COPPE
Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia
UFRJ

INTERACTIVE CUTAWAY RENDERING OF CORNER-POINT MODELS

Felipe Moura de Carvalho

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Ricardo Guerra Marroquim
Antonio Alberto Fernandes Oliveira

Rio de Janeiro
Dezembro de 2014

INTERACTIVE CUTAWAY RENDERING OF CORNER-POINT MODELS

Felipe Moura de Carvalho

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

____
Prof. Ricardo Guerra Marroquim, D.Sc.

____
Prof. Antonio Alberto Fernandes Oliveira, D.Sc.

____
Prof. Claudio Esperança, Ph.D.

____
Prof. Mario Costa Sousa, Ph.D.

____
Prof. Waldemar Celes Filhos, D.Sc.

____
Prof. Rodrigo Penteado Ribeiro de Toledo, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2014

Carvalho, Felipe Moura de

Interactive Cutaway Rendering of Corner-Point Models/Felipe Moura de Carvalho. – Rio de Janeiro: UFRJ/COPPE, 2014.

XVI, 80 p.: il.; 29, 7cm.

Orientadores: Ricardo Guerra Marroquim
Antonio Alberto Fernandes Oliveira

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2014.

Referências Bibliográficas: p. 71 – 79.

1. Cutaway. 2. Interactive. 3. Computer Graphics. I. Marroquim, Ricardo Guerra *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

A minha mãe, Bernadete Moura.

Acknowledgement

Quero agradecer ao Prof. Ricardo Marroquim por tudo, tanto dentro e fora do mundo acadêmico.

Quero agradecer a todos o membros do LCG que eu convivi nesses últimos 8 anos de luta.

Agradeço ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e a Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) pelo suporte financeiro.

Enfim, todos aqueles que do fundo do coração, me ajudaram nessa longa jornada , que ainda continua. =)

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

RENDERIZAÇÃO INTERATIVA DE *CUTAWAYS* EM MODELOS DE *CORNER-POINT*

Felipe Moura de Carvalho

Dezembro/2014

Orientadores: Ricardo Guerra Marroquim
Antonio Alberto Fernandes Oliveira

Programa: Engenharia de Sistemas e Computação

A indústria de Oil&Gas utiliza simulação numérica para a previsão de produção e melhorar a recuperação de hidrocarbonetos de campos de petróleo. Uma vez que os reservatórios são entidades encontradas a grandes profundidades na crosta terrestre, um modelo geológico $3D$ simplificado que imita este ambiente é gerado para compreender conceitos geológicos e geofísicos. Para aplicar simulações, este modelo é discretizado em uma grade $3D$ de células hexaédricas. Além disso, esse modelo é utilizado posteriormente para apresentar o resultado de simulações, atribuindo um escalar para cada célula. Neste cenário, é importante investigar o comportamento interno das células que são muitas vezes ocluídas por outras. Métodos de renderização Cutaway reduzem a oclusão removendo seletivamente os objetos de menos importância para, ao mesmo tempo, expor características importantes e manter parte dessas células, a fim de assistir a visualização e manter informações contextuais. Esta tese introduz um método interativo baseado em *GPU* (*Graphics Process Unit*) para renderização de modelos de reservatórios. A nossa técnica é baseada no valor da profundidade gerada a partir da geometria das células em foco. Apresentamos também modificações no *pipeline* de gráfico para renderizar as parede do corte, bem como inserção de linhas para ajudar a transmitir forma e noção de percepção de profundidade da cena.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

INTERACTIVE CUTAWAY RENDERING OF CORNER-POINT MODELS

Felipe Moura de Carvalho

December/2014

Advisors: Ricardo Guerra Marroquim
Antonio Alberto Fernandes Oliveira

Department: Systems Engineering and Computer Science

The Oil&Gas industry uses numerical simulation to forecast production and enhance the hydrocarbons recovery of oil fields. Since hydrocarbons reservoirs are entities buried deep in the earth's crust, a simplified $3D$ geological model that mimics this environment is generated to understand geological and physical concepts. In order to run simulations this model is discretized in a $3D$ grid of hexahedral cells. In addition, it is also used to present the results of simulations by assigning scalar values to each cell. In this scenario, it is important to investigate the behavior of internal cells that are often occluded by others. The Cutaway rendering method manages the occlusion problem by selectively discarding portions of these overlapping objects with less importance to, at the same time, expose the important features and keep part of the secondary cells to support the visualization with contextual information. This thesis introduces a *GPU* interactive method for cutaway rendering of $3D$ reservoir models. Our technique is based on the depth value generated on the fly by the proxy geometry of the cells in focus. We also present modifications of the polygonal rendering pipeline to properly render the cutaway wall, as well as the wireframe lines that are important to convey shape and amplify depth perception.

Contents

List of Figures

List of Tables

Lista de Abreviaturas

BRICS	*The acronym for an association of five major emerging national economies: Brazil, Russia, India, China and South Africa*, p. 10
CPU	*Central Processing Units*, p. 5
CSG	*Constructive Solid Geometry*, p. 18
Cg	*C for Graphics*, p. 6
EOR	*Enhanced Oil Recovery*, p. 13
E&P	*Exploration and Production*, p. 12
GESY	*Global Energy Statistical Yearbook*, p. 10
GLSL	*OpenGL Shading Language*, p. 5
GPU	*Graphics Process Unit*, p. vii
HLSL	*High Level Shading Language*, p. 5
PCA	*Principal Component Analysis*, p. 26
R&D	*Research and Developement*, p. 1
WEO	*World Energy Outlook*, p. 10

Chapter 1

Introduction

"All we have to decide is what to do with the time that is given us."
— J.R.R. Tolkien, The Fellowship of the Ring

The energy requirements around the world are expected to grow. Even with blooming and expanding green energy *Research and Development* (*R&D*), the developing society will still need petroleum for at least another few decades. A common concern is that the oil and gas reserves that are easily accessible have, to a large extent, been already explored. The sources that are left are either in geologically complicated areas (e.g. Arctic environment, deep sea reservoirs, jungles) or contain heavy oil or oil sands, that are difficult to extract and process [1]. Therefore, new technologies and any possible improvements to all stages of the oil recovery can contribute to the world energy supply.

In this thesis we present a method for generating cutaway visualizations of reservoir models. We follow suggestions on how to generate meaningful cutaways from previous studies, and transfer these good practices to corner-point models. However, due to the volumetric nature of reservoirs, a straightforward application of known methods performs poorly with these complex models.

1.1 Motivation

Due to the advance in data acquisition and computational power, the amount of information created worldwide is growing exponentially year by year [2]. Acquiring and storing the data has limited importance. The real value is achieved when we comprehend it and, consequently, the data becomes a rich source of information for decision making. It is common for the acquired data to come in the form of $3D$ models, containing hundreds or thousands of objects in close proximity and often nested within other objects. The complexity to comprehend the data increases significantly due to the amount of information available to analyze, and is more critical when the data has some time varying attributes. This scenario has created an urge for effective visual communication techniques that allow the user to intuitively and interactively explore and comprehend the data.

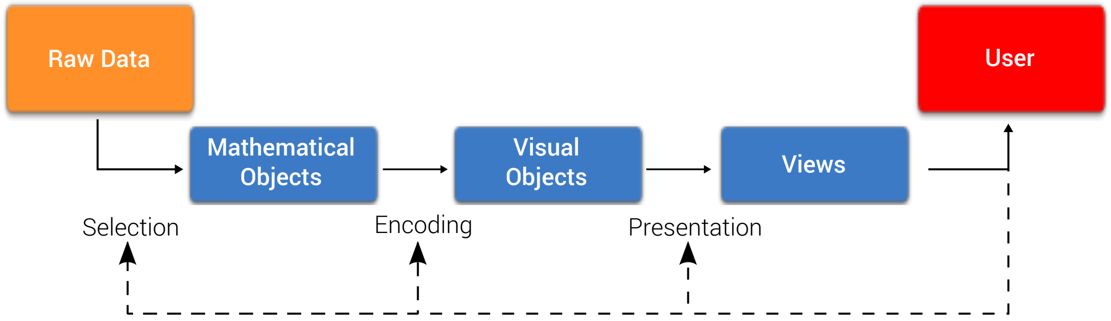


Figure 1.1: The visualization pipeline. The **Selection** process converts raw data of some distinctive format for encoding into a visual form in the next step of the pipeline. The **Encoding** transformation accepts structured data and generates a visual representation. Simply rendering the visual object is not enough to get all information we desire. We also want to gain insight and find patterns and inconsistencies, since the data in most cases is noisy, and the visual object itself is not capable to provide these answers. To this end, a **Presentation** stage is necessary. Nonetheless, it is not always possible to view the entire object due to its complexity or size, thus, we employ various transformations to reveal relevant details, without losing the context. Image adapted from [3].

Research has been carried out to transform raw data in effective representations to allow the user to interpret the data in new and more efficient ways. These representations are broadly called *visualization techniques* [4]. The field of visualization is concerned with graphical representations of any kind of data to help in creating a *mental model* of it [3, 5] and consequently, making the data an useful source of information for the purpose of aiding cognition. Figure 1.1 depicts the general visualization pipeline. Before we go any further, it is important to make a clear distinction between data and information. The former is raw material for data processing. It is related to unorganized facts, events and transactions. Meanwhile, information is when the data is processed, organized, structured or presented in a given context, in a manner that it becomes comprehensible to the users. It is anything that is communicated. In brief, users wish to derive information from data [6].

Essentially, visualization researchers are concerned with how to draw pictures (or rather, have the computer draw pictures and use animations in some situations) that encode the information we are trying to depict from the data. As a result, it helps the user to understand it more easily, or spread the information to an audience with no background in the subject being exposed. There is a statement that a picture is worth more than a thousand words; thus, it can be a way to solve the plethora of information issues outlined in the beginning of the section.

KOSARA [7] points out two cultures in visualization: the part that is classified as *pragmatic visualization* is mostly practiced by the computer science community with no background in art or design. On the other end, artists and designers often work on visualization without much knowledge of technical work being done in computer science.

This thesis joins knowledge from both sides: we make use of computer graphics techniques to realize *Cutaways* rendering in an interactive way to solve the occlusion problem in the manner artists usually follow when creating static illustrations. Cutaways is a specific visualization technique that will be further detailed in this thesis.

1.1.1 Reservoir Visualization

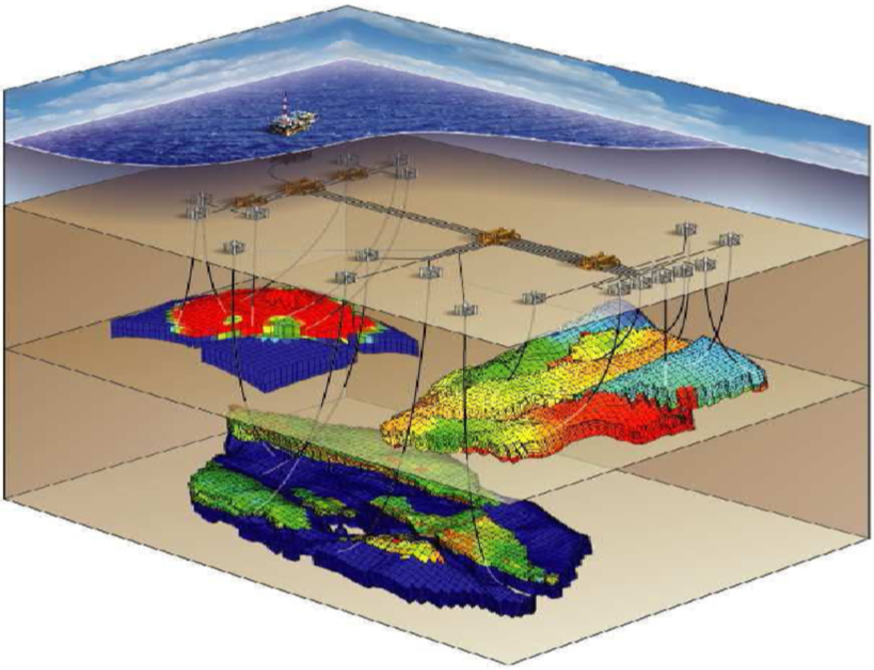


Figure 1.2: Illustration of an off-shore oil field with its respective $3D$ Reservoir Model [8]

The Oil and Gas industry is considered one of the largest industries of all times, involving many large-scale companies worldwide. The increasing demand for energy, along with the gradual depletion of the easily accessible hydrocarbons reservoirs, has motivated the industry to maximize the oil recovery of existing fields. This shift stimulated intensive research to better understand complex fluid flow mechanisms that occur inside the oil fields, as well as the development of advanced reservoir visualization and simulation techniques [9].

Obtaining interpreted results from raw data can sometimes be done automatically; however, there are numerous situations where there is a need, during all processing stages, to visualize the data in an interactive way. This enables the domain experts to gain intuition, discover unexpected patterns, and find guidance about subsequent analysis steps. However, visualization is by its very nature problem-oriented, each of the techniques in the area are more or less tailored for a specific kind of information and a specific domain. This means that a single visualization technique is unlikely to ever become the “best” for all data representation [6].

In order to understand the flow behavior of a reservoir, a $3D$ computational geological model is created by domain experts. This model represents the reservoir geometry, its intrinsic properties and its fluid content. Once the geometry is constructed, the other properties of the model should be filled, and a tuning process, known as *history matching*, is triggered. Based on sparse data the model is populated with static properties, then the flow is computed through a simulation and the results are tested against observed data. If the computed result diverges from the observed data, the model parameters are manipulated and the simulation is restarted. This interactive process is repeated until an acceptable degree of accuracy is reached.

As a reservoir matures and more wells are drilled, more pieces of the puzzle are available. The conceptual model of the reservoir changes with time. In most cases, the model will be refined continually. In some instances, accepted ideas will have to be discarded and a newer model developed.

Visual inspection is an important asset for analysis in all steps of this interactive process, as well as when using the final model as a predictive tool. The standard representation for oil reservoirs is **Corner-Point Grids**, and there are in the order of millions corner-point models currently used by the industry [10]. Nonetheless, these models have volumetric characteristics, and some important phenomena may occur in the interior cells that are usually occluded over several layers of outer cells, and are hard to visualize using naive methods.

1.1.2 Occlusion and Scene Complexity

The increasing computation power of modern computers, and the desire of more accurate models and the representation of the world around us, have driven the conception of high fidelity representations, and more objects are tied together in order to precisely describe a scene. Apart from the processing and modeling stages, there is also an advance in acquisition devices, which leads to highly detailed surfaces and volumes, as well as an increased structural complexity of the scenes.

With the increase in computational power, rendering algorithms also become

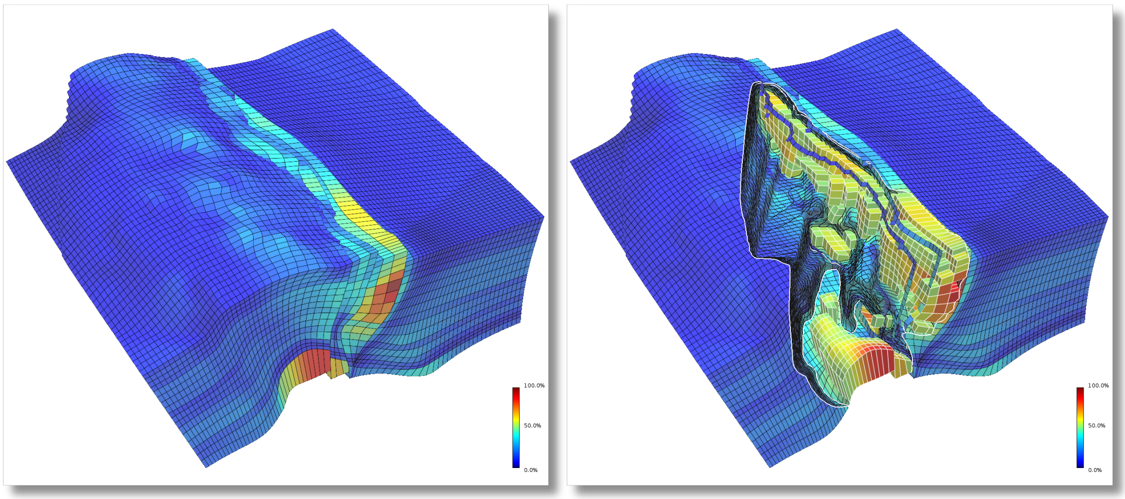


Figure 1.3: $3D$ Reservoir visualization with our Cutaway technique.

more efficient and are able to tackle highly detailed models. However, the problem is not only the shear size of the data, but also its complexity, mainly due to the occlusion problem. In order to reduce the occlusion effect and expose internal information, novel visualization techniques must be proposed.

Illustrative techniques are an usual way to solve this issue. A commonly used method is *Cutaways*, where the artist creates the illusion of the object being cut to expose its interior without losing the general context. In the Computer Graphics literature, researchers have investigated solutions to generate these illustrations automatically. Given that an object in focus (the part or object of interest) is selected or identified, the goal is to compute the appropriate cut surface that eliminates occluding parts. Figure 1.3 illustrates the idea of the Cutaway.

1.2 GPU Programming

GPUs are specialized processors designed to reduce the workload of the *Central Processing Units* (**CPU**) when computing graphic or video intensive tasks. Over the last two decades, *GPUs* have become an integral part of computing platforms for video games, interactive simulations, and high-end $3D$ rendering. Nowadays, *GPUs* are flexible, that means, the *standard pipeline* is programmable, making it extensible to solve tasks direct on the device, and leaving the *CPU* free to do others jobs.

A pipeline is a sequence of several stages, where each stage takes its input from the previous one, performs some operations on it, and then sends the output to the next one. In order to display geometry on the screen, the data has to go through such

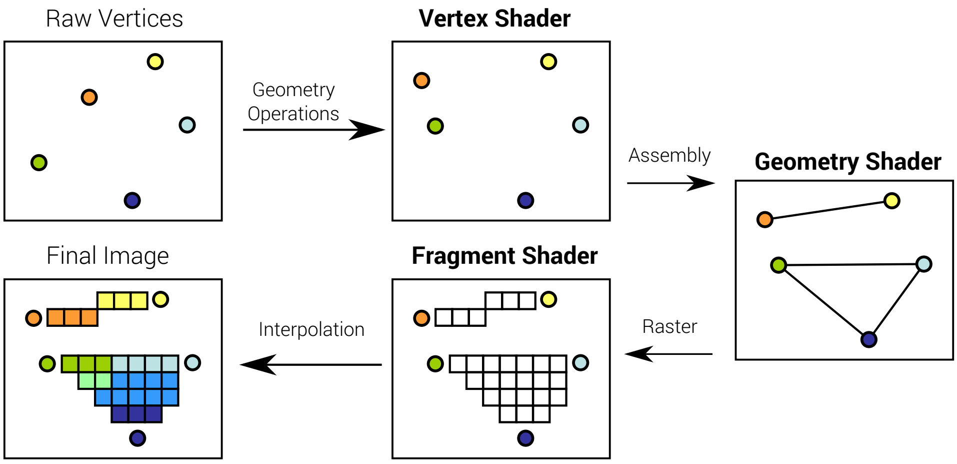


Figure 1.4: Simplified OpenGL Pipeline. Image adapted from [11].

a pipeline on the *GPU*; see Figure 1.4. All the geometries are usually represented by triangles, and every triangle contains three vertices. These vertices, along with some of their attributes (original position, color, texture coordinates, etc.), are sent from the *CPU* to the *GPU*, and processed through the first stage of the pipeline – **Vertex Shader**. In this stage, the vertices are transformed and some new attributes such as illuminated color, transformed normal, etc. are also computed. The processed vertices are then sent to the second stage – **Geometry Shader**. The vertices are assembled into triangles, and the triangles usually represent a mesh. These triangles are then rasterized into many fragments. Note that fragments are in some sense similar to pixels, but they are conceptually different. One fragment corresponds to one pixel on the screen, but one pixel can have more than one fragment. The third stage in the pipeline is the **Fragment Shader**, where the attributes of these fragments are computed. The processed fragments are then sent to the final stage – **Raster Operations**. In this stage, they must go through some common graphics tests, such as stencil test, depth test, etc. Only the surviving fragments can affect the contents of the corresponding pixels. The updated pixels are sent to the frame buffer and finally displayed on the screen.

Even though there are other available programmable stages in modern graphics cards, we will not discuss them in this thesis, since we only make use of the three shaders described above.

1.3 Thesis Contribution

Our main contribution is a cutaway visualization method for reservoirs represented as corner-point models. The technique is based on a flexible screen space representation for clipping volumetric cells. In our case, these cells are selected by varying ranges of attribute values, such as pressure or porosity. By making extensive use of *GPU* techniques the method achieves interactive frame-rates even in the face of complex topology and geometry. In addition, we describe some feature emphasis techniques to enhance the cuts and provide more visual clues during the inspection process. In Figure 1.3 we show a corner point model and the proposed technique in action.

1.4 Thesis Outline

The remainder of this thesis is organized as follows:

- *Chapter 2*
  A brief overview of Oil&Gas Exploration and Production and where they make use of visualization to gain insight about the behavior of the field and make decisions to enhance production.

- *Chapter 3*
  The related works are divided in three domains that make use of certain kinds of occlusion reduction methods based on cutaways: Volume rendering 3.1, Polygon models 3.2 and Geological models 3.3.

- *Chapter 4*
  The detailed description of our cutaway method for corner point models. Results are also presented in this chapter.

- *Chapter 5*
  Analyze of performance, as well as a design critique of the proposed method are present

- *Chapter 6*
  Conclusions and directions for future works.

Chapter 2

Oil&Gas Background

"if you can't measure it, you can't manage it."

— Peter Drucker

Petroleum, or crude oil, and other hydrocarbons are known by mankind since the dawn of civilization [12, 13]. The word petroleum has roots on the Latin words *petra*, meaning rock, and *oleum*, denoting oil, which combined literally means rock-oil. This term is coined by the German mineralogist Georgius Agricola in the treatise *De Natura Fossilium* [14]. However, some languages, such as Russian or Arabic, use variants of the ancient word *naphtha* as the word for petroleum.

Petroleum is one of the most important natural resource of the industrialized nations. It can be used as an energetic source to generate heat, drive machinery and fuel vehicles and airplanes. Apart from its primary usage, some petroleum derivatives are used to manufacture synthetic rubber, synthetic fibers, drugs, detergents, and plastic, just to cite a few.

A common concern is that the oil and gas reserves that are easily accessible have, to a large extent, been already explored. The sources that are left are either in geologically complicated areas (e.g. Arctic environment, deep sea reservoirs, jungles) or contain heavy oil or oil sands, that are difficult to extract and process [1]. Therefore, new technologies and any possible improvements to all stages of the oil recovery can contribute to the world energy supply.

Understanding the petroleum systems on a formation basis is fundamental in the tasks of identifying possible reservoirs and knowing whether sufficient hydrocarbons have filled potential traps with economically viable reserves. Assimilation of subsurface data and the construction of subsurface geologic models are critical to hydrocarbon exploration and production. Therefore, it is not surprising that visualization of digital geologic models has impacted the work processes of domain experts.

This chapter overviews the Oil&Gas Exploration and Production cycle, and where this thesis is inserted.

2.1 World Energy Demand

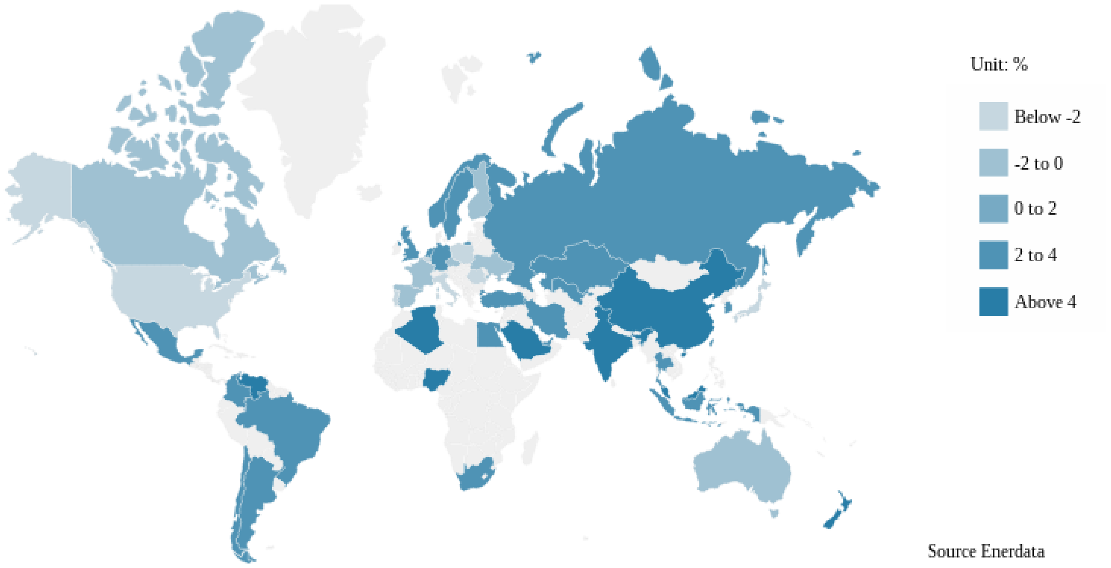


Figure 2.1: World energy consumption growth between 2011 – 2012. The growth is leveraged by the BRICS [15].

The global energy demand increases every year. According to the *Global Energy Statistical Yearbook* (*GESY*) 2013 [15], there will be a significant increase in energy consumption in the coming years, particularly in countries of the *BRICS* (**The acronym for an association of five major emerging national economies: Brazil, Russia, India, China and South Africa**). At the same time, renewable energy becomes more important. Its production is less harmful to the environment and starts to become economically competitive compared to the production of energy from fossil fuels. Thus, as stated by the *World Energy Outlook* (*WEO*) 2013 [16] the use of renewable fuels is expected to grow strongly over coming years. Still, oil and natural gas will remain the main energy source until at least 2050.

Petroleum has been used primarily as an energy for heating, electricity and transportation. Particularly, some chemicals (also known as petrochemicals) including, kerosene, diesel and gasoline are the most convenient fuels for the internal combustion engine, wildly used on automobiles, aircraft and ships. Although most petroleum is used to generate energy, it is also an important feedstock. The second use of petroleum is for synthesizing organic compounds. By 1965, about 80% of the world organic chemicals were synthesized from petroleum, and this figure rose to 98% in 1980 and 99% in the year 2000 [12]. Components of petroleum also serve as lubricating oils and solvents. Because of these valuable uses, many have suggested that we are wasting a valuable feedstock resource by continually burning it for energy [17].

Every year it becomes more difficult for the Oil companies to meet this huge demand for petroleum. First of all, many large fields are already at a mature stage. Other fields, recently discovered, are often too small to be exploited efficiently. It is therefore essential to develop new technologies that allow reducing costs for maintaining oil fields and increasing the rate of oil recovery from the existing fields.

2.2 Origin of Oil&Gas

Oil&Gas formation are created by the accumulation of organic material being compressed and heated in the sea bed over several millions of years. Over time, this mix of organic and mineral material were transformed into a complex mixture of hydrocarbons, together with various others chemical components. Because the earth is filled by layers of soil (mostly at significant depths), the contained Oil&Gas cannot exist within a "lake" or "pool", but must reside in the small fraction of space (or pores) that exists in the rocks, also known as source rock [18, 19]. The relative light mass of the hydrocarbons, causes them to move upwards, in a process called migration, but a seal rock formation, known as trap, can prevent the hydrocarbons from escaping to the surface and allows them to stay at place. This system, often referred as petroleum system [20], can be reduced to three essential components: a source rock, maturation, and trap. A simple illustration of this process is shown in the Figure 2.2.

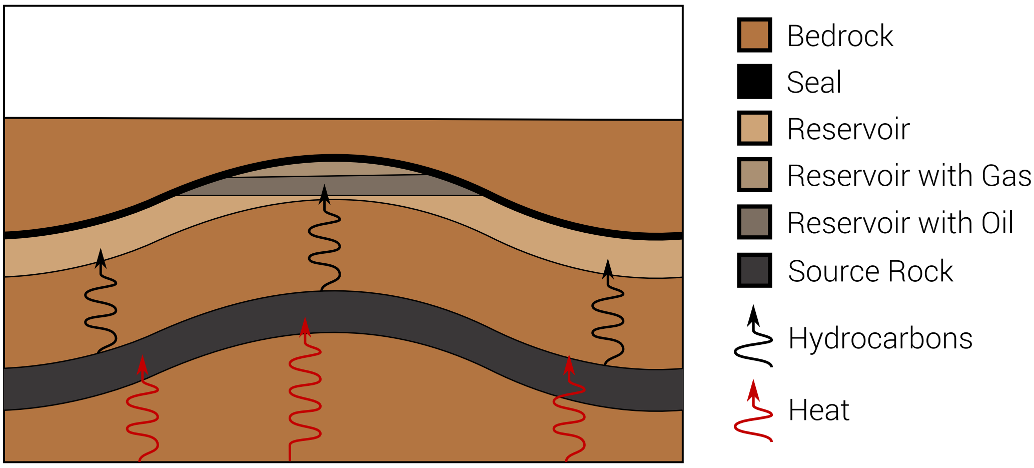


Figure 2.2: Petroleum system, consisting of a source-rock and hydrocarbons after maturation, that migrates to the reservoir-rock where it is trapped by the seal. Image adapted from [19].

2.3 Exploration and Production (E&P) Process

The Oil&Gas industry is formally classified in two main sectors: *upstream* and *downstream*. The upstream sector is designated to the exploration and production of the raw material. The downstream is concerned with the refining and processing of this raw material and marketing its derived products. It is also responsible for the logistic: transportation from the field and the distribution of the products [21]. First of all, the *Exploration and Production* (**E&P**) involves locating potential hydrocarbons reservoirs (the portion of the earth crust that contains oil and gas, also referred as trap, as shown in Figure 2.2). This task is carried out by analyzing rock samples collected in the field, and through *seismic data observations*. To gather seismic data, a sound wave is emitted from the surface and the signal of the reflected waves is recorded. The seismic waves reflect from the subsurface structure in different ways depending on the structure's density, creating a seismic image. Once a potential field is located, exploration wells are drilled and log data are gathered. Well logs, i.e recording of the physical properties by measurement tools in the well bore, in addition with core samples, are used to further characterize the geological environment, and estimate the potential of hydrocarbons production. In case the oil field is considered to be economically profitable, a development strategy can be further determined and eventually the field can be taken into production. In the production phase, the domain experts develop strategies to extract the hydrocarbons such as well planning and fluid injection [18, 19, 22]. This process is illustrated in Figure 2.3.

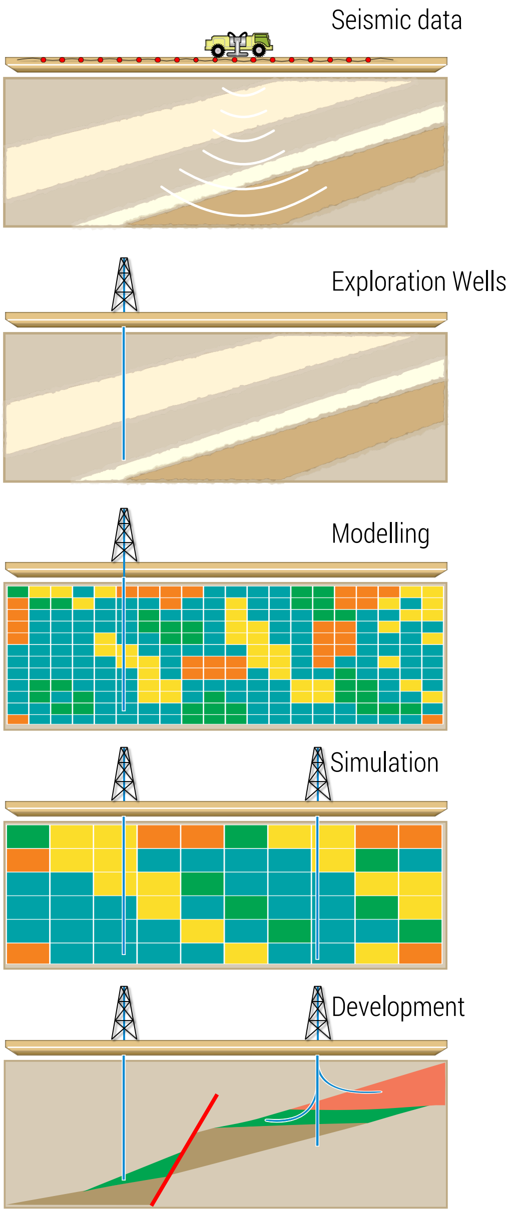


Figure 2.3: Stages of the exploration and production process.

2.4 Reservoir Management

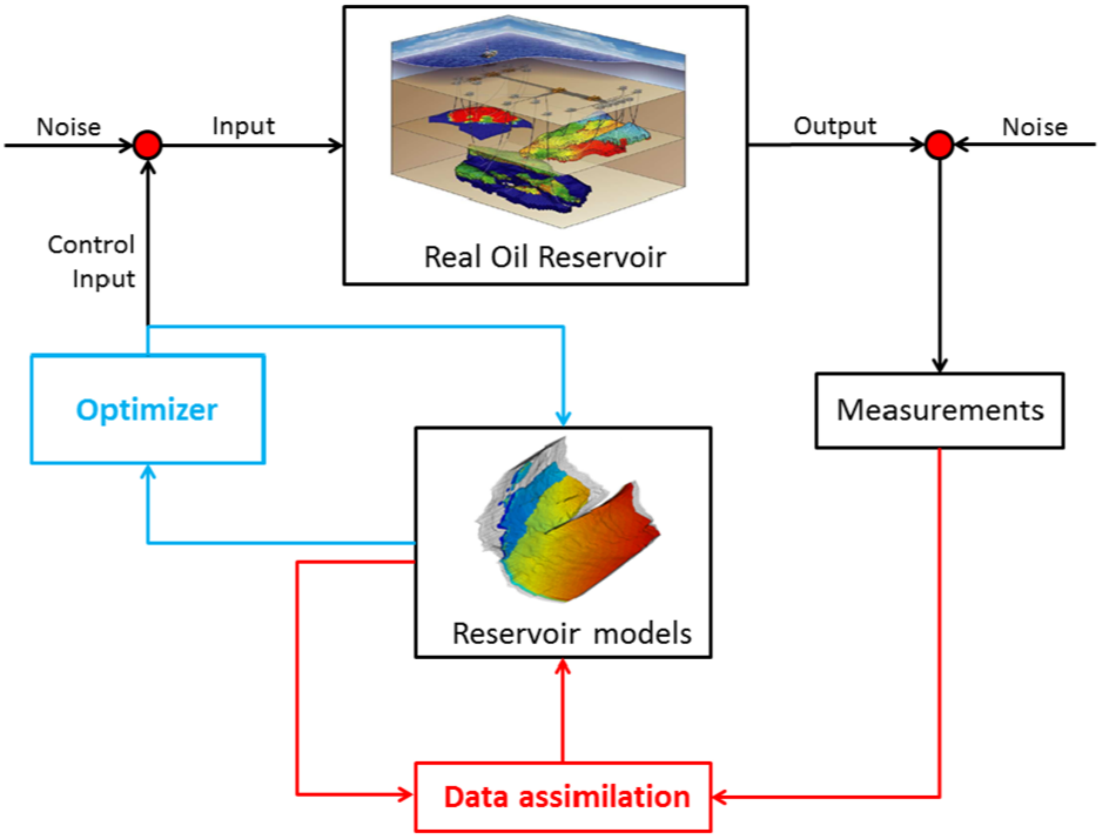


Figure 2.4: Closed-loop reservoir management as presented in JANSEN *et al.* [22]

Nowadays reservoir management is an integrated workflow that covers reservoir life-time from exploration to abandonment. The ultimate goal is to maximize the oil production or another economic objective and reduce the risk of failure, particularly in expensive high-risk E&P projects.

Various technologies used to understand a potential reservoir provide information at many different scales. Core plugs are a few inches in size. Well logs can detect properties within a few feet around the well. Seismic imaging covers a huge volume, but its typical resolution is limited to a few meters vertically and tens of kilometers horizontally. Collecting samples from the reservoir is a time consuming and costly task. Furthermore, it only provides a sparse representation of the subsurface environment. Therefore, geological interpretations based on seismic surveys and understanding of sedimentary processes are used to interpolate or extrapolate the measured data in order to yield complete reservoir descriptions.

Constructing reservoir models has become a crucial step in resource development as it provides a venue to integrate and reconcile all available data and geological concepts. These models represent the best guess as to what the reservoir looks like, in terms of its geometry, its intrinsic properties, and its fluid content. The standard

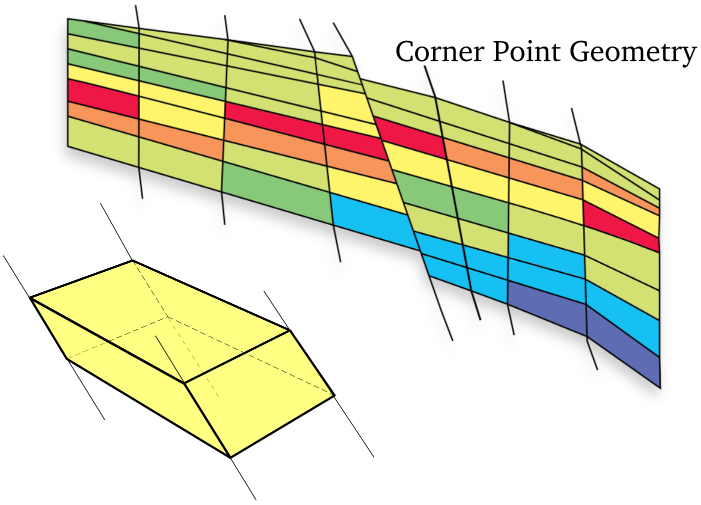


Figure 2.5: Cross section of a corner-point model. It is widely used by the petroleum industry as it can better adapt the grid to reservoir boundaries and some internal features (e.g., faults, horizons, wells, and flow pattern) [23] .

representation for oil reservoirs are **Corner-Point Grids**, which consist of irregular hexahedral cells arranged in a $3D$ grid. Each cell has static geological properties and fluid flow dynamic properties. Degenerated cells can be created during the modeling phase, and some may also completely disappear, introducing connections between cells that are not initially set as neighbors. Another feature of corner-point grids is that they easily allow for discontinuities across faces, allowing for the inclusion of fractures and faults in the model, as illustrated in Figure 2.5. Visualization of these models do not only support the role of understanding complex geometries and spatial data relationships, but it also can synergistically influence and subsequently improve the data sampling, transformation, and modeling methods used to gain insight of the subsurface environment.

A recent trend is to manage the production of a reservoir as a model-based control process, which is referred to as *closed loop reservoir management*, or also known as *real-time*, *reservoir management*, *e-fields* or *smart fields* [22, 24]. The concept is relatively simple. Improvements in sensors and hardware over the last decades have made it possible to perform a larger variety of measurements continuously in productive fields. The aim of the closed-loop concept is to incorporate all these data in models while respecting the different uncertainties associated. These models can then be used to optimize the production strategy.

The first loop, referred to as data assimilation or model updating step, consists of a continuous update of the models incorporating all data available. The second loop consists of optimizing the control strategy using the data assimilated models.

2.4.1 Data Assimilation

In reservoir engineering the process of data assimilation (the red loop in Figure 2.4) is often referred to as "*History Matching*" [25–27]. This name is explained by the objective of the process itself: obtain a model that fits historical observations. Data assimilation is widely used in fields like meteorology [28], oceanography [29] and groundwater flow [30]. This process is mainly concerned with the reconstruction of unknown quantities based on the available measurements in the presence of uncertainties (usually as noise). It allows to combine available observations with a given dynamic model. The information present in the measurements is combined with the information obtained when performing numerical simulations in order to produce more realistic results. One starts with an ensemble of prior models and updates them each time a new measurement becomes available. When this process ends, the model is simulated forward in time, thus, a future reservoir performance can be predicted and uncertainties can be estimated. At this point the model is used to forecast production and new development tasks explained in the following section.

2.4.2 Reservoir Optimization

When developing a field, the goal is often to maximize an economic criterion (e.g. oil and gas revenues minus field development costs). In the optimization loop (the blue loop in Figure 2.4), one tries to identify the optimal exploitation strategy, both through optimization of production controls in a given well configuration [31, 32], as well as through determining the optimal positions of new wells [33, 34]. As oil and gas is produced from the reservoir, new pressure and production data become available. This data can be used from the models updating step, to obtain a more accurate and reliable reservoir model.

When the data assimilation loop and the optimization loop are combined, they form a framework for model-based closed-loop reservoir management. Both loops are repeated sequentially as new data becomes available [35–37]. As illustrated in Figure 2.4, the reservoir model, represented as a corner-point grid, is the heart of field development from the beginning until its abandonment. Visualization is important during data assimilation to spot inconsistencies in the data extrapolation/interpolation, and when the model is used to simulate scenarios, like well planning and injections fluid.

2.5 Summary

In this chapter, we discussed the importance of Oil&Gas to the development society as well as, overview the Oil and Gas Industry, and where this thesis is inserted.

GOMES and ALVES [18] list the main questions that should be answered by reservoir engineers, and not surprisingly, almost all are influenced by geometrical information of the reservoir model. For instance, “Where should the wells be positioned to maximize oil recovery?”; it is a question that the engineers answer by correlating the behavior of the flow with the well’s position. To achieve these correlations the experts must have a good model that allows for flow simulations based on the rock properties, as well as analysis tools. In history matching, the reservoir engineer is generally forced to estimate parameters for the entire reservoir model only based on the information related to sparsely distributed production data [38]. To this end, visualization is of paramount importance during all development stages of an oil field.

Chapter 3

Literature Review

> *"Without libraries what have we? We have no past and no future."*
>
> —Ray Bradbury

Complex models arise in various domains, such as architecture, manufacturing industry, and medical imaging. The conception of even larger and more detailed models has brought challenges for visual comprehension and making inferences on the data. One important issue is occlusion, i.e., when important parts of the models are nested inside or hidden behind others. In this case, artists often make use of the cutaway technique, that removes less important parts of the object to reveal the most important ones. Researchers in computer graphics have been inspired by these technical illustrations and developed algorithms to automate the process of creating cutaway illustrations.

We have categorized three main areas where certain types of occlusion reduction methods were used, most of them with inspiration in cutaway illustrative technique. In Section 3.1, works on volume rendering are presented and in Section 3.1.1 a detailed description is given for the seminal work of VIOLA *et al.* [39]. Subsequently, in Section 3.2, we present some related works in polygon models. In general they use *Constructive Solid Geometry* (*CSG*) techniques to reduce occlusion, like the work proposed by LI *et al.* [40] detailed in Section 3.2.1. Among the works presented in Section 3.2, the work of BURNS and FINKELSTEIN [41], Section 3.2.2, is most related to our method, where a real-time cutaway rendering was develop to support dynamic scenes and animation. Finally, in Section 3.3, an overview of occlusion reduction methods to aid in visualization of geological models is presented. In particular, in Section 3.3.1, the work of LIDAL *et al.* [42] is presented. Their main contribution is a set of Design Principles to efficiently apply cutaway to geological models. Although they have used a model different from the one used in our work, some design principles still fit to our case study.

3.1 Cutaway in Volume Rendering

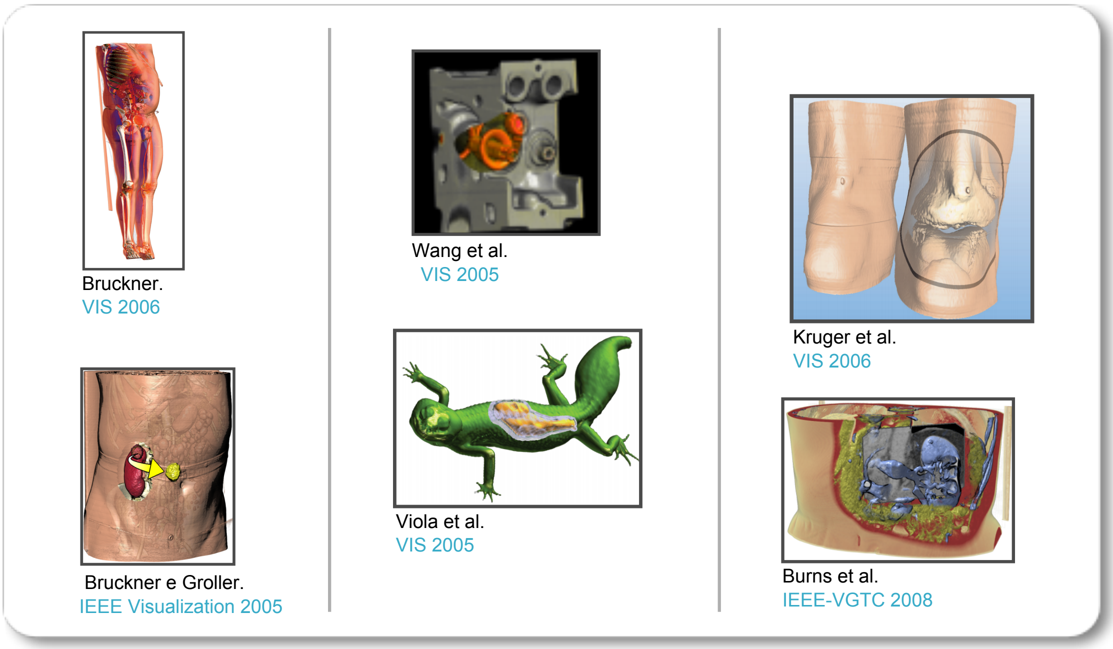


Figure 3.1: Some volume rendering works that make use of cutaways. BRUCKNER *et al.* [43], WANG *et al.* [44], BRUCKNER and GRÖLLER [45], VIOLA *et al.* [39], KRUGER *et al.* [46], BURNS *et al.* [47].

Volume rendering methods create visualizations by employing opacity functions to reveal the interior structures, or by extracting iso-surfaces. However, designing a suitable transfer function can be a challenging task. BRUCKNER *et al.* [48] describe a *focus+context* method to highlight regions of interest in volumetric models. They use the idea of lighting-driven feature classification, creating images that resemble artistic illustrations. In a previous work, BRUCKNER and GRÖLLER [45] proposed a discrimination of interior and exterior parts by preserving clear shape cues in order to maximize context. By the same token, WANG *et al.* [44] proposed a framework based on the Magic Lens metaphor [49]. They advocate the use of various types of lenses to magnify regions of interest. KRUGER *et al.* [46] implemented a similar system by combining several layers of the volumetric data in the hot spot region.

3.1.1 Importance Driven *Cutaway* on Volume Rendering

Traditionally, features within the volume dataset are classified by optical properties such as color and opacity. VIOLA *et al.* [39] use the approach of importance-driven visualization, where another dimension is assigned to features, describing their importance.

With this importance value, they developed a cutaway method for this type of data set that will be further described below.

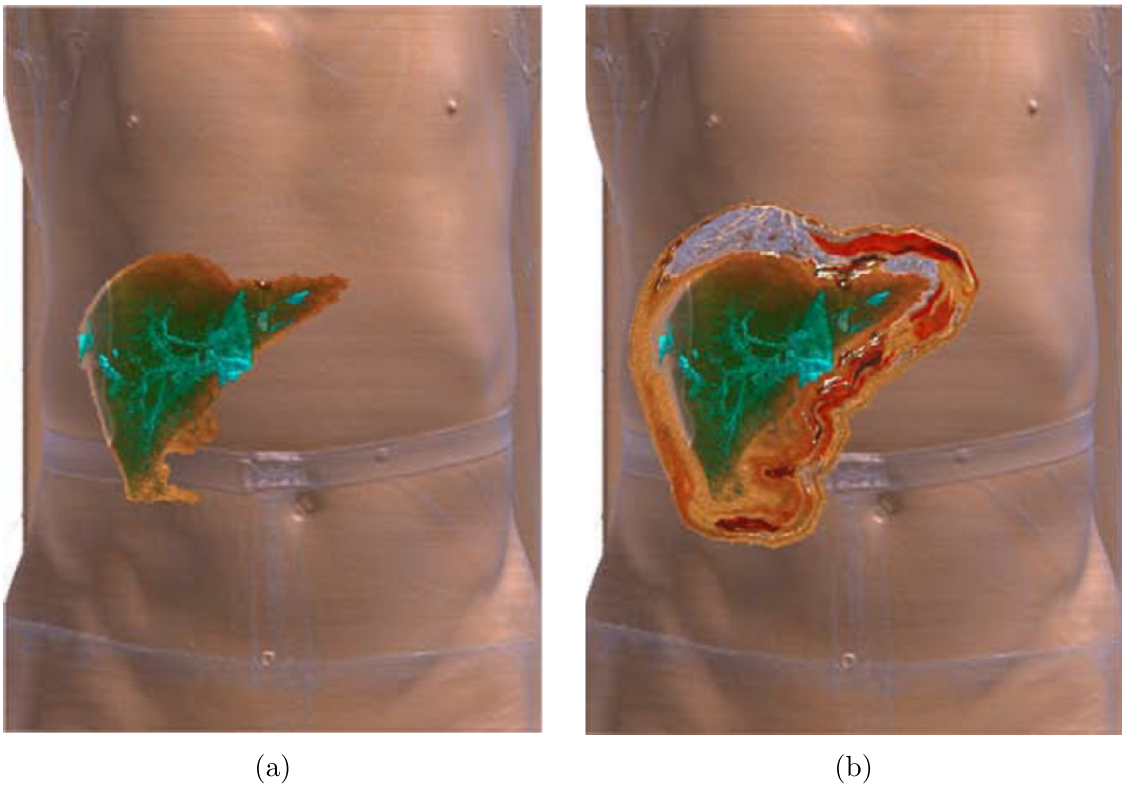

(a) (b)

Figure 3.2: Images resulting from the application of the method proposed in [39]. 3.2a cutaway applied with the initial buffer. 3.2b images generated by the buffer modified by Equation 3.1, exposing the surrounding layers.

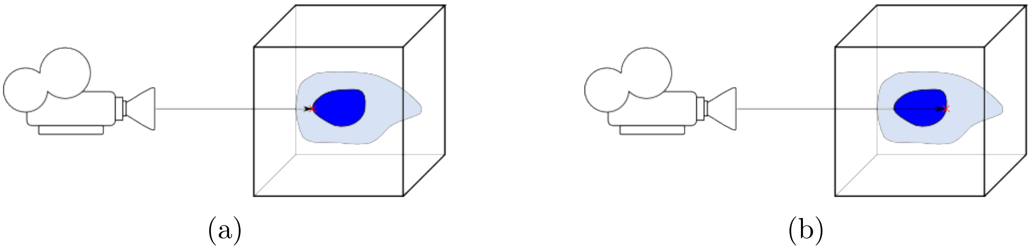
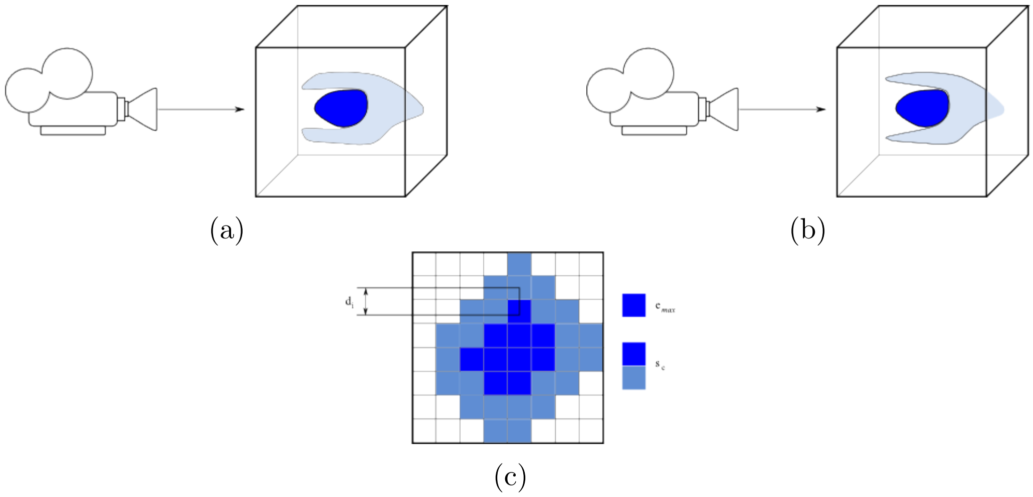

(a) (b)

Figure 3.3: Generating the initial buffer. First hit of the ray in 3.3a and the last hit 3.3b. In a second instant the buffer is recorded. Image adapted from [50].

3.1.1.1 Cutaway Generation

The algorithm is based on a depth buffer, which has depth values of the region with the highest relevance and is used to determine the level of opacity of a given pixel. This buffer is generated from the projected pixels of the object in focus. These pixels contain depth information, which is acquired by the intersection of the rays along the view vector, as illustrated in Figure 3.3b. The buffer is initialized with the rear hull of the object in focus ($e_{\max}$ in Equation 3.1).

With this initial buffer, the geometry of the cut volume is similar to a cylinder, as illustrated in Figures 3.2a and 3.4a. With a naive buffer it is possible to visualize the object of interest, but the layers that overlap the focus object are not visible.


(a)

(b)

(c)

Figure 3.4: The cone cut volume generation. 3.4a the initial cylindrical cut volume. The cone cut volume, generated by the modified buffer 3.4b. 3.4c we have the footprint of the modified buffer. In dark, the initial buffer and in light modified values. Image adapted from [50].

These layers are important because they provide context and the notion of spatial arrangement of the region of interest in relation to the components that surround it. To make these layers visible it is necessary to modify the initial buffer. The cut geometry creates the effect of a truncated cone, by applying a distance transform to the initial buffer. To this purpose they use the *Chamfer* [51] method detailed below.

The *Chamfer* [51] algorithm computes an approximation of the distance transform in $2D$. The algorithm scans the image in two steps: first step, from top to bottom and right to left, and in the second step in the reverse order.

For each pixel p, a small window with dimension $k \times k$ is used to find the smallest distance of p to a pixel already processed in the image. Operating sequentially for each pixel, the distances are propagated in the first step, forming a partial distance transform. In the second step, the values are updated, if necessary, using the reverse scan, forming the final buffer.

The values of the modified buffer e_i allow the visualization of the surrounding layers generated by the initial buffer $e_{\max}$ of the focus object, with an aperture angle with factor s_c, and distance d_i in image space from the pixel i to the nearest pixel with depth value of $e_{\max}$. This is defined by the equation below:

$$e_i = e_{\max} - \frac{d_i}{s_c} \tag{3.1}$$

The footprint of the modified buffer is illustrated in Figure 3.4c and the promoted effect is shown in Figures 3.4b and 3.2b.

3.1.2 Discussion

Even if volume rendering techniques are able to achieve high quality visualizations, there is one drawback that has led us to take a different route in our research. Basically, we would like to show a clear cut of the volume, and thus, avoid using transparency. The main reason is to emphasize the structure of the sliced volumetric cells, that is, the interface between the cutaway surface and the volume. As will be shown in Chapter 4, a well defined and clear cut is able to reveal internal structures of the reservoir's layers, which would most certainly be lost, or at least, not as evidenced, using volume rendering approaches.

3.2 Cutaway in Polygonal Rendering

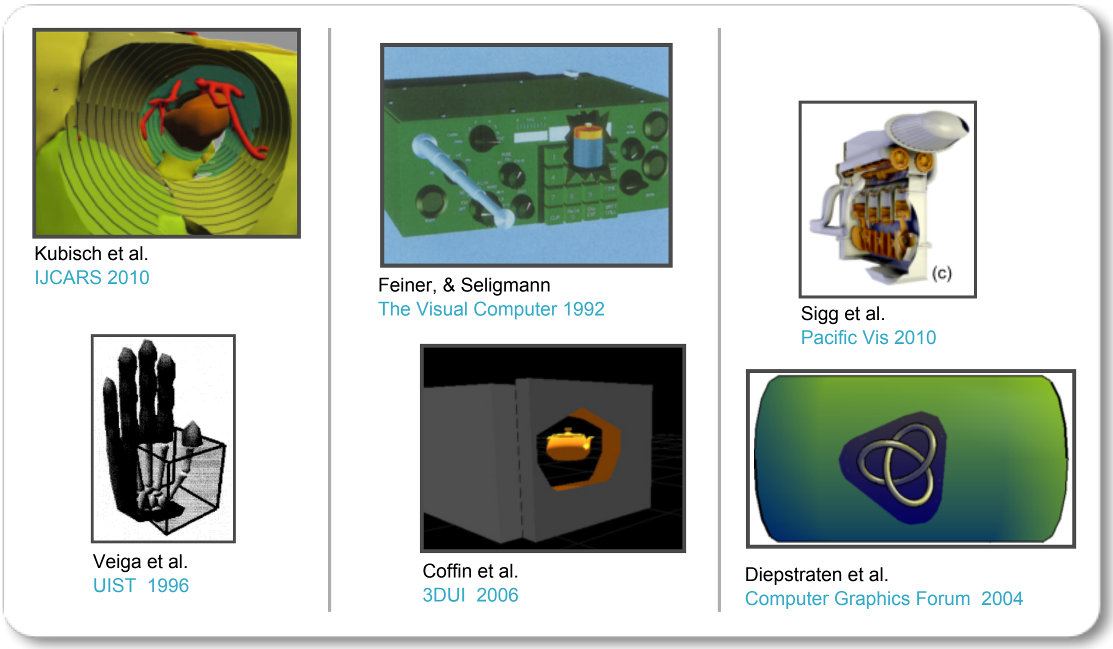


Figure 3.5: Cutaways on polygon models KUBISCH *et al.* [52], FEINER and SELIGMANN [53], SIGG *et al.* [54], VIEGA *et al.* [55], COFFIN and HOLLERER [56], DIEPSTRATEN *et al.* [57].

For cutaway illustrations in polygon models, two different approaches can be applied: constructive solid geometry (CSG)-based cutouts and stencil/depth buffer-based cutouts. In this section, we summarize some works on polygon model and go in further details in the work of LI *et al.* [40], in Section 3.2.1 and BURNS and FINKELSTEIN [41], Section 3.2.2.

Following the concept of Intent-Based Illustration Systems [58], FEINER and SELIGMANN [53] developed a system that makes use of image processing techniques to expose hidden objects. The system generates two masks: a *cutaway-mask* based on the z-buffer for clipping the models; and another called *edge-mask* to highlight the cutaway boundaries. Another early work that resembles the cutaway illustrations was proposed by VIEGA *et al.* [55], that extends to a $3D$ environment the metaphor of a see-through interface of the Magic Lenses proposed by BIER *et al.* [49].

DIEPSTRATEN *et al.* [57] classified cutaways in two sub classes: cutout illustrations and breakaway illustrations. For both cases, they presented several methods to create cutaways in polygonal models as well as a set of rules to apply the cuts. In order to enhance the experience in virtual environments, COFFIN and HOLLERER [56] proposed an interface to make arbitrary cuts on the occluding geometry. This method enables the ability to see through solid walls with a virtual x-ray vision, the analog for cutaways in virtual environments.

KUBISCH *et al.* [52] developed a system that applies a set of “Smart Visibility” techniques to help in planning a surgery. Their method uses explicit cut volumes to apply the cutaway technique on a specific organ. Based on the bounding sphere of the object of interest, a cone is generated in clip space to serve as the cut volume. The cone is rendered and the depth and illumination information is stored in an $RGBA$ buffer. This buffer is then used as the cut surface to clip the model appropriately.

SIGG *et al.* [54] proposed a fully automatic method for placing the cut volume and interactively specifying important features in a model. To achieve comprehensibility of the cutaway they suggest the use of simple cut geometries, such as cuboids, spheres, or cylinders. We have also followed this direction by using rectangular shapes for our cut volumes.

3.2.1 Semi-Automatic Cutaway Generation

In this section we discuss the work of LI *et al.* [40], who presented a formal survey of illustration conventions to reduce occlusion, as well as techniques to algorithmically apply them. In the case of cutaways, they developed an authoring system to help users to establish parameters, place the cut object, and set good views that mimic the artist’s choice for static illustration. While their technique reveals objects of interest, it is intended to produce a static view of the scene, thus providing a limited solution for exploring complex $3D$ dynamic and interactive scenarios. Their technique is based on two premises:

The cuts should respect the geometry of parts of interest. Based on the analysis of several books on scientific illustration ([60], [61], [62], [63]), they concluded that the most effective cuts are performed to partially remove occlusive parts allowing them to be mentally reconstructed. Thus, the location and shape of the cuts depend not only on the geometry of occlusive parts but also on the location of parts to be exposed.

Cutaway Illustrations should support interactive exploration. Static cutaway illustrations reveal limited information from the chosen point of view and not always clearly illustrate the structures of interest. Interactive controls can help improve comprehensibility of the relationship between the parts.

3.2.1.1 Traditional Cutaways Conventions

Traditional cutaway illustrations present a series of conventions that help emphasize the shape and position in spatial relations to the structures in a complex $3D$ model.

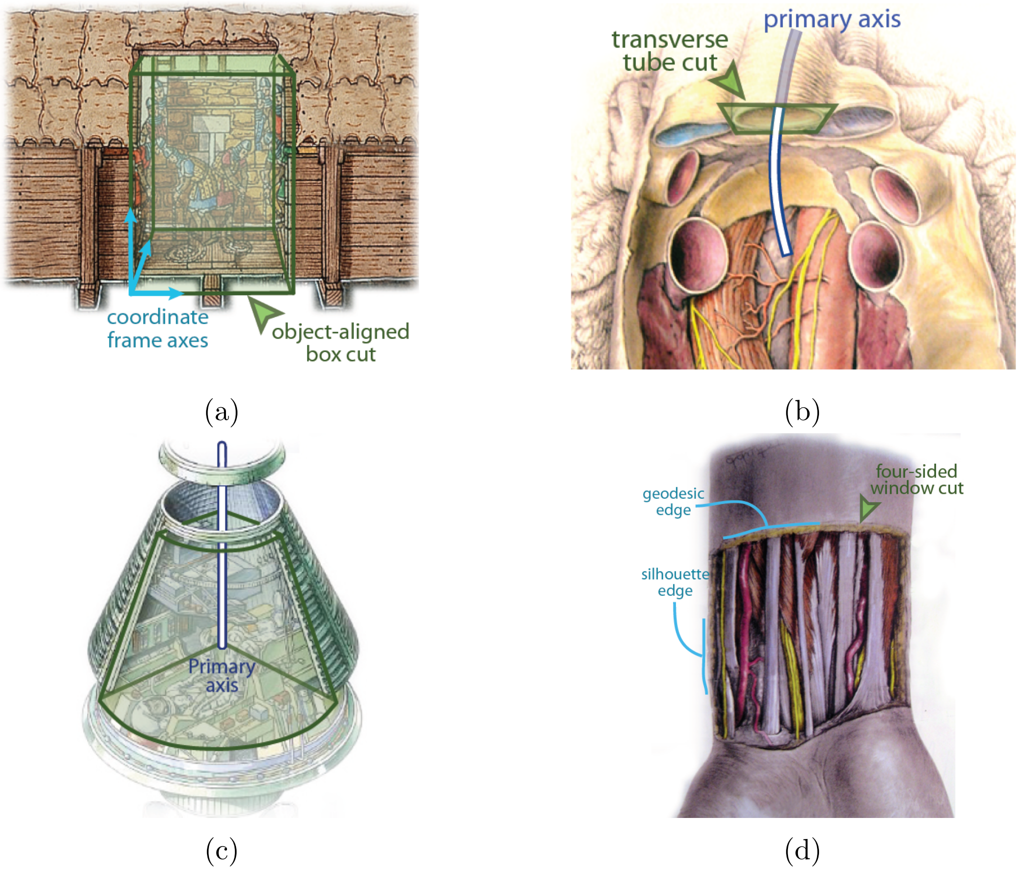


Figure 3.6: Geometric conventions for cut volumes. Illustrators choose cut volumes based on the geometry of the object in focus: 3.6a Object-aligned box cuts are used for rectangular objects; 3.6b Tube cuts for long cylindrical parts; 3.6c Wedge cuts for radially symmetric objects; 3.6d Window cuts for exposing parts that are surrounded by thin structures such as skin [59].

Besides these features, some rendering techniques are applied to enhance this relationship. To identify these conventions they examined technical manuals, anatomy atlas and illustration books of buildings and complex machinery, in addition they worked directly with the artists that produce these materials. The results are a few conventions listed below.

3.2.1.2 Geometry-based conventions

The geometry of the parts determines the most appropriate direction for the cut. Below, are cited the most common forms and conventions according to the authors.

Object-Aligned Box cuts Illustrators often use boxes aligned with the principal axes of the object. These objects are modeled respecting the 3 orthogonal axes, and in many cases resemble rectangular solids. Aligning a box cutter with the principal axes of the object helps to infer the shape of the geometry that will be removed, as illustrated in Figure 3.6a.

Tube Cuts. $3D$ models, especially for anatomy and some machinery parts, possess structures that resemble tubes, either because they exhibit radial symmetry (e.g. pipes and gears) or they are long and narrow (e.g. muscles and bones). Illustrators usually line up the cuts with the main axis of the object, removing a section of the structure using a cutting plane perpendicular to the main axis, as shown in Figure 3.6b. A second variation has a slice format (Figure 3.6c), removing less material than the previous one, and hence, more context. Furthermore, this variation better preserves the structure of the cylindrical tube. The slice-shaped cuts are typically used in radially symmetric (or almost symmetric) structures.

Window Cuts. Many $3D$ models are surrounded by thin structures (such as the skin or the body of a car) that occlude large parts of the internal structures. To expose these parts, illustrators often create a small window on the surrounding structure and remove the material in this area.

The edges defined by this window provide useful information about the shape of surrounding structures. Stylizations of these edges help emphasize the contours as described in the following section. Another convention, usually in technical illustrations, is drawing detailed edges to differentiate the edges of windows cut from others that belong to the scene.

3.2.1.3 Viewpoint Conventions

Illustrators carefully choose viewpoints that help the user to understand the spatial relationships between the part of interest and the occluder. Typically, the point of view not only centralizes the part of interest, but also minimizes the number of occlusive structures. This strategy allows exposing parts of interest with only a few slices, removing less of surrounding structures, and increasing the context of the scene.

3.2.1.4 Method Overview

Based on the two premises of Section 3.2.1 and the conventions of previous sections, LI *et al.* [40] developed a system for visualizing complex $3D$ models using cutaways. The system has two interfaces: the authoring interface allows the user to add information to the model for the construction of the cutaway; the viewing interface takes the model pre-processed by the previous interface and allows the user to explore the model with high-level cutting tools. The system input is a $3D$ model, where each part has its particular structure or geometry. The system cuts the model by removing material from a region, which they call *volume cutting* using CSG subtraction.

During the authoring process the system requires the user to specify a set of good viewpoints. When the model is examined using the visualization interface, the system selects between the points of view which provides the best visibility for the focus structure. The selection algorithm takes into account the layers between the occlusive structures encoded in an occlusion graph defined for each viewpoint, and a metric-based visibility in the exhibition area.

To render the model, the system uses *OpenCGS* [64], an open source library based on *GPU* acceleration algorithms. Furthermore, various rendering techniques for styling the cuts are used to improve depth perception and orientation of several layers.

3.2.2 Adaptive Cutaway for Comprehensible Rendering of Polygonal Scenes

As described in previous sections, in complex scenes with lots of overlapping objects, it is essential to preserve the context while focusing the visualization on a specific object, as illustrated in Figure 3.7. In this section we present the work of BURNS and FINKELSTEIN [41] for real-time cutaway rendering. One detail is that this technique requires a polygonal model with double walls to simulate thickness of the cut wall. The resulting renders are similar to technical illustrations and can be explored interactively.

3.2.2.1 Introduction

The cutaway techniques presented in this section was based on the techniques presented by VIOLA *et al.* [39] and LI *et al.* [40]. The core of the method is the generation of the depth image representation of the cutaway surface, as detailed in the subsequent sections.

3.2.2.2 *Cutaway* Function

In Section 3.1.1, we discussed how VIOLA *et al.* [39] used the *Chamfer* method to create a depth image of the cutaway surface. The algorithm renders the depth values of the back hull of the object of interest in a buffer, which is transferred to CPU where it is processed using the *Chamfer* method, and then transferred back to video memory. The *Chamfer* algorithm is a technique that approximates the Euclidean Distance Transform in $2D$ using L_2 norm. The method is applied in a binary image I and determines the minimum distance d for each pixel p to a seed pixel $q \in R$:

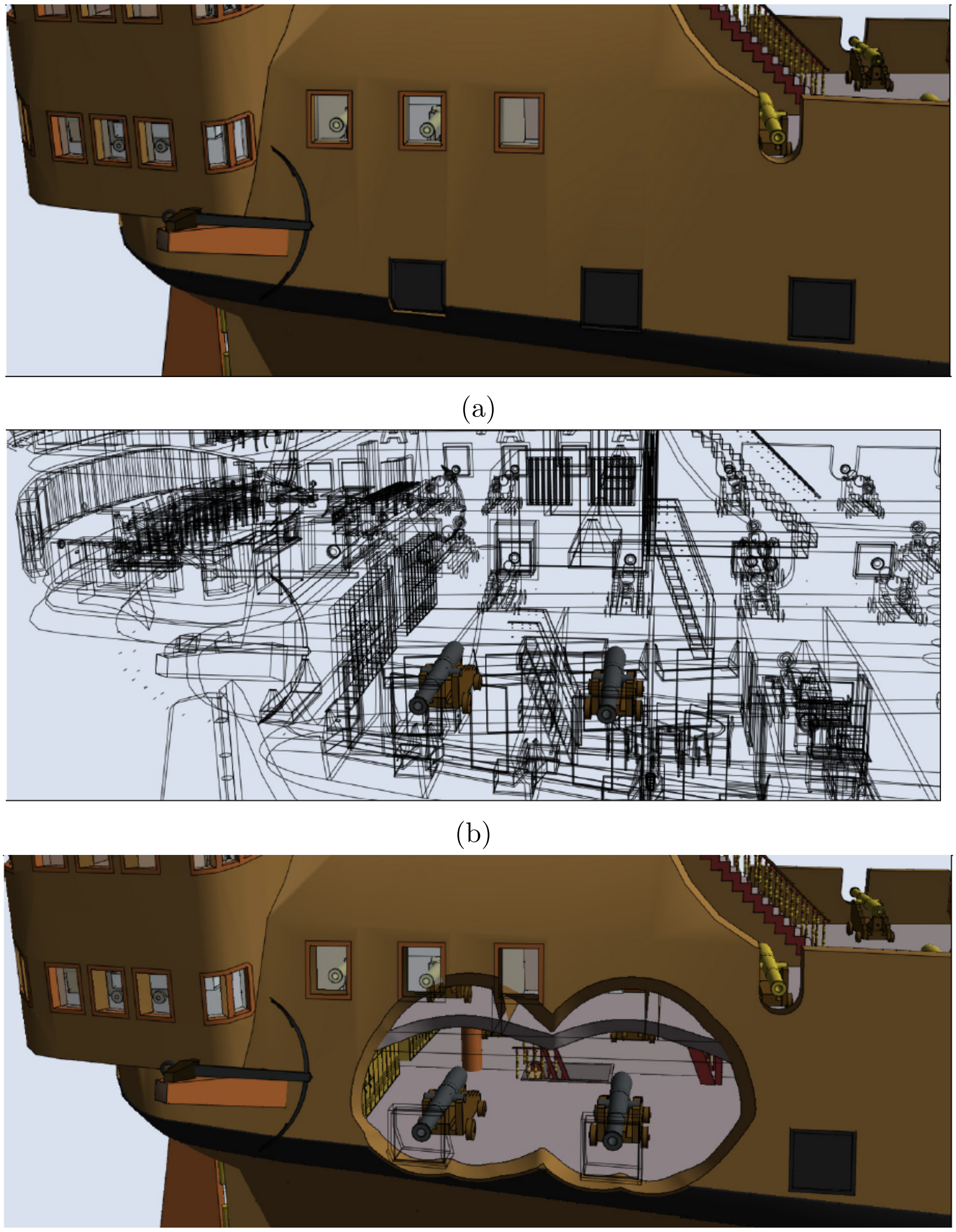
$$d(p) = \min_{q \in R} ||q - p|| \tag{3.2}$$

This minimization function can be defined alternatively as a function of maximization, where $d_{\max}$ is the maximum value of d, and $d(p) = d_{\max}$ when $p \in R$.

$$d(p) = \max_{q \in R} (d_{\max} - ||q - p||) \tag{3.3}$$

Initializing the image I with depth values $z(p)$ instead of $d_{\max}$, and using a scale value m for the *Chamfer* method, the function can be defined as follows:

$$d(p) = \max_{q \in I} (z(p) - m||q - p||) \tag{3.4}$$



(a)

(b)

(c)

Figure 3.7: Global and local application of occlusion reduction techniques. 3.7a the body of the ship occludes interior objects. 3.7b in order to expose the cannons, applying an overall occlusion reduction on the rest of the model makes it difficult to understand the context of the scene. 3.7c local occlusion reduction, the rest of the model is preserved, providing valuable spatial information to understand the scene [65].

3.2.2.3 Algorithms for the *Cutaway* Surface Computation

As mentioned in previous sections, the first step to compute an image space cutaway surface is to define the region of interest. The initial $q \in R$ is set by rendering the rear hull of the region of interest in a buffer with depth values. Each pixel q represents

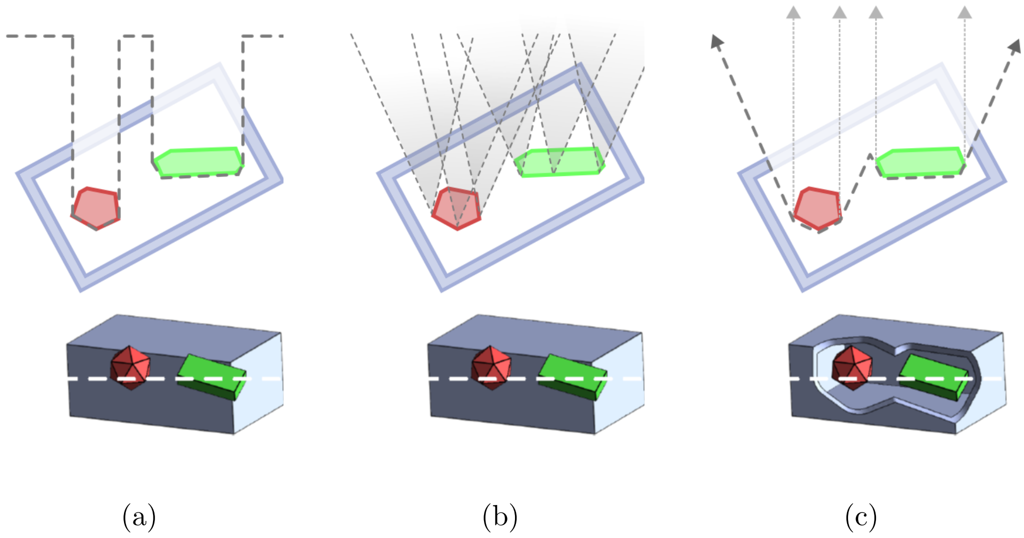


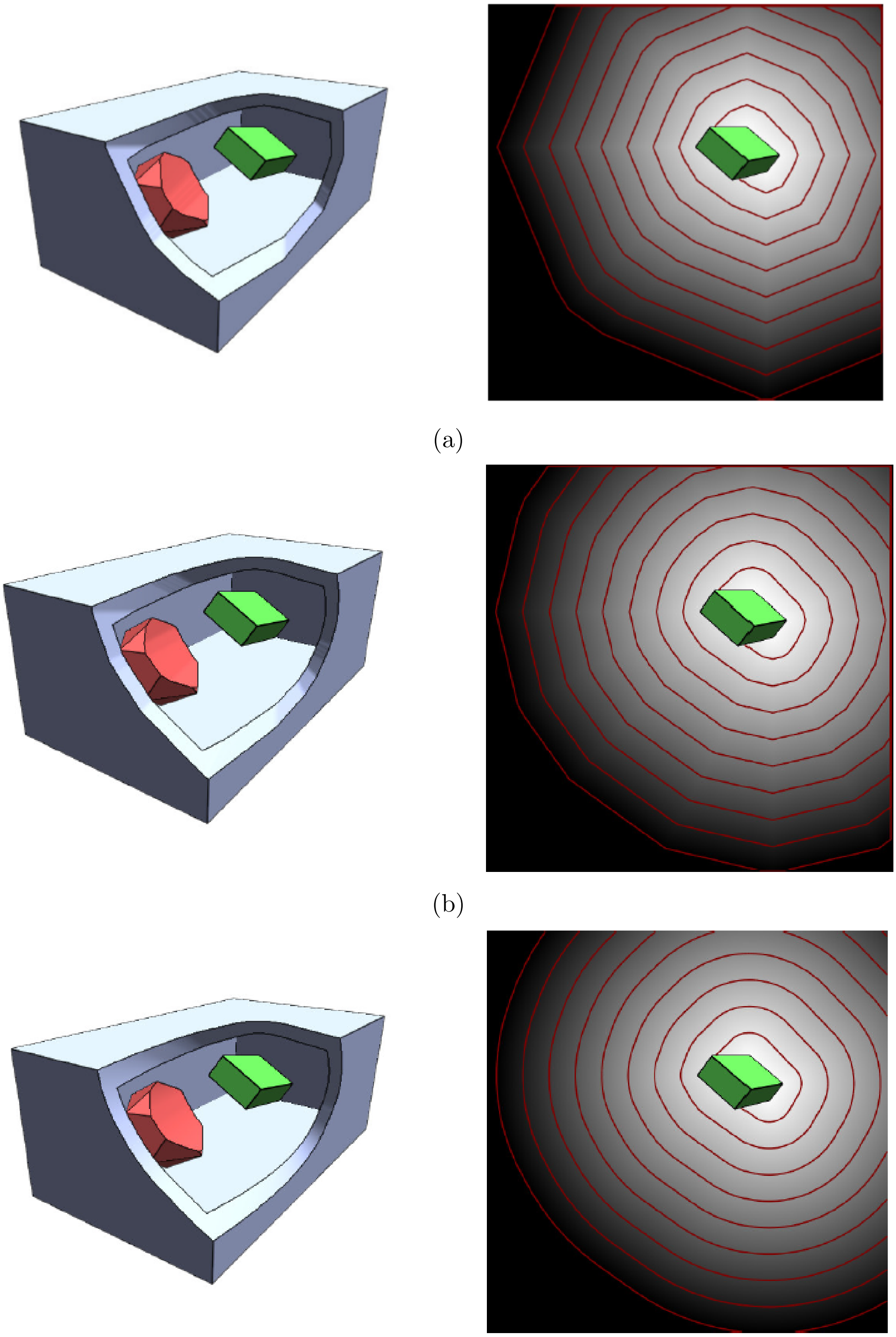
Figure 3.8: 3.8a buffer initialization with the back hull of the object of interest I; 3.8b visualization of the per pixel cone; 3.8c the modified buffer I, but with the cutaway surface defined with a given angle Θ [65].

an occlusion volume, defined by the cone with apex at the pixel and aligned with the viewing direction and aperture Θ. The cut volume is the union of all cones defined in R, and the cutaway surface for this volume is the rear hull of the union of these cones. This process is illustrated in Figure 3.8. Given a pixel p, its depth is calculated using Equation 3.4.

The authors list three ways to calculate the cutaway surface C in image space from the depth buffer: a brute force method, the *Chamfer* method, and a *GPU* based called *Jump Flooding* [66].

Brute Force. A basic method that offers an accurate result is to compute $c(p, q)$ for each pixel p from the seed pixels $q \in R$ and take the maximum of these values. For a buffer of size $n \times n$, this method requires $O(n^4)$ operations, making it less appreciated for real-time applications.

Chamfer A less expensive alternative is to use the previously described *Chamfer* algorithm. Although the *Chamfer* algorithm has complexity $O(n^2)$ for a buffer n x n, it is an approximate algorithm, and thus, there are possible artifacts in the final image, especially visible when changing the view direction (Figure 3.9). Another disadvantage is the sequential way the algorithm deals with pixels, making it less suitable for a *GPU* implementation.



(a)

(b)

(c)

Figure 3.9: Cutaway surface and the respective depth image representation for different kernel sizes of the *Chamfer* method. 3.9a for a kernel of 3 × 3, note that the surface is not smooth. 3.9b with a kernel 5 × 5 there is an improvement, however, it implies in a higher computational cost. 3.9c the exact solution [65].

Jump Flooding Information propagation is a very common task in many applications. For example, for the ”Paint Bucket” tool in an image processing software (i.e. Photoshop or GIMP), a point is chosen with the purpose of spreading its color information to all other points in a region (determined manually or by similarity to the starting point). In other words, we want to fill an area with a specific color, spreading from a seed to all others points. However, this is a naive way of filling regions, reminding the effect of a wave propagation, and generally being slow and inadequate for real-time applications. To illustrate the method above, suppose an $n \times n$ image containing 1 pixel located at the bottom left with some information we wish to propagate (see Figure 3.10). In each step, the information is passed pixel by pixel. At the end of $n-1$ steps, the entire image is filled. Using this method, the number of steps is linear regarding the image resolution. Analyzing the sequence of filling, we find that each shaded pixel is actually used once. In each step only the boundary pixels are required for propagation.

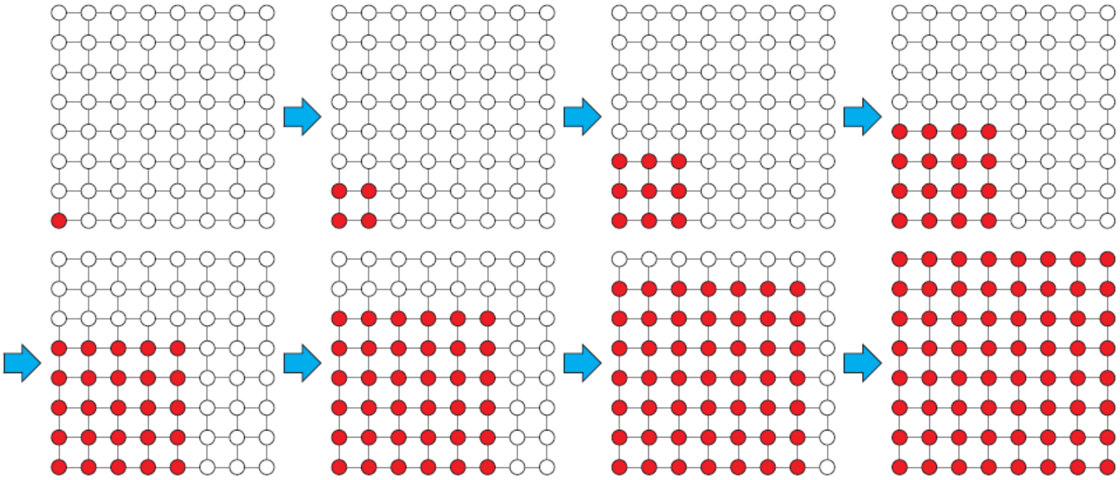
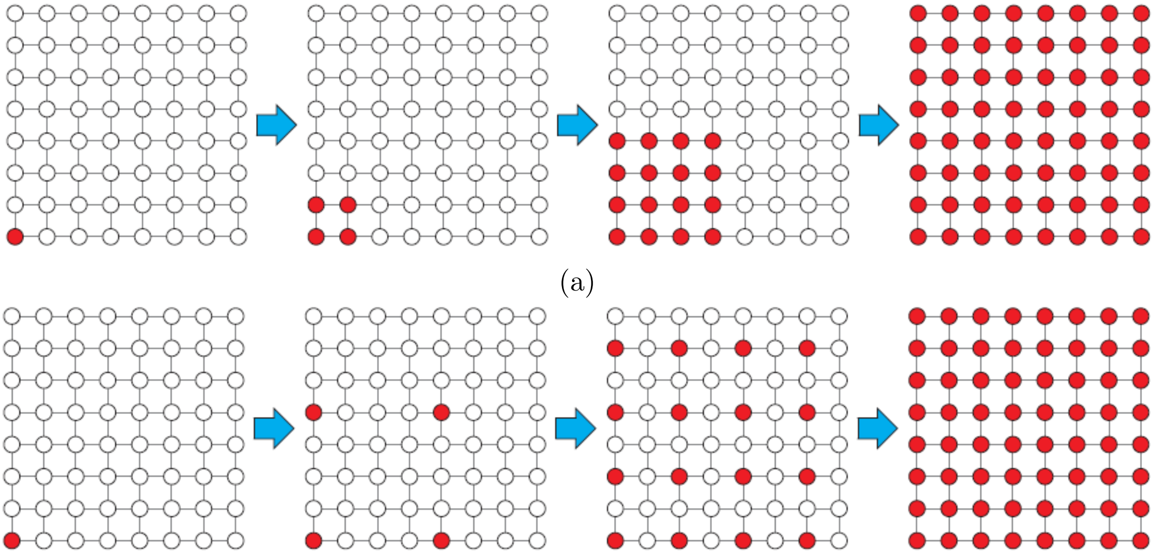


Figure 3.10: Naive approach to fill a regions. Only border points are used for information propagation. Image adapted from [67]

To make the process suitable for *GPU*, RONG [66] developed the algorithm called *Jump Flooding Algorithm* (JFA - fill through jumping). In the naive approach, a pixel (x, y) passes its information to (at most) eight neighboring position $(x+i, y+j)$, where $i, j \in \{-1, 0, 1\}$. This step is called “step size 1”.

A more efficient way is achieved by changing the step size in each iteration. There are two approaches: in 3.11a the initial step size is 1, and doubled in the next one; in 3.11b the initial step is large (usually $\lceil n/2 \rceil$ for an image with dimensions $n \times n$) and halved at each step until reaching the value 1. Both approaches have logarithmic complexity with respect to image resolution.

In an $n \times n$ image, the JFA method propagates information of several seeds at the same time. Without loss of generality, we assume n a power of 2 in the following discussion, so the JFA runs in $\log n$ steps. During the step with size k,



(a)

(b)

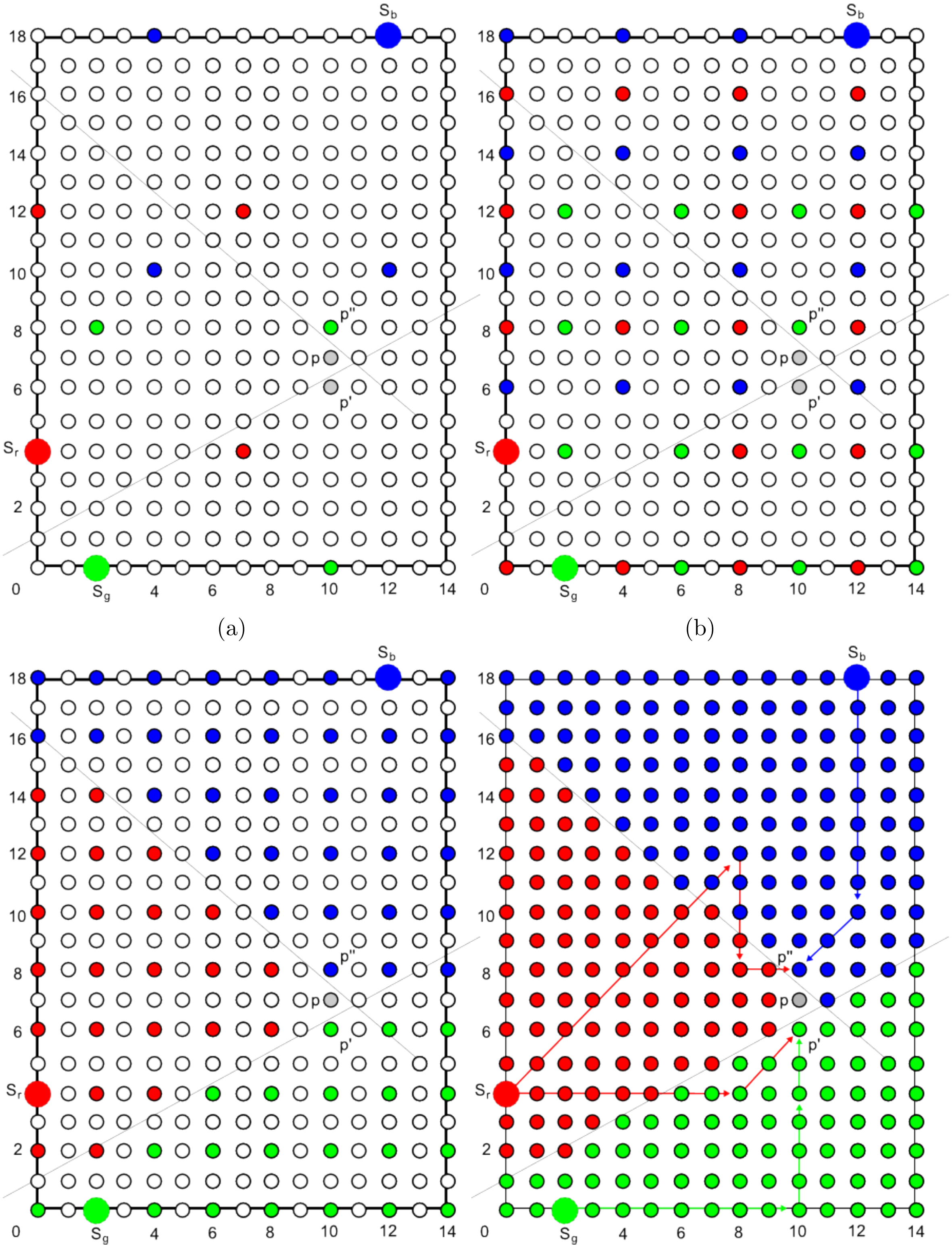
Figure 3.11: Two approaches that are more efficient than the standard propagation information. 3.11a doubling the step size. 3.11b halving the step size. Image adapted from [67].

each pixel (x, y) passes its information of seeds (if any) to other pixels in positions $(x+i, y+j)$, where, $i, j \in \{-k, 0, k\}$. With the information received and its own (if any), a criterion is used to decide which seed information should be maintained.

Thus, the best seed information is stored and propagated in the subsequent steps. In order for a pixel p to receive information of its best seed s, the information of s has to follow a sequence of steps $p1, p2, ...pk$, where p_i transfers its information to p_{i+1}, until reaching p.

Nonetheless the JFA does not always guarantee an exact result. Depending on the application of the algorithm and the criterion used for propagation, the result may contain errors, i.e. a pixel may not receive information from its ideal seed.

Figure 3.12 illustrates an example of error in the JFA method, where it is used for the computation of a Voronoi diagram. The seed contains its own position as initial information, and the ideal information for a given pixel is its closest seed. In Figure 3.12, the ideal seed of p is s_r, however, for this particular configuration the JFA failed to pass the information from seed s_r to p . At the end of the iterations, p can receive s_g or s_b. The reason is that for s_r to reach p, it must have a path passing through $p'(10.6)$ or $p''(10.8)$. But neither p' or p'' have selected s_r as their nearest seed, thus, this path does not exist. Nevertheless, in practice, the results are very close to the exact result, and only a small percentage of errors occur, making the JFA suitable for many real time applications.



(a)

(b)

(c)

(d)

Figure 3.12: 3.12d Note that for the step size 1, the point p can only receive information provided by points p' and p'', which lack the seed information s_r. Thereby, for this configuration, the JFA will have a small error, however, it does not cause major disruptions for many applications. Image adapted from [67].

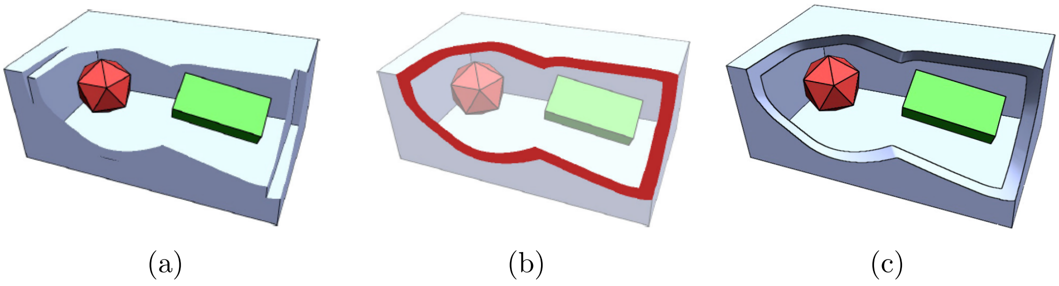


Figure 3.13: 3.13a applying a cutaway exposes the objects of interested and the inner walls of the box. 3.13b fragments coming from the “back-facing” triangles, provide a solid appearance to the box. 3.13c these fragments are painted using the cutaway surface normals and lines are used to highlight the edges of the cut [65].

3.2.2.4 Rendering the Walls

As mentioned in Section 3.2.2.1, the technique requires a polygonal model with double walls to render the cut walls. For scenes composed of “closed” models, a cut in the model always exhibits its interior. To provide the appearance of a cut, the fragments from the back-facing triangles are rendered using the cutaway surface normals. As illustrated in Figure 3.13b, only “visible” back-facing triangles appear in the place where the surface is cut. The rendering of the cut surface enhances the idea of solid models being carved along the cutaway surface, and assists in the realization of the layers of occlusion.

3.2.2.5 Contour Lines

To create non-photorealistic renderings, which mimic artistic techniques, the polygonal models rendering is incremented with contour of lines. These lines serve the same purpose as lines in artistic design, to convey shape and objects detail. In cutaway illustrations, the edge surface that delimits the occlusion region, form a contour providing the features mentioned above. The contours are identified as the fragments neighboring the “back-facing” triangles from the previous section. Figure 3.13c illustrates how these pixels corresponding to the edge of the cutaway surface can help the perception of the cut location and shape.

3.2.3 Discussion

Even though we are able to gather a few guidelines from these previous methods, none alone is suitable for corner point models. Most methods work with surface models, where there exists a clear distinction between different objects. In our case, our grid has no obvious prior separation since our set of primary cells is dynamic, i.e., depends on the range of selected attributes. Furthermore, even though each cell has to be handled individually, we need to, at the same time, keep the idea of a global context.

3.3 Cutaway in Geological Models

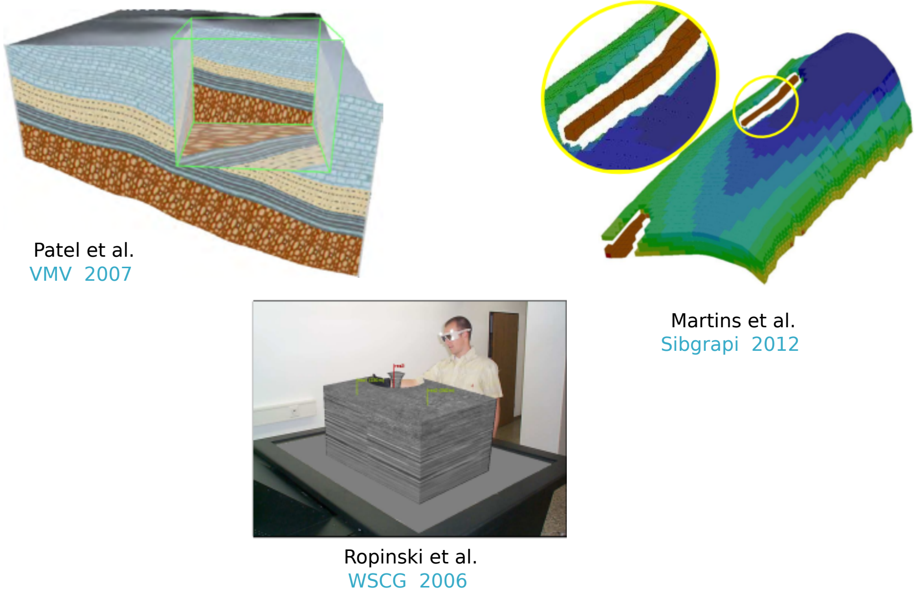


Figure 3.14: Cutaways on geological models PATEL *et al.* [68], MARTINS *et al.* [69], ROPINSKI *et al.* [70].

In the literature, there are very limited examples of illustrative techniques applied to geological models, and even fewer that use cutaways.

In ROPINSKI *et al.* [70], a method is presented to provide interactive exploration of seismic data sets using volume rendering based on two specialized transfers functions. One is used to render the volume of the region of interest defined by the *lens volume* (box or sphere), and the other to render the part external to the cut geometry. They also extend a set of immersion techniques to apply their approach to virtual reality environments.

PATEL *et al.* [68] extended the previous method by using a similar approach in combination with a $2D$ texture transfer function to produce illustrative rendering of interpreted seismic volume data. They took inspiration from geology illustration books. Both methods focus on visualizing important geological features for seismic data sets (e.x. faults and horizons).

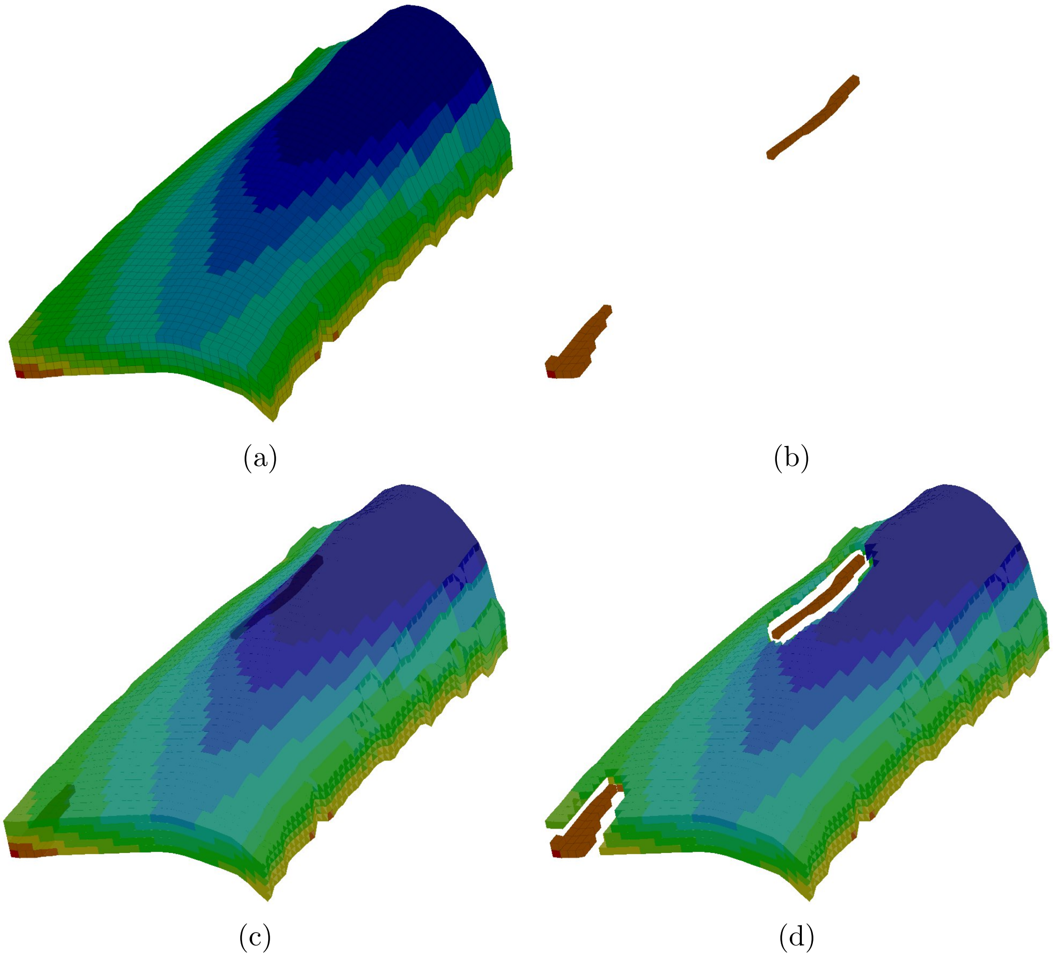


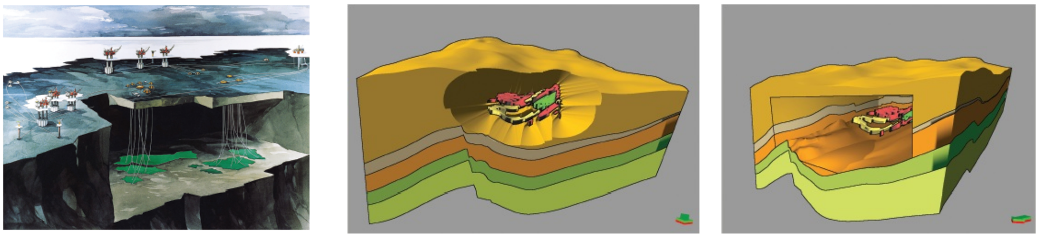
Figure 3.15: Visualization of primary cells. In 3.15a we have the model itself and in 3.15b some cells are selected to be visualized. In Figure 3.15c transparency is applied to the secondary cells and in 3.15d the ray cast procedure is depicted.

The work proposed by [42] and [69] are the most related to our domain of study. LIDAL *et al.* [42], in a similar investigation as [40], but in a specific context, comprised a study with domain experts and geological illustrators to come up with a series of design principles to effectively apply cutaways to geological models. Their approach aims at solving the problem of depth and shape perception to emphasize features inside the geological models. Unlike [52], [39] and [47], their cut geometry is decoupled from the camera, and is defined in model space using a two render pass strategy. The first pass extracts the bounding rectangle from the depth footprint of the objects in focus. This rectangle serves as the back plane of the proxy geometry generated in a second pass. We take a similar path, but we generate the cutaway surface representation in one render pass, since we explicitly represent the cut volume. Another notable difference from our method, is that their input is much simpler since they work with a set of surfaces representing geological layers. Corner-points, on the other hand, impose a further challenge to visually separate the primary and secondary objects.

The method proposed by MARTINS *et al.* [69] uses a combination of cutaway and transparency to resolve occlusion problems. Their approach for cutaway is a simple procedure that casts rays from the center of the camera to the centroid of the primary cells. Secondaries cells that are intersected by these rays are eliminated. Although they resolve the occlusion problem, the discrete elimination criterion conservatively eliminates secondary cells, as shown in Figure 3.15.

The choice between binary and continuous was also taken in account during our research. In our case, we clip secondary cells in a continuous and tighter manner. Again, this is very related to the previous discussion about the advantages of not using volumetric rendering approaches. A clear and continuous cut saves and evidences valuable context information in the neighborhood of the cells in focus.

3.3.1 Design Principles for Cutaway Visualization of Geological Models



(a) (b) (c)

Figure 3.16: 3.16a Illustration manually produced by a graphic designer. 3.16b Using the technique proposed by BURNS and FINKELSTEIN [41] applied to geophysical models where the focus cells apparently are placed in the yellow layer. 3.16c Cutaway technique proposed by LIDAL *et al.* [42], elucidating that the cells are actually in the orange layer [42].

In a recent work, LIDAL *et al.* [42] made a study where traditional cutaway methods (specially the method proposed by BURNS and FINKELSTEIN [41], Section 3.2.2) were applied to geological models, and observed that it had a low degree of depth perception, as illustrated in Figure 3.16b. This characteristic leads to a series of difficulties in understanding the model and accomplishing the tasks listed below:

- understand the spatial order of objects in focus;
- measure relative distances between objects in focus and the edges of the cutaway surface, in addition, the position of objects in focus within the cutaway;
- understand in which layer the object in focus resides;
- depict the shape and topology of the object in focus;
- understand the shape of the cut on the model.

In order to understand why the traditional cutaway methods are not adequate for geological models, they cite the work of TURNER [71], where he describes that in these models there are a number of articles or units he named *geo-objects*. Turner argues that these *geo-objects* have characteristics that represent a major challenge for modeling and visualization. Among many, he cites the complex geometry and topology (e.g. containing discontinuities to represent faults or very narrow layers) as main features that make current visualization methods fail.

To improve the visualization, LIDAL *et al.* [42] were inspired by geophysical illustrations, Vision and Perception sciences, as well as discussions with geologists and illustrators. Thus, they proposed a series of design principles for visualization of geophysical and other models that share the same characteristics.

3.3.2 Design Principle

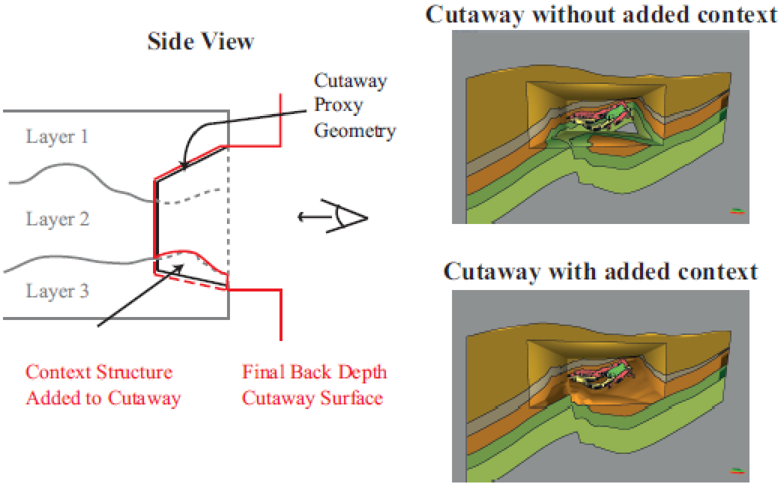

Side View
Cutaway Proxy Geometry
Layer 1
Layer 2
Layer 3
Context Structure Added to Cutaway
Final Back Depth Cutaway Surface
Cutaway without added context
Cutaway with added context

Figure 3.17: Adding structural parts from the context. Here the layer where the cells belong is kept [42].

Principle 1: Use an oblique view to visualize cutaways. Analyzing cutaway illustrations such as Figure 3.16a, they have observed that illustrators rarely use a frontal view, but one with a certain inclination. This oblique view exposes the walls of the occlusion volume aiding the perception of depth.

Principle 2: Simple cut volumes for complex models Due to the complexity of geological models, illustrators perform cutaways using simple cuts. Generally, rectangular boxes or something similar to the cut volumes proposed by LI *et al.* [40] (Section 3.2.1.2).

Principle 3: Include familiar context Artifacts related to object and context are expressed by keeping some parts of the latter intact, even though, in some cases, they occlude the object of interest. In Figure 3.17 part of the layer where the object of interest resides is maintained to provide context.

Principle 4: Use suitable lighting model Illustrators use some rendering techniques to highlight features in the model. One is to apply shadows to emphasize notions of distance and depth.

Principle 5: Use parallax Parallax is the perception of relative distances by observing the scene from different viewpoints. Although, in general, they have studied static graphics, animations and interactivity are mechanisms that can increase depth and shape understanding.

3.3.3 Implementing the Principles

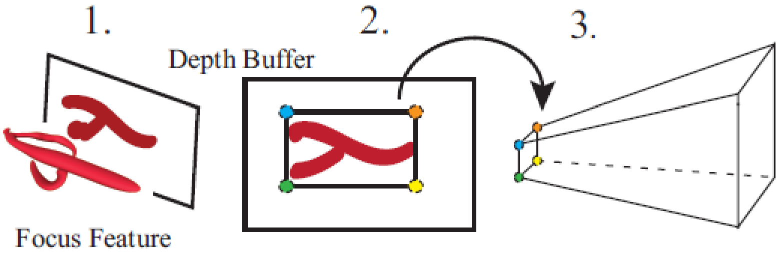
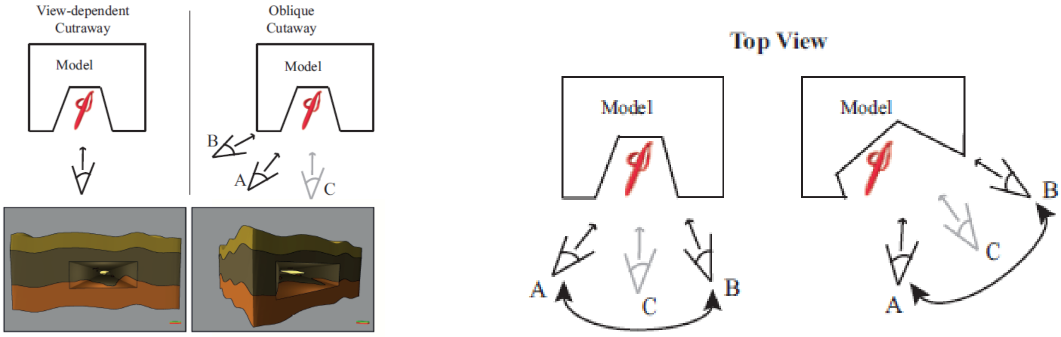


Figure 3.18: Cut volume generation. Differently from 3.2.2, this technique has more control over the cut volume [42].

LIDAL *et al.* [42] developed a cutaway visualization technique based on the principles listed in the previous section. The input is a set of polygon objects, and the solution is based on one *proxy geometry* that has the shape of a truncated pyramid defined by 8 points in model space. This will serve as the cut volume and, consequently, the cutaway surface (Figure 3.18). The following are the steps necessary to generate the *proxy geometry*.

1. Render the object of interest in a buffer, see (1) in Figure 3.18.
2. Extract the bounding box of the footprint from the buffer, see (2) in Figure 3.18.
3. Transform the vertices of the bounding box to view space.
4. Define the pyramid's frontal plane near the camera.
5. A constant is used to determine the opening angle of the truncated pyramid.
6. After computing the 8 vertices of the pyramid, they are transformed into model space, decoupling the cut volume from the camera.

With the cut volume decoupled from the camera, they can achieve Principle 1 by using a camera view to generate the geometry of the cut volume and another for rendering, as shown in Figure 3.19. The simple geometry of the cut volumes



(a)

(b)

Figure 3.19: 3.19a left: front view of the region of interest; 3.19a right: the same cut volume, but with another point of view [42].

meets the requirements of Principle 2. Principle 3 is carried out by maintaining the base layer where the region of interest resides, as shown in Figure 3.17. To satisfy Principle 4, some effects are used on the model to emphasize the form, perception of relative distance and depth.

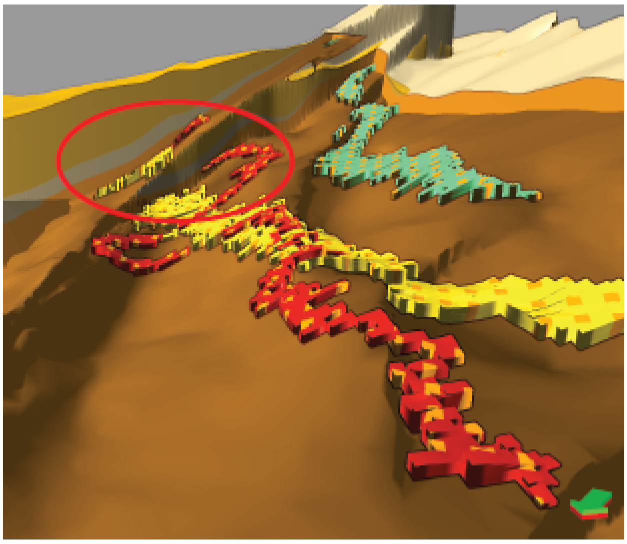


Figure 3.20: Close-up view where some rendering styles are used. In particular shadows and lines stripes on the context [42].

The parallax effect implies that when there are moving objects near the camera, they seem to move faster than distant objects. This difference in angular velocity may be used to convey distance relationships between objects. For this end, it is necessary to compare the speed of the objects in focus with some fixed reference. With the cut volume generated previously this task can be easily achieve. However, this dissociation of the camera and the object in focus, in some cases, can cause total occlusion of the object in focus if the rotation angle is very large, as shown in Figure 3.19a. For such, a limit is set for the rotating camera that updates the cut volume when it reaches a specific threshold, as shown in Figure 3.19b. As illustrated in Figure 3.20, they use dark stripes where there is a discontinuity between the objects in focus, as well as, some shadow.

3.4 Sumary

The cutaway technique proposed by LI *et al.* [40] is strongly influenced by traditional illustration, coming from the analysis of scientific illustration books, as well as interviews with professional that produce these materials. Their technique, although interactive, is restricted to some views chosen during the authoring phase, providing limited exploration. The parts of the model have to be assigned with an appropriated cut volume in a pre-processing stage, making this solution prohibitive to our needs, since the segmentation of our model in study is not pre-defined. The works of VIOLA *et al.* [39] and BURNS and FINKELSTEIN [41], proposed image based cutaway techniques that allow revealing the focus features based only on their geometry. Their techniques allow a more automatic manner to create cutaway views by only modifying the initial depth buffer of the focus feature to reveal not only the focus features, but the intermediate layers providing context during the visualization. The cutaway solution of VIOLA *et al.* [39] targets volumetric models. Volume rendering naturally renders interior material of solid objects, that have been exposed by the cutaway, so no special care is necessary to render the cut surface and the interior of the model. On the other hand, the context frontier is somewhat fuzzy due to the transparency effect.

In the case of polygon models, when applying cutaway, its interior looks ”empty”, and have to be reconstructed. BURNS and FINKELSTEIN [41] overcome this issue by using a two pass rendering procedure, with an extra pass to reconstruct the guide lines. LIDAL *et al.* [42] follow the methodology used by LI *et al.* [40] and defined a series of design principles to effectively apply cutaways to geological models. Following LI *et al.* [40], their work is intended to produce static views. They also propose a set of rendering styles to enhance depth perception and illumination models to communicate shape and spatial order of focus features. Other previous works in cutaways, in general, have created explicit cutaway surface geometry using *CSG* and stencil buffer techniques. These works were limited by the graphic hardware of their time.

Chapter 4

Cutaway for Corner Point Models

> *"Simplicity is only achieved with enormous effort"*
>
> — Clarice Lispector (A Hora Da Estrela)

Oil&Gas reservoirs are entities located deep in the earth crust. A reservoir engineer's job is to understand and predict what cannot be seen or touched. To this end, a number of indirect tools and measurements are required, involving considerable uncertainty.

Gather information about the reservoir is a costly and time consuming task. A reservoir model is build to converge all the available data and extrapolate over a $3D$ geological model, in order to characterize the subsurface environment to be used in numerical simulators. It guides the planing and the production of the field in its whole lifetime.

The standard representation for oil reservoirs is corner-point grids, and there are in the order of millions corner-point models employed by the industry nowadays [10]. In this chapter we detail our method for generating cutaway visualizations of corner-point models, to help domain experts to gain insight about internal structures with a simple and interactive solution.

4.1 Introduction

Features visualization in a corner-point model aims to show information about a set of one or more *cells of interest*, which we also call the object in focus, or the set of *primary cells*. These cells may be surrounded by other cells, known as *cells of secondary interest*. The cells of interest are the focus of the visualization, while the secondaries exist to provide spatial and functional context.

This scenario suggests a binary notion of importance, where cells of interest are assigned high importance and all others cells with low importance. A good visualization should emphasize and expose the cells of interest and, at the same time, omit as few as possible the secondaries cells, to provide valuable information about the importance of the primary cells during the inspection. Cutaway techniques are often used for conveying *in situ* spatial relationships between parts in focus from the context in complex $3D$ models.

The method presented in this chapter is based on the depth buffer generation of the cutaway surface, and is closely related to the approach described in Section 3.2.2 and 3.3.1, but with more control over the cutaway shape and a more flexible cutaway structure. The previous applications of cutaway are mostly viewpoint-dependent, i.e., the shape and location of the cut is directly dependent on the viewpoint information. We also follow this direction, but we relax it, allowing others angles of view for the cutaway visualization, hence improving depth perception of focus features (Section 4.3.4). Furthermore, our input model is much more complex than most representations treated by previous approaches, thus requiring special attention in many points.

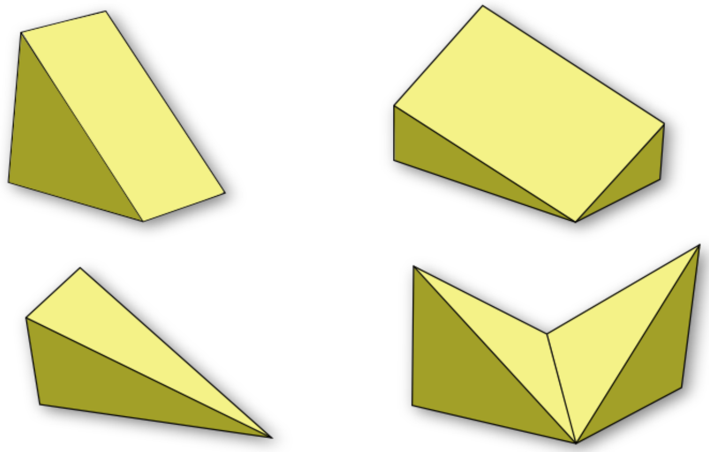


Figure 4.1: Example of deformed cell arising in corner-point models

4.1.1 The Model

The input model, consists in an up-scaled corner point model as described in Section 2.4. Each cell represents a portion of the earth's subsurface, where the rock's representative attributes (static attributes) and fluid contents (dynamic attributes) are assigned to each cell. In order to fit the geological features, corner-point models allow for degenerate cells as shown in Figure 4.1. For this reason, we use their bounding boxes in the cutaway process. More details in Section 4.3.

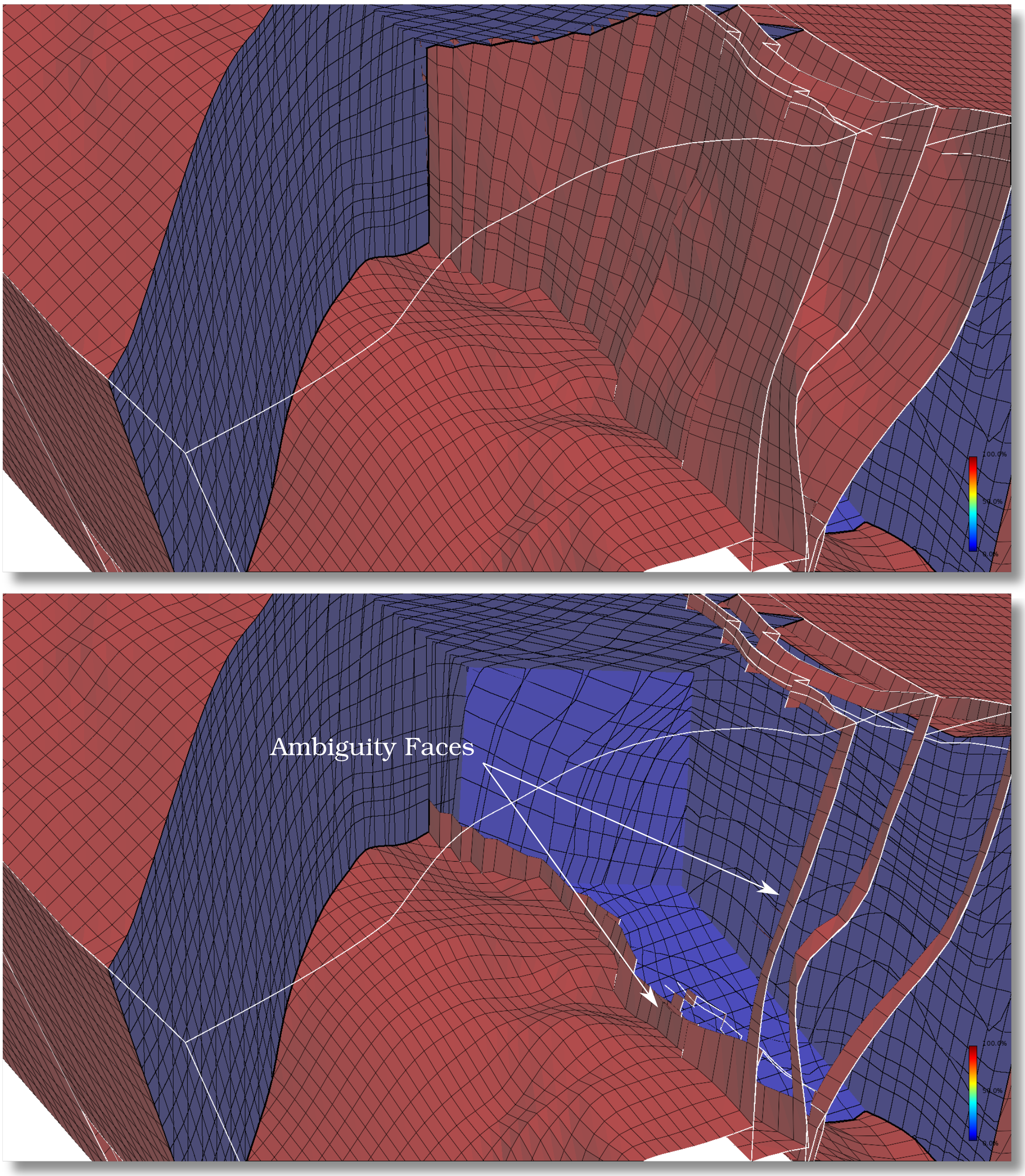


Figure 4.2: In blue, interior faces, and in red faces that belong to the shell and fault. Some faces were wrongly classified like those on the edge of the faults.

4.1.2 Pre-processing

To help render the cutaways, we classify cell's faces into two groups: faces that are internal to the model (interior faces); and the faces on the surface's boundary (shell faces), see Figure 4.3. The interior faces we also classify as faults or not faults. Since we are working with posprocessed models, we lose the discrimination of the important features, which in the modeling phase were used by the domain expert to create the representation that resembles the complex geological structures of the reservoir. We have tried to reverse engineer them in order to extract back these features, but in some models, where the geometry/topology are very complex, it is not possible to classify all faces accurately, as shown in Figure 4.2.

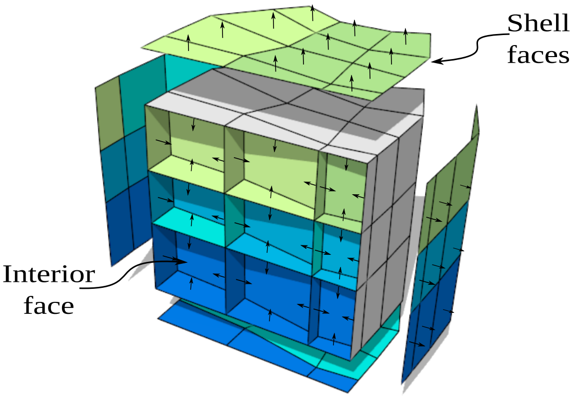
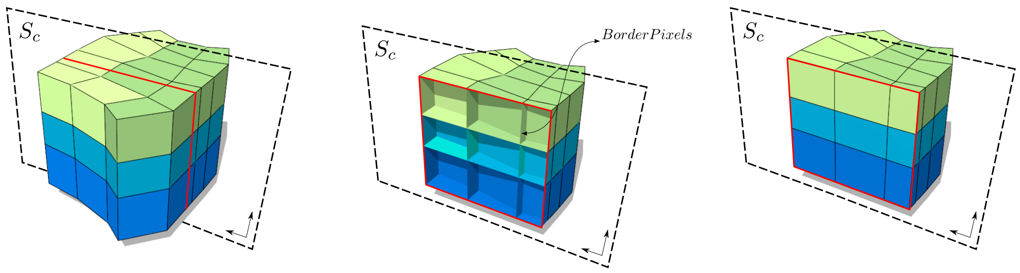


Figure 4.3: Faces classification.

4.2 Basic Cutaway Structure



(a) Secondary cells intersected by the cutaway surface S_c.

(b) Clipping fragments in front of S_c.

(c) Reconstructing the cells and its respective lines.

Figure 4.4: Secondaries cells intersected by the cutaway surface S_c (4.4a). Eliminating the fragments in front of S_c creates a hollow impression (4.4b). Rendering the cells and its respective lines appropriately to give the impression of solid elements being sliced (4.4c).

Our models consist of a set of cells defining a volume that represents the subsurface reservoir. The discrimination of what are the focus features depends on the chosen range values for the visualization. In other words, the focus features can be any cell or set of cells. In this scenario, the cutaway has to adapt in order to follow the dynamic segmentation. When the cut is not defined along the view direction, no geometry for its interior exist, therefore cells appear "empty", as illustrated in Figure 4.4b. Generating new triangle surfaces on the fly, in order to fill these gaps and conveying the appearance of something being cut, may not be practical for real time rendering. One may suggest the use of Constructive Solid Geometry libraries [64, 72, 73], among others, to realize the cuts. The basic cutaway structured can be more clearly defined using a CSG paradigm. CSG is a geometric approach that combines simple geometry primitives to form more complicated shapes in 3 dimensional space. The primitives must be solid with interior and exterior well defined. For example, a sphere and cube are solid primitives, whereas a triangle is not. CSG primitives are volumetrically combined into more complicated shapes by boolean algebra operators. The operators are *union*, *intersection* and *difference*. A general cutaway approach may be defined by employing CGS subtraction operations on the model. The part to be subtracted is the actual *cut volume*.

In practice, the subtraction operator can be realized in image space, since there are only two regions separated by a cutaway surface. The occlusion test is based on which region the fragment lies, and can be summarized when its depth value is greater or less than the edge of the cutaway surface. This classification can be made by comparing the depth of the fragment, p_z with the value $C(p)$ of the cutaway surface. The occlusion function is defined as follows:

$$\Omega(p) = \begin{cases} 0 & \text{if} \quad p_z \geq C(p) \\ 1 & \text{if} \quad p_z < C(p) \end{cases} \tag{4.1}$$

This particular definition is binary, but the occlusion function can be redefined for intermediate values between 0 and 1, as realized in [39], where the value between this range is used to assign a degree of transparency to the object in question. In our work, we will use only the binary approach. In practice, a cutaway structure when used with simple occlusion reduction exhibits the following important characteristics:

- interior objects should be visible from any given viewing angle. In other words, portions of secondary objects that occlude the region of interest have to be detected, and discarded;
- The amount of material discarded can be controlled by the angle Θ of the cut volume as desired, providing a balance between the amount of material removed and the ability to locate objects within nested structures;
- Multiple regions of interest can be defined as a union of cutaway surfaces, with the resulting cutaway surface exposing all objects of interest simultaneously;
- The shape of the cutaway approximates the silhouettes of the region of interest,

reducing the amount of material that has to be removed.

4.2.1 Object of Interest

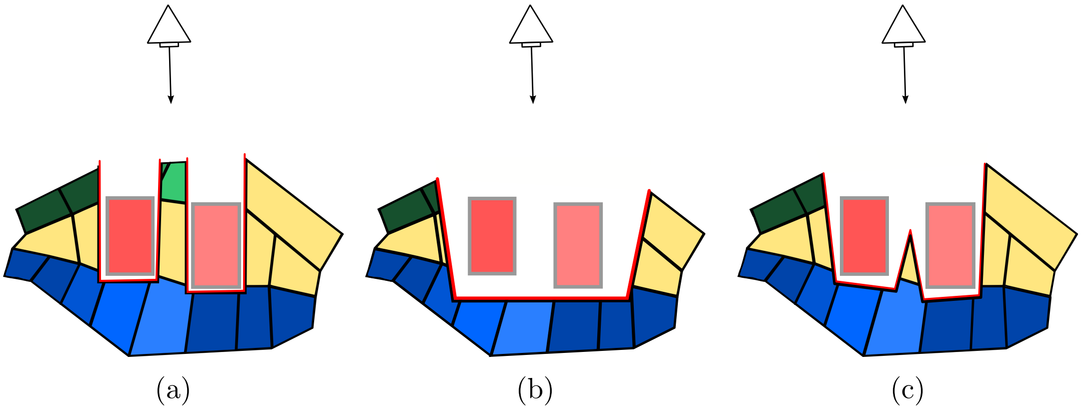


Figure 4.5: Definitions of region of interested. In 4.5a, object geometry based, 4.5b proxy geometry of the union of the objects of interest and union of proxy geometries of individual objects 4.5c.

At the core of cutaway illustrations, lies the definition of the shape and extension of the region of interest, being crucial for the effectiveness of the exposure. Depending on the application requirements, and given a set of objects of interest, there are several manners for constructing the region of interest, as described bellow.

Object Geometry The region of space occupied by the object of interest is the most elementary definition of the region of interest, as shown in red in Figure 4.5. Its simplicity emerges from the fact that the object's geometry is readily available, and can be directly used to construct the cutaway surface, since any full exposure of the region of interest guarantees complete exposure of the object of interest. This is the minimal definition of region of interest, and results in the minimal amount of material to be removed. In this way, only the object of interest is exposed, while nearby objects or materials may remain occluded (Figure 4.5a).

Proxy Geometry When visualizing a collection of objects of interest, the region of interest can be expanded from the minimal definition to include all the space around and between the objects of interest. This can be accomplished by computing a proxy geometry from the union of the objects of interest, as illustrated in Figure 4.5b, and using it as the basis for the cutaway surface computation. However, if objects of interest are far apart, the proxy geometry can be quite large, resulting in a large amount of material being removed, hence providing less context in the visualization. One may suggest the use of the convex hull, however, the convex hull can generate a smooth cutaway shape that creates ambiguity in certain visualization scenarios. As pointed in Section 4.1.1, we will use the bounding box of the cells of interest as the proxy geometry and their union to feed the cutaway surface computation (Figure 4.5c), since it provides a good balance between exposition and amount of secondary material to be removed.

4.3 The Method

As explained in the previous section, our target is corner point grids, where each element is defined by a cuboid with possible irregular shape. The objects of interest are cells themselves and the primary/secondary discrimination is dependent on the range of chosen values for a specific attribute.

To reduce ambiguity during visualization, we shade the wireframe lines of the primary and secondary cells with different colors. Since the attribute's color code is an important inspection feature, we try to respect it as much as possible, i.e., we avoid recoloring the cells even though it would provide a better contrast between primaries and secondaries. As our focus object is a set of small irregular cells, we construct our cut volumes in model space, similar to [40, 42, 54]. However, we build a cut geometry for each primary cell, and unify them based on the depth footprint of their rasterization. The main reason is that we have no prior knowledge of the location of primary cells, ex. they might be tightly packed or highly dispersed inside the model.

In the following subsections we describe in details our method to generate cutaway renderings of $3D$ reservoir models. Our approach is performed in three main stages:

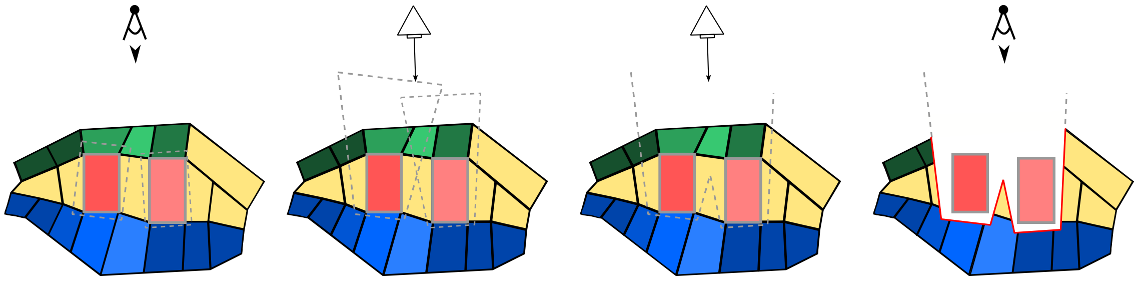
1. First, we generate the cut surface and represent it as a depth image: Section 4.3.1.

2. Second, we render the secondary cells and continuously clip them against the cut surface, rendering the remaining ones: Sections 4.3.2 , 4.3.3 and 4.3.4.

3. Finally, we render the primary cells in a very straightforward manner.

4.3.1 Cutaway Surface

To achieve interactive frame rates, we employ a modification on the rendering pipeline to create the cut volumes and the unified cut surface in a single render pass.



(a) Initial Bounding Box (b) Proxy geometry generation (c) Cutaway surface generation (d) Cutaway surface clipping

Figure 4.6: Proxy Geometry Generation: 4.6a first the bounding box of objects of interest are computed; 4.6b the bounding boxes are then transformed into frustums; 4.6c the frustums are rendered with an inverted depth test to register their union; 4.6d the rest of the model (everything that is not in focus) is clipped against the depth image of the cut surface.

We start with the bounding box of the cells in focus, and reshape them into frustums facing the camera. Each frustum defines a cut volume for a single primary cell (see Figures 4.6a and 4.6b).

For a single primary cell the depth image generation is trivial. The bounding box is transformed into a frustum with a given aperture angle and rendered into a depth buffer. This buffer is our screen space representation of the cutaway (cut surface), and is used to remove occluding geometry. This is trivially achieved using a depth test, i.e., during the rasterization of the secondary cells each generated fragment's depth is tested against the value in the depth image.

When we have a set of primary cells, it is possible that their cut volumes overlap, as shown in Figure 4.6b. A straightforward way to treat the overlap is by inverting the default depth test when rendering the frustum in order to keep only the rear hull of the proxy geometry. This simple modification allows the union of any number of cut volumes, as shown in Figure 4.6c. The accumulated depth buffer of all rasterized frustums completely defines the cut surface for our rendering purposes. A $2D$ simplification of the complete process is illustrated in Figure 4.6.

Following, we briefly describe these sequence of steps inside a shader pipeline perspective:

Vertex Shader Extract the bounding box and reshape it into a frustum facing the camera.

Geometry Shader Receive the 8 points defining the frustum and generate its faces for rasterization.

Fragment Shader Save the rasterized frustum into an $RGBA$ texture, where the RGB channel stores the normal of the cut surface and the alpha channel stores the depth information. We also store in a separate buffer the model space coordinates of the surface, since they will be necessary for the SSAO algorithm described further on.

4.3.2 Clipping

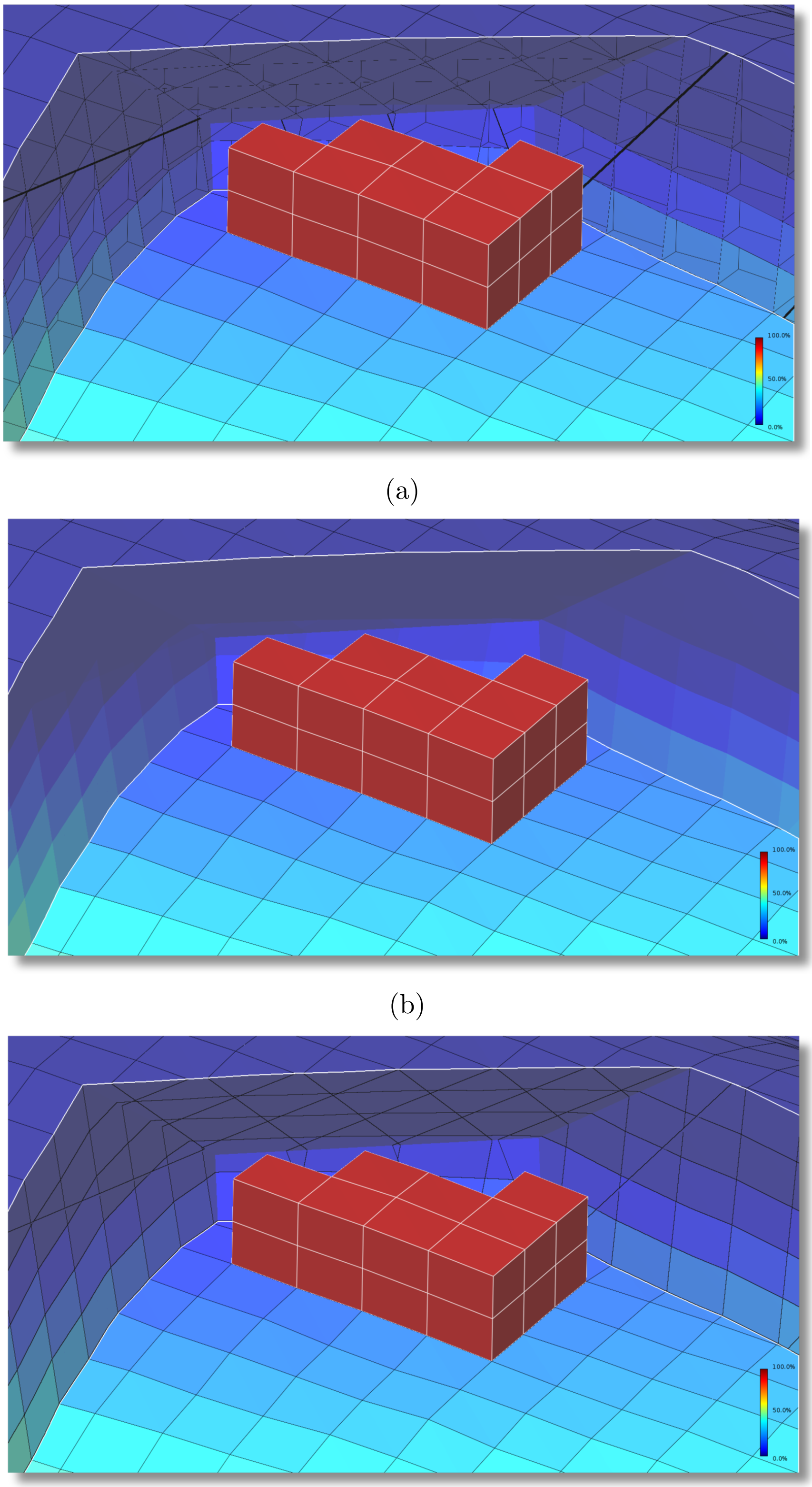
As aforementioned, given a depth image representation of the cut surface, clipping of the secondary cells is realized in a second render pass with a simple depth test. By simply eliminating fragments with depth in front of the cut surface, a hollow geometry is rendered, as shown in Figure 4.7a. However, secondary cells should be clipped in a continuous manner, i.e., a cell can be partially clipped giving the appearance of a solid object being cut. To achieve this visual impression, the normals of the fragments of the interior faces are set as the normals of the cut surface in the corresponding pixel. Note that the cut surface's normals were stored during the first render pass (Section 4.3.1). The whole process is illustrated in Figure 4.7.

4.3.3 Rendering Lines

Wireframe rendering is commonly realized in two render passes. The first renders the filled triangles, and the second renders the lines, using previous generated depth buffer to remove hidden lines. This procedure not only involves passing the geometry twice to the graphics card, but creates some artifacts (also known as z-fight) on the final image due to the small difference between the depth buffer of the rasterized lines and triangles. We use an approach proposed by [74] to render the model with its respective wireframe in one render pass. The main idea is to compute the distances from fragments to triangle edges, as illustrated in Figure 4.8. If a fragment is within a threshold distance (half the line width) from a triangle edge, the fragment is rendered with the line color, otherwise it is rendered with the triangle color. A smoothing function I_p is applied at the boundary between triangle and line to remedy aliasing artifacts.

However, since we do not render the cut surface explicitly, the lines representing the boundaries between the cut surface and the clipped cells are not known a prior, and have to be determined during rendering. Thus, we have to detect the border pixels representing where a cell face is clipped by the cut surface.

A simple approach to detect the border pixels is to use a threshold between the depth image of the cutaway and the depth of the fragment. The idea is to identify fragments that were not clipped, but are very close to the cut surface. Unfortunately, although some lines are rendered correctly, others are rendered with erroneous thickness (Figure 4.9a), mainly due to discretization issues and to the fact that the depth test has limited precision [75]. To avoid this undesirable effect



(a)

(b)

(c)

Figure 4.7: Naive cut reveals primary cells, but hollow geometry is rendered for secondary cells 4.7a. In 4.7b, we apply a naive solution, also used by [41], to proper render the secondary cells. In 4.7c, we reconstruct the lines in the same rendering pass using our ray-casting procedure.

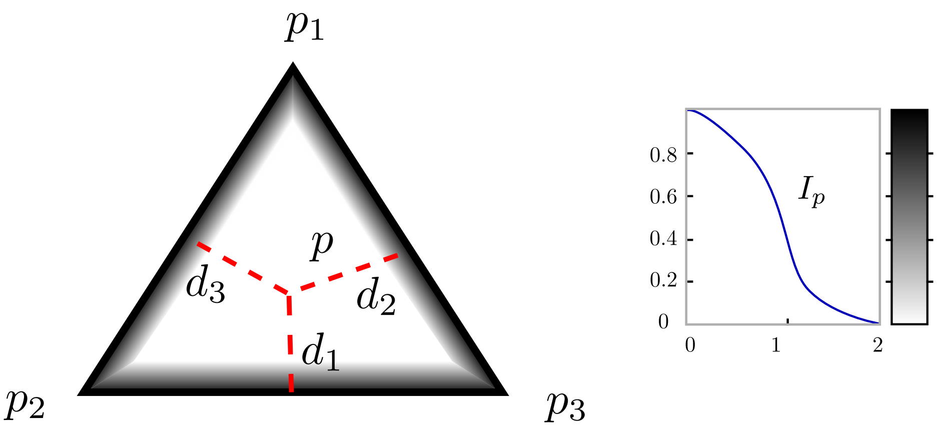
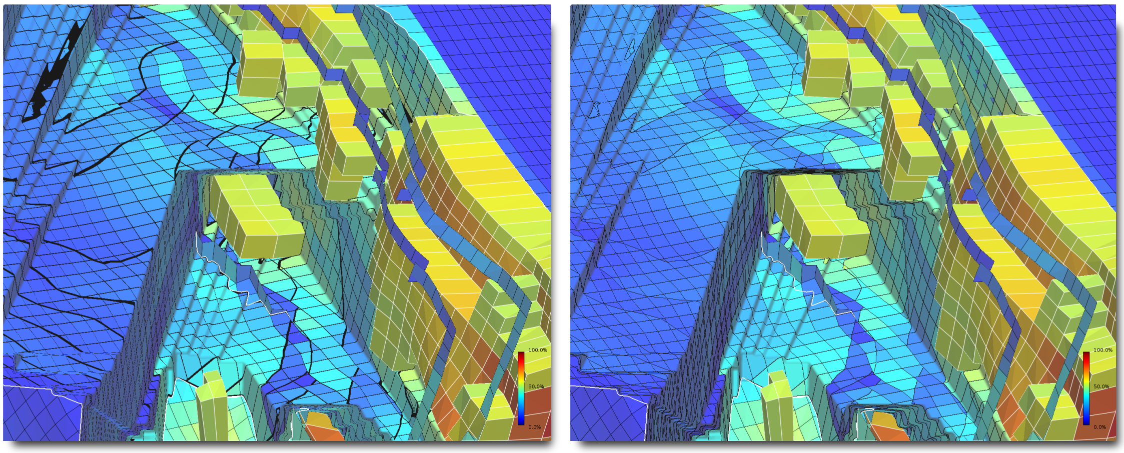


Figure 4.8: The intensity, I_p, of a fragment centered at p is computed based on the window space distances d_1,d_2, and d_3 to the three edges of the triangle. More details in [74].



(a) Threshold based line rendering

(b) Raycasting based line rendering

Figure 4.9: A corner-point model with some cells in focus, and using a narrow cutaway angle. Figure 4.9a illustrates the internal lines rendered using a threshold between the cut surface and the depth of the fragment. In Figure 4.9b internal lines are rendered using our ray casting procedure. Note how the thickness of the black lines varies arbitrarily using a threshold approach, and is stable using the ray-casting test.

a more precise test is carried out by checking the neighbors of the fragments. If one or more adjacent fragments were clipped, it means that this is a border pixel. Since in the fragment shader we do not have access to others fragments being processed, we employ a simple ray casting algorithm to check if the geometry projected in the neighboring fragment would be clipped against the cut surface. Our only assumption is that the neighbors belong to the same cell face, i.e., are coplanar in model space.

A four pixel neighborhood is used: top, down, right and left. To test a neighbor, we start by reading the depth value z_{p_i} from the cutaway buffer at its position (Figure 4.10). To predict the neighbor's depth, we re-project the neighboring pixel

p_i onto the plane defined by the current pixel position in world coordinates and the normal n_f of the face that generated it. The distance $d(p_i, p_i')$ is tested against the value z_{p_i} to determine if the neighbor pixel would be clipped or not. If at least one neighbor is clipped, the current pixel is marked as a border pixel and is shaded in a different manner to emphasize the cut. The size of the neighborhood determines the thickness of the lines, for example, if we test a neighbor two pixels away, a two-pixel thick line is produced.

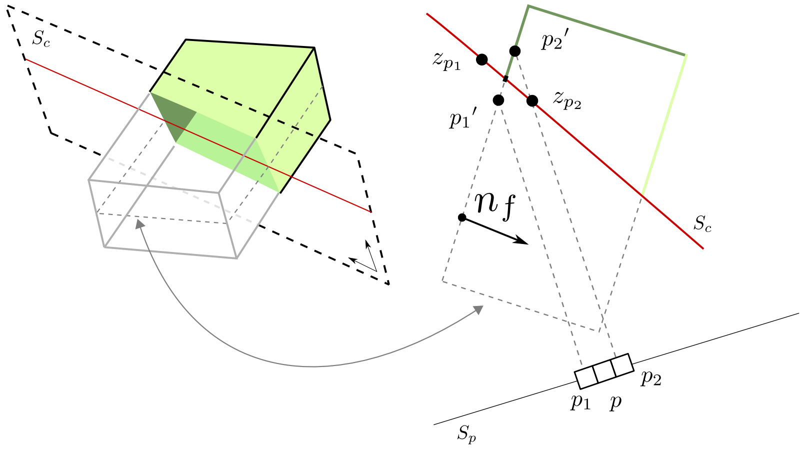
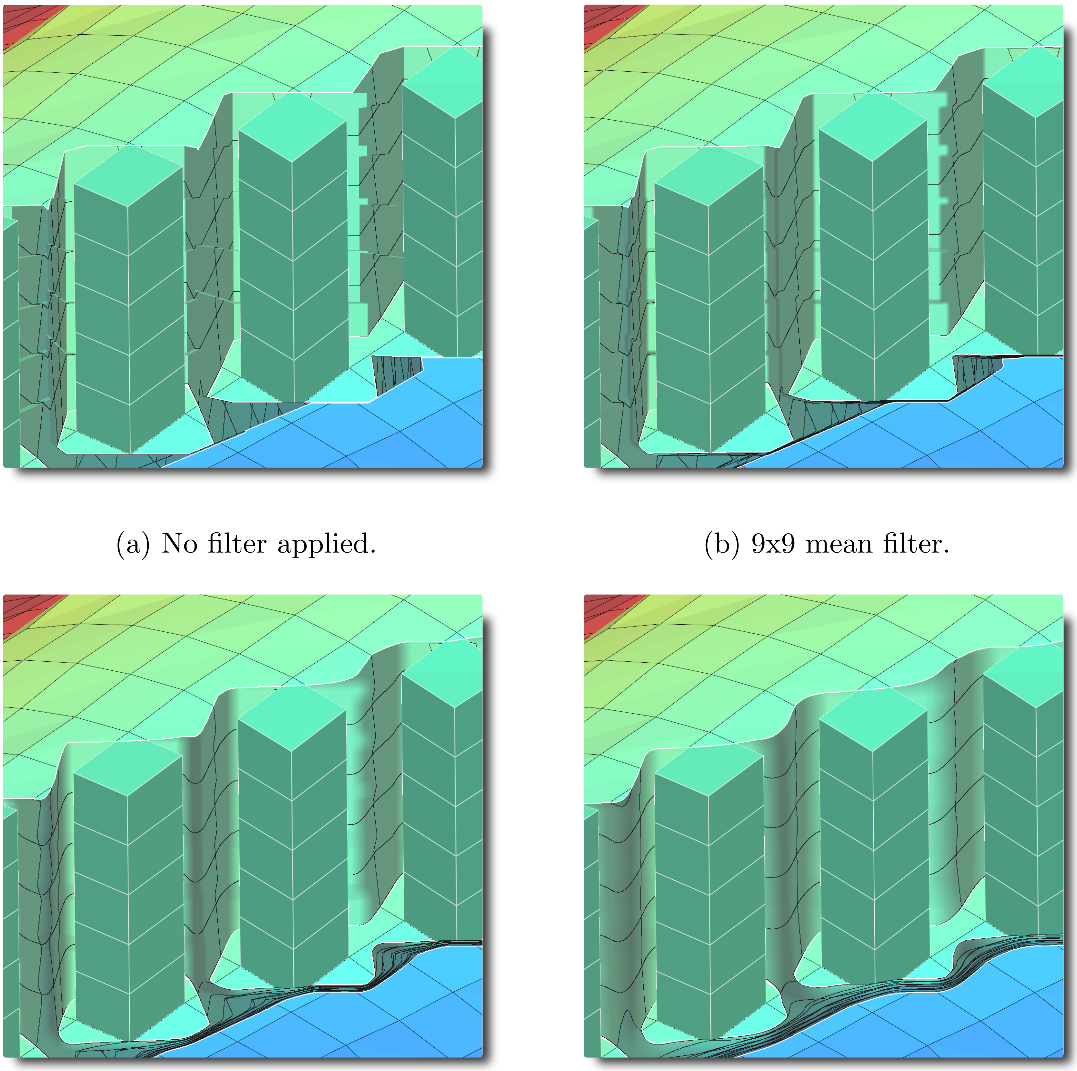


Figure 4.10: Ray Casting border detection: the pixel p tests its neighbor's projection onto the face (indicated by normal n_f) to test if their depth value z_{p_i} falls behind the cut surface (line in red). In this case p is a border pixel, since p_1 falls in front of the cut surface.

4.3.4 Extra Lines and Features

Lines are an important feature style in many illustrative techniques [76]. They offer a minimal visual representation of a scene with little visual clutter [77]. Contours, for example, are feature lines that convey the structure of the object. We extract these contours from the shell faces in a pre-processing stage, and render them in a last render pass. Another important feature line is the one that defines the intersection of the shell faces with the cut surface, emphasized in red in Figure 4.7a. These are also rendered in this final pass. The overhead of these feature lines is minimal and they provide useful clues about the global structure of the model.

Another visual improvement is the smoothing of the cutaway depth image. When many frustums are generated, a staircase effect is produced after unifying them in screen space. To smooth the transition between the different frustums a mean filter is applied directly on the depth image. Figure 4.11 illustrates the effect of applying the filter with different kernel sizes. The wider the kernel the more impact it causes on the rendering performance, however, usually a small filter, such as 9×9, is enough.



(a) No filter applied.

(b) 9x9 mean filter.

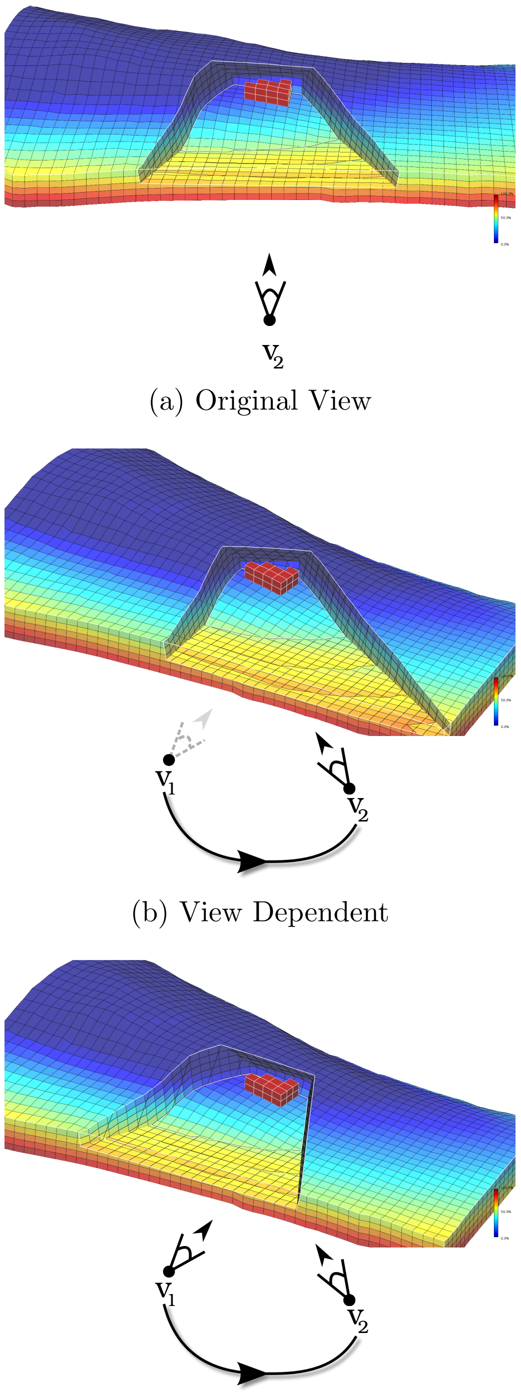
(c) 33x33 mean filter.

(d) 65x65 mean filter.

Figure 4.11: Application of a mean filter to smooth the union of the cut surfaces in image space. This is an extreme case for illustration purposes, in practice a small filter produces enough smoothing.

Note that large kernels might violate the occlusion premise, since some primary cells can be partially occluded by the smoothed surface. Even though a more accurate or controlled smoothing could be explored, we have decided to not hinder performance since no significant visualization issue was noted with this artifact. This effect can be noted in Figure 4.11d.

To further highlight the primaries, and increase spatial perception, we add indirect shading, following a Screen Space Ambient Occlusion (SSAO) approach by SHANMUGAM and ARIKAN [78]. The only required modification in the pipeline, is that we have to save the cutaway surface's coordinates as well as the normals in a buffer. With the same goal in mind, i.e., to enhance the contrast be-



(a) Original View

(b) View Dependent

(c) Freeze View

Figure 4.12: Freeze view feature for improved depth perception as proposed by Lidal et al. [42]. In Figure 4.12a the start view position v_1 from the camera. In Figure 4.12b we have the scene rendered with view dependent cutaway generation. Figure 4.12c shows the freeze view in action, where the original camera view v_1 is used to generate the cutaway rendering and a second camera view v_2 to visualize the scene. With freeze view it is possible to gain more insight about the spatial location of the features in focus.

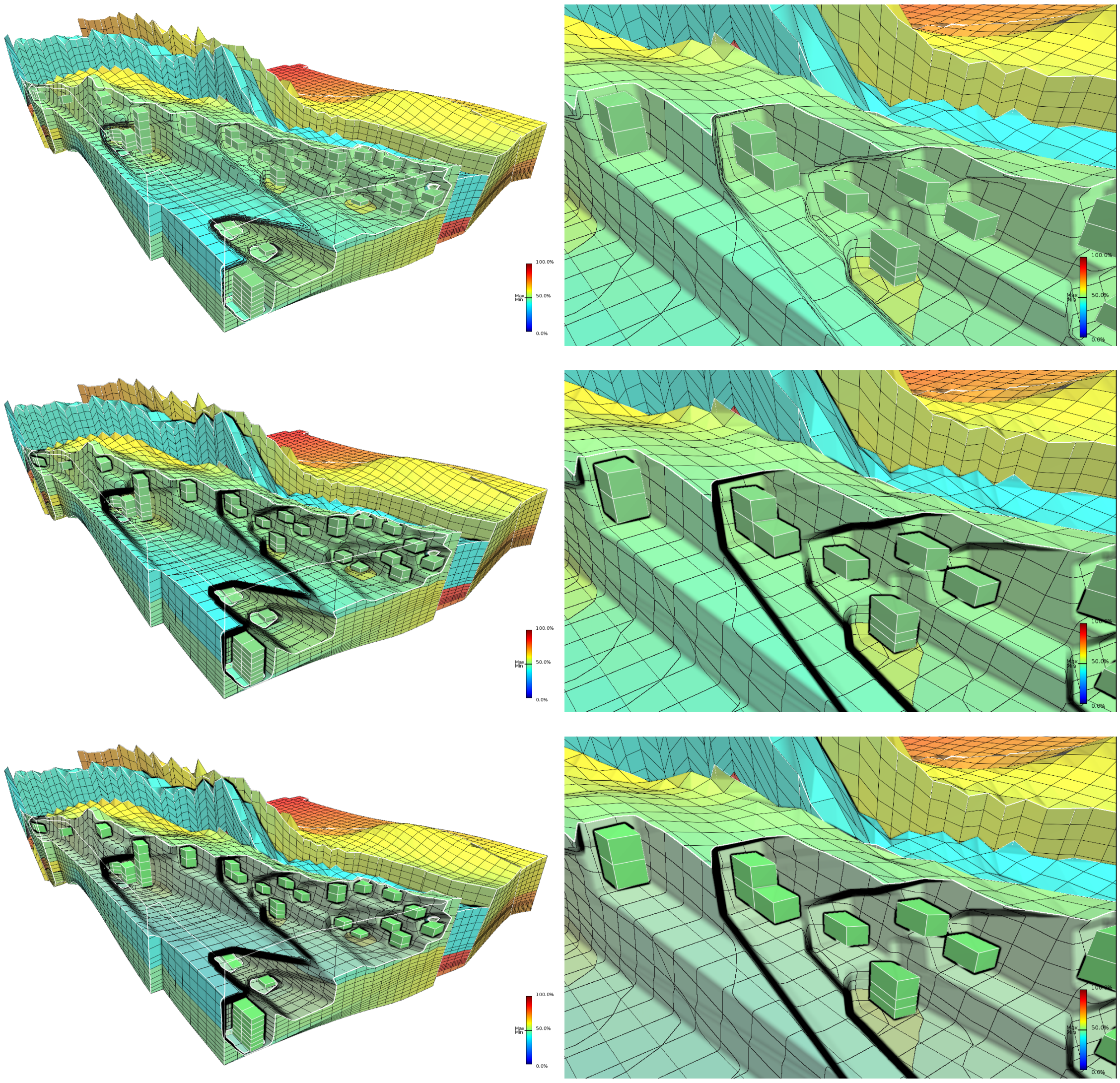


Figure 4.13: From top to bottom: the result of the cutaway; applying SSAO; and increasing/decreasing the saturation channel of the primaries/secondaries. Note how these two techniques help in distinguishing primaries from secondaries. Since the selected value range is very tight, the minimum and maximum values on the color bar are practically overlapping for this example. The SSAO effect is a little exaggerated in this case for illustration purposes.

tween the primaries and secondaries, we slightly increase the saturation channel of the first group while decreasing it for the second group. Figure 4.13 illustrates both effects.

Finally, we also employ the concept of a freeze-view mode, where one can rotate the model without changing the frustums. In other words, the transformation matrices for the frustums are kept still while the model continues to rotate or translate. This allows for an improved depth perception of the cut surface. This effect is illustrate in Figure 4.12. Note, however, that there is a limit to the maximum angle between the initial and final view directions. As long as the projected shape of the frustum can be represented as a height-map, that is, the projection of the walls do not overlap, the freeze view will work as expected. When the angle does not respect this rule, some unclear behavior may be noted, as shown in Figure 4.14.

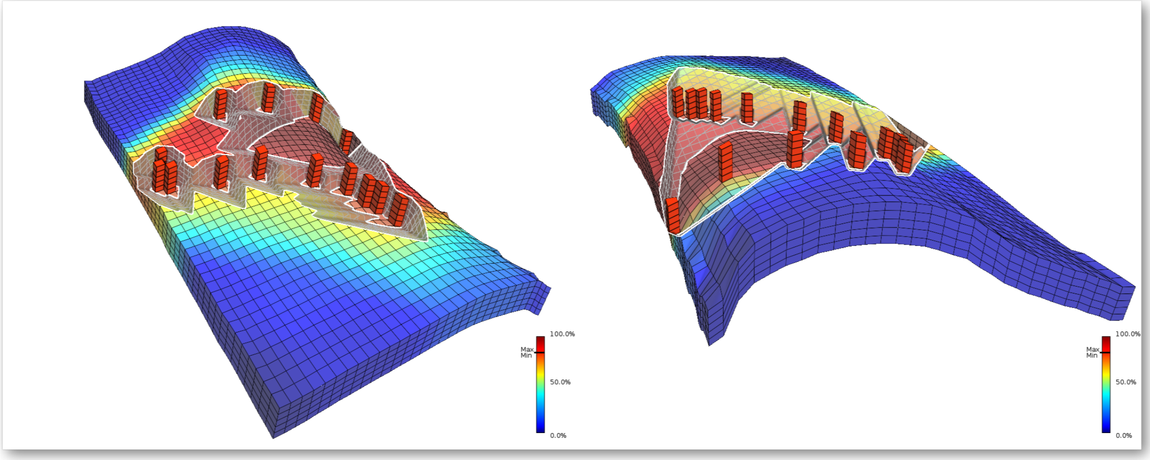


Figure 4.14: Freeze view limitation. Left: the front view of the cutaway surface. Right: when the model is rotated 180 degrees, the cutaway surface is not preserved.

Chapter 5

Results

"I'm a greater believer in luck, and I find the harder I work the more I have of it."
— Thomas Jefferson

In this chapter we discuss our proposed method and its performance. We start by reviewing the work realized in each stage, and in Section 5.1 we show performance results using the models listed in Table 5.1. We also carried out a study with two domain experts that recurrently use corner-point models to inspect pos-processing simulations. In Section 5.2, we discuss the feed-back given by the two domain experts, and how it can be implemented in future works.

Model Name	Number of Cells
Zaphirus	7500
Petra	32760
Opharine	76000
Sapphire	208320

Table 5.1: Four reservoir models using our cutaway method. All the models were rendered using a 1920×1080 screen resolution and a mean filter with kernel size of 9×9.

As described in Section 4.3, our method is performed in three stages. One to select the primary cells and generate the depth image from their bounding boxes. The second clips the secondary cells and reconstructs the internal walls. Finally, the last one emphasizes some features of the model and renders the primary cells.

The first stage computes and orients a bounding box for each primary cell in the vertex shader using only transformation matrices and the geometry of a cell. With this information the shader calculates the points of the bounding box and normals of each face. Even though the number of texture accesses are limited, this computation still involves many cross and dot products for each bounding box.

Moreover, we discriminate the primaries from the secondaries on-the-fly, introducing a further overhead in this stage.

In the second stage we clip the secondary cells against the cut surface, fill the cells, and reconstruct the lines. Filling the cells is trivial, we just need to change the normal of each fragment generated from an internal face. The expensive part is to reconstruct the lines. For a border line of minimum width (i.e., one), we have to access the depth buffer four times per fragment. When we make the surface angle wider, more fragments are discarded, and consequently more of the cutaway wall is exposed, implying in more ray casting tests (texture accesses).

The third stage imposes a very small overhead. This stage renders some contour lines and the primary cells. Even though it would be possible to render the primaries during the second pass, we decided to render them in a separated stage, to allow for more freedom when applying some effects to emphasize the features in focus. For example, it is possible to change the color of the cells and lines since it might be an important contrast enhancement.

5.1 Experiments

In this section, we detail the performance of our method. Table 5.1 shows four reservoir models of various sizes, used in our experiments. The experiments were realized by selecting some primaries cells and rotating the model in different orientations, while, at the same time, changing the cut surface aperture.

We track each stage of the model, including the mean filter and SSAO. Figures 5.1, 5.2,5.3 and 5.4, depict the model and its associate graph with performance numbers. The reported times are averaged frame rates. We noted that the frame rate slightly drops when using a wide angle aperture for the cut surface. Although in this scenario we are eliminating more secondary cells, and consequently sending less cells down the rendering pipeline, we are also defining larger cutaway walls and thus exposing more context. In other words, more time is dedicated to the ray casting algorithm to reconstruct the internal lines and create the walls for the sliced cells.

Figure 5.1 shows the performance times for the Zaphirus model. As can be noted, it is the only one where the first stage consumes more time than the second. The main reason is because the projected area exposed by the cutaway is small, basically on the sides of the primaries cells, as shown in Figure 5.1 (right).

The other models have the same performance pattern. From the graph in Figures 5.2,5.3 and 5.4, we note that there is a period of time where the line describing the Cutaway Render and Cutaway Generation converge. This happens when the area of the cutaway is small and the primaries are rendered in the center of the screen. The image on the right side of the graph depicts this scenario.

The SSAO and Mean filter features are both image space algorithms. Thus, they tend to have a smoother behavior during rendering, varying only according to the projected area of the models.

In Figures 5.5, 5.7, 5.8 and 5.9a, we have more images depicting the method in different scenarios.

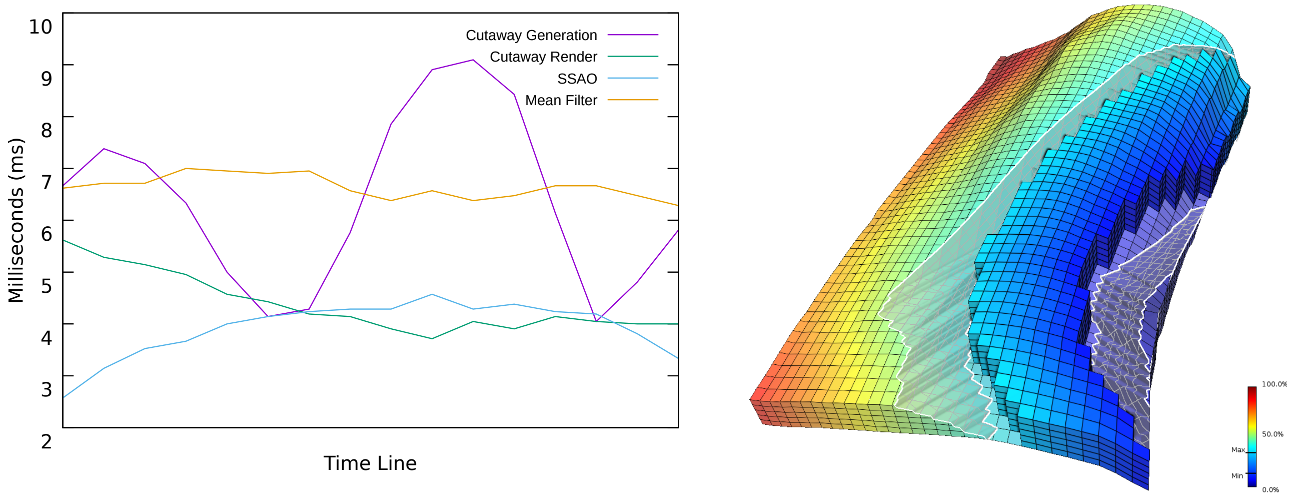


Figure 5.1: Model Zapphirus

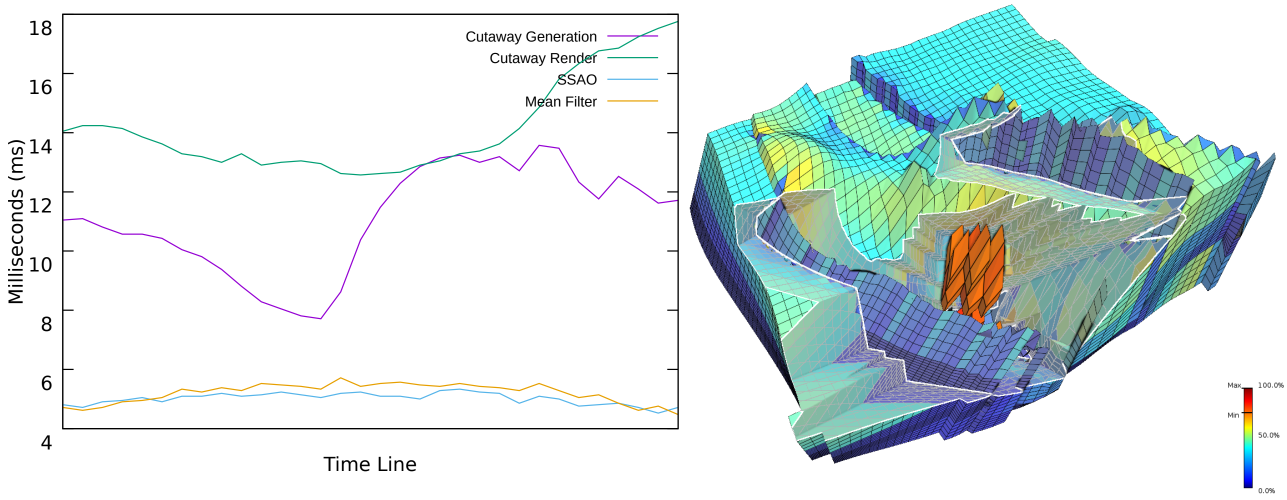


Figure 5.2: Model Petra.

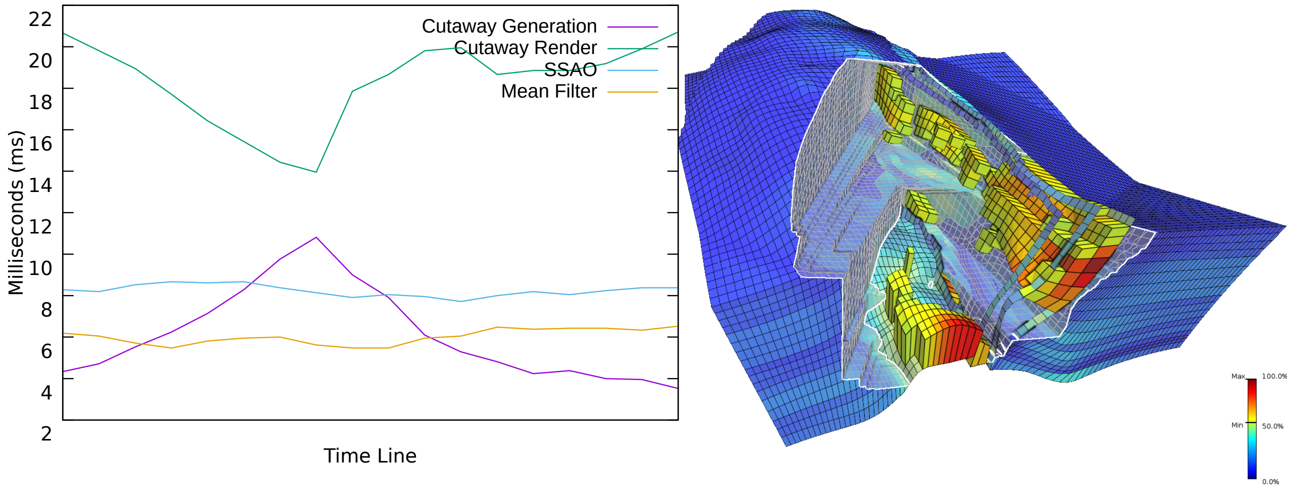


Figure 5.3: Model Opharine.

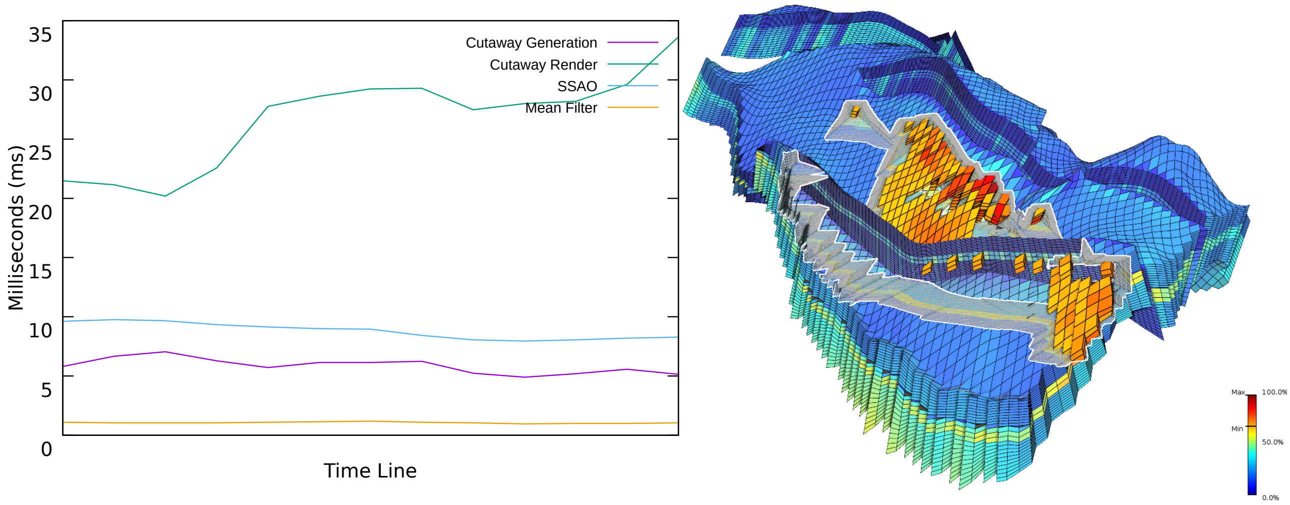


Figure 5.4: Model Zapphirus.

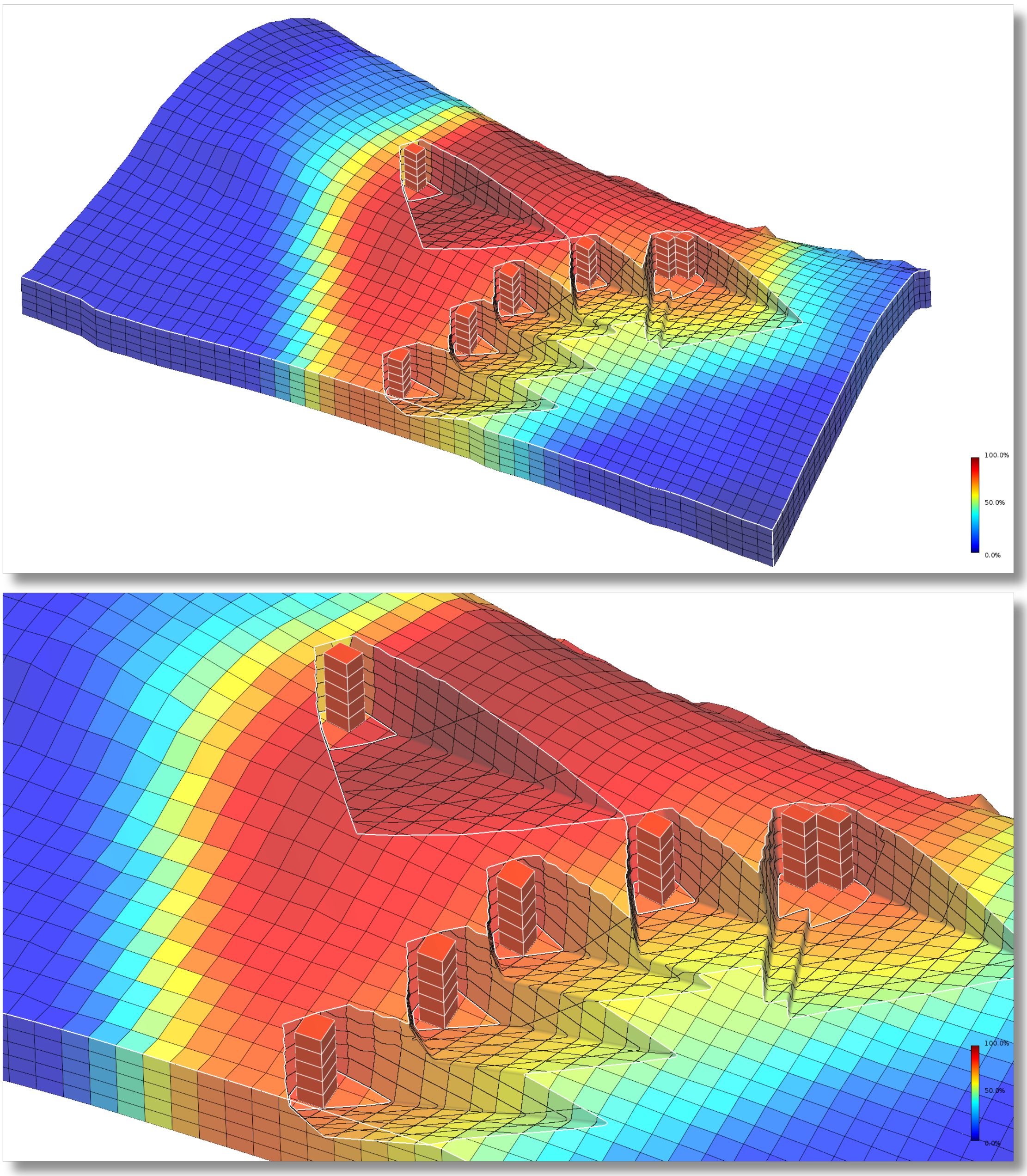


Figure 5.5: Zaphirus model with the static property porosity.

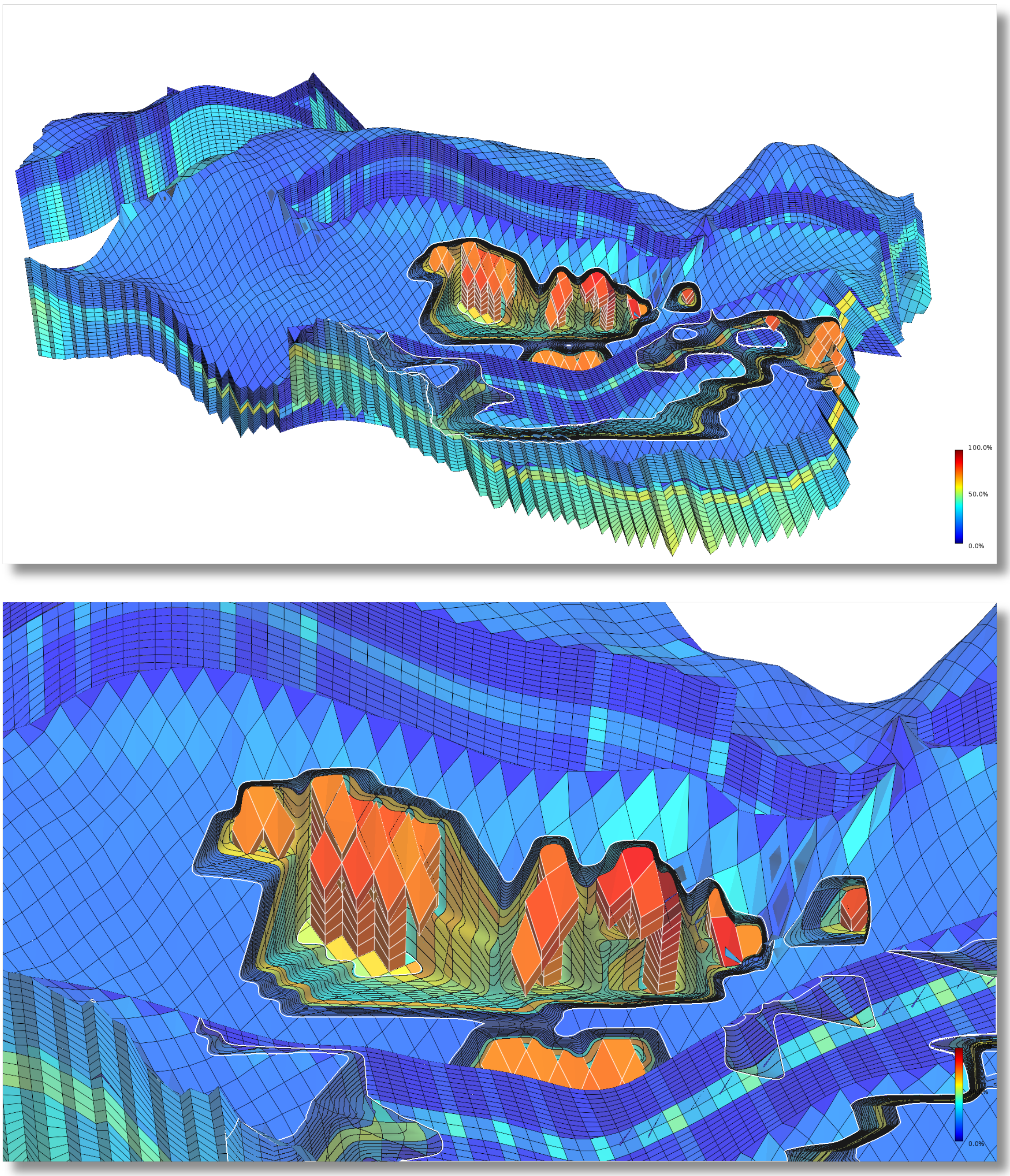


Figure 5.6: Sapphire model with the static property modified block volume.

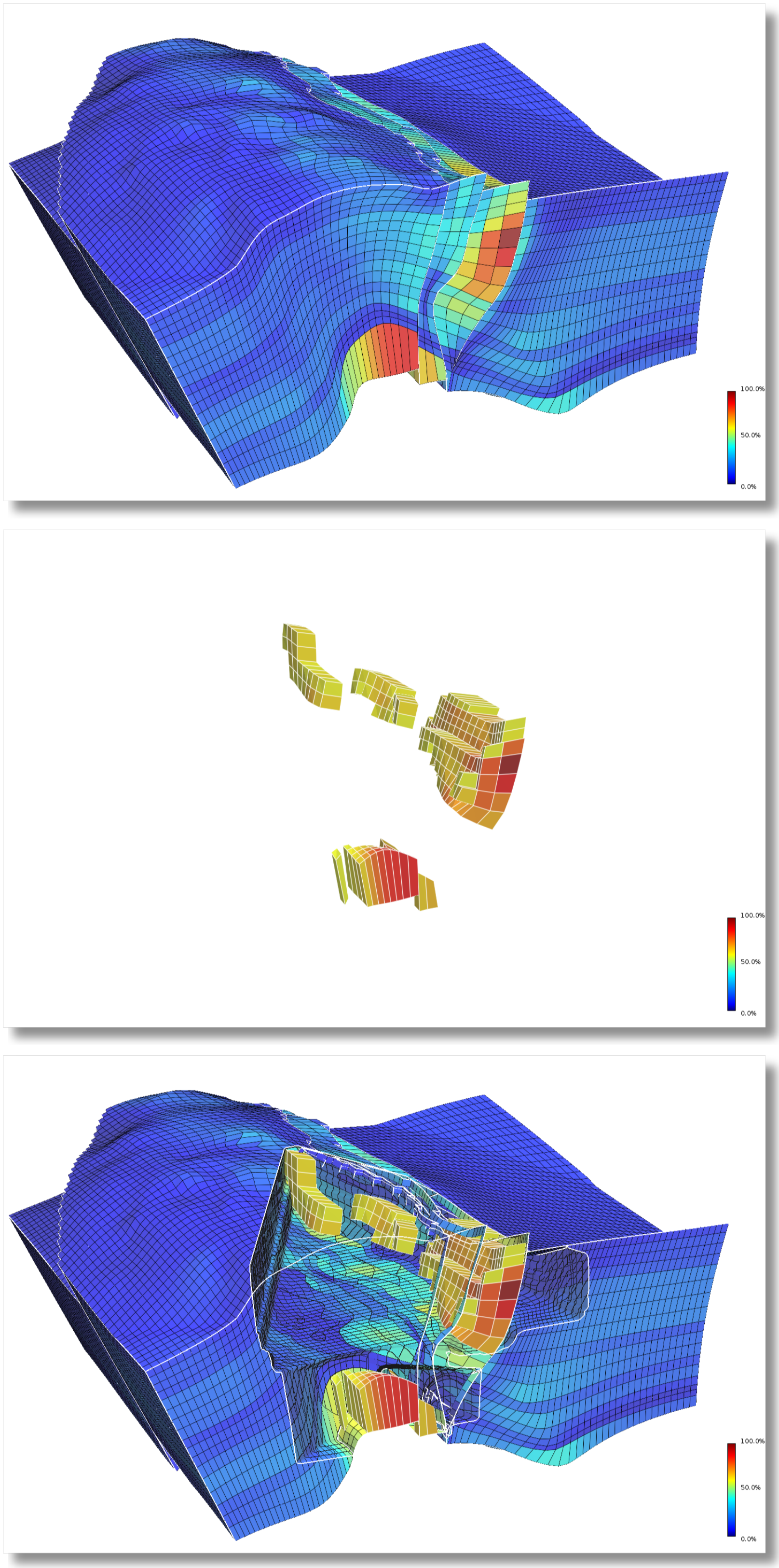


Figure 5.7: Top: the raw model. Middle: some cells are selected for inspection. Bottom: the desired visualization.

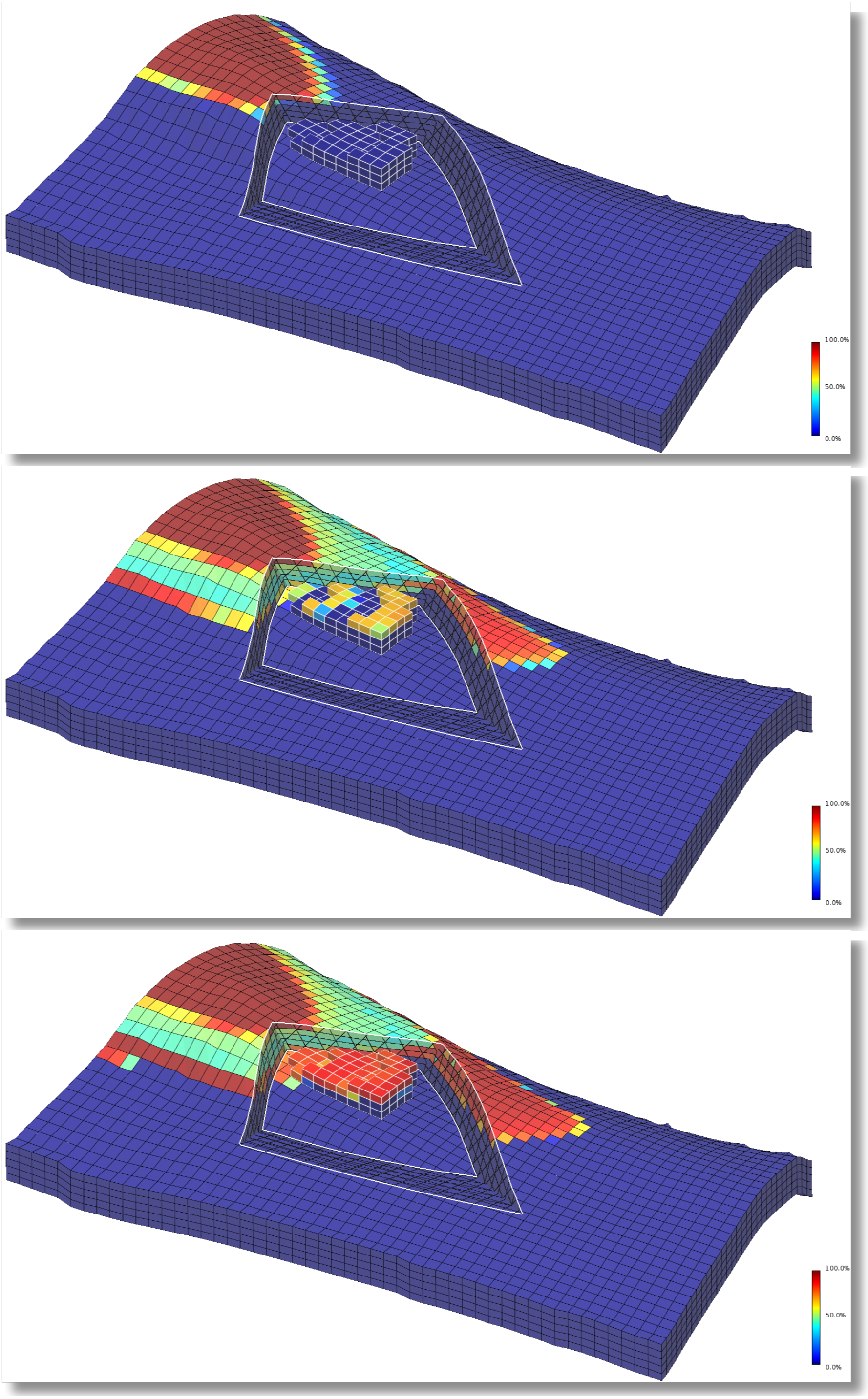
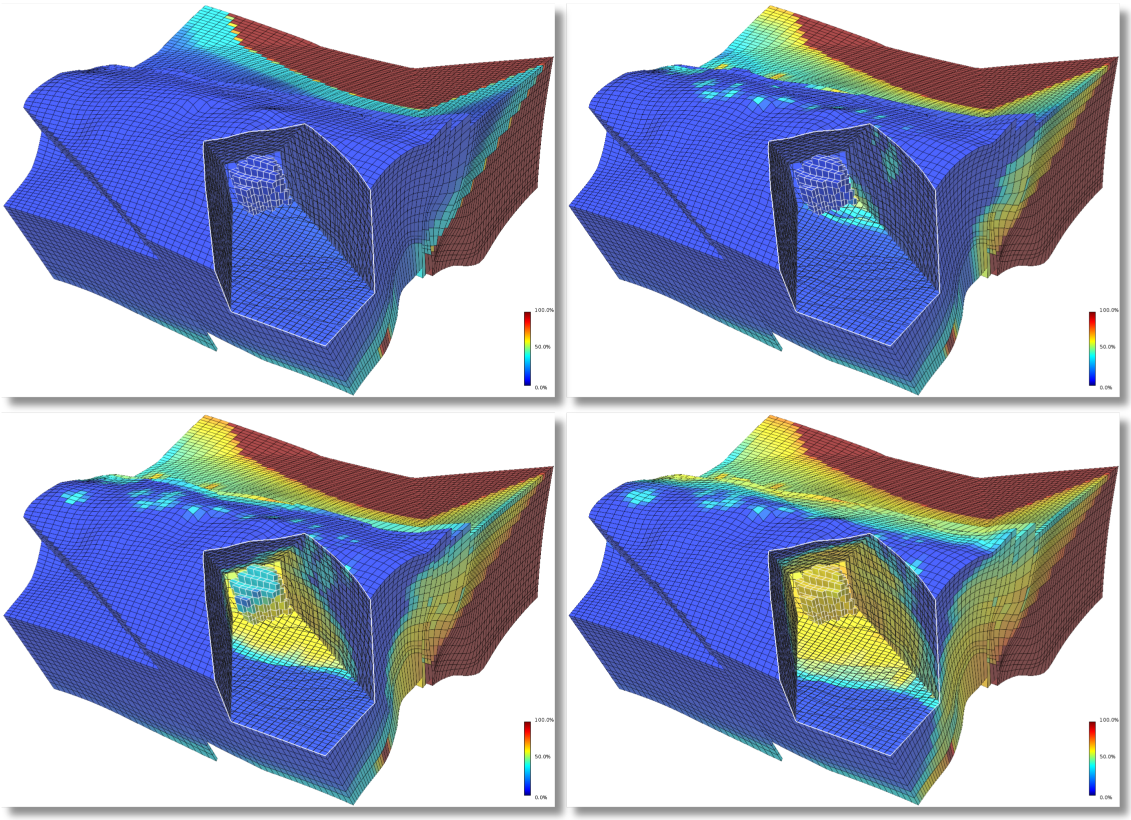


Figure 5.8: Zaphirus model and the dynamic attribute Oil Saturation in different time steps.



(a) Dynamic attributes visualization. Focus cells being visualized with different time steps. The sequence is top-left, top-right, bottom-left and bottom-right.

Figure 5.9: Opharine model and the Water Saturation dynamic attribute.

5.2 Design Critique

Gathering feedback from experts that have experience analyzing corner point models is essential to fine tune the visualization technique achieving the expressiveness required by the real industrial tasks.

We hosted design critique sessions, with two expert practitioners involved in upstream research in the oil and gas industry. One domain expert has more than 30 years of industry experience including research and development of reservoir simulation gridding and visualization techniques and software systems; the other domain expert has 10 years of experience in fluid flow simulation in various gridding methods and models, including corner-point. Both domain experts have PhD. degrees in petroleum engineering.

We have led one independent session with each expert, letting them test the prototype for about 30 minutes. The two experts stated that the visualization technique allows easily locating the primary cells in the reservoir, while keeping the context of the surrounding cells. They also commented that to the best of their knowledge, no current system for the oil-and-gas industry is able to achieve similar results. Both highlighted that the freeze view, silhouettes curves, and smoothing edges features improve the reservoir's spatial perception. They suggested this technique could have a great impact for front track visualization of fluid simulation and inspection of wells inside the reservoirs.

In addition, they pointed out as a limitation the difficulty to differentiate between primary and secondary cells when there is a large number of primaries. To tackle this issue they suggested the following as options that could be enabled/disabled within the system: (1) remove all cells that are between the primary cells and the camera, (2) hide/draw the edges of the secondary blocks, and (3) draw the silhouette of the primary cluster in different color and line style. Nevertheless, all these requests are easily achievable using our method without introducing any technical challenge.

Although this was still an informal study, it sheds light on the potential of the proposed technique to visualize a set of corner-point cells while providing a good understanding of the surrounding reservoir geometry. The study also indicates good application and improvements opportunities.

Chapter 6

Conclusion

"There is no real ending. It's just the place where you stop the story."

— Frank Herbert

In this thesis we have presented an interactive cutaway visualization method to aid in the inspection of Oil&Gas reservoir models represented by corner-points. The method produces visualizations that help to address the challenges outlined in the introduction: convey structural information of internal features of corner point models. The relationships between the focus cells and the context are important for domain experts, helping them to gain a semantic understanding of the phenomena in study.

The method runs in interactive frame rates for models with reasonable size, ie. up to 200K cells. Our solution is based on a depth representation of the cut surface to perform the clipping process. All cuts and lines are realized in image space, making the method partially bounded by the screen resolution, but since all cells are projected we are also bounded by the model's size.

Finally, we made a few enhancements to increase the perceptual contrast of the visualization, by rendering and shading some important contour lines.

During our research we have made a few choices, and an important one in our point-of-view was to achieve a well defined and continuous cut of the volume. This choice strayed us away from volumetric rendering approaches, as well as a binary classification of the cells that should be removed. As can be seen from the resulting images, internal structures such as faults and layers are evidenced and clearly exposed using the boundary lines. Nevertheless, if the complexity of the model greatly increases, too many lines may render a confusing visualization, so in these cases, further research is necessary to draw only the most important features to keep the context clean and comprehensible.

Another point that deserves discussion, is the choice to not used a Distance Transform approach (such as Jump-Flooding) to create the screen space cutaway surface. Basically, as also emphasized by LIDAL *et al.* [42], a more regular shaped cut (such as a truncated pyramid, or frustum) increases visual comprehension, as opposed to a surface that is tighter but more irregular. A frustum shaped cut also conforms better to most reservoirs structure. Furthermore, it is easier to control the frustum aperture, since we can extend/contract it in two axis independently (horizontally or vertically). One disadvantage is that creating and projecting one frustum for each primary cell is costly, and a Distance Transform approach will probably be more efficient.

We have specifically applied our approach to Corner-Point grids, even though there is nothing that restricts it to this model representation. The method could be readily employed in different types of datasets with minimum hassle, since it is generic enough.

6.1 Future Works

Here we summarize some avenues of future research which have come up during the work on this thesis:

Visualization of large models. The results presented in this work produce interactive cutaway visualization of models with a reasonable size. However, we would like to aim at even larger models, with millions of cells. Our solution works with a screen space representation of the cutaway surface, and consequently, is already scalable to some degree. However, we still have to project all cells at least twice during the process, thus some model space approaches might be necessary to avoid projecting the whole set of cells. Also, there are a number of low-level performance issues to address in the current technique that can further improve the frame, such as passing the bounding box centroid and its extensions, instead of its eight vertices. Another idea to improve performance is to create clusters of primary cells, and consequently avoid generating and projecting many small frustums. This would also let us treat the clusters as a single entity, and new effects could be though of, such as enhancing the contour of the clusters with lines or shading strategies.

Illustrative enhancements. We try to respect the color scheme of the cells as much as possible, since it is the way the reservoir engineer usually inspects them. In the future we would like to realize a study with domain experts, in the spirit of the work of LIDAL *et al.* [42] and LI *et al.* [40], to come up with design principles to effective apply cutaways in the other contexts, such as wells visualizations. Furthermore, we would like to explore combining cutaways with other techniques, such as ghosting view, peeled-away and exploded view.

Visualization of time-varying attributes. This base cutaway technique can be extended to dynamic scenarios associated to the dynamic reservoirs simulations attributes. Although the proposed technique can be used to expose internal structures at a particular moment in the time, it would be interesting to develop visualization methods that explicitly consider the dynamic time varying nature of some of its attributes. However, tracking dynamic properties over time may result in visual confusion as cells may arbitrary pop in and out of the primary set. Thus, further studies on how to provide a smooth time-coherent cutaway visualization are necessary.

Authoring tool for visual explanations. In this thesis, we focus on revealing internal structures in order to convey spatial and contextual information in corner-point models, to help expert domain explore the models with a powerful exploration technique. However, in combination with illustration techniques, those models can be used to communicate high level concepts to a non-expert audience, allowing a better understanding the phenomena in study. In combination with illustration techniques, and additional design elements (e.g. arrows, labels, short animation, and so on) can further increase abstractions and convey functional and temporal relationships, in the case of time-varying data.

Intelligent camera. Taking a step further within the dynamic scenario above, a future goal is to also automatically create animations. Given the reservoir model and a dynamic attribute range to track, the camera should be positioned for each frame to render the best possible visualization. However, now the smooth time-coherent issue is even more critical, since the best view direction for one frame might be very different from the subsequent one. In this case a compromise must be achieve, and sometimes the animation should be slowed down to allow for the camera to arrive at its destination, for example.

Bibliography

[1] L.P, B. "Total, Shell Chief Executives Say: Easy Oil Is Gone". http://www.bloomberg.com/apps/news?pid=newsarchive&sid=aH57.uZe.sAI, 2007.

[2] GANTZ, J., REINSEL, D., CHUTE, C., et al. "The Expanding Digital Universe: A Forecast of Worldwide Information Growth Through 2010". http://www.emc.com/leadership/programs/digital-universe.htm, 2007.

[3] CARD, S., MACKINLAY, J., SHNEIDERMAN, B. *Readings in Information Visualization: Using Vision to Think*. Interactive Technologies Series. San Francisco, CA, USA, Morgan Kaufmann Publishers, 1999. ISBN: 9781558605336. URL: <http://books.google.com.br/books?id=wdh2gqWfQmgC>.

[4] MACKINLAY, J. "Opportunities for information visualization", *IEEE Computer Graphics and Applications*, v. 20, n. February, pp. 22–23, 2000. URL: <http://www.computer.org/csdl/mags/cg/2000/01/mcg2000010022.pdf>.

[5] RHYNE, T. "Does the difference between information and scientific visualization really matter?" *IEEE Computer Graphics and Applications*, v. 23, n. June, pp. 6–8, 2003. URL: <http://www.computer.org/csdl/mags/cg/2003/03/mcg2003030006.pdf>.

[6] ELMQVIST, N. *3D Occlusion Management and Causality Visualization*. Phd thesis, Chalmers University of Technology and Göterborg University, 2006. URL: <https://engineering.purdue.edu/~elm/https://engineering.purdue.edu/~elm/projects/phd-thesis/summary.pdf>.

[7] KOSARA, R. "Visualization Criticism - The Missing Link Between Information Visualization and Art". In: *Proceedings of the 11th International Conference Information Visualization*, IV '07, pp. 631–636, Washington, DC,

USA, 2007. IEEE Computer Society. ISBN: 0-7695-2900-3. doi: 10.1109/IV.2007.130. URL: <`http://dx.doi.org/10.1109/IV.2007.130`>.

[8] STATOIL. "Gina Krog Oil Field Illustration". jun 2014. URL: <`http://www.statoil.com/en/OurOperations/FutureVolumes/ProjectDevelopment/Pages/Dagny.aspx`>.

[9] EVANS, F., VOLZ, W., DORN, G., et al. "Future Trends in Oil and Gas Visualization". In: *Proceedings of the Conference on Visualization '02*, VIS'02, pp. 567–570, Washington, DC, USA, 2002. IEEE Computer Society. ISBN: 0-7803-7498-3. URL: <`http://dl.acm.org/citation.cfm?id=602099.602200`>.

[10] AARNES, J., KROGSTAD, S., LIE, K.-A. "Multiscale mixed/mimetic methods on corner-point grids", *Computational Geosciences*, v. 12, n. 3, pp. 297–315, 2008. ISSN: 1420-0597. doi: 10.1007/s10596-007-9072-8. URL: <`http://dx.doi.org/10.1007/s10596-007-9072-8`>.

[11] BORGO, R., BRODLIE, K. "State of the Art Report on GPU Visualization", *The University of Leeds, http://www.viznet.ac.uk/reports/gpu/1*, 2009.

[12] ENEH, O. C. "A Review on petroleum: Source, uses, processing, products and the environment. J", *Applied Sci*, v. 11, pp. 2084–2091, 2011.

[13] HÖÖK, M., BARDI, U., FENG, L., et al. "Development of oil formation theories and their importance for peak oil", *Marine and Petroleum Geology*, v. 27, n. 9, pp. 1995–2004, oct 2010. ISSN: 02648172. doi: 10.1016/j.marpetgeo.2010.06.005. URL: <`http://www.sciencedirect.com/science/article/pii/S0264817210001224`>.

[14] BANDY, M. C., BANDY, J. A., OTHERS. *De Natura Fossilium (Textbook of Mineralogy)*. Mineola, NY, Courier Dover Publications, 2004.

[15] OF ENERGY, U. D. "Global Energy Statistical Yearbook 2013". `http://yearbook.enerdata.net/`, 2013.

[16] VAN DER HOEVEN, M. "World Energy Outlook", *International Energy Agency*, Nov 2013. ISSN: 2072-5302. URL: <`http://dx.doi.org/10.1787/weo-2013-en`>.

[17] LIBES, S. *Introduction to Marine Biogeochemistry*. Waltham, Massachusetts, Elsevier Science, 2011. ISBN: 9780080916644. URL: <`http://books.google.com.br/books?id=KVZJUw4nORgC`>.

[18] GOMES, J. S., ALVES, F. B. *The Universe of the Oil and Gas Industry: From Exploration to Refining.* 1st ed. Brazil, Partex Oil and Gas, 2013. ISBN: 9892037782.

[19] JAGER, G. D. *The influence of geological data on the reservoir modelling and history matching process.* PhD thesis, Technische Universiteit Delft, 2012.

[20] LAWNICZAK, W. *Feature-Based Estimation for Application in Geosciences.* PhD thesis, Technische Universiteit Delft, 2012.

[21] DEVOLD, H. "Oil and Gas Production Handbook An Introduction to Oil and Gas Production, Transport, Refining and Petrochemical Industry", *ABB, 3rd*, 2013.

[22] JANSEN, J.-D., BOSGRA, O. H., VAN DEN HOF, P. M. "Model-based control of multiphase flow in subsurface oil reservoirs", *Journal of Process Control*, v. 18, n. 9, pp. 846–855, out. 2008. ISSN: 09591524. doi: 10.1016/j.jprocont.2008.06.011. URL: <`http://www.sciencedirect.com/science/article/pii/S0959152408001066`>.

[23] DING, Y., LEMONNIER, P. "Use of Corner Point Geometry in Reservoir Simulation". 1995.

[24] JANSEN, J.-D., BROUWER, D., NAEVDAL, G., et al. "Closed-loop reservoir management", *First Break*, v. 23, n. 1, 2005.

[25] CHAVENT, G., DUPUY, M., LEMMONIER, P., et al. "History matching by use of optimal theory", *Society of Petroleum Engineers Journal*, v. 15, n. 01, pp. 74–86, 1975.

[26] GAVALAS, G., SHAH, P., SEINFELD, J. H., et al. "Reservoir history matching by Bayesian estimation", *Society of Petroleum Engineers Journal*, v. 16, n. 06, pp. 337–350, 1976.

[27] GOSSELIN, O., AANONSEN, S., AAVATSMARK, I., et al. "History matching using time-lapse seismic", *Journal of petroleum technology*, v. 56, n. 4, pp. 66–67, 2004.

[28] BENNETT, A., CHUA, B., LESLIE, L. "Generalized inversion of a global numerical weather prediction model", *Meteorology and Atmospheric Physics*, v. 60, n. 1-3, pp. 165–178, 1996. ISSN: 0177-7971. doi: 10.1007/BF01029793. URL: <`http://dx.doi.org/10.1007/BF01029793`>.

[29] BENNETT, A. F. *Inverse modeling of the ocean and atmosphere.* Cambridge, England, Cambridge University Press, 2002.

[30] VALSTAR, J. R., MCLAUGHLIN, D. B., TE STROET, C., et al. "A representer-based inverse method for groundwater flow and transport applications", *Water Resources Research*, v. 40, n. 5, 2004.

[31] BROUWER, D., JANSEN, J., OTHERS. "Dynamic optimization of waterflooding with smart wells using optimal control theory", *SPE journal*, v. 9, n. 04, pp. 391–402, 2004.

[32] SARMA, P., AZIZ, K., DURLOFSKY, L. J., et al. "Implementation of adjoint solution for optimal control of smart wells". In: *SPE Reservoir Simulation Symposium*. Society of Petroleum Engineers, 2005.

[33] ZANDVLIET, M., HANDELS, M., VAN ESSEN, G., et al. "Adjoint-based well-placement optimization under production constraints", *SPE Journal*, v. 13, n. 04, pp. 392–399, 2008.

[34] ZANDVLIET, M. *Model-based lifecycle optimization of well locations and production settings in petroleum reservoirs.* PhD thesis, Delft University of Technology, Delft, The Netherlands, 2008.

[35] NÆVDAL, G., BROUWER, D. R., JANSEN, J.-D. "Waterflooding using closed-loop control", *Computational Geosciences*, v. 10, n. 1, pp. 37–60, 2006.

[36] SARMA, P., DURLOFSKY, L. J., AZIZ, K., et al. "Efficient real-time reservoir management using adjoint-based optimal control and model updating", *Computational Geosciences*, v. 10, n. 1, pp. 3–36, 2006.

[37] CHEN, Y., OLIVER, D. S., ZHANG, D., et al. "Efficient ensemble-based closed-loop production optimization", *SPE Journal*, v. 14, n. 04, pp. 634–645, 2009.

[38] TRANI, M. *From time-lapse seismic inversion to history matching of water flooded oil reservoirs.* PhD thesis, Technische Universiteit Delft, 2011.

[39] VIOLA, I., KANITSAR, A., GROLLER, M. "Importance-driven volume rendering". In: *IEEE Visualization 2004*, pp. 139–145. IEEE Comput. Soc, oct 2004. ISBN: 0-7803-8788-0. doi: 10.1109/VISUAL.2004.48. URL: <`http://dl.acm.org/citation.cfm?id=1032664.1034441`>.

[40] LI, W., RITTER, L., AGRAWALA, M., et al. "Interactive cutaway illustrations of complex 3D models", *ACM Trans. Graph.*, v. 26, n. 3, pp. 31, 2007. doi: http://doi.acm.org/10.1145/1276377.1276416.

[41] BURNS, M., FINKELSTEIN, A. "Adaptive cutaways for comprehensible rendering of polygonal scenes". In: *SIGGRAPH Asia'08: ACM SIGGRAPH Asia 2008 papers*, pp. 1–7, New York, NY, USA, 2008. ACM. doi: http://doi.acm.org/10.1145/1457515.1409107.

[42] LIDAL, E. M., HAUSER, H., VIOLA, I. "Design Principles for Cutaway Visualization of Geological Models". In: *Proceedings of the 28th Spring Conference on Computer Graphics*, SCCG '12, pp. 47–54, New York, NY, USA, 2013. ACM. ISBN: 978-1-4503-1977-5. doi: 10.1145/2448531.2448537. URL: <`http://doi.acm.org/10.1145/2448531.2448537`>.

[43] BRUCKNER, S., GRIMM, S., KANITSAR, A., et al. "Illustrative Context-Preserving Exploration of Volume Data", *IEEE Transactions on Visualization and Computer Graphics*, v. 12, n. 6, pp. 1559–1569, 11 2006. ISSN: 1077-2626. URL: <`http://www.cg.tuwien.ac.at/research/publications/2006/bruckner-2006-ICE/`>.

[44] WANG, L., ZHAO, Y., MUELLER, K., et al. "The magic volume lens: an interactive focus+context technique for volume rendering". In: *Visualization, 2005. VIS 05. IEEE*, pp. 367–374, Oct 2005. doi: 10.1109/VISUAL.2005.1532818.

[45] BRUCKNER, S., GRÖLLER, M. E. "VolumeShop An Interactive System for Direct Volume Illustration". In: C. T. Silva, E. Gröller, H. R. (Ed.), *Proceedings of IEEE Visualization 2005*, pp. 671–678", oct 2005. ISBN: 0780394623. URL: <`http://www.cg.tuwien.ac.at/research/publications/2005/bruckner-2005-VIS/`>.

[46] KRUGER, J., SCHNEIDER, J., WESTERMANN, R. "ClearView: An Interactive Context Preserving Hotspot Visualization Technique", *Visualization and Computer Graphics, IEEE Transactions on*, v. 12, n. 5, pp. 941–948, Sept 2006. ISSN: 1077-2626. doi: 10.1109/TVCG.2006.124.

[47] BURNS, M., HAIDACHER, M., WEIN, W. "Feature emphasis and contextual cutaways for multimodal medical visualization", *. . . on Visualization*, pp. 275–282, maio 2007. doi: 10.2312/VisSym/EuroVis07/275-282. URL: <`http://dl.acm.org/citation.cfm?id=2384179.2384223http://dl.acm.org/citation.cfm?id=2384223`>.

[48] BRUCKNER, S., GRIMM, S., KANITSAR, A., et al. "Illustrative Context-Preserving Volume Rendering". In: *Proceedings of EuroVis 2005*, pp. 69–76, may 2005.

[49] BIER, E. A., STONE, M. C., PIER, K., et al. "Toolglass and Magic Lenses: The See-through Interface". In: *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pp. 73–80, New York, NY, USA, 1993. ACM. ISBN: 0-89791-601-8. doi: 10.1145/166117.166126. URL: <http://doi.acm.org/10.1145/166117.166126>.

[50] VIOLA, I. *Importance-Driven Expressive Visualization.* Phd thesis, Vienna University of Technology, Favoritenstrasse 9-11/186, A-1040 Vienna, Austria, 2005. URL: <http://www.cg.tuwien.ac.at/research/publications/2005/phd-viola/http://www.ii.uib.no/vis/team/viola/>.

[51] BORGEFORS, G. "Distance transformations in digital images", *Computer Vision, Graphics, and Image Processing*, v. 34, n. 3, pp. 344 – 371, 1986. ISSN: 0734-189X. doi: 10.1016/S0734-189X(86)80047-0.

[52] KUBISCH, C., TIETJEN, C., PREIM, B. "GPU-based smart visibility techniques for tumor surgery planning", *International Journal of Computer Assisted Radiology and Surgery*, v. 5, n. 6, pp. 667–678, 2010. ISSN: 1861-6410. doi: 10.1007/s11548-010-0420-0. URL: <http://dx.doi.org/10.1007/s11548-010-0420-0>.

[53] FEINER, S. K., SELIGMANN, D. D. "Cutaways and ghosting: satisfying visibility constraints in dynamic 3D illustrations", *The Visual Computer*, v. 8, n. 5-6, pp. 292–302, set. 1992. ISSN: 0178-2789. doi: 10.1007/BF01897116. URL: <http://link.springer.com/10.1007/BF01897116>.

[54] SIGG, S., FUCHS, R., CARNECKY, R., et al. "Intelligent cutaway illustrations", *2012 IEEE Pacific Visualization Symposium*, pp. 185–192, fev. 2012. doi: 10.1109/PacificVis.2012.6183590.

[55] VIEGA, J., CONWAY, M. J., WILLIAMS, G., et al. "3D magic lenses". In: *Proceedings of the 9th annual ACM symposium on User interface software and technology - UIST '96*, pp. 51–58, New York, New York, USA, nov. 1996. ACM Press. ISBN: 0897917987. doi: 10.1145/237091.237098. URL: <http://dl.acm.org/citation.cfm?id=237091.237098>.

[56] COFFIN, C., HOLLERER, T. "Interactive Perspective Cut-away Views for General 3D Scenes". In: *Proceedings of the IEEE Conference on Virtual Reality*, VR '06, pp. 118–, Washington, DC, USA, 2006. IEEE Computer Society. ISBN: 1-4244-0224-7. doi: 10.1109/VR.2006.88. URL: <http://dx.doi.org/10.1109/VR.2006.88>.

[57] DIEPSTRATEN, J., WEISKOPF, D., ERTL, T. "Interactive Cutaway Illustrations", *Computer Graphics Forum*, v. 22, n. 3, pp. 523–53, 2003. doi: 10.1111/1467-8659.t01-3-00700.

[58] SELIGMANN, D. D., FEINER, S. "Automated Generation of Intent-based 3D Illustrations", *SIGGRAPH Comput. Graph.*, v. 25, n. 4, pp. 123–132, jul. 1991. ISSN: 0097-8930. doi: 10.1145/127719.122732. URL: <http://doi.acm.org/10.1145/127719.122732>.

[59] LI, W. *Interactive Illustrations for Visualizing Complex 3D Objects*. Phd thesis, University of Washington, 2007. URL: <http://www.wilmotli.com/>.

[60] LOECHEL, W. *Medical Illustration: A Guide for the Doctor-author and Exhibitor*. Springfield, C.C. Thomas, 1964. URL: <http://books.google.com.br/books?id=ID4JaAEACAAJ>.

[61] WOOD, P. *Scientific Illustration: A Guide to Biological, Zoological, and Medical Rendering Techniques, Design, Printing, and Display*. Design and Graphic Design Series. Hoboken, New Jersey, John Wiley & Sons, 1994. ISBN: 9780471285250. URL: <http://books.google.com.br/books?id=tHzjM4GmPdQC>.

[62] ZWEIFEL, F. *A Handbook of Biological Illustration*. Chicago Guides to Writing, Editing, and Publishing. Chicago, University of Chicago Press, 1988. ISBN: 9780226997018. URL: <http://books.google.com.br/books?id=nTAHk4zbnqwC>.

[63] HODGES, E. *The Guild Handbook of Scientific Illustration*. Hoboken, New Jersey, John Wiley & Sons, 2003. ISBN: 9780471360117. URL: <http://books.google.com.br/books?id=gD8pWkA6TcwC>.

[64] KIRSCH, F., DÖLLNER, J. "OpenCSG: a library for image-based CSG rendering". In: *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '05, pp. 49–49, Berkeley, CA, USA, 2005. USENIX Association. URL: <http://dl.acm.org/citation.cfm?id=1247360.1247409>.

[65] BURNS, M. *Efficient and Comprehensible Visualization of Complex 3-D Scenes*. Phd thesis, Princeton University, 2011. URL: <http://www.cs.princeton.edu/~mburns/http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Efficient+and+Comprehensible+Visualization+of+Complex+3-D+Scenes#0>.

[66] RONG, G. "Jump flooding in GPU with applications to Voronoi diagram and distance transform", *Proceedings of the 2006 symposium on Interactive*, p. 109, mar 2006. doi: 10.1145/1111411.1111431. URL: <http://dl.acm.org/citation.cfm?id=1111411.1111431http://www.comp.nus.edu.sg/~tants/jfa.html>.

[67] GUODONG, R. *Jump flooding algorithm on graphics hardware and its applications*. PhD thesis, National University of Singapore, 2008.

[68] PATEL, D., GIERTSEN, C., THURMOND, J., et al. "Illustrative Rendering of Seismic Data". In: Hendrik. Lensch, Bodo Rosenhahn, H. S. (Ed.), *Proceeding of Vision Modeling and Visualization 2007*, pp. 13–22, nov 2007.

[69] MARTINS, Z., DE CARVALHO, F., VITAL BRAZIL, E., et al. "Cutaway Applied to Corner Point Models". In: *SIBGRAPI 2012 - WGARI (Workshop on Industry Applications)*, aug 2012.

[70] ROPINSKI, T., STEINICKE, F., HINRICHS, K. H. "Visual Exploration of Seismic Volume Datasets", *Journal Proceedings of the 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision (WSCG06)*, v. 14, pp. 73–80, 2006. URL: <http://viscg.uni-muenster.de/publications/2006/RSH06a>.

[71] TURNER, A. "Challenges and trends for geological modelling and visualisation", *Bulletin of Engineering Geology and the Environment*, v. 65, pp. 109–127, 2006. ISSN: 1435-9529. URL: <http://dx.doi.org/10.1007/s10064-005-0015-0>. 10.1007/s10064-005-0015-0.

[72] POPINET, S. "GTS: GNU Triangulated Surface library". http://gts.sourceforge.net/, 2000–2006.

[73] CGAL. "CGAL, Computational Geometry Algorithms Library". 2014. http://www.cgal.org.

[74] BÆRENTZEN, A., NIELSEN, S. L., GJØL, M., et al. "Single-pass Wireframe Rendering". In: *ACM SIGGRAPH 2006 Sketches*, SIGGRAPH '06, New York, NY, USA, 2006. ACM. ISBN: 1-59593-364-6. doi: 10.1145/1179849.1180035.

[75] SHREINER, D., SELLERS, G., KESSENICH, J. M., et al. *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.3*. 8th ed. Boston, Addison-Wesley Professional, 2013. ISBN: 0321773039, 9780321773036.

[76] SOUSA, M. C., PRUSINKIEWICZ, P. "A few good lines: Suggestive drawing of 3d models", *Computer Graphics Forum (Proc. of EuroGraphics*, v. 22, pp. 2003, 2003.

[77] COLE, F., SANIK, K., DECARLO, D., et al. "How Well Do Line Drawings Depict Shape?" In: *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, pp. 1–9, New York, NY, USA, 2009. ACM. ISBN: 978-1-60558-726-4. doi: 10.1145/1576246.1531334. URL: <http://doi.acm.org/10.1145/1576246.1531334>.

[78] SHANMUGAM, P., ARIKAN, O. "Hardware Accelerated Ambient Occlusion Techniques on GPUs". In: *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, I3D '07, pp. 73–80, New York, NY, USA, 2007. ACM. ISBN: 978-1-59593-628-8. doi: 10.1145/1230100.1230113. URL: <http://doi.acm.org/10.1145/1230100.1230113>.

Appendix A

Source Code

Source code and video of the project available at:
https://code.google.com/p/irescutaway