Computers & Geosciences 105 (2017) 158–168

Contents lists available at ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

Research paper

Stochastic simulation of channelized sedimentary bodies using a constrained L-system

Guillaume Rongier[a,b,*], Pauline Collon[a], Philippe Renard[b]

[a] GeoRessources (UMR 7359, Université de Lorraine / CNRS / CREGU), Vandœuvre-lès-Nancy F-54518 France
[b] Centre d'Hydrogéologie et de Géothermie, Université de Neuchâtel, 11 rue Emile-Argand, Neuchâtel 2000, Switzerland

ARTICLE INFO

Keywords:
Sedimentary system
Channel
Stochastic simulation
Lindenmayer system
Data conditioning
Constraints

ABSTRACT

Simulating realistic sedimentary bodies while conditioning all the available data is a major topic of research. We present a new method to simulate the channel morphologies resulting from the deposition processes. It relies on a formal grammar system, the Lindenmayer system, or L-system. The L-system puts together channel segments based on user-defined rules and parameters. The succession of segments is then interpreted to generate non-rational uniform B-splines representing straight to meandering channels. Constraints attract or repulse the channel from the data during the channel development. They enable to condition various data types, from well data to probability cubes or a confinement. The application to a synthetic case highlights the method's ability to manage various data while preserving at best the channel morphology.

Introduction

The presence of channelized sedimentary bodies constitutes a key structuring element for the connectivity of a reservoir. These bodies are composed of heterogeneous deposits that can contain fluids or act as flow barriers. Such barriers may compartmentalize the reservoir and make its exploitation harder, especially when combined with sealing faults Gainski et al. (2010). Channel modeling can help to study and anticipate more precisely the flow behavior. It requires a balance between geological concepts and data conditioning, which remains an issue in stochastic simulation.

Stochastic simulation methods are usually split in two categories: cell-based and object-based methods. Cell-based methods (e.g., Deutsch and Journel, 1992; Galli et al., 1994; Strebelle, 2002) attribute a sedimentary body type – e.g., channel – to each cell within a grid, based on a prior model. It makes well data conditioning easy, but the channel continuity is rarely preserved. The resulting connectivity differs from the prior model or from object-based methods (Rongier et al., 2016). Object-based methods (e.g., Viseur, 2001; Deutsch and Tran, 2002; Pyrcz et al., 2009) rely on a geometrical representation of the channels, with parameters such as their width or their wavelength. They preserve the channel continuity. However, data conditioning is difficult, because of the elongated shape of channels and the poor flexibility of their representations.

In numerical biology, tree and root simulation also relies on object-based approaches (e.g., Prusinkiewicz and Lindenmayer, 1996; Leitner et al., 2010; Longay et al., 2012). A formal grammar – the Lindenmayer system or L-system (Lindenmayer, 1968) – mimics tree and root growth to simulate the related objects. An interesting development of such methods introduces the environment influence to improve the simulation realism: influence of gravity, influence of the sun light distribution, influence of other trees, etc. (e.g., Mvech and Prusinkiewicz, 1996; Streit et al., 2005; Taylor-Hell, 2005). Despite this ability to integrate various data, very few works use formal grammars to simulate channels. Hill and Griffiths (2009) rely on an analog model to define some rules for channel stochastic simulation, with a conditioning limited to well data or channel segments interpreted on seismic data.

In this paper, we present a new method to simulate channels based on L-systems. L-system rules control the channel morphology to simulate straight to sinuous channels (Section 1). During the simulation, constraints influence the channel growth to condition various data (Section 2). The purpose of this research is to facilitate data conditioning as compared to other object-based approaches while keeping their main added-value: the preservation of the channel shape. A synthetic case study applies the method to the simulation of submarine channels within an incised valley (Section 3). It leads to a discussion on the channel simulation with L-system and constraints (Section 4).

1. L-system for channel stochastic simulation

This object-based method relies on a L-system to give form to a parameterized channel geometry.

* Corresponding author at: GeoRessources (UMR 7359, Université de Lorraine / CNRS / CREGU), Vandœuvre-lès-Nancy, F-54518 France.
E-mail address: guillaume.rongier@univ-lorraine.fr (G. Rongier).

http://dx.doi.org/10.1016/j.cageo.2017.05.006
Received 12 December 2016; Received in revised form 16 May 2017; Accepted 16 May 2017
Available online 18 May 2017
0098-3004/

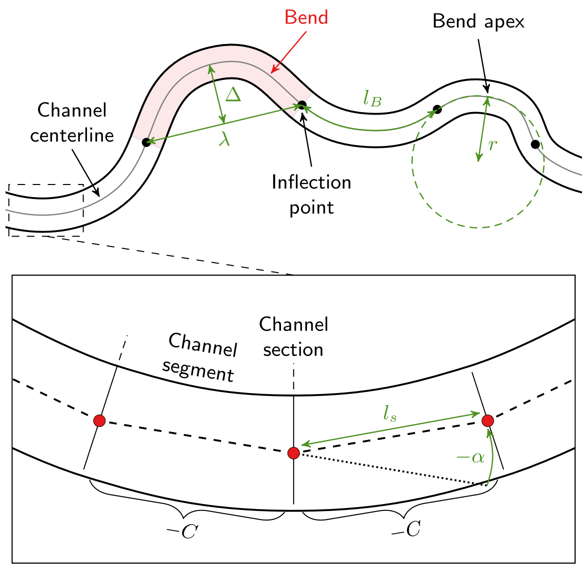

Fig. 1. Channel discretization for L-system simulation. The red dots locate the channel sections obtained from the interpretation. l_s is the distance between two successive sections, i.e., the channel segment length. α is the angle between two successive sections, i.e., the change of orientation between two successive channel segments. The other parameters characterize a channel bend, with l_B the bend length, λ the bend half-wavelength, Δ the bend amplitude, and $r = 1/c$ the radius of curvature – radius of a circle fitted at the bend apex – with c the curvature.

1.1. Channel object definition and simulation principle

A channel object is built upon Non-Uniform Rational B-Splines (NURBS) – a generalization of Bézier curves (Piegl and Tiller, 1995) – as defined by Ruiu et al. (2015). In this model, a channel appears as a discretized object composed of channel segments separated by transversal channel sections (Fig. 1). A section is at the end of its respective segment. An orientation and a distance from the previous section characterize a given section.

A L-system builds the succession of sections and determines their location. A sequential Gaussian simulation (Deutsch and Journel, 1992) simulates a width and thickness for each channel section, with a possible influence of the channel curvature. The section locations, widths and thicknesses are the foundation for the NURBS representation. This method is designed for a classical modeling workflow, which consists in building a geological grid from a structural model, and simulating the channels inside the parametric space of this grid (Dubrule et al., 1997).

1.2. Brief introduction to L-systems

The Lindenmayer system is a formal grammar designed by Lindenmayer (1968) to simulate the development of filamentous organisms. Since then, it has been expanded to simulate the development of plants (e.g., Mvech and Prusinkiewicz, 1996; Prusinkiewicz et al., 2001; Leitner et al., 2010). A L-system aims at rewriting an initial string, the axiom, with production rules, all composed of letters from a predefined alphabet. The rules include a set of letters to replace, the predecessor (*pred*), and a set of replacement letters, the successor (*succ*):

$$lc < pred(param) > rc\colon cond \xrightarrow{p} succ(param')$$

The application of a rule may depend on the letters before and after the predecessor, called respectively the left and right context (*lc* and *rc*), on a probability (p), and on parameters (*param* and *param′*) and conditions (*cond*). The following example contains an axiom ω and three production rules p_1, p_2, and p_3:

$$\begin{aligned}
\omega &: b(0) \\
p_1 &: a(h) > b(h) \xrightarrow{0.75} b(h+1) \\
p_2 &: a(h) > b(h) \xrightarrow{0.25} a(h)a(h) \\
p_3 &: b(h) \quad : h < 1 \longrightarrow a(h)b(h)
\end{aligned}$$

The rules may rewrite the axiom as follows:

$$\begin{aligned}
\mu_0 &: b(0) && (\omega) \\
\mu_1 &: a(0)b(0) && (p_3) \\
\mu_2 &: b(1)a(0)b(0) && (p_1 \& p_3) \\
\mu_3 &: b(1)a(0)a(0)a(0)b(0) && (p_2 \& p_3)
\end{aligned}$$

The resulting string constitutes a sequence of commands, which control an interpreter called a turtle (Prusinkiewicz, 1986). The turtle's state is represented by a position vector $\overrightarrow{P}$ and an orthonormal coordinate system centered on this position, with:

- $\widehat{H}$ the forward direction.
- $\widehat{L}$ the left direction.
- $\widehat{U}$ the up direction.

This state is built from an initial position and orientation. Then, the letters update it and draw object elements, such as segments, along the way to progressively build the object. More details about L-systems can be found in Prusinkiewicz and Lindenmayer (1996).

1.3. Alphabet for channel simulation

A channel is decomposed into bends separated by inflection points (Fig. 1). A bend is a succession of channel segments whose orientations change along the same direction. Seven letters constitute the alphabet to model this succession of segments (Table 1).

A channel divides into two branches that grow in opposite directions. Brackets [and] symbolize this branching structure, with the letters of the first branch in between. The letter C builds the channel object, while the letters + and - control the channel sinuosity. A channel segment is symbolized by a $\pm C$ pair. Thus, a channel is a succession of $\pm C$ letters, with $\pm$ being + or − but remaining the same along the bend. The letter I has no influence on the string rewriting. It only states the initial position during the geometrical interpretation. The letter T has no influence on the geometrical interpretation. It only ensures the channel growth during the rewriting.

1.4. Production rule definition

The rules aim at building bends from channel segments. This includes converting the input parameters that describe the channel morphology into L-system parameters.

Table 1
L-system alphabet for channel simulation.

Letters (parameters)	Interpretation
I	First letter of a L-system string that contains the initial position
T	Do nothing
$C(l_s)$	Move forward of a length l_s and draw a channel segment between the new position and the former one
$+(\alpha)$	Turn left by an angle α
$-(\alpha)$	Turn right by an angle α
[	Start a branch, i.e., push the current turtle's state into a stack
]	End a branch, i.e, pop a state from the stack to be the current turtle's state