FACULTY OF ENGINEERING AND ARCHITECTURE

Consensus-based position estimation in modular motor drives

Thymen Vandenabeele
Student number: 01606059

Supervisors: Prof. dr. ir. Hendrik Vansompel, Prof. dr. ir. Peter Sergeant
Counsellor: Lynn Verkroost

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Electromechanical Engineering

Academic year 2020-2021

GHENT UNIVERSITY

FACULTY OF ENGINEERING AND ARCHITECTURE

Consensus-based position estimation in modular motor drives

Thymen Vandenabeele
Student number: 01606059

Supervisors: Prof. dr. ir. Hendrik Vansompel, Prof. dr. ir. Peter Sergeant
Counsellor: Lynn Verkroost

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Electromechanical Engineering

Academic year 2020-2021

GHENT UNIVERSITY

Permission of Usage

The author gives permission to make this master dissertation available for consultation and to copy parts of this master dissertation for personal use. In all cases of other use, the copyright terms have to be respected, in particular with regard to the obligation to state explicitly the source when quoting results from this master dissertation

Thymen Vandenabeele, June 2021

Preface

This dissertation concludes my five years engineering studies at the University of Ghent. It was an amazing part of my life with many ups and even some downs. This year was definitely the most bizarre of them all: my final year as a student and a COVID-19 pandemic. It must be said: those two are not an optimal combination. Living as a young person during this pandemic is sometimes mentally though. However, it is a moral duty to try to turn something negative into something positive. Therefore, I used a lot of the extra spare time to work hard on this dissertation, let this text be the result of that.

In the first semester of this academic year I mainly conducted a research into different average consensus algorithms. Meanwhile, understanding and tuning the vector-tracking observer had to be done in the first semester. I had quite some fun because I am genuinely interested in many different subjects and I learned a lot of new things in that semester.
In the second semester, the practical side of this research was conducted: the algorithms had to be tested on a real measurement setup. Days full of soldering small connections and trying to understand the working principles of the MicroLabBox were no exception. Eventually, encouraging results were obtained and that is one of the best feelings a researcher can have.

I sincerely hope that this dissertation is an added value for the development of modular motor drives. Personally, I am more than satisfied with the result. This result was however, never obtained without certain people. I would like to thank my roommates in Ghent for the pleasant environment to work on my dissertation. I would like to thank my parents and my brother: they kept supporting me, even in my moody days (mostly during exams). My parents gave me every opportunity I wanted and the freedom to choose my own path, thank you. Furthermore I want to thank Vincent from the laboratory for being so friendly and helpful whenever I had technical issues with the setup. And I am thankful to Prof. dr. ir. Peter Sergeant for his structured advice. But most of all I would like to thank Prof. dr. ir. Hendrik Vansompel and Lynn Verkroost, they were the perfect mentors: always available, genuinely interested in my work and most of all extremely helpful with all my questions! I could not wish for anything more.

Thymen Vandenabeele, June 2021

Consensus-based position estimation in modular motor drives

by Thymen Vandenabeele

Master's dissertation submitted in order to obtain the academic degree of Master of Science in Electromechanical Engineering

Academic year 2020-2021

Supervisors: Prof. dr. ir. Hendrik Vansompel, Prof. dr. Ir. Peter Sergeant
Counsellors: Ir. Lynn Verkroost

Faculty of Engineering and Architecture
Ghent University

Abstract: The goal of this master's dissertation is to make a decentralised position measurement of the rotor in a modular motor drive (MMD). A MMD has a stator that is split up into modules. Using a distributed control strategy, a single centralised controller for the whole drive can be replaced by segmented control units. This makes the control of the drive fault tolerant. Up until now, the position measurement of the rotor is still conducted by one single encoder. In this dissertation, a decentralised strategy to measure the position of the rotor using cheap and robust Hall sensors is proposed. A combination of three Hall sensors is called an agent. A vector-tracking observer is used to obtain an initial position measurement for each agent. This position measurement is further improved by using information from other agents in a consensus-based averaging algorithm. To make this position measurement concept fault tolerant, a fault detection algorithm was introduced. The consensus-based averaging algorithm and the fault detection algorithm were tested on a real test setup. This dissertation ends with a discussion about sensor misalignment.

Keywords: Consensus-based algorithm, distributed control, modular motor drives, vector-tracking observer, sensor misalignment

Consensus-based position estimation in modular motor drives

Thymen Vandenabeele
Supervisors: prof. dr. ir. Peter Sergeant, prof. dr. ir. Hendrik Vansompel and ir. Lynn Verkroost

***Abstract*—This article proposes a strategy for decentralised position estimation in modular motor drives. This is the last step in the decentralisation process of the control of modular drives. In this new strategy, every module of the modular drive is equipped with a Hall sensor. The output of the sensors are used in multiple vector-tracking observer schemes to track the position of the rotor. To improve the estimates of the rotor position by the vector-tracking observers, a consensus-based averaging algorithm is introduced. This distributed algorithm estimates the average of the outputs of the vector-tracking observers based on limited information. A fault detection algorithm is proposed to exclude faulty signals from the consensus-based averaging algorithm. Furthermore, the influence of misalignment of the Hall sensors is discussed.**

***Keywords*— Consensus-based algorithm, distributed control, modular motor drives, vector-tracking observer, sensor misalignment**

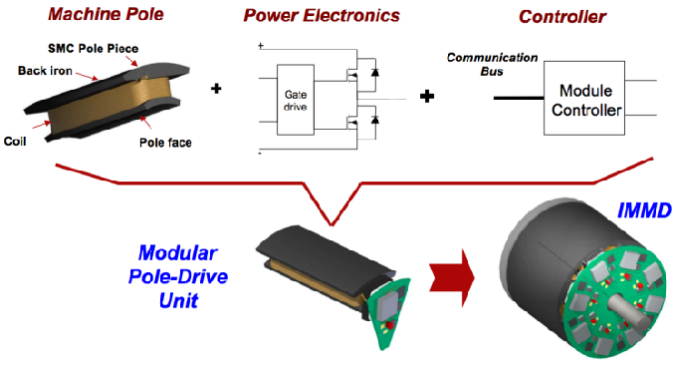


Fig. 1: Schematic drawing of the IMMD concept [2]

I. Introduction

The technology of power electronics is constantly evolving, which leads to power electronic components that are more compact and more resistant to heat. This creates an opportunity to integrate the power electronic components of a motor drive close to the stator of an electrical machine. In an integrated motor drive (IMD), the power electronics and the motor are not physically split anymore [1]. The design of an IMD is more compact than a separate control and motor unit. This leads to shorter cables, which reduces the risk of emitted electromagnetic interference (EMI) and avoids overvoltage due to transmission line effects. IMDs are also cheaper: there is no need for an output filter because of the shorter cables. Also, the integration of the power electronics into the housing of the motor leads to an increased automation of the manufacturing process. Next to that, a singular cooling can be used for both the power electronics and the motor.

An important step in the evolution towards the complete absorption of the motor drive into the motor enclosure, is the discretization of both the motor and the power electronics into modules. This gives rise to the concept of an Integrated Modular Motor Drive (IMMD). An example of such an IMMD is presented in Fig. 1. The modular approach in an IMMD allows individual control of every stator winding, this leads to fault-tolerant techniques and a decentralised control of the machine. In most cases in literature, every module is controlled by a centralised controller [3]. This control strategy of an IMMD is fault-tolerant to a certain degree. The centralised controller is able to recognise faulted modules and can prevent them from disturbing the operation of the motor [4]. However, when the centralised controller breaks down, no control of the motor is possible anymore. This makes the control technique with one centralised controller not entirely fault tolerant. A new generation of IMMDs demonstrates that it is possible to segment this centralised controller as well. When a module of such an IMMD fails, the other modules continue working and the machine is still controllable.

When the controller is segmented, distributed control is needed. In [2], a simple decentralised control strategy is proposed where the controllers of the different modules operate as synchronized peers running identical programs while sharing sensor information as needed. This sensor information is the current feedback of every phase. All the controllers share a single position measurement from an encoder.

The centralised controller was segmented to enhance the fault tolerancy, however, all the modules are up to now still dependent on one single encoder or resolver for the position measurement of the rotor. These encoders or resolvers are delicate and expensive components. Therefore it is not economically feasible to install multiple encoders in a modular drive. This means that the position measurement of the rotor in modular drives is a single point of failure. In this article a distributed algorithm that results in a decentralised, robust and cheaper estimation of the rotor position is introduced. This distributed position estimation is the last step towards an entirely decentralised control of an IMMD.

In this article, a modular motor drive (MMD) instead of an IMMD will be used because the concept of an IMMD still faces a lot of challenges (e.g. thermal management) before it can be commercialized. In a MMD, the controller of the machine is still segmented but not integrated into the stator. Nevertheless, to integrate the controller into the stator stays

the ultimate objective for future research.
For the proposed distributed measurement technique, each module of the MMD will get its own cheap, low resolution Hall sensor. At first, the outputs of three of these Hall sensors will be combined to obtain an initial position estimate per group of three modules, based on the technique proposed in [5]. In a second step, these initial estimates will be improved by means of communication between the modules.
In Section II, it is explained how the output of three Hall sensors can be combined into one rotor position measurement. A vector-tracking observer (VTO) will be used in combination with two other features. The combination of three modules that make up one position measurement is called an agent in the remainder of this article. Section III uses the output of all the agents in the modular drive in a consensus-based averaging algorithm to further improve the estimates of the agents. In Section IV a fault detection algorithm is proposed that enables the agents in the modular drive to recognise faulty signals from its neighbouring agents or from itself. Measurements of the performance of the consensus-based averaging algorithm and the fault detection algorithm are discussed in Section V. Due to sensor misalignment, there is a difference in accuracy, when estimating the rotor position, between simulations and the measurements from Section V. This topic is discussed in Section VI. The article ends with a conclusion in Section VII

II. Low-Resolution Position Sensor

As mentioned in the introduction, an initial rotor position measurement is conducted by a combination of three modules. This combination is called an agent. A modular drive consists of multiple agents. In this section, the position estimation by one agent is described. The output of three Hall sensors will be used in combination with a vector-tracking observer (VTO).

A. Low Resolution Position Vector

Using a low-resolution position estimation, the interval of 360 electrical degrees or two pole pitches τ_p is subdivided in a limited number of intervals. In this article, one agent is made up by three modules. Because every module is equipped with a Hall sensor, one estimation of the position of the rotor relies on three Hall sensors. This gives six sectors over 360 electrical degrees. During rotation, the rotor position vector jumps to subsequent sectors. Therefore it can only take six discrete values. This vector is a low resolution position vector and denoted by $\vec{H}_{\alpha\beta}$. Another important vector is $\vec{H}$. This vector corresponds with the real rotor position, θ. It is a unit vector rotating continuously and given by

$$\vec{H} = e^{j\theta}, \tag{1}$$

with θ continuously varying and attached to the rotor. It is shown in [5] that $e^{j\theta}$ is the fundamental component of $\vec{H}_{\alpha\beta}$. This fundamental is the same rotating vector as $\vec{H}$=$e^{j\theta}$. And thus it can be stated that $\vec{H}_{\alpha\beta,1}$ is equal to $\vec{H}$.

B. Vector-tracking Observer

To control a permanent magnet synchronous machine (PMSM), the vector $\vec{H}_{\alpha\beta}$ is not accurate enough. The large difference with the real angle leads to a torque ripple. To reduce the deviation from the actual rotor position, one could use a vector-tracking observer (VTO), whose structure is shown on Figure 2. The basic VTO topology consists of a PID controller and a feedback loop. Initially the speed of the rotor is estimated ($\hat{\omega}$) and this result is then integrated to obtain the estimated rotor position ($\hat{\theta}$). The VTO has to track θ as

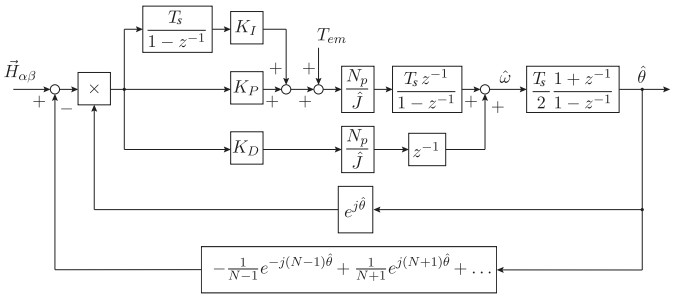


Fig. 2: Topology of the VTO, with decoupling of additional harmonics added.

accurate as possible. Therefore the bandwidth has to be chosen as such that at maximum speed, the VTO is still able to track θ. This is not the only limitation to the bandwidth of the VTO. At low speed and with high bandwidth, $\hat{\theta}$ will follow the discrete structure of $\vec{H}_{\alpha\beta}$. This will lead to differences with the real position of the rotor that are unacceptable for good control of the PMSM. So, in order to keep the fault relatively small at low speed, the bandwidth of the VTO cannot be too high. Two additional features are added to the basic VTO structure.

1) Variable Gains: To address the trade-off for the bandwidth from above, a speed dependent bandwidth imposes itself naturally. The bandwidth is made speed dependent by linearly varying PID parameters K_i, K_p and K_d with the speed. Based on the estimated speed $\hat{\omega}$, a value for every gain is calculated. To prevent the VTO from working in open loop, a minimum value for the gains is imposed at very low speeds.

2) Decoupling Additional Harmonics: In simulations in [5], it was seen that the deviation $\Delta\theta$, with $\Delta\theta = \hat{\theta} - \theta$, is at its largest when $\vec{H}_{\alpha\beta}$ jumps to next sector. This is caused by the reaction of the VTO to the discrete structure of $\vec{H}_{\alpha\beta}$. When the additional harmonics are decoupled from $\vec{H}_{\alpha\beta}$, a rotating vector with strongly reduced harmonic content remains. Now the bandwidth can be increased without resulting in a VTO that follows strongly the discrete structure of $\vec{H}_{\alpha\beta}$.

III. Consensus-Based Averaging Algorithm

In the previous section, a vector-tracking observer with variable gains and decoupling of harmonics was proposed to obtain $\hat{\theta}$, based on the output of three low resolution Hall sensors. This combination of three Hall sensors is called an agent. A modular drive consists of multiple agents. Each agent has its own estimate of θ. The notation for such an estimate is $\hat{\theta}_i$ with $i \in$ 1,...,n. n being the number of agents in the modular drive. The algorithm, that is proposed in this section, improves the position estimate for every agent. This is done by estimating the average of the estimates of the five agents:

$\bar{\hat{\theta}}$. The hypothesis behind this is that all the agents estimate θ, an average of these estimates should improve the position measurement. The estimation of the average is not executed by a single calculation unit. This would make the position measurement of the rotor centralised, while it is the goal of this article to decentralise the position measurement of the rotor. Therefore, every agent will conduct its own calculations to get an approximation of the average. A major difficulty to do this, is that every agent only communicates with its two closest neighbours. In Fig. 3, the communication possibilities between the agents are shown for five agents. In the proposed consensus

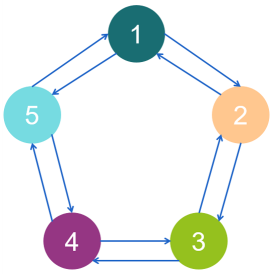


Fig. 3: Visual representation of the communication between five agents. Every agent only communicates with its two closest neighbours.

algorithm, the input function is the sine and the cosine of $\hat{\theta}_i$. This is done to use continuous functions in the algorithm. The average sine and the average cosine are then used together to obtain an estimation for $\bar{\hat{\theta}}$. The proposed consensus algorithm is expressed in equation (2) for agent i. Analogue equations are conducted for the cosine of the estimated rotor position.

$$\overline{(\sin\hat{\theta})}_i(k) = \frac{1}{3}\Big(\sin\hat{\theta}_h(k) + \sin\hat{\theta}_i(k) + \sin\hat{\theta}_j(k)\Big) \tag{2a}$$

$$\begin{aligned}\overline{(\sin\hat{\theta})}_i(k+2) =& \overline{(\sin\hat{\theta})}_i(k)\cos(2\hat{\omega}T_s) \\ &+ \overline{(\cos\hat{\theta})}_i(k)\sin(2\hat{\omega}T_s)\end{aligned} \tag{2b}$$

$$\begin{aligned}\overline{(\sin\hat{\theta})}_{hij}(k+2) =& \frac{1}{3}\Big(\overline{(\sin\hat{\theta})}_h(k+2) + \overline{(\sin\hat{\theta})}_i(k+2) \\ &+ \overline{(\sin\hat{\theta})}_j(k+2)\Big)\end{aligned} \tag{2c}$$

$\sin\hat{\theta}_i(k)$ is the sine of the estimated rotor position $\hat{\theta}$, estimated by agent i. Agent i calculates the average of three estimates it has at its disposal: its own estimate, and the estimates from its two closest neighbours h and j. This is done in equation (2a). To be able to include information from every agent in the network, two communication steps are needed. Therefore, a prediction step of two timesteps imposes itself. Below, the reason for using two communication steps is explained. In the prediction step, a prediction of the position of the rotor for two timesteps into the future is made, based on the estimated speed $\hat{\omega}$ of the rotor. This speed $\hat{\omega}$ is determined by the VTO before it is integrated to obtain the rotor position. The formula for the prediction step is in equation (2b). In this equation, $\overline{(\cos\hat{\theta})}_i(k)$ from the parallel algorithm for the cosine is needed. $\overline{(\sin\hat{\theta})}_i(k+2)$ is the estimation of the average sine of agent i's and both its neighbours' estimates $\hat{\theta}$ two steps further in time. Then, agent i computes the average of $\overline{(\sin\hat{\theta})}_i(k+2)$, $\overline{(\sin\hat{\theta})}_h(k+2)$, $\overline{(\sin\hat{\theta})}_j(k+2)$. As can be seen in equation (2c). When the average of these values is calculated, not only the neighbours of agent i have an influence on the average. Also the neighbours of the neighbours from agent i have an influence on the average calculated by agent i. This leads to an estimate of the average based on five agents, calculated by agent i. An overview of what is communicated in this algorithm for agent i can be seen in Fig. 4. In this

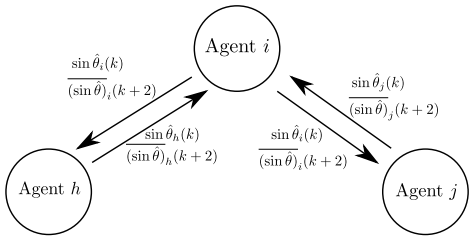


Fig. 4: Scheme to illustrate the communication between the agents.

dissertation, the modular drive consists of only five agents. When the modular drive would consist of more agents, not all agents are included in the average calculated by agent i. It could be possible to include more agents in one estimation. This would require more communication steps and therefore a prediction step for more than two timesteps. This would make the algorithm quite dependent on the speed estimated by the VTO: $\hat{\omega}$.

IV. Fault Detection Algorithm

In order to obtain a fault-tolerant consensus algorithm, a fault detection algorithm should be introduced. A fault detection algorithm enables the agent to conclude whether its neighbour or the agent itself is faulty. Next to that, there is the possibility for a faulted agent to use the signals of its neighbours to obtain a good estimation of the rotor position. This fault detection algorithm also uses the sine and the cosine of the rotor position. In this article, the fault detection algorithm will be elaborated for the sine. Parallel to this, the same happens for the cosine. A visual representation of every step in the fault detection algorithm is in Fig. 5. Agent i receives the sine waves from its neighbours and subtracts them from its own sine wave. The notation for this is Δ_{hi} or Δ_{ji}, depending on which neighbour is subtracted. From now on, expressions will only be shown for the interaction between agent h and agent i to keep it concise. An expression for Δ_{hi} is shown below.

$$\Delta_{hi} = \sin(\hat{\theta_i}) - \sin(\hat{\theta_h}) \tag{3}$$

The absolute value of Δ_{hi} is taken, $|\Delta_{hi}|$. To ease out instantaneous errors, a moving average of $|\Delta_{hi}|$ is calculated, $|\overline{\Delta_{hi}}|$. The number of timesteps for which the moving average has to be calculated is arbitrary. The higher the number, the

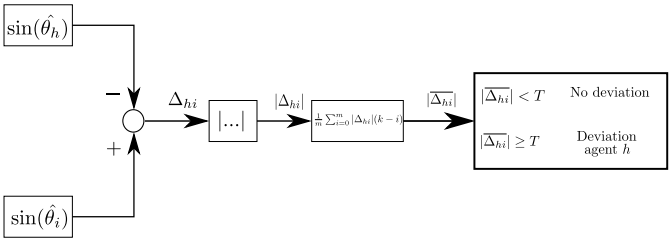


Fig. 5: Procedure executed by agent i to examine if its neighbour agent h deviates. An analog procedure is executed by agent i to examine if agent j deviates.

more robust the fault detection is during dynamical situations such as a speed reversal. However, choosing a large amount of data to average makes the detection algorithm slower to react to mistakes. $|\overline{\Delta_{hi}}|$ is now compared to a threshold value, T. If this threshold value is exceeded, then there is a so-called deviation. The decision tree of Fig. 6 shows the logic reasoning for deciding which agent is faulted. If there is at

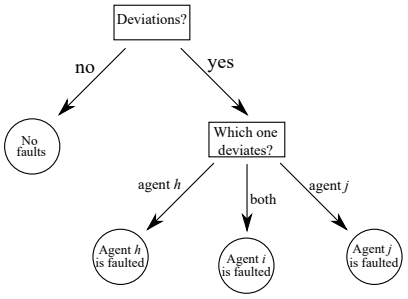
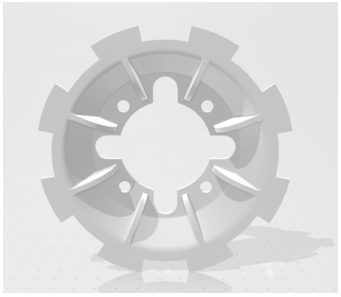


Fig. 6: Decision three of agent i for determining whether agent h, agent j or agent i itself is faulted.

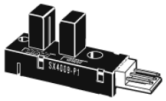
least one deviation, there is an agent that outputs a signal that is different from the others. If only $|\overline{\Delta_{hi}}| > T$, then it can be concluded that agent h is faulted. The same reasoning holds for agent j. If both $|\overline{\Delta_{hi}}|$ and $|\overline{\Delta_{ji}}|$ exceed the threshold, it can be concluded that agent i is faulted. An agent that is faulted is excluded from the consensus algorithm. Two fault situations are discussed in this article.

1) Single Sensor Failure: When a Hall sensor fails, the output of that sensor is a constant zero signal. Simulations have shown that the output of the VTO of an agent with a faulted Hall sensor is congruent to the output of a healthy agent for a large part of a period. This makes it important to have an accurate fault detection algorithm that only exclude the agent with the faulted sensor in moments where the deviation is unacceptable.

2) Agent Shutdown: An agent shutdown in this article is defined as an agent that has a constant zero signal as output from the VTO, it will only communicate a constant zero signal to its closest neighbours in both steps in the consensus algorithm defined in Section III.



(a) Customized disc; Design of the disc done by Prof. Dr. Ir. Hendrik Vansompel



(b) Optical switch © OMRON Corporation

Fig. 7: Parts of the construction to mimic the rotating magnetic field and Hall sensor interaction.

V. Measurements

Measurements to verify the performance of the consensus-based averaging algorithm and the fault detection algorithm were conducted for different situations. In every situation, $\Delta\theta$ from the output of a single agent based on its VTO solely and $\Delta\theta$ from the consensus-based averaging and fault detection algorithms are compared. In each plot, the upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm and fault detection algorithm.

A. Test Setup

The machine that is used in the test setup is an axial flux permanent magnet synchronous machine (AFPMSM). The AFPMSM is coupled with an induction machine and works as a generator. This enables straightforward speed control of the AFPMSM using the induction machine as a motor with variable speed. The concept that is elaborated in this article describes that every module of the AFPMSM should be equipped with a Hall sensor. The installation of a Hall sensor into a stator module is however technically difficult and less relevant for this research. Therefore, a construction with a rotating customized disc (Fig. 7a) and fifteen optical switches (Fig. 7b) has been developed. The optical switches are mounted on a stationary panel and are evenly distributed around the rotation axis. The customized disk has eight openings, representing a rotor with eight pole pairs. This sequence of openings and impermeable parts of the disk rotates through the legs of the optical switch. The output of the optical switches is now the same square wave as if Hall sensors would be used in combination with a rotor with eight pole pairs. To determine the accuracy of the measurement method in this dissertation, the position of the rotor has to be known. The reference position of the rotor is measured by an encoder. Fifteen sensor outputs and the encoder signal has to be processed for the measurements. All these signals arrive on a dSPACE MicroLabBox using a D-sub connector. Via the MicroLabBox, which is connected to a computer, the signals can be processed in dSPACE ControlDesk and plotted using MATLAB. The VTO, the consensus algorithm and the fault

detection algorithm are constructed in Simulink and uploaded as an sdf-file to dSPACE ControlDesk. Also the rotational speed of the induction machine is controlled using dSPACE ControlDesk. In Fig. 8 a picture of the measurement setup is presented.

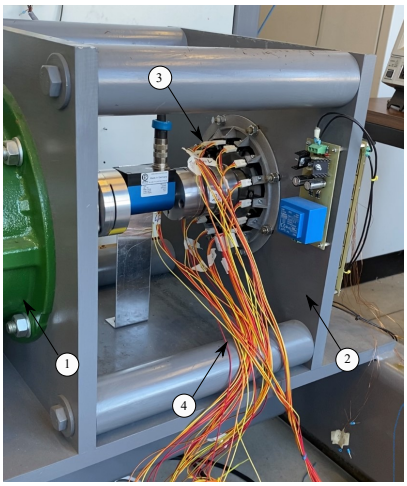


Fig. 8: Picture of the measurement setup. 1: Induction machine; 2: Panel for sensor mounting; 3: Optical switch; 4: Sensor cables to MicroLabBox

B. Regime Measurement

The first measurement discussed in this article is a regime measurement at 500 rpm. The results of this measurement can be seen in Fig. 9. There is an improvement by the consensus-based averaging algorithm: the amplitude of $\Delta\theta$ is reduced from 4.5 electrical degrees to 2.5 electrical degrees.

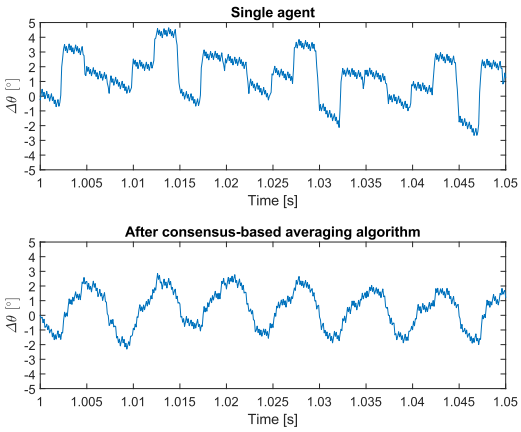


Fig. 9: Measurement in regime at a mechanical speed of 500 rpm.

C. Speed Reversal Measurement

The consensus algorithm was also tested for dynamical situations. In Fig. 10 and Fig. 11 the measurement results for a speed reversal caused by a negative torque of 20 Nm are plotted. Also in these measurements, similar conclusions can be drawn as the consensus-based averaging algorithm leads to improved results.

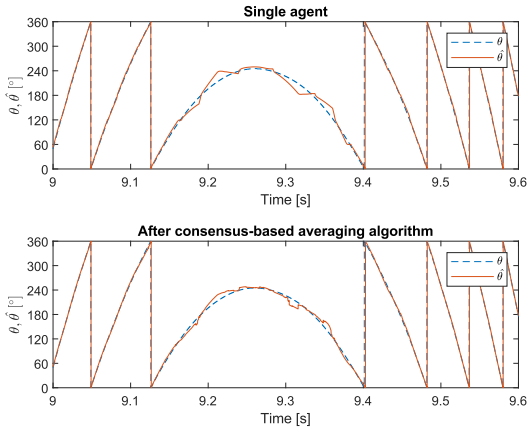


Fig. 10: Measurement of a speed reversal. In this figure, the waveforms of $\hat{\theta}$ are plotted instead of $\Delta\theta$.

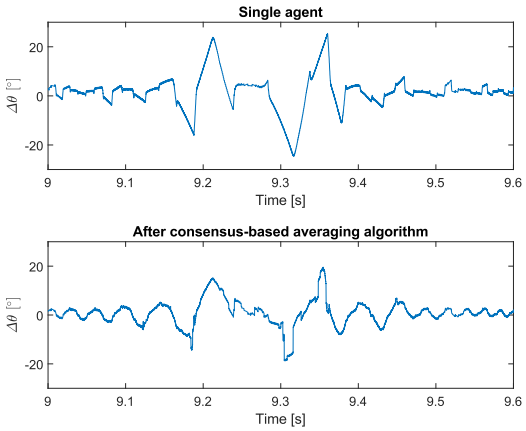


Fig. 11: Measurement of a speed reversal.

D. Faulted Sensor Measurement

In Fig. 12, $\Delta\theta$ of the agent with one faulted sensor and $\Delta\theta$ of the same agent with the consensus-based averaging algorithm and the fault detection algorithm is plotted. As one can see, the error of the single agent reaches almost minus sixty degrees. This estimation is not useful for vector control of the machine. Thanks to the combination of algorithms, the error is made a lot smaller.

E. Agent Shutdown Measurement

For this situation, it might be more interesting to examine what happens at the neighbour of the agent that is shut down. The results can be seen in Fig. 13. At four seconds in the measurement, the shut down agent is excluded from the

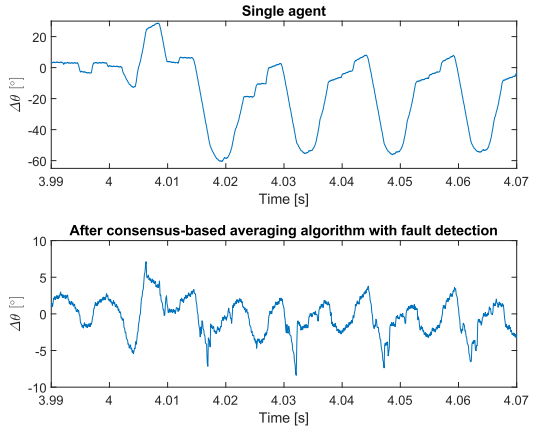


Fig. 12: Measurement when one sensor is faulted at a mechanical speed of 500 rpm. The sensor fails at four seconds.

average and its neighbour performs the consensus algorithm with his own input and with the input from its other neighbour. The magnitude of the error $\Delta\theta$ increases after four seconds but stays smaller than without the consensus-based averaging algorithm.

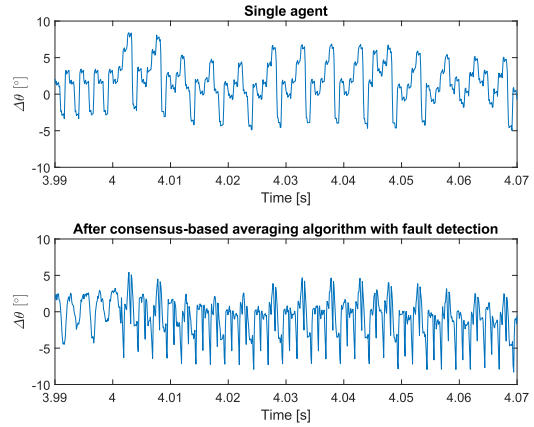


Fig. 13: Measurement when one agent is shut down at a mechanical speed of 1500 rpm. In this Figure, the output of the neighbour of the shutdown agent is plotted. The agent is shut down at four seconds.

F. Numerical Analysis of the Measurements

For all these measurements, a measure for the deviation (dev) was calculated to analyse numerically the improvement by the consensus and the fault detection algorithms. The formula for dev is given in equation (4). The length of the period over which dev is calculated was two seconds and if dev was calculated for a fault situation, a new steady state regime was always already established. The numerical values for dev are in Table I.

$$dev = \sum_{t=0\text{s}}^{2\text{s}} |\Delta\theta(t) - \overline{\Delta\theta}|, \tag{4}$$

TABLE I: Measure for the deviation from the real rotor position for the measurements.

	Deviation dev [electr. rad]	
Measurement	Single Agent	After C. and Fault Det. Algorithm
Regime	610	403
Reversal	1362	910
Sensor failure	6997	671
Module shutdown	1507	774

with $\overline{\Delta\theta}$ the average of $\Delta\theta$ in that period of two seconds.

VI. Sensor Misalignment

When the values of dev of the measurements were compared to those obtained in the simulations, a large difference was observed. This is due to misplacement of the Hall sensors. To simulate the effect of misplaced sensors, the real locations of the rising edges and the falling edges of the sensors need to be measured. These locations are then used in simulations and the value of dev became much larger than in the simulations with ideal sensor positions. In [6], a self calibrating technique is proposed to mitigate the effect of misaligned sensors.

VII. Conclusion

By using Hall sensors and a vector-tracking observer, the position measurement of the rotor in a MMD is now decentralised. It is now prevented from being a single point of failure of the MMD. The estimated rotor position is improved by introducing a consensus-based averaging algorithm. Additionally, a fault detection algorithm has been created, which makes the consensus-based averaging algorithm fault tolerant.

References

[1] G. Calzo, G. Vakil, B. Mecrow, S. Lambert, T. Cox, C. Gerada, M. Johnson, and R. Abebe, "Integrated motor drives: State of the art and future trends," *IET Electric Power Applications*, vol. 10, 04 2016.

[2] A. Shea and T. M. Jahns, "Hardware integration for an integrated modular motor drive including distributed control," *2014 IEEE Energy Conversion Congress and Exposition (ECCE)*, pp. 4881–4887, 2014.

[3] N. R. Brown, T. M. Jahns, and R. D. Lorenz, "Power converter design for an integrated modular motor drive," in *2007 IEEE Industry Applications Annual Meeting*, 2007, pp. 1322–1328.

[4] M. J. Duran and F. Barrero, "Recent advances in the design, modeling, and control of multiphase machines—part ii," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 1, pp. 459–468, 2016.

[5] H. Vansompel, "Regeling van een permanente-magneet-bekrachtige synchrone machine met een positiesensor met lage resolutie," *Master thesis to obtain the degree of Master of Science in Electromechanical Engineering*, 2009.

[6] D. A. Papathanasopoulos, D. V. Spyropoulos, E. D. Mitronikas, and A. D. Karlis, "Commutation angle self-calibrating technique for brushless dc motor drives with defective hall-effect position sensors," in *2020 International Conference on Electrical Machines (ICEM)*, vol. 1, 2020, pp. 1301–1307.

Contents

Permission of Usage v

Preface vii

Abstract ix

List of Figures xx

List of Tables xxviii

List of Abbreviations and Notations xxix

1 Introduction 1
- 1.1 Integrated Modular Motor Drives . . . 1
  - 1.1.1 Integrated Motor Drives . . . 1
  - 1.1.2 Integrated Modular Motor Drives . . . 2
- 1.2 Distributed Control . . . 3
- 1.3 Goal of the Master's Dissertation: a Distributed Position Estimation . . . 6
  - 1.3.1 Low Resolution Position Estimation . . . 6
  - 1.3.2 Consensus Based Position Estimation . . . 6
- 1.4 Conclusion . . . 8

2 Low-Resolution Position Sensor 9
- 2.1 Introduction to Position Measurement . . . 9
  - 2.1.1 Vector Control . . . 9
  - 2.1.2 Conclusion . . . 11
- 2.2 Methods to Measure or Estimate the Rotor Position . . . 11
  - 2.2.1 The Encoder . . . 11
  - 2.2.2 Sensorless Control . . . 12
  - 2.2.3 Estimation Methods using Hall Sensors . . . 12
- 2.3 Vector-tracking Observer . . . 13
  - 2.3.1 Low-resolution Position Vector . . . 14

2.3.2 Discrete Time Vector-tracking Observer (VTO) 16
2.3.3 Variable gains 26
2.3.4 Decoupling Additional Harmonics 30
2.4 Conclusion 32

3 Consensus-Based Averaging Algorithm 37
3.1 Introduction 37
3.1.1 Communication Between the Agents 38
3.1.2 The goal 38
3.2 Converging Dynamic Average Consensus Algorithms 39
3.2.1 Static Average Consensus Algorithms 39
3.2.2 From Static to Dynamic Average Consensus 39
3.2.3 Simulations 42
3.2.4 Conclusion 45
3.3 Consensus-based Averaging Algorithm 47
3.4 Simulations 49
3.4.1 Regime Simulations 50
3.4.2 Start-up Simulation 52
3.4.3 Speed Reversal Simulation 53
3.4.4 Numerical Analysis of the Simulations 58
3.5 Practical Implementation and Measurements 59
3.5.1 The Electrical Machine 59
3.5.2 Construction and Placement of the Sensors 59
3.5.3 Reference Position Measurement 61
3.5.4 Processing Signals and Controlling Speed 61
3.5.5 Parameters Used in the Measurements 62
3.5.6 Results of the Measurements 63
3.6 Conclusion 71

4 Fault Detection Algorithm 73
4.1 Introduction 73
4.2 Fault Detection Algorithm 74
4.3 Simulations and Measurements 76
4.3.1 Simulations 76
4.3.2 Measurements 89
4.4 Conclusion 99

5 Sensor Misalignment 100
5.1 Introduction 100
5.2 Real Sensor Outputs 100

5.3 Simulations 101
5.4 Conclusion 104

6 Conclusion and Further Research **107**
6.1 Conclusion 107
6.2 Further Research 108

A Code Simulations DAC **110**
A.1 Introduction and Credits 110
A.2 Generation of Angle Waveforms 110
A.3 Dynamic Average Consensus 111

Bibliography **114**

List of Figures

1.1 Difference between integrated motor drive concept and conventional drive [2] 2
1.2 Schematic drawing of the IMMD concept [3] 4
1.3 Basic six phase IMMD configuration. [3] 5
1.4 Diagram of IMMD distributed control configuration. [3] 5
1.5 Visual representation of the communication between five agents. Every agent only communicates with its two closest neighbours. 7

2.1 Graph with two reference frames: the $\alpha\beta$-frame attached to the stator and the qd-reference frame attached to the rotor. The instantaneous angle between the rotor and the stator is θ 10
2.2 Three graphs with respectively the output of sensor 1, sensor 2 and sensor 3. Every single sensor has a resolution of 180 electrical degrees. By placing them strategically, six sectors can be created. The sectors are separated by the vertical lines. The output of the sensors is multiplied with a scaling factor to obtain logical 1's and 0's. 14
2.3 Example of how the angle of the discrete vector $\vec{H}_{\alpha\beta}$ looks in comparison to the angle of the continuous vector $\vec{H}$. The angle is a function of time with a rotational speed of 5 electrical radians per second. 17
2.4 Basic scheme of the VTO in discrete time 18
2.5 Scheme of the vector-tracking observer (VTO) with linearisation of the feedback through the vector product. Notice that some blocks have been moved in comparison to Figure 2.4. This is to highlight and isolate the PID controller. 20
2.6 Simulation in regime at a mechanical speed of 500 rpm. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 24

2.7 Simulation in regime at a mechanical speed of 1000 rpm. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 25
2.8 Simulation in regime at a mechanical speed of 1500 rpm. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 26
2.9 Simulation in regime at a mechanical speed of 500 rpm with variable gains. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 28
2.10 Simulation in regime at a mechanical speed of 1000 rpm with variable gains. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 29
2.11 Simulation in regime at a mechanical speed of 1500 rpm with variable gains. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 30
2.12 Decoupling of the additional harmonics added to the scheme of the vector-tracking observer (VTO) in discrete time. 32
2.13 Result after subtraction of the additional harmonics. When a discontinuous jump occurs in the harmonics, a distortion occurs in the sine waveform. . . 33
2.14 Simulation in regime at a mechanical speed of 500 rpm with decoupling of additional harmonics. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 34
2.15 Simulation in regime at a mechanical speed of 1000 rpm with decoupling of additional harmonics. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 35
2.16 Simulation in regime at a mechanical speed of 1500 rpm with decoupling of additional harmonics. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles. 36

3.1 Visual representation of the communication between five agents. Every agent only communicates with its two closest neighbours. 38
3.2 A block diagram of a static average consensus algorithm. When the initial conditions for $\mathbf{x}$ are equal to $\mathbf{u}$, this scheme converges to the exact average of the elements of $\mathbf{x}_0$, if $\mathbf{L}$ meets the requirements from the theorem in subsection 3.2.2. 40

3.3 A block diagram of an accelerated and robust dynamic average consensus algorithm. This system has two internal states. 41
3.4 Composition of one element of the input vector $\mathbf{u}_k$. The lower graph is the summation of both graphs above. 44
3.5 Simulation that shows the difference with the real average before and after the dynamic average consensus algorithm. The mechanical speed of the electrical machine is 500 rpm. 45
3.6 Simulation that shows the difference with the real average before and after the dynamic average consensus algorithm. The mechanical speed of the electrical machine is 1000 rpm. 46
3.7 Simulation that shows the difference with the real average before and after the dynamic average consensus algorithm. The mechanical speed of the electrical machine is 1500 rpm. 47
3.8 Scheme to illustrate the working principle of the consensus-based averaging algorithm. The result is the estimate of the average by agent i. The same scheme is followed to calculate the cosine of the average. 48
3.9 Scheme to illustrate the communication between the agents. 49
3.10 Simulation in regime at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 50
3.11 Simulation in regime at a mechanical speed of 1000 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 51
3.12 Simulation in regime at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 52
3.13 Simulation of a start-up for an acceleration of 570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent that is based solely on the VTO of this agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm. 54
3.14 Simulation of a start-up for an acceleration of 570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 55
3.15 Simulation of a reversal for an acceleration of -570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm. 56

3.16 Simulation of a reversal for an acceleration of -570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 57
3.17 Figure of the optical switch. © OMRON Corporation 60
3.18 3D model of the disc that is mounted on the rotating shaft. The outer part has eight openings to imitate a machine with eight pole pairs. Design of the disc done by Prof. Dr. Ir. Hendrik Vansompel 61
3.19 Picture of the measurement setup. 1: Induction machine; 2: Panel for sensor mounting; 3: Optical switch; 4: Sensor cables to MicroLabBox 63
3.20 Measurement in regime at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 64
3.21 Measurement in regime at a mechanical speed of 1000 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 65
3.22 Simulation in regime at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 66
3.23 Measurement of a start-up with an acceleration of 570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm. 67
3.24 Measurement of a start-up with an acceleration of 570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 68
3.25 Measurement of a speed reversal with an acceleration of -570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm. 69
3.26 Measurement of a speed reversal with an acceleration of -570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm. 70

4.1 Procedure executed by agent i to examine if agent h deviates. An analog procedure is executed by agent i to examine if agent j deviates. 74
4.2 Decision three of agent i for determining whether agent h, agent j or agent i itself is faulted. 75

4.3 Simulation when one sensor is faulted at a mechanical speed of 1000 rpm. The dashed line is the sinewave of the real angle θ. The full line is the waveform of the estimated angle $\hat{\theta}$ by an agent with one faulted sensor. . . 77
4.4 Simulation of $|\overline{\Delta_{hi}}|$ for the sine part of the fault detection algorithm at a mechanical speed of 1500 rpm. In the upper figure, normal operation is simulated. In the middle figure, agent h has a faulted sensor. In the lower figure, agent h is shut down. A horizontal line is drawn at $|\overline{\Delta_{hi}}| = 0.05$. . . 78
4.5 Simulation of $|\overline{\Delta_{hi}}|$ for the cosine part of the fault detection algorithm at a mechanical speed of 1500 rpm. In the upper figure, normal operation is simulated. In the middle figure, agent h has a faulted sensor. In the lower figure, agent h is shut down. A horizontal line is drawn at $|\overline{\Delta_{hi}}| = 0.05$. . . 79
4.6 Simulation when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 80
4.7 Simulation when one sensor is faulted at a mechanical speed of 500 rpm. The error $\Delta\theta$ of the agent with the faulted sensor is plotted with implementation of the consensus algorithm but without the fault detection algorithm. The sensor fails at four seconds. 81
4.8 Simulation when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor based on its VTO solely. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 82
4.9 Simulation when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 83
4.10 Simulation when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 84

4.11 Simulation when one agent is shut down at a mechanical speed of 500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down with implementation of the consensus and fault detection algorithm. The lower figure is the zoomed version of the upper figure. The agent is shut down at four seconds. 85
4.12 Simulation when one agent is shut down at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds. 86
4.13 Simulation when one agent is shut down at a mechanical speed of 1500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down with implementation of the consensus and fault detection algorithm. The lower figure is the zoomed version of the upper figure. The agent is shut down at four seconds. 87
4.14 Simulation when one agent is shut down at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds. 88
4.15 Measurement when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 91
4.16 Measurement when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 92

4.17 Measurement when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 93
4.18 Measurement when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds. 94
4.19 Measurement when one agent is shut down at a mechanical speed of 500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down. The agent is shut down at four seconds. 95
4.20 Measurement when one agent is shut down at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds. 96
4.21 Measurement when one agent is shut down at a mechanical speed of 1500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down. The agent is shut down at four seconds. 97
4.22 Measurement when one agent is shut down at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds. 98

5.1 Simulation in regime at a mechanical speed of 500 rpm with real sensor outputs. The upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm. . 104
5.2 Simulation in regime at a mechanical speed of 1000 rpm with real sensor outputs. The upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm. . 105

5.3 Simulation in regime at a mechanical speed of 1500 rpm with real sensor outputs. The upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm. . 106

List of Tables

2.1 Properties from each sector and its representation. The sequence starts from the positive α-axis and goes counterclockwise. 16
2.2 Results of the analytical calculation for the PID parameters 24

3.1 Comparison between average consensus algorithms. 42
3.2 Measure for the deviation from the real rotor position for the simulations. . 59
3.3 Characteristics of the AFPMSM 60
3.4 Parameters used in the measurements 62
3.5 Measure for the deviation from the real rotor position for the measurements. 72

4.1 Measure for the deviation from the real rotor position for the simulations with fault situations. 90
4.2 Measure for the deviation from the real rotor position for the measurements with fault situations. 92

5.1 Locations of the rising and falling edges of the square waves of the sensors for one mechanical rotation. Sensor 1 to 8 102
5.2 Locations of the rising and falling edges of the square waves of the sensors for one mechanical rotation. Sensor 9 to 15 103
5.3 Measure for the deviation from the real rotor position for the simulations with sensor misalignment. 105

List of Abbreviations and Notations

Abbreviations

AFPMSM	Axial flux permanent magnet synchronous machine
DC	Direct current
EMI	Electromagnetic interference
FET	Field-effect transistor
FOC	Field oriented control
GaN	Gallium nitride
IMD	Integrated motor drive
IMMD	Integrated modular motor drive
MMD	Modular motor drive
PE	Power electronics
PID	Proportional - integral - derivative
PMSM	Permanent magnet synchronous machine
PWM	Pulse-width modulation
rpm	Revolutions per minute
SiC	Silicium carbide
VSI	Voltage source inverter
VTO	Vector-tracking observer

Notations

$\mathbf{x}$	Bold lowercase letter: array
$\mathbf{Q}$	Bold uppercase letter: matrix

Chapter 1

Introduction

The goal of this master's dissertation is to make a decentralised position measurement of the rotor in a modular motor drive. This position is necessary to make use of field oriented control. This introductory chapter will explain the background and context that motivated the goal of this master's dissertation.

1.1 Integrated Modular Motor Drives

1.1.1 Integrated Motor Drives

Technology evolves constantly, certainly in the world of power electronics. This progress leads to more compact power electronic components which are more resistant to heat and vibrations. One of the consequences is that it is now possible to integrate the inverter close to the challenging environment of a stator in electrical machines. In an integrated motor drive (IMD), the power electronics and the motor are not physically split anymore. One way to realize this is mounting the power electronics on the surface of the motor. An example of this can be seen in Figure 1.1. According to [1] an IMD is defined as *the result of the functional and structural integration of the power electronics (PE) converter with the machine as a single unit taking into consideration the electrical and structural and thermal impacts both components have on each other and the system as a whole.* This definition points towards the advantages of an IMD and also to the challenges. The design of an IMD is more compact than a separate control and motor unit. This leads to shorter cables, which reduces the risk of emitted electromagnetic interference (EMI) and avoids overvoltage due to transmission line effects. IMDs are also cheaper: there is no need for an output filter because of the shorter cables. Also, the integration of the power electronics

into the housing of the motor leads to an increased automation of the manufacturing process. Next to that, a singular cooling can be used for both the power electronics and the motor. This brings us to the main challenge of an IMD: since the temperature in the stator windings can become very high, thermal management becomes crucial in order to preserve the power electronics [1]. This problem and the practical size of the converter puts a limit on the maximal power rating of an IMD. Siemens has developed in 2015 an IMD with the power electronics mounted on the surface of the stator housing, this solution is called the SIVETEC MSA 3300. It has the potential for power ratings up to 200kW [2].

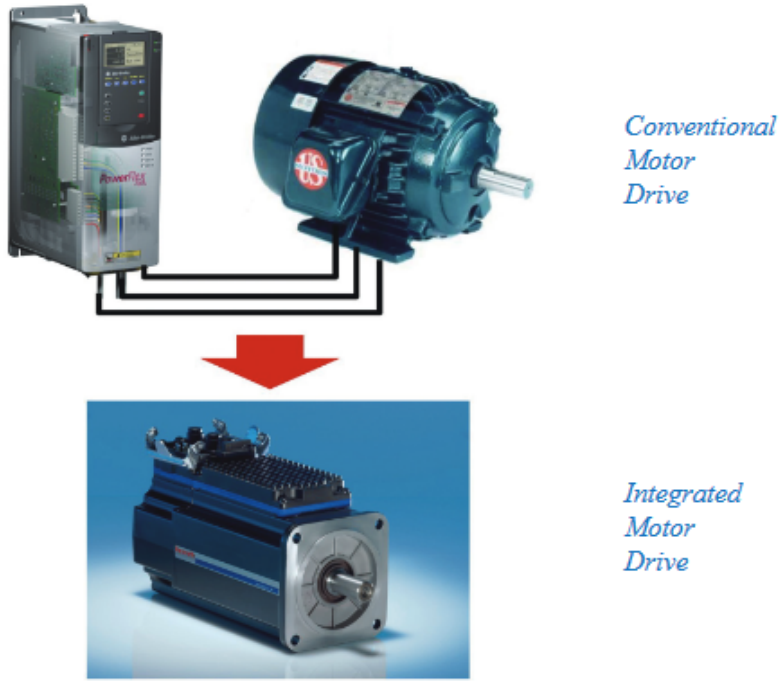


Figure 1.1: Difference between integrated motor drive concept and conventional drive [2]

1.1.2 Integrated Modular Motor Drives

An important step in the evolution towards the complete absorption of the motor drive into the motor enclosure, is the discretization of both the motor and the power electronics into modules. This gives rise to the concept of an Integrated Modular Motor Drive (IMMD). An example of such an IMMD is presented in Figure 1.2. The modules can be seen as the building blocks of an IMMD, simplifying the construction of the drive system. One module

consists of a stator core element with a concentrated winding (which is called a machine pole in Figure 1.2), fed by a dedicated power electronic converter and controller. The purpose of this modular approach was dual: on the one hand to improve the manufacturability and on the other hand to make the motor fault tolerant [2]. Next to these two important advantages of an IMMD, another beneficial consequence of segmenting the stator into modules arise. It is now also possible to segment the converter. The authors of [4] propose a modular multilevel converter where the modules can be series connected. When the modules are series connected, low-voltage (<200 V) FET-structures based on Silicium Carbide (SiC) en Gallium Nitride (GaN) can be used. These low-voltage switches enable higher switching frequencies which reduce the size of the DC-link capacitors [5]. In [6], a comparison between different types of converters is conducted.

The concept of an IMMD is very promising, it is the end stage of the road to compact electrical machines with easily replaceable parts and straightforward control. However, the concept nowadays still faces some important challenges as for example issues concerning thermal management. A more general type of modular drives is the modular motor drive (MMD). The difference with an IMMD is that the controller and the power electronics are not necessarily integrated into the stator, omitting some challenges from the IMMD concept. In [7], the name 'IMMD' is used for a modular drive with the PE components mounted on top of the stator. This realisation is slightly different than the concept from Figure 1.2, but can also be seen as an IMMD. It shows that a MMD and an IMMD are not distinctively different concepts. Initially, the algorithms introduced in this dissertation will be used in MMDs since this concept covers a wide range of modular drives, but the exact same algorithms are applicable in IMMD concepts. The integration of the controller and the power electronics into the stator is no subject of this dissertation.

1.2 Distributed Control

The modular approach in a MMD allows individual control of every stator winding, this leads to fault-tolerant techniques and a decentralised control of the machine. The additional degrees of freedom that are introduced in this way, are often used in literature for multiphase control. Conventionally, an electrical machine is controlled with one centralised controller. Also in the majority of the presented MMD concepts, the power electronics are integrated in every module but a centralised controller operates all these pole drive units [8]. A centralised controller can be fault tolerant: it is able to exclude the faulted modules from the drive system [9]. A major disadvantage is that there is no redundancy in the

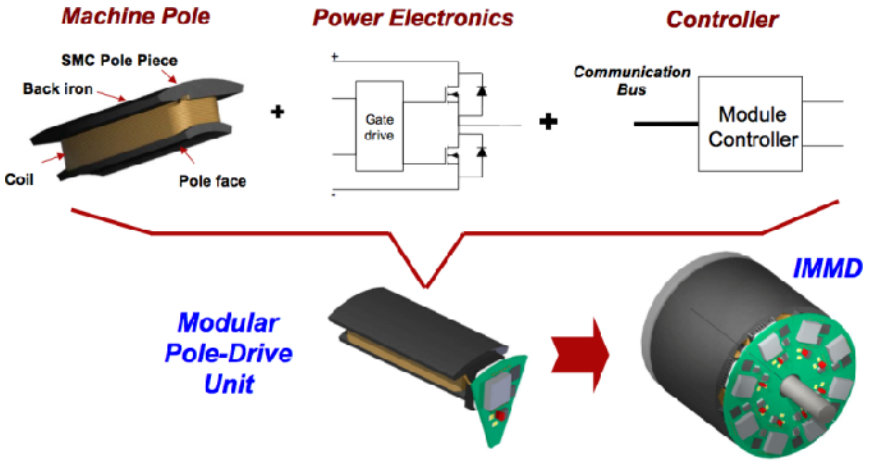


Figure 1.2: Schematic drawing of the IMMD concept [3]

control unit itself. If this controller breaks down, no control is possible anymore. A new generation of MMDs demonstrates that it is possible to segment this centralised controller as well. When a module of such an MMD fails, the other modules continue working and the machine is still controllable. The control unit of the machine is not a single point of failure anymore[1]. This enables one to use the full potential of the MMD.

A major difficulty in decentralised control, however, is that a module only has local information to compute the appropriate output to obtain a smooth torque characteristic. In [10] a distributed control strategy is proposed to cope with this difficulty. Also mutual coupling of the phases in a multiphase motor complicates accurate decentralised control. In [10], neighbouring modules can communicate with each other intelligently to exchange information, which leads to similar performances as a centralized control strategy. Another strategy that relies on communication with neighbouring modules is proposed in [3]. The authors of [3] opted for six phases divided in two groups. A scheme of this concept is in Figure 1.3. A diagram of the distributed control topology is in Figure 1.4. Each pole drive unit is equipped with a controller, named after the phases in Figure 1.3. The controllers operate as synchronized peers running identical programs while sharing sensor information as needed. That sensor information is the current feedback of every phase. All the controllers share a single position measurement from an encoder.

[1]This is true except for the position measurement [3], hence the purpose of this research.

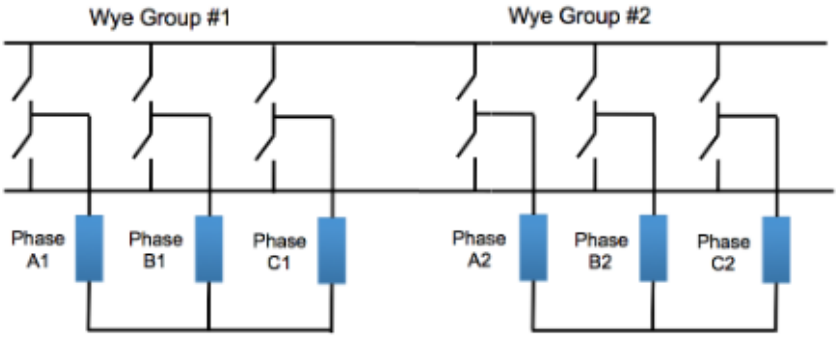


Figure 1.3: Basic six phase IMMD configuration. [3]

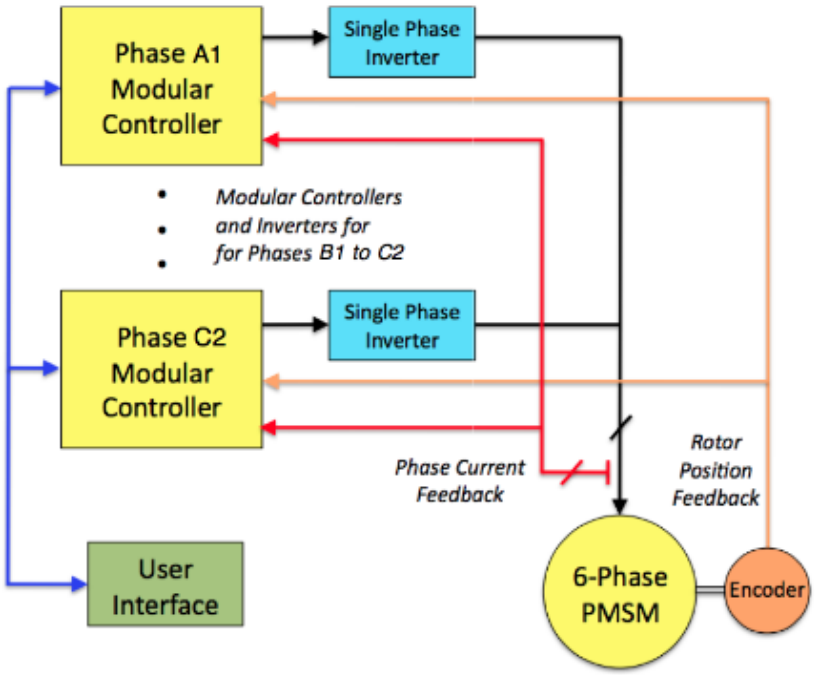


Figure 1.4: Diagram of IMMD distributed control configuration. [3]

1.3 Goal of the Master's Dissertation: a Distributed Position Estimation

The modules need the position of the rotor to calculate the appropriate currents to use field orientation. When this position is measured in a centralised way by means of a single, delicate resolver or encoder, the position measurement remains a single point of failure in the MMD. As resolvers and encoders are expensive components that need to be mounted on the rotor shaft, it is not an option to equip each module with its own dedicated resolver or encoder.

This leads to the goal of this master's dissertation: to propose a distributed algorithm that results in a decentralised, robust and cheaper estimation of the rotor position. This distributed position estimation is the last step towards an entirely decentralised control of a MMD. For the proposed distributed technique, each module of the MMD will get its own cheap, low resolution Hall sensor. At first, the outputs of three of these Hall sensors will be combined to obtain an initial position estimate per group of three modules, based on the technique proposed in [11]. In a second step, these initial estimates will be improved by means of communication between the modules. These two steps are summarized in the following two subsections.

1.3.1 Low Resolution Position Estimation

In [11], a strategy is proposed to estimate the position of the rotor based on cheap and robust Hall sensors. In Chapter 2 the exact working principle is explained thoroughly. In short, m Hall sensors are equally distributed over the whole circumference of the stator. One Hall sensor is able to measure whether the rotor is in a sector of 180 electrical degrees or not. When, for example, three Hall sensors are equally distributed over the circumference of the stator, six different sectors are defined. This gives the rotor position estimation an initial resolution of 60 degrees. The authors of [11] propose two additional techniques to dramatically increase the resolution of this sensor system: using a vector-tracking observer with variable observer gains, and the reduction of the harmonic content in the input vector of the vector-tracking observer.

1.3.2 Consensus Based Position Estimation

The modules of the MMD will have each their own Hall sensor to measure the position of the rotor. To improve the initial position estimation, a communication network will

be established between the modules. A group of modules that makes a first estimate of the position of the rotor is called an agent. This estimation is done with the algorithm in Chapter 2. Next to the initial communication between the modules of a single agent, there is also communication between the agents themselves. An example to make this clear: suppose that the stator consists of fifteen modules. That is for example, five groups of three modules or five agents. Each agent will have their own estimation of the position based on the three Hall sensors in that group of modules. Communication between the agents will lead by a consensus algorithm proposed in this dissertation to a consensus-based position estimate. This consensus algorithm is introduced in Chapter 3. The structure of the communication between the five neighbours can be seen on Figure 1.5.

Another advantage of the information exchange is that it introduces fault tolerance. In the fault detection algorithm proposed in this dissertation, a faulted agent is recognised by its neighbours, and consequently, no weight is given to the position estimate of the faulted agent in the consensus algorithm. A module with a faulted sensor will also be able to continue operating by the position estimates of the other agents. This fault detection algorithm is introduced in Chapter 4.

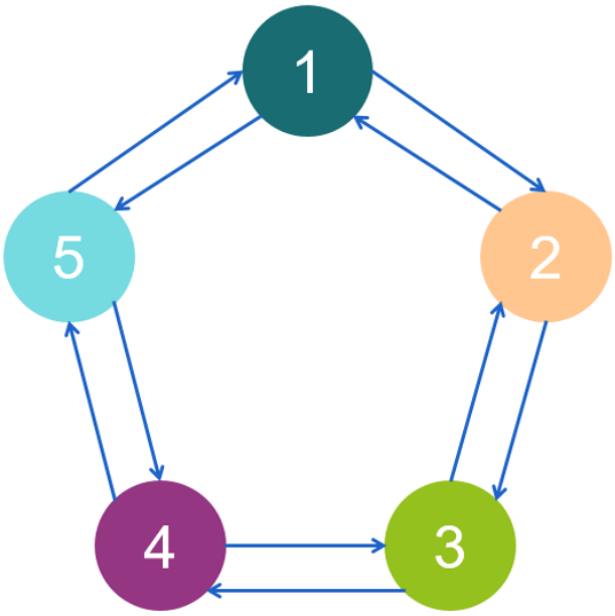


Figure 1.5: Visual representation of the communication between five agents. Every agent only communicates with its two closest neighbours.

1.4 Conclusion

The main contribution of this dissertation is the decentralisation of the position measurement in a MMD by means of cheap and robust, but low-resolution Hall sensors. This makes the rotor position estimation more fault tolerant. Low resolution position estimation techniques, improved by means of the consensus-based algorithm proposed in this dissertation, results in an adequate estimate of the rotor position.

Chapter 2

Low-Resolution Position Sensor

In this chapter, the principle of the low-resolution position estimation by one agent is explained. The position will be estimated by means of Hall sensors. A single agent consists of three Hall sensors in this dissertation. The number of sensors that are used for making one estimation, i.e. the number of Hall sensors in one agent, can be varied. It depends on the resolution that is needed, but minimum two sensors are required. However, with the techniques proposed in this chapter, three sensors will be used to control a Permanent Magnet Synchronous Machine (PMSM). This chapter is based on the master's dissertation of prof. Dr. Ir. Hendrik Vansompel [12].

2.1 Introduction to Position Measurement

An accurate position measurement is vital in many control algorithms for electrical machines. Many industrial applications require an accurate control of the torque and/or the speed of the electric drive. To see how important knowledge about the position of the rotor is to control torque and speed, one has to dive into the principles of vector control.

2.1.1 Vector Control

In the following paragraph, vector control of a PMSM will be explained briefly. The MMD on which the algorithms will be tested is an axial flux permanent magnet synchronous machine (AFPMSM).

In rotating field machines, torque is produced by the interaction between a rotating sinusoidal current layer and a rotating magnetic field distribution. The magnitude of the

torque is proportional to the magnitude of the field, the magnitude of the current layer and to the cosine of the angle between the symmetry axis of the field and the current layer.

Straightforward control of a rotating field machine is obtained when the stator current can be controlled independently from the rotor field. This torque control loop is called vector control. When the stator current is kept orthogonal to the rotor field, the term field oriented control (FOC) is used. In Figure 2.1 the rotor reference frame or qd-reference frame is shown. The d-axis is always attached to the field flux axis. I_s is a fixed vector when using vector control. I_s can be decomposed in I_{sq} and I_{sd}, these values are constants in vector control. In FOC I_s is orthogonal to the field flux axis and thus I_{sd} is always zero. The stator current has only the torque producing current I_{sq}.

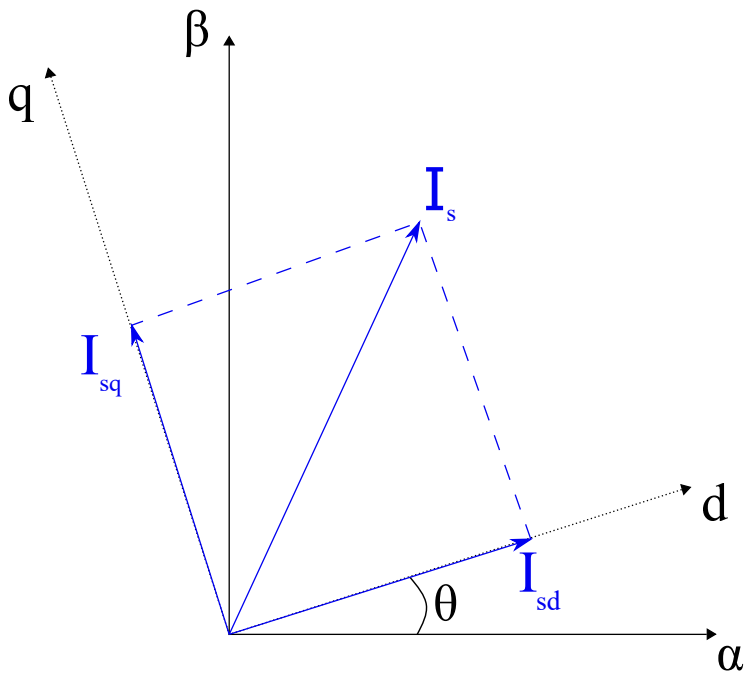


Figure 2.1: Graph with two reference frames: the $\alpha\beta$-frame attached to the stator and the qd-reference frame attached to the rotor. The instantaneous angle between the rotor and the stator is θ

To apply FOC, the position of the rotor has to be known, because the position of the rotor determines the location of the field flux axis. Based on that information, it is possible to keep the armature current orthogonal to the field flux axis. There are two external inputs for controlling an electric drive: i_q^s and i_d^s. These inputs are obtained via a torque control loop, sometimes supplemented with a speed control loop. These set values have

to be transformed into useful phase currents: i_a^s, i_b^s and i_c^s. The first step is the inverse Park-transformation to go from the rotor reference frame (qd-reference frame) to the two phase stator reference frame ($\alpha\beta$-reference frame). In the second step, the inverse Clarke-transformation is used to go to the three-phase reference system (abc-reference system). Based on these currents, a voltage-source inverter (VSI-PWM) can calculate the required switching angles to obtain vector control. When i_d^s is zero, FOC is applied. An accurate position measurement results in a good Park-transformation. This leads to perfect sinusoidal currents and voltages, which leads to a smooth torque characteristic. This shows the importance of an accurate position measurement. The position of the rotor is expressed by θ. When the estimated position $\hat{\theta}$ is used in the inverse Park transformation, the transformation will not be perfect. This causes i_d currents and a different phase angle. This causes current ripples.

2.1.2 Conclusion

Above, a general introduction into vector control and FOC is given. This is done to illustrate the importance of a correct position measurement of the rotor. For the control of a modular drive, the principles stay the same. However, there are some extra subtleties and difficulties when controlling a modular drive. These challenges will not be discussed in this dissertation. For this research, it is sufficient to know that the better the position estimation is, the smoother the torque characteristic of the machine will be.

2.2 Methods to Measure or Estimate the Rotor Position

2.2.1 The Encoder

The encoder can measure the position of the rotor with high accuracy, independent of the speed of the rotor. This is beneficial during start-up of the machine. It is currently one of the most common ways to measure θ. However, the encoder has some disadvantages. Firstly, it is an expensive device that is not very resistant to vibrations and other tough circumstances in industrial workplaces such as dust. The second disadvantage is that it is not practical for a modular drive. The main principle of the modular drive is that when some parts (modules) fail, the machine will keep on working. A modular drive is inherently fault tolerant. Up until now, all the modules of the modular drive use the position measurement of the same encoder. That makes the encoder a single point of

failure. A solution could be to use multiple encoders. This is not a valuable option since encoders are expensive and an encoder is normally mounted on a free shaft end. To address this problem, another method is proposed in this text.

In this dissertation, an encoder is only used to measure θ accurately during lab tests. This θ is than compared to $\hat{\theta}$ to validate the results of the algorithm.

2.2.2 Sensorless Control

In the last decades, a lot of research is conducted towards sensorless control of PMSMs [13]. This control of electrical machines based on the back emf of the rotor, often used with sinusoidal or square signal injection, is more robust and cheaper than an encoder. The most obvious disadvantage of a sensorless drive is that there is no back emf at standstill. Therefore, no position measurement is possible. To address this problem, the authors of [14] propose a method with an adjusted rotor-design and search coils in the stator. Unfortunately the technology of sensorless control is still not considered as mature enough to guarantee an acceptable performance in a wide range of torque and speed [15]. Especially in traction applications: the motors are designed to be heavily saturated to obtain large power density. According to the authors of [16], the performance under saturation of sensorless control is heavily affected. This is why sensorless control is not a viable option for this research: an IMMD aims for high power density.

2.2.3 Estimation Methods using Hall Sensors

When working with a PMSM, Hall sensors can be used to track the position of the rotor. If the magnets are mounted on the surface of the rotor, a binary Hall sensor integrated in the stator outputs a square wave. The output state is high when the rotor is in a specific zone of 180 electrical degrees and otherwise. Three Hall sensors equally distributed over the stator create six sectors and thus a resolution of 60 electrical degrees. Advantages of using Hall sensors are their robustness and cheaper price. They do not need to be mounted on a free shaft end and thus every module of the modular drive can be equipped with a Hall sensor.

A resolution of 60 degrees is not good enough to obtain accurate vector control. The signals of the Hall sensors will need to be processed further. This can be done in two ways: not based on the model of the drive system or based on the (electrical and mechanical) model [15].

Non-model-based Estimation Methods

This solution exploits signal processing technologies. There are several different data processing technologies that are used: first-order or higher order approximations, least squares interpolation on position trajectory, etc. An advantage of these methods is that no prior knowledge of the machine and the load is necessary to obtain an estimate of the position and the speed of the rotor. Mathematically, it is equivalent to the problem of determining a continuous, and thus differentiable, function that interpolates the quantised discontinuous position measurement from the Hall sensors [17]. To obtain the speed of the rotor, the function of the position of the rotor is differentiated. These methods are easily implemented, but the estimators are quite dependent on the application for which the motor is used. Therefore, these non-model-based estimation methods underperform in situations with large speed variations [17]. It is also not possible to avoid a time delay and spikes in the speed estimation by these methods, this can result in unstable motions [15]. For smooth operation of the electrical machine in a wide array of different speeds, it is better to choose for a model-based estimation method.

Model-based Estimation Methods

Methods using Kalman filters or state observers are based on information about the electrical and mechanical model of the system. Another difference with the non-model-based estimation methods is that in the model-based methods, initially the speed of the rotor is determined. To obtain the rotor position, the speed function is integrated. In [18], the authors propose a vector-tracking observer (VTO). This model-based method is capable of tracking the rotor position with zero lag, unlike e.g. phase-locked loops. The principle of the VTO as well as the tuning of the VTO is explained in the following section. Because of its favourable properties in dynamical situations and its possible zero phase lag, this method is chosen in this dissertation to track the rotor position.

2.3 Vector-tracking Observer

In the first part of this section, a low-resolution position vector is mathematically defined. Subsequently, the basic topology of the VTO is explained. In the last part of this section two additional features are proposed to further improve the estimation of the position of the rotor.

2.3.1 Low-resolution Position Vector

Using a low-resolution position estimation, the interval of 360 electrical degrees or two pole pitches τ_p is subdivided in a limited number of intervals. In this dissertation, one agent is made up by three modules. Because every module is equipped with a Hall sensor, one estimation of the position of the rotor relies on three Hall sensors. This gives six sectors over 360 electrical degrees. This is a choice of the author, using more or less Hall sensors is also possible. The sectors can be seen on Figure 2.2.

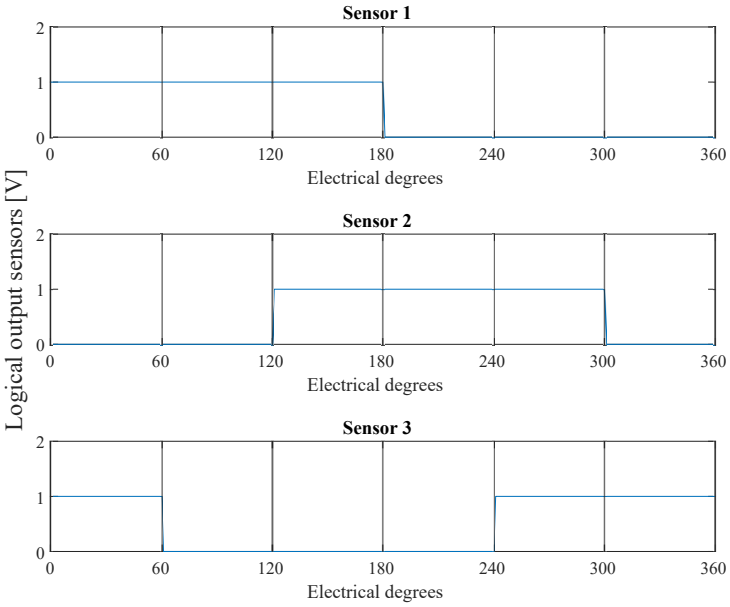


Figure 2.2: Three graphs with respectively the output of sensor 1, sensor 2 and sensor 3. Every single sensor has a resolution of 180 electrical degrees. By placing them strategically, six sectors can be created. The sectors are separated by the vertical lines. The output of the sensors is multiplied with a scaling factor to obtain logical 1's and 0's.

During rotation, the rotor position vector jumps to subsequent sectors. Therefore it can only take six discrete values. Each value is the representation of a sector. The middle of the sector is chosen as representation. Before we go on with the mathematical analysis of this vector, some notations are introduced.

- N. This is the number of sectors in which 360 electrical degrees are subdivided. These sectors are numbered from 1 to N in counterclockwise direction with sector 1

containing the zero degrees mark.

- $\vec{H}_{\alpha\beta}$. This vector shows in which sector of the $\alpha\beta$-reference frame the rotor is located. It can only have N different values, one specific value for each sector. Its magnitude is $H_{\alpha\beta}$. Its phase in the $\alpha\beta$-reference frame, denoted by $\theta\Delta\frac{2\pi}{N}$ to emphasize the discrete character, is dependent on the sector in which it is located. Here the middle of each sector is chosen to represent the sector. Now $\vec{H}_{\alpha\beta}$ can be written as

$$\vec{H}_{\alpha\beta} = H_{\alpha\beta}e^{j\theta\Delta(\frac{2\pi}{N})}. \tag{2.1}$$

- $\vec{H}$. This vector corresponds with the real rotor position. It is a unit vector rotating continuously and given by

$$\vec{H} = e^{j\theta}, \tag{2.2}$$

with θ continuously varying and attached to the rotor.

$\vec{H}_{\alpha\beta}$ is the discrete version of $\vec{H}$ with a resolution of $\frac{360^\circ}{N}$. As can be seen on Figure 2.3, $\vec{H}$ is a continuously rising function with a reset from 2π to 0. $\vec{H}_{\alpha\beta}$ is a staircase function with a step every time $\vec{H}$ enters a new sector.

From this staircase function it is possible to construct a Fourier series in function of θ.

$$\vec{H}_{\alpha\beta}(\theta) = \sum_{k=-\infty}^{+\infty} a_k e^{jk\theta} \tag{2.3a}$$

with a_k given by

$$a_k = \frac{1}{2\pi}\sum_{n=1}^{N} e^{j\theta\Delta(\frac{2\pi}{N})} \int_{2\pi n/N}^{2\pi(n+1)/N} e^{-jk\theta}\mathrm{d}\theta \tag{2.3b}$$

The integration to determine the Fourrier coefficients a_k can be simplified by subdividing the integral in N different sectors. In that way, the expression for the Fourrier coefficients a_k can be simplified to

$$a_k = \begin{cases} 0, & k \neq mN+1 \\ \frac{1}{k}e^{-j\pi/N+\phi}, & k = mN+1. \end{cases} \tag{2.4}$$

In this equation is $\phi = \pi/N$ and $m \in \{0, \pm1, \pm2, \ldots\}$. This ϕ appears because the middle of each sector is chosen to represent the sector. Because of this choice, $\vec{H}_{\alpha\beta}$ and $\vec{H}$ are in phase.

Table 2.1: Properties from each sector and its representation. The sequence starts from the positive α-axis and goes counterclockwise.

$\theta\,(^\circ)$	sector	state $\vec{H}_{\alpha\beta}$	$\theta\Delta\left(\frac{2\pi}{N}\right)(^\circ)$
$[0, 60]$	$s1$	$\vec{H}_{\alpha\beta,1}$	30
$[60, 120]$	$s2$	$\vec{H}_{\alpha\beta,2}$	90
$[120, 180]$	$s3$	$\vec{H}_{\alpha\beta,3}$	150
$[180, 240]$	$s4$	$\vec{H}_{\alpha\beta,4}$	210
$[240, 300]$	$s5$	$\vec{H}_{\alpha\beta,5}$	270
$[300, 360]$	$s6$	$\vec{H}_{\alpha\beta,6}$	330

Combining equations (2.1), (2.3a) and (2.4) gives

$$\begin{aligned}\vec{H}_{\alpha\beta} =& e^{j\theta} - \frac{1}{N-1}e^{-j(N-1)\theta} + \frac{1}{N+1}e^{j(N+1)\theta} \\ &- \frac{1}{2N-1}e^{-j(2N-1)\theta} + \frac{1}{2N+1}e^{j(2N+1)\theta} + \dots .\end{aligned} \tag{2.5}$$

The first term on the right hand side $e^{j\theta}$ is the fundamental component of $\vec{H}_{\alpha\beta}$. This fundamental is the same rotating vector as $\vec{H}$=$e^{j\theta}$. And thus it can be stated that $\vec{H}_{\alpha\beta,1}$ is equal to $\vec{H}$.

2.3.2 Discrete Time Vector-tracking Observer (VTO)

Introduction

To control a PMSM, the vector $\vec{H}_{\alpha\beta}$ does not represent accurately enough the rotor position. The large difference with the real angle leads to a torque ripple. To reduce the deviation from the actual rotor position, one could use a vector-tracking observer (VTO), whose structure is shown on Figure 2.4.

Working Principle

$\vec{H}_{\alpha\beta}$ and the physical torque (T_{em}) are the inputs of the VTO. It gives as output an estimate of the rotor position $\hat{\theta}$. Different parts of the VTO are: a vector product, a PID controller, a model of the mechanical system and a feedback loop.

The mechanical system calculates a rotor speed based on the sum of two torques. On the one hand the physical torque T_{em} which is ought to be given as an external input, and on the other hand a torque, which is outputted by the PID controller, based on $\vec{H}_{\alpha\beta}$. The

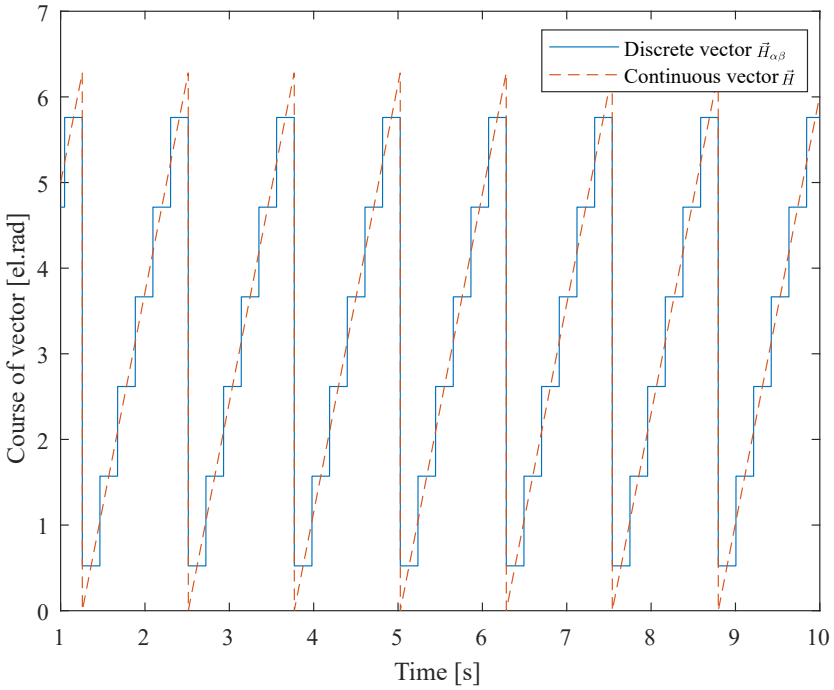


Figure 2.3: Example of how the angle of the discrete vector $\vec{H}_{\alpha\beta}$ looks in comparison to the angle of the continuous vector $\vec{H}$. The angle is a function of time with a rotational speed of 5 electrical radians per second.

mechanical system consists of an integrator multiplied with a constant. This system is based on Newton's equation for rotational motion: $T_m - T_{em} = \frac{J}{N_p} \mathrm{d}\omega/\mathrm{dt}$. T_m is the torque produced by the electrical machine (in Nm), J is the inertia of the system (in kgm^2) and N_p the number of pole pairs. The output is an estimate for the electrical speed $\hat{\omega}$. Then, another integration follows to obtain an estimate for the rotor position $\hat{\theta}$. This estimation is now done based on two torque components. The contribution of one component (T_{em}) is calculated in an open loop. The other contribution (based on $\vec{H}_{\alpha\beta}$) is calculated in a closed loop. Namely, $\hat{\theta}$ is coupled back via $e^{j\hat{\theta}}$ to the input $\vec{H}_{\alpha\beta}$. The residual vector after the feedback is the result of a vector product between the vectors $\vec{H}_{\alpha\beta}$ and $e^{j\hat{\theta}}$. The outcome is a vector but only the magnitude of the vector is used as input for the PID controller. The output of the PID controller is the component of the torque based on $\vec{H}_{\alpha\beta}$.

These two torque inputs for the mechanical system are complementary. If the speed of the machine stays within the bandwidth of the controller, position estimation based on $\vec{H}_{\alpha\beta}$ suffices. When the frequency goes out of bounds, the open loop calculation based on T_{em}

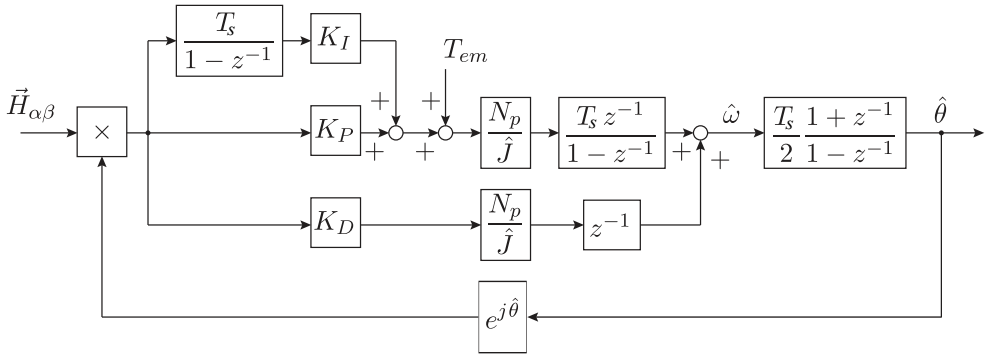


Figure 2.4: Basic scheme of the VTO in discrete time

provides a good estimate of θ. Theoretically, the open loop calculation gives an accurate result for $\hat{\theta}$, only when the inertia J is estimated correctly. When this is not the case, the closed loop calculation of $\hat{\theta}$ will still converge to a good estimate. This means that even with a wrong estimation of J, a good estimation of θ remains.

Linearisation of the VTO

To tune the parameters of the PID-controller, a linearisation of the VTO is conducted. First the vector product is studied in more detail. The purpose of the vector product is to generate a vector that expresses the residual ε after the feedback. This residual is a measure for the difference between $e^{j\hat{\theta}}$ and the fundamental component $\vec{H}_{\alpha\beta,1} = e^{j\theta}$ of the discrete position vector $\vec{H}_{\alpha\beta}$. Remember that the phase of the fundamental $\vec{H}_{\alpha\beta,1}$ is the exact θ at any moment. So one could say that the goal is to track $\vec{H}_{\alpha\beta,1}$ with $e^{j\hat{\theta}}$ as good as possible. Hence the name: vector-tracking observer.

If we now express the vectors in their orthogonal components we get

$$\vec{H}_{\alpha\beta,1} = \cos(\theta) + j\sin(\theta) \tag{2.6a}$$

$$e^{j\hat{\theta}} = \cos(\hat{\theta}) + j\sin(\hat{\theta}) \tag{2.6b}$$

which gives us ε, the magnitude of the residual vector.

$$\varepsilon = \sin(\theta)\cos(\hat{\theta}) - \cos(\theta)\sin(\hat{\theta}) \tag{2.7}$$

Looking at (2.7), feedback by a vector product can be explained. First of all, ε is zero when $\theta = \hat{\theta}$. The PID controller does not have to compensate. There are other values for θ and $\hat{\theta}$ for which ε becomes zero. When this occurs, the VTO is not stable. The VTO

has only local stability. The stability criteria of the VTO is not addressed any further in this text, a study of the stability is conducted in [12].

Now, a small-signal model of equation (2.7) is built. Small perturbations from the regime values denoted with subscript 0 (θ_0 and $\hat{\theta}_0$) will be studied. Differentiating equation (2.7) and evaluation in the regime point gives

$$\begin{aligned} \mathrm{d}\varepsilon = &\left[\sin(\theta)\frac{\mathrm{d}}{\mathrm{d}\hat{\theta}}\cos(\hat{\theta})\right]_0 \mathrm{d}\hat{\theta} \quad + \quad \left[\cos(\hat{\theta})\frac{\mathrm{d}}{\mathrm{d}\theta}\sin(\theta)\right]_0 \mathrm{d}\theta \\ &- \quad \left[\cos(\theta)\frac{\mathrm{d}}{\mathrm{d}\hat{\theta}}\sin(\hat{\theta})\right]_0 \mathrm{d}\hat{\theta} \quad - \quad \left[\sin(\hat{\theta})\frac{\mathrm{d}}{\mathrm{d}\theta}\cos(\theta)\right]_0 \mathrm{d}\theta. \end{aligned} \tag{2.8}$$

When elaborating this further in the regime point, it results in

$$\mathrm{d}\varepsilon = -\sin(\theta_0)\sin(\hat{\theta}_0)\mathrm{d}\hat{\theta} + \cos(\theta_0)\cos(\hat{\theta}_0)\mathrm{d}\theta - \cos(\theta_0)\cos(\hat{\theta}_0)\mathrm{d}\hat{\theta} + \sin(\theta_0)\sin(\hat{\theta}_0)\mathrm{d}\theta. \tag{2.9}$$

Using the angle sum and difference identities gives

$$\begin{aligned} \mathrm{d}\varepsilon &= \cos(\theta_0 - \hat{\theta}_0)\mathrm{d}\theta - \cos(\theta_0 - \hat{\theta}_0)\mathrm{d}\hat{\theta} \\ &= \cos(\theta_0 - \hat{\theta}_0)(\mathrm{d}\theta - \mathrm{d}\hat{\theta}). \end{aligned} \tag{2.10}$$

$\mathrm{d}\varepsilon$ is proportional to ($\mathrm{d}\theta$-$\mathrm{d}\hat{\theta}$) with a factor $\cos(\theta_0 - \hat{\theta}_0)$. The change in $\mathrm{d}\varepsilon$ when $\mathrm{d}\theta$ changes is given by

$$\frac{\mathrm{d}\varepsilon}{\mathrm{d}\theta} = \cos(\theta_0 - \hat{\theta}_0). \tag{2.11}$$

And the change in $\mathrm{d}\varepsilon$ when $\mathrm{d}\hat{\theta}$ changes is given by

$$\frac{\mathrm{d}\varepsilon}{\mathrm{d}\hat{\theta}} = -\cos(\theta_0 - \hat{\theta}_0). \tag{2.12}$$

When the case $\theta_0 = \hat{\theta}_0$ is considered, equations (2.11) and (2.12) become much more simple: $\frac{\mathrm{d}\varepsilon}{\mathrm{d}\theta}$= 1 en $\frac{\mathrm{d}\varepsilon}{\mathrm{d}\hat{\theta}}$= -1. This is equal to the control loop shown in Figure 2.5. Notice that the input is now θ. This is because the vector product is now omitted and it shows the real purpose of the VTO: to track the real angle θ.

Bandwidth of the VTO

The VTO has to track θ as accurate as possible. Therefore the bandwidth has to be chosen as such that at maximum speed, the VTO is still able to track θ. Based on Figure 2.5 this would be the only restriction for the bandwidth. However, the input of the original VTO is the discrete vector $\vec{H}_{\alpha\beta}$. At low speeds and with a high bandwidth, $\hat{\theta}$ will follow

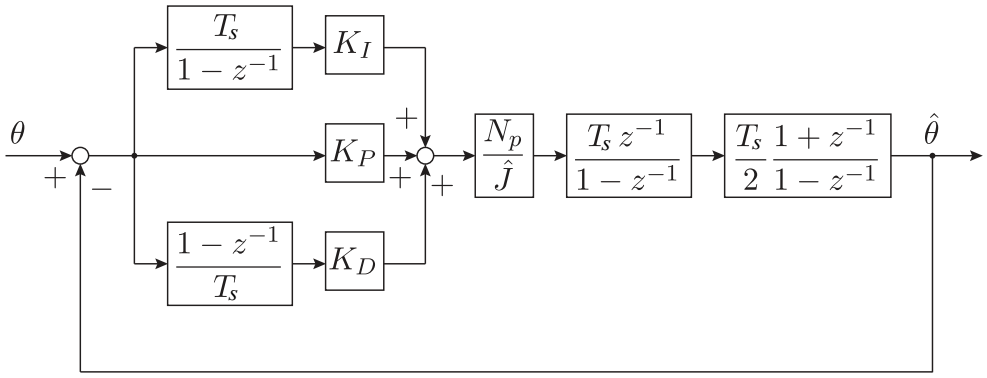


Figure 2.5: Scheme of the vector-tracking observer (VTO) with linearisation of the feedback through the vector product. Notice that some blocks have been moved in comparison to Figure 2.4. This is to highlight and isolate the PID controller.

the discrete structure of $\vec{H}_{\alpha\beta}$. This will lead to differences with the real position of the rotor that are unacceptable for good control of the PMSM. So, to keep the fault relatively small at low speeds, the bandwidth of the VTO cannot be chosen too high. When the bandwidth of VTO is chosen too low, it cannot react to sudden changes in speed and T_{em}. A solution for this problem is suggested in subsection 2.3.3.

It is not useful to choose a bandwidth larger than the bandwidth of the speed-control loop. It only leads to losses and does not improve the dynamic performances of the machine. With the bandwidth of the speed-control loop comes a limit speed ω_{lim}. This limit speed can be determined based on the sample ratio (SR). This is the ratio of the sample speed of the rotor position and the desired bandwidth (B) of the VTO. The sample speed of the rotor position is dependent on the electrical speed of the machine (ω), expressed in radians per second. This gives us for sample ratio SR

$$SR = \frac{N\omega}{2\pi B}. \tag{2.13}$$

Usually, the sample ratio SR is chosen between 8 and 10 [18]. The minimum value possible for the sample ratio (SR) is 2, corresponding with the Nyquist frequency. The lower the value of the sample ratio SR, the more aggressive the PID controller is. Now, an expression for the limit speed ω_{lim} can be found

$$\omega_{lim} = \frac{2\pi B\ SR}{N}. \tag{2.14}$$

Tuning the PID Controller

To tune the PID controller, the discrete-time transfer functions for the controller $C(z)$ and the system $G(z)$ are necessary. Since the controller is a PID controller, it has a standard expression that is the following

$$C(z) = K_i \frac{T_s}{1 - z^{-1}} + K_p + \frac{1 - z^{-1}}{T_s} K_d. \tag{2.15}$$

This equation can be rewritten and this leads to

$$C(z) = \frac{\left(K_i T_s^2 + K_p T_s + K_d\right) z^2 + \left(-K_p T_s - 2K_d\right) z + K_d}{T_s z - T_s}. \tag{2.16}$$

In this equation T_s is the sample period. K_i, K_p and K_d are the gains for the, respectively, integrating, proportional and differential action of the controller. The transfer function of the system is as follows

$$G(z) = \frac{N_p}{\hat{J}} \frac{T_s z^{-1}}{1 - z^{-1}} \frac{T_s}{2} \frac{1 + z^{-1}}{1 - z^{-1}} \tag{2.17}$$

or more compactly formulated

$$G(z) = \frac{N_p T_s^2}{2\hat{J}} \frac{z + 1}{z^2 - 2z + 1}. \tag{2.18}$$

In this equation, N_p is the number of pole pairs of the machine and $\hat{J}$ is an estimate for the inertia of the rotor. As already mentioned, this estimate does not have to be entirely correct. This is because the closed loop calculation of $\hat{\theta}$ converges to an acceptable value anyway.

Tuning a PID controller means choosing the most appropriate value for the gains. It will be the desired bandwidth of the control loop that will determine the values for the gains. To assign a bandwidth to the system, it suffices to determine the eigenfrequencies of the closed-loop system. This system has three poles o_i, $i = 1, 2, 3$ and therefore it has three eigenfrequencies f_i, $i = 1, 2, 3$. The highest eigenfrequency f_1 is chosen to be equal to the bandwidth B. This was calculated based on the limit speed ω_{lim}. The other frequencies f_2 and f_3 are each chosen to be a factor 10 smaller than the previous one. This means that f_1 is 10 times larger than f_2 and 100 times larger than f_3. In that way, the pole o_1 dominates with the bandwidth.

The desired eigenfrequencies of the closed-loop poles are determined. With a root-locus analysis it is now possible to match the correct eigenfrequencies to the closed-loop poles

by changing the position of the open-loop zeros and by changing the value of the total gain. Based on the open-loop zeros and the total gain, values for K_i, K_p and K_d can be calculated. This method is good to see the influence of the position of the open-loop zeros and the total gain on the position of the closed-loop poles. However, usually, an analytical method is used to tune the paramaters of a PID-controller. This is how it was done in this dissertation. For more info about the root-locus method see [12].

The analytical method calculates, based on the eigenfrequencies, directly the values for the parameters K_i, K_p and K_d. It can be shown that if the eigenfrequencies of the three closed-loop poles have to be different, the closed-loop poles have to be real. And, when a pole is on the real axis, a damping factor $\xi = 1$ corresponds to that pole. A general expression for the poles is

$$o_i = e^{-\zeta 2\pi f_i T_s}, i = 1, 2, 3. \tag{2.19}$$

Next, an expression for the characteristic equation $K(z)$ is composed. This is given by

$$K(z) = 1 + C(z)G(z). \tag{2.20a}$$

Substitution of equations (2.16) and (2.18) in equation (2.20a) results in

$$K(z) = 1 + \frac{N_p T_s}{2\hat{J}} \frac{\left(K_i T_s^2 + K_p T_s + K_d\right) z^3 + \left(-K_s + K_i T_b^2\right) z^2 + \left(-K_p T_s - K_d\right) z + K_d}{z^3 - 3z^2 + 3z - 1}. \tag{2.20b}$$

By substitution of the poles o_i in equation (2.20b) one obtains three equations with K_i, K_p and K_d as unknowns. Since there are three different eigenfrequencies, these equations are linearly independent. Therefore a solution can be found for the three unknowns. Solving these equations gives the PID parameters for this system:

$$K_i = \frac{8\hat{J}}{N_p T_s^3} \frac{-o_1 - o_2 - o_3 + o_1 o_2 + o_1 o_3 + o_2 o_3 - o_1 o_2 o_3 + 1}{o_1 + o_2 + o_3 + o_1 o_2 + o_1 o_3 + o_2 o_3 + o_1 o_2 o_3 + 1}; \tag{2.21a}$$

$$K_p = \frac{4\hat{J}}{N_p T_s^2} \frac{o_1 + o_3 + o_2 - 3o_1 o_2 - 3o_1 o_3 - 3o_2 o_3 + 5o_1 o_2 o_3 + 1}{o_1 + o_2 + o_3 + o_1 o_2 + o_1 o_3 + o_2 o_3 + o_1 o_2 o_3 + 1}; \tag{2.21b}$$

$$K_d = \frac{2\hat{J}}{N_p T_s} \frac{o_1 + o_2 + o_3 + o_1 o_2 + o_1 o_3 + o_2 o_3 - 7o_1 o_2 o_3 + 1}{o_1 + o_2 + o_3 + o_1 o_2 + o_1 o_3 + o_2 o_3 + o_1 o_2 o_3 + 1}. \tag{2.21c}$$

Implementation and Simulation

In this dissertation, the PID parameters are determined by the analytical method. As limit speed, the maximal speed of the motor that is used in the test setup is chosen. This speed

is limited to 1500 rpm. The speed is limited in the test setup for safety reasons. This value has to be multiplied by N_p and multiplied with $\frac{2\pi}{60}$ to get ω_{lim}. In this test setup N_p is 8. The number of sections N in which the 360 electrical degrees are subdivided is 6. The sample ratio is chosen to be equal to 8. This gives us a bandwidth $B = \frac{\omega_{lim}\, N}{2\pi BG} =$ 150 Hz. This is eigenfrequency one f_1. The other eigenfrequencies are $f_2 =$ 15 Hz and $f_3 =$ 1.5 Hz. This leads to the PID parameters from Table 2.2. Furthermore the inertia J is estimated to be 0.0351 kg m^2.

Simulations are done for three mechanical speeds of the motor: 500 rpm, 1000 rpm and 1500 rpm. Because the start of a simulation is a step that corresponds with transients, enough time was waited before examining the results. The sample period T_s is 100 μs. Results of the simulations can be seen in Figures 2.6, 2.7 and 2.8. In these figures $\Delta\theta$ is plotted. This is defined as

$$\Delta\theta = \hat{\theta} - \theta. \tag{2.22}$$

Table 2.2: Results of the analytical calculation for the PID parameters

Parameter	Value
K_p	431.9089
K_i	3.6703×10^3
K_d	4.5653

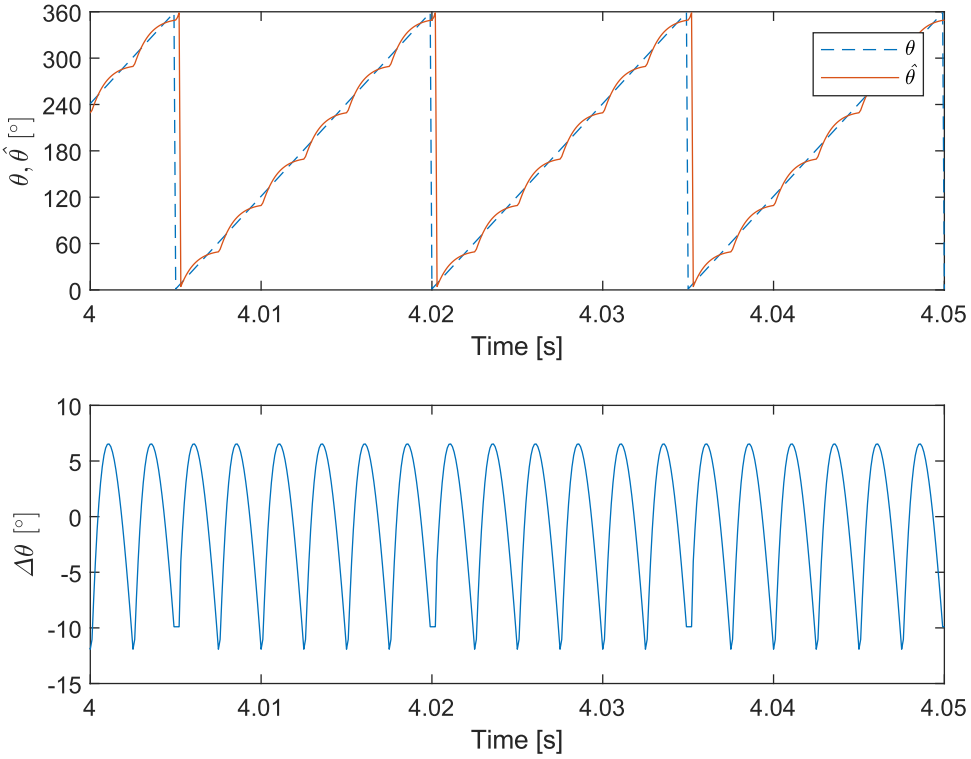


Figure 2.6: Simulation in regime at a mechanical speed of 500 rpm. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

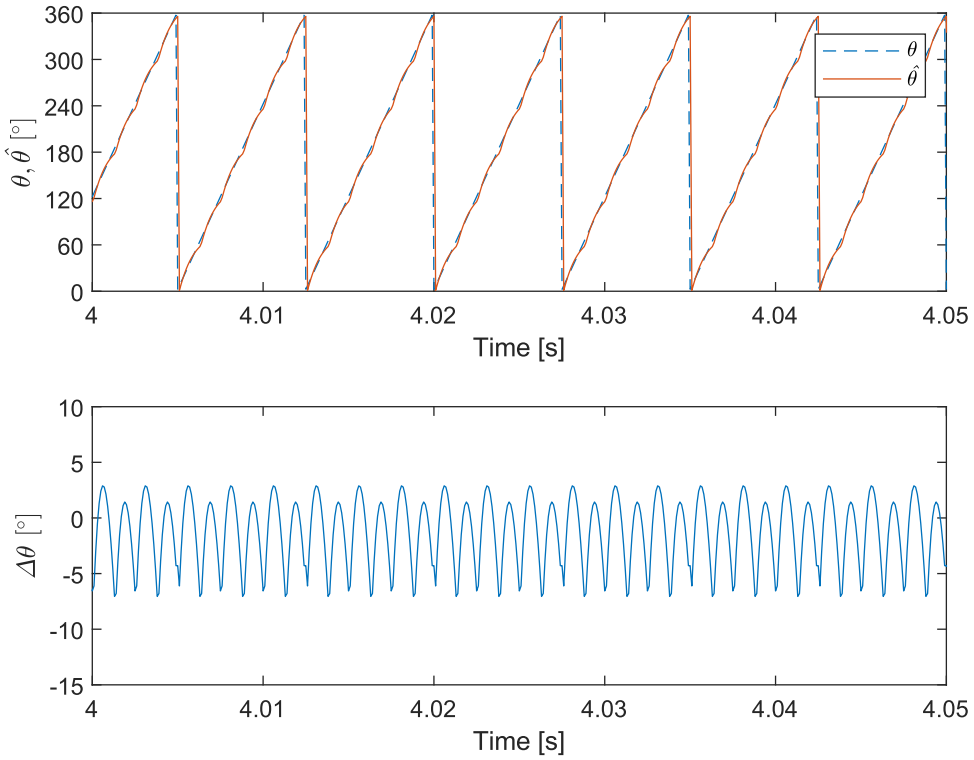


Figure 2.7: Simulation in regime at a mechanical speed of 1000 rpm. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

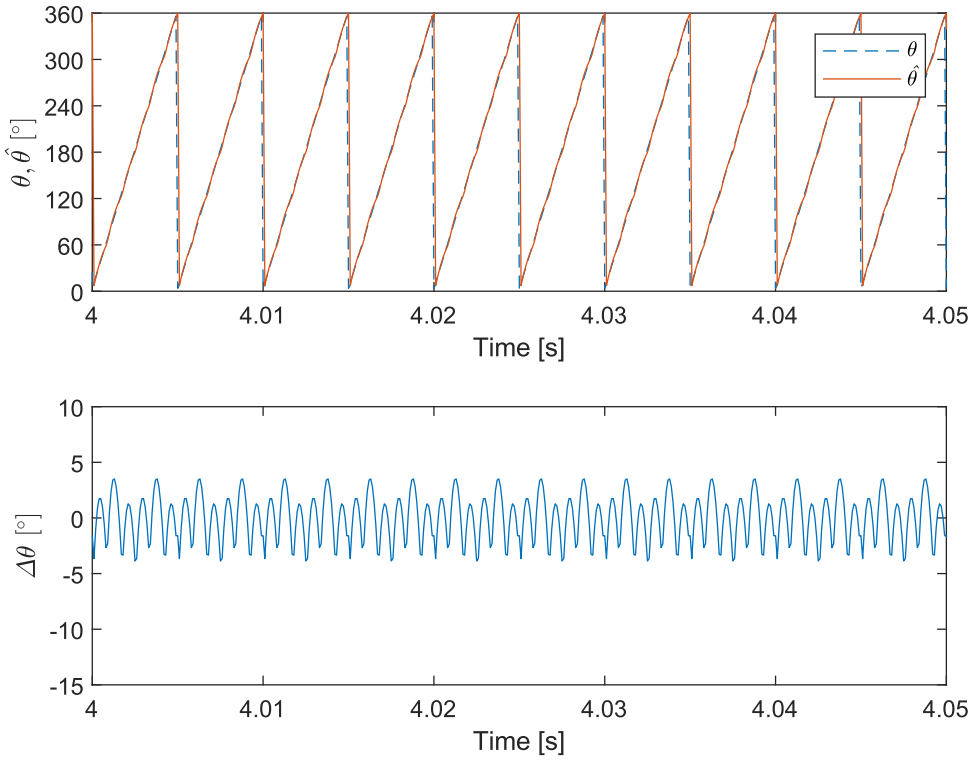


Figure 2.8: Simulation in regime at a mechanical speed of 1500 rpm. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

Conclusion

From these simulations, some conclusions can be drawn. At low speeds, the VTO tracks more the discrete form of $\vec{H}_{\alpha\beta}$. This results in a resolution for $\hat{\theta}$ that is not precise enough to perform vector control. At higher speeds, the estimation is better.

This is caused by the fixed bandwidth of the system and the reaction of the system on higher harmonics of the input $\vec{H}_{\alpha\beta}$. These higher harmonics are the result of the discrete jumps of $\vec{H}_{\alpha\beta}$. For both these problems, a solution is proposed in the following subsections.

2.3.3 Variable gains

The accuracy of the VTO is speed dependent, as was seen in the previous subsection. The PID parameters are tuned based on the desired bandwidth of the system. This needs to be

chosen high enough: in that way disturbances in the steady state regime can be followed precisely. However, if the bandwidth is chosen too high, the VTO tracks the discrete waveform of $\vec{H}_{\alpha\beta}$ almost exactly. This leads to a staircase wave that has a large fault $\Delta\theta$ between θ and $\hat{\theta}$. Therefore a speed dependent bandwidth imposes itself naturally.

In this dissertation, the bandwidth is made speed dependent by linearly varying PID parameters K_i, K_p and K_d with the speed. Based on the estimated speed $\hat{\omega}$, a value for every gain is calculated.

In practice this means that the gains are calculated with equation (2.21) based on the desired eigenfrequencies. These desired eigenfrequencies are derived from the bandwidth that is calculated with limit speed ω_{lim}. To obtain speed dependent gains, those gains are multiplied with a scaling factor K_ω. K_ω is the same for every gain. This scaling factor is linear with the estimated speed $\hat{\omega}$ and is maximally equal to 1.

$$K_\omega = \frac{1}{\omega_{lim}}\hat{\omega} \tag{2.23}$$

When the speed goes to zero, the scaling factor will cause the PID gains to become zero. When this happens, the estimation of θ is only done by the mechanical model and $\hat{J}$. The inertia $\hat{J}$ is only an estimated value and there is no feedback loop anymore that converges to the correct θ. This will lead to a $\Delta\theta$ that is unacceptably large at very low speeds. That is why a minimum value for K_ω is chosen. The minimal value for the scaling factor is usually chosen at 10% or 0.1. Now, there is a feedback loop present at every possible speed. A consequence of this limit for K_ω is that at very low speeds, the estimation for θ will be less good. From now on in this thesis, variable gains will always be used.

Simulation

Now, the same setup for simulations as in subsection (2.3.2) is used. Additionally to this basic setup, scaling factor K_ω is used in the gains. As can be seen in the following figures, the fault $\Delta\theta$ is almost speed-independent.

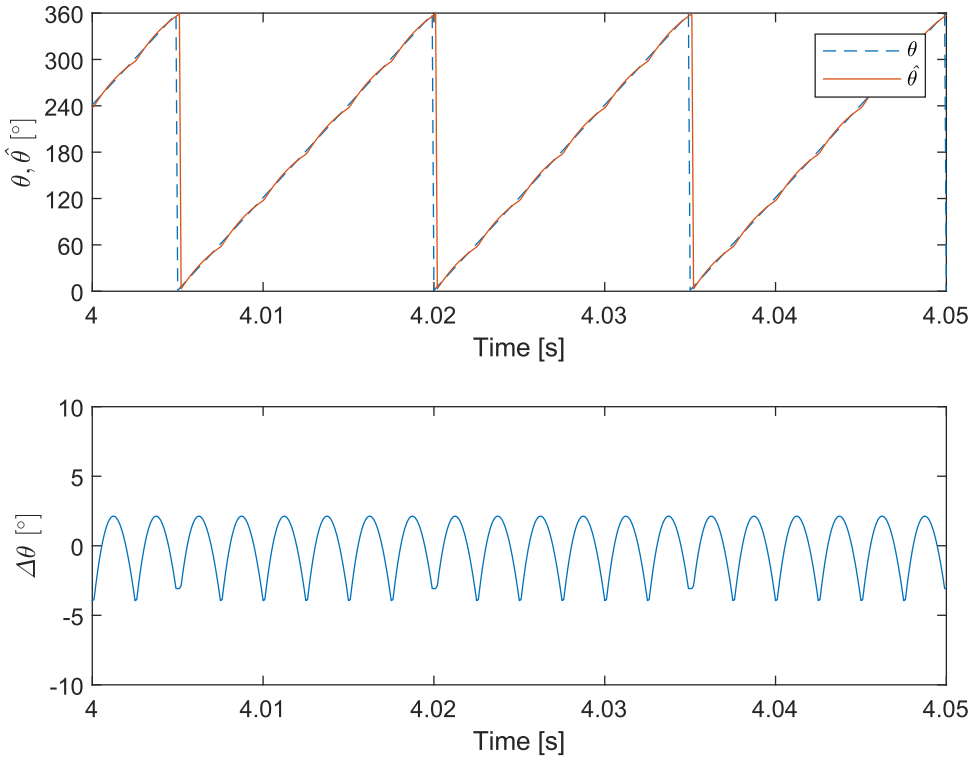


Figure 2.9: Simulation in regime at a mechanical speed of 500 rpm with variable gains. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

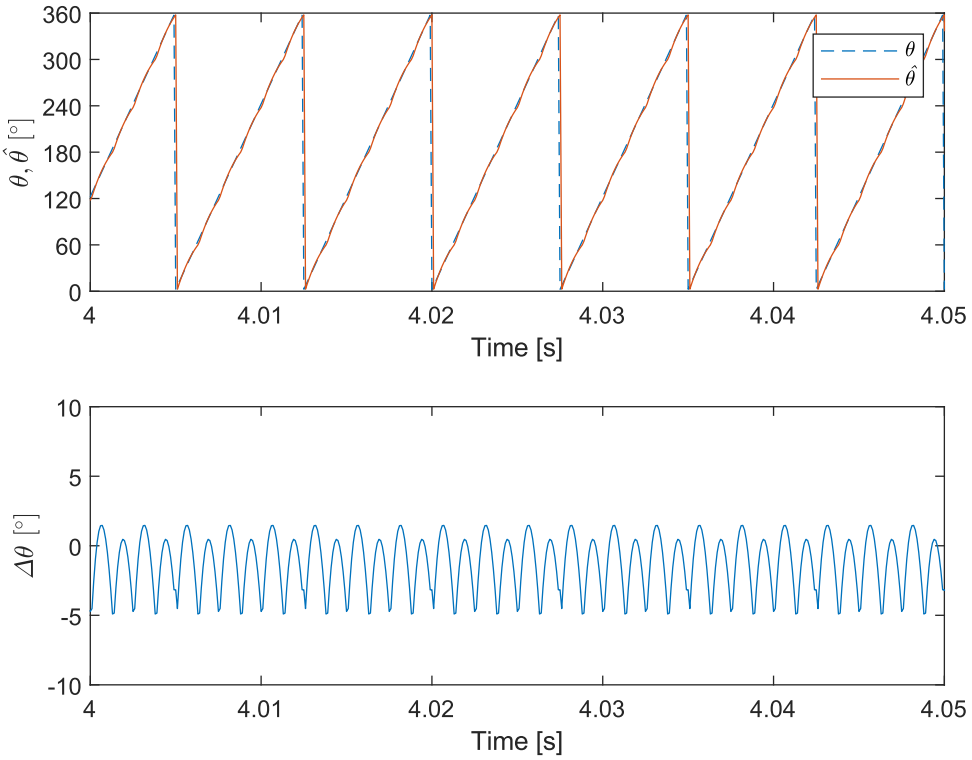


Figure 2.10: Simulation in regime at a mechanical speed of 1000 rpm with variable gains. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

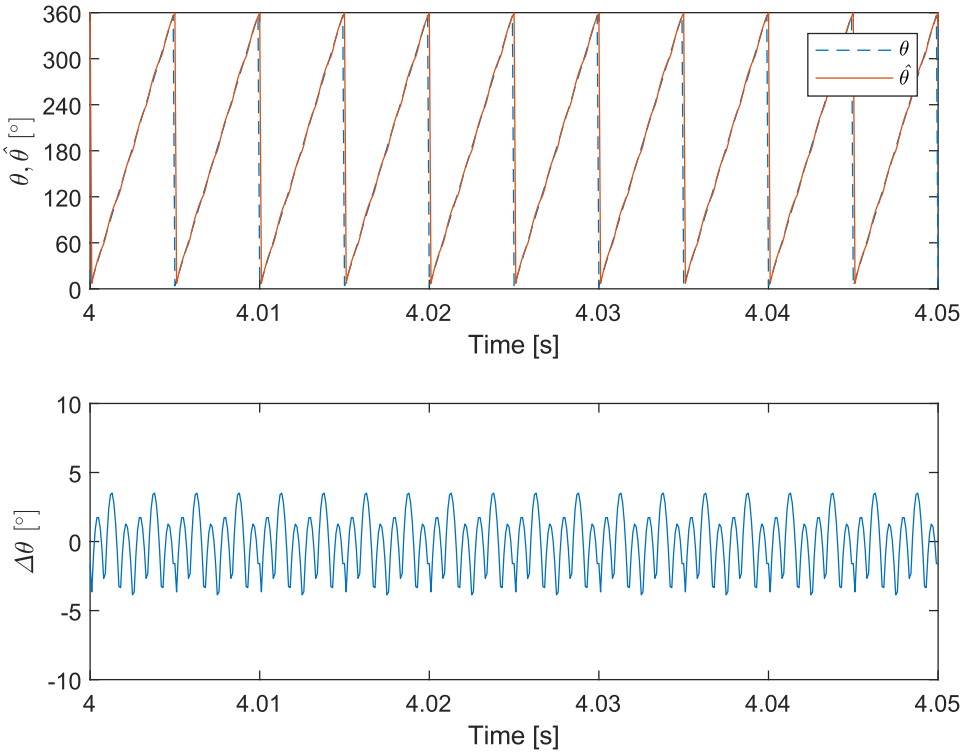


Figure 2.11: Simulation in regime at a mechanical speed of 1500 rpm with variable gains. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

2.3.4 Decoupling Additional Harmonics

Simulations (Figures 2.9, 2.10 and 2.11) show that $\Delta\theta$ is the largest when $\vec{H}_{\alpha\beta}$ jumps to the next sector. The cause of this can be found in the Fourrier series representation of the vector. In subsection (2.3.1), the following expression was found:

$$\begin{aligned}\vec{H}_{\alpha\beta} =& e^{j\theta} - \frac{1}{N-1}e^{-j(N-1)\theta} + \frac{1}{N+1}e^{j(N+1)\theta} \\ & - \frac{1}{2N-1}e^{-j(2N-1)\theta} + \frac{1}{2N+1}e^{j(2N+1)\theta} + \dots .\end{aligned} \tag{2.24}$$

The first term in this expression $e^{j\theta}$ corresponds with the real angle of $\vec{H}$. This term is called the fundamental and is denoted by $\vec{H}_{\alpha\beta,1}$. The other harmonics, denoted as $\vec{H}_{\alpha\beta,h}$, cause the discrete character of $\vec{H}_{\alpha\beta}$. Since only the real angle θ is of interest, it is preferable

to only use $\vec{H}_{\alpha\beta,1}$ as input of the VTO. Then, it is not possible anymore to see a difference between $\hat{\theta}$ and θ when $\vec{H}$ enters a new sector.

To obtain the fundamental $\vec{H}_{\alpha\beta,1}$, the higher harmonics have to be subtracted from $\vec{H}_{\alpha\beta}$ [18].

$$\begin{aligned} e^{j\theta} = \vec{H}_{\alpha\beta} - \frac{1}{N-1}e^{-j(N-1)\theta} + \frac{1}{N+1}e^{j(N+1)\theta} \\ + \frac{1}{2N-1}e^{-j(2N-1)\theta} - \frac{1}{2N+1}e^{j(2N+1)\theta} - \dots . \end{aligned} \tag{2.25}$$

Every term in equation (2.25) is dependent on θ. This means that subtracting the higher harmonics $\vec{H}_{\alpha\beta,h}$ can be done easily, on the condition that θ is known. When the higher harmonics are subtracted from $\vec{H}_{\alpha\beta}$ only the fundamental $e^{j\theta}$ remains. If the fundamental is the input of the VTO, the bandwidth of the VTO can be made larger. Because now, the discrete structure of $\vec{H}_{\alpha\beta}$ is not followed anymore, only the continuous fundamental $e^{j\theta}$.

Working Principle

As can be seen in Figure 2.12, the additional harmonics are subtracted from $\vec{H}_{\alpha\beta}$ before the vector product. The additional harmonics are determined with $\hat{\theta}$. This leads to an estimation of the harmonic content $\hat{\vec{H}}_{\alpha\beta,h}$

$$\begin{aligned} \hat{\vec{H}}_{\alpha\beta,h} = -\frac{1}{N-1}e^{-j(N-1)\hat{\theta}} + \frac{1}{N+1}e^{j(N+1)\hat{\theta}} \\ - \frac{1}{2N-1}e^{-j(2N-1)\hat{\theta}} + \frac{1}{2N+1}e^{j(2N+1)\hat{\theta}} + \dots . \end{aligned} \tag{2.26}$$

When this estimate of the harmonics is subtracted from $\vec{H}_{\alpha\beta}$, the result is $\hat{\vec{H}}_{\alpha\beta,1}$.

Representation of the Additional Harmonics

The ideal representation of the additional harmonics as it is shown in equation (2.26), leads to a discontinuous waveform. This can be seen in Figure 2.13. Theoretically this gives the most exact estimation $\hat{\vec{H}}_{\alpha\beta,1}$. However, if $\hat{\theta}$ differs from θ at a discontinuous instant in the waveform of $\hat{\vec{H}}_{\alpha\beta,h}$, large differences with $\vec{H}_{\alpha\beta,1}$ occur. To address this problem, another representation of the harmonic content is introduced. Main goal of this representation is to make the discontinuous edges less steep.

To make the waveform continuous, the waveform is filtered. The filter that is chosen takes a moving average. This makes the waveform continuous and the edges less steep. The

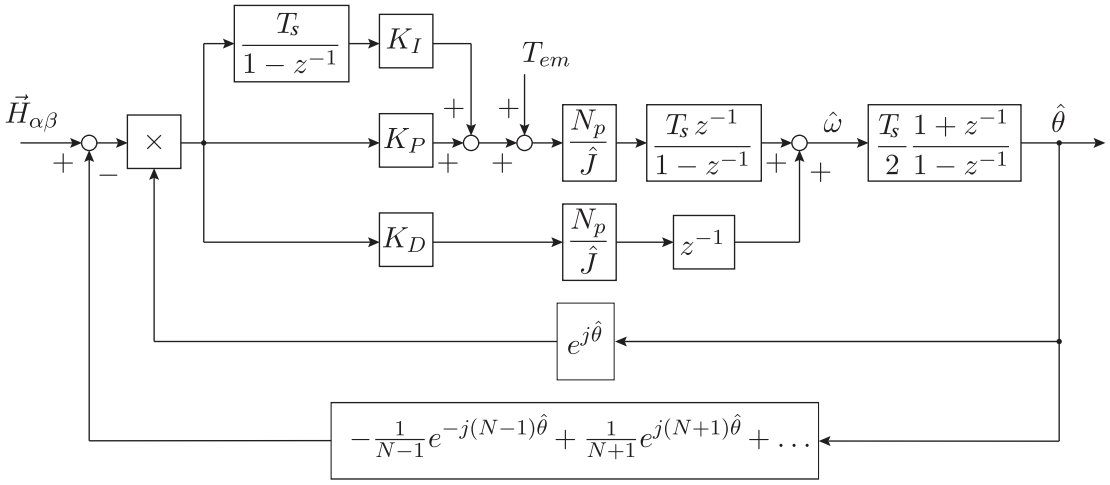


Figure 2.12: Decoupling of the additional harmonics added to the scheme of the vector-tracking observer (VTO) in discrete time.

higher the number of consecutive values that are used to average, the less steep the edges are. The filter in this dissertation averages 5 consecutive values. Another method to obtain less steep edges is to only use the first few harmonics. In this research the first seventy harmonics are used.

Simulation

In the simulations from Figures 2.14, 2.15, 2.16, the same test setup as in subsection 2.3.3 is used. The decoupling of additional harmonics is added. Filtered harmonics up to the order of 70 are subtracted from the input $\vec{H}_{\alpha\beta}$. As one can see, the fault $\Delta\theta$ has become smaller than before. What might seem unexpected is that the estimation $\hat{\theta}$ does not improve with the speed anymore. This could be caused by the peaks in $\vec{H}_{\alpha\beta}$ when the additional harmonics are subtracted, these peaks can be seen in Figure 2.13. At higher speeds, the VTO has a higher bandwidth and is therefore more responsive to these peaks which causes the estimation to not improve at higher speeds in comparison to lower speeds.

2.4 Conclusion

In this chapter the working principle of the low-resolution position estimator is explained. Based on the input of three Hall sensors a vector-tracking observer (VTO) is able to give an estimate for the position of the rotor θ. This estimation is improved by two extra

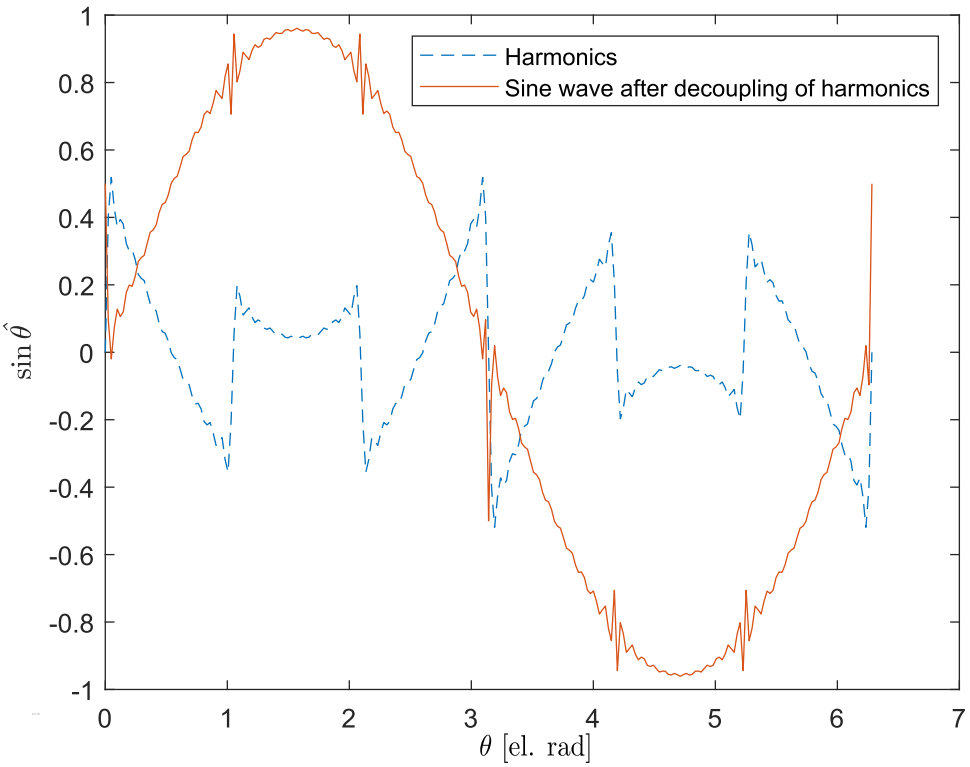


Figure 2.13: Result after subtraction of the additional harmonics. When a discontinuous jump occurs in the harmonics, a distortion occurs in the sine waveform.

features: variable gains and the decoupling of additional harmonics. As can be seen in Figures 2.14, 2.15 and 2.16 the resolution is now quite accurate. This measurement of the position θ based on three Hall sensors will from now on be called the measurement of a single agent. Remember that a modular motor drive consists of several modules. Every module is equipped with a single Hall sensor. An agent is a combination of three modules.

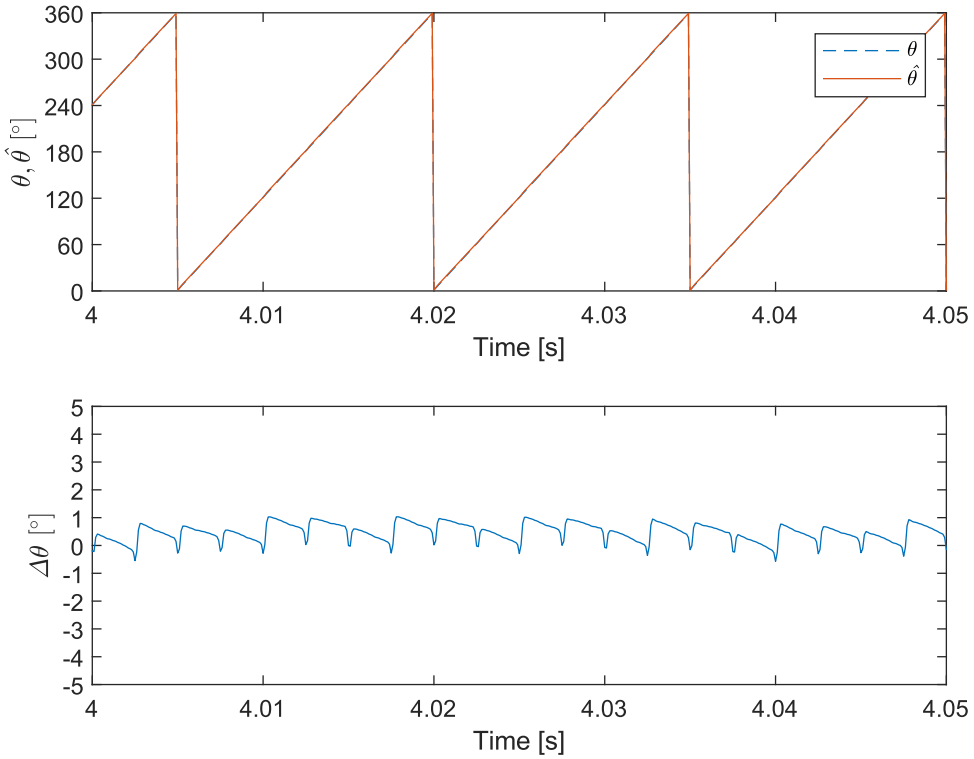


Figure 2.14: Simulation in regime at a mechanical speed of 500 rpm with decoupling of additional harmonics. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

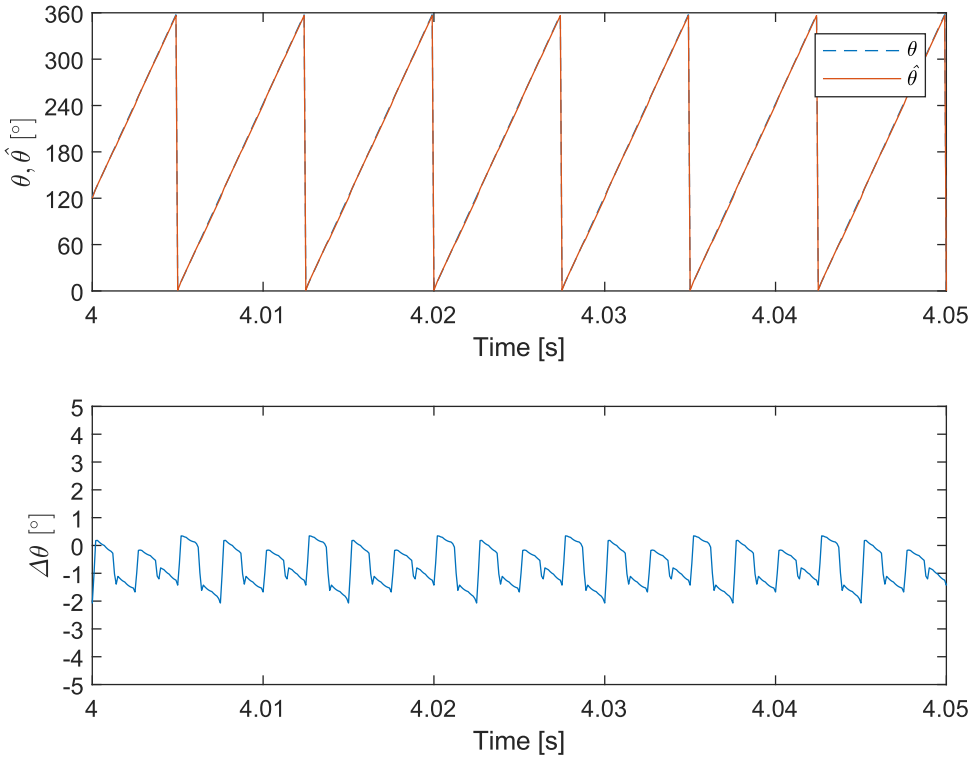


Figure 2.15: Simulation in regime at a mechanical speed of 1000 rpm with decoupling of additional harmonics. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

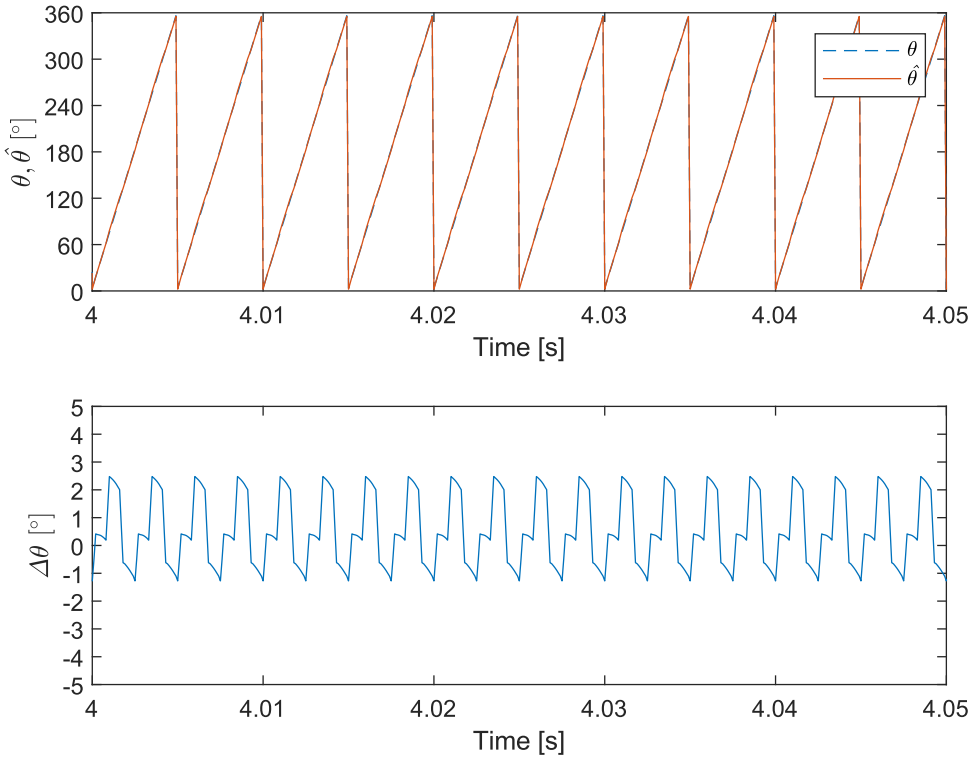


Figure 2.16: Simulation in regime at a mechanical speed of 1500 rpm with decoupling of additional harmonics. The upper figure shows $\hat{\theta}$ in comparison to θ. The lower figure shows the difference $\Delta\theta$ between these two angles.

Chapter 3

Consensus-Based Averaging Algorithm

3.1 Introduction

The goal of this dissertation is to decentralise the position measurement of the rotor in a modular motor drive (MMD). When the position measurement is decentralised, the different modules in the modular drive are not dependent on one single sensor. In order to obtain independent modules, every module will be equipped with a Hall sensor.

In Chapter 2, it is demonstrated that the low-resolution outputs of three binary Hall sensors can be combined into one estimate of the rotor position with high resolution. For this estimate $\hat{\theta}$, three modules have to communicate. In this dissertation, the combination of three modules, each equipped with a Hall sensor, will be called an agent. One agent has one high-resolution position estimate $\hat{\theta}$.

Usually, modular drives consist of more than three modules. The MMD in this dissertation consists of fifteen modules. This means that there are five agents in this drive. Each agent has its own estimate of θ. The notation for such an estimate is $\hat{\theta}_i$ with $i \in 1,...,n$. n being the number of agents in the modular drive. The algorithm, that is proposed in this chapter, improves the position estimate for every agent. This is done by estimating the average of the estimates of the five agents $\bar{\hat{\theta}}$. The hypothesis behind this is that all the agents estimate θ, and that averaging these estimates should improve the position measurement.

$$\bar{\hat{\theta}} = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_i \tag{3.1}$$

3.1.1 Communication Between the Agents

The estimation of the average is not executed by a single calculation unit. This would make the position measurement of the rotor centralised while it is the goal of this dissertation to decentralise the position measurement of the rotor. Therefore, every agent will conduct its own calculations to get an approximation of the average. A major difficulty to do this, is that every agent only communicates with its two closest neighbours. In Figure 3.1, the communication possibilities between the agents are shown for the example of five agents. In section 3.2 this problem of calculating $\bar{\hat{\theta}}$ with limited communication is discussed.

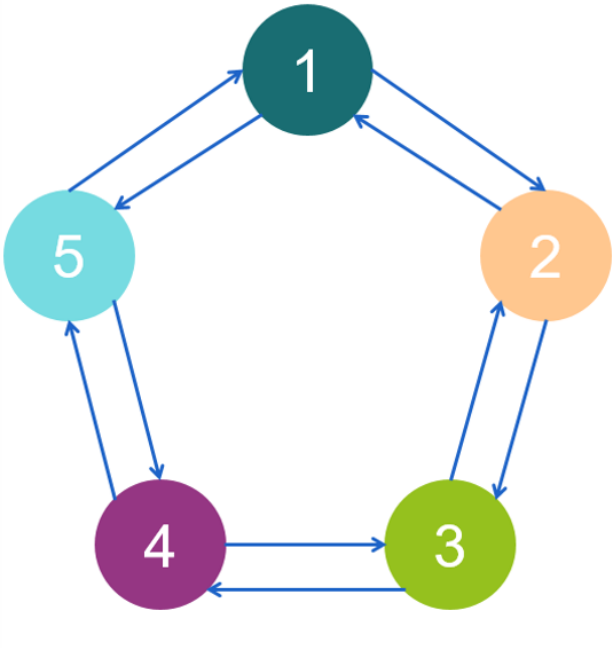


Figure 3.1: Visual representation of the communication between five agents. Every agent only communicates with its two closest neighbours.

3.1.2 The goal

In a modular drive there are n estimations of the rotor position θ. This gives an opportunity to improve these estimations. The goal of this chapter is to do this. The following constraints have to be taken into account: all the calculations are done in a decentralised way and there is limited communication between the agents.

3.2 Converging Dynamic Average Consensus Algorithms

The agents in a modular drive can be seen as a sensor network. With limited communication in the network, an average of the output of all the sensors has to be calculated. The output of an agent is a time-varying signal. The "dynamic average consensus problem" is that the multiagent network has to compute the average of the time-varying signals collectively [19]. The challenge is that each agent is only capable of doing local computations and communicating with local neighbours. The time-varying signals from the agents are not continuous (they reset at 2π) and therefore these are not optimal to be averaged. To address this issue, the sine and the cosine of the estimated rotor positions are used in the consensus algorithms. These are continuous functions. Every consensus algorithm in this section has to be executed simultaneously for the sine function as well as for the cosine function.

3.2.1 Static Average Consensus Algorithms

Algorithms to solve static average consensus problems are extensively documented in literature. In the static average consensus problem, the multiagent network has to determine the average of the constant outputs of the agents. These outputs do not vary in time and therefore some iterations are allowed in order to arrive at the average. The outputs of the agents in this research ($\hat{\theta}_i$) vary in time as the rotor rotates. If a static average consensus algorithm would be used to calculate the dynamic average consensus problem in a modular drive, the static algorithm should converge in one timestep. That would require very high communication speed because some iterations are needed. Since this is practically not possible, a fault is induced with a magnitude depending on the converging speed of the static algorithm and on how fast the outputs of the agents change.

3.2.2 From Static to Dynamic Average Consensus

Now, some notations are introduced.

- $\mathbf{I}_n$, this is the identity matrix of size n.
- $\mathbf{u}_k$, each element of this nx1 vector represents the output of the corresponding agent at time instant k. $\mathbf{u}_k$ is hence the input of the consensus algorithm. In this dissertation that is $\sin\hat{\theta}_i$ and simultaneously $\cos\hat{\theta}_i$. The average of this input vector is $\mathbf{u}^{avg}$.

- $\mathbf{x}_k$, each element of this nx1 vector represents the estimate of the average by the corresponding agent at time instant k. $\mathbf{x}_k$ is hence the output of the consensus algorithm and is communicated to the neighbours.
- $\mathbf{L}$, this is the Laplacian matrix of the graph that represents the multiagent network.
- ρ, this is the convergence rate of the algorithm.

In Figure 3.2 a block diagram of a static consensus algorithm is displayed. As one can see, the time-invariant input $\mathbf{u}$ of the static consensus algorithm is fed into the system as the initial condition $\mathbf{x}_0$. Equation (3.2a) describes the static algorithm from Figure 3.2, equation (3.2b) also expresses the static algorithm, but in vector- and matrix notation. This algorithm converges in discrete time according to the following theorem [19]: "As $k \to \infty$, every agreement state x_k^i, $i \in \{1, ..., n\}$ of the discrete-time static average consensus algorithm (3.2b) converges to $\mathbf{u}^{avg}$ with an exponential rate no worse than $\rho \in (0, 1)$, provided that the Laplacian matrix satisfies $\rho = \|\mathbf{I}_n - \mathbf{L} - \mathbf{1}_n\mathbf{1}_n^T/n\| < 1$."

$$x_{k+1}^i = x_k^i - \sum_{j=1}^{n} a_{ij}(x_k^i - x_k^j) \tag{3.2a}$$

$$\mathbf{x}_{k+1} = (\mathbf{I}_n - \mathbf{L})\mathbf{x}_k \tag{3.2b}$$

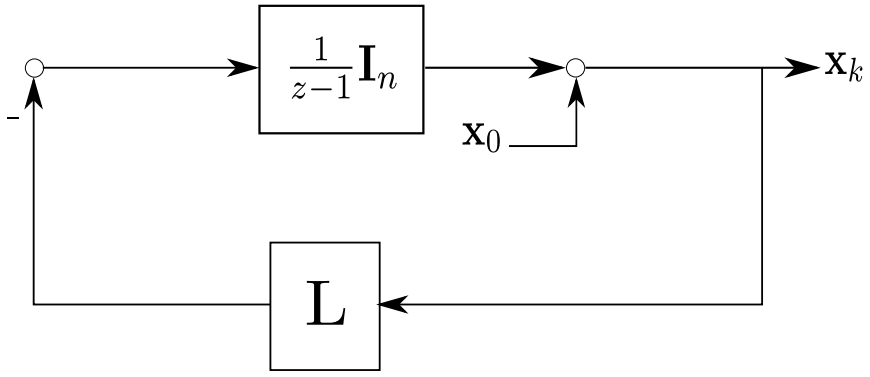


Figure 3.2: A block diagram of a static average consensus algorithm. When the initial conditions for $\mathbf{x}$ are equal to $\mathbf{u}$, this scheme converges to the exact average of the elements of $\mathbf{x}_0$, if $\mathbf{L}$ meets the requirements from the theorem in subsection 3.2.2.

The block diagram of Figure 3.2 can be used as basis for a dynamic average consensus algorithm. For dynamic average consensus, the time-varying signals $\mathbf{u}_k$ need to be continuously

injected into the dynamical system as inputs [19]. This introduces the need for storing an internal state vector $\mathbf{p}_k$. In [19], some features are added to accelerate the convergence and to make the algorithm robust. A robust algorithm is not dependent on any initial values. To obtain this, an additional internal state vector $\mathbf{q}_k$ needs to be stored and $\mathbf{p}_k$ needs to be communicated to the neighbouring agents. The acceleration of the convergence is obtained by storing next to $\mathbf{p}_k$ and $\mathbf{q}_k$, also $\mathbf{p}_{k-1}$ and $\mathbf{q}_{k-1}$. The block diagram of such an accelerated robust dynamic average consensus algorithm can be seen in Figure 3.3. Equation (3.3) expresses how the algorithm is implemented. The difference with equation (3.2a) lies in the storage of two internal states (q^i_{k+1} and p^i_{k+1}) and the communication of p^i_k, this is summarized in Table 3.1.

$$q^i_{k+1} = 2\rho q^i_k - \rho^2 q^i_{k-1} + k_p \sum_{j=1}^{n} a_{ij}((x^i_k - x^j_k) + (p^i_k - p^j_k)) \tag{3.3a}$$

$$p^i_{k+1} = (1+\rho^2)p^i_k - \rho^2 p^i_{k-1} + k_i \sum_{j=1}^{n} a_{ij}((x^i_k - x^j_k) \tag{3.3b}$$

$$x^i = u^i_k - q^i_k \tag{3.3c}$$

$$p^i_0, q^i_0 \in \mathbb{R}, \quad i \in \{1, ..., n\}$$

Root locus techniques can be used to determine the optimal values for k_p, k_i and ρ. The

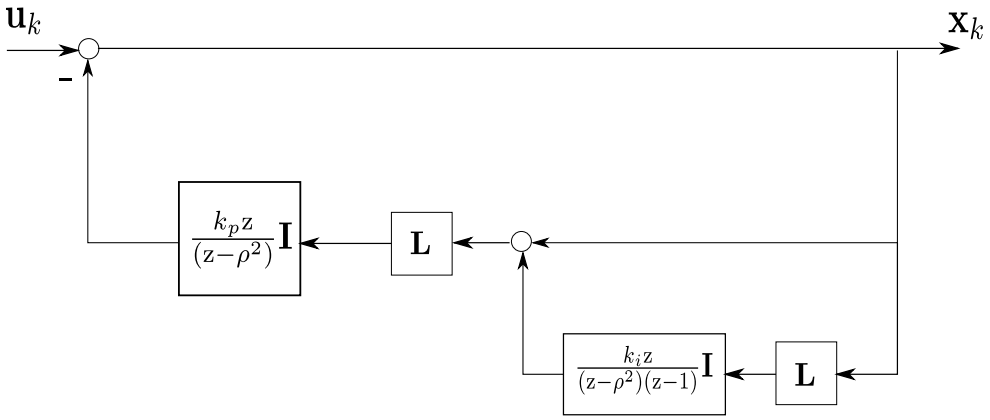


Figure 3.3: A block diagram of an accelerated and robust dynamic average consensus algorithm. This system has two internal states.

system is designed to minimize ρ. The root locus analysis is done based on the eigenvalues λ_i ($i \in \{1, ..., n\}$) of the Laplacian $\mathbf{L}$. Since the block diagram contains two Laplacians $\mathbf{L}$, the root locus analysis is not linear. This complicates the process, but nevertheless a

Table 3.1: Comparison between average consensus algorithms.

Average consensus algorithm	Stored	Communicated
Static	/	x_k^i
Dynamic	p_k^i	x_k^i
Robust + dynamic	p_k^i , q_k^i	x_k^i , p_k^i
Accelerated + robust + dynamic	p_k^i , q_k^i , p_{k-1}^i , q_{k-1}^i	x_k^i , p_k^i

solution can be found [20]. The optimal parameters are given in (3.4). In these equations, λ_r=λ_2/λ_n with λ_n the largest eigenvalue of $\mathbf{L}$ and λ_2 the second smallest eigenvalue of $\mathbf{L}$.

$$\rho = \begin{cases} \frac{6-2\sqrt{1-\lambda_r}+\lambda_r-4\sqrt{2-2\sqrt{1-\lambda_r}+\lambda_r}}{2+2\sqrt{1-\lambda_r}-\lambda_r}, & 0 < \lambda_r \leq 2(\sqrt{2}-1) \\ \frac{-3-2\sqrt{1-\lambda_r}+\lambda_r+2\sqrt{2+2\sqrt{1-\lambda_r}-\lambda_r}}{-1-2\sqrt{1-\lambda_r}+\lambda_r}, & 2(\sqrt{2}-1) < \lambda_r \leq 1 \end{cases} \tag{3.4a}$$

$$k_i = \frac{(1-\rho)^2}{\lambda_2} \tag{3.4b}$$

$$k_p = (2+2\sqrt{1-\lambda_r}-\lambda_r)k_i \tag{3.4c}$$

3.2.3 Simulations

The goal of the dynamic average consensus algorithm in this dissertation is to further improve the position estimation of the rotor in a MMD. The agents have their own estimates of the rotor position obtained by the method described in Chapter 2. By averaging all the estimates, it is expected that the rotor position estimate is improved.

In this dissertation, the graph that represents the communication between the agents is in Figure 3.1. The Laplacian of this graph $\mathbf{L}$ is given by:

$$\mathbf{L} = \begin{bmatrix} 1 & -0.5 & 0 & 0 & -0.5 \\ -0.5 & 1 & -0.5 & 0 & 0 \\ 0 & -0.5 & 1 & -0.5 & 0 \\ 0 & 0 & -0.5 & 1 & -0.5 \\ -0.5 & 0 & 0 & -0.5 & 1 \end{bmatrix}. \tag{3.5}$$

The first (or smallest) eigenvalue from $\mathbf{L}$ is 0. That is because $\mathbf{L}$ is the Laplacian of a constant, connected and undirected graph. With the second eigenvalue (λ_2=0.6910) and the largest eigenvalue (λ_n=1.8090), λ_r is calculated to be equal to 0.3810. With the equations from (3.4) this leads to the following ideal parameters:

- $\rho = 0.3794$
- $k_i = 0.5574$
- $k_p = 1.7783$

These parameters, together with the block diagram of Figure 3.3, are used in the simulations. It has to be determined whether the algorithm can follow the variations in $u_k^i = \hat{\theta}_{i,k}$, when the communication frequency is equal to the sample frequency of the position: $f_s = 1/T_s = 10\text{kHz}$.

The MATLAB code for the simulations is included in Appendix A. It is constructed for the application of this dissertation: a MMD with fifteen modules and thus five agents. In the MATLAB code, the input u_i or $\hat{\theta}_i$ for each agent is the summation of two parts as can be seen in Figure 3.4. The first part is for every agent the same sawtooth function θ that rises to 2π and then resets. The second part is a sinusoidal wave with small amplitude, high frequency and a different phase for every agent. This is done to introduce differences between the agents. The phase difference between every agent is chosen as such that averaging all the agents leads to the original sawtooth function or θ.

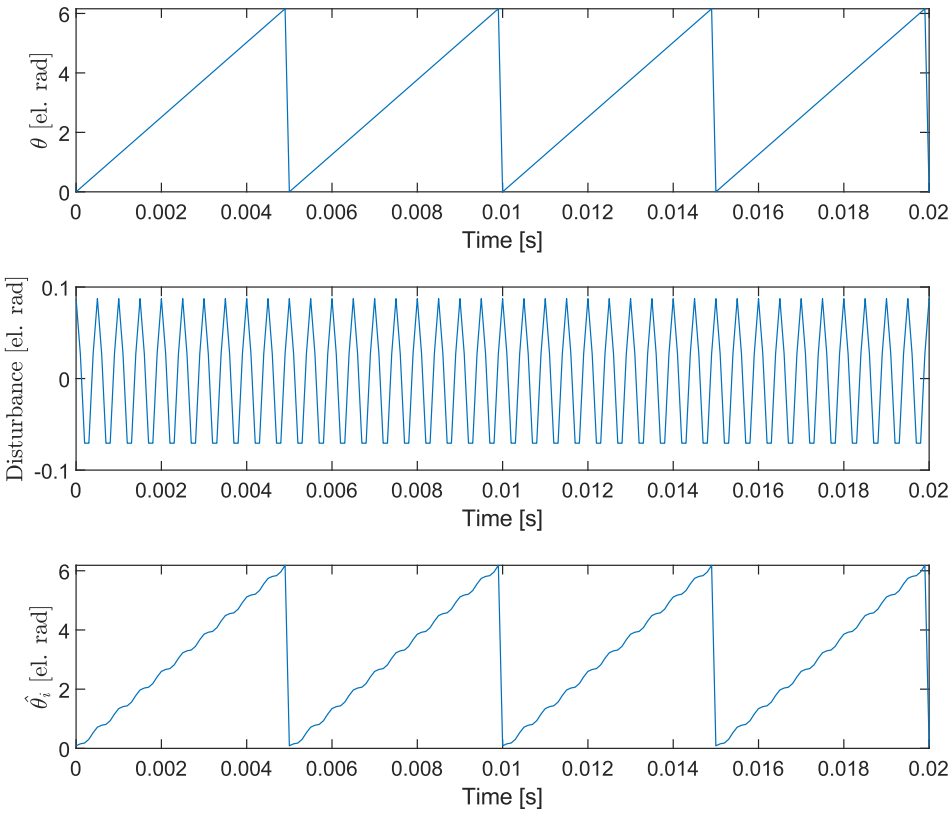


Figure 3.4: Composition of one element of the input vector $\mathbf{u}_k$. The lower graph is the summation of both graphs above.

First, the algorithm is tested for a mechanical speed of 500 rpm. The results can be examined in Figure 3.5. In this figure, the difference with the real average is plotted. With error θ_1 equal to $\theta - \hat{\theta}_1$. Remember that in these simulations, the real average is equal to θ. The amplitude of the error after the dynamic average consensus algorithm has become a lot smaller. This means that the algorithm is able to converge to the average when the rotational speed is 500 rpm.

When the rotational speed is 1000 rpm, the algorithm still converges to the average. The results are slightly better than a single agent. This can be seen in Figure 3.6.

Problems arise when the rotational speed is 1500 rpm. As can be seen in Figure 3.7, the dynamic average consensus algorithm cannot converge to the average anymore. The

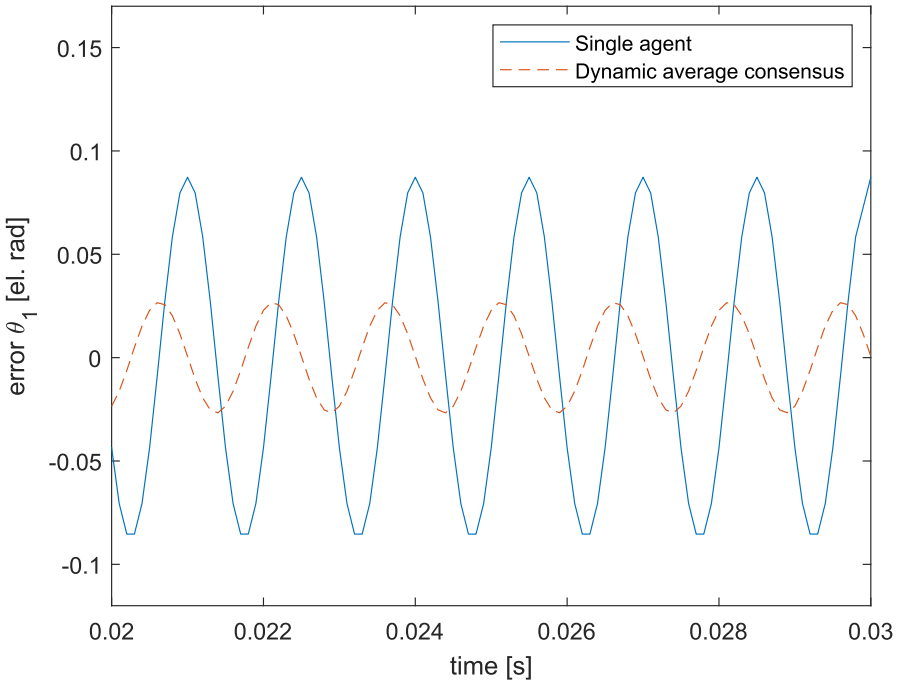


Figure 3.5: Simulation that shows the difference with the real average before and after the dynamic average consensus algorithm. The mechanical speed of the electrical machine is 500 rpm.

convergence is not fast enough for the algorithm to track the signals and calculate their average. The rate of convergence is determined by the network topology. The network topology in this dissertation dictates that every agent only communicates with its neighbours. A solution to this problem could be to raise the communication frequency to 50 kHz. This is five times the sampling frequency. This would bring the bit rate to 1.6 Mb/s[1] Unfortunately this is not feasible with standard CAN communication protocols because the maximum bit rate is 1 Mb/s in these protocols [21].

3.2.4 Conclusion

From Figure 3.7 it can be concluded that a general dynamic average consensus algorithm cannot converge fast enough, and is therefore not suited for the application in this disserta-

[1]The communicated signals are represented by 16 bits and there are five agents each communicating $\mathbf{p}_k$ and $\mathbf{q}_k$. That is 0.8 Mb/s, but this has to be multiplied by 2 because the algorithm is simultaneously calculating for the sine and the cosine.

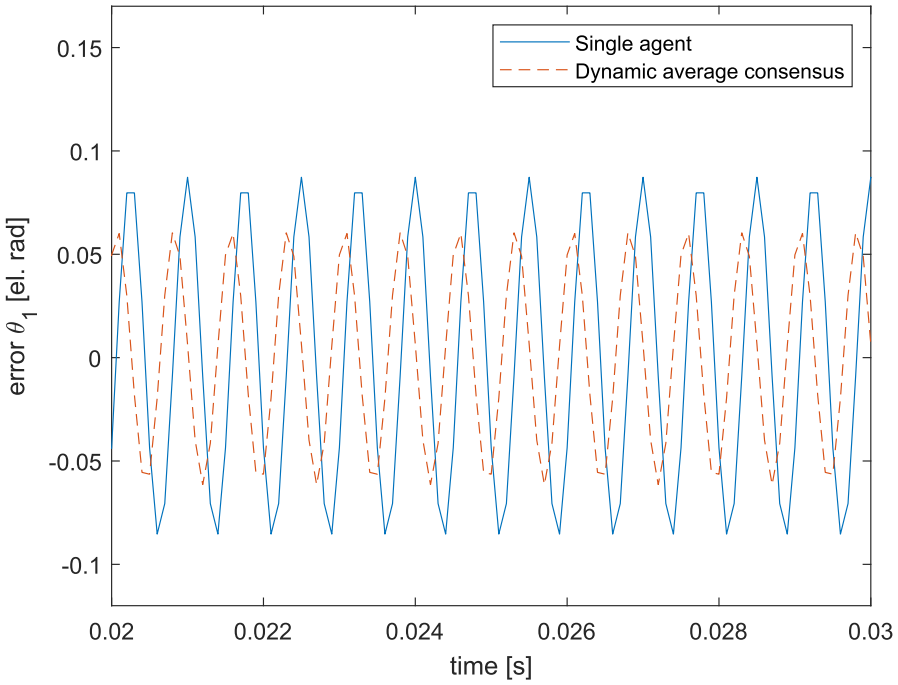


Figure 3.6: Simulation that shows the difference with the real average before and after the dynamic average consensus algorithm. The mechanical speed of the electrical machine is 1000 rpm.

tion. In [22], it is suggested to use the prior knowledge that the signals are sinusoids. This algorithm was tested, but failed to improve the results, again because the communication frequency of 10 kHz is not high enough. Other algorithms with each their specific features, e.g. time-varying agent roles [23], were tested but none of them improved the results at high rotational speeds.

From this literature study it can be concluded that no dynamic average consensus algorithm that converges to the average, converges fast enough under a limited communication frequency. To address this problem, an algorithm has to be found that does not need to converge. In the following section, such an algorithm is presented.

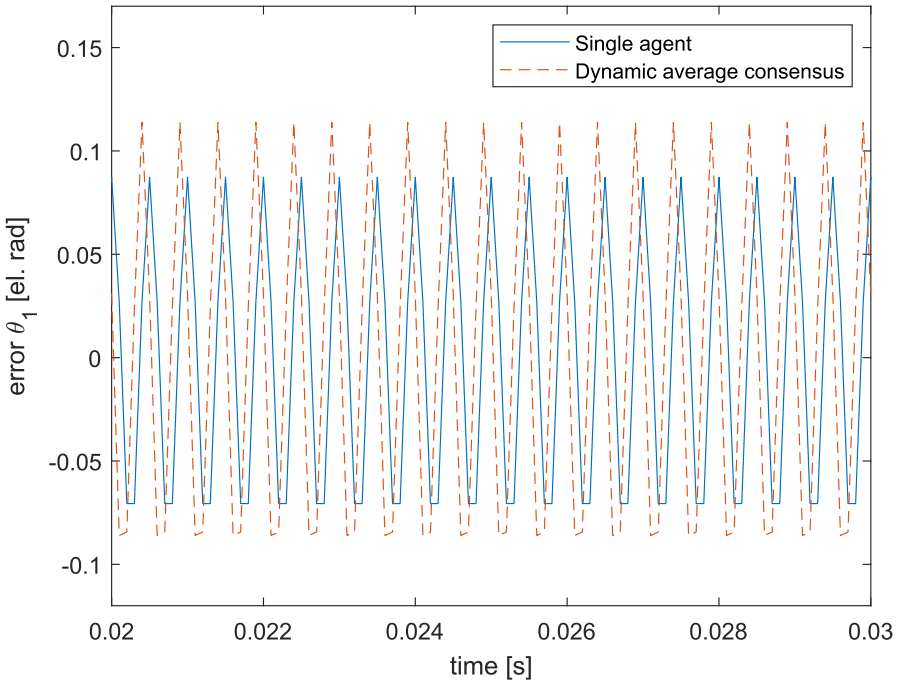


Figure 3.7: Simulation that shows the difference with the real average before and after the dynamic average consensus algorithm. The mechanical speed of the electrical machine is 1500 rpm.

3.3 Consensus-based Averaging Algorithm

In section (3.2), it is shown that standard communication protocols do not allow communication that is fast enough for the above dynamic average consensus algorithms to obtain a good estimate of the average for high speeds. However, the agents track the same rotor position. Therefore, an average of the output of these agents should lead to a better estimation of the rotor position, at every speed. In this section, a new algorithm is proposed that estimates the average of the agents without trying to track the average perfectly and without having to converge.

In this consensus algorithm, the input function is not $\hat{\theta}_i$ but the sine and the cosine of this estimate. This is done to obtain continuous functions. By division of these two, the tangent is found. With the inverse trigonometric function arctangent, an angle for the rotor position is calculated. From here, only the calculation of the average sine is discussed, the calculation of the average cosine is analog and executed simultaneously.

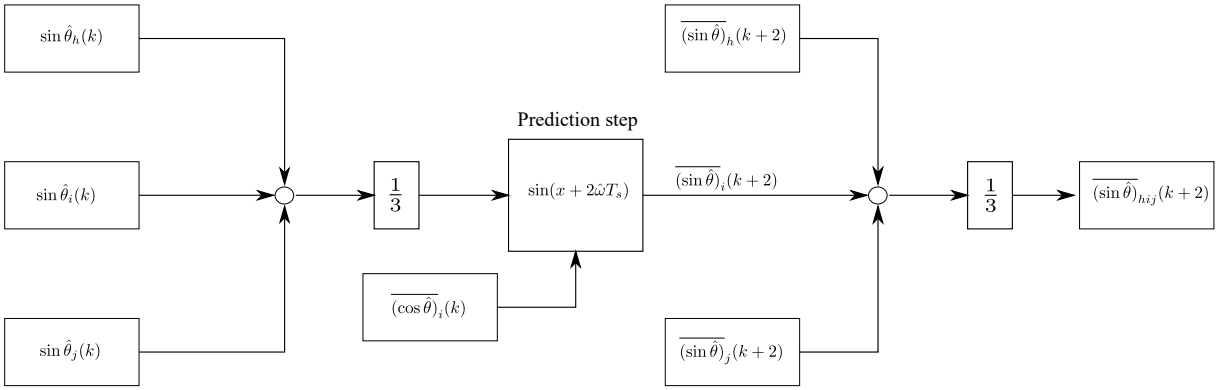


Figure 3.8: Scheme to illustrate the working principle of the consensus-based averaging algorithm. The result is the estimate of the average by agent i. The same scheme is followed to calculate the cosine of the average.

In this consensus algorithm, every agent communicates with its neighbours. This communication occurs in two steps, as can be seen in Figure 3.8. The algorithm is expressed in equation (3.6). Analog equations are conducted for the cosine of the estimated rotor position.

$$\overline{(\sin\hat{\theta})}_i(k) = \frac{1}{3}\Big(\sin\hat{\theta}_h(k) + \sin\hat{\theta}_i(k) + \sin\hat{\theta}_j(k)\Big) \tag{3.6a}$$

$$\overline{(\sin\hat{\theta})}_i(k+2) = \overline{(\sin\hat{\theta})}_i(k)\cos(2\hat{\omega}T_s) + \overline{(\cos\hat{\theta})}_i(k)\sin(2\hat{\omega}T_s) \tag{3.6b}$$

$$\overline{(\sin\hat{\theta})}_{hij}(k+2) = \frac{1}{3}\Big(\overline{(\sin\hat{\theta})}_h(k+2) + \overline{(\sin\hat{\theta})}_i(k+2) + \overline{(\sin\hat{\theta})}_j(k+2)\Big) \tag{3.6c}$$

$\sin\hat{\theta}_i(k)$ is the sine of the estimated rotor position $\hat{\theta}$, estimated by agent i. In the first step, these estimated values are communicated to the neighbours. A summary of what is communicated in this consensus algorithm can be seen in Figure 3.9. Then, each agent i calculates the average of three estimates it has at its disposal: its own estimate, and the estimates from its two closest neighbours h and j. This is done in equation (3.6a). To be able to include information from every agent in the network, two communication steps are needed. The reason for this is explained below. Because algorithm uses two communication steps, a prediction step is needed. In this step, based on the estimated speed $\hat{\omega}$ of the rotor, a prediction of the position of the rotor two timesteps further is made. This speed $\hat{\omega}$ is determined by the VTO before it is integrated to obtain the rotor position. The formula for the prediction step is in equation (3.6b). In equation (3.6b), $\overline{(\cos\hat{\theta})}_i(k)$ from the parallel algorithm for the cosine is needed. $\overline{(\sin\hat{\theta})}_i(k+2)$ is the estimation of the average sine of agent i's and both its neighbours' estimates $\hat{\theta}$ two steps further in time.

Then, this $\overline{(\sin\hat{\theta})}_i(k+2)$ value is communicated to the neighbours, and agent i again computes the average of these values, as can be seen in equation (3.6c). When the average is calculated of these values, not only the neighbours of agent i have an influence on the average. Also the neighbours of the neighbours from agent i have an influence on the average calculated by agent i. This leads to an estimate of the average based on five agents, calculated by agent i.

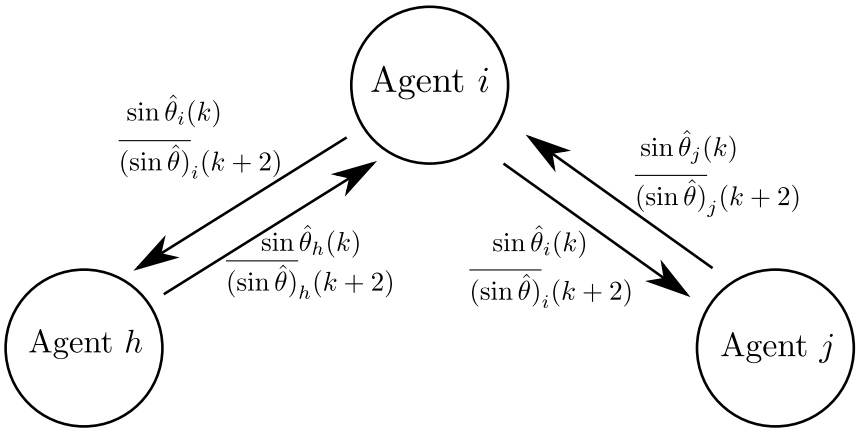


Figure 3.9: Scheme to illustrate the communication between the agents.

In this dissertation, only five agents are used in the modular drive. When the modular drive would consist of more agents, not all agents are included in the average calculated by agent i. In a converging dynamic average consensus algorithm, as discussed in section (3.2), every agent of the system is included. It could be possible to include more agents in one estimation. This would require more communication steps. And therefore a prediction step for more than two timesteps. This would make the algorithm quite dependent on the speed estimated by the VTO: $\hat{\omega}$.

3.4 Simulations

This algorithm was simulated to test whether it leads to better estimations of the rotor position. It is simulated for the same mechanical speeds as in Chapter 2: 500 rpm, 1000 rpm and 1500 rpm. Next to regime simulations, also simulations were performed for two dynamical situations: start-up and reversal of direction.

3.4.1 Regime Simulations

The simulation for 500 rpm can be seen in Figure 3.10. The error with respect to the real rotor position θ ($\Delta\theta = \hat{\theta} - \theta$) is plotted in function of time. As can be seen, the result is very good. The magnitude of the error is clearly reduced in comparison to the estimate of a single agent that is based solely on the VTO described in Chapter 2 of this agent. A numerical analysis of the performance of the consensus algorithm is conducted in section (3.4.4).

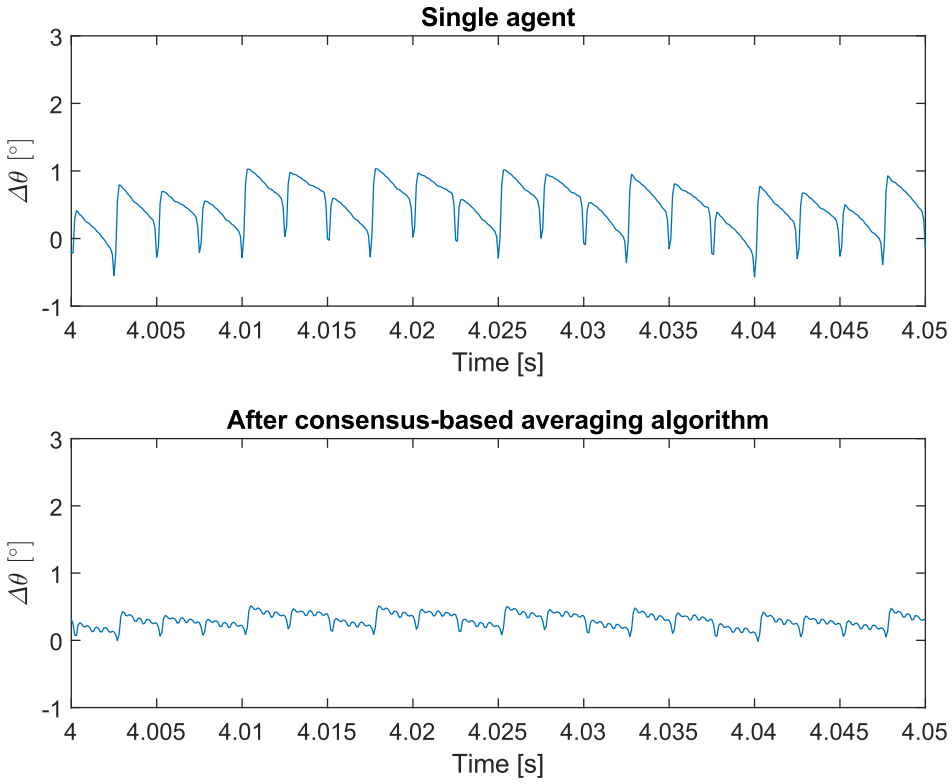


Figure 3.10: Simulation in regime at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

The simulation for 1000 rpm can be seen in Figure 3.11. Again, the deviation with the real rotor position θ is plotted in function of time. Also at this speed, the fault is reduced when using the consensus-based averaging algorithm.

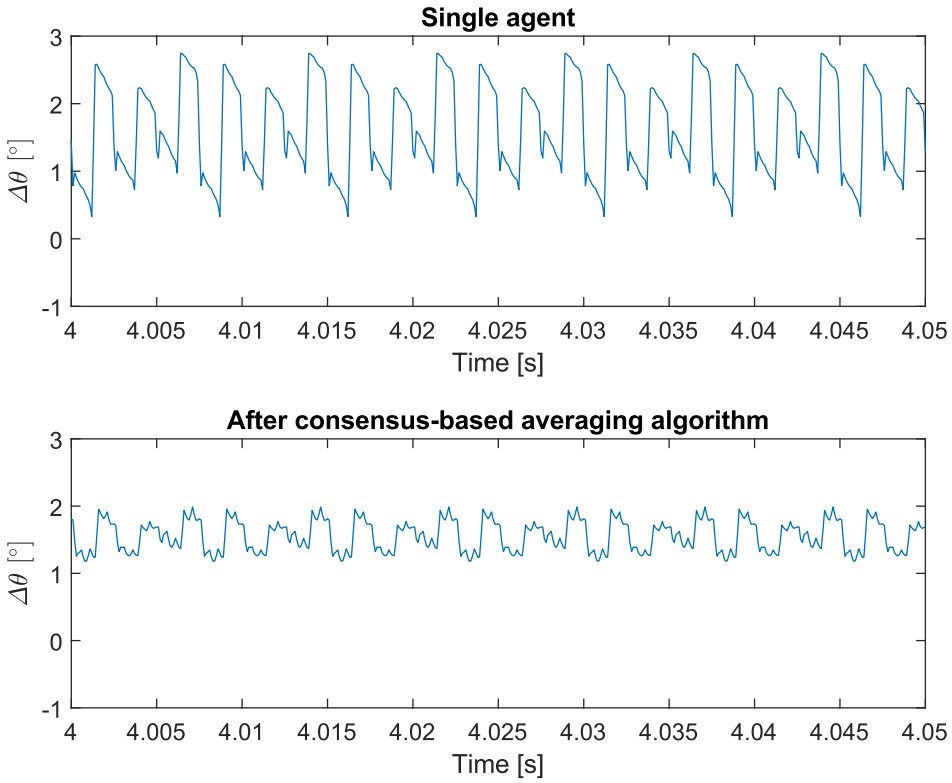


Figure 3.11: Simulation in regime at a mechanical speed of 1000 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

The same conclusions can be drawn for the simulation for 1500 rpm, Figure 3.12. At this speed, the improvement is even better: the magnitude of the error is only one third of the original fault. This shows that the algorithm performs better than the converging algorithm of section (3.2).

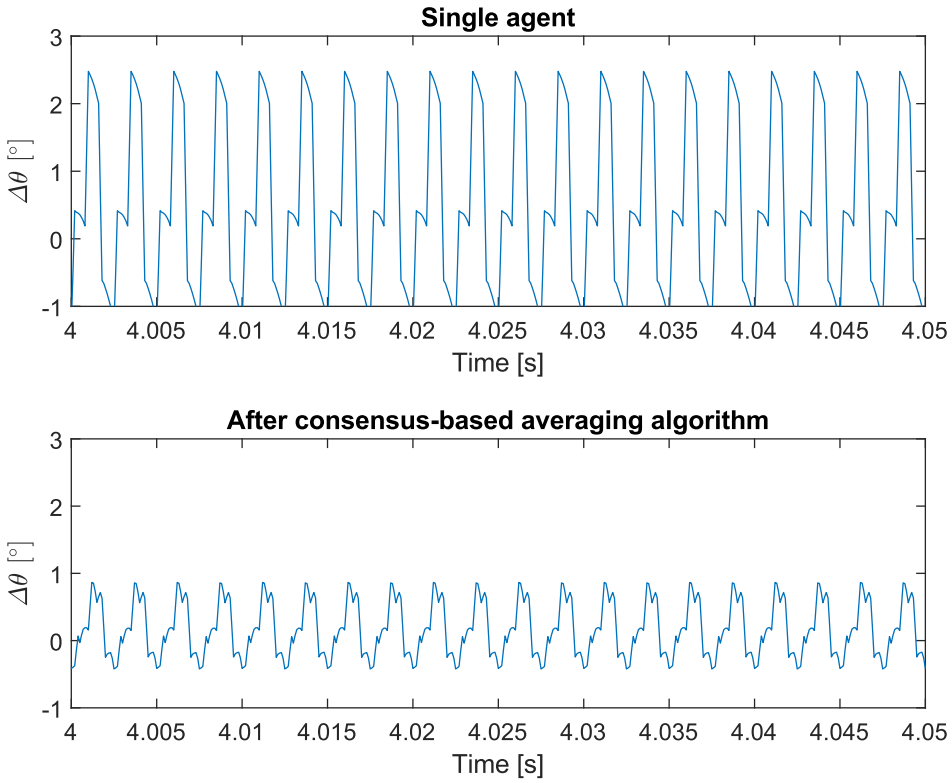


Figure 3.12: Simulation in regime at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

3.4.2 Start-up Simulation

Next, a start-up situation is simulated. The machine starts from zero rotational speed and will accelerate with an acceleration that is defined by a torque of 20 Nm. Together with an inertia J, which is estimated to be 0.0351 kgm^2, this results in an acceleration $\alpha = 570$ rad/s^2. In Figure 3.13 the estimates for the rotor position by one agent and by the average-consensus algorithm are displayed. The consensus algorithm has a bad estimate at the start but quickly performs better than a single agent. This can also be seen in Figure 3.14, there it is even more clear that the algorithm outperforms a single agent.

The initial bad start by the consensus algorithm can be explained. In Simulink, the relative positions of the agents are simulated by adding a constant to the input position of the rotor: $\theta_i = \theta + \gamma_i$, with $\gamma_i = (i-1)\frac{360°}{15}$. Fifteen is the number of modules of the modular drive in

this dissertation. This means that, for example, the input of the VTO of the second agent is shifted 24° relative to the first agent. The input of the VTO of the third agent is shifted 48° relative to the first agent, and so on. Before the average consensus algorithm estimates the average of the outputs of the agents, the outputs of the VTO of the agents ($\hat{\theta}_{i,VTO}$) have to be referred to the same reference position: $\hat{\theta}_i = \hat{\theta}_{i,VTO} - \gamma_i$. Ideally, the output of the VTO of the agents starts at γ_i and then it is shifted to zero degrees by the term $-\gamma_i$. However, it is observed in the simulations that the output of the VTO of every agent starts at zero degrees instead of γ_i degrees. The PID controller in the VTO compensates for the input step function starting from zero degrees, independent of what initial conditions are given. When this output of zero degrees is shifted with $-\gamma_i$, $\hat{\theta}_i$ becomes negative. From this negative value, modulo 360 is taken, and an output of $360° - \gamma_i$ appears. Every agent, except agent 1, has this incorrect input for the consensus algorithm. This explains the initial offset in estimation by the consensus algorithm. The offset disappears when the VTO of every agent has reached $\hat{\theta}_{i,VTO} \approx \gamma_i$. To avoid this in the measurements, a start-up measurement is initiated a few seconds after the initialization of the system, to avoid influence of any initial conditions.

3.4.3 Speed Reversal Simulation

To end the simulations, a change of direction is simulated. A negative acceleration of -570 rad/s^2 is used. First, to slow the machine down to zero rotational speed and then to instantly flip the rotation sense of the rotor. In Figures 3.15 and 3.16 the results of this simulation are shown. In Figure 3.15 a comparison between the waveforms of a single agent and the consensus-based averaging algorithm can be made. The difference in accuracy is less clear with the reversal than with the start-up. In Figure 3.16 this observation is confirmed. The difference in magnitude of $\Delta\theta$ between the single agent and the consensus-based averaging algorithm is minimal.

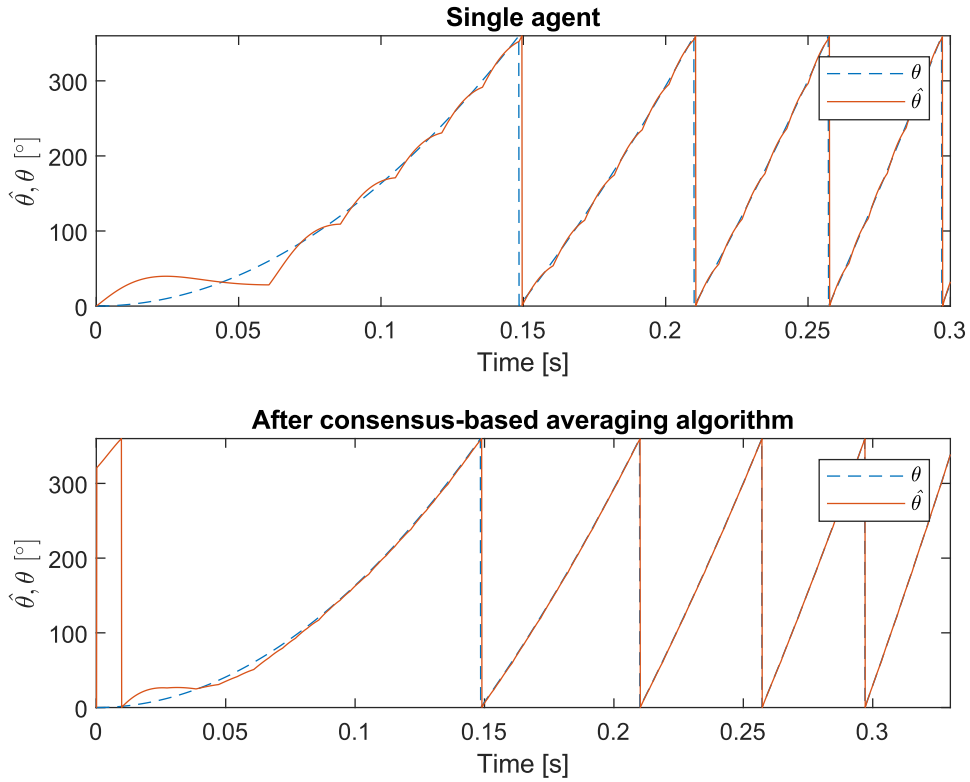


Figure 3.13: Simulation of a start-up for an acceleration of 570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent that is based solely on the VTO of this agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm.

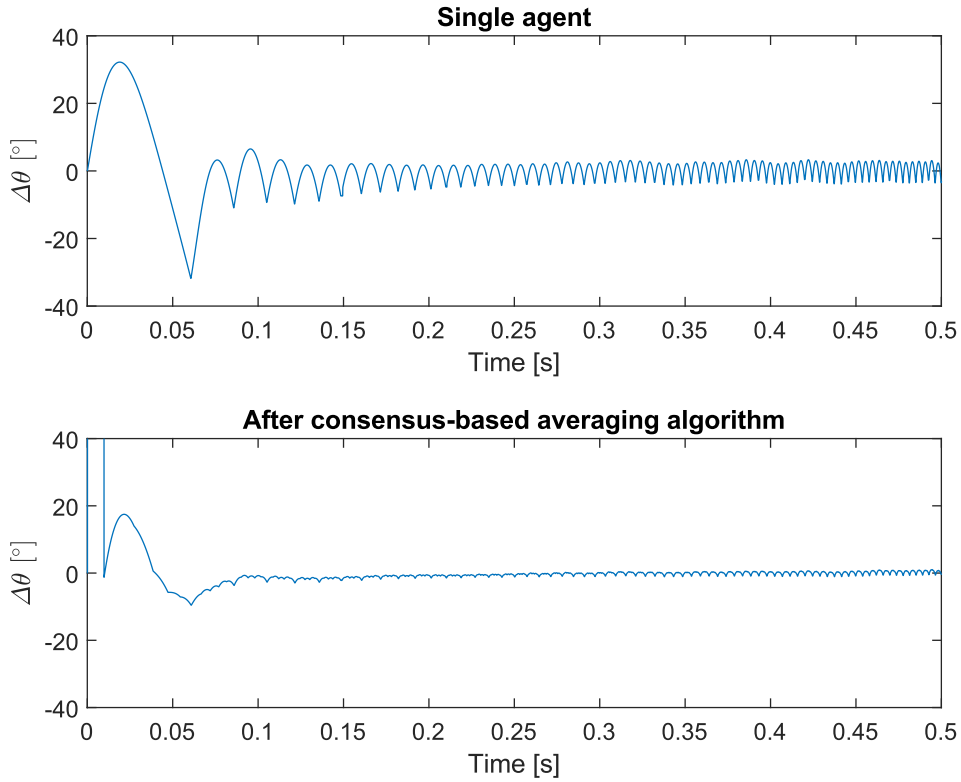


Figure 3.14: Simulation of a start-up for an acceleration of 570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

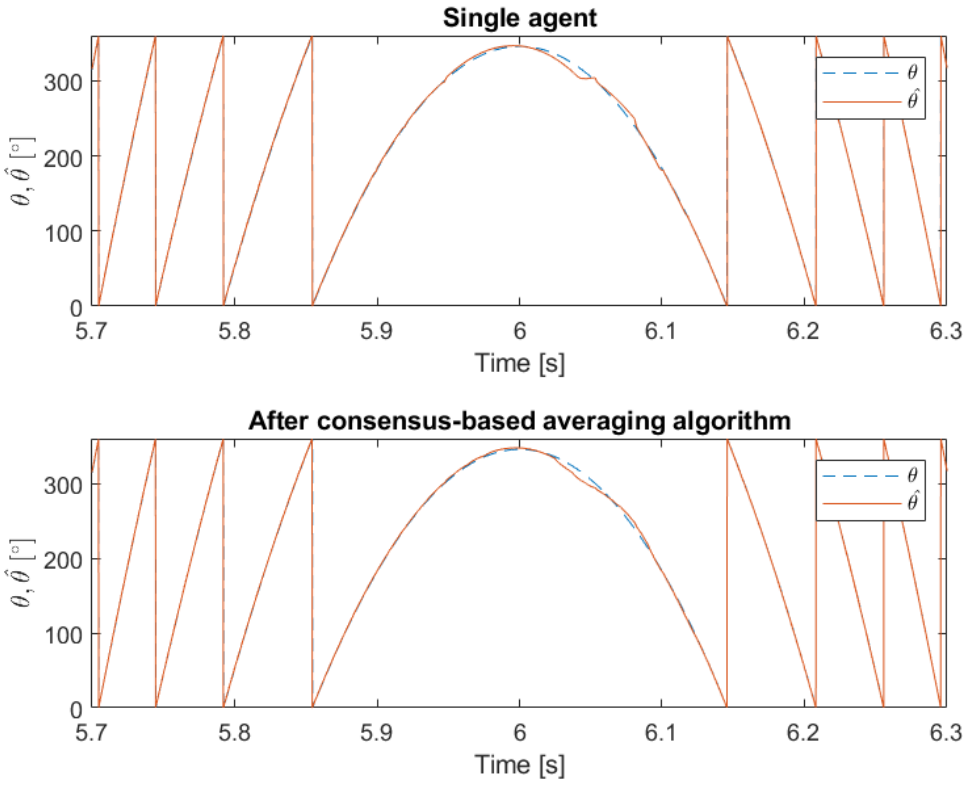


Figure 3.15: Simulation of a reversal for an acceleration of -570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm.

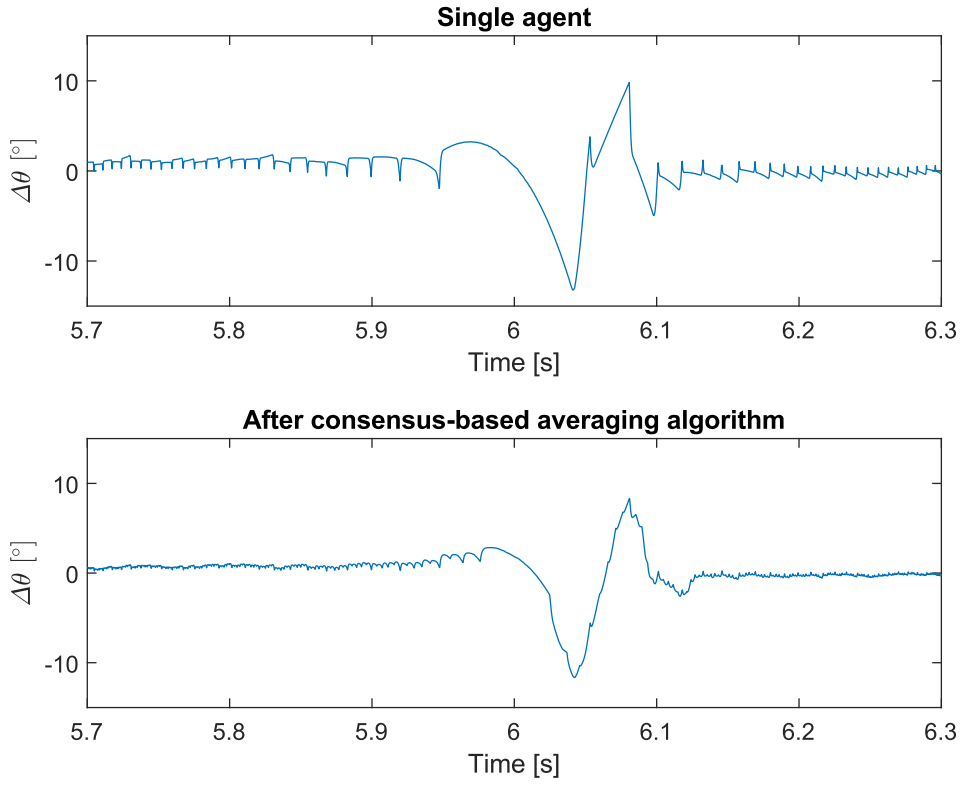


Figure 3.16: Simulation of a reversal for an acceleration of -570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

3.4.4 Numerical Analysis of the Simulations

Although it is very informative to analyse graphs, it can be convenient to compare the different simulations on a numerical basis. In order to do so for these simulations, a measure for the deviation (dev) is set. This measure is the sum of elements over a period of two seconds. Each element in the sum is the absolute value of the difference between the deviation with the real position θ, ($\Delta\theta(t)$), and the average of this deviation ($\overline{\Delta\theta}$) for the whole measurement period of two seconds. This difference is chosen to cancel out a possible error in synchronizing the encoder with the sensors in the measurement setup, which is discussed in the following section. This measure for the deviation is also shown in the following equation [2]:

$$dev = \sum_{t=0\text{s}}^{2\text{s}} |\Delta\theta(t) - \overline{\Delta\theta}|. \tag{3.7}$$

For all the simulations above, dev is calculated and listed in Table 3.2. The average consensus algorithm leads to improved results. dev is reduced with a factor 3 for each regime measurement. The consensus algorithm does not reduce dev for the start-up, in fact dev is now larger. That is due to the incorrect initial conditions as described above. To analyse the real improvement by the consensus algorithm, dev is also calculated for the period 0.1 s to 2.1 s instead of 0 s to 2 s. There, the improvement by the consensus algorithm is clear.

[2]Please notice that this equation is expressed in continuous time functions. This is done to emphasize the time period of two seconds. In practice, with a sample frequency f_s of 10 kHz, this corresponds to an interval length of 20 000 elements.

Table 3.2: Measure for the deviation from the real rotor position for the simulations.

	Deviation *dev* [electr. rad]	
Simulation	Single Agent	After Consensus Algorithm
Regime: 500 rpm	91	30
Regime: 1000 rpm	219	72
Regime: 1500 rpm	372	120
Start-up (0 s - 2 s)	840	1229
Start-up (0.1 s - 2.1 s)	628	202
Reversal	493	310

3.5 Practical Implementation and Measurements

3.5.1 The Electrical Machine

The electrical machine that is used in the setup to test the consensus-based averaging algorithm is an axial flux permanent magnet synchronous machine (AFPMSM). The specifications for this machine are in Table 3.3. The AFPMSM is coupled with an induction machine. When the AFPMSM is working as a motor, the induction machine is a load for the AFPMSM. In this dissertation, the AFPMSM will operate in generator mode. That is because the induction machine is controlled by a frequency converter with a speed-control loop. This enables a straightforward control of speed for the AFPMSM. The AFPMSM is constructed as a modular drive, that means that the stator poles can be controlled independently. Eventually, for this research, every stator pole is equipped with a Hall sensor. Now, for testing and measuring, the sensors are not integrated in the modules as will be explained in the following section.

3.5.2 Construction and Placement of the Sensors

The sensors are not integrated in the stator modules. The integration of the sensors in the modules is beyond the scope of this dissertation. This dissertation focuses on the principles of decentralised position estimation in modular motor drives. When the sensors are not integrated in the AFPMSM, a more simple and accessible setup can be made.

The magnets of the AFPMSM are mounted on the surface of the rotor. The field of these magnets is a block wave over one pole pitch. The binary Hall sensor indicates which part of the magnetic field is passing when the rotor rotates. That is how a single Hall sensor

Table 3.3: Characteristics of the AFPMSM

Property		Unit	Value
Nominal power	P_n	kW	4
Nominal speed	N_n	rpm	2500
Nominal voltage (per module)	V_n	V	30
Nominal torque	T_n	Nm	15
Number of pole pairs	N_p	-	8

has a resolution of 180 electrical degrees, in Figure 2.2 this was shown.

However, if the Hall sensors are not integrated in the stator of the AFPMSM, there is no magnetic field to measure. To address this problem, the combination of the magnets on the rotor and a Hall sensor is recreated by means of a customized disc and an optical switch.

The optical switch used in this setup is an integrated combination of a led and a phototransistor. The optical switch has two legs (see the arrows on Figure 3.17): one leg contains the led and the other contains the base of the phototransistor. The led is aimed at the phototransistor, resulting in a low output voltage of the optical switch. When something impermeable passes in between the led and the phototransistor, the output voltage is high. Fifteen of these optical switches are mounted on a stationary panel, evenly spaced around the rotating shaft.

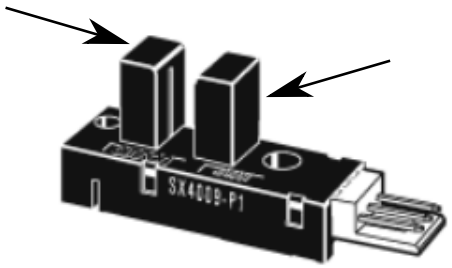


Figure 3.17: Figure of the optical switch. © OMRON Corporation

The customized disc is pictured in Figure 3.18. This disc is mounted on the rotating shaft. In the outer part, openings are provided. This outer part rotates through the legs

of the stationary optical switches. The output of these switches is a square wave: low voltage output when an opening passes, high voltage output when the impermeable part passes. These openings create the same effect on the output of an optical switch as a rotating magnet on a Hall sensor's output, as in Figure 2.2. This disc in particular has eight openings, this is equivalent to a machine with eight pole pairs.

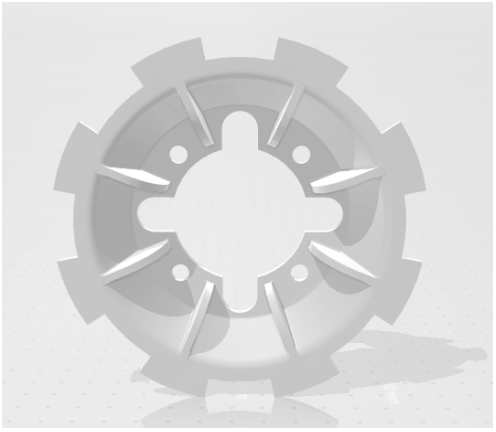


Figure 3.18: 3D model of the disc that is mounted on the rotating shaft. The outer part has eight openings to imitate a machine with eight pole pairs. Design of the disc done by Prof. Dr. Ir. Hendrik Vansompel

3.5.3 Reference Position Measurement

To determine the accuracy of the measurement methods in this dissertation, the position of the rotor has to be known. This position is determined by means of an encoder. The encoder is mounted on the shaft end on the side of the induction machine. The encoder in this setup has 1024 lines on its disc, resulting in a resolution of 4096 pulses per revolution.

3.5.4 Processing Signals and Controlling Speed

The following signals have to be processed: fifteen sensor outputs and the encoder signal. All these signals arrive on a dSPACE MicroLabBox via a D-sub connector. Via the MicroLabBox, which is connected to a computer, the signals can be processed in dSPACE ControlDesk and plotted via MATLAB.

In dSPACE ControlDesk, an sdf file can be uploaded. Such a file is built from a MATLAB Simulink file. An sdf file contains all the outputs and blocks from the MATLAB Simulink

file. Therefore it performs in real time and based on real signals the same calculations as in the simulations from section (3.4).

The output of the encoder Simulink/dSPACE block is in mechanical degrees. This has to be converted to electrical radians for FOC. Therefore, the original signal is multiplied with a factor $8\frac{2\pi}{360}$, since there are eight pole pairs. The modulus 2π of this result is taken. Next, the encoder signal should be at the zero radians crossing at the same moment as the sensors. To obtain this, a constant is added to the encoder signal and the modulus is taken again. This constant is determined during testing: when the position output of the encoder equals zero radians, the position output of one agent is stored. That position is the constant that is added to the encoder signal.

The speed of the induction machine is controlled via the MicroLabBox in combination with ControlDesk. The MicroLabBox is connected with a CAN cable to a Siemens Sinamics frequency converter. This converter controls the speed of the induction machine that is coupled to the AFPMSM. This enables a straightforward control of the speed in real time.

3.5.5 Parameters Used in the Measurements

Parameters like the PID parameters of the VTO, the inertia of the machine, etc. have the same numerical values as for the simulations. All the parameters used in the measurements are summarized in Table 3.4. In Figure 3.19 a picture of the measurement setup is presented.

Table 3.4: Parameters used in the measurements

Parameter	Value
K_p	431.9089
K_i	3.6703×10^3
K_d	4.5653
N_p	8
J	0.0351 kgm^2
T_s	0.0001 s

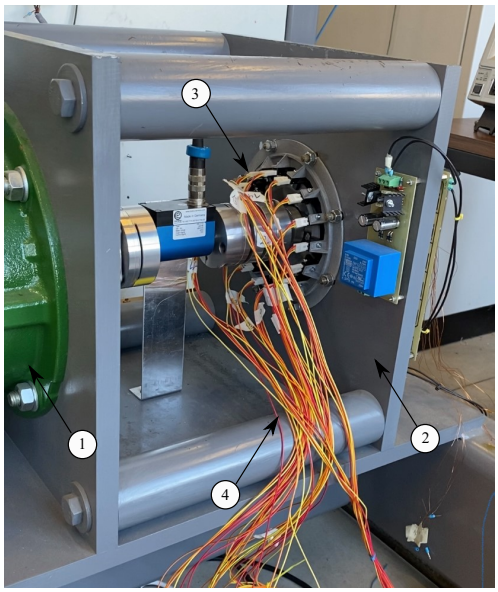


Figure 3.19: Picture of the measurement setup. 1: Induction machine; 2: Panel for sensor mounting; 3: Optical switch; 4: Sensor cables to MicroLabBox

3.5.6 Results of the Measurements

Regime Measurements

The consensus algorithm of Figure 3.8 is tested for three different speeds in regime: 500 rpm, 1000 rpm and 1500 rpm. The results of these measurements are shown in Figures 3.20, 3.21 and 3.22, together with the experimentally obtained position estimates that are based solely on the VTO of this agent, described in Chapter 2. It can be observed that the consensus algorithm leads to a smaller error $\Delta\theta$ in the estimated rotor position $\hat{\theta}$ in comparison with a single agent that is based solely on the VTO from Chapter 2. Remarkable is that the error is generally larger than in the simulations. That is probably caused by misplacement of the sensors and printing errors of the rotating disc. This topic will be further elaborated in Chapter 5. The magnitude of the reduction is also relatively smaller than in the simulations with ideal sensor outputs. However in absolute terms, especially for higher speeds, the decrease in magnitude of $\Delta\theta$ is around two degrees, which is the same as in the simulations.

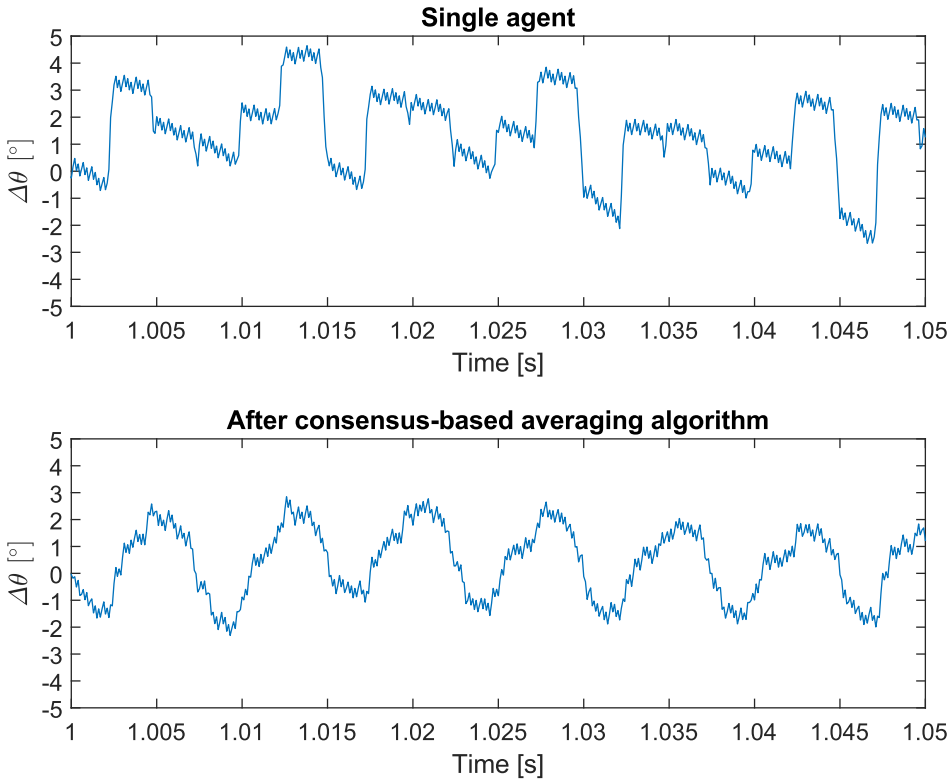


Figure 3.20: Measurement in regime at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

Start-up Measurement

During start-up the gain of the consensus algorithm is less clear. It is certain that start-up is a challenging situation for the vector-tracking observer. This method is based on the frequent discrete jumps from sector to sector, see Table 2.1. When the rotor is at standstill, the error can go up to the width of a full sector. This means, using three sensors for one measurement, an error of 60 electrical degrees. In the measurement of Figures 3.23 and 3.24 the error at standstill is approximately 45 electrical degrees. The start-up was executed with the same acceleration as in the simulations, i.e. an acceleration α of 570 rad/s^2 caused by a torque of 20 Nm. A difference with the simulations is that the initial rotor position is not 0°, that is because it is difficult to obtain an exact 0° initial position in the test setup.

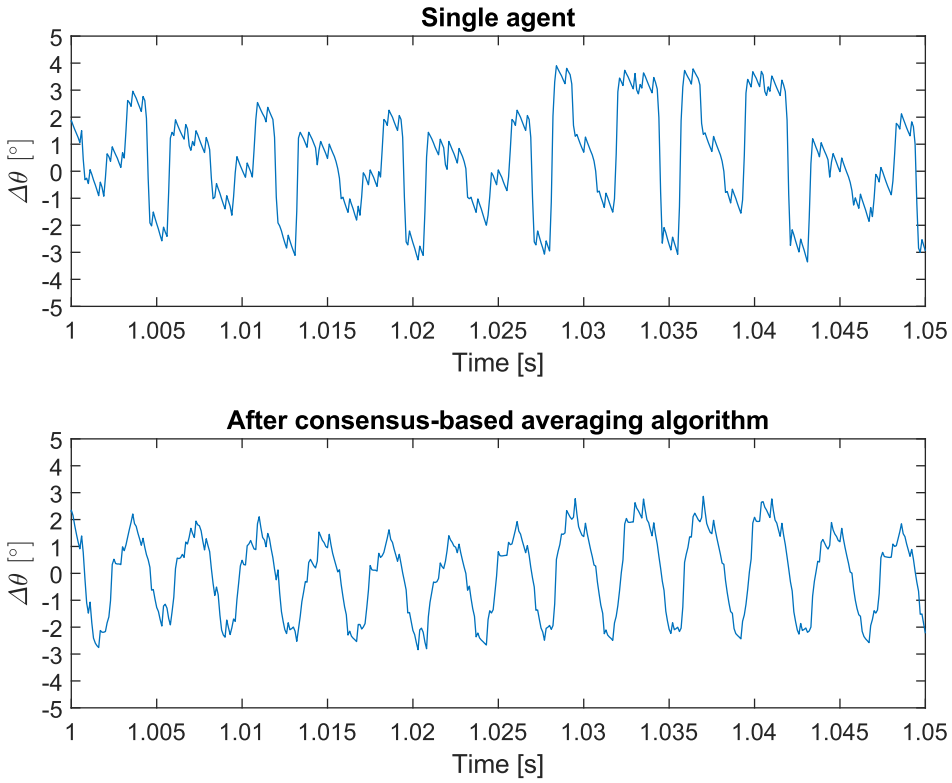


Figure 3.21: Measurement in regime at a mechanical speed of 1000 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

Speed Reversal Measurement

In the measurement of a speed reversal (α=-570 rad/s^2), the improvement by the consensus algorithm is more distinct than in the measurement of the start-up. This can be seen in Figures 3.25 and 3.26. However, the error is still much larger than in the simulations. This is due to the fact that the sensors are not placed entirely exact and to the fact that the disc its openings are not exactly 180 electrical degrees, this topic is further elaborated in Chapter 5.

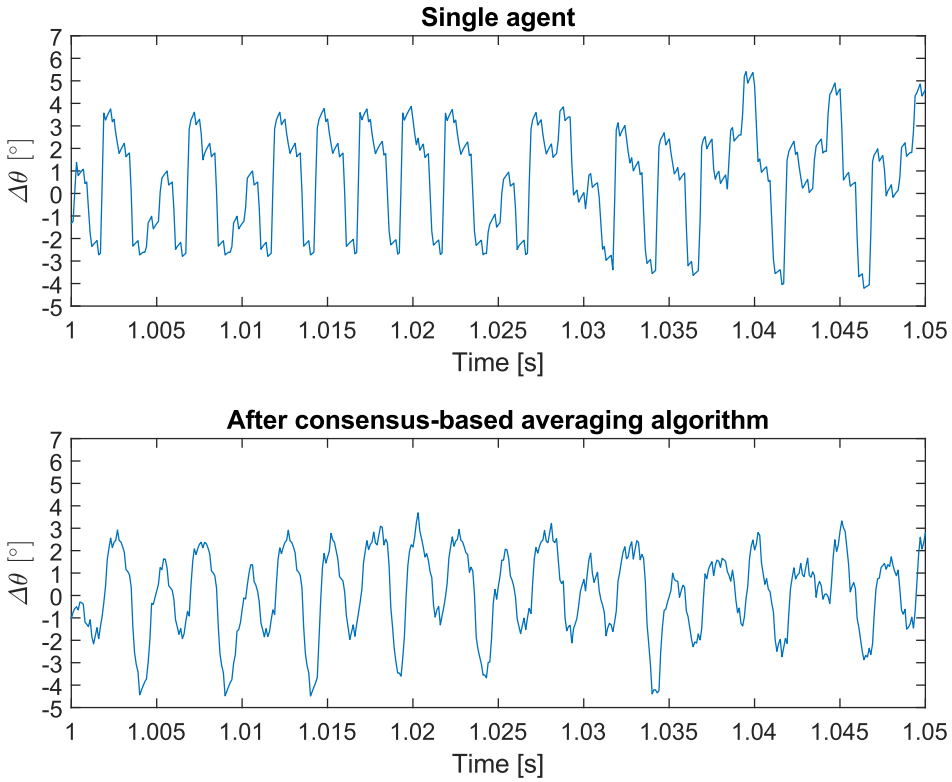


Figure 3.22: Simulation in regime at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

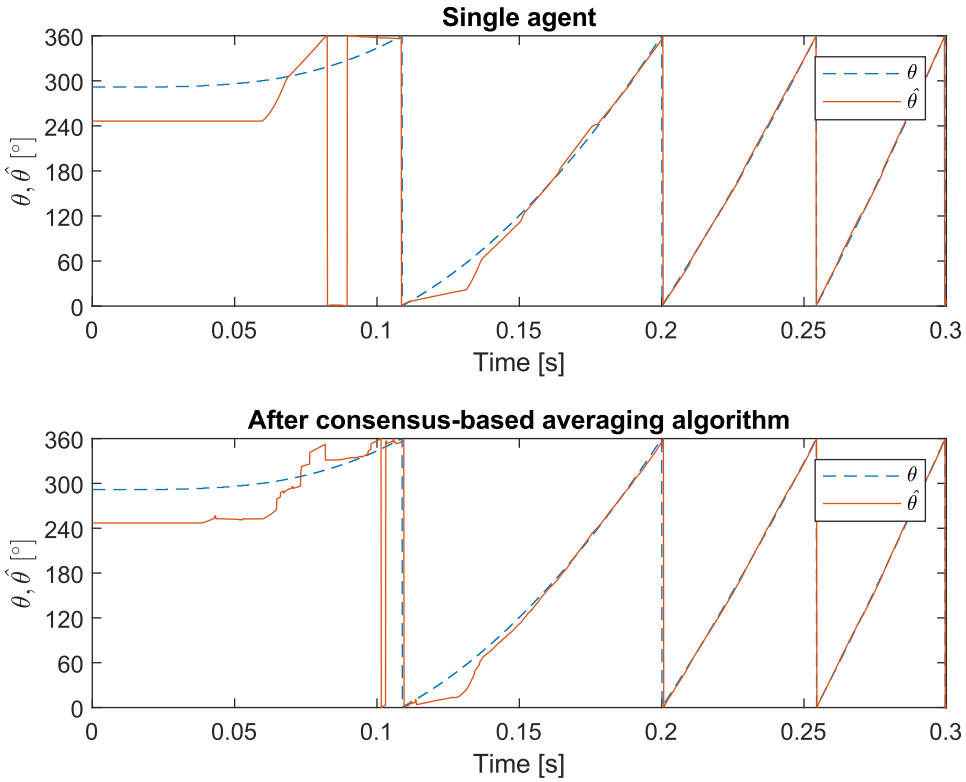


Figure 3.23: Measurement of a start-up with an acceleration of 570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm.

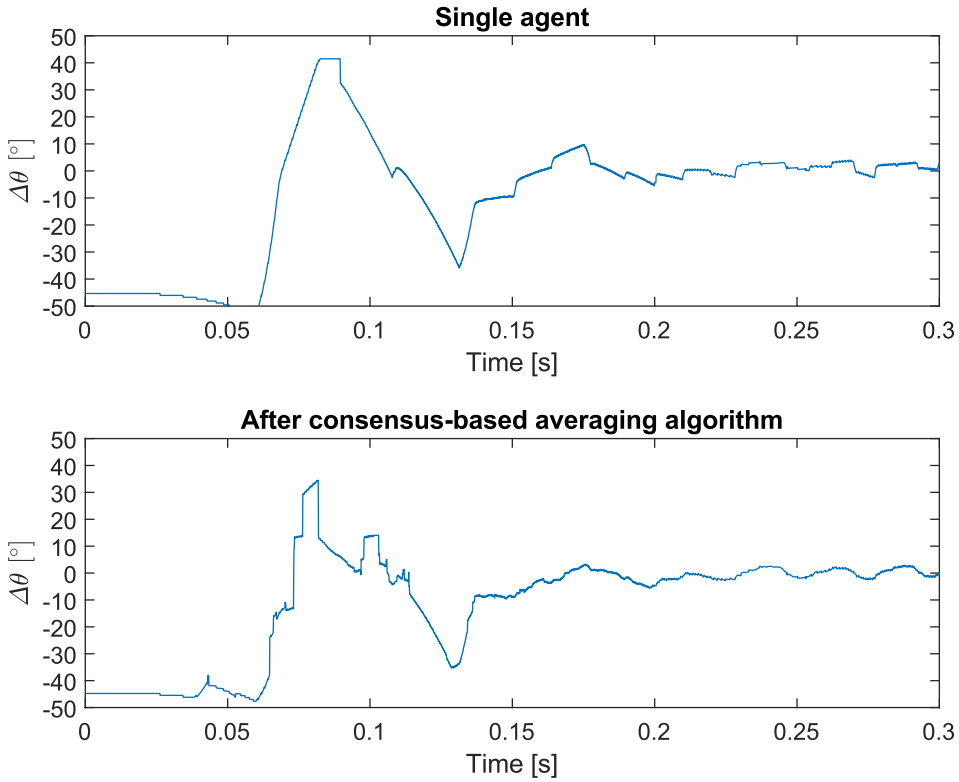


Figure 3.24: Measurement of a start-up with an acceleration of 570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

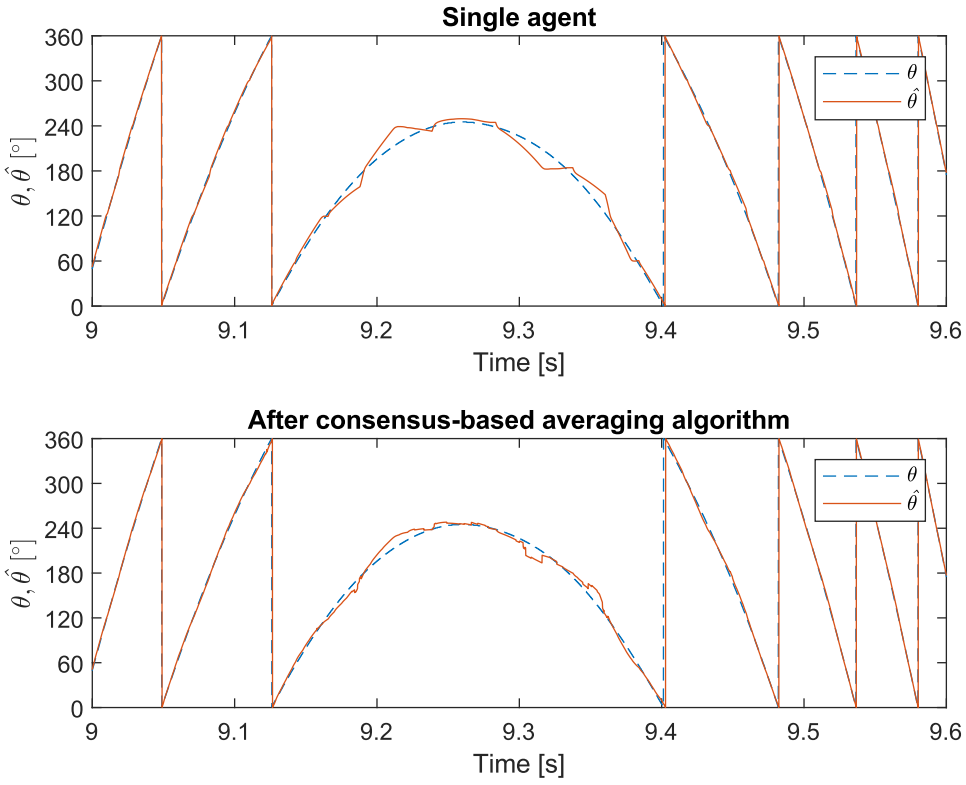


Figure 3.25: Measurement of a speed reversal with an acceleration of -570 rad/s^2. The upper figure shows the waveform $\hat{\theta}$ of a single agent. The lower figure shows the waveform $\hat{\theta}$ after the consensus-based averaging algorithm.

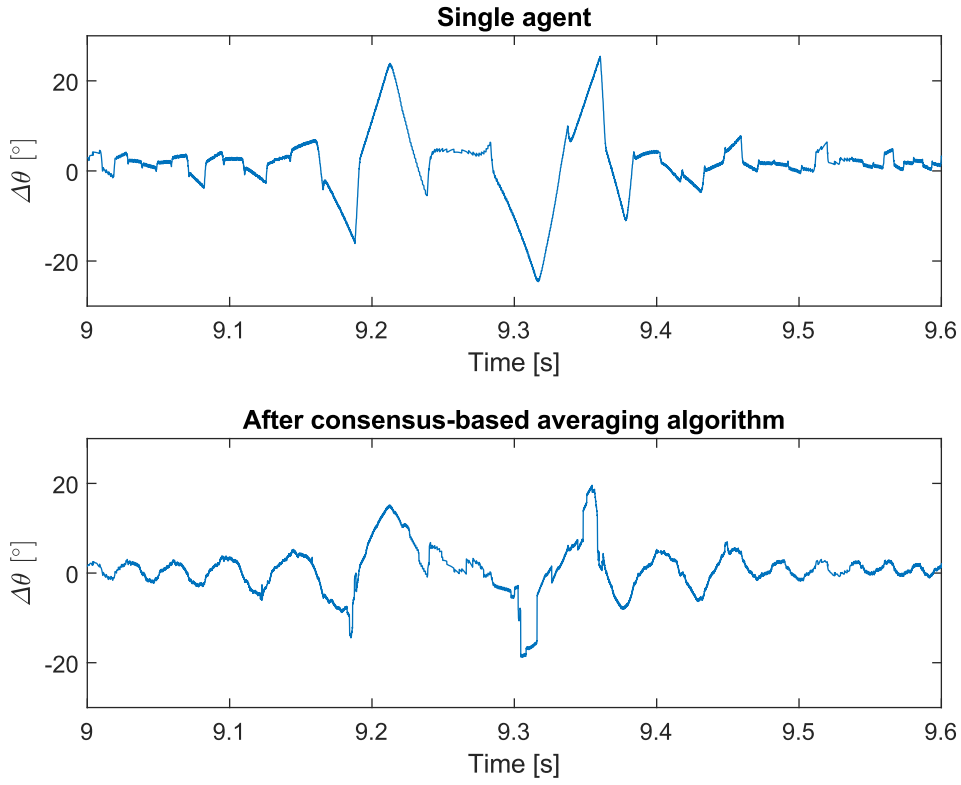


Figure 3.26: Measurement of a speed reversal with an acceleration of -570 rad/s^2. The upper figure shows the error $\Delta\theta$ of a single agent. The lower figure shows the error $\Delta\theta$ after the consensus-based averaging algorithm.

Numerical Analysis of the Measurements

The results of the measurements are not only analysed by using graphs, but also by conducting a numerical analysis. The same measure for the deviation as defined for the simulations is used here: *dev*, see equation (3.7). The results are listed in Table 3.5. The improvement by the consensus algorithm is, as in the graphs, distinct. However, the numerical values of *dev* are a lot higher than in the simulations, this was expected from examining the graphs. In Chapter 5, the difference between the simulations and the measurements is further elaborated.

3.6 Conclusion

In this chapter, an algorithm to estimate the average of the estimates of the rotor position by different agents is proposed. This is done to exploit the fact that in a modular drive there are multiple agents estimating the position of the rotor. This algorithm is executed by every agent in the modular drive and not at one single place. This keeps the estimation of the position of the rotor decentralised and therefore fault-tolerant. Another difficulty is that every agent only communicates with its neighbours.

Using simulations it is shown that the consensus algorithm leads to a better estimate of the rotor position than the estimate of a single agent based solely on the VTO described in Chapter 2. This improvement applies to any speed up to 1500 rpm. Dynamic average consensus algorithms from subsection (3.2) were not able to track the average at those high speeds. Also in dynamical situations such as start-up and reversal, the proposed algorithm led to better position estimates. The better the position estimate, the smoother the torque characteristic of the motor.

In the test setup, optical switches instead of Hall sensors were used. This creates the same output as if Hall sensors would have been used but are far more easy in implementation. From the measurements it can be concluded that the algorithm leads to better estimates of the rotor position. Unfortunately, the relative improvement is not that distinct as in the simulations. This is because the position of the sensors and the length of the sectors on the disc are not exact. This is discussed in Chapter 5.

Table 3.5: Measure for the deviation from the real rotor position for the measurements.

	Deviation *dev* [electr. rad]	
Measurement	Single Agent	After Consensus Algorithm
Regime: 500 rpm	610	403
Regime: 1000 rpm	1359	475
Regime: 1500 rpm	1629	577
Start-up	2340	1333
Reversal	1362	910

Chapter 4

Fault Detection Algorithm

4.1 Introduction

One of the advantages of a modular drive is that it is fault tolerant. This means that when a module fails, the other modules keep on working. Fault tolerant machines are used in environments where failing machines have large consequences concerning safety or financially.

This robustness to failing modules has to be incorporated in the consensus-based averaging algorithm. When a module fails, an agent will output a wrong signal. That is because an agent needs a fixed number of Hall sensors to estimate the rotor position. If an agent outputs a signal that is wrong, the other agents should recognise this. When the failing agent is recognised, his signal should not be included in the calculation of the average. Because a wrong estimation of the position of the rotor by one agent, would lead to an error for all the other agents.

Next to recognising an agent that outputs a faulty signal, another feature could be added to the fault detection algorithm. If an agent is indicated as faulty, but its communication still works with the other agents, it should be able to use the signals of its neighbours. Doing this, the agent calculates the average of the estimations of its neighbours. This estimation could be used for the control of the modules that make up the faulted agent.

To conclude, a fault detection algorithm enables an agent to conclude whether its neighbour or the agent itself is faulty. Next to that, there is the possibility for a faulted agent to use the signals of its neighbours to obtain a good estimation of the rotor position. In this dissertation the following two faults will be discussed: a single Hall sensor that fails

and a single agent that is shut down. An agent shutdown in this dissertation is defined as an agent that has a constant zero signal as output from the VTO, it will also only communicate a constant zero signal to its closest neighbours in both steps in the consensus algorithm from Chapter 3.

4.2 Fault Detection Algorithm

Like the averaging algorithm, this fault detection algorithm uses the sine and the cosine of the rotor position. Remember that this is done to work with continuous functions. In this text, the fault detection algorithm will be elaborated for the sine. Parallel to this, the same happens for the cosine.

A visual representation of every step in the fault detection algorithm is in Figure 4.1. Agent i has two neighbours: agent h and agent j. Agent i receives the sine waves from its neighbours and subtracts them from its own sine wave. The notation for this is Δ_{hi} or Δ_{ji}, depending on which neighbour is subtracted. From now on, expressions will only be shown for the interaction between agent h and agent i to keep it concise. An expression for Δ_{hi} is shown below.

$$\Delta_{hi} = \sin(\hat{\theta}_i) - \sin(\hat{\theta}_h) \tag{4.1}$$

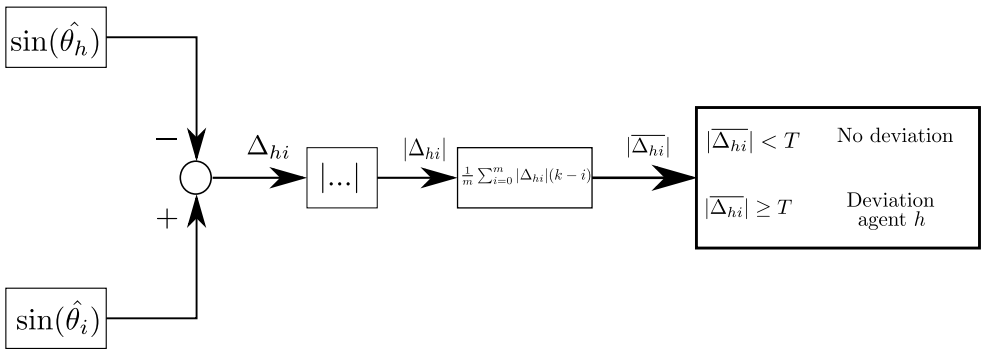


Figure 4.1: Procedure executed by agent i to examine if agent h deviates. An analog procedure is executed by agent i to examine if agent j deviates.

The absolute value of this subtraction is taken, $|\Delta_{hi}|$. To ease out instantaneous errors, a moving average of $|\Delta_{hi}|$ is calculated: $|\overline{\Delta_{hi}}|$. The number of timesteps for which the moving average has to be calculated is arbitrary. The higher the number, the more robust the fault

detection is during dynamical situations such as start-up. During start-up, the differences between the neighbours are usually larger, but all agents are performing normally. It would be incorrect to mark an agent as faulty during that phase. However, choosing a large amount of data to average makes the detection algorithm slower to react to mistakes. It also takes more memory capacity when choosing a large amount of data to average.

$|\overline{\Delta_{hi}}|$ is now compared to a threshold value, T. If this threshold value is exceeded, then there is a so-called deviation. The decision tree of Figure 4.2 shows the logic reasoning for deciding which agent is faulted. If only $|\overline{\Delta_{hi}}| > T$, then it can be concluded that agent

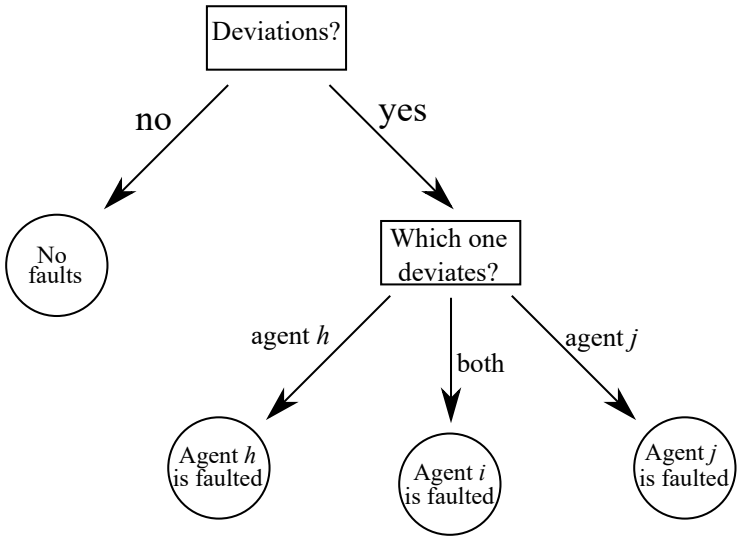


Figure 4.2: Decision three of agent i for determining whether agent h, agent j or agent i itself is faulted.

h is faulted. The same reasoning holds for agent j. If both $|\overline{\Delta_{hi}}|$ and $|\overline{\Delta_{ji}}|$ exceed the threshold, it can be concluded that agent i is faulted.

This fault detection algorithm works when there is only one agent not performing as expected. This could happen when a Hall sensor is broken or the agent is shut down. However, when two agents are communicating a signal that is not correct, this algorithm will not always indicate correctly which agents should be excluded from the consensus-based averaging algorithms. Therefore, it is important to notice that only the two faults described in the beginning of the chapter will be discussed in this dissertation.

When an agent is indicated as defective, it is excluded from the consensus-based averaging algorithm. Please remember that the averaging happens in two steps, to allow more agents

to be involved in the average. This fault detection algorithm is implemented before each step. To explain why the algorithm is implemented before each step, two scenarios are discussed. First, a single sensor failure. Second, an agent that is shut down.

Single Sensor Failure

When one Hall sensor of an agent fails, the output of that sensor is a constant zero signal. The agent should now be indicated as defective. However, the agent is still able to communicate. Therefore, the agent can perform the consensus-based averaging algorithm based on the input of its neighbours, see equation (4.2a). This leads to a prediction $\overline{(\sin\hat{\theta})}(k+2)$, see equation (4.2b), that is communicated to its neighbours. After this second communication, the fault detection algorithm is run again with these signals as input. This is to determine whether the averages based on two agents do not deviate too much from the averages based on three agents. If this is the case, the output is calculated according to equation (4.2c).

$$\overline{(\sin\hat{\theta})}_i(k) = \frac{1}{2}\left(\sin\hat{\theta}_h(k) + \sin\hat{\theta}_j(k)\right) \tag{4.2a}$$

$$\overline{(\sin\hat{\theta})}_i(k+2) = \overline{(\sin\hat{\theta})}_i(k)\cos(2\hat{\omega}T_s) + \overline{(\cos\hat{\theta})}_i(k)\sin(2\hat{\omega}T_s) \tag{4.2b}$$

$$\overline{(\sin\hat{\theta})}_{hij}(k+2) = \frac{1}{2}\left(\overline{(\sin\hat{\theta})}_h(k+2) + \overline{(\sin\hat{\theta})}_j(k+2)\right) \tag{4.2c}$$

Agent is Shut Down

When an agent is shut down or does not communicate anymore it is important to look at what happens at its neighbours. The neighbours should be able to recognise the shut down agent in both communication steps. That is another reason why the fault detection algorithm is also implemented at the second step.

4.3 Simulations and Measurements

4.3.1 Simulations

For the simulations a window length for the moving average of five timesteps is chosen. This value is a compromise. For the situation of one faulted sensor, it is advisable to choose the window length as small as possible. This is because the waveform of the agent with a faulted sensor only deviates from a normal waveform in particular areas: this can be

seen in Figure 4.3. To be able to exclude the agent only in the instants that the waveform deviates too much, the window length should be chosen small.

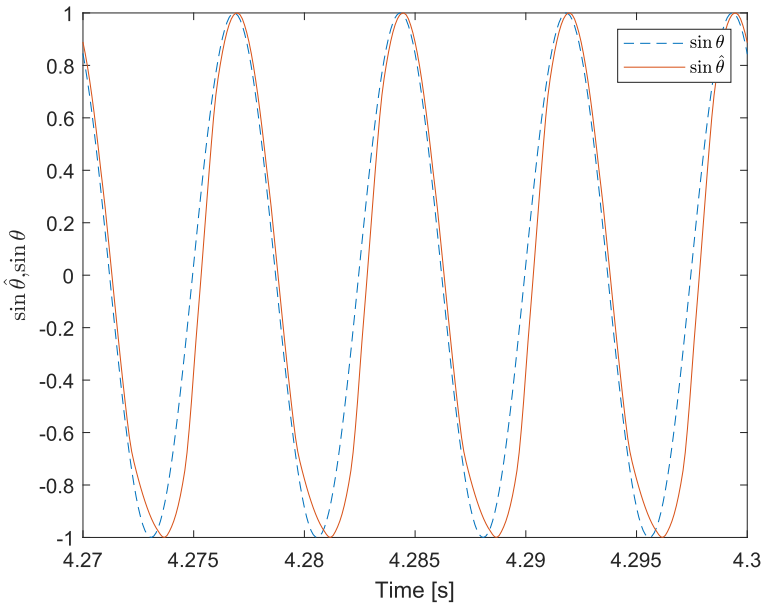


Figure 4.3: Simulation when one sensor is faulted at a mechanical speed of 1000 rpm. The dashed line is the sinewave of the real angle θ. The full line is the waveform of the estimated angle $\hat{\theta}$ by an agent with one faulted sensor.

In the situation that an agent is shut down, it is better to have a larger window length. Because it is not desired that any signal of that agent is included in the consensus-based averaging algorithm. Remember that in this dissertation "an agent shutdown" means that the output of its VTO is a constant zero signal and the agent communicates in both communication steps a constant zero signal. In this dissertation, a window length of five timesteps is chosen.

The threshold value T is chosen to be 0.05. In Figure 4.4 $|\overline{\Delta_{hi}}|$ is plotted in normal operation, in operation with a faulted sensor and in operation with an agent shutdown, all for the sine part of the fault detection algorithm. A horizontal line is drawn at $|\overline{\Delta_{hi}}|$ equal to 0.05 in all plots. This choice for T leads to a correct decision in normal operation and excludes the agent with the faulted sensor when its difference with a healthy agent becomes too large. The agent that is shut down, is always excluded from the consensus algorithm. In Figure 4.5, the same plots are made as in Figure 4.4 now for the cosine part

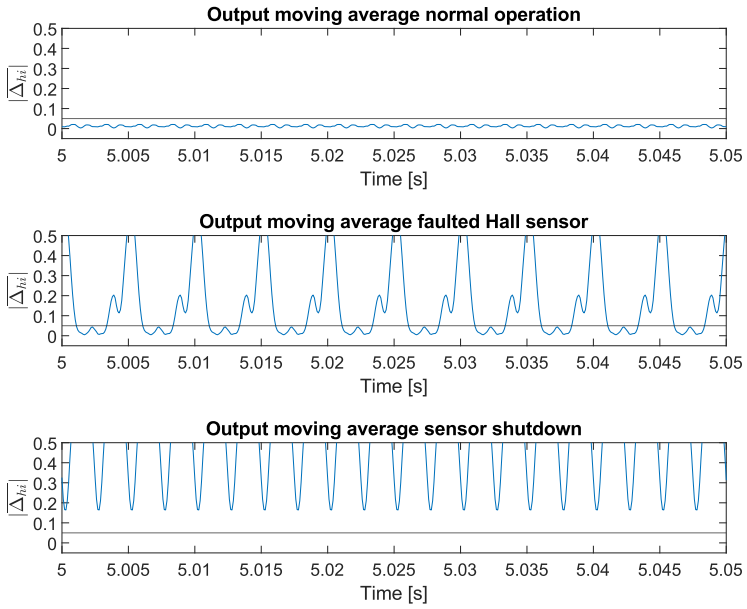


Figure 4.4: Simulation of $|\overline{\Delta_{hi}}|$ for the sine part of the fault detection algorithm at a mechanical speed of 1500 rpm. In the upper figure, normal operation is simulated. In the middle figure, agent h has a faulted sensor. In the lower figure, agent h is shut down. A horizontal line is drawn at $|\overline{\Delta_{hi}}| = 0.05$.

of the fault detection algorithm. In this figure, it is clear that T equal to 0.05 is a good choice as threshold value. Although it could be better for recognising faults to lower T, it cannot be chosen any lower than 0.05 since the fault detection algorithm would indicate normal operation as faulty.

Below two scenarios will be discussed: single sensor failure and an agent shutdown.

Single Sensor Failure

In the simulation of this situation, the fault is started at four seconds into the simulation. The error of the agent ($\Delta\theta$), without the fault detection algorithm and the error of the agent with the fault detection averaging algorithm will be discussed. Next to that, the error of its neighbour with and without the fault detection algorithm is discussed. The simulations are at a mechanical speed of 500 rpm and at a mechanical speed of 1500 rpm.

In Figure 4.6 the error of the agent with one faulted sensor and the error of the same agent with the consensus-based averaging algorithm and the fault detection algorithm is

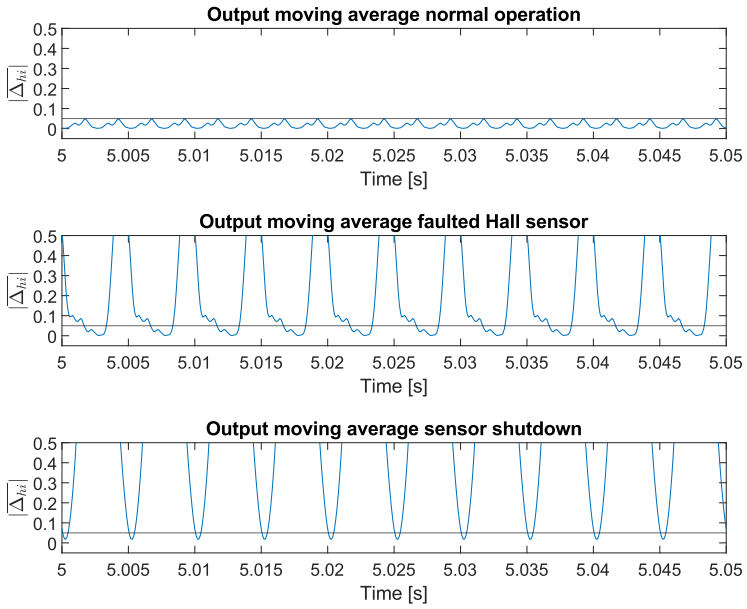


Figure 4.5: Simulation of $|\overline{\Delta_{hi}}|$ for the cosine part of the fault detection algorithm at a mechanical speed of 1500 rpm. In the upper figure, normal operation is simulated. In the middle figure, agent h has a faulted sensor. In the lower figure, agent h is shut down. A horizontal line is drawn at $|\overline{\Delta_{hi}}| = 0.05$.

plotted. As one can see, the error of the single agent goes up to almost thirty degrees. This estimation is not useful for vector control of the machine. With the combination of algorithms, the error is made a lot smaller.

To show the importance of the fault detection algorithm, in Figure 4.7, the error after the consensus-based averaging algorithm without the fault detection algorithm is shown. The results in Figure 4.7 prove the utility of the consensus-based averaging algorithm together with the fault detection algorithm.

The results of the algorithms are examined for the neighbour of the agent with the faulted sensor as well. Figure 4.8 shows the error of the agent before the combination of algorithms and after. As can be seen, the faulted sensor has an influence on its neighbours. This causes the estimation of the rotor position at the neighbours to be less accurate after the consensus-based averaging algorithm. This is not desirable. However the increase in magnitude of the error (about 1 degree increase) is rather small in comparison to the reduction of the error at the agent with the faulted sensor. Therefore, it can be concluded

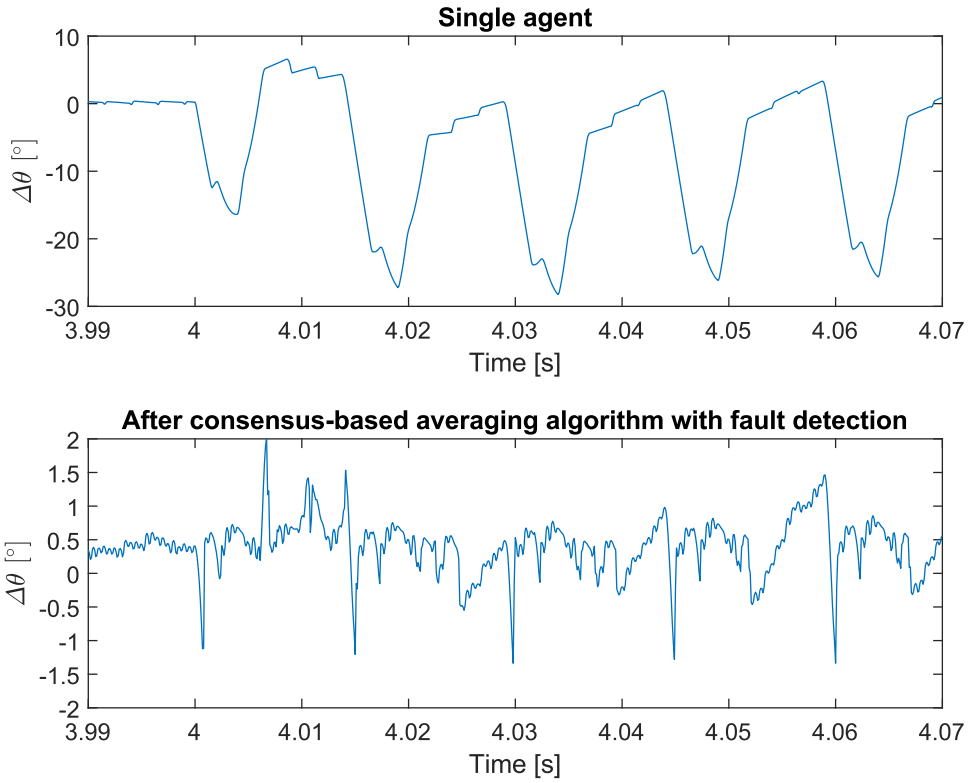


Figure 4.6: Simulation when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

that the consensus-based averaging algorithm provides a more accurate estimate of the rotor position when considering the modular drive as a whole.

The same simulations are performed at higher speed: 1500 rpm. The results are plotted in Figures 4.9 and 4.10. In Figure 4.9 the error before and after the consensus-based averaging algorithm and fault detection algorithm for the agent with one faulted sensor is plotted. The results are similar as when the machine was rotating at 500 rpm. The error is reduced considerably due to both the consensus and fault detection algorithm. In Figure 4.10 the results are plotted for the neighbour of the agent with the faulted sensor.

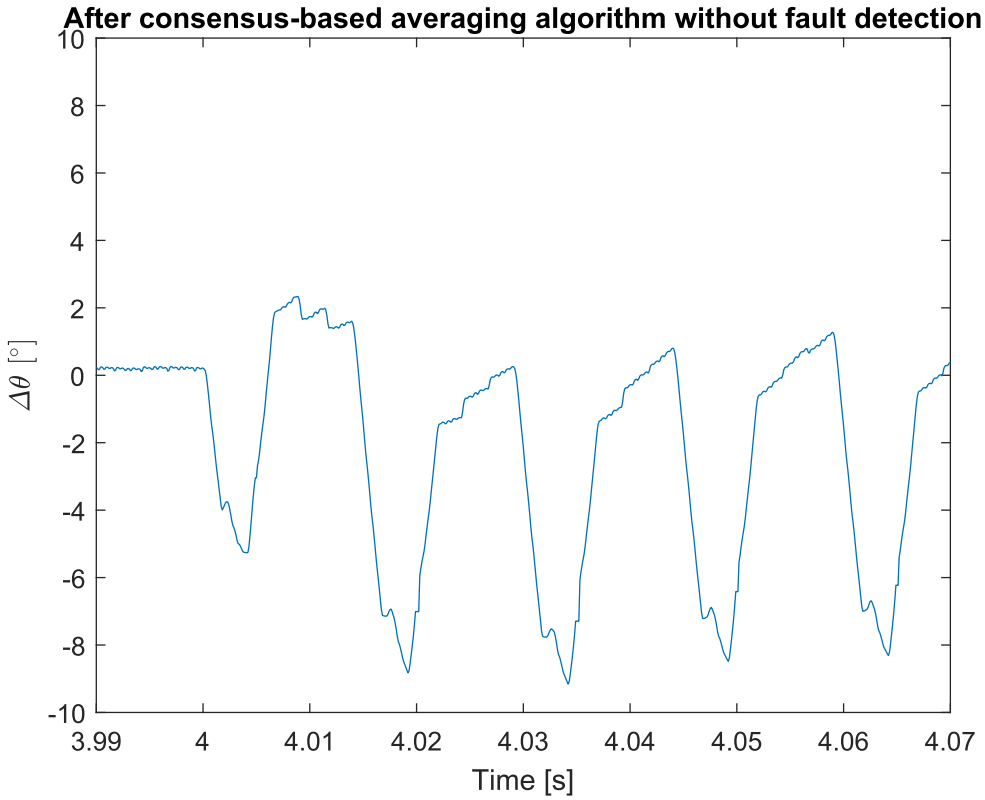


Figure 4.7: Simulation when one sensor is faulted at a mechanical speed of 500 rpm. The error $\Delta\theta$ of the agent with the faulted sensor is plotted with implementation of the consensus algorithm but without the fault detection algorithm. The sensor fails at four seconds.

Agent is Shut Down

To simulate this situation, it is chosen to only partially shut down the agent. It is shut down in the sense that it will communicate a constant (zero) signal to its neighbours in both communication steps. However, the agent keeps performing the calculations for the consensus algorithm. In this way, it can be examined how the error $\Delta\theta$ looks like when an agent calculates the average rotor position only based on information from its neighbours. So for the neighbours, it seems that the agent is shut down. The agent itself performs calculations based on communication with the neighbours.

In Figure 4.11 the result for the agent that is shut down is displayed. Remember that the agent is not entirely shut down as it still receives signals from its neighbours and can perform the algorithm. It is only its own signal, which is used as input for itself and for

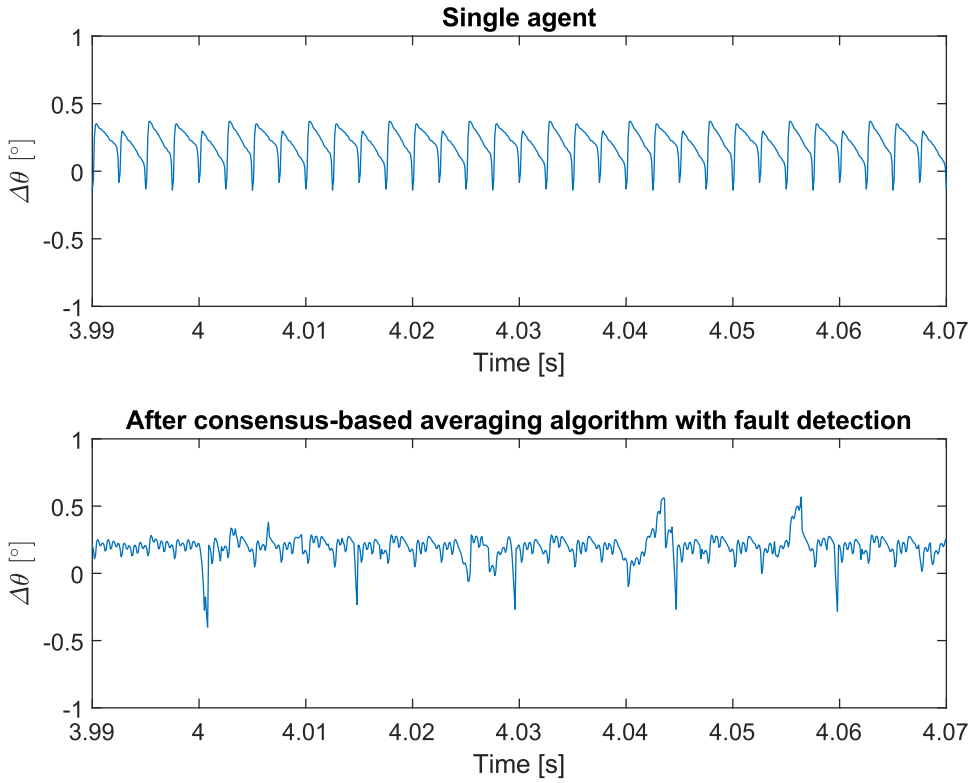


Figure 4.8: Simulation when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor based on its VTO solely. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

its neighbours in both communication steps, that is set to zero starting from four seconds into the simulation. At this moment, a peak in the error can be observed. After the peak, the error stays within the same boundaries as before the shutdown.

Looking to the neighbour of the agent that is shut down, similar results can be observed. This is seen in Figure 4.12. At four seconds in the simulation, the shut down agent is excluded from the average and its neighbour performs the algorithm with his own input and with the input from its other neighbour. The magnitude of the error increases after four seconds but stays smaller than without the consensus-based averaging algorithm.

The shut down of an agent is also studied at higher speed. The results can be seen in

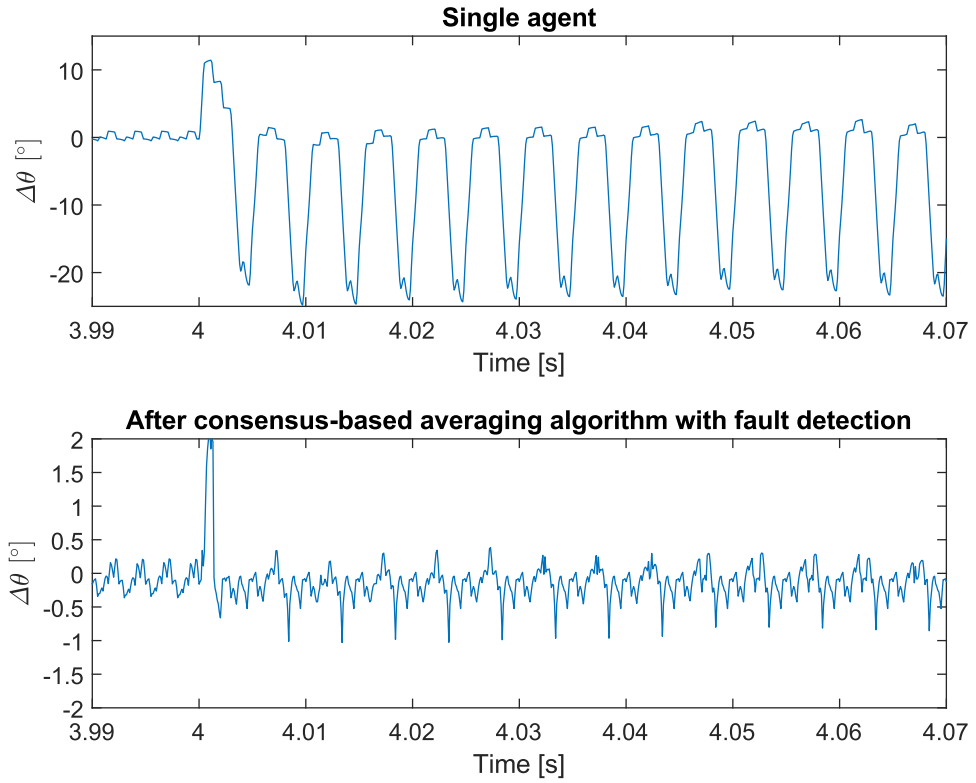


Figure 4.9: Simulation when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

Figures 4.13 and 4.14. The results are similar. The peak in the graph from the agent that is shut down has become smaller in comparison to the low speed simulation. At the neighbour, a larger peak occurs at four seconds. After that it recovers and the magnitude of the error increases. However, the error still stays smaller than without the consensus algorithm.

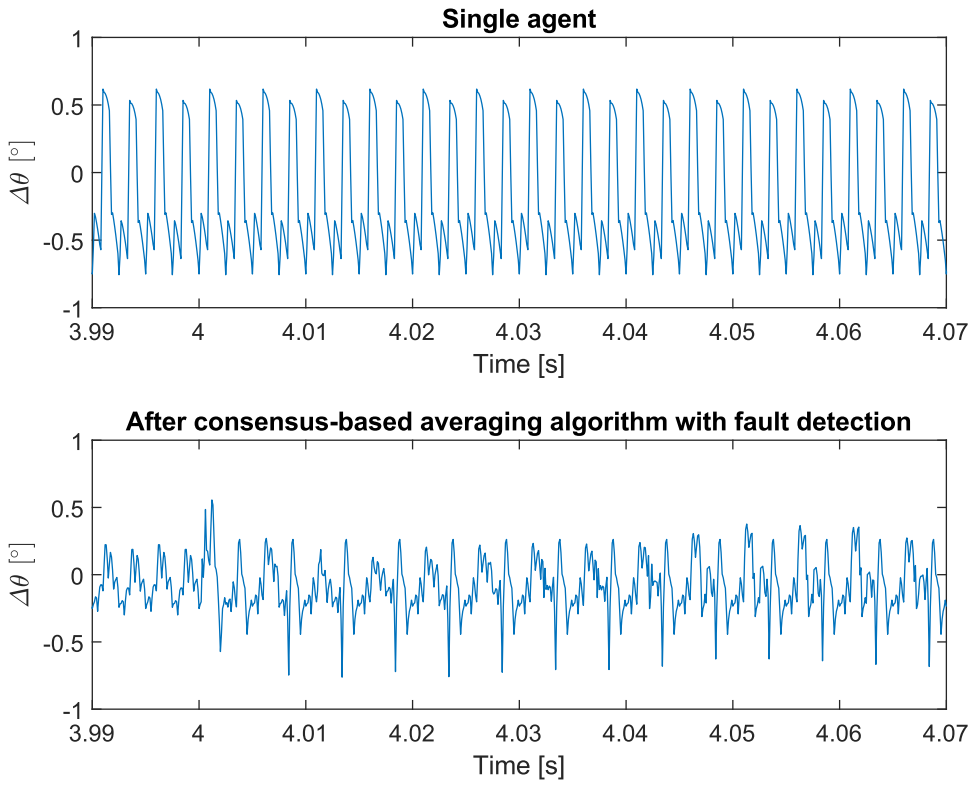


Figure 4.10: Simulation when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

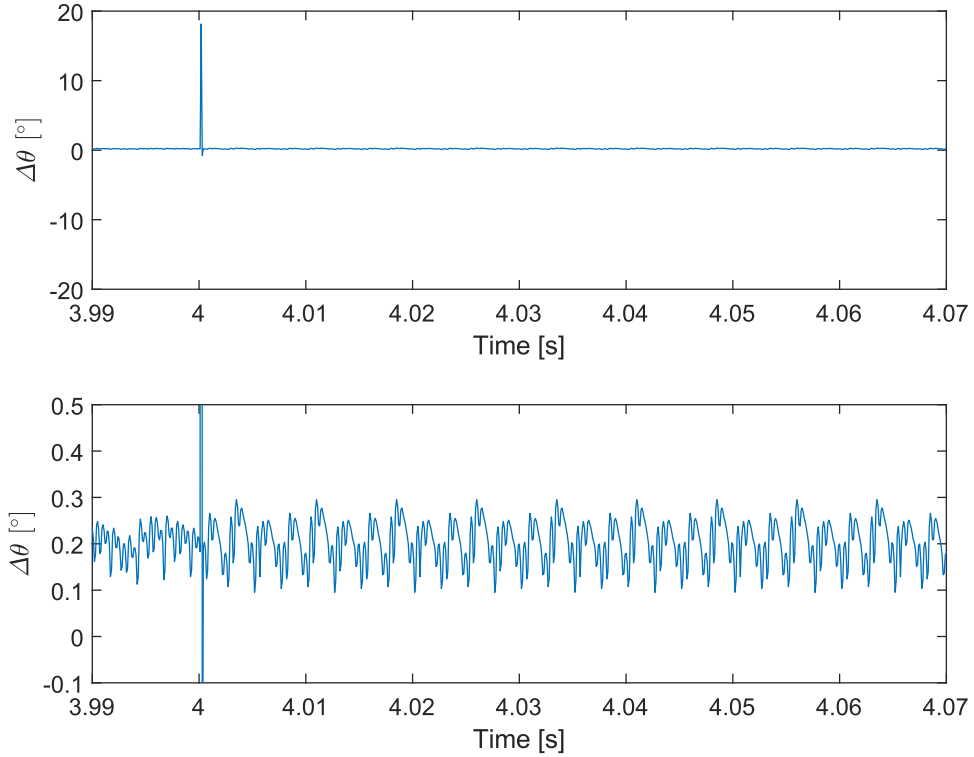


Figure 4.11: Simulation when one agent is shut down at a mechanical speed of 500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down with implementation of the consensus and fault detection algorithm. The lower figure is the zoomed version of the upper figure. The agent is shut down at four seconds.

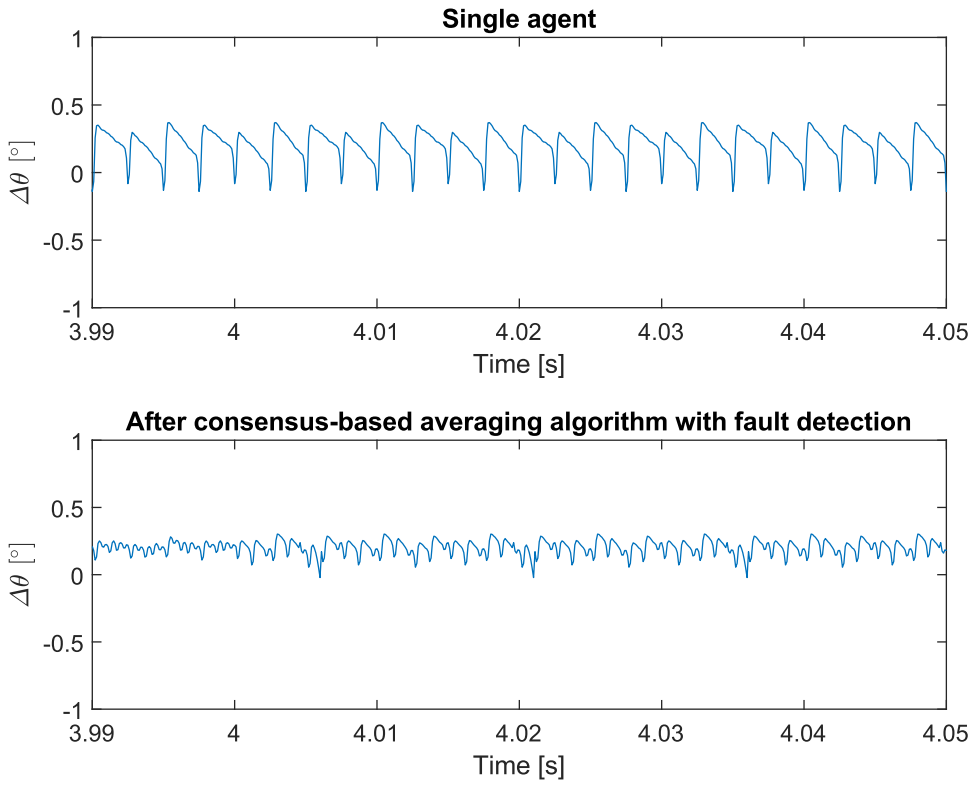


Figure 4.12: Simulation when one agent is shut down at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds.

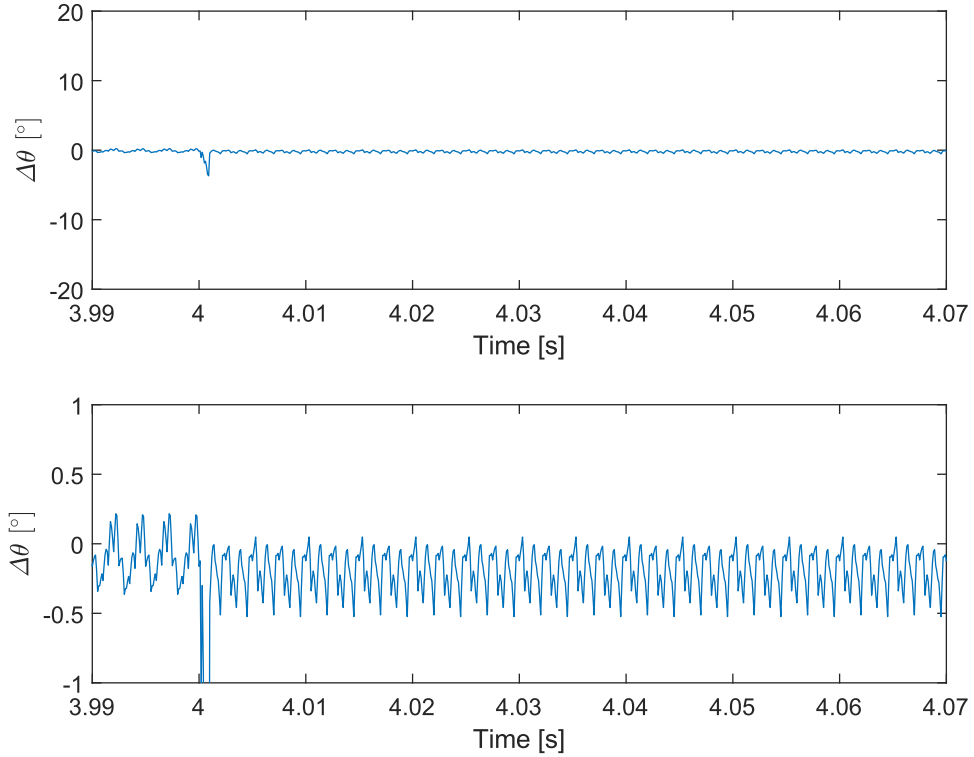


Figure 4.13: Simulation when one agent is shut down at a mechanical speed of 1500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down with implementation of the consensus and fault detection algorithm. The lower figure is the zoomed version of the upper figure. The agent is shut down at four seconds.

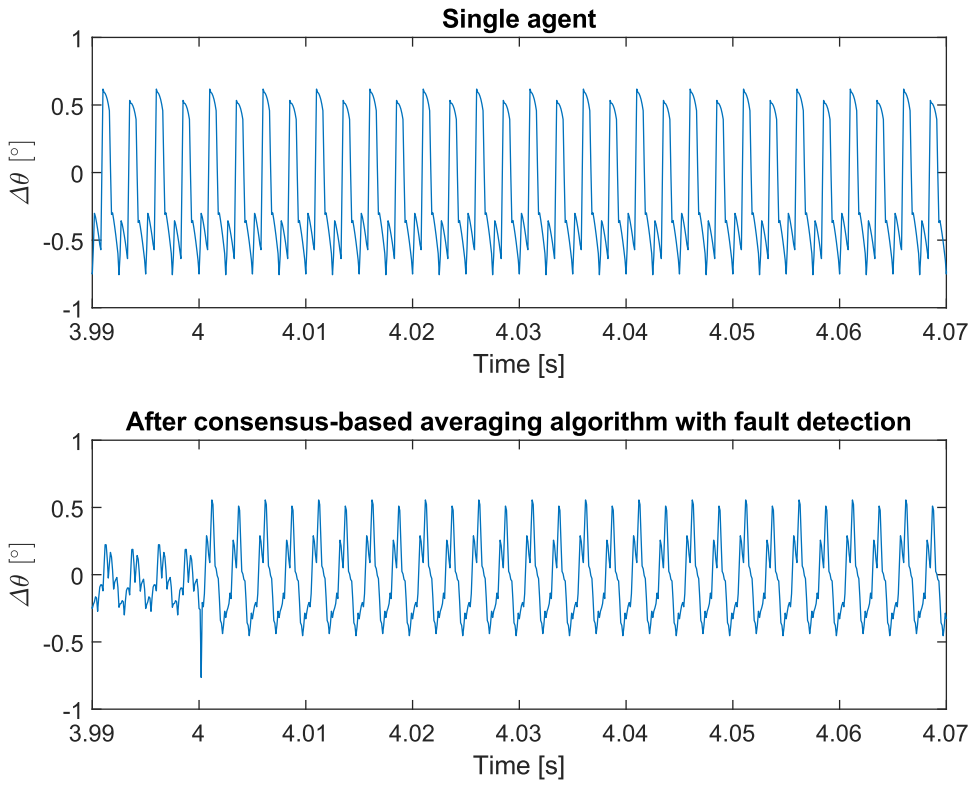


Figure 4.14: Simulation when one agent is shut down at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds.

Numerical Analysis of the Simulations

In Chapter 3, a numerical analysis was conducted to examine the improvement in position estimation by the consensus algorithm. For the analysis, a measure for the deviation was introduced (*dev*), see equation (3.7). Also in this chapter, *dev* is used to compare the results of the position estimation with and without the consensus and the fault detection algorithms. For every simulation, *dev* is calculated over a period of two seconds. These two seconds are always chosen in a new steady-state period after the initialization of the fault. The results are listed in Table 4.1.

4.3.2 Measurements

The situations from above are also measured on the test setup. The same setup is used as in the previous chapter. The window length of the moving average in the fault detection algorithm is five timesteps, just as in the simulations. Also the threshold T is the same: 0.05.

Single Sensor Failure

In Figure 4.15 the measurement of the agent with one faulted sensor is shown for a mechanical speed of 500 rpm. The error of the VTO of the agent goes up to 60 electrical degrees. Because the agent is still able to communicate, it can use the signals from its neighbours to improve the estimation of the rotor position. It results in a fault that is reduced to maximally 10 electrical degrees. This measurement is not as accurate as the simulations. This is caused by wrong sensor outputs due to sensor misplacements. In Chapter 5, this issue is discussed.

Also a measurement of the neighbour of the agent with the faulted sensor was performed. This can be seen in Figure 4.16. The neighbour is affected a little by the agent with a faulted sensor. However, the beneficial effect of the consensus algorithm on the position estimation of the agent with the faulty sensor outweighs the adverse effect on the position estimation of its neighbour.

At higher speed, 1500 rpm, the same conclusions can be drawn. The results for the measurements at 1500 rpm can be seen in Figures 4.17 en 4.18.

Table 4.1: Measure for the deviation from the real rotor position for the simulations with fault situations.

	Deviation *dev* [electr. rad]	
Simulation	Single Agent	After Cons. & Fault Det. Algorithm
Sensor failure 500 rpm	6938	94
Sensor failure 500 rpm neighbour	91	73
Sensor failure 1500 rpm	6469	160
Sensor failure 1500 rpm neighbour	373	147
Module shutdown 500 rpm neighbour	91	48
Module shutdown 1500 rpm neighbour	372	270

Agent is Shut Down

Measurements for this situation are performed with the same conditions as for the simulations. The faulted agent does not use nor communicates its own signal in both calculation steps of the consensus algorithm. It is, however, capable of receiving signals from its neighbours and calculating the average rotor position based on the signals of its neighbours. In Figure 4.19 the result for the faulted agent at 500 rpm is plotted. As one can see, the error slightly increases when the fault occurs.

For the neighbour of the faulted agent, an increase in error can be seen. The error is now even slightly larger than without the averaging algorithm at some instants, this is also calculated in Table 4.2. Since this was not the case in the simulations, an explanation can be the misalignment of the sensors, this is further elaborated in Chapter 5. The result of this measurement is in Figure 4.20.

The same measurements are performed for high speed also. The results for 1500 rpm can be seen in Figures 4.21 en 4.22. The conclusions are similar as for 500 rpm, except that for the neighbour of the agent that is shut down, the consensus and fault detection algorithms lead to better results than the single agent.

Numerical Analysis of the Measurements

Also for these measurements, a numerical comparison is conducted. The results can be seen in Table 4.2. The numerical values show an improvement when applying the consensus and fault detection algorithms except for the measurement of the error of the neighbour of an agent that is shut down at 500 rpm. At 1500 rpm there is for the same situation a

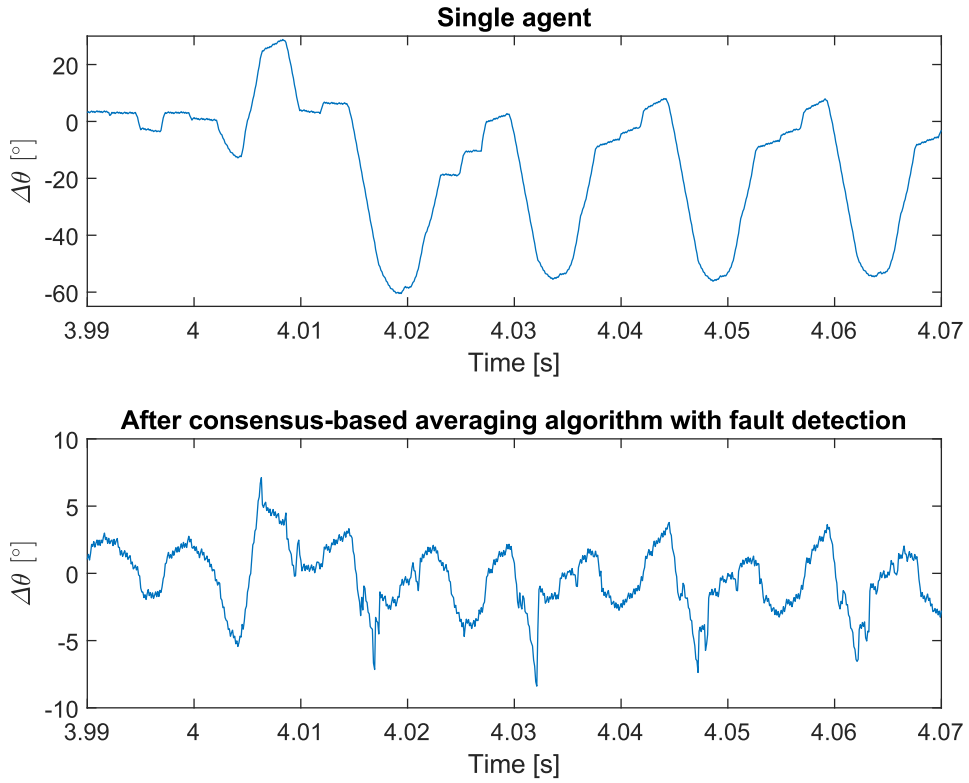


Figure 4.15: Measurement when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

large improvement by the algorithm. The odd result of the measurement of the neighbour of the agent that is shut down at 500 rpm could be caused by misalignment of the sensors, also in Chapter 5, where sensor misalignment is discussed, one odd result is obtained by simulations with realistic sensor positions. Another explanation could be that a mistake was made during that measurement.

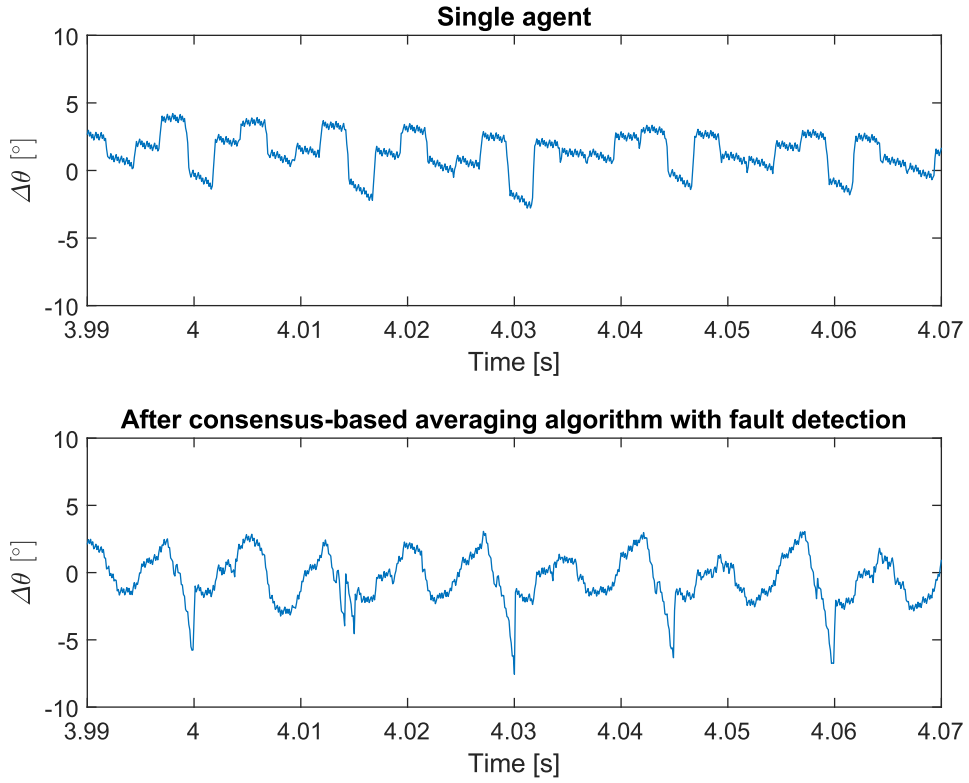


Figure 4.16: Measurement when one sensor is faulted at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

Table 4.2: Measure for the deviation from the real rotor position for the measurements with fault situations.

	Deviation *dev* [electr. rad]	
Measurement	Single Agent	After Cons. & Fault Det. Algorithm
Sensor failure 500 rpm	6997	671
Sensor failure 500 rpm neighbour	629	553
Sensor failure 1500 rpm	6733	745
Sensor failure 1500 rpm neighbour	1631	655
Module shutdown 500 rpm neighbour	624	764
Module shutdown 1500 rpm neighbour	1507	774

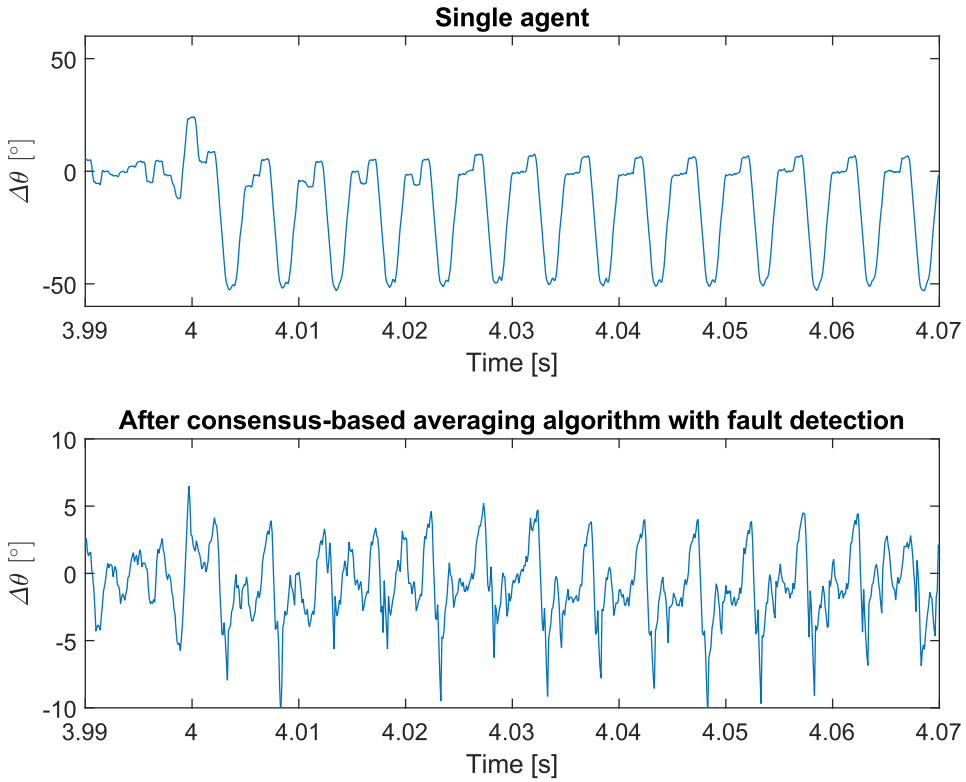


Figure 4.17: Measurement when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the single agent. The lower figure shows the error $\Delta\theta$ of the agent after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

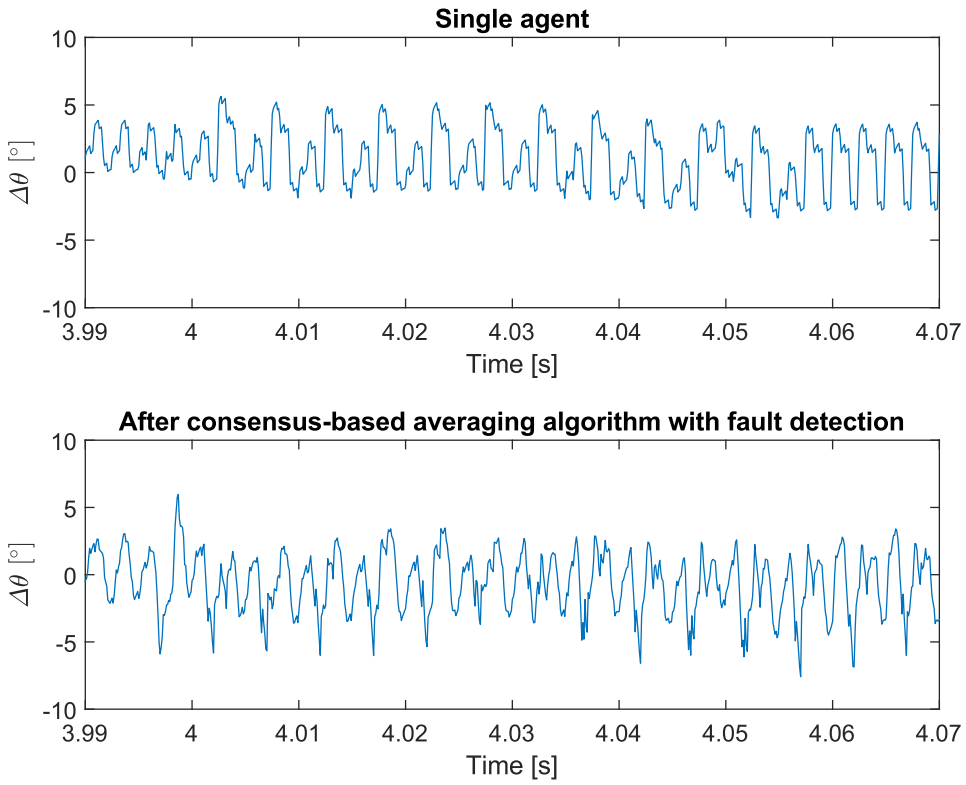


Figure 4.18: Measurement when one sensor is faulted at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent with the faulted sensor after the consensus-based averaging algorithm together with the fault detection algorithm. The sensor fails at four seconds.

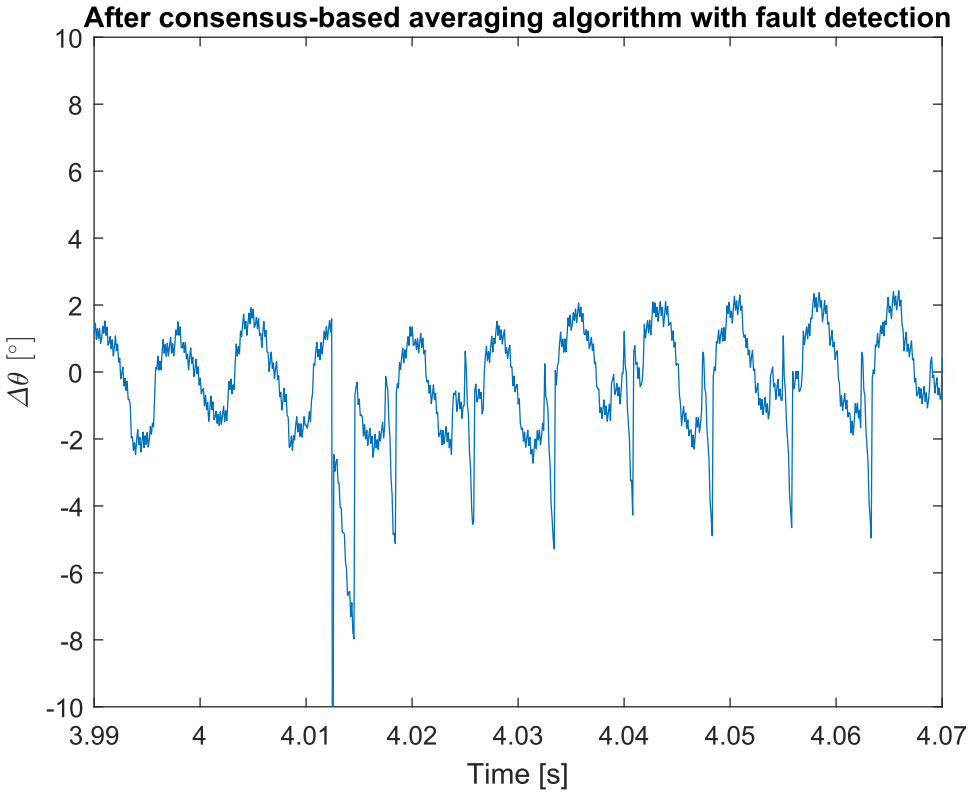


Figure 4.19: Measurement when one agent is shut down at a mechanical speed of 500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down. The agent is shut down at four seconds.

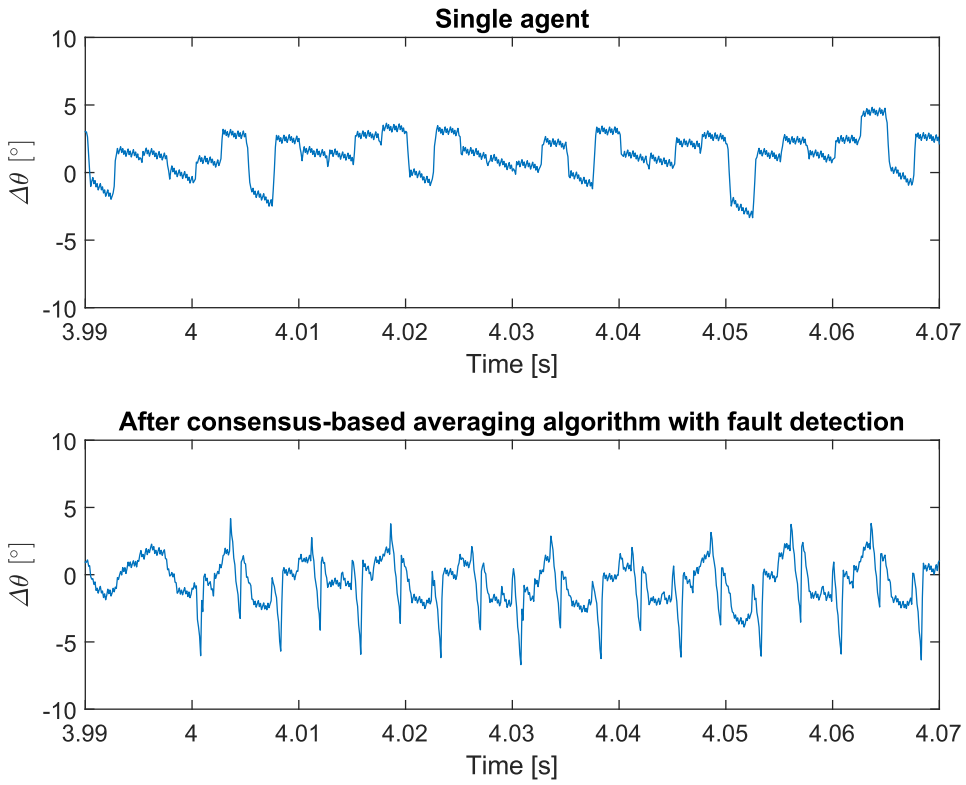


Figure 4.20: Measurement when one agent is shut down at a mechanical speed of 500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds.

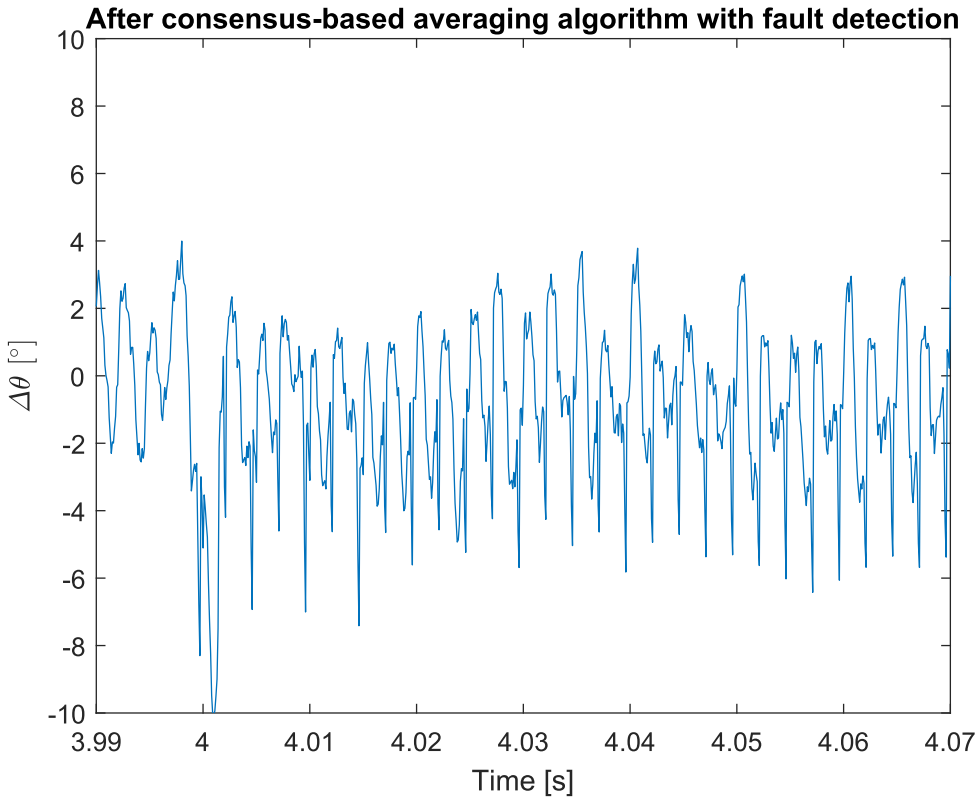


Figure 4.21: Measurement when one agent is shut down at a mechanical speed of 1500 rpm. This figure plots the error $\Delta\theta$ of the agent that is partially shut down. The agent is shut down at four seconds.

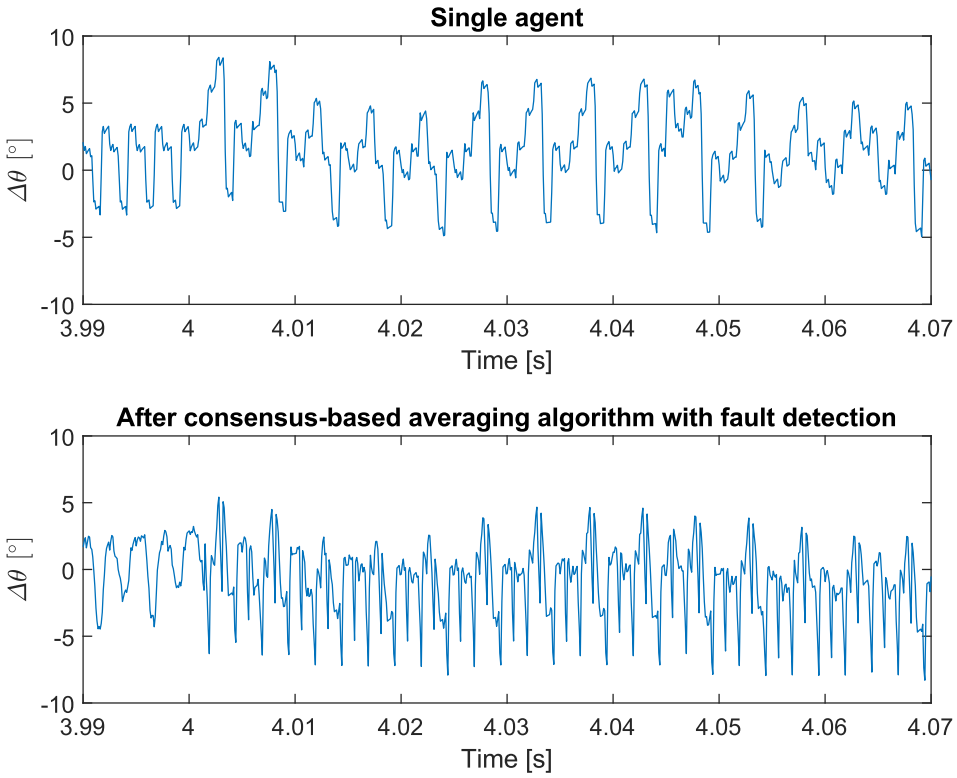


Figure 4.22: Measurement when one agent is shut down at a mechanical speed of 1500 rpm. The upper figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down. The lower figure shows the error $\Delta\theta$ of the neighbour of the agent that is shut down after the consensus-based averaging algorithm together with the fault detection algorithm. The agent is shut down at four seconds.

4.4 Conclusion

One of the most important advantages of a modular drive is its fault-tolerancy. In order to make the position estimation robust for faults, a fault detection algorithm is needed. A fault detection algorithm was proposed in this chapter and it was tested for two fault situations.

The first situation was the shut down of one single Hall sensor. Simulations and measurements show that the error of the agent with the faulted sensor can go up to 60 electrical degrees. However, with the averaging and fault detection algorithm, this error is reduced to 6 electrical degrees. The price to pay for this improvement is a slight increase in the fault of the neighbours' estimate.

The second situation is that one agent's signal falls to zero. It cannot use its own signal and its signal is not communicated with the neighbours. For the calculation of the average it relies entirely on the signals of its neighbours. Due to the combination of the averaging algorithm and the fault detection algorithm, this agent still has an estimate for the rotor position. Therefore, the modules of this agent can keep on working even without an own position estimation. Again, at the neighbour of the faulted agent, there is a slight increase in the error.

Chapter 5

Sensor Misalignment

5.1 Introduction

In this chapter, an inquiry into the cause of the increased magnitude of $\Delta\theta$ in the measurements is performed. From the previous chapters it can be concluded that both the VTO of a single agent and the consensus algorithm perform less good than expected from the simulations. It is suspected that a misalignment of the sensors could be the cause of that issue. In the following sections it will be demonstrated that this is indeed a large factor in the deviation from the real rotor position.

5.2 Real Sensor Outputs

Three factors play a role in the output of the sensors: the width of the permeable/non-permeable sections in the rotating disc, the radius on which the sensor is mounted and the spacing between the different sensors. The first two factors affect the output of every single sensor, while the third factor affects the cooperation between multiple sensors. The width of the different sections can variate because the disc is 3D-printed. This method is not exact and deviations can occur. The most important factor, is probably the radius on which the sensors are mounted. Because the edges of a sector are not in line with the middle of the disc, there is only one specific radius for which the subsequent sectors are equally long. Since the sensors are mounted on the stationary panel with screws, the radius is not exact. The third factor influences the estimation of the rotor position because the vector-tracking observer assumes that the sensors are equally spaced around the circumference.

The three factors discussed above can be examined by measuring the output of every sensor

during one mechanical rotation. Since the rotor has eight pole pairs, one mechanical rotation leads to eight electrical revolutions or 2880 electrical degrees. Remember that during one electrical revolution, the sensor's waveform should be a square wave with 180 electrical degrees high voltage output and 180 electrical degrees low voltage output. In Tables 5.1 and 5.2 the result of this measurement is shown. For every sensor, numbered from one to fifteen, the ideal and the real positions of the rising and the falling edges of the square wave are shown. From this table it can be concluded that sometimes, the position of the edges are way of the ideal edge positions.

5.3 Simulations

To see what the influence is of these sensor outputs, the measured locations of the rising and falling edges of the square waves are used in a MATLAB Simulink simulation. In Chapters 2, 3 and 4 simulations were done based on ideal positions of the sensors. The results of the simulations with real positions can be seen in Figures 5.1, 5.2 and 5.3. It shows that the estimation of a single agent is not as accurate anymore as in the simulations with ideal sensor positions. The error $\Delta\theta$ varies now between plus and minus three degrees. With the ideal sensor positions, $\Delta\theta$ varied between plus and minus one degree. This explains for a great part why $\Delta\theta$ in the measurements at the real setup were larger than in the simulations with the ideal sensor positions.

Another remark about the plots is that the consensus algorithm still leads to improved estimations of the rotor position. The consensus algorithm averages out the deviations in position estimates caused by sensor misalignment. This decreases the adverse effect of sensor misalignment. This is another advantage of the consensus based averaging algorithm. The relative improvement is, however, much larger in the simulations with the ideal sensor positions. Therefore it is advisable to place the sensors with great care.

These simulations are also analysed using *dev*, see equation (3.7). The results are listed in Table 5.3. These results confirm the conclusions drawn from the graphs. *dev* is generally higher than in Table 3.2, this means that misalignment of the sensors is an explanation for higher *dev* in the measurements. The difference in *dev* between the simulations and the measurements is now reduced. Next to that, the numerical results show that the consensus algorithm still leads to improved estimates of the rotor position in comparison to the VTO of an agent individually. There is one remarkable result: at 1000 rpm, *dev* for the position estimation by a single sensor is unusually large. This was not the case with ideal sensor

		Pole number							
	Position edge [°]	1	2	3	4	5	6	7	8
1	Ideal rising	240	600	960	1320	1680	2040	2400	2760
	Real rising	239.7	599.0	958.3	1319	1678.3	2036.9	2397.6	2759
	Ideal falling	60	420	780	1140	1500	1860	2220	2580
	Real falling	57.63	416.93	776.23	1136.93	1495.52	1855.52	2216.93	2576.93
2	Ideal rising	120	480	840	1200	1560	1920	2280	2640
	Real rising	123	483.7	843.7	1203	1563	1923.7	2283.7	2642.3
	Ideal falling	300	660	1020	1380	1740	2100	2460	2820
	Real falling	300.9	660.9	1021.6	1380.2	1739.5	2100.9	2460.2	2820.2
3	Ideal rising	360	720	1080	1440	1800	2160	2520	2880
	Real rising	358.6	718.6	1076.5	1437.2	1797.9	2157.9	2517.9	2878.6
	Ideal falling	180	540	900	1260	1620	1980	2340	2700
	Real falling	176.5	535.1	894.4	1255.1	1615.1	1976.5	2335.1	2695.1
4	Ideal rising	72	432	792	1152	1512	1872	2232	2592
	Real rising	73.1	432.4	792.4	1152.4	1511.7	1871	2231.7	2593.1
	Ideal falling	252	612	972	1332	1692	2052	2412	2772
	Real falling	251	609.6	968.9	1329.6	1688.9	2048.9	2410.3	2770.3
5	Ideal rising	312	672	1032	1392	1752	2112	2472	2832
	Real rising	315.7	676.4	1037.1	1397.1	1756.4	2116.4	2477.1	2837.1
	Ideal falling	132	492	852	1212	1572	1932	2292	2652
	Real falling	133.6	494.3	854.3	1214.3	1573.6	1932.9	2476.4	2833.6
6	Ideal rising	192	552	912	1272	1632	1992	2352	2712
	Real rising	188.4	548.4	906.3	1266.3	1627.7	1987.7	2347.7	2707.7
	Ideal falling	12	372	732	1092	1452	1812	2172	2532
	Real falling	5.6	364.2	724.2	1084.9	1444.9	1806.3	2165.6	2524.9
7	Ideal rising	264	624	984	1344	1704	2064	2424	2784
	Real rising	267.9	628.6	987.9	1347.2	1707.9	2067.9	2425.8	2787.2
	Ideal falling	84	444	804	1164	1524	1884	2244	2604
	Real falling	85.8	446.5	805.1	1164.4	1525.1	1883.6	2244.4	2605.1
8	Ideal rising	144	504	864	1224	1584	1944	2304	2664
	Real rising	146.2	506.9	867.6	1227.6	1586.9	1946.9	2307.6	2667.6
	Ideal falling	324	684	1044	1404	1764	2124	2484	2844
	Real falling	324.8	684.8	1044.8	1405.5	1764.1	2124.1	2485.5	2844.8

Table 5.1: Locations of the rising and falling edges of the square waves of the sensors for one mechanical rotation. Sensor 1 to 8

		Pole number							
	Position edge [°]	1	2	3	4	5	6	7	8
9	Ideal rising	24	384	744	1104	1464	1824	2184	2544
	Real rising	23.9	383.9	743.2	1101.8	1461.8	1823.2	2183.2	2543.9
	Ideal falling	204	564	924	1284	1644	2004	2364	2724
	Real falling	201.1	561.8	920.4	1279.7	1640.4	2001.1	2361.8	2721.8
10	Ideal rising	96	456	816	1176	1536	1896	2256	2616
	Real rising	101.9	461.9	821.2	1181.2	1541.2	1901.2	2260.5	2621.2
	Ideal falling	276	636	996	1356	1716	2076	2436	2796
	Real falling	279.1	639.8	998.4	1357.7	1719.1	2077.7	2438.4	2799.1
11	Ideal rising	336	696	1056	1416	1776	2136	2496	2856
	Real rising	338.2	696.8	1056.8	1417.5	1777.5	2137.5	2497.5	2858.2
	Ideal falling	156	516	876	1236	1596	1956	2316	2676
	Real falling	154.7	514.7	874.7	1234.7	1596.1	1954.7	2314.7	2676.1
12	Ideal rising	216	576	936	1296	1656	2016	2376	2736
	Real rising	217.2	577.2	936.5	1295.1	1655.8	2016.5	2377.2	2737.2
	Ideal falling	36	396	756	1116	1476	1836	2196	2556
	Real falling	33.7	394.4	753	1113	1474.4	1834.4	2195.1	2554.4
13	Ideal rising	48	408	768	1128	1488	1848	2208	2568
	Real rising	48.5	407.8	767.8	1127.8	1485.7	1846.4	2207.8	2568.5
	Ideal falling	228	588	948	1308	1668	2028	2388	2748
	Real falling	225	584.3	945	1303.6	1663.6	2025	2385.7	2745.7
14	Ideal rising	288	648	1008	1368	1728	2088	2448	2808
	Real rising	294.6	655.3	1015.3	1374.6	1733.9	2094.6	2454.6	2813.2
	Ideal falling	108	468	828	1188	1548	1908	2268	2628
	Real falling	112.5	472.5	832.5	1191.1	1550.4	1911.8	2271.1	2631.8
15	Ideal rising	168	528	888	1248	1608	1968	2328	2688
	Real rising	170.8	528.7	888.7	1249.4	1610.1	1970.1	2330.1	2690.8
	Ideal falling	348	708	1068	1428	1788	2148	2508	2868
	Real falling	347.3	707.3	1068	1427.3	1788.7	2147.3	2507.3	2868.7

Table 5.2: Locations of the rising and falling edges of the square waves of the sensors for one mechanical rotation. Sensor 9 to 15

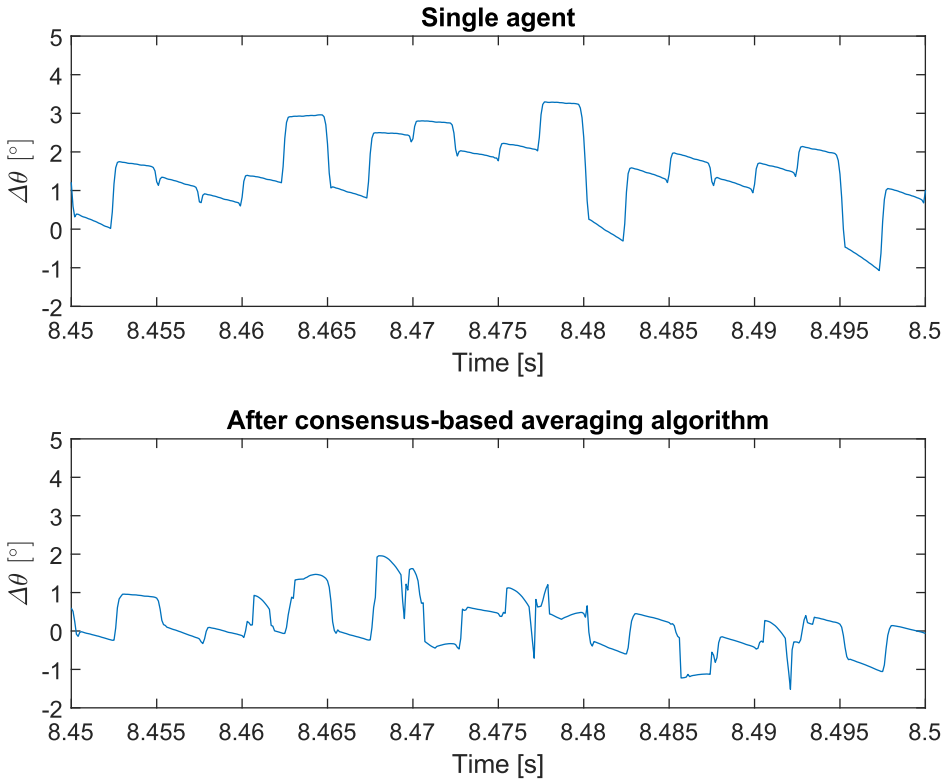


Figure 5.1: Simulation in regime at a mechanical speed of 500 rpm with real sensor outputs. The upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm.

positions. Therefore, one could link this odd result to the misalignment of the sensors. However, an explanation why *dev* is so high only at this speed is not found. Also in the measurements of Chapter 4, one odd result was obtained (module shutdown 500 rpm: neighbour). A simulation of the specific situation with the real sensor positions did not lead to the same unexpected results. So chances are that a mistake was made during the measurement of that situation.

5.4 Conclusion

The measurements from the previous chapters showed that the error with the real rotor position is larger than expected from simulations. The hypothesis is that this is caused by misalignment of the sensors. This hypothesis is tested by measuring the real output

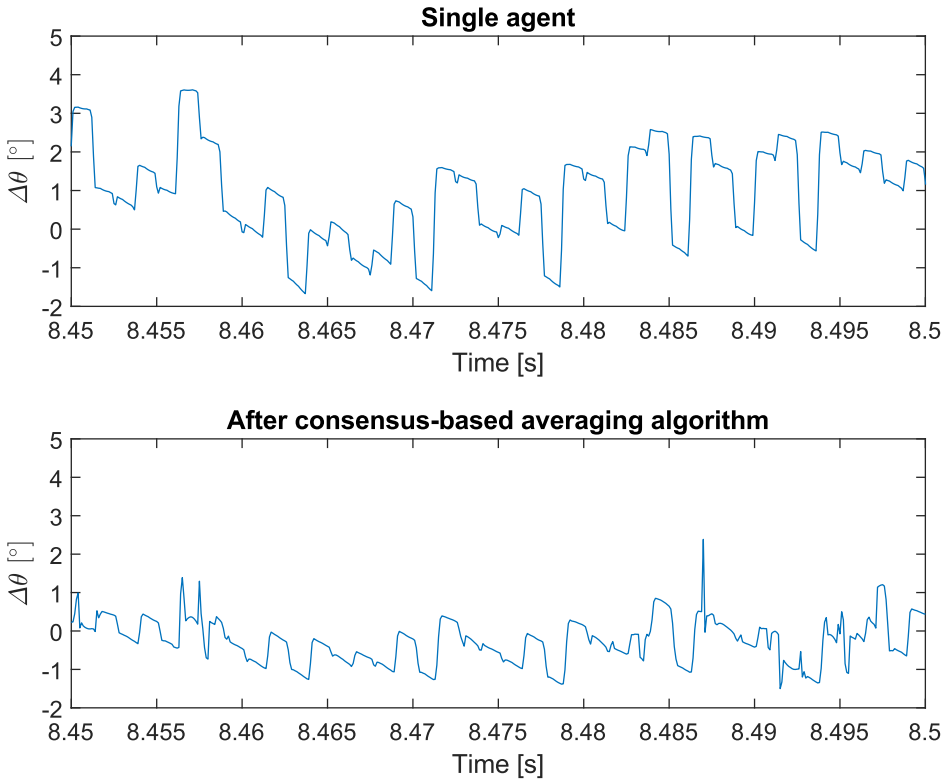


Figure 5.2: Simulation in regime at a mechanical speed of 1000 rpm with real sensor outputs. The upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm.

Table 5.3: Measure for the deviation from the real rotor position for the simulations with sensor misalignment.

	Deviation *dev* [electr. rad]	
Simulation	Single Agent	After Consensus Algorithm
Regime: 500 rpm	432	235
Regime: 1000 rpm	897	275
Regime: 1500 rpm	671	308

of the sensors and using this information for simulations. The result of these simulations showed that the misalignment of the sensors does lead to larger deviations and explain for a large part the larger deviations of the measurements in comparison to the simulations. The authors of [24] propose a self calibrating technique to mitigate the effect of misaligned

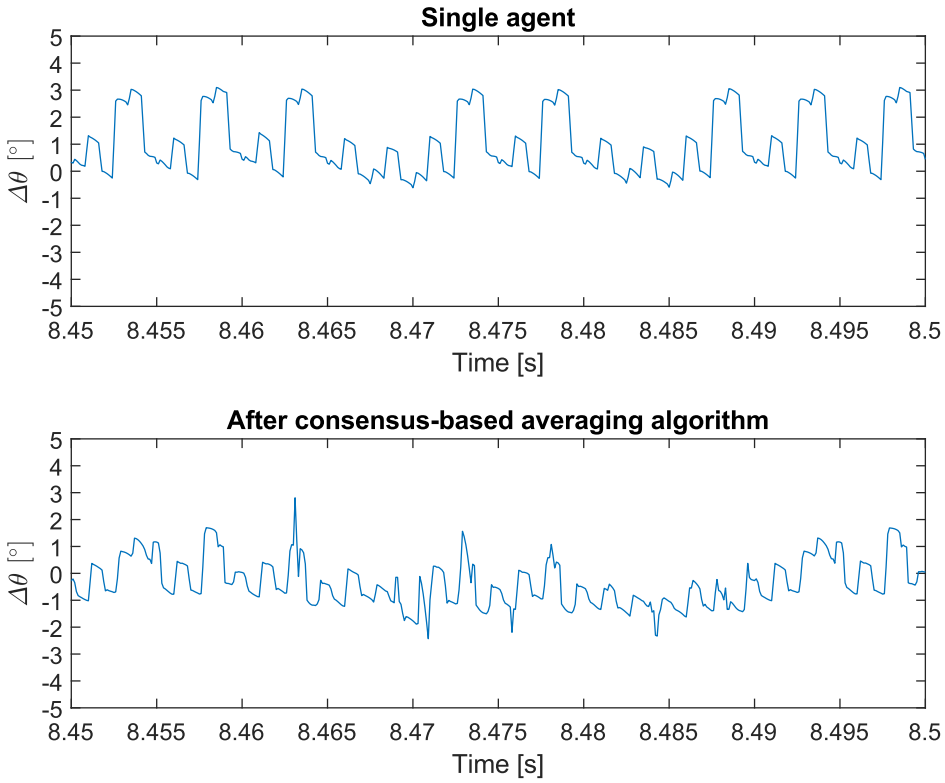


Figure 5.3: Simulation in regime at a mechanical speed of 1500 rpm with real sensor outputs. The upper figure shows the fault $\Delta\theta$ of a single agent. The lower figure shows the fault $\Delta\theta$ after the consensus-based averaging algorithm.

sensors. The technique is based on an analysis of the current spectrum of the DC link, thus in order to use this technique, a current sensor needs to be added to the modules in the stator.

Chapter 6

Conclusion and Further Research

6.1 Conclusion

The goal of this master's dissertation was to make a decentralised position measurement of the rotor in a modular motor drive (MMD). A MMD has a stator that is split up into modules. Using a distributed control strategy, a single centralised controller for the whole drive can be replaced by segmented control units. This makes the control of the drive fault tolerant. Up until now, the modules all receive the rotor position as an input from one single encoder or resolver. This means that the position measurement of the rotor is still a single point of failure. In this dissertation, a decentralised strategy to measure the position of the rotor using cheap and robust Hall sensors was proposed. The concept includes that every module should be equipped with a binary Hall sensor. Every combination of three Hall sensors is called an agent and is able to track the rotor position using a vector-tracking observer (VTO). This position measurement of one agent is further improved by using information from other agents (the neighbours of the former agent) in a consensus-based averaging algorithm. To make this position measurement concept robust and fault tolerant, a fault detection algorithm was introduced that enables the agents to exclude faulty agents from the consensus-based averaging algorithm.

In Chapter 2, the position estimation by one agent, consisting of three modules, is elaborated. It is shown that based on the input of three Hall sensors a vector-tracking observer is able to give an accurate estimate for the position of the rotor. This estimation is then further improved by using variable gains for the PID controller of the observer and by decoupling additional harmonics at the input of the control loop. This algorithm was simulated for different speeds and good position measurements were obtained.

Usually, a MMD consists of more than three modules and thus it has more than three agents. In Chapter 3, a consensus-based averaging algorithm was proposed to exploit the fact that multiple measurements of the same rotor position are available. A distributed algorithm is used to estimate the average of the estimates of the agents. One difficulty is that an agent only communicates with its neighbour, however, it is expected to estimate the average of all the estimates of the agents in the drive. The proposed consensus algorithm uses two communication steps to overcome this issue. The consensus algorithm was simulated and also tested on a test setup. The results of these simulations and measurements show that the consensus-based averaging algorithm leads to better estimates for the rotor position than a single agent's estimates based on the VTO described in Chapter 2.

When using information from all the modules in the MMD, it is important that every module communicates the correct signals. In Chapter 4, a fault detection algorithm was introduced to enable agents to recognise whether one of its neighbours or the agent itself is faulted. This fault detection algorithm was tested for two situations. The first situation is the breakdown of one Hall sensor. The agent with the faulted sensor has a position estimation of the rotor based solely on the VTO that is not usable to control the MMD. However, using information from its neighbours and excluding its own measurement from the consensus algorithm, the agent obtains an excellent position measurement of the rotor. In the second fault situation, the agent has no own estimate of the rotor position anymore, also the output communication to the neighbours is interrupted. It was shown that also in this situation, the faulty agent is quickly recognised by the fault detection algorithm and that, by the input communication from its neighbours, an excellent position estimate was obtained.

Chapter 5 discusses the influence of sensor misalignment on the measurements to test the algorithms. It is concluded that a great part of the difference in accuracy of the position estimations between the simulations and the measurements is due to sensor misalignment.

By using Hall sensors and a vector-tracking observer, the position measurement of the rotor in a MMD is now decentralised. Therefore it is not a single point of failure anymore. The estimated rotor position is improved by introducing a consensus-based averaging algorithm. A fault detection algorithm makes the consensus-based averaging algorithm fault tolerant.

6.2 Further Research

Further improvement to this research can be made on three different domains.

The first improvement that could be made is the scalability of the consensus-based averaging algorithm. In this dissertation, the average output of five agents is estimated by using two communication steps. When more agents are involved, more communication steps would be necessary. This would need a prediction step that goes further in the future than two timesteps. Any deviation in speed estimation from the actual speed of the rotor would lead to larger faults in position estimation than when the prediction is for less timesteps into the future. A solution could be not to involve every agent of the drive in the consensus-based averaging algorithm of a given agent. Or a dynamic average consensus algorithm from [19] could be improved as such, that it converges fast enough to track the average at high speed. These algorithms are less dependent on the number of agents. The convergence speed was dependent on the communication frequency of the agents. If this frequency could be increased, dynamic average consensus algorithms could probably be used in the future.

Furthermore, the fault detection algorithm is only simulated and tested for the case of one fault. Further research has to show what happens if multiple sensors or agents break down. Also, a comparative study between different fault detection algorithms could be conducted to find the optimal algorithm for this specific application.

Lastly, further research could be conducted on how to mitigate the misplacement of the Hall sensors. In literature, there are some methods proposed to reduce the effect of misaligned sensors. For example in [24], as was mentioned in Chapter 5. New methods or a comparative study between the existing methods could be two meaningful additions to the research into rotor position estimation based on low-resolution sensors.

Appendix A

Code Simulations DAC

A.1 Introduction and Credits

```
clear all
close all
clc

% S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch and S. Martinez,
"Tutorial on Dynamic Average Consensus: The Problem, Its Applications,
and the Algorithms,"
in IEEE Control Systems Magazine,
vol. 39, no. 3, pp. 40-72, June 2019, doi: 10.1109/MCS.2019.2900783.
% -> robust accelerated dynamic average consensus algorithm (31), p. 62
% MATLAB code by Lynn Verkroost
```

A.2 Generation of Angle Waveforms

```
Omega_mech = 1500*2*pi/60; % [mech. rad/s], mechanical speed
p = 8; % [-], number of pole pairs
omega_el = Omega_mech*p; % [el. rad/s], electrical speed

n = 5; % [-], number of subgroups

fs = 10e3; % [Hz], sampling frequency
```

```
Ts = 1/fs; % [s], sampling period

Tsim = 2*pi/omega_el*20; % [s], simulation time (5 electrical periods)
time = 0:Ts:Tsim; % [s], time

thetas = zeros(n,size(time,2)); % [el. rad], angle measurement
                                of subgroups

for i = 2:size(time,2)
    thetas(:,i) = mod(thetas(:,i-1) + omega_el*Ts,2*pi);
    % angle = discrete integral of speed, in interval [0,2*pi)
end

omega_dist = 10*omega_el;
disturbance = 5*pi/180*[cos(omega_dist*time);
    cos(omega_dist*time-2*pi/5); cos(omega_dist*time-2*2*pi/5);
    cos(omega_dist*time-3*2*pi/5); cos(omega_dist*time-4*2*pi/5)];
% [el. rad], disturbances on angle measurements

thetas_meas = thetas + disturbance;
% [el. rad], disturbed angle measurements of subgroups

thetas_meas_av = mod(mean(thetas_meas),2*pi);
% [el. rad], average of the disturbed angle measurements of the subgroups

thetas_meas = mod(thetas_meas,2*pi);
% [el. rad], disturbed angle measurements of subgroups
```

A.3 Dynamic Average Consensus

```
delta = 1/2;
A = [0 delta 0 0 delta;
    delta 0 delta 0 0;
    0 delta 0 delta 0;
    0 0 delta 0 delta;
```

```
    delta 0 0 delta 0]; % adjacency matrix
Dout = eye(n)*2*delta; % out-degree matrix
L = Dout - A; % Laplacian matrix

lambdas = sort(eig(L)); % eigenvalues of the Laplacian matrix
lambda2 = lambdas(2);
lambdan = lambdas(end);
lambdar = lambda2/lambdan;

rho0 = (6-2*sqrt(1-lambdar)+lambdar-4*sqrt(2-2*sqrt(1-lambdar)+lambdar))
    /(2+2*sqrt(1-lambdar)-lambdar); % see Table 3, p.63
rho1 = (-3-2*sqrt(1-lambdar)+lambdar+2*sqrt(2+2*sqrt(1-lambdar)-lambdar))
    /(-1-2*sqrt(1-lambdar)+lambdar); % see Table 3, p.63
if (0 < lambdar) && (lambdar <= 2*(sqrt(2)-1)) % see Table 3, p.63
    rho = rho0;
else
    rho = rho1;
end

ki = (1-rho)^2/lambda2; % see Table 3, p.63
kp = (2+2*sqrt(1-lambdar)-lambdar)*ki; % see Table 3, p.63

cos_meas = cos(thetas_meas); % cosine of the measured angles
sin_meas = sin(thetas_meas); % sine of the measured angles

cos_av = zeros(n,size(time,2));
% average of the cosine of the angle measurements
    by means of dynamic average consensus
sin_av = zeros(n,size(time,2));
% average of the sine of the angle measurements
    by means of dynamic average consensus
p_cos = zeros(n,size(time,2)); %
p_sin = zeros(n,size(time,2)); %
q_cos = zeros(n,size(time,2)); %
q_sin = zeros(n,size(time,2)); %
thetas_av = zeros(n,size(time,2));
```

```
% [el. rad], average of the angle measurements
    by means of dynamic average consensus

for it = 2:size(time,2)-1
% dynamic average consenus algorithm (31), p. 62
    q_cos(:,it+1) = 2*rho*q_cos(:,it) - rho^2*q_cos(:,it-1) +
                    kp*L*(cos_av(:,it)+p_cos(:,it));
    p_cos(:,it+1) = (1+rho^2)*p_cos(:,it) - rho^2*p_cos(:,it-1)
                    +  ki*L*cos_av(:,it);
    cos_av(:,it+1) = cos_meas(:,it+1) - q_cos(:,it+1);
    q_sin(:,it+1) = 2*rho*q_sin(:,it) - rho^2*q_sin(:,it-1) +
                    kp*L*(sin_av(:,it)+p_sin(:,it));
    p_sin(:,it+1) = (1+rho^2)*p_sin(:,it) - rho^2*p_sin(:,it-1)
                    + ki*L*sin_av(:,it);
    sin_av(:,it+1) = sin_meas(:,it+1) - q_sin(:,it+1);
    for i = 1:n
        thetas_av(i,it+1) = angle(cos_av(i,it+1)+1i*sin_av(i,it+1));
        if thetas_av(i,it+1) < 0
            thetas_av(i,it+1) = thetas_av(i,it+1) + 2*pi;
        end
    end
end
```

Bibliography

[1] Giovanni Calzo, Gaurang Vakil, B.C. Mecrow, Simon Lambert, Tom Cox, Chris Gerada, Mark Johnson, and Robert Abebe. Integrated motor drives: State of the art and future trends. *IET Electric Power Applications*, 10, 04 2016.

[2] Thomas M. Jahns and Hang Dai. The past, present, and future of power electronics integration technology in motor drives. *CPSS Transactions on Power Electronics and Applications*, 2(3):197–216, 2017.

[3] A. Shea and T. M. Jahns. Hardware integration for an integrated modular motor drive including distributed control. *2014 IEEE Energy Conversion Congress and Exposition (ECCE)*, pages 4881–4887, 2014.

[4] Lukas Lambertz, Rainer Marquardt, and Anna Mayer. Modular converter systems for vehicle applications. In *2010 Emobility - Electrical Power Train*, pages 1–6, 2010.

[5] Tatsuo Morita, Satoshi Tamura, Yoshiharu Anda, Masahiro Ishida, Yasuhiro Uemoto, Tetsuzo Ueda, Tsuyoshi Tanaka, and Daisuke Ueda. 99.3 percent efficiency of three-phase inverter for motor drive using gan-based gate injection transistors. In *2011 26th Annual IEEE Applied Power Electronics Conference and Exposition, APEC 2011*, Conference Proceedings - IEEE Applied Power Electronics Conference and Exposition - APEC, pages 481–484, May 2011.

[6] Staffan Norrga, Lebing Jin, Oskar Wallmark, Anna Mayer, and K. Ilves. A novel inverter topology for compact ev and hev drive systems. pages 6590–6595, 11 2013.

[7] Jordi Van Damme, Lynn Verkroost, Hendrik Vansompel, Frederik De Belie, and Peter Sergeant. A holistic dc link architecture design method for multiphase integrated modular motor drives. In *2019 IEEE International Electric Machines Drives Conference (IEMDC)*, pages 1593–1598. IEEE, 2019.

[8] N. R. Brown, T. M. Jahns, and R. D. Lorenz. Power converter design for an integrated modular motor drive. In *2007 IEEE Industry Applications Annual Meeting*, pages 1322–1328, 2007.

[9] Mario J. Duran and Federico Barrero. Recent advances in the design, modeling, and control of multiphase machines part ii. *IEEE Transactions on Industrial Electronics*, 63(1):459–468, 2016.

[10] L. Verkroost, H. Vansompel, F. De Belie, and P. Sergeant. Distributed control strategies for modular permanent magnet synchronous machines taking into account mutual inductances. In *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 66–71, 2020.

[11] Hendrik Vansompel, Frederik De Belie, and Jan Melkebeek. Improving the rotor position estimation in permanent-magnet synchronous machines with a low-resolution position sensor. In *2010 XIX international conference on Electrical Machines (ICEM)*, page 6. IEEE, 2010.

[12] H. Vansompel. Regeling van een permanente-magneet-bekrachtige synchrone machine met een positiesensor met lage resolutie. *Master thesis to obtain the degree of Master of Science in Electromechanical Engineering*, 2009.

[13] Seung-Ki Sul, Yong-Cheol Kwon, and Younggi Lee. Sensorless control of ipmsm for last 10 years and next 5 years. *CES Transactions on Electrical Machines and Systems*, 1(2):91–99, 2017.

[14] Yong-Cheol Kwon, Seung-Ki Sul, Noor Aamir Baloch, Shinya Morimoto, and Motomichi Ohto. Design, modeling, and control of an ipmsm with an asymmetric rotor and search coils for absolute position sensorless drive. *IEEE Transactions on Industry Applications*, 52(5):3839–3850, 2016.

[15] Qinan Ni, Ming Yang, Shafiq Ahamed Odhano, Mi Tang, Pericle Zanchetta, Xiaosheng Liu, and Dianguo Xu. A new position and speed estimation scheme for position control of pmsm drives using low-resolution position sensors. *IEEE Transactions on Industry Applications*, 55(4):3747–3758, 2019.

[16] Nicola Bianchi, Emanuele Fornasiero, and Silverio Bolognani. Effect of stator and rotor saturation on sensorless rotor position detection. *IEEE Transactions on Industry Applications*, 49(3):1333–1342, 2013.

[17] Luca Bascetta, Gianantonio Magnani, and Paolo Rocco. Velocity estimation: Assessing the performance of non model-based techniques. In *2007 IEEE/ASME international conference on advanced intelligent mechatronics*, pages 1–6, 2007.

[18] M. C. Harke, G. De Donato, F. Giulii Capponi, T. R. Tesch, and R. D. Lorenz. Implementation issues and performance evaluation of sinusoidal, surface-mounted pm machine drives with hall-effect position sensors and a vector-tracking observer. *IEEE Transactions on Industry Applications*, 44(1):161–173, 2008.

[19] S. S. Kia, B. Van Scoy, J. Cortes, R. A. Freeman, K. M. Lynch, and S. Martinez. Tutorial on dynamic average consensus: The problem, its applications, and the algorithms. *IEEE Control Systems Magazine*, 39(3):40–72, 2019.

[20] B. Van Scoy, R. A. Freeman, and K. M. Lynch. Design of robust dynamic average consensus estimators. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 6269–6275, 2015.

[21] Filippo Savi, Giampaolo Buticchi, Chris Gerada, Pat Wheeler, and Davide Barater. Information technologies for distributed machine drives: An overview. In *2019 IEEE International Electric Machines Drives Conference (IEMDC)*, pages 1805–1809, 2019.

[22] H. Bai, R. A. Freeman, and K. M. Lynch. Robust dynamic average consensus of time-varying inputs. In *49th IEEE Conference on Decision and Control (CDC)*, pages 3104–3109, 2010.

[23] J. D. Peterson, T. Yucelen, J. Sarangapani, and E. L. Pasiliao. Active-passive dynamic consensus filters with reduced information exchange and time-varying agent roles. *IEEE Transactions on Control Systems Technology*, 28(3):844–856, 2020.

[24] Dimitrios A. Papathanasopoulos, Dionysios V. Spyropoulos, Epaminondas D. Mitronikas, and Athanasios D. Karlis. Commutation angle self-calibrating technique for brushless dc motor drives with defective hall-effect position sensors. In *2020 International Conference on Electrical Machines (ICEM)*, volume 1, pages 1301–1307, 2020.