Pontifícia Universidade Católica
do Rio de Janeiro

**Viviana Lorena Vargas Grajales**

# Image based simulation methods for depositional systems modeling

**Tese de Doutorado**

Thesis presented to the Programa de Pós–graduação em Matemática of PUC–Rio in partial fulfillment of the requirements for the degree of Doutor em Matemática.

Advisor: Prof. Sinesio Pesco

Rio de Janeiro
December 2017

**Viviana Lorena Vargas Grajales**

# Image based simulation methods for depositional systems modeling

Thesis presented to the Programa de Pós–graduação em Matemática of PUC–Rio in partial fulfillment of the requirements for the degree of Doutor em Matemática. Approved by the undersigned Examination Committee.

Rio de Janeiro, december 13, 2017

**Viviana Lorena Vargas Grajales**

The author completed her undergraduate studies in Mathematics in 2007 from Universidad del Valle - Colombia. The author also obtained the degree of Master in Mathematics in 2012 at the same institution.

Bibliographic data

Vargas Grajales, Viviana Lorena

Image based simulation methods for depositional systems modeling/ Viviana Lorena Vargas Grajales; advisor: Sinesio Pesco. — Rio de Janeiro: PUC–Rio, Departamento de Matemática, 2017.

v., 73 f.: il. ; 29,7 cm

1. Tese (doutorado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Matemática.

Inclui referências bibliográficas.

1. Matemática – Teses. 2. Geoestatística multiponto. 3. Imagem de treinamento. 4. Campo de direções. 5. Modelagem baseada em objetos. 6. Esqueleto.
I. Pesco, Sinesio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Matemática. III. Título.

CDD: 620.11

## Acknowledgements

First, I want to thank Department of Mathematics of PUC-Rio for opening me the doors.

I want to thank CNPq and Petrobras for the scholarship that supported me during the doctoral years.

I would like to thank and recognize the competence and dedication of the staff of the Mathematics Secretariat of PUC-Rio, for their help and cooperation throughout my study period.

I am deeply grateful to professor Sinesio Pesco for accepting me to do this doctoral thesis under his direction and for introducing me to these interesting subjects. He was always by my side helping me whenever I found myself lost.

To my parents and my brother for their unconditional support.

I want to thank my husband Mauricio Zamudio for his love and patience.

All my friends and colleagues, especially those who have accompanied me during these years in Brazil. I am grateful to Alexandro C., Antonio G., Clea A., Dania G., Erick C., Fiorella R., Gabrielle N., João S., Lilian B., Orlando F., Pablo F., Rafael S., Raimundo N., Sebastian P., Tamires P., Thayz P., Thiago D., Yunelsy N.

## Abstract

Vargas Grajales, Viviana Lorena; Pesco, Sinesio (Advisor). **Image based simulation methods for depositional systems modeling**. Rio de Janeiro, 2017. 73p. Tese de Doutorado — Departamento de Matemática, Pontifícia Universidade Católica do Rio de Janeiro.

In this work, we present two geostatistical methods to model geological structures that exhibit directional features in a tree structure, like fan deltas and turbidite channels. The first method is a multiple point geostatistical algorithm called directional field-based simulation (DIR-SIM). The directional feature of the training image is used to create a new object that we call training directional field (TDF), which contains the direction in each point of the image. This TDF represents the training image in a broader sense because both the training image and the direction followed by the reservoir are contained there. We propose to apply this object as a fundamental tool in the simulation. The second method is an object- based simulation called SKE-SIM which uses a training image to extract the distribution of selected parameters to build the turbidite channel system. The idea is based on the premise that the training image can be well represented by a one-dimensional object that we call, skeleton.

## Keywords

Multiple-Point geostatistics; Training image; Directional field; Object-based modeling; Skeleton.

## Resumo

Vargas Grajales, Viviana Lorena; Pesco, Sinesio. **Métodos de simulação baseados em imagem para modelagem de sistemas deposicionais**. Rio de Janeiro, 2017. 73p. Tese de Doutorado Departamento de Matemática, Pontifícia Universidade Católica do Rio de Janeiro.

Neste trabalho, apresentamos dois métodos geostatísticos para modelar estruturas geológicas que exibem características direcionais em uma estrutura de árvore, como leques deltaicos e canais turbidíticos. O primeiro método é um algoritmo geoestatístico multi-ponto chamado simulação baseada em campo de direções (DIR-SIM). A característica direcional da imagem de treinamento é usada para criar um novo objeto que chamamos de campo direcional de treinamento (TDF), que contém a direção em cada ponto da imagem. Este TDF representa a imagem de treinamento em um sentido mais amplo por que tanto a imagem de treinamento quanto a direção seguida pelo reservatório estão contidas nele. Propomos aplicar esse objeto como uma ferramenta fundamental na simulação. O segundo método é uma simulação baseada em objetos chamada SKE-SIM, que usa uma imagem de treinamento para extrair a distribução de parâmetros selecionados para construir o sistema de canais turbidíticos. A idéia baseia-se na premissa de que a imagem de treinamento pode ser bem representada por um objeto unidimensional que chamamos esqueleto.

## Palavras chave

Geoestatística multiponto; Imagem de treinamento; Campo de direções; Modelagem baseada em objetos; Esqueleto.

# Table of contents

1 Introduction **10**
Dissertation Outline 12

2 Theoretical framework **13**
2.1 Geostatistics 13
2.2 Object based modeling 14
2.3 Multiple point simulations (MPS) 15
2.4 Problem 18

3 DIR-SIM **22**
3.1 Training directional field 22
3.2 Directional field synthesis 27
3.3 Simulations 32
3.4 Conclusions 39

4 Skeleton based simulation **44**
4.1 Training Skeleton 44
4.2 Extracting information from the training skeleton 45
4.3 Skeleton definition 46
4.4 Skeleton synthesis 48
4.5 3D turbidite channels simulation 50
Single-valued B-spline curves modeling 53
Connecting B-Spline Curves 54
Object in the grid 55
Skeleton inside the lobe 56
3D Channel formation from the skeleton 57
Channel system in the lobe 58
4.6 Simulations 59
4.7 Conclusions 68

Bibliography **71**

# List of Figures

1.1 (a) The Wax Lake Delta. (b) The channel system highlighted in red is the training image. 10

2.1 (a) Object based simulation of FLUVSIM [5]. (b) Simulation parameters of the channel and levee [4]. 14
2.2 The training image is divided in ten regions based on its geometry [15]. 17
2.3 Comparison between 3 images to illustrate the notion of tree-like image. 20
2.4 Three directions are perceived, $v$ and $w$ in the channel development direction and $u$ in the width direction of the channel. 20
2.5 Application of the directional interval to define a local orientation. 21
2.6 Visual definition of the directional interval in a tree-like image. 21

3.1 Training Image. 22
3.2 Structuring element used to determine the contour. 24
3.3 Sequences $T_n$ and $C_n$. 25
3.4 (a) Two possible paths in the contour. (b) Two secants passing through point $p$ and $q_1$, $q_2$ respectively. 26
3.5 Steps of size 1 and 3 from point $p$. 27
3.6 (a) The TDF defined in the first element $C_0$ of the sequence $C_n$. (b) TDF defined in the contours $C_0, C_1, C_2$, the yellow points do not have a assigned direction yet. 28
3.7 Illustration of the template concept with radius 2, $T^2$. 29
3.8 Pattern database of the training directional field given by the figure 3.6. (b) using a template $T^2$. 29
3.9 Initialization and order of the simulation. 32
3.10 Training Image. 33
3.11 Sequence $C_n$ and $T_n$. 33
3.12 TDF of the training image 3.10 sampled in spaced points. 34
3.13 (a) Realizations with $\beta = 1$. (b) Realizations with $\beta = 0$. 34
3.14 Realizations for $\beta = 0.5$. 35
3.15 (a) Alluvial Fan, China. (b) The channel system highlighted in red. 36
3.16 Training image obtained from 3.15. 36
3.18 The training directional field of the image 3.16. 36
3.17 Erosion applied two times to the image 3.16. 37
3.19 Realizations obtained from the simulation using $\beta = 0.5$. 38
3.20 E-types (averages) over 30 realizations with $\beta = 0.5$. 39
3.21 Other simulation order and template. 40
3.22 (a) Training image. (b) Realization using Gabor filters to divide the simulation area. (c) Realization including the spatial location of the patterns into the similarity search. 41
3.23 Training image to be rotated and dilated for the simulation of the different regions given in the figure 3.24 42

3.24 (a) Realization using rotation. (b) Realization using both rotation and scaling. 43

4.1 (a) Channel system. (b) One dimensional approximation of the channels. 45
4.2 (a) Training image. (b) One-dimensional representation obtained by using FIJI. 45
4.3 Training skeleton obtained from the figure 4.2(b). 46
4.4 Bifurcation angles and lengths. 46
4.5 Illustration of the possible configurations of the nodes. 47
4.6 Skeleton bifurcation. 49
4.7 Depositional system model of turbidites. 51
4.8 Lobe parameters. 52
4.9 B-Spline curves in the planes $xy$ and $xz$. 53
4.10 Lobe parameters. 54
4.11 Control points to the 3 lobe depositional system. 54
4.12 Two quarters of ellipses forming $ellip_L(x)$. 55
4.13 Volume from the B-spline curves as the union of quarter of ellipses. 56
4.14 Simulation of a turbidite lobes [16]. 57
4.15 Skeleton built inside the region limited by the curves $B_{right}$ and $B_{left}$ in the plane $xy$. 57
4.16 Channel formation from one skeleton edge and a channel system limited by the B-spline curves. 58
4.17 (a) Training image. (b) Training skeleton. 59
4.18 Six skeletons obtained using the training skeleton 4.17(b) 60
4.19 One channel system simulation. 61
4.20 Three channels system simulation. 62
4.21 Three channels system simulation in one lobe. 63
4.22 (a) Training image. (b) Training skeleton. 64
4.23 Six skeletons obtained using the training skeleton 4.22(b) 64
4.24 3D realization from the skeleton in the figure 4.23(e) 65
4.25 3D realization of three channel systems limited by the B-spline curves. 66
4.26 3D realization of three channel systems inside its respective lobes. 67
4.27 3D realization of three channel systems inside its respective lobes. 68

# 1 Introduction

An important problem in geology is the modeling of certain type of reservoirs whose image representation cannot be assumed as the realization of a stationary point process, however it has a well-defined geometric structure, like fan deltas and turbidites deposits (as the presented in the figure 1.1(a)). In fact, frequently natural processes occurring in the earth can be better understood by non-stationary models.

The purpose of this work is the modeling of geologic structures, which spatial continuity can be assumed to be reflected by training images like the red channel system in the figure 1.1(b); these images present a particular geometry given by the orientation and width of the channels varying throughout the image; this kind of geometry will be called tree-like.

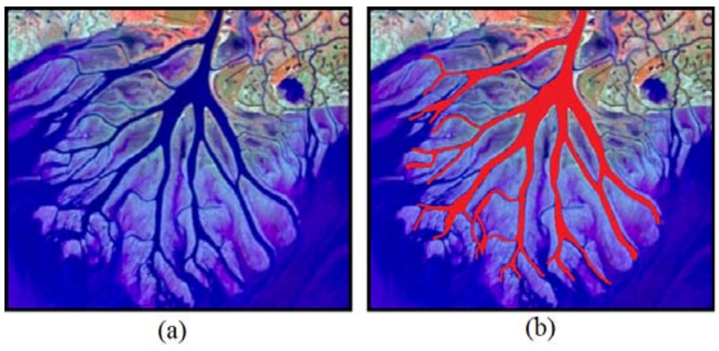


Figure 1.1: (a) The Wax Lake Delta. (b) The channel system highlighted in red is the training image.

In this thesis two simulation methods for modeling the aforementioned geological structures are proposed. The first method uses techniques from multi-point geostatistics. It is based on the fact that our training images allow us to determine a vector field representing them. The second method is object-based and it originates from the idea of interpreting tree-like training images as the thickening of a graph.

Multi-point geostatistics was developed to overcome difficulties that appeared in traditional methods, such as variogram-based. Traditional methods lack the capability of representing complex objects, as curvilinear and continuous structures. Multi-point methods depend on the concept of training image as source of spatial continuity; this training image is a conceptual image that is assumed to contain all possible structures of interest believed to appear in the geological body. In practice the training image is obtained from the realizations of an object-based model, but also may come from geologists sketches. The training image has been used to obtain statistics and conditional probabilities with the aim to reproduce them. A change of perspective is given when this probabilistic approach is replaced by pattern reproduction approach, where the training image is used as a source of patterns. Those patterns are located in the simulation area trying to provide consistency between them.

One of the main characteristics of tree-like images is their directional feature. This information was used in previous works: in [1] the training image is partitioned in regions where it is assumed that a stationary model can be applied, this is done visually, and only one direction is associated with each area. This direction is employed to obtain the training image used in the stationary simulation at the area. In [2], also the training image is divided in regions and stationary simulation is applied in each one. Here the process to divide the image is automated and done using Gabor filters. This uses the directional feature of the image.

In our first method DIR-SIM, directions are not defined regionally but pointwise, a vector field that represents the flow on each point is obtained, it incorporates local orientation information and represents the training image in a broader sense.

The vector field associated to the training image, called training directional field (TDF), is the main novelty in the first algorithm proposed in this dissertation. The TDF is used to control the simulation such that the realizations obtained have a similar directional pattern to that of the training image.

The second simulation proposed, SKE-SIM, is object-based. This type of geostatistical method is applied to geological structures that can be described by parametric geometries. The geological structure is represented by a collection of well-defined geometric objects. Such geometric objects are defined by parameters deduced from the available information. Then, the simulation proceeds sequentially creating the objects and placing them in the simulation grid until a criteria is attained.

The objective of SKE-SIM is to build a 3D model of the channel system using information extracted from a tree-like image (like the one in figure 1.1(b)). The 3D channel system is assumed to be located in the half-space $\{z \leq 0\}$ and the training image is thought as its projections to the plane $xy$. The channel system is interpreted as a thickening and deepening of a unidimensional object, called skeleton. The training image is used to define probability distributions from which the parameter values of the skeleton are sampled.

The main idea behind this method is that the unidimensional representation of the channel system, simplifies the simulation and makes it fast. In addition, the geometrical representation allows the introduction of more information into the modeled reservoir, like petrophysical properties.

## Dissertation Outline

This thesis is organized as follows:

Chapter 2 presents the theoretical framework of the problem. In the last section, the problem dealt with in this thesis is explained in more detail.

Chapter 3 proposes a non-stationary multiple point method called DIR-SIM (directional simulation). It introduces the concept of training directional field, which is used to guide the simulation.

Chapter 4 presents a 3D object-based modeling. A training image is represented as a simple linear structure (the skeleton). This structure is used to find parameter values for the modeled object. Based on these parameters new skeletons are generated, they are turned into 3D objects and placed in a lobe.

# 2
# Theoretical framework

## 2.1
## Geostatistics

Geostatistics is a set of tools to model and predict a spatial-temporal natural phenomena. Measurements are expensive and time consuming, so the phenomena is usually characterized by taking only some samples at different locations. Geostatistics methods provides estimates for locations where samples were not taken, together with the uncertainty of that predictions [3].

A geostatistical model generates multiple equiprobable scenarios representing the same spatial continuity supposed in an unknown real phenomena. The model should cover the space of uncertainty and satisfy the available data measured locally.

Geostatistics is widely applied in many areas, for example in:

Permeability or porosity estimation in oil reservoirs.

The mining industry to quantify mineral resources.

Meteorology to predict temperatures, rainfall, etc.

Currently, geostatistics techniques are used commonly for the modeling of petroleum reservoirs. Multiple realizations are simulated considering different sources of data as well logs, cores, seismic information, geologic interpretations, etc.

The data available to model the system is limited, for example, the porosity and permeability are only obtained at sparse drilled well locations.

In addition, the geologic processes are heterogeneous, i.e. strongly varying in space and in time. This causes a considerable uncertainty or variability in the modeling results.

Traditionally, this problem can be attacked by geostatistical methods such as variogram, kriging or indicator simulation, interpolating data.

But in particular, we are interested in object based and multipoint geostatistical methodologies which are briefly explained in the following sections.

## 2.2 Object based modeling

Object models (also called boolean models) generate spatial distributions of objects, like channels, rock types, turbidite lobes, etc. in the simulation region. The simulation consists of placing the entire objects directly on the grid.

The main requirements for this type of methods are the geometry parametrization and geometry placement [3]. The first consists in the description of the object by parametric geometries, that means, the object can be characterized by simple geometries depending on parameters. These parameters (for example sinuosity, depth, length, width, etc.) must be properly chosen according to the available information as seismic data, outcrops, etc. The second requirement, geometry placement, refers to the disposition of the objects created on the grid. They are randomly placed until a established criteria is reached, for example a global proportion.

In the figure 2.1(a), one simulation using the algorithm FLUVSIM [4] is shown. It is an object based model of fluvial systems. The cross sections of channels and levees together with the parameters used, are presented in the figure 2.1(b). In addition, the sinuosity of the channel is another parameter in the modeling.

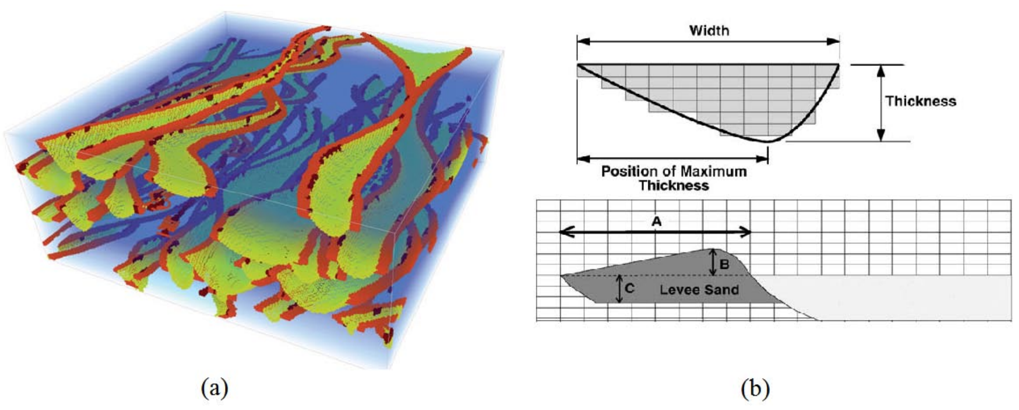


Figure 2.1: (a) Object based simulation of FLUVSIM [5]. (b) Simulation parameters of the channel and levee [4].

Another example of object based modeling is the algorithm PETBOOL [6], [7], which has been used to model different reservoirs in Brazil (fluvial channels and lobes in the Açu and formation at the Potiguar basin, channelized turbidites in the Espirito Santo basin etc.). This algorithm uses as basic geological objects: plane beds, wedges, ellipsoids, lobes, sigmoids, channels, turbidite channels and dunes. The simulation can be constrained by wells, global object density and interval object density given by a vertical facies proportion curve.

One of the main ideas supporting the object-based method presented in this dissertation is the extraction of information from a training image, and the use of this information for establishing the value of parameters. This way of thinking is also explored in the work [8].

The paper [8] presents an object-based simulation of reservoirs in turbidite channel systems. These systems are constituted by sets of channels with meander architecture. The algorithm uses a training image to establish the azimuth angles, width and thickness in the modeled object. In a particular case studied, the training image was obtained by analyzing 3D seismic images of the reservoir area (turbidite channel system in the Lower Congo basin, offshore Angola, on the west Africa). A uniform distribution was used to generate the width and thickness of the channels. In addition, Gaussian distribution laws for porosity and permeability were adopted.

## 2.3 Multiple point simulations (MPS)

Multiple point simulation (MPS) methods are geostatistical techniques with the ability to reproduce the connectivity of many locations, because they can account for correlations between more than two positions at a time. Thus, they can reproduce complex geological structures as sinuous geometries, that two-point statistics cannot model.

MPS borrows multiple-point statistics from a conceptual geological model called *training image*, that represents the prior geological knowledge of the modeled phenomena.

Training images are explicit examples of the heterogeneity that is believed to be presented in a natural phenomena. They can be obtained from different sources such as interpreted outcrop photographs, geologist s sketch, etc. In practice, the main source are object-based simulations. Training images does not need to be conditioned to any specific data, they need only to reflect a prior geological concept.

Several approaches using training images have been developed, which handle patterns in different ways of defining, storing or searching them. Some simulations are focused on the calculation of probabilities from the patterns and others have an image-construction approach.

In [9], a pixel-based (one pixel is simulated each time) method called ENESIM is presented. If the node $x$ is being simulated, and $N_x$ is a neighborhood, then all the occurrences of $N_x$ in the training image are found. Then, the conditional distribution is determined as the histogram of the central node values for all the occurrences of $N_x$. A value is sampled from the distribution and it is assigned to the node $x$. This method is highly CPU demanding because it directly scans the complete training image for each node.

To overcome the drawback of ENESIM, Strebelle proposed an extension of this work called Single Normal Equation Simulation (SNESIM) [10]. It is considered the first computationally-efficient MPS method [11]. In this method, a training image is scanned to obtain the conditional probabilities. Before starting the simulation a search tree storing these probabilities is constructed. To simulate an uninformed node, according to the valued already simulated in a neighborhood of the node, a probability distribution is retrieved from the search tree, and a value is sampled from it.

The first methods proposed in MPS determine the value of every single pixel sequentially, i.e. pixel-based methods. This kind of methods are flexible but computationally demanding. To overcome this drawback, it was developed a second family of methods, MPS pattern-based, i.e. simulating a group of pixels simultaneously.

In [1], the SIMPAT method is presented. It creates a pattern database $B$ from the training image and defines a random simulation path in the grid. Sequentially, each node in the path is visited. Then for each position $x$, a neighborhood $N_x$ formed by the already simulated nodes is extracted. $N_x$ is compared with the patterns in $B$, looking for the most similar, using a similarity function. The found patterns is placed in the simulation grid in the position $x$.

Due to the search of the most similar pattern, SIMPAT is CPU demanding. The FILTERSIM algorithm [12], proposed a solution to this efficiency problem. The patterns are classified in classes using selected filter statistics. So the simulation proceeds to determining the pattern class most similar to the data event. Then, randomly one pattern from the corresponding class is chosen and placed in the simulation region. Similar to FILTERSIM, the algorithm DISPAT presented in [13], proposes an improvement to the search of a similar pattern, in this case the filters are replaced by kernel k-means applied on an MDS transform of the TI patterns. Further developments in this line include the method LSHSIM [14], where locality sensitive hashing together with a compression technique is applied in order to increase the efficiency of the algorithm.

The previous works are all stationary models. In our work, we want to model geological structures typically represented by images of a non-stationary process. Some non-stationary elements are required in this type of model.

One approach is partitioning the training image in regions where a stationary modeling can be applied. In each region a different training image could be used.

In Arpat [15], the training image is divided in regions observing the morphology of the image. For example, the variation in the inclination or the density defines different regions in the model. Considering this two characteristics in the figure 2.2 a training image is divided in ten regions. Then, each region is associated with a different training image containing patterns considered representative of that region. Another option would be to use only one training image that is transformed (by rotations and dilations) depending on the features of each region. Then, the algorithm is applied in each region in a stationary way.

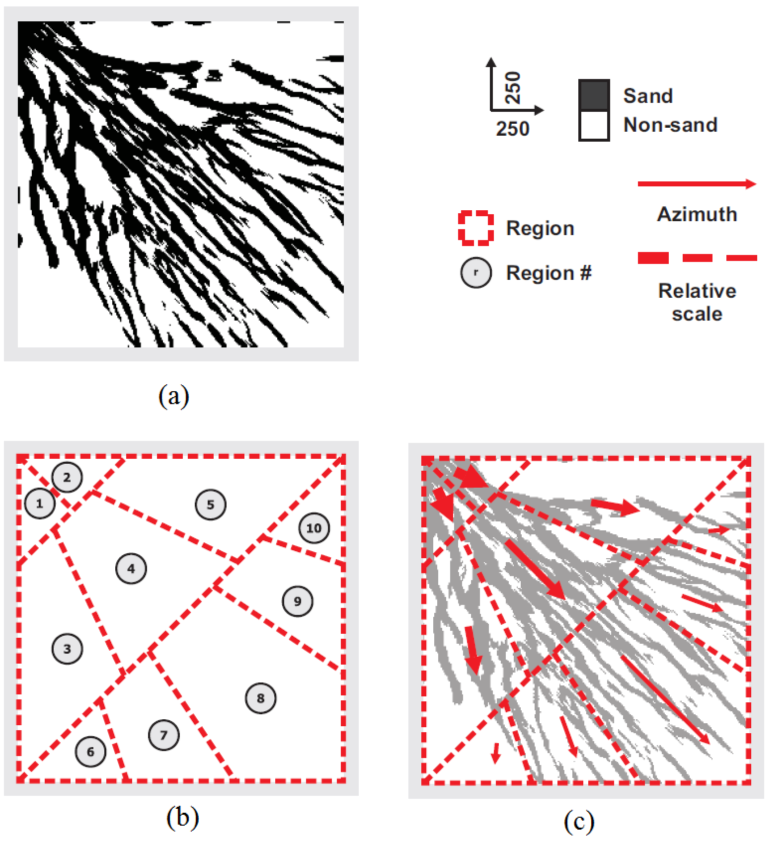


Figure 2.2: The training image is divided in ten regions based on its geometry [15].

In [2], Gabor filters are used to do an automatic segmentation of the training image into regions where the stationary model is applied. Then, each region is simulated using the segmented region as source of patterns. In this same paper another method is presented. In it, the simulation is performed following the usual stationary algorithm but a new element is introduced when computing the similarity between patterns. A spatial coordinate with the pattern position is employed in the similarity calculations. This change makes the algorithm non-stationary.

In this dissertation two methods for the modeling of deltas, turbidite channels or structures with a similar geometry, are proposed. Those structures are represented by training images having a particular geometry called tree-like. One of the methods is an object-based model and the other is a non-stationary multiple point simulation. In the next subsection, the problem dealt in this dissertation and the definition of tree-like is explained.

## 2.4 Problem

Turbidite deposits are generated by turbidity currents and related gravity flows. These currents are caused by catastrophic events like a storm surges, shocks induced by earthquakes, failures of sediments due to steep slopes, river floods, etc. Turbidity currents move huge amounts of sand, mud and pebbles forming turbidites in oceans, lakes either in shallow or deep waters.

Turbidite reservoirs are distinguished by a complex structure of sand bodies arranged in channels and lobes deposited in deep water environments. This kind of reservoirs still represent an important source of oil exploration in Brazil. Due to the high investment made to exploit these types of reservoirs, the production has to be high enough to compensate. The cost of drilling a single well can easily exceed 100 million dollars and the success rates are around 15 to 30 percent, then the risk involved in the exploitation must be determined.

It is necessary an adequate representation of these reservoirs, so the main objective of this work is to present new methodologies for modeling the depositional architecture of turbidite channels, creating different scenarios that are equally likely and realistic.

Two methods are presented in this work, the first is called directional simulation DIR-SIM, it is a multiple point simulation method and the other is an object based method, called skeleton simulation SKE-SIM. Both methods use, in different ways, an image that is an example of the distribution and continuity believed to be presented in the turbidite channels.

The turbidite channel system that will be modeled is represented by binary 2D images. The black color in a certain position means the object occupies that position and white means the contrary, for example in the figure 2.3(b), a black point represents sand in a channel and white no sand. Because of the particular structure of the modeled object, this work is restricted to images with a special geometry, that we call tree-like. We have not formal definition, but tree-like images should have the following features:

1. At each black point, one direction in which the channel appears to be developing can be perceived. In the figure2.3(a), if a black point is taken in the image, there is no distinguishable direction in which the object is being formed, it is not a tree-like image. Both images in the figure 2.3(b and c) meet this condition.

2. The tree-like image has a directional interval. If a point $p$ is taken in a tree-like image, tree directions stand out, see figure 2.4. In the direction $u$ the point gets away from the object in a short distance, because in this direction the channel is crossed through its width. In directions $w$ and $v$ the point $p$ tends to continue for a longer distance within the object. In fact these two directions give the idea of the tendency of the channel to develop. But one of them goes against the flow of the channel. To decide which one, the small region where the point $p$ is analyzed, has to be seen as part of to the entire channel system, see figure 2.5. Observing the general structure of the image, $v$ can be chosen like the appropriate direction in the point $p$.

3. Like a tree, the body has a starting region $O$ from which branches are born, ramify and continues. In the figure 2.3(c) the source is located in the upper-right corner. On the contrary in figure 2.3(a) there is not any starting region, and figure 2.3(b) has a lot of starting regions.

The image given by figure 2.3(b) is not a tree-like however it verifies property (1). Hence at each point there are two possible directions in which the channel develops. In fact, we can use the algorithm described in chapter 3 to define a directional field for this image. Then DIR-SIM method is not restricted only on tree-like image, but the second algorithm SKE-SIM, in chapter 4 applies only for these kind of images.

The *directional interval* is defined as an interval containing the directions that are presented in the tree-like image. It can be established visually, for example in figure 2.6 the directions in the tree-like are inside the red cone. Then the directional interval is the interval determined by the opening angle of the cone and its position.

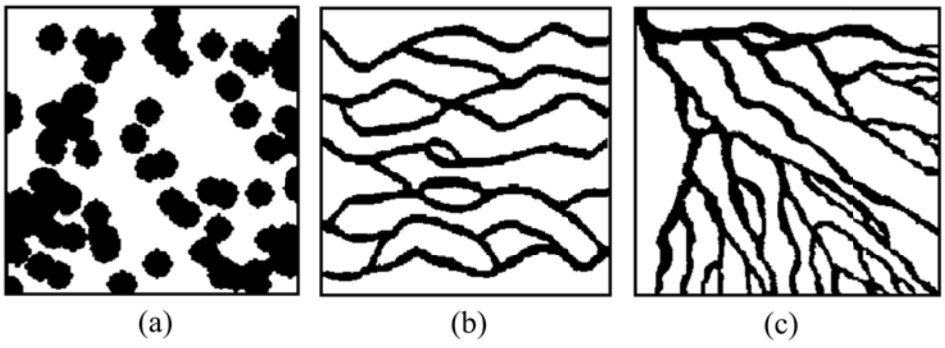


Figure 2.3: Comparison between 3 images to illustrate the notion of tree-like image.

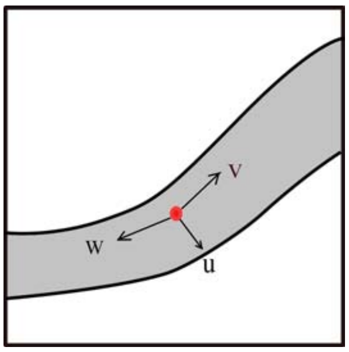


Figure 2.4: Three directions are perceived, $v$ and $w$ in the channel development direction and $u$ in the width direction of the channel.

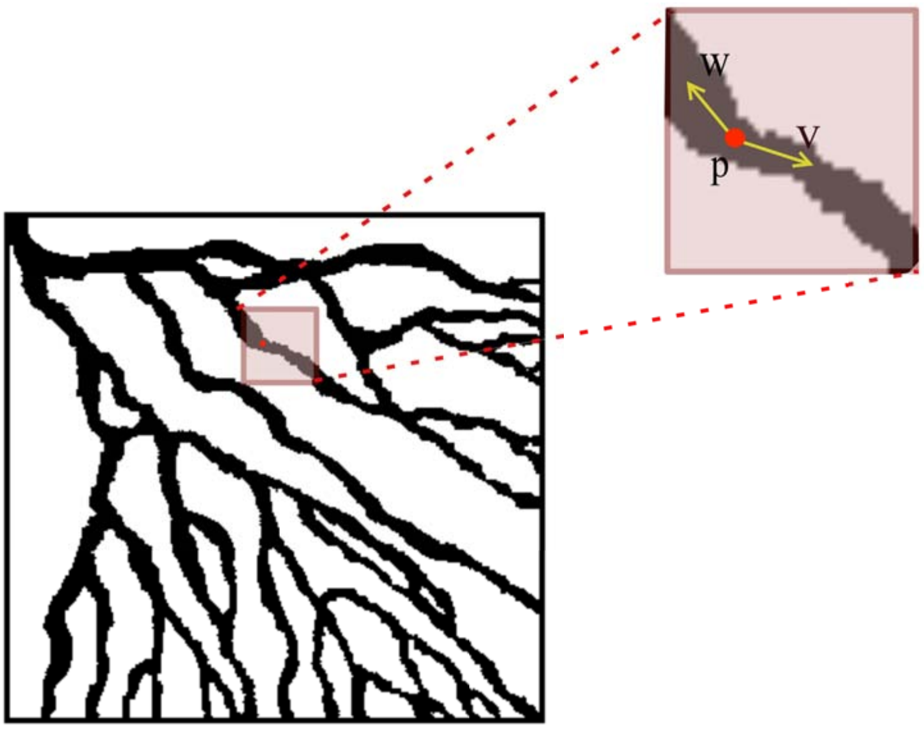


Figure 2.5: Application of the directional interval to define a local orientation.

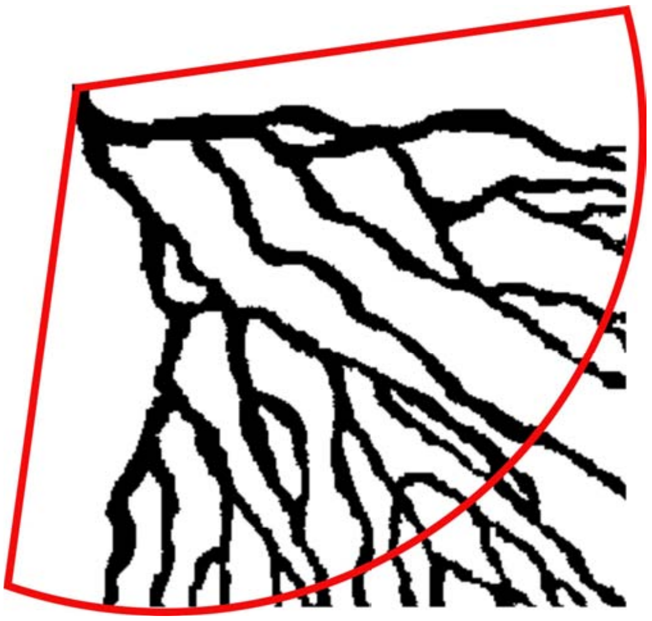


Figure 2.6: Visual definition of the directional interval in a tree-like image.

# 3
# Directional Field-based Simulation of Non-Stationary Geostatistical Models

In this chapter it is proposed a multipoint-geostatistics method to non-stationary models using tree-like training images. The proposed method is divided in two parts. In the first part, due to the directional characteristics of the training images used, a directional field associated to the training image is obtained defining a direction at each point within the body. To illustrate this step, the simple binary training image given by the figure 3.1, will be used. In the second part, this field is employed to guide the simulation algorithm using directions in a context of similarity between patterns.

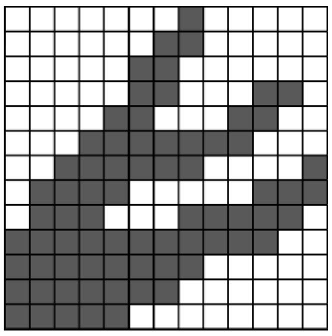


Figure 3.1: Training Image.

## 3.1
## Training directional field

From the training image **TI**, given by figure 3.1, $ti$ is defined as the function that assigns the value of the training image **TI** to the position $p = (i, j)$ in image. Since this work is restricted to binary images, the function $ti$ is defined by:

$$ti(p) = \begin{cases} 1, & \text{if } \mathbf{TI} \text{ contains sand in p} \\ 0, & \text{if } \mathbf{TI} \text{ contains no sand in p} \end{cases}$$

This training image characterization shows if one point is inside the channel system or not. However, the tree-like training images present directional features that allow to describe it in a broader way, considering also one direction to each point in the reservoir. Motivated by this directional property, a new object called **training directional eld (TDF)** is determined, containing the directions over the reservoir in the training image. To each black point in the **TI** one direction is assigned. This direction will be defined by a vector of norm 1, this will be represented by the angle between the vector and the positive $x$ axis. To represent the **TDF** it is used a function $tdf$ given by:

$$tdf(p) = \begin{cases} \alpha_p, & \text{if } ti(p) = 1 \\ \text{ND}, & \text{if } ti(p) = 0 \end{cases}$$

Where $\alpha_p$ is a direction that represents how the channels is developing at the point $p$. It will be defined in the algorithm presented in section 3.1.2. At points outside of the channel system there is not a defined direction, this is denoted by **ND**. As it will be seen below, the value $tdf(p)$ is defined as an approximation of the slope of the tangent to the image contour at point $p$. If the body is a geological structure formed by the deposition of material, like turbidite deposits, then this directional field can be interpreted as the set of directions followed by this material when the structure was formed.

The first step of DIR-SIM is the construction of the TDF, which is mainly based on two things: (1) the training image contour and (2) the directional interval in the image. The contour, is a curve on the body that limits it, i.e. where there is a change from being inside or outside the body. The directional interval is the range of values that can take the directions, for example $\left[-\frac{\pi}{4}, \frac{\pi}{4}\right]$. This allows to determine the appropriated direction at each point, as was explained in the section 2.4 when defining tree-like.

## 3.1.1 Image decomposition

The algorithm starts with a tree-like training image $T_0$ (figure 3.1). Let $p$ in $T_0$ be a point such that $ti(p) = 1$. $p$ will be defined as a contour point if, when centering the structure given by figure 3.2 on it, $ti(u_j) = 0$ for one or more the positions $u_1, u_2, u_3$ and $u_4$, i.e. $p$ has at least one white point in the neighborhood taken. The contour $C_0$ is a new image from $T_0$ defined by

$C_0(p) = 1$ if $p$ is a contour point of $T_0$ and $C_0(p) = 0$ otherwise. Figure 3.3(b) shows the contour $C_0$ of the training image $T_0$.

Now, it is generated a new image $T_1$ formed by the black points in $T_0$ that are not on its contour $C_0$, so $T_1$ is the difference $T_0 - C_0$ (Figure 3.3(c))

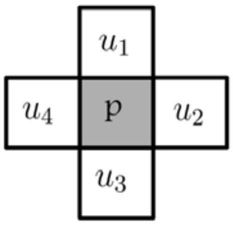


Figure 3.2: Structuring element used to determine the contour.

The previous process is an erosion of a binary image $T_0$, using the structuring element, which determines the result of erosion on the image. The basic effect is to erode away the boundaries of regions of foreground pixels (black pixels). Thus areas of foreground pixels shrink in size, and holes within those areas become larger.

Then the erosion is applied to the image $T_1$. This process, is repeated successively generating two images sequences, one with the contours $C_0, C_1, \ldots, C_k$ and another with $T_0, T_1, \ldots, T_k$, see the figure 3.3. Since $T_{n+1} = T_n - C_n$ is strictly smaller than $T_n$, the body in the sequence $T_n$ is becoming increasingly smaller at each step.

It must be decided how many times erosion is applied. There are multiple criteria that can be used to stop the process. For example when the number of points in the eroded image $T_n$ is less than a certain rate of the total points in the initial image. In addition, the number of connected components can increase for each erosion, then other alternative is to observe the number of connected components of the image $T_n$ and base on that, one decides when to stop.

The training image $T_0$ is described as the following union

$$T_0 = C_0 \cup C_1 \cup \ldots C_{k-1} \cup T_k \tag{3.1}$$

in the explicit example given by the figure 3.3, $T_0 = C_0 \cup C_1 \cup T_2$. Determining TDF is based on the fact that $T_0$ can be described as the union given by (3.1).

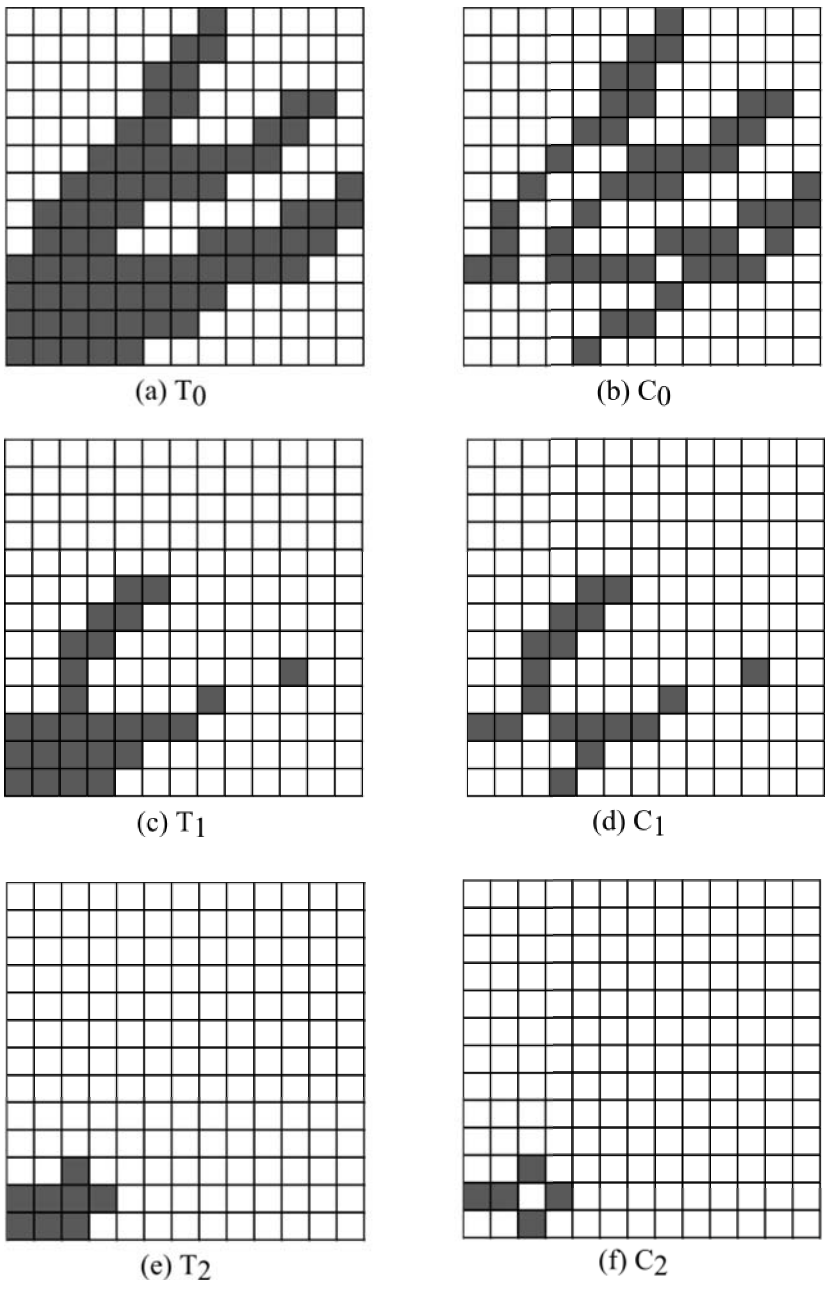


Figure 3.3: Sequences $T_n$ and $C_n$.

## 3.1.2 Directions on the contour

The direction at point $p \in T_0$ is defined using the contour to which it belongs, because in contour points it is possible to define the directions that the point must follow to continue within the contour.

Given a point $p$ in the contour, it has two possible paths to follow in the contour, see figure 3.4(a), and the selection is made according to the directional interval of the tree-like image. The idea is to give at a point $p \in T_0$ a value for the slope of the tangent line to the contour curve that passes through that point.

To determine the tangent to a curve at the point $p$, it is considered the secant lines passing through $p$ and nearby points $q$ (figure 3.4(b)). The tangent is obtained at the limit when the point $q$ tends to $p$. Since the contour is discrete, the limit when $q$ tends to $p$ cannot be taken. The tangent will be computed using two secants, as will be explained in the next paragraphs.

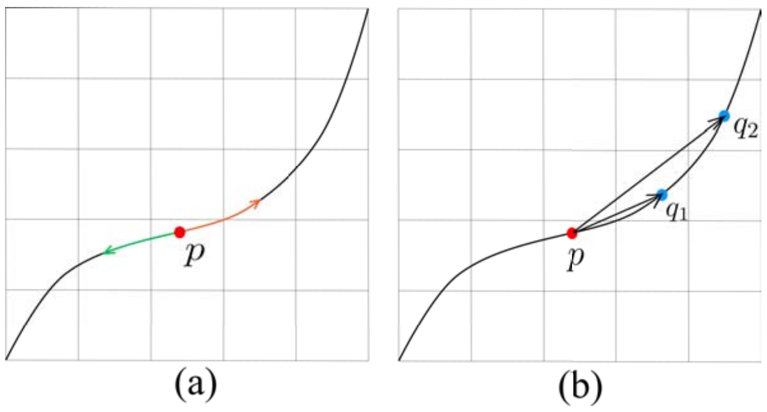


Figure 3.4: (a) Two possible paths in the contour. (b) Two secants passing through point $p$ and $q_1$, $q_2$ respectively.

To take a step of size 1 from the point $p$ on the contour is to choose one neighbor point $q$ ($q$ is immediately next to $p$), which is on the contour and meets the established directional interval, i.e. the direction of the segment $\overline{pq}$ is inside the interval defined by the directional interval. If more than one point satisfies the previous condition then one of them is randomly selected. A step of size $n$ is to successively take $n$ steps of size 1.

To construct the TDF, two step sizes $n$ and $m$ are fixed. By taking these steps from a point $p$, two final positions are reached, points $q_n$ and $q_m$. The direction $\alpha_i$ of the secant passing through $p$ and $q_i$ is defined as the angle between the line $\overline{pq_i}$ and x axis, for $i = n, m$. The direction at point $p$ is defined as the average of these two directions. Then the function *tdf* which determines TDF, at point $p$ is given by:

$$tdf(p) = \frac{\alpha_n + \alpha_m}{2}$$

In the figure 3.5 it can be observed the two points $q_1$ and $q_3$ taking steps of size 1 and 3 respectively from the point $p$. The arrows indicate the direction, $\alpha_1$ and $\alpha_3$ the angle formed with $x$ axis. It also can be noted that for points as the point $r$ in the image 3.5 no steps from them can be taken, in this case the definition of the direction is reserved for later and will be based on the directions of points prior to it.

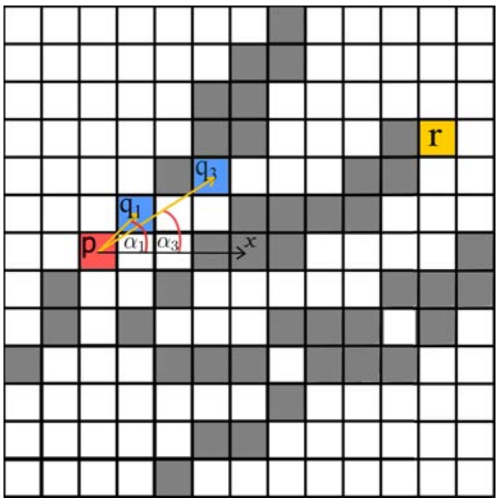


Figure 3.5: Steps of size 1 and 3 from point $p$.

In the figure 3.6(a), the set of directions determined in the first element of the contour sequence, $C_0$, are shown. Yellow positions have no following point and thus do not have a established direction yet.

The TDF is defined in the points of the successive contours $C_n$. However, it remains to assign the directions to the interior points, $T_k$, and those that belong to some $C_n$ but do not have a direction yet. For them, the TDF is defined by the average of the known directions, in a convenient neighborhood (usually one or two points of distance). This process continuous until every point has a direction defined.

The TDF in the successive contours $C_0, C_1, C_2$, given in the figure 3.3(b, c, d) is shown in the figure 3.6(b). The yellow points represent the points in $T_3 = T_2 - C_2$ together with the points in the contours whose do not have a direction yet. A red template is centered in these points. The direction in a yellow point is defined as the average of the known directions inside the template.

The previous method to find the TDF can be summarized in the following algorithm.

## 3.2 Directional field synthesis

This section describes the steps of the simulation algorithm, called directional simulation **DIR-SIM**. The algorithm can be described in two basic steps: (1) The TDF is preprocessed to extract and collect the patterns. (2)

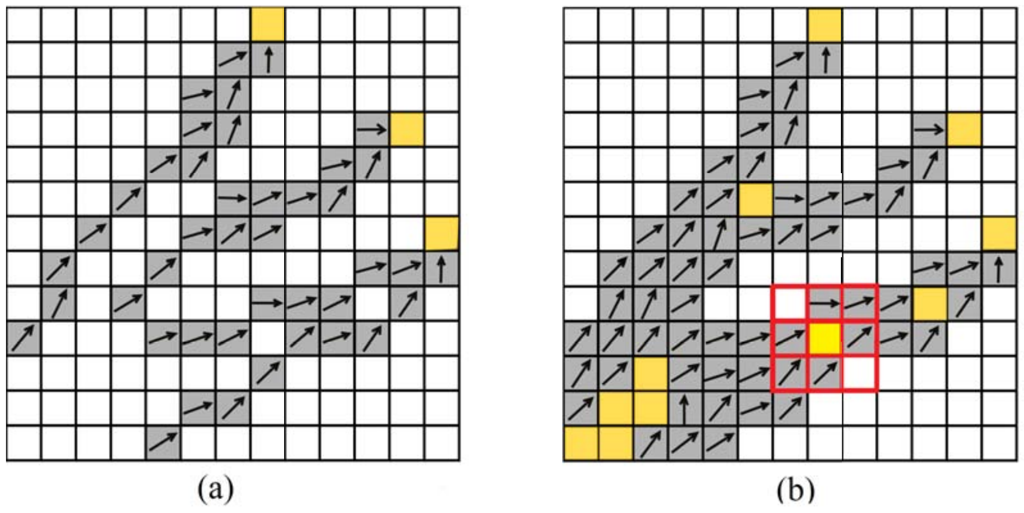


Figure 3.6: (a) The TDF defined in the first element $C_0$ of the sequence $C_n$. (b) TDF defined in the contours $C_0, C_1, C_2$, the yellow points do not have a assigned direction yet.

**Algorithm 1** TDF generation

```
1: Generate the contour $C_0$ of the training image $T_0$.
2: A new image $T_1 = T_0 - C_0$ is obtained.
3: Keep repeating (1),(2) to get $C_i$ and $T_{i+1}$ for each step $i$.
4: Find the direction at each contour point in the sequence $C_n$, using steps
   of size $n$ and $m$.
5: For all points where no direction has been defined, it is found by
   interpolation of known neighbor values.
```

The patterns are used in the sequential simulation to simulate one point each time. This is a MPS method, but instead of a training image it uses a training directional field as source of information. The TDF contains the training image data and additionally the directional information.

## 3.2.1 Pattern database

A template $T$ is defined as a subset of $\mathbb{Z} \times \mathbb{Z}$ that contains the center of the template (the pair $(0,0)$). Because of the simulation order, it will be used templates with L-form and radius $r$ given by:

$$T^r = \{(i,0) : -r \leq i \leq 0\} \cup \{(i,j) : -r \leq i \leq r, -r \leq j \leq -1\}$$

The figure 3.7 illustrates a template with radius 2, represented by the red points.

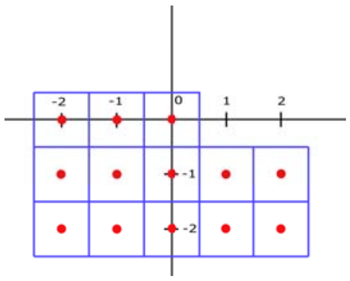


Figure 3.7: Illustration of the template concept with radius 2, $T^2$.

When the template $T^r$ is centered at a point $p$ in the TDF, it defines a set of directions called the pattern with center $p$. The function $Pat_p^{T^r}$ is used to described the pattern, it is given by:

$$\begin{aligned} Pat_p^{T^r} : T^r &\to \mathbb{R} \\ x &\to tdf(x+p) \end{aligned}$$

The TDF is scanned using the template $T^r$, that means the template is centered at each point $p$ in the TDF, obtaining the patterns $Pat_p^{T^r}$ that are stored in a database $PatBase$. The figure 3.8 presents all the patterns of the TDF given by the figure 3.6(b). As it can be seen in the figure 3.8, the patterns are defined at the points where the translated template to $p$ is completely contained in the TDF.

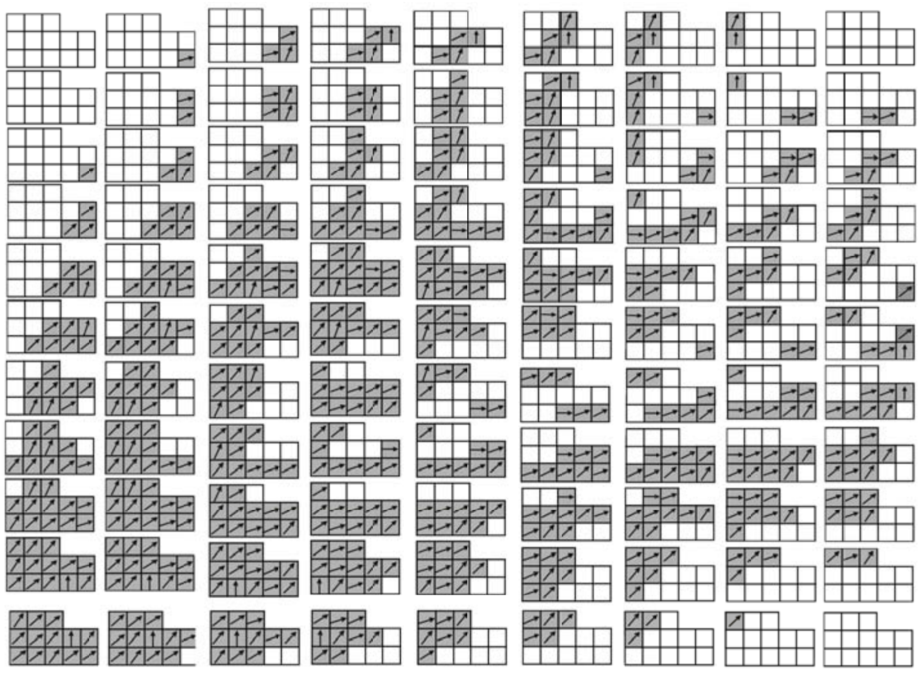


Figure 3.8: Pattern database of the training directional field given by the figure 3.6. (b) using a template $T^2$.

The type of training image dealt in this dissertation presents characteristics that differ between regions. If regions in different locations are observed in the image, the inclination and thickness of the channels can be considerably no similar. Then, the patterns are not independent of their location. For these reason together with the pattern $Pat_p^{Tr}$, the position of the point $p(x, y)$ is stored.

### 3.2.2 Comparison criterion

Given two patterns, it is defined a measure of similarity between them. Although squared Euclidean distance is not a metric, because it does not meet the triangular inequality, it will be used because only comparisons between distances are considered.

The similarity, $d_{tdf}$, between the pattern centered at $p$ and the pattern centered at $q$, both with template $T_r$, is defined by

$$d_{tdf} = \underset{x \quad T_r}{\phantom{\sum}} (Pat_p^{T_r}(x) - Pat_q^{T_r}(x))^2 \tag{3.2}$$

As can be seen in the formula above, the distance $d_{tdf}$ is only a summation of the angles differences between two patterns. However, since not all points in the training image have an assigned direction, then it must be specified what happens when $tdf(u) = \mathrm{ND}$. Below is shown how it is calculated the difference $Pat_p^{T_r}(x) - Pat_q^{T_r}(x)$.

$$Pat_p^{T_r}(x) - Pat_q^{T_r}(x) = \begin{matrix} tdf(x+p) - tdf(x+q), & \text{if } Pat_p^{T_r}(x) = \ \mathrm{ND} \\ & \text{and } Pat_q^{T_r}(x) = \ \mathrm{ND} \\ 0, & \text{if } Pat_p^{T_r}(x) = \ \mathrm{ND} \\ & \text{and } Pat_q^{T_r}(x) = \ \mathrm{ND} \\ \frac{\pi}{b}, & \text{otherwise} \end{matrix}$$

In the case when a position in one pattern has a defined direction but the corresponding position in the other pattern does not, i.e. $Pat_p^{T_r}(x) = \mathrm{ND}$ and $Pat_q^{T_r}(x) = \mathrm{ND}$, the assigned value is $\frac{\pi}{b}$. The value of the parameter $b$ depends on the directional interval adopted. For example, it can be chosen $b$ such that $\frac{\pi}{b}$ corresponds to the length of the directional interval interval, this makes sense because it is the maximum value that the difference between two directions can take.

Remember that in the pattern database *PatBase* together with the pattern, it was also stored its position. Using this position, it is defined the local distance between the patterns $Pat_p^{Tr}$ and $Pat_q^{Tr}$ as the squared Euclidean distance of the points $p$ and $q$, this will be denoted by $d_{loc}$:

$$d_{loc} = (p_x - q_x)^2 + (p_y - q_y)^2$$

Given the structure of the training image, the comparison between two patterns used in the algorithm will not depend only on the similarity between them but also in their positions in the image. Then, it is defined the total distance $d_{total}$ that considers both the distance in terms of its angles, i.e. the similarity between patterns, and the distance of their locations. It is defined as the convex combination of these two distances:

$$d_{total} = \beta d_{tdf} + (1 - \beta) d_{loc}$$

where $\beta \in [0, 1]$. The value of the weight $\beta$ controls the amount of non-stationarity in the simulation. If $\beta = 0$, only the location is considered then the modeling is totally not stationary and the realization is very similar with the training directional field. When $\beta = 1$, the pattern positions are not taken into account, then the modeling is stationary. Hence, when a result of the simulation is considered it will be observed that it has similar features at different regions in the image.

### 3.2.3 Simulation Algorithm

The algorithm starts with an empty simulation region $R$. To initialize the simulation, if the radius of the template used is $r$, then the *tdf* values of the first $r$ columns on the left and the $r$ lower rows are pasted in $R$; this values are used as a seed.

The model simulates one point each time, following a unilateral path, from bottom to top and left to right starting on the left bottom corner. The first point to be simulated and the simulation order is illustrated in the figure 3.9.

To simulate one position $q$ in $R$, $PatEv_q^{T_r}$ (the pattern with center $q$ in $R$ with template $T^r$) is compared against the patterns in $PatBase$. Comparing the entire base of patterns for each point, makes the algorithm very slow. For this reason, an interval $[0, a]$ is chosen (this is a parameter fixed before running the program), then the database is randomly traversed until finding one pattern whose distance $d_{total}$ to $PatEv_q^{T_r}$ is in the interval. If no pattern is at that distance, then the closest pattern is selected. The value in the center of the selected pattern is assigned to the position $q$. The same process is repeated for each position in $R$ until all the values are determined.

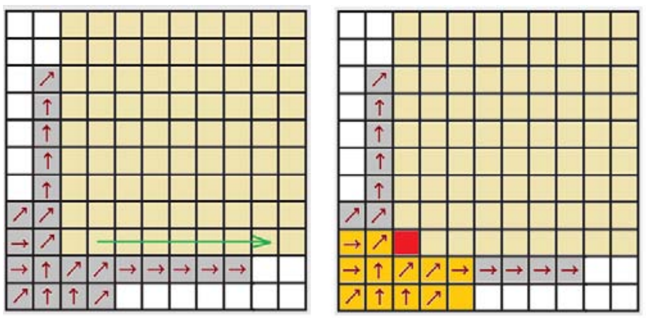


Figure 3.9: Initialization and order of the simulation.

An advantage of using the interval to select a pattern, instead of choosing the closest one, is that this introduces some degree of randomness into the program, which produce more variability in the realizations. Another argument in pro of the introduction of the interval is that the simulation is faster.

The goal of this simulation is to ensure that the newly assigned value will maintain as much local similarity between the two patterns as possible.

A sketch of the algorithm:

**Algorithm 2** DIR-SIM

```
1: A patterns base $PatBase$ is created from the TDF, using a template with L-form and radius $r$. Together with each pattern the location is stored.
2: We start with an empty grid. The values in the TDF first $r$ rows and $r$ columns (from left to right and bottom to top) are copied in the simulation grid; this is used as a simulation seed.
3: Simulating the angle in the location $t$. The simulation is done in a scan order line, starting in the left bottom corner. The template is centered at $t$ and the values are extracted, forming a pattern $P_t$. Each pattern in the base is compared with $P_t$ using the distance
       $$D = \beta \quad dis_{tdf} + (1-\beta) \quad dis_{loc}$$
4: Move to the next node
```

## 3.3 Simulations

This section shows some of the realizations obtained from the application of DIR-SIM.

### 3.3.1 Example 1

The training image used is given by the figure 3.10, with size 210 $\times$ 210. The sequences of contours $C_n$ and the difference $T_n - C_n$ is obtained and showed by figure 3.11. Erosion is applied four times and it was assigned the direction to 77.69% of the total black points. The TDF was obtained using two steps of length 1 and 3, to find the direction in each point, in the figure 3.12 it is presented a sample of the TDF, with a zoom of the gray region.

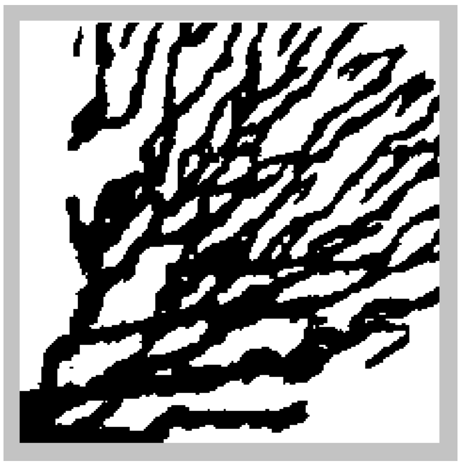


Figure 3.10: Training Image.

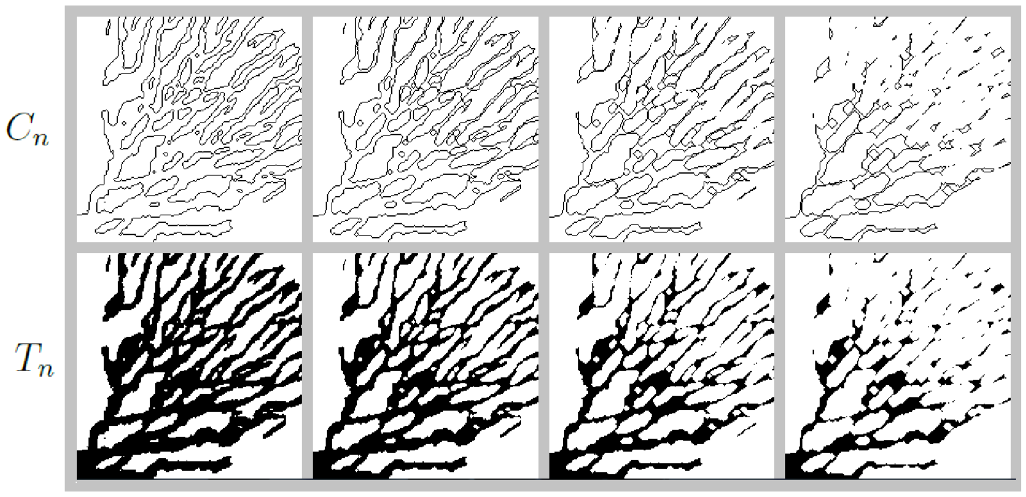


Figure 3.11: Sequence $C_n$ and $T_n$.

Figure 3.13 shows the result of the algorithm in the limit cases, when $\beta = 0$ or $\beta = 1$. The result is the expected, in the first case the distance of the positions between patterns have all the weight, for this reason the realization

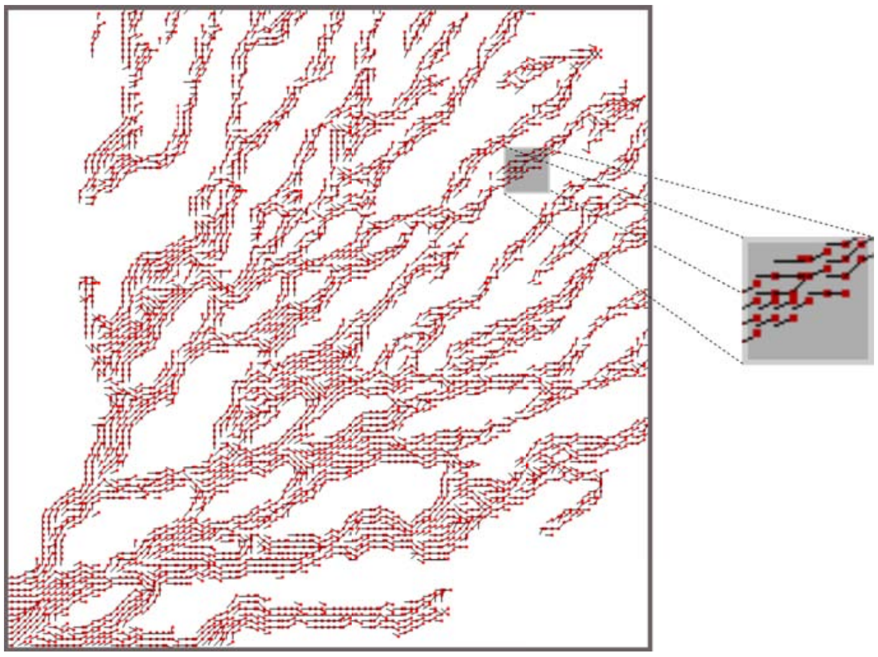


Figure 3.12: TDF of the training image 3.10 sampled in spaced points.

is almost identical to the training image. In the second case, we can see that the realizations have a stationary behavior, because the pattern position in the image has no influence in the pattern search.

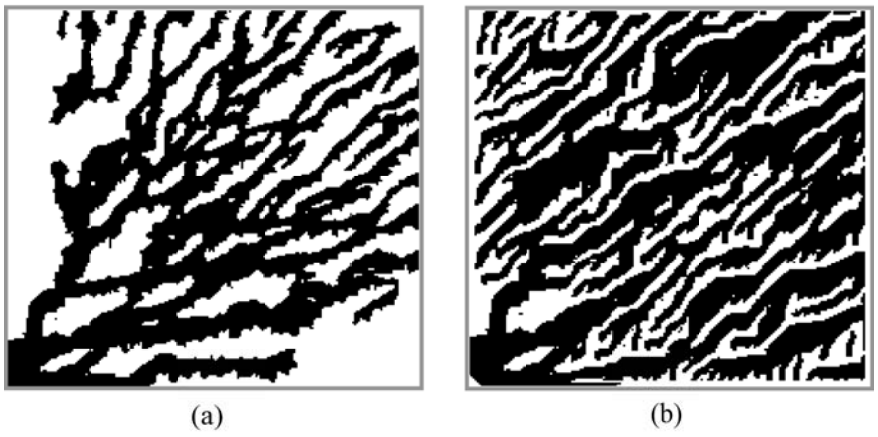


Figure 3.13: (a) Realizations with $\beta = 1$. (b) Realizations with $\beta = 0$.

In the figure 3.14 is presented four realizations for the value $\beta = 0.5$. It can be seen that the simulations are very similar to the training image but not equal, which is desired. In addition the realizations are preserving the connective expected, however some of the channels are thicker that the ones in the training image.

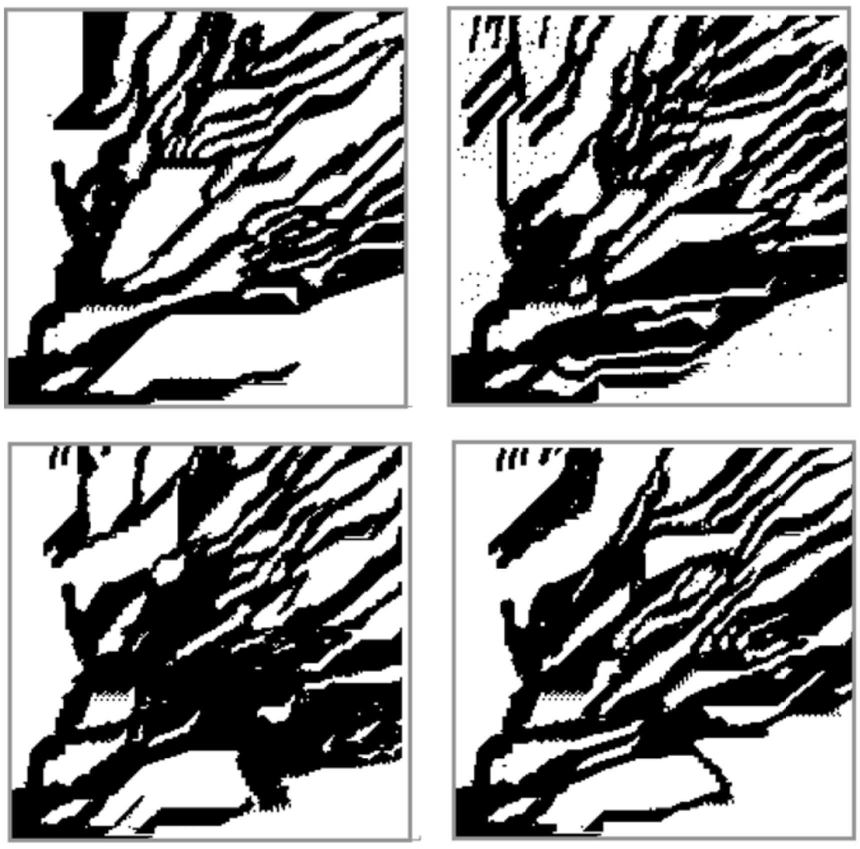


Figure 3.14: Realizations for $\beta = 0.5$.

### 3.3.2 Example 2

The training images used in MPS methods come from several different sources such as geologist drawings, photographs, or realizations of object based or process-based simulation. In this example the training image comes from a photograph of an alluvial fan blossoms across the landscape between the Kunlun and Altun mountain ranges that form the southern border of the Taklimakan Desert in China, given by figure 3.15(a). The blue left region is the active part of the fan, from it the training image will be generated, in figure 3.15(b) these blue channels were highlighted in red. Finally, figure 3.16 is the binary image used as training image.

The size of the image (figure 3.16) is 183 X 183. In figure 3.17, it can be observed the two sequences $T_n$ and $C_n$, erosion was applied 2 times. It was assigned the direction to 87.4% of the points that belong to some contour, $C_i, i = 1, 2$ . After interpolation was applied 6 times, all the remaining points has one direction designated.

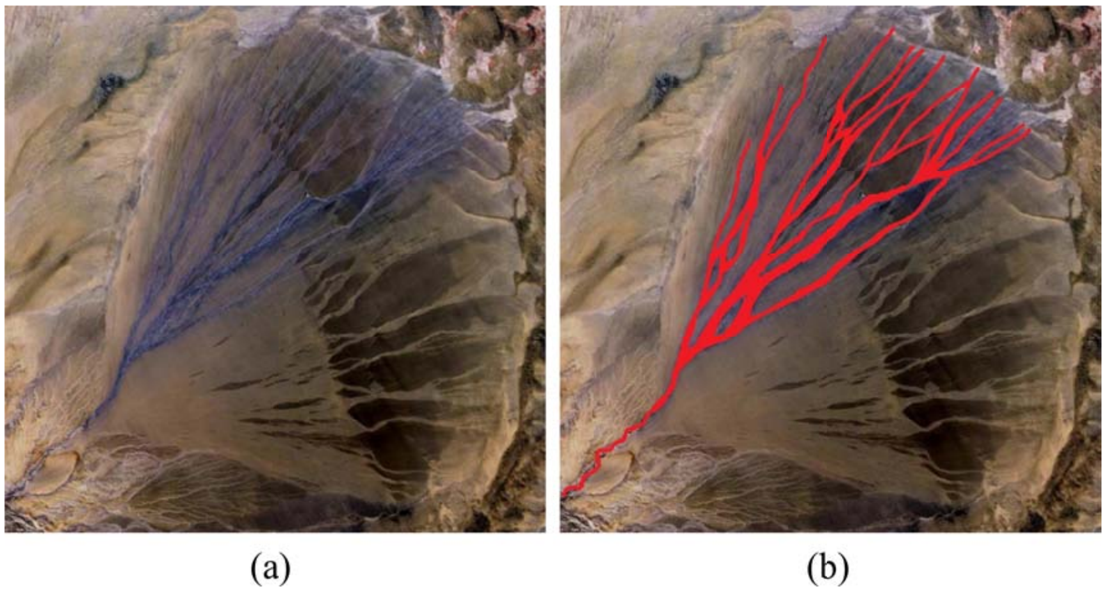


Figure 3.15: (a) Alluvial Fan, China. (b) The channel system highlighted in red.

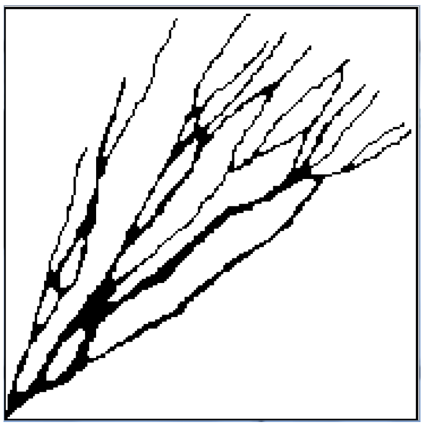
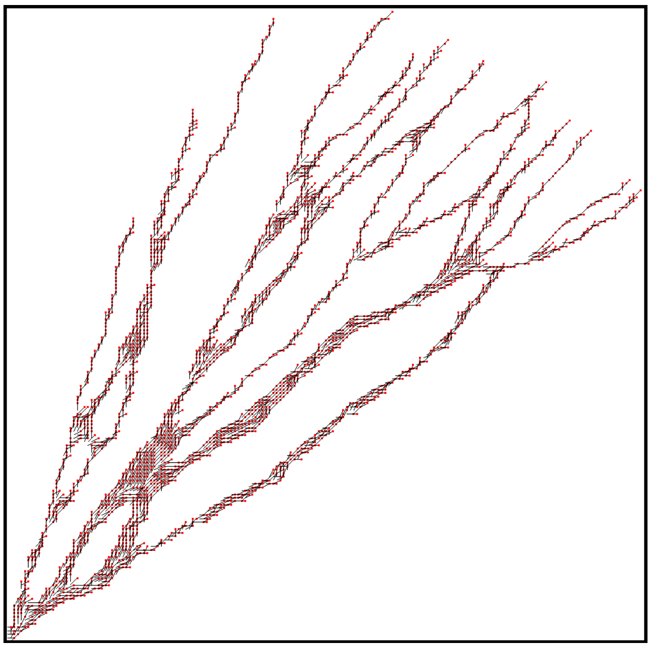


Figure 3.16: Training image obtained from 3.15.



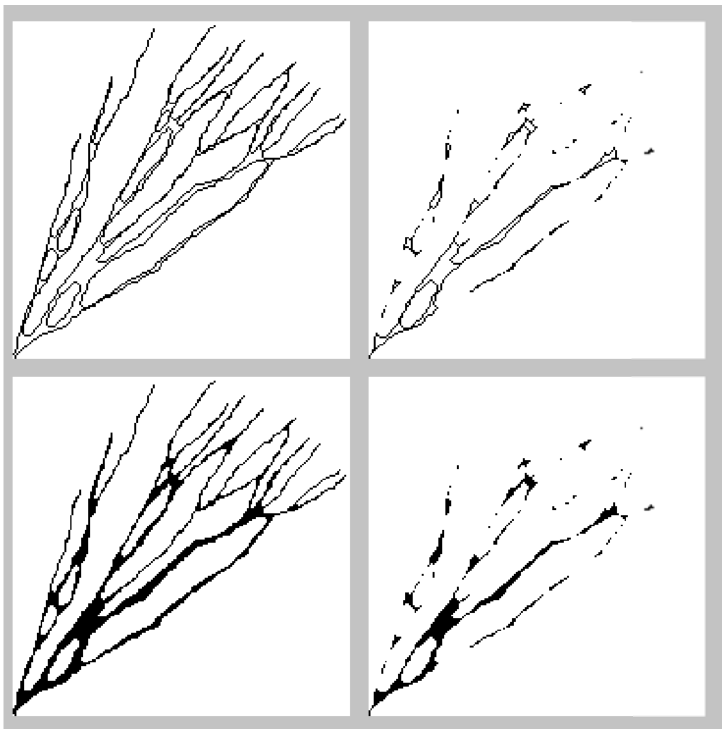


Figure 3.17: Erosion applied two times to the image 3.16.

To obtain the TDF, steps of size 1 and 3 were used. The directional interval assumed for this image, is $\left[0, \frac{\pi}{2}\right]$. The TDF can be observed in figure 3.18. In the image 3.19 it is shown 6 realizations using the parameter $\beta = 0.5$

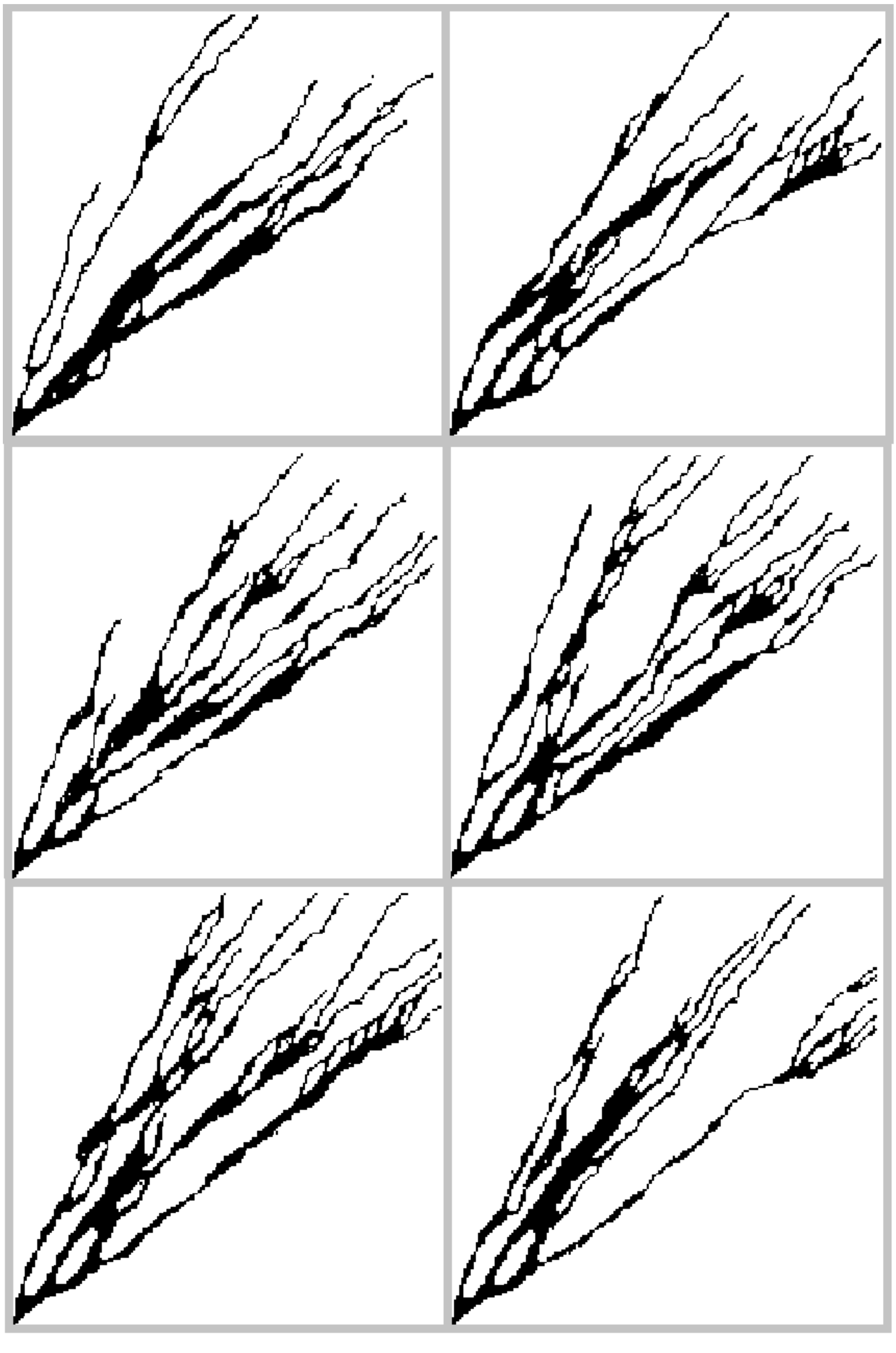


Figure 3.19: Realizations obtained from the simulation using $\beta = 0.5$.

For $\beta = 0.5$, 30 realizations were generated and the E-type (averages) was computed. Figure 3.20 shows the ensemble of all these realizations. It can be appreciated that the simulations present a good variability, they are well distributed inside a region limited by the leftmost and the rightmost channels. The only structure repeated in the simulations is the root of the image that was use as a seed.

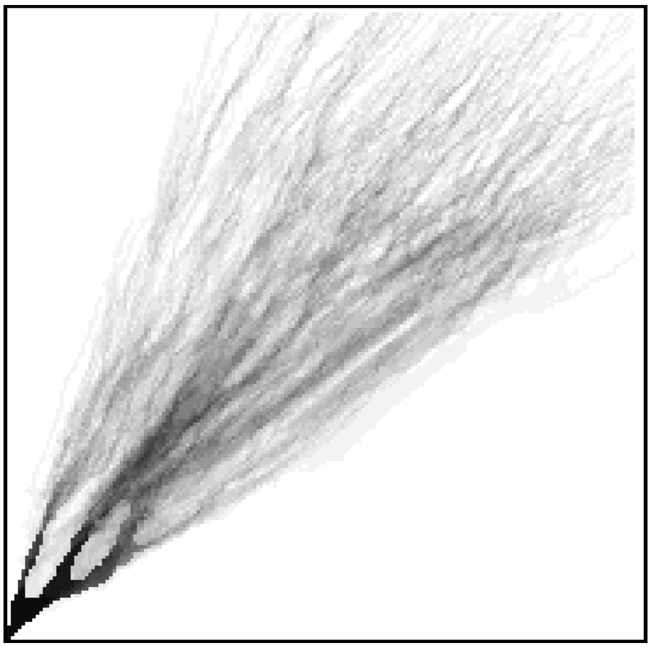


Figure 3.20: E-types (averages) over 30 realizations with $\beta = 0.5$.

## 3.4 Conclusions

In this chapter we presented a non-stationary multiple point method using training images with a particular geometry, that was called tree-like. In fact the method is more general, it can be applied to images in which a direction can be defined at each point. The main contribution is the introduction of a new object called training directional field TDF, defined from the training image. This field can be interpreted as the directions followed by the tree-like image. The idea is that the image being simulated should follow the TDF. This is done by including the directions in the calculation of the distance between patterns.

One of the advantages of this method is that the realizations obtained by the simulation, present the same geometrical characteristics that the training image. This does not mean that the simulation is very similar to the training image, on the contrary looking through the examples it can be seen that the variability between realizations is good. Although the simulation path is not random and goes in a scan order line, the variability comes from not choosing the most similar pattern but using an interval that defines the maximum allowed distance between two patterns.

The main drawback of the algorithm is the large CPU-time cost. This is caused by the search process in the pattern base. We use a strategy where the base is randomly traversed looking for a pattern with distance in an established interval. Although this is faster that checking all the patterns looking for the closest one, we need a more optimal search process.

Conditioning is an important subject for reservoir modeling. Though, conditioning was not a goal of this dissertation, in future works it can be adapted using MPS strategies usually employed in similar situations.

The similarity between patterns is computed using the position in the training image and the position simulated , so the algorithm only works when the simulation region has the same dimensions as the training image.

It was used a scan line path for the simulation. There are other possibilities as defining a path according to the the direction of the tree like image. For example, the path that goes diagonal by diagonal. Notice that in this case the template should be modified. This is illustrated in figure 3.21.

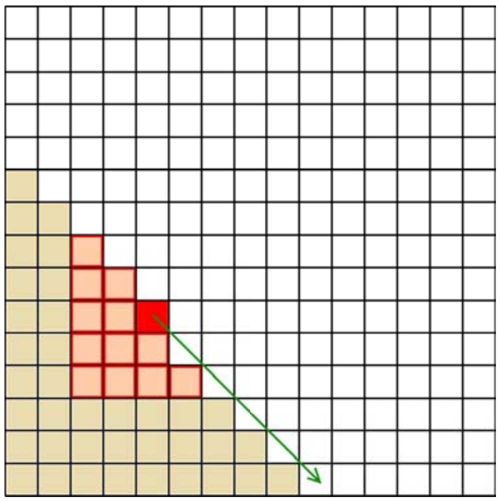


Figure 3.21: Other simulation order and template.

A recurrent problem when using MPS methods for elongated structures is that the realizations lack enough connectivity [11]. The algorithm proposed, DIR-SIM, preserves the connectivity as it appears in the training image. Normally, when the algorithm generates non-connected images it can be seem that the different connected components are related to the components of the training image. This connectivity can be compared with others works, for example in [2], two non-stationary simulations are presented. As it was explained in section 2.3 one of them generates an automatic segmentation of the training image in regions where a stationary modeling is applied. In the figure 3.22(b), one realization of this method is shown, using the training image given by the figure 3.22(a). Using the same training image one realization of the second method is shown in the figure 3.22(c), in this case the non-stationarity comes from the use of the pattern position in the training image in the simulation. In these two realizations one can observe that the continuity presented in the training image is not preserved. It is clear that the training image has some disconnected components, but in the realizations the quantity of this components is considerably higher. Another remark, is that it is clear that the realizations we obtained are tree-like images. However, it is not clear that this is the case for the realizations presented in the figures 3.22(b, c).

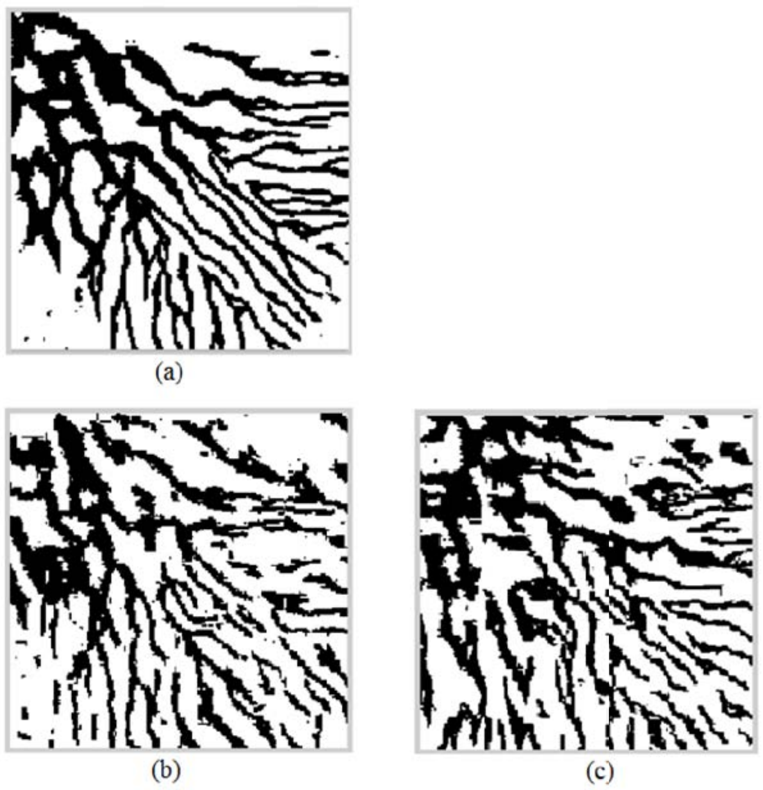


Figure 3.22: (a) Training image. (b) Realization using Gabor filters to divide the simulation area. (c) Realization including the spatial location of the patterns into the similarity search.

In the figure 3.24, two realizations of the method presented in [15] are shown. As it was explained in section 2.3, the idea is to divide the training image in regions where a stationarity simulation is performed. In 3.24 only one training image is used, given in the figure 3.23. The image 3.24(a) is obtained by applying a stationary simulation in each region, the training image used is a rotation of the original one (the rotation angle depends on the orientation presented at the region). The use of rotations explains why the channel system has a width similar in all regions. In 3.24(b) in addition to rotation, dilation was also employed to transform the original training image shown in the figure 3.23. Thus, the realization presents a width that depends on the position in the image. In this realizations it can be observed that the continuity is maintained.

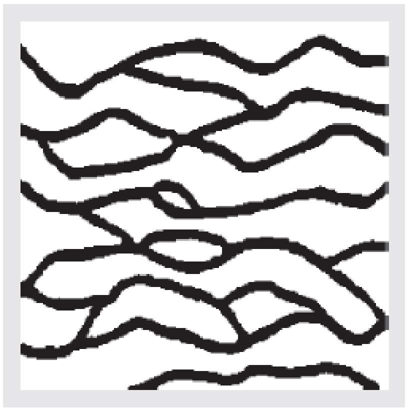
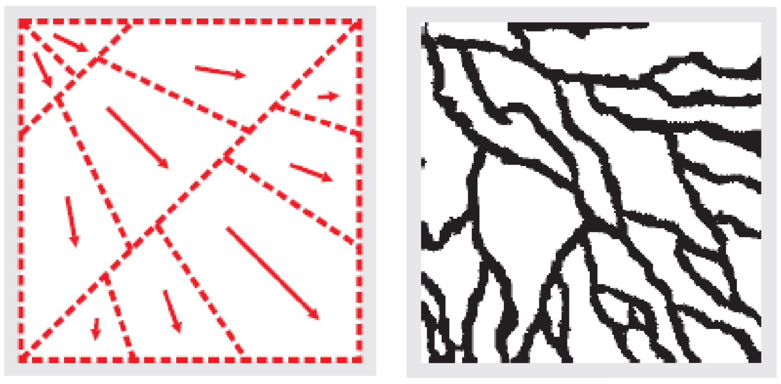


Figure 3.23: Training image to be rotated and dilated for the simulation of the different regions given in the figure 3.24

The training images used in the previous examples, present white regions, i.e. regions without channels. However, in the simulations this characteristic is not maintained, it can be observed that all the regions in the images are filled with channels. On the contrary, one characteristic presented in our simulations, is that the white regions are preserved.

The main idea that we proposed was the introduction of the TDF, and its application computing distances. Observe that the use of the TDF can be introduced in many other methods from MPS and textures synthesis, the basic idea would be to use the TDF to compute distances. Clearly, this would only work for training images where the TDF makes sense (satisfy the condition 1 of tree-like). This can be thought of as passing from using $C^0$ distance to a $C^1$ distance, here we are making an analogy with $C^0$ and $C^1$ distances for differentiable functions.


(a)

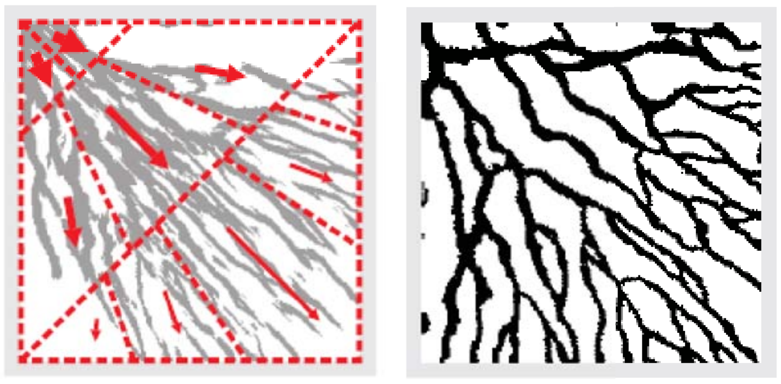

(b)

Figure 3.24: (a) Realization using rotation. (b) Realization using both rotation and scaling.

# 4
# Skeleton based simulation

In this chapter it is presented another approach to model turbidite channels using tree-like training images. The method is called **skeleton based simulation**, SKE-SIM and it is an object based model that uses some elements of multipoint geostatistics. SKE-SIM starts with a training image which is represented by a one-dimensional object called training skeleton. From this new object, information about the direction and length of the channels is extracted and it is used to simulate others skeletons. These new skeletons are used to create a 3D model of channels inside a turbidite lobe.

## 4.1
## Training Skeleton

A channel system such as the one shown in the figure 4.1(a) can be approximated by a one dimensional structure given by the red lines in the figure 4.1(b), that will be called *skeleton*. This structure reflects the global behavior of the channel system. It contains information of the original object such as the points representing regions where there is a channel bifurcation and edges representing channels in the system.

The skeleton is a graph in the plane formed by edges that are straight lines. The nodes represent the channel bifurcations and all skeleton has a special node corresponding to the root of the graph.

In the same way of the algorithm presented in the previous chapter, SKE-SIM starts with a 2D binary training image, as presented in figure 4.2(a). From this image, a first one-dimensional representation of the channels is generated. It is obtained by erosion of the object but preserving the connectivity of the branches.

To obtain this representation, an image processing program called FIJI was used. It can be found in https://imagej.net/Fiji. The plug-in Skeletonize3D performs the process described in the last paragraph. The figure 4.2(b) is the result obtained using this plug-in.

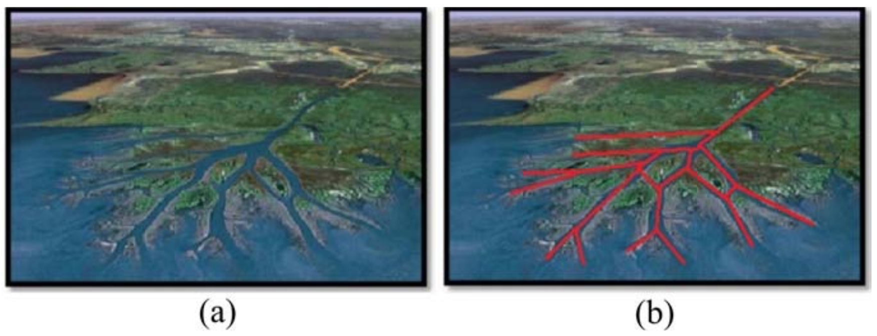


Figure 4.1: (a) Channel system. (b) One dimensional approximation of the channels.

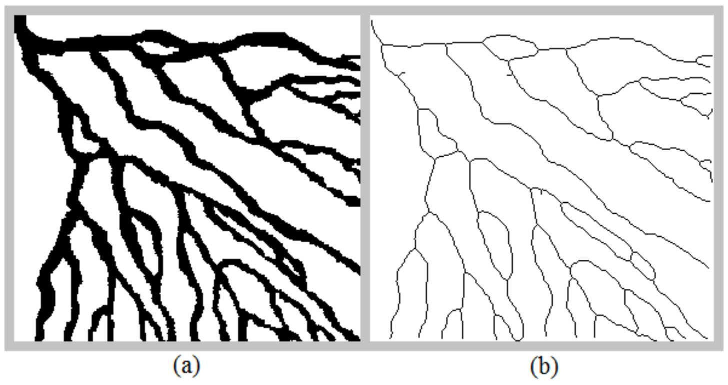


Figure 4.2: (a) Training image. (b) One-dimensional representation obtained by using FIJI.

Using FIJI the bifurcation points are found, and how they are connected. Now the skeleton is built using this information, connecting the nodes by straight lines. The figure 4.3 is the skeleton obtained from 4.2(b), this will be called **training skeleton**.

## 4.2 Extracting information from the training skeleton

Now, two relevant information will be extracted of the training skeleton to guide the generation of others skeletons. The fist one is the bifurcation angle. To define this element, first we observe that at the end point of any edge , two channels come out. The bifurcation angles, at this node, are defined as the angles that form the new edges with respect to the direction of the channel from which they originated. In the figure 4.4, $\alpha$ and $\beta$ are the bifurcation angles of the edges $a$ and $b$.

The other data that is extracted from the training skeleton is the length

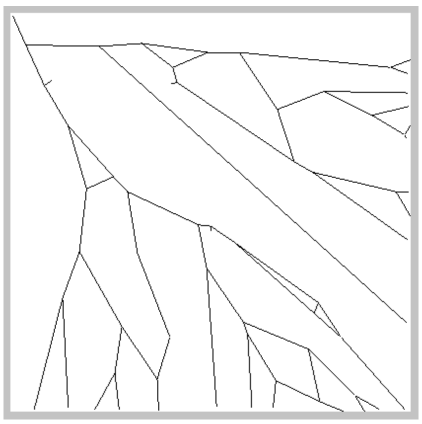


Figure 4.3: Training skeleton obtained from the figure 4.2(b).

of the edges. This number can be interpreted as the distance the channel moves before it bifurcates.

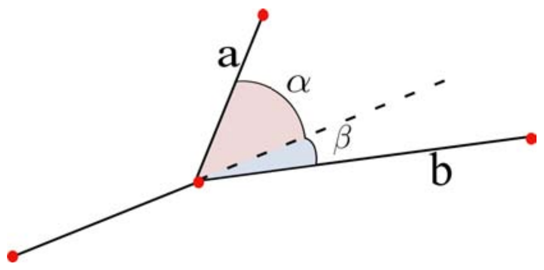


Figure 4.4: Bifurcation angles and lengths.

The information extracted from the training skeleton, the length of each edge and the bifurcation angles, is analyzed and organized. The data is used to obtain probability distribution functions for the angles and lengths (constructing histograms). However, this is only appropriate if the amount of information is sufficient. On other cases, it can be used to define an interval from which random values will be chosen.

In the next section it will be explained how the extracted information is used to construct new skeletons.

## 4.3 Skeleton definition

A skeleton $Sk$ is given by two sets: the edges $E$ and the nodes $N$. The edges can be interpreted as the channels, and the nodes as bifurcation or convergence points of channels.

Each edge $e \in E$ is a directed segment defined by two points in $\mathbb{R}^2$, the father point $f_e$ and the son point $s_e$. The edge is directed because it goes from the initial point $f_e$ to the final point $s_e$. It is expressed as $f_e \overset{e}{\to} s_e$.

A node $n \in N$ is an object formed by 3 elements: a point $p_n \in \mathbb{R}^2$, a vector $\alpha_n \in \mathbb{R}^2$ and an integer $M_n \in \mathbb{N}$. The point $p_n$ is the initial or final point of one or more edges in $E$. $\alpha_n$ contains the directions of the edges that arrive at the point $p_n$, it can be one or two edges. $M_n$ is the number of edges that arrive or left the point $p_n$, it is called the *mark* of the node.

The skeletons that are going to be used verify that $M_n$ is always a value in the set $\{1, 2, 3\}$. The figure 4.5 shows examples of nodes, with its respective points, marks and the angles of the edges that arrives on it. In the node $n$ where two edges come out from it, this can be interpreted as a channel that meets an obstacle and it is bifurcated into two channels. In node $m$ where two edges arrive, it means that two channels meet. The case when the node has mark 1 or 2 can be understood as the situation before the node turns into something like $n$ or $m$. For example, the node $p$ has mark one and it represents a channel ending at $p$ that could bifurcate generating two more edges. Similarly, the node $q$ has mark 2 and this is interpreted as two channels meeting at $q$, from the fusion of this two channels one new edge could be created.

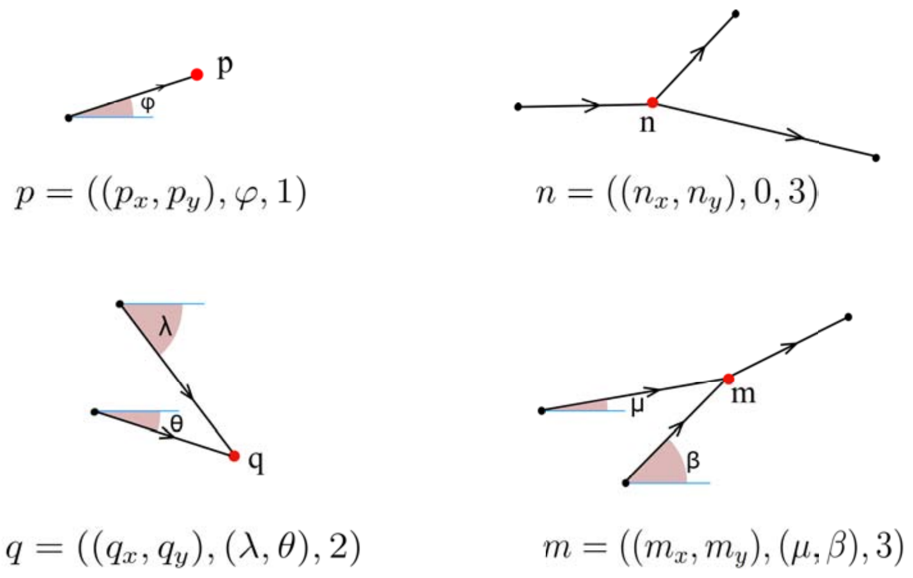

p
φ
$p = ((p_x, p_y), \varphi, 1)$
n
$n = ((n_x, n_y), 0, 3)$
λ
θ
q
$q = ((q_x, q_y), (\lambda, \theta), 2)$
μ
m
β
$m = ((m_x, m_y), (\mu, \beta), 3)$

Figure 4.5: Illustration of the possible configurations of the nodes.

## 4.4 Skeleton synthesis

The construction of the skeleton aims to mimic the channel system development. The idea is to generate a sequence of skeletons

$$Sk_0, Sk_1, Sk_2, \ldots, Sk_n, \ldots$$

where each skeleton $Sk_{m+1}$ is obtained by bifurcating the previous skeleton $Sk_m$. This can be described as the successive application of the function $Bif : SKE \to SKE$, where $SKE$ is the set of all the skeletons. We have that:

$$Bif(Sk_m) = Bif^{m+1}(Sk_0) = Sk_{m+1}.$$

For the skeleton $Sk_m$, this function $Bif$ bifurcates each node $n \in N_m$ with mark different to 3. If the node has mark 1 then it is bifurcated in two channels, on the contrary if it has mark 2, only one channel is generated. To bifurcate a node and create new edges, the function $Bif$ uses the directions of the edges that arrive at this node. The angle and the length of the new edge is chosen using the probability distributions extracted from the training skeleton, as it was explain in the section 4.2.

The sequence is initialized with a skeleton $Sk_0$ formed by one edge $e_0$ and one node $n_0$. Without loss of generality this initial skeleton is constructed from the origin in the direction of the positive $x$-axis. Then the edge $e_0$ has slope 0 and the initial point is $f_e(0, 0)$. Its length is chosen using the distribution function of the lengths obtained from the training skeleton. The node $n_0$ is formed by the final point of the edge $e_0$, the direction of $e_0$ and the mark $m_{n_0} = 1$ because only the edge $e_0$ is related to it.

Given a skeleton $Sk_m$, its set of nodes $N_m$ is traversed and the nodes with mark different to 3 are bifurcated generating new edges. All the resulting edges are stored in a set denoted by $E_{pos}$ of the possible edges to be included in the new skeleton $Sk_{m+1}$. In the figure 4.6(b) the green edges form the set $E_{pos}$ from the skeleton in the figure 4.6(a). This set is traversed in a random order, that in the figure 4.6 is given by the number at the end of each edge. Every time one edge in $E_{pos}$ is analyzed, another skeleton is created by adding the new edge, so we have a sub-sequence of skeletons, given by

$$Sk_{m,1}, Sk_{m,2}, \ldots, Sk_{m,l} = Sk_{m+1},$$

where $Sk_{m,n}$ is the skeleton obtained by inserting the edge in the position $n$ in $E_{pos}$ to the previous skeleton $Sk_{m,n-1}$. In the case of $Sk_{m,1}$ the previous

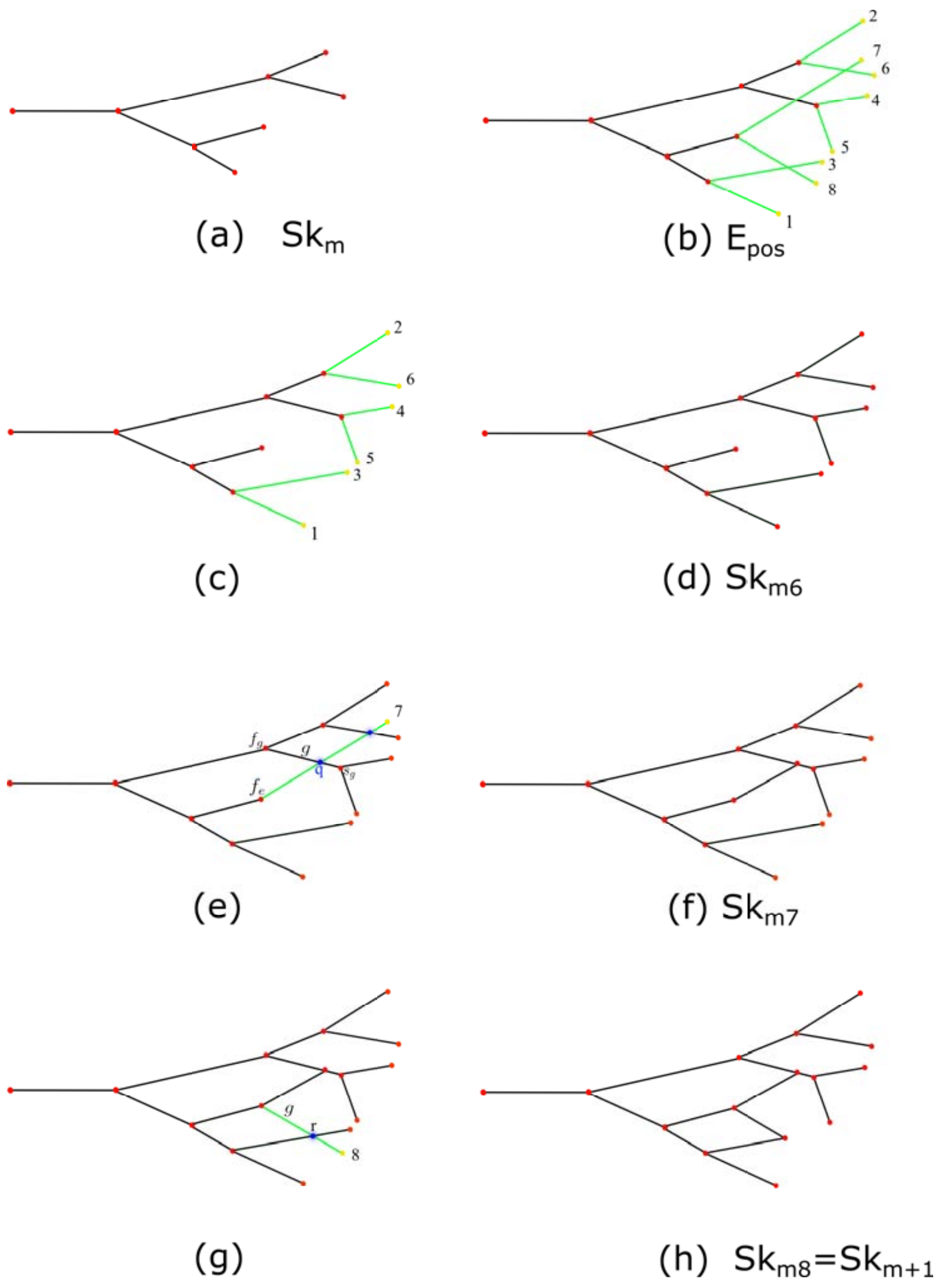


Figure 4.6: Skeleton bifurcation.

skeleton is $Sk_{m-1}$. $l$ is the number of elements in $E_{pos}$. The insertion process to generate $Sk_{m,n}$ from $Sk_{m,n-1}$ is not just adding the $n$-th edge, since this edge could intersect the already generated skeleton. First, it must be determined whether the edge intersects the previous skeleton or not. If it has not a intersection then it is included. In the figure 4.6(c) it is observed that the edges from 1 to 6 have no intersection with the initial skeleton nor with the edges of the set $E_{pos}$ which are successively included in the skeleton. Then these edges are included, generating 6 skeletons, $Sk_{m,1}, \dots, Sk_{m,6}$. In figure 4.6(d) $Sk_{m,6}$ is shown.

It can happen that the $n$-th edge in $E_{pos}$, $f_e \overset{e}{\to} s_e$, intersects the skeleton $Sk_{m,n-1}$. In this case, all the intersection points between $e$ and $Sk_{m,n-1}$ are found and the closest one to $f_e$, denoted by $q$, is selected. Let $f_g \overset{g}{\to} s_g$ be the edge in $Sk_{m,n-1}$ to which $q$ belongs. We consider two cases:

1. If the mark of the node at $s_g$ is not 1: to generate $Sk_{m,n}$ the edge $g$ is deleted from $Sk_{m,n-1}$ and three new edges and one node are added $f_g \overset{g_1}{\to} q$, $q \overset{g_2}{\to} s_g$, $f_e \overset{e_n}{\to} q$ and one node at $q$. This situation is described in the figure 4.6(e). It is observed that the edge 7 has two intersections with the previous skeleton $Sk_{m,6}$ and $q$ is the closest to the initial point. Then the edge that starts at the initial point of the edge 7 and ends at $q$ is inserted, this can be seen in the resulting skeleton $Sk_{m,7}$ given in the figure 4.6(f).

2. If the mark of the node at $s_g$ is 1: the edge $e \in E_{pos}$ and $g$ are modified, too. These two edges will preserve their initial points but their end points are now the point of intersection $q$. The node that contain the point $q$ is inserted in the list of nodes and has mark equal 2. In the figure 4.6(g) this case is illustrated, the edge 8, the last in the set $E_{pos}$, and the intersected edge are modified. Figure 4.6(h) is the resulting skeleton $Sk_{m,8}$.

The bifurcation process continues until some threshold is attained. This threshold could be the number of times the function $Bif$ is applied. Moreover, if the skeleton must be generated within a specific region then the growth of a branch of the skeleton stops when the branch exits this region.

A sketch of the algorithm is presented in the algorithm 3.

## 4.5 3D turbidite channels simulation

The skeleton is one dimensional representation in the plane for channel systems with a tree-like geometry. The actual dimension of the channel system is 3D, the images considered are thought as projections to a plane of the 3D object. In this section it is presented a way to transform a skeleton representation in a 3D image.

This work was mainly motivated by turbidite channels, those channels formed inside turbidite lobes (see figure 4.7). The first step in the 3D modeling will be to create the lobes, inside which the channels will be placed. It is adopted the turbidity lobe modeling proposed in the work [16]. This is a simple depositional model with three turbidites lobes. Once the lobe has been

**Algorithm 3** Skeleton generation

```
1: Initialize with one skeleton $Sk$, that contains one node and one edge.
2: **While** $Sk$ does not meet the threshold do the following:
3: The node list in $Sk$ is traversed, and those with mark different to 3 are selected.
4: Each selected node is bifurcated according to its mark. If the mark is 1 then it generates two new edges. If the mark is 2 then it generates one edge. To find the direction and the length of the new edges the probability distributions from the training skeleton are used.
5: All new edges form the set $E_{pos}$.
6: Choose a random order $e_1, ..., e_l$ to transverse $E_{pos}$.
7: $Sk_0 := Sk$.
8: **For** $n$ from 1 to $l$ **do**:
        If the edge $e_n$ does not intersect any edge in $Sk_{n-1}$, then $e_n$ is added to $Sk_{n-1}$ to obtain $Sk_n$.
        If the first point of intersection between $e_n$ and $Sk_{n-1}$ is $q$, which belongs to the edge $g$. Then the new edge $f_e \overset{e_n}{\to} q$ is added to $Sk_{n-1}$ and $g$ is modified according to the mark of the node at $s_g$, this gives $Sk_n$.
9: Define $Sk = Sk_l$ and return to the start of the while.
10: Once the while is exited the result of the simulation is the skeleton $Sk$.
```

created the skeleton synthesis is done inside it. This means that the skeleton development is constrained by the lobe boundary.

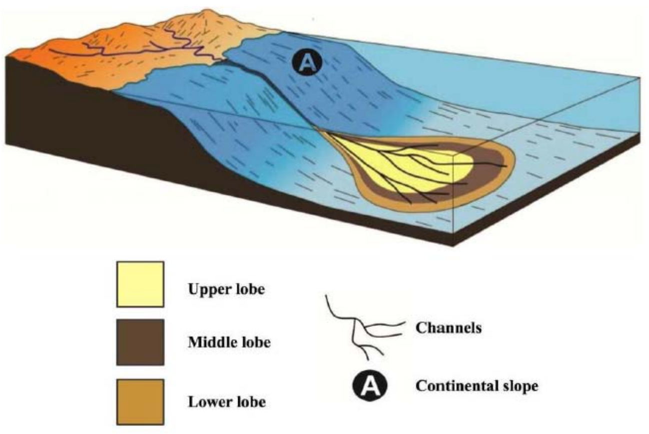


Figure 4.7: Depositional system model of turbidites.

In the next subsection the modeling of lobes is explained, following the work [16].

### 4.5.1 Turbidite Lobe Modeling

The turbidite lobe is constructed using basically three parameters: depth, width and length $l$. In the figure 4.10(a) the general structure given to the lobe modeled and the parameters used can be observed. To create a non-symmetrical lobe, the width and depth are divided in two, right width $w_r$, left width $w_l$, superior depth $d_s$ and inferior depth $d_i$. In the figure 4.10(b) a depositional system with three turbidite lobes is shown. First the deepest lobe is built, and according to its parameters the following lobes are created eroding the previous ones. In this case the green lobe erodes the blue one, and the red lobe erodes the green.

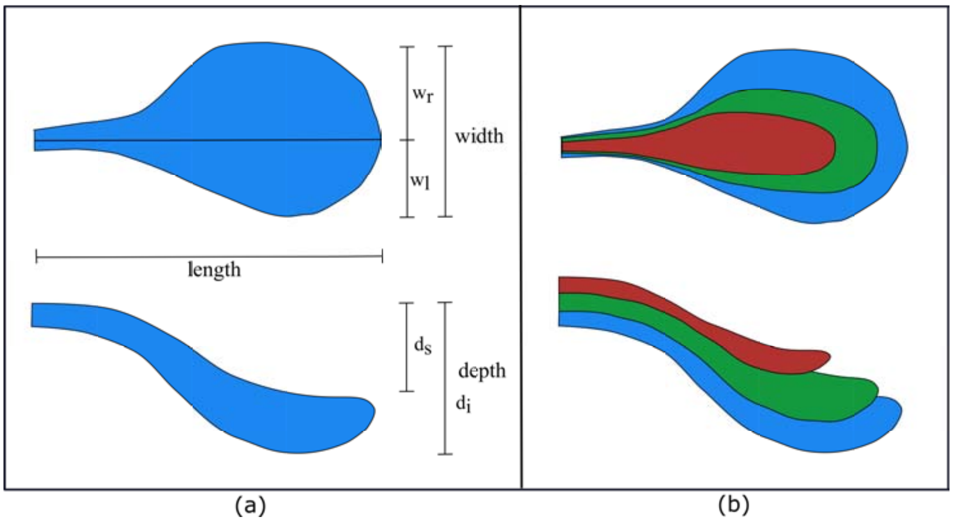


Figure 4.8: Lobe parameters.

The environment used to created the lobe system is a grid in $\mathbb{R}^3$ which is a partition of the space by parallelepipeds. In this work the partition is given by cubes.

The grid is defined in a cube $[x_{min}, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$ where each side is partitioned in $n_x, n_y, n_z$ equal parts respectively. This partition generate a set of cubes, also called cells. Each cell forming the grid is determined by a triple $(i, j, k)$, where $i = 0, \ldots, n_x - 1$, $j = 0, \ldots, n_y - 1$ and $k = 0, \ldots, n_z - 1$. This triple is used to refer to the cube $C_{i,j,k}$ given by:

$$\begin{aligned} C_{i,j,k} = &[x_{min} + i\Delta x, x_{min} + (i+1)\Delta x] \\ &\times [y_{min} + j\Delta y, y_{min} + (j+1)\Delta y] \\ &\times [z_{min} + k\Delta z, z_{min} + (k+1)\Delta z] \end{aligned} \tag{4.1}$$

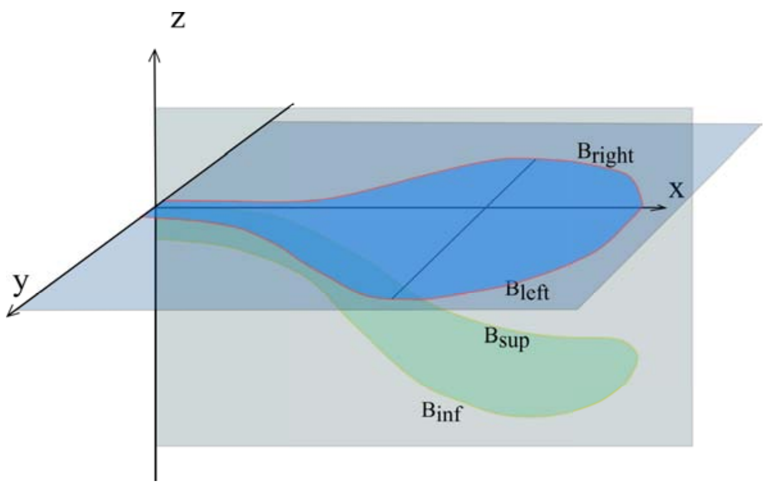


Figure 4.9: B-Spline curves in the planes $xy$ and $xz$.

where $\Delta x = \frac{x_{max}-x_{min}}{n_x}$, $\Delta y = \frac{y_{max}-y_{min}}{n_y}$ and $\Delta z = \frac{z_{max}-z_{min}}{n_z}$.

## Single-valued B-spline curves modeling

The algorithm of [16] creates the lobe by first creating two regions, one in the plane $xy$ and another in the plane $xz$, so that these regions should coincide with the projections of the lobe to those planes. To create the regions four functions are defined, $B_{right}, B_{left}$, $B_{inf}, B_{sup}$, the graphs of those functions are the boundary of the regions. The graph of $B_{right}, B_{left}$ limit the region in the plane $xy$, they are constructed as single-value B-spline curves and their definition depends on the parameters $w_{right}$, $w_{left}$ and $l$. The graph of $B_{inf}, B_{sup}$ limit the region in the plane $xz$, they are constructed as single-valued B-spline curves modeling B-spline curves and their definition depends on the parameters $d_{inf}$, $d_{sup}$ and $l$. This is described in figure 4.9.

The control points to define the B-spline curves lie in the planes $xy$ and $xz$, they are chosen so that their $x$ coordinates are equidistant. The single value B-spline curves are functions with respect to the variable $x$; so, given a point, it will be easy to decide in which side of the curve it lies. The interval $[0, l]$, on the $x$-axis, is divided evenly into the number of control points desired $n + 1$. The partition $\{0, \frac{l}{n}, \frac{2l}{n}, \dots, l\}$ is obtained, which will be the coordinate $x$ of the control points. Remember $l$ is the length of the lobe. The algorithm of [16] was done with a fixed number of control points for each lobe, since it is used in this work, our simulations are also restricted to these quantities. The work of [16] can be modified to allow a different number of control points. However, as the realizations show, using these numbers of control points one gets curves with the desired features.

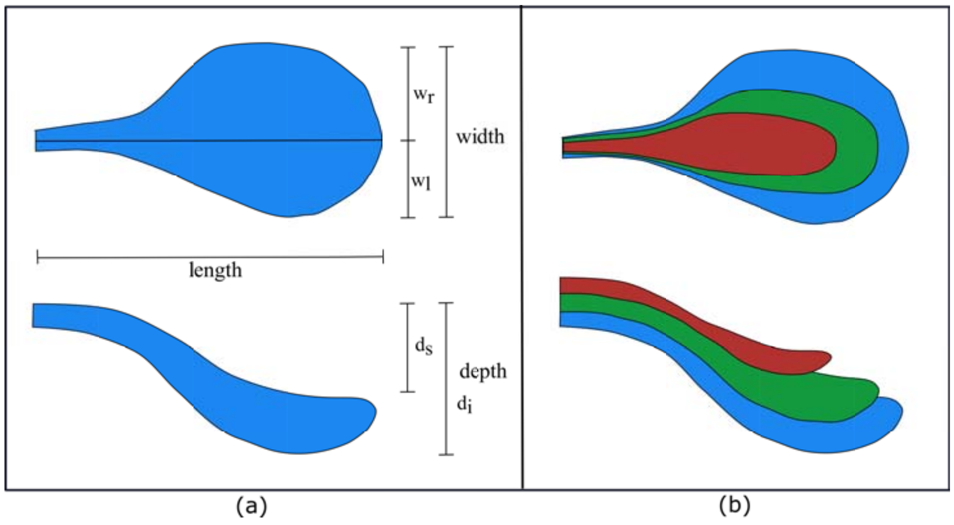


Figure 4.10: Lobe parameters.

The control points are chosen according to the turbidite parameters and the lobulated structure wanted in the curve. In the figure 4.11, B-spline curves with the coordinate $x$ of its control points are shown. First, it is built the deepest lobe using fourteen points. The second and third lobes have thirteen and twelve points respectively.

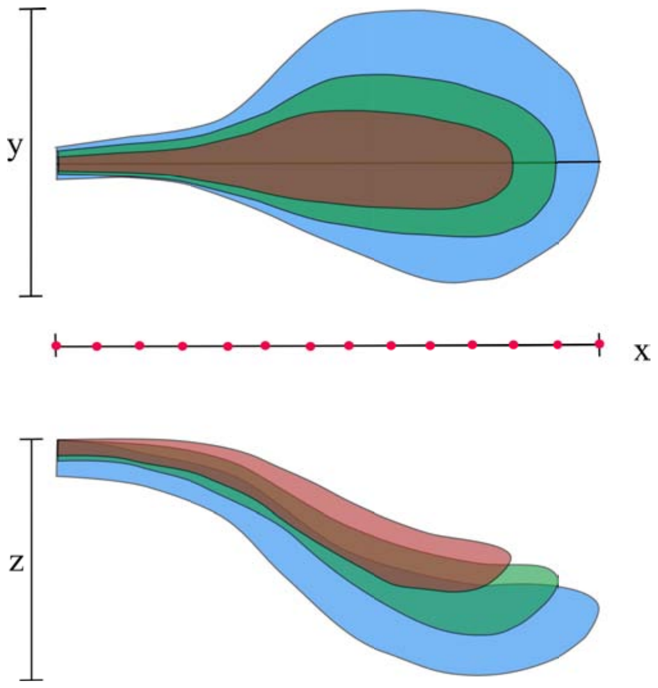


Figure 4.11: Control points to the 3 lobe depositional system.

## Connecting B-Spline Curves

The four B-spline curves created in the planes will determine the shape of the lobe, here it will be explained how they generate the volume.

Given $x \in [0, l]$, define $y_r = B_{right}(x)$, $y_l = B_{left}(x)$, $z_i = B_{inf}(x)$ and $z_s = B_{sup}(x)$, see the figure 4.12(a).

The idea is to construct for each $x_0 \in [0, l]$ two quarters of ellipses in the plane $x = x_0$ (parallel to $yz$) using the images of $x_0$ through the four B-spline curves. When all this ellipses are joined a volume is obtained, this is the lobe.

Given $x_0 \in [0, l]$, consider the points $(x_0, y_r, z_s)$, $(x_0, y_l, z_s)$ and $(x_0, 0, z_i)$ in the plane $x = x_0$. Using this points a region in the plane $x = x_0$ is defined, that corresponds to the intersection between the lobe and the plane. The two first points are joined by a segment. Then, it is constructed two quarters of ellipses in the plane $x = x_0$. The ellipses have center $(x_0, 0, z_s)$, one passes through the points $(x_0, y_r, z_s)$, $(x_0, 0, z_i)$ and the other through $(x_0, y_l, z_s)$, $(x_0, 0, z_i)$. The object thus constructed is called $ellip_L(x_0)$. The ellipses have the equation

$$\frac{y^2}{a^2} + \frac{(z - z_s)^2}{b^2} = 1, \ x = x_0.$$

One of them with $a = |z_i|$ and $b = |y_r|$ and the other with $a = |z_i|$ and $b = |y_l|$. Figure 4.12(a,b) shows the two ellipses in $ellip_L(x)$. The union of all the $ellip_L(x)$ for $x \in [0, l]$ gives the lobe. It is illustrated in the figure 4.13.

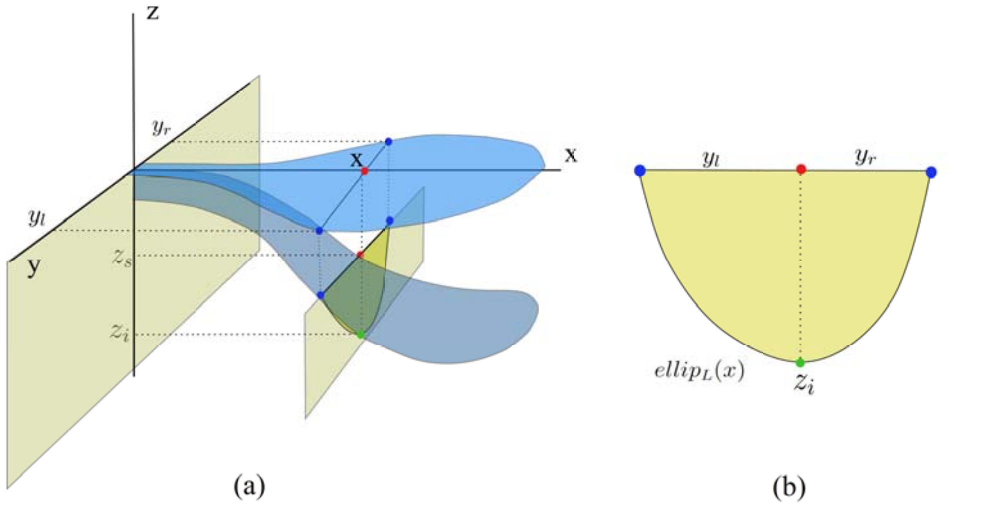


Figure 4.12: Two quarters of ellipses forming $ellip_L(x)$.

Bellow it is shown how the previously defined concepts are used to represent the lobe in the grid.

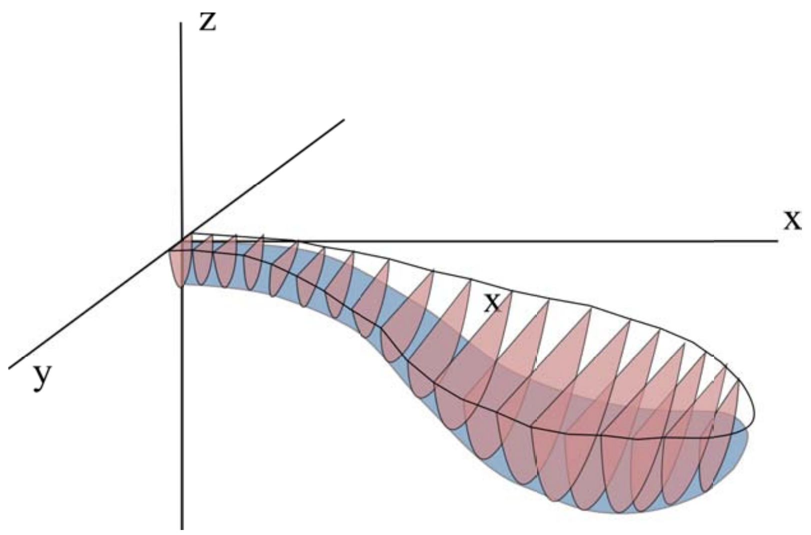


Figure 4.13: Volume from the B-spline curves as the union of quarter of ellipses.

### Object in the grid

The lobe system is represented in a 3D grid. To determine whether a cube in the grid is inside a lobe or not, its center is evaluated in the equations that determine the lobe. The cube $C$ is inside the lobe $L$ if and only if its center $(p_x, p_y, p_z)$ lies in $ellip_L(p_x)$.

Given $C$ a cube in the grid, a mark is assigned to $C$ in such a way that the mark characterizes to which lobes $C$ belongs. The mark of $C$ is the product $a \cdot b \cdot c$ where $a = 2$ if $C$ is in the lower lobe, otherwise $a = 1$, $b = 3$ if $C$ is in the middle lobe, otherwise $b = 1$, $c = 5$ if $C$ is in the upper lobe, otherwise $c = 1$. Those marks are used to paint the grid and generate a 3D image of the lobe.

In the figure 4.14, it is shown some lobes generated in a grid with dimensions $100 \times 100 \times 100$.

## 4.5.2 Turbidite Channel Modeling

Now, the skeleton will be used to construct a 3D model of a turbidite channels. The basic idea is to built the skeleton in the plane $xy$ conditioned to the limits given by the B-spline curves $B_{left}$ and $B_{right}$. Then the skeleton is thickened and also it is given a depth according to the lobe depth parameter.

### Skeleton inside the lobe

When a skeleton is created, a criterion for stopping its growth must be given, it could be the number of bifurcations from the initial skeleton $Sk_0$.

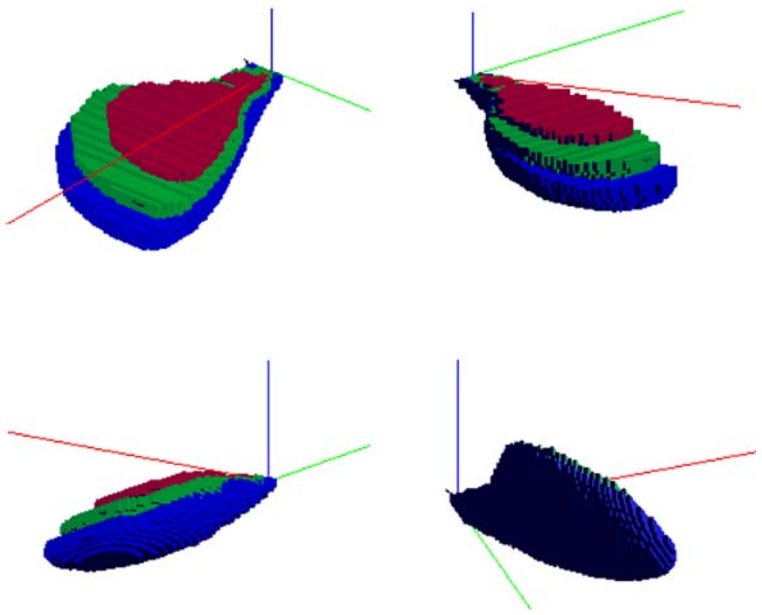


Figure 4.14: Simulation of a turbidite lobes [16].

Now, the skeleton is created in a 3D environment, in the plane $xy$ inside the region limited by the curves $B_{right}$ and $B_{left}$. Then, the growth is limited by these curves, i.e. the skeleton must be fitted into the lobe, as illustrated in the figure 4.15. The skeleton synthesis changes slightly with this constraint. First, the initial edge in the skeleton $Sk_0$ is set to have the initial point $(0, 0, 0)$ and the direction is the positive axis $x$. The algorithm is basically the same, but now if the final point of an edge $f_e \overset{e}{\rightarrow} s_e$ in the set $E_{pos}$ is outside of the the region limited by the curves $B_{right}$ and $B_{left}$, then the mark $M_n$ of the node $n$ with point $f_e$ is set as 3, i.e. the skeleton stops growing from this node. The algorithm continues until all the nodes have mark 3.

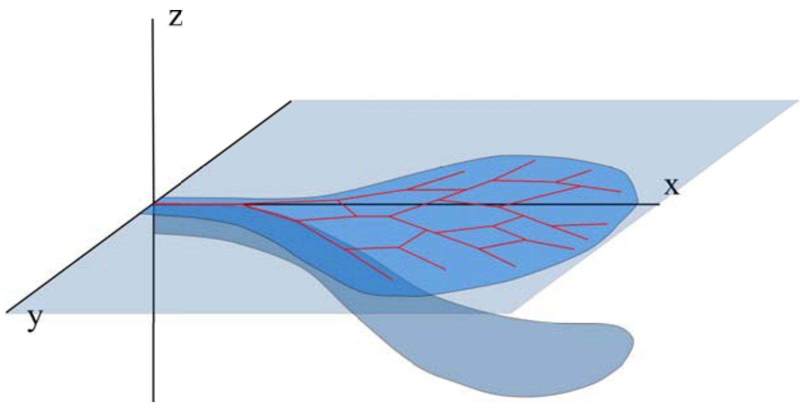


Figure 4.15: Skeleton built inside the region limited by the curves $B_{right}$ and $B_{left}$ in the plane $xy$.

### 3D Channel formation from the skeleton

After the skeleton $S$ has been created in the plane $xy$ inside the region limited by the B-splines curves in this plane, a 3D model of the turbidite channels is constructed from it. The channel system is built by giving volume to each edge in the edge set $E$ of the skeleton $S$. Each edge is used as the center line of a channel. The cross section orthogonal to the edge is a half-ellipse, with parameters the width $w_{ch}$ and depth $d_{ch}$ of the channel (see figure 4.16). This is applied to all the skeleton edges.

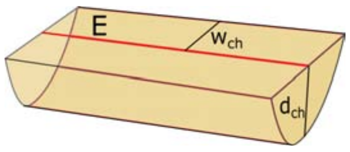


Figure 4.16: Channel formation from one skeleton edge and a channel system limited by the B-spline curves.

### Channel system in the lobe

Previously, a volume was given to the skeleton by thickening and deepening each edge. Now, the channels system should be placed inside a lobe, this is explained in this subsection.

Until now the channels system has its surface in the $xy$ plane. To fit it inside the lobe, each point $(x, y, z)$ in it is translated downward by the vector $(0, 0, z_s)$. Remember that $z_s = B_{sup}(x)$. This procedure projects the former channels system to the superior surface of the lobe, which is given by the B-spline curve $B_{sup}$.

Like the lobes, the channels will also be represented in the grid. A cell in the grid is inside the channels system if its center is in it.

Three channels systems are simulated, one for each lobe. The cells in the grid are then marked according to which object they belong. To define the marking, the objects drawn in the grid are ordered as follows: lower lobe, middle lobe, upper lobe, lower lobe channel, middle lobe channel, upper lobe channel. The mark of a cell $C$ in the grid is defined as the product $p_1 \cdot p_2 \cdot p_3 \cdot p_4 \cdot p_5 \cdot p_6$ where $p_i$ is equal to the $i$-th prime number, if $C$ is inside the $i$-th object (in the given order) or $p_i = 1$ otherwise.

With the definition given for the marking, from the mark of a cell it is easy to determine to which lobes or channels it belongs (just use prime factorization). This information is used to paint the grid.

## 4.6 Simulations

In this section some realizations using SKE-SIM are presented.

### 4.6.1 Example 1

The training image used is given by the figure 4.17(a). From this image, using FIJI, the training skeleton is obtained and it is illustrated in the figure 4.17(b). This training skeleton is the source of information used to generate others skeletons.

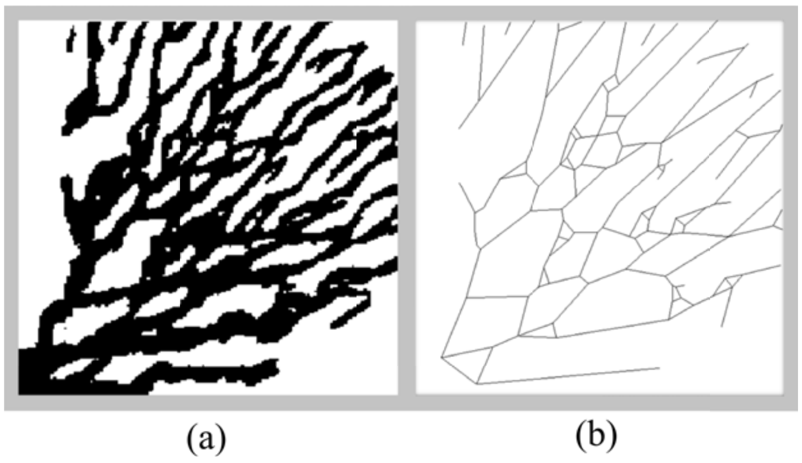


Figure 4.17: (a) Training image. (b) Training skeleton.

From the training skeleton given by the figure 4.17(b), a uniform distribution of the bifurcation angles and channel lengths are obtained and used to generate others skeletons. These new skeletons are constructed following the algorithm 3 explained in the section 4.6. In the figure 4.18 six simulations of skeletons are presented. The B-spline curves in plane $xy$ that limit the skeleton can be appreciated too.

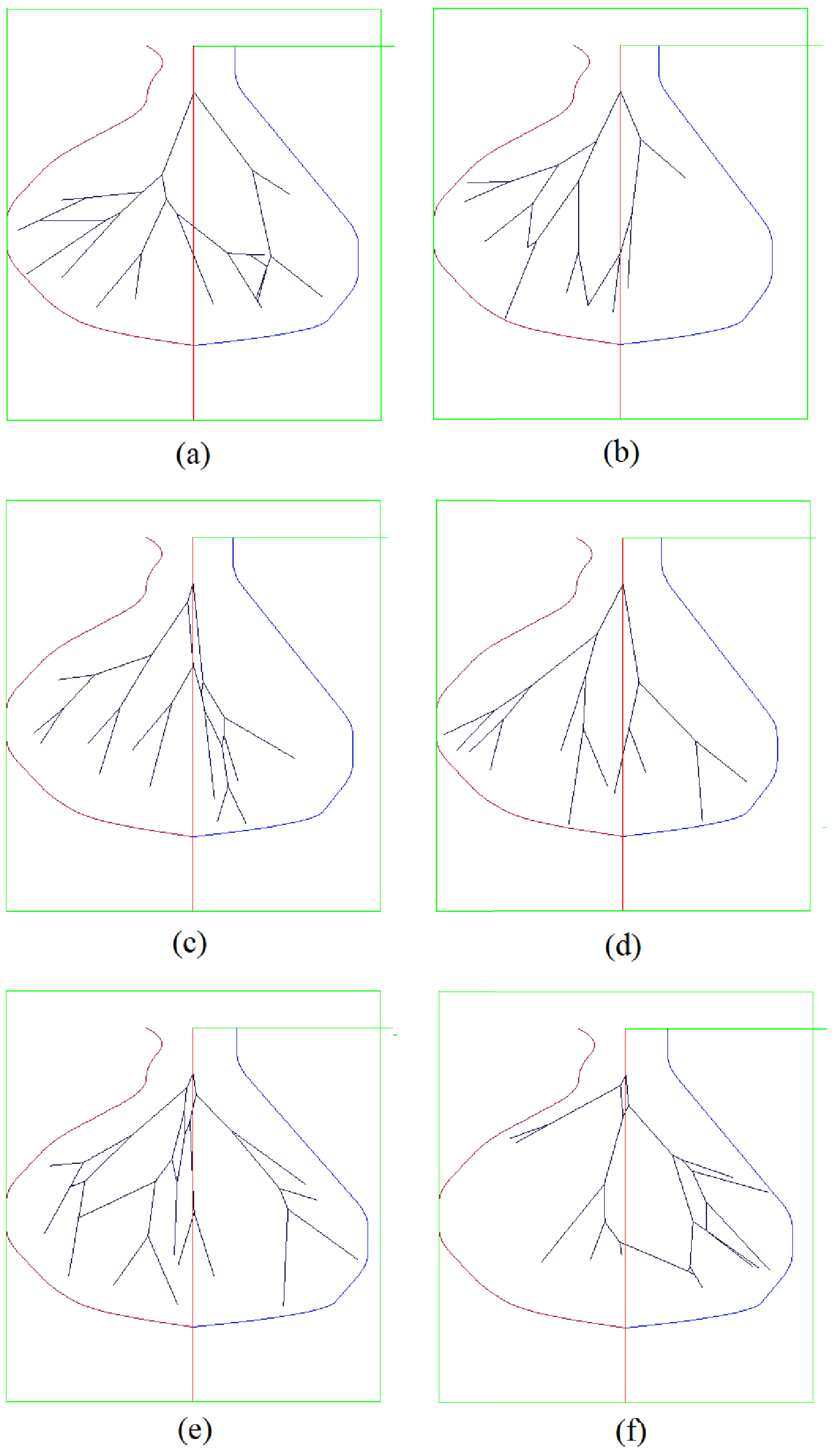


Figure 4.18: Six skeletons obtained using the training skeleton 4.17(b)

From the skeletons, that lie in the $xy$ plane, a 3D model is generated. In the figure 4.19 a simulation of one lobe and one channel system is presented. Figures 4.19(a, b) show only the channel system, in (a) an upper view is shown. In the figure 4.19(c), the same channel system is illustrated inside the lobe. The figure 4.19(d) shows a cross-section, perpendicular to the $x$ axis, of the total system, here the ellipsoidal form of the channels is appreciated.

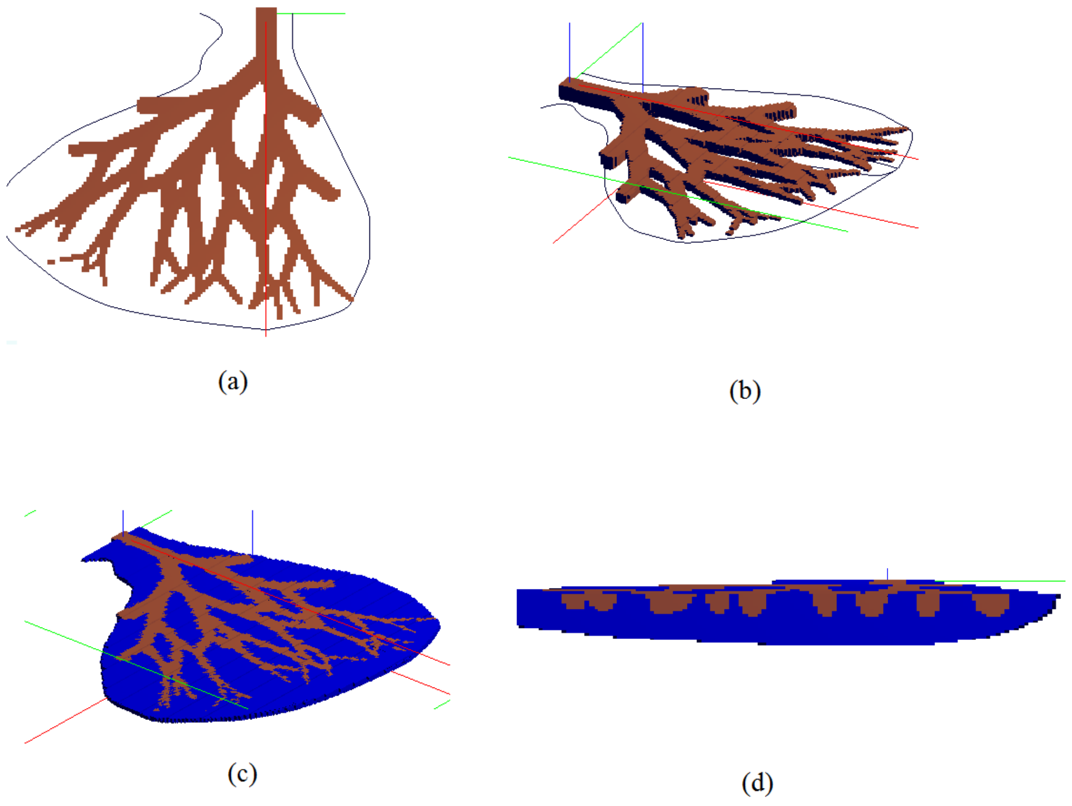


Figure 4.19: One channel system simulation.

In the figure 4.20, three channel systems were simulated inside three lobes. In the figure 4.20(a) an upper view of the channel set is presented. A different perspective of the same system is shown in the figure 4.20(b). Together with the channels, the lobes that contains them are drawn in the figure 4.20(c). In the figure 4.20(d), a lateral view of the three channels is shown. In this same image it can also be observed the b-spline curves in the $xz$ plane (these curves define the depth of the lobe).

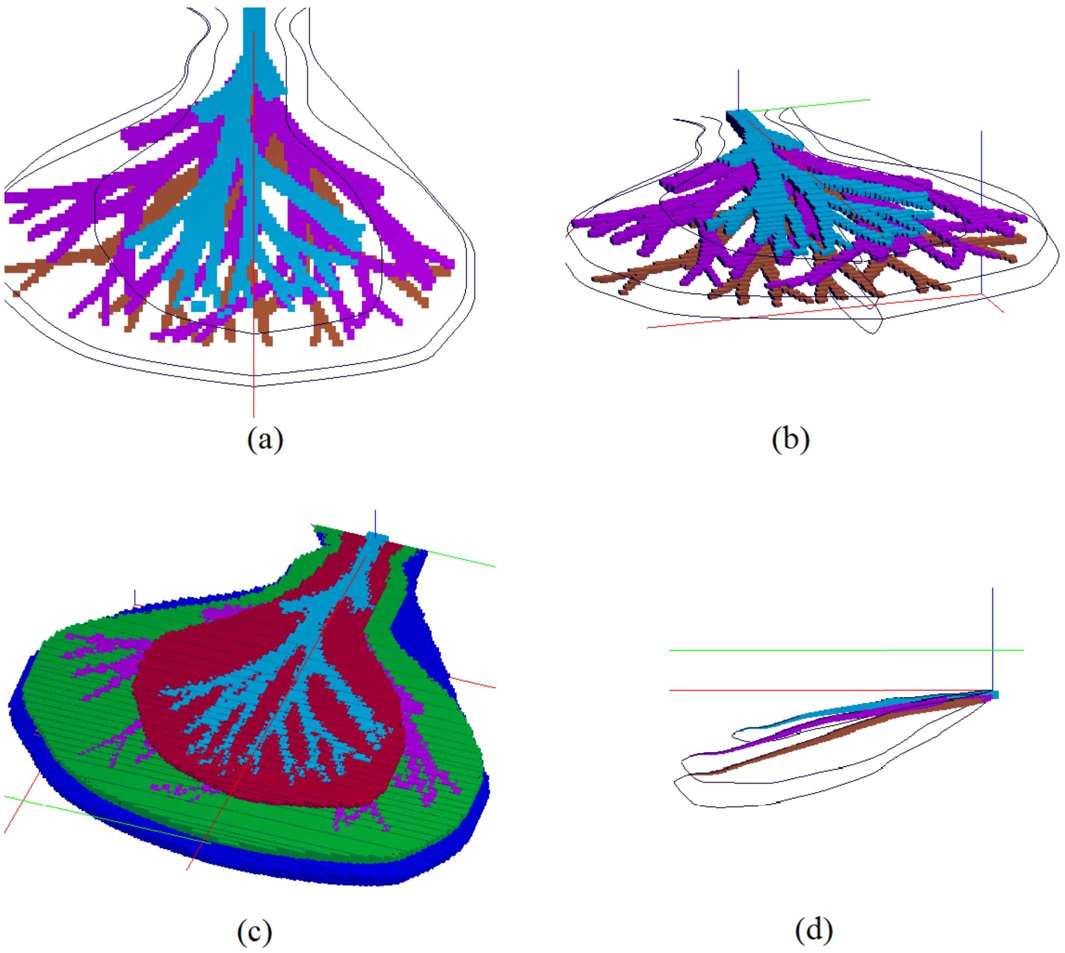
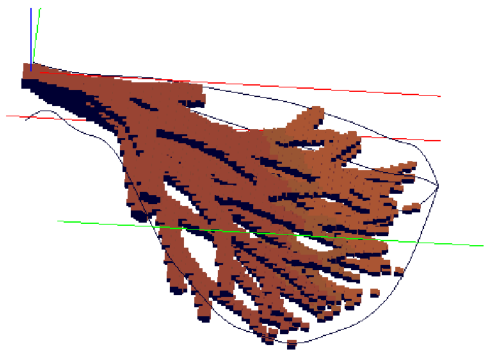
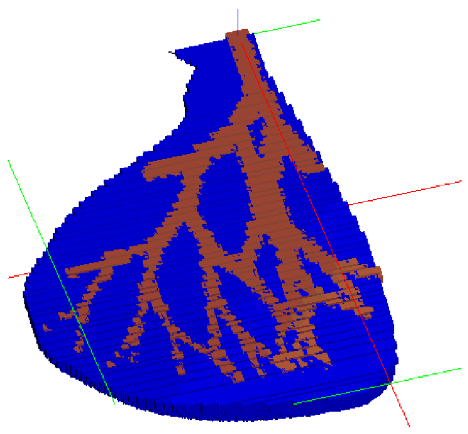


Figure 4.20: Three channels system simulation.

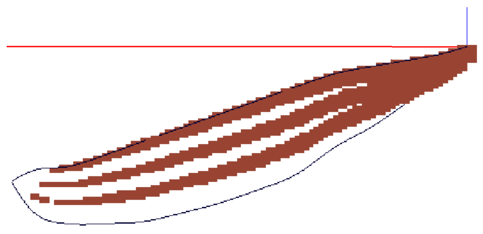
In the previous realizations, in each lobe one channel system was generated. The SKE-SIM simulation can be easily modified to allow the generation of more than one channel system in the same lobe. In the figure 4.21, three channels systems were simulated inside one lobe. In figure 4.21(a), only the three channels are visualized. In figure 4.21(b) the system is shown together with the lobe that contains them, in this case only the system in the surface of the lobe can be seen. In the figure 4.21(c), the lateral view (parallel to the plane $xz$) shows how the three systems are distributed inside a lobe, the black curve surrounding the systems are the b-spline curves in the plane $xz$.



(a)



(b)



(c)

Figure 4.21: Three channels system simulation in one lobe.

### 4.6.2 Example 2

For this example we use the same training image of the previous chapter 3.3, obtained from the alluvial fan given by the figure 3.15. The training image is shown in figure 4.22(a) and the training skeleton, obtained from the information computed using FIJI, is presented in the figure 4.22(b).

In the figure 4.23, six skeletons by applying SKE-SIM are shown. Remember that these skeletons lie in the plane $xy$ and from them the 3D modeling is performed. To generate this skeletons the information about the bifurcation angles and channel lengths, obtained from the training skeleton 4.22(b), was used.

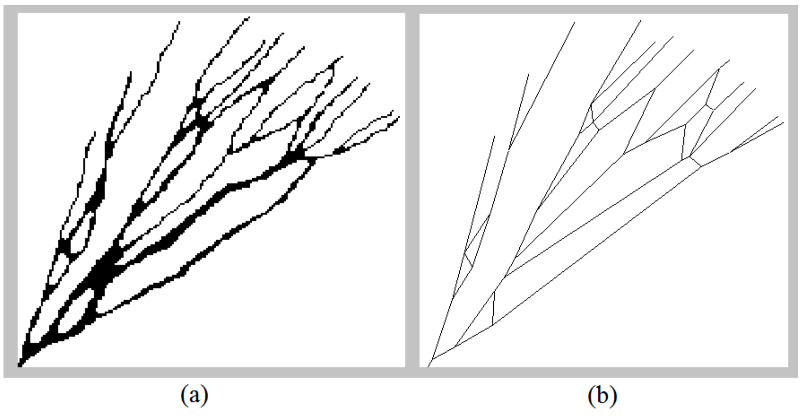


Figure 4.22: (a) Training image. (b) Training skeleton.

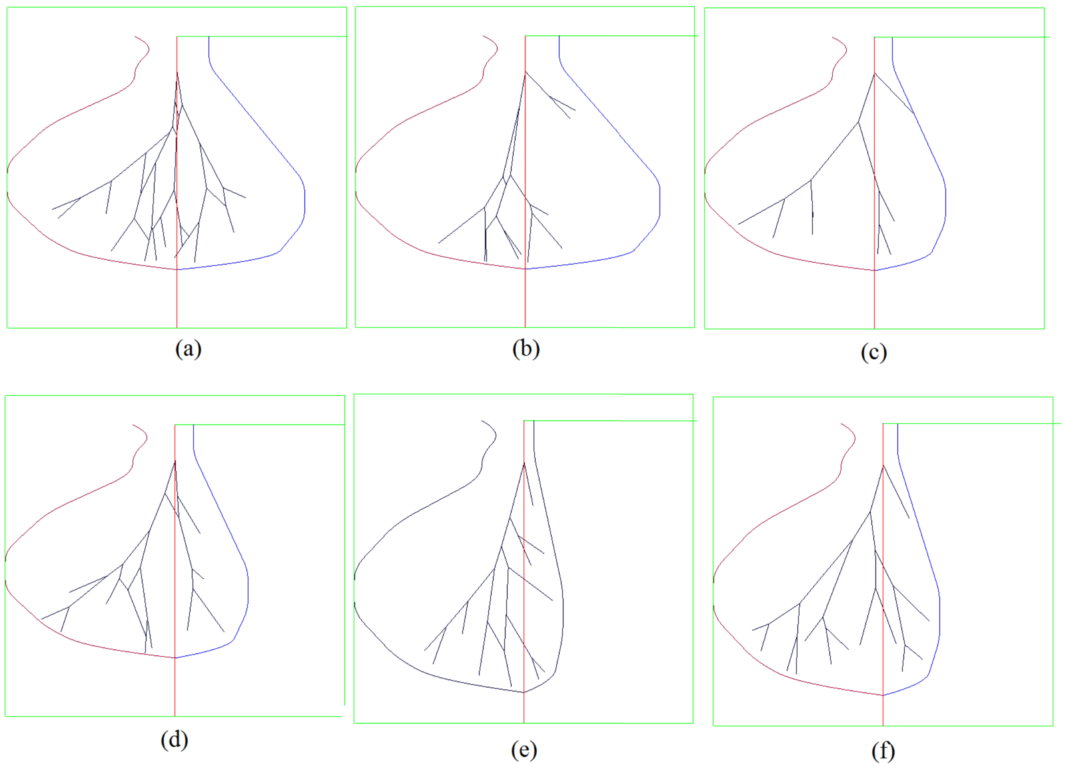


Figure 4.23: Six skeletons obtained using the training skeleton 4.22(b)

From the skeleton given by the figure 4.23(e), one 3D channel system is created inside a lobe. In the figure 4.24(a, b) only the channel system, limited by the B-spline curves in the $xy$ plane, is shown. In the figure 4.24(c, d) the same channel system is illustrated inside the lobe.

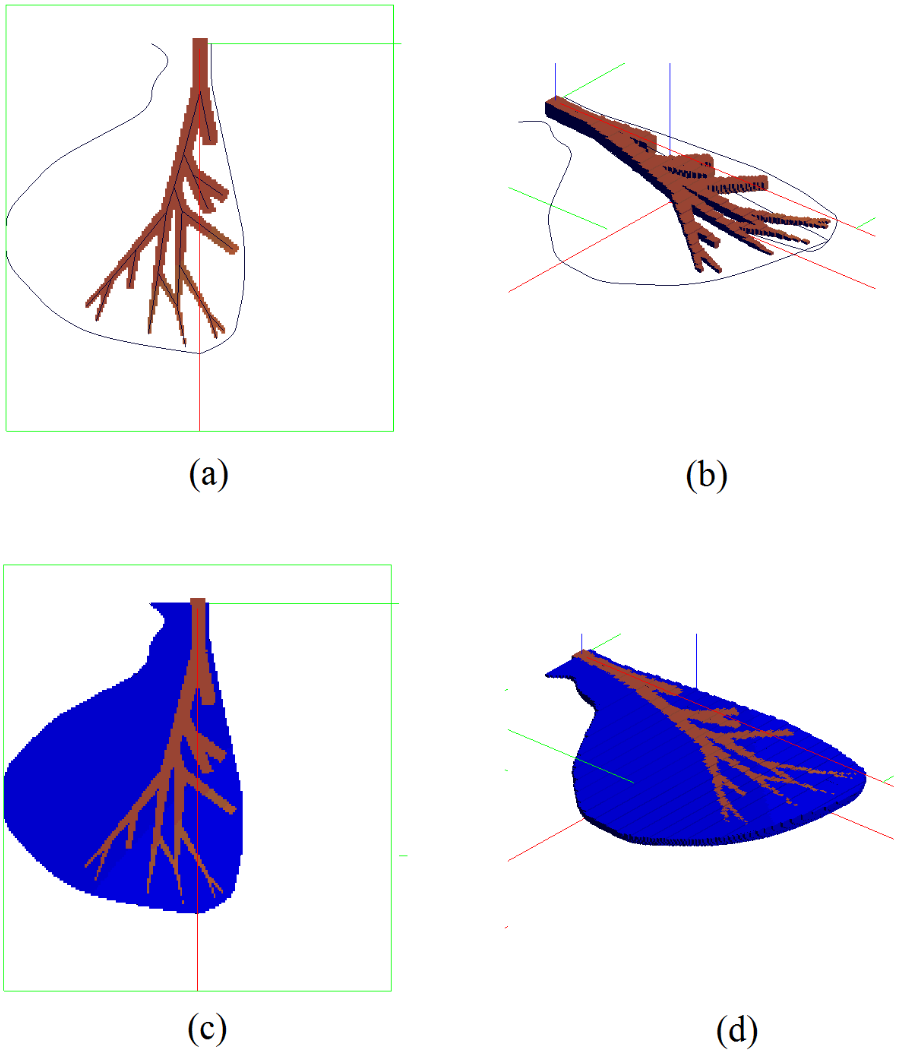


Figure 4.24: 3D realization from the skeleton in the figure 4.23(e)

In the figure 4.25, a depositional system of three channel systems is illustrated. In this figure the B-spline curves of the three lobes are observed. In the figure 4.25(a), an upper view of the system is shown. A lower view is presented in the figure 4.25(c). In the figure 4.26 the three channel systems of the figure 4.25 are shown inside three lobes, one lobe for each one. In the figure 4.26(c) a frontal view of the system is shown, there can be observed how the channels in superior positions erode the inferior channels.

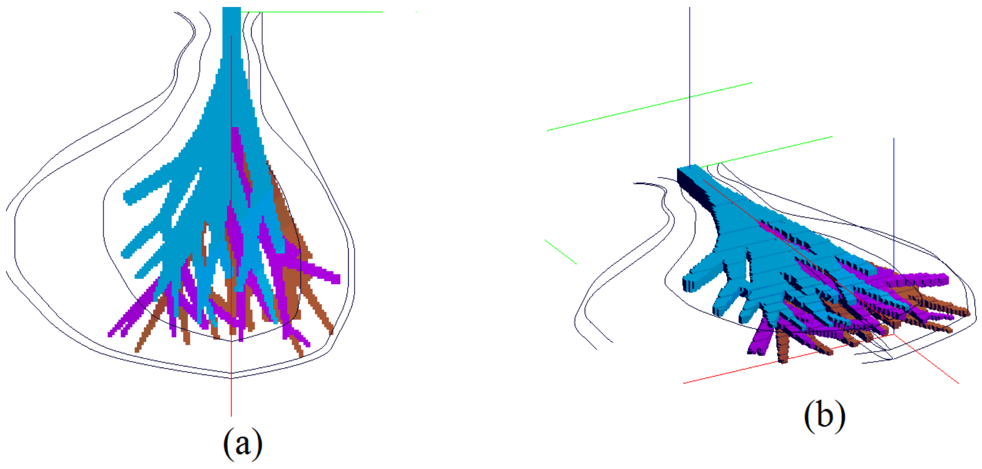
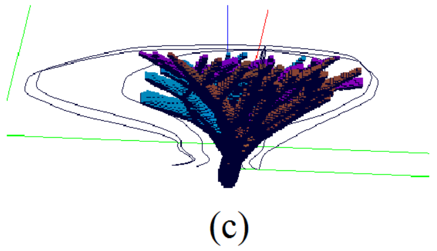



Figure 4.25: 3D realization of three channel systems limited by the B-spline curves.

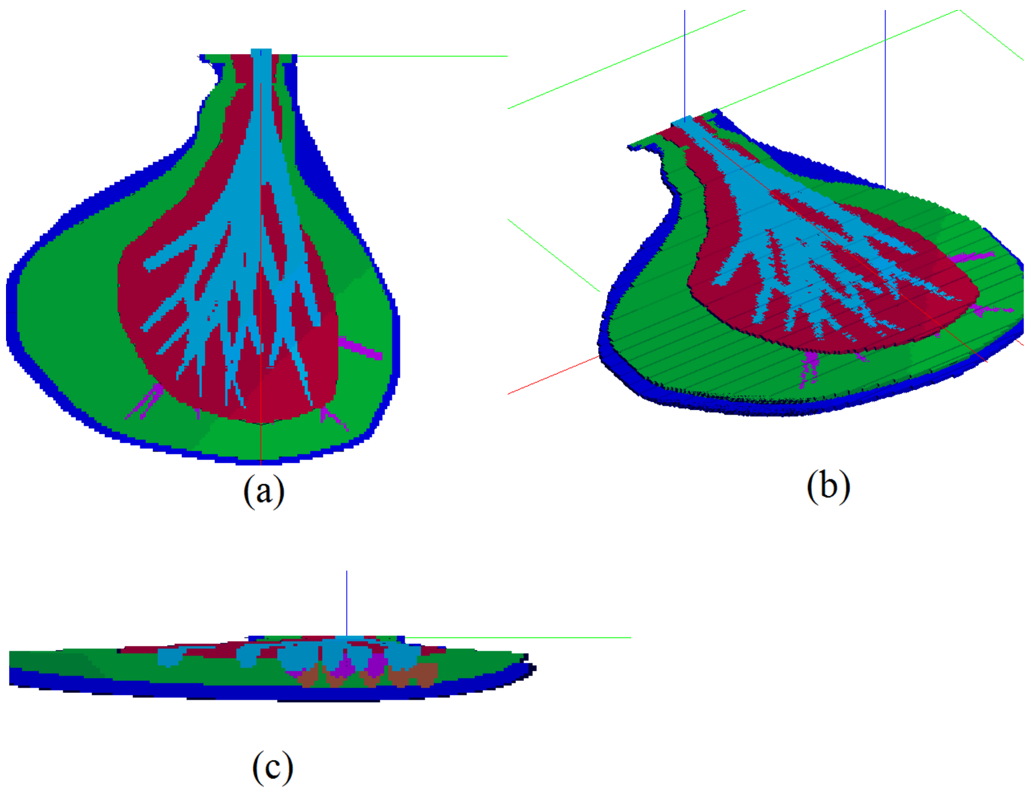


Figure 4.26: 3D realization of three channel systems inside its respective lobes.

In the figure 4.27, another realization is shown. Figures 4.27(a, b) present only the channel systems. The system together with the lobes are depicted in the figures 4.27(c, d). Figure 4.27(e) gives a frontal view.

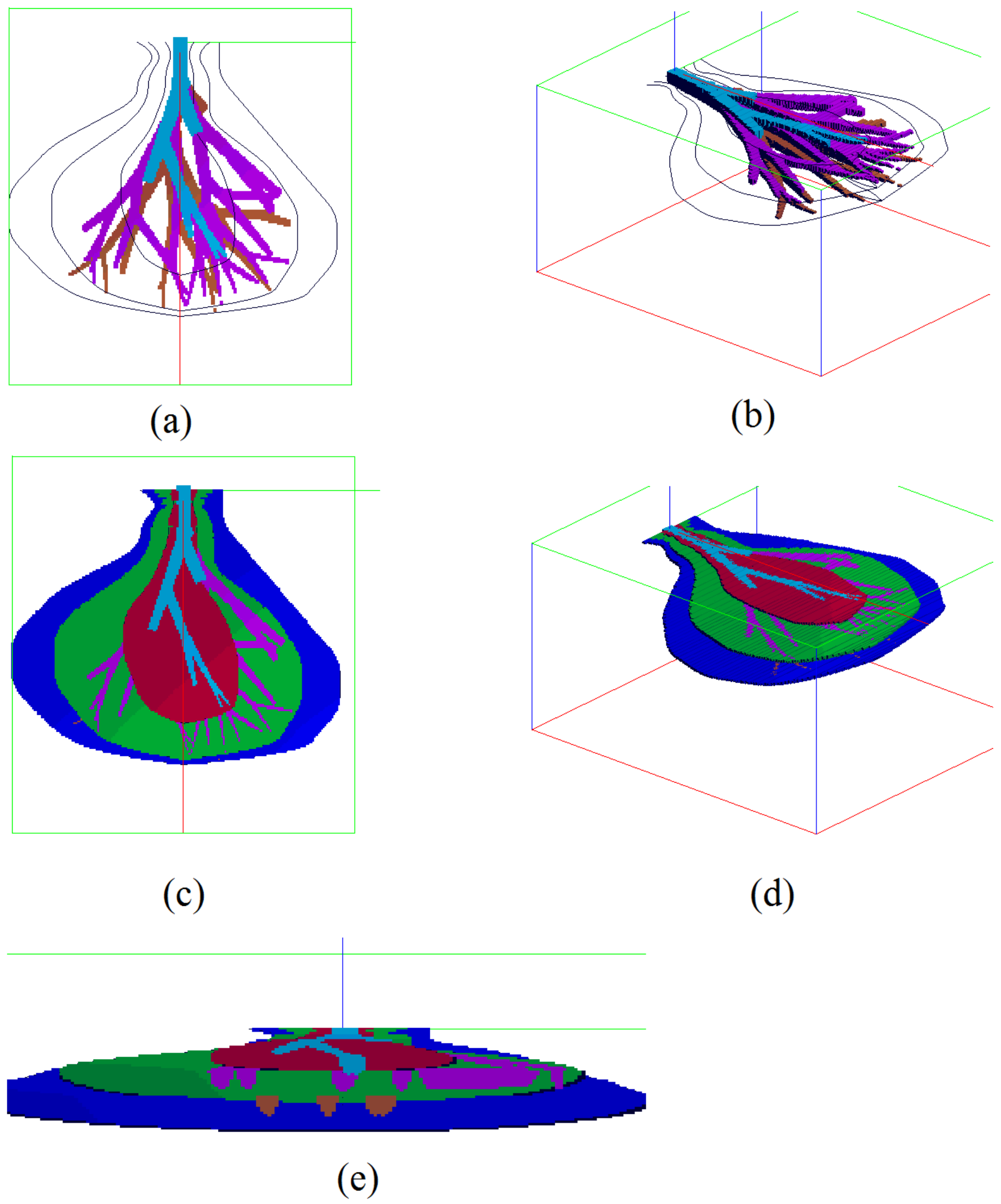


Figure 4.27: 3D realization of three channel systems inside its respective lobes.

## 4.7 Conclusions

In this chapter a 3D object-based modeling for turbiditic channels inside a lobe was presented. The lobe is created using the work of [16]. To build the channel system, information is required to determine the parameters values.

Our idea is to use a tree-like image that represents the projection to the plane $xy$ of the channel system. This is the training image and contains the basic geometry desired in the object. A linear approximation of this image is obtained, this is called training skeleton. From it, two probability distributions for the parameters values (bifurcation angles and length) are obtained. This is employed to generate new skeletons.

The generation of the skeletons is very simple. The skeleton is constructed sequentially. In each step, a couple of new edges emerge from each of the final nodes. The directions and length of those edges are determined using the probability distributions. Some rules are applied for especial edges or when an intersection occurs. The skeleton continues growing until a threshold is reached. Here, the boundary of the projection of the lobe to the plane $xy$ was used as constraint.

Once the skeleton is simulated, the 3D channel system is created inside the lobe. The idea is to give a volume to the skeleton, each edge is thickened, deepened and translated downward such that it lies inside the lobe.

There are two fundamental assumptions upon which the method is supported: the geometry of the channel system can be reasonably represented by a thickening of a unidimensional structure, the skeleton. Secondly, the distance traversed by a channel before something happens making it bifurcates is controlled by a random variable whose probability distribution does not depend on the position of the channel. This also happens for the bifurcation angles. Thus, these two quantities can be simulated using the same random variables (one for each quantity) no matter where the phenomenon occurs.

SKE-SIM is not a complex method, it does try to mimic the detailed physical behavior of the phenomena. Nevertheless, the results are visually appealing. The small time of simulation is one of the advantage of this method. The construction of the skeleton is very fast. The main factor increasing the simulation time is the number of cells in the grid.

Another advantage of this method is that the simulation (a 3D image) has a well-defined geometry. This helps in computing relations between the different parts of the channel system. For example, given a point in a channel the distance to its center can easily be obtained. This can be used to insert internal properties of the rocks in the channel system, as porosity and permeability.

One of the main steps in the method is to extract information from the training image. This is used to define the probability distribution of the parameters. This is a problem since finding a good training image is difficult.

One possibility that could be introduced to our method is to allow curves, not necessarily straight lines, connecting the nodes. However, notice that for the training image used in our simulations it happened that: given two bifurcation points, the length of the actual channel between them was, in average, less than 10% the distance between the nodes. This last remark justified our choice of straight segments.

One possible future research line would be to consider the modeling of a channel system based on the influence of the surface topography on the channel development. In this work the channel system was only restrained to be contained in the lobe. In the surface case the building procedure for the skeleton should consider the geometry of the surface. For example, one could imagine that the simulation grid comes together with a function indicating the high of the surface (interpreting the surface as the graph of a function from $\mathbb{R}^2$ to $\mathbb{R}$). Then the skeleton simulation uses this function to determine whether an edge can follow a certain path or should bifurcate.

An interesting possibility for future research could be the introduction of seismic information, this could be used as some sort of conditioning and it would play a role at estimating the bifurcation angles and channel lengths.

# Bibliography

[1] MARIETHOZ, G.; CAERS, J, ***Multiple-point geostatistics: Stochastic modeling with training images***, Wiley-Blackwell, 2015.

[2] DEUTSCH, C., TRAN, T, ***FLUVSIM: a program for object-based stochastic modeling of fluvial depositional systems***, Computers and Geosciences, volume 28, pp. 525-535, 2002.

[3] ARPAT, B, ***Stochastic simulation with patterns***, Doctoral dissertation, Stanford University, 2005.

[4] ALZATE, Y, ***Object-based Modelling of Turbidity Lobes using non parametric B-Splines***, Master dissertation, Pontifícia Universidade Católica do Rio de Janeiro, 2016.

[5] ARPAT, B.; CAERS, J, ***Stochastic simulation with patterns***, Mathematical Geology , 39(2):177-203, 2007.

[6] HONARKHAH, M.; CAERS, J, ***Direct pattern-based simulation of non-stationary geostatistical models***, Math Geosci 44: 651 - 672, 2012.

[7] PYRCZ, M.; DEUTSCH, C, ***Geostatistical reservoir modeling***, Oxford university press, 2014.

[8] BARRETO, A.; FARIAS, R.; KRUEL, R.; LANZARINI, W.; LOPES, H.; PESCO, S.; TAVARES, G, ***Petbool: simulador estocástico da geometria e heterogeniedades de reservatorios petroliferos usando o método booleano***, Applied Mathematical Sciences, Primeiro Workshop em Simulação e Engenharia de Reservatórios, p.26-27, 1995.

[9] LOPES, H.; PESCO, S.; POLETTO, C.; TAVARES DOS SANTOS, G, ***Petbool: a software for stochastic modeling of geological objects***, Chapter in TERRA NOSTRA Series, Proceeding of the mathematical geology'02, IAMG, pp. 179-184, 2002.

[10] MARQUES, I.; ALMEIDA, J.; QUININHA, M.; LEGOINHA, P, ***Combined Use of Object-Based Models, Multipoint Statistics and Direct Sequential Simulation for Generation of the Morphology, Porosity and Permeability of Turbidite Channel Systems***, Springer International Publishing. Geostatistics Valencia 2016, Quantitative Geology and Geostatistics 19, DOI 10.1007/978-3-319-46819-8-43, 2017.

[11] GUARDIANO, F.; SRIVASTAVA, M, ***Geostatistics - Troia'92***, Springer Netherlands, Dordrecht, Ch. Multivariate Geostatistics: Beyond 516 Bivariate Moments, pp. 133-144, 1993.

[12] STREBELLE, S, ***Conditional simulation of complex geological structures using multiple-point statistics***, Mathematical Geology, 34: 1-22, 2002.

[13] MARIETHOZ, G.; LEFEBVRE, S, ***Bridges between multiple-point geostatistics and texture synthesis: Review and guidelines for future research***, Computers & Geosciences 66: 66-80, 2014.

[14] ZHANG, T.; SWITZER, P.; JOURNEL, A, ***Filter-based classification of training image patterns for spatial simulation***, Mathematical Geology, 38, 63-80, 2006.

[15] HONARKHAH, M.; CAERS, J, ***Stochastic simulation of patterns using distance-based pattern modeling***, Math Geosci 42:487-517, 2010.

[16] MOURA, P.; LABER, E.; LOPES, H.; MESEJO, D.; PAVANELLI, L.; JARDIM, J.; THIESEN, F.; PUJOL, G, ***LSHSIM: A Locality Sensitive Hashing Based Method for Multiple-Point Geostatistics***, Computers and Geosciences. Volume 107, pages 49-60, 2017.

[17] DE VRIES, L.M.; CARRERA, J.; FALIVENE, O.; GRATACÓS, O.; SLOOTEN, L.J, ***Application of multiple point geostatistics to non-stationary images***, Math Geosci 41: 29-42, 2009.

[18] HONARKHAH, M, ***Stochastic simulation of patterns using distance-based pattern modeling***, Doctoral dissertation, Stanford University, 1993.

[19] STREBELLE, S.; PAYRAZYAN, K.; CAERS, J, ***Modeling of a deepwater turbidite reservoir conditional to seismic data using multiple-point geostatistics***, Society of petroleum Engineers Paper number 77425, 2002.

[20] TJELMELAND, H, ***Models in Reservoir Characterization and Markov Random Fields for Compact Objects***, Doctoral dissertation, Norwegian University of Science and Technology, Trondheim, Norway, 1996.