



**NTNU – Trondheim**  
Norwegian University of  
Science and Technology

# An Integrated Visualization Technique for Wellbore Placement

**Mats Stivang Ramfjord**

Master of Science in Engineering and ICT

Submission date: June 2014

Supervisor: Børge Arntsen, IPT

Co-supervisor: Ahmed Adnan Aqrawi, Schlumberger

Norwegian University of Science and Technology

Department of Petroleum Engineering and Applied Geophysics



---

# Acknowledgement

---

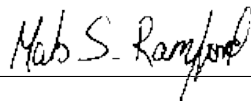
This report is the result of a master's thesis assigned by the Department of Petroleum Engineering and Applied Geophysics at the Norwegian University of Science and Technology in Trondheim, Norway.

I would like to thank my supervisor Professor Børge Arntsen for the support with the project. He has provided me with excellent help and guidance. I would also like to thank Ahmed Adnan Aqrabi and Hallgrim Ludvigsen of Schlumberger for their support and interest, in addition to the initiative of starting this project. They have provided invaluable ideas, development tools and data sets. I would also like to thank Susann Jøraandstad of Schlumberger for providing the computer that the development and testing was done with. In addition to this, I would like to thank Trond Benum of Schlumberger for helping me getting started in the initial phase of the project. I would not have been able to complete this project without the help and support from all these people. Thank you all.

A special thanks goes to my girlfriend, Henriette, for supporting me throughout the project and for all the great years we have had together in Trondheim.

A final word goes to my family. Thank you for all the support you have given me over the years. You have always been there for me and it has always been highly appreciated. I could not have done this without you.

*Trondheim, June 2014*



---

Mats Stivang Ramfjord



---

# Abstract

---

The placement of new wells in a mature field may be very challenging. There is always some uncertainty related to the position of the existing wells, as well as the newly planned well. Other hazards such as faults or salt domes may also be present. Well planners will always try to avoid any collisions and other dangers. In a cluster of wells it can be difficult to find a good path to a target. This report proposes an integrated visualization technique for finding safe and drillable areas in a field in 3D. The solution filters out dangerous and unreachable areas. This can possibly save time during the well planning phase. In theory, any kind of hazard can be used as input.

Existing technologies that originally have been developed for the geophysics domain are used as visualization tools, i.e. seismic cubes and geobodies. The final result, a *drilling volume*, is stored as a seismic cube. It quickly draws attention to areas that can be reached by drilling from a given start position. Since the drilling volume is stored as a seismic cube, it can be used in combination with seismic attributes to display areas with high reservoir quality, that also are safe and reachable. This is a powerful result of the drilling volume. However, there is still much that can be improved. The performance of creating a drilling volume is currently not good enough when used with medium-to-large seismic cubes. Also, the resulting drilling volume can possibly exclude areas that in reality are both safe and reachable. Because of this, the solution presented in this report should not be used without being critical to the result.



---

## Sammen drag

---

Planlegging av nye brønner i et velutviklet felt kan være veldig utfordrende. Det er alltid en viss usikkerhet ved posisjonen til eksisterende brønner, samt posisjonen til en planlagt brønn. Flere farer kan være tilstede, som for eksempel forkastninger eller saltformasjoner. Under planlegging av en ny brønn prøver man alltid å unngå kollisjoner og andre problemer. I en klynge av brønner kan det være vanskelig å finne en god vei til et bore mål. Denne rapporten presenterer en integrert visualiseringsteknikk for å finne områder som det både er trygt og mulig å bore inn i fra en gitt startposisjon. Løsningen filtrerer ut områder som er farlige og ikke kan nås. Dette kan spare tid under brønnplanleggingsfasen. I teorien kan hvilken som helst type fare brukes som input.

Eksisterende teknologi som opprinnelig har blitt utviklet for geofysikkdomenet, altså seismikkuber og geobodies, brukes som visualiseringsverktøy. Sluttresultatet, et *drillingvolum*, lagres i form av en seismikkube. Det trekker kjapt oppmerksomheten til områder som kan nås fra en gitt startposisjon. Drillingvolumet kan kombineres med seismiske attributter for å trekke frem områder som har høy reservoarkvalitet og kan nås. Dette er mulig siden drillingvolumet lagres i form av en seismisk kube. Dette er et bruksområde hvor drillingvolumet har høy nytteverdi. Det må likevel påpekes at det er mye som kan forbedres. Ytelsen bak konstrueringen av drillingvolumet er foreløpig for dårlig når det blir brukt mellomstore/store seismikkuber som input. Det kan også oppstå tilfeller der resultatene ekskluderer områder som egentlig er både trygge og mulige å bore til. På grunn av dette burde ikke løsningen som presenteres i denne rapporten brukes uten å være kritisk til resultatet.





# Contents

<b>Acknowledgement</b>	<b>i</b>
<b>Abstract</b>	<b>iii</b>
<b>Sammendrag</b>	<b>v</b>
<b>Contents</b>	<b>vii</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>xi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Goals . . . . .	2
1.2 Contributions . . . . .	3
1.3 Thesis outline . . . . .	3
<b>2 General Background</b>	<b>5</b>
2.1 Geophysics . . . . .	5
2.1.1 Seismic Cubes . . . . .	5
2.1.2 Geobodies . . . . .	7
2.1.3 Faults . . . . .	10
2.2 Drilling . . . . .	11
2.2.1 Well Planning . . . . .	12
2.2.2 Measurements . . . . .	12
2.2.3 Position Uncertainty . . . . .	14
2.2.4 Collision Avoidance . . . . .	15
<b>3 Theory</b>	<b>17</b>
3.1 Algorithms . . . . .	17
3.1.1 Order of growth . . . . .	17
3.2 Graph theory . . . . .	18
3.2.1 Graphs . . . . .	18
3.2.2 Trees . . . . .	18
3.2.3 Tree traversal . . . . .	19

<b>4</b>	<b>Drilling Volume</b>	<b>21</b>
4.1	Basic description . . . . .	21
4.2	Computation . . . . .	24
4.2.1	Initialization . . . . .	24
4.2.2	Moving through the cube . . . . .	26
4.3	Usage . . . . .	31
4.3.1	Adding hazard types . . . . .	32
4.3.2	Visualized as a seismic cube . . . . .	32
4.3.3	Visualized as a geobody . . . . .	32
4.3.4	Cross-plotting with geobodies . . . . .	32
<b>5</b>	<b>Results</b>	<b>33</b>
5.1	Data . . . . .	33
5.2	Visualization . . . . .	34
5.2.1	Without hazards . . . . .	35
5.2.2	With hazards . . . . .	37
5.3	Performance . . . . .	40
5.3.1	Runtimes . . . . .	40
5.3.2	Hardware . . . . .	41
<b>6</b>	<b>Discussion</b>	<b>43</b>
6.1	Visualization . . . . .	43
6.1.1	Shape and areal extent . . . . .	43
6.1.2	Holes . . . . .	44
6.1.3	Limitations vs. reality . . . . .	45
6.2	Performance . . . . .	45
6.2.1	Infinite loops . . . . .	46
6.2.2	Stop criteria . . . . .	46
6.2.3	Runtime vs. hazards . . . . .	47
6.2.4	Node splitting and new directions . . . . .	47
<b>7</b>	<b>Conclusion</b>	<b>49</b>
	<b>References and Bibliography</b>	<b>51</b>

# List of Figures

2.1	A seismic cube. . . . .	6
2.2	Slices in a seismic cube. . . . .	7
2.3	A geobody of the seismic cube in Figure 2.1. . . . .	8
2.4	Opacity settings for the geobody in Figure 2.3. . . . .	8
2.5	The geobody in Figure 2.3 with custom opacity settings. . . . .	9
2.6	Custom opacity settings for the geobody in Figure 2.5. . . . .	10
2.7	Fault types. . . . .	11
2.8	One vertical well and one deviated well. . . . .	12
2.9	Azimuth, inclination, measured depth, northing, easting and true vertical depth. . . . .	13
2.10	Normal well shapes. . . . .	14
2.11	Visualization of error cones for a vertical well (blue) with a side-track (green). . . . .	15
3.1	A tree. . . . .	18
3.2	A rooted tree. . . . .	19
3.3	Breadth-first-search (BFS). . . . .	19
4.1	A drilling volume computed in a heavily faulted area with two existing wells nearby. . . . .	21
4.2	Time slice of a hazard cube. . . . .	25
4.3	Geobody of a hazard cube. . . . .	26
4.4	Example of azimuth spread with 8 start nodes. . . . .	27
4.5	Saved angle. . . . .	29
4.6	Finding inclination cost. . . . .	30
5.1	The seismic data used as input. . . . .	33
5.2	The wells used for testing. . . . .	34
5.3	A seismic cube with faults. . . . .	34
5.4	Time slice (seen from above) of a drilling volume created using an empty hazard cube. . . . .	35
5.5	Inline section of a drilling volume created with using an empty hazard cube. . . . .	36
5.6	Drilling volume visualized as a geobody. . . . .	36
5.7	Drilling volume visualized as a geobody, showing holes. . . . .	37

*LIST OF FIGURES*

---

5.8	Time slice of drilling volume with hazards. . . . .	38
5.9	Inline section of drilling volume with hazards. . . . .	38
5.10	Drilling volume in between a hazard cube, both visualized as geobodies. . . . .	39
5.11	Drilling volume with a time slice of a hazard cube. . . . .	40
5.12	Plot of measured performance. . . . .	41

# List of Tables

4.1	Example of azimuth spread with 8 start nodes. . . . .	26
4.2	All possible new directions by changing inclination and/or azimuth. . . . .	31



## Introduction

---

In the recent years, several attempts have been made to use 3D graphics in conveying information in between disciplines in the petroleum industry. One of the goals behind this has been to provide a good framework for better teamwork and cooperation across disciplines. The drilling engineer does not have knowledge about everything that is important for the geophysicists, and vice versa. It has been focused on including drilling requirements in all stages of the well design. This has been done to streamline the workflow between the geoscience requirements and the drilling limitations (Cayeux et al., 2001).

The exact position of a wellbore trajectory is not known and it is not a trivial task to find it. Normal practice is to derive it from surveying measurements of measured depth, inclination and azimuth. The measurements contain some errors. These errors will contribute to the position uncertainty (Bang and Torkildsen, 2011).

Drilling engineers often use other software tools than the geophysicists and reservoir engineers. This puts some limitations to what can be achieved when it comes to cooperation. The data from all the disciplines should be used in the best practical way to get the most accurate results, such that better decisions can be made. Drilling engineers may find the position uncertainty for a well. The geophysicist and the reservoir engineer may predict the properties of the reservoir. This information can be used in many ways, but the software tools have some limitations. Because of this, the information from the drilling engineer and the geophysicist and reservoir engineer may not be presented in the best possible way.

An integrated solution is proposed where the position uncertainty is displayed with the tools used by the geophysicists. The solution takes an error model for a well and finds the volume that the well may be in, for a given uncertainty model. For a set of wells in the same area, the solution finds the uncertainty volume for all the wells and creates a data set that can be visualized in 2D and 3D. This is done by using well data in combination with geophysical tools, where the visualization is handled by the geophysical

tools. The uncertainty volume is called a *hazard cube*.

The solution can, in theory, create a hazard cube for any type of obstacle. Each obstacle type can have its own uncertainty model. A set of hazard cubes can be merged together. A hazard cube may resemble a labyrinth that new wells have to pass through. This is in fact what is done in this project. A drilling *node* is planted in a hazard cube. The node tries to visit all reachable parts of the labyrinth. Whether a part is reachable or not depends on the shape of the hazards and how quickly the node is allowed to turn. The movement of the node is controlled by an input DLS (dogleg severity).

The final result is called a *drilling volume*. It shows which areas that are possible to drill into with a given set of hazards, a start position and a maximum allowed DLS. It is stored as a seismic cube. This means that it can, of course, be visualized as a seismic cube, but also as a geobody. This can be very useful.

For the drilling engineer it is very important to know which areas to avoid. The geophysicist and reservoir engineer focus more on predicting which areas have high hydrocarbon potential. The drilling volume makes it possible to filter out the areas that are drillable *and* have high hydrocarbon potential. The opposite can also be done. This example shows one of many ways the drilling volume can be used to present the existing information in a better way, such that better decisions can be made.

Skinner (2011) mentions three main reasons why the position of the wellbore is important:

- Intersection with geological targets
- Collision avoidance management
- Future well and reservoir development planning

The first and second point is very important for the drilling volume, as will be shown later.

## 1.1 Goals

The main goal of this thesis is to provide a safe and reachable 3D drilling volume, given a set of obstacles one would like to avoid. The obstacles could be e.g. existing wells, faults or salt domes. To achieve this, the following problems must be addressed:

- Create a quick overview of hazards in a field.
- Visualize the position uncertainty for every input obstacle. In other words: create hazard cubes for every input obstacle.



- Obtaining a visualization technique that can be used for any input hazards.
- Simulating realistic drilling movement in a 3D grid (seismic cube).
- Simplify the early stages of well planning. (Reducing the design cycle time. (Cayeux et al., 2001))

## 1.2 Contributions

These are the main contributions of the work:

- A powerful visualization technique for drillable areas in a field with a cluster of existing wells and other hazards.
- Combining existing technology developed for the geophysics domain with tasks related to the drilling domain. This opens up for new kinds of workflows.
- A filter that filters out unreachable areas. This can save a lot of time in the well planning phase.
- A visualization tool that can be combined with geobodies to show reachable areas with e.g. high reservoir quality.

## 1.3 Thesis outline

The report is structured in the following way:

**Chapter 2:** Presents relevant background material from the petroleum industry, in particular drilling and some technology for geophysics. Topics such as wellbore position uncertainty and collision avoidance are covered, in addition to seismic cubes and geobodies.

**Chapter 3:** Covers some basic theory related to algorithms and graphs.

**Chapter 4:** Describes the drilling volume, the logic behind, how it is computed and examples of how it may be used.

**Chapter 5:** Presents the results. The focus is mainly on the visualization, but performance results are also presented.

**Chapter 6:** Discusses the results. The focus is also here mainly on the visualization.

**Chapter 7:** Concludes the report.



# General Background

---

The petroleum industry is often divided into three main sections: upstream, midstream and downstream. The upstream sector can be further divided into two subsections: exploration and production. Exploration includes the searching of potential oil and gas fields and drilling of exploration wells. This is done at onshore and offshore locations. Production covers drilling and operating the wells that bring hydrocarbons to the surface. The other two sections, midstream and downstream, covers everything else such as transportation, storage, sale, refining, processing, marketing and distribution.

Software has been an increasingly important part of the petroleum industry during the last few decades. It is used throughout all of the industry, ranging from upstream to downstream. Computers are excellent at numerical calculations, giving them strong benefits in simulations and visualizations compared to human calculation power. This has been heavily utilized by the industry for many years.

## 2.1 Geophysics

Some important geophysical terms related to the computation and visualization of the drilling volume factor will be presented here. The computation and visualization of the drilling volume relies on technology that was originally developed for geophysicists. It uses seismic cubes and their visualization techniques to visualize the data. After the drilling volume has been computed, it can be visualized in an even better way using geobodies.

### 2.1.1 Seismic Cubes

Seismic cubes are usually used for displaying seismic data, as seen in Figure 2.1. One cube consists of many cells. Each cell has a given three-dimensional index  $(i, j, k)$  that describes the relative position of the cell. Each cell also

has a value assigned to it. E.g., the cube in Figure 2.1 consists of about 6.4 million cells, each with a given index.

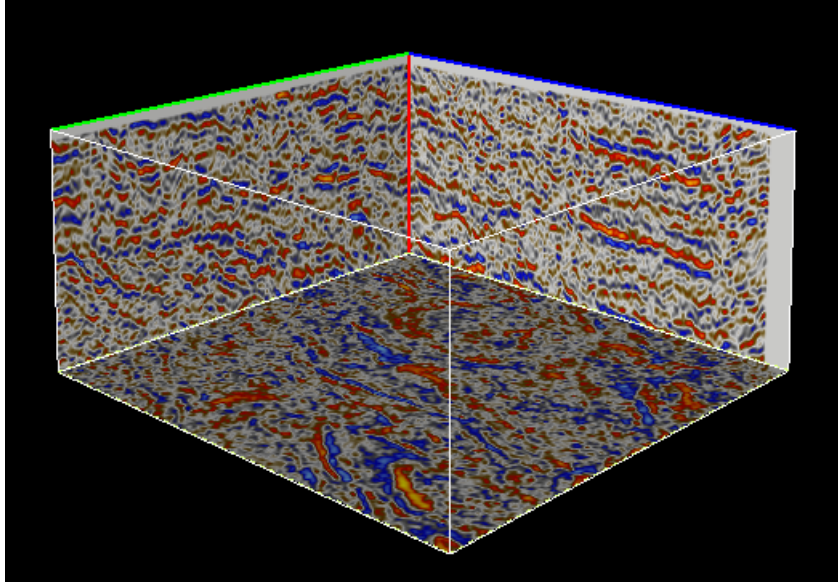


Figure 2.1: A seismic cube.

### Slices

Any part of a seismic cube can be displayed by interactively changing the position of a slice. Figure 2.2 shows the seismic cube in Figure 2.1 where the position of the slices have been changed, such that other parts of the cube is displayed. Vertical sections can be chosen as well as horizontal sections. It is also possible to show any kind of section by using so-called random sections.

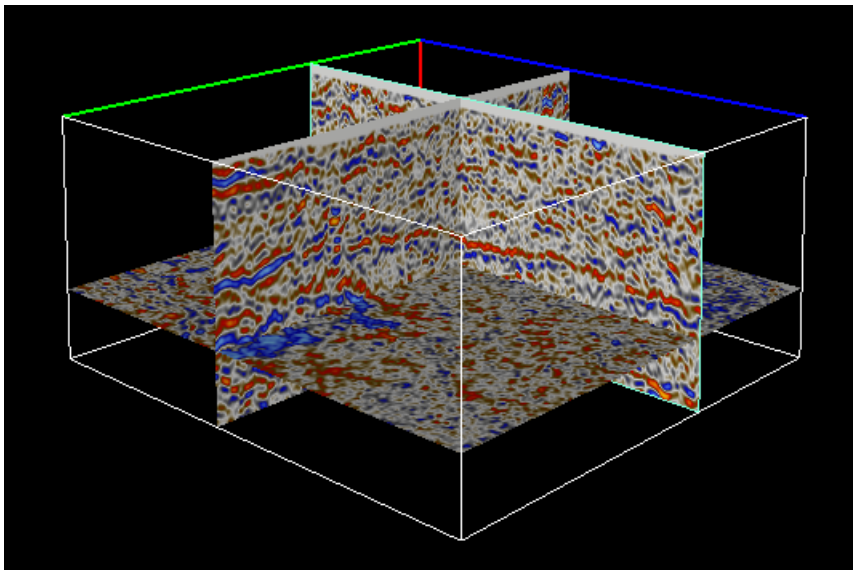


Figure 2.2: Slices in a seismic cube.

### 2.1.2 Geobodies

A geobody is a powerful way to visualize seismic data. A geobody can be created from a seismic cube. It allows the user to display the full seismic cube as a complete volume, instead of only showing certain sections at a time. It is also possible to specify some parts of the data set that should not be visible. This way, one can e.g. hide all values below a given threshold by setting the opacity to zero. Another possibility is to cross-plot two or more cubes in the same geobody, e.g. showing two or more different attributes at the same time. This can be heavily utilized when visualizing a drilling volume, as will be described later.

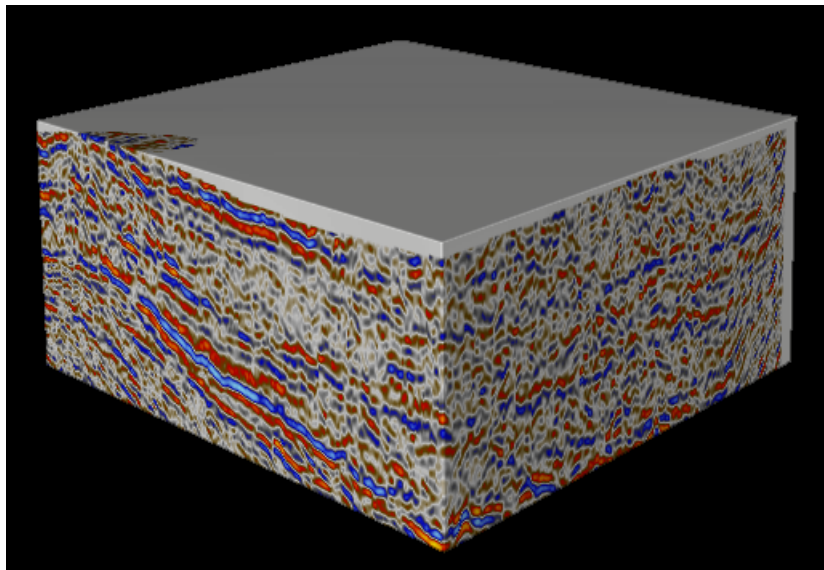


Figure 2.3: A geobody of the seismic cube in Figure 2.1.

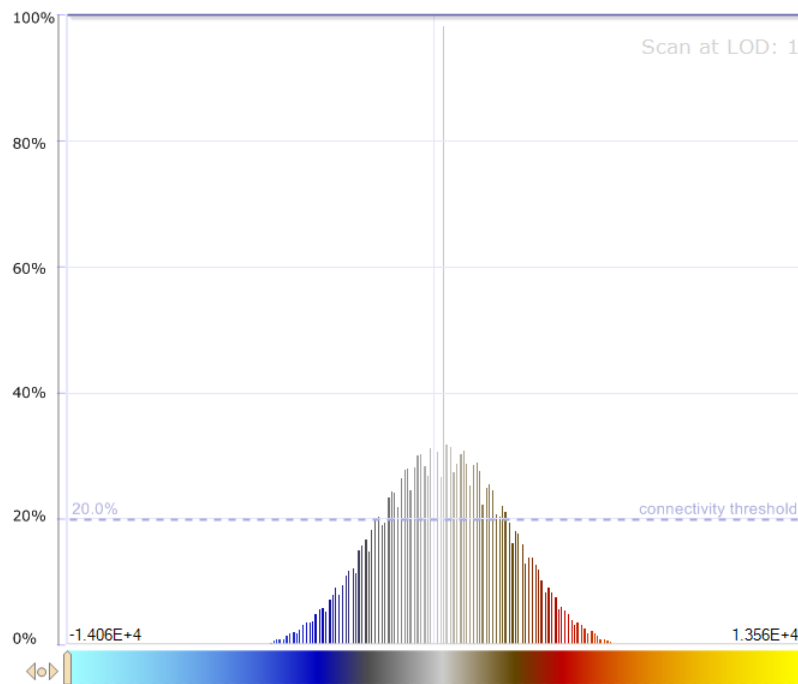


Figure 2.4: Opacity settings for the geobody in Figure 2.3, with 100% opacity for all values.

Figure 2.3 shows a geobody of the seismic cube in Figure 2.1. Here the complete volume is displayed with 100% opacity. The opacity settings for

## 2.1. GEOPHYSICS

---

this geobody is shown in Figure 2.4. By changing the opacity for certain values, it is possible to get a visualization like the one in Figure 2.5. The opacity settings for this geobody is shown in Figure 2.6. This example may not be the best usage of the opacity tool, but it demonstrates how it is possible to display interior parts of a geobody. This is a very useful feature for the drilling volume.

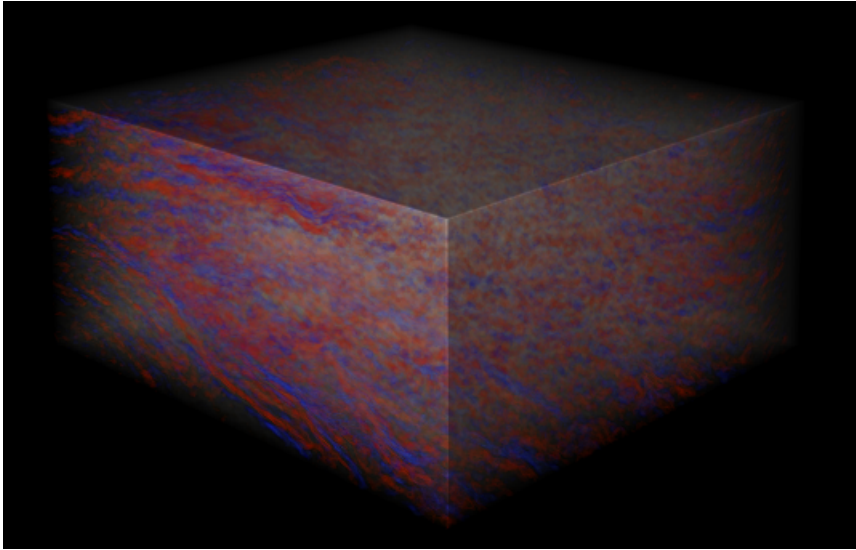


Figure 2.5: The geobody in Figure 2.3 with custom opacity settings.

## 2.1. GEOPHYSICS

---

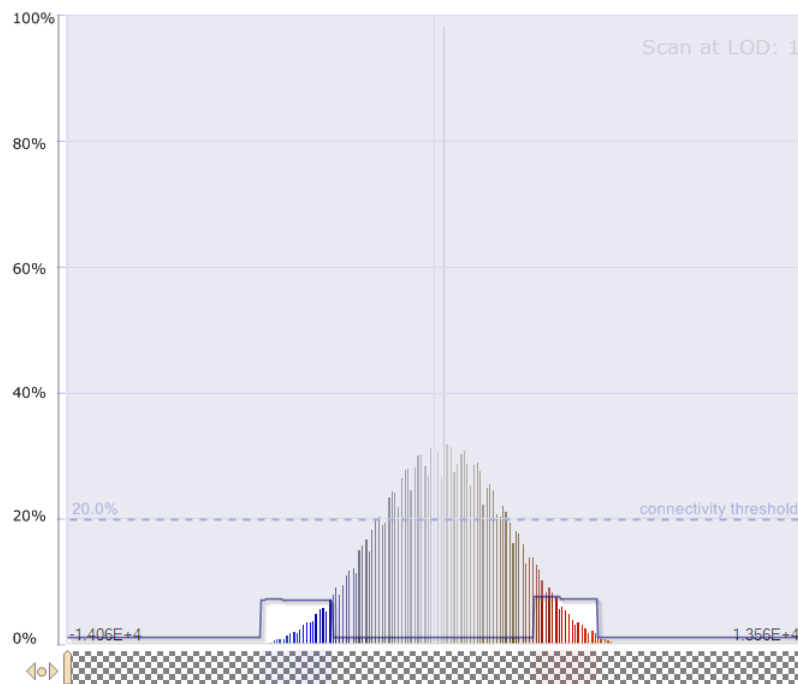


Figure 2.6: Custom opacity settings for the geobody in Figure 2.5.

### 2.1.3 Faults

A fault is described by Jahn et al. (2008, p. 81) as a "plane of failure" in a rock caused by stress. Faults are often observed on seismic sections, shaped as curves or lines. They can cover many hundreds of kilometres. Figure 2.7 shows three types of faults and how they are formed over time.

When planning a new well, it is often desirable to avoid drilling into faults, as they can cause severe problems. Because of this, faults have been used as input to create hazard cubes (see section 4.2.1) in this project.



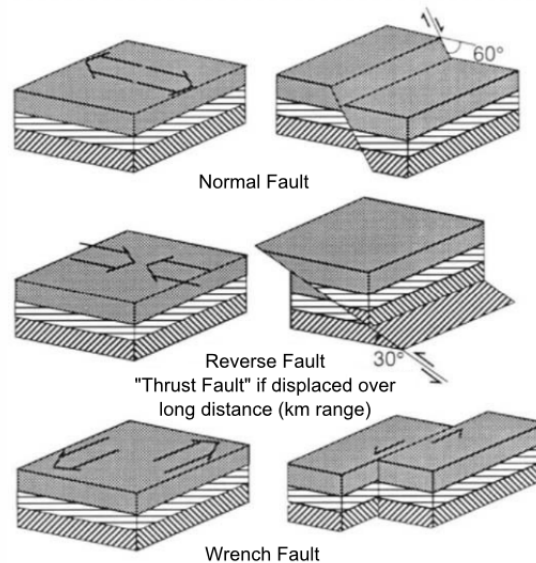


Figure 2.7: Fault types. Image copied from Jahn et al. (2008, p. 82) and modified.

## 2.2 Drilling

Drilling engineering is a big discipline. This report will (obviously) not cover everything related to drilling - only the most relevant topics related to the drilling volume will be presented.

The petroleum industry uses wells to bring hydrocarbons from the subsurface to the surface, i.e. produce oil or gas. There are a few types of wells in the petroleum industry. Two of them are exploration- and production wells. Exploration wells are usually drilled to confirm clear indications of hydrocarbons in the subsurface. The information on where the geological target is may e.g. be given to the drilling engineer from a geophysicist. The exploration wells are usually vertical, as this has a lower cost than directional wells.

Production wells bring hydrocarbons from a reservoir to the surface or inject gas or water into the reservoir. The goal of the injection is often to make it easier to transport the hydrocarbons to the surface. The properties of the hydrocarbons can be altered by the injection fluid and making it flow more easily. Injection can also be used to maintain the pressure in a reservoir. The well paths may be very long and even exceed 10 km, such as the Z-44 well that was drilled at Chayvo with a measured depth of 12 376 m (Sanford et al., 2013).

## 2.2. DRILLING

---

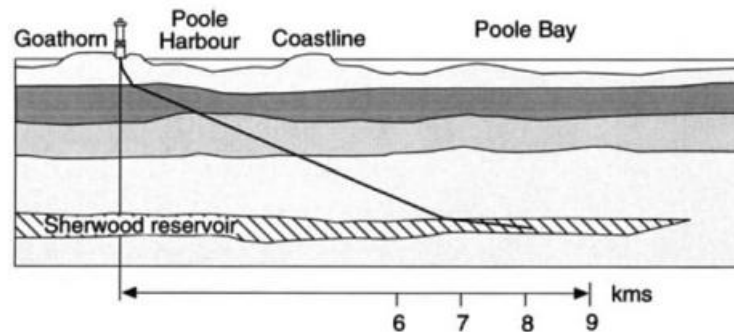


Figure 2.8: One vertical well and one deviated well. Image copied from Jahn et al. (2008, p. 46).

A well is usually drilled vertically from the surface. It may be vertical all the way, or deviate at a predefined depth. This is what happens in directional drilling. The path of the well can be formed in many ways, limited by the drilling equipment that is being used. E.g., the dogleg severity (DLS, how "quickly" a well can turn) is limited by the drilling equipment. Some normal trajectories are shown in Figure 2.10.

### 2.2.1 Well Planning

To drill a new well, a major investment is required. Drilling a new well has to be profitable. An unwanted surprise can be very expensive for the drilling company, so the well planning is crucial to minimize the costs, as well as risks. Jahn et al. (2008) mentions the following four target objectives for drilling a new well:

1. Gather information
2. Produce hydrocarbons
3. Inject gas or water
4. Relieve a blowout

The first and last items are usually done in the exploration phase and the production phase, while the others are typically done in the production phase only. There may occur problems during any of these objectives.

When planning new wells, the existing wells are often referred to as *offset wells*, while the new well is simply referred to as the *planned well*.

### 2.2.2 Measurements

There are three main terms used to describe the measured position of a well: measured depth (MD), azimuth (A) and inclination (I). Azimuth and

## 2.2. DRILLING

---

inclination are calculated from other measurements, while MD is "raw". In the end, these three measurements are used to describe the position of a well. These three measurements can also be used to calculate northing (N), easting (E) and true vertical depth (TVD).

### Measured depth

MWD tools (measurements while drilling) are often used to measure depth. MD is measured at the surface and describes how long the well is. This is not the same as TVD, which can be different if the well is drilled directionally.

### Azimuth

Azimuth describes the compass direction and is usually specified in degrees with respect to the geographic or magnetic north pole. It is often measured using magnetic surveying tools, such as magnetometers.

### Inclination

Inclination specifies how much the wellbore deviates from vertical direction, given in angles. Inclination is usually measured with accelerometers and/or gyroscopes.

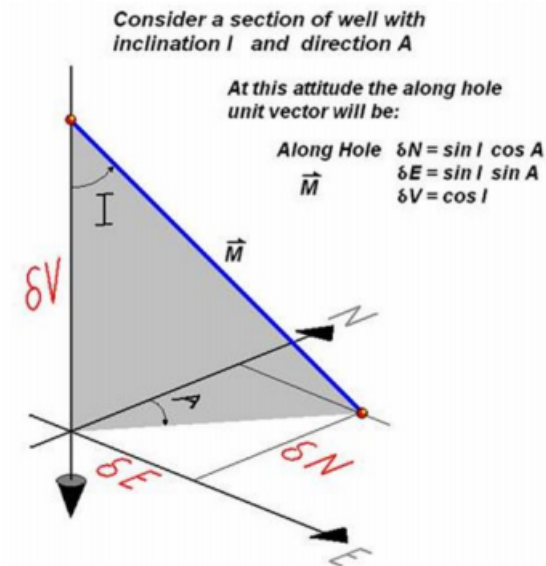


Figure 2.9: Azimuth (A), inclination (I), measured depth (MD), northing (N), easting (E) and true vertical depth (TVD). Image copied from Jamieson (2012, p. 68).

### Dogleg severity

A severe dogleg can be characterized as a *sudden change in angle or direction*. This can cause a pipe to get stuck. To avoid this, the *dogleg severity* (DLS) should be controlled while drilling. DLS is usually given in degrees per 100 ft. or degrees per 30 m. There are several ways to calculate DLS. Most methods use measured depth, inclination and azimuth.

### 2.2.3 Position Uncertainty

Jamieson (2012) mentions that an error in azimuth tends to be the dominant error in a 3D ellipse of uncertainty. An effect of this is that position uncertainty is smaller in vertical wells than in deviated wells. Figure 2.8 shows a vertical well and a deviated well. The deviated well is likely to have a larger error ellipse than the vertical well.

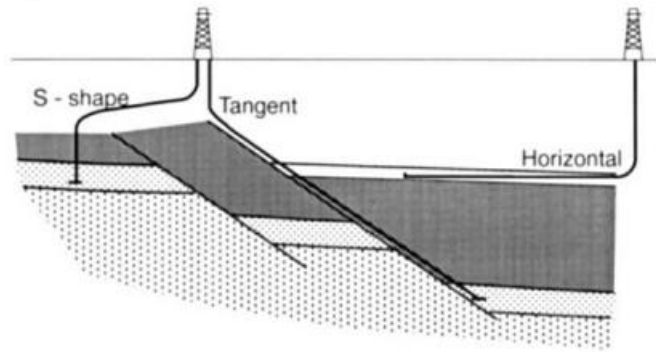


Figure 2.10: Normal well shapes: s-shape, tangent and horizontal. Image copied from Jahn et al. (2008, p. 51).

### Visualization of Positional Uncertainty

In 3D software, position uncertainty is often visualized with a cone along the path of a well. If the cones of a proposed well and an offset well intersect, there may be a collision risk. Figure 2.11 shows a well with an error cone.

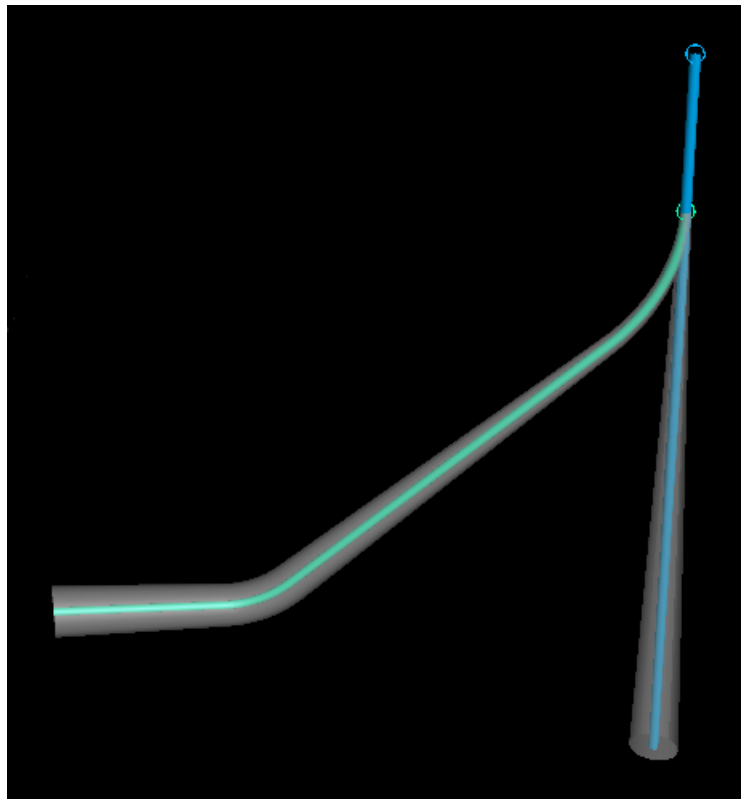


Figure 2.11: Visualization of error cones for a vertical well (blue) with a side-track (green).

### 2.2.4 Collision Avoidance

While planning a well, it is important to identify possible sources of danger (Gluyas and Swarbrick, 2004, p. 278). The consequences of a collision can be huge, both for the humans involved and for the environment. If a collision occurs, there might be a kick, resulting in huge amounts of hydrocarbons flowing uncontrollably to the surface. Oil has a lower density than water, so it floats on top of water. On offshore locations, the oil may spread to large areas, resulting in major environmental damage.

In a scenario like this, the companies involved would have to cover large expenses. It could also heavily damage the reputation of the companies.

In 2010, there was a major oil spill in the Gulf of Mexico (Garg and Gokavarapu, 2012). The effects of the oil spill were huge. The oil spill had spread over 580 square miles in three days, 11 people died and 20 were injured, birds were drowned, breeding was interrupted and the operator company lost more than 60 billions USD.



---

# Theory

---

This chapter presents some basic theory related to algorithms and graphs.

## 3.1 Algorithms

There is no generally accepted *formal* definition of an algorithm. However, Cormen et al. (2009, p. 5) use the following *informal* definition:

*An **algorithm** is any well-defined computational procedure that takes some value, or set of values, as **input** and produces some value, or set of values, as **output**.*

The time it takes for an algorithm to finish is often called the *runtime*, or *running time*.

### 3.1.1 Order of growth

The order of growth of the runtime of an algorithm describes how fast the runtime increases as the problem size increases (Cormen et al., 2009, p. 28). The problem size is often noted as  $n$ . For a function  $f(n)$  with a problem size  $n$ , the big-O notation is denoted as  $O(f(n))$ . Big-O notation is used to describe an upper bound on the runtime of a function or algorithm, i.e. the worst-case runtime. Rosen (2012, p. 205) defines big-O notation as:

Let  $f$  and  $g$  be functions from the set of integers or the set of real numbers to the set of real numbers. We say that  $f(x)$  is  $O(g(x))$  if there are constraints  $C$  and  $k$  such that

$$|f(x)| \leq C|g(x)| \quad (3.1)$$

whenever  $x > k$ . (This is read as "f(x) is big-oh of g(x)".)

As an example: for a function  $f$ , defined as  $f(x) = x^2 + 3x + 1$ , we say that  $f$  is  $O(x^2)$ .

## 3.2 Graph theory

This section covers basic theory of *graphs* and *trees*.

### 3.2.1 Graphs

Graphs are often used to model problems. Rosen (2012, p. 641) defines a *graph* as:

*A graph  $G = (V, E)$  consists of  $V$ , a nonempty set of **vertices** (or **vertices**) and  $E$ , a set of **edges**. Each edge has either one or two vertices associated with it, called its **endpoints**. An edge is said to **connect** its endpoints.*

A vertex may have more than two edges, i.e. be connected to more than two other vertices. If such a vertex exists in a graph, the graph is called a *multigraph*. Furthermore, a *directed graph* is defined as (Rosen, 2012, p. 643):

*A **directed graph**  $(V, E)$  consists of a nonempty set of vertices  $V$  and a set of **directed edges**  $E$ . Each directed edge is associated with an ordered pair of vertices. The directed edge associated with the ordered pair  $(u, v)$  is said to **start** at  $u$  and **end** at  $v$ .*

A graph can be a multigraph and directed at the same time, which makes it a directed multigraph.

### 3.2.2 Trees

A tree is defined by Rosen (2012, p. 746) as:

A *tree* is a connected undirected graph with no simple circuits.

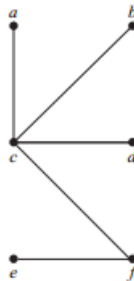


Figure 3.1: A tree with 6 vertices and 5 edges. Image copied from (Rosen, 2012, p. 746).



### 3.2. GRAPH THEORY

---

Figure 3.1 shows a tree. A tree cannot contain multiple edges or loops. A tree can also be *rooted*:

A **rooted tree** is a tree in which one vertex has been designated as the root and every edge is directed away from the root.

Figure 3.2 shows a rooted tree.

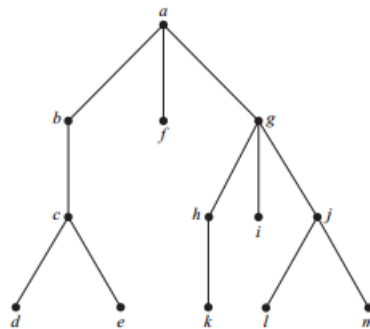


Figure 3.2: A rooted tree. Vertex  $a$  is the root. Image copied from (Rosen, 2012, p. 748) and modified.

#### 3.2.3 Tree traversal

A tree can be traversed in several ways. One normal method is to do a breadth-first-search (BFS). Only this method will be presented here.

In simple terms, when using BFS on a tree, all vertices in one level of the tree are evaluated before continuing to vertices in the next level. An example is shown in Figure 3.3 where BFS is used on a graph.

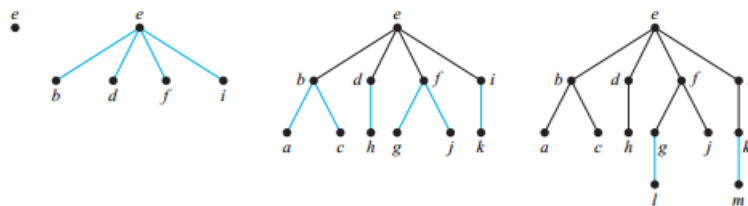


Figure 3.3: Breadth-first-search of a graph. First, the vertex  $e$  is evaluated, then all vertices in the next level are evaluated ( $b$ ,  $d$ ,  $f$  and  $i$ ). This continues until all vertices have been traversed. Image copied from Rosen (2012, p. 790).



# Drilling Volume

---

This chapter presents what a drilling volume is, the preprocessing of a drilling volume, how it is computed, how it is visualized and some examples on how it may be used.

### 4.1 Basic description

A drilling volume is a 3D volume, stored as a seismic cube, that shows where it is possible to drill. A *start position*, *hazards* and *DLS limitations* are required as input. The hazards and DLS limitations control the shape of the drilling volume.

The drilling volume is not meant to provide a single well path nor a small selection of well paths. Instead, it provides the complete volume that a new well can go through.

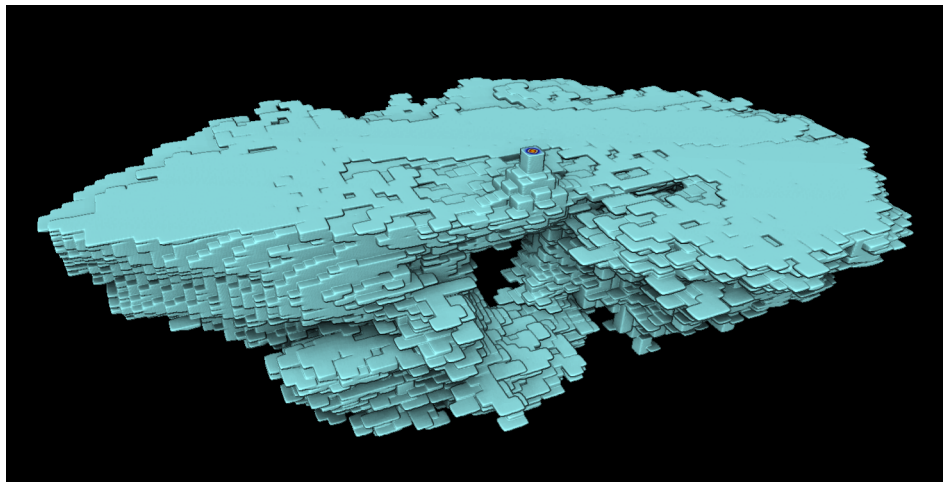


Figure 4.1: A drilling volume computed in a heavily faulted area with two existing wells nearby.

The basic idea behind the creation of a drilling volume is to plant a *node* in a seismic cube at the position where a new well should start (at the *start position*). The node will then start working its way through the seismic cube while trying to simulate the movement of real drilling.

The seismic cube can be considered as a low-resolution 3D grid. It is not possible to create smooth (continuous) arcs in it. However, while moving between indexes in the seismic cube, the computation tries to move in circular arcs that are as smooth as possible. It uses the input DLS to control the curvature of the arcs. This also means that the input DLS controls when a node can *turn*, i.e. change index in a perpendicular direction to its current (grid) direction.

A simple principle is used to decide when a node can turn. While a node is moving without changing its direction, it accumulates an angle (a *saved angle*). This angle is used to decide if the node can turn or not. To check if the *saved angle* is large enough, it is compared to an *angle cost*. When the saved angle is larger than the angle cost, the node can turn. The terms *saved angle* and *angle cost* are described in further detail in section 4.2.2.

Whenever a node does a turn, it changes direction, i.e. inclination and/or azimuth, and splits into more nodes. The new nodes get new directions that they move along. While moving, they accumulate an angle each until they can do a turn. This continues until all reachable indexes are marked, or until the computation is terminated in other ways.

The principles that are used are very simple, but have still proven to be difficult to convert to efficient code. A drilling volume can be very useful, but it must be noted that it is far from a perfect solution right now. The principle works quite well. However, the computation currently suffers too much from problems related to performance.

The main algorithm behind the creation of a drilling volume is shown in Algorithm 4.1. It operates with a queue that nodes are added to once they are created. When the computation is finished with one node, it pops a new node from the queue and starts evaluating it. The queue works as a FIFO-queue (first in-first out). The data structure of all evaluated nodes is in-memory during the computation, and resembles a rooted tree. This causes the algorithm to traverse the solution space similar to a BFS (breadth-first search).

#### 4.1. BASIC DESCRIPTION

---

---

**Algorithm 4.1:** Pseudocode of the drilling volume algorithm.

---

**Input:** hazard cube, start nodes  
**Output:** drilling volume

```
1: add start nodes to queue;
2: while queue is not empty do
3:   node = first node in queue;
   // See node splitting in section 4.2.2
4:   if direction of node is vertical then
5:     | find and add 2 new nodes to queue; // See Alg. 4.2
6:   else if direction of node is diagonal then
7:     | find and add 4 new nodes to queue; // See Alg. 4.2
8:   else if direction of node is horizontal then
9:     | find and add 3 new nodes to queue; // See Alg. 4.2
10:  end
11:  remove node from queue;
12: end
13: return drilling volume;
```

---

Algorithm 4.2 describes how new nodes are found. A run of this algorithm can be seen in Figure 4.6, where inclination cost is used.

---

**Algorithm 4.2:** Pseudocode of finding new nodes.

---

**Input:** node  
**Output:** a new node

```
1: saved angle = 0;
2: angle cost = infinity;
3: while saved angle < angle cost do
4:   move node one index in current grid direction;
5:   if index is outside hazard cube or hazard exists at index then
6:     // Terminate current route. No node is returned.
7:     return null;
8:   else
9:     set value of drilling volume at current index to 1;
10:  end
11:  // Saved angle always increases.
12:  saved angle = DLS · traveled distance;
13:  // Position of test index depends on direction of
14:  // the node and whether inclination and/or azimuth
15:  // is altered.
16:  test index = get new test index;
17:  // Angle cost always decreases.
18:  angle cost = change in angle needed to reach new test index;
19: end
20: create new node;
21: set inclination and azimuth for new node;
22: return new node;
```

---

## 4.2 Computation

Before the drilling volume can be created, the input hazards must be combined into one single hazard cube. Examples of hazards can be existing wells, faults or salt domes.

### 4.2.1 Initialization

In the initialization, the hazard cube and a set of start nodes are created before the creation of the drilling volume starts.

### Hazard cube

A hazard cube is a seismic cube where the values indicate whether there is a hazard at an index or not. The values are binary; a value of 1 means there is a hazard at an index, while 0 means no hazard.

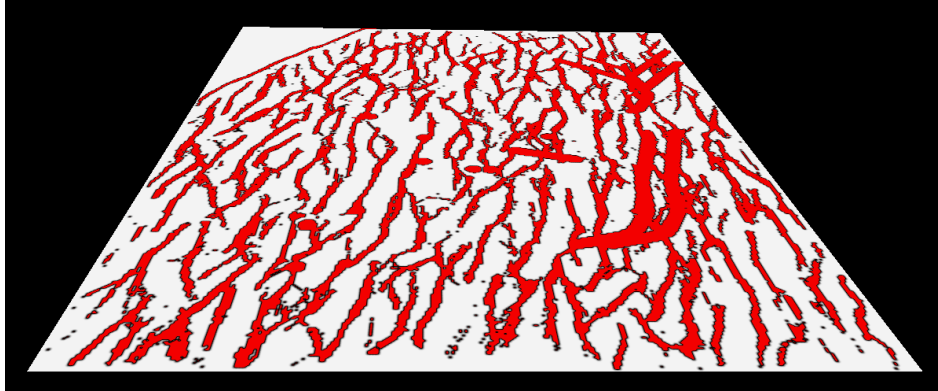


Figure 4.2: Time slice of a hazard cube with some wells and a large amount of faults. Red indicates a hazard, while white indicates "no hazard".

One hazard cube is created for each type of hazard. All the cubes are finally merged into one final hazard cube that is used in the computation of the drilling volume. Figure 4.2 shows a time slice of a hazard cube with some wells and a large amount of faults. The hazards are marked with red, while white indicates "no hazard". Figure 4.3 shows the same hazard cube visualized as a geobody.

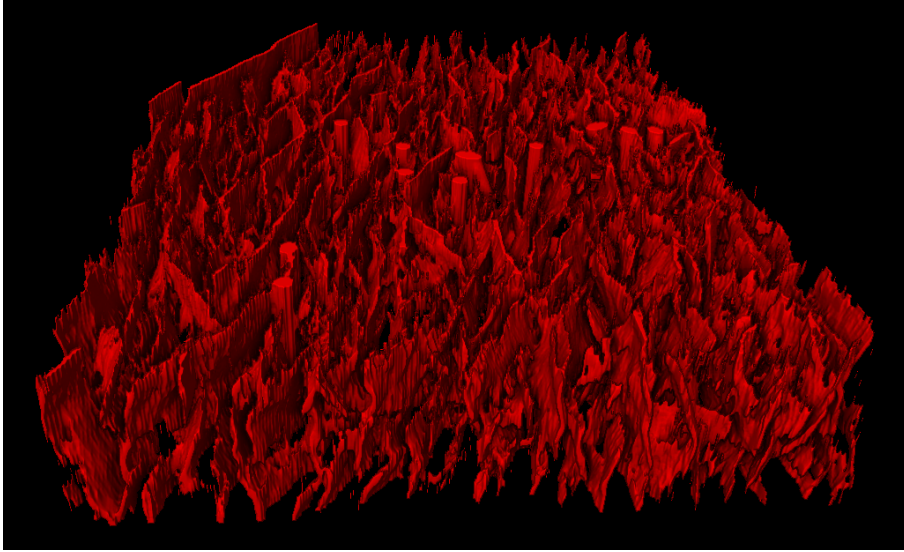


Figure 4.3: Geobody of a hazard cube with some wells and a large amount of faults.

### Start nodes

A set of start nodes are created during the initialization where the azimuth of the start nodes are evenly spread out between 0–360 degrees. This is done to make sure that all azimuth directions are covered in the early stages of the computation. Table 4.1 shows an example with 8 start nodes. The same example is shown in Figure 4.4.

Node	Azimuth (degrees)
1	0
2	45
3	90
4	135
5	180
6	225
7	270
8	315

Table 4.1: Example of azimuth spread with 8 start nodes.

### 4.2.2 Moving through the cube

In the beginning, a start node is moving completely vertical with an inclination of 0. It moves from index to index. When the node has traveled *far enough* (described in section 4.2.2), it will split into more nodes.



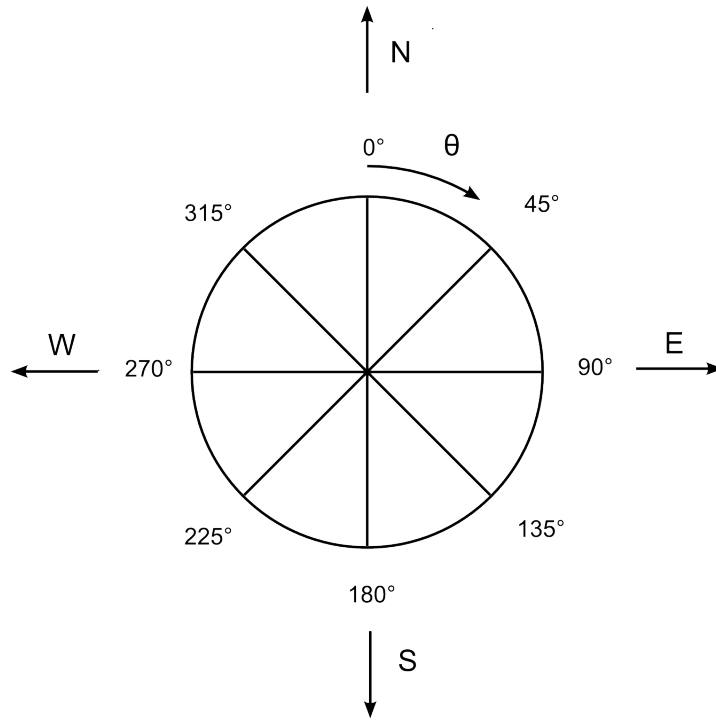


Figure 4.4: Example of azimuth spread with 8 start nodes.

### Node splitting and new directions

The number of new nodes a node splits into depends on the direction of the node:

**Vertical:** the direction of a node is called vertical when its inclination is between 0–30 degrees. When moving vertically, the node will split into **two** new nodes:

1. One node will *increase inclination*.
2. One node will *decrease inclination*.

**Diagonal:** a node is moving diagonally when the inclination is between 30–60 degrees. When the direction is diagonal, it splits into **four** new nodes:

1. One node will *increase inclination*.
2. One node will *decrease inclination*.
3. One node will *increase azimuth*.
4. One node will *decrease azimuth*.

**Horizontal:** finally, when the inclination of a node is between 60–90 degrees, it is said to be moving horizontally. This time it will split into **three** new nodes:

1. One node will *decrease inclination*.
2. One node will *increase azimuth*.
3. One node will *decrease azimuth*.

This set of simplifying rules for node splitting is used to reduce the total number of nodes, thus improving the performance. However, the rules also make sure that a node may never move upwards. This avoids complex infinite loops (discussed in section 6.2.1). The rules also provide all the necessary new directions that are needed to move around in the input seismic cube in a fairly realistic way.

### Saved angle and angle cost

A node has traveled *far enough* to split into more nodes when its *saved angle* is higher than the *angle cost*. The two terms are presented below.

**Saved angle:** as a node moves in one direction, it accumulates an angle — a *saved angle*. The accumulation rate of the saved angle is determined by the input DLS, calculated as:

$$\text{saved angle} = \text{traveled distance} \cdot \text{DLS} \quad (4.1)$$

Figure 4.5 shows how the saved angle ( $\alpha$ ) increases as the traveled distance ( $m$ ) increases during an iteration.

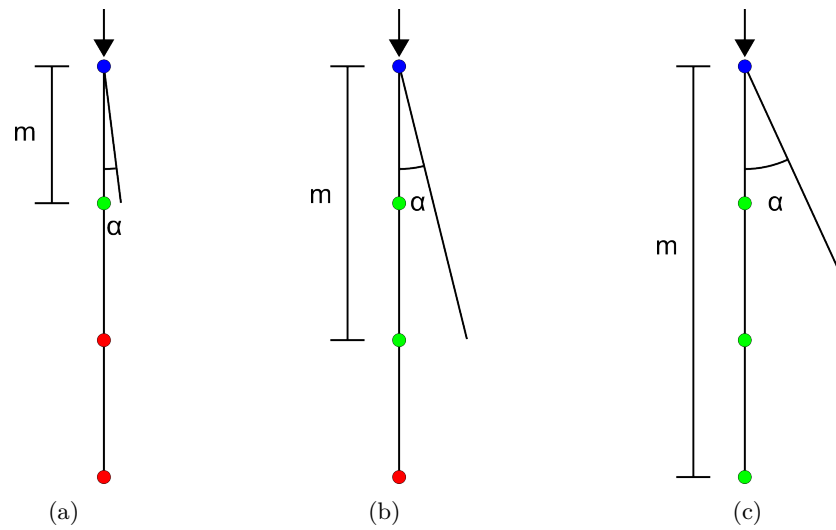


Figure 4.5: Saved angle ( $\alpha$ ) given a traveled distance ( $m$ ). Blue shows where the current node was created (the same position as the parent node). Red indicates *not visited*, while green indicates *visited* or *possible to visit*. The arrow shows the direction of the current node. Note that the saved angle increases as the traveled distance increases.

**Angle cost:** either inclination cost or azimuth cost. The angle cost describes, in terms of an angle, how much the inclination or azimuth of the current node must change in order to move to a given test index. Figure 4.6 shows how inclination cost is found. The same principle is used to find azimuth cost.

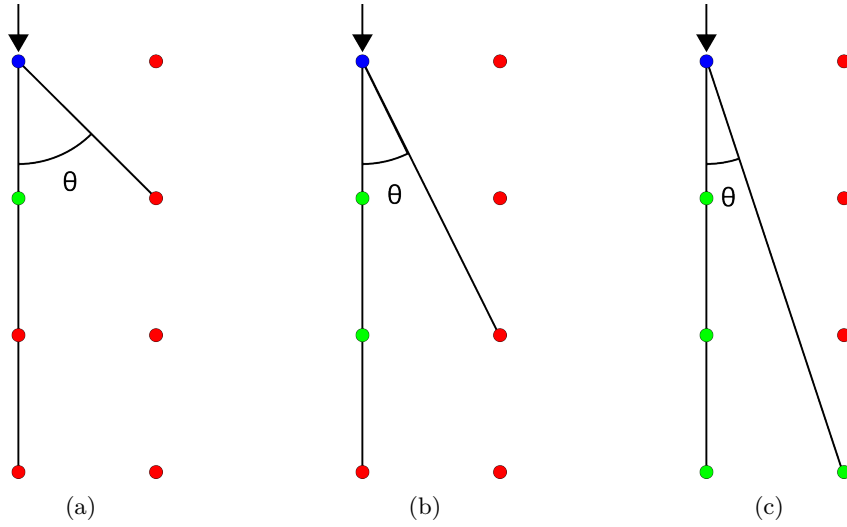


Figure 4.6: Inclination cost ( $\theta$ ) is given by the inclination change needed to move to another (deviated) index. The parent node is blue. Red indicates *not visited*, while green indicates *visited*. The arrow shows the direction (inclination) of the current node. In (a), the current node has visited one index below the parent node, and it was marked. The test index (in red) was not possible to visit, since the inclination cost was too high. In (b), the node has continued along the same direction and visited another index. This one was also marked with green. The inclination cost to the new test index was still too high. In (c), the current node has visited a third index and marked it green. This time, the saved angle was larger than the inclination cost, so the node could jump to the test index (the test index was successful). After this, the current node was split into more nodes and the computation continued.

When the saved angle becomes higher than the angle cost, new nodes are created at the successful test index. The new nodes have other directions than the previous node, i.e. different inclination and/or azimuth values. The inclination value of the new node is given by the direction from the previous node to the new node (in Figure 4.6(c), the direction from the blue node to the bottom right green node). The same applies for azimuth.

Now the cycle is repeated; the new nodes will move along their new directions until they can split into more nodes, and so on. This continues until the whole seismic cube has been traversed, or until the computation is terminated in other ways (see *simplifications* below).

Whenever a node visits an index, the index will be marked (colored) in the seismic cube. This is what causes the final drilling volume to be visible.

### Limitations and simplifications

The computation of the drilling volume suffers from one major weakness: the number of nodes grows very rapidly. The problem is that one node will always split into more nodes, unless a hazard is reached or the node is at the boundary of the seismic cube.

If no simplifications are used, the algorithm can easily grow out of control and may "never" finish. A simple trick to control the behavior of the algorithm is to limit the number of recursive calls that can be made. When this limit is reached, the algorithm terminates. However, as this terminates the algorithm, the solution space will be drastically reduced. The recursive calls limitation works as a compromise between performance and the extent of the resulting drilling volume.

Another major simplification that is used is to reduce how many nodes one node is split into. One node is split into 2, 4 and 3 nodes when moving vertically, diagonally and horizontally, respectively. This is a compromise between performance and the number of holes that appear in the results. Without the simplification, a node could be split into 9 nodes, all with different directions. There would also be more routes in the computation, and thus less holes in the result. The different directions are compounds of changing inclination and/or azimuth. These are presented in Table 4.2.

Direction	Inclination	Azimuth
1	Same	Same
2	Same	Increased
3	Same	Decreased
4	Increased	Same
5	Increased	Increased
6	Increased	Decreased
7	Decreased	Same
8	Decreased	Increased
9	Decreased	Decreased

Table 4.2: All possible new directions by changing inclination and/or azimuth.

### 4.3 Usage

The main purpose of a drilling volume is to display the volume that it is possible to drill into, given the input limitations. This enables the volume to be used as a decision tool while planning a new well. Since the volume only displays reachable parts, everything else can be ignored in the well planning phase. In other words, it works as a filter that filters out areas that can be ignored. This reduces the time spent on planning a new well.

When a drilling volume has been created, there are two main ways to visualize it. One is as a seismic cube, the other one is as a geobody. Which one to use depends on the situation and what to display. Both ways allow the well planner to select points that the new well should go through.

#### 4.3.1 Adding hazard types

In theory, any type of obstacle can be used as input to create a hazard cube. This is because the hazards are generalized into binary values: either there is a hazard at an index, or there is not.

#### 4.3.2 Visualized as a seismic cube

There are both pros and cons of visualizing it as a seismic cube. When a drilling volume is visualized as a seismic cube, it is possible to select a single section of the cube to display. This is useful when one wants to focus on a small part of the volume. The tools of a seismic cube can be used to interactively switch between sections to get an overview of the whole volume, but this does not work too well. Fortunately there are better ways to get an overview of the whole volume.

#### 4.3.3 Visualized as a geobody

A seismic cube can be converted into a geobody. This is very useful when dealing with a drilling volume. The geobody makes it possible to display all of the drilling volume at the same time as a 3D figure. This way of visualizing the drilling volume is useful when a quick overview of an area is required. By cropping a geobody to an appropriate size, it is possible to get a good view and understanding of how a certain part of the volume was reached. On the other hand, it is also possible to see why an area was *not* reached.

#### 4.3.4 Cross-plotting with geobodies

As mentioned earlier in this report, a geobody can be used to cross-plot two or more seismic cubes. It merges the seismic cubes into one volume. This is a major benefit for the drilling volume. It can be used to cross-plot a seismic cube that indicate high reservoir quality. This results in a geobody that shows *reachable and drillable* areas with *high reservoir quality*. This is a very powerful combination that could possibly provide new ideas while planning new wells.

# Results

---

This chapter presents the results. First, the test project is presented. Then the visualization is presented, followed by the measured performance. Finally, a brief description of the hardware that was used is presented.

### 5.1 Data

A fairly simple test project was used for testing. It contained the following data:

**Seismic data:** a seismic cube with real seismic data was used as input. The cube covers an area of approximately 9 km times 7.5 km, with a total depth of approximately 1 km.

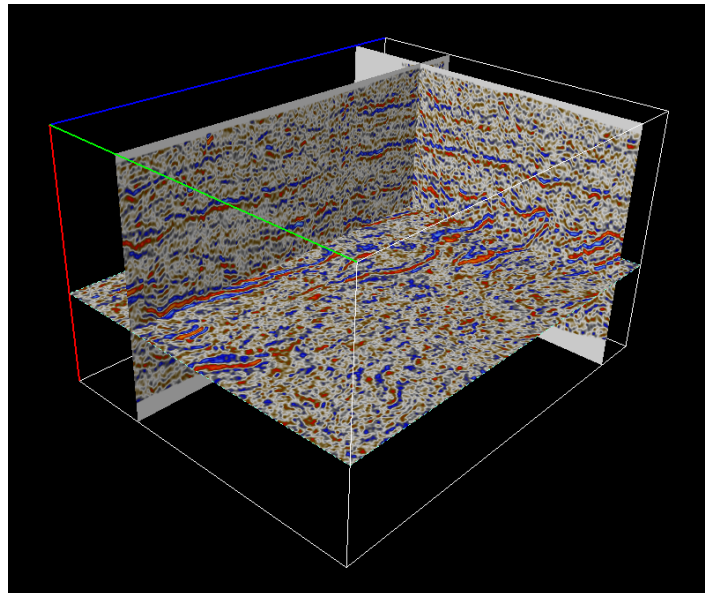


Figure 5.1: The seismic data used as input.

**Wells:** a set of 11 different wells.



Figure 5.2: The wells used for testing.

**Faults:** a set of synthetic faults. Volume attributes were run on the seismic data to create a "fault cube".

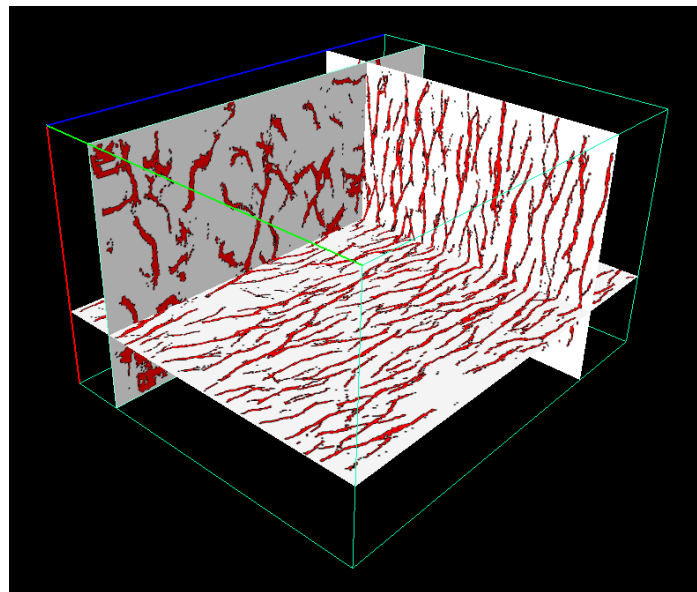


Figure 5.3: A seismic cube with faults.

## 5.2 Visualization

First, the results using no hazards are presented. Then, the results with hazards are presented. Both cases are presented using time slices, inlines and geobodies.



### 5.2.1 Without hazards

Here, the drilling volume has been created using an empty hazard cube (in free space). This results in a symmetrical drilling volume.

#### Using seismic cubes

Figure 5.4 shows a time slice of a drilling volume created using no hazards.

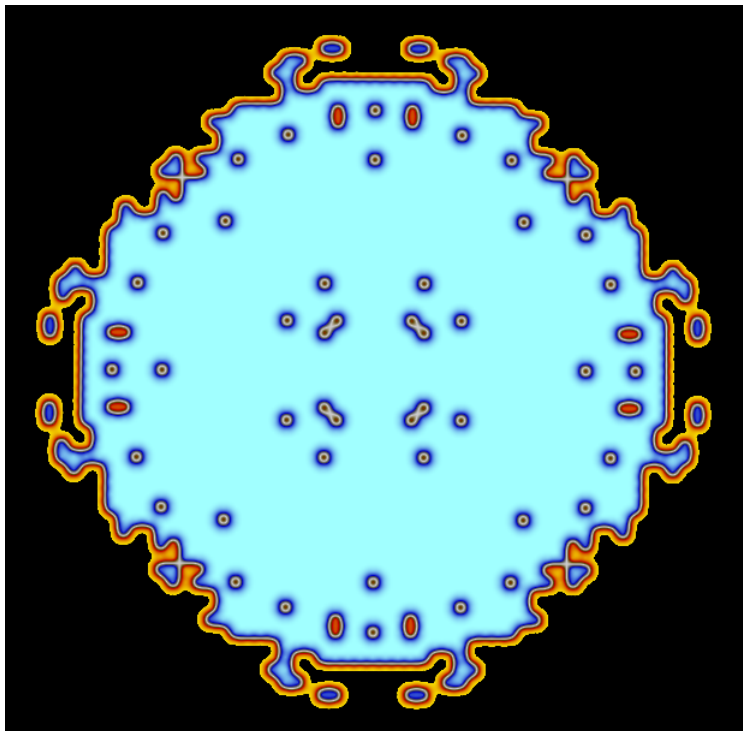


Figure 5.4: Time slice (seen from above) of a drilling volume created using an empty hazard cube.

Figure 5.5 shows an inline section of a drilling volume created using no hazards. The start position can be seen at the top.

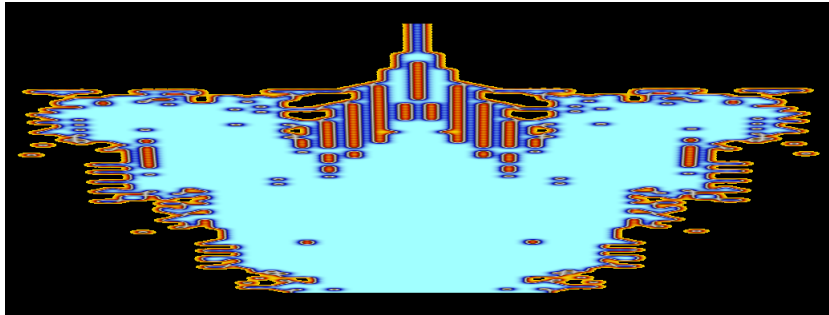


Figure 5.5: Inline section of a drilling volume created using an empty hazard cube. The start position can be seen at the top.

### Using geobodies

Figure 5.6 shows a drilling volume in an area with no hazards. The drilling volume is visualized as a geobody.

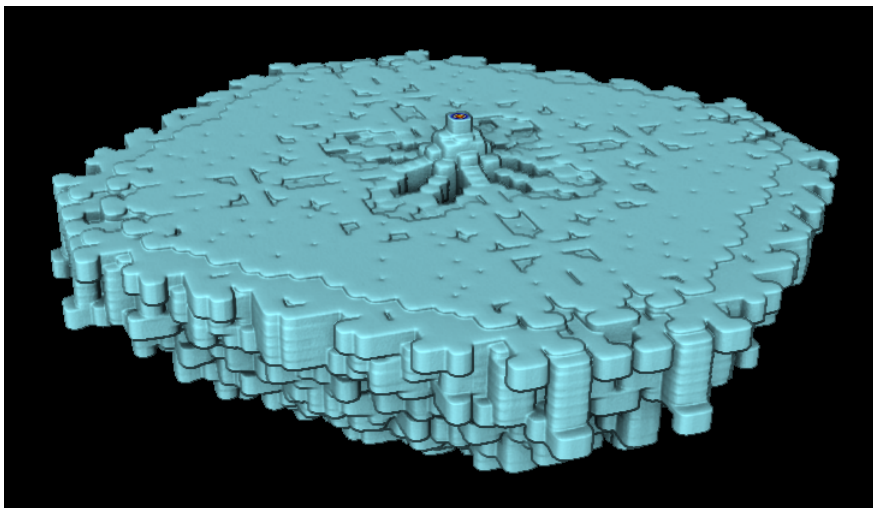


Figure 5.6: A drilling volume created using no hazards, visualized as a geobody. The start position can be seen at the top.

Figure 5.7 shows a drilling volume that was created using no hazards, visualized as a geobody, viewed from a slightly tilted angle. Notice the vertical holes close to the center. The Z-direction (vertical) has been stretched by a factor of 5 to better display the holes.

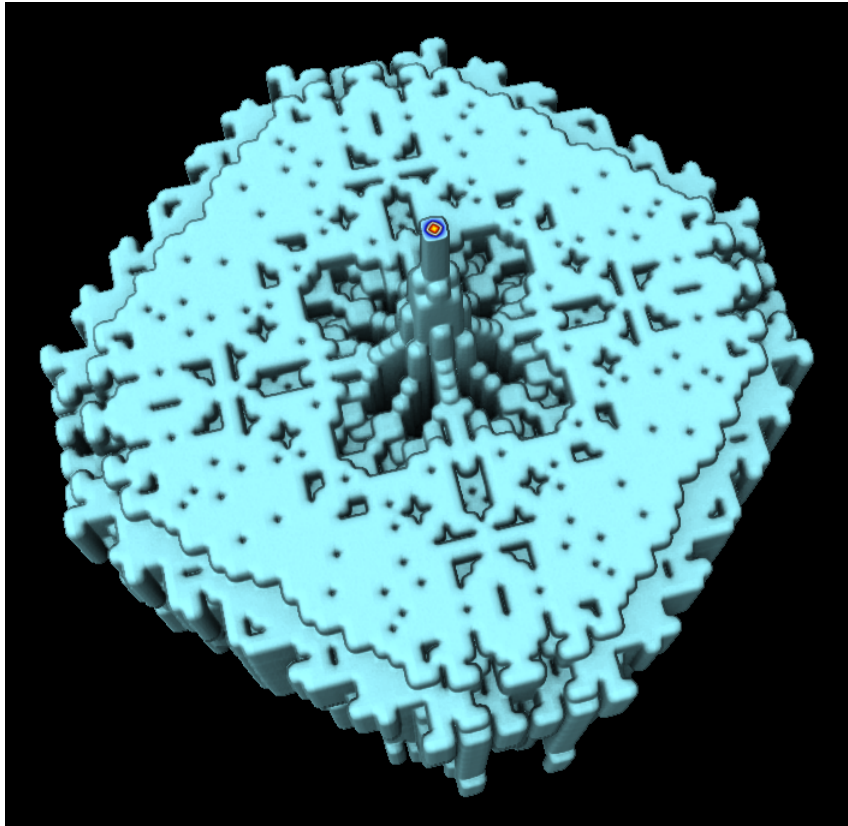


Figure 5.7: A drilling volume created using no hazards, visualized as a geobody. Notice the vertical holes close to the center.

### 5.2.2 With hazards

Here, the drilling volume has been created using a hazard cube full of wells and faults. This results in an asymmetrical drilling volume.

#### Using seismic cubes

Figure 5.8 shows a time slice of a drilling volume and a hazard cube. The drilling volume is colored with a light blue color, while the hazards are red. This is a heavily faulted area with a few wells nearby.

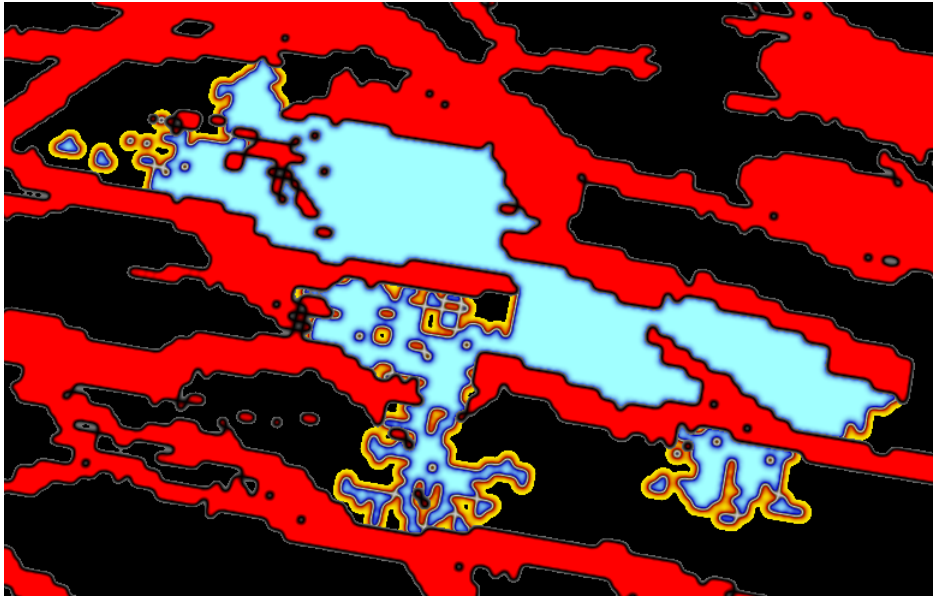


Figure 5.8: Time slice of a drilling volume (blue) with hazards (red). The time slice is viewed from above. The area is heavily faulted with a few wells nearby.

Figure 5.9 shows an inline section of a drilling volume (light blue) and a hazard cube (red).

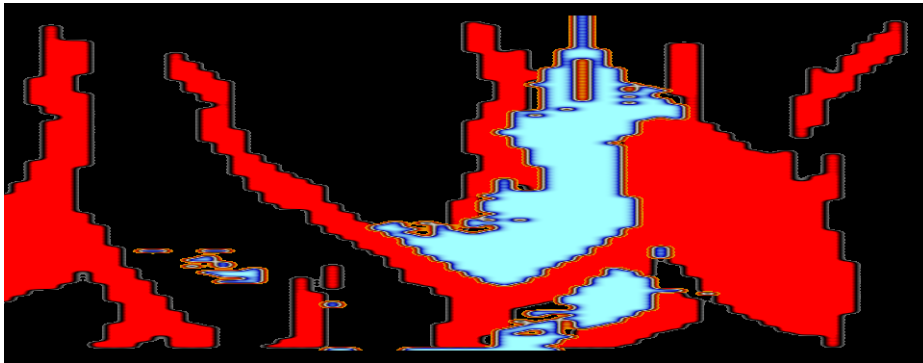


Figure 5.9: Inline section of a drilling volume (blue) with hazards (red). The area is heavily faulted with a few wells nearby.

### Using geobodies

Figure 5.10 shows a drilling volume and parts of a larger hazard cube, visualized as geobodies. The image is taken at a slightly tilted angle from above. The start point can be seen in the center of the image. The drilling volume itself is colored with blue, while the hazards are colored with red

## 5.2. VISUALIZATION

---

(inside of hazards) and dark grey (outside of hazards). The hazard cube has been cropped so that it does not cover all of the drilling volume.

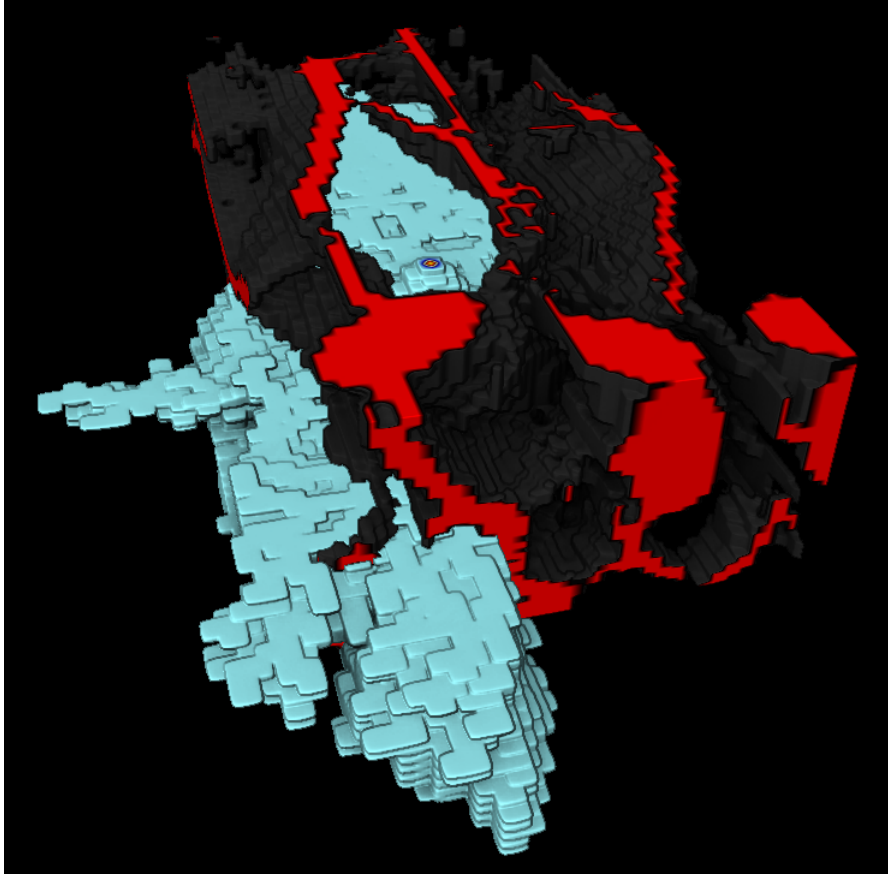


Figure 5.10: Drilling volume (blue) in between a hazard cube (red/black), both visualized as geobodies. The start position can be seen in the center of the image. The hazard cube has been cropped to show more of the drilling volume.

Figure 5.11 shows a drilling volume geobody, a hazard cube time slice and two wells. The start position is seen at the top of the drilling volume. Notice the big hole in the center of the drilling volume.

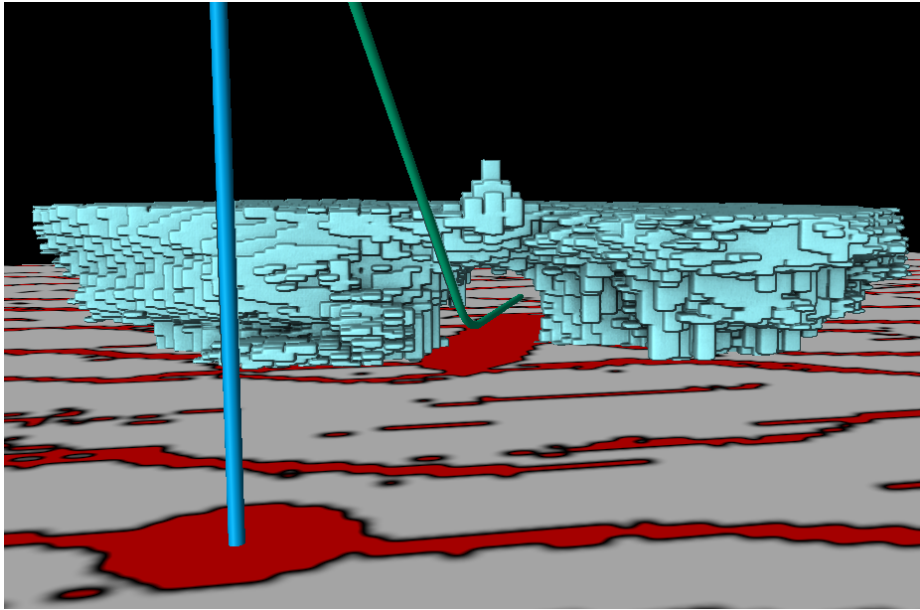


Figure 5.11: Drilling volume (blue) with a time slice of a hazard cube (red/black/white). Two wells are also displayed. The start position can be seen at the top of the drilling volume.

## 5.3 Performance

The runtimes are presented with a plot, followed by the hardware that was used for testing.

### 5.3.1 Runtimes

Here, the measured runtimes are presented using a plot. The exact number of measured seconds is not very important. The important thing is how the plot evolves as the number of recursive calls increases. The performance was measured using a different number of maximum allowed recursive calls. The tests were run for three different scenarios:

1. With no hazards.
2. With boreholes used as hazards.
3. With boreholes and faults used as hazards.

### 5.3. PERFORMANCE

---

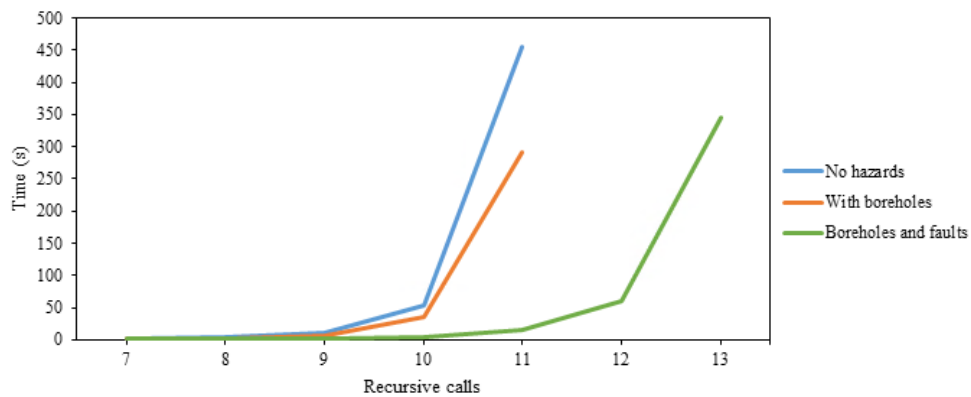


Figure 5.12: Plot of measured performance. The x-axis shows the maximum number of recursive calls that was allowed during the run. The y-axis shows the measured time in seconds.

#### 5.3.2 Hardware

For reference, the testing was done on a fairly powerful laptop with the following hardware:

**Processor:** Intel(R) Core(TM) i7-3920XM CPU @ 2.90GHz, 3.1 GHz, 4 Core(s), 4 Logical Processor(s)

**Memory:** 16 GB DDR3

**Graphics:** Intel HD Graphics 4000 / NVIDIA Quadro K5000M

**Disk:** 512 GB SSD





# Discussion

---

In this chapter the results will be discussed. The main focus will be on the visualization of the drilling volume in different scenarios, but there is also a discussion on performance.

## 6.1 Visualization

The shape and areal extent of the drilling volume is discussed, followed by the holes that appear in the results. Finally, there is a short discussion on DLS limitations.

### 6.1.1 Shape and areal extent

The shape and areal extent of the drilling volume is discussed for the two main cases: with- and without hazards.

#### No hazards

A time slice of a drilling volume is displayed in Figure 5.4. The drilling volume was created in an area with no hazards. Because of this, the computation never terminated by crashing into hazards. The only terminations that happened were caused by the recursive call limit. This is what caused the resulting drilling volume to be symmetrical, as one would expect from the algorithm.

The result from using no hazards tends to expand approximately equally far in every direction when viewed from above. This can easily be seen using time slices — all time slices are symmetrical.

In theory, the computation should be able to reach almost every index in the input seismic cube. However, there are some exceptions, most them being indexes above the topmost horizontal surfaces. If the computation was allowed to run without any recursive call limit, it would have visited almost

all indexes. It would also never finish, since it would be moving around in loops in the cube. See more on this in section 6.2.1.

The higher the number of recursive calls, the larger will the areal extent be. The drilling volume in Figure 5.6 has a lateral areal extent of approximately 750 meters from the start position, in lateral every direction. Since the lateral grid size is 25 m · 25 m, this is the same as 30 grid blocks (750 m/25 m = 30). The input seismic cube had a size of 9 km · 7.5 km, so the drilling volume only covered a small part of the input. On the other hand, it did cover a circle with a diameter of approximately 1.5 km, which is not too bad.

### With hazards

The most interesting results are created in dangerous areas. When there are a lot of hazards nearby, e.g. wells or faults, the computation terminates a lot of the nodes. This reduces the runtime, as there are many fewer nodes being evaluated. The shape of the resulting drilling volume is controlled by the position of the hazards. The time slice in Figure 5.8 shows this in a very good way. The computation has tried to fill all empty areas in-between the hazards (red). Every time a node reached a hazard it was terminated, and the computation continued with another node. This continued until the recursive call limit was reached. The recursive call limit is the reason why the computation did not visit the black areas that were not blocked by the hazards.

The inline section in Figure 5.9 also shows how the drilling volume was shaped by the hazards. Here, the start position is visible at the top and it is possible to see how the computation has moved downwards along the hazards.

The geobodies in Figure 5.10 show both the drilling volume (blue) and the hazards (dark grey/red). Note that the hazard cube is cropped, so not all hazards are visible in the image. The start position is visible in the center of the image. The computation moved downwards in-between the first hazards. From there, it found some paths further down that led into the area on the left side of the figure. Once again, it is clear that the hazards control the shape of the drilling volume. Since the shape is controlled by the hazards, the areal extent in blocked directions will also be affected. The right side of the start position has not been visited at all, since there are hazards blocking that side. In general, the areal extent will be smaller whenever there is a hazard blocking the path.

### 6.1.2 Holes

There are many holes in the drilling volume results. These are all non-visited indexes. Some of them have not been visited since the computation

was terminated before they were reached. Others were not possible to reach because of the DLS limitations or because there were hazards there.

The time slice in Figure 5.4 has many holes in it. The holes close to the center were actually reachable, but they were not visited. This may be due to some of the direction limitations that were used in the algorithm (see directions in section 4.2.2).

The geobody in Figure 5.11 has several big vertical holes close to the center. The reason why these appeared may also be because of the direction limitations. The computation could not find any possible routes with the given limitations. Because of this, those indexes were never visited. This is a weakness in the algorithm, but it was done to reduce the runtime (see section 6.2.4).

A hole may be caused by hazards. These holes are supposed to appear where there are hazards. This is very easily seen in Figure 5.11, where a drilling volume is shown with a time slice of a hazard cube and two nearby wells. The big hole in the center of the drilling volume was caused by the deviated well that goes through the drilling volume.

### 6.1.3 Limitations vs. reality

One of the limitations being used is the DLS. The DLS tries to control how quickly the computation can turn. In a grid, this is analogous to how many indexes one has to move along one direction before it is possible to change index in the direction one tries to turn. Figure 5.5 shows an inline section of a drilling volume created without hazards. The effect of the DLS limit is seen by how the drilling volume is shaped from the start position down to the topmost horizontal parts. The computation has moved from index to index, trying to create a continuous circular arc. This is happening because the computation always tries to turn at the highest allowed rate, i.e. turn at the rate of the DLS limit. The end result is not too bad — it resembles a circular arc.

Another limitation is that the computation is never allowed to have an inclination above 90 degrees, i.e. it is not allowed to move upwards. In reality, it is possible, and some times necessary, to drill upwards to reach a target. This limitation results in close-to-horizontal surfaces once the computation is moving horizontally. This is visible on several of the figures, such as Figure 5.6.

## 6.2 Performance

Several factors affect the performance. The major ones are discussed here. The performance is a problem, but there are ways to improve it. The plot in Figure 5.12 shows the measured performance for three different cases:

Note that the tests were done with a fairly powerful laptop.

1. With no hazards
2. With boreholes as hazards
3. With boreholes and faults as hazards

The number of allowed recursive calls is displayed on the x-axis. The runtime grows at an extreme rate once the computation is allowed to use a certain number of recursive calls. This is discussed further in the sections below.

### 6.2.1 Infinite loops

One problem with the algorithm is that it will create infinite loops, without the recursive call limitation. The loops will occur only in the horizontal plane. The computation may move around in circles at the same depth. This should be handled in a better way, since it decreases the performance by a lot. Infinite loops will, of course, go on infinitely and thus never terminate. The creation of a drilling volume would thus never finish.

Below are two possible (and similar) ways to handle this problem, though there probably exist many more:

**Limited number of turns:** the number of turns can be limited for every single depth level in the seismic cube. This would cause the computation to terminate whenever a node has done a certain number of turns at the current depth.

**Limited change in azimuth:** when moving along a horizontal plane, at one single depth level, it is possible to limit how much azimuth a node is allowed to change. Whenever a node turned, it would record how much it turned (how much the azimuth changed) and accumulate the value. When the value would reach a certain number, the computation could terminate.

If the computation was allowed to move upwards, the problem of infinite loops could occur in all directions. This would require a more complex solution.

### 6.2.2 Stop criteria

The only stop criteria that were used while creating the drilling volumes were the number of recursive calls and hazards. This means that when a route had done a certain number of turns, it would terminate. It would also terminate whenever it hit a hazard.

The stop criteria are the most important things that stop the runtime from growing towards infinity. The stop criteria lowers the runtime by factors — without them, the computation would never finish.

### 6.2.3 Runtime vs. hazards

As can be seen in Figure 5.12, the runtime grows at a very high rate when a drilling volume is created in an area with no hazards. The same thing happens when only boreholes were used as hazard input. The interesting thing is that the runtime starts to grow at a high rate at a later stage when boreholes *and* faults were used. This happened after only two more recursive calls than in the two other cases, so it is not a massive improvement. It is however a clear indication on that the drilling volume works best in areas with a lot of hazards. The performance increases the more limited the non-hazardous space is. In other words, if the space the computation has to evaluate is small, the computation will finish faster. If the space is large, the computation will require more time.

### 6.2.4 Node splitting and new directions

Section 4.2.2 describes what happens when a node splits into new nodes. At every "turning point", a node should really be able to split into 9 new nodes. The results that have been presented here have only split into 2, 4 and 3 nodes when moving vertically, diagonally and horizontally, respectively. This was done to increase the performance. The node splitting affects the growth rate of the algorithm very hard. With the current limitations, the number of nodes will never grow faster than  $O(4^n)$ . However, the real limit would be even lower, as the growth rate depends on the current direction.

No proper tests were done to see how much the runtime would increase by using all 9 possible new directions when splitting nodes. It is however very clear that this would affect the performance heavily.



---

## Conclusion

---

An integrated visualization technique for wellbore placement has been obtained. The solution, a *drilling volume*, shows areas that are safe to drill into and also possible to reach. The resulting data is visualized using existing technology such as seismic cubes and geobodies. The data is stored in a seismic cube, thus making it possible to cross-plot safe and drillable areas with other variables, e.g. variables that indicate high reservoir quality. The solution filters out unreachable and hazardous areas so that they may be ignored while planning new wells. The visualization technique is in theory generic, making it possible to use it with any types of hazard as input. It has proven to be a useful tool that gives a good overview of the hazards in an area and how they may be avoided them while drilling.

The solution presented in this report should not be considered flawless. There are several flaws, especially when it comes to performance. It should however be thought of as a good first try to solve the problem. The full size of a seismic cube, i.e. the number of cells it has, is usually too large to be handled effectively by the solution. The worst-case runtime may be written as  $O(n^4)$ , where  $n$  is the number of nodes required to manoeuvre through a cube. The runtime is however lowered when the computation is started in a very hazardous area, as many routes are terminated once they reach a hazard. In other words, the performance is better in very hazardous areas.

The areal extent of a drilling volume is depending on the performance. If the performance is low, e.g. when evaluating an area with no hazards, the areal extent is also low. There may also be some areas that are not covered by the drilling volume, even though they are reachable. These areas appear as holes in the result.

The drilling volume can be a useful tool while planning new wells in a field with a cluster of wells and other hazards. It should however be used with care, as it does not provide a perfect solution. Even though an area is marked as safe and drillable, it may not actually be the case, since there may be other hazards there that the drilling volume did not evaluate. On the other hand, areas that are not covered by the drilling volume should not

---

necessarily be ignored. The areas may not have been evaluated because of performance-related limitations, and not by hazards or drilling limitations.



---

## References and Bibliography

---

- Bang, J. and Torkildsen, T. (2011). Wellbore anti-collision safety: Separation distances must be increased due to degraded positioning accuracy in northern areas.
- Cayeux, E., Genevois, J.-M., Crepin, S., and Thibeau, S. (2001). Well planning quality improved using cooperation between drilling and geosciences.
- Cormen, T., Leiserson, C. E., Rivest, R. L., and Stein, C. (2009). *Introduction to Algorithms*. The MIT Press, 3 edition.
- Garg, T. and Gokavarapu, S. (2012). Lessons learnt from root cause analysis of gulf of mexico oil spill 2010.
- Gluyas, J. and Swarbrick, R. (2004). *Petroleum Geoscience*. Blackwell Publishing.
- Jahn, F., Cook, M., and Graham, M. (2008). *Hydrocarbon Exploration and Production*. Developments in Petroleum Science. Elsevier Science.
- Jamieson, A. (2012). Introduction to wellbore positioning.
- Rosen, K. H. (2012). *Discrete Mathematics and Its Applications*. McGraw-Hill Education, 7th edition.
- Sanford, S., Mathis, R., Egan, M., Gupta, V., and DiPippo, E. (2013). Case history of a challenging thin oil column extended reach drilling (erd) development at sakhalin.
- Skinner, G. D. M. (2011). Wellbore positioning practices and challenges in africa.