# DOWNHOLE INTELLIGENCE FOR DRILLING SYSTEMS USING SUPERVISED AND DEEP REINFORCEMENT LEARNING TECHNIQUES

A Dissertation

by

NARENDRA VISHNUMOLAKALA

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

| | |
|---|---|
| Chair of Committee, | Eduardo Gildin |
| Committee Members, | Siddarth Misra |
| | Heitor Rodrigues Lima |
| | Satish Bukkapatnam |
| Head of Department, | Timothy Jacobs |

May 2022

Major Subject: Interdisciplinary Engineering

ABSTRACT

Operational decision-making during drilling for hydrocarbons or geothermal energy is challenging due to the complex nature of the process. Many of the times, these decisions have to be taken with incomplete information at hand, either because of uncertainties or errors in the measurements, limited data transmission rates or bandwidths, inaccurate modeling during planning phase, delays in processing and analysing the information or simply due to unexpected situations encountered in the process. This is more evident in geosteering operations while drilling directionally, because a series of high-quality decisions regarding well-trajectory adjustment are required to be taken in timely manner to achieve optimal result. Lack of a systematic and transparent framework to clearly and quantitatively state measurable objectives, key underlying uncertainties, or relevance between underlying uncertainties and real-time information has led to the treatment of geosteering operations as more of an art, rather than science. Furthermore, these challenges encountered in geosteering when it is treated as an independent operation are multiplied when the operation is integrated into the broader framework of drilling operations and it becomes a daunting task for a driller on the surface, or a drilling engineer at the rig or at a remote location, to carry out the operations in an efficient manner. Automating the processes and incorporating autonomous systems into the workflow is a viable solution to this highly complex problem. In this work, machine learning techniques, in particular Reinforcement Learning has been used to develop autonomous geosteering systems and predictive analytics for wellbore cleaning and vibration mitigation problems. The problems are first solved in a modular fashion and a plan to integrate the sub-systems is proposed later.

In this work, I frame the geosteering operation as a sequential decision-making problem under uncertainty. Conventional procedures to automate the process are either model-based which require accurate modeling of the highly complex process, or not universal thus limiting applicability of the methods with freedom, or computationally expensive. The approach

taken in this study utilize reinforcement learning techniques to develop solutions that is model-free, platform-independent, and are near-real-time. A physics-based real-time engine has been used to develop a drilling simulator using which the reinforcement learning models are trained. Traditional dynamic programming solutions are not tractable because of the continuous action space for steering, possible observations within each sequence, and the computational demands of simulating updated environment. Function approximators are used in a policy gradient approach utilizing an advanced technique of Proximal Policy Optimization to train a drilling agent. Two separate solutions for steering have been developed, one for accurately tracking a given wellbore trajectory under uncertainty, and the other to self-learn the trajectory and maximize contact with target zone.

A second set of solutions pertain to improving the performance of the drilling operations. A dysfunction library has been developed using supervised machine learning techniques to identify in real-time the type of dysfunction encountered downhole. The solutions are extended beyond identification, into prediction of a particular type of dysfunction, Stickslip, ahead of time. The models were able to predict Stickslip 10 seconds and 30 seconds into the future and have been validation using field data. Reinforcement Learning has been used to optimize the hole cleaning process essentially minimizing cuttings bed height by changing surface parameters. These solutions are developed as independent sub-systems which could eventually be integrated into the reinforcement learning framework of autonomous geosteering. My main contributions in this study are 1) to formulate the drilling systems in sequential decision-making framework 2) implement reinforcement learning and other machine learning techniques, 3) develop solutions that are interpretable and practical and 4) develop a plan for an integrated optimization solution that could pave way for a fully autonomous downhole drilling system in the future.

# DEDICATION

To my parents and my wife.

# ACKNOWLEDGMENTS

There have been numerous people who have walked by my side during different phases of my life, shaping the person I am today. They have guided me, steered me towards opportunities, and showed me the brighter side of every situation making me stronger and happier with each day. I would like to thank each and every one of them. First and foremost, I would like to express my deep gratitude to Dr. Eduardo Gildin, who has always been there for me supporting me at every stage of my life for the last 8 years. I'm extremely grateful to him for his constant guidance & motivation, and for making me the person I am today. I attribute all of my success in my graduate studies and all my professional accomplishments to him.

I want to thank my committee members Dr. Bukkapatnam, Dr. Lima, and Dr. Lima for sharing their knowledge, providing excellent technical guidance, interest, and motivation. Dr. Lima, in addition to serving on the committee, helped me with teaching assistantship in his labs, has been my mentor and was always had an open door for me to talk to. I m really grateful for his kindness, care and support.

I thank everyone involved in the development of each project. My lab-mates for creating an excellent stress-free work environment. Special thanks to my friend Enrique, for his unwavering commitment, collaboration, and for always pushing me to give my best.

Lastly, a big thank you to my parents and sister for believing in me and loving me unconditionally. Special thanks to my beautiful wife whose unwavering support & faith in me held me strong  composed even in times of distress. I always looked up to her for inspiring me to be better with each day. My friends for creating a family environment, for boosting my morale, and offering great help throughout my studies at TAMU. They imbibed immense confidence in me and supported me at unaccountable instances.

CONTRIBUTORS AND FUNDING SOURCES

## Contributors

This work was supported by a thesis (or) dissertation committee consisting of Professor Dr. Eduardo Gildin, Dr. Heitor Rodrigues Lima, Dr. Siddarth Misra, of the Harold Vance Department of Petroleum Engineering, and Dr. Satish Bukkapatnam, of the Industrial Engineering Department.

The methods, and processes discussed in Chapters 4 & 5, were developed in collaboration with Teale Engineering LLC. The short-hop hardware component development and impedance modeling calculations discussed in Chapter 6 were performed by E-Spectrum Technologies. Code base provided by Dr. Guni Sharon of Computer Science Department at Texas A&M University as a part of coursework was partially used for developments in section 6.2. Data analyzed for Section 6.1 was provided by a U.S.-based service company under in-kind collaboration agreement with the Petroleum Engineering Department at Texas A&M University.

All other work conducted for the thesis (or) dissertation was completed by the student independently.

## Funding Sources

Graduate study and research expenditure were supported by a fellowship, Teaching Assistantship, and Research Assistantship from the Department of Petroleum Engineering, the Dwight Look College of Engineering in Texas A&M UniversityTexas A&M University.

This work and graduate studies were made possible by partial support from:

- Foundation CMG through the FCMG Research Chair at Texas A&M University
- E-Spectrum Technologies,and the U.S.Department of Energy, under contract DE-SC001986
- Teale Engineering LLC, and the National Science Foundation under STTR Phase 1

program grant #2108048

- The National Science Foundation (NSF) I-Corps under grant #2100640
- Teale Engineering LLC through co-op.
- Texas A&M Commercialization Office

TABLE OF CONTENTS

Page

# LIST OF FIGURES

FIGURE Page

# LIST OF TABLES

TABLE Page

# 1. INTRODUCTION

There is an ever-increasing need to develop more cost-effective access to energy, not only from conventional Oil & Gas resources but also from cleaner sources such as Hydrothermal and Enhanced Geothermal Systems (EGS). The United Nations predicted world population growth from 7.7 billion in 2019 to 9.7 billion by 2050 [11]. This, combined with growth prediction in urbanization, from 52% in 2011 to 62% in 2035 and reaching 70% worldwide by 2050 [12], is causing rising demand for energy in the coming years. In contrast, a fall in conventional crude oil resources prompts exploration and production from more complex unconventional reserves. Valued at $4.6 billion in 2018, and projected to reach $6.8 billion by 2026 [13], the geothermal energy market which is driven by the implementation of stringent government regulations related to climate change will play a key role in the shift towards reduced-carbon initiatives. Although rapid efforts are being taken to develop such renewable energy sources, these resources are not mature enough to cater to massive energy demand. Responsible development of the unconventionals that can be achieved by utilizing advanced technologies like artificial intelligence, plays a crucial role in our nation's clean energy future, offering significant economic, energy security, and environmental benefits.

## 1.1 Research Motivation

Drilling for hydrocarbon or geothermal energy is a complex process that involves monitoring of several strongly coupled dynamics under an aggressive and harsh environment that the drill bit endures downhole. In addition, constraints on bi-directional communication data transmission rate and bandwidth between the surface and downhole lead to uncertainties and inaccuracies in sensor data, introducing additional complexity. Because of such issues, it is becoming increasingly difficult to make operational decisions for drilling engineers either on surface or at a remote monitoring site. These engineers drilling for either hydrocarbons or geothermal energy are constantly looking for ways to optimize the drilling process making

it faster, safer, productive, and more predictable. Automated Drilling Systems are seen to be the way forward to address the issues [14, 15]. On one hand, Oil & Gas wells, especially in the US, are getting more tighter and on the other, Geothermal wells pose issues with high temperatures. These complexities are making process automation more necessary than ever. Decision-making based on surface data, sparse and delayed downhole data, and driller's experience is leading to significant inefficiencies and sometimes, unsafe operations [16, 17].

Although drilling for energy resources and extraction still remains a predominately manual operation where major gains in safety and optimization are yet to be fulfilled, there has been a significant advancement of drilling automation technologies over the last decade [18]. An interesting quantifiable measure to understand the technological advancement is to look at the number of research publications in this domain. More than 400 research papers per year (as of 2021) are being published within the sub-discipline of well drilling at major SPE conferences and journals [19]. An increase of 37% is seen in the publications rate from 2015 to 2020. Some of the major industry drivers that contributed to the adaption of automated systems to effectively improve current solutions are as listed below.

- **Increased well complexity and excessive Non-Productive Time (NPT)** - Multi-Directional extended reach wells are becoming more common especially in the offshore business. Ambitious projects with narrower pressure margins, complex trajectories and uncertainties must rely on robust automated systems in the future to achieve extended distances while mitigating costs, risks while reducing NPT.

- **Data Processing Capabilities** - Modern computer systems are capable of processing vast amounts of data from acquiring, storing, categorizing and interpreting it in a relatively short period. Automated systems will be able to run local optimizations based on history or model-based data that will either take control of drilling parameters or advise the driller accordingly.

- **Knowledge Transfer** - Skilled and experienced employees will eventually exit the

industry, leaving their roles to younger professionals. This loss in expertise can be reduced using autonomous systems.

- **Increased Emphasis on Safety** Safety has always been the top priority for companies where dangerous working and extreme conditions are common, such as the Oil and Gas industry. Also, strict environmental regulations must be followed to avoid serious negative consequences to the ecosystem and the health of collaborators. Automated systems will decrease the number of people working in hazardous zones, as well as improve the HSE monitoring processes currently in place.

- **Economic Drivers** - As the global oil prices become more volatile and as the wells become more complex, there is a strong need for deep technologies to enhance operational performance and achieve cost savings. The energy industry is currently undergoing a digital transformation and is rapidly adopting advanced technologies to improve productivity and safety. Energy companies are convinced that the application of deep technologies is the economical way forward.

- **Increasing Demand For Clean Energy** - Environmentalists and several governments worldwide are promoting the need for clean energy production. The effect of such a demand is driving the global geothermal power market growth due to factors such as volatile prices & limited presence of fossil fuels, cost-effectiveness & goals for 'zero-emission' drilling of geothermal energy. Advanced technologies have the potential to make 1) hydrocarbon drilling more cleaner and 2) geothermal drilling operations much cheaper.

The first oil well was drilled almost 1800 years ago and since then we have seen advances in technology in various fronts that have taken us to the moon and allow us to connect with a loved one instantly across the globe. The Wright brothers took their first flight in 1903 and it took less than 100 years to achieve autonomous flight. Yet, the drilling industry, having the largest head start, has lagged behind in achieving autonomous operations [20]. Drawing

parallels with these industries and adapting successful technologies from relevant industries like automated steering will push the limit of current state of automation forward in the drilling domain. The Level Of Automation (LOA) taxonomy proposed by Kaber et. al. [9] and related work [21, 22], is shown in Table 1.1 which can be used for comparison across different functions and different industries.

| **Levels of Automation** | **Monitoring** | **Generating** | **Selecting** | **Implementing** |
|---|---|---|---|---|
| 1. Manual Control | Human | Human | Human | Human |
| 2. Action Support | Human/Computer | Human | Human | Human/Computer |
| 3. Batch Processing | Human/Computer | Human | Human | Computer |
| 4. Shared Control | Human/Computer | Human/Computer | Human | Human/Computer |
| 5. Decision Support | Human/Computer | Human/Computer | Human | Computer |
| 6. Blended Decision Making | Human/Computer | Human/Computer | Human/Computer | Computer |
| 7. Rigid System | Human/Computer | Human/Computer | Human | Computer |
| 8. Automated Decision Making | Human/Computer | Human/Computer | Computer | Computer |
| 9. Supervisory Control | Human/Computer | Computer | Computer | Computer |
| 10. Full Automation | Computer | Computer | Computer | Computer |

Table 1.1: Levels of Automation, Effects on Performance, Situation Awareness, and Workload in a Dynamic Control Task [9]

The complexity of the drilling operation stems from the fact that the operation is a combination of several sub-systems and processes. Naturally, drilling automation is not a single system—it encompasses a hierarchical construct of automated subsystems. The lower ranks of the hierarchy are typically at a higher level of automation. In the geosteering operation for instance, the current downhole rotary-steerable system that requires only supervisory control is at LOA of 8, whereas directional control on surface is (at best) a shared control system at an LOA of 4. Some examples of drilling processes and their LOA ratings [10] are shown in Table 1.2. Furthermore, when other processes such as vibration control, ROP optimization are integrated into the steering system, the combined process will be at a much lower LOA. The hierarchical construct typifies drilling-systems automation, which is concerned with making the individual subsystems work together in creating a quality borehole. The technical challenge is moving the overall drilling-systems automation from a low LOA

(currently 2, human/computer monitoring for many operations) to a higher level to realize gains in productivity, efficiency, and safety.

| **Monitor** | **Advice** | **Control** | **Autonomous** |
|---|---|---|---|
| L2 | L3 - L4 | L5 - L7 | L8 - L10 |
| Wellsite Monitoring Systems<br>Remote Data Centers | Drilling Dynamics Diagnosis<br>Directional Drilling Advisory | Auto-Driller<br>MPD Control Systems | MWD-RSS Systems<br>LWD Formation Samplers |

Table 1.2: Examples of available drilling-automation processes, grouped by automation category and LOA [10]

## 1.2 Problem Statement

The Energy industry's most critical technological concern in the drilling optimization front is the inability to transmit large amounts of data from the downhole to process it on the surface. Monitoring Geothermal energy or Oil & Gas drilling operations in real-time, either on-site, or remotely has its limitations, which results in sub-optimal drilling operations, leading to a slow penetration rate that proved to be expensive for energy service companies. For decades, the industry has been trying to improve the data transmission rate between downhole and surface. Solutions such as wired-pipe telemetry emerged as a result [25]. Although technologically advanced, these processes have seldom been economically feasible. Alternatively, with an intelligent system downhole near bit, the need for transmission of high volumes and high frequency data could be eliminated. Sensors located downhole acquire data at much higher rates than what is transmitted to the surface using the current mud-pulse telemetry communication system [26, 27, 28]. The high frequency data is sometimes logged on to the Bottom Hole Assembly (BHA) board which will typically be used for post-processing and for use in planning of future wells. Usage of this data to improve efficiency of operations while drilling is constrained due to the slow data transmission rates and bandwidth. With appropriate hardware in the BHA, the continuous stream of high-

frequency and real-time data can be utilized to develop in-situ decision-making capabilities using artificial intelligence methods like Reinforcement Learning (RL).

In specific, the problem of automated geosteering and integrating other interconnected drilling processes is tackled in this research. Geosteering is the real-time adjustment of well trajectory while drilling to maximize effective contact with the target zone. Geosteering currently is predominantly a manual process that requires geologists to manually update the geosteering model by constantly changing both the bed dip and length of the segments of the model to obtain the best match between lateral and offset well logs [29]. This process can be challenging when there are thin reservoir sections and precise well placement is required. Furthermore, interactive wellbore navigation helps move beyond reactive geosteering which, as the name indicates, is reacting in an appropriate manner to a geological event that has already been encountered or traversed by the wellbore [30]. A downhole automated geosteering system will enable real-time decision-making near-bit resulting in more efficient, quicker, and high quality wellbores. Parallels can drawn from successful applications from other industries, specifically from that of autonomous vehicles which can have similarities to the geosteering setting. This research shows the modeling of wellbore navigation operation as an autonomous self-steering problem. Given the complexity of the operation due to several factors like large number of process variables involved, data measurement inaccuracies and uncertainties, formation heterogeneities from well-to-well among other factors, it becomes increasingly difficult to accurately model the system. The study explores the use of RL techniques to develop model-free control of the system and train a self-learning drilling agent. As has been mentioned before, drilling, especially directional drilling is a complex process that involves several subsystems interconnected and impacting the overall success of the operation. Two such important silos are drilling performance optimization in terms of dysfunction mitigation and hole cleaning in terms of efficient transport of drill cuttings. While modeling an integrated system is not in the scope of this research, individual models have been developed and a plan to integrate the sub-systems into the unified autonomous

framework has been proposed.

In summary, **the research aims at developing an intelligent downhole system using model-free techniques of reinforcement learning, that can self-steer the drill bit into the target zone while ensuring drilling operational efficiency**.

### 1.2.1 Research Objectives

The above described problem is tackled in a multi-step approach breaking it down into sub-problems. Firstly, the steering problem is treated as a wellbore trajectory tracking problem with the objective of traversing a given wellpath with the least amount of deviation resulting in a semi-autonomous system. The system is considered as semi-autonomous for the reason that well plans developed by drilling engineers and geologists before drilling a well are provided as inputs to the system. A fully autonomous geosteering system learns to find an optimal wellbore trajectory and steers against the uncertainties and deviations encountered while drilling. This is noted as the second step where a drilling agent learns to self-steer a well. Next, optimization of wellbore cleaning based on drilling operational parameters is explored. Finally, dysfunction mitigation techniques are developed based on surface and downhole drilling data. While the third and fourth steps are developed as individual modules, they could eventually be integrated into the steering models.

Some of the key objectives of this study are listed below. This research aims:

- to formulate the sequential decision-making problem of geosteering as a Markov Decision Process (MDP). The MDP model is setup according to the objectives and system variables pertinent to the application. Consequently multiple MDP formulations are investigated for this multi-step approach as described above.

- to develop simulated environments that facilitate learning using advanced artificial intelligence techniques. A computationally efficient work-flow has been developed accounting for uncertainty in data measurements, geology and reservoir rock properties.

- to evaluate and apply deep reinforcement learning for geosteering autonomously and

semi-autonomously. The performance of the reinforcement learning algorithm is improved by utilizing sampling techniques and function approximators.

- to provide a real-time computationally efficient solution to wellbore trajectory tracking application and analyze benefits over traditional techniques

- to interpret policies developed using the reinforcement learning algorithm and suggest improvements in the algorithm. This includes an interpretation of the impact of reward parameters and evaluating the impact of state formulation on the final convergence of the algorithms.

- to apply the developed methodology on several case studies of varying complexity and demonstrate the efficacy and versatility of the reinforcement learning algorithm.

- to provide data-driven solutions to identifying, predicting dysfunctions and optimizing hole cleaning operation to later integrate into steering models. Validation of models to identify and predict dysfunctions is performed on real-world drilling data

- to demonstrate the strategies proposed for implementation of a downhole intelligent system are superior to existing technologies that rely on constrained data transmission between downhole and surface

### 1.2.2 Research Contributions

The knowledge acquired from this study will be beneficial for industry as well as researchers to broaden their concept related to geosteering, drilling performance optimization, and for eventually implementing fully-autonomous systems downhole. Some of the main contributions of this work are listed below.

- Proved that the use of high-frequency downhole measurements can significantly increase the accuracy of dysfunction identification while drilling.

- Developed a data-driven method using recurrent neural networks that is more robust than the existing methods, to predict Stickslip torsional vibration events in drillstring up to 30 seconds ahead of time using only downhole raw drilling dynamics data.

- Developed a simulation framework that balances the trade-off between two extremes of commercial-grade high-fidelity simulators and the inaccurate real-time platforms. The high-fidelity simulators such as finite-element-method-based simulators which are computationally demanding are found to not be a good fit for an interactive learning approach and can be replaced by the proposed physics-based near-real-time simulator.

- Translated the self-correcting methods of reinforcement learning from autonomous vehicles to geosteering of hydrocarbon wells and showed the feasibility of automated pro-active steering operations with minimum human intervention in the process.

- Presented an alternative and a more practical approach to modeling the complex behavior of bit-walk through the use of model-free reinforcement learning techniques.

- Combined two objectives of optimal well placement and improved drilling performance, which have been predominantly treated as independent drilling sub-systems, and demonstrated through simulation case studies that the coupled system will be able to achieve better overall operational efficiency.

### 1.3 Outline of Dissertation

In Chapter 2, a comprehensive review of literature surrounding the current state of directional drilling operations, specifically in the domains of geosteering and performance optimization, is provided. Limitations with respect to telemetry are discussed next. Benefits and drawbacks of existing technologies including model-based control are discussed. Case studies from literature are studied and presented highlighting real-world applications. Deficiencies in the manual or semi-autonomous operations on the field are discussed.

Chapter 3 presents the mathematical foundations of reinforcement learning and the terminology associated with the algorithms. MDP formulation and different elements of the RL framework are explained. Bellman expectation and optimality equations for state and action value functions which are central to RL are shown. Two broad categories, off-policy and on-policy algorithms with examples of popular SARSA and Q-Learning algorithms are highlighted for model-free methods. This is followed by description of an advanced techniques called PPO, that will be heavily used in the subsequent chapters. Simulation environments used for RL implementation and training are presented. Specifically, functionality of three simulators - Unity, Chrono, and OpenLab are explained. Finally, the steering and associated problems are formulated in a reinforcement learning framework and the benefits of addressing the problem in this formulation are discussed.

In Chapter 4, the first step of geosteering modeling that is trajectory tracking problem is discussed in detail. The chapter starts with an overview of the problem and discussion of how the path tracking problem is being approached in other industries. After understanding the objectives and constraints of the problem, an MDP formulation is presented with details on state space, action space, rewards, and stochasticity in the simulator. Several approaches for reward shaping are explained in detail. Implementation is done on ML-Agents platform within the custom-built Unity simulation environment. Application of RL agorithms for training is described in detail. The developed reinforcement learning policy is visualized and metrics demonstrating the convergence of the algorithm are presented. Assessment of training performance is carried through various metrics like cumulative reward, episode length, entropy, value loss, to name a few.

Chapter 5 presents the models developed for the autonomous self-steering systems. The chapter starts with an overview of the problem and discussion of how the problem upgrades from the semi-autonomous system discussed in the prior chapter. A major component of the autonomous downhole system is the availability of information regarding the formation type current being drilled at a given depth. Supervised machine learning classification methods

used to identify rock type while drilling are presented. Results of the classification models and validation on lab-scale drilling test data are shown. Similar to Chapter 4, but with updated goals and constraints, MDP formulation is shown for the autonomous self-steering drilling system. Reward shaping approaches are explained in detail. Learning with an existing baseline plan and reward schema associated with it are presented next. Implementation is done on ML-Agents platform within the custom-built Unity simulation environment. The PPO algorithm has been presented and the learning behaviour from the implementations are discussed. Assumptions with the model setup and consequently, the limitation in the policy behaviour are explained. Performance evaluation of the learning process is described at the end.

Chapter 6 demonstrates the implementation of two drilling sub-systems - Vibration mitigation and Hole cleaning operations. First, the vibration mitigation issue is briefly explained along with advancements like short-hop EM telemetry. A dysfunction detection library is developed using machine learning classification methods. Results of these models in identifying Stickslip are presented. Recurrent neural networks are shown to solve dysfunction prediction problem. Implementation of the classification algorithms and the RNN-LSTM algorithms are briefly explained. Results of predictions for 10 seconds and 30 seconds ahead of time are presented. Later, an MDP formulation for how the vibrations can be controlled using drilling operational parameters like Weight-on-Bit and rotational speed, is presented. The second aspect of drilling performance optimization explored in this study is Hole cleaning. An overview of the process and existing methodologies are briefly discussed. MDP model setup and learning environment using NORCE OpenLab simulator are presented. RL implementation, training process, and results of the learned policy are shown. An RL framework to integrate the drilling performance models into the steering models is presented.

In Chapter 7, the final conclusions and directions for future research on the development of downhole intelligent systems for autonomous geosteering and other related problems have been discussed. Three main recommendations - integration of steering models with two of

the drilling subsystems presented in the prior chapters, field validation of RL steering models, extension of the models into 3-D domain are presented.

# 2. LITERATURE REVIEW: AUTOMATED GEOSTEERING & RELATED PROBLEMS

A comprehensive review of literature surrounding the current state of directional drilling operations, specifically in the domains of geosteering and performance optimization, is provided. Limitations with respect to telemetry are discussed next. Benefits and drawbacks of existing technologies including model-based control are discussed. Case studies from literature are studied and presented highlighting real-world applications. Current advancements in automated geosteering and move towards AI-based techniques are discussed. Finally, a summary of the reviewed methodologies, highlighting their gaps, is presented.

## 2.1 Overview of Directional Drilling & Geosteering

Directional Drilling refers to all the activities that pertain to designing and drilling of a well in a controlled deviated manner to reach a target zone. In other words, the objective of directional drilling is to connect the surface location to hydrocarbons (high temperature zones in the case of geothermal drilling) not located directly below it [1, 31]. There are several applications of directional drilling, some of which are listed below [32, 33].

- **Inaccessible Locations** - To drill into locations directly beneath natural or man-made obstructions where there is risk to the environment or when permissions are not available.
- **Avoid Geological Problems** - To reach the producing formations under a salt dome as drilling vertically through a salt dome can cause drilling problems like washouts, lost circulation and corrosion or to avoid drilling vertically through a steeply dipping fault plane that might cause slippage
- **Optimal Well Placement** - To place the well in the sweet spot i.e. the best producing zones of the reservoir. Some parts of the reservoir may be characterized with

| | | |
|---|---|---|
| Surface Location | Quadrant | Hold Angle |
| Target Location | Polar Coordinate | Start of Drop |
| Kickoff Point | Rectangular Coordinate | True Vertical Depth |
| Buildup Rate | Vertical Section | Target Displacement |
| Turn Rate | Tangent Section | Azimuth |
| Measured Depth | Well Inclination | End of Buildup |
| Horizontal Displacement | Drop Off Rate | End of Drop |

Table 2.1: Factors that impact planning of drilling an Oil & Gas Well

heterogeneities and sedimentological variations not favorable for production

- **Horizontal drilling** - To accurately place the wellbore in a thin reservoir i.e. increased penetration of the producing formation. To increase drainage area, prevent gas or water coning problems, increase efficiency of enhanced oil recovery processes

- **Multi-lateral Drilling** To drill wells with multiple branches or laterals that can target widely spaced reservoir compartments. This is especially critical for offshore development where having individual well platforms is impractical.

- **Sidetracking** - Either to re-drill or re-completion an existing well for exploring potential new producing zones, or because of obstructions or issues like fishing.

### 2.1.1 Well Planning

A well that is carefully planned before spud can lead to significant cost savings not only in drilling but also in completions. Several factors affect the trajectory of the wellbore from formation characteristics, geology of the target zone to availability of downhole equipment for the application [34, 35, 36]. Therefore, there are many components that need consideration when designing a successful well plan. Table 2.1 shows the list of different components of a well plan, some of which are discussed in this section [33].

Fig. 2.1 shows the parameters used to describe location of the wellbore [1]. It is assumed that Point A is the surface location and a straight line connecting Point A and Point B is

the wellbore for the sake of simplicity. The distance measured along the wellbore (AB) is called *Measured Depth (MD)*. The vertical distance along z-axis from Point A to Point B is called *True Vertical Depth (TVD)*. The plane along gravity (z-axis) is called inclination plane and the one that is perpendicular to it and along the true north is called direction plane. The angle made by the wellbore with the z-axis is called *inclination ($\phi$)*. The angle between the geographic north and the projection of the wellbore onto a horizontal plane is called as *azimuth ($\theta$)*.

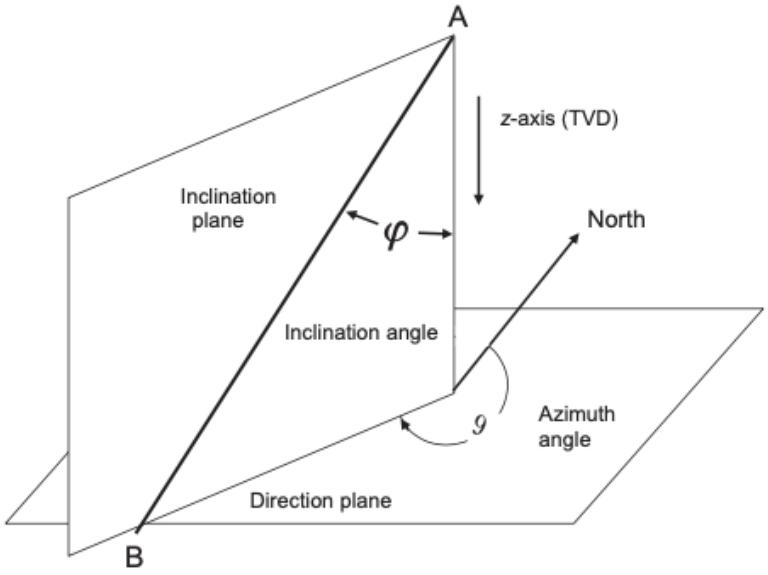


Figure 2.1: Parameters describing geometry of a wellbore [1] - Inclination plane along gravity axis; Directional plane perpendicular to inclination plane; Inclination Angle made by the wellbore with the z-axis; Azimuth angle between the geographic north and the projection of the wellbore

Fig. 2.2 shows a simple two-dimensional well profile [37, 38]. RKB represents Rotary Kelly Bushing or in generalised sense, the surface location. KOP is the point in the well bore where the well deviates from the vertical. Build Section continues from the KOP until EOB (End of Build) where the maximum planned inclination is achieved. The angle is maintained through out the Hold section until End of Hold (EOH). Build up rate (BUR) is the rate at

which the inclination is built measured as degrees of angle change in a given MD section. The angle is maintained through out the Hold section until End of Hold (EOH). If the inclination needs to be dropped, it is done at a drop rate that is similar to BUR but in the -ve direction, along the length of drop section until End of Drop (EOD). Another important parameter while drilling deviated wells is Dogleg Severity (DLS). It indicates the "crookedness" in the wellbore and is measured as change in inclination of the well bore measured per 100 ft. of the course length.

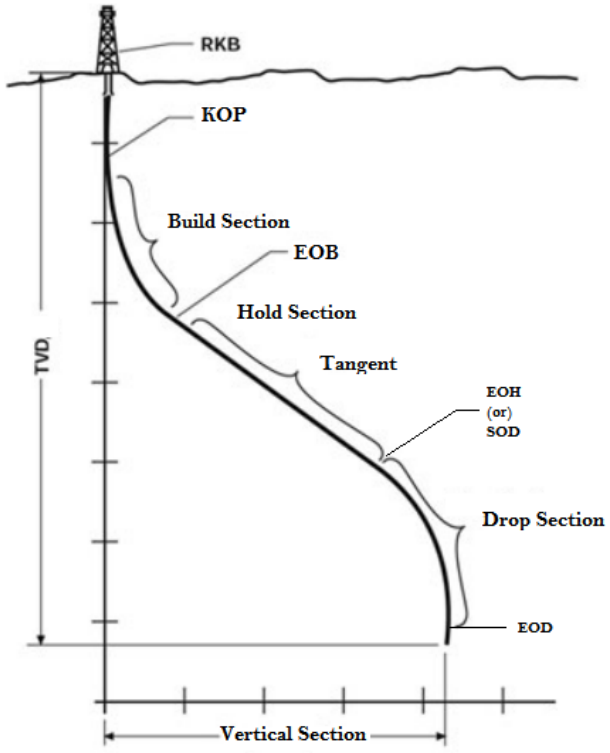


Figure 2.2: Schematic diagram of a 2-dimensional wellbore trajectory [2] showing sections of the wellpath - Build, Hold, Tangent, and Drop sections

### 2.1.2 Measurement While Drilling

The information about the trajectory of the wellbore comes from a combination of measurements - Measured Depth (MD), Inclination, and Azimuth. MD is obtained from surface

measurements, typically from pipe tally and the other two measurements are obtained from downhole through surveys. Measurement While Drilling (MWD) is a technique of directional drilling used to obtain data from bottom of the hole. Surveys are taken with the MWD tools to get the temperature and orientation of the drill bit at the survey point. With the advancement of technology, MWD is now capable of accurately measuring and relaying continuous orientation information to the surface in addition to the survey measurements. Method of transmission (telemetry) of data from downhole to surface [39] is discussed in detail later in the chapter.

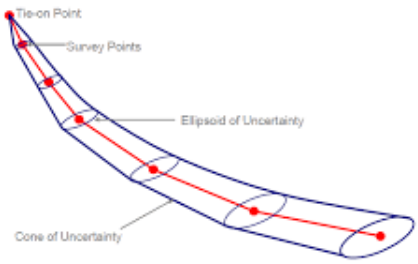


Figure 2.3: Ellipse of Uncertainty in 3-D wellbore positioning constructed at the rig surface using MWD data obtained at each survey point (indicated by red dots) [3]

*Ellipsoid of Uncertainty*

Using the MWD Survey information, calculations are done at the surface to reconstruct the 3-D trajectory of the wellbore. But due to various factors as listed below, pertaining to measurements or calculations [40], there comes certain uncertainty in the 3D wellbore position represented by ellipse of uncertainty [3] as shown in Fig. 2.3.

### 2.1.3 Geosteering

Geosteering is a technique of directional drilling which pertains to the acquisition and interpretation of geological, engineering and directional data, with the goal of maximizing the exposure to a target zone [41, 42]. In a traditional setting, sometimes referred to as geometric

steering, trajectories of the wells pre-designed to a certain target zone are controlled by MWD survey measurements. However, as mentioned in the previous section, survey measurements are not very accurate. In addition, the frequency of these surveys might be too small in many cases. This is especially true in the case of thin reservoirs where precise placement of the wellbore is required. This is where the Logging While Drilling (LWD) comes into play and the wellbore steering decisions are taken based on the real time (RT) LWD response of the geologic formations. The method is known as geologic steering or geosteering. In other words, it is defined as the interactive planned navigation of the wellbore based on the geologic data. A typical geosteering project involves several steps [43], some of which are outlined below.

1. evaluate geological, offset well, field production data
2. design well trajectory profile and establish tolerances for wellbore placement
3. spud, drill vertically to KOP and begin directional drilling
4. establish geological correlations and targeting control
5. adjust trajectory profile as needed in build section
6. land well at deviated entry point
7. evaluate geological structure and location of anticipated anomalies (faults, pinch-outs, lateral channel changes, etc)
8. drill ahead in horizontal section while steering
9. initiate remedial actions upon an unexpected event (geological surprise) to determine cause, take evasive action as necessary, or
10. evaluate sidetrack decision
11. condition hole for completion

## 2.2 Limitations on Telemetry Rate and Bandwidth

Measurements made downhole by the MWD systems are transmitted back to the surface through a communication protocol commonly referred to as telemetry system [39]. This signal transmission although is a key link to use MWD data for efficient operations, it has for long been a major bottleneck in the drilling operations. There are several types of telemetry - both wired and wireless, and each have its advantages and drawbacks [39, 44, 45, 46, 47].

*Mud Pulse Telemetry*

The most widely used method of transmission is mud pulse telemetry where data from MWD systems is encoded into pressure pulses and transmitted to the surface with drilling mud as the medium [48, 49]. The transmitted signal is received and decoded on the surface using pressure transducers. Positive pulse, negative pulse, and continuous pulse are three common modes of mud pulse telemetry. Although reliable and cost-efficient, MPT suffers from low bandwidth issues limiting data rates to very low. Typical data transmission rates are 0.5-5 bits/second.

*Electromagnetic Telemetry*

(EM) telemetry encodes the data from the BHA onto a carrier signal that is transmitted by an EM wave transmitted through the earth [50]. This signal is received and decoded at the surface through ground geophones. Although reliable, cost-effective, and independent of mud equipment, EM waves are short range due to high signal attenuation.

*Wired Pipe*

Wired drill pipe (WDP) is a wired telemetry system that where a conductor is inserted into the drill pipe that connects the surface system to the BHA [51, 52]. While it has benefits with regards to data transmission rates, WDP is not cost-effective and is a single-point failure design [53].

*Comparison of Telemetry Methods*

There are other less common methods of telemetry in the industry. A comparison of the various telemetry methods [39] is done in Table 2.2.

| **Telemetry Method** | **Transmission Depth (m)** | **Transmission Rate (b/s)** | **Reliability** | **Cost** |
|---|---|---|---|---|
| Mud Pulse | >6000 | 1 - 12 | Good | High |
| Electromagnetic | 600-6000 | 1 - 12 | Normal | High |
| Sound wave | 1000-4000 | 100 | Normal | Low |
| Wired Pipe | >6000 | 1 - 2 M | Normal | Very High |

Table 2.2: Comparison of commonly used Telemetry methods showing Mud pulse limitations on slow transmission rate and Wired Pipe telemetry limitations on cost.

## 2.3 Well Planning and Trajectory Optimization

Modern well placement and real-time well navigation methods are enabled by availability of LWD tools like gamma, electrical resistivity, density, neutron-porosity, acoustic velocities, and magnetic resonance among others, in addition to MWD measurements [29, 54, 55]. While geosteering, engineers are constantly faced with the challenge of correctly defining the wellbore position with respect to the target reservoir and other geologic markers from an often incomplete set of data [56]. The uncertainty in geosteering and interpretation of horizontal wells and the necessity for constraints and geometric models are explained by Zhou [57]. Some of the uncertainties can be reduced by the appropriate selection of additional Logging-While-Drilling sensors [58]. When advanced LWD tools, such as azimuthal density and azimuthal resistivity tools, are used in real time, they provide additional data that can be incorporated into the geosteering software [59, 60, 61]. Dip calculations from density tools made within the geosteering software can be considered when altering the geological model to change the formation dip. Azimuthal resistivity logs can either be artificially modeled and compared to the real-time data to identify the distance the wellbore is placed away

from formation boundaries or, using inversion calculations, the distance to these boundaries can be calculated and plotted against the geological model [62]. Automatic inversion of deep-directional-resistivity measurements for well placement and reservoir description are discussed by Dupuis et.al. [58]. Improvement in accuracy of well placement was achieved through enhancing the quality of MWD surveys in real-time [63], Continuous Inclination Measurement From a Near Bit Inclination MWD Sensor [64] and through another technique of applying geomagnetic corrections [65].

Directional drilling in its basic form is following a pre-determined well plan. However, uncertainities arise while drilling with the targeted geological formation not being exactly at the assumed depth, due to unknown lateral variations and other uncertainties in geologic structure and stratigraphy, in addition to pre-drill geological uncertainties [66]. Uncertainties of the actual stratigraphy in the bore hole are compounded by errors in the MWD-surveyed wellbore trajectory. In the order of magnitude, positional uncertainty arise from errors in Inclination & Azimuth measurements [67], Survey Position Relative to Doglegs and Slides [68], Rigsite Survey Procedure, Pipe Stretch and Thermal Expansion [69], and Survey Spacing [70], to name a few. Factors such as these and the general operational challenges encountered while geosteering, from a drilling engineer's perspective is presented by Noynaert et. al. [67].

## 2.4 Model-Based Control

The key operation of geosteering involves measuring the gamma ray intensity ("gamma") in the subject well being drilled and comparing with the gamma ray intensity log from a nearby offset well with known stratigraphy ("type log"). The stratigraphic depth of the subject well is matched to that of the offset well. The matching process may include compression and expansion of certain sections of the two wells leading to an inherent ambiguity in the data which in effect makes the results of this process highly dependent on the experience and expertise of the individuals making the interpretation. In order to make geosteering more transparent, repeatable and robust, the process of identifying and ranking the possible interpretations needs to be automated. This conventional stretch and match process was first

automated by Arbus [71] where the gamma logs are split into blocks and a shortest path algorithm was used to find the sequence of stratigraphy blocks that provides the best overall gamma match for the entire section of the wellbore. This method does not account for faults. Winkler [73] posed geosteering as a probabilistic inverse problem where Bayesian Inference was used to find spatial position of the wellbore by inverting fault offsets and dips of the surrounding geological strata [74]. Automatic inversion of Deep-directional-resistivity (DDR) logging-while-drilling (LWD) measurements was implemented by Dupuis et.al. [75] which was used for interpreting the subsurface and integrating the information into geo-models.

Downton [76] performed systems modeling and design of Automated-Directional-Drilling Systems using an analytic approach. The response of lateral borehole propagation of directional drilling is derived as a simple, first-order delay differential equation. The basis for the delay equation lies in employing Laplace transforms. The method showed how a directional drilling system evolves wellbore which in turn evolves the BHA [77]. The transfer-function response of a drilling system's directional tendency is constructed owing to the influence of gravity, weight on bit (WOB), and active steering actuator influences. The analytical approach gave simple algebraic forms for a tool's dogleg capability and led to simple graphical techniques for determining how the borehole propagates [78]. A fully automated geosteering algorithm has been developed [79] that includes advanced LWD filtering, fault detection, correlation, tracking of multiple interpretations with associated probabilities and visual aids of stratigraphic misfit heatmaps. The wellbore propagation model in combination with two other models - one being a set of cost functions that aims to quantitatively represent the actual value of the well using various metrics, and the other a Genetic Algorithm (GA) solver, a slide drilling guidance systems has been developed to optimize the directional well path. In a different work by Sergey et.al. [79], quantification of uncertainties in the geosteering workflows has been attempted with model realizations using an ensemble Kalman filter. Pitcher et.al. [59] discusse the advances in geosteering technologies from simple to complex solutions using examples of field case studies.

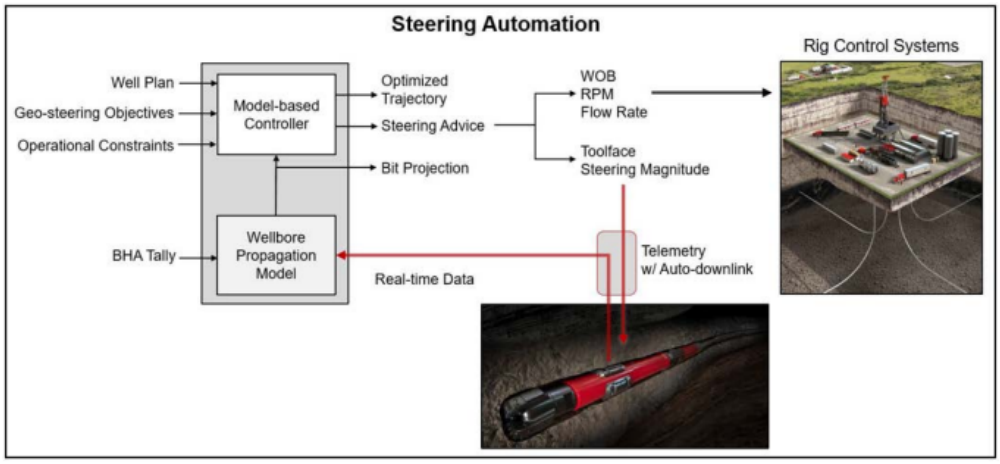


Figure 2.4: Overview of Steering automation and control process using Model-based Control system [4]

A model-based control approach for automated steering for both long laterals and 3-D complex wells has been proposed by Demirer et.al. [4] in a recent study. A 2-D wellbore propogation model is first presented to explain the inclination dynamics of the wellbore and the model is then expanded into the 3-D domain which included azimuthal dynamics also. Such model-based directional drilling controllers have been developed for both mud motors and rotary steerable systems. A model with kinematic equations was developed [80] with inclination and azimuth as states and model predictive control algorithm to hold a constant attitude [81, 82]. Panchal et. al. [83] designed a controller using pole-placement method that led to the development of inclination-hold and azimuth-hold systems.

## 2.5 Automated Geosteering

Process automation in drilling was adopted in some of the early applications and was shown to be technologically feasible and that it improves efficiency. Closed-loop solutions are seen to be crucial in increasing safety, efficiency, and reducing costs of drilling [84, 85]. Attempts have been made to integrated downhole components to that of surface control systems in the closed-loop architecture. Directional drilling automation has been attempted in solutions involving both mud motors and rotary steerable systems (RSS). Decision-making

recommendations of when to slide or rotate [79] with a positive displacement motor to achieve a trajectory with as little tortuosity as possible have been proposed [86, 87]. A recent advancement in RSS technologies is to utilize downhole sensors, high-speed processors and survey packages to provide the backbone for an autonomous downhole navigation system. This led to development of automated features like close-loop attitude-hold similar to the ones mentioned above [83]. In addition to automating a significant part of the labor-intensive directional drilling workload, this technology facilitates delivering consistent and reliable drilling performance across multiple wells with the standardization of steering control practices along vertical, tangent and lateral sections.

Automation of the directional drilling operations, particularly automated decision-making, have led to improvement in the consistency the drilling performance and reduction of operation time of the drilling process [4]. However, implementation of the automated advisor to mud-motor systems comes with complexities and challenges [76, 88]. Few more research studies on automated steering advisory systems include the proposal of the near-bit sensor surveys to enhance the automated drilling features [77], applications an automated directional system in complicated environments [89] to name a few. Other studies regarding high-speed data transmission [90], and the level of human interaction with a partially automated steering system [91, 92] have been conducted to complement a more efficient and accurate automated steering system.

### 2.5.1 Move towards AI-based techniques

The Oil & Gas industry has successfully adapted data-driven solutions for many applications across the industry. Nguyen T. et al. [93] and Mohammadpoor et.al. [94] implemented predictive analytics using Big Data technologies. A relevant topic in geosteering where the data-driven techniques and machine learning have provided solutions is LWD data interpretation. Klyuchnikov et. al. [95] proposed use of Random Forests to predict lithotype from LWD measurements and approach for integrating the LWD interpretations into geological models. Timonov [96] showed the use of XGBoost methods improved the prediction

accuracies compared to random forest methods. Enrique showed a method to identify rock formation type using downhole data collected from a lab-scale drilling rig. Shahriari et.al. [97] implemented deep neural networks and Kristoffersen [98] used Artificial Neural Networks to optimize the horizontal well placement. Bilinchuk A.V. et al. [99] considered the ability of the system to more accurately and quickly determine the target interval for borehole placement as the automation of the geosteering process. An intelligent Geo-Navigation system concept has been introduced by Alenezi et.al. [100] in his study on optimizing automated geosteering using Machine Learning. Differentiating between automated and autonomous systems, Cayeux [101] proposed an autonomous decision-making system while drilling that relies on optimization of time taken to reach the target. The optimization problem is solved using Markov Decision Process methods. A recent study by showed the use of reinforcement learning methods to train an automated directional drilling agent in a simulated environment. The agent learned to read survey data and trajectory information and efficiently switch between sliding and rotating while drilling.

## 2.6 Deficiencies in existing methodologies

Some of the highlights from the above discussed technologies are presented in Table 2.3. below, with handpicked publications relevant to those technologies. Some of the gaps in the existing methods are also identified and shown in the table.

| Description | References | Identified Gaps in the technology |
|---|---|---|
| Data transmission rates are slow and bandwidth is constrained. Mud-pulse is 2-10 bps. Wired-pipe is not always feasible. | Introduction the mud pulse transmission methods of MWD system, Yang, Q. et. al. (2010) [39] | Methods utilizing real-time (or high frequency) measurements is limited. Current literature is focused on surface data due to limitations in telemetry |
| Surface-based systems rely predominantly on survey measurements | Advancement and economic benefit of geosteering and well-placement technology, Bittar (2015) [60], Deverse et. al. (2015) [65] | Focus on improving MWD survey quality and continuous inclination measurements. Uncertainties and measurement errors in surveys not addressed |
| Emphasis is placed on correction on surface more than downhole optimization | Directional Drilling Tests in Concrete Blocks Yield Precise Measurements of Borehole Position and Quality, Stockhausen et. al. (2016)[70] | Additional processes cause further time-delays, loses effectiveness |
| Model Based Systems provides 'real-time' steering advice to drillers. Development of Integrated Steering Advisory for Mud Motors | An Integrated Directional Drilling Simulator with Steering Advisor and Self-Learning Algorithm, Yang, Liu. et. al. (2022) [102] | Rotate/Slide sequence for Mud Motors and may not be applicable for RSS applications; Geosteering is complex and uncertainties like bitwalk are hard to model |
| Self-Learning physics-based model Advisory for RSS. Approach is in a setpoint control framework. | Steering advisory system for rotary steerable systems, Zalluhoglu et. al. (2019) [103] | Limited in scope such as Inclination hold; Digital Twin continuously calibrated while drilling but model may not fully account for biases, uncertainties |
| Multi-agent decision-making system using Deep Reinforcement Learning. Self-learning Steering Advisory system that accounts for uncertainties | DDNet: A Multi-Agent Decision Making and Evaluation in Drilling with Looking-Ahead Simulation, Yingwei et. al. (2022) [8] | Steering decisions for Mud Motors and may not be applicable for RSS applications. Decisions are generated at every survey point may be termed as Reactive geosteering; Real-time navigation/steering methods do not exist. |

Table 2.3: Summary of Review of Existing Technologies highlighting gaps in current technology

# 3. REINFORCEMENT LEARNING & STOCHASTIC SIMULATION ENVIRONMENT

This chapter introduces the concepts of Reinforcement Learning (RL) and describe how RL is used in developing models for complex problems. The mathematical tool used to represent the problems - Markov Decision Process (MDP), is first described. This is followed by a high-level explanation of various components of a standard RL model. Two broad categories of algorithms - off-policy and on-policy, are presented. Popular RL techniques like Q-learning, their function approximators, the exploration-exploitation trade-off problem are discussed briefly. Towards the end, the implementation of the RL models in a discrete and continuous action space are described. The chapter concludes with the formulation of the geosteering problem in the framework of reinforcement learning. It is this formulation and its variations that are used in the remainder of the dissertation.

## 3.1 Reinforcement Learning Concepts

*"Give a man a fish, and he will eat for a day. Teach a man how to fish, he will eat every day, and give the man the taste of fish, he will figure out how to fish, even if the conditions change!"* - Michael Littman, a computer science professor at Brown University, talked about the basic idea around the reward hypothesis underlying reinforcement learning. The first scenario of giving the man a fish is analogous to traditional rules-based programming. In contrast, the second can be formulated as supervised Machine Learning (ML) solutions to a problem. However, in situations where the situational environment frequently changes, it is of utmost importance to quickly make correct decisions, such as in rocket guidance, autonomous driving systems, or downhole Oil & Gas drilling environment. The need for real-time online learning is not only desired but also essential. A technique under the umbrella of Artificial Intelligence (AI), called Reinforcement Learning (RL), has gained exponential popularity in solving sequential decision making problems.

**Evolution of RL**

The modern field of Reinforcement Learning evovled from the combination of three concepts. The first one concerns learning by trial and error which has roots in the psychology of animal learning. The second concerns the problem of optimal control originally developed by Bellman in 1950s [104] and its solution using value functions and dynamic programming. The final concept involves use of temporal difference methods. Even though dynamic programming provides a good framework for determining the optimal solution to the problem at hand, the determination of the solutions of the sub-problems becomes progressively difficult with increasing state and decision space dimensionality. This phenomenon is referred to as Bellman's curse of dimensionality. This calls for the development and utilization of efficient methods to approximate the optimal solution to a multi-stage decision problem under uncertainty. Several techniques have been developed to approximately solve dynamic programming problems and reinforcement learning is one of the most popular of these approximate methods.

Reinforcement Learning has been in practice with variations in the methodology in several applications under different terminology as shown in Fig. 3.1. Machine Learning can be broadly classified into three types of learning - supervised, unsupervised, and reinforcement learning. Supervised learning is learning from a training set of labeled examples provided by a knowledgeable external supervisor. The goal of supervised learning is for the system to extrapolate, or generalize, its responses so that it acts correctly in situations not present in the training set. While this can be a valuable approach in many applications, it is not adequate when the environment is interactive. Unsupervised learning, on the other hand is about finding patterns in unlabeled data. RL is different from unsupervised learning in that it does not rely on examples of correct behavior. In the RL framework, actions are taken and the system evaluates the feedback to indicate whether the action taken was the right one. It is up to the agent, and thus the learning algorithm, to use and interpret this feedback or reward signal. RL acts in a long-term goal-oriented approach that is concerned about

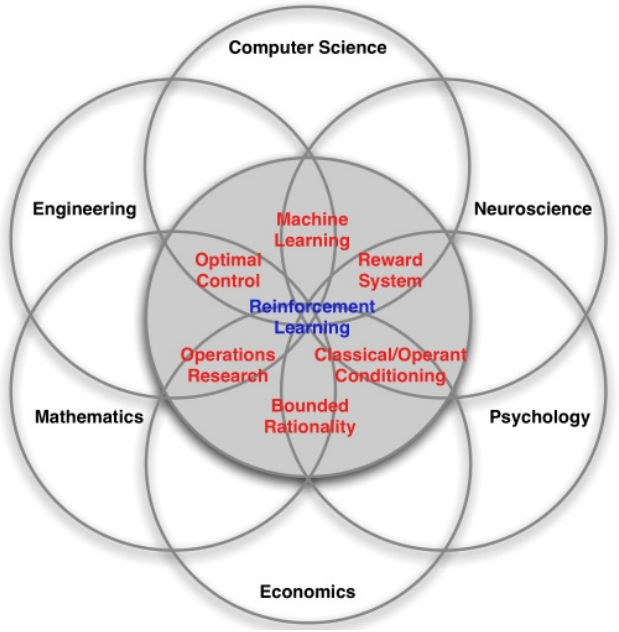


Figure 3.1: Reinforcement learning really is the culmination of many fields and has a rich history in optimization and behavioral psychology. [5]

maximization of a reward signal generated in the system. Several of the concepts discussed in the chapter are adapted from the hugely popular book by Richard and Sutton [5].

### 3.1.1 Markov Decision Processes

Markov property refers to the memoryless property of a stochastic process. In simpler terms, Markov processes are where the future is independent of the past given the present. Markov Decision Processes (MDP) are a classical formalization of sequential decision making [105]. MDPs are a mathematically idealized form of the reinforcement learning problem. A block diagram representation of the MDP structure is shown below in Fig. 3.2. A finite MDP is defined as a tuple (S, A, T, R, g) as shown below. Some of the key elements of the MDP and the RL framework are briefly described next.

- $\mathcal{S}$ is a state space
- $\mathcal{A}$ is an action space
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the state transition probability function

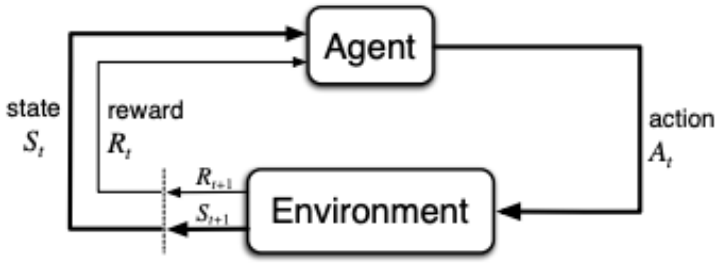


Figure 3.2: Block Diagram representation of Markov Decision Process where Agent takes an action and observes state space of the Environment and receives reward for the action. [5]

- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to R$ is the reward function
- $\gamma \in [0, 1]$ is discount factor

**Elements of RL system**

*Agent and Environment*

The learner and decision-maker of the system is called an *agent*. Everything outside the agent that it interacts with is called an *environment*.

*States, Actions, and Rewards*

The agent and environment interact at each of a sequence of discrete time steps, t. At each time step t, the agent receives some representation of the environment's state, $S_t \in$ S, and on that basis selects an action, $A_t \in A(s)$. The transition probability matrix P , sometimes referred to as the model of the system determines where the agent lands in next state $S_t + 1$. The agent receives a reward $R_{t+1} \in R$.

$$p(s', r|s, a) = Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\} \tag{3.1}$$

$$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r|s, a) = 1, \forall s \in \mathcal{S}, a \in \mathcal{A}(s) \tag{3.2}$$

*Returns and Episodes*

The goal of the agent is maximization of the expected value of the cumulative sum of a received scalar signal (reward, r). The sequence of steps the agent takes before naturally breaking the interaction with the environment is termed as an episode. Reward is the variable that the reinforcement learning agent learns to maximize. At each time step of an episode, the environment sends scalar rewards to the reinforcement learning agent. The agent trains with an objective to maximize the total reward it receives over the long run. The reward signal thus defines what are the good and bad actions for the agent. Reward shaping is performed to alter the priorities of the learning agent similar to the use of optimization functions in gradient-based optimization methods. An appropriate reward definition is critical to the success of the reinforcement learning agent; if the reward is not well aligned with the eventual goals of the user then the agent may learn sub-optimal behavior or suffer from local optima. Reward definition should ensure that the agent doesn't look for immediate returns but is long-term goal-oriented. The discounted cumulative reward expected (at a discount factor, $\gamma$) for continuous tasks is denoted by Eq. 3.3.

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \tag{3.3}$$

*Policy*

The policy is defined as a mapping from states to actions. The policy could be deterministic, and depend only on the state, $\pi(s)$ as determined by Eq. 3.3, or stochastic, $\pi(a|s)$ as determined by Eq. 3.4, such that it defines a probability distribution over the actions, given a state.

$$a = \pi(s) \tag{3.4}$$

$$\pi(a|s) = \mathcal{P}[A_t = a|S_t = s] \tag{3.5}$$

*Value Function*

Value function is central to reinforcement learning as most RL algorithms involve estimating value functions. These functions estimate how good it is for the agent to be in a particular state, or how good it is to perform a particular action by being in a particular state. Quantification of "goodness" is done using expected return. Two types of value functions are defined - state value function and action value function. The state value function of a state v(s) under a policy $\pi$, is the expected return when starting in $s$ and following $\pi$ thereafter. State Value function calculation is shown in Eq. 3.6. Similarly, action-value function shown in Eq. 3.7 is defined as the value of taking action $a$ in state $s$ under policy $\pi$, denoted q(s, a), as the expected return starting from $s$, taking the action $a$, and thereafter following policy $\pi$.

$$V_\pi(s) = E_\pi(G_t|S_t = s) \tag{3.6}$$

$$Q_\pi(s, a) = E_\pi(G_t|S_t = s, A_t = a) \tag{3.7}$$

*Bellman Expectation Equations*

A policy's value functions assign to each state, or state–action pair, the expected return from that state, or state–action pair, given that the agent uses the policy. Bellman equations formulate an iterative approach to solving the value functions and determining an optimal policy. The value function is decomposed as the sum of immediate return and the value of the successor state as shown in eq.

$$v(s) = \mathcal{R}_s + \gamma \sum_{s' \in S} \mathcal{P}_{ss'} v(s') \tag{3.8}$$

In the stochastic framework, probabilistic assessment of values are done under policy pi. State value function which is the value of a state s under a policy $\pi$, represented as $v_\pi(s)$, is the expected return when starting in s and following $\pi$ thereafter. It can be mathematically represented as shown in Eq. 3.9. Similarly, action value function which is the value of taking an action a in a state s under a policy $\pi$, represented as $q_\pi(s, a)$, is the expected return when starting in s, taking an action a, and following $\pi$ thereafter. It can be mathematically represented as shown in Eq. 3.10.

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}|S_t = s)] \tag{3.9}$$

$$q_\pi(s) = E_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}|S_t = s, A_t = a)] \tag{3.10}$$

Backup diagram (graphical representation of the system which shows transition between states per actions and associated rewards) as presented in Fig. 3.3 shows the Bellman Expectation equations for calculation of state and value functions.

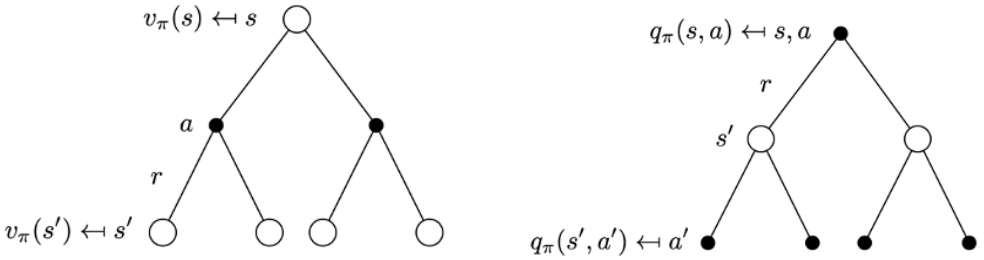


Figure 3.3: Backup diagram for state (left) and action (right) value functions using Bellman expectation equations

*Bellman Optimality Equations*

The agent learns to find the optimal policy $\pi^*$ that maximizes the long-term performance and the optimal policy is defined as the policy for which $v_{\pi^*}(s) \geq v_\pi(s) \forall s, \pi$. Bellman Optimality equations are presented for optimal state and value functions respectively in Eqs. 3.11 & 3.12.

$$v_*(s) = max_a \mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'} v_*(s') \tag{3.11}$$

$$q_\pi(s) = \mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'} max_a q_*(s', a') \tag{3.12}$$

Backup diagram as presented in Fig. 3.4 shows the Bellman Optimality equations for calculation of optimal state and value functions; highlighting that actions are taken so as to maximize state value or action value function

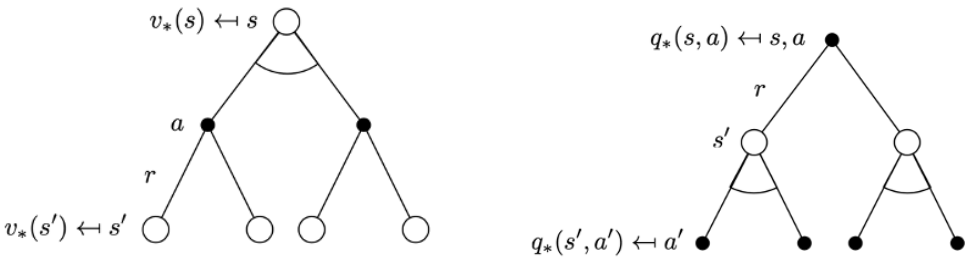


Figure 3.4: Backup diagram for optimal state (left) and action (right) value functions using Bellman optimality equations

### 3.1.2 Model-free Methods

Majority of the algorithms discussed and implemented in this research work are model-free methods which do not rely on knowledge about MDP transitions. In contrast, model-based methods are those RL methods that utilize the information about behavior of the

environment, for planning and learning. Model-based techniques rely on the availability of state-transition probability matrix, for instance. Putting equation which calculates value functions of states under policy $\pi$, in a matrix form results in .

$$v = \mathcal{R} + \gamma \mathcal{P} v \tag{3.13}$$

$$v = (1 - \gamma \mathcal{P})^{-1} \mathcal{R} \tag{3.14}$$

This is a system of linear equations that can be solved by various linear solvers. As can be seen in the Eq. 3.14, the solution to this problem relies on the knowledge of the model, or the transition probabilities ($\mathcal{P}$). Given the high complexity of computations, $\mathcal{O}(n^3)$, direct solution is possible for small MDPs. And for the larger MDPs, there are iterative methods available such as dynamic programming, Monte-carlo evaluation, Temporal-Difference learning, to name a few. More advanced methods likes Dyna-Q have been proposed that integrate model learning, planning and direct RL. The model-based methods suffer from the curse of dimensionality and the learned policies also result in sub-optimal behaviors if the models are not accurate. The drilling processes are highly complex in nature and involve a significant amount of uncertainty in modeling thus limiting the usage of model-based methods.

The other set of algorithms are model-free methods. They assume the transition probabilities are not known and they estimate the value functions and policy by performing roll outs on the system. Monte Carlo, temporal difference and policy search methods are the most common model-free algorithms used. Two approaches are proposed - 1) Value function based methods that learn value function and infer optimal policy from that and 2) Policy-search methods that directly search the parameter space for optimal policy. OpenAI provides a taxonomy of the important algorithms in modern RL as shown in Fig. 3.5.

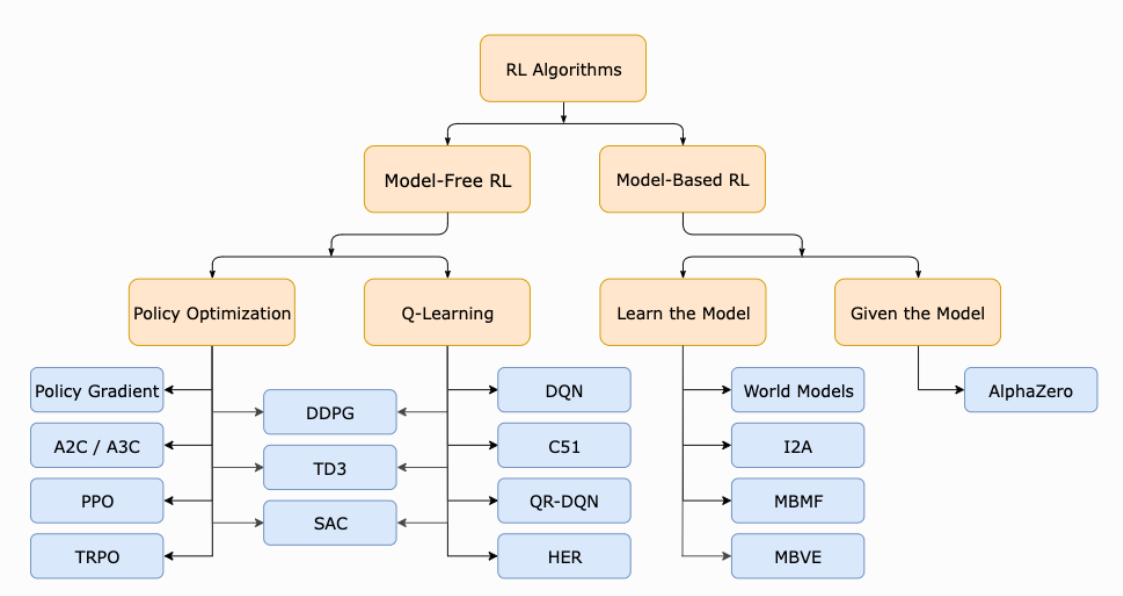


Figure 3.5: Taxonomy of algorithms in modern RL *(Source: www.spinningup.openai.com)* which broadly categorizes the algorithms into model-based and model-free methods out of which two sub-categories of polcicy gradient and Q-learning methods are shown.

*On-policy vs Off-policy*

Training a policy involves collecting data from the environment, sometimes referred to as gathering experiences. Sample-based learning methods can be broadly categorized as off-policy learning and on-policy learning methods. These methods fundamentally differ in the manner experiences are collected. On-Policy learning algorithms are the algorithms that evaluate and improve the same policy which is being used to select actions. In other words, the learning improves the same policy that the agent is already using for action selection. In mathematical terms, Target Policy = Behaviour Policy. A popular on-policy method is SARSA algorithm. Other examples include Policy Iteration, PPO, TRPO. Off-Policy learning algorithms on the other hand evaluate and improve a policy that is different from Policy that is used for action selection. In mathematical terms, Target Policy != Behaviour Policy. A popular off-policy method is Q-Learning algorithm. Other examples include DQN, DDQN, DDPG.

SARSA and Q-Learning are widely used algorithms to solve sequential decision-making problems in a wide variety of applications. Both the algorithms are Temporal-Difference (TD) methods that calculate temporal error in value functions instead of total cumulative reward. The value function is then updated towards an estimated return as opposed to an actual return in Monte Carlo methods. The update equation (Eq. 3.15) is as shown below.

$$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1} - V(S_t)) \tag{3.15}$$

*SARSA*

SARSA is an on-policy temporal difference algorithm which tries to learn an action value function instead of a state value function. The update equation of the SARSA method is shown in Eq. 3.16 and a backup diagram is shown in Fig. 3.6.

$$Q(S, A) \leftarrow Q(S, A) + \alpha(R + \gamma Q(S', A') - Q(S, A)) \tag{3.16}$$

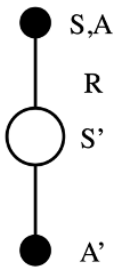


Figure 3.6: Backup diagram of SARSA algorithm

*Q-Learning*

The Q-learning algorithm [106] is one of the most popular reinforcement learning techniques. It is a model-free off-policy value iteration algorithm. Being an off-policy method

gives it sample efficiency over SARSA. The action here is chosen with respect to a $\epsilon$-greedy policy (Target Policy) while the Q-function is updated by using a policy (Behaviour Policy) which is directly greedy with respect to the current Q-function. Q-learning differs from SARSA in the way the Q-function is updated. The update equation is as shown in Eq. 3.17 and the backup diagram is shown in Fig. 3.7.

$$Q(S,A) \leftarrow Q(S,A) + \alpha(R + \gamma max_{a'} Q(S',a') - Q(S,A)) \tag{3.17}$$

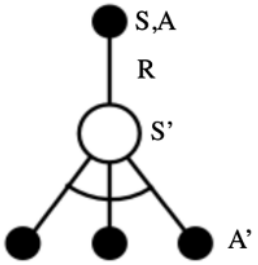


Figure 3.7: Backup diagram of Q-Learning algorithm

### 3.1.3 Approximate Solution Methods

As the number of states grow in the state space, or if the application calls for a continuous state space, applying reinforcement learning algorithms and finding an optimal policy could become an impossible task. Along with the limitations on memory, data, and time, have such state space will result in a sparse environment where a state would be almost never visited again during learning. The solution to this problem is to find a generalization from the states that have already been encountered. The use of function approximators, an instance of supervised machine learning techniques like neural networks, has been a successful approach. The use of Deep Neural Networks as function approximators for value functions or policy in a RL framework is referred to as Deep Reinforcement Learning (DRL). For our application

of training a drilling agent to perform the complex process of geosteering, DRL is the right approach as it can handle high dimensional state and action space. Three popular techniques of implementing DRL are policy gradient methods, Q-learning variations, and actor-critic methods. Deep Q-Learning (DQN) method is briefly discussed next, followed by a policy gradient DRL method.

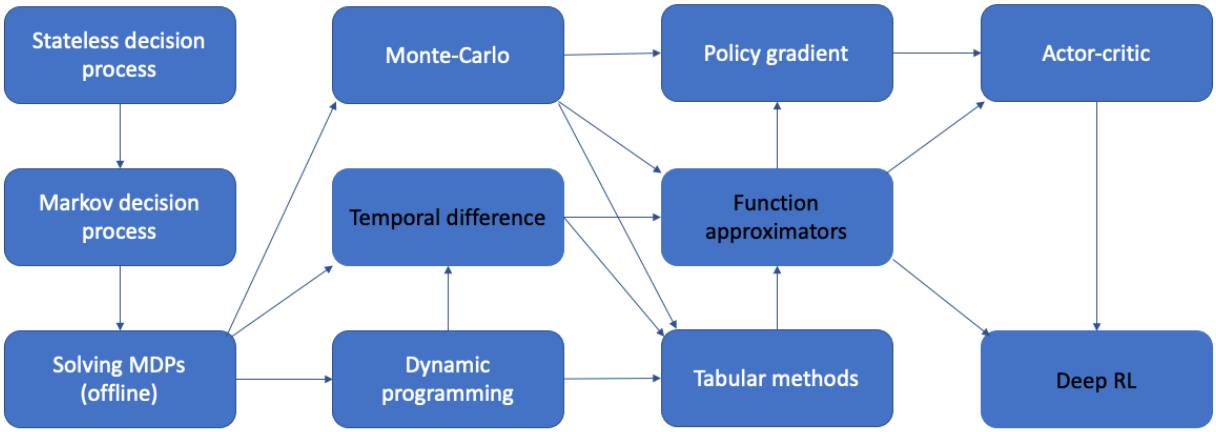


Figure 3.8: High-level view of RL methods and their interconnect highlighting the emergence of Deep Reinforcement Learning from Temporal Difference methods utilizing function approximators.

#### *3.1.3.1 Deep Q-Learning (DQN)*

Mnih [107] developed the first deep learning model to successfully learn control policies directly from high-dimensional sensory input using reinforcement learning. The model termed as Deep Q-Learning (DQN) is a convolutional neural network, trained with a variant of Q-learning. DQN has since been widely popular in several applications of model-free reinforcement learning. RL learns from a scalar reward signal that is frequently sparse, noisy and typically sequences of highly correlated state. Further, the data distribution changes as the algorithm learns new behaviors. DQN was first to demonstrate that a convolutional neural network can overcome these challenges to learn successful control policies from high-

dimensional raw data in complex RL environments. Mnih addressed these problems with the use an experience replay mechanism and randomly sampling and training on previous transitions. Psuedocode for the DQN algorithm is presented in Appendix C.

### 3.1.4 Proximal Policy Optimization

In many domains, it is more efficient to learn the policy directly instead of deriving one from state and action values. This approach is the policy gradient (PG) method where a policy is determined without using a policy function. In this method, the policy is a parametrized function $\pi(a|s)$ as shown in Eq. 3.18. $J(\theta$ is a scalar policy performance measure (sum of discounted rewards) with respect to the policy parameters Eq. 3.19. PG methods find the best parameters ($\theta$) to maximize (optimize) a score function $J(\theta$, given the discount factor $\gamma$ and the reward r. In other words, first the quality of the policy is measured with a policy score function and then policy gradient ascent is used to find the best parameter that improves the policy.

$$\pi_\theta(a|s) = \mathcal{P}[a|s] \tag{3.18}$$

$$J(\theta) = E_{\pi\theta}[\sum \gamma r] \tag{3.19}$$

Although PG methods optimize the policy, they are not sample efficient. As a solution, data is sampled from a given policy to find a new policy that maximizes the expected return using Trust Region methods. The idea is to approximate the objective function, in the case the policy, with a simpler function. The method referred to as Trust Region Policy Optimization (TRPO) updates policies by taking the largest step possible to improve performance, while satisfying the special constraint on how close the new and old policies are allowed to be. Proximal Policy Optimization (PPO) [108] shares the same motivation with TRPO but achieves increase policy improvement without the risk of performance collapse. This is done with a Clipped Surrogate Objective Function which limits the magnitude of

change between successive policies such that an improvement (natural policy) is guaranteed. The details of the PPO algorithm are shown in Appendix C. PPO algorithms have been used for the implementation of steering models discussed in Chapters 4 & 5.

## 3.2 Drilling Simulation Environment

Simulation is the imitation of an operation or behaviour of a real-world process or a physical system over time. Reinforcement Learning methods that operate in an iterative manner required data corresponding different combinations of control actions and the state space of the environment. RL typically leverages a simulator to obtain the training data. In recent years, there have been significant advancements in the domain of DRL research and algorithm design [107, 108, 109, 110]. Essential to the rapid development has been the presence of robust and scalable simulation platforms such as the Arcade Learning Environment [111], VizDoom [112], MuJoCo [113], and many others [114, 115, 116].

In this research study, multiple simulation platforms have been used at various stages for different problems. Relevant functionality and application of four such platforms for RL are briefly discussed below.

### 3.2.1 Unity Physics Engine

Unity is a real-time 3D development platform that consists of a rendering and physics engine as well as a graphical user interface called the Unity Editor. Unity has received widespread adoption in the gaming, AEC (Architecture, Engineering, Construction), auto, and film industries and is used by a large community of game developers to make a variety of interactive simulations, ranging from small mobile and browser-based games to high-budget console games and AR/VR experiences. Unity has a built-in 2-D physics engine (Box2D) and a 3-D physics engine (Nvidia PhysX). There are options to incorporate high-fidelity physics into the simulator, but it is not in the scope of current work. Unity simulation is seen as a powerful cloud platform to train AI in a safe, virtual environment, test at a massive scale, and validate before taking a solution into production. This is especially critical for

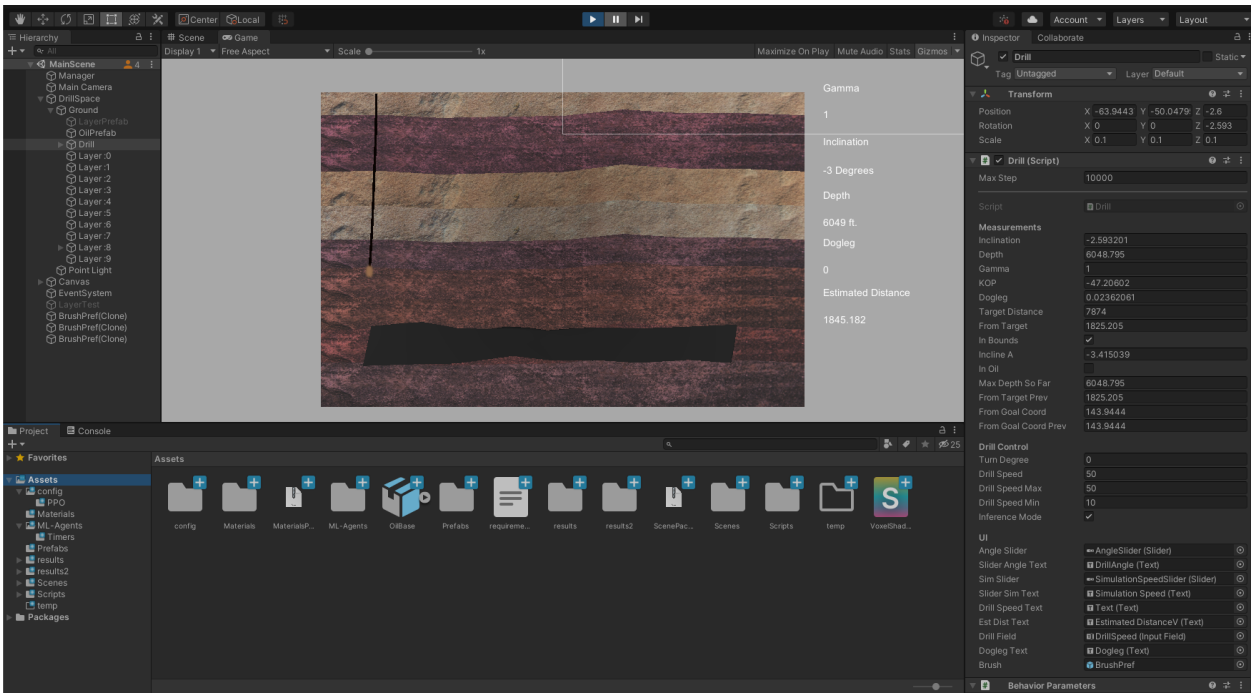


Figure 3.9: Snapshot of Unity Editor showing various 'GameObjects', Project space, and behavior parameters setup for building the Drilling Simulator

applications like drilling operations where high capital costs and high risk are involved. A basic 2-D simulator as shown in Fig. 3.9 has been developed in the Unity platform to train a drilling agent to perform geosteering. While this section is by no means a tutorial for Unity, some of the key elements of the platform are explained in reference to the Drilling Simulator (DS).

*Key elements of Simulator Development*

Assets correspond to various project items that can be implemented in the environment. An example of asset used in the DS is textures of formations. Every object present within the simulator is a GameObject. They don't append any functionality themselves but merely acts as holders for components like the Transform, Light, Script, and RigidBody components. Components are the basic building blocks of objects and their activities in the simulator. The simulations run on Scenes. The GameObjects can be setup as Prefabs so they can be reusable in the project. Some examples of GameObjects are formations, oil layers, drillbit, well path,

etc. Behaviours are added to the simulator elements using C# scripts for GameObjects. Some examples include Motion, Transform, Rotation, Colliders etc. when the agent drills through formations.

*Functionality of the Simulator*

Different layers on the simulation environment represent different rock formations below the surface (Top boundary of the simulator is considered as surface). All the rock formation have predefined LWD properties of Gamma and Resistivity values. The Oil layer is treated as a separate object for the sake of simplicity. Thickness of formation and oil layer are predefined and are randomized during training if required. Measurements made bottomhole are assumed to be near-bit and include TVD, and Inclination. Dogleg severity calculations are done at every time step and added to the data logs. The data logs consist of measurements of Gamma, Resistivity, TVD, Inclination, Dogleg and timestamp. In the manual mode, the agent can drill in any direction based on user-input for steering angle and drilling speed.

### 3.2.2 ML-Agents

The Unity Machine Learning Agents (ML-Agents) is an open-source toolkit that enables Unity simulations to serve as environments for training intelligent agents. ML-Agents consist of several components. First is the Learning environment that is developed in Unity that contains the scene and other environment elements. A Python API where RL algorithms can be written connects through an external communicator to the learning environment, to our Drilling Simulator in this case. Inside the learning environment, a GameObject can be set as an Agent that is trained by optimizing its policy, sometimes referred to as Brains. In the drilling simulator, the drillbit is assigned the role of the agent. Behaviour parameters of the agent (a sample is shown in Fig. 3.10) are setup in such a way to receive information about states, and to take actions. In addition C# scripts of the agent are modified to setup the behaviour as designed in the model. Some examples of the behaviours changed are listed below.

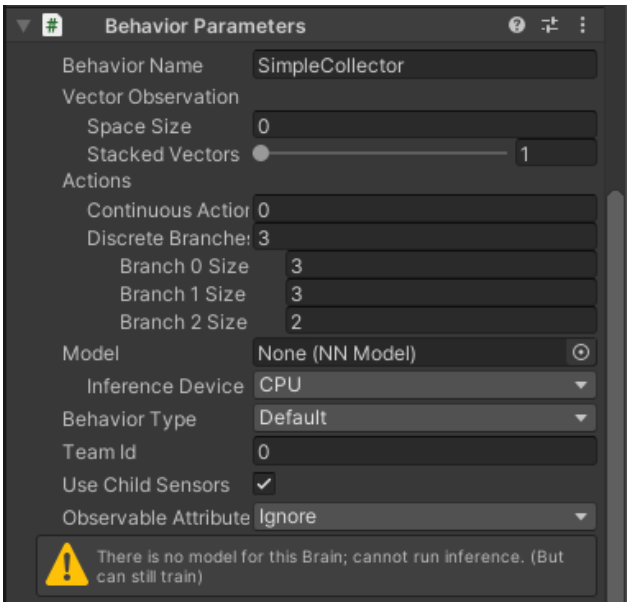


Figure 3.10: Snapshot of Unity Editor showing Behavior Parameters of an Agent for a discrete action space setup as shown by 3 element Actions configuration.

- OnEpisodeBegin is called whenever a new episode starts
- CollectObservations is where observations are sent to the agent
- OnActionReceived is where an action is sent from the agent

ML-Agents has the capability to run off-policy and on-policy algorithms (PPO, SAC for example) directly interfacing with the Unity simulator. Batch training, where multiple environments are trained simultaneously to reduce learning time, is also implemented into ML-Agents.

### 3.2.3 NORCE OpenLab

A drilling simulator called *OpenLab* (Figure 3.11) that is developed and managed by the Drilling & Well Modeling group of NORCE Energy in collaboration with the University of Stavanger, has been used for training some of the RL models in this study. OpenLab is an integration of the physical and virtual drilling and well operations, which is new and unique to the Oil & Gas drilling world. The simulator can be run interactively through a


Figure 3.11: NORCE OpenLab Simulator *(Source: https://openlab.app/)* showing a sample simulation configuration with wellplan and trajectory setup

web browser, or also programatically through Matlab or Python packages. A simulation in the Web Enabled Drilling Simulator is based on NORCE's computer models of well flow and drillstring mechanics. In this work, a Python client has been used to create a a gym environment. A drilling configuration is setup on OpenLab with realistic physical values to several drilling parameters.

### 3.2.4 Multi-Body Dynamics Simulator

Chrono is an OpenSource physics-based modelling and simulation infrastructure based on a platform-independent design implemented in C++. ProjectChrono library, a multi-physics simulation engine has been used by Losoya et.al. [117] to model the rig, drillstring, and other rigid-body dynamics. The main capabilities of the multi-physics engine used are:

- Multi-body Dynamics - Able to run simulations of mechanisms made of rigid bodies. Apply a constraint to parts using a wide set of joints. Add motors, linear actuators,

springs, and dampers while applying forces and torque.

- Finite Elements - Create finite elements, and model flexible parts including beams, cables, and shells. Apply local or distributed loads. Perform non-linear analysis with large deformations.

- Large Scale Simulation - Simulate large scenarios such as granular flows, vehicle-soil inter- action, and fluid-solid interaction. Perform co-simulation with other CFD and FEA software packages.

- Collision Detection - Define collision shapes using meshes or primitives. Compute frictional contact forces using automated collision detection algorithms. Define surface properties and surfaces

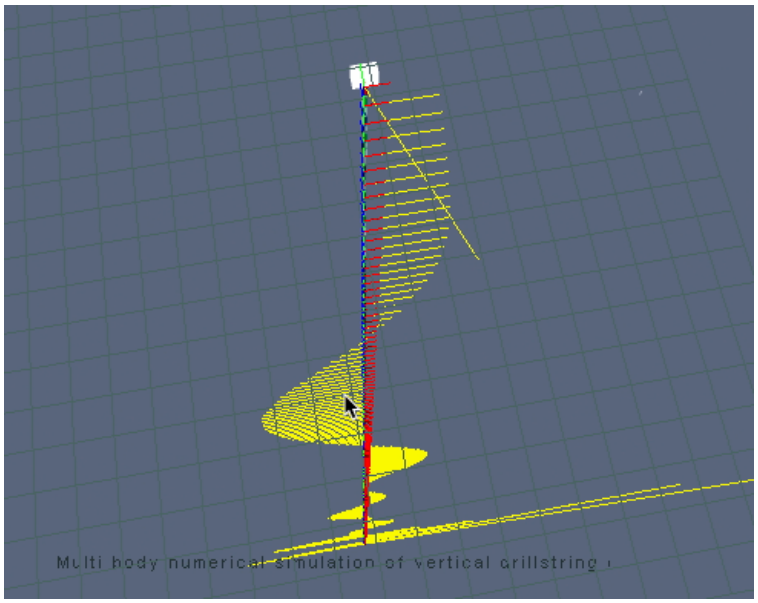


Figure 3.12: Snapshot of Chrono Multibody Dynamics Simulator showing forces in 3-dimensional space for a vertical drillstring configuration with 5 drillpipes and simulated WOB

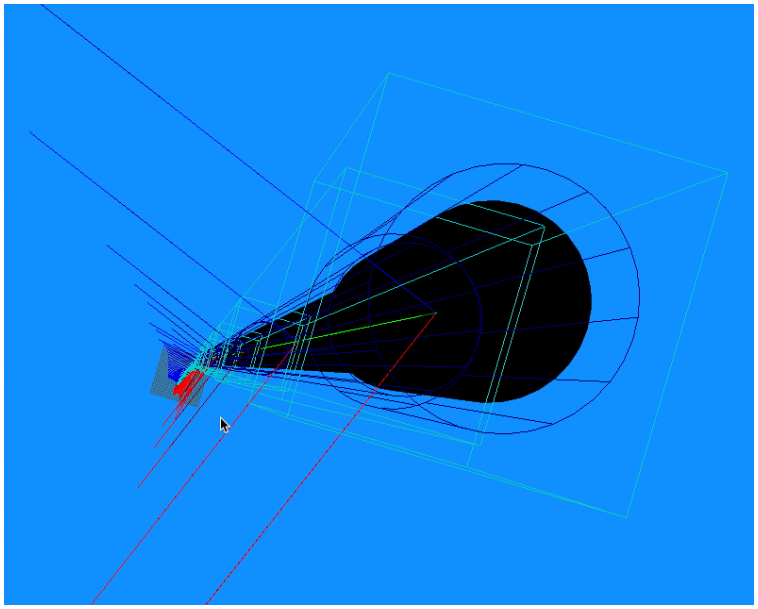

Figure 3.13: Snapshot of Chrono Multibody Dynamics Simulator showing Horizontal Drill-string dynamics for a 10 drill pipe configuration

Chrono simulator has been used to generate synthetic data for drilling performance optimization. Specifically, measurements of tri-axial forces and torque at the bit is done for a given drillpipe and BHA configuration, to study the behaviour of vibrations in drill pipe. WOB and RPM parameters are altered at the surface and change in behaviour downhole is logged. Screenshots of the multi-body dynamics simulator displaying forces on a drill pipe section are shown in Fig. 3.12 & Fig. 3.13.

# 4. SEMI-AUTONOMOUS SYSTEM: PLANNED WELL TRAJECTORY TRACKING

This chapter shows the implementation of the first level of automated geosteering drilling system that is trajectory tracking of a planned well. The chapter begins with an overview of the problem by also summarizing the current state of well planning and directional drilling operations. Trajectory tracking is a setup in the RL framework next. The model is tuned by modifying various reward schemes. Training is performed on a 2-D well simulator and the performance is evaluated with different metrics.

## 4.1 Overview

The arrival of Rotary Steerable Systems (RSS) significantly raised the level of automation in directional drilling control operations which were previously considered as highly complex processes that could only be handled by specialists. However, the operation of steering, that is keeping the well within an acceptable range of a predefined trajectory is predominantly a manual process. Directional driller at the rig surface continuously monitors surface parameters and downhole measurements to ensure the actual trajectory coincides with that of the well plan is developed by directional drilling engineer. Steering decisions are taken at the surface and downlinked to RSS. The more modern *Steering Advisory Systems* act as semi-autonomous system by providing steering recommendations automatically. Yet, uncertainties such as encountering hard formations, changes in drilling parameters, hard-to-model behaviour like bit-walk, etc. make it difficult to fully automate the path tracking operation.

Directional drillers downlink steering commands like toolface angle, steering magnitude to follow the predefined trajectory and specified targets. As the RSS tools matured, automation of steering individual sections of the path, such as attitude control for verticals, tangents, and horizontal sections evolved wherein only setpoints are sent downhole instead of frequent downlinking of steering commands. The remaining significant challenges for autonomous trajectory tracking include integration of surface and downhole information, and automatic

curve control for geological steering while compensating for disturbances and avoiding high local dogleg, and delivery high-quality wellbore.

### 4.1.1 Parallels from Other Industries

Reinforcement Learning algorithms were used for path tracking of a real car-like mobile robots [118]. Data was generated during off-line simulations using a random path generator to cover different curvatures and initial positions, headings and velocities of the vehicle for the RL agent to learn [119]. It was shown that the trained RL agent was able to control the car smoothly and reduce the velocity adaptively to follow a given track in comparison to conventional controllers. For a similar application, RL was used to develop self-optimizing path tracking controller for intelligent vehicles [121, 122, 123]. For the lateral control of an autonomous vehicle, a steering method based on the fusion of the RL and traditional PID controllers is designed to adapt to various tracking scenarios. According to the pre-defined path geometry and the real-time status of the vehicle, the interactive learning mechanism, based on an RL framework using actor– critic—a symmetric network structure, can realize the online optimization of PID control parameters in order to better deal with the tracking error under complex trajectories and dynamic changes of vehicle model parameters. The concepts of utilizing deep reinforcement learning for path tracking was also able to solve the curved path-following problem for underactuated vehicles or marine vessels subjected to unknown ocean current influence [6]. Transfer learning was used to extend a trained policy which resulted in computational cost advantage.

## 4.2 Model Setup

The Directional Drilling environment is modeled as a Markov Decision Process (MDP). The MDP is designed for a trajectory tracking scenario where a predefined well plan is provided to the driller. Consequently the model assumes that the provided trajectory is the optimal path to be followed in order to achieve the purpose of drilling the well. The different components of the MDP framework - Agent, Environment, States, Actions, Rewards, and

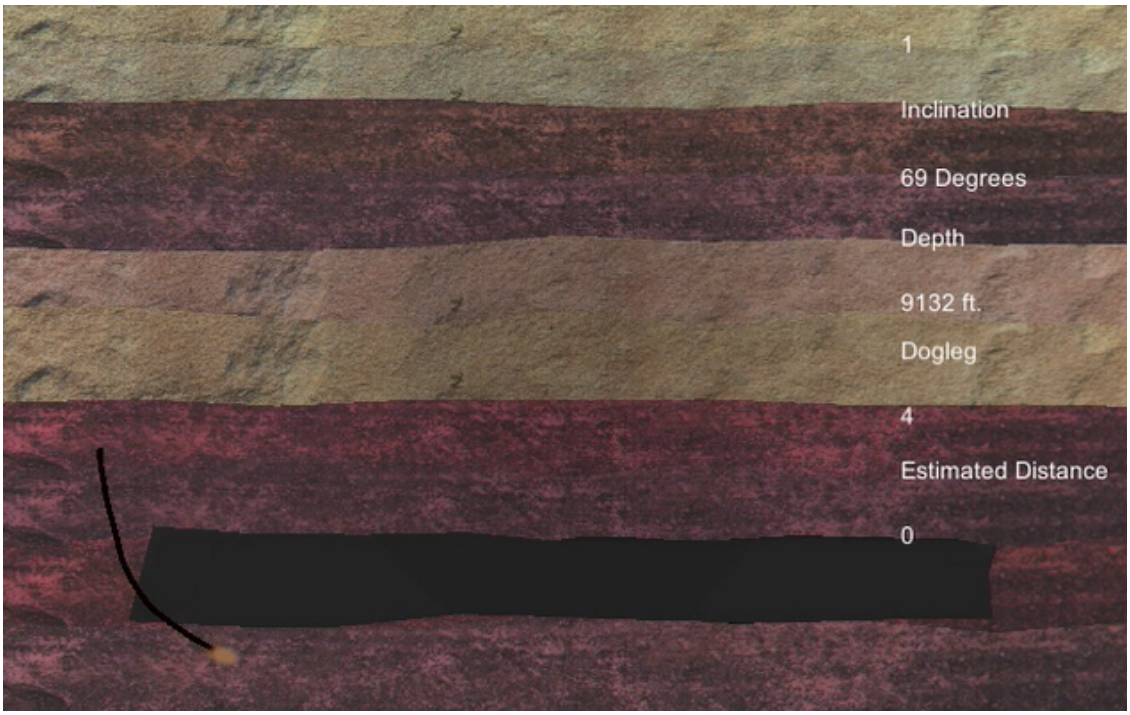


Figure 4.1: Snapshot of Unity Simulator depicting formations (target zone shown in black) with drilling agent (shown as yellow dot at the tip of trajectory) learning to drill through target zone and measurements shown on the right

Goals are defined accordingly. Stochasticity is incorporated into the model by introducing noise into some of the actions. Details of the model setup are as below.

**Simulator**

A simplified 2-D earth model is built in the Unity simulator as described in section 3.2. that represents different layers of formations with varying thicknesses, shapes, gamma and resistivity values. A total vertical depth of 1000 ft. is used in the simulations. A thin oil layer is positioned towards the lower portion of the environment. Drilling starts from the surface (TVD = 0 ft.). A pre-defined well trajectory for drilling through the formations is given as input to the agent. A snapshot of the simulator depicting formations and sample trajectory is provided in Fig 5.4.

*Domain randomization*

Producing agents that can generalize to a wide range of environments is a significant challenge in reinforcement learning. One method for overcoming this issue is domain randomization [124], whereby at the start of each training episode some parameters of the environment are randomized so that the agent is exposed to many possible variations. There are commonly three types of 2-D well profiles that are in practice for Oil & Gas well drilling as described in Miska et.al. [1] and shown in Fig 4.2. Domain randomization is done on well paths and positions of oil layers. With enough variability in the simulator environment, the real world may appear to the model as a close version of one of the domain variations. Without such randomization, the agent only learns to follow sample paths used during training and can not be general.

A set of trajectories are generated based on the above mentioned three major well profiles in 2-D space. Random paths are generated for each episode with varying start locations on the surface, kick off points, build rate, horizontal section lengths and measure depths. In addition, training is also performed on complex trajectories that may not be practically feasible for real-world drilling operations. This is done to ensure the agent does not suffer from local optima and to achieve a robust model.

**Agent**

The drilling agent in this setup mimics the directional driller on the rig surface. Thus, the goal of the agent is to traverse the predetermined well trajectory without any deviations. At each step of an episode, the agent observes the states and takes actions so as to maximize the rewards in achieving this goal. RL is long-term goal oriented. Although the long-term goal of the driller is to drill a high quality wellbore that maximizes the contact with the target zone, in the framework of this setup, the drilling agent is only interested in traversing the path because of the assumption that the provided path is optimal in achieving the above-mentioned long-term goal.

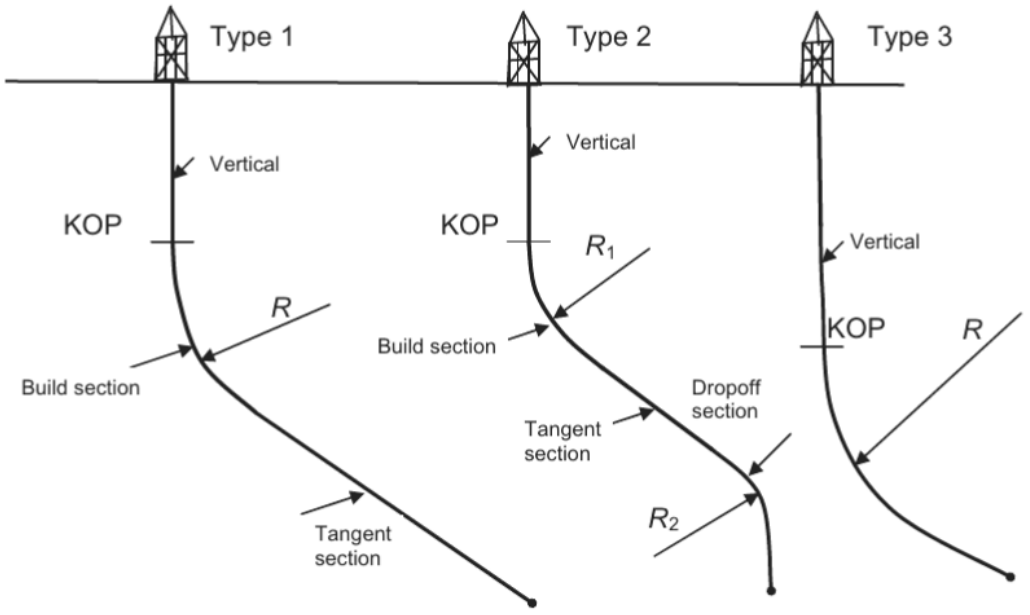


Figure 4.2: Three major types of 2-D wellbore trajectories which are used as basis for generating trajectories for testing the learned RL agent

**Environment**

A wellpath is assumed to be available before the drilling simulation runs start. Waypoints are placed along the trajectory of the wellpath. The waypoints defined in the 2D space carry the information of the wellbore position (x,y). At every step, the agent will have information about the current position, two closest waypoints - termed as 'home' and 'destination', and position of 10 waypoints ahead. In addition, dogleg severity at every timestep is also available. Two additional parameters - Cross Track Error and Heading Angle Error are also part of the space. These two parameters are described in detail in the 'Rewards' section. The observations that the agent makes in the environment which define the states are as listed below.

**State Space** ($\mathcal{S}$)

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth

- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Home waypoint position
- Destination waypoint position
- 'Next' 10 waypoints position
- Cross Track Error
- Heading Angle Error
- Dogleg Severity ($dg$)

It is assumed that all the above observations are available in real-time while drilling. In the real-world, some of the sensors are located behind the bit, sometimes about 60 ft. farther. But for the simulations in this model, and for the rest of the chapters unless it is specified otherwise, it is assumed that the observations are made near-bit.

**Actions**

A continuous action space is defined for the model where the agent can steer left or right adhering to mechanical constraints. Two different scenarios - one with a constant drilling (or Steering) speed and the other with drill speed as an action are implemented. The idea behind having steering speed as an action is to simulate and observe the behaviour when hard formations are encountered.

**Action Space** ($\mathcal{A}$)

- Steer: $[-6°, +6°]$ ($\dot{\theta}$)
- Throttle (Case 1): constant speed ($v$)
- Throttle (Case 2): variable speed [1,10] ft/s ($v$)

**Stochasticity**

Uncertainties of the real-world drilling environments such as hard formations, bit walk are incorporated into the simulator by intentionally adding Gaussian noise to the actions at each step of the episode. For the case of variable speed control, the added noise is a function of gamma, inclination, and drill speed.

**Rewards**

For the trajectory following control problem, the objective is to control the drill agent in such a way that it converges to the predefined wellpath. The main indicator of the convergence used in this setting is the cross-track error, $e$ (XTE) which tells how far the drill is from the path. The agent is penalized in proportion to the XTE and rewarded for staying close to the target path. Waypoints and cross-track error calculation are shown in Fig. 4.3. Assuming the blue curve represents a section of the wellpath, waypoints represented as maroon dots are placed along the wellpath. XTE is calculated from the drill bit that is deviated from the planned trajectory. XTE is calculated as the distance normal to the two closest waypoints ('home' and 'destination'). In addition, to limit excess steering which would result in high bending moments on the drill pipe, a negative reward is imposed for larger doglegs. The agent is also rewarded positively whenever the waypoints are switched ahead because that indicates the drillbit propagation. Finally, a small living reward is added to ensure drilling is completed in shorter time. This is particularly useful for variable speed control setup.

**Reward Function** ($\mathcal{R}$)

- $s_t$ : state at time=$t$, $(x_t^{pos}, y_t^{pos}, \theta_t, dl_t, d_t^{oil})$
- $a_t$ : action at time=$t$, $(\dot{\theta}_t, v_t)$
- $R(s_t, a_t, s_{t+1})$ : reward obtained on reaching $s_{t+1}$ after executing $a_t$ at $s_t$

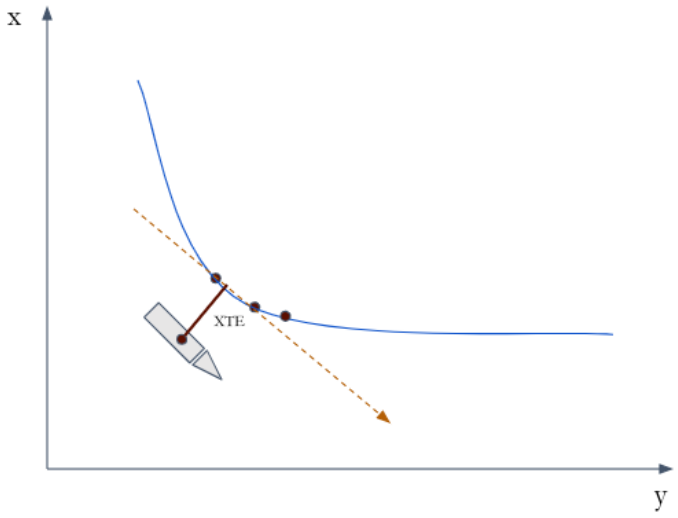


Figure 4.3: Schematic showing waypoints and cross track error calculated as the distance of the current position of the agent normal to the tangent of the two closest waypoints on the trajectory

$$\mathcal{R}(s_t, a_t, s_{t+1}) : \begin{cases} -k_1(e), & \text{if } s_{t+1} = \text{deviation from path} \\ +k, & \text{if } s_{t+1} = \text{new waypoint set as 'home'} \\ -k_2(dg), & \text{if } s_{t+1} = \text{steering in bends} \\ +0.0001, & \text{if } s_{t+1} = \text{drilling is not idle} \\ Terminal, & \text{if } s_{t+1} = \text{moving in a circle} \end{cases}$$

### 4.2.1 Reward Shaping

Several 'waypoints' are defined along the given target wellpath to calculate XTE. In addition to the proportional rewards for XTE as described above, a different implementation with a Gaussian reward with amplitude alpha and standard deviation sigma, is proposed for XTE.

$$\mathcal{R}(s_t, a_t, s_{t+1}) = -k_2(dg^2) + 0.0001 + \begin{cases} \alpha e^{(-c_e^2/2\sigma)}, & \text{if } s_{t+1} = \text{deviation from path} \\ 0, & \text{if } s_{t+1} = \text{target path} \end{cases}$$

Hyperparameters of the model include k2 from the first term of the reward function that ensures less bending moment on the drill pipe, and the Gaussian function variables alpha and sigma. A sample Gaussian reward function is as shown in Fig 4.4.

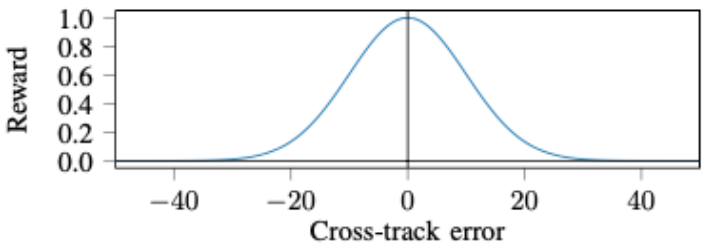


Figure 4.4: Gaussian Reward function for Cross Track Error (XTE) [6]

*Heading Angle Error*

Although XTE ensured the agent learns minimizing deviations from the planned trajectory, the inference models showed scope for improvement in accuracy, especially at the curved sections. A new variable called Heading Angle error ($\alpha$) has been defined and added to the reward criteria. As shown in Fig, two tangents are drawn - one between the home and destination waypoints, and the other between the next two waypoints ahead of the current position. The angle made between the two tangents is defined as the heading angle error. The agent is given a negative reward proportional to the magnitude of the heading angle error. The intuition behind the addition of this parameter is to help the agent learn, anticipate and prepare for bends in the trajectory.

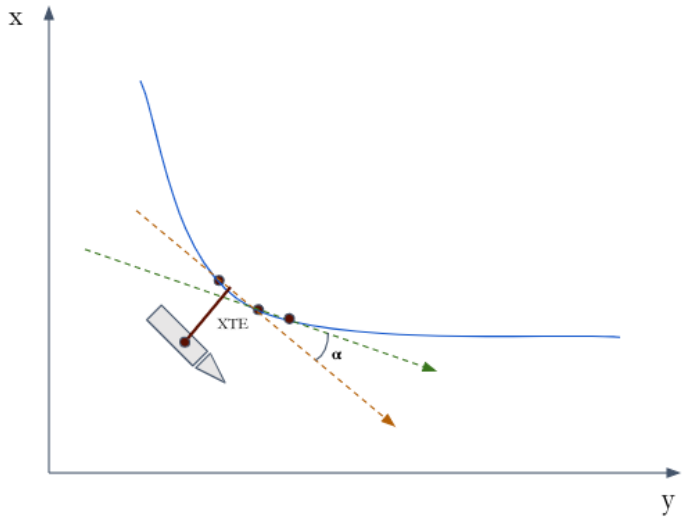


Figure 4.5: Schematic showing addition of Heading Angle Error parameter which is the difference in angles made by the current direction and target direction defined by future waypoints

A summary of the MDP formulation and the model setup for RL implementation is depicted in Fig. 4.6.

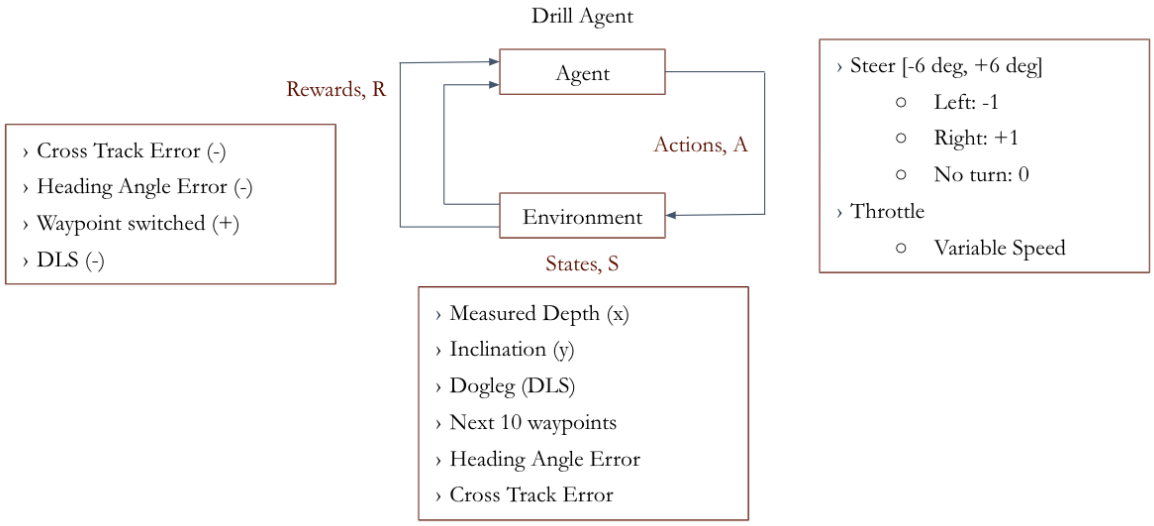


Figure 4.6: RL Model Setup for Semi-Autonomous Steering System with continuous actions, state space and reward configuration

### 4.3 Implementation

The agents starts by exploring off the baseline wellpath provided. At each time step of an episode, the current state is observed, and an action is taken based on the policy. The action taken determines the next state and reward received. The process is continued and rewards are accumulated until a terminal state is reached. PPO, an on-policy algorithm is first implemented on the system. Psuedocode of the PPO algorithm is shown below. The configuration of the algorithm and the associated hyperparameters are shown in Table 4.1. The model is trained for 300,000 episodes on the simulator.

**Algorithm 1** Proxy Policy Optimization

**Require:** initial policy parameters $\theta_0$ and value function parameters $\phi_0$
**for** k = 0,1,2 ... **do**
Collect set of Trajectories $D_k$ by running policy $\pi_k = \pi(\theta_k)$
Compute Rewards, $R_t$
Compute Advantage Estimates, $A_t$ based on current value function, $V_\phi$
Update the Policy by maximizing PPO-Clip objective:
$\theta_{k+1} = argmax_\theta \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} min(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)))$
typically via stochastic gradient ascent with Adam.
Fit Value Function by regression on mean-squared error
$\phi_{k+1} = argmin_\phi \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} (V_\phi(s_t) - R_t)^2$
typically via a gradient descent algorithm.
**end for**

| parameter | value |
|---|---|
| learning rate | 0.0003 |
| batch size | 128 |
| buffer size | 1024 |
| number of layers | 2 |
| hidden units | 256 |
| maximum steps | 300000 |
| summary frequency | 1000 |

Table 4.1: PPO Configuration for Trajectory Tracking learning showing hyperparameters - learning, hidden units of neural network layer, maximum number of episodes.

Although majority of the deviated O&G wells are variations of the three common well profiles discussed in the chapter previously, it is found that training only on such trajectory variations is not efficient for learning. The agent could not be trained on the curved sections for a large number of episodes. A solution implemented to overcome the situation was to use complex trajectories such as the one shown in Fig. 4.7 for training the agent. Testing of the trained model is performed on new trajectories that are similar to real-world well profiles.

Every episode of the training phase starts with a random wellpath generated from a set of predefined trajectories. Waypoints are generated along the selected trajectory. As can be seen in Fig. 4.8, waypoints marked as blue dots represent the target trajectory that the agent is required to learn to traverse. The simulation starts from the top of the 2-D space (represents surface). The black curve in Fig. 4.8 shows the path the drill agent has taken in that episode.

In the first case, the agent is allowed to only drill at constant speed. This set a limitation on the agent's action space. The agent could only take steering decisions (left, right, straight) within the range of predefined steering settings. Bit-walk is induced into the environment

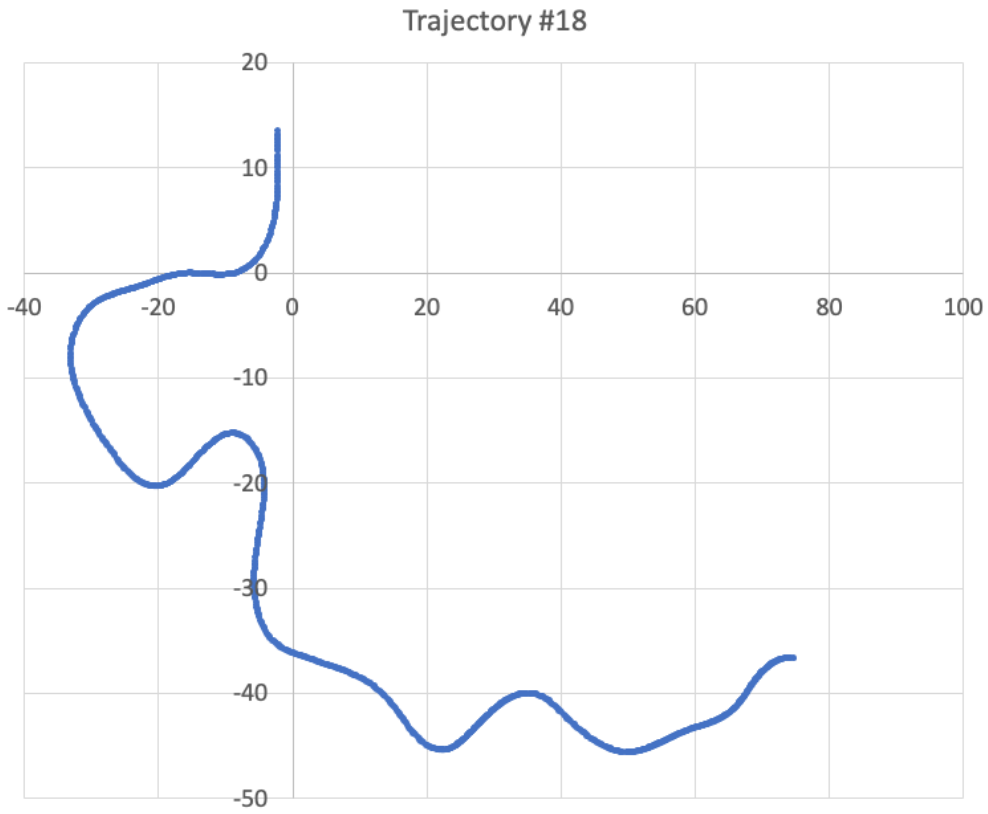


Figure 4.7: Example of a complex trajectory used for training that is generated based off the common well profiles; horizontal axis represents inclination and vertical axis represents the vertical depth in relative terms

affecting the agent's actions. The deviations due to bit-walk are set to be dependent on formation being drilled, current inclination, and drilling speed. After learning for about 300,000 episodes, the agent is tested on a new wellpath that was not encountered by the agent during training. One such test case is shown in Fig. 4.8. The performance of the agent is satisfactory but it can be seen that the deviations are large at high build rate sections.

The agent is now provided with the freedom to change to drilling speed (within specified range). The state space and reward structure being the same, the agent was set up for training for 300,000 episodes. Testing on the same trajectory as that of previous case, the agent performed much better as can be seen in Fig. 4.9. The agent learnt to reduce drilling speed at the bends (or the sections where build rate is high) thus achieving minimal

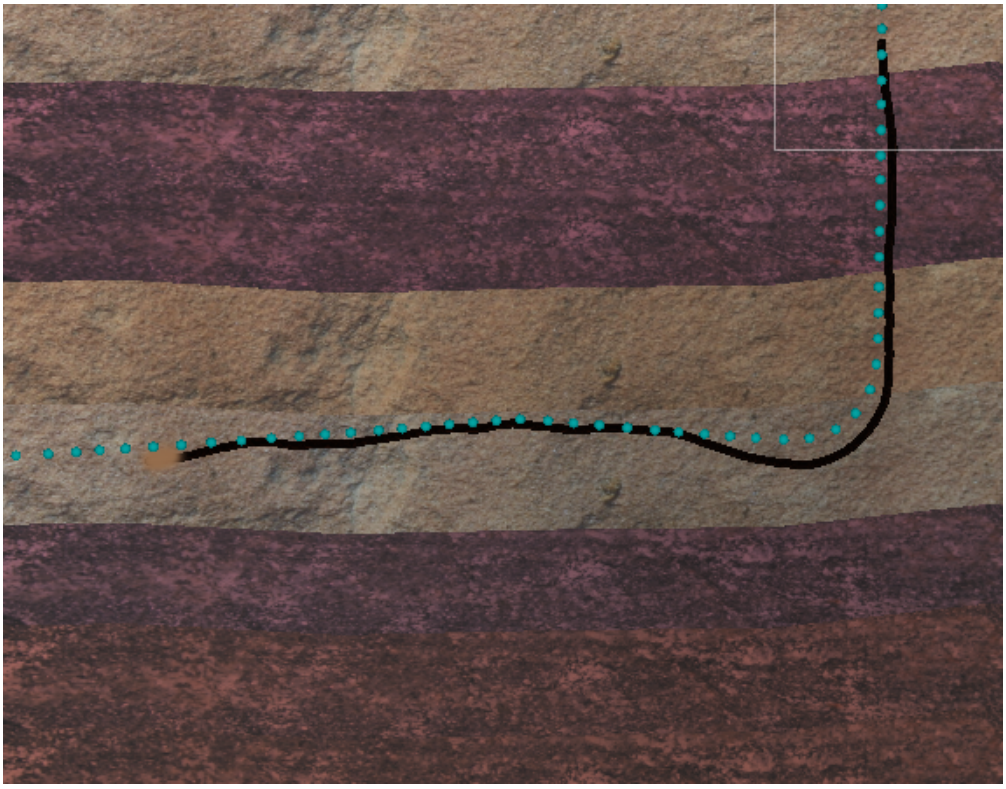

Figure 4.8: Trained Model put to test on a new trajectory shows the agent suffering deviations due to Bitwalk uncertainties

deviations.

## 4.4 Performance Evaluation

Statistics pertaining to PPO implementation as discussed above are assessed to understand the learning process. The training process information is collected and visualized using a Tensorflow utility called Tensorboard. The performance evaluation metrics are discussed below.

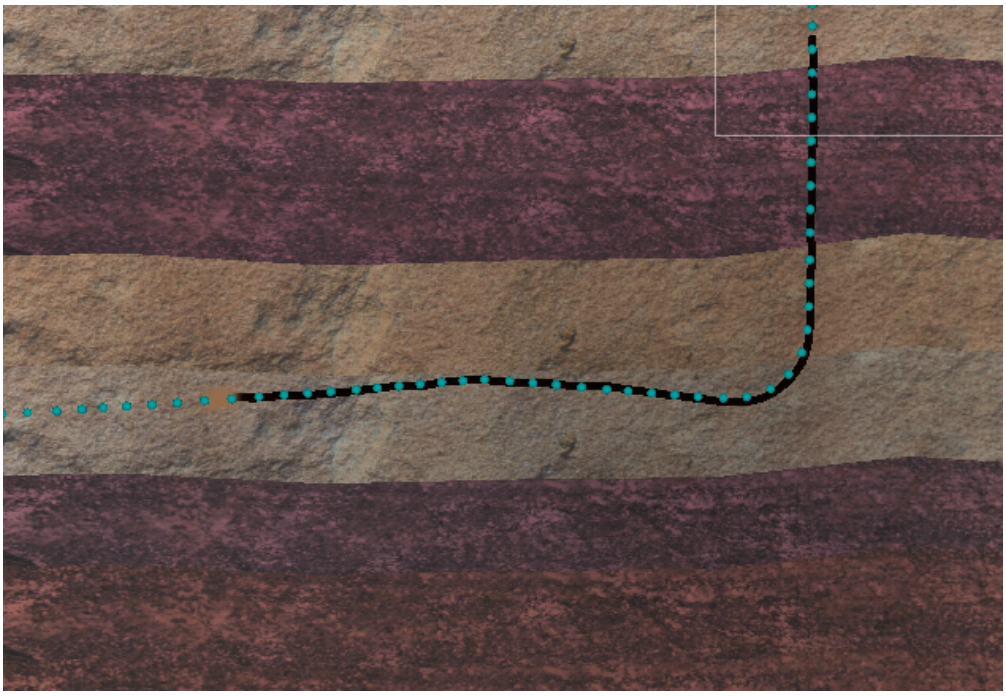

Figure 4.9: Model learned to accurately follow trajectory by slowing down at high build sections after the action space is expanded to include speed control

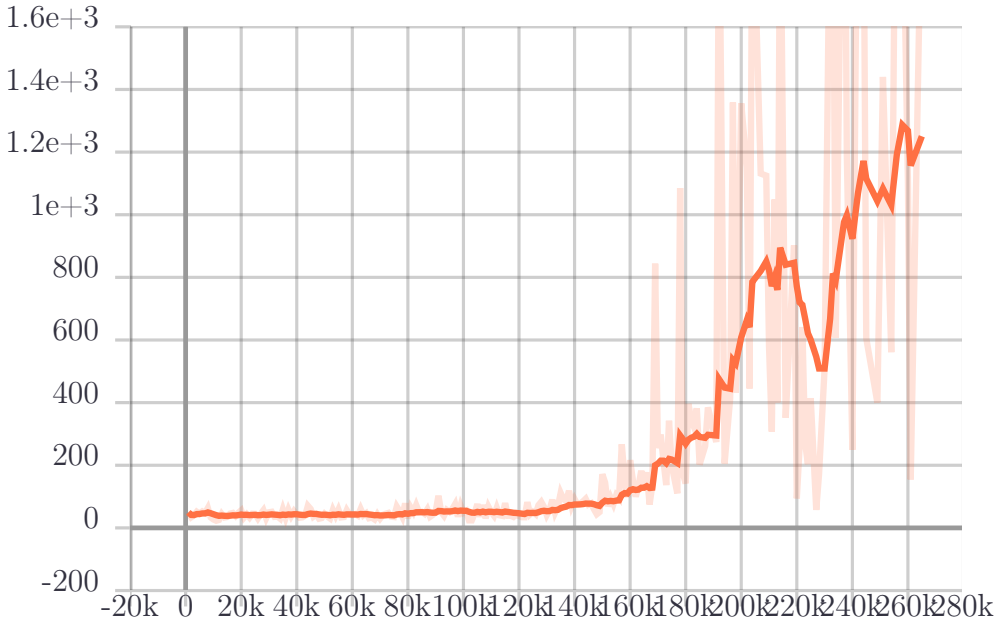


Figure 4.10: Cumulative Reward (vertical axis) is seen to consistently increasing in each episode (horizontal axis) showing the success of learning process

The first metric, shown in Figure 4.10 represents the mean cumulative episode reward over all agents. An increasing trend in the cumulative reward signifies a successful training session. Ups and downs in the cumulative reward value during the training is expected. Given the complexity of the process, significant increase in the reward does not occur until after 180,000 episodes.

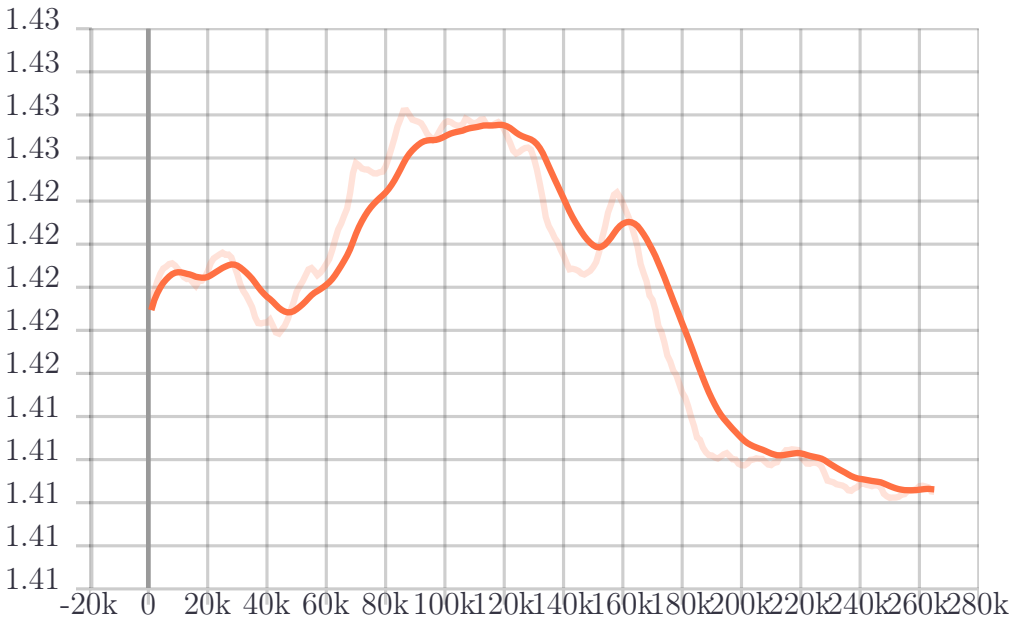


Figure 4.11: Entropy (vertical axis) follows the desirable behavior of decreasing over time in each episode (horizontal axis)

Figure 4.11 shows the entropy of the agent in each episode of the training process. Entropy represents how random the actions taken by the agent are. The consistent decreasing behaviour seen here is desirable, as opposed to too quick drop to too slow changes as it would mean the *Brain* taking random decisions.

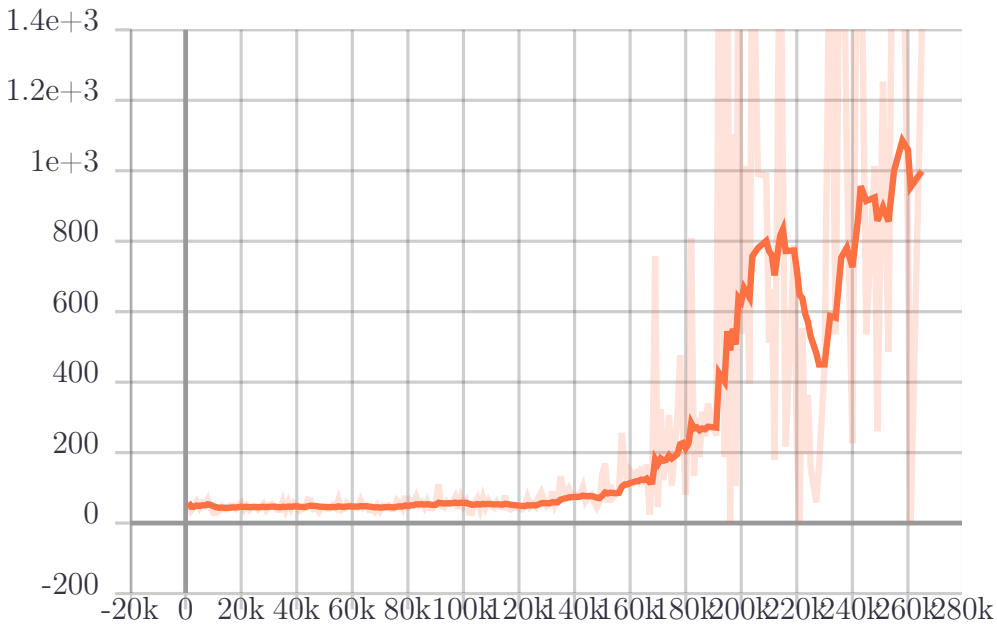


Figure 4.12: Number of steps before terminal step (vertical axis) of each episode (horizontal axis) are seen to be low in the early stage while exploring and to be large as the agent learned to take higher number of correct actions

Figure 4.13 shows the length, in terms of time steps for each episode over the course of training. As expected, in the initially stages the episode length is shorter because the agent is exploring more. As the learning stabilizes, similar episode lengths are seen. This means the agent is performing more of exploitation and terminal states are not encountered early in the episode.

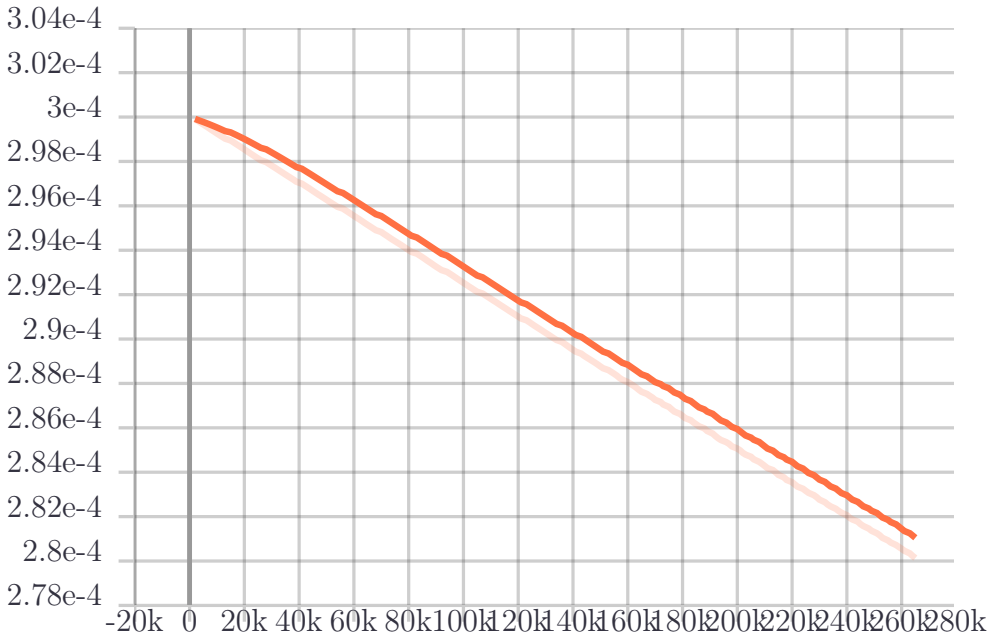


Figure 4.13: Learning rate (vertical axis) in each episode (horizontal axis) shown to be starting at the rate setup in PPO configuration and is gradually decreased as the updates need to be less frequent

Learning rate represents how large a step the training algorithm takes as it searches for the optimal policy. In a successful training process, like in our case, the learning rate as shown in Figure 4.13 decreases in a linear schedule. In addition, advanced parameters like KL convergence and Clipping fraction for PPO implementation are used to diagnose the learning process.

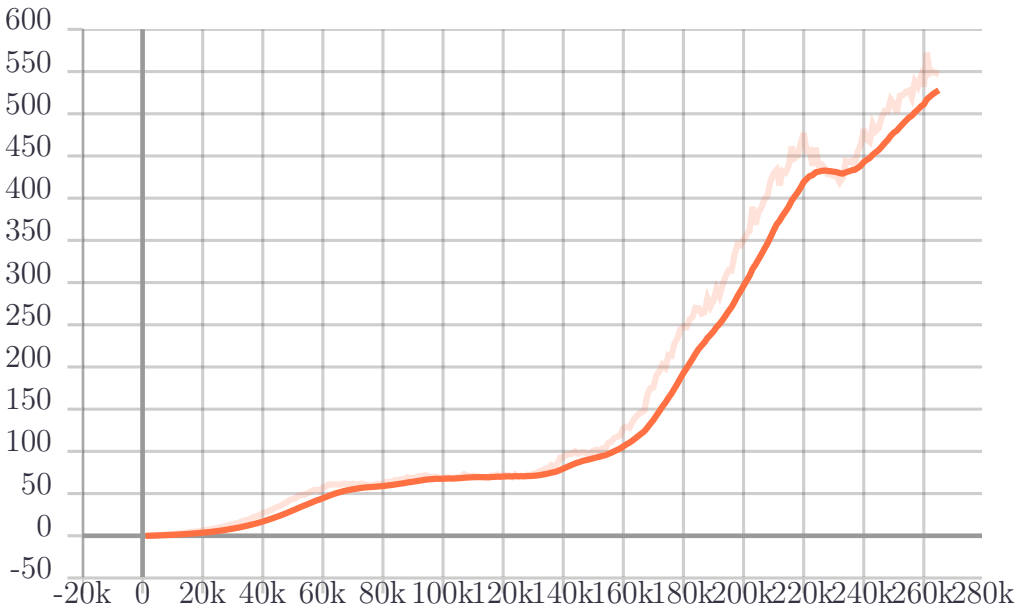


Figure 4.14: Value Estimate (vertical axis) in each episode (horizontal axis) shows the desirable behavior of consistent increase over time.

Figure 4.14 shows the mean value estimates of all states visited by the agent. These values should increase over each episode as the cumulative reward increases. They correspond to how much future reward the agent predicts itself receiving at any given point.

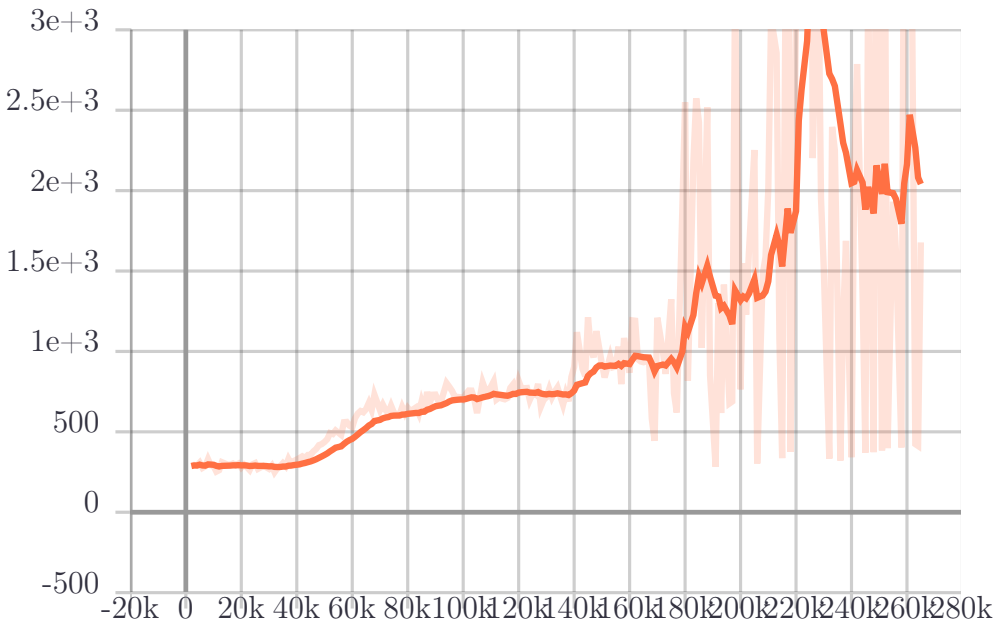


Figure 4.15: Value Loss (vertical axis) in each episode (horizontal axis) is seen to be increasing while the agent is still exploring and eventually dropping at around 230k episodes as the learning become more stable.

Figure 4.15 shows the mean loss of the value function update. This parameter correlates to how well the model is able to predict the value of each state. This should increase while the agent is learning (as the reward increases), and then decrease when the agent is exploiting more and once the reward becomes stable. The behaviour seen in the figure signifies a successful training session.

# 5. AUTONOMOUS SYSTEM: SELF-STEERING TRAJECTORY OPTIMIZATION

This chapter extends the scope of the steering problem from the previous chapter into a more autonomous system. The chapter starts with an overview of the problem and discussion of the scope of the problem attempted in this study highlighting the limitations and assumptions of the models. A major component of the autonomous downhole system is the availability of information regarding the formation type current being drilled at a given depth. Supervised machine learning classification methods used to identify rock type while drilling are presented. Results of the classification models and validation on lab-scale drilling test data are shown. Similar to Chapter 4, but with updated goals and constraints, MDP formulation is shown for the autonomous self-steering drilling system. Reward shaping approaches are explained in detail. Learning is done in four separate cases with different objectives. Implementation is done on ML-Agents platform within the custom-built Unity simulation environment. The PPO algorithms have been presented and the learning behaviour from the implementations are discussed. The performance evaluation of training process is done and presented using different metrics.

## 5.1 Overview

In Chapter 4, the goal was to accurately traverse the wellpath trajectory predetermined by the drilling engineers during planning phase. Naturally, this comes with an assumption that the wellpath provided during the planning phase is accurate to drill hydrocarbons from the well efficiently. Course corrections made while drilling are assumed either to be non-existent or that they are instantaneously available to the agent while drilling. Assuming that the predefined wellpath is accurate may not be feasible in the real-world scenario as there are several uncertainties such as formation tendencies, measurement or estimate errors. In this chapter, an alternative approach is provided to autonomously steer a well, eliminating the need for continuous corrections while drilling. The agent is shown to start with a base

plan and then react to formations, deviations, and other uncertainties at each time step while drilling. Given the geological information of a well, the RL agent learns to drill a quality wellbore with maximum contact with the target zone. The formation being drilled at a particular time step is identified using Gamma ray logs in the simulations shown in the chapter. Chapter 5.2 shows an alternative approach of identifying formation rock type using downhole drilling data. This approach will be later used in an integrated system that is not in the scope of this work, therefore only gamma ray logs are used. There are certain assumptions to this model setup as well. These assumptions are listed below.

- Geological logs are accurate and are alone sufficient to identify a formation type and distinguish from target zone
- Continuous inclination & Gamma measurements are near-bit, accurate, and available in real-time
- Drilling Dynamics do not impact the steering process
- Cuttings accumulation do not impact the steering process
- Bit-walk is induced into the actions as a Gaussian error
- Rotary-Steerable Systems (RSS) are in use for geosteering

### 5.1.1 Problem Scope

Despite the very general nature of this research problem, the scope of the work is limited. Firstly, surface operations such as drill pipe connections are ignored and the drilling agent is always assumed to be on-bottom. In real-world scenario, this assumption implies that operations such as tripping, reaming, fishing, and back-reaming are not considered. Second, only geosteering and wellbore placement are considered, therefore excluding actions related to well construction, casing, cementing, drilling performance, formation strength evaluation, etc. Drilling performance optimization is considered separately later in Chapter 6 and a

framework for integration is provided but is not in the scope of this chapter of autonomous geosteering. Next, the models are limited to conditions in which annulus is not closed and where a single drilling fluid is used. Because of this assumption, operations such as well control, managed pressure drilling, dual gradient operations are not considered. Overall, the steering operation is treated as a subsystem with minimum or no interference from other sub-systems while drilling. Nevertheless, the proposed solution should be valid for:

- Land rigs, fixed platforms, or floaters
- Any depth ranges and well shapes
- Simple or tapered drill-strings and wellbore architectures
- Heterogeneous formations and thin reservoirs
- Low and high bandwidth telemetry communication methods

## 5.2 Rock Formation Identification Downhole

The proposed self-steering models take into account the availability of downhole drilling data including directional Measurement-While-Drilling (MWD) data as well Logging-While-Drilling (LWD) data in real-time. LWD data, specifically Gamma ray logs are used to identify the type of rock formation currently being drilled. An alternate approach considered here is to estimate the rock formation type from downhole parameters excluding LWD data. The motivation behind this approach is to achieve both computational efficiency as well as cost advantage by excluding gamma ray sensors in the Bottom-Hole-Assembly (BHA).

Data-driven techniques like supervised machine learning classification algorithms are used to carry out a layer-based determination and change detection of properties of rock formation currently being drilled in near real-time. Real-world drilling data obtained through lab-scale drilling of multiple formations is used to train, test and validate the models. Output of the models will be later fed into an integrated RL geosteering system. Details of the rock formation identification models are discussed below.

| Formation ID | 1 | 2 | 3 | 5 | 6 | 8 |
|---|---|---|---|---|---|---|
| # Observations | 13217 | 805 | 899 | 45 | 28 | 5214 |

Table 5.1: Distribution of data points for different formation types drilled shows data pertaining to formations 5 & 6 is limited

**Data Source**

A lab-scale autonomous rig was built with an objective of addressing several issues in drilling operations with automation for the SPE Drillbotics competition, an international student competition designed to accelerate the uptake of automated drilling systems [125]. The rig acquires data from eight different analog and digital sensors. A laser-based sensor measures the traveling block distance and its position related to the top of the structure. A Wheatstone bridge load-cell measures the tension on the draw- works line that is generated by the total weight of the top drive motor and the overall assembly. The difference between the hook load tension and the original weight is used to get an approximation of the weight on the bit (WOB). Triaxial accelerometers are mounted at the BHA to record vibration data. Several experiments were conducted on the rig and data was generated by drilling into multiple formations. Raw data is acquired from the above mentioned sensors at 2000 Hz which is then downsampled and filtered to 10 Hz. A total of 115 predictors including measured and derived parameters are used for training the models. Fig 5.2 shows the various predictor parameters used for training. A total of 9 different formations (numbered 1 through 9) have been drilled during the experiments. The data points with Formation Type 4, 7, and 9 were removed as there was only seen 8, 3, and 1 data points out of a total of 20,220 data points available. The remaining data is distributed across the formations as shown in Table 5.1.

| Input Parameter | Response |
|---|---|
| 1. Depth (in) | |
| 2. Program Run time (estimate in mills) | Rock Formation Type |
| 3. WOB (bf) | |
| 4. RPM | |
| 5. Torque (in-lb) | |
| 6. Avg ROP (over dx data points) (ft/hr) | |
| 7. DOC (mm/cutter) | |
| 8. Z vibrations (G's) | |
| 9. X Vibrations (G's) | |
| 10. Y vibrations (G's) | |
| 11. Rig displacement (mm) | |
| 12. Hookload (lb) | |
| 13. UP Vib Z (G's) | |
| 14. Instantaneous MSE (psi) | |
| 15. MSE (psi) | |
| 16. Desired WOB setpoint | |
| 17. Desired RPM setpoint | |
| 18. Z ACC mean | |
| 19. Z ACC RMS | |
| 20. Z ACC Max | |
| 21. Z ACC Min | |
| 22. Z ACC STD | |
| 23. Z ACC Kutosis | |
| 24. Z ACC Skewness | |
| 25. Z ACC Variance | |
| 26. Z ACC MAD | |
| 27. Z ACC IQR | |
| 28. Z ACC Energy | |
| 29. Z ACC Entropy | |
| 30. Z ACC Dominant Frequency | |
| 31. Z ACC Magnitude | |
| 32-45 X ACC | |
| 46-59 Y ACC | |
| 60-73 Z Jerk | |
| 74-87 X Jerk | |
| 88-101 $Up_Y$ Jerk | |
| 102-115 Gyro/Magnetometer data | |

Figure 5.1: List of predictor parameters used for classification that includes surface measurements, downhole measurements, and derived parameters

### ML Implementation

Identification of rock formation type is done using Machine Learning classification techniques. Different algorithms have been implemented and compared (Fig. 5.2). Firstly, the most important predictors are identified and separated from the total 115 variables. Modeling is done on the top 10 predictors instead of the entire list of predictors to avoid issues like overfitting while training. Two measurements of variable importance were used as a criterion to determine the most important predictors. The first was Mean Decrease Accuracy (MDA), also known as permutation importance. This factor shows the decrease of accuracy for each predictor terms on the out of bag errors when a given variable is excluded from the model. The second one is the mean Gini index, which illustrates a decrease in node impurity as each variable is excluded from the training set.

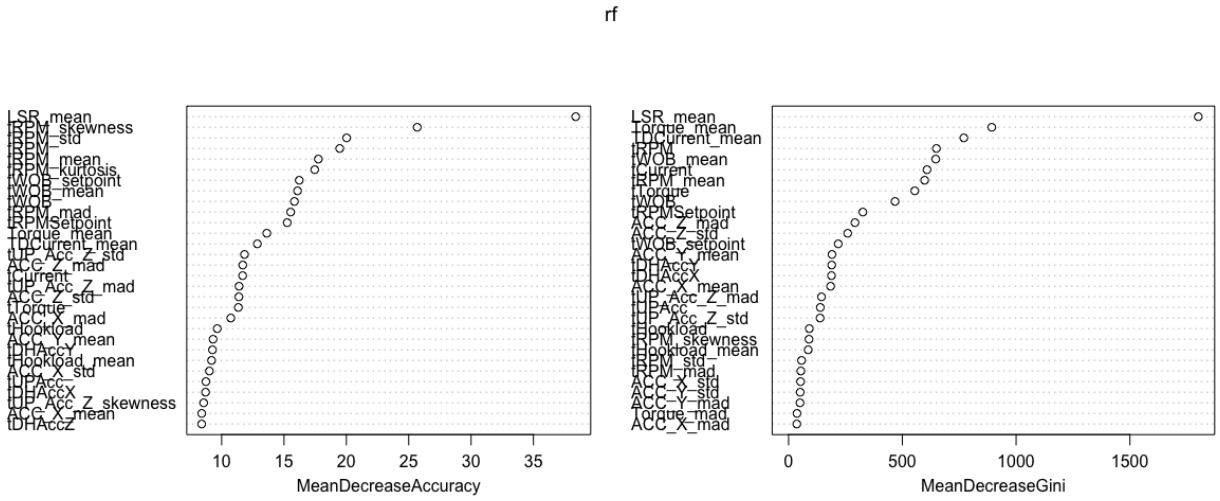


Figure 5.2: Importance of predictors according to a Mean Decrease Accuracy (left) and mean GINI indexes (right).

After separating out the training and test datasets, experiments were performed with the below listed classifier algorithms using the 10 most important predictors. Results ML models are presented next. Table 5.2 shows the confusion matrix obtained from Random

Forests Model. Similar results are obtained with the other techniques and a summary of the accuracy results are shown in Table 5.3.

- Logistic Regression (LR)
- Linear Discriminant Analysis (LDA)
- Support Vector Machines (SVM)
- Random Forests (RF)
- Artificial Neural Networks (ANN)

| 0 | 1 | 2 | 3 | 5 | 6 | 8 | Class error |
|---|---|---|---|---|---|---|---|
| 1 | 13215 | 2 | 0 | 0 | 0 | 0 | 0.0001513203 |
| 2 | 2 | 801 | 2 | 0 | 0 | 0 | 0.0049689441 |
| 3 | 0 | | 894 | 0 | 0 | 1 | 0.0055617353 |
| 5 | 0 | 0 | 13 | 28 | 0 | 4 | 0.3777777778 |
| 6 | 0 | 1 | 0 | 3 | 23 | 1 | 0.1785714286 |
| 8 | 0 | 0 | 0 | 1 | 0 | 5213 | 0.0001917913 |

Table 5.2: Confusion matrix using the 10 most important predictors in Random Forest model

| **Algorithm** | LR | LDA | SVM | RF | ANN |
|---|---|---|---|---|---|
| **Accuracy** | 96.56% | 96.78% | 98.94% | 99.48% | 99.58% |

Table 5.3: Results from different machine learning methods using prominent predictors.

While the results are encouraging, it is worth noting that the dataset was limited to few, known formation types. Figure 5.3 shows results of a test case where four different formations (identified by different colors 1-4) are drilled. The algorithm was able to identify formation types 1, 3 and 4 accurately and 2 with some error due to similarity in properties of adjacent rocks. The models can be further improved with more experimental data involving complex formations encountered in real-world drilling. Training on such data will lead to the development of a more robust rock formation identification model.

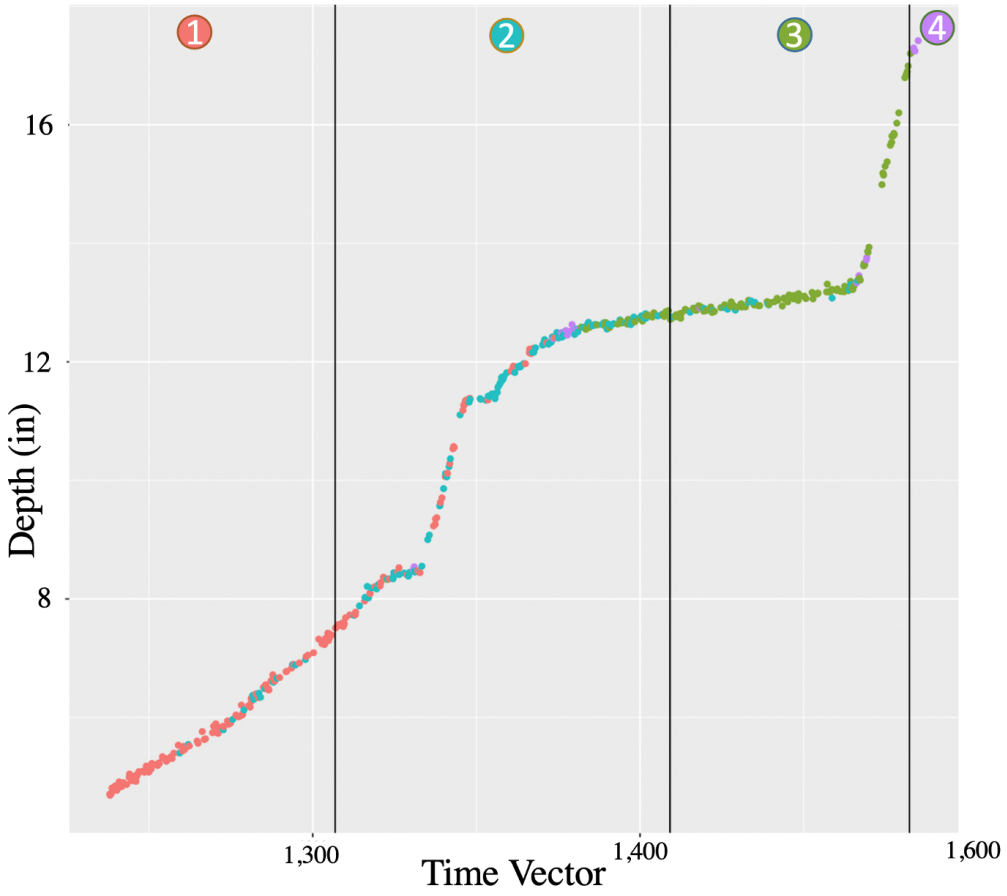


Figure 5.3: Neural network classifier using prominent predictors results during critical region where the algorithm was able to identify formation types 1 (red), 3 (green) and 4 (purple) accurately and 2 (blue) with some error due to similarity in properties of adjacent rocks

## 5.3 Model Setup

The Directional Drilling environment is modeled as a Markov Decision Process (MDP). The MDP is designed for an autonomous system where only the geological model data is provided to the drilling agent. The agent learns to design a well plan that satisfies specific objectives. Reward scheme depends on the desired objectives. Although there could be different goals of a successful drilling operation, for the sake of this study, only the objectives pertaining to drilling a high-quality wellbore that maximizes contact with the target zone and also drills fast. The different components of the MDP framework - Agent, Environment, States, Actions, Rewards, and Goals are defined accordingly. Stochasticity is incorporated into the model by introducing noise into some of the actions. Details of the model setup are as below.

### Simulator

A simplified 2-D earth model is built in the Unity simulator as described in section 3.2. that represents different layers of formations with varying thicknesses, shapes, gamma and resistivity values (Fig. 5.4). A total vertical depth of 10000 ft. is used in the simulations. A thin oil layer is positioned towards the lower portion of the environment. Drilling starts from the surface (TVD = 0 ft.). Kickoff happens when the drill agent gets close to the target zone. Measured data 'at-bit' consists of LWD measurements of Gamma, MWD measurements of TVD and Inclination, and other parameters such as dogleg and timestamp.

*Domain randomization*

Producing agents that can generalize to a wide range of environments is a significant challenge in reinforcement learning. One method for overcoming this issue is domain randomization, whereby at the start of each training episode some parameters of the environment are randomized so that the agent is exposed to many possible variations. With enough variability in the simulator environment, the real world may appear to the model as a close version of one of the domain variations. Without such randomization, the agent tries to

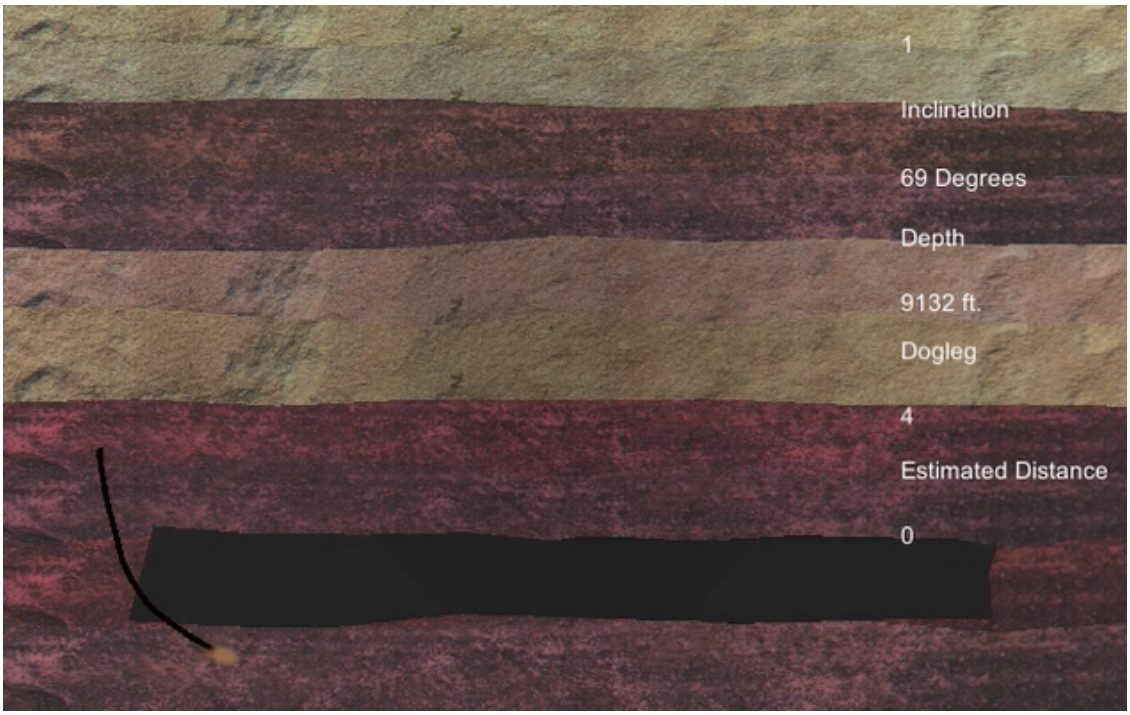


Figure 5.4: Snapshot of Simulator depicting formations with varied thickness and properties, and sample trajectory as drilled by the RL agent

over-fit and can not be general. In this particular setup, start positions at the surface, and KOP are randomized every episode.

**Agent**

The drilling agent in this setup mimics the directional driller on the rig surface and also an operational geologist. The goal of the agent is to reach the oil zone as per the provided geological models and to keep drilling in the zone until the end of the target zone. In the real-world (Fig. 5.5), the driller, directional driller, drilling engineer, and operational geologist work together to achieve drilling in the target zone with minimal tortuosity and faster operations. The agent works towards achieving the same goals. At each step of an episode, the agent observes the states and takes actions so as to maximize the rewards in achieving this goal. RL is long-term goal oriented.


Figure 5.5: Snapshot of Real-time Geosteering operation monitoring highlighting the tortuous wellpath (red curve represents the actual trajectory drilled and the dots on the curve are survey points) in the target formation (represented as yellow strip in the center. *(Source: Petrolink)*

### Environment

The observations that the agent makes in the environment which define the states are as listed below.

**State Space** ($\mathcal{S}$)

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth
- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Gamma value of the formation ($\gamma$)
- Resistivity of the formation ($R$)
- Dogleg Severity ($dg$)
- Wellbore Tortuosity ($\mathcal{T}$)

It is assumed that all the above observations are available in real-time while drilling. In the real-world, some of the sensors are located behind the bit, sometimes about 60 ft. farther. But for the simulations in this model, and for the rest of the chapters unless it is specified otherwise, it is assumed that the observations are made near-bit.

**Action Space**

A continuous action space is defined for the model where the agent can steer left or right adhering to mechanical constraints. Two different scenarios - one with a constant drilling (or Steering) speed and the other with drill speed as an action are implemented. The idea behind having steering speed as an action is to simulate and observe the behaviour when hard formations are encountered.

**Action Space** ($\mathcal{A}$)

- Steer: $[-6°, +6°]$ ($\dot{\theta}$)
- Throttle (Case 1): constant speed ($v$)
- Throttle (Case 2): variable speed [1,10] ft/s ($v$)

**Stochasticity**

Uncertainties of the real-world drilling environments such as hard formations, bit walk are incorporated into the simulator by intentionally adding Gaussian noise to the actions at each step of the episode. For the case of variable speed control, the added noise is a function of both gamma and drill speed.

**Rewards**

Rewards are defined for the agent in accordance with the objectives. Consequently, the agent gets reward when it drills in the oil zone and no reward anywhere else in the formations. In order for the agent to move forward, an additional reward is given as it gets closer to the end of the target. The agent is penalized for dogleg to avoid tortuosity in the well path.

**Reward Function** ($\mathcal{R}$)

- $s_t$ : state at time=$t$, $(x_t^{pos}, y_t^{pos}, \theta_t, \gamma, dg, d_t^{oil})$
- $a_t$ : action at time=$t$, $(\dot{\theta}_t, v_t)$
- $R(s_t, a_t, s_{t+1})$ : reward obtained on reaching $s_{t+1}$ after executing $a_t$ at $s_t$

$$\mathcal{R}(s_t, a_t, s_{t+1}) : \begin{cases} \{0, +1\}, & \text{if } s_{t+1} = \text{drilling in target zone} \\ k_1(d_t^{oil}), & \text{if } s_{t+1} = \text{drilling towards end} \\ -k_2(dg), & \text{if } s_{t+1} = \text{steering in bends} \\ +0.0001, & \text{if } s_{t+1} = \text{drilling ahead} \\ Terminal, & \text{if } s_{t+1} = \text{moving in a circle} \end{cases}$$

## 5.4 DRL Implementation

Due to the nature of the drilling operation, the well path is divided into vertical and deviated sections. The agents starts by exploring formations around the KOP until it 'hits' the oil zone and then learns to stay in the zone. At each time step of an episode, the current state is observed, and an action is taken based on the policy. The action taken determines the next state and reward received. The process is continued and rewards are accumulated until a terminal state is reached. PPO, an on-policy algorithm is implemented on the system.

Learning for the agent is done in four stages each with individual objectives. This is done to understand the behaviour and impact of each of the variables involved in the geosteering process. The agent first considers a baseline case and complexities are added thereafter with relaxed assumptions. Model setup, reward structure, and results of each of the four stages are discussed next.

### 5.4.1 Setup 1: Staying in the Oil Zone

*Objectives:*

- The agent should reach the oil zone.

- The agent should drill through the oil zone by always staying within the zone.
- Drill until End of Target is reached.

*States* ($\mathcal{S}$)*:*

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth
- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Gamma value of the formation ($\gamma$)

*Stochasticity:*

- No Bit-Walk phenomenon

*Actions* ($\mathcal{A}$)*:*

- Steer: $[-6°, +6°]$ ($\dot{\theta}$)
- Throttle: constant speed ($v$)

*Rewards* ($\mathcal{R}$)*:*

- $\{0,+1\}$, 1 if $s_{t+1}$ = drilling in target zone
- $k_1(d_t^{oil}),$ if $s_{t+1}$ = drilling towards end

This model serves as a baseline for the drill agent to generate a well plan based on geological data. Ignoring deviations due to bit walk, and other heterogeneties in the formations, the agent was trained to drill to and through the oil zone. Dogleg severity was not taken into consideration in this scenario and speed was kept constant. The agent learned to reach the oil and stay in the oil zone. Since no rewards were considered for dogleg severity, as expected the trajectory seemed tortuous as can be seen in Fig. 5.6.

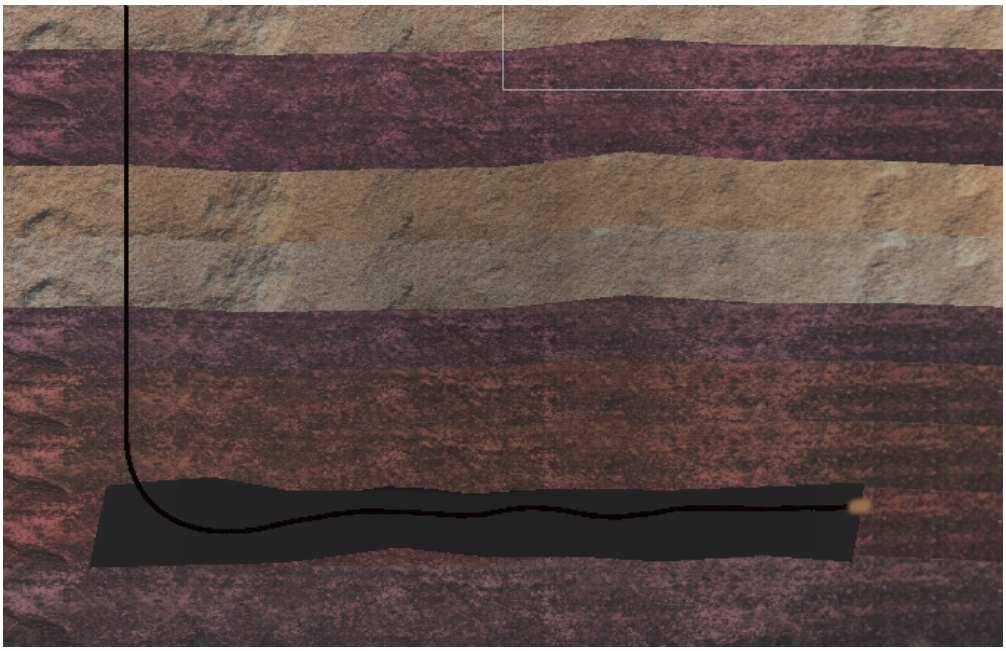

Figure 5.6: Model trained on the setup described in Stage 1 - shows the agent learning to drill through target zone while always staying in the zone, but with tortuousity

### 5.4.2 Setup 2: Facing the Uncertainty

*Objectives:*

- The agent should reach the oil zone.
- The agent should drill through the oil zone by always staying within the zone.
- Drill until End of Target is reached.

*States* ($\mathcal{S}$)*:*

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth
- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Gamma value of the formation ($\gamma$)

*Stochasticity:*

- Bit-Walk phenomenon is introduced

*Actions* ($\mathcal{A}$)*:*

- Steer: $[-6°, +6°]$ $(\dot{\theta})$
- Throttle: constant speed $(v)$

*Rewards* ($\mathcal{R}$)*:*

- $\{0,+1\}$, 1 if $s_{t+1}$ = drilling in target zone
- $k_1(d_t^{oil}),$ if $s_{t+1}$ = drilling towards end

This model represents one added complexity to the previous baseline setup. Bit walk deviation is introduced into the simulation. In the real-world, bit walk or hole deviation in general, is a result of several factors like heterogeneous nature of formation and dip angle, BHA, Stabilizers (location, number, and clearances), Applied weight on bit (WOB), Hydraulics at the bit, Improper hole cleaning, to name a few. It is extremely hard to model the bit walk. It is added as Gaussian noise to the steering actions taken by the agent. The model is setup with same objectives as those of setup #1 above. The agent failed to learn a good behaviour as shown in Fig. 5.7 to stay in the oil zone and to keep drilling until end of target section. The actions helped the agent reach the target zone and drill through it but the trajectory was highly tortuous. The reason behind this behaviour is due to the fact that the rewards scheme in this setup does not reward or penalize dogleg severity.

### 5.4.3 Setup 3: Address Tortuousity

*Objectives:*

- The agent should reach the oil zone.
- The agent should drill through the oil zone by always staying within the zone.
- Drill until End of Target is reached.
- Minimize dogleg severity

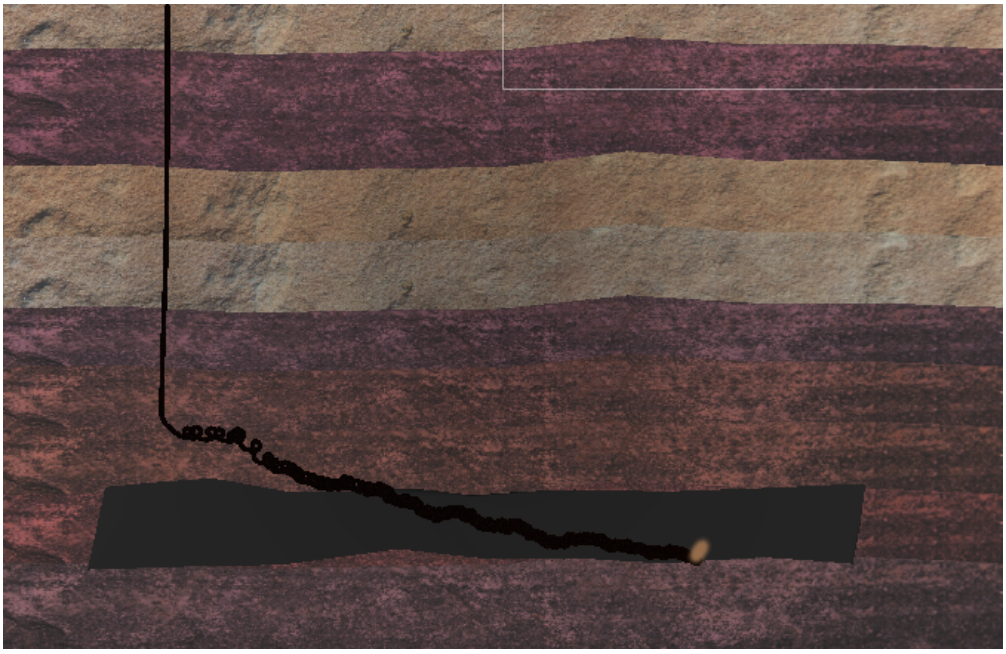

Figure 5.7: Model trained on the setup described in Stage 2 - shows the agent learned to reach and drill through target zone but resulted in high local toruosities

*States* ($\mathcal{S}$)*:*

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth
- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Gamma value of the formation ($\gamma$)
- Dogleg Severity ($dg$)

*Stochasticity:*

- Bit-Walk phenomenon is introduced

*Actions* ($\mathcal{A}$)*:*

- Steer: $[-6°, +6°]$ ($\dot{\theta}$)
- Throttle: constant speed ($v$)

*Rewards* ($\mathcal{R}$)*:*

- $\{0,+1\}$, 1 if $s_{t+1}$ = drilling in target zone
- $k_1(d_t^{oil})$, if $s_{t+1}$ = drilling towards end
- $-k_2(dg)$, if $s_{t+1}$ = steering in bends

The problem of high tortuousity encountered in the previous stage is addressed here by adding reward (or penalty) in proportion to the dogleg severity. The model learned to reduce the number of bends in the trajectory while in the oil zone and before reaching the end of target. However, this setup led the agent to learn to drill at high build rates as can be seen in Fig.5.8. This behaviour leads to reduction in the contact area of the wellbore with the target section that might have long-term impacts in the future. Rewards need to be shaped in order to reduce the build rate.

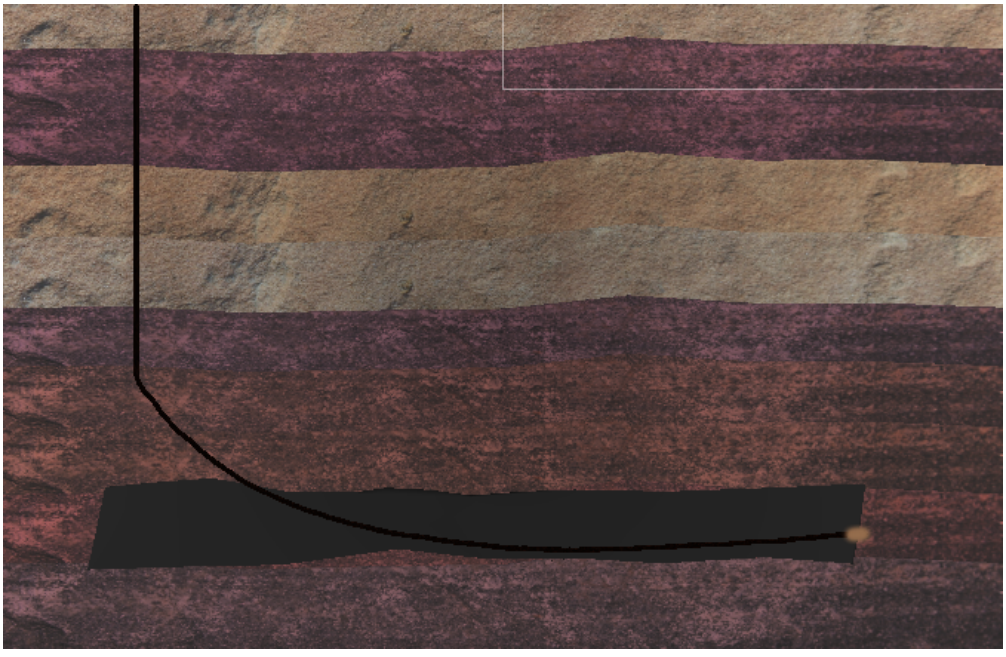

Figure 5.8: Model trained on the setup described in Stage 3 - shows the learned model minimizing local tortuosities while drilling through target zone but resulting in higher build-rate

### 5.4.4 Setup 4: Maximize Contact with High Quality Wellbore

*Objectives:*

- The agent should reach the oil zone.
- The agent should drill through the oil zone by always staying within the zone.
- Drill until End of Target is reached.
- Minimize dogleg severity

*States* ($\mathcal{S}$)*:*

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth
- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Gamma value of the formation ($\gamma$)
- Dogleg Severity ($dg$)

*Stochasticity:*

- Bit-Walk phenomenon is introduced

*Actions* ($\mathcal{A}$)*:*

- Steer: $[-6°, +6°]$ ($\dot{\theta}$)
- Throttle: variable speed [1,10] ft/s ($v$)

*Rewards* ($\mathcal{R}$)*:*

- $\{0,+1\}$, 1 if $s_{t+1}$ = drilling in target zone
- $k_1(d_t^{oil})$, if $s_{t+1}$ = drilling towards end
- $-k_2(dg)$, if $s_{t+1}$ = steering in bends

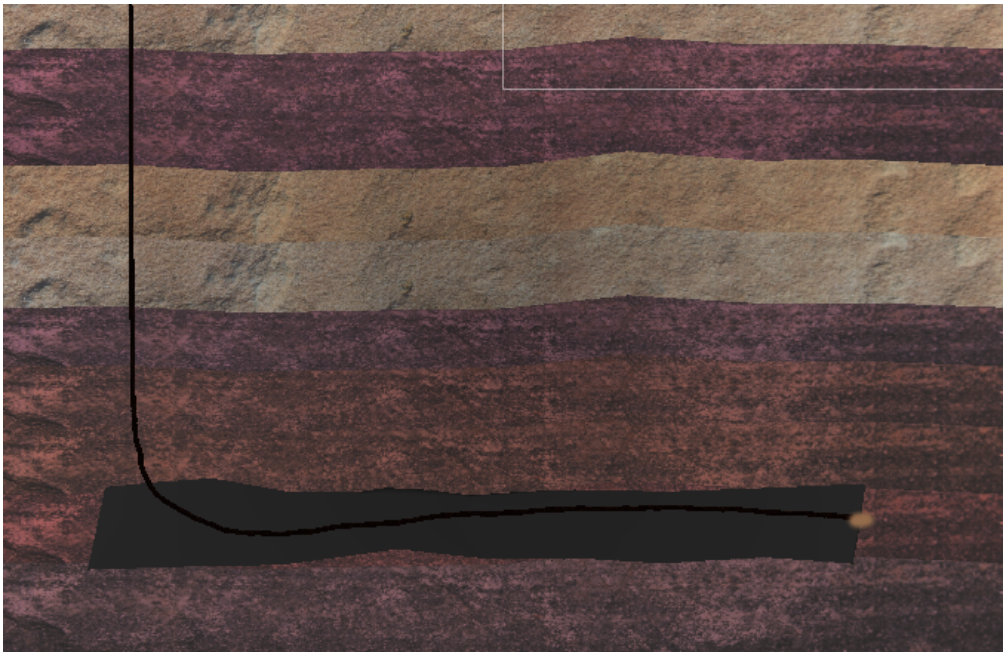

Figure 5.9: Model trained on the setup described in Stage 4 - shows the model is tuned and achieves the objectives of drilling through target zone maximizing exposure and minimizing toruosities

This trained model encompasses all the objectives that were predetermined for the drill agent. Specifically, the agent was able to drill through the target zone with minimum local tortuousities resulting in a high-quality wellbore (Fig. 5.9). Introducing speed control to the agent also helped in improving the contact area of the wellbore in the target section.

## 5.5 Performance Evaluation

Statistics pertaining to PPO implementation as discussed above are assessed to understand the learning process. The training process information is collected and visualized using a Tensorflow utility called Tensorboard. The metrics corresponding to the Stage 4 model setup are presented first followed by all of the stages combined.

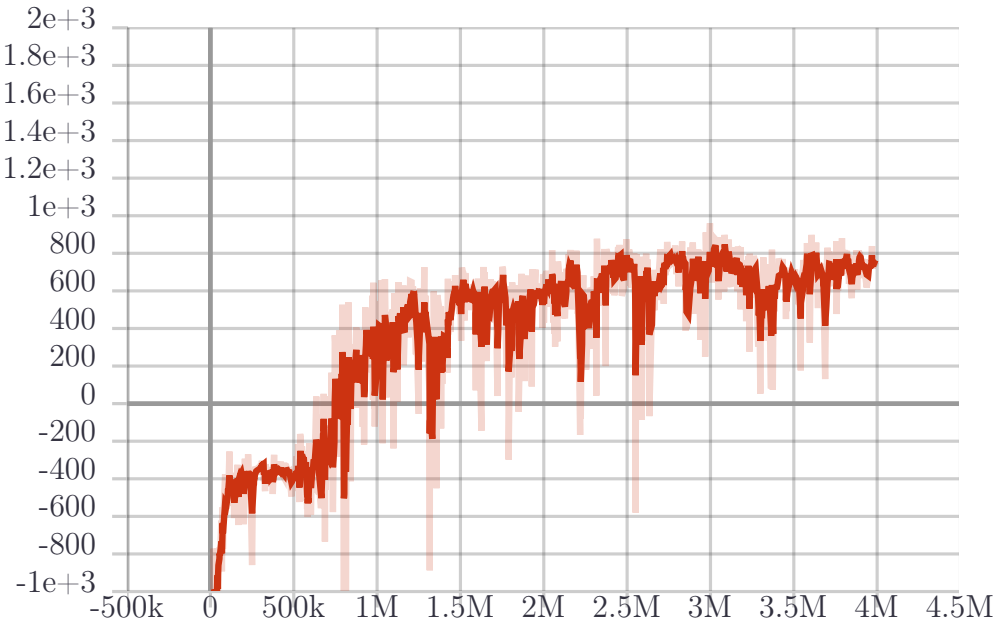


Figure 5.10: Stage 4 shows a successful learning process where the cumulative Reward (vertical axis) is seen to consistently increasing in each episode (horizontal axis)

The first metric, shown in Figure 5.10 represents the mean cumulative episode reward over all agents. An increasing trend in the cumulative reward signifies a successful training session. Ups and downs in the cumulative reward value during the training is expected. Given the complexity of the process, significant increase in the reward does not occur until after 1 million episodes.

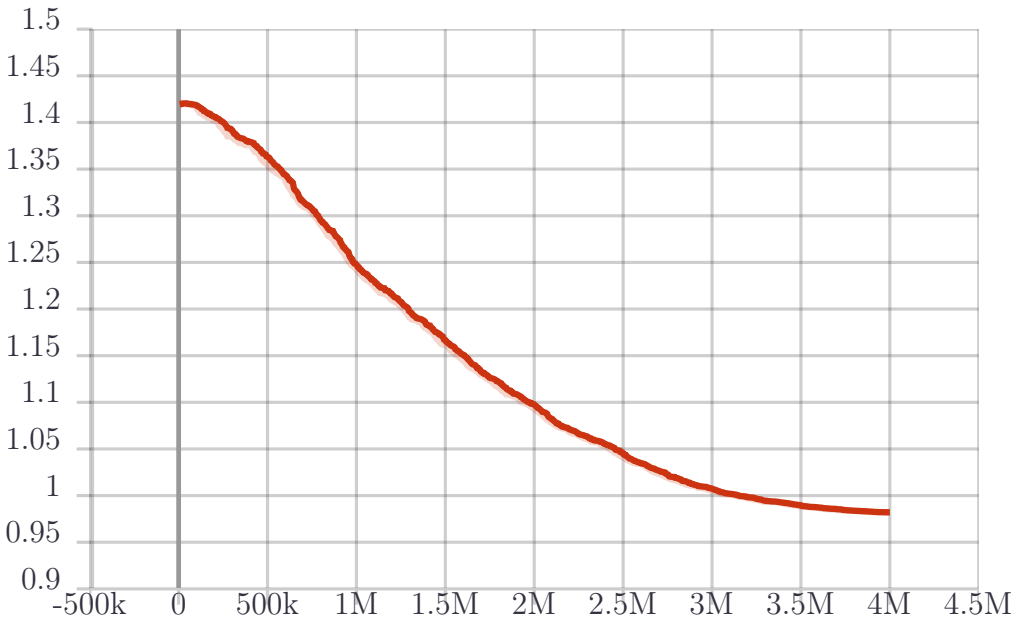


Figure 5.11: Stage 4: Entropy (vertical axis) follows the desirable behavior of decreasing over time in each episode (horizontal axis)

Figure 5.11 shows the entropy of the agent in each episode of the training process. Entropy represents how random the actions taken by the agent are. The consistent decreasing behaviour seen here is desirable, as opposed to too quick drop to too slow changes as it would mean the *Brain* taking random decisions.

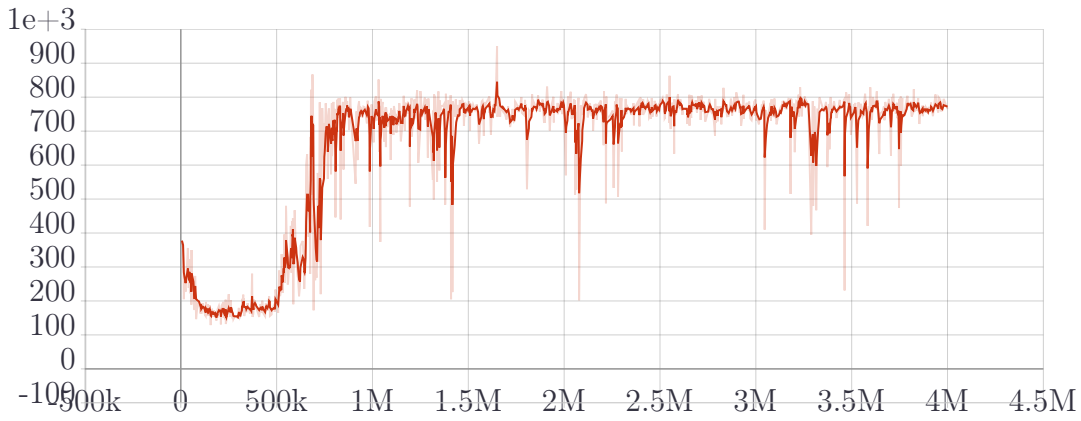


Figure 5.12: Stage 4: Length (vertical axis) of each episode (horizontal axis) are seen to be low in the early stage while exploring and to be large and consistent as the agent learned to take higher number of correct actions

Figure 5.13 shows the length, in terms of time steps for each episode over the course of training. As expected, in the initially stages the episode length is shorter because the agent is exploring more. As the learning stabilizes, similar episode lengths are seen. This means the agent is performing more of exploitation and terminal states are not encountered early in the episode.

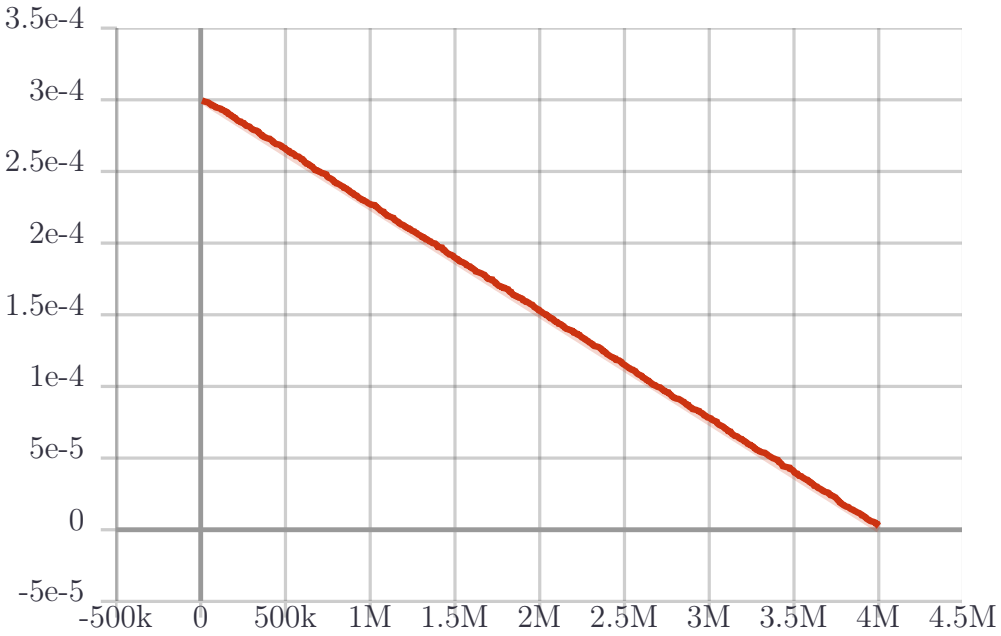


Figure 5.13: Stage 4: Learning rate (vertical axis) in each episode (horizontal axis) shown to be starting at the rate setup in PPO configuration and is gradually decreased as the updates need to be less frequent

Learning rate represents how large a step the training algorithm takes as it searches for the optimal policy. In a successful training process, like in our case, the learning rate as shown in Figure 5.13 decreases in a linear schedule. In addition, advanced parameters like KL convergence and Clipping fraction for PPO implementation are used to diagnose the learning process.

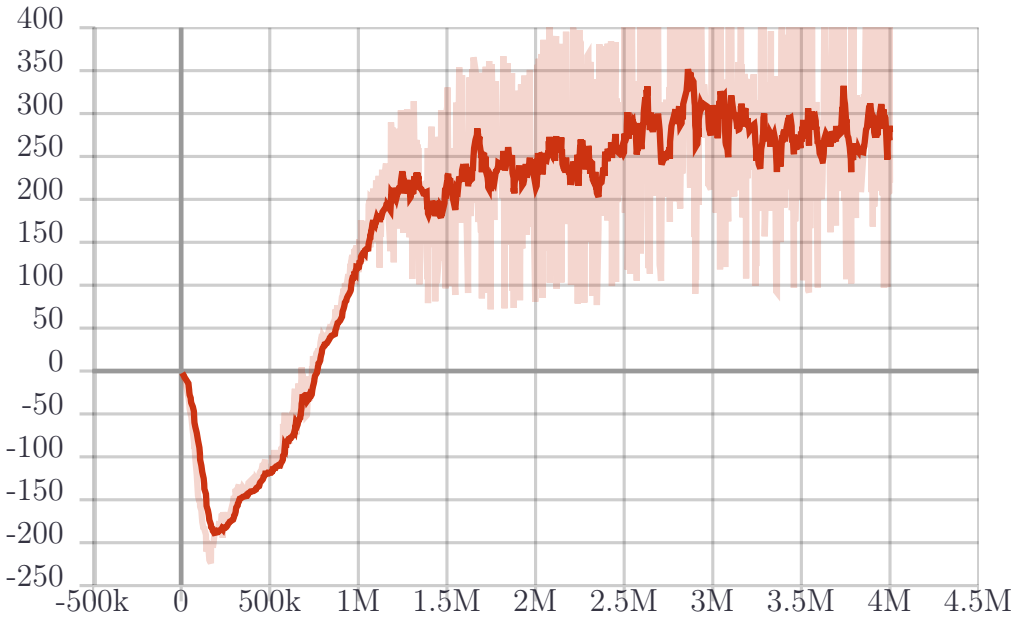


Figure 5.14: Stage 4: Value Estimate (vertical axis) in each episode (horizontal axis) shows the desirable behavior of gradual increase over time after the initial exploration phase.

Figure 5.14 shows the mean value estimates of all states visited by the agent. These values should increase over each episode as the cumulative reward increases. They correspond to how much future reward the agent predicts itself receiving at any given point.

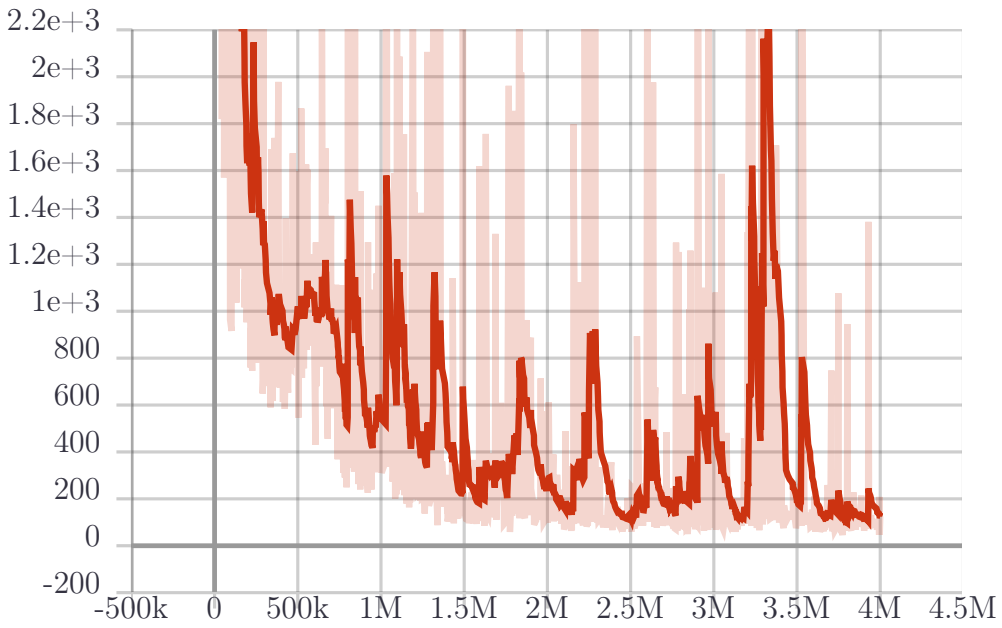


Figure 5.15: Stage 4: Value Loss (vertical axis) in each episode (horizontal axis) shows the desirable behavior of increasing while the agent is exploring and gradually decreasing over time.

Figure 5.15 shows the mean loss of the value function update. This parameter correlates to how well the model is able to predict the value of each state. This should increase while the agent is learning (as the reward increases), and then decrease when the agent is exploiting more and once the reward becomes stable. The behaviour seen in the figure signifies a successful training session.

Similar trends in the performance evaluation metrics are seen in the other stages. The four stages discussed in previous section and a failure case for Stage 1 (dark blue) is also added for comparison. Stage 1 is shown in pink, Stage 2 in green, Stage 3 in cyan, and Stage 4 in orange.

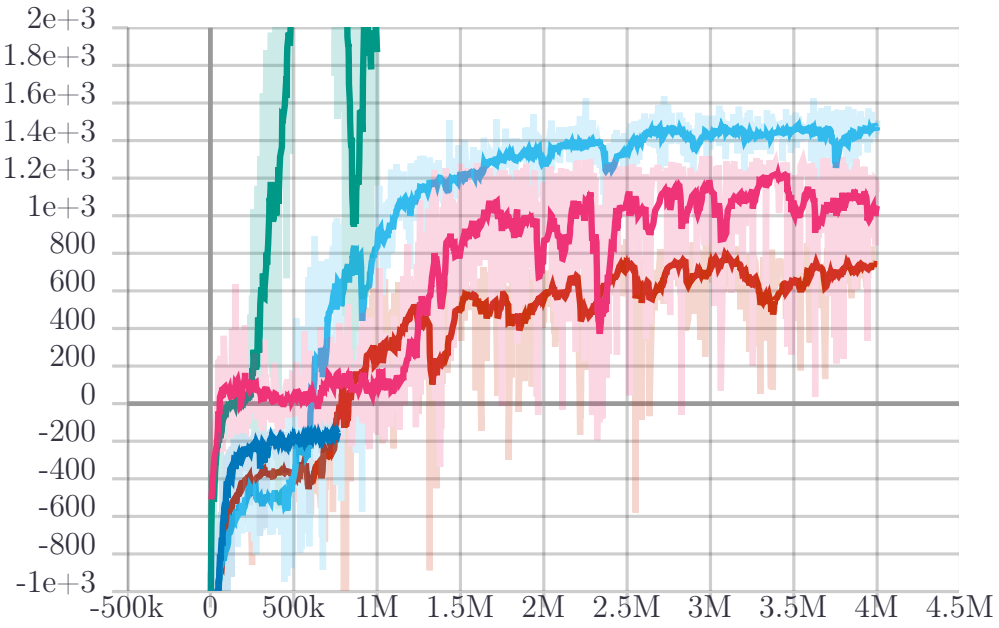


Figure 5.16: Cumulative Reward (vertical axis) in each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange) shows consistently increasing behavior for all stages except Stage 2

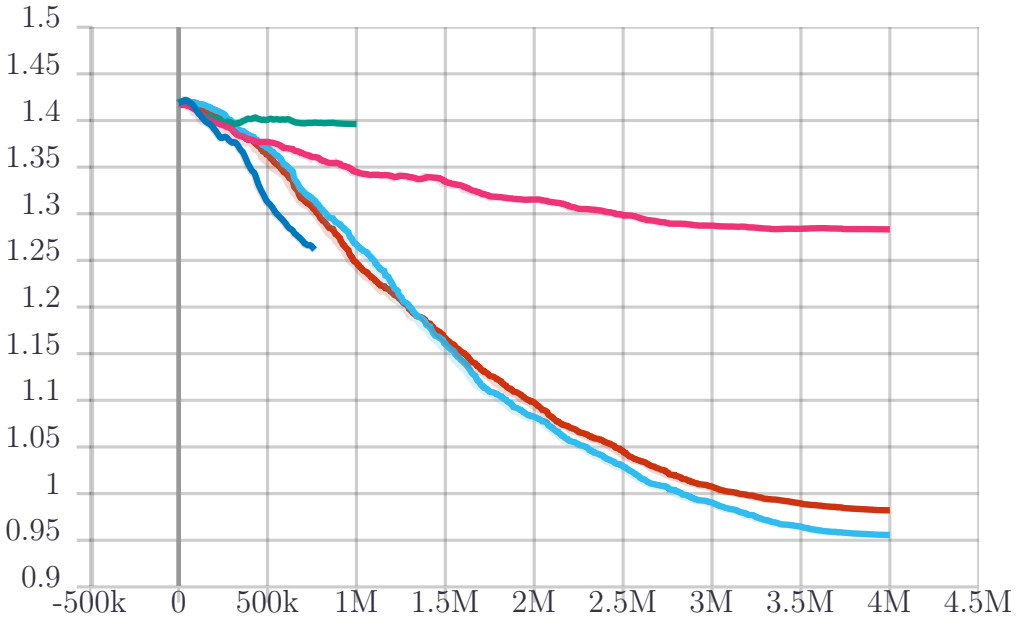


Figure 5.17: Entropy (vertical axis) in each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange) - Stage 2 (green) can be seen as flat due to high number of random actions

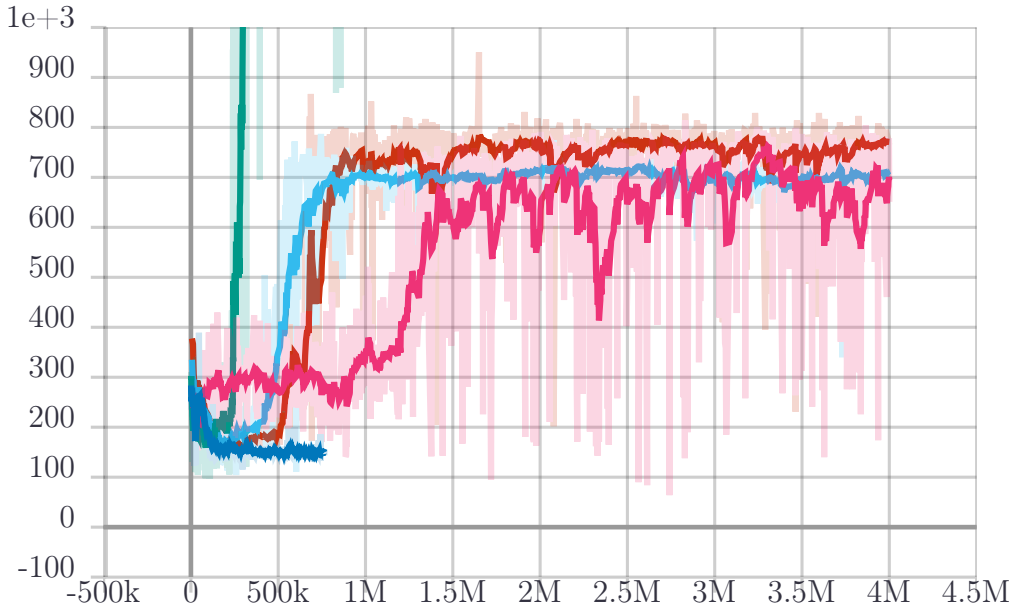


Figure 5.18: Length (vertical axis) of each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange) - Stage 2 (green) has large episode length due to relaxed terminal state condition

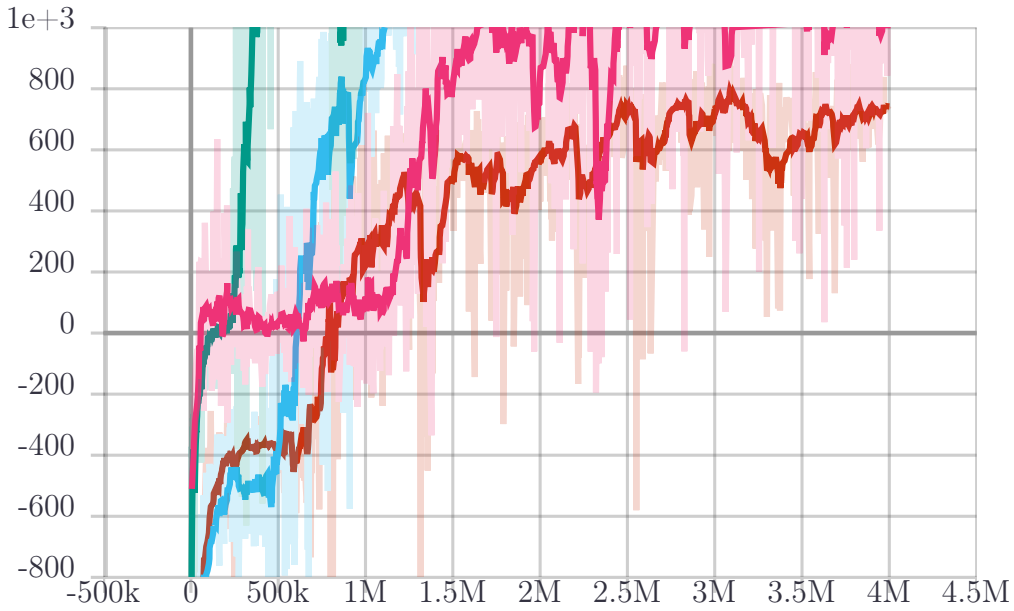


Figure 5.19: Extrensic Reward (vertical axis) in each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange)

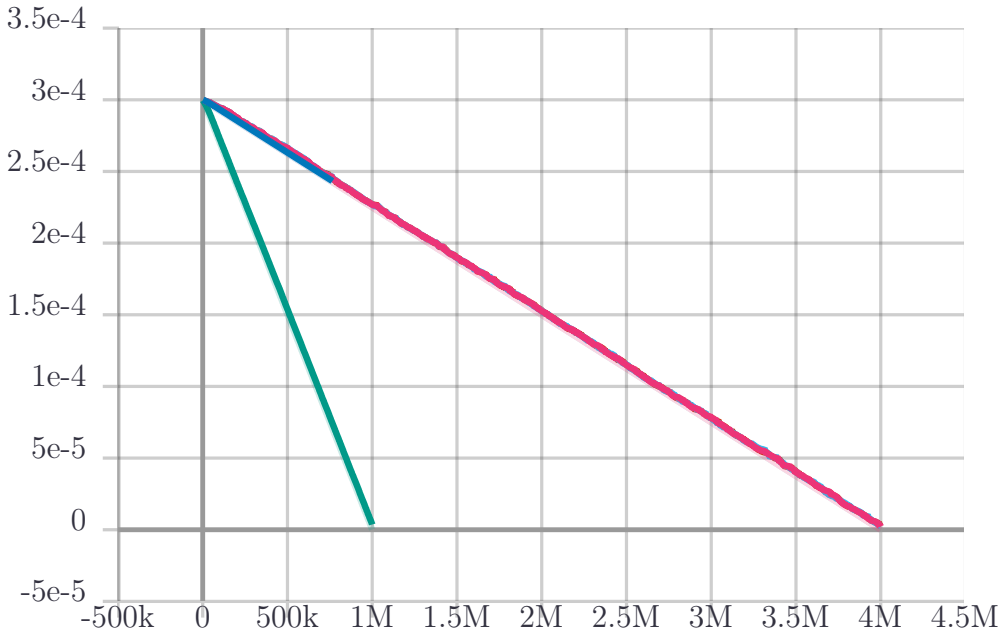


Figure 5.20: Learning rate (vertical axis) in each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange)

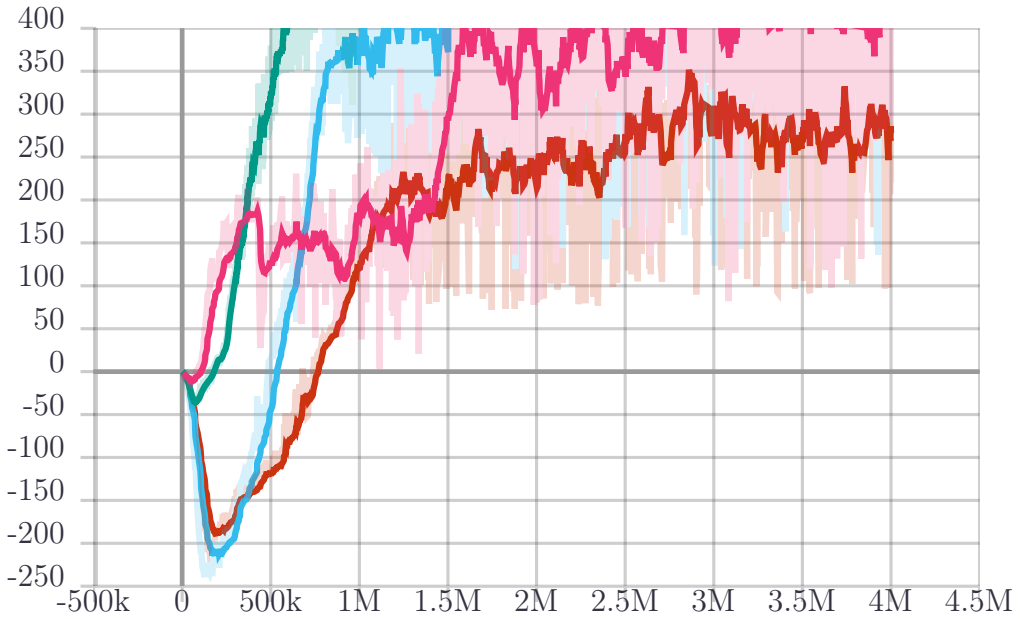


Figure 5.21: Value Estimate (vertical axis) in each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange)

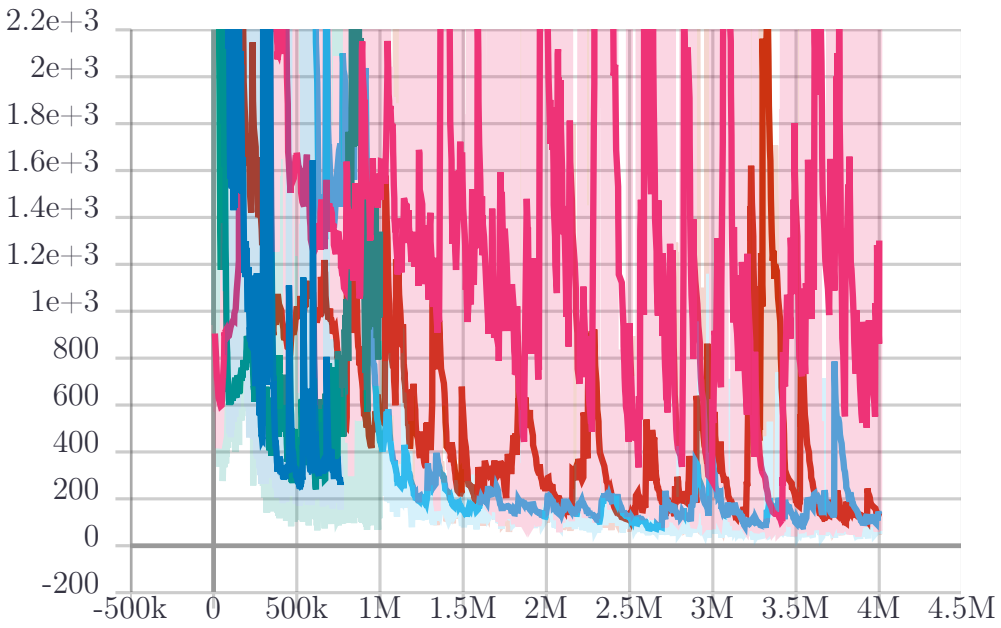


Figure 5.22: Value Loss (vertical axis) in each episode for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan), 4(orange) shows higher ups and downs due to exploration but overall decreasing trend shows improvement in learning over time.

*Histogram Visualization*

Cumulative rewards are shown in Figures 5.10 & 5.16 for Stage 4 and combined overlay of all stages respectively. A different way to visualize these cumulative rewards is using histogram representation. These visualizations provide intuition about whether a model is actually learning or not, and if it is, then how well it is learning. Figure 5.23 shows the histogram of cumulative reward. As can be seen in the figure, there are two clusters where rewards are concentrated. One on the left represents the period when the agent is exploring more and still learning and the one towards the right represents the behaviour when there is more exploitation and when the learning process is stablized. Similar representation is provided for other stages in Figure 5.24.

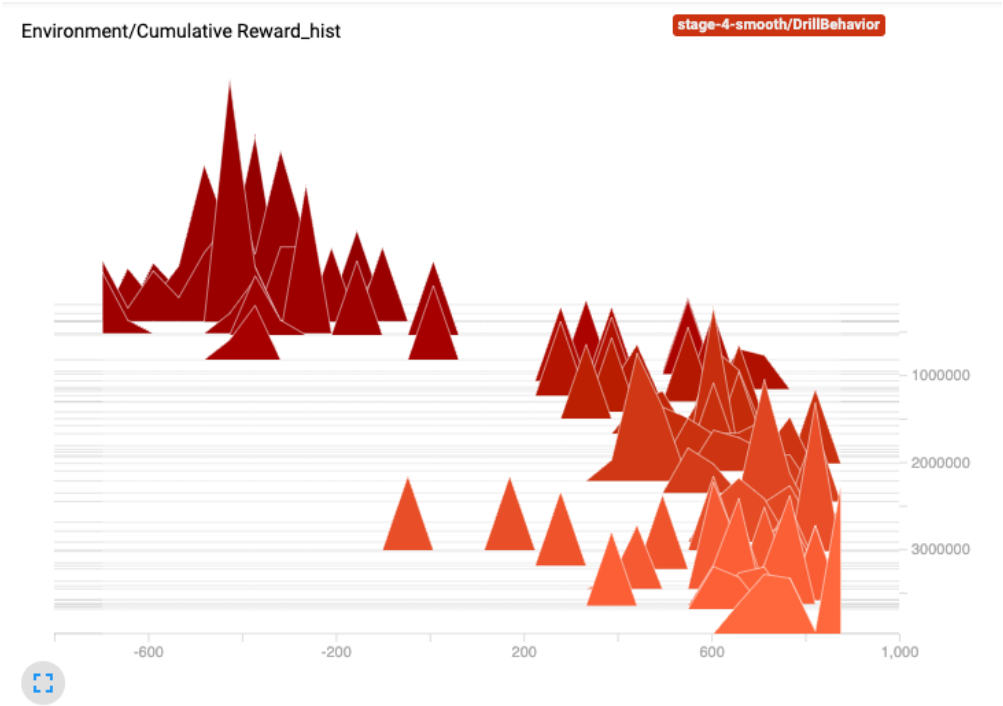


Figure 5.23: Cumulative Reward Histogram for Stage 4 shows two clusters where rewards are concentrated - One on the left represents the period when the agent is exploring more and still learning and the one towards the right represents the behaviour when there is more exploitation and when the learning process is stablized

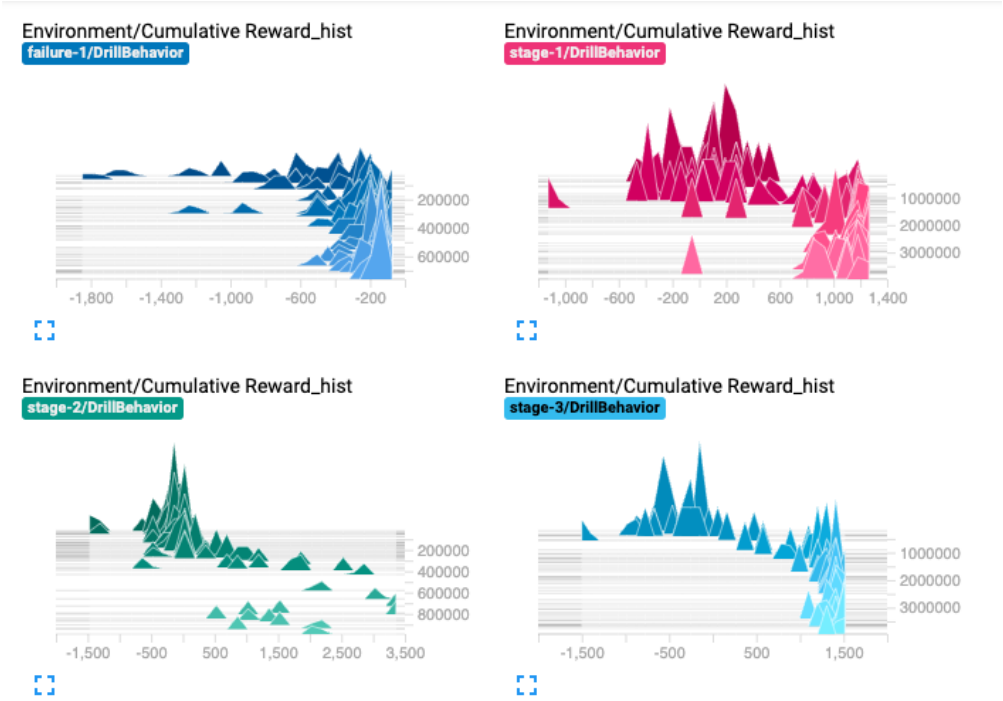


Figure 5.24: Cumulative Reward Histogram for Stages 1 (pink), 1-F (dark blue), 2 (green), 3 (cyan) - as can be seen in Stage 2 histogram, the agent was stuck in exploration for a longer period of time and less rewards accumulated over time.

# 6. DRILLING OPERATIONAL PERFORMANCE OPTIMIZATION

In this chapter, two drilling sub-systems - Vibration mitigation and Hole cleaning operations are studied and modeled. First, the vibration mitigation issue is briefly explained along with advancements like short-hop EM telemetry. A dysfunction detection library is developed using machine learning classification methods. Results of these models in identifying Stickslip are presented using surface and downhole measurements. Recurrent neural networks are shown to provide predictive analytic solutions for Stickslip dysfunction. Implementation of the classification algorithms and the RNN-LSTM algorithms are briefly explained. Results of predictions for 10 seconds and 30 seconds ahead of time are presented. The second aspect of drilling performance optimization explored in this study is the wellbore cleaning operation. An overview of the process and existing methodologies are briefly discussed. MDP model setup and learning environment using NORCE OpenLab simulator are presented. RL implementation, training process, and results of the learned policy are shown.

## 6.1 Vibration Mitigation While Drilling *

The key aspects of drilling operations that currently needs to be improved can be grouped into 3 broad buckets. The first one is in regards to drilling complexity and sensors inaccuracies, followed by high reliance on surface data processing & analysis, and lastly inefficient downhole-to-surface data transmission systems. The first one relates to the complexity of drilling operations. Drilling is a complex process by itself that involves monitoring of several strongly coupled dynamics under an aggressive and harsh environment that the bit endures downhole. This limits the ability to utilize downhole information effectively and the existing drilling optimization systems rely predominately on rig instrumentation data available at the surface. Even the surface instrumentation such as hook load, drill string revolutions

*Parts of this section are adapted with permission from *Machine Learning-based Intelligent Downhole Drilling Optimization System Using An Electromagnetic Short-hop Bit Dynamic Measurements* by Narendra Vishnumolakala, 2020, SPE Annual Technical Conference and Exhibition (ATCE) by Society of Petroleum Engineers.

per minute (RPM), surface torque, pump pressure, flow rates, etc. suffer from measurement errors in the field [126]. Furthermore, majority of sensor data acquisition is at low frequency. Electronic Device Recorders (EDR) service companies typically offer acquisition rates of about 1 Hz [127], which is insufficient for high-frequency control systems or robust semi-autonomous optimization systems that require an increased level of processing power.

Even if the challenges pertaining to sensor accuracy and measurement errors mentioned above were eliminated, we would still need to transmit the data from several thousands of feet downhole to the EDR at surface. This problem has severely hindered the adoption of some of the automated solutions. High-speed communication alternatives have been proposed but are still in the early adoption phase of their development. They are not economically viable for mass adoption on lower-cost wells, especially on the onshore and geothermal markets. Technologies like short-hop telemetry systems and downhole processing units offer solutions to counter the issue of low data transmission rate and bandwidth. The short-hop telemetry system is briefly discussed below.

**Short-Hop Telemetry**

The component-level computing architecture and data communication pathways of the proposed Short-hop telemetry system as deployed in drilling a well is shown in Fig. 6.1. The downhole processing component is modular, consisting of a Short-hop Transmitter Sub, located near the bit and below the mud motor, and a Short-hop Receiver Sub, located above the mud-motor and attached to the Electromagnetic (EM) Measurement-While-Drilling (MWD) Instrument Collar. The Short-hop Transmitter Sub gathers real- time bit dynamics data and transmits this data to the Short-hop Receiver Sub via a high-speed short-hop EM transmission pathway. The Short-hop Receiver Sub receives and pre-processes the data and passes the data to the EM MWD tool via a wired communication tool bus. The EM MWD tool transmits the data to the surface via a long-hop EM uplink to a Drilling Advisory System (DAS), which combines the downhole data with surface data gathered from the rig.

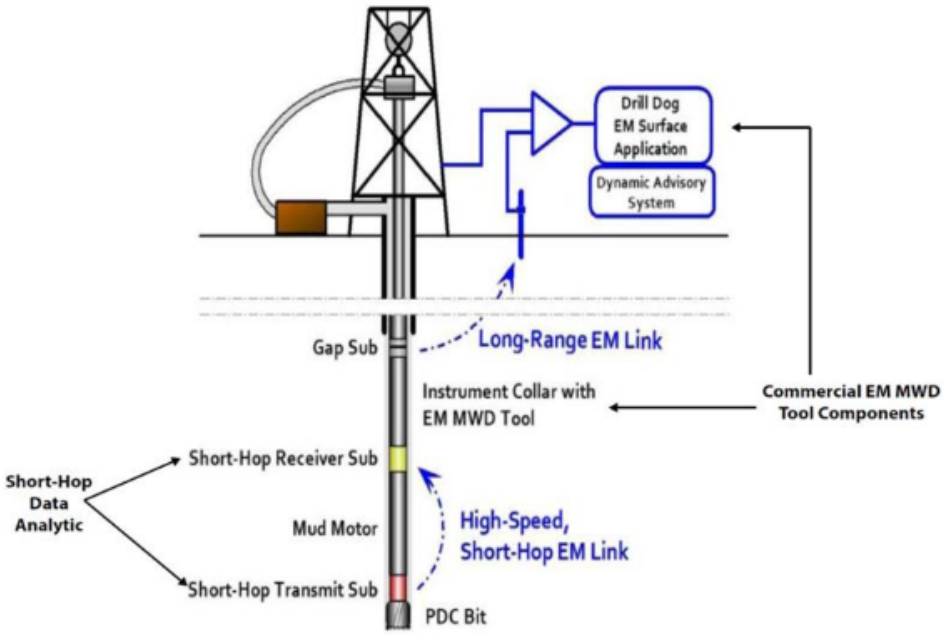


Figure 6.1: Optimized Telemetry System Architecture: The downhole processing component is modular, consisting of a Short-hop Transmitter Sub, located near the bit and below the mud motor, and a Short-hop Receiver Sub, located above the mud-motor and attached to a EM MWD Instrument Collar. The Short-hop Transmitter Sub gathers real-time bit dynamics data and transmits this data to the Short-hop Receiver Sub via a high-speed short-hop EM transmission pathway. The Short-hop Receiver Sub receives and pre-processes the data and passes the data to the EM MWD tool via a wired communication tool bus. The EM MWD tool transmits the data to the surface via a long-hop EM uplink to the Drilling Advisory System (DAS), which combines the downhole data with surface data gathered from the rig. *Reprinted from [128].*

### 6.1.1 ML-Based Dysfunction Identification

A bit-dysfunction library was developed using drilling logs from open sources as well as numerical drill string simulations. The objective was to derive and improve the performance of the identification algorithm and reduce the effects of latency for a driller to make use of crucial downhole information such as weight- on-bit, torque, and rotary speed by sending the essential downhole parameters in a single moving window information package over the proposed electromagnetic communication system. Furthermore, all surface drilling measurements, including measurements of those same parameters, are used more effectively to

control the drilling process.

*Exploratory Analysis*

High-frequency real-time drilling data was obtained from Equinor, an International Energy company. The data corresponds to Volve fields on the Norwegian Continental Shelf [129]. The Volve field and a snapshot of WITSML data format is shown in Fig. 6.2. The data is an aggregate from multiple service companies, which makes it inconsistent and needs preprocessing. The WITSML data is parsed to generate a single stream of data constituting all the input variables concerned - WOB, RPM, Torque, MSE, Bending Moment, Depth, and responses - ROP, Whirl, Stick-slip, Axial, Lateral Vibrations. The data used for modeling consisted of 7 variables (Fig. 6.4 and four outputs, each with two labels - yes/ no. Not all variables, not outputs, are from a common data source. Therefore, combining all of them into one dataset required a significant amount of data cleaning. Depth of drilling was chosen to be the indexing criteria, and all the data points are matched based upon the depth index. At the end of this step, labeled data was generated and consolidated in one common dataset.

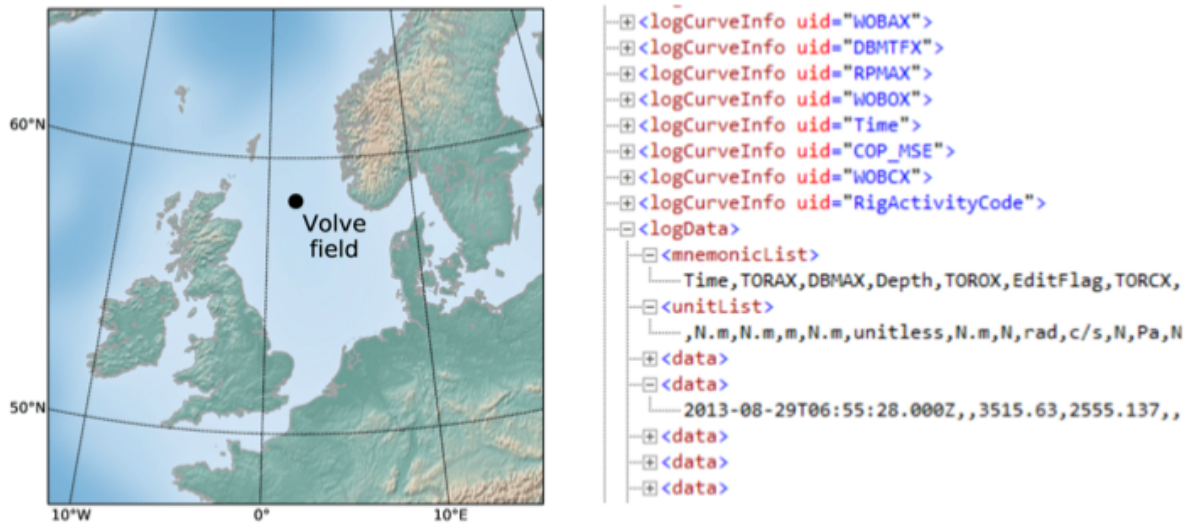


Figure 6.2: Location of the Volve field and an extract of the WITSML-based data showing an overview of the XML-formatted variables. *Reprinted from [128].*

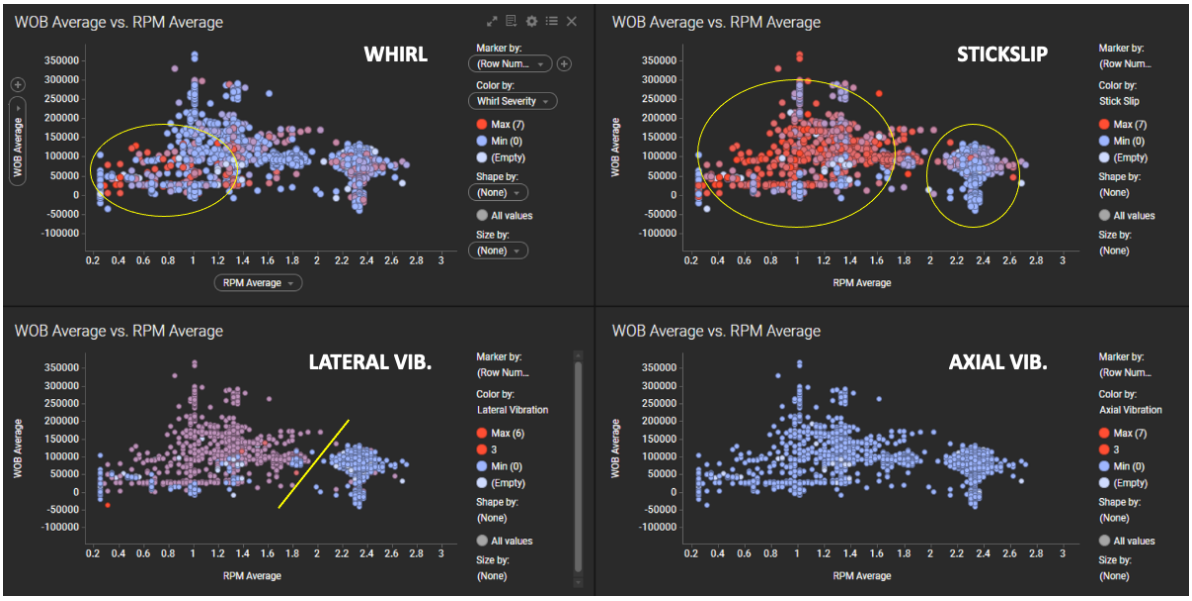


Figure 6.3: Exploratory Analysis of Volve Field Data showing dysfunctions (Whirl, Stickslip, Lateral Vibrations, Axial Vibrations) for a range WOB and RPM values. *Reprinted from [128].*

An exploratory analysis was performed on a sample subset of data from one well to observe for patterns in each of the dysfunctions. Fig. 6.3 shows the result of a well response for (a) Stick-Slip and (b) Whirl severity with WOB and RPM as input variables.

*Methodology*

Three main methods were analyzed in order to optimize drilling operations by detecting and mitigating dysfunctions as listed below.

- Firstly, develop regression models to estimate Rate of Penetration (ROP) from the available data without dysfunctions. And then detect anomalies in the ROP value while testing. These anomalies are the result of dysfunctions downhole while drilling.
- Apply classification models to classify and predict the presence of dysfunction using mutually exclusive binary classification
- Using unsupervised, clustering techniques to find patterns in unlabeled raw drilling data with available variables and group the results into dysfunction yes/no (1 or 0).

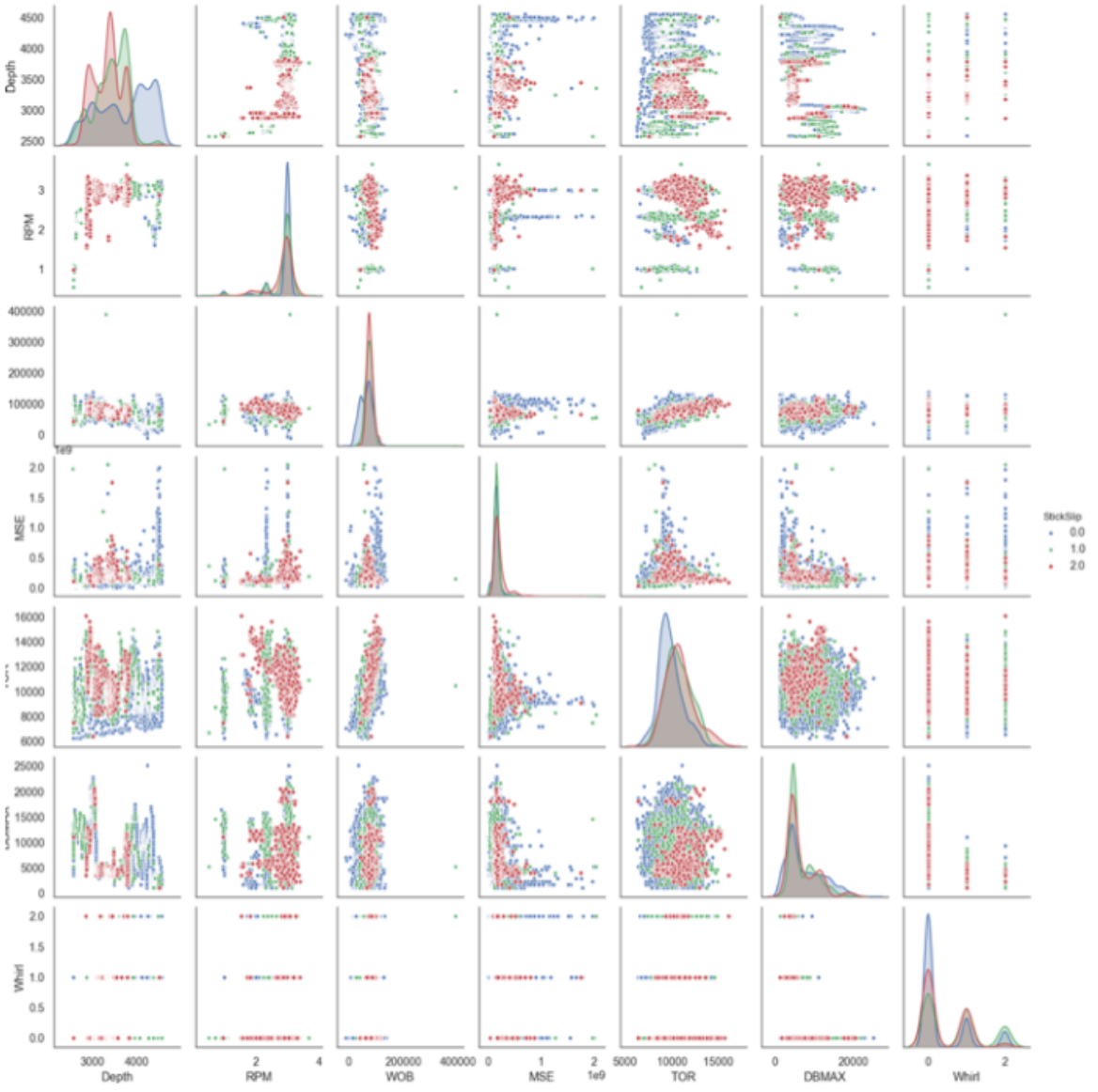


Figure 6.4: Pair plot of the variables used in the preliminary training set shows the distribution of parameters distinguished by a color hue for Stickslip severity (0-blue, 1-green, 3-red). *Reprinted from [128].*

Several well-known regression Machine Learning techniques can be used to estimate ROP. We use a different type of Machine Learning models to estimate the value of ROP. First, the database was separated into train, validation, and test sets. The validation set is set aside to help fine-tune our hyperparameters for different ML models. The goal is to improve the accuracy of ROP prediction further. The results can then be compared to determine the best models for estimating the value of ROP.

Only one out of the three approaches showed promising results. Estimating ROP and performing anomaly detection turned out to be non-feasible with the available data (Fig. 6.5). The clustering of data delivered poor results for the lack of large and clean datasets. The results are detailed below, along with improvements proposed for future work.

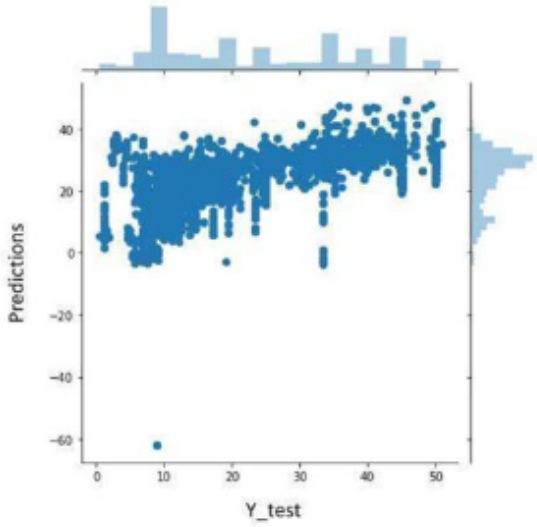


Figure 6.5: Initial ROP-based linear regression model shows poor performance of the model in estimating ROP on the test set. *Reprinted from [128].*

*ML Implementation*

The data was trained on multiple classification models and the results from each of the classifiers were compared on test accuracy and model complexity. Classifiers namely Logistic

Regression (LR), K-Nearest Neighbors (KNN), Decision Trees (DT), Random Forests (RF), and Neural Networks (NN) were used to train of the processed dataset containing 36,415 data points. This is not ideal for training complex models such as a neural network since we risk overfitting the data set and perform poorly on unseen samples. Another limitation on the data is that it is skewed with some classes occurring more often than the other. This further complicates the training and testing phase since some adjustments are expected to accommodate this attribute of our dataset. Fig. 6.6 shows the correlation matrix of the most important variables of the training dataset used to derive the results discussed in this section.

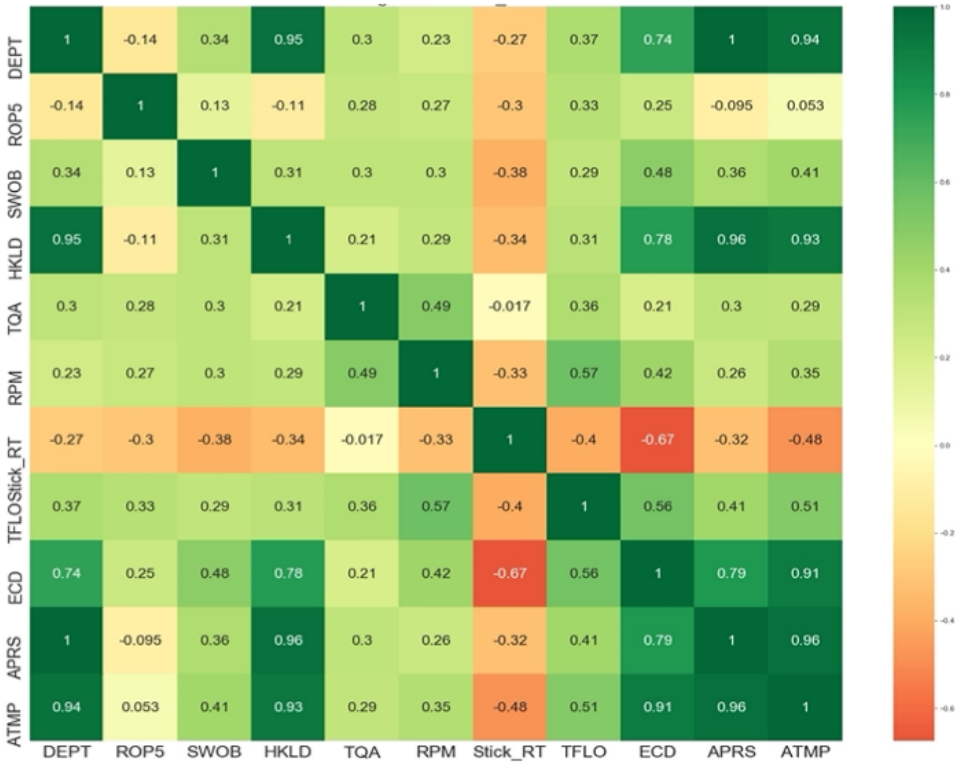


Figure 6.6: Correlation matrix plot of the variables used in the preliminary training phase used to identify the most important parameters to be used for the model. *Reprinted from [128].*

*Results*

Random Forests model achieved the highest overall accuracy of 89% or least test error. Tree-based methods, in general, are less complex to implement and have good interpretability. Decision Trees suffer from high variance, and in order to decrease variance at the expense of slightly increased bias, bootstrap aggregating or bagging of trees is suggested. In the Random Forests model, bagged trees are built with sampled parameters instead of a full set.

Classification results of Random Forests model are as shown in Table 6.1. As can be seen in Figure 8 the data is skewed towards label '0' or 'no stick slip' condition. Therefore, instead of overall accuracy, a better metric like f1-score will provide more insights into predictions. Recall of class '0' is 0.95 which means true positives are 95% of all actual labels. Recall of class '1' is 0.70 which means true positives are 70% of all predicted labels. In other words, it can be explained for this case of stick slip detection in this fashion. The model predicted stick slip 5% of the time when it was normal (No stick slip) condition. Similarly, the model predicted that there was no stick slip 30% of the time when there was in fact stick slip happening downhole.

| Class | Precision | Recall | f1-score |
|---|---|---|---|
| 0 | 0.91 | 0.95 | 0.93 |
| 1 | 0.80 | 0.70 | 0.74 |

Table 6.1: Random Forest model results on one test well shows overall high accuracy in classification of Stickslip but suffers from low precision and recall values for the case of Stickslip presence.

Other classifier models have been implemented on the dataset. For example, a more advanced model based on neural networks (NN) was implemented to achieve an overall 87% accuracy. Although a superior model, NN works best with larger datasets. For this reason,

a simple, fully connected model that reduces the risk of overfitting of overly complex neural networks is developed. Various Neural Network configurations were examined by changing the number of hidden layers (2,3) with different numbers of hidden units for each layer. It was observed that the more complex models (more layers, hidden units) did not give better performance compared to the simpler models and in some cases perform worse than their simple counterparts. To this end, a fully connected architecture which consists of one input layer (6 input nodes), two hidden layers (each with 128 hidden units) and an output layer with two output nodes (1 for each possible classes) was implemented. The architecture is shown in Fig. 6.7. Results of the NN model and other classifiers are summarized in Table 6.2 below.

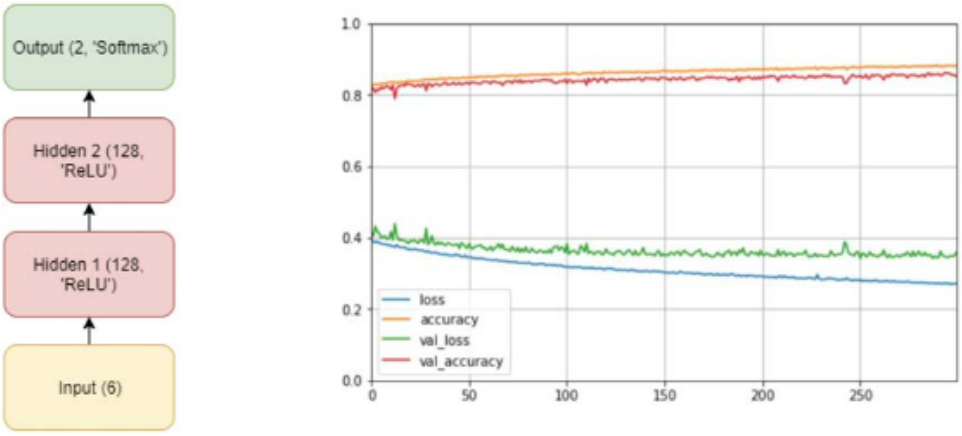


Figure 6.7: A fully connected architecture (on left) of one input layer (6 input nodes), two hidden layers (each with 128 hidden units), and an output layer with two output nodes (1 for each possible classes). Tensorboard scalars are shown on the right indicate performance of the model with metrics loss (blue), validation loss (green), accuracy (orange) and validation accuracy (red) on vertical axis and number of epochs on the horizontal axis. *Reprinted from [128].*

| **Algorithm** | LR | KNN-10 | DT | RF | ANN |
|---|---|---|---|---|---|
| **Accuracy** | 77% | 75% | 85% | 89% | 86% |

Table 6.2: Accuracy results from the several supervised machine learning models on the initial drilling dataset identifying stick-slip.

### 6.1.2 Prediction of Stickslip with RNN-LSTM Model *

#### Background

Sub-optimal drilling operations can cause excess vibrations. These vibrations occur in different directions (axially, laterally, and around the axis of rotation) and can lead to drilling dysfunctions based on the type of vibrations. Axial vibrations lead to bit bouncing, lateral vibrations lead to bit whirl, and torsional vibrations lead to stick-slip. Drilling dysfunctions can damage the drill string, cause premature bit wear, or create stuck-pipe situations. Drill string damage results in excess NPT and sub-optimal economics when developing a field. In a worst-case scenario, the entire drilling operation may be jeopardized in the event of a severely damaged drill string [130]. This can result in a lost well and an economic loss of over $2 MM. Less severe dysfunctions can cost hundreds of thousands to perform corrective operations. Drilling advisory systems are utilized to detect the status of downhole tools and relay that information to the driller. Current drilling advisory systems are dependent on data collected at surface and extrapolated to what may or may not be the conditions downhole. Collection of data at the surface instead of at- or near-the-bit can result in an inaccurate depiction of the situation downhole. This study will use real-time, high frequency downhole data collected at-the-bit and apply supervised machine learning classification models to the data in order to determine when drilling dysfunctions are occurring in real-time and Recurrent Neural Networks to predict them into the future.

*Parts of this section are adapted with permission from *Predicting Dysfunction Vibration Events while Drilling Using LSTM Recurrent Neural Networks* by Narendra Vishnumolakala, 2021, SPE/IATMI Asia Pacific Oil & Gas Conference and Exhibition by Society of Petroleum Engineers.

Prior works related to vibration predictions downhole relied extensively on drill string configuration inputs with complex calculations and high simulation times. Zha et. al. [131] used data deep learning techniques with a parallel combination of Convolution neural networks (CNN's) and recurrent neural networks (RNN's) to predict downhole vibration using surface drilling data. However, the intricacies of continuous change/unpredictable nature in downhole boundary conditions can still be a major hurdle to rely on surface data [132]. Tian and Horne [133] successfully implemented a recurrent neural network method to analyze downhole gauge data. RNNs are best suited for domains where sequential data needs to be analyzed or processed. Successful applications include predicting vibration events in aircraft engines [134], forecast on time series data in waterways using LSTM networks [135] to predict water levels, to name a few.

**Model Setup & Implementation**

Data-driven models to predict vibrations ahead of time are developed using Recurrent Neural Networks (RNN). The architecture of the model is discussed, first with a brief description of RNN machine learning models. Data used for training and subsequent problem formulation is shown next. Recurrent neural networks (RNN) are a class of neural networks best suited for processing sequences of values by mapping sequences to vectors, vectors to sequences or sequences to sequences. Unlike traditional neural networks where all inputs (and outputs) are typically assumed independent of each other, RNNs have a "memory" which captures information about what has been calculated so far. This makes them powerful in solving problems involving sequential data. Although robust, RNNs suffer from the problem of vanishing gradients. The gradients carry information used in the RNN, and when the gradient becomes too small, the parameter updates become insignificant. This makes the learning of long data sequences difficult. A brief introduction to the concepts of RNN and LSTM is provided next.

As explained in Olah [7] and Brandenburg [136], a typical LSTM network consists of three stages (Figure 1). The first stage (I) is called the "forget gate layer" where it is decided

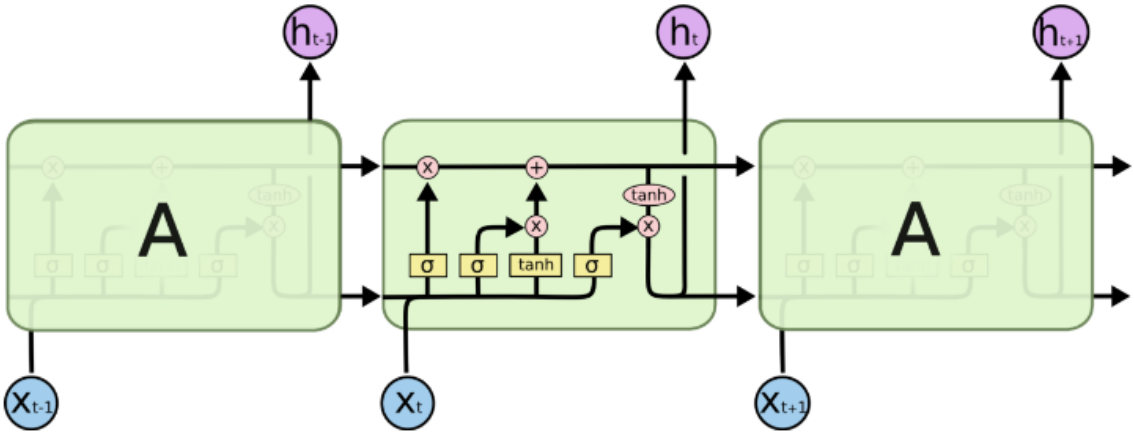


Figure 6.8: High-level view of a repeating module in RNN LSTM Architecture [7]

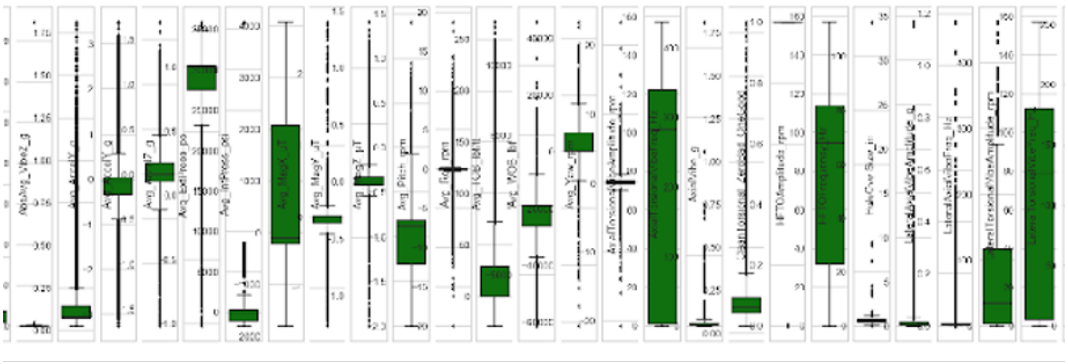

Figure 6.9: Exploratory Analysis of dataset using Boxplots. *Reprinted from [137]*

what information is retained and what is discarded from the cell state. Input at the current timestep and output from the previous timestep are fed through a sigmoid layer that outputs a number between 0 and 1 as a measure of information retained. The second stage (II) is an "update layer" where a sigmoid layer called "input gate layer" combined with a tanh layer creates an update to the state. The final stage (III) is the "output layer" where a sigmoid in combination with a tanh layer creates a filtered version of the information to output.

Data from drilling logs is gathered from various fields. High-frequency downhole data is utilized for training. Using downhole data instead of surface measurements significantly re-

| | **Wells A & B** | **Well C** |
|---|---|---|
| 14***Input** | Temperature | Temperature |
| | Avg Accel X | Avg Accel X |
| | Avg Accel Y | Avg Accel Y |
| | Avg Accel Z | Avg Accel Z |
| | Avg Mag X | Avg Mag X |
| | Avg Mag Y | Avg Mag Y |
| | Avg Mag Z | Avg Mag Z |
| | Wobble | |
| | Hole Oversize | |
| | Avg WOB | |
| | Avg TOB | |
| | Avg BOB | |
| | Avg Internal Pressure | |
| | Avg External Pressure | |
| **Output** | StickSlip Indicator | StickSlip Indicator |

Table 6.3: List of Parameters used for modeling in each of the 3 wells datasets

duces data uncertainties. The data consists of downhole logs from 3 different wells - referred to as Well A, Well B, Well C for the sake of this discussion. Data pertaining to Wells A & B consisted of 101 parameters - measured and derived combined. Measured parameters pertain to data obtained from sensors measuring Temperature, Weight-on-Bit (WOB), Torque, Bending Moment, Accelerometer data in 3 axes and Magnetometer data in 3 axes. Derived parameters are generated from accelerometer and magnetometer raw readings in both time and frequency domains. Examples of derived parameters are Vibration frequencies, Spectrogram values etc. Well C consisted of 95 parameters only including accelerometer and magnetometer measured and derived parameters. Pre-processed data (Fig. 6.9) obtained from three different wells are used to train supervised classification models as well as RNN LSTM networks using the parameters listed in Table 6.3. Model formulation and training process are explained in detail.

First, the data is split into training and test datasets. 70% of data from each well was used for training the models and the remaining 30% to test the accuracy of the trained

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 60, 50)            13000
_________________________________________________________________
dropout_3 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_4 (LSTM)                (None, 60, 50)            20200
_________________________________________________________________
dropout_4 (Dropout)          (None, 60, 50)            0
_________________________________________________________________
lstm_5 (LSTM)                (None, 50)                20200
_________________________________________________________________
dropout_5 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 51
=================================================================
```

Figure 6.10: Model Configuration of an LSTM Run shows the setup of layers in the first column - 3 LSTM layers and corresponding dropout, shape of each layers in second, and the number of parameters in the layers are shown in third column.

models. A regression model is built using LSTM architecture - a sample configuration is shown in Fig. 6.10. The Input layer has 14 nodes for Wells A & B and 9 nodes for Well C. Three hidden layers each with 50 neurons were used to build the LSTM network. To prevent the model from overfitting, a dropout of 20% was used in each of the layers. Finally, an output layer with 1 neuron was used to predict Stickslip vibration level. The number of timesteps for the LSTM varied between 10 and 100 seconds. Most of the experiments were run at 50 unit timesteps and is referred to as the 'default' configuration for the remainder of the paper. Training was performed for 100 epochs with a batch size of 32. Figure 6 is a sample configuration of an LSTM run. Several runs were performed by varying 1) the data set (Wells A, B, C) , 2) timestep size 3) input data frequency 4) LSTM hyperparameters like batch size, dropout regularization.

**Results**

An LSTM neural network is trained on the data for 300 epochs to learn. Training and validation results are shown below. Hyperparameters are altered and a comparison of the

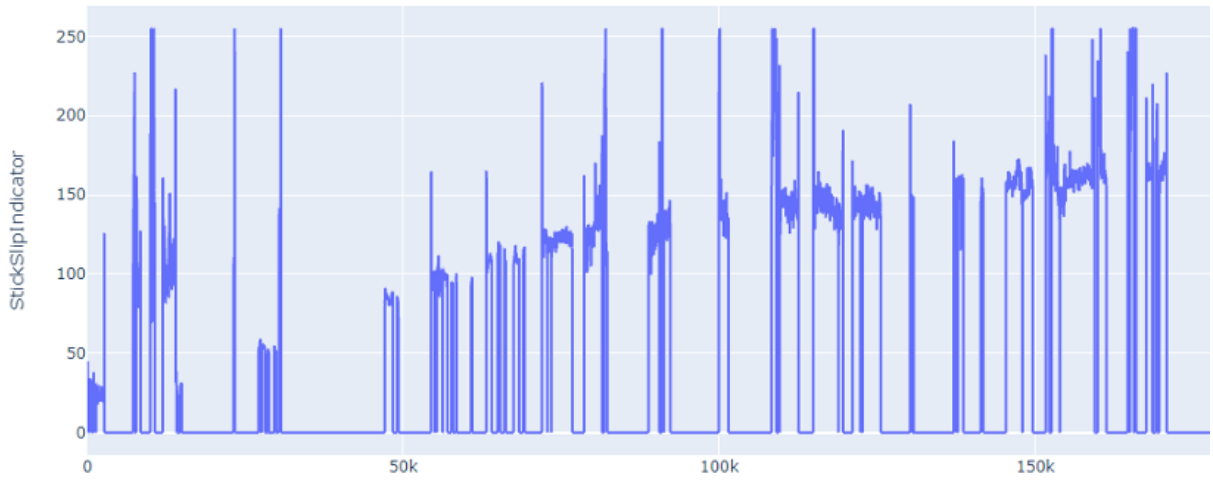


Figure 6.11: Simple Line Plot of StickSlipIndicator in Well A dataset with timestamps (horizontal axis) shows that the stickslip severity values shown on vertical axis (distributed from 0 to 250) are not heavily skewed. *Reprinted from [128].*

performance is also shown. The models performed best on Well A dataset. Well B dataset is skewed and has several missing data points. Well C dataset suffers from similar issues in addition to having less number of predictors. Comparison of results from each of the wells is provided. Predictions are made 10 seconds and 30 seconds into the future. As would be expected, 10 second predictions are found to be more accurate.

Figure 6.11 shows the distribution of Stickslip data from Well A dataset. As can be seen, the data is not skewed. Although the dataset contained 370k data points, only the first 170k data points contained relevant data and hence only part of the full dataset was used for modeling. 0-120k data points were used for training and 120k-170k (50,000 data points) was used for testing the model results. Fig. 6.12 shows the results of the trained models for 10 second predictions into the future. Predictions were made on the test dataset and compared with actual data. The predicted vibration levels seem to be in line with the actuals. Fig. 6.13 is a "zoomed in" version of Figure 10 showing predicted versus actual vibration levels on a portion of the test dataset. Figures 6.14 and 6.15 show results of predictions on Wells B & C datasets. As explained earlier, due to skewed and noisy data, these results are not

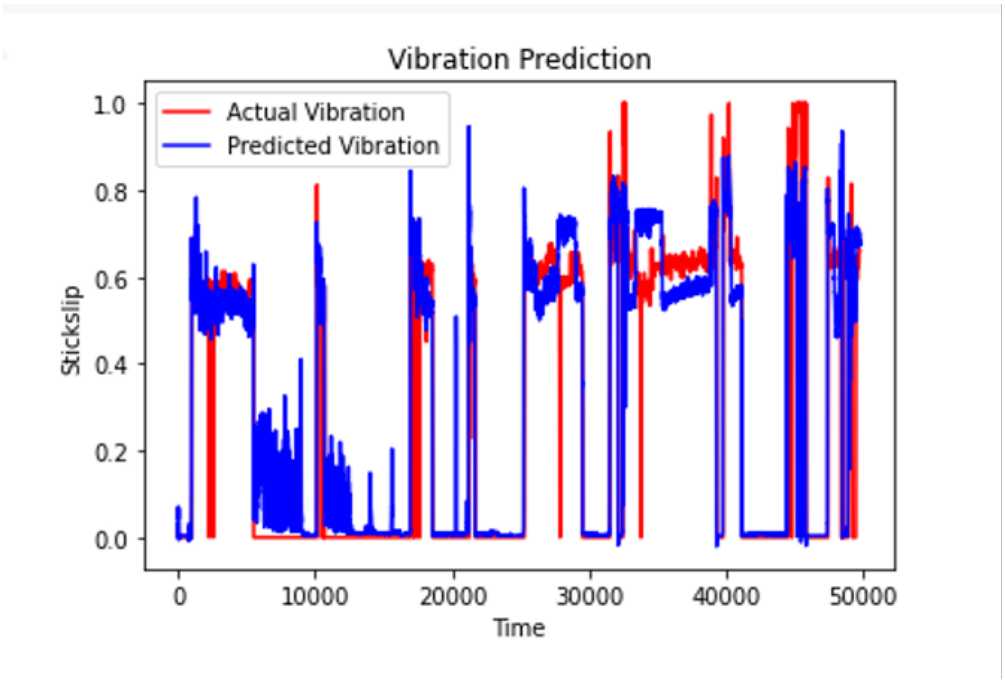


Figure 6.12: Vibration Prediction Results: Well A: 10 second prediction shows good match between the predicted scaled stickslip values (blue) and actual scaled stickslip values (red) on vertical axis over time in horizontal axis.

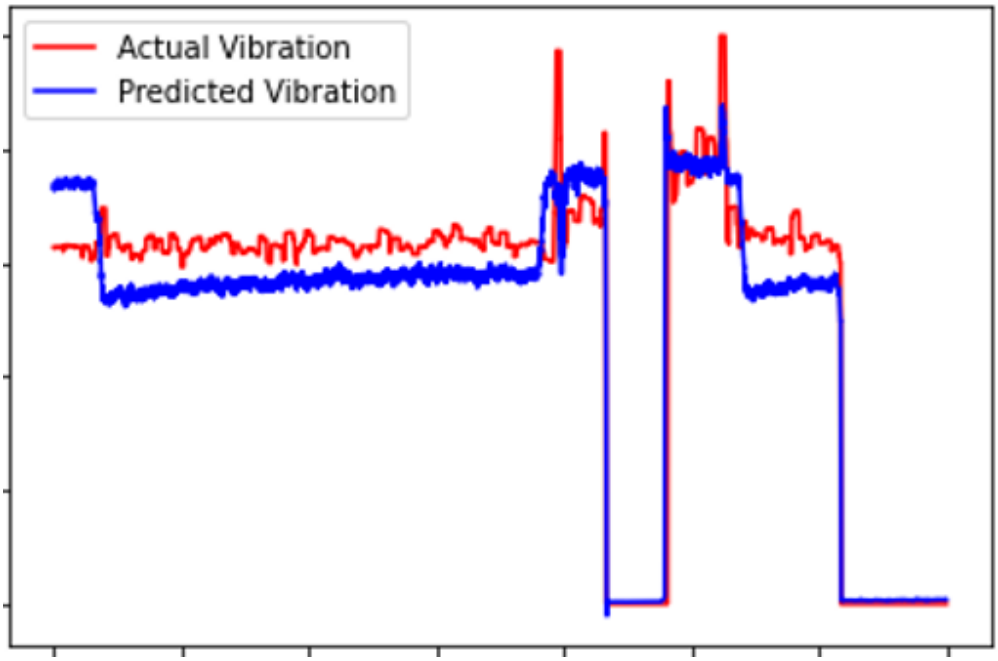


Figure 6.13: Vibration Prediction Results: Well A: 10 second prediction; Zoomed into a shorter time interval highlights the close match between the actual and predicted values.

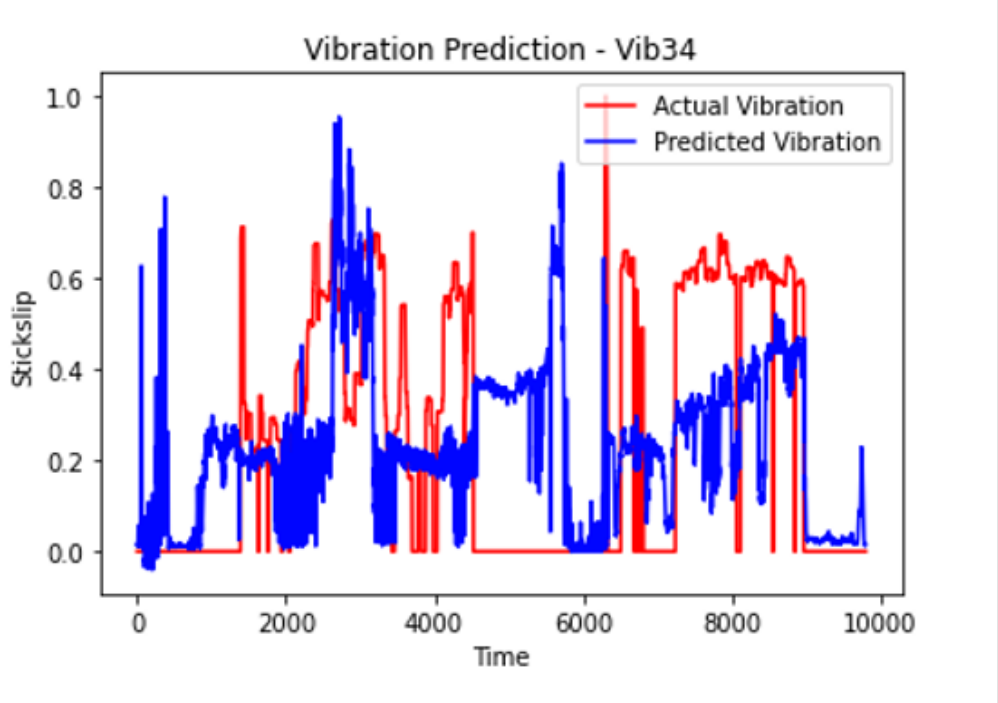


Figure 6.14: Vibration Prediction Results: Well B: 10 second prediction shows poor match between the predicted scaled stickslip values (blue) and actual scaled stickslip values (red) on vertical axis over time in horizontal axis.

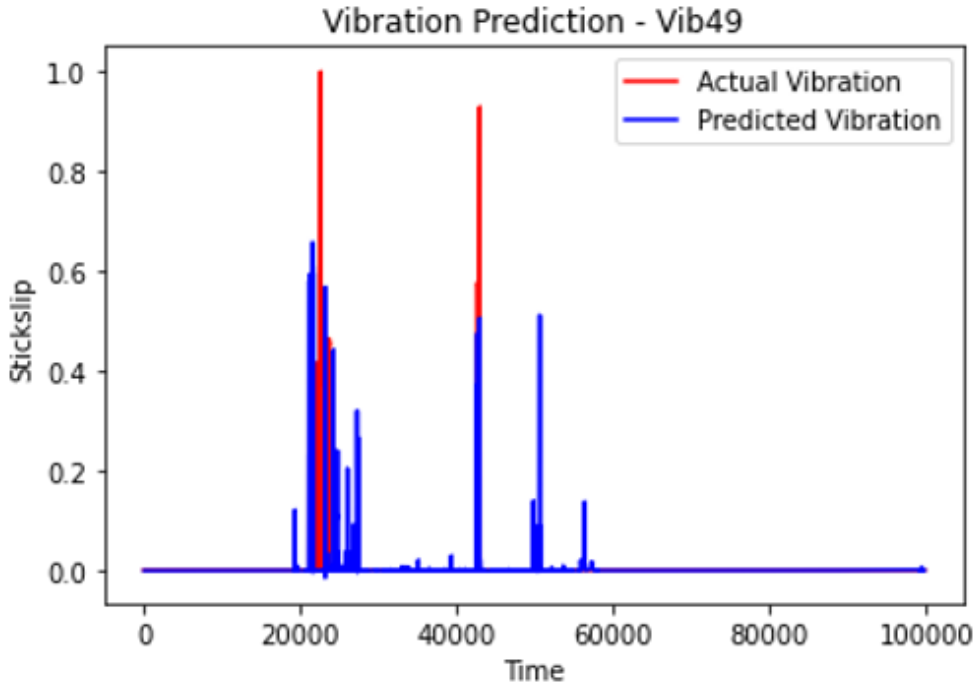


Figure 6.15: Vibration Prediction Results: Well C: 10 second prediction shows poor match between the predicted scaled stickslip values (blue) and actual scaled stickslip values (red) on vertical axis over time in horizontal axis due to skewed dataset.

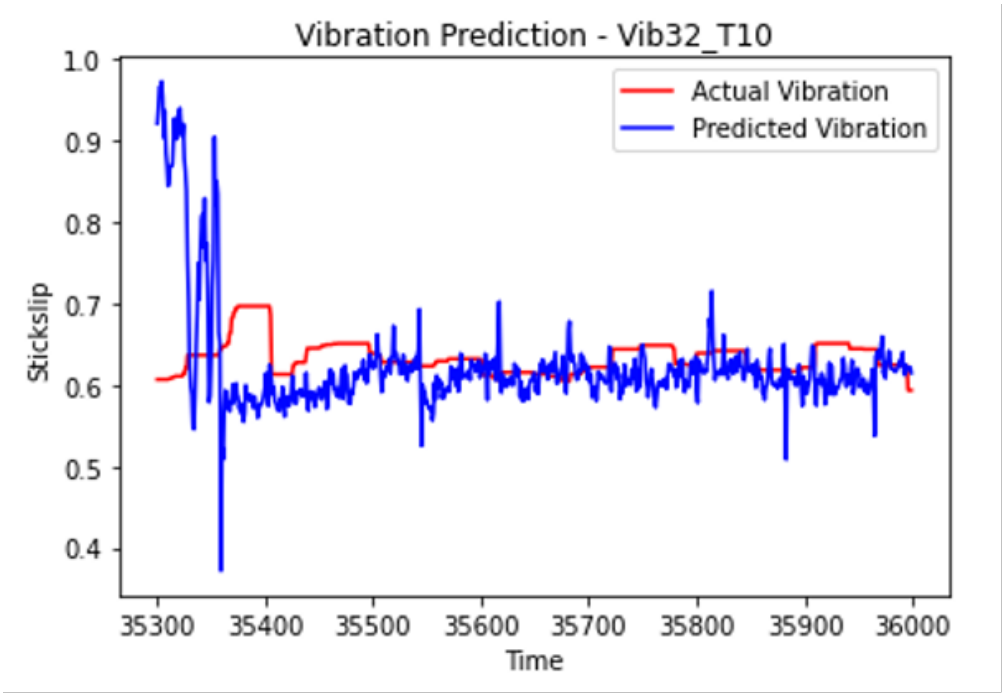


Figure 6.16: Vibration Prediction Results: Well A: 10 second prediction; Timestep window: 10 units - shows deterioted performance in prediction of Stickslip (vertical axis) over time (horizontal axis).

accurate enough with predictions being highly sensitive and possible overfitting.

With the highly encouraging results obtained from Well A, further analysis was done to improve the predictions. Optimizing hyperparameters of the RNN LSTM model has the potential to significantly improve the performance of the models. Few experiments were performed in this regard. The default timestep was changed from 50 units to 10 units and 100 units. The 10 unit run as expected underperformed (Fig. 6.16) compared to the default setting while increasing the timestep window improved prediction accuracy as shown in Fig. 6.17.

The experiments were repeated for predicting Stickslip events 30 seconds ahead of time. The results are very encouraging with trends similar to those of 10 second predictions. The accuracy is less than that of the 10-second predictions as expected. Fig. 6.18 shows the 30-second prediction results on Well A dataset. Fig. 6.19 is the "zoomed in" version of the

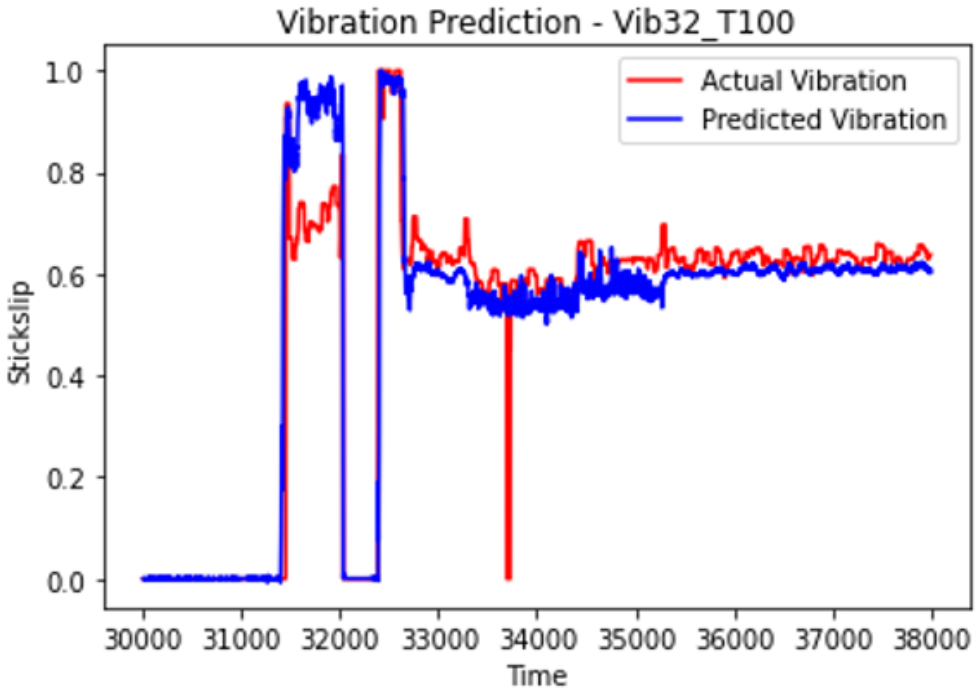


Figure 6.17: Vibration Prediction Results: Well A: 10 second prediction; Timestep window: 100 units - shows improved performance in prediction of Stickslip (vertical axis) over time (horizontal axis).

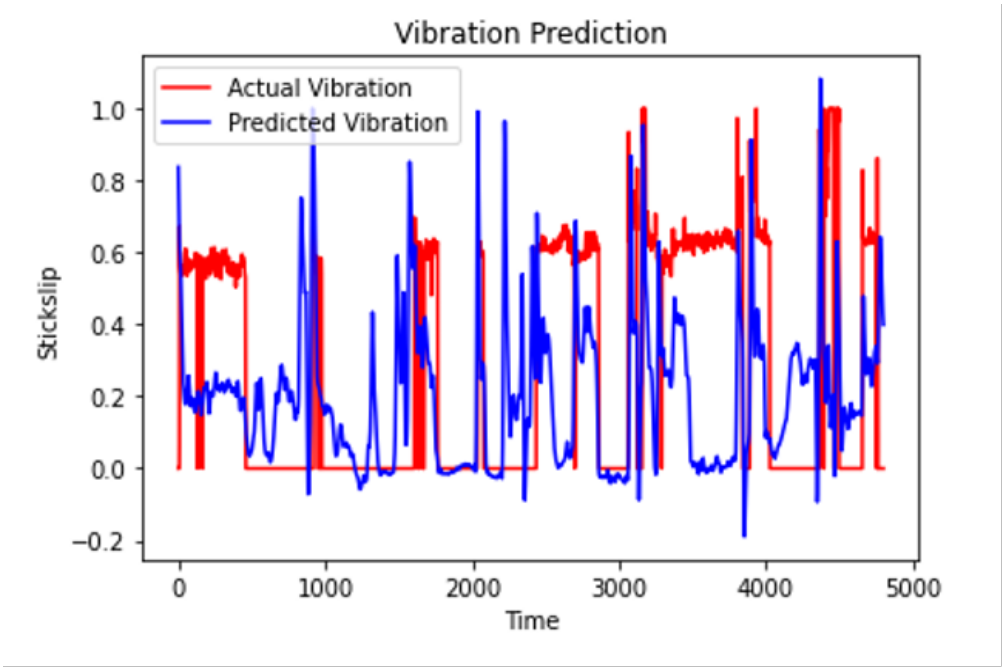


Figure 6.18: Vibration Prediction Results: Well A: 30 second prediction shows comparable match between the predicted scaled stickslip values (blue) and actual scaled stickslip values (red) on vertical axis over time in horizontal axis.

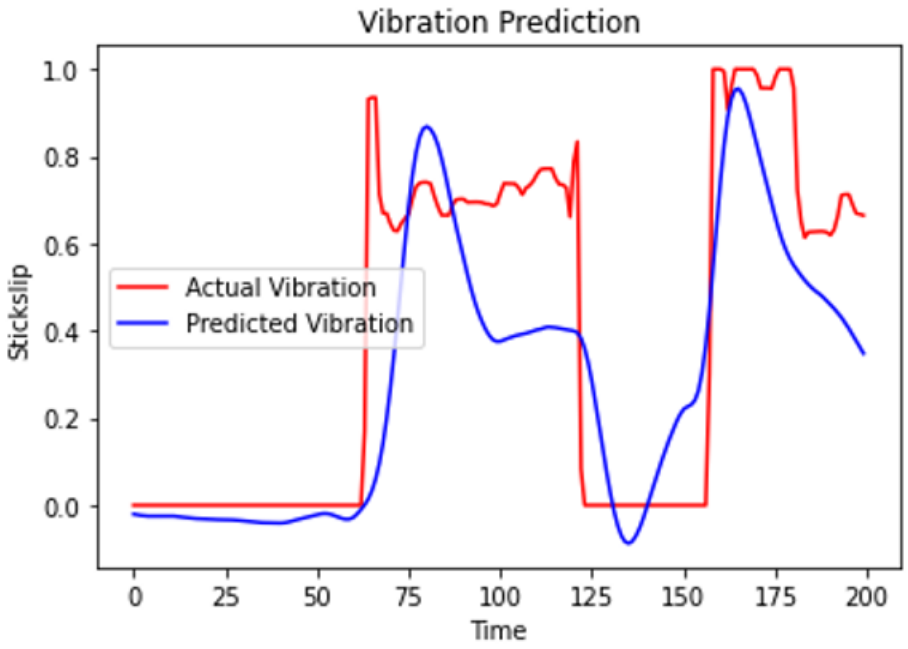


Figure 6.19: Vibration Prediction Results: Well A: 30 second prediction; Zoomed into a shorter time interval highlights the close match between the actual and predicted values.

results showing predicted versus actual vibration levels on a portion of the test dataset.

Mean Squared Error (MSE), Mean Absolute Error (MAE) and Normalized RMSE were used as a measure of accuracy for the regression models discussed above. An MSE of 0.02 resulted from the 10-second model training and an MSE of 0.10 resulted from the 30-second model. MSE penalizes larger prediction errors by square whereas Mean absolute error (MAE) treats all errors the same. Normalized RMSE (NRMSE) facilitates the comparison between models with different scales and is calculated by dividing the Root mean squared error with the standard deviation. Table 6.4 summarizes the evaluation metrics obtained from the results of predictions on Well A dataset.

## 6.2 DRL for Improved Hole Cleaning

The steering decisions taken by the RL models discussed in Chapters 4 & 5 might affect the transportation of drill cuttings from downhole to surface resulting in overall sub-optimal directional drilling operations. Effective cleaning of wellbore is achieved by integrating the

| Well A | MSE | MAE | NRMSE |
|---|---|---|---|
| 10-sec Model | 0.02 | 0.164 | 0.844 |
| 30-sec Model | 0.1 | 0.274 | 1.195 |

Table 6.4: Well A Testing Results - Error calculations summary shows that 10-second predictions are far more accurate than 30-second predictions.

hole cleaning models into the RL framework. Integration with steering models is not in the scope of this study. Stand-alone implementation of RL in improving hole cleaning efficiency is presented.

### 6.2.1 Background

Several attempts have been made to optimize the cuttings transport process while drilling. Broadly, there are three approaches to tackle this complex phenomenon - Experimental, Numerical and Data-driven methods. The original experimental work of Sifferman and Becker [138] resulted in determining critical parameters that impact hole cleaning. About 10 variables with respect to drilling fluid properties, circulation system, directional properties were found to be the key parameters affecting the process. Although the work validated several of the hypotheses surrounding the hole cleaning process, it needed further research to develop a good correlation between the variables. Several of these variables had been individually studied like in the work of Hopkin [139] testing the effect of mud rheology and Mohammadsalehi et. al. [140] studying the effect of wellbore configuration. Although the experimental approach gave a simplistic explanation for the effect of one or more variables, run in artificially controlled environments coupled with limited variables as chosen by the experimenter. Hence they may not represent real wellbore drilling conditions that are usually characterized by uncertainties due to changing downhole conditions. It also involved large-scale settling velocity flow loops that would adequately represent field conditions that may be costly to set up [141]. To overcome this issue, numerical simulations and physics-based modeling techniques have been proposed.

Cayeux et. al. [142] proposed a real-time, transient cuttings transport model that can calculate the distribution of cuttings along the wellbore. A good match between the surface measurements, observations, and the model prediction was attained in the use cases tested. Ozbayoglu et al. [143] developed a cuttings-transport mechanistic model that not included the primary variables concerning mud rheology but the effects of drill-pipe rotation and eccentricity. The model can estimate the volumetric distribution phases in three-phase flow and the pressure losses in the horizontal sections. The numerical simulations approach proved to be useful when applied to phenomena having complex mathematical models such as the hole cleaning process, which are difficult to establish analytical solutions, especially non-linear systems. The disadvantages of the methods are high computational cost and implementation complexity. Some approaches to tackle this issue were proposed like in Feder et al. [144] where a steady-state model is used to extend it to a pseudo-transient model. This is a practical approach but leads to uncertainties and loss of accuracy due to the numerical model.

Data-driven models using the Artificial Intelligence approach were then developed. Tombul et al. [145] applied several data-driven models (linear and nonlinear regression, support vector regression (SVR), support vector machine (SVM), and artificial neural networks (ANN)) to predict the velocity and direction of the cuttings using experimental data collected via a particle image velocimeter. Cayeux et al. [146] discuss the development and application of a system that defines and estimates some key indicators using physical models and real-time data to detect deviations from normal expected behavior and provide warnings to the drilling crew. However, most methods are reactive in nature i.e. these methods require the dysfunction to have already taken place to be able to detect the problem, or they rely on human operators to take the right decision based on their understanding of the state of the system.

Alawami et.al. [147] developed a real-time indicator for the evaluation of hole cleaning efficiency using raw data. The developed system automatically calculates the Carrying Ca-

pacity Index (CCI) in real-time that gives a measure of the cuttings transport efficiency. Although this system takes us one step closer toward the ultimate goal of having an integrated and fully automated hole cleaning evaluation and intervention tool, the calculations are based on a deterministic formula that is not comprehensive of all the possible drilling variables thus making it inaccurate in practical applications. Hole cleaning is a stochastic sequential decision-making problem that needs to be tackled with Reinforcement Learning (RL) methodology.

RL has been used to solve some of the drilling inefficiencies and improve decision-making. A highly non-linear process in drilling is managed pressure drilling (MPD) that requires the development of a complex dynamic model for the process. A smooth update deep Q learning algorithm is used by Arno et. al. [148] instead to train an agent embedded in a managed pressure drilling system. This study tackled several issues surrounding MPD but hasn't investigated using RL for hole cleaning. Another similar study by Yingwei Yu et. al. [149] is the usage of Deep Q-learning as proposed by Mnih et. al. [150] to train an automated directional drilling agent. Cuttings transport and hole cleaning issues are alleviated in deviated wells. The study did not cover the issue in their modeling.

Based on our literature review, the closest study so far to our proposed project is the work of Saini et. al. [151] where the application of Digital Twinning and RL have been proposed for predictive action planning for hole cleaning optimization. The hole cleaning problem, which inherently is a decision-making problem under uncertainty, has been set up in a simplified sense as a Markov reward process (MRP). States, actions, and rewards associated with state-action pairs have been proposed for this system.

### 6.2.2 Model Setup & Implementation

The hole cleaning process is modeled as a Markov Decision Process (MDP). A lot of factors affect the cuttings accumulation on a drilling field. However, to reduce the complexity, limited parameters are picked in the state and action spaces. Most of the configuration parameters are kept constant across all the experiments. Stochasticity is incorporated into

the framework by randomizing Mud rheology - a drilling parameter that significantly affects cuttings transport as discussed in Nazari et. al. [152]. The rate of penetration (ROP) which is the drilling speed and the string velocity are kept constant across all experiments as well. Even with these simplifications, the cuttings do get accumulated if appropriate planning is not done to ensure their removal. Once the cuttings start to accumulate, the pump flow rate has to be increased sufficiently to remove the cuttings. A planned agent would proactively control the pump flow with the increase in depth to ensure that the cuttings bed never increase and this is the intended behaviour of a learnt agent.

**MDP Framework**

- State Space ($\mathcal{S}$)
  - Max Cuttings Bed Height (cm)
  - Cuttings Bed Height Volume (cm)
  - Drill depth (m)
- Discrete Action Space ($\mathcal{A}$)
  - Flow Rate (l/min): [1800, 2500] in steps of 50
- Continuous Action Space ($\mathcal{A}$)
  - Flow Rate (l/min): [1800, 2500]
- Rewards ($\mathcal{R}$)
  - Max Cuttings Bed Height (cm): -5x
  - Cuttings Bed Height Volume (cm): -1x
  - Pump Flow Rate (l/min): -0.1x
- Stochasticity
  - Mud Rheology: random in [2.0 - 2.1]

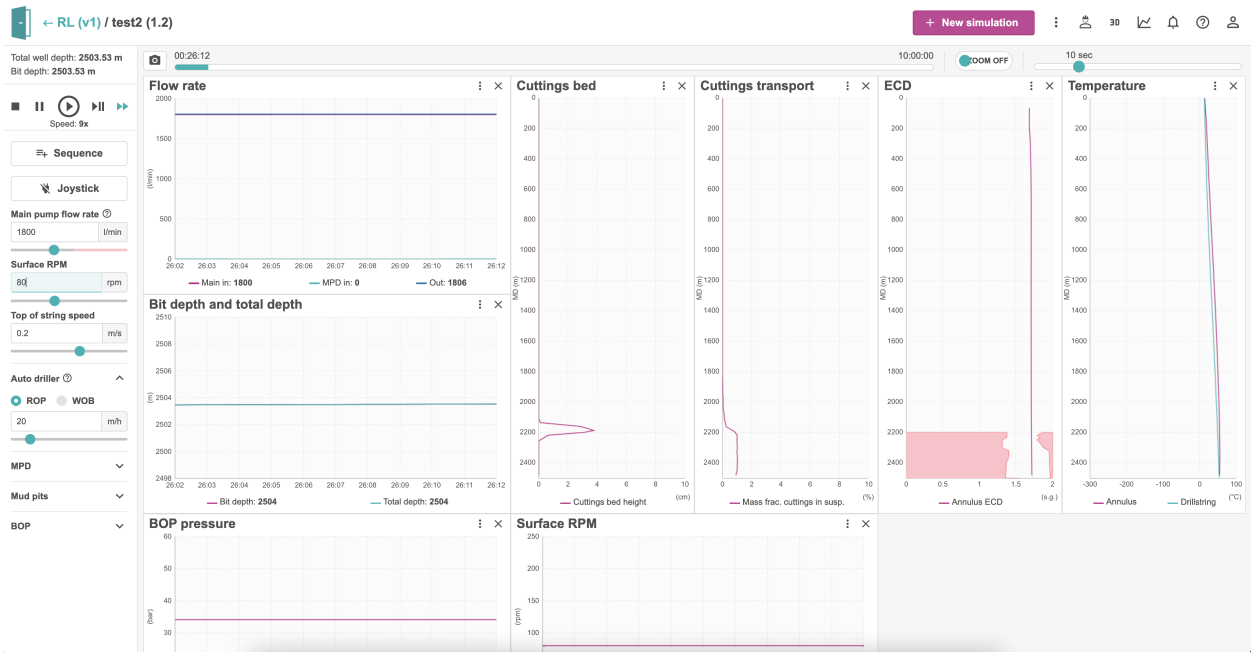


Figure 6.20: NORCE OpenLab Simulator showing a sample simulation configuration setup for an experiment on hole cleaning process for training the RL agent

### Learning Environment

A drilling simulator called *OpenLab* (Fig. 6.20) that is developed and managed by the Drilling & Well Modeling group of NORCE Energy in collaboration with the University of Stavanger, is used for training the models. OpenLab is an integration of the physical and virtual drilling and well operations, which is new and unique to the Oil & Gas drilling world. The simulator can be run interactively through a web browser, or also programatically through Matlab or Python packages. In this work, the Python client has been used to create a a gym environment. A drilling configuration is setup on OpenLab with realistic physical values to several drilling parameters. In this configurations, the stochasticity parameters are changed before each episode and the interactive simulations commenced. On each step, the action space variables are updated in the simulator and the drilling operation is run for 1 minute. At the end of each step, observations and rewards are collected and the state space is populated.

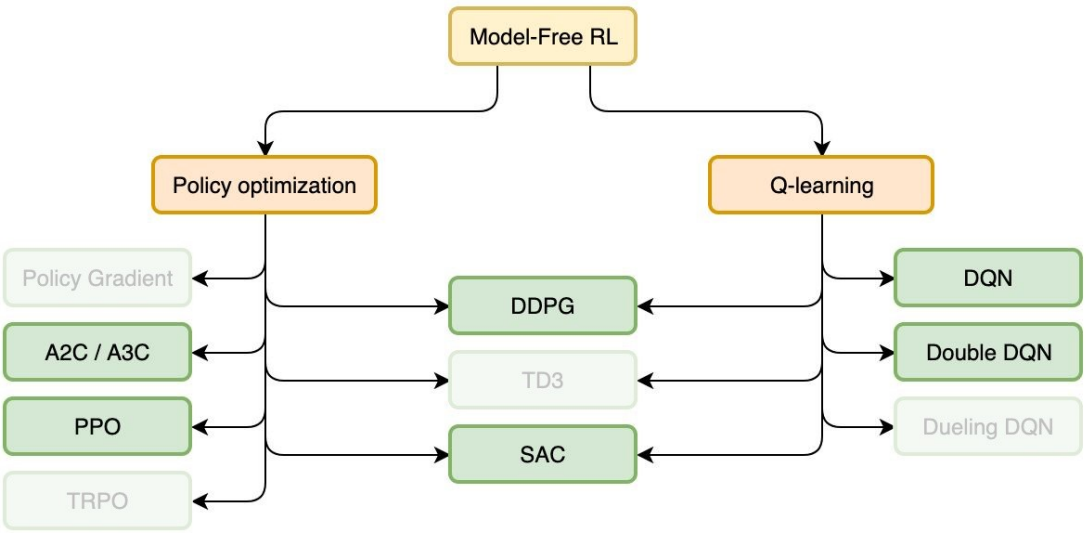


Figure 6.21: RL algorithms experimented with for the Hole Cleaning problem using the defined model and running episodes in OpenLab simulator

### Methodology

Multiple Reinforcement Learning algorithms are used to train the model - firstly with a discrete action space and experimented with DQN and Double DQN algorithms. The intention was to get a baseline to compare advanced algorithms with. One important limitation of the model was that there were limited steps and running them on the simulator was costly. To work around this, more updates were done using the same sample in the DQN algorithms. To contrast the Q-learning algorithms, an Advantage Actor-Critic (A2C) algorithm was implemented and the performance was compared. However, it is expected that A2C would take longer than DQN and variants to train.

In the continuous action space, the Deep Deterministic Policy Gradient (DDPG) algorithm was first implemented. As an off-policy algorithm it has the potential to provide improved sample efficiency which would be beneficial for the domain at hand. Current domain's sensitivity to hyper-parameters is yet to be tested. Proximal Policy Optimization PPO is trained as it is touted to be much less prone to hyper-parameter tuning and hence

could provide a robust reward curve. Finally, Soft Actor-Critic (SAC) was implemented for its many benefits including inherent state space exploration and robustness against local optima.

Initial experiments were run for tuning the action space, state space and reward function. During this time, the OpenLab simulator was manually controlled for multiple episodes to better understand the effect of change in actions on the cuttings bed. Once the model and environment was fixed, identical experiments were run for all the algorithms. All the experiments ran for 100 episode and in each episode the agent drilled till 20 m depth. On each step, the simulation is advanced by 1 minute as the changes in states are quite slow and simulator closely follows the real-world field parameters. Each episode took 4 minutes to run and training an algorithm took roughly 7 hours.

### 6.2.3 Simulation Results

First, the performance of different DQN variants are compared. In the experiments performed, two variants of DQN with batch size 32 and with batch size 128 are analuzed. As the work is in a sample scarce domain, updates to the neural network are done using the same number of samples. Increasing the batch size to 128 would increase the sample size from the replay buffer and 4 times more updates from the same number of environment steps would be possible, potentially improving the learning. Figure 6.22 shows the result comparison between these three DQN variants. No significant improvement in either of the three variants is observed. The hypothesis is that these algorithms require more episodes to learn efficiently.

Comparing the rewards for all the discrete action space algorithms in Figure 6.23, it is seen that A2C has outperformed all the DQN variants in terms of initial as well as final rewards. However, this improvement could be attributed to the constant epsilon used for training the A2C network. While training A2C network, a constant epsilon value of 0.1 was used which would mostly take greedy steps right from the start. In contrast, all the DQN variants were trained with an initial epsilon of 1 and a decay of 0.95. This explains the

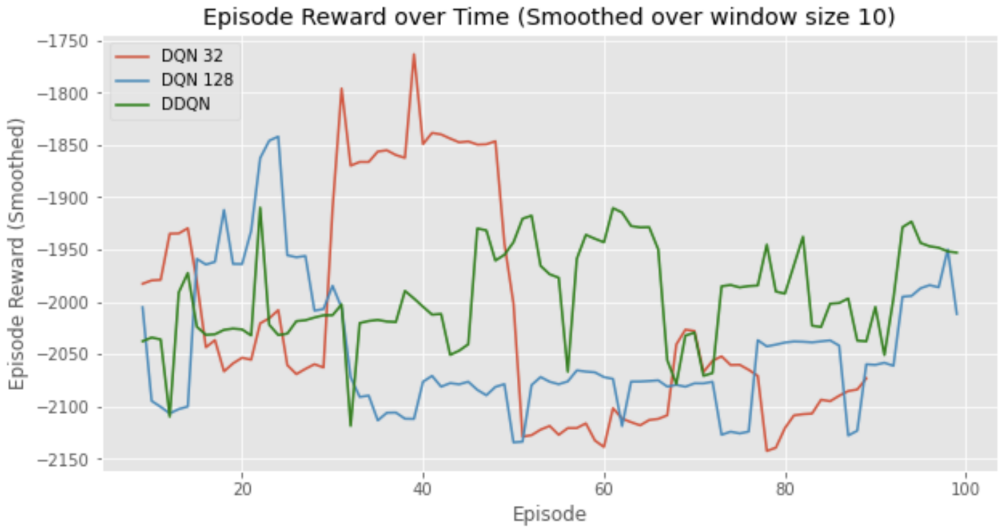


Figure 6.22: Rewards comparison for DQN-32, DQN-128 and DDQN shows no significant improvement in learning with three of the Deep Q-Learning variations.

initial jump in rewards and subsequent increase in A2C. Although, A2C performed better it is believed that the current setup is simplified and such low initial epsilon would severely limit the state exploration.

In the continuous action space (Figure 6.24), it is seen that PPO starts out with very high rewards due to a different sampling strategy. However, SAC algorithm shows the highest increase in rewards. The sudden increase in the rewards after 50th episode is because till the 50th episode, actions are taken randomly and after that the policy governs the actions. Once that comes into effect, SAC learns quickly and consistently. PPO in comparison show slower learning rate but still the rewards keep on increasing.

Contrasting between the continuous and discrete action space algorithms (Figure 6.25), it is seen that both of them have comparable performance. A2C's high rewards are suspected to be because of a simpler model formulation which allows the greedy approach to outperform higher exploration approaches. SAC has the steepest reward curve once policy control kicks in. DDPG flattens out in terms of rewards and performs worse than DQN variants.

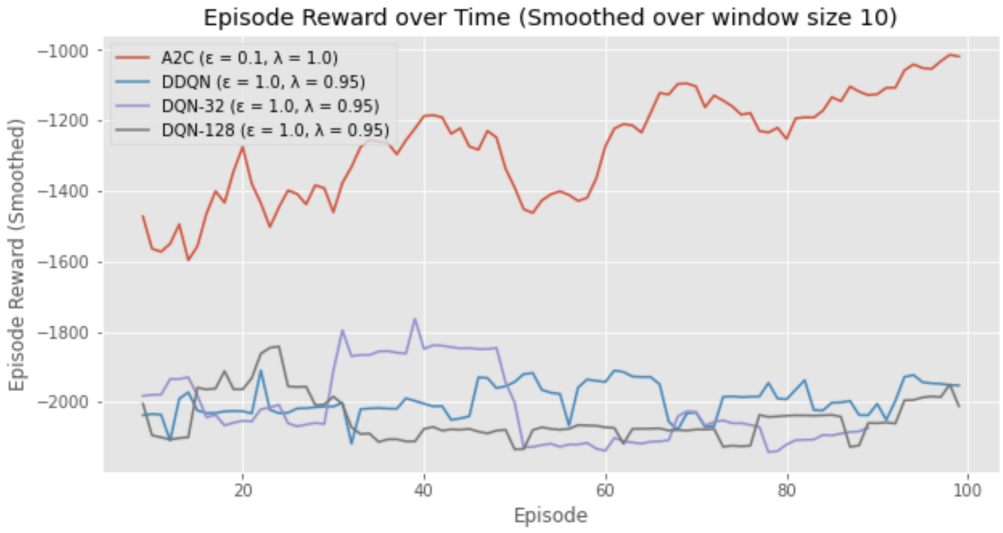


Figure 6.23: Rewards for discrete action space algorithms highlighting improvement in learning process with the use of Advantage Actor Critic (A2C) method

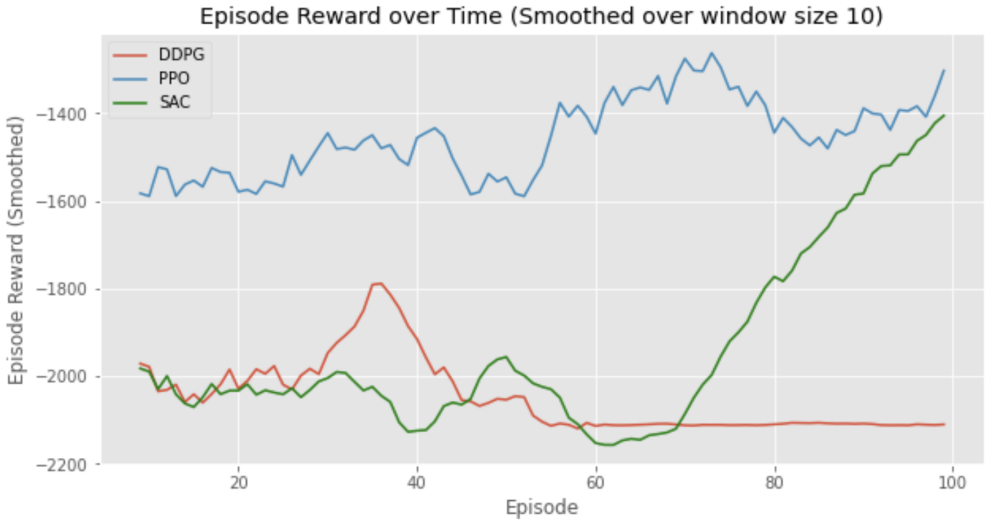


Figure 6.24: Rewards for continuous action space RL algorithms shows that Soft-Actor-Critic (SAC) method performed significantly better in comparison to other RL algorithms.

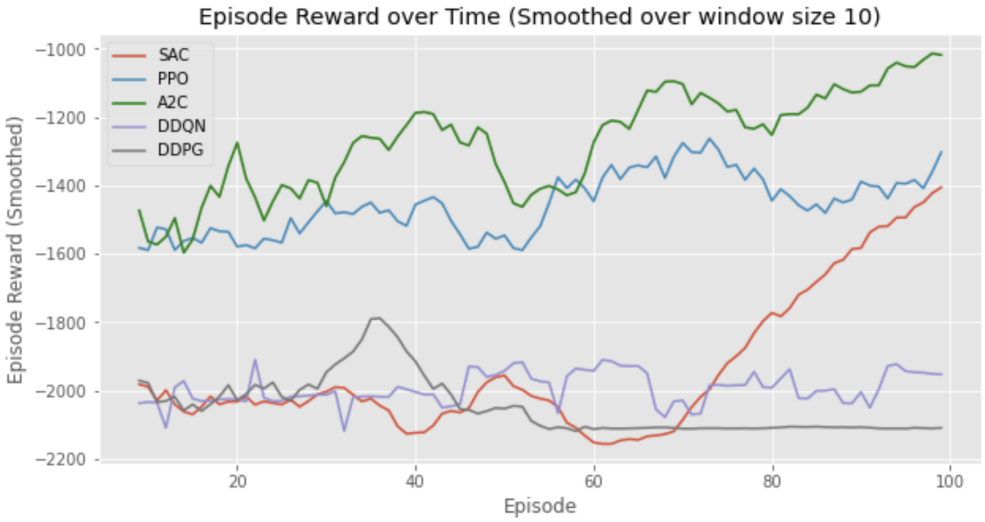


Figure 6.25: Rewards comparison for discrete and continuous action space algorithms - A2C, SAC, DDPG. Actor Critic methods yielded greater performance both in discrete (A2C) and continuous (SAC) setups.

This simplified model formulation demonstrates that RL agents have the potential to learn the course of action in the hole cleaning problem. In particular, it is observed that the continuous action space algorithms show more stability in terms of learning in this domain. Training results show that having a greedy approach is beneficial and discrete actions are a reasonable simplification for the Hole Cleaning problem. The gym environment developed in this work will be useful in future developments when integrating into the steering models.

## 6.3 Setup for Integration into Steering Models

The Steering models developed in Chapters 4 & 5 treat the directional drilling downhole environment as an MDP where either geological data or well plans are provided to the drilling agent and the agent learns to optimize well placement. A model setup for integrating the other components of drilling that have been covered in this chapter - dysfunction mitigation and wellbore cleaning with the steering system is explored now. The implementation of such an integrated system is not in the scope of this current work. This topic is further discussed

in Chapter 7 as a recommendation for future work.

### 6.3.1 Model Setup

A high-level MDP setup is proposed for the system that can be utilized for RL implementation, with the assumption that a unified interactive learning environment is available. The different components of the MDP framework - Agent, Environment, States, Actions, Rewards, and Goals are defined accordingly. Stochasticity is incorporated into the model by introducing noise into some of the actions. Details of the model setup are as below.

**Agent**

The drilling agent in this setup mimics the roles of the driller, directional driller on the rig surface, a drilling engineer at an RTOC, and also an operational geologist. The goal of the agent is to reach the oil zone as per the provided geological models and to keep drilling in the zone until the end of the target zone while also ensuring efficient cuttings transport, control of vibrations, and ROP optimization. In the real-world, the driller, directional driller, drilling engineer, and operational geologist work together to achieve drilling in the target zone with minimal tortuosity and faster operations. The agent works towards achieving the same goals. At each step of an episode, the agent observes the states and takes actions so as to maximize the rewards in achieving this goal. RL is long-term goal oriented.

**Environment**

The observations that the agent makes in the environment which define the states are as listed below.

**State Space** ($\mathcal{S}$)

- Position (x) of the drill ($x^{pos}$) = True Vertical Depth
- Position (y) of the drill ($y^{pos}$) = Inclination ($\theta$)
- Gamma value of the formation ($\gamma$)
- Resistivity of the formation ($R$)

- Dogleg Severity ($dg$)
- Wellbore Tortuosity ($\mathcal{T}$)
- Estimated Distance to Target (TD)
- Triaxial Accelerometer measurements ($a_x, a_y, a_z$)
- Stickslip Severity ($SS_t$)
- Predicted Stickslip ($SS_{t+n}$)
- BHA Configuration
- Cuttings Concentration
- Cuttings Bed Height

It is assumed that all the above observations are available in real-time while drilling and the measurements are made near-bit.

**Action Space**

A continuous action space is defined for the model where the agent can steer left or right adhering to mechanical constraints. Two different scenarios - one with a constant drilling (or Steering) speed and the other with drill speed as an action are implemented. In addition, surface control variables namely Weight-On-Bit, Pipe rotation, and circulation flow rate are also possible actions.

**Action Space** ($\mathcal{A}$)

- Steer: $[-6°, +6°]$ ($\dot{\theta}$)
- Weight-On-Bit ($WOB$)
- Drill-pipe Rotational speed ($RPM$)
- Circulation Flow Rate ($GPM$)

**Stochasticity**

Uncertainties of the real-world drilling environments such as hard formations, bit walk are incorporated into the simulator by intentionally adding Gaussian noise to the actions at each step of the episode. Furthermore, uncertainties in the formation tendencies, errors in estimations will also have to be considered.

**Rewards**

Rewards are defined for the agent in accordance with the objectives. Consequently, the agent gets reward when it drills in the oil zone and no reward anywhere else in the formations. In order for the agent to move forward, an additional reward is given as it gets closer to the end of the target. Faster drilling will be rewarded. The agent is penalized for dogleg to avoid tortuosity in the well path, delays or slow drilling, increased vibration, and high penalty for events like tool failure.

**Reward Function** ($\mathcal{R}$)

$$\mathcal{R}(s_t, a_t, s_{t+1}) : \begin{cases} \{0, +1\}, & \text{if } s_{t+1} = \text{drilling in target zone} \\ k_1(d_t^{oil}), & \text{if } s_{t+1} = \text{drilling towards end} \\ -k_2(dg), & \text{if } s_{t+1} = \text{steering in bends} \\ +0.0001, & \text{if } s_{t+1} = \text{drilling ahead} \\ -k_3(SS), & \text{if } s_{t+1} = \text{magnitude of vibration} \\ Terminal, & \text{if } s_{t+1} = \text{tool failure} \\ Terminal, & \text{if } s_{t+1} = \text{moving in a circle} \end{cases}$$

The above configuration is presented for reference and reward shaping exercise will have to be done in order to tune the reward function. This is an iterative process that will have to be performed during the training phase.

# 7. SUMMARY

*There are two paths to achieve fully autonomous drilling: either rely on the presence of a human operator in case the autonomous system is unable to recover from an unexpected situation or design the system from scratch to never need any human assistance of any form. The second alternative may be mandatory for very constrained environments such as space exploration missions where distances and communication delays render human interventions impractical or impossible, however in the context of drilling operations, it is always possible to rely on the presence of a human operator. Taking advantage of this situation will significantly reduce the complexity of some of the challenges by eliminating the necessity of developing fool-proof systems, thus resulting in a quicker, more practical and feasible approach to autonomous drilling systems.* - paraphrasing Eric Cayeux (2022) [101].

The same approach has been adapted and several of the solutions developed in this study are based on the availability of a fallback "pilot" (the driller).

## 7.1 Conclusion

Geosteering of a hydrocarbon well involves taking decisions regarding placement of the well in target zone by accurately steering the wellbore while drilling. This process, when combined with sometimes conflicting objectives of ROP optimization, or minimizing wellbore tortuosity, brings in multitude of operational challenges. Traditionally, these operations have been carried out in a manual fashion predominantly relying on the personnel's expertise and experience. Steering advisory systems have been recently proposed that proved to aid and improve the geosteering performance. Several gaps have been identified in the current methods such as the process being reactive with decision-making at prolonged intervals typically of 90 feet, reliance on modeling complex uncertainties like bit-walk, limitations on translating the methods to more latest methodologies like RSS, to name a few. Some of these gaps are addressed in this study through an alternative approach of utilizing data-driven

artificial intelligence techniques like reinforcement learning, through a self-learning approach in an interactive environment to better utilize the breadth of information available both at the surface and downhole, to better characterize the uncertainties and to provide solutions for this multi-objective problem. The key findings of the study have been summarized below.

- The sequential decision-making problem of geosteering was formulated as a Markov Decision Process (MDP). This novel technique was shown to be capable of combining multiple objectives of optimal well placement and improved drilling performance which have been predominantly treated as independent drilling sub-systems. It was demonstrated through simulation case studies that the coupled system will be able to achieve better overall operational efficiency.

- Two specific variations of the geosteering problem have been studied. First in a semi-autonomous setting where wellpaths are available, the RL agent learned to accurately track a given trajectory. Second, the agent was trained to 'explore' hydrocarbons in a self-steering autonomous fashion. Demonstrated a proof-of-concept that reinforcement learning techniques used in autonomous vehicles and other similar domains can be utilized to efficiently 'geosteer' a directional drilling O&G well. The self-correcting methodology exemplified the feasibility of automated pro-active steering operations with minimum human intervention in the process.

- The Steering models were shown to learn in a stochastic environment affected by bit-walk. The MDP formulation and RL implementation approach was demonstrated to be an alternative and a more practical approach to modeling the complex behavior of bit-walk through the use of model-free reinforcement learning techniques.

- Developed a simulation framework that balances the trade-off between two extremes of commercial-grade high-fidelity simulators and the inaccurate real-time platforms. The high-fidelity simulators such as finite-element-method-based simulators which are computationally demanding are found to not be a good fit for an interactive learning

approach and can be replaced by the proposed physics-based near-real-time simulator. The simulator has been developed in a such a manner that it would be applicable to several more drilling scenarios. Specifically, the simulator can be extended to devise new methods, to apply reinforcement learning to automated well planning, managed pressure drilling, simulate formation tendencies, among many other potential use cases.

- Machine Learning classification techniques were used to identify drilling dysfunctions. It was proved that the use of high-frequency downhole measurements can significantly increase the accuracy of dysfunction identification while drilling. Developed a data-driven method using recurrent neural networks that is more robust than the existing methods, to predict Stickslip torsional vibration events in drillstring up to 30 seconds ahead of time using only downhole raw drilling dynamics data. The results are utilized to emphasize the importance of analysing high-frequency drilling data as opposed to reliance only on surface measurements.

- An integrated framework combining the steering operations with drilling performance subsystems such as vibration monitoring, dysfunction prediction, wellbore cleaning, rock formation identification, ROP optimization, was proposed with unified objectives.

- The knowledge acquired from this study will be beneficial for industry as well as researchers to broaden their concept related to geosteering and for implementing autonomous systems downhole. This knowledge can also be used for Geothermal extended reach well which has the potential to act as a clean, stable and unlimited source of energy.

## 7.2 Recommendations for future research

### 7.2.1 Field Validation

Simulation and training results shown in Chapters 4 & 5 correspond to the drill agent learning to either traverse a given wellbore trajectory or to self-steer into formations in a

quest for hydrocarbons. All the models have been trained and tested in a 2-D simulated environment. Performance of the learning process has been evaluated using several RL performance metrics. The next step in the process is to validate these trained models in a real-world environment. Comparison in terms of operational metrics will provide valuable insights. Some of the relevant parameters in the geosteering or drilling performance applications are as listed below.

- Increase in percentage of payzone exposure
- Improvement in quality of wellbore quantified by reduction in dogleg severity
- Reduction in amount of local tortuosities
- Increase in average ROP or reduction in total drilling time
- Reduction in severe vibration events
- Accumulation of wellbore cuttings

In lieu of appropriate real-world drilling data, a comparison of performance can be made utilizing a high fidelity drilling simulator. DDNet developed by Schlumberger is a directional drilling agent aimed at making efficient steering decisions at survey points [8]. DDNet models are compared against their proprietary high-fidelity 3-D drilling simulator. Estimated Total Reward (ETR) and some of the other metrics as described above have been compared. Fig. 7.1 shows comparison of dogleg severity (DLS) values when the DDNet agent is employed. Black solid line until about 2000 ft. (on horizontal axis) represents the drilling before engaging the agent (red line is smoothed version of DLS). Colored dots starting a little ahead of 2000 ft., represent the drilling section steered by DDNet.

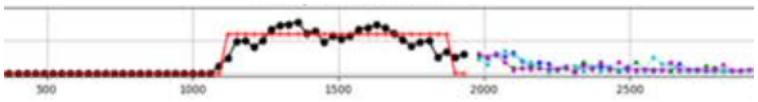


Figure 7.1: Comparison of Dogleg Severity using DDNet framework [8] - shows drilling before the agent is engaged; Black solid line until about 2000 ft. (on horizontal axis); red line is smoothed version of DLS; Colored dots starting a little ahead of 2000 ft., represent the drilling section steered by DDNet.

### 7.2.2 Integration of RL Models

The RL models in this work have been discussed in a modular fashion. Chapter 4 showed a semi-autonomous steering system, Chapter 5 showed a near-fully-autonomous geosteering system, and Chapter 6 illustrated drilling performance models for vibration control and hole cleaning. An integration framework with a basic MDP formulation is discussed towards the end of Chapter 6. Implementation of the integrated system is recommended for future work. Fig. 7.2 shows a high-level overview of the proposed models and how they tie into the proposed integrated intelligent drilling system.

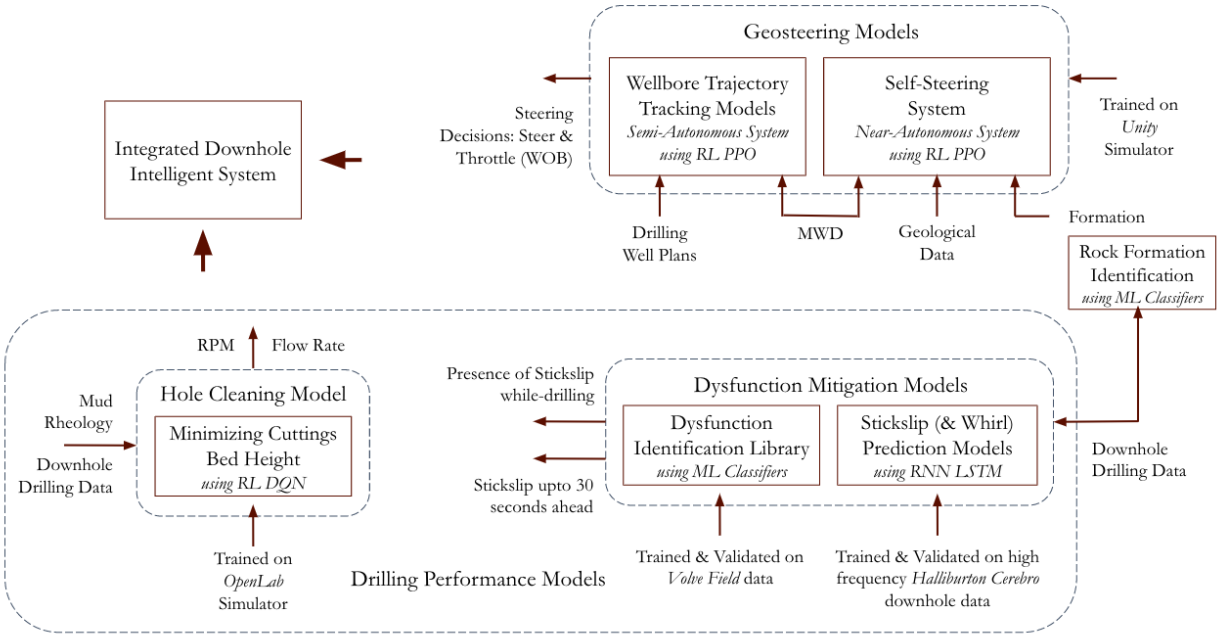


Figure 7.2: High-level Overview of Integrated Intelligent Drilling System demonstrates the interconnect between the various models presented in the study.

# REFERENCES


[1] R. Mitchell and S. Miska, *Fundamentals of drilling engineering*. Society of Petroleum Engineers, 2011.

[2] "Art of directional drilling," September 17, 2015 2015.

[3] R. Griffiths, *Well placement fundamentals*. Schlumberger, 2009.

[4] N. Demirer, U. Zalluhoglu, J. Marck, and R. Darbe, "Automated steering with real-time model-based control," in *International Petroleum Technology Conference*, OnePetro.

[5] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[6] A. B. Martinsen and A. M. Lekkas, "Curved path following with deep reinforcement learning: Results from three vessel models," in *OCEANS 2018 MTS/IEEE Charleston*, pp. 1–8, IEEE.

[7] C. Olah, "Understanding lstm networks," 2015.

[8] Y. Yu, C. Jeong, W. Chen, Y. Shen, V. Vesselinov, and S. Chambon, "Ddnet: A multi-agent decision making and evaluation in drilling with looking-ahead simulation," in *IADC/SPE International Drilling Conference and Exhibition*, OnePetro, 2022.

[9] M. R. Endsley and D. B. Kaber, "Level of automation effects on performance, situation awareness and workload in a dynamic control task," *Ergonomics*, vol. 42, no. 3, pp. 462–492, 1999.

[10] J. D. Macpherson, J. P. de Wardt, F. Florence, C. D. Chapman, M. Zamora, M. L. Laing, and F. P. Iversen, "Drilling-systems automation: Current state, initiatives, and potential impact," *SPE drilling  completion*, vol. 28, no. 04, pp. 296–308, 2013.

[11] U. N. DESA, "Key findings of world population prospects 2019: Highlights," *New York (US), United Nations Department for Economic and Social Affairs*, 2019.

[12] U. Nations, "Revision of world urbanization prospects," *United Nations: New York, NY, USA*, 2018.

[13] "Global geothermal energy market will reach usd 9 billion by 2025: Zion market research."

[14] J. P. de Wardt, M. Behounek, C. Chapman, and D. Putra, "Drilling systems automation-preparing for the big jump forward," in *SPE/IADC Drilling Conference*, OnePetro.

[15] S. C. Geekiyanage, E. A. Løken, D. Sui, and T. Wiktorski, "Architectures and algorithms for a smart drilling robot," in *International Conference on Offshore Mechanics and Arctic Engineering*, vol. 58875, p. V008T11A028, American Society of Mechanical Engineers.

[16] M. K. Bak, "Model based design of electro-hydraulic motion control systems for offshore pipe handling equipment," 2014.

[17] C. Carpenter, "Real-time analysis for remote operations centers," *Journal of Petroleum Technology*, vol. 65, no. 09, pp. 160–163, 2013.

[18] C. Temizel, C. H. Canbaz, H. Aydin, B. F. Hosgor, D. Y. Kayhan, and R. Moreno, "A comprehensive review of the fourth industrial revolution ir 4.0 in oil and gas industry," in *SPE/IATMI Asia Pacific Oil  Gas Conference and Exhibition*, OnePetro.

[19] E. Z. Losoya, *Data-Driven Numerical Simulation And Optimization Using Machine Learning Methods for Drilling Dysfunction Identification and Automation.* Thesis, 2022.

[20] M. R. Isbell, A. C. Groover, B. Farrow, and D. Hasler, "What drilling automation can teach us about drilling wells," in *SPE Annual Technical Conference and Exhibition*, OnePetro.

[21] M. R. Endsley, *Automation and situation awareness*, pp. 163–181. CRC Press, 2018.

[22] D. B. Kaber, *The effect of level of automation and adaptive automation on performance in dynamic control environments.* Texas Tech University, 1996.

[23] R. Parasuraman, "Designing automation for human use: empirical studies and quantitative models," *Ergonomics*, vol. 43, no. 7, pp. 931–951, 2000.

[24] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation," *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.

[25] R. Foster and R. Macmillan, "High speed telemetry on wired drill pipe, history, and impact on drilling process," in *Offshore Technology Conference*, OnePetro.

[26] R. Foster and R. Macmillan, "High speed telemetry on wired drill pipe, history, and impact on drilling process," in *Offshore Technology Conference*, OnePetro.

[27] O. Sehsah, A. Ghazzawi, G. Vie, T. Al-Tajar, A. Ali, A. Al-Mohammed, M. Itani, S. Ullah, H. Escalera, and M. Balka, "Intelligent drilling system: Expanding the envelope of wired drill pipe," in *Abu Dhabi International Petroleum Exhibition  Conference*, OnePetro.

[28] J. Macpherson, I. Roders, K. Schoenborn, R. Mieting, and F. Lopez, "Smart wired pipe: Drilling field trials," in *SPE/IADC International Drilling Conference and Exhibition*, OnePetro.

[29] C. Torres-Verdín and J. Zhou, "Introduction to this special section: Well geosteering," *The Leading Edge*, vol. 34, no. 5, pp. 494–494, 2015.

[30] R. Chemali, M. Bittar, F. Hveding, M. Wu, and M. Dautel, "Improved geosteering by integrating in real time images from multiple depths of investigation and inversion of azimuthal resistivity signals," *SPE Reservoir Evaluation  Engineering*, vol. 13, no. 02, pp. 172–178, 2010.

[31] B. S. Aadnoy, I. Cooper, S. Miska, R. F. Mitchell, and M. L. Payne, *Advanced drilling and well technology.* SPE Richardson, Texas, 2009.

[32] J. A. Short, *Introduction to directional and horizontal drilling.* Pennwell Corporation, 1993.

[33] T. Inglis, *Directional drilling*, vol. 2. Springer Science  Business Media, 1988.

[34] P. Aird, *Deepwater drilling: well planning, design, engineering, operations, and technology application.* Gulf Professional Publishing, 2018.

[35] S. Devereux, *Practical well planning and drilling manual.* PennWell Tulsa, OK, 1998.

[36] X. Liu, R. Liu, and M. Sun, "New techniques improve well planning and survey calculation for rotary-steerable drilling," in *IADC/SPE Asia Pacific drilling technology conference and exhibition*, OnePetro.

[37] A. T. Bourgoyne, K. K. Millheim, M. E. Chenevert, and F. S. Young, *Applied drilling engineering*, vol. 2. Society of Petroleum Engineers Richardson, 1986.

[38] G. Liu, *Applied Well Cementing Engineering.* Gulf Professional Publishing, 2021.

[39] Q. Yang, Z. Wang, and Y. Zhang, "Introduction the mud pulse transmission methods of mwd system," *Hunan Agricultural Machinery*, vol. 37, no. 3, pp. 27–28, 2010.

[40] X. Liu, R. Liu, and M. Sun, "New techniques improve well planning and survey calculation for rotary-steerable drilling," in *IADC/SPE Asia Pacific drilling technology conference and exhibition*, OnePetro.

[41] W. Lesso and S. Kashikar, "The principles and procedures of geosteering," in *IADC/SPE drilling conference*, OnePetro.

[42] E. Christiaansen, D. J. Bourgeois, C. Macdonald, K. C. Longmuir, T. A. Natras, I., Mcilreath, and EnCana, "Aade-07-ntce-13 proactive geosteering with directional deep resistivity and rotary steerable tool in thin coalbed methane ( cbm ) reservoirs,"

[43] W. Lesso and S. Kashikar, "The principles and procedures of geosteering," in *IADC/SPE drilling conference*, OnePetro.

[44] S. Hank, "Innovation delivers practical solutions to real-world drilling challenges [j]," *The American Oil Gas Reporter*, vol. 44, no. 1, pp. 68–77, 2002.

[45] M. Reeves, J. D. MacPherson, R. Zaeper, D. R. Bert, J. Shursen, W. K. Armagost, D. S. Pixton, and M. Hernandez, "High speed drill string telemetry network enables new real time drilling and measurement technologies," in *IADC/SPE Drilling Conference*, OnePetro.

[46] L. Xinping, F. Jun, and J. Youhai, "Application status and prospect of lwd data transmission technology," *Well Logging Technology*, vol. 32, no. 3, pp. 249–253, 2008.

[47] J. Wang and H. Wen, *The Research of Data Transmission Technology in Measurement While Drilling*. Scientific Research Publishing, Inc. USA, 2018.

[48] C. Klotz, I. Wassermann, and D. Hahn, "Highly flexible mud-pulse telemetry: a new system," in *SPE Indian Oil and Gas Technical Conference and Exhibition*, OnePetro.

[49] Y. Liang, X. Ju, A. Li, C. Li, Z. Dai, and L. Ma, "The process of high-data-rate mud pulse signal in logging while drilling system," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[50] L. X. H. X. T. Vulin and Y. Chunguo, "Developments of electromagnetic measurement while drilling," *Petroleum Drilling Techniques*, 2006.

[51] J. Fay, H. Fay, and A. Couturier, "Wired pipes for a high-data-rate MWD system," 1992.

[52] M. J. Manning, J. D. Macpherson, D. E. Taylor, N. Baksh, C. Peveto, L. Farnsworth, and S. R. Lemke, "Wired pipe-enabled logging while drilling applications," vol. d, 2008.

[53] C. McCartney, S. Allen, M. Hernandez, D. MacFarlane, A. Baksh, and M. E. Reeves, "Step-change improvements with wired-pipe telemetry," 2009.

[54] T. Gee, S. Maus, A. M. Mitkus, K. McCarthy, D. M. Velozzi, and R. Mottahedeh, "Improving wellbore placement accuracy using stratigraphic misfit heatmaps," in *Asia*

*Pacific Unconventional Resources Technology Conference, Brisbane, Australia, 18-19 November 2019*, pp. 386–394, Unconventional Resources Technology Conference.

[55] S. Maus and S. DeVerse, "Enhanced wellbore placement accuracy using geomagnetic in-field referencing and multi-station correction," in *SPE/AAPG/SEG Unconventional Resources Technology Conference*, OnePetro.

[56] J. Pitcher and F. Gallice, "Beyond gamma ray: Determining a geosteering program for a new unconventional reservoir," *The Leading Edge*, vol. 34, no. 5, pp. 514–522, 2015.

[57] J. Zhou, "Uncertainty in geosteering and interpretation of horizontal wells—the necessity for constraints and geometric models," *The Leading Edge*, vol. 34, no. 5, pp. 492–499, 2015.

[58] J. Gremillion*, M. Flowers, N. Tvrdy, M. Okoro, and Z. Newnam, "Selection of logging while drilling measurements for geosteering of horizontal wells in unconventional reservoirs," in *Unconventional Resources Technology Conference, Denver, Colorado, 22-24 July 2019*, pp. 1344–1360, Unconventional Resources Technology Conference (URTeC); Society of ....

[59] J. Pitcher, N. Clegg, C. Burinda, R. Cook, C. Knutson, M. Scott, and T. Løseth, "Advances in geosteering technology: From simple to complex solutions," in *IADC/SPE Drilling Conference and Exhibition*, OnePetro.

[60] M. Bittar and A. Aki, "Advancement and economic benefit of geosteering and well-placement technology," *The Leading Edge*, vol. 34, no. 5, pp. 524–528, 2015.

[61] D. Seifert, S. Aramco, R. Chemali, M. Bittar, G. Althoff, and A. Lotfy, "Hydrocarbon reservoirs where proactive geosteering is most likely to succeed," in *SPE/IADC Drilling Conference and Exhibition*, OnePetro.

[62] M. Bittar, R. Chemali, M. Morys, J. Wilson, F. Hveding, S. Li, S. Knizhnik, and D. M. Halverson, "The depth-of-electrical image a key parameter in accurate dip computation and geosteering," in *SPWLA 49th Annual Logging Symposium*, OnePetro.

[63] S. DeVerse and S. Maus, "Optimizing lateral well spacing by improving directional survey accuracy," in *SPE Liquids-Rich Basins Conference-North America*, OnePetro.

[64] P. Berger and R. Sele, "Improving wellbore position accuracy of horizontal wells by using a continuous inclination measurement from a near bit inclination mwd sensor," *Journal of Canadian Petroleum Technology*, vol. 39, no. 10, 2000.

[65] S. Maus and S. DeVerse, "Enhanced wellbore placement accuracy using geomagnetic in-field referencing and multi-station correction," in *SPE/AAPG/SEG Unconventional Resources Technology Conference*, OnePetro.

[66] S. F. Noynaert, "What i wish my geologist knew about drilling: Geosteering from a drilling engineer's perspective," in *SPE/AAPG/SEG Unconventional Resources Technology Conference*, OnePetro.

[67] R. Woodward and S. Noynaert, "If it's so easy, why don't you come do it yourself? a response to what i wish my geologist knew about drilling: A drilling engineer's view of geosteering," in *SPE/AAPG/SEG Unconventional Resources Technology Conference*, OnePetro.

[68] R. E. Studer and L. P. Y. Macresy, "Improved bha sag correction and uncertainty evaluation brings value to wellbore placement," in *SPE Annual Technical Conference and Exhibition*, OnePetro.

[69] G. A. Bordakov, A. V. Kostin, J. C. Rasmus, D. Heliot, H. Laastad, and E. J. Stockhausen, "Improving lwd image and formation evaluation by utilizing dynamically corrected drilling-derived lwd depth and continuous inclination and azimuth measurements," in *SPE Annual Technical Conference and Exhibition*, OnePetro.

[70] E. Stockhausen and W. Lesso, "Continuous direction and inclination measurements lead to an improvement in wellbore positioning," in *SPE/IADC Drilling Conference*, OnePetro.

[71] T. Arbus and S. Wilson, "Cybersteering: Automated geosteering by way of distributed computing and graph databases in the cloud," in *SPE/AAPG/SEG Unconventional Resources Technology Conference*, OnePetro.

[72] B. Kristoffersen, T. Silva, M. Bellout, and C. Berg, "An automatic well planner for efficient well placement optimization under geological uncertainty," in *ECMOR XVII*, vol. 2020, pp. 1–16, European Association of Geoscientists  Engineers.

[73] H. Winkler, "Geosteering by exact inference on a bayesian network," *Geophysics*, vol. 82, no. 5, pp. D279–D291, 2017.

[74] J. Fierstien, H. Winkler, P. Strauss, and A. Klokov, "Optimization and drilling of horizontal wells using a bayesian network," in *Unconventional Resources Technology Conference, Houston, Texas, 23-25 July 2018*, pp. 152–157, Society of Exploration Geophysicists, American Association of Petroleum ....

[75] C. Dupuis and J.-M. Denichou, "Automatic inversion of deep-directional-resistivity measurements for well placement and reservoir description," *The Leading Edge*, vol. 34, no. 5, pp. 504–512, 2015.

[76] G. C. Downton, "Systems modeling and design of automated-directional-drilling systems," *SPE Drilling  Completion*, vol. 30, no. 03, pp. 212–232, 2015.

[77] J. Sugiura, R. Samuel, J. Oppelt, G. Ostermeyer, J. Hedengren, and P. Pastusek, "Drilling modeling and simulation: Current state and future goals," in *SPE/IADC Drilling Conference and Exhibition*, OnePetro.

[78] C. P. Andrade, J. L. Saavedra, A. Tunkiel, and D. Sui, "Rotary steerable systems: mathematical modeling and their case study," *Journal of Petroleum Exploration and Production Technology*, vol. 11, no. 6, pp. 2743–2761, 2021.

[79] C. Pehlivantürk, J. D'Angelo, D. Cao, D. Chen, P. Ashok, and E. Van Oort, "Slide drilling guidance system for directional drilling path optimization," in *SPE/IADC International Drilling Conference and Exhibition*, OnePetro.

[80] N. Panchal, M. T. Bayliss, and J. F. Whidborne, "Vector based kinematic closed-loop attitude control-system for directional drilling," *IFAC Proceedings Volumes*, vol. 45, no. 8, pp. 78–83, 2012.

[81] M. Bayliss, C. Bogath, and J. Whidborne, "Mpc-based feedback delay compensation scheme for directional drilling attitude control," in *SPE/IADC Drilling Conference and Exhibition*, OnePetro.

[82] N. Demirer, U. Zalluhoglu, J. Marck, H. Gharib, and R. Darbe, "A model predictive control method for autonomous directional drilling," in *SPE Annual Technical Conference and Exhibition*, OnePetro.

[83] N. Panchal, M. T. Bayliss, and J. F. Whidborne, "Attitude control system for directional drilling bottom hole assemblies," *IET control theory  applications*, vol. 6, no. 7, pp. 884–892, 2012.

[84] A. Kyllingstad and P. J. Nessjoen, "Hardware-in-the-loop simulations used as a cost-efficient tool for developing an advanced stick-slip prevention system," *IADC / SPE Drilling Conference and Exhibition*, 2010.

[85] S. W. Lai, J. Ng, A. Eddy, S. Khromov, D. Paslawski, R. van Beurden, L. Olesen, G. S. Payette, and B. J. Spivey, "Large-Scale Deployment of a Closed-Loop Drilling Optimization System: Implementation and Field Results," *SPE Drilling & Completion*, vol. 36, pp. 47–62, 03 2021.

[86] M. Ignova, M. Montois, and K. Mantle, "An automated trajectory control for drilling operations," in *SPE Middle East Oil and Gas Show and Conference*, vol. Day 4 Thu, March 21, 2019, (D041S035R002).

[87] C. Hansen, M. Stokes, R. Mieting, F. Quattrone, V. Klemme, K. Nageshwara Rao, I. Wassermann, and R. Zaeper, "Automated trajectory drilling for rotary steerable systems," in *IADC/SPE International Drilling Conference and Exhibition*, vol. Day 3 Thu, March 05, 2020, (D101S017R005).

[88] Z. Liu and R. Samuel, "Probabilistic real-time trajectory control considering uncertainties of drilling parameters and rock properties," in *IADC/SPE Drilling Conference and Exhibition*, vol. Day 1 Tue, March 06, 2018, (D011S006R002).

[89] S. Coffey and A. Groover, "Achieving automated directional drilling across us onshore basins," in *IADC/SPE International Drilling Conference and Exhibition*, OnePetro.

[90] C. Gillan, M. Isbell, and T. Visitew, "Automated directional drilling software and remote operations centers drive rig fleet well delivery improvement," in *IADC/SPE Drilling Conference and Exhibition*, OnePetro.

[91] B. Chmela, S. Kern, T. Quarles, S. Bhaduri, R. Goll, and C. Van, "Directional drilling automation: Human factors and automated decision-making," in *IADC/SPE International Drilling Conference and Exhibition*, OnePetro.

[92] R. Wylie, K. McClard, and J. de Wardt, "Automating directional drilling: Technology adoption staircase mapping levels of human interaction," in *SPE Annual Technical Conference and Exhibition*, OnePetro.

[93] T. Nguyen, R. G. Gosine, and P. Warrian, "A systematic review of big data analytics for oil and gas industry 4.0," *IEEE access*, vol. 8, pp. 61183–61201, 2020.

[94] M. Mohammadpoor and F. Torabi, "Big data analytics in oil and gas industry: An emerging trend," *Petroleum*, vol. 6, no. 4, pp. 321–328, 2020.

[95] N. Klyuchnikov, A. Zaytsev, A. Gruzdev, G. Ovchinnikov, K. Antipova, L. Ismailova, E. Muravleva, E. Burnaev, A. Semenikhin, A. Cherepanov, *et al.*, "Data-driven model for the identification of the rock type at a drilling bit," *Journal of Petroleum science and Engineering*, vol. 178, pp. 506–516, 2019.

[96] A. V. Timonov, R. A. Khabibullin, N. S. Gurbatov, A. R. Shabonas, and A. V. Zhuchkov, "Automated geosteering optimization using machine learning," in *Abu Dhabi International Petroleum Exhibition & Conference*, OnePetro, 2021.

[97] M. Shahriari, D. Pardo, B. Moser, and F. Sobieczky, "A deep neural network as surrogate model for forward simulation of borehole resistivity measurements," *Procedia Manufacturing*, vol. 42, pp. 235–238, 2020.

[98] B. S. Kristoffersen, M. C. Bellout, T. L. Silva, and C. F. Berg, "An automatic well planner for complex well trajectories," *Mathematical Geosciences*, vol. 53, no. 8, pp. 1881–1905, 2021.

[99] A. Bilinchuk, F. F. Khaliullin, A. Sitnikov, A. Pustovskikh, A. Margarit, I. Zhdanov, and T. Andzhukaev, "Automated solution to unlock base production potential (russian)," *Neftyanoe khozyaystvo-Oil Industry*, vol. 2016, no. 12, pp. 84–86, 2016.

[100] B. Alenezi, N. Diaz, D. Qaddoura, P. Odiase, A. Al-Mutairi, F. Otaibi, and D. Chawla, "Methodology for planning and execution of multilateral well design together with enhanced geo-navigation techniques for optimum well placement in complex geological environment," in *Abu Dhabi International Petroleum Exhibition & Conference*, OnePetro, 2020.

[101] E. Cayeux, B. Daireaux, A. Ambrus, R. Mihai, and L. Carlsen, "Autonomous decision-making while drilling," *Energies*, vol. 14, no. 4, p. 969, 2021.

[102] Y. Liu, J. Marck, K. Tian, N. Demirer, and V. Pho, "An integrated directional drilling simulator with steering advisor and self-learning algorithm," in *International Petroleum Technology Conference*, OnePetro, 2022.

[103] U. Zalluhoglu, N. Demirer, J. Marck, H. Gharib, and R. Darbe, "Steering advisory system for rotary steerable systems," in *SPE/IADC International Drilling Conference and Exhibition*, OnePetro, 2019.

[104] J. B. Keller and R. Bellman, *Stochastic equations and wave propagation in random media*, vol. 16. American Mathematical Society Providence, RI, 1964.

[105] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley  Sons, 2014.

[106] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[107] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[108] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[109] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International conference on machine learning*, pp. 387–395, PMLR, 2014.

[110] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, *et al.*, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *International Conference on Machine Learning*, pp. 1407–1416, PMLR, 2018.

[111] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[112] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski, "Vizdoom: A doom-based ai research platform for visual reinforcement learning," in *2016 IEEE conference on computational intelligence and games (CIG)*, pp. 1–8, IEEE, 2016.

[113] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033, IEEE, 2012.

[114] C. Beattie, J. Z. Leibo, D. Teplyashin, T. Ward, M. Wainwright, H. Küttler, A. Lefrancq, S. Green, V. Valdés, A. Sadik, *et al.*, "Deepmind lab," *arXiv preprint arXiv:1612.03801*, 2016.

[115] M. Johnson, K. Hofmann, T. Hutton, and D. Bignell, "The malmo platform for artificial intelligence experimentation.," in *IJCAI*, pp. 4246–4247, Citeseer, 2016.

[116] E. Coumans, Y. P. Bai, and A. PyBullet, "a python module for physics simulation for games, robotics and machine learning. 2016."

[117] E. Z. Losoya, E. Gildin, S. F. Noynaert, Z. Medina-Zetina, T. Crain, S. Stewart, J. Hicks, *et al.*, "An open-source enabled drilling simulation consortium for academic and commercial applications," in *SPE Latin American and Caribbean Petroleum Engineering Conference*, Society of Petroleum Engineers, 2020.

[118] D. Kamran, J. Zhu, and M. Lauer, "Learning path tracking for real car-like mobile robots from simulation," in *2019 European Conference on Mobile Robots (ECMR)*, pp. 1–6, IEEE.

[119] G. Hess and W. Ljungbergh, "Deep deterministic path following," *arXiv preprint arXiv:2104.06014*, 2021.

[120] S. Wang, X. Yin, P. Li, M. Zhang, and X. Wang, "Trajectory tracking control for mobile robots using reinforcement learning and pid," *Iranian Journal of Science and Technology, Transactions of Electrical Engineering*, vol. 44, no. 3, pp. 1059–1068, 2020.

[121] E. Meyer, H. Robinson, A. Rasheed, and O. San, "Taming an autonomous surface vehicle for path following and collision avoidance using deep reinforcement learning," *IEEE Access*, vol. 8, pp. 41466–41481, 2020.

[122] L. Jiang, Y. Wang, L. Wang, and J. Wu, "Path tracking control based on deep reinforcement learning in autonomous driving," in *2019 3rd Conference on Vehicle Control and Intelligence (CVCI)*, pp. 1–6, IEEE.

[123] J. Ma, H. Xie, K. Song, and H. Liu, "Self-optimizing path tracking controller for intelligent vehicles based on reinforcement learning," *Symmetry*, vol. 14, no. 1, p. 31, 2022.

[124] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30, IEEE.

[125] E. Zarate Losoya, *Real-time Control and Vibrations Analysis of a Completely Automated Miniaturized Rig.* PhD thesis, 2016.

[126] F. Florence, C. Chapman, J. Macpherson, and M. Cavanaugh, "Implementation of drilling systems automation-halifax workshop summary: Industry standards, business models and next steps," in *SPE Annual Technical Conference and Exhibition*, OnePetro, 2015.

[127] B. Lesso, M. Ignova, F. Zeineddine, J. Burks, and B. Welch, "Testing the combination of high frequency surface and downhole drilling mechanics and dynamics data under a variety of drilling conditions," in *SPE/IADC Drilling Conference and Exhibition*, OnePetro, 2011.

[128] E. Z Losoya, N. Vishnumolakala, E. Gildin, S. Noynaert, Z. Medina-Cetina, J. Gabelmann, D. Glowka, and R. Houston, "Machine learning based intelligent downhole drilling optimization system using an electromagnetic short hop bit dynamic measurements," in *SPE Annual Technical Conference and Exhibition*, OnePetro, 2020.

[129] Equinor, "Disclosing all volve data," 2018.

[130] M. H. Nordin, L. K. Looi, P. Slagel, M. H. Othman, A. R. Affandi, and M. S. Zurhan, "Minimising torsional vibration due to stick slip using z technology for drilling energy efficiency in multiple hard stringers field in offshore malaysia," in *International Petroleum Technology Conference*, OnePetro, 2021.

[131] Y. Zha and S. Pham, "Monitoring downhole drilling vibrations using surface data through deep learning," in *SEG Technical Program Expanded Abstracts 2018*, pp. 2101–2105, Society of Exploration Geophysicists, 2018.

[132] F. Clayer, J. Vandiver, and H. Lee, "The effect of surface and downhole boundary conditions on the vibration of drillstrings," in *SPE Annual Technical Conference and Exhibition*, OnePetro, 1990.

[133] C. Tian and R. N. Horne, "Recurrent neural networks for permanent downhole gauge data analysis," in *SPE Annual Technical Conference and Exhibition*, OnePetro, 2017.

[134] A. ElSaid, B. Wild, J. Higgins, and T. Desell, "Using lstm recurrent neural networks to predict excess vibration events in aircraft engines," in *2016 IEEE 12th International Conference on e-Science (e-Science)*, pp. 260–269, IEEE, 2016.

[135] L. Youwei, J. Zhaobing, and T. Kun, "Predictions to the water level of changjiang waterway employing deep learning algorithm lstm," in *The 31st International Ocean and Polar Engineering Conference*, OnePetro, 2021.

[136] B. Justin, "Applying deep learning to time series forecasting with tensorflow," *MapR. June*, vol. 10, 2017.

[137] N. Vishnumolakala, D. M. Murphy, T. Nguyen, E. Z. Losoya, V. R. Kesireddy, and E. Gildin, "Predicting dysfunction vibration events while drilling using lstm recurrent neural networks," in *SPE/IATMI Asia Pacific Oil & Gas Conference and Exhibition*, OnePetro, 2021.

[138] T. Sifferman and T. E. Becker, "Hole cleaning in full-scale inclined wellbores," *Spe Drilling Engineering*, vol. 7, pp. 115–120, 1992.

[139] E. Hopkin, "Factors affecting cuttings removal during rotary drilling," *Journal of Petroleum Technology*, vol. 19, pp. 807–814, 1967.

[140] M. MohammadSalehi and N. Malekzadeh, "Optimization of hole cleaning and cutting removal in vertical, deviated and horizontal wells," 2011.

[141] O. E. Agwu, J. U. Akpabio, S. Alabi, and A. Dosunmu, "Settling velocity of drill cuttings in drilling fluids: A review of experimental, numerical simulations and artificial intelligence studies," *Powder Technology*, vol. 339, pp. 728–746, 2018.

[142] E. Cayeux, T. Mesagan, S. Tanripada, M. Zidan, and K. Fjelde, "Real-time evaluation of hole-cleaning conditions with a transient cuttings-transport model," *Spe Drilling & Completion*, vol. 29, pp. 5–21, 2014.

[143] M. Ozbayoglu, R. E. Osgouei, and E. Yuksel, "Hole cleaning performance of gasified drilling fluids in horizontal well sections," 2010.

[144] J. Feder, "Comprehensive cuttings-transport model optimizes drilling operations for hole cleaning," *Journal of Petroleum Technology*, vol. 72, pp. 57–58, 2020.

[145] H. Tombul, A. Ozbayoglu, and M. Ozbayoglu, "Computational intelligence models for piv based particle (cuttings) direction and velocity estimation in multi-phase flows," *Journal of Petroleum Science and Engineering*, vol. 172, pp. 547–558, 2019.

[146] E. Cayeux, B. Daireaux, E. Dvergsnes, G. Saelevik, and M. Zidan, "An early warning system for identifying drilling problems: An example from a problematic drill-out cement operation in the north-sea," *Distributed Computing*, 2012.

[147] M. Alawami, M. K. Bassam, S. Gharbi, and M. M. A. Rubaii, "A real-time indicator for the evaluation of hole cleaning efficiency," 2019.

[148] M. Arnø, J. Godhavn, and O. Aamo, "Deep reinforcement learning applied to managed pressure drilling," 2020.

[149] Y. Yu, W. Chen, Q. Liu, M. Chau, V. Vesselinov, and R. Meehan, "Training an automated directional drilling agent with deep reinforcement learning in a simulated environment," 2021.

[150] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[151] G. S. Saini, P. Ashok, and E. V. Oort, "Predictive action planning for hole cleaning optimization and stuck pipe prevention using digital twinning and reinforcement learning," 2020.

[152] T. Nazari, G. Hareland, and J. J. Azar, "Review of cuttings transport in directional well drilling: Systematic approach," 2010.

[153] H. Kung Hsiang, "Introduction to various reinforcement learning algorithms.," 2018.

# APPENDIX A

# RL ALGORITHMS PSUEDOCODES

A few of the Reinforcement Algorithms were briefly described in Chapter 3 of this work. Pseudocodes for these RL algorithms are provided here. This material has been adapted from various sources [153] and from graduate coursework of Dr. Guni Sharon, Assistant Professor with the Computer Science & Engineering department at Texas A&M University.

## Q-Learning

Q-Learning is an off-policy and model-free methods that has the objective to maximize the value function (Q-value).

---

$Q$-learning: Learn function $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$

---

**Require:**
  Sates $\mathcal{X} = \{1, \ldots, n_x\}$
  Actions $\mathcal{A} = \{1, \ldots, n_a\}$, $A : \mathcal{X} \Rightarrow \mathcal{A}$
  Reward function $R : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$
  Black-box (probabilistic) transition function $T : \mathcal{X} \times \mathcal{A} \to \mathcal{X}$
  Learning rate $\alpha \in [0, 1]$, typically $\alpha = 0.1$
  Discounting factor $\gamma \in [0, 1]$
  **procedure** QLEARNING($\mathcal{X}$, $A$, $R$, $T$, $\alpha$, $\gamma$)
    Initialize $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ arbitrarily
    **while** $Q$ is not converged **do**
      Start in state $s \in \mathcal{X}$
      **while** $s$ is not terminal **do**
        Calculate $\pi$ according to Q and exploration strategy (e.g. $\pi(x) \leftarrow \arg\max_a Q(x, a)$)
        $a \leftarrow \pi(s)$
        $r \leftarrow R(s, a)$ ▷ Receive the reward
        $s' \leftarrow T(s, a)$ ▷ Receive the new state
        $Q(s', a) \leftarrow (1 - \alpha) \cdot Q(s, a) + \alpha \cdot (r + \gamma \cdot \max_{a'} Q(s', a'))$
        $s \leftarrow s'$
    **return** $Q$

---

### SARSA

The structure of SARSA method is similar to that Q-Learning but it is an on-policy algorithm that the Q-value based on the action performed by the current policy instead of the greedy policy.

SARSA($\lambda$): Learn function $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$

**Require:**
Sates $\mathcal{X} = \{1, \ldots, n_x\}$
Actions $\mathcal{A} = \{1, \ldots, n_a\}$, $A : \mathcal{X} \Rightarrow \mathcal{A}$
Reward function $R : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$
Black-box (probabilistic) transition function $T : \mathcal{X} \times \mathcal{A} \to \mathcal{X}$
Learning rate $\alpha \in [0, 1]$, typically $\alpha = 0.1$
Discounting factor $\gamma \in [0, 1]$
$\lambda \in [0, 1]$: Trade-off between TD and MC
**procedure** QLEARNING($\mathcal{X}$, $A$, $R$, $T$, $\alpha$, $\gamma$, $\lambda$)
    Initialize $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ arbitrarily
    Initialize $e : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ with 0. ▷ eligibility trace
    **while** $Q$ is not converged **do**
        Select $(s, a) \in \mathcal{X} \times \mathcal{A}$ arbitrarily
        **while** $s$ is not terminal **do**
            $r \leftarrow R(s, a)$
            $s' \leftarrow T(s, a)$ ▷ Receive the new state
            Calculate $\pi$ based on $Q$ (e.g. epsilon-greedy)
            $a' \leftarrow \pi(s')$
            $e(s, a) \leftarrow e(s, a) + 1$
            $\delta \leftarrow r + \gamma \cdot Q(s', a') - Q(s, a)$
            **for** $(\tilde{s}, \tilde{a}) \in \mathcal{X} \times \mathcal{A}$ **do**
                $Q(\tilde{s}, \tilde{a}) \leftarrow Q(\tilde{s}, \tilde{a}) + \alpha \cdot \delta \cdot e(\tilde{s}, \tilde{a})$
                $e(\tilde{s}, \tilde{a}) \leftarrow \gamma \cdot \lambda \cdot e(\tilde{s}, \tilde{a})$
            $s \leftarrow s'$
            $a \leftarrow a'$
    **return** $Q$

### Deep Q-Learning (DQN)

DQN is an off-policy model-free algorithm that addresses problems of correlated data and non-stationary distributions. It uses an experience replay mechanism and randomly samples and trains on previous transitions resulting in a smoother training distribution over many past behaviors.

Initialize replay memory $D$ to capacity $N$
Initialize action-value function $Q$ with random weights $\theta$
Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$
**For** episode = 1, $M$ **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  **For** $t$ = 1,T **do**
    With probability $\varepsilon$ select a random action $a_t$
    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $D$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $D$
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters $\theta$
    Every $C$ steps reset $\hat{Q} = Q$
  **End For**
**End For**

### Double Deep Q-Learning (DDQN)

The max operator in Q-learning uses the same values both to select and to evaluate an action that makes it more likely to select overestimated values, resulting in overoptimistic value estimates (Maximization bias). DDQN solves this by training two value functions. Updates are done on only one of the two value functions at each step while considering the maximum value from the other.

Initialize replay memory $D$ to capacity $N$
Initialize action-value function $Q$ with random weights $\theta$
Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$
**For** episode = 1, $M$ **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  **For** $t$ = 1,T **do**
    With probability $\varepsilon$ select a random action $a_t$
    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $D$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $D$
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters $\theta$
    Every $C$ steps reset $\hat{Q} = Q$
  **End For**
**End For**

### Deep Deterministic Policy Gradient (DDPG)

DDPG, similar to DQN is a model-free off-policy algorithm but can be used for learning continuous actions. It uses Experience Replay and slow-learning target networks from DQN, and it is based on Policy Gradient methods. It uses off-policy data and the Bellman equation to learn the Q-function, and uses the Q-function to learn the policy.

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights $\theta^Q$ and $\theta^\mu$.
Initialize target network $Q'$ and $\mu'$ with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer $R$
**for** episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state $s_1$
  **for** t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action $a_t$ and observe reward $r_t$ and observe new state $s_{t+1}$
    Store transition $(s_t, a_t, r_t, s_{t+1})$ in $R$
    Sample a random minibatch of $N$ transitions $(s_i, a_i, r_i, s_{i+1})$ from $R$
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_i}$$

    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  **end for**
**end for**

### Proximal Policy Optimization (PPO)

PPO is an on-policy algorithm that can be used with both discrete and continuous action spaces. PPO strikes a balance between ease of implementation, sample complexity, and ease of tuning, trying to compute an update at each step that minimizes the cost function while ensuring the deviation from the previous policy is not a big jump.

Input: initial policy parameters $\theta_0$, initial value function parameters $\phi_0$

**for** $k = 0, 1, 2, \ldots$ **do**

Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.

Compute rewards-to-go $\hat{R}_t$.

Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function $V_{\phi_k}$.

Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left( \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), \quad g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$$

typically via stochastic gradient ascent with Adam.

Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left( V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.

**end for**

# APPENDIX B

## CODE SNIPPETS

Steering models discussed in Chapters 4 & 5 are implemented in C++ based Unity platform. Snippets of sections of the code is provided below. Same structure of MDP formulations discussed through out the study is used to implement the program.

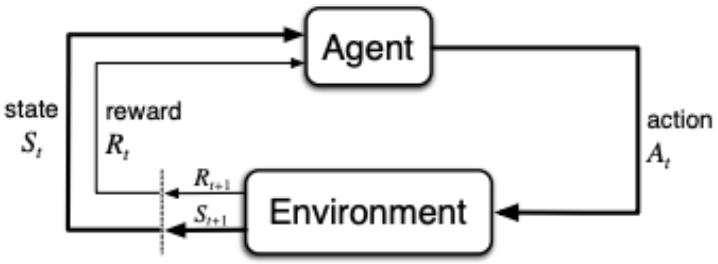


Figure B.1: Block Diagram representation of Markov Decision Process [5]

### Simulation I/O

```
[Header("Measurements")]
public float inclination; //Angle difference from orginal position (vertical)
public float depth; //Distance from surface
public int gamma; //Current rock type
public float KOP; //Depth at which drill will begin changing angle.
public float dogleg; //Change of inclination ind degrees per 100ft
public float targetDistance; //Tracks estimated distance from the oil layer.
public float fromTarget;
public bool inBounds; //Checks to see current position of drill within the ground.
public float inclineA; //Inclination measure taken 100ft prior to the displayed inclination. Used for dogleg.
public bool inOil; // Checks if agent was in the oil zone in the previous step.
public float maxDepthSoFar = 0;
public float fromTargetPrev = 0;
public float fromGoalCoord = 0;
public float fromGoalCoordPrev = 0;

[Header("Drill Control")]
public float turnDegree; //Current Degree rotation of drill, between -6 to 6 degrees
public float drillSpeed; //Speed of drill descending, 0-10ft
public float drillSpeedMax; // Maximum drilling speed
public float drillSpeedMin; // Minimum drilling speed
float startPos; //Initial position of the drill
float rockResist; //Determines amount of resistance rock has.
float rotateDeform; //Adds noise to the rotation of the drill
float rotControl;
float speedControl;
public bool inferenceMode = false;
```

### Agent Actions

```
// DISCRETE actions
// Rotation
// 0: left, 1: right, 2: do nothing
// rotControl = actions.DiscreteActions[0];
// Speed
// speedControl = actions.ContinuousActions[0];

if (inferenceMode)
{
    if (transform.position.y > KOP)
    {
        rotControl = 0f;
        speedControl = 1f;
    }
    else
    {
        // CONTINUOUS actions
        // Rotation
        rotControl = actions.ContinuousActions[0];
        // Speed
        speedControl = actions.ContinuousActions[1];
    }
}

else
{
    rotControl = actions.ContinuousActions[0];
    speedControl = actions.ContinuousActions[1];
}

drillSpeed = drillSpeedMin + 0.5f * (drillSpeedMax - drillSpeedMin) * (speedControl + 1);

// Draw trajectory point
DrawHole();

// Translate
gameObject.transform.Translate(new Vector3(0, -drillSpeed * .010000f * Time.deltaTime * simulationSpeed));
```

### States Observed

```
public override void CollectObservations(VectorSensor sensor)
{
    // Observations
    // sensor.AddObservation(depth / 10000f);
    sensor.AddObservation(transform.position.x / 80f); // x-coord
    sensor.AddObservation(transform.position.y / 80f); // y-coord
    sensor.AddObservation(transform.eulerAngles.z / 360f); // angle
    sensor.AddObservation(dogleg / 90f);
    sensor.AddObservation(fromTarget / 5000f); // becomes negative after crossing oil zone
}
```

### Rewards Received

```
SetReward(0f);

if (Physics.Raycast(transform.position, transform.forward, out hit))
{
    if (hit.collider.gameObject.GetComponent<Layer>() != null)
    {
        gamma = hit.collider.gameObject.GetComponent<Layer>().gamma;
        rockResist = hit.collider.gameObject.GetComponent<Layer>().resist;
        rotateDeform = hit.collider.gameObject.GetComponent<Layer>().rockNoise;
    }

    // Reached oil zone
    if (hit.collider.gameObject.GetComponent<Oil>() != null)
    {
        AddReward(+1f);
        inOil = true;
    }
    // else
    // {
    //     // Agent goes out of oil zone: high negative reward
    //     if (inOil == true)
    //     {
    //         SetReward(-10f);
    //         inOil = false;
    //     }
    // }

}

// Agent moves up/tries to go in a circle: high negative reward
if (inferenceMode)
{
    if (depth < maxDepthSoFar - 2000f)
    {
        EndEpisode();
    }
}

{
    if (depth < maxDepthSoFar - 500f)
    {
        AddReward(-100f);
        EndEpisode();
    }
}

// Small living cost
// SetReward(-0.01f);

// Dense Reward: distance to goal
fromGoalCoord = Mathf.Pow(Mathf.Pow(transform.position.x - 80f, 2f), 0.5f);
AddReward(10f * (fromGoalCoordPrev - fromGoalCoord));
fromGoalCoordPrev = fromGoalCoord;
```

# APPENDIX C

# INDUSTRY SURVEY

With an objective to close the gap between research in academia and industry, a customer discovery project has been taken up by the research group at Texas A&M University. The project was performed as a part of the National Science Foundation (NSF) I-Corps program. A total of 135 customer interviews have been performed to understand the unmet needs and unsolved problems the industry is currently facing. Several insights were drawn from the interviews from drillers at the rig site, drilling engineers, geologists, to executives at major Oil & Gas (O&G) companies. The interviews helped develop a thorough understanding of the drilling ecosystem of both geothermal and O&G industries. The findings can be broadly categorized into 3 main aspects as listed below.

1. Technology gaps in the industry
2. Domains with biggest pain-points
3. Sense of urgency for a solution

We learned that drilling faster, safer, and more predictably is a significant pain point and a pressing need for drilling engineers at numerous companies. Additionally, small-scale drilling operators and service companies with short-term assets are best positioned to be early adopters of new technology. Key findings such as these helped narrow down our focus and prioritize the research efforts towards solving the immediate pressing need in the industry. A business model canvas as shown in Fig. C.1 has been developed after multiple iterations of validations of hypotheses surrounding the product. We learnt that while the industry is moving towards using Rotary Steerable Systems and other advanced control systems, building a truly autonomous system is constrained due to the lack of accurate depth measurements downhole. The workaround in practice is to rely on surface measurements or

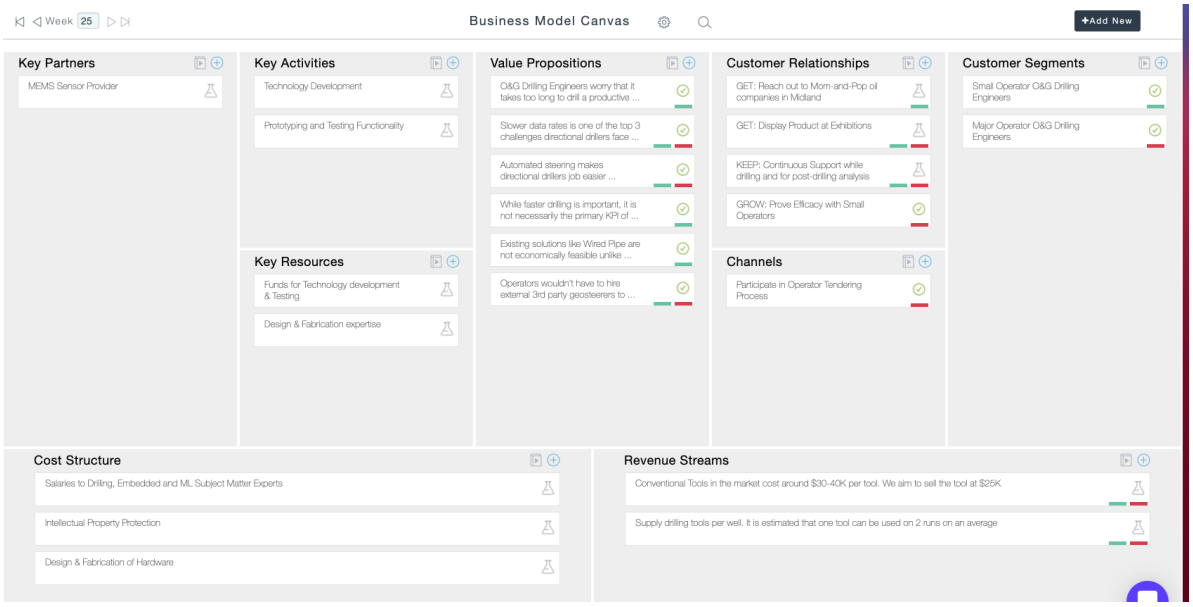


Figure C.1: Business Model Canvas developed from the Customer Discovery Project showing the various hypothesis tested for each of elements in the business model.

estimations – both lead to a significant error due to slow downlinking data rates and harsh downhole conditions. Adding to the complexity are changes in mechanical properties of drill string in rotation, which requires accurate modeling in real-time that is currently far from reality. Knowing an accurate position of the drillbit downhole has several multi-faceted benefits – better steering control of the wellbore thus enhancing production & collision avoidance helping companies mitigate legal and regulatory issues, in addition to cost-savings and improved safety.