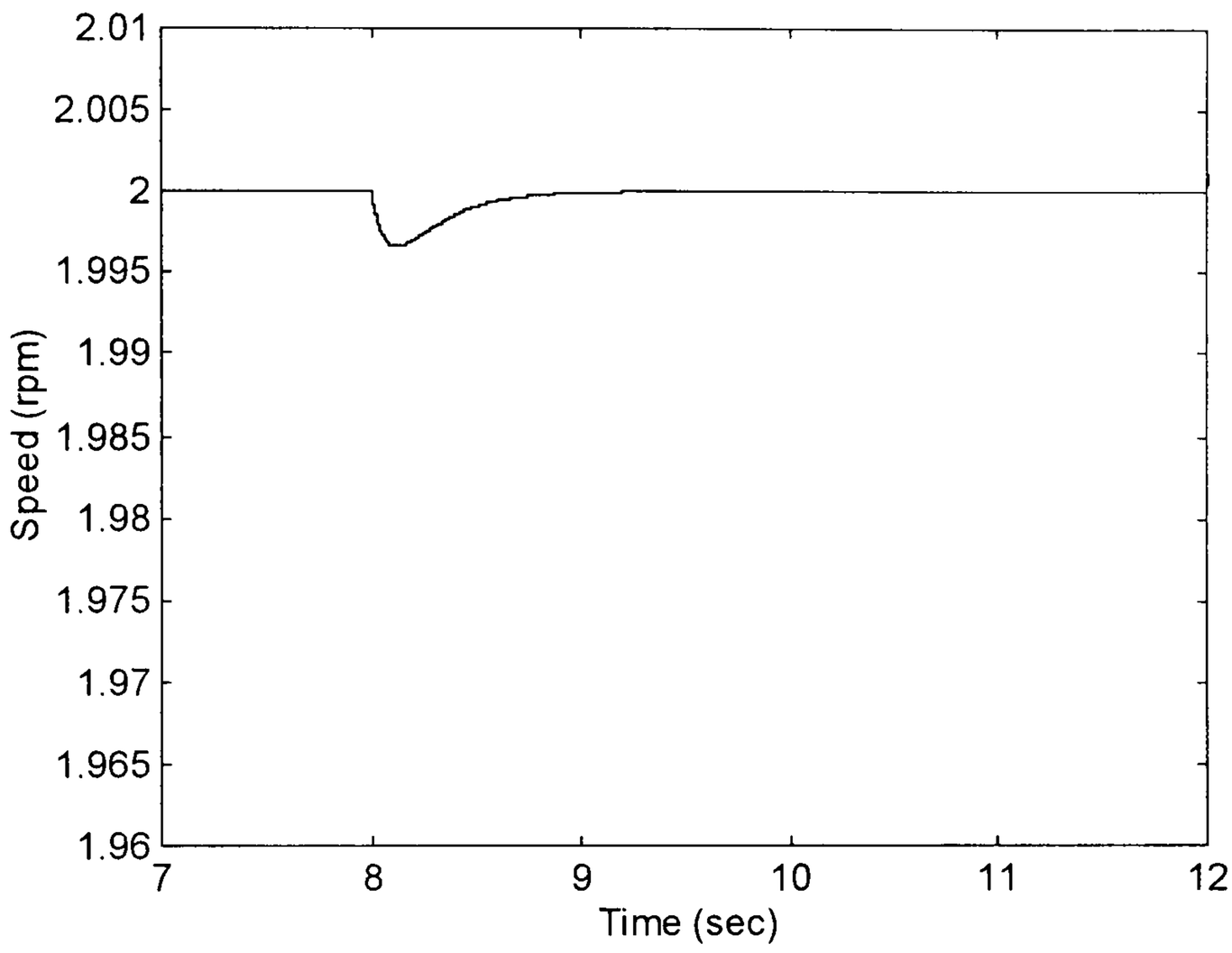



Fig. D.11 – Speed response in enlarged scale showing the variation due to the load disturbance.

Fig. D.12 – D.15 show the speed and current controller output signal. Regarding to the speed controller, it is saturated at 15 volts for fractions of a second, during the acceleration time. However, as it lasts for a very short period of time, it does not cause any problem. Furthermore, if the speed demand increases in smaller steps, this saturation may not even take place. On the other hand, due to the higher gains of the controllers, the speed holding capability is better than that shown in section D.4 – Fig. D.9, with torque feedforward control. The difference is shown in Fig. D.16.

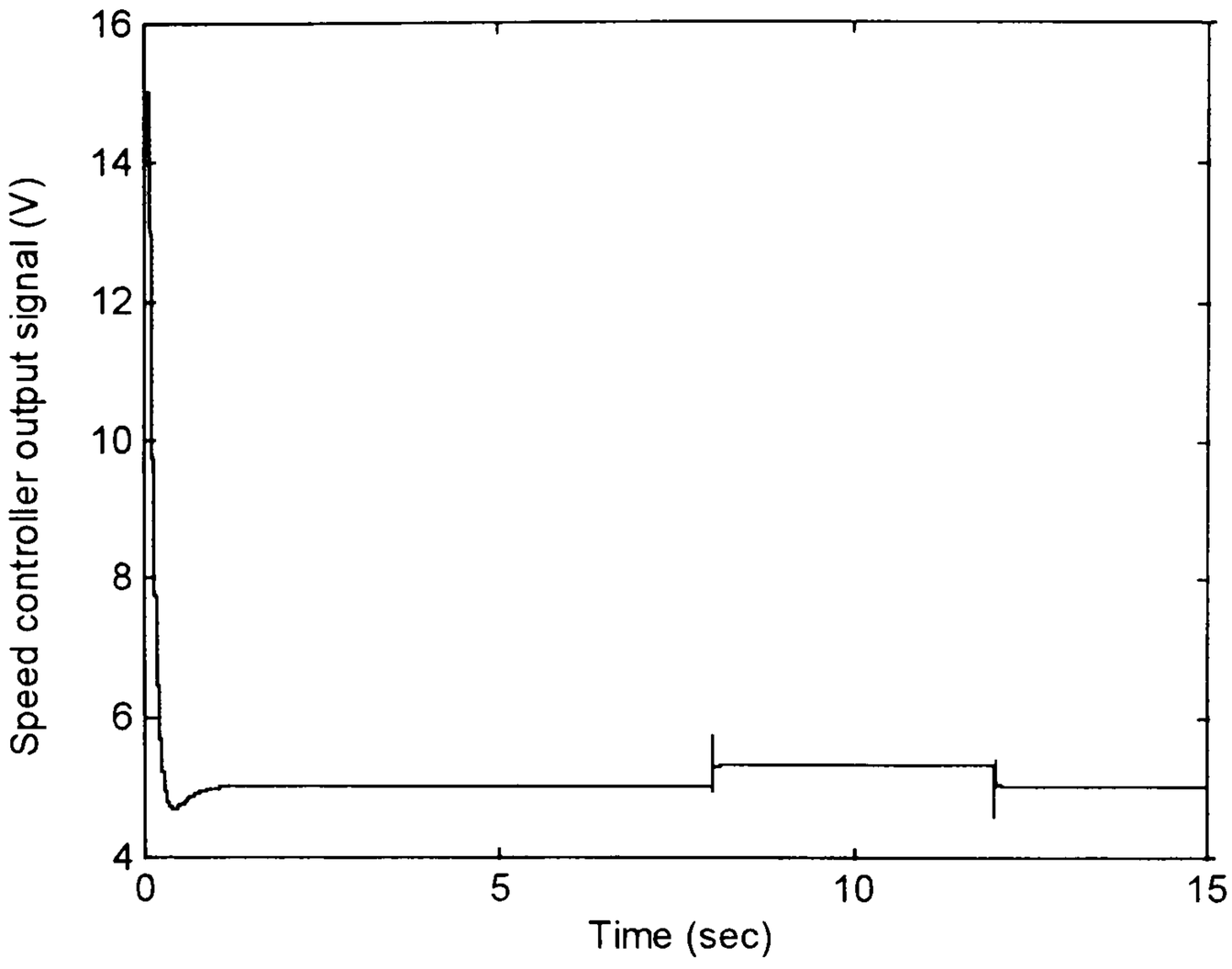



Fig. D.12 – Speed controller output signal.

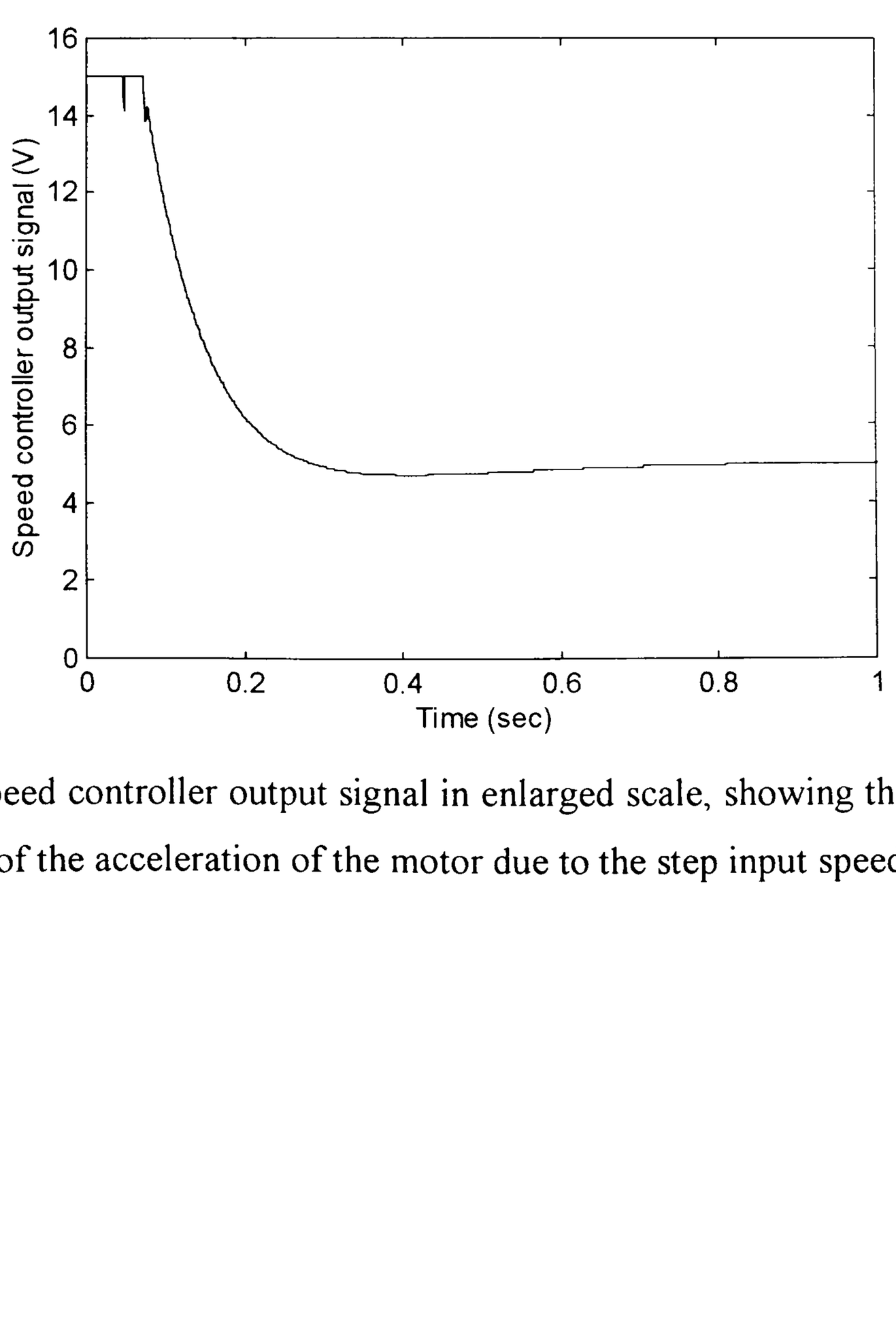



Fig. D.13 – Speed controller output signal in enlarged scale, showing the saturation at the beginning of the acceleration of the motor due to the step input speed demand.

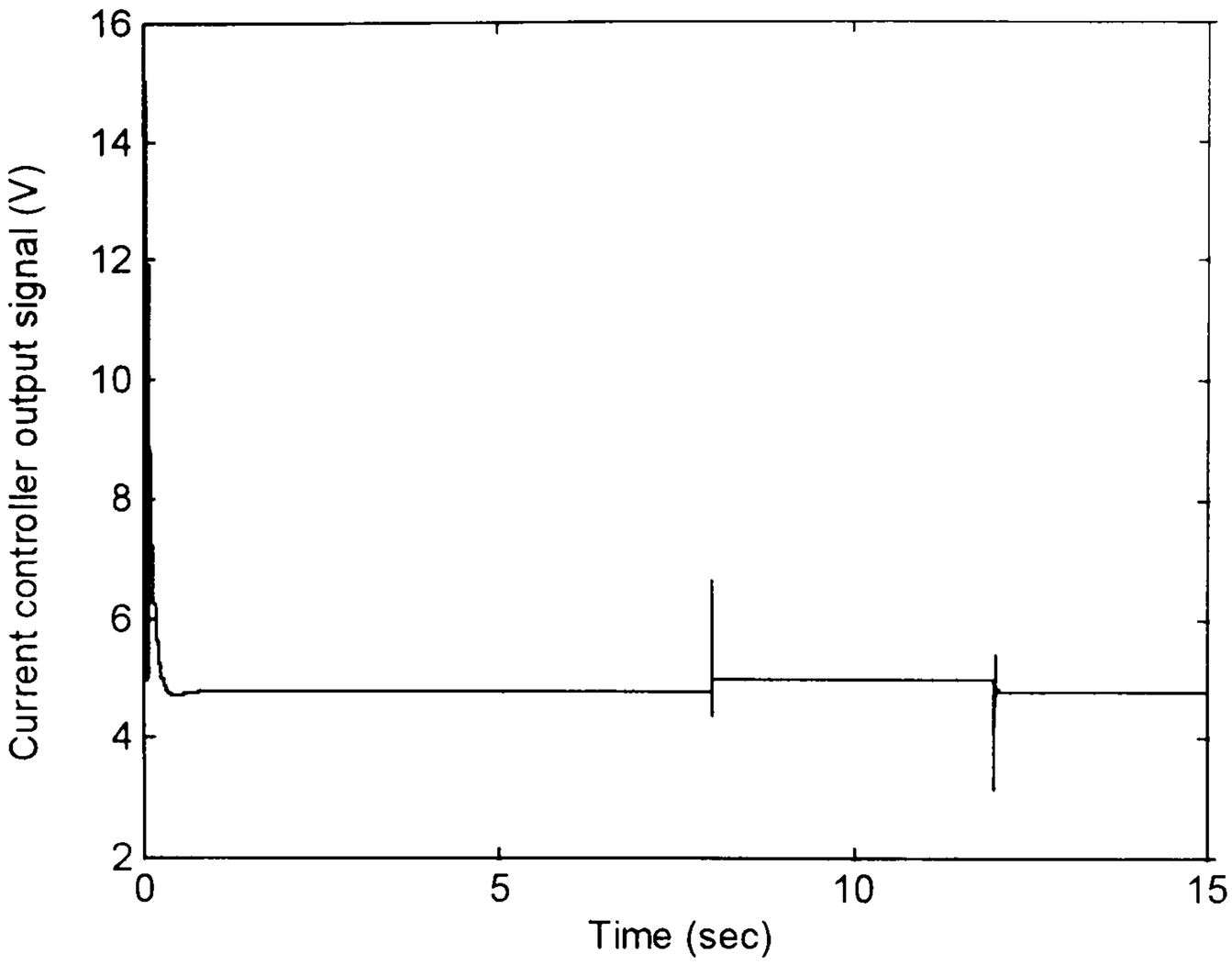



Fig. D.14 – Current controller output signal.

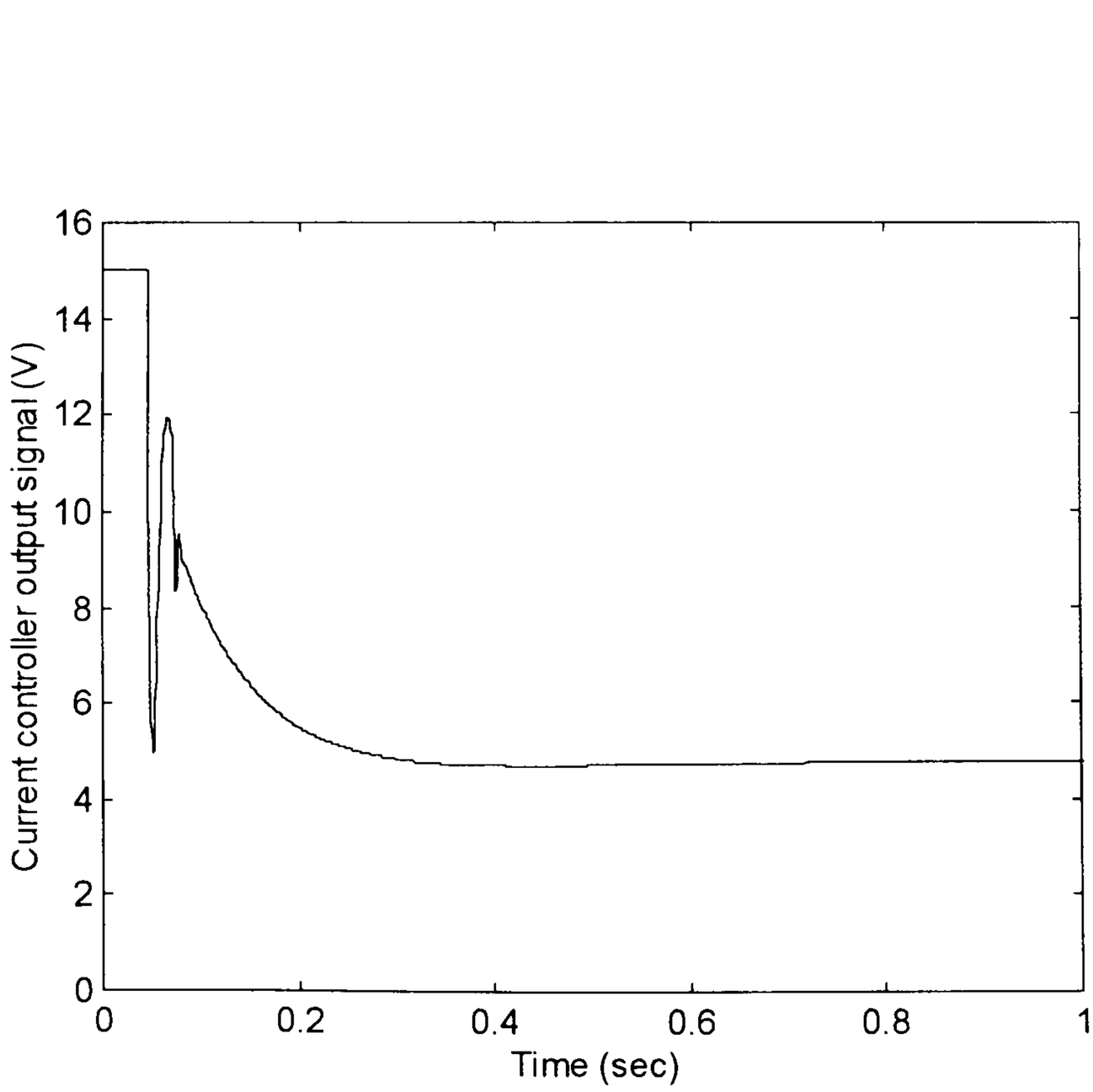
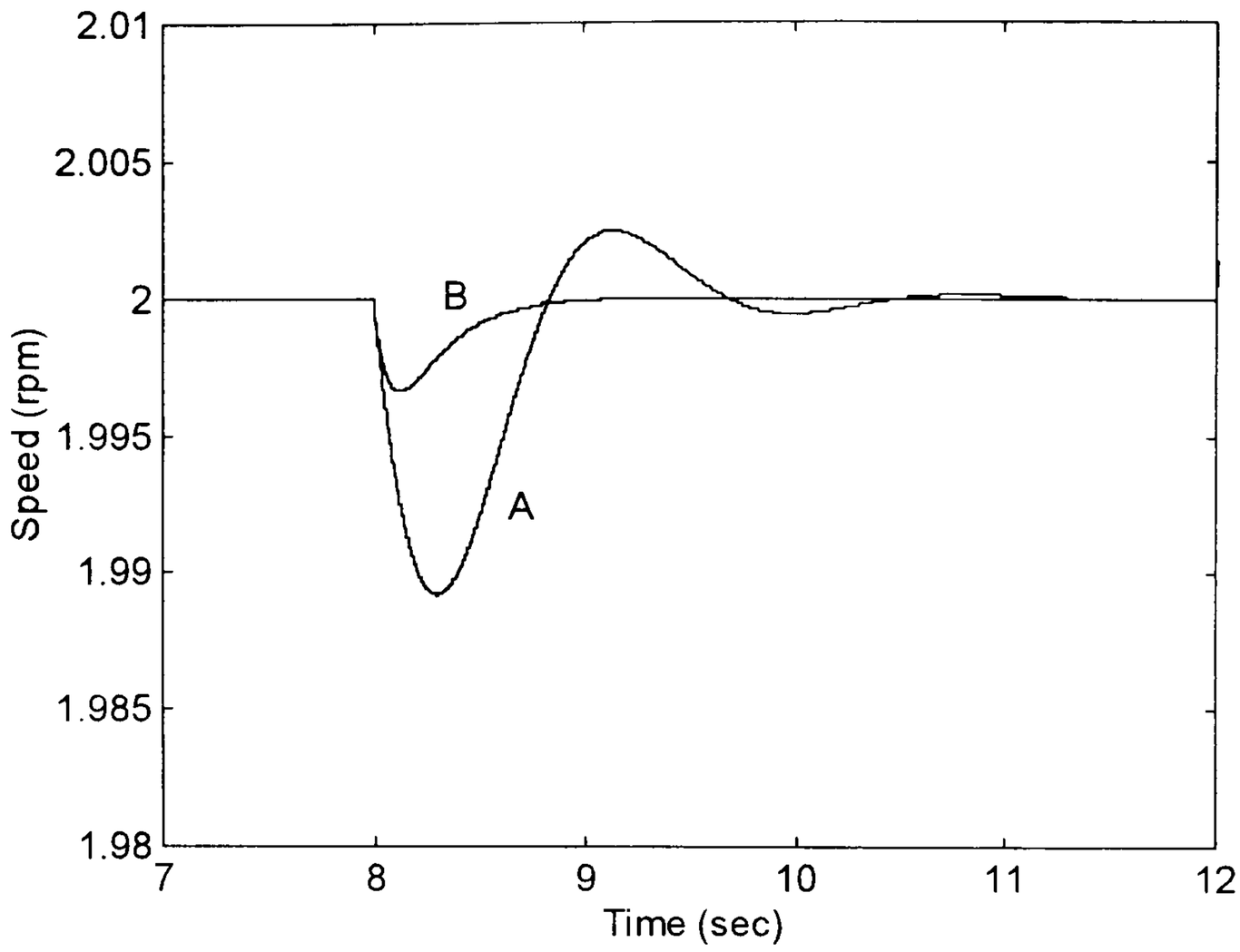



Fig. D.15 – Current controller output signal in enlarged scale, showing the saturation at the beginning of the acceleration of the motor due to the step input speed demand.




"A" - with torque feedforward control and lower gains; "B" - with torque feedforward control and higher gains.

Fig. D.16 – Comparison of the speed holding capability of the dc motor drive with load estimator and torque feedforward control, with the controller's gains as in sections D.4 – Fig. D. 9 and section D.5 – Fig. D.11.

## D.6 – THE IMPACT OF THE PRESENCE OF NOISE ON THE SPEED SIGNAL

The impact of noise on the speed signal can change the performance of the drive, mainly when the load estimator and torque feedforward control is used. The reason is that the speed differentiation is needed and when a noisy signal is differentiated, the result is another signal even noisier.

In order to diminish this problem, the time constant of the filter used, *Tc*, has to be increased. On the other hand, for a **noise free signal**, the higher this time constant, the poorer the performance (bigger speed drop due to the disturbance and longer the speed recovery time). As a consequence, the speed signal has to be as good as possible, in order to ensure optimal performance.

In order to illustrate its impact, a 3 mV- 100 Hz noise signal is added to the speed signal which comes from the tacho generator. This magnitude corresponds to 0.2% of the signal which comes out of the tacho generator due to a 2 rpm (0.209 rad/s) motor speed. Two values of the time constant, $Tc$ = 1 ms, as it originally was, and $Tc$ = 50 ms are used on the filter and the speed responses shown in Fig. D.16 and D.17. The gains of the current and speed controller are kept as used in section 6, however, the simulation time has been reduce to 10 sec and the disturbance taking place at *time* = 3.5 sec, lasting for 4 sec. Regarded to the response with $Tc$ = 1 ms, the filter has no effect on the noise reduction as the cut off frequency is above the frequency of the noise. However, for $Tc$ = 50 ms, the level of noise fed forward is reduced.

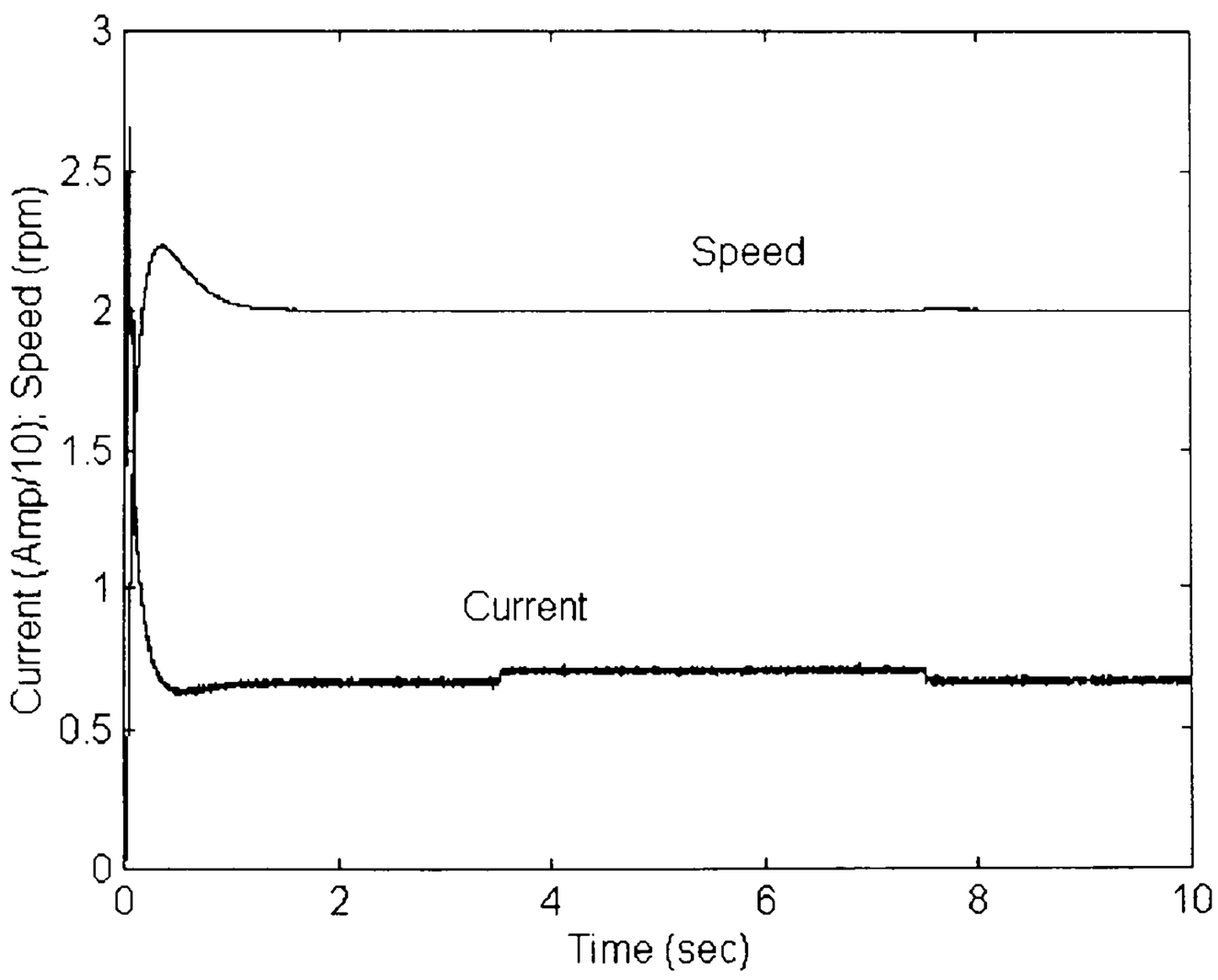



Fig. D.16 – Fig. 14 - Speed response of the dc motor drive with torque feedforward control in the presence of 3 mV noise added to the speed signal from the tacho generator, with $Tc$ = 50 ms.

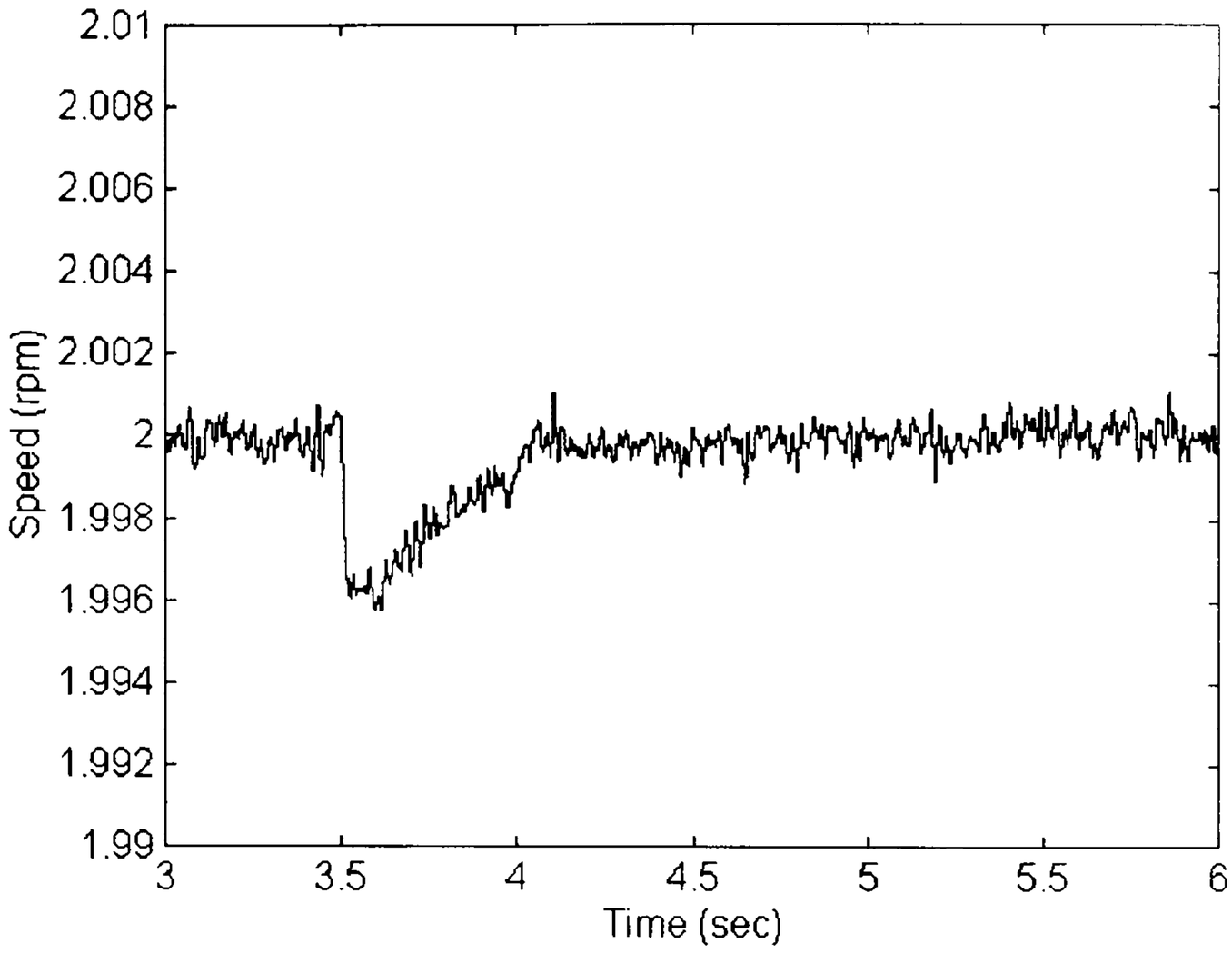



Fig. D.17 – Speed in enlarged scale showing the impact of the load disturbance.

## D.7 - CONCLUSION

The dc motor drive of Line 51 has been studied and its speed holding capability against load disturbance investigated. In order to make improvement on the performance of the motor drive against variations of the load, a load disturbance observer and torque feedforward control has been used. However, by retuning the speed controller with higher proportional and integral gains could improve even more the performance of the dc motor drive in the presence of load disturbance: the speed drop due to load disturbance as well as its recovery time can be significantly reduced. This mean that the speed holding capability against load variation can be improved, compared to the way it is today in the production plant.

## D.8 PARAMETERS OF THE DC MOTOR DRIVE AS DETERMINED BY THE DuPONT ENGINEER.

### D.8.1 – Components value

| | | |
|---|---|---|
| R1 = 4700 Ohms | R38 = 1000 Ohms | XP1 = 0.1 |
| R2 = 100 Ohms | R42 = 1800 Ohms | |
| R3 = 12 KOhms | R43 = 64.9 KOhms | P3 = 20 KOhms |
| R4 = 0 Ohms | R44 = 68.1 KOhms | XP3 = 0.3 |
| R8 = 22 KOhms | | |
| R10 = 1800 Ohms | | P4 = 20 KOhms |
| R13 = 15 KOhms | R48 = 470 KOhms | XP4 = 0.5 |
| R14 = 22 KOhms | R52 = 10 KOhms | |
| R17 = 15 KOhms | | |
| R21 = 1800 Ohms | WN1 = 10 KOhms | C1 = 6.8 nF |
| R22 = 4700 Ohms | WN2 = 10 KOhms | C2 = 6.8 nF |
| R23 = 10 KOhms | WN3 = 22 KOhms | C5 = 47 nF |
| R25 = 330 Ohms | | C6 = 0.1 μF |
| R26 = 100 KOhms | | C7 = 1.0 μF |
| R27 = 330 Ohms | | C8 = 0.6 μF |
| R32 = 330 KOhms | | C11 = 0.15 μF |
| R33 = 100 KOhms | P1 = 20 KOhms | C13 = 6.8 nf |

### D.8.2 – Mechanical Parameters

Inertia – *Jtot* = 193.5 Kg.m2

Friction – *Dtot* = 113.5 Nm/rad/s

### D.8.3 – Parameters used along the block diagram of the dc motor drive system

The gains used along the block diagram of the dc motor drive system were determined as follow:

### D.8.3.1 - Tacho generator

Tacho constant:

$Kt = 7.2$ V/rad/s

Tacho Filter:

$$Ktf = \frac{R3 + XP1.P1}{P1 + R3 + R2 + R1} = \frac{12e^3 + 0.1x20e^3}{20e^3 + 12e^3 + 100 + 4700} = 0.38 \text{ V/V}$$

### D.8.3.2 - Speed demand filter

$$Ksd = \frac{WN3.R13}{(WN1 + WN2)(R4 + R8)} = \frac{22e^3.15e^3}{(10e^3 + 10e^3)(0 + 22e^3)} = 0.75 \text{ V/V}$$

### D.8.3.3 - Speed loop

Speed controller proportional gain:

$$Ksf = \frac{XP4.P4 - (XP4)^2 P4 + R32}{R14(1 - XP4)} = \frac{0{,}5.20e^3 - (0{,}5)^2 20e^3 + 330e^3}{22e^3(1 - 0{,}5)} = 30.45 \text{ V/rad/s}$$

Speed controller integral gain:

$$Tsf = (\text{'8}.R14(1 - XP4) = 0{,}6e^{-6}.22e^3(1 - 0{,}5) = 0.0066 \text{ V/rad}$$

Speed setpoint gain:

$$Ksl = \frac{R14}{R17 + R22} = \frac{22e^3}{15e^3 + 4700} = 1.116 \text{ V/V}$$

Constant gain:

$$Kcsl = \frac{R21 + XP3.P3}{R10 + R21 + P3} = \frac{1800 + 0{,}3.20e^3}{1800 + 1800 + 20e^3} = 0.33 \quad \text{V/V}$$

### D.8.3.4 - Current loop

Current controller proportional gain:

$$Kcf = \frac{R48}{R44} = \frac{470e^3}{68{,}1e^3} = 6.9 \ \text{V/A}$$

Current controller integral gain:

$$Tcf = C13.R44 = 6{,}8e^{-9}\,68e^3 = 4.6e^{-4} \quad \text{V.s/A}$$

Current setpoint gain:

$$Kcl = \frac{R44}{R38 + R42 + R43} = \frac{68{,}1e^3}{1000 + 1800 + 64{,}9e^3} = 1.005 \ \text{V/V}$$

Current feedback gain:

$Kc = 0.25$ V/A

Drive gain:

$Ki = 4$ V/V

### D.8.3.5 - Motor constants

Winding resistance:

$Rm = 2.06$ Ohms

Winding inductance:

$Lm = 0.035$ Henry

Torque constant:

$Ka = 26.032$ Nm/A

Back EMF:

$Kb = 26.032$ V/rad/s

$$Ke = \frac{1}{Rm} = \frac{1}{2.06} = 0.48 \text{ Ohms}^{-1}$$

$$Te = \frac{Lm}{Dtot} = \frac{0.0035}{2.06} = 0{,}017 \quad \text{sec}$$

### D.8.3.6 - Casting drum constants

$$Km = \frac{1}{Dtot} = \frac{1}{113.5} = 8{,}81e^{-3} \ \text{(Nm.s/rad)}^{-1}$$

$$Tm = \frac{Jtot}{Dtot} = \frac{193{,}52}{113{,}5} = 1{,}705 \ \text{sec}$$

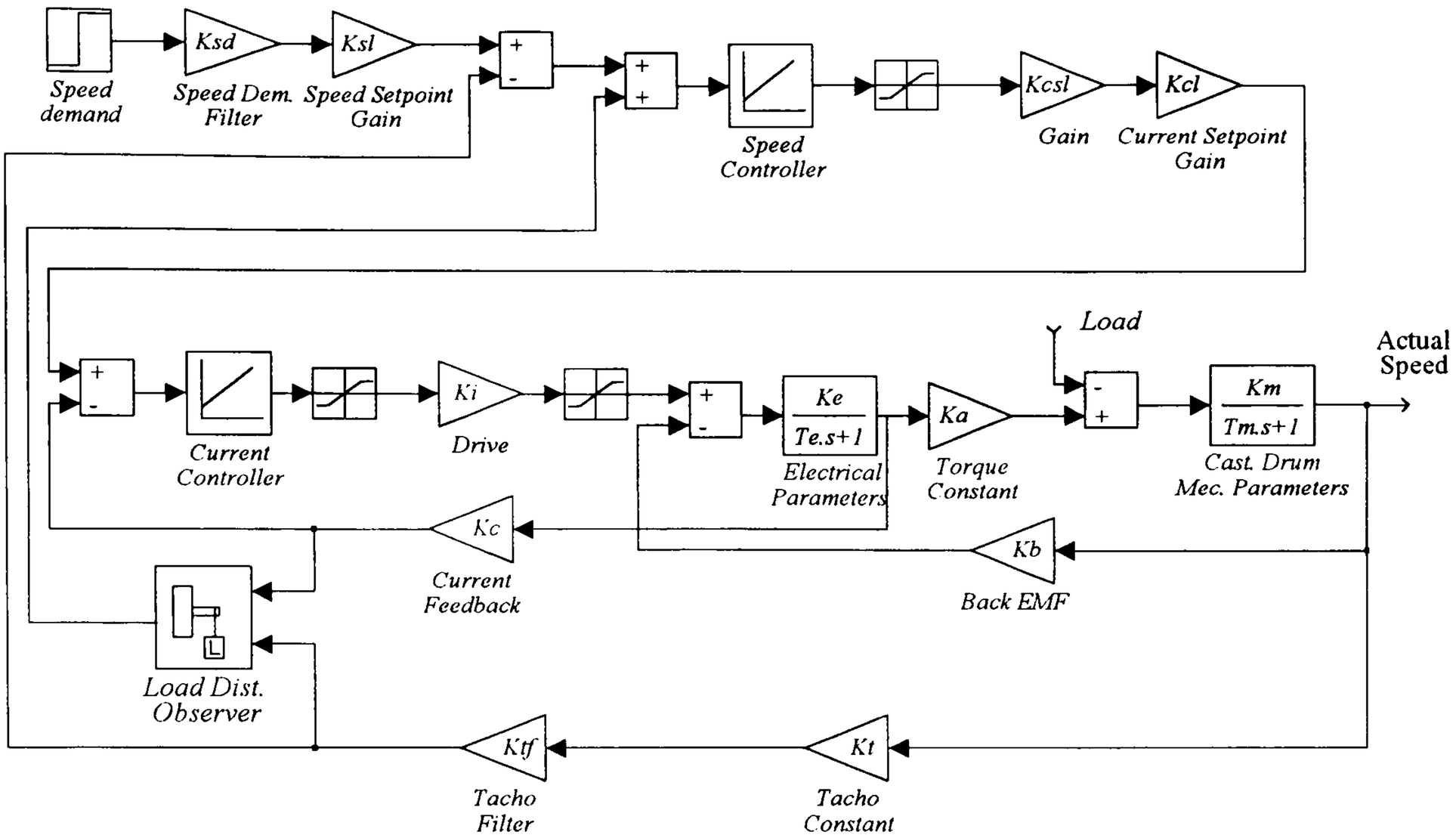



Fig. Aii – SIMULINK block diagram of the casting drum dc motor drive of Line 51 with load estimator.