NTNU

Norwegian University of
Science and Technology

The Design and Implementation of a Control System for an Autonomous Miniature Drilling Rig

Mikkel Leite Arnø

Master of Science in Industrial Cybernetics
Submission date: June 2018
Supervisor: Lars Imsland, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Preface

This master's thesis is written for the Department of Engineering Cybernetics at NTNU, in cooperation with the Department of Geoscience and Petroleum. The work presented is in relation to the Drillbotics competition final, which the NTNU team qualified for in the design phase last semester. The NTNU Drillbotics team consists of myself and four petroleum students: Alexander Handeland, Sebastian Knoop, Andreas Thuve and Per Øystein Turøy.

The Drillbotics project has been educational and challenging, providing an arena to couple theory with practical problem solving, along with interdisciplinary teamwork between cybernetics and petroleum students. The mother lode of my work has consisted of programming from scratch the control system along with assisting programs like automatic file saving in LabVIEW in cooperation with one of the petroleum students, Andreas Thuve. LabVIEW was new to both of us, so at the start of the semester, considerable time was spent taking National Instruments' LabVIEW courses, Core 1, which was online, and Core 2, which was arranged at Gløshaugen. Along the way, setting up communication infrastructure, mechanical fixes on the rig and working around hardware limitations has been time consuming. Several parts, like the ballscrew and the Ethernet/IP module on the hoisting motor drive have been replaced during the semester. Also, an Ethernet/IP to Modbus converter and a cDAQ chassis plus modules have been acquired. At these occurrences, contact with vendors, and providing them with the details of our system has been essential to ensure that the new parts were compatible with our setup. Due to these delays priorities have been made, meaning that some features on the design have been omitted or not tested as thoroughly as we have intended.

As engineering students, the team has had an evolving vision of the end result, and how we wanted the rig to function. However, tasks like selecting and setting up communication protocols with actuators, and mechanical fixes have been outside the team's immediate competence. I would like to thank Noralf Vedvik, Steffen Wærnes Moen, Terje Bjerkan and Håkon Myhren for their help in this capacity, as without their insight and expertise resolving these practical issues, this project would not have been possible.

I would also like to thank the team supervisors, Lars Struen Imsland, Alexey Pavlov, Sigve Hovda and Lars Berge Gjersvik. The biweekly status meetings have been very helpful in terms of project planning and in learning from their experience in control theory and the drilling industry. Continuous contact with the supervisors has aided in keeping up the progression throughout the semester, and helped us avoid various pitfalls in our design.

Lastly, I would like to thank our sponsors. Equinor, formerly known as Statoil, has provided the team with funding for the project, a necessity for hardware and mechanical upgrades on the design. Lyng Drilling has guided the team

in drillbit design, and manufactured it as well. Sorte Skiferbrudd, Nidaros Domkirke Restaureringsarbeid and Heimdal Naturstein have all provided us with various rock types, allowing us to set up our own formations for testing of the rig. All these contributions are greatly appreciated, and have been paramount to our work.

Abstract

The work presented in this thesis is concerned with the control system of the autonomous miniature drilling rig NTNU has entered and competed with in the international Drillbotics competition. A brief listing of the status of automation in the drilling industry is given, along with challenges, both current and in the future. Furthermore, relevant theory regarding digital filter design, Cohen-Coon PID tuning and least squares estimation is provided for a deeper understanding of the various aspects of the design.

The entire implementation of the system is given, including a brief introduction to LabVIEW and the underlying motivation for choosing this software. An overview of the setup provides the details revolving hardware, software and communication protocols in the design, including actuators and sensors. The process of selecting sampling frequency and the self-coding of digital third-order Butterworth filters is explained. Then, an overview of the various control schemes is given before explaining a fully autonomous state machine, of which a simplified version was used on the competition day. As the team's work is handed over to next year's team, measures to document data well, along with modularizing and describing the LabVIEW code well have been taken. Also, the graphical user interface for the autonomous program is presented separately, explaining the train of thought when coming up with a user-friendly design. Safety has been a priority during this project, so a shutdown sequence also functioning as an emergency shutdown has been implemented in all LabVIEW programs. As previously mentioned, documenting results has also been a priority, which is why an automatic file saving module was coded and implemented in all programs to systematically save all drilling data.

After the implementation is explained, the results of the most important tests are presented. This includes limit tests, PID controller tuning, formation response tests to determine drilling parameters to move forward with and estimator tests to identify drilled formation. Finally, the crown of the work, the results for the two state machines are discussed. One section includes the Drillbotics competition drilling session, where the simplified state machine without the estimator was used. The final section elaborates on a drilling run performed after the competition, where the estimator is put to the test.

Sammendrag

Arbeidet som presenteres i denne avhandlingen er opptatt av styringssystemet til den autonome miniatyrboreriggen NTNU har konkurrert med i den internasjonale konkonkurransen Drillbotics. En kort oversikt over statusen for automatisering i boreindustrien er gitt, sammen med utfordringer, både nåværende og i fremtiden. Videre er relevant teori om digitalt filterdesign, Cohen-Coon PID-tuning og least squares estimering gitt for en dypere forståelse av de ulike aspektene av designet.

Implementeringen av hele systemet er gitt, inkludert en kort introduksjon til LabVIEW og den underliggende motivasjonen for å velge denne programvaren. En oversikt over oppsettet gir detaljer om maskinvare-, programvare- og kommunikasjonsprotokoller i designet, inkludert aktuatorer og sensorer. Prosessen med å velge samplingfrekvens og selvkodingen av digitale tredjeordens Butterworth-filtre er forklart. Deretter gis en oversikt over de ulike kontrollmetodene før en helautonom tilstandsmaskin presenteres, hvorav en forenklet versjon ble brukt på konkurransedagen. Siden lagets arbeid blir overlevert til neste års lag, har det blitt gjort tiltak for å dokumentere data godt, samt modularisering og god beskrivelse av LabVIEW-koden. Også det grafiske brukergrensesnittet for det autonome programmet presenteres separat, og forklarer tankegangen når det kommer til et brukervennlig design. Sikkerhet har vært en prioritet under dette prosjektet, så en avslutningssekvens som også fungerer som en nødstans har blitt implementert i alle LabVIEW-programmer. Som tidligere nevnt har dokumentasjon av resultater også vært en prioritet, og en automatisk fillagringsmodul ble derfor kodet og implementert i alle programmer for systematisk lagring av all boredata.

Etter at implementeringen er forklart, presenteres resultatene av de viktigste testene. Dette inkluderer tester av systemets grenser, PID kontroller tuning, formasjonsresponstester for å bestemme boreparametere for å gå videre med og estimatortester for å identifisere boret formasjon. Til slutt diskuteres arbeidets krone, resultatene for de to tilstandsmaskinene. En seksjon inkluderer Drillbotics konkurranseboringen, hvor den forenklede tilstandsmaskinen uten estimator ble brukt. Den siste seksjonen utdyper på en boring som utføres etter konkurransen, hvor estimatoren blir testet.

Contents

Chapter 1

Introduction

1.1 Drillbotics

It is hard to think of an industry more demanding and more complex than the oil and gas industry. The demanding environments at which oil reserves can be located have been a major drive in the further development of technology and expertise which has even been applied to the space industry. Depleting reserves and increasing worldwide energy demand forces companies to move their operations to even more remote and challenging environments where the cost of operations are even higher. Though economically feasible production in these environments are challenging, companies willing to develop innovative technology to lower their costs and mitigate risk can definitely overcome these challenges [1].

Among initiatives to accelerate development of innovative technology in the oil and gas industry is the Drilling Systems Automation Technical Section (DSATS), which is a technical section of the Society of Petroleum Engineers (SPE) dedicated to development of automation techniques and systems to improve both productivity and safety of the well drilling process. DSATS arranges the Drillbotics competition, which is an international competition for university students to design and construct a miniature drilling rig to autonomously drill a 60 cm rock sample. This rock sample is unknown to the contenders, hidden in a wooden crate, and has previously consisted of formations such as cement, asphalt, rubber mats and bathroom tiles. The teams are judged on a variety of areas, such as safety, robustness of the control scheme, performance and the quality of the wellbore. The bottom line of the competition however, is to "drill a vertical well as quickly as possible while maintaining borehole quality and integrity of the drilling rig and drillstring", as stated on the Drillbotics webpage [2].

1.2 Automation in the Drilling Industry

Drilling automation is the control of the drilling system and drilling operation, combining several subsystems like the drilling rig, drillstring, drilling fluid and the downhole bottomhole assembly (BHA). The aim of drilling automation is to reduce human intervention, increase quality and efficiency of operations, and improve personnel safety. It is a challenging task to safely automate these operations, as they often take place in geopressured, possibly corrosive, rheologically complex lithologies. Another challenge is that drilling operations involve several different companies working together: equipment suppliers, service companies, drilling contractor and operator. Each of these companies may bring their own hardware and software, which is an inhibition to progress in drilling automation since devices have no means to communicate across organizations.

As mentioned previously, drilling operations consist of several subsystems. The same goes for drilling automation: it is not a single system, but rather a grouping of many automated subsystems. These subsystems are available in different levels of automation: **Monitor**, **Advice**, **Control** and **Autonomous**, where autonomous subsystems should at least be able to run without human intervention for some period of time. An example of a subsystem in the **Autonomous** level is the downhole rotary-steerable system, which only needs supervisory control, while directional control on surface is at best available in the **Advice** level of autonomy. Drilling automation is a reality, as the required automation technology exists. The challenge ahead is to further develop the level of autonomy of the separate subsystems within drilling automation to increase efficiency of operations in a safe manner.

There are many reasons why automation of the drilling process is so appealing. An automated system will continuously perform a repeated task at a consistent performance level, never tiring. An example is the makeup of connections, which when made manually can vary from taking under a minute in some cases, while taking over 10 minutes in other cases, while an autonomous connections system would predictably make the connections in a constant pace from one time to the next. This consistency extends to drilling several similar profile wells per oil field, where autonomous systems will continuously perform to minimize cost and risk. Another example could be rate of penetration (ROP) improvement systems, which have convincingly outperformed manual and advice mode drilling, as the autonomous system continuously and frequently adjusts drilling parameters to adapt to changing environments, while the driller is often less willing to make so many adjustments.

Complex well profiles with narrow bottomhole pressure margins is another case where automation could outperform human drillers as such wells could require more frequent adjustments. Moving forward in time, we have access to more and more data and real-time measurements, which can be of great

assistance in telling us something about the downhole conditions and the stability of the well. However, displaying these large volumes of raw data in a user interface is more likely to overwhelm the driller than be of any help. This is where automated systems come into play, converting raw data into useful information for the driller, or forming decisions and managing control on its own, depending on the system's level of automation.

Another drive for drilling automation is that expert resources are not currently readily available for all operations, and several experts are retiring from the industry these years. This loss of competence can be mitigated by further development of autonomous drilling systems. Currently, drilling automation is progressing towards systems requiring less understanding of the drilling operation to operate. A situational awareness among operating crew, and an understanding of the features and limitations of the automatic systems will still be of great importance, but as the level of autonomy of these systems rises, the system itself could take on more and more of the role as expert resource.

An obvious effect of increasing levels of autonomy is a lower presence of human resources on site. This is an HSE benefit, as it reduces the number of people working in hazardous areas. It is however also important to remember that implementing autonomous systems that are receptive to remote monitoring or control are exposed to cyberattacks, which could compromise the safety of the drilling operation, both for human resources on site and the environment. There are available industry standards to assist in the safe implementation of autonomous systems, such as the International Organization for Standardization (ISO) and the International Electrotechnical Commission. These organizations provide standards for organizational security, technical security, risk assessment/management, and ensuring security continuously. It should be noted that people are an important part of the autonomy loop, and can provide useful experience and judgment in critical phases or in case of system failures. This means that although automation can take over repetitive, dangerous and complex tasks, it should aim to utilize the strengths of both human and machine. [3]

1.3 Motivation and Goals

As the work presented in this master's thesis is related to the Drillbotics 2018 competition, a natural source of motivation is the drive to deliver a well tested, high-performance set-up for the competition day. The design of a robust control system capable of handling a variety of contingencies and drilling dysfunctions is a prerequisite to do well in the competition, and thus also a cornerstone goal for this thesis. The list below summarizes some of the main tasks related to achieving this goal:

- Setting up the communication infrastructure for the system.
- Choosing a suitable control scheme and PID controller tuning.
- Setting up manual PID modes for test-drilling.
- Implementing efficient filtering for sensor data.
- Implementing an estimator to determine drilled formation.
- Design of a user-friendly and intuitive GUI.
- Setting up automatic file saving for post-drilling analysis.
- Implementing a functioning state machine, representing the different phases of drilling.

Being a multi-disciplinary team consisting of a cybernetics student and petroleum students, the Drillbotics competition allows for learning across field of study, as well as the solving of practical challenges as means to a very specific goal. The competition also calls for good communication within the group and planning ahead to properly manage the project. These conditions are similar to those an engineer can expect to face during his or her career, which is also a motivation factor.

Chapter 2

Relevant Theory

2.1 Filtering

2.1.1 The Butterworth Filter

The Butterworth filter family is well known within signal conditioning. They are also referred to as maximally flat magnitude filters, as their frequency response contains no ripples in the passband, meaning that the passed frequencies are equally amplified, all the way up towards the cut-off frequency. The complexity of filters are defined by their order, where a first-order lowpass filter has a transition band slope of -20 dB/dec, and an n-th order lowpass filter has a transition band slope of -20n dB/dec. In several control system applications, the transition band slope of a first-order filter is insufficient to effectively attenuate the unwanted high-frequency components of the signal, warranting higher order filters.

As mentioned above, the Butterworth filter has the desirable trait of a flat passband. This comes at the expense of a wide transition band, unlike for example Elliptic filters, which have steeper transition bands but ripples across the passband. To some extent, the wide transition band of the Butterworth filter can be shortened by increasing the order of the filter, but this in turn contributes to poor phase characteristics, i.e. time delay. [4]

To find the transfer function of an n-th order Butterworth filter, one can use the normalized Butterworth polynomials, which are normalized around $\omega_c = 1$, and then have the general form:

$$B_n(s) = \prod_{k=1}^{\frac{n}{2}} \Big[s^2 - 2s\cos\big(\frac{2k+n-1}{2n}\pi\big) + 1 \Big], \text{ n = even} \tag{2.1}$$

$$B_n(s) = (s+1) \prod_{k=1}^{\frac{n-1}{2}} \Big[s^2 - 2s\cos\big(\frac{2k+n-1}{2n}\pi\big) + 1 \Big], \text{ n = odd.} \tag{2.2}$$

Table 2.1 shows the normalized Butterworth polynomials for the first-, second-, and third-order versions of the filter.

n	Denominator of Transfer Function
1	$s+1$
2	$s^2 + 1.4142s + 1$
3	$(s+1)(s^2+s+1)$

Table 2.1: Normalized transfer function denominator polynomials of different order Butterworth filters.

These can in turn be used to calculate the transfer function for a Butterworth filter of any cut-off frequency ω_c by setting:

$$H(s) = \frac{K}{B_n(\alpha)}, \tag{2.3}$$

where $\alpha = \frac{s}{\omega_c}$ and K is the amplification along the passband [5].

2.1.2 The Bilinear Transform

The bilinear transform is a method used in digital signal processing when transforming continuous-time signals to discrete-time. It can used to convert the transfer function of an analog filter to a transfer function for a digital filter by mapping the points of the s-plane to corresponding points in the z-plane. Points on the imaginary axis are mapped to the unit circle $|z| = 1$, and the entire left half-plane inside it.

The exact transform from the z-plane to s-plane is as follows:

$$z = \mathrm{e}^{sT}, \text{ where T is the sampling time.} \tag{2.4}$$

The bilinear transform utilizes a first-order Padè approximation of the exact transform, leading to:

$$z \approx \frac{1 + sT/2}{1 - sT/2}, \tag{2.5}$$

meaning that

$$s \approx \frac{2}{T}\frac{z-1}{z+1}. \tag{2.6}$$

In other words, the bilinear transform from the s-plane to the z-plane is performed by plugging Equation (2.6) into the transfer function of the analog filter [6]:

$$H(z) = H(s)\big|_{s=\frac{2}{T}\frac{z-1}{z+1}}. \tag{2.7}$$

After obtaining the transfer function $H(z)$ of the digital filter, it can be arranged into the following form:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0 + b_1 z^{-1} + ... + b_n z^{-N}}{1 + a_1 z^{-1} + ... + a_n z^{-M}}, \tag{2.8}$$

which is the form of a recursive filter. The difference equation used further to implement the filter is then [7]:

$$y[n] = -\sum_{k=1}^{M} a_k y[n-k] + \sum_{k=1}^{N} b_k x[n-k], \tag{2.9}$$

where

$y[n] =$ current filtered output,
$y[n-k] =$ k-th to last filtered output,
$x[n] =$ current raw input,
$x[n-k] =$ k-th to last raw input.

2.2 Cohen-Coon PID Tuning

Cohen-Coon is a PID tuning method known to work well on most self-stabilizing processes, which means that the process stabilizes at some equilibrium dependent on process design and controller output. This tuning method

is based on open loop testing to find a transfer function for the system, which is then used to find appropriate PID gains. A particularly common transfer function form used is the first-order plus dead time (FOPDT), which has the form:

$$H(s) = \frac{K\mathrm{e}^{-\theta s}}{\tau s + 1}, \tag{2.10}$$

where K is the process gain, θ is the dead time, and τ is the time constant.

Typically, a step response test is performed for this tuning. When the process has achieved steady state, a step change is made in the controller output (CO), large enough to move the process variable (PV) out of the process noise and disturbance level. Then, the controller output and process variable are ranged between 0-100 % of upper and lower calibration limits, which are chosen based on the expected values of controller output and process variable in the closed loop. Then, the process gain K is calculated by:

$$K = \frac{\Delta \mathrm{PV}_{\%}}{\Delta \mathrm{CO}_{\%}}. \tag{2.11}$$

After this, the point of inflection is found on the process variable curve, and a tangential line is drawn through the curve at this point. The dead time θ is then given as the difference in time from the intersection between the tangential line and the original process variable level before the step change and the time of the step change in controller output. The next step is to find the time constant τ. First, the 63 % value of total PV change is calculated, and the time constant τ is then given as the time difference between the time at this point and the time at the end of the dead time.

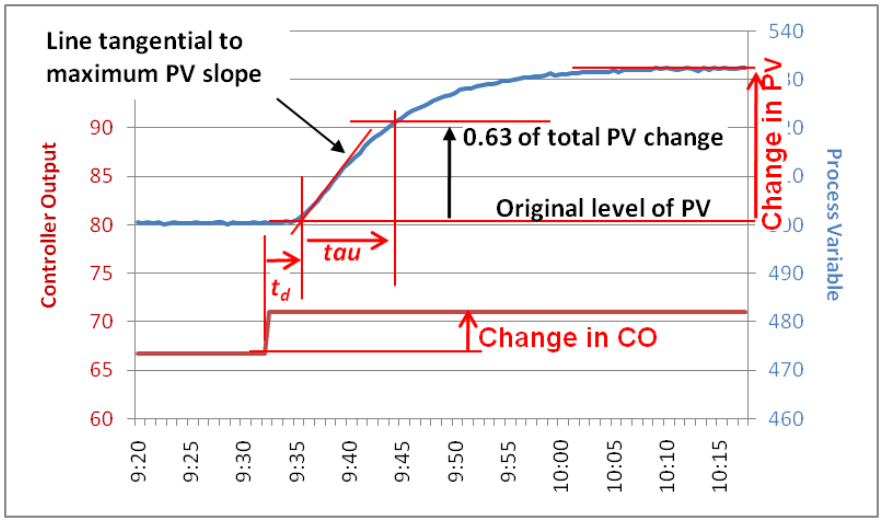

Figure 2.1: Illustration of the step test [8].

Often, the steps mentioned above are repeated 2-3 times to get good average values for the process parameters. Then, the table below gives the rules for tuning the different controllers. [9]

	Controller Gain	Integral time, t_i	Derivative time, t_d
P Controller	$K_c = \frac{1.03}{K}\left(\frac{\tau}{\theta} + 0.34\right)$		
PI Controller	$K_c = \frac{0.9}{K}\left(\frac{\tau}{\theta} + 0.092\right)$	$t_i = 3.33\theta\frac{\tau+0.092\theta}{\tau+2.22\theta}$	
PD Controller	$K_c = \frac{1.24}{K}\left(\frac{\tau}{\theta} + 0.129\right)$		$t_d = 0.27\theta\frac{\tau-0.324\theta}{\tau+0.129\theta}$
PID Controller	$K_c = \frac{1.35}{K}\left(\frac{\tau}{\theta} + 0.185\right)$	$t_i = 2.5\theta\frac{\tau+0.0185\theta}{\tau+0.611\theta}$	$t_d = 0.37\theta\frac{\tau}{\tau+0.185\theta}$

Table 2.2: Cohen-Coon tuning rules.

These tuning rules are designed to give a fast response, aiming for what is called quarter amplitude damping (QAD), which eliminates error between setpoint and process variable very fast, overshooting the setpoint and oscillating a few times before stabilizing. The error from setpoint then becomes smaller with a ratio of 4:1 for each overshoot cycle. Although this thinking gives very fast disturbance rejection, it could be a poor tuning choice, as it gives rise to several problems. It makes the loop oscillatory, and it causes overshoot every time it rejects disturbance. Also, these loops are not very stable nor very robust, and can result in closed loop instability if the process characteristics change, for example if the process gain K doubles.

For this reason, it can be necessary to further tune the controller after this initial tuning. A modified version of the Cohen-Coon method aiming to minimize the drawbacks mentioned above, is to divide the controller gain K_c by 2, which to a large extent reduces oscillations and overshoot and increases the loop's robustness. [10]

2.3 Least Squares

This section is an excerpt from the Relevant Theory section of my TTK 4551 - Engineering Cybernetics Specialization Project, "Recursive Least-Squares Estimator to Classify Drilled Formation for Autonomous Miniature Drilling Rig". It is added in this thesis for a more complete understanding of the estimator implemented in the control system.

The least squares principle chooses unknown parameters of a mathematical model to minimize the sum of squares of the differences in observed and computed values, multiplied by numbers that describe the degree of precision. A mathematical model, or *regression model* in the following form can be set up

$$y(i) = \phi_1(i)\theta_1 + \phi_2(i)\theta_2 + ... + \phi_n(i)\theta_n = \phi^T(i)\theta, \tag{2.12}$$

where $y(t)$ is an observed variable, $\phi_1...\phi_n$ are known functions, or *regressors*, and $\theta_1...\theta_n$ are parameters to be determined. The purpose of the least squares method is then to determine the parameters to minimize the cost function

$$V(\theta, t) = \frac{1}{2}\sum_{i=1}^{t}(y(i) - \phi^T(i)\theta)^2. \tag{2.13}$$

Defining

$$Y(t) = [y(1) \quad ... \quad y(t)]^T, \tag{2.14}$$

$$\Phi(t) = \begin{bmatrix} \phi^T(1) \\ \vdots \\ \phi^T(t) \end{bmatrix}, \tag{2.15}$$

$$P(t) = (\Phi^T(t)\Phi(t))^{-1}, \tag{2.16}$$

the least squares estimation is given by

$$\hat{\theta} = (\Phi^T\Phi)^{-1}\Phi^T Y = P(t)\left(\sum_{i=1}^{t}\phi(i)y(i)\right), \tag{2.17}$$

provided that $\Phi^T\Phi$ is a nonsingular matrix.

In online estimation, there are several considerations to be taken into account, such as computational complexity and time-varying parameters. These challenges can be met with recursive computation and forgetting, respectively. The recursive least squares equations with exponential forgetting, λ, are as follows:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)\left(y(t) - \phi^T(t)\hat{\theta}(t-1)\right) \tag{2.18a}$$

$$K(t) = P(t-1)\phi(t)\left(\lambda I + \phi^T(t)P(t-1)\phi(t)\right)^{-1} \tag{2.18b}$$

$$P(t) = \left(I - K(t)\phi^T(t)\right)P(t-1)/\lambda \tag{2.18c}$$

with $0 < \lambda \leq 1$,

where initial condition for P is obtained by choosing $t = t_0$ so that $\Phi^T(t_0)\Phi(t_0)$ is nonsingular, yielding

$$P(t_0) = \left(\Phi^T(t_0)\Phi(t_0)\right)^{-1} \tag{2.19a}$$

$$\hat{\theta}(t_0) = P(t_0)\Phi^T(t_0)Y(t_0). \tag{2.19b}$$

The recursive least squares equations with exponential forgetting minimize the following cost function:

$$V(\theta, t) = \frac{1}{2}\sum_{i=1}^{t}\lambda^{t-i}(y(i) - \phi^T(i)\theta)^2. \tag{2.20}$$

From this, it is clear to see that the recursive least squares with exponential forgetting is equivalent to least squares, solving the cost function in Equation (2.13) when $\lambda = 1$.

Time-varying parameters can be divided into two different classes. The first class consists of parameters changing continuously but slowly. The other class, which might be most relevant to this project, consists of the cases where parameters change abruptly but infrequently.

Assuming the first case, where parameters change continuously but slowly, exponential forgetting is a common approach for online estimation. Here, the most recent data is given unity weight, while data being n samples old are weighted λ^n [11]. The choice of λ here determines how fast the system forgets old data. Typically, the choice of λ is somewhere above 0.98, as values below this value renders the system too susceptible to noise since the estimation is based mostly on the last few samples. If the horizon of samples used in the estimation is too short, the zero-mean property of white noise is not captured by the estimator, which explains the increasing susceptibility to noise with lower forgetting factor.

For the second case, where parameters are assumed to change abruptly but infrequently, there exist other approaches. One approach can be to reset the matrix P to δI with regular intervals, where a recommended value for δ may be

$$\delta > 100\sigma^2. \tag{2.21}$$

Here, σ^2 is the variance of the parameters. [12]

Chapter 3

Implementation

3.1 Considerations

As the master theses of the team members are related to the Drillbotics competition, our work progress will be handed over to a new team to participate in Drillbotics 2019. Handover of work is a delicate matter that could be problematic even for experienced professionals, which is why we wanted to take extra care to ensure a smooth transition. To strive for this, we came up with some points we wanted to prioritize:

- Having clean, understandable code with descriptive variable names and comments.
- Making a user-friendly GUI.
- Ensuring that data is well-documented.

Anyone with experience in programming knows the importance of descriptive variable names and comments in the code. Even understanding ones own code written some time back can prove difficult if there are no descriptions available. Since the scope of the Drillbotics control system is quite large, it is extremely important to keep this in mind. Having a user-friendly user interface is also important for the handover. Inheriting a system where the GUI consists of many controls can be overwhelming if there is no apparent guide telling which controls to operate. We let us inspire of common practice in commercial software, where controls are greyed out and disabled if they should not be operated, and aimed to implement this kind of functionality in our GUI as well. When working in projects with several members involved, it is important that files are saved in a consistent manner to avoid not finding data, duplicates, old versions, etc. We aspired to eliminate this problem by programmatically saving all drilling data after each test run. This way,

project members would not be free to save files as they please, but prompted after each run if they wanted to save drilling data, and if so, the data would be saved with a pre-specified name structure to a pre-specified folder.

3.2 LabVIEW

Laboratory Virtual Instrument Engineering Workbench, or LabVIEW for short, is a development environment for a graphical programming language from National Instruments commonly used for supervisory control and data acquisition (SCADA) purposes. The programming language manifests as different blocks which represent functions. These blocks are wired together into a block diagram, and the ordering of the blocks relative to each other determine the execution order of the code, since the programming paradigm is based on data availability.

The LabVIEW software is comparable to Simulink, which was used by NTNU's Drillbotics team last year, although allows lower level programming. Also, Simulink is primarily intended for simulation purposes, as the name suggests, while LabVIEW is commonly used for real-time operations. A new LabVIEW engineer who has experience with script programming will quickly recognize structures such as while loops, for loops, case structure, local and global variables. The fact that LabVIEW is programming at a lower level means that the engineer is free to manipulate the design of the program at a lower level, including:

- greying out and disabling irrelevant controls in the GUI.
- customizing graph indicator and control appearance.
- setting the iteration speed of the program.
- implementing a state machine.
- forcing the order of execution using flat structures.

Often, these manipulations can be carried out using the same logic as in script programming since the presentation of data structures in LabVIEW are similar to those used in scripting. The changes can be used both to create an intuitive and user-friendly GUI, as well as saving computer memory by specifying run speed of the program.

Also, programs can be called within the main program, which bears resemblance to functions in scripting. The main program is called a VI (Virtual Instrument), and sub-programs called within this VI are reasonably enough called subVIs. This encourages a modular design of the program, which makes streamlining and debugging much simpler. As mentioned earlier, it has been of great importance to the team to have a user-friendly GUI and

a modular, well-documented, and easy to understand code when handing the project over to next year's team, which strongly advocated the use of LabVIEW for our system.

3.3 Overview of Setup

The miniature drilling rig has a hollow steel beam frame, weighing approximately 100 kg. It is 70 cm wide, 285 cm tall when erect, and 235 cm long when folded down. To imitate the functionalities of a full-scale drilling rig, it consists of three actuators, a **top drive motor**, a **hoisting motor** and a **pump motor**. The top drive motor provides rotation to the drillstring and the drillbit, causing scraping and crushing of drilled formation. The hoisting motor pushes downwards, resulting in weight on bit and torque. This is necessary on a miniature drilling rig, as the weight of the drillstring and bit alone are insufficient, in contrast to a full-scale rig. The hoisting motor is connected to a ballscrew, which turns the rotational velocity of the motor to a linear, translational movement up and down. The pump motor circulates fluid by injecting it into a swivel which transports it through the drillpipe, out of a nozzle in the bit, and up the annulus. In the drilling industry, the choice of drilling fluid is important to manage downhole pressure, among other things. For the miniature drilling rig described in this thesis, the drilling fluid used is simply water, which serves as a transportation agent of drilling cuttings, removing them from the well, and up towards the surface. This is important, as effective hole cleaning is necessary for effective drilling. The drilling fluid from the pump also provides internal pressure in the drillstring, stiffening it. Lastly, is cools down the drillbit, which otherwise would reach high temperatures due to the friction between the bit and the drilled formation.

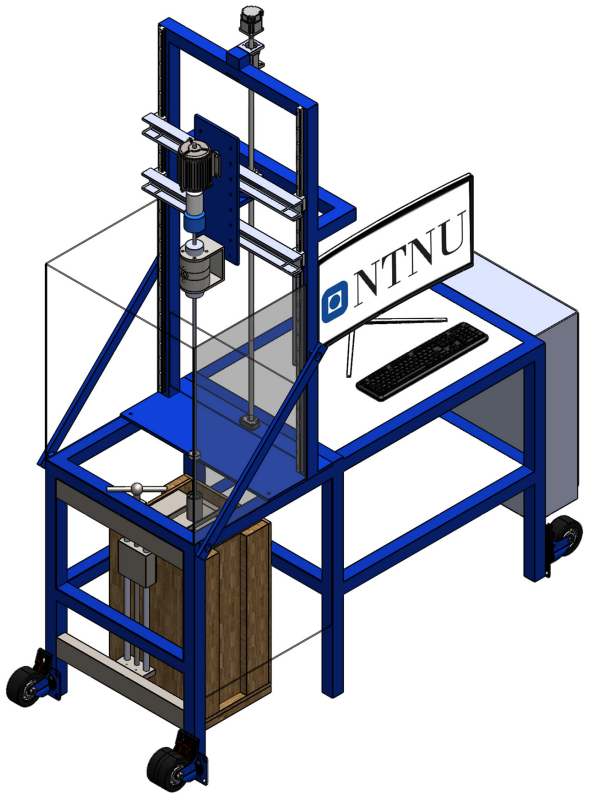

Figure 3.1: Overview of rig setup [13].

The drillstring is supported at four points during the drilling operation, and as the operation progresses, the lengths of the unsupported segments will vary. As can be seen in Figure 3.1, the top of the drillstring is connected to the top drive motor, and runs down through the rotary kelly bushing (RKB) which is a teflon bearing located on the drill floor. Further down, a riser is located, which is fastened to both the rig and the formation to ensure alignment throughout the operation. Inside the riser lies another teflon bearing to support the drillstring and mitigate lateral vibrations. At the end of the drillstring lies the bottomhole assembly (BHA), housing the downhole sensor card, and then the drillbit. The last support point is given by the formation when weight is applied to the bit.

Figure 3.2 shows the frequency converters, or drives, for the three actuators. The rightmost is the drive for the top drive motor, which includes a digital

display to navigate settings, and an internal PID controller for rotational velocity. The middle drive is for the hoisting motor, which is set up using a software called Lenze Engineering. This software was also used to scale measurements from the drive. The drive contains an internal PID controller for rotational velocity of the hoisting motor which acts upon the ballscrew, converting it to a translational movement up and down. The leftmost drive controls the pump motor, and simply includes a knob to increase and decrease the rotational velocity along with a seven-segment display to view the RPM. This drive also uses an internal PID controller.

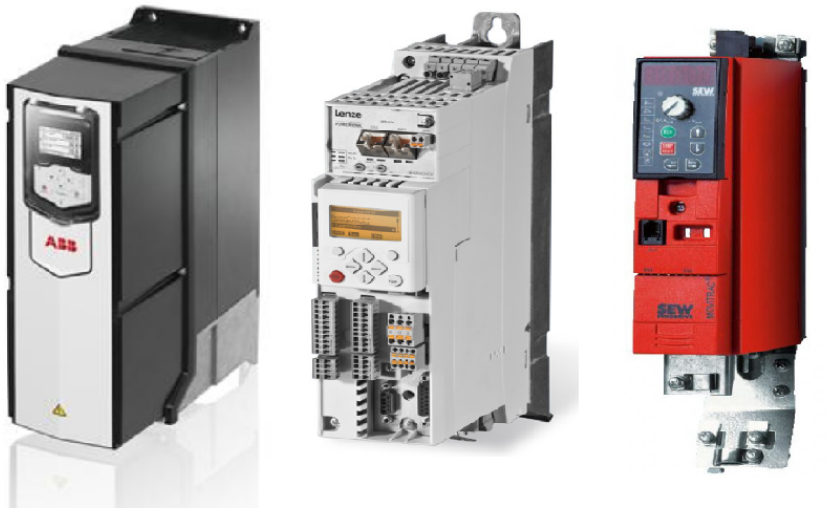

Figure 3.2: Drives for top drive, hoisting motor and pump motor.

A cDAQ-9174 chassis with NI-9207 (voltage and current input), NI-9263 (voltage output) and NI-9375 (digital I/O) modules were acquired, and are illustrated in Figure 3.3. These modules were meant to acquire load cell and pressure measurements, along with reading and writing to the pump motor. However, due to time limitations, we were unable to implement this solution, and the load cell and pressure measurements were acquired using the USB-6009 DAQ instead, while the pump motor was manually operated.

Figure 3.3: cDAQ-9174 chassis with NI-9207, NI-9263 and NI-9375 modules. The rightmost is the NI USB-6009 Multifunction I/O device used in the final setup.

Figure 3.4 gives an overview of the communication infrastructure for the system. It can be seen that the pump motor and ClampOn sensor are not connected to the LabVIEW control system, but rather run in parallel. The data acquisition for the load cell and pressure gauge is done by the USB-6009 Multifunction I/O, and data acquisition for the utilized inputs on the device is set up in LabVIEW. The downhole sensor card is wired directly by USB to the computer, and data acquisition is performed using VISA in LabVIEW. The top drive and hoisting motors are connected to the drives, which communicate using Ethernet cable as physical layer, and Modbus as communication protocol. Note that the hoisting motor drive is connected via a Modbus TCP - Ethernet/IP gateway, as the drive itself is not directly compatible with Modbus communication.

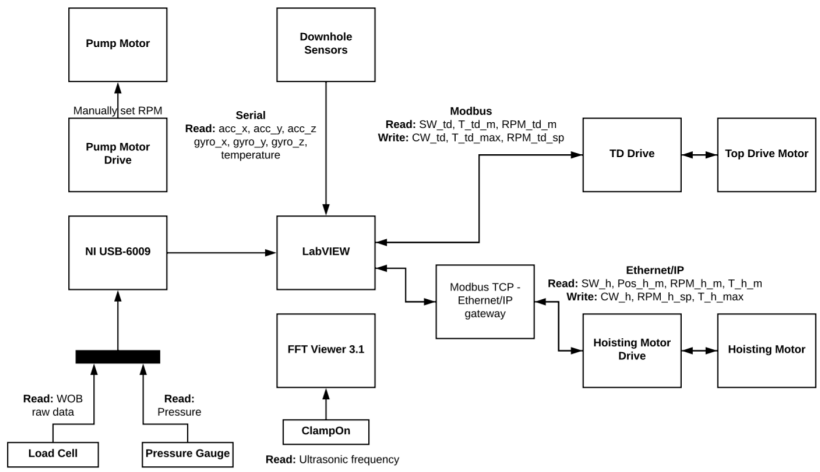

Figure 3.4: Communication infrastructure overview.

3.4 Communication

3.4.1 Top Drive Motor

The top drive motor is an *ABB 3GAA 091520-ASJ*, which is connected to LabVIEW via the *ABB ACS 880-01-05A6-3* drive. The drive is connected via Ethernet cable, communicating using the Modbus TCP/IP protocol, which establishes a client/server hierarchy between the LabVIEW interface and the drive, respectively. This hierarchy means that the drive (server) will not perform any actions until it receives a request from the LabVIEW interface (client). The communication allows for reading a wide variety of measurements from the holding registers of the drive, but for the purposes of this project, the measurements of interest are the RPM of the top drive motor and the torque. Also, a binary status word can be read from the coils of the drive, indicating whether or not the top drive motor is ready for operation or if some error has occurred. The set up communication also permits writing to the coils of the drive. To enable control of the top drive motor, a binary control word must first be written to coils. As the drive contains an integrated PID controller for the RPM of the top drive motor, communication to write setpoints to this holding register was also set up.

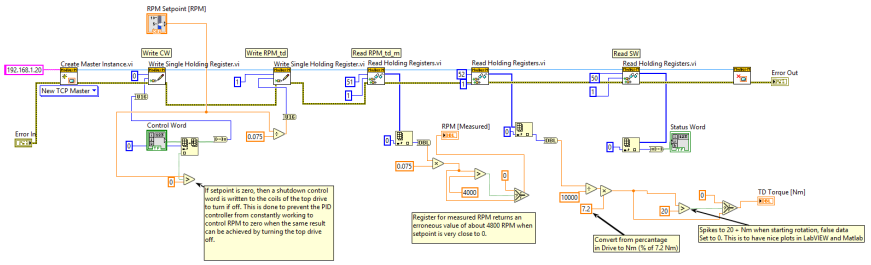

Figure 3.5: Top drive communication subVI block diagram.

The leftmost block in Figure 3.5 initiates the code by establishing a master instance (client) to control the top drive motor located at the given IP address. The next two blocks writes the control word and the top drive RPM setpoint. To avoid having the internal PID controller of the motor constantly working when RPM setpoint was set to zero, some additional logic was added, saying that if RPM setpoint is equal zero, the first bit of the control word was set to logic low, which effectively shuts down the motor. The successive three blocks read top drive RPM, torque and the status word, before the master instance is closed.

3.4.2 Hoisting Motor

The hoisting motor is a *Lenze GST 03-2M VBR 063C42*, which is connected via the *Lenze 8400 Topline C* drive. An Ethernet/IP module for the drive, along with a Modbus adapter was acquired to set up communication with LabVIEW. This made setting up communication quite similar to that of the top drive motor, and the structure of the communication subVI could to some extent be copy/pasted. Changing the IP address, input/holding register addresses, and scaling measurements were however required. Also here, the first bit of the control word was set to logic low when the weight on bit or speed setpoints were zero, effectively disabling the motor to avoid that the internal PID controller in the drive constantly worked when not necessary.

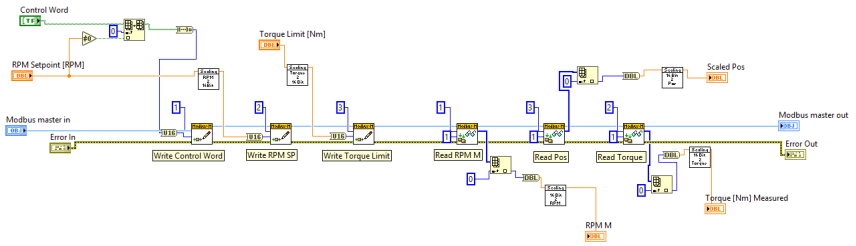

Figure 3.6: Hoisting motor communication subVI block diagram.

3.4.3 Pump Motor

The pump motor is a *Hawk HC980A*, which is connected via the *SEW Movitrac MC07B0040-5A3-4-00* drive. Though it was initially planned to implement the pump motor in the control system for the rig, there was not enough time due to several other issues regarding setup of communication throughout the semester. This forced prioritizing other tasks, leading to manual operation of the pump motor when drilling. This did not affect the drilling operation significantly, as the pump motor was planned to run at a constant RPM, and the only difference was turning on the drive in the electrical cabinet at the beginning of the drilling operation rather than using LabVIEW.

3.4.4 Load Cell

The load cell is a *TC4AMP* force transducer, initially planned to be connected via the NI-9207 C Series Voltage and Current Input Module on the cDAQ-9174 chassis, was finally connected via the USB-6009 Multifunction I/O. The load cell has a range of +/- 5 kN, converted from a +/- 10 V differential

input voltage. Wanting to display weight on bit in kg, the linear interpolation equation becomes:

$$WOB = -509.7 \text{ kg} + \frac{V_{in} + 10 \text{ v}}{20 \text{ v}} 1019.4 \text{ kg} \tag{3.1}$$

3.4.5 Pressure Gauge

The pressure gauge used is an *Aplisens PCE-28* pressure transmitter, also initially planned to be connected via the NI-9207 C Series Voltage and Current Input Module on the cDAQ-9174 chassis, was finally connected via the USB-6009 Multifuncition I/O. It is set up to convert a 4-20 mA current signal to 0-100 bar by interpolating within its pressure range:

$$p = \frac{i_{in} - 4 \text{ mA}}{20 \text{ mA} - 4 \text{ mA}} 100 \text{ bar} \tag{3.2}$$

3.4.6 ClampOn SandQ

The ClampOn SandQ is a high-sensitive, non-intrusive sensor commonly used in the oil and gas industry for detection of sand in production through the ultrasonic signal that is generated by the sand particles flowing inside the pipe. It contains onboard digital signal processing, and generates an FFT plot as output. [14]

This sensor was not included in the LabVIEW program, but run in parallel on separate software. It was found interesting that the sensor indicated events by changes in the frequency spectrum that could also be seen on other measurements like weight on bit and torque. Although this sensor was not incorporated in the autonomous program, it should be considered that it may serve as an extra set of eyes for example in detecting change in formation, and further work on this should be performed. Figure A.1 in Appendix A shows the SandQ and the weight on bit throughout an autonomous run.

3.4.7 Downhole Sensor Card

The downhole sensor card included an *MPU-6050* 6-axis accelerometer + gyroscope, along with a temperature sensor. The card itself was soldered and programmed by Steffen Wærnes Moen to output raw measurements, and was connected to the computer by USB. The communication was set up using VISA in LabVIEW.

3.5 Signal Conditioning

3.5.1 Sampling Frequency

In design of the discrete-time process, determining the necessary sampling frequency was first on the agenda. Last year's communication with the top drive and hoisting motor ran through a PLC and an OPC layer, which was limited to 10 Hz. It was early established that this limitation could result in problems due to more time delay, which is why this year's communication towards said motors were changed to Modbus to be able to increase the sampling frequency of the system. Several scenarios require short reaction time, like tagging of top formation layer, a change in drilled formation, and inhomogeneous formations resulting in torque spikes. Therefore, to avoid severe drilling dysfunctions such as buckling and twist-off, the top drive and hoisting motor should be sampled at short intervals. Also, the load cell must be sampled at the same frequency as the hoisting motor to precisely control the weight on bit. Though it is not planned to use the accelerometer measurements in any control capacity, they will be important in the testing phase to monitor downhole vibration dynamics. When choosing appropriate drilling parameters, the rate of penetration will obviously be among the most important criteria, but monitoring vibrations will be equally important. Observing amplitudes at different vibration frequencies will help determine the eigenfrequencies of the system in different formation types, so these can be avoided. To capture higher-frequency vibrational dynamics, it will be necessary to sample these measurements at a higher frequency than 10 Hz. It was early suggested to read the following measurements at 100 Hz:

- weight on bit from load cell,
- torque on drillstring from top drive,
- rotational velocity from top drive,
- rotational velocity from hoisting motor,
- position from hoisting motor,
- and torque from hoisting motor.

Also, setpoints to the internal PID controllers of the top drive and hoisting motors are written at 100 Hz. This just leaves the pump motor read/write, and the pressure gauge. The pressure control is expected to be simpler, as the conditions within the drillstring are more or less constant as long as no drilling dysfunctions occur. For this reason, it was initially decided to run the pump motor and pressure gauge in a parallel loop in the LabVIEW program at 10 Hz, although it was later found that the structure of the program and the logging of this data would be simpler if run in the main while loop of the program. We ended up doing this, and sampling also this sensor at 100 Hz.

The downhole sensor card was implemented in a parallel loop in the LabVIEW program running at 500 Hz, which was necessary to observe high frequency vibrations. The ClampOn SadQ sensor was, as previously mentioned, not incorporated in LabVIEW, but rather run in parallel on a separate software called FFT Viewer 3.1. This software allows selecting several ranges of sampling frequencies, from a couple of kHz to some MHz.

3.5.2 Digital Filtering

As expected, measurements from sensors were subject to high-frequent noise. To attenuate the noise component in the signals, digital filters were implemented. The LabVIEW Signal Processing library contains a variety of filters, which were initially implemented in the system. However, these built-in filters contained several settings and functionalities, complicating the process of finding the appropriate filters in this library. Also, some of these filters gave completely erroneous outputs, which indicated that we did not have a complete understanding of the wiring and settings of the built-in filter blocks. For debugging purposes and a more complete understanding, it was therefore decided to code filter blocks from scratch.

The first self-coded lowpass filter was a first-order Butterworth filter. This was seen to attenuate some noise, although not effectively enough due to the fact that its transition band of -20 dB/dec is not steep enough. Wanting to steepen the transition band to -60 dB/dec, a third-order Butterworth filter was implemented. The continuous-time transfer function was found using the normalized Butterworth polynomial introduced in Chapter 2.1.1, and using Equation (2.3) to get:

$$H(s) = \frac{1}{\tau^3 s^3 + 2\tau^2 s^2 + 2\tau s + 1}. \tag{3.3}$$

Further, this transfer function was discretized using the bilinear transform, given in Equation (2.7) to get:

$$H_d(z) = \frac{1}{\tau^3\left(\frac{2}{T}\frac{z-1}{z+1}\right)^3 + 2\tau^2\left(\frac{2}{T}\frac{z-1}{z+1}\right)^2 + 2\tau\left(\frac{2}{T}\frac{z-1}{z+1}\right) + 1}, \tag{3.4}$$

which was rearranged to the standard form for the recursive filter given in Equation (2.8). Furthermore, using the difference equation defining the relationship between input and output of the filter given in Equation (2.9), this yields:

$$y[n] = -a_1 y[n-1] - a_2 y[n-2] - a_3 y[n-3] + b_0 x[n] + b_1 x[n-1] + b_2 x[n-2] + b_3 x[n-3], \tag{3.5}$$

where:

$$b_0 = \left(8\frac{\tau^3}{T^3} + 8\frac{\tau^2}{T^2} + 4\frac{\tau}{T} + 1\right)^{-1} \tag{3.6a}$$

$$b_1 = 3b_0 \tag{3.6b}$$

$$b_2 = b_1 \tag{3.6c}$$

$$b_3 = b_0 \tag{3.6d}$$

$$a_1 = \left(-24\frac{\tau^3}{T^3} - 8\frac{\tau^2}{T^2} + 4\frac{\tau}{T} + 3\right)b_0 \tag{3.6e}$$

$$a_2 = \left(24\frac{\tau^3}{T^3} - 8\frac{\tau^2}{T^2} - 4\frac{\tau}{T} + 3\right)b_0 \tag{3.6f}$$

$$a_3 = \left(-8\frac{\tau^3}{T^3} + 8\frac{\tau^2}{T^2} - 4\frac{\tau}{T} + 1\right)b_0. \tag{3.6g}$$

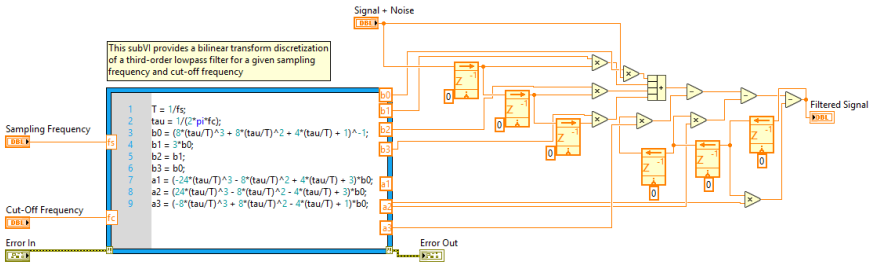

Figure 3.7: Third-order Butterworth lowpass filter subVI.

3.6 PID Control Modes

3.6.1 Purpose

The PID control modes are separate from the autonomous state machine mode. The purpose of the different modes are to compare control schemes to find which parameters can be best controlled. This is important both for avoiding drilling dysfunctions such as twist-off and buckling, as well as finding trends in input parameters which result in best possible rate of penetration. The PID modes will be used for gathering data and gaining drilling experience before choosing the appropriate control scheme in the autonomous mode.

3.6.2 RPM/WOB Mode

The RPM/WOB mode involves controlling the rotational velocity of the drillstring using the internal RPM PID controller in the top drive motor, and controlling weight on bit using the internal RPM PID controller in the hoisting motor.

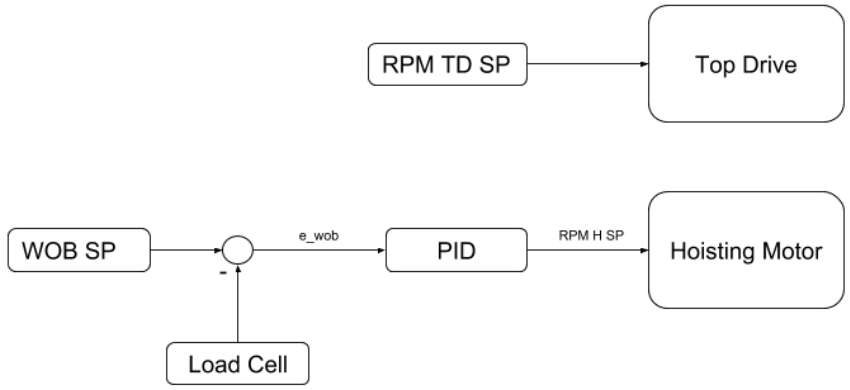

Figure 3.8: Overview of RPM/WOB control scheme.

As Figure 3.8 shows, the setpoint for top drive rotation is fed directly to the internal RPM PID of the top drive. For controlling the weight on bit, the error between setpoint and load cell measurement is fed into a PID controller. The output of this PID controller is the setpoint for hoisting motor rotational velocity, which is fed to the internal RPM PID controller of the hoisting motor. The benefit of this control scheme is that both top drive RPM and weight on bit are quite easily controlled since the internal RPM PID controller of the top drive is well-tuned, and the weight on bit dynamics are quite slow. The logged drilling data from NTNU's Drillbotics team last year shows that they managed to control these variables quite well. However, the torque on the drillstring is not controlled in this scheme, which can be problematic. Twist-off of the drillstring is a drilling dysfunction which occurs at too high torque values, and from early testing, it was found that these critical torque values occurred before reaching weight on bit near the buckling limit, making torque values the bottleneck for the setup.

3.6.3 RPM/Torque Mode

The RPM/Torque mode controls the rotational velocity of the drillstring in the same manner as the RPM/WOB mode, using the internal RPM PID

controller in the top drive motor. In addition, it takes the error between torque measurements from the top drive motor and a setpoint, as input to the internal RPM PID controller in the hoisting motor.

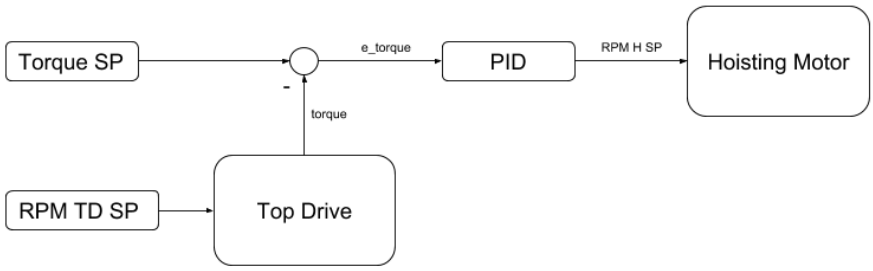

Figure 3.9: Overview of RPM/Torque control scheme.

From Figure 3.9 it can be seen that the top drive rotational velocity is controlled in the same manner as the RPM/WOB mode, by feeding the setpoint directly to the internal RPM PID controller of the top drive. The top drive contains a torque sensor, and a PID controller takes the difference between the torque setpoint and this measurement as input, and gives the RPM setpoint for the internal PID controller in the hoisting motor as output. In this control scheme, the weight on bit is not controlled, although this is not believed to be a problem since the critical torque values have proven to occur before the critical weight on bit values, as mentioned previously. It has been experienced in previous testing that the torque dynamics are much faster than weight on bit dynamics, which could be an issue. Especially in inhomogeneous formation types, torque spikes have been known to occur. This presents a control issue, as these spikes happen instantaneously and can reach critical levels, while the control system may not be quick enough to handle them.

3.6.4 Torque/WOB Mode

The Torque/WOB mode controls the torque using the rotational velocity of the top drive, in contrast to the RPM/Torque mode, which uses the hoisting motor. It does so by taking the error between the torque measurements of the top drive motor and a setpoint as input to the internal RPM PID controller.

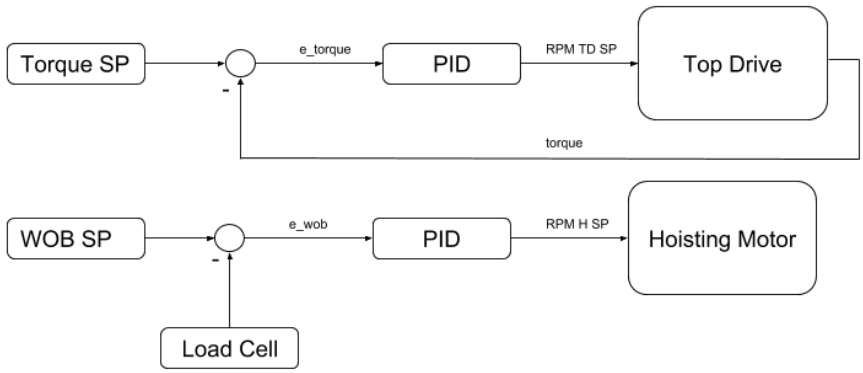

Figure 3.10: Overview of Torque/WOB control scheme.

Last year's team also experimented with torque control, although only considering the hoisting motor as actuator. Since the drillstring torque is a function of both weight on bit and rotational velocity, it was initially thought useful to implement both types of torque control to compare which control scheme best could handle the fast torque dynamics. One benefit of this configuration is the direct control of both variables responsible for the most common and catastrophic drilling dysfunctions, torque and weight on bit, causing twist-off and buckling, respectively. It would however be necessary to add some additional logic to the top drive RPM setpoint limits, since this variable is not directly controlled. Homogeneous and soft formations may not give rise to high torque values, and in these cases it will be necessary to define an upper limit for the top drive RPM to avoid integrator wind-up if the system is unable to reach the torque setpoint. Also, in the case of inhomogeneous formations causing high torque values, some lower limit for top drive RPM should be defined to avoid RPM values approaching zero. However, this control scheme was not tested after finding that the top drive frequency converter ramps between setpoints, which results in slow, yet robust control, as well as less tear on the motor. With this configuration, the top drive would be too slow to accurately control torque. Also, since this control scheme requires more logic than the other modes, it was concluded that it would not be fruitful to further pursue the idea.

3.6.5 Speed Mode

The Speed mode controls the RPM of the hoisting motor by feeding a setpoint to its internal PID controller, while the top drive is not controlled at all. This mode will be used to set up the rig for new drilling experiments, but not for the drilling itself. The thought is to use this mode to hoist the

drillbit downwards until almost tagging the top formation layer, and then switching to the control mode that will be used in the drilling test. The drive for implementing this mode is mostly convenience, so one can quickly maneuver the rig using LabVIEW to set up for new experiments, rather than switching to other software in the set up, and then back to LabVIEW for the experiment itself.

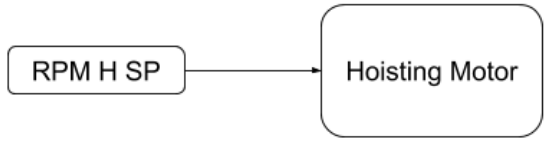

Figure 3.11: Overview of Speed mode control scheme.

3.7 State Machine

3.7.1 Purpose

While the various PID modes and additional programs are used for testing purposes and data gathering, the state machine is what will be used on the competition day. As the Drillbotics competition revolves around performing a completely autonomous drilling operation from start to finish, it makes sense to divide the operation into different stages. The purpose of the state machine is to define these stages, the actions performed within them, and the logics leading to the transition between the stages. The way a state machine is implemented in LabVIEW is by setting up a case structure in a while loop, where each case represents a state. The state contains code defining its scope, along with transition criteria implemented using selectors, which choose an enumerated constant, or enum, based on some boolean logic. This enum is passed through a shift register to the next iteration, either leading

to a change in state, or maintaining the same state, depending on the logic of the selector.

3.7.2 Initialization

The Initialization phase is the first phase of the state machine, and involves lowering the drillbit until tagging the top formation layer. In this phase, there is no rotation of the drillstring, meaning that the only actuator used in the control system is the hoisting motor. Note that the pump motor is run in parallel also in this phase. In this state, the hoisting motor is run in speed mode until reaching a weight on bit threshold, indicating that the bit has tagged formation, and then the state machine propagates to the Hoist Up phase, which is explained in the next section.

Through experimentation it was found that a hoisting motor RPM corresponding to 3 cm/min rate of penetration (downward drilling velocity) along with a 5 kg weight on bit threshold was an appropriate combination. The weight on bit threshold is needed due to several sources of friction on the rig. The main friction sources are between the ballscrew and load cell, the rotary kelly bushing, and the downhole bearing lying in the riser. All this friction causes the load cell measurements to vary around 0 kg with an amplitude of approximately 1.5 kg, however the weight on bit threshold was set a bit higher to avoid that the state machine logic would erroneously detect tagging formation. Since this phase involves no drillstring rotation, it enters a state of compression instantaneously when tagging formation, making the weight on bit increase very rapidly, so downward speed of the drillbit cannot be too high, as the control system would not be able to detect tagging formation before reaching critical weight on bit values leading to buckling. At the same time, Drillbotics is a competition, advocating a fast overall drilling process. Testing with different hoisting motor RPMs, it was found that 5 cm/min was the best value, as it is fast enought, and at the same time, the system is capable of stopping in time to avoid buckling. Several runs showed that the system would typically be able to stop the hoisting when reaching a weight on bit between 5-7 kg.

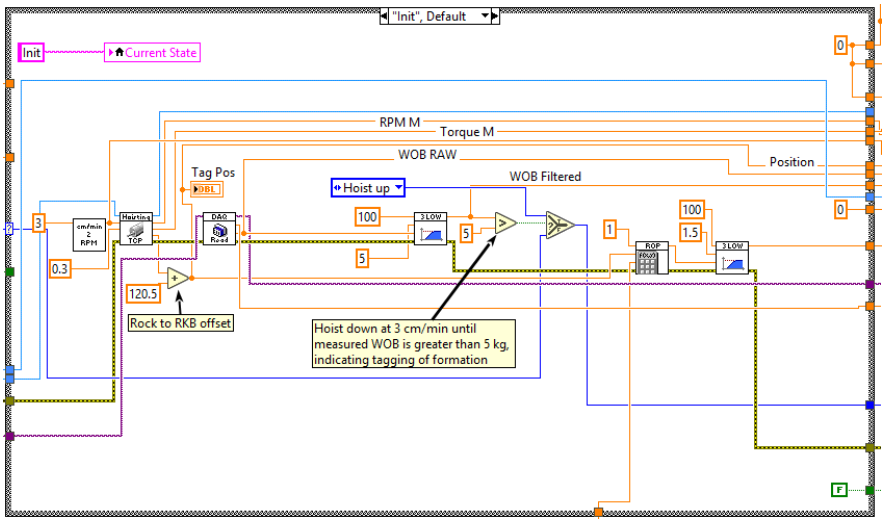

Figure 3.12: Initialization phase structure.

Figure 3.12 shows the block diagram within the Identification phase. The tag position is stored within this case and used later for termination criterion when reaching a target depth of tag position plus a pre-defined drilling depth.

3.7.3 Hoist Up

The Hoist Up phase is entered from the Initialization phase, and from this phase, the state machine will always propagate to the Start Rotation phase. This phase is necessary, as starting drillstring rotation while the bit is in contact with the formation often leads to bit whirl. Bit whirl is a phenomenon where the bit "walks" around in the well, which expands the well diameter and generally leads to poor borehole quality.

The phase serves as a necessary transitional state to safely start rotation in the next phase before commencing drilling. Also here, the only actuator controlled within LabVIEW is the hoisting motor. Here, the hoisting motor is run in speed mode like in the Initialization phase. The bit is hoisted up with a hoisting motor RPM corresponding to -1 cm/min for 4 seconds before hoisting down for 2 seconds and coming to a full stop. The 2 seconds of hoisting down is due to friction in the ballscrew leading to an offset in the weight on bit measurements when hoisting up. As the drilling process is expected to primarily run downwards, the load cell has been calibrated at zero weight on bit when in standstill and moving down in air. By hoisting 2 seconds down, the weight on bit measurements stabilize around zero, and the initial measurements in the next phase will be correct.

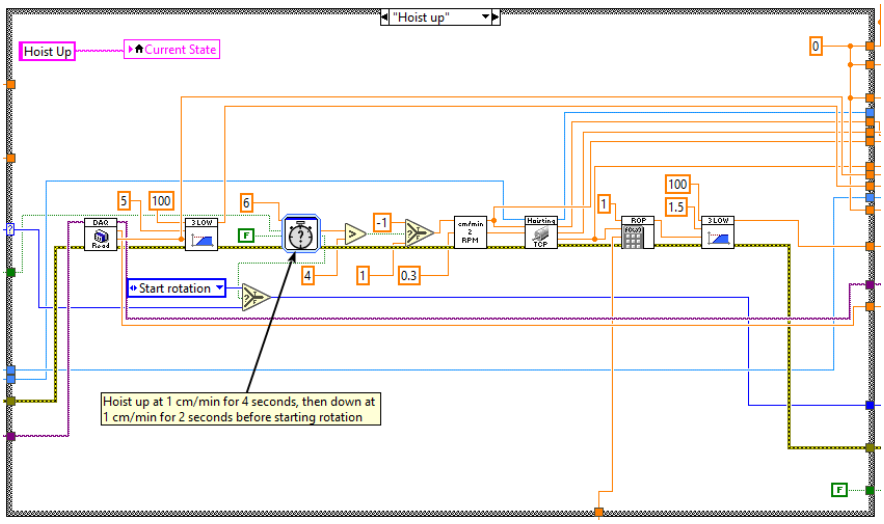

Figure 3.13: Hoist Up phase structure.

The Hoist Up phase is shown in Figure 3.13. This phase simply utilizes an internal timer to check whether the time elapsed within this state has reached 4 seconds, at which point it goes from hoisting up to hoisting down. After the successive 2 seconds, the phase is concluded. This timer is reset at first iteration of the phase.

As mentioned, this phase does not include top drive rotation. However, in the event of new layer detection, the state machine will transition to this state, retaining the setpoint for top drive rotation from the previous Drilling phase. This means that the first time the state machine enters this phase (after tagging top formation), there will be no top drive rotation. In every consecutive Hoist Up phase, however, there will be rotation although not programmed in this phase. This setup was used rather than stopping rotation since the top drive motor will reach its setpoint faster, leading to shorter down-time in operations.

3.7.4 Start Rotation

As the phase name suggests, this phase starts the rotation of the drillstring, getting ready to commence drilling. The setpoint for the drillstring rotation is 700 RPM, as this is the starting rotation used in the Identification phase. This phase terminates when the measured drillstring rotation has reached 690 RPM, and the state machine will move on to the Identification phase.

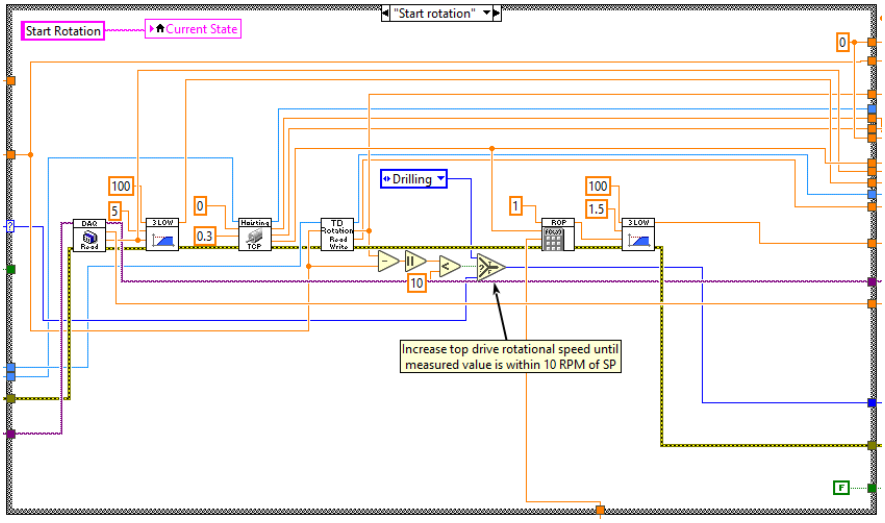

Figure 3.14: Start Rotation phase structure.

For termination of this phase, a selector checks whether the top drive rotational velocity is within 10 RPM of the setpoint using an absolute value block. This is to ensure that the phase concludes correctly in the cases where the RPM setpoint from the previous Drilling Phase is higher or lower than the setpoint used in the Identification phase. The block diagram is shown in Figure 3.14.

3.7.5 Identification

The Identification phase is based on a recursive least-squares estimator, and attempts to identify the drilled formation to choose the optimal drilling parameters based on previous testing. It does this by trying to relate the drilling parameters' and the formation's effect on rate of penetration through a linearized regression model which is given below:

$$ROP = \theta_1 + \theta_2(\omega - \omega_N) + \theta_3(WOB - WOB_N), \tag{3.7}$$

where

$$ROP : \text{measured rate of penetration,} \tag{3.8a}$$
$$\omega : \text{measured drillstring rotational velocity,} \tag{3.8b}$$
$$\omega_N : \text{normalized drillstring rotational velocity,} \tag{3.8c}$$
$$WOB : \text{measured weight on bit,} \tag{3.8d}$$
$$WOB_N : \text{normalized weight on bit,} \tag{3.8e}$$

and

$$\theta_1 : \text{effect of formation strength,} \tag{3.9a}$$
$$\theta_2 : \text{effect of rotary speed,} \tag{3.9b}$$
$$\theta_3 : \text{effect of weight on bit.} \tag{3.9c}$$

This regression model is a simplified version of that presented by Bourgoyne, et al [15], and the simplifications made are discussed in more detail in my TTK 4551 - Engineering Cybernetics Specialization Project, "Recursive Least-Squares Estimator to Classify Drilled Formation for Autonomous Miniature Drilling Rig".

One significant change to the estimator setup from that of last semester is that the regressors (drillstring rotational velocity and weight on bit) are now normalized. This is a measure taken as it was found that the estimator was unable to accurately determine the effect of formation strength, θ_1, when using the model without normalized regressors. It was then learned that a common rule of thumb is that an estimator will struggle to accurately determine regression model parameters when the model contains less regressors than regression model parameters. From this is makes sense that the previous setup struggled especially with θ_1, as no regressor in the form of measurement is related to this parameter.

In this phase, the PID parameters tuned to shale is used, as these parameters were found experimentally to give the best overall tuning results for every tested formation. To ensure bumpless transfer, the PID controller is always reinitialized in the first iteration of the Drilling phase, effectively eliminating any potential integral action stored in the controller. As the name of the phenomenon suggests, this is important to avoid cases where previous controller actions cause massive overshoot. Another safety measure was limiting the output of the weight on bit PID controller to +/- 1000 hoisting motor RPM actuation. The reason for this is to avoid excessively aggressive controller outputs, and to avoid integral wind-up. It was found in the LabVIEW documentation of the built-in PID block that when the actuation reaches the

limit of the controller output, the controller is reinitialized, which removes the possibility for wind-up.

The Identification phase runs through the same sequence every time. It starts with a constant weight on bit equal to 10 kg, and a constant drillstring rotational velocity equal to 700 RPM. After 5 seconds into the phase, the weight on bit is ramped up to 30 kg over a time span of 15 seconds, followed by a ramp up of drillstring rotational velocity up to 1200 RPM, also over 15 seconds. These ramp functions were self-coded, and are illustrated in Appendix B, Figure B.1. By choosing normalization of $\omega_N = 700$ RPM and $WOB_N = 10$ kg, the second and third terms in the regression model will be approximately equal to zero during the first 5 seconds of the Identification phase, allowing the estimator to more confidently determine θ_1.

Figure 3.15 shows the recursive least squares estimation code, which was implemented using a Matlab script written in a subVI. In the first iteration of the Identification phase, the boolean input "reset" is set to TRUE by implementing a delay node in the state machine and comparing the current state to that of the previous iteration. In this case, the covariance matrix is initialized as $P = 0.5I$, where I is the identity matrix. The regression model parameters θ_n for $n = 1, 2, 3$ are initialized according to Equation (2.19b). In the successive iterations of this phase, the gain vector K, covariance matrix P and regression model parameters θ_n are updated recursively using Equations (2.18).

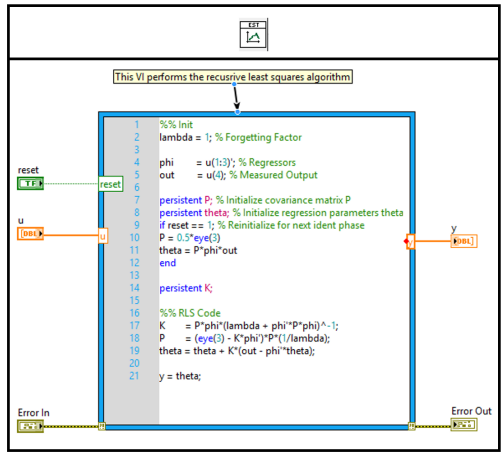

Figure 3.15: Matlab scripted recursive least-squares estimator subVI.

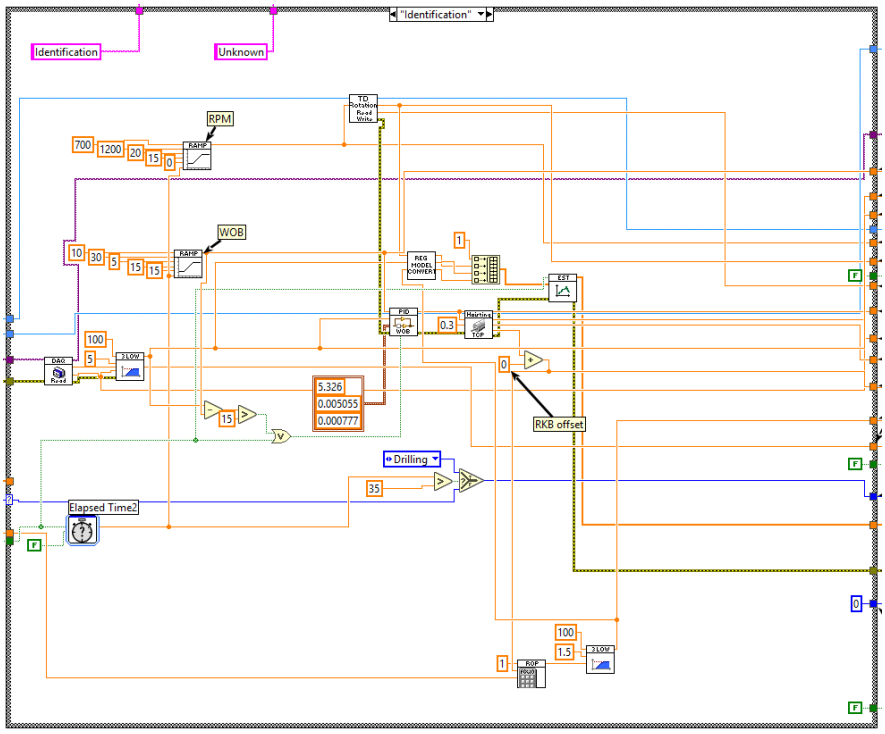

Figure 3.16: Identification phase structure.

Figure 3.16 shows the Identification phase. In addition to the actuator communication blocks and weight on bit PID controller, it includes self-coded ramp functions, the estimator itself, and an internal timer used to conclude the phase when the time elapsed in the phase has passed 35 seconds. Since the estimator uses such conservative drilling parameter setpoints, a pure weight on bit PID is used. However, in the Drilling phase, logic is included to switch between weight on bit control and torque control. This is further elaborated in the next section.

3.7.6 Drilling

The Drilling phase always follows the Identification phase. This phase contains the Library VI which stores the regression model parameters for each formation that has been tested beforehand, comparing the regression model parameters of the formation to be identified to those stored in the library. The regression model parameters are represented as points in $\mathbb{R}^3$, and the Library subVI simply calculates the Euclidean distance between the point representing the unknown formation and all existing points stored in

the library, which represent known formations. These distances are appended to an array, and the minimum element in the array is found. The reasoning behind this solution is that the closest fit of regression model parameters will belong to the known formation that is most similar to that formation being drilled. This means that if the autonomous system encounters a formation type not previously tested, it will simply allocate it to the closest fit in the library. For this to be a good solution, the library should contain formations of a variety of hardnesses to cover the spectrum of formations we expect to drill in.

The Library VI also contains setpoints for weight on bit and drillstring rotational velocity, for each of the stored formations. These setpoints are found from the formation response tests explained in section 4.3, and are those found to be best in terms of rate of penetration, vibrations, and torque values, as the aim is to optimize the drilling process while maintaining the integrity of the rig the safety of personnel in the vicinity.

The PID parameters used are obtained using the Cohen-Coon method for the formation types tested beforehand, and the experiments regarding the PID tuning are described in section 4.2. Initially, PID controllers for a variety of formation types were tuned, as it was found that a controller tuned for one formation type might give a completely different response in another formation with different properties. However, after a considerable amount of tests, it was found that the PID controller tuned in shale resulted in an overall good tuning for all tested formations. Another feature to avoid overshoot was to ramp up the weight on bit setpoint in the beginning of the Drilling phase. As in the Identification phase, the same measures are taken to ensure bumpless transfer and avoid integral wind-up.

Throughout the semester, we experienced stuck pipe multiple times, which led to the implementation some logic to switch between weight on bit control and torque control to handle this dysfunction. The logic is quite simple: if the torque exceeds a user-defined threshold, the controller switches to torque mode, resulting in backing up and allowing the top drive to rotate again. In these occurrences, we wanted slow control to avoid overshooting when pulling out. This was solved by scaling down the torque error and running experiments until finding that multiplying the torque error with 0.5 yielded satisfactory results. Implementing this in our design was important, as we actually experienced stuck pipe in the competition drill. This is explained in section 4.5.1.

Figure 3.17 shows the Library VI within the Drilling phase. The top of the script contains the library of regression model parameters for known formations. Further, all Euclidean distances are calculated and appended to an array. Lastly, some simple if-statements find the minimum of the array, and outputs the rock type, a formation index (for Matlab plotting), and setpoints for further drilling.

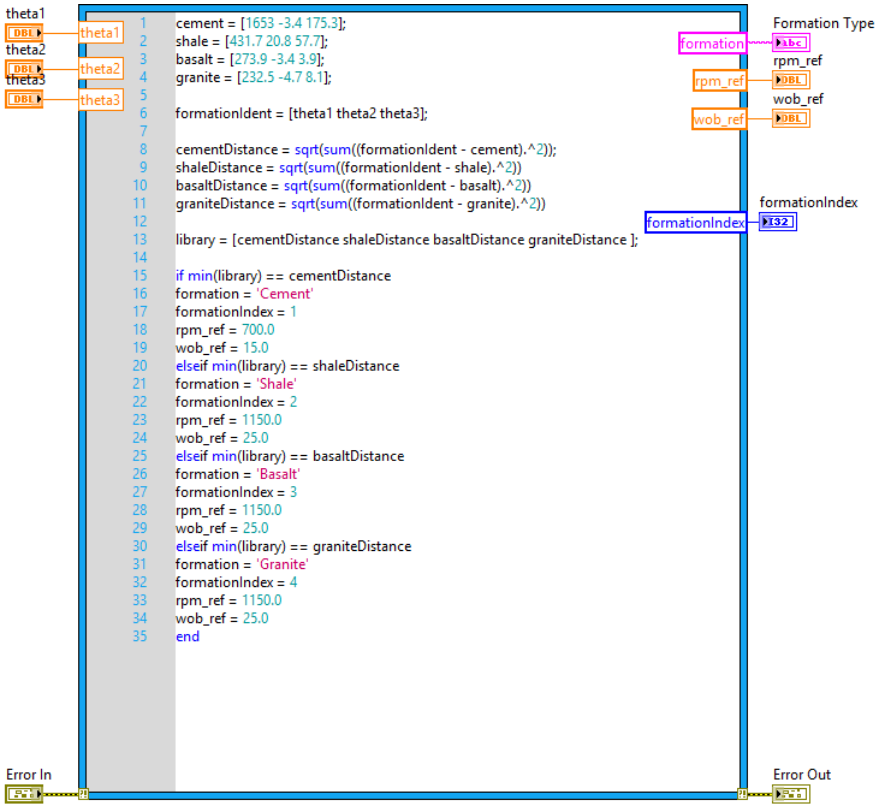

Figure 3.17: Library subVI containing pre-defined formations, along with setpoints.

To detect change in drilled formation, a simple change detection algorithm based on the rate of penetration was implemented in this phase. It was found that the rate of penetration within a formation closely resembled a Gaussian probability distribution. This led to the idea of comparing the mean rate of penetration for the last 2 seconds to the probability distribution of the previous 8 seconds. These values were not found instantly, but rather based on simulations performed on previous drilling experiments. To detect change in drilled formation, the algorithm says that if the mean is within a threshold of 2.75 standard deviations in the obtained distribution, the drilled formation is still the same. If not, a new layer is encountered. Also this threshold was found by running simulations on previous runs.

Figure 3.18 shows the block diagram for the Drilling phase. During the first 10 seconds, the change detection for formations is disabled while the PID controllers reach their setpoints to avoid erroneous detection. It can also be seen that the PID controller is subject to re-initialization of integral action

in the first iteration of the state, if measured weight on bit is more than 15 kg above setpoint, and when switching between weight on bit control and torque control.

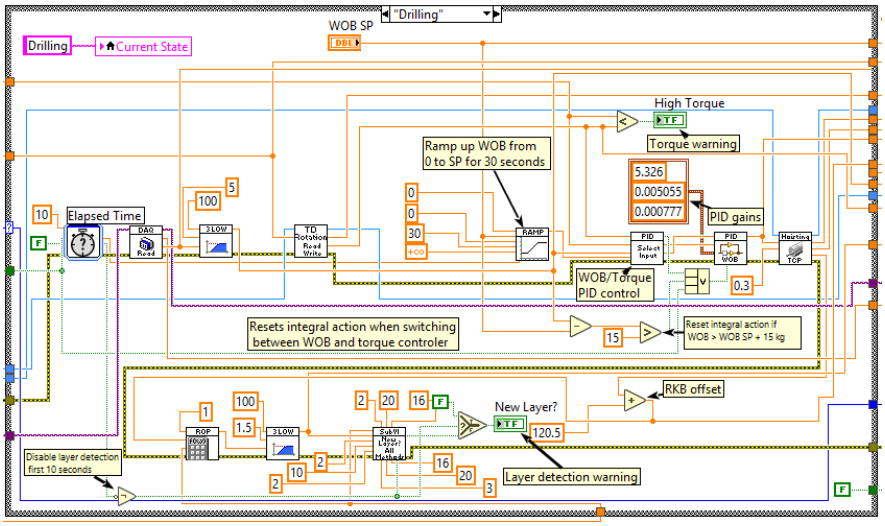

Figure 3.18: Drilling phase structure.

3.7.7 Trip Out

The Trip Out phase is the final state of the state machine, and is activated when reaching target depth. Here, the hoisting motor hoists the drillstring upwards at -10 cm/min, while maintaining 50 RPM on the top drive motor. The top drive rotation is a measure taken to avoid getting stuck when pulling out. The phase runs until the bit reaches the tag depth at the top of the formation, at which point the state is concluded. This is illustrated in Figure 3.19. Then, the script terminates, and the user is prompted to save the drilling data.

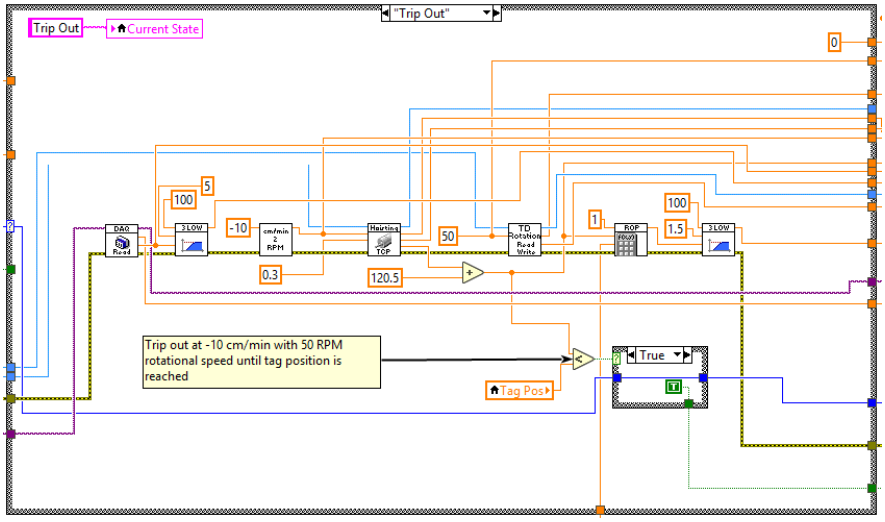

Figure 3.19: Trip Out phase structure.

3.8 Graphical User Interface

Initially, the manual PID control testing and the autonomous state machine were implemented in the same VI, and use property nodes to disable and enable relevant controls. However, this was found to be memory consuming, making the program lag, and execution time was inconsistent. Because of this, the programs were split up, and this section explains the GUI for the fully autonomous VI.

Figure 3.20 shows the GUI. At the top left corner, boolean LEDs indicate the status of the system:

- Top drive communication,
- Hoisting motor communication,
- DAQ communication,
- New layer indication,
- Twist off warning,
- High torque warning.

Moving to the right, the current state of the state machine, and the current formation drilled are indicated by string indicators. The indicator for the current drilled formation displays "Unknown" for every state except the

Drilling phase, where this string is chosen based on the logic in the Library VI.

Then, some numeric indicators display elapsed time, position [mmRKB] which is the position relative to the rotary kelly bushing, drilled depth from tag position, and downhole temperature from the downhole sensor. Numeric controls are located to the top right. Target depth can be entered before startup of the system, which defines the position at which the state machine will initiate the Trip Out phase. Weight on bit limit is a safety precaution which can be entered, so that if for any reason the system should become unstable and exceed the weight on bit limit, the program will terminate immediately. Finally, a torque limit can be entered which defines the torque where the PID controller switches to torque mode to back off. All the way to the right is a progress bar showing drilling progress towards the target depth as well.

Four waveform charts display important drilling data throughout the process. These are the weight on bit measured and setpoint, rate of penetration, torque and torque limit, and finally axial vibrations from the downhole accelerometer. At the bottom, gauges also display rate of penetration and weight on bit measured/setpoint. Also, pressure and mechanical specific energy are shown. Top drive measured rotational velocity and setpoint are displayed by sliders. Finally, a large stop button is located on the bottom right, which immediately terminates the script.

Figure 3.20: Front panel of the Autonomous VI

The final GUI has been designed to only provide the essential drilling data in an attempt to not overwhelm the operator. Less important data is represented using numerical indicators, while more important data is represented graphically. Errors and warnings, along with the current state and drilled for-

mation are displayed so the operator has a better situational awareness of the drilling operation. Also, the entire layout is designed to be intuitive, splitting up indicators and controls, and sectioning and aligning data representation to make it look less messy.

3.9 Emergency Stop

Although an emergency stop button was already implemented in the electrical cabinet, a shutdown sequence was implemented in all LabVIEW programs as well, which also served as an extra emergency stop. Using the subVIs for communication with top drive motor and hoisting motor (described in sections 3.4.1 and 3.4.2 respectively), zero setpoints were written to the drives, which effectively results in writing shutdown control words for the motors using the implemented logic.

3.10 Automatic File Saving

In any experiment process, gathering data for post-analysis is a very important part of the job. It is also of importance to store this data in an appropriate manner so that it is easy to find afterwards. This project was expected to require a great amount of different experiments, like PID tuning, limit testing, estimator testing, formation response testing, and state machine logic tests. For this reason, an automatic file saving system was implemented in all LabVIEW programs.

At the end of each run, whether the program was terminated by an error or by the operator, the operator is prompted to fill in a comment section, operator name, and run number if the data of the run is to be saved, as shown in Figure 3.21. The Save File subVI then checks whether or not a folder called Drilling Data already exists in the same directory as the LabVIEW program. If the folder does not exist, the subVI creates it, and if it already exists, it skips this step, and saves all data to a .txt file which is automatically named after the date of the run and the run number. In addition, an Excel spreadsheet was created with more descriptions of each run, to make navigating information simpler. A Matlab script was also created to automatically generate plots from all data, along with a PDF summary.

Though this solution took some time to implement, it made post-drilling analysis very simple and effective, allowing the team to quickly analyze data from one run, and then start the subsequent run shortly after. This allowed the team to run upwards of 30 tests in a normal working day, making the progression in experiments much more efficient than one would expect using

manual data handling and saving. The Save File subVI overview is depicted in Appendix B, Figure B.14.

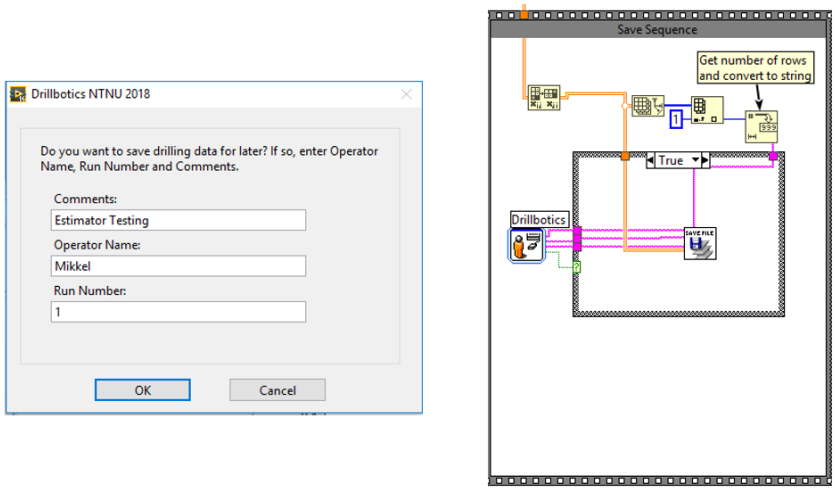

Figure 3.21: User prompt for file saving, and automatic file saving module block diagram

Chapter 4

Results

4.1 Limit Testing

Obviously, knowing the limits of your plant is of immense importance when designing a control system. One very relevant limit for the NTNU Drillbotics rig setup is the buckling limit of the drillpipe. This limit is defined by the amount of applied weight on bit that results in plastic deformation of the pipe, which can be seen by bending of the pipe.

At first, a static buckling test was performed, using the hoisting motor to push the drillpipe into the formation while reading the load cell measurements. The test ended when plastic deformation of the pipe was achieved, and the load cell measurements were analyzed afterwards. As can be seen in Figure 4.1, the weight on bit plot resembles a stress-strain curve, and plastic deformation is achieved around 130 kg, where the weight on bit starts to flat out, followed by a decrease. Note that for these tests, no setpoint is used for the weight on bit.

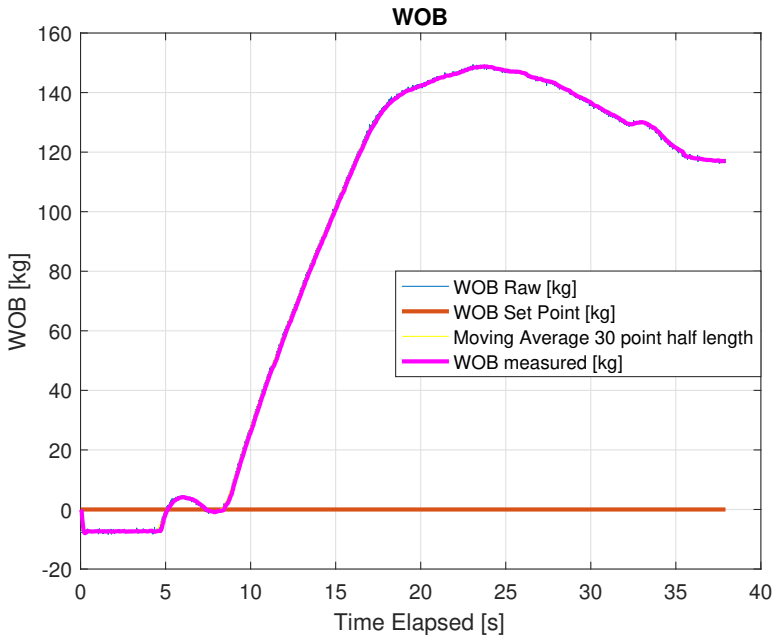

Figure 4.1: Weight on bit measurements from static buckling test.

Although the static buckling test gives useful information regarding the strength of the drillpipe, the static environment at which this test is performed is not representative for the conditions during drilling. When drilling, the drillpipe rotates at high velocities, water is circulated through it, and vibrations occur when scraping and crushing the underlying formation. Hence, a dynamic buckling test was performed under these conditions. The experiment was run in cement with 1000 RPM on the drillstring and a ramp function on the hoisting motor RPM, which would directly lead to higher weight on bit. The longest length of unsupported drillpipe was the 60 cm segment of the pipe above the rotary kelly bushing, and thus the limiting segment of pipe in terms of weight on bit. This setup represents the critical scenario where the drillbit starts drilling in the top formation layer of a tall rock sample. As the bit eventually reaches depth, the top unsupported segment length decreases, and the buckling limit will increase.

As Figure 4.2 shows, the weight on bit almost linearly increases until approximately 67 kg before it rapidly shoots in the air. This explosion in weight on bit is most likely due to stiction in the rotary kelly bushing when the pipe buckles, and it was concluded that the dynamic buckle limit for the drillpipe was around 67 kg for a 60 cm unsupported drillstring segment. By comparing Figure 4.1 and Figure 4.2, it becomes clear how much more noise

is present in dynamic conditions versus the static. The blue plot here is the raw data, while the purple plot is the filtered data using the self-coded third-order Butterworth lowpass described in Section 3.5.2. The yellow plot is an offline average which is calculated post-drilling in Matlab. This line lies beneath the filtered signal.

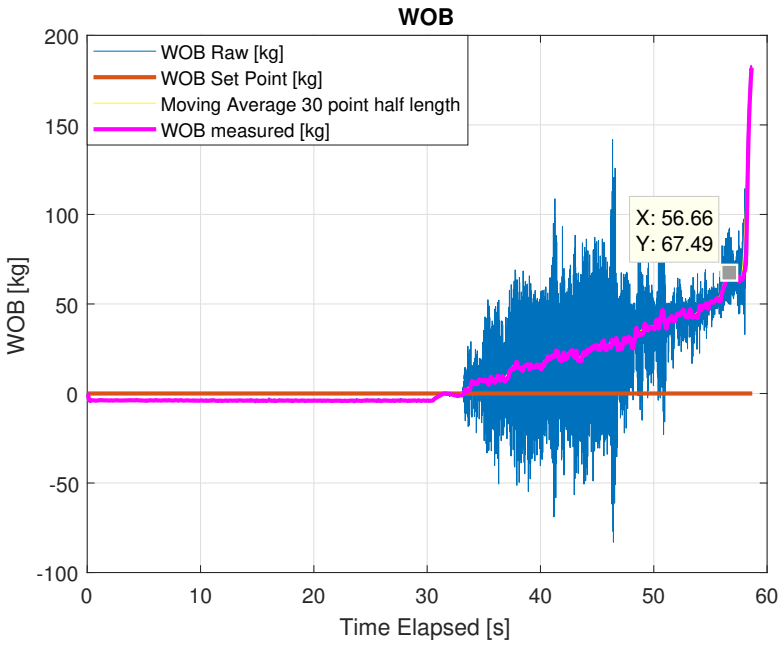

Figure 4.2: Weight on bit measurements from dynamic buckling test.

Figure 4.3 shows the hoisting motor RPM and torque during the dynamic buckling test. The hoisting motor torque is a function of the weight applied by the hoisting motor, as the motor itself meets resistance when acting on the ballscrew to push the carriage with the drillpipe and bit down into the formation. It can be seen that the hoisting motor torque increases linearly with increasing hoisting motor RPM up until the point of buckling, where it shoots upwards similarly to the weight on bit in Figure 4.2. The safety limit of the hoisting motor is set to 0.3 Nm, which is shown by the red line in the second subplot, and it is both interesting and reassuring to see that the limit is not reached even in the extreme case of buckling.

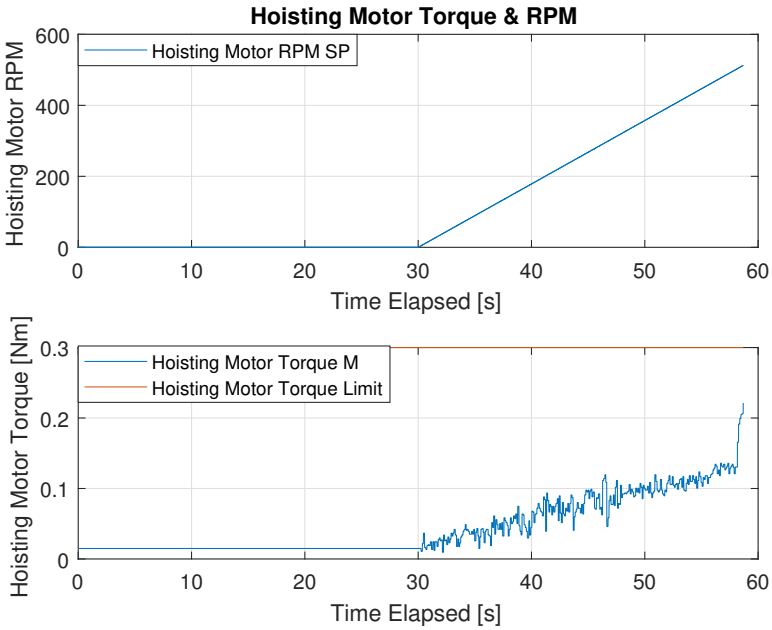

Figure 4.3: Hoisting motor measurements from dynamic buckling test.

Figure 4.4 shows the top drive rotational velocity and torque during the dynamic buckling test. The top drive rotational velocity is controlled by an internal PID controller in the frequency converter/drive, to which LabVIEW simply writes the setpoint. This PID controller came pre-tuned, and it can be seen in the first subplot. During the dynamic buckling test, the top drive rotational velocity was first subject to a step to 500 RPM setpoint, followed by a new step up to 1000 RPM before weight on bit was applied. As the plot shows, the tuning seems quite overdamped, making the response a bit slow. However, it is also a very robust controller, able to maintain the setpoint very well even in the extreme case of buckling. As the rotational velocity setpoint is not expected to be changed very often during a drilling run, the slow response was decided to be an acceptable drawback of such a robust controller, and it was decided not to tune this PID controller any further.

The second subplot shows the top drive torque, which also increases linearly as more weight on bit is applied. Finding the limit for drillstring torque has also been a central focus, as twist-off is a direct consequence of passing it. This limit was challenging to find, as it varied from run to run, leading to the conclusion that twist-off often occurred due to fatigue. At first, a standard bearing lied on top of the bottomhole assembly and bit. After drilling, it could be seen that this bearing dug into the drillpipe due to vibrations,

reducing its integrity. Using this solution, twist-off could occur at values between 6-8 Nm, although in some cases, it twisted off at much lower torque. There was also a situation with very high torque, at which the drillpipe did not twist off. This happened while drilling in cement, and one of the polydiamond-crystalline (PDC) cutters of the bit broke. The bit then got stuck in the broken off cutter, unable to drill through it. In this situation, the top drive torque value exceeded 13 Nm without twisting off. Due to the inconsistencies in twist-off data, it was decided to implement a teflon bearing rather than the standard type, which did not damage the pipe nearly as much as the former solution. After this was implemented, twist-off occurrances reduced drastically, and mostly happened at around 10-12 Nm, except for some cases where the drillstring had been used for many consecutive runs.

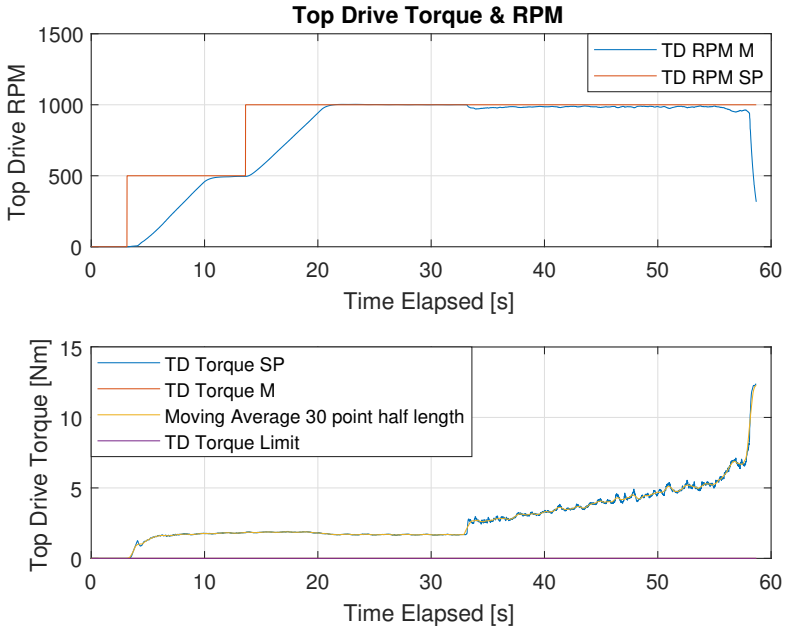

Figure 4.4: Top drive measurements from dynamic buckling test.

4.2 PID Controller Tuning

The initial attempt at tuning the weight on bit PID controller was performed in cement using the Cohen-Coon tuning method described in section 2.2. A stand-alone VI was programmed for this purpose, utilizing LabVIEW's existing Cohen-Coon tuning block, which takes the process variable (weight on bit) and controller output (hoisting motor RPM) as input and gives PID

values as output. The VI's front panel and block diagram are illustrated in Appendix B, Figure B.16 and Figure B.17, respectively.

Since the Cohen-Coon tuning process uses percentage of process variable and controller output to tune by, the first step was to define the expected ranges of weight on bit and hoisting motor RPM when drilling. The range of weight on bit was set from 0-70 kg, and the range of hoisting motor RPM was set to values corresponding to a range between 1 cm/min and 15 cm/min rate of penetration, as these were the extrema in rate of penetration from previous testing in different formations.

The experiment was set up so that the drillpipe rotational velocity was first set to 1000 RPM, before a step input in hoisting motor RPM corresponding to 7 cm/min was performed. Simultaneously, the weight on bit was monitored, and after reaching steady state, a new step was performed on the hoisting motor RPM corresponding to 15 cm/min. These step inputs can be seen in the first subplot of Figure 4.5. When the weight on bit again reached steady state, the experiment was over, and the data from the last step input was used as input to the Cohen-Coon PID tuning block.

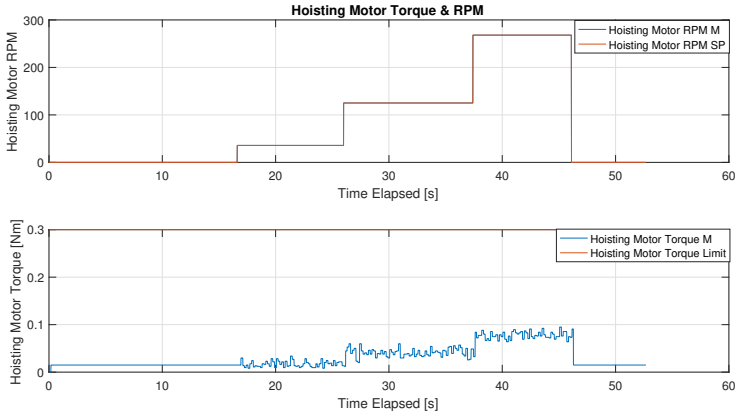

Figure 4.5: Step input in hoisting motor RPM for open loop weight on bit response testing. Hoisting motor torque also monitored.

Figure 4.6 shows the weight on bit response during the Cohen-Coon PID tuning test. Comparing this plot to that of the step inputs in hoisting motor RPM in Figure 4.5, one can observe a change in weight on bit trend at about 37 seconds, corresponding to the time of the last step input. The weight on bit plot is zoomed in on the relevant trend changing area to better observe the change.

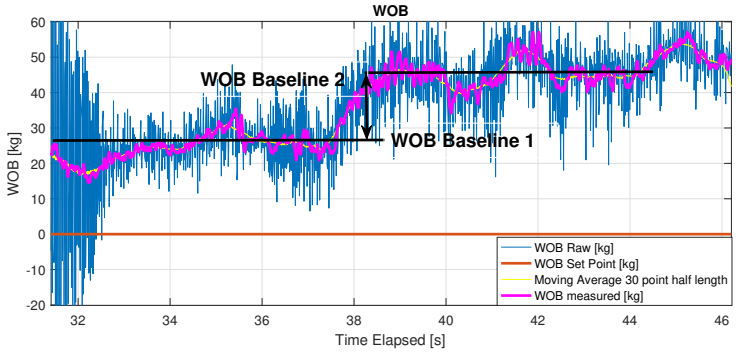

Figure 4.6: Measured weight on bit response

LabVIEW's built-in Cohen-Coon block gave output PID parameters which were highly varying, and did not yield acceptable results. For this reason, it was decided to graphically observe the weight on bit change, hoisting motor RPM change, time delay and time constant of the response, to then use Equation (2.11) and the PID controller tuning formula in Table 2.2. The weight on bit baseline before the last hoisting motor RPM step input was found to be 25 kg, and the second baseline, after the hoisting RPM step input was found to be 47 kg. To find the process gain K in Equation (2.11), the weight on bit change and hoisting motor RPM were converted into percentages of the expected operating range. As the upper limit of the PID controller output was set to 1000 RPM, this value was also used to convert hoisting motor RPM change into percentage. As previously mentioned, and illustrated in Figure 4.2, the dynamic buckling limit of the pipe was 67 kg. Thus, a 70 kg weight on bit range was used to convert the weight on bit change into percentage in the graphical interpretation method, as it also was while using the built-in Cohen-Coon block. In retrospect, this range might be a bit high, as the rig should never operate at this setpoint. However, the process gain was from this calculated to be $K = 3.84722$.

The time constant of the weight on bit change was found by calculating:

$$WOB_\tau = WOB_1 + 0.63\big(WOB_2 - WOB_1\big), \tag{4.1}$$

where WOB_τ is the weight on bit at one time constant, while WOB_1 and WOB_2 are the first and second weight on bit baselines, respectively. After this, the time of the start in weight on bit change was subtracted from the time at $WOB = WOB_\tau$, and it was found that the time constant of the response was $\tau = 0.00833$ [min]. From graphical comparison of the hoisting

motor RPM and the weight on bit plot, the time delay was found to be $\theta = 0.00333$ [min].

Plugging these values into the Cohen-Coon PID tuning in Table 2.2, the PID parameters for cement were found to be $K_p = 0.94217$ (proportional gain), $t_i = 0.00719$ (integral time in minutes), $t_d = 0.00115$ (derivative time in minutes). Note that these are plugged into the *standard form* of the PID controller, i.e:

$$u(t) = K_p\Big(e(t) + \frac{1}{t_i}\int_0^t e(\tau)d\tau + t_d\frac{d}{dt}e(t)\Big), \tag{4.2}$$

so that

$$K_i = K_p\frac{1}{t_i}, \tag{4.3a}$$

$$K_d = K_p t_d, \tag{4.3b}$$

where K_i is the integral gain and K_d is the derivative gain [16].

The Cohen-Coon weight on bit tuning performed in cement gave a good and quick response right away. It could also be seen that the hoisting motor RPM setpoint, which is the output of the controller, was much less varying than an early tuning using a trial-and-error approach, indicating that the actuation also became more cost-effective as well, as the motor could track this less varying reference much better. It was however observed that this tuning gave a much poorer response when drilling in other formations. For this reason, it was initially believed necessary to tune PID controllers for each formation type.

As the PID tuning process resulted in good tuning for cement, it continued to be the tuning method of choice for the other formation types as well. The process continued to provide good tuning for the other formations, and very little further tuning was necessary. For this specific system, it was difficult to conclusively see the quarter amplitude damping response expected when using Cohen-Coon, however, it was often seen that the tuning resulted in fast convergence, and an underdamped response, with overshoot at step changes in controller setpoint. Out of PID tunings in cement, granite and shale, it was found that the controller tuned in shale gave the best overall results for all tested formations. This led to discarding the idea of tuning multiple controllers, allowing further simplification of the design by using just this controller independent of drilled formation. For this controller, the Cohen-Coon PID gains obtained were $K_p = 5.326$, $t_i = 0.005055$, $t_d = 0.000777$. This tuning was used in all subsequent testing.

As the plant dynamics can change abruptly in this process, it was of interest to ascertain whether the PID controller tuned for one formation type could safely handle a change in drilled formation. The two scenarios of interest were when transitioning into a harder formation type, and transitioning into a softer formation type. Figure 4.7 is from a run with shale-tuned PID parameters, starting in shale, and then hitting cement. At approximately 65 seconds, the formation change occurs, at which point the weight on bit drops. This is due to the fact that for a harder formation like shale, less actuation is needed to achieve higher weight on bit. However, the response quickly converges to the 40 kg setpoint in cement as well, indicating that a change to softer formation should not be an issue.

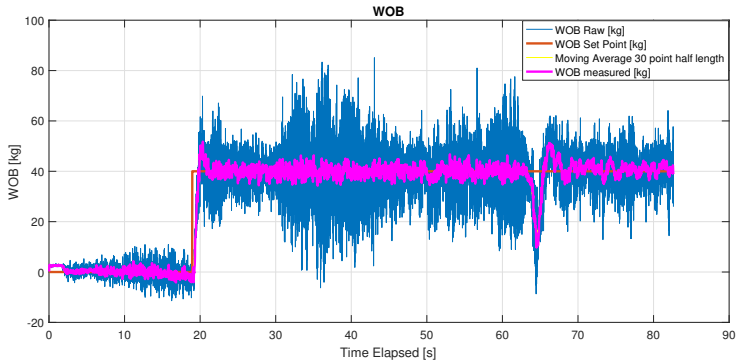

Figure 4.7: Effect on weight on bit when transitioning from shale to cement using Cohen-Coon PID parameters for shale.

During experiments, it was found that the event of transitioning into a harder formation type was more critical. Since several of the PID controllers tuned for softer formations have lower gains, they were slower in response, which resulted in excessive overshoots when hitting harder formations, followed by a slower control action back to setpoint. To avoid buckling the pipe, it was deemed necessary to take measures to avoid this type of response. The method of choice was to re-initialize the PID block, effectively removing integral action when measured weight on bit reached a threshold of 15 kg above setpoint. Figure 4.8 shows a run from shale to granite with shale-tuned PID parameters. At 32 seconds, the change occurs, and the weight on bit quickly increases. Then, the weight on bit rapidly sinks towards the 30 kg setpoint, converging nicely. Upon inspection of Figure 4.9, it can be seen that there is a quick drop in actuation at the time of the formation change, which is the result of the re-initialization of the PID block.

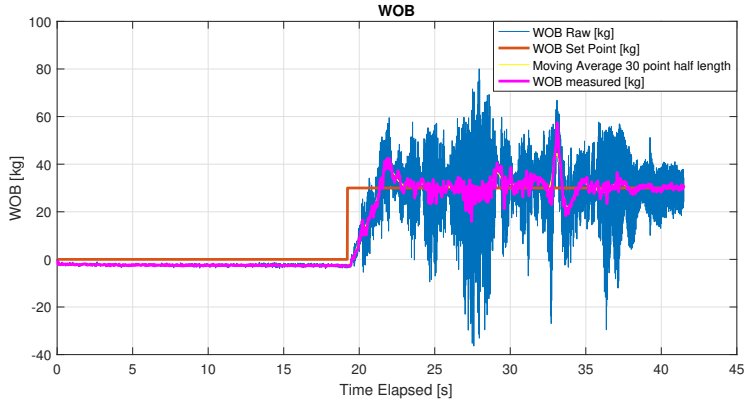

Figure 4.8: Effect on weight on bit when transitioning from shale to granite using Cohen-Coon PID parameters for shale along with re-initialization of integral action.

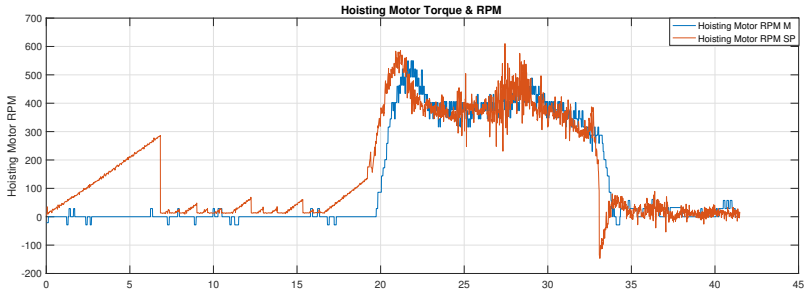

Figure 4.9: Hoisting motor RPM during transition from shale to granite with re-initialization of PID parameters

4.3 Drilling Parameters

This section describes the formation response testing sequence used to determine optimal drilling parameters for the tested formations. As the competition crate contains a variety of unknown formation types, it was apparent that tests should be performed on formations we might expect to encounter on the competition day, ranging from soft to hard. To perform these tests, a separate Formation Response VI was programmed to run a pre-defined sequence. This sequence was constant for each run, and was tested on granite, shale, basalt and cement.

The test program ran one-minute cycles, where the first cycle starts at 10

kg weight on bit, and is step-wise incremented by 5 kg each successive cycle. Meanwhile, the drillstring rotational velocity is ramped up and down between 700 and 1800 RPM every other cycle. To give the weight on bit controller some time to reach steady state at each cycle, the cycles start off with 10 seconds of constant top drive rotational velocity before commencing 50 seconds of ramping. The block diagram is given in Appendix B, Figure B.20.

The formation testing sequence ensures that rate of penetration can be logged for every combination of the 5 kg increments of weight on bit and every rotational velocity within the previously defined range. Figures 4.10 and 4.11 illustrate the testing sequence for an actual run in granite, the hardest formation subject to these tests.

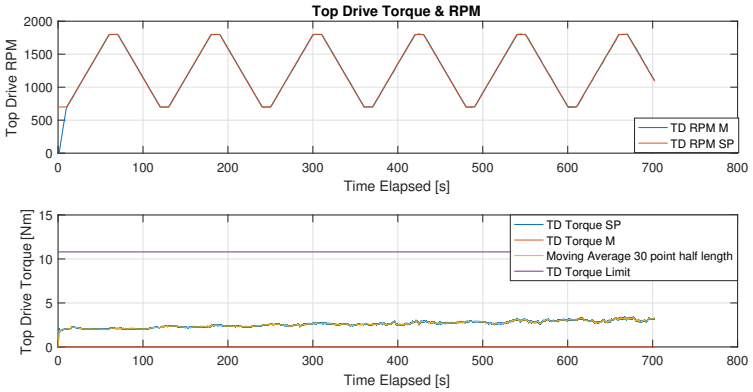

Figure 4.10: Top drive rotational velocity sequence for formation response testing.

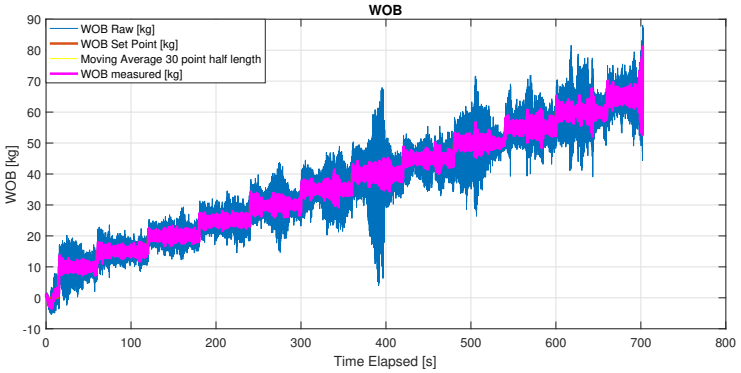

Figure 4.11: Weight on bit sequence sequence for formation response testing.

As the purpose of these tests were to find optimal drilling parameters in terms of rate of penetration, the RPM, weight on bit, and ROP data collected from testings were plotted in a 3D scatter plot to find the optimal drilling parameters. Figure 4.12 shows this response in granite. It is clear to see that for this specific formation type, increasing weight on bit and drillstring rotational velocity will yield increasing rate of penetration. The global optimum of this data set can upon inspection be seen to be $ROP = 1.45$ cm/min at $\omega_{ds} = 1800$ RPM and $WOB = 66$ kg. Though this weight on bit value is close to the buckling limit found in the dynamic buckling test described in section 4.1, this test was performed in a lower rock sample, leading to shorter unsupported drillstring segments. Under these conditions, the buckling limit of the pipe may be much higher, explaining why buckling did not occur during this test. In a later run, a similar formation response test was performed in granite up until 95 kg weight on bit, still without buckling. Still, in an autonomous run, the depths of formation layers are not known beforehand, and finding the buckling limits for all depths would require extensive testing, for which there was not enough time. For this reason, it was decided to choose drilling parameters in compliance with the previously found limits.

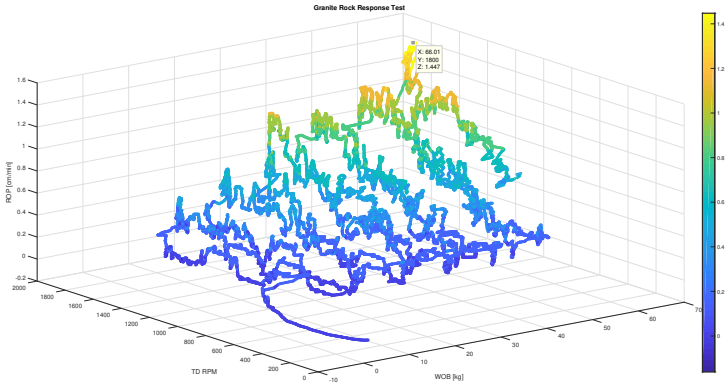

Figure 4.12: Formation response for granite.

The results were conclusive for all the tested formation types. With the allowable setup for the competition, the drilling operation is not limited by the Founder's point, but rather by the mechanical limits of the aluminum drillstring. This was indicated by an increase in rate of penetration for increasing drilling parameters in all tests. The optimization of the drilling process therefore simply turned into a search for the maximum allowable drilling parameters while still maintaining the integrity of the well. Through extensive testing and experience gained along the way, it was found to be

safe to drill the competition sample at 50 kg weight on bit and 1300 RPM top drive rotational velocity since the rock was low, resulting in shorter unsupported drillstring segments.

4.4 Estimator

The Identification phase in the state machine was used to obtain regression model parameters to put together a library of formations. As explained in section 3.7.5, both weight on bit and rotary speed is constant during the first 5 seconds of estimation, at 10 kg and 700 RPM, respectively. Then, weight on bit is ramped to 30 kg over the next 15 seconds while keeping rotary speed constant. Lastly, rotary speed is ramped to 1200 RPM over the next 15 seconds while keeping weight on bit constant. Then, the estimation is finished. This sequence is shown in Figures 4.13 and 4.14.

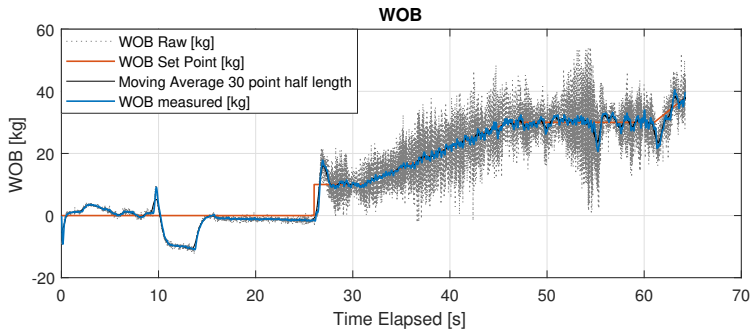

Figure 4.13: Weight on bit sequence during estimation.

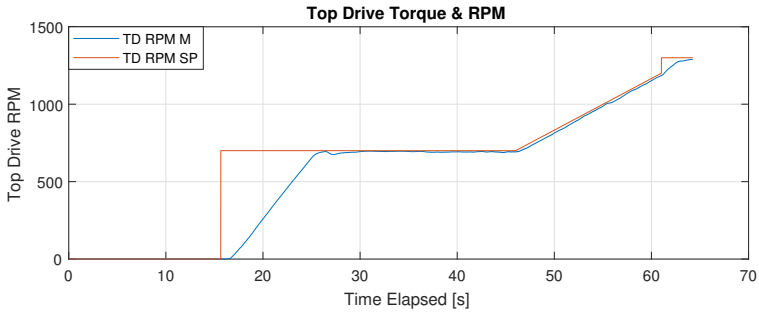

Figure 4.14: Top drive rotational velocity sequence during estimation.

Figure 4.15 depicts obtained regression model parameters for an estimation

run in shale. The estimation starts at approximately 25 seconds elapsed time, and it can be seen that at 30 seconds, and 45 seconds, the parameters change quite drastically. At these points in time, the ramping of weight on bit and top drive rotational velocity begins, at which points the estimator is obtaining new information regarding the effects of the drilling parameters on rate of penetration. This explains these changes in the plot. It can also be seen that towards the end, the parameters converge very nicely towards their final values, which indicates that the estimator is confident in its calculation of the parameters.

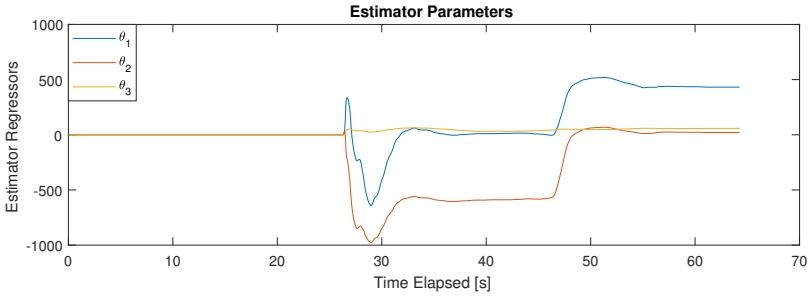

Figure 4.15: Resulting regression model parameters for an estimation test performed in shale.

In an attempt to prove the concept of the recursive least squares estimator to determine drilled formation, it was tested on four different formation types: cement, shale, basalt and granite, where cement is the softest of the four, then shale, basalt, and finally granite being the hardest. As can be seen from Table 4.1, the regression model parameters support this statement by the way θ_1, describing the effect of formation drillability, trends from high to low moving from cement to granite. Looking at θ_2, it can be seen that drillstring rotational velocity has a negative effect on rate of penetration for cement, basalt and granite, which is in opposition to the team's experience this semester. This could be due to coincidences like formation inhomogeneities and vibrations due to small misalignments.

	Cement	Shale	Basalt	Granite
θ_1	1653	431.7	273.9	232.5
θ_2	-3.4	20.8	-3.4	-4.7
θ_3	175.3	57.7	3.9	8.1

Table 4.1: Obtained regression model parameters for the tested formation types.

Figure 4.16 represents the regression model parameters in $\mathbb{R}^3$. It shows that

cement and shale are quite far apart, while basalt and granite have quite similar properties, which should challenge the estimator. For this reason, it was decided to test the state machine with the estimator in a formation containing several layers of cement and basalt. This way, a simple reservoir containing only the two formations could be made, but at the same time giving the estimator two erroneous options to choose from when identifying the formations. The results from this experiment is further elaborated in 4.5.2.

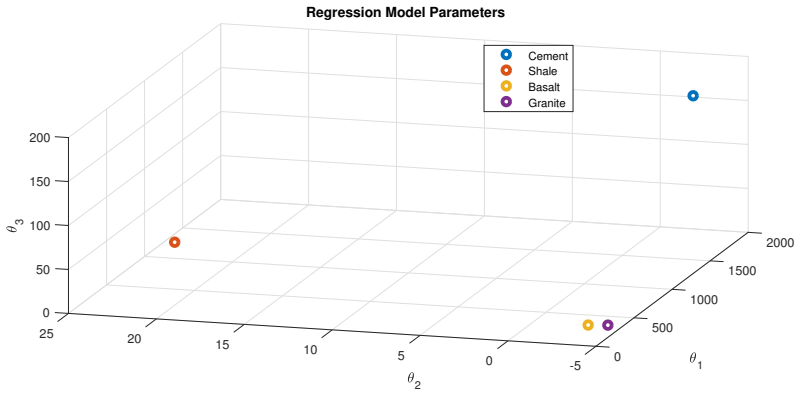

Figure 4.16: Regression model parameters for the formation library represented in $\mathbb{R}^3$

4.5 State Machine Propagation

4.5.1 Competition Day Script

Since the formation response testings showed that we were unable to reach the Founder's point in the tested formations, the team concluded that we were limited by the mechanical limits of our setup. This led to the conclusion that higher weight on bit and higher top drive rotational velocity would result in higher rate of penetration. For this reason, the script used on the competition day was simplified by removing the Identification phase altogether. Other than this, the script was exactly the same.

The provided formation sample was 35 cm in height, not 60 cm as initially believed, and it consisting of sandstone, cement, an inclined layer of tile followed by an inclined layer of limestone, then cement again, and flatstone followed by a second layer of sandstone. Of these, the limestone and flatstone are the hardest formation types, while the cement is less homogeneous,

generating more drillstring torque spikes. The sandstone was expected to be the easiest to drill.

Figure 4.17 shows the weight on bit setpoint and measurement throughout the competition run. As can be seen, when entering the Drilling phase, the weight on bit is ramped up over 30 seconds to its setpoint of 50 kg before flattening out. During ramping up the weight on bit, the drillbit passes the sandstone and hits the cement. As approximately 120 seconds, the bit hits the tile layer and gets stuck briefly. Here, the PID switches to torque control, and starts hoisting up to get unstuck before commencing drilling once more. The thin tile layer is quickly passed and the limestone is drilled after 140 seconds elapsed to 170 seconds elapsed. After this, the remaining cement, flatstone and sandstone is drilled before reaching the bottom of the wooden crate containing the formation sample at approximately 230 seconds elapsed, leading to a drop in weight on bit as it is hanging in free air.

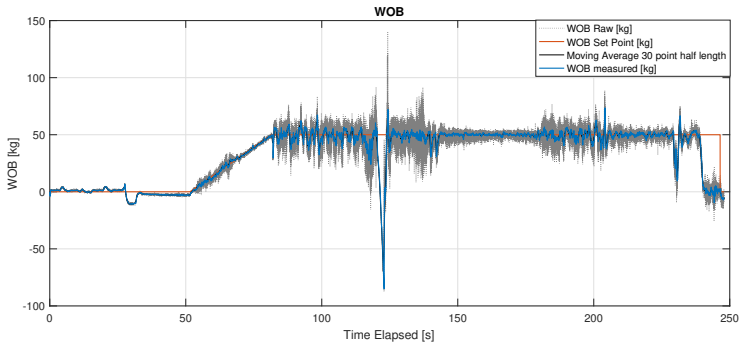

Figure 4.17: Weight on bit profile in the competition run.

As previously mentioned, the drillstring gets stuck when hitting the tile layer, which is clear to see from Figure 4.18. The top drive rotational velocity drops down towards zero while the torque skyrockets above the pre-defined torque limit.

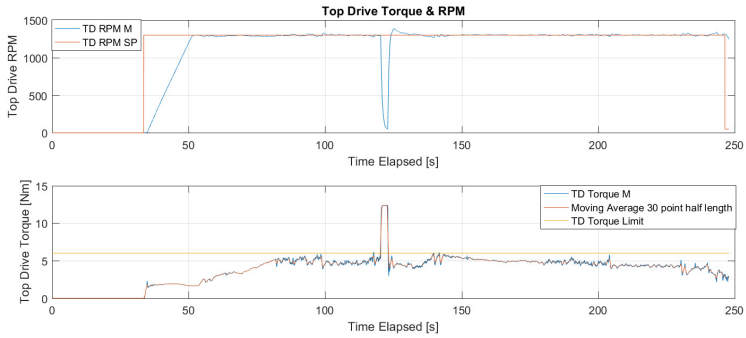

Figure 4.18: Top drive rotational velocity and torque throughout the competition drilling.

Figure 4.19 shows the propagation of the state machine throughout the competition drilling session. It starts out in the Initialization phase, tagging the formation, before it hoists up, creating room between the bit and the underlying formation. Then drillstring rotation is initiated before moving into the Drilling phase. It can be seen at the very end of the plot that the state machine transitions into the Trip Out phase at the end. This phase should have lasted longer, while the bit slowly pulls out of the formation while maintaining a low rotational velocity. However, when the team observed that the drillbit had gone through the wooden crate, closing in on the floor below, the operator manually terminated the program. It was first seen post-drilling that the state machine had worked perfectly, and had initiated tripping out. The drilling operation itself took 3 minutes and 15 seconds through the 35 cm reservoir. This gives an average rate of penetration of approximately 10.8 cm/min, which was much faster than anticipated. To our knowledge, most other teams in the competition spent between 15 to 18 minutes on identical reservoirs.

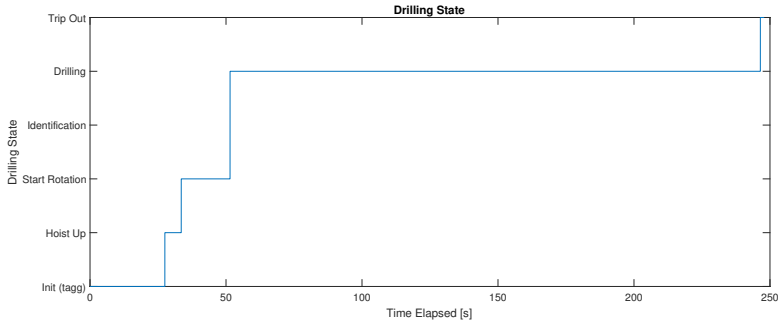

Figure 4.19: State machine propagation when drilling through the unknown competition formation.

4.5.2 Estimator Script

The estimator script was tested in a formation containing several layers of basalt and cement. For the particular run presented in this section, the drilling starts out in basalt. As can be seen in Figure 4.20, the estimator correctly identifies the formation. This was an uplifting result, as the granite lays close to basalt in terms of drillability properties and regression model parameters, as seen in Table 4.1 in section 4.4. The same day, several runs were performed in this formation, and the estimator was able to correctly identify basalt in every run after fixing a bug resulting in integrator wind-up in the autonomous script. It was desirable to perform a run successfully showing the entire state machine propagation, including identifying basalt, detecting new layer, and then identifying cement before tripping out. However, there were problems with the connection on the drillstring, which caused it to loosen mid-run, so these results were unfortunately not obtained.

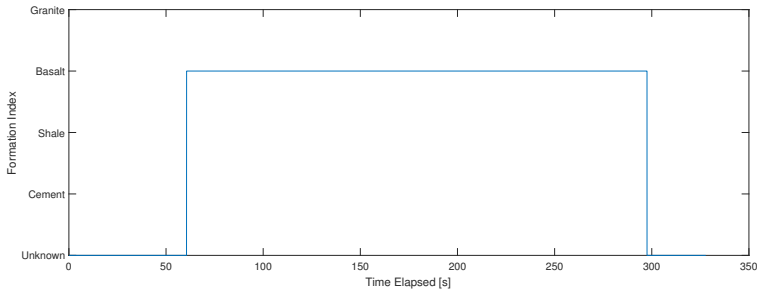

Figure 4.20: Correct identification of basalt in an autonomous run.

Figure 4.21 shows the state machine propagation for the run. As always, the

system starts by tagging formation, hoisting up and starting rotation before entering the Identification phase. After this phase, the system continues drilling in the basalt before encountering a basalt/basalt interface, which is correctly detected. At this point, the bit is hoisted up, and rotation is set to 700 RPM again, before initiating the Identification phase once more. In the middle of this phase, the connection to the drillstring loosened, and the script was terminated.

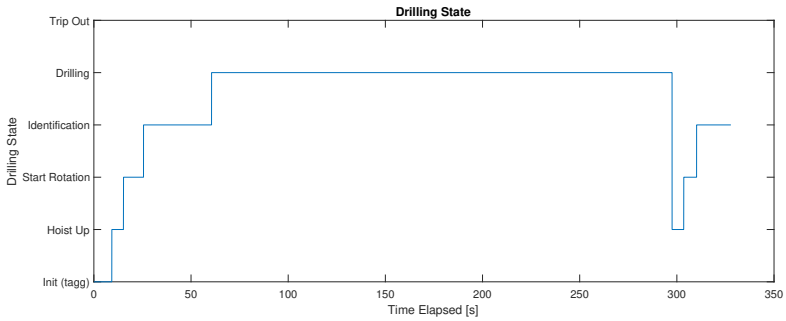

Figure 4.21: Propagation of the state machine.

In Figure 4.22, it can once more be seen that the regression model parameters for the first estimation has converged nicely at a little past 50 seconds of elapsed time. Comparing this to Figure 4.21, it is observed that the state machine is in the Identification phase until approximately 60 seconds of elapsed time. When the state machine transitions into the Drilling phase afterwards, the program simply retains the regression model parameters throughout the phase, which is why they have flattened completely out. At the very end, the final estimation is performed, where the script has been terminated, so the parameters have not had the time to converge well.

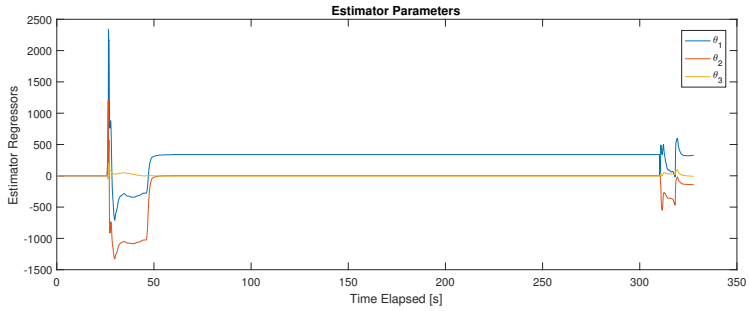

Figure 4.22: Calculated regression model parameters for the drilled basalt.

While this run indicates that the recursive least-squares estimator is a feasible solution to determining drilled formation, not all results looked this promising. A couple of days after this successful run, more attempts were made to perform a more complete run, showing identification of both basalt and cement, and then tripping out. On this day, the parameters varied much more, and the program misidentified basalt as granite or shale. Also on this day, the drillstring connection loosened, consuming too much time to get any good runs. This led to the hypothesis that there might be more affecting factors than the regression model captures. One factor could be drilled depth. On this day, the well from the successful run was continued. This leads to a longer segment of unsupported pipe downhole, which was seen to cause friction in the bearings, most likely giving slightly erroneous readings on the load cell. The bit used was also worn, and it was considered whether the bit had been worn more since the previous runs, also affecting the rate of penetration response. The last considered effect was the loosened drillstring connection. As it was found to be loosened at the end of the runs, there was no definitive way of determining when it occurred. If the connection had started to loosen early in the run, the top drive rotational velocity measurements would not be 1:1 in ratio with the drillbit, which again obviously would affect the rate of penetration response. These factors should be further investigated in the upcoming semester by the next team if further research on the estimator is considered. More ideas on improvement on the estimator is given in section 7.

Chapter 5

Discussion

Overall, the final control system proved to be robust and safe, which were the main priorities for the design. The team was enthused with the functionalities of the end product, both regarding the control system and the novel mechanical solutions implemented. The multi-disciplinary nature of the project has allowed the team to learn new things across the petroleum and cybernetics fields, and has required close cooperation between all members, giving valuable practice in project management and planning as well.

The estimator has been verified in practice, and has proven to be a candidate solution for identifying drilled formation, and further work on this is encouraged. The setup herein presented is vulnerable to changes on the rig, including but not limited to mechanical fixes, PID tuning, bit selection, bit wear etc. It could be interesting to include effect of bit selection and bit wear to the model, and the effects of these are given in Bourgoyne et al [15].

The scope of the work has been comprehensive and ambitious, and almost all elements discussed in the beginning of the project have been successfully implemented. However, the design is not perfect, and further work is always required to maintain and improve on solutions. The team has put significant effort in all our endeavors to modularize code and to achieve systematic documentation of drilling data and lessons learned to lay a foundation for an efficient handover to next year's team. We believe we have been successful in this, and hope that our successors can use our work to quickly get up to speed on understanding the design and its functionalities, strengths and weaknesses.

Chapter 6

Conclusion

The work revolving the control system design for the Drillbotics competition has proven to be very educational, allowing coupling theory with practical problem solving. It has been a comprehensive project, requiring diving into many aspects of control system and drilling theory. As stated in section 1.3, the goals of the project were:

- Setting up the communication infrastructure for the system.
- Choosing a suitable control scheme and PID controller tuning.
- Setting up manual PID modes for test-drilling.
- Implementing efficient filtering for sensor data.
- Implementing an estimator to determine drilled formation.
- Design of a user-friendly and intuitive GUI.
- Setting up automatic file saving for post-drilling analysis.
- Implementing a functioning state machine, representing the different phases of drilling.

Although the selection of communication protocols and hardware modules for the top drive and hoisting motors was done by Steffen Wærnes Moen, setting up the communication infrastructure within LabVIEW, both with actuators and sensors, was performed in its entirety by myself and Andreas Thuve. A lot has been learned about data acquisition using National Instruments' DAQ modules and setup of Modbus communication via Ethernet cable, and along the way, the communication setup has been improved to streamline the code and use less memory, effectively leading to a reliable code consistently running at 100 Hz over time.

The selected control scheme was primarily weight on bit controlled by the hoisting motor, and rotational velocity controlled by the top drive. In the event of torque exceeding a pre-defined torque limit, the weight on bit controller switches to torque control, also utilizing the hoisting motor as actuator. Among several discussed control schemes, this has been believed to be the most reliable and simplistic, as weight on bit is much easier to control than torque. The torque control implemented in the control system is primarily meant to be used for dysfunction handling like stuck pipe, and as explained in 4.5.1, this setup worked perfectly in the competition drilling session. The PID controller was tuned using the Cohen-Coon process response method, which proved to work very well for this setup. Although it was initially believed necessary to tune PID controllers for several rock types, it was found that the shale-tuned PID worked well for all tested formation types, allowing further simplification of the control system. Also, essential elements in control, like integrator anti-windup and bumpless transfer have been implemented, and the importance of these features have been observed first-hand.

A manual drilling VI for PID controller testing was implemented for tuning and implementing the integrator anti-windup and bumpless transfer. Also, several other programs like the estimator testing VI and drilling parameter response VI have been made. These supporting VIs have allowed the team to gain extensive experience in the drilling operation before implementing the fully autonomous state machine, which in turn has allowed us to "walk" before "running".

In data acquisition and control, signal conditioning is paramount for robust control. Self-coded third-order Butterworth filters have been implemented for weight on bit, rate of penetration and mechanical specific energy. For control purposes, the appropriate filtering of the weight on bit measurements have proven important, while the filtering of rate of penetration has been important for proving the concept of the recursive least squares estimator. This is due to the fact that we have calculated the rate of penetration based on position measurements from the hoisting motor's internal incremental encoder, and the resolution of this sensor led to very discrete measurements. The self-coding of the filters have given insight and understanding in how a digital filter works.

While the estimator was not used in the competition script due to a fundamental conflict in objectives: drilling fast versus knowing drilled formation, the concept was still verified in a separate showcase script, as explained in section 4.5.2 with four rock types in the Library VI. Differentiating between basalt and granite, two rock types with very similar properties, has been especially satisfying, and proves the concept that with a good estimation setup and reliable measurements, it is possible to confidently determine the properties of drilled formation using this method.

Effort has been put into making the GUI of the script as intuitive and user friendly as possible. It shows the status of data acquisition from sensors, communication with actuators, along with indicators for new formation layers, high torque values, and twist-off. Also, controls for inputting drilling parameter setpoints and limits are present. Live charts, gauges and sliders present essential drilling information as well.

An automatic file saving module, saving drilling data to .txt files, was implemented in all test programs for post-drilling analysis. A Matlab script was also written to load the .txt files and create all relevant plots along with a short .pdf summary report. This work took some time, but in the end saved the team many hours of manual data handling. Data from each run could be analyzed instantly after each experiment, and allowed the team to log and systematically save data for well over 300 drilling runs just over the last month of the semester.

The state machine divides the drilling process into phases that fully automate from tripping in to tripping out, including termination criteria like critical weight on bit values, torque values and reaching target depth. In addition to the state machine with the estimator, a simplified version omitting the identification feature has been implemented and used on the competition day. Both scripts have performed well in different testing environments, thus verifying that the designs are robust and versatile. The simplified script used on the competition day performed beyond expectations, drilling through the competition rock in 3 minutes 15 seconds, yielding an average rate of penetration throughout the operation of 10.8 cm/min. The estimator script was not used in this capacity due to the fact that the Identification phase requires approximately 35 seconds to confidently determine the parameters of the drilled formation. Since the competition rock was unknown, one could assume that there might be very thin layers within it, which could be drilled through before the Identification phase was finished. This was a complicating factor which was concluded could lead to problems. However, this design was verified separately, and could be a building block towards parameter estimation in full-scale drilling operations.

Chapter 7

Further Work

Although a lot of the work has gone very well, several areas for improvement and further development in the control system have been identified. Due to limited time, the estimator was not tested to the extent initially desired. For the experiment described in 4.5.2, regression model parameters for only one run in each tested formation type was implemented in the Library VI, and the Euclidean distance between the unknown formation and the formations stored in the Library VI are calculated. Then, the minimum distance is chosen to classify the unknown formation. This leaves the setup vulnerable to small changes in the setup, especially for formation types with similar properties. One way to improve this setup is to implement an algorithm commonly used in machine learning, i.e. the k-nearest neighbors algorithm, also known as k-NN. To do this, one would perform several runs in the tested formation types and define them in the Library VI. One would then need to select an appropriate value for k, and then classify the unknown formation based on what other formation type is the most common among its k nearest neighbors. There exists several variants of this algorithm, utilizing different measures of distance, but because of the low dimensionality of the regression model, it is believed that Euclidean distance is a good starting point also in this setup.

Also, in the presented setup, there is a separate Identification phase, using a recursive least squares estimator with forgetting factor $\lambda = 1$, meaning no forgetting. Another idea for implementation could be to completely omit the Identification phase, continuously running the estimator throughout the drilling operation and introducing forgetting to continuously identify drilled formation. This would also simplify the design since a separate change detection algorithm would then be superfluous. For this to work, it should be considered to excite the system continuously as well, by for example stepping or ramping drilling parameters back and forth to provide the estimator

with information regarding the effect of the drilling parameters on rate of penetration.

Implementing the pump in the control system was initially planned, though it was eventually down-prioritized due to more important matters. In the current setup, the pump is manually controlled by turning on the motor drive and using a knob to control its rotational velocity. It was planned to implement pressure control in LabVIEW, using feedback from the pressure gauge, and further work on this for a more elegant solution should be considered. If this setup is desired, communication with the pump motor could be set up via the National Instruments DAQ modules.

The cDAQ chassis and analog/digital I/O modules acquired were not implemented, and a USB-6009 DAQ was used instead due to simplicity when setting up. The modules compatible with the cDAQ chassis include analog filters for noise rejection, which could further improve on the signal conditioning and avoid aliasing, although this was not experienced to be problematic for this year's team. This change would also modularize the hardware setup, as modules can be removed and appended to the chassis for example when deciding to remove or add sensors or control of new actuators if so desired.

The ClampOn SandQ sensor and downhole accelerometer/gyroscope were implemented towards the end of the semester, and further work to utilize these in the control system should be consideres. The ClampOn SandQ sensor is not implemented in LabVIEW currently, but could be useful to observe eigenfrequencies of the system along with change detection when drilling. The team has been in contact with ClampOn this semester for help installing their software and connecting to the sensor. They have also said that they are quite familiar with LabVIEW, so they can be of assistance to next year's team if further work is performed on this. The DSATS jury for the Drillbotics competition have said that they would like teams to focus on actively using the downhole sensors for control, rather than just logging and monitoring this data, like most teams have done so far. Finding the appropriate way to utilize these measurements could also be an area of focus further on.

Bibliography

[1] http://www.digitalistmag.com/industries/oil-and-gas/2015/01/14/top-challenges-facing-oil-gas-industry-conquer-02050223, 18.04.2018

[2] https://drillbotics.com/, 18.04.2018

[3] John D. Macpherson, John P. de Wardt, Fred Florence, Clinton D. Chapman, Mario Zamora, Moray L. Laing, Fionn P. Iversen, *Drilling-Systems Automation: Current State, Initiatives, and Potential Impact,* SPE, 2013

[4] https://www.electronics-tutorials.ws/filter/filter_8.html, 16.04.2018

[5] https://en.wikipedia.org/wiki/Butterworth_filter, 16.04.2018

[6] https://en.wikipedia.org/wiki/Bilinear_transform, 17.04.2018

[7] https://en.wikipedia.org/wiki/Digital_filter, 17.04.2018

[8] http://blog.opticontrols.com/archives/383, 20.04.2018

[9] https://www.dataforth.com/tuning-control-loops-for-fast-response.aspx, 20.04.2018

[10] http://blog.opticontrols.com/archives/1066, 20.04.2018

[11] K. J Åström, B. Wittenmark, *Adaptive Control, 2. edition,* Addison Wesley Longman, 1995

[12] S. Haykin, *Adaptive Filtering Theory,* Prentice Hall, 2002

[13] A. Handeland, S. Knoop, A. Thuve, P. Ø. Turøy, *Designs Considerations for Miniature Autonomous Drilling Rig,* 2018

[14] http://www.clampon.com/wp-content/uploads/2014/11/ClampOn_SandQ_Aug2014.pdf, 07.06.2018

[15] A. T. Bourgoyne Jr., F. S. Young Jr., *A Multiple Regression Approach to Optimal Drilling and Abnormal Pressure Detection,* SPE, 1974

[16] https://en.wikipedia.org/wiki/PID_controller#Ideal_versus_standard_PID_form, 20.05.2018

Appendix A

SandQ vs Weight on Bit

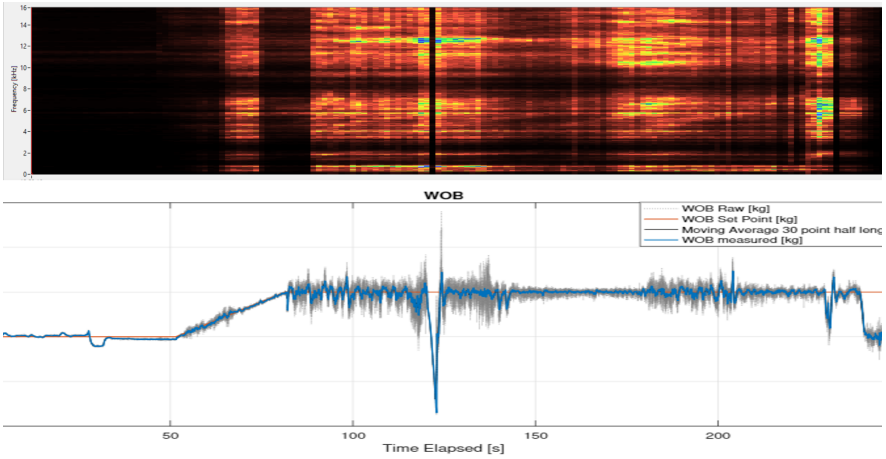

Figure A.1: SandQ sensor plotted with weight on bit measurements on the competition day. It can be observed that major events occurring at approximately 145 seconds and 230 seconds are indicated in both plots.

Appendix B

Additional LabVIEW Code

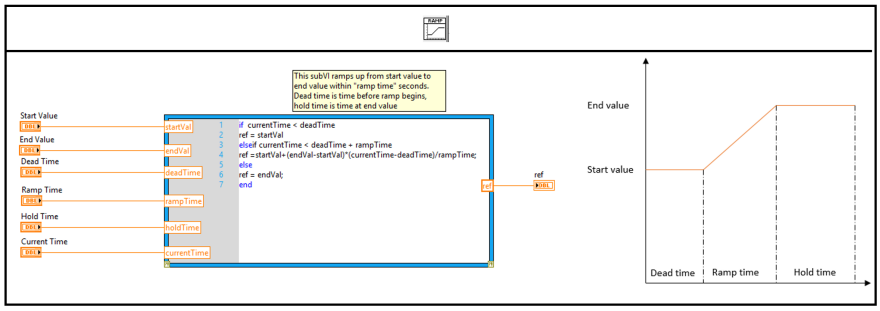

Figure B.1: Ramp function subVI used in the Identification phase, Drilling phase, and the formation response testing.

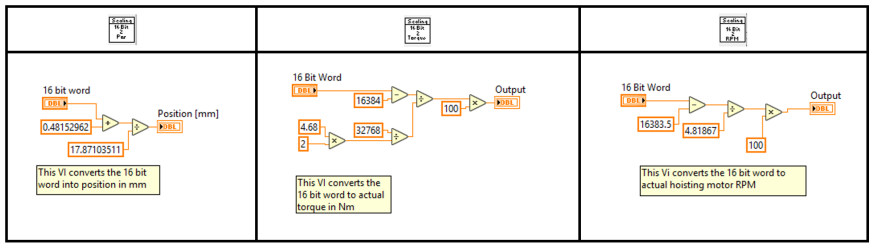

Figure B.2: Conversions for the hoisting motor Modbus communication, from 16 bit words to correct measurements.

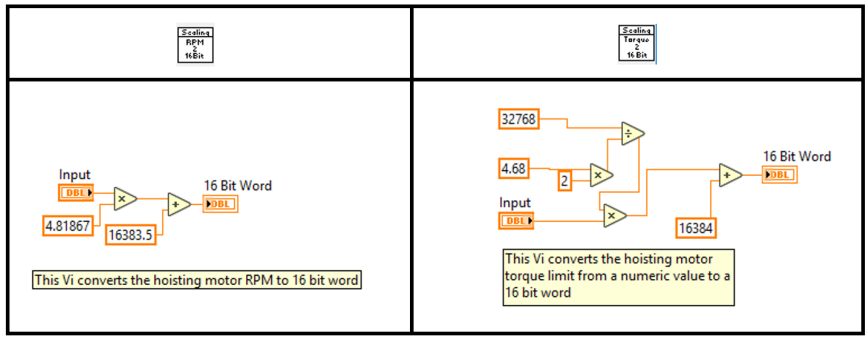

Figure B.3: Conversions for the hoisting motor Modbus communication, from setpoints to 16 bit words.

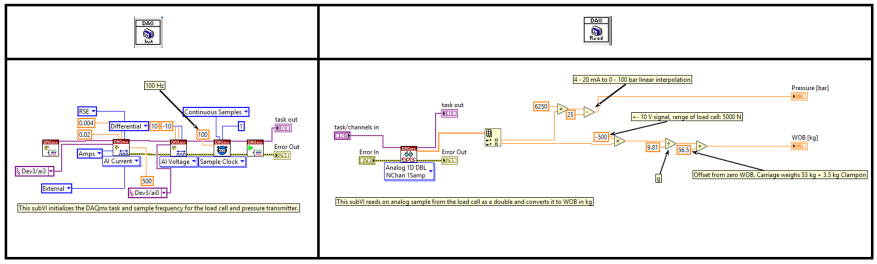

Figure B.4: Leftmost: initialization of DAQ pressure and load cell measurements. This subVI is located outside the while loop in the program and executes only at the first iteration. Rightmost: Read subVI for measurements, continuously updated each iteration.

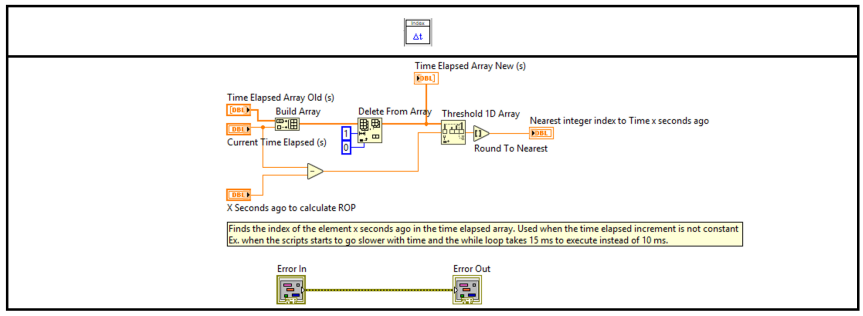

Figure B.5: SubVI created to find the element at x seconds ago. This due to small variations in iteration speed. This was used for ROP calculations, and also within the formation change detection.

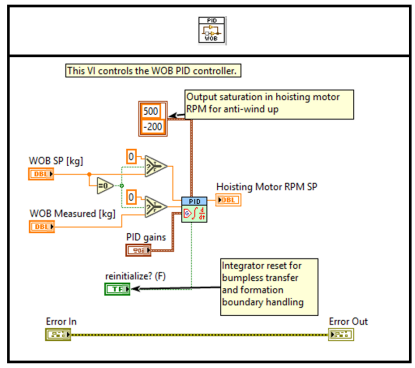

Figure B.6: Weight on bit PID subVI. Includes re-initialization of integral action. Also implemented logic to write measurements to zero if setpoint is zero to avoid wind-up.

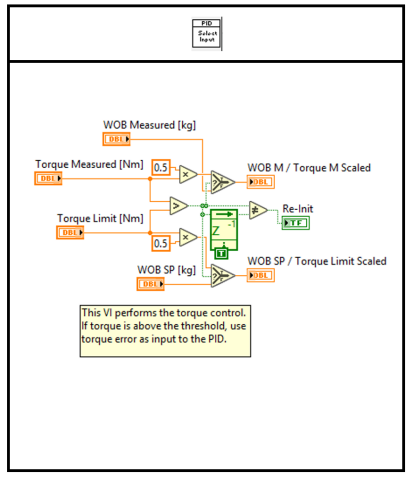

Figure B.7: PID mode selector subVI. If torque is above torque threshold, switch to torque mode. If not, use weight on bit mode. Re-initialization of integral action when switching between modes.

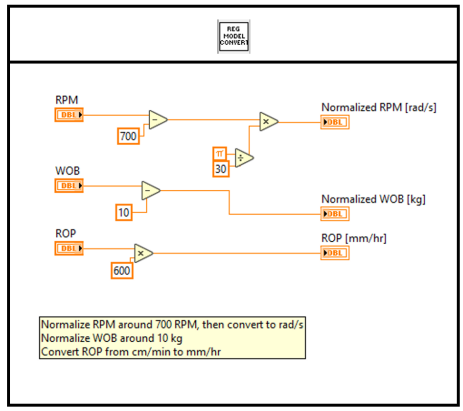

Figure B.8: Conversion of ROP to mm/hr and RPM to rad/s. Also normalization of weight on bit and rotational velocity at 10 kg and 700 RPM, respectively. Unit conversions purely for aesthetic purposes.

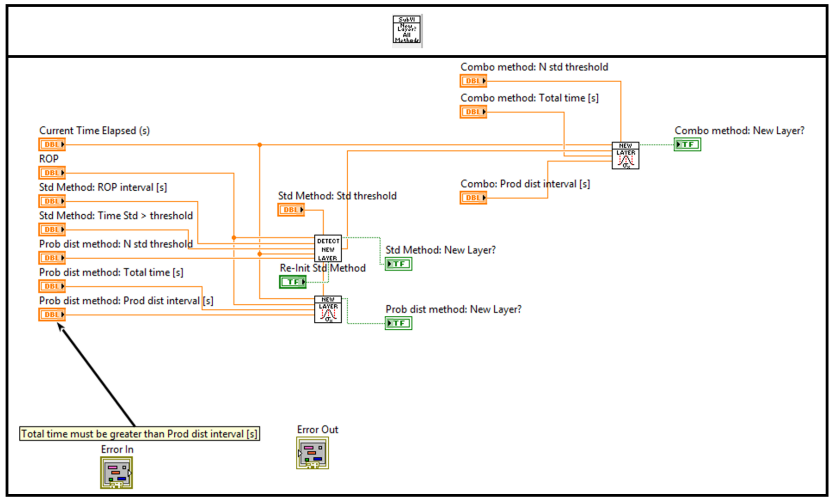

Figure B.9: SubVI containing 3 different change detection algorithms tested. Ended up using the method comparing recent mean ROP to previous probability distribution.

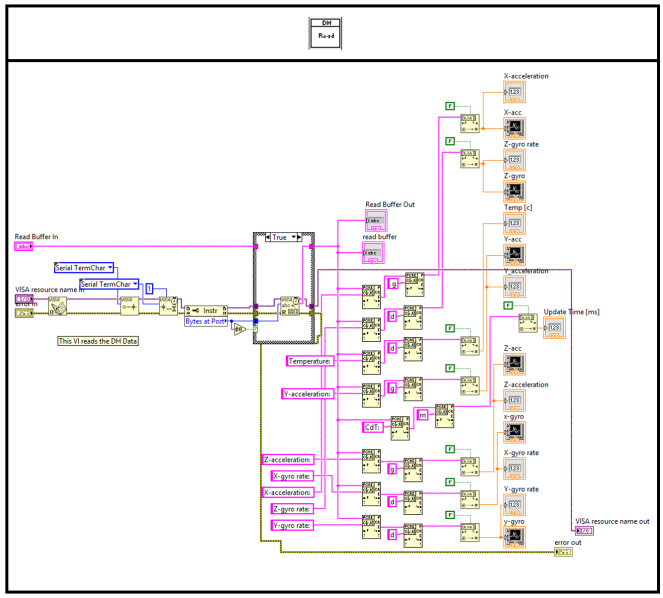

Figure B.10: Dowhnhole sensor reading subVI programmed by Steffen Wærnes Moen.

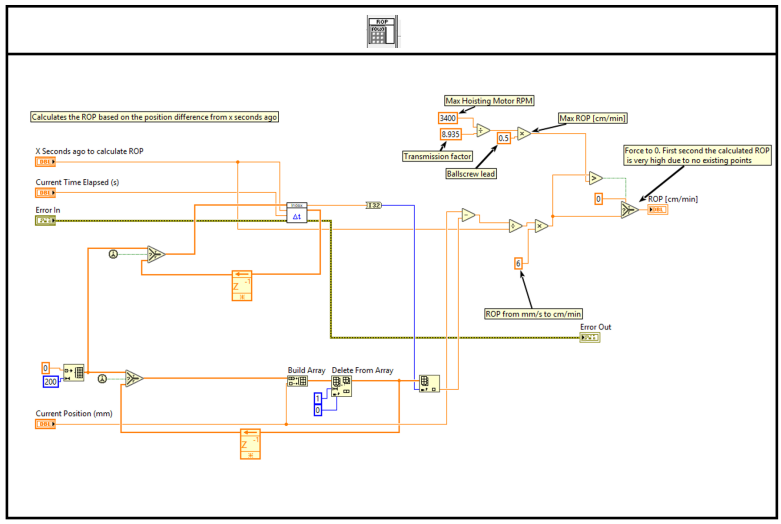

Figure B.11: SubVI calculating rate of penetration using position difference over time.

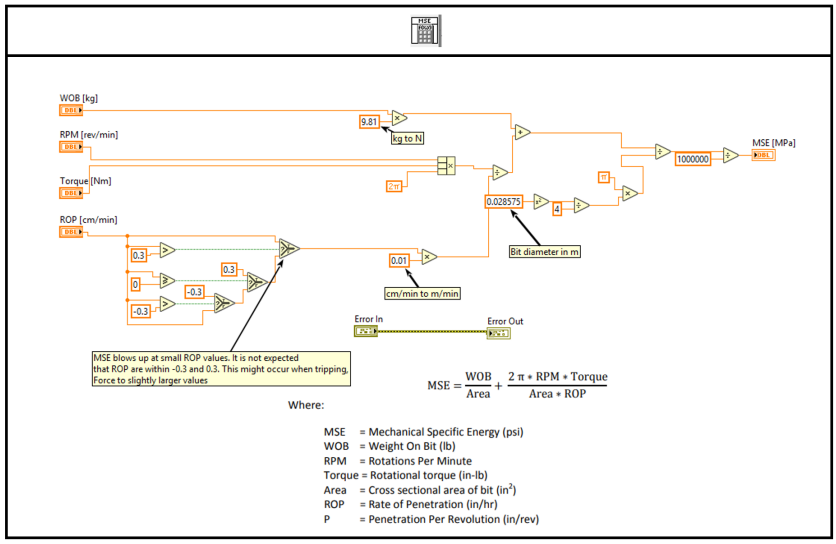

Figure B.12: SubVI calculating mechanical specific energy (MSE).

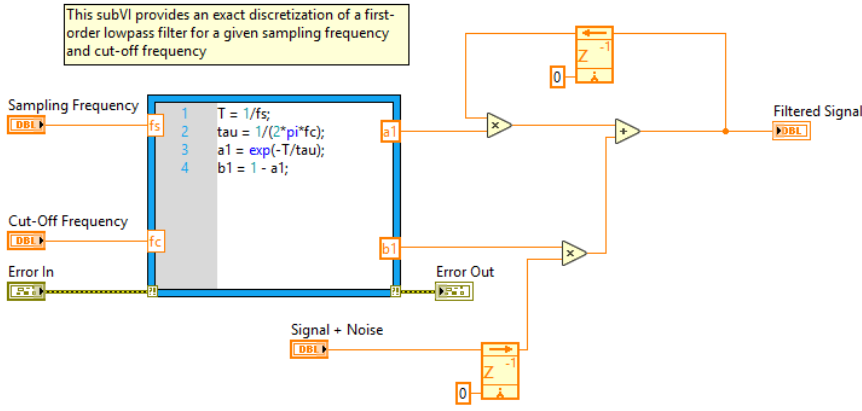

Figure B.13: First-order Butterworth lowpass filter subVI. This was implemented before realizing a third-order was needed.

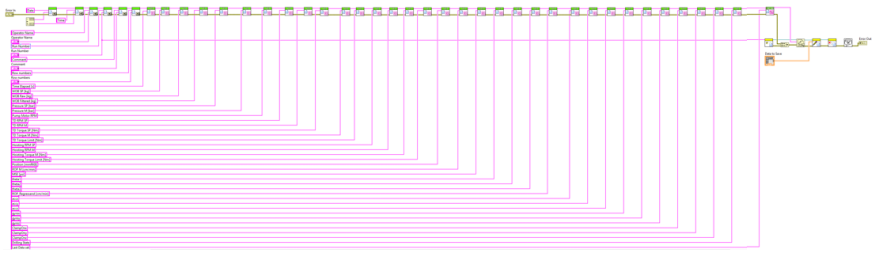

Figure B.14: File saving VI block diagram.

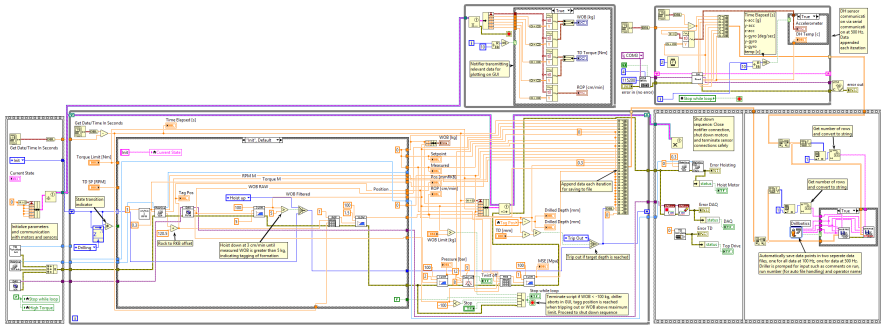

Figure B.15: Block diagram of the Autonomous VI.

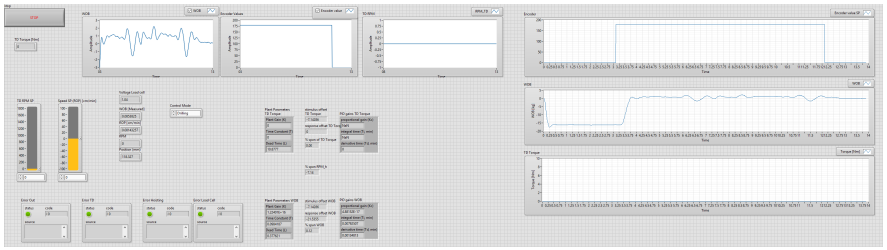

Figure B.16: Front panel of VI used for Cohen-Coon PID tuning.

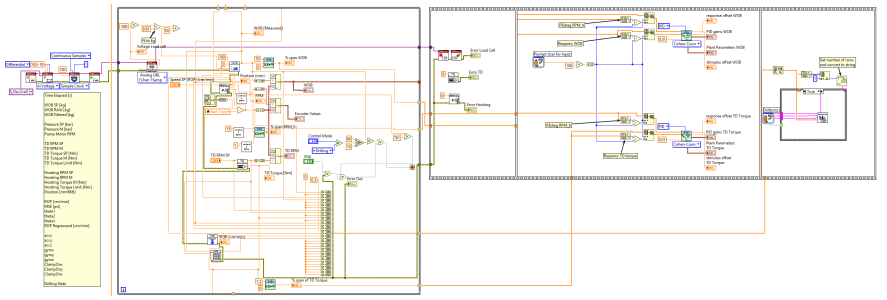

Figure B.17: Block diagram of VI used for Cohen-Coon PID tuning.

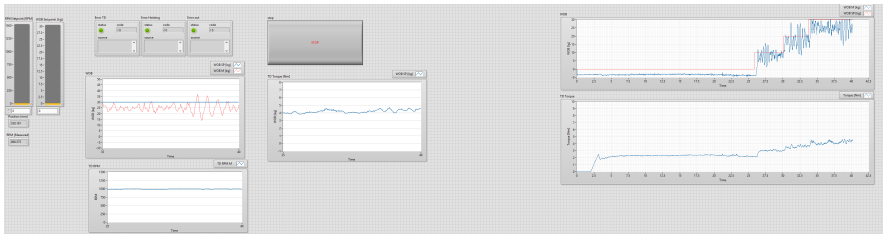

Figure B.18: Front panel of VI used for PID testing.

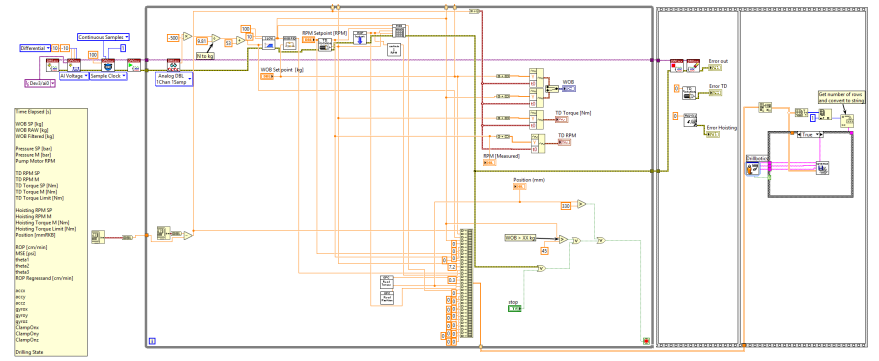

Figure B.19: Block diagram of VI used for PID testing.

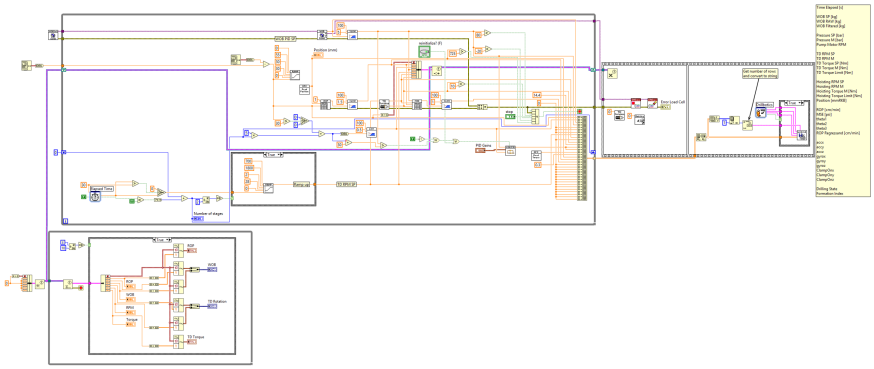

Figure B.20: Block diagram of VI used for formation response testing.