Qualitative diagnostic model for sensor network assessment, applied to wind turbines

August 2008
Wibert Huberts
Department of BioMechanical Engineering
Delft University of Technology
Qualitative diagnostic model for sensor network assessment, applied to wind turbines

Wibert Huberts
August 2008

Supervisors

Prof. T. Tomiyama
Delft University of Technology

E. Echavarria
Delft University of Technology
Preface

This report is the result of a Master thesis carried out at the section Intelligent Mechanical Systems (department of BioMechanical Engineering) of the Delft University of Technology. The section Intelligent Mechanical Systems aims at education and research in design of mechanical systems that exhibit non-traditional, advanced features (e.g. Evolvability, Adaptability, Reconfiguration, and Self-Maintenance). Research is performed to gain more knowledge in reliability and reconfiguration of offshore wind turbines.

I would like to thank Erika Echavarria for her guidance and support throughout my Master thesis. I would like to thank André Bos from Science and Technology, for letting me use their software to test my qualitative models. I would like to thank Prof. Tetsuo Tomiyama for the social life in the form of dinners and barbeques as well for his guidance during my whole Master of Science education.

I would like to thank my girl friend for her support and help with writing during my academic education. Last but certainly not least, I would like to thank my friends and fellow students Frank, Henk, Stephan, Henk-Jan, Jurgen, Richard, An and Wouter for the pleasant time, discussions, and encouragement during my study.

Delft, August 2008
Wibert Huberts
Summary

In the last 25 years, the size of the largest commercial wind turbines has increased from approximately 50 kW to 3 MW, with machines up to 5 MW under design (Manwell et al., 2006). In these years, hundreds of sensors have been added. This high number of sensors leads to a high complexity of the wind turbine. As a result, wind turbines suffer more and more under frequent short stops.

In order to evaluate the need of a particular sensor, a tool is needed that gives more insight in the whole sensor network. This tool should give more insight in the relations between the sensors and their values. Some sensors can become obsolete because other sensors are able to measure these values.

The goal of this research is to come up with a tool that can assess the sensor network. In order to find such a tool, more insight is needed in the sensors on the wind turbine and what the criteria are of an optimal sensor network.

In literature, 145 sensors at different locations are found in the wind turbine. These sensors together with the reasoning system form the sensor network. The sensor network is responsible for supplying the controllers, protection system, and condition monitoring system with the appropriate and correct information. The reasoning system uses the redundancies in the sensor network to make an as good as possible representation of the physical world. These redundancies are:

- Physical laws dictate the relation between physical variables
- Operation states give the expected variable combinations
- Parallel sensors should measure the same value
- Failure modes help to interpreted unhealthy behaviour
- Expected sensor values according to control signals

There are four criteria on which an optimal sensor network can be assessed. First, the sensor network for a wind turbine must provide the physical variables for control, protection system, and condition monitoring system. Second, the sensor network must provide redundant information for fault diagnosis. Third, the sensor network must have redundant information to have a sufficient high chance of detecting limit values. As long as variables do not reach their limit value, it is safe to continue operation. Fourth, the sensor network has as few sensors as possible in order to keep costs for designing, building and maintaining the sensor network low.

The third criterion is assessed with a qualitative diagnostic model (QDM), which is a promising tool for fault detection. Of a single pitch system, a qualitative model is build to test if a QDM with an additional health variable is capable to detect situations where possible limit values remain undetected. The results are promising. The output shows all the situations (component failure combinations) were a limit value could remain undetected.
# Table of contents

Preface ........................................................................................................ iii
Summary ....................................................................................................... v
Table of contents ........................................................................................ vii
List of figures .............................................................................................. viii
List of tables ............................................................................................... ix

1 Introduction .......................................................................................... 1
2 Sensors in wind turbines ......................................................................... 3
  2.1 Principle of operation of sensors ....................................................... 3
  2.2 Reasons to put sensors in the wind turbine ...................................... 6
    2.2.1 Sensors for control ................................................................... 6
    2.2.2 Sensors for limit detection ......................................................... 7
    2.2.3 Sensors for condition monitoring .............................................. 9
  2.3 Sensors in the wind turbine according to literature ......................... 11
    2.3.1 Sensors hub ............................................................................ 11
    2.3.2 Sensors gearbox .................................................................... 13
    2.3.3 Sensors yaw drive mechanism ................................................ 15
    2.3.4 Sensors hydraulic group for mechanical and yaw brakes .......... 16
    2.3.5 Sensors fixed speed induction generator .................................. 18
    2.3.6 Sensors control cabinet .......................................................... 20
    2.3.7 Sensors main bearing .............................................................. 20
    2.3.8 Other sensors ......................................................................... 21
    2.3.9 Sensor overview .................................................................... 21
3 Assessing a sensor network .................................................................. 23
  3.1 Sensor network .............................................................................. 23
    3.1.1 Sensor network redundancies ............................................... 24
    3.1.2 Definition of optimal sensor network ...................................... 26
  3.2 Qualitative diagnostic model (QDM) as a tool to assess a sensor setting 28
    3.2.1 Explanation of QDM ............................................................... 28
    3.2.2 QDM and the chance of detecting limit values ......................... 29
4. Example: assessing sensor network of the pitch mechanism ............... 31
  4.1 Construction of QDM .................................................................... 31
    4.1.1 Variable declaration ............................................................... 32
    4.1.2 Healthy behaviour .................................................................. 35
    4.1.3 Sensor network redundancies ............................................... 35
    4.1.4 Additional health variable ..................................................... 45
  4.2 Construction of post processor ......................................................... 46
  4.3 Experiment with the QDM ............................................................... 49
5. Conclusions ......................................................................................... 53
6 Recommendations .................................................................................. 55

References ............................................................................................... 57
Appendices ............................................................................................... 59
Appendix 1: Script of the QDM model ....................................................... 61
Appendix 2: Script of the post processor .................................................. 74
Appendix 3: Experiment inputs and outputs ............................................. 80
List of figures

Figure 1 The first automatically operating wind turbine (12 kW) for electricity generation (1888) .................................................................................................................. 1
Figure 2 A 5 MW wind turbine prototype under development by REpower Systems (2008) ................................................................. 1
Figure 3 Hub (Harakosan) ......................................................................................................................................................................... 12
Figure 4 Electrical motor in the hub (Burton, 2002) .................................................................................................................................................................................. 12
Figure 5 Sensor overview in the hub .................................................................................................................................................................................. 13
Figure 6 Sensor overview of the gearbox ......................................................................................................................................................... 14
Figure 7 Typical arrangement of Yaw Bearing, Yaw Drive and Yaw Brake (Burton, 2002) ................................................................. 16
Figure 8 Sensor overview of hydraulic group for mechanical and yaw brakes ......................................................................................... 17
Figure 9 Sensor overview of the hydraulic system of the generator ........................................................................................................... 19
Figure 10 Cross section of the generator ......................................................................................................................................................... 19
Figure 11 Induction generator (DWAI, 2003), green cross section is shown in Figure 9, red cross section is shown in Figure 10 .................................................................................................................. 19
Figure 12 Sensor overview of the other sensors ......................................................................................................................................................... 22
Figure 13 Sensor overview of wind turbine ................................................................................................................................................. 22
Figure 14 Definition of sensor network ......................................................................................................................................................... 23
Figure 15 Single pitch system .............................................................................................................................................................................. 31
Figure 16 The healthy behaviour of a sensor (thermometer on the motor) ................................................................................................. 35
Figure 17 When the brake is on, the rotational speed of the motor will decelerate or stay at zero .................................................................................................................................................................................. 39
Figure 18 Four steady states and seven transition states for the pitch mechanism ................................................................................................................. 40
Figure 19 The additional health variable in the syntax of the model ......................................................................................................................... 46
Figure 20 A print screen of the generated rough output ............................................................................................................................................. 47
Figure 21 A print screen of the post processor output ............................................................................................................................................. 48
# List of tables

Table 1: Most used sensors with their principle of operation .................................................. 4
Table 2: Four main operation states ......................................................................................... 6
Table 3: Motor and generator failures ..................................................................................... 10
Table 4: List with electrical equipment and their sensors ....................................................... 20
Table 5: Failure modes of an electrical motor ......................................................................... 25
Table 6: Post processed output example of the QDM ............................................................... 30
Table 7: Used symbols ............................................................................................................. 32
Table 8: Relations for rotational speed, rotational acceleration and current ......................... 36
Table 9: Additional rules for deceleration ............................................................................... 36
Table 10: Rotational speed, vibrations in the nacelle and pitch health determine the vibrations in the motor ........................................................................................................... 37
Table 11: Vibrations in the motor determine the rotational speed ........................................... 37
Table 12: Calculation table to calculate $T_{\text{motor}}$ when the motor is stopped ................. 38
Table 13: Calculation table to calculate $T_{\text{motor}}$ in operation with $T_{\text{motor}} = \text{zero}$ ........ 38
Table 14: Calculation table to calculate $T_{\text{motor}}$ in operation with $T_{\text{motor}} = \text{low}$ .......... 38
Table 15: Calculation table to calculate $T_{\text{motor}}$ in operation with $T_{\text{motor}} = \text{norm}$ ....... 39
Table 16: Calculation table to calculate $T_{\text{motor}}$ in operation with $T_{\text{motor}} = \text{red}$ .......... 39
Table 17: Calculation table to calculate $T_{\text{motor}}$ when $w_{\text{motor}} = \text{zero}$ ...................... 39
Table 18: All the variations of controller signal, timer and current operation state with their healthiness ...................................................................................................................... 42
Table 19: Output of the 6 input files ......................................................................................... 50
Table 20: Summary of the output ............................................................................................ 51
1 Introduction

In the last 25 years, the size of the largest commercial wind turbines has increased from approximately 50 kW to 3 MW, with machines up to 5 MW under design (Manwell et al., 2006. (See Figure 1 for the first automatically operating wind turbine and Figure 2 for a 5 MW prototype.) In these years there have been hundreds of sensors added to the wind turbines.

These sensors provide the controllers with the current conditions of the wind turbine. However, it looks like that the number of sensors are becoming far too many and are reducing the reliability (resistance to failure) of the sensor network. This high number of sensors leads to a complex sensor network. As a result, wind turbines suffer more and more under frequent short stops caused by malfunctioning sensors. Placing wind turbines in remote areas (e.g. offshore) increases the problem even more. Reaching those wind turbines for quick repairs is sometimes (with heavy weather) even impossible.

Equipping a system with more sensors than strictly needed does not necessarily lead to less reliability. Superfluous sensors lead to redundant sensor outputs. When these redundancies are found, they can be used to detect failures and make it even possible to continue wind turbine operation when one or more sensors are broken. For instance, a wind vane on a wind turbine measures the angle of the misalignment between the wind direction and the nacelle. This angle changes when the yaw mechanism starts rotating the nacelle (e.g. to get the rotor blades in the direction of the wind). One can say, that the wind vane directly measures misalignment with the wind direction and indirectly measures the rotational speed of the nacelle. When the yaw mechanism is in use, the wind vane output can be compared with the rotational sensor of the yaw mechanism, to check if both sensors are still working correctly. When the rotational sensor of the yaw system gets broken, it might be even possible to rely on the wind vane to
measure roughly the rotations of the yaw system, as long as this rotational sensor is not replaced / repaired.

Improving the sensor network is not straightforward. There is a clear trade off between reducing the number of sensors and having redundant sensor information. A tool is needed that is aware of the redundancies in the sensor network and that can show which sensors are really needed.

The goal of this research is to come up with a tool that can assess the sensor network. In order to find such a tool, the following method is used.

Firstly, an overview of the implemented sensors in a wind turbine needs to be built. Knowing what the principles of operation are, which variables are needed for the wind turbine operation, and where the sensors are placed in the wind turbine gives a better understanding of the sensor network in the wind turbine.

Secondly, the redundancies among the individual sensors are of great importance for the reliability of the sensor network. Different sources of sensor network redundancies should be defined in order to know where possible redundancies can be found.

Thirdly, a qualitative diagnostic model (QDM) has shown in previous research (Echavarria et al., 2008; Huberts, 2008) to be a promising tool for wind turbine diagnosis. The QDM uses as input the measured values of the sensors and the signals from the controllers. It uses the network redundancies to detect faulty components. The strength of the QDM is twofold. First, the model of the QDM is relatively easy to build because of the qualitative statements. Its statements are represented as humans perceive the world, by common sense and values like negative, positive, normal, high, etc. This makes it possible to build models of complex systems relatively easily. Second, the reasoning engine can search through all possible variable combinations because of the qualitative variables. Possibly the QDM can combine these reasoning capabilities with sensor network redundancies to serve as a tool to assess the sensor network.

Fourthly, when a tool is created for sensor network assessment, it should be presented in an example to show the capabilities of this tool.

This report consists of 6 chapters. In the second chapter an overview of sensors in the wind turbine is presented. Then, the sensors working principle, wherefore the sensors are placed, and sensor locations in the wind turbine are discussed. In chapter 3 a definition is given for the sensor network and sensor network redundancies. Also a concept is built to assess a sensor network with a QDM. An example of a QDM assessment is given in the fourth chapter. The last two chapters are the conclusions and recommendations.
2 Sensors in wind turbines

A complete list of all sensors equipped on a certain wind turbine was not available for this project. Companies were reserved to share this information. Sensor information would tell too much about how their wind turbines operate, it is just too confidential for companies to share this information. The only information that was available is: “There are hundreds of sensors in wind turbines”.

To tackle the problem of high complexity of sensor networks in wind turbines, more understanding about the sensors is needed. In this chapter, an overview of sensors is presented from information that is found in literature. This includes technical descriptions of five wind turbine manufacturers (Enercon, 2005; Gamesa, 2007; Nordex, 1998; Harakosan; Suzlon, 2005), books of wind turbines (Burton et al., 2002; Manell, 2006; Rademakers et al., 1996) as well a sensor technology handbook (Wilson, 2005) and a condition monitoring handbook (Rao, 1996).

The wind turbine that is discussed in this chapter is a general modern wind turbine. It is a horizontal axis wind turbine (HAWT) with an upwind rotor. It is pitch controlled with three independent electrical motors. The (three stage) gearbox and generator are cooled with a fluid that is connected with a heat exchanger at the rear. The generator is a fixed speed induction generator.

In section 2.1 the operation principles of most common sensors in the wind turbine are discussed. In section 2.2 are the sensors that are needed for control (2.2.1), limit detection (2.2.2), and condition monitoring (2.2.3) presented. In the last section (2.3), the main parts of the wind turbine are described in detail, followed by a list of sensors that are in, on, or around this part. For each main part a drawing is made, which shows the locations of the sensors.

2.1 Principle of operation of sensors

A sensor is a device which measures a physical property [Oxford dictionary]. In the wind turbine a human does not read these values. The central controller, the computer of the wind turbine, receives this sensor information. Therefore, the output of the sensor must be an electrical signal.

In this section the different working principles of some sensors are described. It is out of the scope of the project to discuss them all. The most important and most common used sensors are discussed. Only the sensors that measure the physical properties that are available and commonly measured in the wind turbine are described.

The working principle gives insight how a sensor works and how its output can be influenced. As an example, when a sensor makes use of a resistor to measure humidity, it will measure the change of humidity because it influences the resistance of the resistor. However, caution is needed when the temperature of the working environment changes, temperature changes change the resistance of a resistor as well. Another example, a magnetic sensor can be influenced by the magnetic field of the generator when placed at or in the generator. Therefore, it is important to know what a sensor measures and by which physical variables the sensor is influenced.
<table>
<thead>
<tr>
<th>Sensor type</th>
<th>Transducer</th>
<th>Principle of operation</th>
<th>Notes / considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thermometer</td>
<td>Thermistor</td>
<td>temperature -&gt; electrical resistance -&gt; electrical signal</td>
<td>-100 to 300 °C / fast thermal response</td>
</tr>
<tr>
<td>Thermometer</td>
<td>Bi-metal thermostats</td>
<td>temperature -&gt; mechanical -&gt; open or close contact</td>
<td>-85 to 371 °C / less accurate</td>
</tr>
<tr>
<td>Pressure sensor</td>
<td>Piezoresistive pressure</td>
<td>pressure -&gt; force -&gt; electrical resistance -&gt; output voltage</td>
<td>Dominates the pressure sensors marked for many good reasons</td>
</tr>
<tr>
<td>Wind vane</td>
<td>Wind vane</td>
<td>wind direction -&gt; rotational movement</td>
<td></td>
</tr>
<tr>
<td>Wind speed</td>
<td>Anemometer</td>
<td>flow rate -&gt; rotational speed*</td>
<td></td>
</tr>
<tr>
<td>Rotational speed sensor</td>
<td>DC Generator</td>
<td>rotational speed -&gt; induction -&gt; Voltage</td>
<td>Does not require electricity</td>
</tr>
<tr>
<td>Rotational speed sensor</td>
<td>Hall effect magnetic sensor</td>
<td>rotational speed -&gt; magnetic field -&gt; electrical signal</td>
<td>Contact free</td>
</tr>
<tr>
<td>Rotational speed and movement sensor</td>
<td>Incremental optical encoder</td>
<td>rotational movement -&gt; optical signal -&gt; electrical pulses</td>
<td>Measures relative angles</td>
</tr>
<tr>
<td>Rotational movement sensor</td>
<td>Absolute optical encoder</td>
<td>rotational movement -&gt; optical signal -&gt; electrical pulses</td>
<td>Disk is divided into N sections</td>
</tr>
<tr>
<td>Rotational/linear movement sensor</td>
<td>Potentiometer</td>
<td>movement -&gt; electrical resistance -&gt; output voltage</td>
<td>Has significant advantage above other types of accelerometers</td>
</tr>
<tr>
<td>Accelerometers</td>
<td>Differential capacitor</td>
<td>acceleration -&gt; mass of beam moves centre plate away of / closer to differential capacitor -&gt; amplitude of square wave -&gt; voltage signal</td>
<td></td>
</tr>
<tr>
<td>Humidity sensors</td>
<td>Capacitive sensors</td>
<td>humidity -&gt; capacitance change in three layers -&gt; voltage output signal</td>
<td>Wide ranging in use / capable of operating accurately down to 0% relative humidity / contaminants do not cause failure</td>
</tr>
<tr>
<td>Humidity sensors</td>
<td>Resistive</td>
<td>humidity -&gt; resistance change -&gt; voltage</td>
<td>Life expectancy is less than five years / small size / widely used in air conditioning controls</td>
</tr>
<tr>
<td>Over current</td>
<td>Fuse</td>
<td>over current -&gt; heat melts wire -&gt; no current</td>
<td>For one time use</td>
</tr>
<tr>
<td>Over current</td>
<td>Circuit breaker (electromagnet)</td>
<td>over current -&gt; electromagnet activates switch -&gt; no current</td>
<td></td>
</tr>
<tr>
<td>Over current</td>
<td>Circuit breaker (Bi-metal)</td>
<td>over current -&gt; heat -&gt; bi-metal activates switch -&gt; no current</td>
<td></td>
</tr>
<tr>
<td>Over current</td>
<td>Ground fault circuit</td>
<td>constantly monitors the current in a circuit's neutral wire and hot wire</td>
<td>Designed to protect people from electrical shock</td>
</tr>
</tbody>
</table>

*1 The anemometer first changes the wind direction into a rotation movement. This rotational movement can be measured with an incremental optical encoder, an absolute optical encoder, or a Potentiometer.
The most common sensors in the wind turbine are put in Table 1. In the second column the transducer method is shown. When there is more than one common transducer type, multiple transducers are put in the table for the sensor type.

In the third column the principle of operation shows how the physical variable is set in an electrical signal. For example, a Thermistor has as principle of operation “temperature -> electrical resistance -> electrical signal”. This means that the temperature changes the electrical resistance of a resistor. By measuring this resistance an electrical signal is obtained.

For some transducers additional notes / considerations are given.

More detailed information can be found in Wilson (2005).
2.2 Reasons to put sensors in the wind turbine

When is known wherefore a sensor is placed one can determine if a sensor is really needed. The wind turbine needs sensors for its controllers, for its safety system, and for condition monitoring of its components. Controllers are responsible for safe and efficient operation of the wind turbine. When the controllers fail to keep the wind turbine within the limits of safe operation, the safety system comes into action to stop the wind turbine. To predict failures of components there are sensors on the individual components to record their behaviour (condition monitoring).

The central question of this section is: Which variables are needed for control (2.2.1), limit detection (2.2.2) and condition monitoring (2.2.3)?

2.2.1 Sensors for control

The wind turbine reacts on changes of the outside world. The wind turbine itself depends on several physical variables for its energy production. The controller measures these physical variables with sensors to control the operation of the wind turbine.

This section gives a short overview of the physical variables that are needed for wind turbine control. Explained is when and wherefore these physical variables are needed.

Four main operation states are used to describe when these physical variables are needed for control. In Table 2 these four operation states are written down. The operation states are split up in transitional states and stationary states. The turbine may remain in a stationary state for long periods of time, depending on the wind and operating conditions. Other operating states may only be transitional states that are entered during changes from one stationary operating state to another.

<table>
<thead>
<tr>
<th>Operation state</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (stationary)</td>
<td>Waiting for good wind speeds</td>
</tr>
<tr>
<td>2 (transitional)</td>
<td>Start up</td>
</tr>
<tr>
<td>3 (stationary)</td>
<td>Generating electricity</td>
</tr>
<tr>
<td>4 (transitional state)</td>
<td>Shut down</td>
</tr>
</tbody>
</table>

Wind speed is used for operation state chances. The wind speed is needed to know when the wind turbine has to start up and shut down. Configuration changes at generating electricity (operation state 3) depend on the wind speed too. The optimal pitch angle is used between cut in and rated wind speed. Power control by pitching partly out of the wind is used between rated and cut out wind speed.

Wind direction is needed in operation states start up and generating electricity (2 and 3). In operation state start up the rotor blades will be yawed into the wind direction. In operation state generating electricity the rotor blades will follow the wind direction. The wind direction is needed to know in which direction the wind turbine will be able to generate the most electricity.
Rotational speed of the generator is measured for power control. At above rated wind speeds, the pitch angle is in a feedback loop with the rotational speed of the generator (Manwell et al., 2006). This way the rotational speed is kept between limits and maximum power output is achieved.

Number of rotations of the nacelle (with respect to the tower) is a measure for cable twist. This cable twist is caused when the nacelle is yawed around to follow the wind direction. Too much yaw activity in one direction can lead to problems with the cables that are going to the nacelle. Therefore, the yaw activity must be monitored to count the number of rotations in one direction and unwinding must be started in time.

Phase, voltage, and frequency of the grid (grid status) are very important for the generator. The grid dictates the phase of the generated electricity in an induction generator and a sudden change of the phase can damage the generator. When the phase of the grid suddenly changes the generator is quickly disconnected from the grid. The grid status is therefore especially important during generator operation (operation states start up and generating electricity).

Temperature determines if the wind turbine can operate. When the weather conditions are too cold or hot the wind turbine cannot operate and the wind turbine has to stay in operation state waiting for good wind speeds.

Feedback from the components is needed for the controller to check if everything is still working fine. Pitch angles of the blades are checked to know if they all have the same angle, switches on brakes will tell if brakes are really applied, etc. These sensors are all described in section 2.3.

2.2.2 Sensors for limit detection

In a wind turbine the controller is responsible for maintaining the machine operating parameters within their normal limits. However, if for some reason the controller does not keep the wind turbine within these limits there is a protection system. In this section recommended practices from Rademakers et al. (1996) are followed for limit detection. First limit detection by the controller is described. Second, limit detection of the protection system is written down.

2.2.2.1 Limit detection by the controller

The controller is responsible to keep operation between limits. It has sensors that provide the controller with the internal variables of the wind turbine. When a limit is reached the controller will directly shut down the wind turbine.

Below an overview of alarms and their physical variables are given as found in Rademakers et al. (1996).

- Grid
  - Current trafos to measure phase asymmetry
  - Voltage trafos to measure high voltage
  - Voltage trafos to measure low voltage
  - 0-point detector to measure frequency error
  - Number of cut-in/cut out to measure too many cut-in / cut-out at low load
- **Hydraulic system**
  - Pump frequency to measure hydraulic leakage
  - Pump time to measure Valve defect
  - Circuit breaker to measure hydraulic Overload
  - Etc.

- **Brakes**
  - Level switch to measure brake fluid low
  - Brake active and rpm to measure bad disk brake or valve
  - Power and rpm to measure bad brake valve

- **Electrical system**
  - Power after cut-out to measure short circuit in thyristors or welded by-pass switch
  - Temp. on thyristors to measure thyristors overheated
  - PTC in generator windings to measure hot Generator
  - Bypass switch feedback to measure by-pass switch
  - Circuit breaker feedback to measure overload or circuit breaker trip
  - Counter of rotations to measure cable twist

- **Mechanical Systems**
  - Thermal relay, yaw switch feedback to measure overload yaw motor or bad switch
  - Temperature sensor in gear oil to measure hot gear oil
  - Temperature sensor at bearing to measure hot gear bearing

- **Control of sensors and control system**
  - Comparison between rpm-sensors to measure bad rpm-sensors
  - Internal control of software to measure bad controller or program error
  - Yaw time to measure bad wind vane
  - Wind Speed and power to measure bad anemometer
  - All temperature sensors to measure bad temperature sensors

- **Emergency button**
  - Press on to detect an emergency stop request

### 2.2.2.2 Limit detection by protection system

When the controller fails to keep the wind turbine within its limits there is a protection system (also called safety system) that comes into action. The purpose of the protection system is to ensure that, should a critical variable exceed its normal limit as a result of a fault or failure in the wind turbine or the control system, the machine is maintained in a safe condition (Burton et al., 2002). In practise this means that the protection system directly shuts down the wind turbine without inference of the controller (Rademakers et al., 1996).

The critical variables are:
- Rotational speed of the turbine to measure overspeed
- Power output to measure generator overload
- Generator temperature to measure generator overload
- Vibrations to measure excessive vibrations
- Phase, voltage and frequency to measure grid failure
- Too much pressure on the cables (measured by a limit switch in the yaw system) to measure abnormal cable twist
- Activity of the controller (measured by controller watchdog timer) to measure health of controller
- Press on (the emergency button) to measure an emergency stop request

For each parameter it is necessary to set an activation level at which the safety system is triggered. This has to be set at a suitable margin above the normal operating limit to allow for overshooting by the control system, but sufficiently far below the maximum safe value of the parameter to allow scope for the safety system to rein it in.

2.2.3 Sensors for condition monitoring

The emphasis of condition monitoring is on preventive techniques rather than simple detection. A comprehensive description of condition monitoring of wind turbine components could be found in Rao (1996). It is out of the scope of this project to discuss all the condition monitoring options for every component in the wind turbine. In this section a summary of condition monitoring practices is given for the following components: Gearbox, ball/roller bearing, hydraulic systems and electrical motor/generator. These components are described because condition monitoring on these components are most valuable. These components are critical parts of the system, there are good techniques developed to predict their failure and faults, and hydraulic / electrical failure is capable of causing fire which can lead to lost of the whole wind turbine.

2.2.3.1 Gearbox
Most prevalent observables for wear and failure in gearboxes are increasing noise and vibration, generation of abnormal sizes and amounts of metallic debris, and increasing temperature due to increased power losses within the gearbox.

Vibration (and noise) sensing can be done in three ways. First with accelerometers, they measure the vibrations of the gearbox. Shaft bearing looseness and gear mesh misalignment are the result of wear and failure. These bearing looseness and gear mesh misalignment can be measured with velocity sensors. The third option is not widely applied to gear box vibration monitoring. A displacement sensor can detect faults that are manifested as output shaft radial motion.

Metallic debris can be found in the lubrication oil. The best and most complete detection of the debris is done (off line) in a laboratory. However, online detection of the amount of oil debris can be done in several ways. Electromagnetic sensing can detect the debris when it passes through or capture the debris and register that event. A permanent (electric) magnet can attract the particles. Disturbances in the coil or measuring the inductance change of the coil gives the amount of debris. A filter can be used to capture the metallic debris. By measuring the pressure before and after the filter will give the pressure drop. This pressure drop is an indication of the amount of captured metallic debris. Relatively new in metallic debris detection is optical particle detection and image processing.

Temperature sensing is not capable of giving an early warning. Rate of temperature increase and rate of increase of this rate (acceleration) are sometimes useful in detecting the later stages of failure. The oil temperature can be measured as well as the gearbox cap temperature.
2.2.3.2 Ball / Roller bearing

Vibrations are the best early warning for bearings. *Enveloping* is observing the high frequencies. This is a technique to detect early machinery malfunction and deterioration, so the right maintenance can be planned in advance.

2.2.3.3 Hydraulic systems

The presents of dirt in the hydraulic fluid is the single most important aspect controlling the life and reliability of fluid systems. There are many methods of detecting the dirt: Visual inspection, contaminant monitoring (pressure drop at filters), wear debris analysis, energy methods (like power, current, torque and temperature measurements), operational parameters (pressure, flow speed), dynamic measurements (vibration, stress waves, noise spectrum analyses and acoustic emissions). (See also section 2.2.3.1)

Most important to detect failures in the hydraulic system is leakage. Leaks means that there could be problems with high vibrations or that seals are beginning to wear. Visual inspection is a very effective for detecting leakage. Other ways to detect leakage is measuring flow rate, pressure measurements and operation time of the valves and pistons. The last is very easy to measure and is an indication of internal leakage too.

2.2.3.4 Electrical motor and generator

The squirrel-cage induction motor works the same as the squirrel-cage induction generator. In the wind turbine there are electric motors everywhere, at the yaw mechanism, pitch mechanism, fan and hydraulic pumps. The most common failures are described in Table 3.

<table>
<thead>
<tr>
<th>Component</th>
<th>Failure chance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bearings</td>
<td>30%</td>
</tr>
<tr>
<td>Burn out/ thermal trip</td>
<td>30%</td>
</tr>
<tr>
<td>Misalignment and unbalance</td>
<td>30%</td>
</tr>
<tr>
<td>Electrical connections</td>
<td>5%</td>
</tr>
<tr>
<td>Winding and castings</td>
<td>5%</td>
</tr>
</tbody>
</table>

The majority of the failures (bearings, burn out, misalignment and unbalance) are mainly secondary effects. Possible causes for burn out are e.g. high loads, high resistance or high ambient temperature. The failure of bearings may be caused by lack of lubrication. Failure will be induced by excessive loads which reduce the life and cause an early failure. Signs of misalignment or unbalance may indicate installation faults, such as misalignment with the driven equipment or uneven mounting of the motor frame, or manufacturing faults such as an eccentric rotor.

Current running through the motor, temperature of the motor, resistance of the windings and vibrations can all be used for monitoring the electrical motor / generator. These four ways are described in the continuation of this section.

1. Current

The developed torque (T) can be expressed in terms of magnetic flux (Ψ) and current (I) as follows: $T \propto \Psi \cdot I$

Current which varies proportional to \{torque + a constant\}, may indicate gross changes in load. However, it is not very sensitive to the real cause of the faults. The early signs of failure will be lost in the normal variations. This technique can be used to compare currents between phases, which can identify problems in individual phases. A single measurement is not likely
to be very informative, but comparison between identical machines operating under the same conditions can give immediate results.

2. Temperature
The current, which normally passes through the stator and rotor conductors, create heat which must be dissipated. The heat generated is proportional to the square of the current. The induction motor responds to an increased load by slowing down. Because motors are usually supplied with a shaft-mounted fan, which blows ambient air across fins attached to the stator, the cooling fan is slowed down too. This leads to rapid overheats of the machine. Severe damage is often prevented by use of over-current and over-temperature trips. The early signs of failure will be difficult to detect with temperature monitoring.

3. Resistance
High resistance in the windings cause inefficiency. Low resistance indicates short circuits, which lead to high localised currents and high temperatures. High temperature will cause further degradation of the insulation, which makes the short circuit even worse, and rapid failure follows. Since a large proportion of failures occur at start-up, it is logical to test the condition of the motor prior starting.

4. Vibration
Vibration is not a fault itself, but is symptomatic of developing failures, and gives early warnings. The following faults are observable from the vibration data at the relevant frequencies (where $n$ is the rotational speed):

1. Rotating imbalance $n$
2. Looseness $2n, 3n, \ldots$
3. Bent shaft $n$
4. Misalignment $n, 2n$
5. Bearings impact frequencies, noise at high frequency.

Drawback of vibration monitoring is that the analysis is complicated.

2.3 Sensors in the wind turbine according to literature

In this section the main parts of the wind turbine are described in detail, followed by a list of sensors that are in, on, or around this part. For each main part a drawing is made, which shows the locations of the sensors.

2.3.1 Sensors hub

Description:
In the hub the pitch system is located. The pitch system has been designed with a triple redundancy. This means that every single blade has its own drive system, consisting of a motor, a gearbox, a converter and a control system (Suzlon, 2005; Burton et al., 2002). Each blade has its own battery backup to ensure the functionality even when the electrical grid is not available (Suzlon, 2005). Their drive pinions are interlocked with the inner gear wheel of the blade bearings, which support the rotor blades (Suzlon, 2005; Harakosan). The motors are equipped with an internal brake, which holds the blades in safe position between the pitch processes (Suzlon, 2005). There is a redundant blade angle monitoring system at each blade (Enercon, 2005). Very precise control of the pitch on each blade is needed in order to avoid
unacceptable pitch angle differences during normal operation (Burton et al., 2002). Their synchronization is checked by the controller (Harakosan).

The electrical motor is a critical component of the wind turbine. Therefore, it is valuable to condition monitor the temperature, the current and the vibrations of the electrical motor (see also 2.2.3).

**Measured values:**
- Voltage of the battery, battery check to guarantee the working of the battery backup when the electrical grid is not available.
- Pitch angle of each blade, important value for power regulation and the working of the pitch mechanism (including motor).
- Temperature of the motor for overload pitch motor.
- Current flowing through electrical motor, for condition monitoring.
- Vibrations of the electrical motor, for condition monitoring.
- Temperature in the hub, for control.
- Feedback of the internal brake electrical motor.

**Overview:**
In Figure 3 can be seen how the hub looks like without any blades. In Figure 4 a close view of an electrical motor in the hub is shown. For a better understanding a schematic overview of the sensors in the hub is presented in Figure 5. (The sensors in / on the controllers are described in section 2.3.6.)

![Figure 3 Hub (Harakosan)](image1.png)

![Figure 4 Electrical motor in the hub (Burton et al., 2002)](image2.png)
2.3.2 Sensors gearbox

Description:
The gearbox consists of 3 stages: 1 planetary and 2 helical stages convert the low rotational speed into high speed. The gear oil is cooled by a heat exchanger, connected to the cooling system of the generator. Monitoring of the oil temperature ensures that the oil reaches its optimum temperature as fast as possible, and at the same time is kept constantly at the optimum temperature. (Nordex, 1998)

To transport the oil, there is a pump and an auxiliary pump installed (Gamesa, 2007). The oil is filtered after the pump by an oil filter.

Measured values:
- Accelerometers at bearings (Stork-Gears) for bearing / gearbox condition
- Rotational speed both sides of the gearbox
- Bearing cap temperature for bearing / gearbox state
- Pump frequency for: (Rademakers et al., 1996)
  1. Feedback of activation of the pump
  2. State of hydraulic system
- Pressure sensors before and after oil filter for: (Suzlon, 2005)
  1. State of oil filter
  2. State of hydraulic system

Figure 5 Sensor overview in the hub
- Oil temperature for: (Suzlon, 2005; Nordex, 1998)
  - State of gearbox
  - Controlling the oil temperature for optimal temperature
  - State of heat exchanger
- Oil level for detection of low oil level.

**Overview:**
For a better understanding a schematic overview of the sensors in/on/around the gearbox is presented in Figure 6.

![Figure 6 Sensor overview of the gearbox](image)

- P  Pressure sensor (piezoresistive) (4x)
- T  Electronic (oil) thermometer (thermistor) (7x)
- ω  Rotational sensor (optical encoder) (5x)
- Hz  Accelerometers (differential capacitor) (2x)
- L  Oil level switch (electrical bridge) (1x)
2.3.3 Sensors yaw drive mechanism

**Description:**
The nacelle (bedplate) and tower are connected with a slewing (ring ball) bearing which can support axial, radial and moment loads simultaneously. Three yaw drives (consisting of planetary gears and an electrical motor) turn the nacelle. The yaw brake consists of a large disk brake activated by 6 hydraulic brake calibers. (Nordex, 1998; Vestas, 2005; Suzlon, 2005)

The yaw motion is monitored by: counting the rotations of the yaw motor, measuring the temperature for motor overload (Rademakers *et al.*, 1996) and checking the yaw time for plausibility (Enercon, 2005). There is a separated limit sensor that checks the cable twist caused by the yaw (Enercon, 2005; DWIA, 2003). This can be like a pin in a socket that will be pulled out when the cables are turned around each other. It is recognized that motors are a critical part of any system (Rao, 1996). For condition monitoring current and vibrations can be measured as well (see 2.2.3).

In the next section (see 2.3.4) the yaw brakes itself are discussed.

**Measured values:**
- Rotational speed and direction of yaw motor, for control (motor feedback and nacelle orientation)
- Temperature of yaw motor for overload yaw motor
- Current over the yaw motor for condition monitoring
- Vibrations of the yaw motor for condition monitoring
- Excessive cable twist for limit detection
Overview:
For a better understanding a schematic overview of the sensors on the yaw drive mechanism is presented in Figure 7.

![Diagram of Yaw Drive Mechanism](image)

<table>
<thead>
<tr>
<th>Sensor Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ω</td>
<td>Rotational and directional sensor (Absolute optical encoder) (3x)</td>
</tr>
<tr>
<td>T</td>
<td>Electronic Thermometer (thermistor) (3x)</td>
</tr>
<tr>
<td>I</td>
<td>Current (Ammeter) (3x)</td>
</tr>
<tr>
<td>Hz</td>
<td>Vibrations (differential capacitor) (3x)</td>
</tr>
<tr>
<td>S</td>
<td>Switch (electrical bridge) (1x) (for cable twist)</td>
</tr>
</tbody>
</table>

Figure 7 Typical arrangement of Yaw Bearing, Yaw Drive and Yaw Brake (Burton et al., 2002)

2.3.4 Sensors hydraulic group for mechanical and yaw brakes

Description:
The yaw brake and mechanical disk brake are actuated with a hydraulic system. The pressure switches should have the same position and are compared for this to find bad pressure switches (Rademakers et al., 1998).

The yaw brake consists of a large disk brake activated by 6 hydraulic brake callipers (Nordex, 1998). There are 2-3 pumps that set 10 litre oil at a pressure of 140-150 bar (Vestas, 2005).

A wind turbine mechanical disc brake typically consists of a steel brake disc acted on one or more brake callipers. The brake callipers are almost always arranged so that the brakes are spring applied and hydraulically retracted, i.e. fail-safe. (Burton et al., 2002)
Measured values:
- On/off switches on the brakes for feedback of the brakes.
- Pump frequency for: (Rademakers et al., 1998)
  1. Feedback of activation of the pump
  2. State of hydraulic system
- Pressure sensors before and after oil filter for: (Suzlon, 2005)
  1. State of oil filter
  2. State of hydraulic system
- Oil level in the fluid reservoir for detection of low oil level.

Overview:
For a better understanding a schematic overview of the sensors in / on the hydraulic group is presented in Figure 8.

Figure 8 Sensor overview of hydraulic group for mechanical and yaw brakes
2.3.5 Sensors fixed speed induction generator

Description:
The controller will constantly monitor the voltage and frequency of the alternating current of the grid to protect the drive train for voltage or frequency outside certain limits of the grid. The controller will disconnect the generator from the grid when there are problems with the grid.

Thyristors are installed for the start up of the generator. They connect and disconnect gradually to the grid in order to create a soft start. They are equipped with temperature sensors. (Rademakers et al., 1998)

As soon as the wind turbine is operating, the thyristors are omitted to prevent loss of energy. This is done with a bypass switch.

The generator is a fixed speed induction generator. The correct working of the generator is checked by its temperature at each winding, vibrations at the bearings and power output (phase, current and voltage).

The generator is liquid cooled. The cooling system consists of 2 pumps, heat exchanger, 2 fans, a liquid reservoir, a pressure relieve valve and several temperature sensors.

Measured values:
- Grid voltage for protection of the drive train for high/low voltage (Rademakers et al., 1998)
- Grid phase for protection of the drive train
- State of bypass switch of the thyristors for feedback
- Temperature thyristors for monitoring of thyristors
- Vibrations of the generator for monitoring of the generator including their bearings
- Phase / Current / Voltage of the generator output for statistics and monitoring the generator
- Temperature of the stator windings for monitoring the generator and controlling / monitoring the coolant system
- Pump frequency for
  1. Feedback of activation of the pump
  2. State of hydraulic system
- Liquid pressure after the pumps for monitoring the coolant pumps
- Temperature of coolant for monitoring the heat exchanger
- Temperature of the motor of the pump for condition monitoring
- Rotational speed of fans for feedback of activation of the fan
- Oil level for detection of low oil level

Overview:
For a better understanding a schematic overview of the sensors in / on / around the generator is presented in Figure 9. In Figure 10, a cross section of the generator is shown. In Figure 11 is shown how a generator looks like from the outside. Table 4 gives a list with sensors that can be found at the electrical equipment.
Figure 9 Sensor overview of the hydraulic system of the generator

T  Electronic (coolant) thermometer (thermistor) (13x)
ω  Rotational sensor (optical encoder) (4x)
Hz  Accelerometers (differential capacitor) (2x)
P  Pressure sensor (piezoresistive) (4x)
L  Oil level switch (electrical bridge) (1x)

Figure 10 Cross section of the generator

Figure 11 Induction generator (DWAI, 2003), green cross section is shown in Figure 9, red cross section is shown in Figure 10.
Table 4: List with electrical equipment and their sensors

<table>
<thead>
<tr>
<th>Grid</th>
<th>Thyristors (3x)</th>
<th>Generator</th>
<th>Coolant System</th>
<th>Generator Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>3x phase</td>
<td>3x bypass switch</td>
<td>6x temperature</td>
<td>2x pump frequency</td>
<td>3x phase</td>
</tr>
<tr>
<td>3x voltage</td>
<td>3x temperature</td>
<td>2x vibration</td>
<td>4x pressure</td>
<td>3x voltage</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4x temperature</td>
<td>3x current</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2x fan frequency</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1x coolant reservoir</td>
</tr>
</tbody>
</table>

Sensors that are not in the figures:
V Voltage (voltage trafos) (6x)
ϕ Phase (current trafos) (6x)
S Switch (electrical bridge) (3x)
I Current (ammeter) (3x)

2.3.6 Sensors control cabinet

Description:
Each control cabinet has a watchdog timer, fan, temperature sensors and a humidity sensor. The controller has a watchdog timer, which the controller resets every controller time step. If it is not reset within this time, this indicates that the controller is faulty and the safety system should shut down the turbine. (Burton et al., 2002)

Temperature and humidity sensor are placed to activate the fan. In total there are five control cabinets. There are three control cabinets located in the hub, one in the nacelle and one in the tower.

Measured values:
- Controller activity for safety
- Temperature sensor for control of the fan
- Humidity sensor for control of the fan

Sensors:
t Timer (5x)
T Electronic thermometer (thermistor) (5x)
H Humidity sensor (electrical resistance) (5x)

2.3.7 Sensors main bearing

Description:
The main bearing is located between the gearbox and the hub. This important bearing is monitored with vibration sensors and temperature sensors.
Measured values:
- Bearing cap temperature for condition monitoring
- Accelerometers at bearings for condition monitoring

Sensors:
Accelerometers (differential capacitor) (2x)
Electronic (Oil) Thermometer (thermistor) (2x)

2.3.8 Other sensors

On top of the nacelle are 2 anemometers and 2 wind vanes located (Suzlon, 2005; Harakosan, 2008; Nordex 1998). The wind vane is standing on the nacelle and therefore giving the (local) angle between the front of the nacelle and the wind direction. (This is not the real wind rose direction, this can be calculated by taking in account the rotations of the yaw mechanism. The real wind rose direction is not needed.) The anemometers measure the wind speed.

Temperature is measured in wind turbines to detect extreme conditions like too cold (<-25°C) or hot (>50°C) for safe operation (Rademakers et al., 1996).

The access hatch to the roof and the hub are equipped with sensors to check if they are closed. The rotor lock, which is used to lock the rotor during maintenance, is also equipped with such a sensor. (Rademakers et al., 1996)

Emergency switches are located in the nacelle and tower for safety. There is another limit switch that measures the accelerations at the top of the tower. When the tower starts shaking, this sensor will stop the wind turbine (trip).

Measured values:
- Angle between wind direction and nacelle for operation
- Wind speed for operation
- Temperature in the nacelle and hub for checking correct operation conditions
- Open / closed access hatch to roof for safety
- Open / closed access hatch to hub for safety
- On / Off rotor lock for safety
- Push on emergency switch for safety
- Accelerations and vibrations of the tower for safety

Overview:
For a better understanding a schematic overview of the other sensors is presented in Figure 12.

2.3.9 Sensor overview

In total there are 145 sensors found in previous sections. These sensors are drawn in Figure 13 to get an overview of where the sensors are placed.
θ Angular displacement (absolute optical encoder) (2x)
ω Rotational sensor (optical encoder) (2x)
T Electronic thermometer (thermistor) (1x)
S Switch (electrical bridge) (5x)
Hz Accelerometers (differential capacitor) (3x)

Figure 12 Sensor overview of the other sensors

Figure 13 Sensor overview of wind turbine
3 Assessing a sensor network

In the previous chapter, the sensors of the wind turbine were introduced. A high number of individual sensors were found and discussed. These sensors together with a reasoning system form the sensor network (Figure 14). In section 3.1 is explained what exactly is meant by a sensor network and what the redundancies of the sensor network are. Finally, a definition is presented of an optimal sensor network. In section 3.2 is explained what a qualitative diagnostic model (QDM) is, how it is build up, and how it can be used as a powerful tool to assess a sensor network.

3.1 Sensor network

The sensor network creates a representation of the physical world (see Figure 14). This representation of the physical world supplies the controllers, the protection system, and the condition monitoring system with the values they need.

The sensor network in Figure 14 for example, has six physical world values as input and four values as output. This surplus of input values can lead to redundancies in the sensor network. The reasoning system determines what the representation of the physical world values are by making use of these redundancies. The reasoning system is aware of the fact that a sensor can be broken. In Figure 14, all four values that represent the physical world are used by the controllers, and only two are used by the protection system and the condition monitoring system.

The different types of redundancies in the sensor network are described in section 3.1.1. In order to assess a sensor network, criteria must be set on which the sensor network can be tested. These criteria are defined in section 3.1.2.
3.1.1 Sensor network redundancies

For operation of a wind turbine, the controller, protection system, and condition monitoring system need to know the physical world values. A sensor network is used to get a representation of these physical values. As seen in Figure 14, a sensor network consists of sensors and a reasoning system.

Ideally, the representation of the physical world values are the same as the physical world values. However, there is always a chance that a sensor is broken and is giving a wrong output value. By including redundancies in the sensor network, the representation of the physical world can still be correct, even when there is a sensor faulty.

Using more than one sensor to measure the same physical variable is an example of creating sensor network redundancies. These sensor network redundancies reduce the chance of ending up with an incorrect representation of the physical world.

Sensor network redundancies can be achieved in multiple ways. Placing the same sensor more than once is an easy to understand redundancy of sensor information. In this section, five different sources of sensor network redundancies are discussed: sensors and physical laws, sensors and operation states, sensors in parallel, sensors and failure modes and sensors and control.

1. Sensors and physical laws

There may be interactions among the physical variables (redundancies) of the individual sensors. Often these interactions between the different physical variables create redundant sensor information (Iyengar et al., 1995). For instance, a wind vane on a wind turbine measures the angle of the misalignment between the wind direction and the rotor blades. This angle changes when the yaw mechanism starts rotating the nacelle. We can say that the wind vane directly measures misalignment with the wind direction and indirectly measures the rotational speed of the wind turbine.

All physical interactions follow the rules of physics. The correctness of measured values can be tested with these physical rules.

2. Sensors and operation states

The behaviour of components can be split up in operation states. In an operation state certain output is expected. When a measured value of a sensor does not fit in the operation state, there must be some faulty component.

As an example of operation state sensor network redundancy the electrical motor will be given. The operation of an electrical motor can be divided in 5 healthy operation states:

1. Cooled down and turned off
2. Starting up
3. Up and running
4. Shutting down
5. Just stopped and still cooling down

When the electrical motor is in operation state up and running, it is not likely that the measured physical variable ‘rotational speed’ suddenly drops to zero. When this happens, there must be something wrong. The motor could be broken and got stuck all of a sudden (e.g. by an external object). However, there are more physical variables measured at the sensor network of the electrical motor (like vibration, temperature and current). When these
measured variables all indicate that the electrical motor is in operation state *up and running*, the chance increases that the measured value of the rotation speed is incorrect.

3. **Sensors in parallel**

When there are more than one sensor placed in a sensor network to measure one physical variable, than it is expected that their measured variables are the same. When they give contradictory output, one or more of them must be incorrect.

As an example of *sensors in parallel sensor network redundancy* the wind vane will be discussed. When there are two wind vanes placed on a wind turbine and their output is the same, there must be at least one sensor that is incorrect. The cause can be for example an incorrect wiring by manufacturing or ice that could prevent the movement of a wind vane. Because the sensors are working on the same principle, manufactured by the same manufacturer and working at the same environment, both can stop working properly because of the same cause. However, stop working properly at the same moment in time is not very likely.

4. **Sensors and failure modes**

Failure modes are the different known failures of a component with their failure rate of appearing. These failure modes tell how likely it is that a certain faults appears and which measured values (sensor output) are expected when this fault appears.

These failure modes can be used in two ways to increase sensor network redundancies.

Failure modes can increase sensor network redundancies when sensors are placed to monitor a component. When the failure modes of the component are known, then on forehand it is known what the measured values are when a fault appears. The chances of having this failure mode are often known, this helps to know how likely it is that the failure takes place.

In Table 5 an example of the failure modes of an electrical motor are given, obtained from Rao (1996). When the measured sensor values do not fit in any of the failure modes, the chance increases that one of the sensors is broken instead of the electrical motor.

<table>
<thead>
<tr>
<th>Table 5: Failure modes of an electrical motor</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Fault type</th>
<th>Cause</th>
<th>Chance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bearings</td>
<td>Lack of lubrication or excessive loads</td>
<td>30%</td>
</tr>
<tr>
<td>Burn-out</td>
<td>Heat generation has exceeded heat loss</td>
<td>30%</td>
</tr>
<tr>
<td></td>
<td>(high loads, high resistance, high ambient</td>
<td></td>
</tr>
<tr>
<td></td>
<td>temperature)</td>
<td></td>
</tr>
<tr>
<td>Misalignment &amp; unbalance</td>
<td>Installation or manufacturing problems</td>
<td>30%</td>
</tr>
<tr>
<td>Electrical connections</td>
<td>Lose, broken or other problems with electrical connections</td>
<td>5%</td>
</tr>
<tr>
<td>Windings and castings</td>
<td>Winding or castings</td>
<td>5%</td>
</tr>
<tr>
<td>Other</td>
<td></td>
<td>&lt;1%</td>
</tr>
</tbody>
</table>

The second way that failure modes can help, are the failure modes of sensors themselves. These help to predict when a sensor is giving incorrect output. The chance that a
sensor gets broken without any information is more difficult to predict than when this information is known.

As an example, anemometers can get stuck by icing. However, when it is sunny and warm it is not possible to have a stuck anemometer by icing.

5. Sensors and control
In a wind turbine there are controllers that actuate the different components of the wind turbine. With these actions some reactions are expected. These reactions can be used to check if sensors and components are still in good condition.
Two examples: When the wind vane is iced, it will look like that the wind direction is changing with the same rotational speed as the nacelle is turning around (by the yaw mechanism). A control valve that is stuck will not react to its control request (e.g. the pressure should rise but remains constant) (Bos, 2005).

3.1.2 Definition of optimal sensor network
An optimal sensor network for wind turbines will satisfy the following four criteria:
- It provides physical variables for control, protection system and condition monitoring system.
- It has redundant information for fault diagnosis.
- It has redundant information to have a sufficient high chance of detecting limit values.
- It has as few sensors as possible.

These four criteria for an optimal sensor network are discussed below.

1. Physical variables for control, protection system and condition monitoring system
The sensor network should supply the variables that are needed for control, limit detection and condition monitoring. (See section 2.2.) Below these three practices are explained shortly.

Without some physical variables for control, the wind turbine will be less effective. Therefore a good sensor network should provide these variables.

Sensors for limit detection are very important. When the limits of operation are exceeded and not detected, catastrophic failures can occur. These physical variables must be supplied for safe wind turbine operation.

Sensors for condition monitoring are not directly needed for healthy operation of the wind turbine. However, condition monitoring can prevent failure by giving an early warning for component failure. Placing sensors for condition monitoring is therefore a choice of the wind turbine designers.

The literature study in chapter 2, especially section 2.2, comes up with the sensors that are needed for the wind turbine control, protection system, and condition monitoring system. By comparing the variables that are supplied by the sensor network with the variables that are needed this criterion can be assessed. For now, there is no tool needed to assess this criterion.

2. Redundant information for fault diagnosis
Fault diagnosis of sensors and other wind turbine components decreases down time of the wind turbine for two reasons.

Firstly, knowing the failure means that is known what has to be replaced / repaired. There is no need to send first a mechanic to find the failure. Directly the right tools and components can be brought and directly the right mechanic (skilled for the job) can be send. Therefore, fault diagnosis will decrease the down time of the wind turbines, especially for
remote located wind turbines (like offshore). These wind turbines are less easy to visit for inspection.

Secondly, if it is crystal clear what the source of the problem is, there can be searched for new ways to continue operation without human intervention. This will decrease the down time even more. Possibilities to continue operation by responding to faults by reconfiguring the system or by continuing operation at reduced performance are under research (Echavarria et al., 2007).

To make the source of the problem crystal clear good arguments, that show that a component is faulty, are needed. For example, when a sensor is measuring a limit value and there is thought that this sensor is faulty, a very good proof is needed to allow continuing operation. Sensor network redundancies can give this proof.

In previous research (Huberts, 2008), a QDM is build for fault detection. A simple QDM with some basic sensors and sensor network redundancies was used with a wind turbine simulation program. The first results of this simple qualitative model were promising for failure detection. QDM’s could be used to test if the redundancies in a sensor network are capable of fault detection.

3. Redundant information to have a sufficient high chance of detecting limit values
When a faulty sensor is detected remotely, because there is a good diagnostic system, there is no technical reason to stop producing electricity. The wind turbine is still able to generate electricity. ("Only a sensor is broken.") However, the sensor network has to provide its representation of the physical world without one of its sensors. One important question before continuing operation is: "Is it still safe to continue operation?"

In general, safe operation is guaranteed by having a sufficient high change of detecting limit values (i.e. values where continuing operation could lead to destruction of (a part of) the wind turbine). As long as there are no limit values it is safe to operate. A good sensor network has enough redundancies to have a high chance to detect limit values. When the chance of detecting limit values is still high when one sensor is ignored (because it is faulty), there is an opportunity to continue operation safely.

When there is a variable at its limit value, the chance that this is actually measured as a limit value depends on the robustness of the sensors and on the redundancies in the sensor network. When a sensor is measuring a normal temperature, the physical world temperature can still be at a limit value. This is true because, there is a chance that the temperature sensor is broken and gives the wrong temperature. When there are redundancies in the sensor network that can verify that a measured value is not at a limit value, the sensor network is more reliable.

How large the chance must be to detecting a limit value is a task of risk analysers and insurance companies. It depends on the chance that a sensor gets faulty, the chance of having limit values, the chance that more than one sensor will get faulty at the same time, and the consequences when the limit value would not be detected.

It is not easy to see directly what the chance is to detect a limit value because it is often unclear how the network redundancies are related to each other. To determine remotely if it is safe to continue operation, a tool is needed that can give directly insight in the chance of detecting limit values. The opportunity to do this with a QDM will be discussed in section 3.2.

4. As few sensors as possible
The more sensors there are in the sensor network, the more complex the network becomes. High complexity of the sensor network increases costs for designing, building and
maintaining the sensor network. Therefore, an optimal sensor network consists only of sensors that are really needed.

### 3.2 Qualitative diagnostic model (QDM) as a tool to assess a sensor setting

Four criteria for an optimal sensor setting were described in section 3.1.2. To assess a sensor network, especially a tool is needed to test if the sensor network has a sufficient high chance of detecting limit values (see section 3.1.2).

The QDM is a good candidate to be this tool. It can be build up easily with redundancies because it is qualitative. In more detail the QDM is explained in section 3.2.1. In section 3.2.2 is explained how a QDM can give the chance that limit values would not be detected.

#### 3.2.1 Explanation of QDM

The QDM uses a model of the system (e.g. wind turbine) to reason about its behaviour. It simulates what the system is supposed to do based on this model, and compares it to what it does in reality with the sensor outputs and controller signals. In this way, any not expected behaviour is detected (Darwiche, 2000).

The system suggests candidates as faulty components based on this comparison process. First, it compares the input values with its model and tries to match these values with healthy behaviour according to this model. Thereafter, it searches for combinations with one faulty component that can match the input variables, followed by two and more faulty components. For example, if there is no match with healthy behaviour and there is only one match with one faulty component, the diagnostic model concludes that this component is faulty and there is no indication that other components are faulty.

The diagnostic model cannot verify if there are any healthy components in the wind turbine. When the healthy behaviour matches the input variables, the conclusion is that there is no indication for any failure. This is because it is still possible that a component is broken but does not show faulty behaviour in the current state of the system. For example, when a electro motor is stuck, it will show healthy behaviour when it is not in use. However, as soon as the motor is actuated, it will appear to be broken.

The diagnostic model is qualitative, qualitative reasoning works with a finite number of variables (usually three or four) instead of numbers (quantitative reasoning). Variables are expressed by domains like signs \{negative, zero, positive\} or magnitudes \{low, moderate, red\}.

Qualitative reasoning allows researchers to model physical systems using incomplete information. It is difficult to know precisely the changes within a system. However, the basic relations between the variables in the system are often well known. In this situation, qualitative models can be used to capture incomplete knowledge in a model, which can be simulated to obtain a rough outline of the system behavior. (Houten et al., 1998; Sokolsky et al.)

A qualitative model still can use mathematical and differential equations. These equations provide information on how the system behaves. By modifying one side of an equation, conclusions can be made on what may happen to the variables on the other side of
the equation. For instance, having $a + b = c$, if $a$ increases and $b$ remains constant, it can be said that $c$ will increase. (Echavarria et al., 2008)

In previous research, a QDM is build for simple fault detection in wind turbines (Echavarria et al., 2008). This QDM was tested with a wind turbine simulation program (Huberts, 2008), these first results were promising and consistent enough to continue research. How the model is build up will be shown in chapter 4 by working out an example of the QDM.

### 3.2.2 QDM and the chance of detecting limit values

In this section will be described how the QDM can be taken to a higher level. The strategy is to let the QDM search for limit values while they are not measured (because of sensor failure). The QDM will give the chance that it is possible to have an undetected limit value. This chance can be used to assess the sensor network.

An additional health variable is added that keeps track of any possible limit values. This health variable is not used to tell if a component is healthy, it is a trick to check for any limit values. The new health variable is healthy when there are no limit values and is unhealthy when there are one or more limit values. With the new health variable the QDM will search in its comparison process also for limit values and report if they were found.

The later the first limit values are found in the comparison process, the better the sensor network is. For example, if the first limit value is found when two components are broken, the sensor network is more robust than when the first limit value is found already by one broken component.

The QDM will also indicate how the sensor network can be improved. This will be explained with an example. The post processed output (see section 4.2) of the QDM for this example is printed in Table 6. First the important features of this output are explained, thereafter the tips for improvement will be discussed.

In the first level of the comparison process (indicated by 01) there is printed: "-no faulty components detected-". This means there is no indication, based on the input values, that there is any faulty component. In the second level of the comparison process (indicated by 02) there are the possibilities of one broken component printed.

When "-might be a limit value-" is printed behind the component, it means that when this sensor is broken, there is not enough information to check if one or more variables are at their limit value. In other words, by ignoring this sensor, there might be a limit value, which will not be measured.

In the example "-might be a limit value-" is printed behind the temperature sensor and not behind the other three components. This means that the temperature sensor is the weak link in the sensor network. A physical variable (probably the temperature) can get red without notice of the sensor network when the temperature sensor is faulty.

In the next chapter a full example is worked out of a QDM as a tool to assess a sensor network on the chance of detecting limit values.
Table 6: Post processed output example of the QDM

01 -no faulty components detected-
02 Temperature sensor -might be a limit value-
02 Current sensor
02 Electrical_motor component
02 Vibration sensor
4. Example: assessing sensor network of the pitch mechanism

In the previous chapter, a strategy was described to assess a sensor network by finding the chance to detect limit values. In this chapter a single pitch system, which is responsible for the pitch angle of one blade (see Figure 15), will be put in a qualitative model to assess its sensor network.

In section 4.1 the construction of the QDM is described step by step. A post processor is used to make the output of the QDM understandable without tedious searching (4.2). In section 4.3 is looked at the output of the model and discussed how the sensor network of this example could be improved according to this assessment.

Figure 15 Single pitch system

4.1 Construction of QDM

The QDM for sensor network assessment is build up in four steps:

1. Variable declaration to define the qualitative variables with their quantitative meaning. Also is described which components will get a health variable. A health variable means that the QDM will check the model to determine if the component could be healthy (section 4.1.1).

2. Healthy behaviour description, to describe which behaviour is expected and will lead to a healthy component (section 4.1.2).

3. Redundancies of the sensor network are included to know the relations between the different physical values (section 4.1.3).

4. Declaration of an additional health variable that is needed to assess the sensor network by finding the chance to detect limit values (section 4.1.4).

The whole code of the QDM model can be found in Appendix 1.
In this chapter are symbols used in the code and tables. In Table 7 the symbols are defined.

Table 7: Used symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Not</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>And</td>
</tr>
<tr>
<td>==</td>
<td>Is equal to</td>
</tr>
<tr>
<td>=&gt;</td>
<td>Implies ((A =&gt; B) means if A is true than B is true)</td>
</tr>
</tbody>
</table>

4.1.1 Variable declaration

Constructing a QDM of the pitch system starts with the declaration of the variables and the definition of the qualitative variables.

Sensors:
Sensors have a health variable in the QDM. The output of the model will show if the sensors are healthy or faulty. The following sensors are included in the model of the pitch system and have a health variable:
- Thermometer that measures the air temperature in the hub
- Voltmeter that measures the electrical potential of the battery
- Switch that tells what the electricity source is
- Thermometer that measures the temperature of the motor
- Accelerometer that measures the vibrations of the motor
- Ammeter that measures the current that is running through the motor
- Incremental meter that measures the blade angle and the rotational speed of the blade
- Switch that tells if the internal brake is applied

Other components:
Besides sensors, there are a battery, an electrical motor, a gearbox, a pitch controller, switches, and electricity connections in the pitch system. When the model is used for diagnosis, all these components should have their own health variable. Now we are doing a sensor assessment. Therefore, these components will have one single health variable:
- Pitch mechanism

Input variables:
Output values of some sensors and controllers in the wind turbine, which are not located in this single pitch system, are useful for the sensor network redundancies of the single pitch system. To be able to use these variables, they are included and called input variables. These sensors and controllers itself are not assessed and the output of the sensors is assumed to be the same as the physical world values (see Figure 14). When needed, these sensors could be assessed later on in another model. The input variables of the following components are used, they do not have a health variable:
- Timer of the central controller, which counts the seconds that have past since the last operation state change of the pitch system
- The operation state of the wind turbine
- Condition of the grid
- Vibrations of the nacelle bedplate
Domain of the qualitative variables:
All the variables have to be transformed into qualitative variables. In alphabetic order, the thresholds for the qualitative variables are defined. For some variables an axis is drawn, above the line the qualitative variables are shown, under the line their quantitative meaning is explained.

- Battery voltage:
  True: the voltage is as expected
  False: something is wrong with the voltage

- Rate of change of the pitch angle (direction of the rotational speed of the motor) (θ' motor)
  neg  zero  pos
  -------------------------------
  ----------------------------------
  0

- Rate of change of the rotational speed (acceleration of the motor) (ω' motor)
  neg  zero  pos
  -------------------------------
  ----------------------------------
  0

- Rate of change of the temperature of the motor (T' motor):
  neg  zero  pos
  -------------------------------
  ----------------------------------
  0

- Current motor (I motor):
  zero low norm red
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>-----------------------------</td>
</tr>
</tbody>
</table>
  zero min. I max. I
  during during
  operation operation

- Grid status:
  True: the voltage, phase, and frequency are as expected
  False: something is wrong with the voltage, phase, or frequency

- Pitch angle (θ motor):
  red neg brake zero low optimal stall red pos
  -------------------------------|-------------------------------|
  red neg: strange (negative) angle of blade
  brake: aerodynamic brake
  zero: 0, angle when the blades are pitched out of the wind
  low: for transitions between 0 and optimal angle
optimal: optimal pitch angle
stall: above rated wind speeds of the wind turbine
redpos: strange (positive) angle of blade

• Rotational speed ($\omega$ motor)
  
  zero  low  norm  red
  -----------------      -------
  -----------------      -------
  0  max  speed

• Temperature of the hub (T hub):
  
  red  norm  red
  --------      ------
  --------      ------
  too cold  too hot

• Switch for electricity source
  True: Grid supply
  False: Battery supply

• Switch for internal brake
  True: Internal brake is in use
  False: Internal brake is not in use

• Temperature of the motor (T motor):
  
  zero  low  norm  red
  ---------      -------
  ---------      -------
  normal  operation
  ambient  temperature

• Vibrations in the nacelle bedplate (Hz nacelle):
  The sensor interprets the vibration spectrum.
  Low:  low background vibrations
  Norm: during normal operation of the wind turbine
  Red: Vibrations that interfere with other vibration sensors. These vibrations are not
  produced by any healthy component.

• Vibrations motor (Hz motor):
  The sensor interprets the vibration spectrum.
  Low: background vibrations, motor not in use
  Norm: motor in use
  Red: vibrations that are related to failure of the motor or external vibrations

• Watch dog timer
  The timer clocks the time that has expired since the last operation state change of the central
  controller. This input value is used to know if the pitch mechanism is still in the transition
  state or in the steady state (see Figure 18).
4.1.2 Healthy behaviour

The QDM needs to know when a component is healthy. Healthy behaviour means that a component is showing expected behaviour.

Healthy behaviour of a sensor is that its output has the same value as the physical world value (see Figure 14). The model does not know the physical world variable. The model builds its own representation of the physical value by making use of the redundancies in the network (see section 3.1.1). The QDM determines if a sensor is healthy by comparing its sensor output with its representation of the physical world.

See figure 2 for an example of the code for the temperature sensor on the electrical motor. The statement in this code is: If the temperature sensor is healthy, then the measured temperature is equal to the representation of the temperature and the sensed first derivative (rate of change) of the temperature is equal to the representation of the first derivative of the temperature.

```plaintext
(S_T_motor) =>
    ( (sv_T0_motor == rv_T0_motor)
        &&
        (sv_T1_motor == rv_T1_motor)
    );
```

Figure 16 The healthy behaviour of a sensor (thermometer on the motor)

The healthy behaviour of the whole pitch mechanism is more complex. Its healthy behaviour is described in the redundancies of the sensor network, especially in the operation states description (see the next section).

4.1.3 Sensor network redundancies

The sensor network redundancies as described in section 3.1.1 will be implemented in the qualitative model. All five are described below. The nomenclature of the symbols can be found in Table 7.

1. Physical laws

There are many physical laws implemented in the model. They are described one by one below.

```plaintext
one       two       three       four
 Artifact |----------------|----------------|----------------|
          |----------------|----------------|----------------|
zero      transition somewhere steady state
state     in state three, the steady state will be reached
```
Physics for electricity, acceleration and rotational speed

When the pitch system is healthy, there is a relation between use of electricity (current) by the motor, rotational acceleration of the motor shaft, and rotational speed of the motor shaft. (The rotational speed of the motor is directly related to the rate of change of the pitch angle.)

The pitch system must be healthy for this physical law for two reasons. First, when there is something wrong with the pitch system (e.g. internal brake), the motor is not the only one who can change the angle of the blades. Forces by the wind or gravity can change the angle of the blades when the pitch system is not healthy. Second, when the pitch mechanism is not healthy, it is possible that the motor does not transfer electricity into motion.

In Table 8, the different relations are written down. For acceleration and continuous rotational speed, current is needed. When there is deceleration, current is not necessary. Still we can say something when there is no current. Table 9 shows these relations. Assuming there is some rotational speed and no current, there must be deceleration.

| Acceleration | Positive | Positive | & & True | => | ! Zero |
| Acceleration | Negative | Negative | & & True | => | ! Zero |
| No acc. | Positive | Negative | & & True | => | ! Zero |
| No acc. | Negative | Zero | & & True | => | ! Zero |

Deceleration

| Deceleration | Positive | Negative | & & True | => | ?? |
| Deceleration | Negative | Positive | => | ?? |
| Stand still | Zero | Zero | => | Zero |

| Deceleration | Positive | Zero | & & True | => | Negative |
| Deceleration | Negative | Zero | & & True | => | Positive |

| Deceleration | Positive | Zero | & & True | => | ! Positive |
| Deceleration | Negative | Zero | & & True | => | ! Negative |

Motion physics

The accelerometer is installed for condition monitoring of the electrical motor, it measures the vibrations of the motor. External vibrations can interfere with the vibrations of the electrical motor. External vibrations will be measured by the accelerometer in the nacelle as well. The definition of the qualitative values is as follows:
• When the vibrations in the nacelle are called red, these vibrations will interfere with the vibration measurement at the motor. The vibrations at the motor will be red as well.
• When the vibrations of the motor are called red, these vibrations do not necessarily interfere with the vibration measurement at the nacelle.

When the pitch system is healthy and there are no external vibrations, then the vibrations of the motor reflect the rotational speed (see Table 10). The vibrations created by a healthy motor cannot be generated by any external device or failure. Therefore a normal vibration always means a normal rotational speed (whatever the health of the pitch system is) (see Table 11).

<table>
<thead>
<tr>
<th>Table 10: Rotational speed, vibrations in the nacelle and pitch health determine the vibrations in the motor</th>
</tr>
</thead>
<tbody>
<tr>
<td>ω  (rotational speed)</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>Zero</td>
</tr>
<tr>
<td>Low</td>
</tr>
<tr>
<td>Norm</td>
</tr>
<tr>
<td>Red</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 11: Vibrations in the motor determine the rotational speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hz (vibrations motor)</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td>Low</td>
</tr>
<tr>
<td>Norm</td>
</tr>
<tr>
<td>Red</td>
</tr>
<tr>
<td>Red</td>
</tr>
</tbody>
</table>

An additional rule can be added for the motion physics. When there is no rotational speed and no current used, the vibrations variable cannot be equal to normal:

\[(rv\_Io\_motor == sens.zero) \&\& (rv\_w0\_motor == sens.zero)) => (rv\_Hz0\_motor != sens.normal);\]

• **Electricity physics**
  The electricity switch is responsible for selecting the electricity source. When there are problems with the grid, the switch will choose the battery as power source (healthy behaviour of the pitch system controller).

  Heat flows from cold to hot. In the electromotor current that is going through the coils generates heat. This heat is air cooled by a fan attached on the shaft. In order to cool, the ambient air may not be too high.

  Heat conduction is the spontaneous transfer of thermal energy through matter, from a region of higher temperature to a region of lower temperature, and acts to equalize temperature differences. It is also described as heat energy transferred from one material to another by direct contact.
When the electrical motor is stopped, it will cool down as long as:
- The ambient temperature is lower than the motor temperature
- There is no current running through the motor
- The rotational speed of the motor is zero

In Table 12 the rate of change of the motor temperature ($T'$) is calculated. The table can be read as follows. The possible qualitative temperature variables of the motor are in the first row, the possible qualitative temperature variables of the hub are in the first column. For each combination of these two qualitative values is the sign for the rate of change of the temperature of the motor given. For example, when the motor temperature is low, and the hub temperature is normal, the temperature change will be negative. When they are respectively low and red, the temperature change will be positive, zero or negative. (More information about calculation tables can be found in Echavarria et al., 2008)

<table>
<thead>
<tr>
<th>T hub \ T motor</th>
<th>Zero</th>
<th>Low</th>
<th>Norm</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Norm</td>
<td>+, 0, -</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Red</td>
<td>+, 0, -</td>
<td>+, 0, -</td>
<td>+, 0, -</td>
<td>+, 0, -</td>
</tr>
</tbody>
</table>

When the electrical motor is in use, it will heat up by the current that is going through the motor and is cooled down by the fan that is attached to the motor shaft. The amount of cooling depends on the rotational speed. In the following tables (Table 13, Table 14, Table 15 and Table 16), all the different options are filled in to calculate the temperature rate of change of the motor. Conditions for these tables are normal hub temperature and a healthy pitch mechanism. All the possible combinations of the following variables are put in the tables:
- Current running through the motor ($I_{motor}$)
- Rotational speed of the motor ($w_{motor}$)
- Temperature of the motor ($T_{motor}$)

Table 13: Calculation table to calculate $T'_{motor}$ in operation with $T_{motor} = \text{zero}$

<table>
<thead>
<tr>
<th>$I \backslash w_{motor}$</th>
<th>Zero</th>
<th>Low</th>
<th>Norm</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Low</td>
<td>0, +</td>
<td>0, +</td>
<td>0, +</td>
<td>0</td>
</tr>
<tr>
<td>Norm</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0, +</td>
</tr>
<tr>
<td>Red</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0, +</td>
</tr>
</tbody>
</table>

Table 14: Calculation table to calculate $T'_{motor}$ in operation with $T_{motor} = \text{low}$

<table>
<thead>
<tr>
<th>$I \backslash w_{motor}$</th>
<th>Zero</th>
<th>Low</th>
<th>Norm</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Low</td>
<td>-, 0, +</td>
<td>-, 0, +</td>
<td>-, 0, +</td>
<td>-</td>
</tr>
<tr>
<td>Norm</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-, 0, +</td>
</tr>
<tr>
<td>Red</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-, 0, +</td>
</tr>
</tbody>
</table>
Table 15: Calculation table to calculate $T^\prime_{motor}$ in operation with $T_{motor} = \text{norm}$

<table>
<thead>
<tr>
<th>$w_{motor}$</th>
<th>Zero</th>
<th>Low</th>
<th>Norm</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Low</td>
<td>–, 0, +</td>
<td>–, 0, +</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Norm</td>
<td>+</td>
<td>+</td>
<td>0</td>
<td>–, 0</td>
</tr>
<tr>
<td>Red</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>–, 0, +</td>
</tr>
</tbody>
</table>

Table 16: Calculation table to calculate $T^\prime_{motor}$ in operation with $T_{motor} = \text{red}$

<table>
<thead>
<tr>
<th>$w_{motor}$</th>
<th>Zero</th>
<th>Low</th>
<th>Norm</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Low</td>
<td>–, 0, +</td>
<td>–, 0, +</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Norm</td>
<td>0, +</td>
<td>–, 0, +</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Red</td>
<td>0, +</td>
<td>–, 0, +</td>
<td>–, 0, +</td>
<td>–, 0, +</td>
</tr>
</tbody>
</table>

We can also say something about the temperature when we do not assume that the pitch mechanism is healthy. When there is no rotation, we do not need the assumption of a working fan (a healthy pitch mechanism), the only assumption is a normal hub temperature (see Table 17).

Table 17: Calculation table to calculate $T^\prime_{motor}$ when $w_{motor} = \text{zero}$

<table>
<thead>
<tr>
<th>$T_{motor}$</th>
<th>Zero</th>
<th>Low</th>
<th>Norm</th>
<th>Red</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero</td>
<td>0</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Low</td>
<td>0, +</td>
<td>–, 0, +</td>
<td>–, 0, +</td>
<td>–, 0, +</td>
</tr>
<tr>
<td>Norm</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0, +</td>
</tr>
<tr>
<td>Red</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>0, +</td>
</tr>
</tbody>
</table>

Motion is generated by the electrical motor. The motor is equipped with an internal brake, which holds the blades in safe position between the pitch processes (Suzlon, 2005). When the brake is on, it will decelerate the rotational speed or the rotational speed will stay at zero, or there is something wrong with the pitch mechanism. The code is in Figure 17.

```c
((rv_s0_brake == true) (rv_theta1_motor == sign.pos))
  => ((rv_w1_motor == sign.neg)||(C_pitch == false));
((rv_s0_brake == true) (rv_theta1_motor == sign.neg))
  => ((rv_w1_motor == sign.pos)||(C_pitch == false));
((rv_s0_brake == true) (rv_theta1_motor == sign.zero))
  => ((rv_w1_motor == sign.zero)||(C_pitch == false));
```

Figure 17 When the brake is on, the rotational speed of the motor will decelerate or stay at zero.
2. Operation states
There can be distinguished 11 operation states that describe the pitch mechanism: four steady states and seven transition states (see Figure 18). Eleven states are too much for the qualitative diagnostic system (large increase of calculation time). Therefore, the transition states are split up in positive pitch angle change and negative pitch angle change. The rate of change of the pitch angle is an important variable, because it is indicating which direction the pitch angle is heading. We end up with six operation states. The six operation states are described below.

![Figure 18 Four steady states and seven transition states for the pitch mechanism](image)

1. Aerodynamic brake:
The pitch angle is pitched in order to slow down the rotational speed. It is assumed that the pitch angle does not change in this state. When the pitch mechanism is healthy, the representation of the physical variables should be:
- Vibrations low
- Current zero
- Blade angle brake
- Rate of change of the blade angle zero
- Internal brake applied true

2. Out wind:
The blades are pitched out of the wind because of too low or too high wind speeds. When the pitch mechanism is healthy, the representation of the physical variables should be:
- Vibrations low
- Current zero
- Blade angle zero
- Rate of change of the blade angle zero
- Internal brake applied true

3. Optimal pitch angle:
Below rated wind speeds, the blades catch as much wind as possible. When the pitch mechanism is healthy, the representation of the physical variables should be:
- Vibrations low
- Current zero
- Blade angle optimal
- Rate of change of the blade angle zero
- Internal brake applied true

4. Pitch to stall:
Above rated wind speeds, the blades are pitched to stall in order to miss the surplus of wind power. When the pitch mechanism is healthy, the representation of the physical variables should be:
- Vibrations low or normal
- Current not red
- Blade angle stall
- Rate of change of the blade angle -can be any value-
- Internal brake applied -can be any value-

5. Transition state neg
This transition state is used for the transition of the following states:
Pitch to stall -> Optimal
Optimal -> Aerodynamic brake
Pitch to stall -> Aerodynamic brake
When the pitch mechanism is healthy, the representation of the physical variables should be:
- Vibrations norm
- Current low or normal
- Blade angle -can be any value-
- Rate of change of the blade angle negative
- Internal brake applied false

6. Transition state pos
This transition state is used for the transition of the following states:
Aerodynamic brake -> out wind
Out wind -> Optimal
Out wind -> Pitch to stall
Optimal -> Pitch to stall
When the pitch mechanism is healthy, the representation of the physical variables should be:
- Vibrations norm
- Current low or normal
- Blade angle -can be any value-
- Rate of change of the blade angle positive
- Internal brake applied false

3. Parallel sensors
There are no parallel sensors included in the model, every sensor is placed only once.

4. Failure modes
There is not enough information available to include the failure modes.
5. Control
There is a controller for each pitch motor. These three controllers get their tasks from the central controller in the nacelle and from the safety system. The nacelle and safety system are not included in the model. Only their signals are important to get more insight in the optimal sensor setting for the pitch mechanisms.

Signals from the central controller are:
1. Aerodynamic brake
2. Out of the wind
3. Optimal pitch angle
4. Pitch to stall

When the signal from the central controller changes, the controller of each electrical motor takes care of going to the desired state. To get to the desired state, the pitch mechanism will go into a transition state (see Figure 18). These states and state changes are included in the model.

The angles are fixed in the first three states. In the fourth state, the rotational speed of the small shaft depends directly on the pitch angle. The pitch system is responsible in this state to keep the small shaft speed between limits.

The nacelle controller is not simulated. The output of this central controller is treated as physical input variables (see section 4.1.1).

A timer is used to check if the system is still healthy. If the maximum time is reached and the pitch system did not reach its pitch angle, there is something wrong. In Table 18 all the possible value combinations are given.

<table>
<thead>
<tr>
<th>Signal central controller</th>
<th>Timer</th>
<th>Current operation state</th>
<th>Healthy pitch mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Aerodynamic brake</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>Impossible^1</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>2. Out wind</td>
<td>False^2</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>3. Optimal</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>4. Stall</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>6. Transition (θ = pos)</td>
<td>False^3</td>
</tr>
<tr>
<td>Two</td>
<td>1. Aerodynamic brake</td>
<td>Impossible^4</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>2. Out wind</td>
<td>False^5</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>3. Optimal</td>
<td>False^5</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>4. Stall</td>
<td>False^5</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>6. Transition (θ = pos)</td>
<td>False^3</td>
<td></td>
</tr>
<tr>
<td>Three</td>
<td>1. Aerodynamic brake</td>
<td>True</td>
<td></td>
</tr>
<tr>
<td>Three</td>
<td>2. Out wind</td>
<td>False^6</td>
<td></td>
</tr>
<tr>
<td>Three</td>
<td>3. Optimal</td>
<td>False^6</td>
<td></td>
</tr>
<tr>
<td>Three</td>
<td>4. Stall</td>
<td>False^6</td>
<td></td>
</tr>
<tr>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
<td></td>
</tr>
</tbody>
</table>

Table 18: All the variations of controller signal, timer and current operation state with their healthiness
<table>
<thead>
<tr>
<th>Signal central controller</th>
<th>Timer</th>
<th>Current operation state</th>
<th>Healthy pitch mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Aerodynamic brake</td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>False^3</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>1. Aerodynamic brake</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>2. Out wind</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>3. Optimal</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>4. Stall</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>5. Transition (θ = neg)</td>
<td>False^7</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>6. Transition (θ = pos)</td>
<td>False^3</td>
</tr>
<tr>
<td>2. Out wind</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>2. Out wind</td>
<td>Impossible^1</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>3. Optimal</td>
<td>False^2</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>4. Stall</td>
<td>False^2</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>1. Aerodynamic brake</td>
<td>False^5</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>2. Out wind</td>
<td>Impossible^4</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>3. Optimal</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>4. Stall</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>1. Aerodynamic brake</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>2. Out wind</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>3. Optimal</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>4. Stall</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>1. Aerodynamic brake</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>2. Out wind</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>3. Optimal</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>4. Stall</td>
<td>False^6</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>5. Transition (θ = neg)</td>
<td>False^7</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>6. Transition (θ = pos)</td>
<td>False^7</td>
</tr>
<tr>
<td>3. Optimal</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False^2</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>2. Out wind</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>3. Optimal</td>
<td>Impossible^1</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>4. Stall</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>One</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td>Signal central controller</td>
<td>Timer</td>
<td>Current operation state</td>
<td>Healthy pitch mechanism</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------</td>
<td>------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>3. Optimal</td>
<td>Two</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;5&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>2. Out wind</td>
<td>False&lt;sup&gt;5&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>3. Optimal</td>
<td>Impossible&lt;sup&gt;4&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>4. Stall</td>
<td>False&lt;sup&gt;5&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Two</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td>Three</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Two</td>
<td>2. Out wind</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>3. Optimal</td>
<td>True</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>4. Stall</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>True</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td>Four</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Four</td>
<td>Two</td>
<td>2. Out wind</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Four</td>
<td>Three</td>
<td>3. Optimal</td>
<td>True</td>
</tr>
<tr>
<td>Four</td>
<td>Three</td>
<td>4. Stall</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Four</td>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>False&lt;sup&gt;7&lt;/sup&gt;</td>
</tr>
<tr>
<td>Four</td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>False&lt;sup&gt;7&lt;/sup&gt;</td>
</tr>
<tr>
<td>4. Stall</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;2&lt;/sup&gt;</td>
</tr>
<tr>
<td>One</td>
<td>Two</td>
<td>2. Out wind</td>
<td>True</td>
</tr>
<tr>
<td>One</td>
<td>Three</td>
<td>3. Optimal</td>
<td>True</td>
</tr>
<tr>
<td>One</td>
<td>Three</td>
<td>4. Stall</td>
<td>Impossible&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>One</td>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>False&lt;sup&gt;3&lt;/sup&gt;</td>
</tr>
<tr>
<td>One</td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td>Two</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;5&lt;/sup&gt;</td>
</tr>
<tr>
<td>Two</td>
<td>Two</td>
<td>2. Out wind</td>
<td>False&lt;sup&gt;5&lt;/sup&gt;</td>
</tr>
<tr>
<td>Two</td>
<td>Three</td>
<td>3. Optimal</td>
<td>False&lt;sup&gt;5&lt;/sup&gt;</td>
</tr>
<tr>
<td>Two</td>
<td>Three</td>
<td>4. Stall</td>
<td>Impossible&lt;sup&gt;4&lt;/sup&gt;</td>
</tr>
<tr>
<td>Two</td>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>False&lt;sup&gt;3&lt;/sup&gt;</td>
</tr>
<tr>
<td>Two</td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td>Three</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Two</td>
<td>2. Out wind</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>3. Optimal</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>4. Stall</td>
<td>True</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>5. Transition (θ = neg)</td>
<td>False&lt;sup&gt;3&lt;/sup&gt;</td>
</tr>
<tr>
<td>Three</td>
<td>Three</td>
<td>6. Transition (θ = pos)</td>
<td>True</td>
</tr>
<tr>
<td>Four</td>
<td>One</td>
<td>1. Aerodynamic brake</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

44
<table>
<thead>
<tr>
<th>Signal central controller</th>
<th>Timer</th>
<th>Current operation state</th>
<th>Healthy pitch mechanism</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Stall</td>
<td>Four</td>
<td>2. Out wind</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>3. Optimal</td>
<td>False&lt;sup&gt;6&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>4. Stall</td>
<td>True</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>5. Transition (θ = neg)</td>
<td>False&lt;sup&gt;3&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>Four</td>
<td>6. Transition (θ = pos)</td>
<td>False&lt;sup&gt;7&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

Failures:
1) When the timer is zero, the signal is new and different. This is clearly not the case; not a possible situation.
2) A change of states that is not within the healthy behaviour. When the system is pitched out of the wind, it should not return into aerodynamic brake state (see Figure 18). Fault of central controller, because the central controller itself is not in the system, this change is marked as impossible in the model.
3) This transition state is in the wrong direction. There must be a fault in the electrical motor or pitch controller.
4) When the timer is low, there is not enough time past in order to get into the requested state. Impossible situation like 1).
5) The pitch system did not get into the transition state. The pitch system is broken.
6) The pitch system did not get into the transition state or ended up in the wrong steady state.
7) The system needs too much time to reach its new steady state.

4.1.4 Additional health variable

As described in section 3.2.3, an additional health variable is added to track if a possible red value would not be measured. In Figure 19 the syntax is shown of the additional health variable. In this section this syntax will be explained.

The additional health variable is called Alarm. It has a probability of being healthy of 50% and a probability of being unhealthy of 50%. By setting similar chances, the comparison process will check directly for every output line if the variable alarm is true, false or both. If all the physical variables can be set on other values than red, "Alarm = true" is a possible output. If one of the physical variables can be set on red, "Alarm = false" is a possible output. Both possibilities are checked and both are printed down when they are a possible combination.

We are interested in what the physical world variables are. (The sensor output will give an indication of what the physical world values are (see also 4.1.2).) Therefore, we want to detect if the representation of the physical world variable is red.

Possible red values:
For the pitch mechanism, the following physical values are used for the assessment:
- Hub temperature
- Voltage of battery
- Temperature motor
- Vibrations motor
- Current motor
- Rotational speed motor
These are all the variables that indicate that something is wrong and that operation of the motor should stop.
bool Alarm;
attribute health (Alarm) = true;
attribute probability (Alarm) = Alarm ? 0.5 : 0.5;

(rv_T0_hub == sens.red) => (Alarm == false);
(rv_V_battery == false) => (Alarm == false);
(rv_T0_motor == sens.red) => (Alarm == false);
(rv_Hz0_motor == sens.red) => (Alarm == false);
(rv_I0_motor == sens.red) => (Alarm == false);
(rv_w0_motor == sens.red) => (Alarm == false);

(Alarm == false) => (rv_T0_hub == sens.red) ||
(rv_V_battery == false) ||
(rv_T0_motor == sens.red) ||
(rv_Hz0_motor == sens.red) ||
(rv_I0_motor == sens.red) ||
(rv_w0_motor == sens.red)
);

Figure 19 The additional health variable in the syntax of the model

4.2 Construction of post processor

A post processor is needed to make the output understandable without tedious searching in the rough output to find the information that is needed. The rough output is a long list with suggested faulty and healthy components. Because the healthy components are printed too, a lot of searching is needed without a post processor. In Figure 18 an example of the rough output is shown.

The rough output is structured as follows. The output is ordered from zero faulty components in the first line to all components suggested as faulty in the last line. Impossible healthiness combinations, based on comparing the input variables with the model, are not printed. The rough output in this experiment is cut to 100 lines, which are more than enough for the sensor network assessment.
The post processor itself is a small computer program that presents the information of the rough output less lengthy. An example of the output from the post processor is shown in Figure 21. In three steps the post processor creates its output, these steps are explained below.

Printing the healthy components as well as the faulty components is needless, a health variable can only be healthy or unhealthy. Therefore, the post processor prints only the faulty components, which makes the output immediately less tedious to read. When there are no faulty components it will print: "-no failure-".

When the additional health variable (see section 3.2.2) detects a physical variable that can be red without detection of the sensor network, the post processor will indicate this by printing the words: "-might be a red value-". This way, it is directly clear which component has enough coverage and which components not.

Repetition of the output is removed. For example, when you have the following two lines: "Temperature hub -might be a red value-" and the line "Temperature hub, Temperature motor -might be a red value-", then the second line is a repetition of the first line and has no new information. We already know, when the temperature sensor in the hub is ignored, that there is not enough coverage in the sensor network. Ignoring another sensor will not change this.

The code of the post processor program (written in php/html) can be found in Appendix 2.
01  -no failure-
02  -might be a red value-  S\_T\_hub
02  -might be a red value-  S\_T\_motor
02  -might be a red value-  S\_V\_battery
02  C\_pitch
02  S\_H\_z\_motor
02  S\_I\_motor
02  S\_s\_brake
02  S\_s\_elec
02  S\_theta\_motor
03  -might be a red value-  C\_pitch  S\_H\_z\_motor
03  -might be a red value-  C\_pitch  S\_I\_motor
03  C\_pitch  S\_s\_brake
03  C\_pitch  S\_s\_elec
03  C\_pitch  S\_theta\_motor
03  S\_H\_z\_motor  S\_I\_motor
03  S\_H\_z\_motor  S\_s\_brake
03  S\_H\_z\_motor  S\_s\_elec
03  S\_H\_z\_motor  S\_theta\_motor
03  S\_I\_motor  S\_s\_brake
03  S\_I\_motor  S\_s\_elec
03  S\_I\_motor  S\_theta\_motor
03  S\_s\_brake  S\_s\_elec
03  S\_s\_brake  S\_theta\_motor
03  S\_s\_elec  S\_theta\_motor

Figure 21 A print screen of the post processor output
4.3 Experiment with the QDM

This experiment is to show what is possible with the QDM as a tool to assess a sensor network. First, the used input for the model is explained. Second, the results are presented and discussed.

The input files for the model consist of sensor and controller variables that correspond with the operation states (see Figure 18) for the healthy pitch mechanism. Each of the six operation states is simulated once. This way, the sensor network is tested in the different operation states of the wind turbine. The input files can be found in Appendix 3. The six tested operation states are:
1. Operation state aerodynamic brake
2. Operation state out wind
3. Operation state optimal pitch angle
4. Operation state pitch to stall
5. Transition state negative angle change (from optimal to aerodynamic brake)
6. Transition state positive angle change (from out wind to optimal)

The first two levels of the post processed output (see section 3.2.2 for more explanation) is printed in Table 19. The whole output can be found in Appendix 3. Two conclusions directly stand out from the output:

- In the first level of the output there is no failure. This means that the input files reflect a healthy wind turbine. In other words, by comparing the input values with the model, the conclusion is: There is no reason to assume unhealthy components. This is what was intended. If there was a failure, the input file did not reflect a healthy wind turbine or the model would contain faults.

- In the second level one can see in the output files two or three times the tag: "-might be a red value-". In Table 20 the second level of the output is summarised. In all six operation states there are no redundancies that cover the temperature sensor of the motor and the measured voltage. The temperature sensor of the hub has only redundancies in two operation states.
Table 19: Output of the 6 input files

### 1. Operation state aerodynamic brake:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hzmotor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_I_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
</tbody>
</table>

### 2. Operation state out wind:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_I_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
</tbody>
</table>

### 3. Operation state optimal pitch angle:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_I_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
</tbody>
</table>

### 4. Operation state pitch to stall:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_I_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
</tbody>
</table>

### 5. Transition state negative:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_I_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
</tbody>
</table>

### 6. Transition state positive:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_I_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
</tbody>
</table>

C_pitch: The whole pitch mechanism
S_T_hub: Temperature sensor located in the hub
S_T_motor: Temperature sensor located at the motor
S_V_battery: Volt meter located at the battery
S_Hz_motor: Vibration sensor located at the motor
S_I_motor: Ammeter measuring the amperes going through the motor
S_s_brake: Position of the brake
S_theta_motor: Rotational speed / pitch angle sensor of the blade
The temperature sensor in the hub is only backed up by sensor network redundancies at the operation states *out wind* and *optimal pitch angle*. In these two operation states, the electrical motor is for a long time not used and has time to cool down to the hub temperature. This way, the temperature sensor of the hub is backed up by the temperature sensor of the motor. This means that the hub temperature can be checked on healthiness in these two operation states.

The model warns that one or more physical variables could turn red totally undetected when the temperature sensor on the motor or the voltage sensor of the battery would get faulty. There are three options to improve this situation.

- First, the model could be improved by coming up with additional network redundancies (see 3.1.2). As good as the model is, as good is the sensor network. By using more network redundancies, the model improves and possibly can back up one of these sensors. For example, when it is known how long it takes to cool the motor down to the temperature of its surroundings, then by including a timer the motor temperature sensor could be checked by the hub temperature sensor in the operation states *out wind* and *optimal pitch angle*.

- Second, the sensor network could be expanded with adding an additional sensor. This new sensor would create new redundancies. These new redundancies should back-up at least one of these sensors (e.g. add another voltage sensor on the battery).

- Third, there could be thought of a regular check to discover any mall functioning of these sensors. For the voltage sensor could be thought of switching to battery supply and start the motor. This way, the voltage sensor, electricity switch, and the battery itself are checked.

The table below summarizes the output of the model:

<table>
<thead>
<tr>
<th>Operation State</th>
<th>Temperature Sensor Hub</th>
<th>Temperature Sensor Motor</th>
<th>Voltage Sensor Battery</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Aerodynamic brake</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
</tr>
<tr>
<td>2. Out wind</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
</tr>
<tr>
<td>3. Optimal pitch angle</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
</tr>
<tr>
<td>4. Pitch to stall</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
</tr>
<tr>
<td>5. Transition state neg.</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
</tr>
<tr>
<td>6. Transition state pos.</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
<td>- might be a red value-</td>
</tr>
</tbody>
</table>
5. Conclusions

There are four criteria on which an optimal sensor network can be assessed. (A sensor network consists of the individual sensors and a reasoning system. The reasoning system determines the representation of the physical world.)

- The sensor network for a wind turbine must provide the physical variables for control, protection system, and condition monitoring system. By comparing the variables that are supplied by the sensor network with the variables that are needed, this criterion can be assessed.
- The sensor network must provide redundant information for fault diagnosis. If the sensor network is able to do fault diagnosis, the down time of the wind turbine will decrease. In previous research (Huberts, 2008) a qualitative diagnostic model (QDM) was used for fault diagnosis. The QDM was capable to assess a sensor network on fault detection.
- The sensor network must have redundant information to have a sufficient high chance of detecting limit values. As long as variables do not reach their limit value it is safe to continue operation. A QDM with an additional health variable can detect situations where possible limit values remain undetected. Such a QDM is promising to use for assessing a sensor network.
- The sensor network has as few sensors as possible. The more sensors there are in the sensor network, the more complex the network becomes. High complexity of the sensor network increases costs for designing, building and maintaining the sensor network. An optimal sensor network has as few sensors as possible.

The QDM with an additional health variable as is presented in this research is able to detect situations where possible limit values remain undetected (third criterion of an optimal sensor network).
- The output shows how many sensors and components must be broken before a limit value could remain undetected. By knowing the chance that a component gets broken, the chance of ending up in this situation could be calculated.
- The output of the QDM shows in which situations there could be limit values. It is directly clear which sensors need some back up by other sensors to improve the sensor network.
6 Recommendations

To validate if the qualitative model represents a wind turbine is difficult. A small error by the programmer is often difficult to detect. When the model becomes more complex, it is difficult to verify if one possible variable combination leads to correct output of the QDM. To check the whole model by verifying all the variable combinations is almost impossible. Possibilities of verifying the model lie in building the qualitative model with small building blocks that are completely checked. More research on verifying the model would make the QDM more reliable.

In order to validate the QDM itself as a tool for diagnoses and sensor network assessment, it should be tested with data of real wind turbines. This will give the opportunity to show that the QDM is a great tool with many possibilities.
References

Bos, A., A description of a diagnostic model of the Vulcain Rocket Engine (2005), Andre Bos, S&T corp, Delft, the Netherlands.


General Specification V82-1.65 MW MK II (2005) Vestas, Randers, Denmark.


Sokolsky, O., H.S. Hong (year unknown). Qualitative modeling of hybrid systems. University of Pennsylvania, USA.


Appendices
Appendix 1: Script of the QDM model

// Enum

// type sign
type sign = enum {neg, zero, pos};
// type sign is used by:
// electric motor: T1, theta1

// type sens
type sens = enum {zero, low, norm, red};
// type sens is used by:
// electric motor: T0, Hz0, A0

// type four
type four = enum {one, two, three, four};
// type four is used by:
// iv_ccontroller (one = aerobrake, two = out wind, three = optimal, four = stall)
// iv_t_ccontroller (one = new signal, two = transition state, three = transition state or at steady state, four = steady state)
// rv_t_ccontroller (one = new signal, two = transition state, three = transition state or at steady state, four = steady state)

// type six
type six = enum {one, two, three, four, five, six};
// type six is used by:
// electrical motor states

// type angle
type angle = enum {redneg, brake, zero, low, optimal, stall, redpos};
// type angle is used by:
// electrical motor: theta0

// type bool
// no need for declaration
// type used for healthynes sensors (S_*)
// iv_safetysystem

// End enum

// Calculation tables

system subs101_sign_magn_magn(sign a, sens b, c)
{
    // a = b - c
    // T'motor = Thub - Tmotor
    // Note: Thub cannot be zero or low

    // (b == sens.norm) && (c == sens.zero) => (a == ?)
(b == sens.norm) && (c == sens.low) => (a == sign.neg);
(b == sens.norm) && (c == sens.norm) => (a == sign.neg);
(b == sens.norm) && (c == sens.red) => (a == sign.neg);

(b == sens.red) && (c == sens.zero) => (a == sign.pos);
// (b == sens.red) && (c == sens.low) => (a == ?)
// (b == sens.red) && (c == sens.norm) => (a == ?)
// (b == sens.red) && (c == sens.red) => (a == ?)
}

system function_T1_motor(sign a, sens b, c, d)
{
    //T'motor = A + (T_hub - T_motor) * w_motor
    //T'motor = A + (T_motor) * w_motor
    //a = b + c * d
    //conditions:
    //T_hub = sens.norm
    //fan on motor working properly
    if (c == sens.zero)
    {
        (b == sens.zero) => (a == sign.zero);
        // ((b == sens.low) && (d == zero)) => (a == ?);
        // ((b == sens.low) && (d == sens.low)) => (a == ?);
        ((b == sens.low) && (d == sens.norm)) => (a == sign.pos);
        ((b == sens.low) && (d == sens.red)) => (a == sign.neg);
        // ((b == sens.norm) && (d == sens.red)) => (a == ?);
        // ((b == sens.red) && (d == sens.red)) => (a == ?);
        ((b == sens.red) && (d == sens.red)) => (a == sign.pos);
        ((b == sens.red) && (d == sens.red)) => (a == sign.neg);
    }
    if (c == sens.low)
    {
        (b == sens.zero) => (a == sign.neg);
        // ((b == sens.low) && (d == zero)) => (a == ?);
        // ((b == sens.low) && (d == sens.low)) => (a == ?);
        ((b == sens.low) && (d == sens.norm)) => (a == sign.neg);
        ((b == sens.low) && (d == sens.red)) => (a == sign.neg);
        // ((b == sens.norm) && (d == sens.red)) => (a == ?);
        // ((b == sens.red) && (d == sens.red)) => (a == ?);
        ((b == sens.red) && (d == sens.red)) => (a == sign.pos);
        ((b == sens.red) && (d == sens.red)) => (a == sign.neg);
    }
    if (c == sens.norm)
    {
        (b == sens.zero) => (a == sign.neg);
        // ((b == sens.low) && (d == zero)) => (a == ?);
        // ((b == sens.low) && (d == sens.low)) => (a == ?);
        ((b == sens.low) && (d == sens.norm)) => (a == sign.neg);
        ((b == sens.low) && (d == sens.red)) => (a == sign.neg);
    }
}
((b == sens.norm) && (d == sens.zero)) => (a == sign.pos);
((b == sens.norm) && (d == sens.low)) => (a == sign.pos);
((b == sens.norm) && (d == sens.norm)) => (a == sign.zero);
((b == sens.red) && (d == sens.zero)) => (a == sign.pos);
((b == sens.red) && (d == sens.low)) => (a == sign.pos);
((b == sens.red) && (d == sens.norm)) => (a == sign.pos);
//((b == sens.red) && (d == sens.red)) => (a == ?);
}

if (c == sens.red)
{
    (b == sens.zero) => (a == sign.neg);
    //((b == sens.low) && (d == zero)) => (a == ?);
    //((b == sens.low) && (d == sens.low)) => (a == ?);
    ((b == sens.low) && (d == sens.norm)) => (a == sign.neg);
    ((b == sens.low) && (d == sens.red)) => (a == sign.neg);
    ((b == sens.norm) && (d == sens.zero)) => (a != sign.neg);
    //((b == sens.norm) && (d == sens.low)) => (a == ?);
    ((b == sens.norm) && (d == sens.norm)) => (a == sign.neg);
    ((b == sens.norm) && (d == sens.red)) => (a == sign.neg);
    ((b == sens.red) && (d == sens.zero)) => (a != sign.neg);
    //((b == sens.red) && (d == sens.low)) => (a == ?);
    //((b == sens.red) && (d == sens.norm)) => (a == -);
    //((b == sens.red) && (d == sens.red)) => (a == ?);
}

system function_T1_motor2(sign a, sens b, c)
{
    //T'motor = I - (T_motor - T_hub)
    //a = b - (c - norm)
    //conditions:
    //T_hub = sens.norm
    //w == zero
    //T_hub = sens.norm
    //w == zero
    (b == sens.ZERO) && (c == sens.ZERO) => (a == sign.ZERO);
    (b == sens.ZERO) && (c != sens.ZERO) => (a == sign.neg);
    (b == sens.ZERO) && (c == sens.ZERO) => (a != sign.neg);
    (b == sens.ZERO) && (c != sens.ZERO) => (a = sign.pos);
    (b == sens.ZERO) && (c == sens.ZERO) => (a == sign.pos);
    (b == sens.ZERO) && (c != sens.ZERO) => (a != sign.pos);
    //end calculation tables
    //
    //Declaration of sensors and variables
    //
    system motor ()
    {
        //external physical variables
    }
bool S_T_hub;
attribute health (S_T_hub) = true;
attribute probability (S_T_hub) = S_T_hub ? 0.9995 : 0.0005;

sens sv_T0_hub;
attribute observable (sv_T0_hub) = true;
sens rv_T0_hub;
attribute observable (rv_T0_hub) = true;
(rv_T0_hub != sens.zero);
(rv_T0_hub != sens.low);
sign sv_T1_hub; // not in use
attribute observable (sv_T1_hub) = true;
sign rv_T1_hub; // not in use
attribute observable (rv_T1_hub) = true;

(S_T_hub) =>
(sv_T0_hub == rv_T0_hub) && (sv_T1_hub == rv_T1_hub);

bool S_V_battery;
attribute health (S_V_battery) = true;
attribute probability (S_V_battery) = S_V_battery ? 0.9995 : 0.0005;

bool sv_V_battery;
attribute observable (sv_V_battery) = true;
bool rv_V_battery;
attribute observable (rv_V_battery) = true;

(S_V_battery) =>
(sv_V_battery == rv_V_battery);

// physical variables at electro motor for pitch
bool S_T_motor;
attribute health (S_T_motor) = true;
attribute probability (S_T_motor) = S_T_motor ? 0.9995 : 0.0005;

sens sv_T0_motor;
attribute observable (sv_T0_motor) = true;
sens rv_T0_motor;
attribute observable (rv_T0_motor) = true;
sign sv_T1_motor;
attribute observable (sv_T1_motor) = true;
sign rv_T1_motor;
attribute observable (rv_T1_motor) = true;

(S_T_motor) =>
((sv_T0_motor == rv_T0_motor) && (sv_T1_motor == rv_T1_motor));

bool S_Hz_motor;
attribute health (S_Hz_motor) = true;
attribute probability (S_Hz_motor) = S_Hz_motor ? 0.9995 : 0.0005;

sens sv_Hz0_motor;
attribute observable (sv_Hz0_motor) = true;
sens rv_Hz0_motor;
attribute observable (rv_Hz0_motor) = true;

(S_Hz_motor) =>
(sv_Hz0_motor == rv_Hz0_motor);
//assumption: there are always some vibrations
(rv_Hz0_motor != sens.zero);

bool S_l_motor;
attribute health (S_l_motor) = true;
attribute probability (S_l_motor) = S_l_motor ? 0.9995 : 0.0005;

sens sv_IO_motor;
attribute observable (sv_IO_motor) = true;
sens rv_IO_motor;
attribute observable (rv_IO_motor) = true;

(S_l_motor) =>
(sv_IO_motor == rv_IO_motor);

bool S_theta_motor;
attribute health (S_theta_motor) = true;
attribute probability (S_theta_motor) = S_theta_motor ? 0.9995 : 0.0005;

angle sv_theta0_motor;
attribute observable (sv_theta0_motor) = true;
angle rv_theta0_motor;
attribute observable (rv_theta0_motor) = true;
sign sv_theta1_motor;
attribute observable (sv_theta1_motor) = true;
sign rv_theta1_motor;
attribute observable (rv_theta1_motor) = true;

sens sv_w0_motor;
attribute observable (sv_w0_motor) = true;
sens rv_w0_motor;
attribute observable (rv_w0_motor) = true;
sign sv_w1_motor;
attribute observable (sv_w1_motor) = true;
sign rv_w1_motor;
attribute observable (rv_w1_motor) = true;

(S_theta_motor) =>
((sv_theta0_motor == rv_theta0_motor) && (sv_theta1_motor == rv_theta1_motor));
(S_theta_motor) =>
((sv_w0_motor == rv_w0_motor) && (sv_w1_motor == rv_w1_motor));
((rv_theta1_motor == sign.zero) == (rv_w0_motor == sens.zero));

bool S_s_brake; // checks if internal brake is equipped
attribute health (S_s_brake) = true;
attribute probability (S_s_brake) = S_s_brake ? 0.9995 : 0.0005;

bool sv_s0_brake;
attribute observable (sv_s0_brake) = true;
bool rv_s0_brake;
attribute observable (rv_s0_brake) = true;

(S_s_brake) =>
(sv_s0_brake == rv_s0_brake);

bool S_s_elec; // checks if internal elec is equipped
attribute health (S_s_elec) = true;
attribute probability (S_s_elec) = S_s_elec ? 0.9995 : 0.0005;

bool sv_s0_elec; // true = connected with grid, false = connected with battery
attribute observable (sv_s0_elec) = true;
bool rv_s0_elec;
attribute observable (rv_s0_elec) = true;

(S_s_elec) =>
(sv_s0_elec == rv_s0_elec);

///////////////
// Input values
// these sensors fall outside the scope of the example
four iv_t_ccontroller;
attribute observable (iv_t_ccontroller) = true;
four rv_t_ccontroller;
attribute observable (rv_t_ccontroller) = true;
(iv_t_ccontroller == rv_t_ccontroller);

four iv_ccontroller;
attribute observable (iv_ccontroller) = true;
four rv_ccontroller;
attribute observable (rv_ccontroller) = true;
(iv_ccontroller == rv_ccontroller);

bool iv_grid;
attribute observable (iv_grid) = true;
bool rv_grid;
attribute observable (rv_grid) = true;
(iv_grid == rv_grid);
sens iv_HzO_nacelle;
attribute observable (iv_HzO_nacelle) = true;
sens rv_HzO_nacelle;
attribute observable (rv_HzO_nacelle) = true;
(rv_HzO_nacelle != sens.zero);
(rv_HzO_nacelle == rv_HzO_nacelle);

/\end Declaration of sensors and variables   //
//_______________________________________________________________________________//
//Behaviour
//_______________________________________________________________________________//

//healthy beahviour
//C_pitch is electrical motor and blade-bearing

bool C_pitch;
attribute health (C_pitch) = true;
attribute probability (C_pitch) = C_pitch ? 0.9995 : 0.0005;

//operation states
six st_motor;
attribute observable (st_motor) = true;
if (C_pitch)
{
//1 aerodynamic brake
if (st_motor == six.one)
{
(rv_HzO_motor == sens.low);
(rv_I0_motor == sens.zero);
(rv_theta0_motor == angle.brake);
(rv_theta1_motor == sign.zero);
    (rv_sO_brake == true);
}

//2 zero, out wind
if (st_motor == six.two)
{
(rv_HzO_motor == sens.low);
(rv_I0_motor == sens.zero);
(rv_theta0_motor == angle.zero);
(rv_theta1_motor == sign.zero);
    (rv_sO_brake == tme);
}

//3 Optimal pitch angle
if (st_motor == six.three)
{
(rv_HzO_motor == sens.low);
(rv_I0_motor == sens.zero);
(rv_theta0_motor == angle.optimal);
(rv_theta1_motor == sign.zero);
    (rv_sO_brake == true);
}
//4 Pitch to stall:
if (st_motor == six.four)
{
    (rv_Hz0_motor == sens.low)&&(rv_Hz0_motor == sens.norm));
    (rv_theta0_motor != sens.red);
    (rv_theta0_motor == angle.stall);
}
//5 Transition state theta = neg
if (st_motor == six.five)
{
    (rv_Hz0_motor == sens.norm);
    ((rv_IO_motor == sens.low)||(rv_IO_motor == sens.norm));
    (rv_theta1_motor == sign.neg);
        (rv_s0_brake == false);
}
//6 Transition state theta = pos
if (st_motor == six.six)
{
    (rv_Hz0_motor == sens.norm);
    ((rv_IO_motor == sens.low)||(rv_IO_motor == sens.norm));
    (rv_theta1_motor == sign.pos);
        (rv_s0_brake == false);
}

////////////////////////////////////////////////////////////////////////////////
//reactions on central controller inputs.

((rv_ccontroller != four.one)&&(rv_t_ccontroller != four.one)&&(st_motor != six.one));
((rv_ccontroller != four.one)&&(rv_t_ccontroller != four.one)&&(st_motor != six.two)); //=> central controller != healthy
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.one)&&(st_motor == six.three)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.one)&&(st_motor == six.four)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.one)&&(st_motor == six.five)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.one)&&(st_motor == six.six)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.two)&&(st_motor == six.one));
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.two)&&(st_motor == six.two)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.two)&&(st_motor == six.three)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.two)&&(st_motor == six.four)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.two)&&(st_motor == six.five)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.two)&&(st_motor == six.six)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.three)&&(st_motor == six.one)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.three)&&(st_motor == six.two)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.three)&&(st_motor == six.three)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.three)&&(st_motor == six.four)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.three)&&(st_motor == six.five)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.three)&&(st_motor == six.six)) => (C_pitch == false);
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.four)&&(st_motor == six.one)) => true
((rv_ccontroller == four.one)&&(rv_t_ccontroller == four.four)&&(st_motor == six.two)) => (C_pitch == false);
((rv_controller == four.one)&&(rv_t_controller == four.four)&&(st_motor == six.three)) => (C_pitch == false);
((rv_controller == four.one)&&(rv_t_controller == four.four)&&(st_motor == six.four)) => (C_pitch == false);
((rv_controller == four.one)&&(rv_t_controller == four.four)&&(st_motor == six.five)) => (C_pitch == false);
((rv_controller == four.one)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.two)&&(rv_t_controller == four.one)&&(stmotor == six.one)) => true
((rv_controller != four.two)||(rv_t_controller != four.one)||(st_motor != six.two));

//((rv_controller == four.two)&&(rv_t_controller == four.one)&&(st_motor == six.three)); \(\Rightarrow\) central controller != healthy
((rv_controller != four.two)||(rv_t_controller != four.one)||(st_motor != six.four)); \(\Rightarrow\) central controller != healthy

//((rv_controller == four.two)&&(rv_t_controller == four.one)&&(st_motor == six.five)) => true

//((rv_controller == four.three)&&(rv_t_controller == four.one)&&(st_motor == six.three)) => true
((rv_controller == four.three)&&(rv_t_controller == four.one)&&(st_motor == six.four)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.one)&&(st_motor == six.five)) => true

((rv_controller == four.three)&&(rv_t_controller == four.one)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.one)) => true
((rv_controller != four.three)||(rv_t_controller != four.four)||(st_motor != six.four));

//((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.five)) => true
((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.five)) => true

((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => (C_pitch == false);

((rv_controller != four.three)||(rv_t_controller != four.four)||(st_motor != six.five));

//((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.five)) => true
((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => true
((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.two)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => true

((rv_controller == four.three)&&(rv_t_controller == four.four)&&(st_motor == six.six)) => (C_pitch == false);

((rv_controller != four.three)||(rv_t_controller != four.three)||(st_motor != six.three));

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.three)) => true
((rv_controller != four.three)||(rv_t_controller != four.three)||(st_motor != six.four));

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.four)) => true

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => true
((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => true

((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => true

((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => (C_pitch == false);

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => true

((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => (C_pitch == false);

((rv_controller != four.three)||(rv_t_controller != four.three)||(st_motor != six.five));

//((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st MOTOR == six.five)) => true
((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => (C_pitch == false);

((rv_controller == four.three)&&(rv_t_controller == four.three)&&(st_motor == six.six)) => (C_pitch == false);


// Electric switch
// Operation state for grid failure is not included above. Therefore the following rule
// for healthy behaviour of the pitch mechanism can be added.

if (C_pitch == true)
{
    (iv_grid == true) => (rv_s0_elec == true);
    (iv_grid == false) => (rv_s0_elec == false);
}

//end behaviour

physics Temperature electric motor

// T'motor = I - (T_motor - T_hub) * w_motor - (T_motor - T_hub)
//(w_motor == zero)&&(l == zero)
//T\text{motor} = T\text{hub} - T\text{motor}
sign a1535;
sens b1535;
sens c1535;
system subs101_sign magn magn d01535;
d01535 (a1535, b1535, c1535);
if ((rv_w0\_motor == sens\_zero) & (rv_io\_motor == sens\_zero))
{
(a1535 == rv_T1\_motor);
(b1535 == rv_T0\_hub);
(c1535 == rv_T0\_motor);
}

//T\text{motor} = A + (T\text{hub} - T\text{motor}) \cdot w\text{motor}
//if hub is sens\_red I dont know what happens
//if hub is sens\_norm temp depends on A T and w\text{\_motor)
//assumption fan on motor is working properly \(C\_pitch\)
sign a1615;
sens b1615;
sens c1615;
sens d1615;
system function\_T1\_motor d01615;
d01615 (a1615, b1615, c1615, d1615);
if ((C\_pitch) && (rv_T0\_hub == sens\_norm))
{
(a1615 == rv_T1\_motor);
(b1615 == rv_IO\_motor);
(c1615 == rv_T0\_motor);
(d1615 == rv_w0\_motor);
}

//When \(C\_motor == \?\) and \(w\text{\_motor == zero}\) and \(T\text{hub == norm}\)
sign a1600;
sens b1600;
sens c1600;
system function\_T1\_motor2 d01602;
d01602 (a1600, b1600, c1600);
if ((rv_T0\_hub == sens\_norm) & (rv_w0\_motor == sens\_zero))
{
(a1600 == rv_T1\_motor);
(b1600 == rv_IO\_motor);
(c1600 == rv_T0\_motor);
}

//end physics Temperature electric motor

//end physics Temperature electric motor
/physics motion

//if there is motion there must be current
if (C_pitch)
{
  ((rv_theta1_motor == sign.pos) && (rv_w1_motor == sign.pos)) => (rv_l0_motor != sens.zero);
  ((rv_theta1_motor == sign.neg) && (rv_w1_motor == sign.neg)) => (rv_l0_motor != sens.zero);
  ((rv_theta1_motor == sign.pos) && (rv_w1_motor == sign.zero)) => (rv_l0_motor != sens.zero);
  ((rv_theta1_motor == sign.neg) && (rv_w1_motor == sign.zero)) => (rv_l0_motor != sens.zero);
  ///((rv_theta1_motor == sign.pos) && (rv_w1_motor == sign.neg)) => (rv_l0_motor == ??);
  ///((rv_theta1_motor == sign.neg) && (rv_w1_motor == sign.pos)) => (rv_l0_motor == ??);
  ((rv_theta1_motor == sign.zero) && (rv_w1_motor == sign.zero)) => (rv_l0_motor == sens.zero);
  ((rv_theta1_motor == sign.pos) && (rv_w1_motor == sign.pos)) => (rv_l0_motor == sens.zero);

  ((rv_theta1_motor == sign.pos) && (rv_l0_motor == sens.zero)) => (rv_w1_motor == sign.neg);
  ((rv_theta1_motor == sign.neg) && (rv_l0_motor == sens.zero)) => (rv_w1_motor == sign.pos);
  ((rv_w1_motor == sign.pos) && (rv_l0_motor == sens.zero)) => (rv_theta1_motor != sign.pos);
  ((rv_w1_motor == sign.neg) && (rv_l0_motor == sens.zero)) => (rv_theta1_motor != sign.neg);
}

//no vibration without rotation

(rv_Hz0_nacelle == sens.red) => (rv_Hz0_motor == sens.red);

if ((rv_Hz0_nacelle != sens.red) && (C_pitch))
{
  (rv_w0_motor == sens.zero) => (rv_Hz0_motor == sens.low);
  (rv_w0_motor == sens.low) => ((rv_Hz0_motor == sens.low) || (rv_Hz0_motor == sens.norm));
  (rv_w0_motor == sens.norm) => (rv_Hz0_motor == sens.norm);
}

(rv_w0_motor == sens.red) => (rv_Hz0_motor == sens.red);

if (rv_Hz0_nacelle == sens.red)
{
  (rv_Hz0_motor == sens.low) => ((rv_w0_motor == sens.zero) || (rv_w0_motor == sens.low));
  (rv_Hz0_motor == sens.norm) => ((rv_w0_motor == sens.norm) || (rv_w0_motor == sens.low));
}

//no normal vibrations without current and without rotation
//when there are no external vibrations

((rv_l0_motor == sens.zero) && (rv_w0_motor == sens.zero)) => (rv_Hz0_motor != sens.norm);

//end physics motion

////////////////////////
//Physics internal brake

((rv_s0_brake == true) (rv_theta1_motor == sign.pos)) => ((rv_w1_motor == sign.neg) || (C_pitch == false));
((rv_s0_brake == true) (rv_theta1_motor == sign.neg)) => ((rv_w1_motor == sign.pos) || (C_pitch == false));
((rv_s0_brake == true) (rv_theta1_motor == sign.zero)) => ((rv_w1_motor == sign.zero) || (C_pitch == false));
bool Alarm;
attribute health (Alarm) = true;
attribute probability (Alarm) = Alarm ? 0.5 : 0.5;

(rv_T0_hub == sens.red) => (Alarm == false);
(rv_V_battery == false) => (Alarm == false);
(rv_T0_motor == sens.red) => (Alarm == false);
(rv_Hz0_motor == sens.red) => (Alarm == false);
(rv_I0_motor == sens.red) => (Alarm == false);
(rv_w0_motor == sens.red) => (Alarm == false);

(Alarm == false) => (rv_T0_hub == sens.red) ||
(rv_V_battery == false) ||
(rv_T0_motor == sens.red) ||
(rv_Hz0_motor == sens.red) ||
(rv_I0_motor == sens.red) ||
(rv_w0_motor == sens.red);

//end Alarms //


Appendix 2: Script of the post processor

```php
// get contents of a file into a string
$filename = "output.txt";
$handle = fopen($filename, "r");
$contents = fread($handle, filesize($filename));
fclose($handle);

$contents = str_replace("@ stop output", "", $contents);

$array1 = explode(", "$contents);

// get clean lines
foreach($array1 as $key=>$value){
    if ($key != 0)
        $line[] = "$value";
}

foreach ($line as $key=>$value){
    if ($key != 0)
        $string = explode(", ", $value);
        $chance[$key] = ($string[0]);
        $rest[$key] = ($string[1]);
}

foreach ($rest as $key=>$value){
    print "rest[$key] = $value<br>";
}

$deep = 1;
$n = $chance[1];
foreach ($chance as $key=>$value){
    if ($key > 0)
        {
```
if ($nu != $value) {
    $deep++;
    $nu = $value;
}
if ($deep < 10) {
    $chance2[$key] = "0":$deep;
} else {
    $chance2[$key] = $deep;
}

foreach ($chance2 as $key=>$value) {
    print "chance2[$key] = $value <br>
}
*/

foreach ($rest as $key=>$value) {
    $value3 = str_replace(" ", "", $value);
    $restarray = explode (" ", $value3);
    foreach ($restarray as $key2=>$value2) {
        if (ereg("false", $value2)) {
            if (ereg("Alarm", $value2)) {
                if ($rest2[$key] !=""){
                    $rest2[$key] = "__";
                    $rest2[$key] = "Alarm=false"
                } else {
                    if ($rest2[$key] !=""){
                        $rest2[$key] = "__";
                        $rest2[$key] = $value2;
                    }
                }
            } else {
                if ($rest2[$key] !=""){
                    $rest2[$key] = "__";
                    $rest2[$key] = $value2;
                }
            }
        } else {
            if ($rest2[$key] !="") {
                $rest2[$key] = "-no failure-";
            }
        }
    }
}
/*
foreach ($rest2 as $key=>$value) {
    print "rest2[$key] = $value <br>
}*/
foreach ($rest2 as $key=>$value) {
    $rest3[$key] = str_replace("=false", "", $value);
}

foreach ($rest3 as $key=>$value) {
    print "$rest3[$key] = $value <br>
}

foreach ($rest3 as $key=>$value) {
    unset ($array);
    $array = explode(" ", $value);
    sort ($array);
    $rest4[$key] = $chance2[$key] . "_" . implode("_", $array);
}

foreach ($rest4 as $key=>$value) {
    print "$rest4[$key] = $value <br>
}

sort($rest4);

// don't want that values with an alarm are repeated without an alarm
$telforbidden = 0;
foreach ($rest4 as $key=>$value) {
    if (ereg("Alarm", $value)) {
        $telforbidden++;
        unset ($array);
        $value2 = str_replace("Alarm", "", $value);
        $value2 = str_replace(" ", " ", $value2);
        $value2 = trim ($value2, "_");
        $forbidden[$telforbidden] = explode(" ", $value2);
        // print "<br>$forbidden[$telforbidden][0].".$forbidden[$telforbidden][1].".$forbidden[$telforbidden][2].".$forbidden[$telforbidden][3]."<br>
    }
}
foreach ($rest4 as $key=>$value) {
    //print "$key: $value <br>
    $forbidden[$key] = 0;
    unset ($array);
    $array = explode('_', $value);
    $countj = count($array);
    $i = 1;
    while ($i <= $selfforbidden) {
        $match = 1;
        $alarm = "t";
        $countf = count($forbidden[$i]);
        foreach ($forbidden[$i] as $keyf=>$valuef) {
            if ((+$keyf == 0) && ($array[0] * 1) == $deep) // not all lines of this chance are written in inputfile
                $match = $countf;
        break;
    }
    if ($keyf != 0) {
        foreach ($array as $keya=>$valuea) {
            if ($valuea == "Alarm")
                $alarm = "t";
            if ($valuef == $valuea)
                $match++;
        }
    }
    }
    if ($match == $countf) {
        //print "$key is forbidden<br>
        $teller++;
        $forbidden[$key]++;
        if (($alarm == "t") && ($forbidden[$key] == 2)) {
            $forbiddenline[] = $key;
            break;
        }
        if (($alarm == "f") && ($forbidden[$key] == 1)) {
            $forbiddenline[] = $key;
            break;
        }
    }
}
Si++;

$i = 0;
Print "<table>";
foreach ($rest4 as $key=>$value)
{
    if ($forbidden[$i] == $key)
        $i++;
    else
        print "<tr><td».str_replace("_", "<td>", $value);
}
Print "</table>";
/

foreach ($rest4 as $key=>$value)
{
    $rest5 = str_replace("_", "<td>", $rest4);
}
print "<table>";
foreach ($rest5 as $key=>$value)
{
    $print = "t";
    foreach ($forbidden as $key2=>$value2)
    {
        if ($value == $value2)
        {
            $print = "f";
        }
    }
    if ($print == "t")
    {
        print "<tr><td";
        print "$value";
    }
}
print "</table>";
/*
print "<table>";
foreach ($rest5 as $key=>$value)
{
    print "<tr><td";
    print "$value";
}
print "</table>";
*/
Appendix 3: Experiment inputs and outputs

1 Operation state aerodynamic brake:
Input:
cut solutions 100
set sv_T0_hub norm
set sv_V_battery true
set sv_T0_motor norm
set sv_T1_motor neg
set sv_Hz0_motor low
set sv_I0_motor zero
set sv_theta0_motor brake
set sv_theta1_motor zero
set sv_w0_motor zero
set sv_w1_motor zero
set sv_s0_brake true
set sv_s0_elec true
set iv_ccontroller one
set iv_t_ccontroller four
set iv_grid true
set iv_Hz0_nacelle norm
fm
<table>
<thead>
<tr>
<th>01</th>
<th>-no failure-</th>
</tr>
</thead>
<tbody>
<tr>
<td>02</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td></td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td></td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td></td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>02</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td></td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
</tbody>
</table>
2. Operation state out wind:
Input:
cut solutions 100
set sv_T0_hub norm
set sv_V_battery true
set sv_T0_motor zero
set sv_T1_motor zero
set sv_Hz0_motor low
set sv_I0_motor zero
set sv_theta0_motor zero
set sv_theta1_motor zero
set sv_w0_motor zero
set sv_w1_motor zero
set sv_s0_brake true
set sv_s0_elec true
set iv_ccontroller two
set iv_t_ccontroller four
set iv_grid true
set iv_Hz0_nacelle norm
fm
Output:

| 01 | -no failure-         | 01 | -might be a red value- S_T_motor |
| 02 | -might be a red value- S_V_battery  |
| 02 | C_pitch              |
| 02 | S_Hz_motor           |
| 02 | S_l_motor            |
| 02 | S_T_hub              |
| 02 | S_s_brake            |
| 02 | S_s_elec             |
| 02 | S_theta_motor        |
| 03 | -might be a red value- C_pitch S_Hz_motor |
| 03 | C_pitch S_l_motor    |
| 03 | C_pitch S_T_hub      |
| 03 | C_pitch S_s_brake    |
| 03 | C_pitch S_s_elec     |
| 03 | C_pitch S_theta_motor|
| 03 | S_Hz_motor S_l_motor |
| 03 | S_Hz_motor S_T_hub   |
| 03 | S_Hz_motor S_s_brake |
| 03 | S_Hz_motor S_s_elec  |
| 03 | S_Hz_motor S_theta_motor |
| 03 | S_l_motor S_T_hub    |
| 03 | S_l_motor S_s_brake  |
| 03 | S_l_motor S_s_elec   |
| 03 | S_l_motor S_theta_motor |
| 03 | S_T_hub S_s_brake    |
| 03 | S_T_hub S_s_elec     |
| 03 | S_T_hub S_theta_motor|
| 03 | S_s_brake S_s_elec   |
| 03 | S_s_brake S_theta_motor |
| 03 | S_s_elec S_theta_motor |
3. Operation state optimal pitch angle:
Input:
cut solutions 100
set sv_T0_hub norm
set sv_V_battery true
set sv_T0_motor zero
set sv_T1_motor zero
set sv_Hz0_motor low
set sv_I0_motor zero
set sv_theta0_motor optimal
set sv_theta1_motor zero
set sv_w0_motor zero
set sv_w1_motor zero
set sv_s0_brake true
set sv_s0_elec true
set iv_cccontroller three
set iv_t_cccontroller four
set iv_grid true
set iv_Hz0_nacelle norm
fm
Output:

| 01 | -no failure- |
| 02 | -might be a red value- | 85 |
| 02 | -might be a red value- | S_T_motor |
| 02 | -might be a red value- | S_V_battery |
| 02 | C_pitch |
| 02 | S_Hz_motor |
| 02 | S_l_motor |
| 02 | S_T_hub |
| 02 | S_s_brake |
| 02 | S_s_elec |
| 02 | S_theta_motor |
| 03 | -might be a red value- | C_pitch | S_Hz_motor |
| 03 | C_pitch | S_l_motor |
| 03 | C_pitch | S_T_hub |
| 03 | C_pitch | S_s_brake |
| 03 | C_pitch | S_s_elec |
| 03 | C_pitch | S_theta_motor |
| 03 | S_Hz_motor | S_l_motor |
| 03 | S_Hz_motor | S_T_hub |
| 03 | S_Hz_motor | S_s_brake |
| 03 | S_Hz_motor | S_s_elec |
| 03 | S_Hz_motor | S_theta_motor |
| 03 | S_l_motor | S_T_hub |
| 03 | S_l_motor | S_s_brake |
| 03 | S_l_motor | S_s_elec |
| 03 | S_l_motor | S_theta_motor |
| 03 | S_T_hub | S_s_brake |
| 03 | S_T_hub | S_s_elec |
| 03 | S_T_hub | S_theta_motor |
| 03 | S_s_brake | S_s_elec |
| 03 | S_s_brake | S_theta_motor |
| 03 | S_s_elec | S_theta_motor |
4. Operation state pitch to stall:
Input:
cut solutions 100
set sv_T0_hub norm
set sv_V_battery true
set sv_T0_motor norm
set sv_T1_motor zero
set sv_Hz0_motor norm
set sv_I0_motor norm
set sv_theta0_motor stall
set sv_theta1_motor pos
set sv_w0_motor norm
set sv_w1_motor neg
set sv_s0_brake false
set sv_s0_elec true
set iv_ccontroller four
set iv_t_ccontroller four
set iv_grid true
set iv_Hz0_nacelle norm
fm
Output:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>C_pitch</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_l_motor</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
<td></td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
<td></td>
</tr>
<tr>
<td>03</td>
<td>-might be a red value-</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>-might be a red value-</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
<td>S_theta_motor</td>
</tr>
</tbody>
</table>
5. Transition state negative:
Input for (Optimal -> Aerodynamic brake):
cut solutions 100
set sv_T0_hub norm
set sv_V_battery true
set sv_T0_motor low
set sv_T1_motor pos
set sv_Hz0_motor norm
set sv_I0_motor norm
set sv_theta0_motor low
set sv_theta1_motor neg
set sv_w0_motor low
set sv_w1_motor neg
set sv_s0_brake false
set sv_s0_elec true
set iv_ccontroller one
set iv_t_ccontroller one
set iv_grid true
set iv_Hz0_nacelle norm
fm
<table>
<thead>
<tr>
<th>Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>01  -no failure-</td>
</tr>
<tr>
<td>02  -might be a red value- S_T_hub</td>
</tr>
<tr>
<td>02  -might be a red value- S_T_motor</td>
</tr>
<tr>
<td>02  -might be a red value- S_V_battery</td>
</tr>
<tr>
<td>02  C_pitch</td>
</tr>
<tr>
<td>02  S_Hz_motor</td>
</tr>
<tr>
<td>02  S_I_motor</td>
</tr>
<tr>
<td>02  S_s_brake</td>
</tr>
<tr>
<td>02  S_s_elec</td>
</tr>
<tr>
<td>02  S_theta_motor</td>
</tr>
<tr>
<td>03  -might be a red value- C_pitch S_Hz_motor</td>
</tr>
<tr>
<td>03  -might be a red value- C_pitch S_I_motor</td>
</tr>
<tr>
<td>03  C_pitch S_s_brake</td>
</tr>
<tr>
<td>03  C_pitch S_s_elec</td>
</tr>
<tr>
<td>03  C_pitch S_theta_motor</td>
</tr>
<tr>
<td>03  S_Hz_motor S_I_motor</td>
</tr>
<tr>
<td>03  S_Hz_motor S_s_brake</td>
</tr>
<tr>
<td>03  S_Hz_motor S_s_elec</td>
</tr>
<tr>
<td>03  S_Hz_motor S_theta_motor</td>
</tr>
<tr>
<td>03  S_I_motor S_s_brake</td>
</tr>
<tr>
<td>03  S_I_motor S_s_elec</td>
</tr>
<tr>
<td>03  S_I_motor S_theta_motor</td>
</tr>
<tr>
<td>03  S_s_brake S_s_elec</td>
</tr>
<tr>
<td>03  S_s_brake S_theta_motor</td>
</tr>
<tr>
<td>03  S_s_elec S_theta_motor</td>
</tr>
</tbody>
</table>
6. Transition state positive
Input for (Out wind -> Optimal):

cut solutions 100
set sv_T0_hub norm
set sv_V_battery true
set sv_T0_motor norm
set sv_T1_motor zero
set sv_Hz0_motor norm
set sv_I0_motor norm
set sv_theta0_motor low
set sv_theta1_motor pos
set sv_w0_motor norm
set sv_w1_motor zero
set sv_s0_brake false
set sv_s0_elec true
set iv_ccontroller three
set iv_t_ccontroller three
set iv_grid true
set iv_Hz0_nacelle norm
fm
<table>
<thead>
<tr>
<th></th>
<th>Output:</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>-no failure-</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td>02</td>
<td>S_T_hub</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td>02</td>
<td>S_T_motor</td>
</tr>
<tr>
<td>02</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td>02</td>
<td>S_V_battery</td>
</tr>
<tr>
<td>02</td>
<td>_C_pitch</td>
</tr>
<tr>
<td>02</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>02</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>02</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>02</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>02</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>-might be a red value-</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>C_pitch</td>
</tr>
<tr>
<td>03</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
<tr>
<td>03</td>
<td>S_l_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_theta_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_brake</td>
</tr>
<tr>
<td>03</td>
<td>S_Hz_motor</td>
</tr>
<tr>
<td>03</td>
<td>S_s_elec</td>
</tr>
</tbody>
</table>