International Journal of Geographical Information Science
Vol. 23, No. 3, March 2009, 295–318

Research Article

Representing geographical objects with scale-induced indeterminate boundaries: A neural network-based data model

J. L. SILVÁN-CÁRDENAS*†‡, L. WANG§ and F. B. ZHAN†

†Texas State University-San Marcos, Texas Center for Geographic Information Science, Department of Geography, 601 University Dr., ELA 139, San Marcos, TX 78666, US
‡Centro de Investigación en Geografía y Geomática, Ing. "Jorge L. Tamayo", Contoy 137, Lomas de Padierna, Mexico, D.F. 14240, MEX
§Department of Geography, University at Buffalo, the State University of New York, 105 Wilkeson Quad., Buffalo, NY 14261, USA

(*Received 11 June 2007; in final form 03 January 2008*)

The degree of uncertainty of many geographical objects has long been known to be in intimate relation with the scale of its observation and representation. Yet, the explicit consideration of scaling operations when modeling uncertainty is rarely found. In this study, a neural network-based data model was investigated for representing geographical objects with scale-induced indeterminate boundaries. Two types of neural units, combined with two types of activation function, comprise the processing core of the model, where the activation function can model either hard or soft transition zones. The construction of complex fuzzy regions, as well as lines and points, is discussed and illustrated with examples. It is shown how the level of detail that is apparent in the boundary at a given scale can be controlled through the degree of smoothness of each activation function. Several issues about the practical implementation of the model are discussed and indications on how to perform complex overlay operations of fuzzy maps provided. The model was illustrated through an example of representing multi-resolution, sub-pixel maps that are typically derived from remote sensing techniques.

Keywords: Fuzzy sets; Artificial neural networks; Indeterminate boundaries

1. Introduction

The representation of geographic phenomena in digital databases is one of the most fundamental issues in geographic information science (GISc) (Zhan 1998, Egenhofer *et al.* 1999, Mennis *et al.* 2000). The prevalent view of spatial knowledge representation follows a dyadic object-field conceptualization of geographic space (Couclelis 1992, Mark and Frank 1996, Peuquet 1988, Cova and Goodchild 2002). According to this conception, one may reason about the world as being populated of discrete entities called objects (object view) or as a continuum of named attributes (field view). Schuurman (1999) has pointed out that this conceptualization of the space was further articulated to match early computer capacities, which resulted in the vector and raster data models.

*Corresponding author. Email: jlsilvan@txstate.edu

International Journal of Geographical Information Science
ISSN 1365-8816 print/ISSN 1362-3087 online © 2009 Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/13658810801932021

Another crucial issue of transforming geographic entities into database objects is uncertainty. Uncertainty may operate at various levels of the cognitive and operational processes (Couclelis 1996) and along any of the quantified dimensions: space, time and theme (Usery 1996). It may be, for example, due to a weak match between concept and reality (vagueness), indiscernibility due to mixture properties (Ahlqvist *et al.* 2003), random fluctuations in data capture (Cheung *et al.* 2004, Shi 1998), neglect of time-induced spatial change (Burrough 1996), and many other generalization processes (Blakemore 1984, Ahlqvist *et al.* 2000). In most cases, the degree of uncertainty of geographical entities is in close relation to the scale of their observation and representation (Couclelis 1996, Goodchild 2001).

Modeling spatial objects with indeterminate boundaries has received considerable attention by the GISc community. Most of the available alternatives for modeling objects with indeterminate boundaries are grounded on either probability theory or fuzzy set theory (Schneider 1999). *Probabilistic models* (e.g. Blakemore 1984, Shi 1998, Cheung *et al.* 2004) predominantly model positional and measurement uncertainties. Probability theory can represent uncertainty and defines the membership grade of an entity in a set by a statistically defined probability distribution function. For example, neglect of time-induced change of the water level of a lake may lead to the need of a probabilistic boundary, where each point is assigned the probability that water is present at the point in any time. For a probabilistic model to achieve a good estimate of the probability distribution function, the observation of various realizations is required. This approach generally makes any practical implementation of probabilistic models for handling the boundary phenomena difficult.

The alternative for handling the boundary phenomena is through *fuzzy models* (e.g. Usery 1996, Wang and Hall 1996, Zhan 1998, Schneider 1999, Zhan and Lin 2003), which are all based on fuzzy set theory. This approach predominantly model vagueness, as it describes the admission of the possibility, given by a so-called membership function, that an individual is a member of a set. It can handle the imprecision of meaning of a concept and can be used to model the boundary of objects such as mountains, valleys and oceans. Ahlqvist *et al.* (2000) have pointed out that one major obstacle of fuzzy set-based uncertainty handling is determining the membership function for a given feature. Furthermore, the implementation of membership functions has been generally limited to raster data models for pragmatic reasons. This, in turn, has made it difficult to create a computational framework for handling fuzzy spatial objects, with explicit linkages between fuzziness, level of detail and scale.

Given that the traditional computing paradigm poses a significant constraint in the representation of objects with indeterminate boundaries, we advocate for a bottom-up approach that assumes a computing paradigm based on neural network models. Artificial neural networks (ANNs) are trainable structures, originally inspired on the biological brain, that have been tremendously simplified to facilitate their implementation with conventional hardware. In spite of their over-simplicity as models of the biological brain, ANNs emulate some essential behavior of biological neural networks. Unlike conventional computers, where memory and data are perfectly separable, neural networks have embedded memory and processing capabilities in each single processing unit (neuron). This characteristic makes them highly parallelizable, adaptable and fault-tolerant. These models enjoy high reputation in solving complex non-linear mapping problems, such as in pattern recognition and classification tasks (Looney 1997). They are also commonly used as predictors or decision support systems across several fields, where the geospatial sciences are not excluded (Dowla and Rogers 1995, Foody 1996, Openshaw and Openshaw 1997). In many geographical applications, ANNs are primarily used for the production of land-use land-cover (LULC) maps from remotely sensed imagery. It is within the context of LULC classification that ANNs have largely been used to build fuzzy partitions of geographic space (Foody 1996, Cheng *et al.* 2001). The focus of that kind of study is on transforming spectral measurements made by remote sensors onto a set of membership values that, at the end, are represented in a raster format. In contrast, our aim here is to upgrade the ANN from 'processing tool' to 'representation artifact' in the hope it may help represent geographic objects with indeterminate boundaries at multiple scales.

Traditionally, an ANN functions as a 'black box' device, as it is not generally possible to explain how the parameters of the system being represented are encoded in the network parameters. This characteristic makes data representation in neural networks sound bizarre to many fields. Although data representation has long been recognized as one of the two major themes in neural-network research, the mathematical intractability of a general case has largely shifted the interest to learning algorithms (Anderson 1990). Within GISc, and to the best of the authors' knowledge, the use of ANNs as representation artifacts is void, with a couple of exceptions (Silván 2005, 2006). Silván (2005) developed an algorithm to build a minimal neural network for representing polygon layers. The method used a constructive solid geometry representation (CSG, Dovkin *et al.* 1988, Walker and Snoeyink 1999) to progressively encode polygon edges and its nested structure into network parameters. Latter on, Silván (2006) extended this representation to consider fuzzy points and fuzzy lines, and loosely discussed the link between the degree of fuzziness and the scale of representation. In the present study, a neural network structure is proposed as a data representation model of objects with scale-induced indeterminate boundaries. A new algorithm for building the neural network representation of polygons is introduced, which produces less distortions under scaling of the activation function at the expense of increasing the number of units. Although the main focus is on representing polygonal regions that are traditionally stored in vector format, the construction of fuzzy points and fuzzy lines is also illustrated.

2. Geographic objects with scale-induced indeterminate boundaries

Every geographical object exists only at a range of scales (*operational scale*) that is intrinsic to the nature of the object, at least in an empirical sense (Couclelis 1996, Bian 1997). This range of scales describes the spatial extents within which a phenomenon comes to existence; it gives a notion of its *size*, e.g. a forest operates at a larger scale than individual trees. A similar statement can be said for the temporal dimension: a phenomenon's range of temporal scales gives a notion of its duration in time. On the other hand, its observation is generally restricted to an independent range of scales of an observation device or process. This range of scales is generally specified by the extent of the smallest observable piece of landscape at one end (e.g. spatial resolution, patch size, grain or cell size, etc.) and by the largest observable unit at the other end (e.g. geographic scale, extent, domain, field of view, etc.). Although some common terms, such as resolution and spatial extent, can clearly imply a range of sizes of observable patterns, they are somewhat tied to the physical limitations of the observation process and generally linked to a particular representation. It is noticeable that the observation of geographic features is not only constrained by physical limitations, but also laden of subjective criteria and

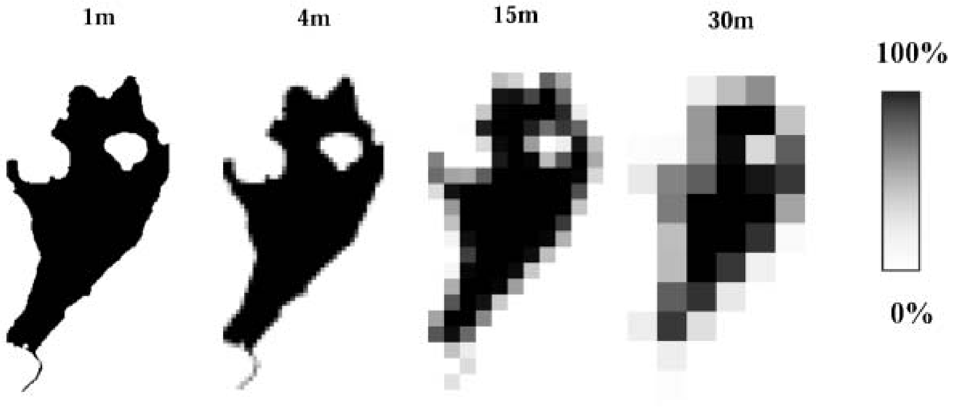


Figure 1. This image shows sub-pixel fraction maps of a lagoon delineated at various spatial resolution. The gray level denotes the area percentage covered by the lagoon within the cell, whereas the cell size is shown at the top of each map.

interests. To more generally reflect the idea of detail filtering by both physical and cognitive processes at the observation level, the concepts of *inner scale* and *outer scale*[1] shall be used to refer to the smallest and largest observable units.

One kind of undetermined boundary arises when the inner scale increases beyond the finest detail of the spatial phenomenon. A typical example is the delimitation of a geographic feature through satellite imagery. The lower the sensor resolution (larger inner scale), the fuzzier or uncertain the boundary is. Figure 1 shows an example of fraction maps of a lagoon delineated by means of remote sensors of various spatial resolution. The level of detail exhibited in the boundary is determined by the resolution of the sensor employed.

Since the optimal observation scale is usually unknown *a priori*, or simply there is no single optimal scale, then it is necessary to delineate the object at various scale levels. Presumably, many complex phenomena do not manifest at a single scale (or a narrow range of scales), but over a wide range of scales, and thus are observable at any overlapping range of scales. For instance, Couclelis (1996) noticed that extensive entities such as ocean, forests, prairies, geological formations, may be subdivided within very wide limits and still maintain their identity. The cognitive process involved on the object's identification and formalization further stresses the need for a multiscale representation. Consider, for example, the delimitation of a forest. Let us assume for a moment there is no physical constraint in the observation scale range. The forest category at each point can be based on the existence of a tree at that location (assuming each tree has perfectly identifiable boundaries). From this definition, it turns out that a single tree can be called forest, and clear spaces between trees are necessarily non-forest. Even though this definition of forest stand may provide high amount of detail in the boundary, it may not be generally acceptable. A more general definition can be based on the tree density within a moving window (filter or kernel). This definition can lead to smoother boundaries (with less details), as it excludes isolated or sparsely distributed trees, while including small non-vegetated patches that are contiguous to dense stands. Evidently, there is no apparent reason to restrict the

[1] The terms inner scale and outer scale were first coined in the context of the scale-space theory of vision (*see* e.g. Florack *et al.* 1994).

kernel size to a single value, as tree crown size and tree distribution may vary within the same forest. Similar arguments can be applied to many other variables, such as slope and population density, for which there is no single correct definition, but only a definition for each given inner scale associated to a kernel size (Goodchild 2001).

In sum, the level of uncertainty, as well as the level of detail, in the boundary of most geographic objects is linked to some scaling operations, mostly related to its observation and representation. The set of representations at all levels of detail comprise a family of maps, which, in principle, has an infinite number of members. Such a family of maps is difficult to represent with conventional data models, as they tend to introduce redundancy, increase memory usage and lead to inconsistencies (Oosterom and Schenkelaars 1995). Models that can handle multiple levels of detail without redundancy are then desirable for handling multiscale representations. While detail levels are closely related to cartographic generalization techniques, our aim here is not to advance such techniques, but rather to describe a neural network-based data model for representing objects with undetermined boundaries arising from scale changes. In doing this, we adopt a point-set paradigm and assume objects have a definable crisp (or nearly crisp) boundary at certain (fine) scale, but may become indeterminate or fuzzy at others (coarser scales).

3. Neural network-based data model

The generalities of the neural network-based data model were first introduced by Silván (2006). These generalities are restated below while emphasizing the particularities of the model used in this study. The ANN model can be seen as composed of two major building blocks (figure 2). The first building block represents a single layer of neural units. Each neural unit produces a membership function that defines a basic geometrical element of the two-dimensional plane. A set of *scaling parameters* allows controlling the degree of fuzziness of basic geometric elements. The second building block consists of a hierarchical network that combines membership values from basic elements into more complex geometrical elements. The final output represents the membership value of a complex geographical entity.

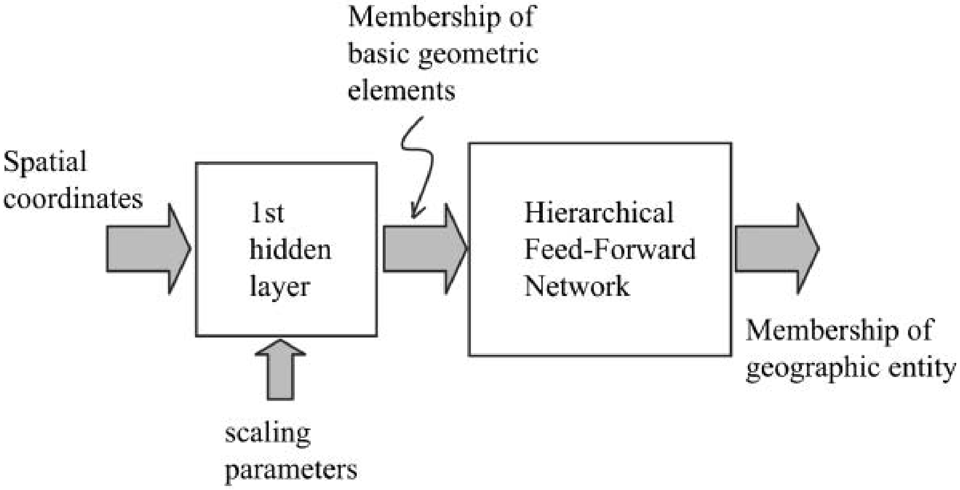

Membership of basic geometric elements
Spatial coordinates
1st hidden layer
Hierarchical Feed-Forward Network
Membership of geographic entity
scaling parameters

Figure 2. This diagram shows the building blocks of the neural network-based data model (*see* the text).

3.1 *Neuron model*

The basic constituent elements of the ANN model are computation units, termed (artificial) neurons. In its most general formulation, a neuron has some number, n, of inputs, and is capable of taking on a number of states, each described by a vector $\mathbf{w}=(w_1, w_2, \ldots, w_n)$ of n real number called *synaptic weights* and a real number b called *activation level* or bias (Looney 1997). Generally, when in the state described by $(\mathbf{w}, b)$, and on receiving input $\mathbf{x}=(x_1, x_2, \ldots, x_n)$, the computation unit produces and activation $f(\mathbf{x}; \mathbf{w}, b)$, where $f{:}\mathrm{R}^n \times \mathrm{R}^{n+1} \to \mathrm{R}$. Different definitions of $f(\mathbf{x}; \mathbf{w}, b)$ lead to essentially different neuron types. Typically, a neuron output takes the form $f(\mathbf{x}; \mathbf{w}, b)=h(g_{\mathbf{w},b}(\mathbf{x}))$, where $g_{\mathbf{w},b}{:}\mathrm{R}^n \to \mathrm{R}$ is a parameterized function termed the *weight function*, and $h{:}\mathrm{R} \to [0, 1]$ is termed the *activation function*. While the weight function determines how the inputs are integrated in the neuron, the activation function determines how the neuron responds to the excitation from the input: typically, low or high.

3.1.1 Weight functions. Depending on the form of the weight function, we shall distinguish two types of neural units. The weight functions used here can be either *dot product* and *Euclidian distance*. For reasons that will become apparent later, the neurons that have the dot-product weight function are referred to as *linear units*, and the neurons that have the Euclidian-distance weight function are referred to as *circular units*. Thus, linear units are neurons that perform linear combinations of the inputs. The output takes the particularly simple form $h(\mathbf{x} \cdot \mathbf{w}-b)$, where $\cdot$ denotes the dot product of vectors, i.e. $\mathbf{x}\cdot\mathbf{w}=w_1x_1+w_2x_2+, \ldots, +w_nx_n$. Likewise, circular units perform quadratic combinations of the inputs. The output of a circular unit takes the form $h(\|\mathbf{x}-\mathbf{w}\|-b)$, where $\|\mathbf{x}\|$ denotes the Euclidian norm of $\mathbf{x}$, i.e. $\|\mathbf{x}-\mathbf{w}\|^2=(x_1-w_1)^2+(x_2-w_2)^2+, \ldots, +(x_2-w_2)^2$. Figure 3 shows the graphic representation of linear and circular units used in this paper.

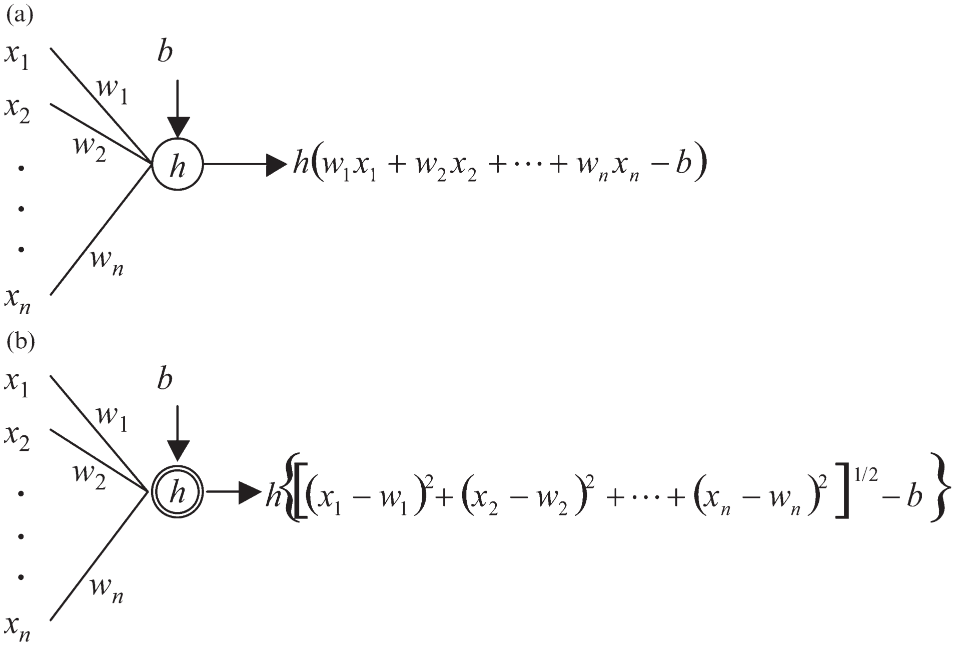

(a)
x_1
x_2
x_n
w_1
w_2
w_n
b
h
$h(w_1x_1 + w_2x_2 + \cdots + w_nx_n - b)$
(b)
x_1
x_2
x_n
w_1
w_2
w_n
b
h
$h\{[(x_1 - w_1)^2 + (x_2 - w_2)^2 + \cdots + (x_n - w_n)^2]^{1/2} - b\}$

Figure 3. Graphic representation of (a) a neural unit with dot-product weight function (linear unit) and (b) a neural unit with Euclidian-distance weight function (circular unit).

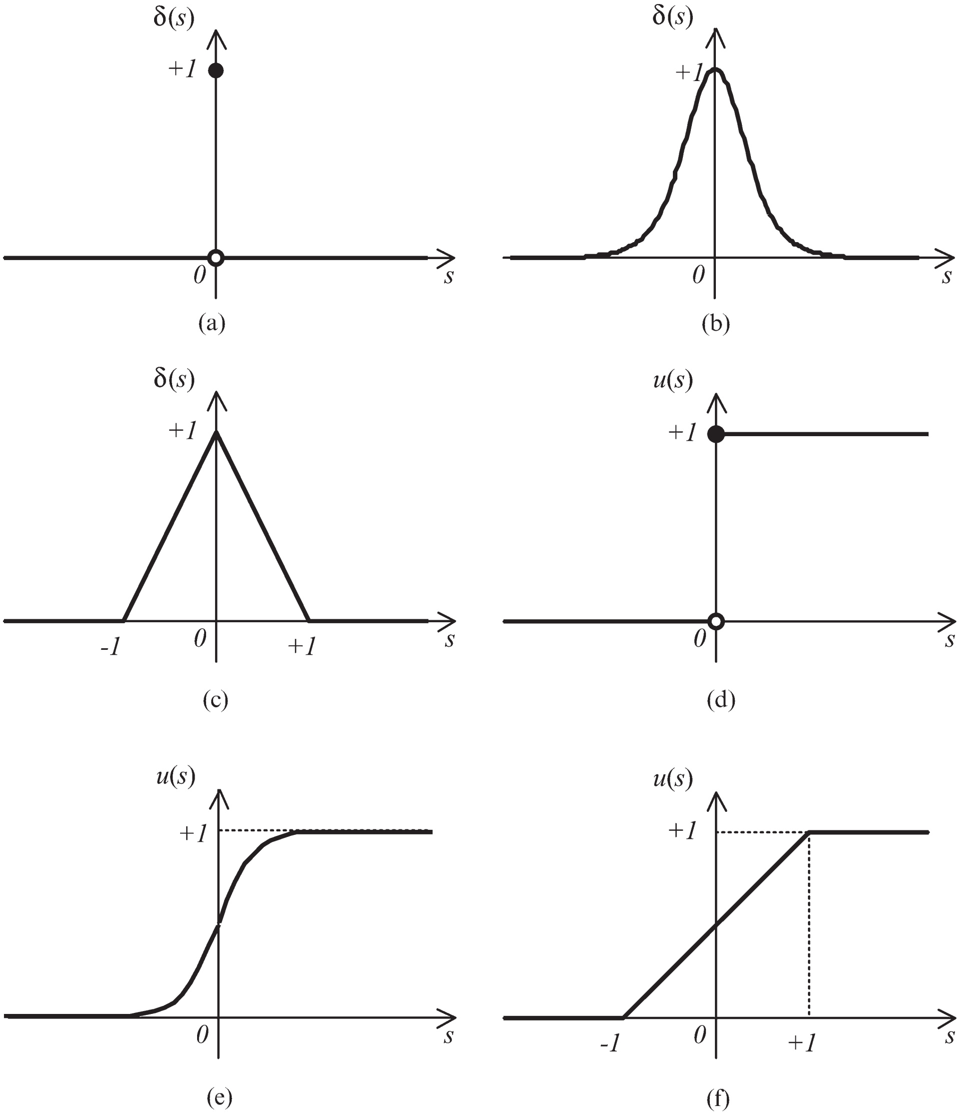


Figure 4. Examples of activation function: (a) unitary impulse, (b) bell-shaped, (c) triangle-shaped, (d) sign, (e) log-sigmoid and (f) ramp. Examples (a), (b) and (c) are of point-spread type, whereas examples (d), (e) and (f) are of cumulative type.

3.1.2 Activation functions. The activation functions used here include the radial basis functions (point-spread type) and sigmoid functions (cumulative type). Figure 4 shows examples of point-spread and cumulative activation functions. In the standard neural network modeling, cumulative functions are typically combined with the dot-product weight function (e.g. as in back-propagation networks, Rumelhart *et al.* 1986, Looney 1997), whereas point-spread activation functions are typically combined with the Euclidian-distance weight functions (e.g. as in radial basis networks, Looney 1997). Here we will not restrict ourselves to these combinations, nor to the standard definitions for point-spread and cumulative functions.

In the most general sense, a *point-spread activation function*, δ: R→[0, 1], is defined as a symmetric and strictly decreasing function in |s|, which fulfils the conditions of equation (1).

$$\lim_{|s|\to\infty}\delta(s)=0,\quad \lim_{s\to 0}\delta(s)=1 \tag{1}$$

On the other hand, a *cumulative activation function*, u: R→[0, 1], is defined by the cumulative integral in equation (2), where w is a normalized point-spread function, so that u fulfils the conditions of equation (3).

$$u(s)=\int_{-\infty}^{s} w(t)\mathrm{d}t, \tag{2}$$

$$\lim_{s\to-\infty}u(s)=0,\quad \lim_{s\to\infty}u(s)=1 \tag{3}$$

We shall distinguish between hard and soft activation functions. An activation function, either cumulative or point-spread, is said to be *hard* if it is bi-valued and piece-wise continuous, and it is said to be *soft* if it is either continuous, or multi-valued and piece-wise continuous. From these definitions, it follows that there is a unique hard point-spread function: the so-called *unitary impulse* defined in equation (4) (*see* figure 4(a)).

$$\delta_0(s)=\begin{cases}1, & \text{if } s=0\\ 0, & \text{if } s\neq 0\end{cases} \tag{4}$$

Examples of soft point-spread function include the standard radial basis function (figure 4(b)) defined as $\delta_1(s)=\exp(-s^2)$, and the triangle-shaped function (figure 4(c)) defined in equation (5).

$$\delta_2(s)=\begin{cases}1-|s|, & \text{if} -1\leq s\leq 1\\ 0, & \text{if} -1>s>1\end{cases} \tag{5}$$

The typical example of hard cumulative activation function is the binary *sign* function, as defined in equation (6) (*see* figure 4(d)).

$$u_0(s)=\begin{cases}1, & \text{if } s\geq 0\\ 0, & \text{if } s<0\end{cases} \tag{6}$$

This function (or any of its variants) is commonly used for several recognition and classification tasks (Looney 1997), and various soft versions exist. The standard unipolar sigmoid is perhaps the most widely used soft version of cumulative activation function (figure 4(e)). This is defined in equation (7).

$$u_1(s)=\frac{1}{1+e^{-s}} \tag{7}$$

Other examples of cumulative functions include the error function, which is introduced in statistics and probability theory, and the ramp function of figure 4(f). These functions are mathematically defined in equations (8) and (9), respectively.

$$u_2(s)=0.5+0.5\mathrm{erf}(s) \tag{8}$$

$$u_3(s) = \begin{cases} 1, & \text{if } s > 1 \\ 0.5 + 0.5s, & \text{if } -1 \le s \le 1 \\ 0, & \text{if } s < -1 \end{cases} \tag{9}$$

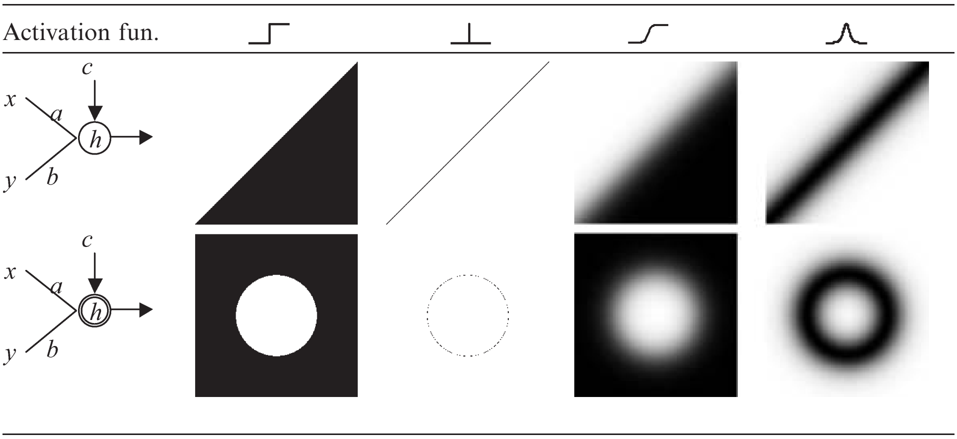
The above examples are meant for illustration purposes only and should not be taken as complete or meant to cover the whole spectrum of types of activation functions one may use to characterize different types of fuzzy geographic objects.

3.2 *First hidden layer: basic geometric elements*

When the inputs of a neuron correspond to points in the plane, its output may be regarded as a membership function defining a spatial entity (Silván 2005, 2006). Several spatial entities can be generated depending on the neuron type and activation function used. The set of all combinations of neuron types and activation functions defines the processing core of the proposed model. More complex shapes are built in terms of these basic elements. The membership values, as produced by single neural units, are displayed as gray level images in table 1. These neural units are the constituent elements of the first layer of the model (figure 2). The kernel so-defined can produce the three basic geometric elements: point, line and region; both fuzzy and crisp. Points are produced as instances of circular neurons with activation level set to 0. Basic lines are created with point-spread type activation functions and can be either linear or circular. Basic regions are produced by cumulative activation functions and can be either infinite-infinite half-spaces produced by linear neurons, or finite-infinite half-spaces produced by circular neurons.

3.2.1 Geometrical interpretation of neural units. Let (x, y) denote any point in the plane and consider a linear neuron with weights (a, b) and bias c. In this case, the weight function becomes $s = ax + by - c$. The equation $ax + by = c$ defines a straight line on the plane. For any given point (x, y) (not necessarily lying on the line), s gives

Table 1. Membership functions simulated from single units. The darker the tone the closer to 1 the membership value is, and the lighter the tone the closer to 0 the membership value is. The shape of the activation function is indicated at the top of each column, whereas the unit denoted by a single circle uses a dot-product weight function, and the unit denoted with a double circle uses an Euclidian-distance weight function.



the oriented distance from the input point to the line, provided that $a^2+b^2=1$. The orientation of the line is such that $s>0$ for a point laying in the right half-plane, and $s<0$ for a point laying in the left half-plane. If the line is horizontal, the half-plane above the line is taken as the right-hand side. Similarly, for a circular neuron with weights (a, b) and bias c, the weight function $s=[(x-a)^2+(y-b)^2]^{1/2}-c$ gives the oriented distance from the input point to the circumference defined by the equation $(x-a)^2+(y-b)^2=c^2$. In this case, if $s>0$, the point lies outside the circumference, and if $s<0$, it lies inside the circumference.

It is worth noting that each activation function at the first layer corresponds to a distance function, provided that its argument, the weight function, defines the oriented distance from the input point to the feature. As we shall see below, this characteristic is useful for linking the concepts of fuzziness, level of detail, and scale.

3.2.2 Scaling parameters. In the proposed model, the original soft cumulative (u) and point-spread (δ) functions are replaced by their scaled versions $u_\lambda(s)=u(s/\lambda)$ and $\delta_\lambda\ (s)=\delta(s/\lambda)$, respectively. The scaling parameter λ controls the smoothness of the transition between 0 and 1, while preserving the shape of the activation function: the greater the value of the scaling parameter the smoother the activation is. Furthermore, in a multiscale framework, each scaling parameter needs to be specified as a function of a common scale index, say t. This functional variation can be denoted by $\lambda_i(t)$ for the i-th unit in the first hidden layer. A feature is said to be uniformly scaled if all its neural units at the first layer share a unique functional variation of the scaling parameter; otherwise, it is non-uniformly scaled.

Since each single unit in the first layer can carry its own scaling parameter, it is convenient to consistently normalize the activation functions, so that all the units share a unique scale reference, even if they have different activations. Here we used the ramp (figure 4(f)) and triangle-shaped (figure 4(c)) activation functions as reference for the normalization of any given cumulative and point-spread function, respectively. Each activation function can be normalized so that it reaches the same value as the reference function at a predefined cut distance $s_{cut}<1$. For example, for $s_{cut}=0.75$ the normalized log-sigmoid function can be written as $u_\lambda\ (s)=0.5+0.5\ \tanh(s/0.77\lambda)$, so that it equals $0.5+0.5s$ at $s=s_{cut}$. Once normalized, the activation function can be scaled by λ. The family of scaled functions contains, by matter of definition, a hard activation function that results in the limit when the scaling factor tends to zero. This is expressed in equation (10).

$$u_0(s)=\lim_{\lambda\to 0} u_\lambda(s), \quad \delta_0(s)=\lim_{\lambda\to 0}\delta_\lambda(s) \tag{10}$$

Strictly speaking, the above definition for the hard cumulative function is slightly different from that in equation (6). Specifically, the limiting process leads to a tri-valued sign function, as given by equation (11).

$$u_0(s)=\begin{cases} 1, & \text{if } s>0 \\ 0.5, & \text{if } s=0 \\ 0, & \text{if } s<0 \end{cases} \tag{11}$$

However, we still consider this function of hard type by extension of the original definition. The tri-valued function so-defined is useful when considering regions as partitions of the space in three disjoint parts: one part inside the object, another on the border of the object, and the remaining part outside the object.

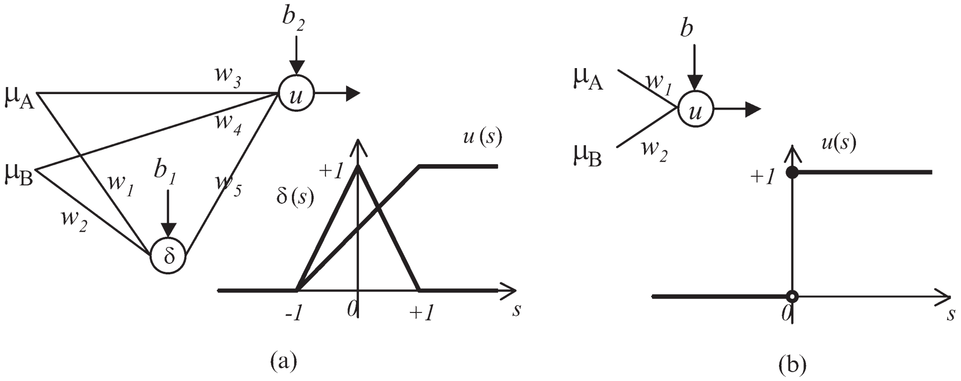


Figure 5. Neural network implementation of (a) fuzzy and (b) Boolean binary operators. Appropriate parameter values are given in table 2, respectively.

3.3 *Hierarchical feed-forward network: basic operations*

In order for the model to be able to build upon basic geometric elements, a set of basic operations need to be defined on the membership space. For this matter, neural networks offer a vast number of possibilities. One obvious approach consists in the implementation of traditional fuzzy algebra operators through neural units. Figure 5(a) shows the diagram of a network capable of implementing any of the basic fuzzy set operations (i.e. union, intersection, and difference), where the inputs μ_A and μ_B denote membership values for two given sets *A* and *B*, respectively. Table 2 gives appropriate weight/bias parameters for each fuzzy set operation. Likewise, figure 5(b) represents the neural implementation of Boolean operations, and table 2 also gives the corresponding weight/bias parameters for each Boolean operation. Notably, ANNs offer a flexible way for implementing traditional operations, even to the point where one may question the uniqueness of parameters and activation functions used. Indeed, one may build slightly different fuzzy operations by relaxing the activation functions in these implementations. A more elaborate approach was previously proposed (Silván 2006) in which a single unit can perform more complex *n*-ary Boolean functions, which can lead to minimal-sized neural networks. Furthermore, the encoding of operations is spread across several weight parameters in a way that makes it difficult, if not impossible, to interpret the representation.

Here we use the implementations of Boolean operators of figure 5(b), but with the ramp activation function (figure 4(f)). This implementation allows for an easy generalization of binary operations to *n*-ary operations (with *n* greater than 2). The general rule for selecting the weight and bias parameters for the *n*-ary AND (intersection) and OR (union) operations is as follows: let *q* and *p* denote the number

Table 2. Weight and bias parameters for the network implementation of fuzzy set operators, as shown in figure 5(a), and Boolean operations, as shown in figure 5(b).

Fuzzy Op.	w_1	w_2	b_1	w_3	w_4	w_5	b_2	Boolean Op.	w_1	w_2	b
$\min(\mu_A, \mu_B)$	1	-1	0	1	1	1	2	$\mu_A \wedge \mu_B$	2	2	3
$\max(\mu_A, \mu_B)$	1	-1	0	1	1	-1	0	$\mu_A \vee \mu_B$	2	2	1
$\min(\mu_A, 1-\mu_B)$	1	1	1	1	-1	1	1	$\mu_A \wedge \neg \mu_B$	2	-2	1

of complimented and non-complimented variables, then the bias parameter is set to $2p-1$ for an AND operator and to $1-2q$ for an OR operator, whereas the weight values are set to -2 and 2 for complimented and non-complimented variables, respectively, for both AND and OR operators.

4. Representation of basic geometric shapes

4.1 *Fuzzy points and fuzzy lines*

Points and lines are non-natural entities of the two-dimensional space, as they can be regarded as degenerated regions. In many cases, their use for representing geographic concepts is tied to scale issues. For example, cities and rivers that are represented by polygons at small geographic scales can be represented by points and lines, respectively, at larger scales. The scales at which regions should be changed into points or lines certainly depend on the purpose and the implementation constraints.

To maintain consistency of the limiting process of fuzzy entities approaching the crisp entities, we may define fuzzy points and fuzzy lines as any relaxed representation coming from a consistent definition of crisp points and lines, respectively. For example, a crisp point can be defined as 'a circle with radius zero', or as 'the intersection of two non-parallel nor coincident lines' (Silván 2006). Figure 6 shows the neural network implementation when using these two definitions.

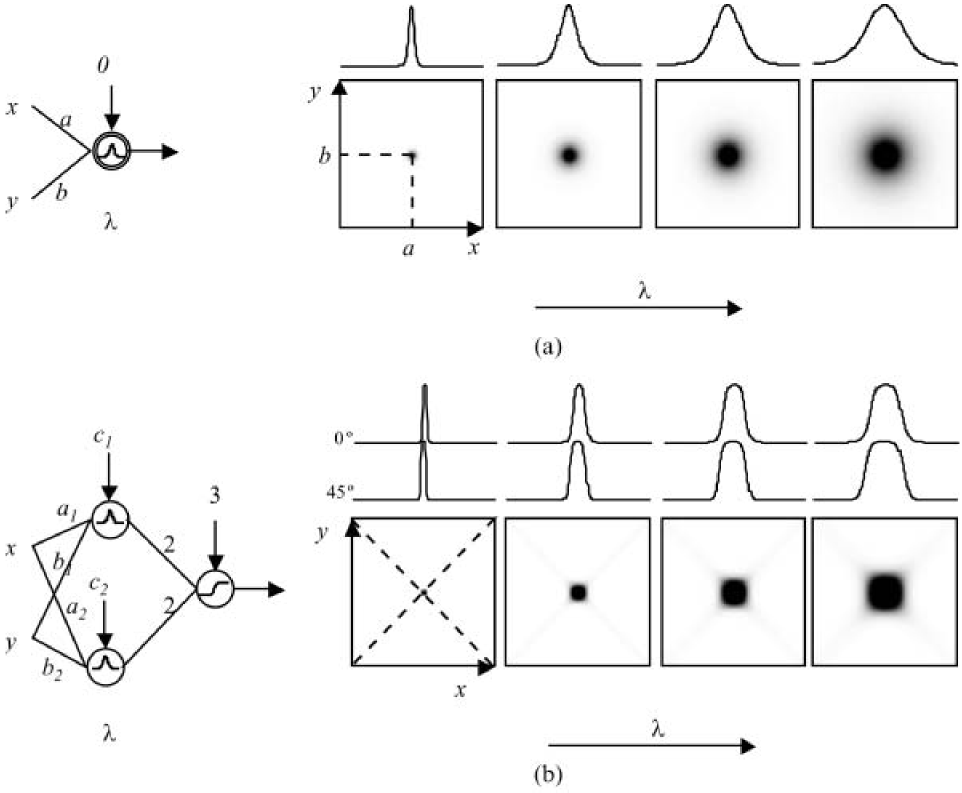


Figure 6. Neural network representation of fuzzy points. In (a), the point is defined as a circumference with radius zero, whereas in (b) the point is defined as the intersection of two straight lines.

The first definition leads to a single circular unit with point-spread activation. This representation provides a general template for membership functions of fuzzy points, of which instances can be found in the data modelling literature (Leung 1997, Schneider 1999). This is the simplest way to define fuzzy points, yet the most restrictive when it comes to representing varying uncertainties along various orientations (anisotropy). The second definition is more amenable for modeling anisotropic uncertainty. Evidently, there is an infinite number of ways for defining a point (e.g. any number of lines meeting at the same point defines a point). Studying all the possible combinations is beyond the scope of this paper. We would rather introduce only the most straightforward definitions.

Likewise, a line segment in the plane can be seen as ‘the intersection between a straight line containing the segment and a circular region centered at the mid point of the segment’. This definition for linear segments can be readily represented with the neural model. The representation has the property that, when the scaling value tends to zero, the membership function takes the value of 0.5 at boundary points, 1 for points interior to the segment, and 0 elsewhere. Moreover, it allows defining fuzzy polygonal lines through the union operation of consecutive segments. Figure 7 shows the general structure of the network for a polygonal line (polyline) with *n* segments. The network parameters a_k, …, f_k, for $k=1$, …, *n*, are determined by the vertex points (x_k, y_k) and (x_{k+1}, y_{k+1}). It should be noted that directed segments cannot be represented with this approach. A more elaborate definition for polylines could be built using a similar approach to that used for polygons below. The construction and use of such data structures are beyond the scope of the present research. We rather turn our attention to the representation of fuzzy polygons. In fact, the network structure for fuzzy polygons might be used to build more complex

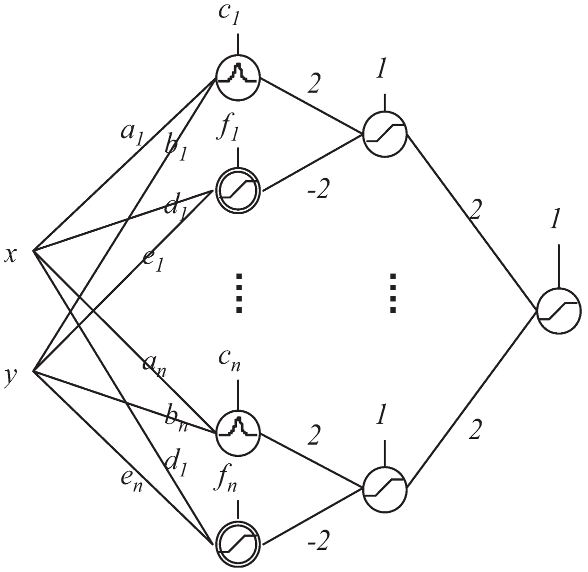


Figure 7. Neural network representation of fuzzy polygonal lines. The parameters (a_i, b_i, c_i), for $i=1$, …, *n*, define the straight lines containing the segments, whereas (d_i, e_i, f_i), for $i=1$, …, *n*, define circumferences centered at the mid point of each segment. The second layer defines the membership function for each segment, which are then combined into a polyline at the output unit.

fuzzy points and lines as degenerated regions, an approach that has shown to be more fruitful (Cohn and Gotts 1996).

4.2 *Fuzzy polygons*

The neural network representation of fuzzy regions is based on the exact model for crisp polygons. Each bounding edge of a polygon defines the boundary of two half-planes. Thus, the first hidden layer of the network consists of a set of linear threshold units, each of which computes the half-planes defined by an edge of the polygon. The intersection and union of such half-planes, performed in the correct order, defines the interior of the polygon. The sequence of such operations can be specified as a Boolean formula on the half-plane that supports (contains) the polygon. This representation is termed the constructive solid geometry (CSG, Dovkin *et al.* 1988, Walker and Snoeyink 1999), and can be further expressed as a direct n-ary tree, where the nodes represent Boolean operators. Figure 8(a) shows an

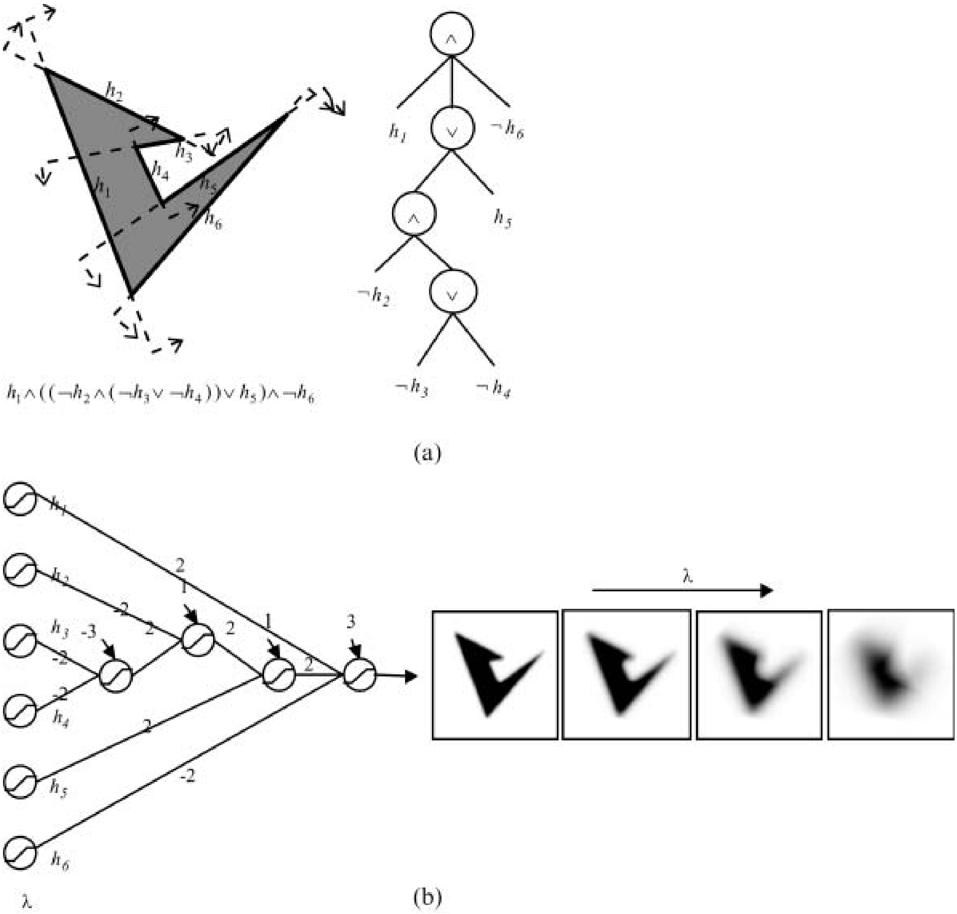


Figure 8. (a) The constructive solid geometry (CSG) representation of a crisp polygon. The arrows on the polygon (left top) indicate the right half-planes, i.e. the region where literals take the value 1 (true). The Boolean equation (bottom) defines the interior of the polygon and is also expressed as a n-ary tree (top right). (b) The neural network representation (left), where the input to the first layer has been obviated. Several simulated outputs for increasing scaling parameter values are also displayed (right). The simulation was performed on a 100-by-100 grid of points.

example of the CSG for a simple polygon, where half-planes to the right-hand side of each polygon edge is indicated with arrows. The translation from the CSG into the network representation is straightforward. In the neural network representation, the Boolean operations are simply implemented using neural units (figure 5(b)), but with soft activation functions. Figure 8(b) illustrates the network representation of a simple fuzzy polygon, and its simulation for various degrees of fuzziness.

An efficient algorithm for constructing the CSG representation for simple polygons has been provided by Dovkin *et al.* (1988). The algorithm builds the Boolean formula in time $n \log_2(n)$ for a polygon with n vertices. A variant of Dovkin *et al.*'s method was implemented to build the network structure with similar time complexity. The algorithm relies on the efficient computation of the convex hull of polygonal chains. The procedure first determines the output neuron and then follows a recursive approach, progressively adding more neurons until the first hidden layer is reached. As an example, consider the polygon of figure 8(a). Table 3 shows the gradual construction of the network using a coded structure. In the coded structure, each pair of square brackets enclose an unprocessed chain, and each pair of curly brackets represents a neural unit implementing a n-ary operation of the form $\{operator, operand_1, operand_2, \ldots, operand_n\}$. In the initial step ($i$=0), the convex hull of the entire polygon is computed and used to decompose the polygon in chains, each of which has extreme vertices corresponding to consecutive vertices of the convex hull, e.g. the polygon represented by $[h_1 \neg h_2 \neg h_3 \neg h_4 h_5 \neg h_6]$ is decomposed in the three chains represented by $[h_1]$, $[\neg h_2 \neg h_3 \neg h_4 h_5]$ and $[\neg h_6]$. Each chain splits the plane in two half-planes when extending the extreme edges toward infinity. The interior of the polygon is defined by the intersection of such half-planes. Then, the output neuron corresponds to an **intersection** operation ($\wedge$) on the half-spaces defined by the chains. The networks for each chain are built recursively. In the next recursion level (i=1), each chain composed of more than one edge is decomposed in sub-chains, e.g. the chain $[\neg h_2 \neg h_3 \neg h_4 h_5]$ results in the sub-chains $[\neg h_2 \neg h_3 \neg h_4]$ and $[h_5]$. This time, the network for the original chain has an output neuron that performs the **union** ($\vee$) of the half-spaces from the sub-chains. As the process continues the operation alternates between union and intersection, until only chains of single edges are left. For single-edge chains, a linear unit is used to represent the half-spaces defined by the edge. This linear unit lies in the first hidden layer.

Once the network has been built, the activation functions and smoothing parameters for the first hidden layer are incorporated in the network data structure. In practice, the selection of the activation function and scaling parameters will mainly depend on the purpose and knowledge about the object being represented. Functions like those of equations (7) and (8) can be used for continuous smooth

Table 3. Progressive construction of the network structure for the polygon of figure 8(a). Each pair of square brackets represents an unprocessed chain, and each pair of curly brackets represents a neural unit implementing a n-ary operation of the form $\{operator, operand_1, operand_2, \ldots, operand_n\}$ (*see* the text).

i	Coded network structure
0	$[h_1 \neg h_2 \neg h_3 \neg h_4 h_5 \neg h_6]$
1	$\{\wedge, h_1, [\neg h_2 \neg h_3 \neg h_4 h_5], \neg h_6\}$
2	$\{\wedge, h_1, \{\vee, [\neg h_2 \neg h_3 \neg h_4], h_5\}, \neg h_6\}$
3	$\{\wedge, h_1, \{\vee, \{\wedge, \neg h_2, [\neg h_3 \neg h_4]\}, h_5\}, \neg h_6\}$
4	$\{\wedge, h_1, \{\vee, \{\wedge, \neg h_2, \{\vee, \neg h_3, \neg h_4\}\}, h_5\}, \neg h_6\}$

fuzzy regions, while functions like that of equation (9) can be used for core-boundary fuzzy regions (*see* Schneider 1999, for a thorough discussion on types of fuzzy regions).

4.2.1 Complex polygons. Complex polygons can be built as overlay operations on simple polygons. Figure 9 illustrates the construction of the network for a multi-part polygon with holes and islands. The hierarchical structure of the network reflects the nested form of the Boolean operations required for defining the interior of the complex region. Notice that it is possible to infer some relationships among the constituents parts, provided that the region is well defined (i.e. it is non-empty and perfectly bounded, and all the constituent parts are essential for the definition of the region). While the connectivity indicates the precedence of operations, the relation weights-bias determines the type of operation performed among the parts (i.e. if the bias equals the sum of positive weights minus one, then the neurons perform a union of the input, or if the bias equals the sum of negative weights plus one, then the neurons perform an intersection of the input). The sign of the weight encodes the orientation of the input polygons (e.g. a negative weight indicates the input polygon represents a hole). Therefore, relationships among constituents parts can be retrieved by inverting some encoding rules from the network structure. Formalizing such a decoding process falls beyond the scope of the present work.

5. Results

5.1 *Scaling up and spatial filtering*

The effect of the scaling parameter on the geometry of crisp regions was investigated using fractal polygons. Several studies have demonstrated the fractal nature of

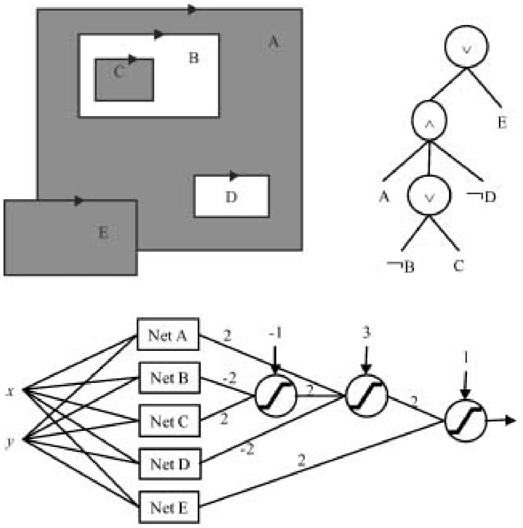


Figure 9. Network representation of complex polygons. Polygons A to E (top-left) are clock-wise oriented so that interior points are to the righthand side when walking around the edge in the direction indicated. The tree structure (top-right) represents the Boolean formula that defines the interior of the region (shaded area). The network representation (bottom) takes the network representation for each polygon and links them through a network that implements the Boolean formula.

various geographic phenomena. For example, lakes and islands distribution in Finland has been found to exhibit a fractal structure (Sarjakoski 1996). Figure 10 shows images of the network simulations for the snowflake and arrowhead fractals, respectively, for increasing values of the scaling parameter. The construction of these fractal polygons can be seen as the union (snowflake fractal) or difference (arrowhead fractal) of scaled triangles with constant scaling factors. Because we wanted to focus on the geometry rather than on the fuzziness, only the first hidden layer used a soft activation function (figure 4(e)), whereas the subsequent layers used a binary hard activation function (figure 4(d)). This setting is equivalent to representing the 0.5-cut of the membership function (i.e. the set of points that have membership values greater or equal to 0.5, Zhan 1998). A uniform scaling was adopted by setting all the scaling parameters to the same value (λ). The simulations reveal that only those details that are in the order of, or larger than, the scaling parameter value can be apparent. Not surprisingly, the maximum achievable level of detail encoded in the network is always reached at $\lambda=0$.

One potential use of the above behavior could be in querying scale-dependent properties of geographical features. For instance, consider the question of Sarjakoski (1996): how many lakes are there in Finland? Figure 10(b) resembles the way lakes (white) versus land (black) looks at various scales. As noted by Sarjakoski (1996) and many others (*see* also, Goodchild 2001), the above question does not have a unique answer, but an answer that depends on the observation/representation scale. Assuming a proper construction of the network structure, the answer to the above question is found by simulating the network on a grid for a small number of meaningful scaling parameters and then counting the number of lakes for each case. Since the network representation is based on the accurate delineation of the feature at the finest detail level, the problem shifts to determining appropriate scaling parameters. Choosing the appropriate scaling parameters might

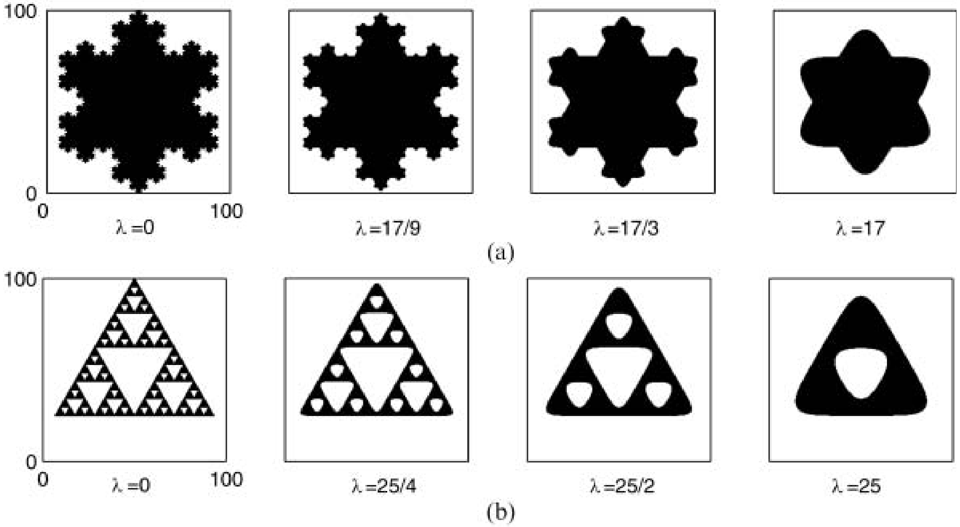


Figure 10. Spatial filtering effect of the scaling parameter (λ) from the neural representation of (a) Koch's snowflake fractal and (b) Sierpinski's arrowhead fractal. The point spacing of the simulation grid is the same in all cases. It was set to a small value with respect to the shortest polygon edge to avoid appearance of spurious details. The output was forced to be crisp by using hard activation functions (*see* the text).

be challenging in practice. It involves a profound knowledge of the mechanisms by which details are filtered out from (or incorporated onto) the apparent shape due to scaling operations, which amounts to determining the scales at which observation processes or abstractions mechanisms produce the patterns being represented. For instance, while an observation device (e.g. remote sensor) can lead to a uniform filtering of detail of observed lake boundaries (uniform scaling), the lake definition itself might not be uniform across the space and scales of observation, as many factors, besides its apparent size, need to be accounted for in its definition (Sarjakoski 1996). In this case, a non-uniform scaling of the activation functions is necessary, making the modeling far more complex.

The spatial filtering effect was also tested on various real datasets. An interactive zooming tool was developed to render multi-resolution maps using the neural network representation. Some examples are shown in figure 11. Figure 11(a)

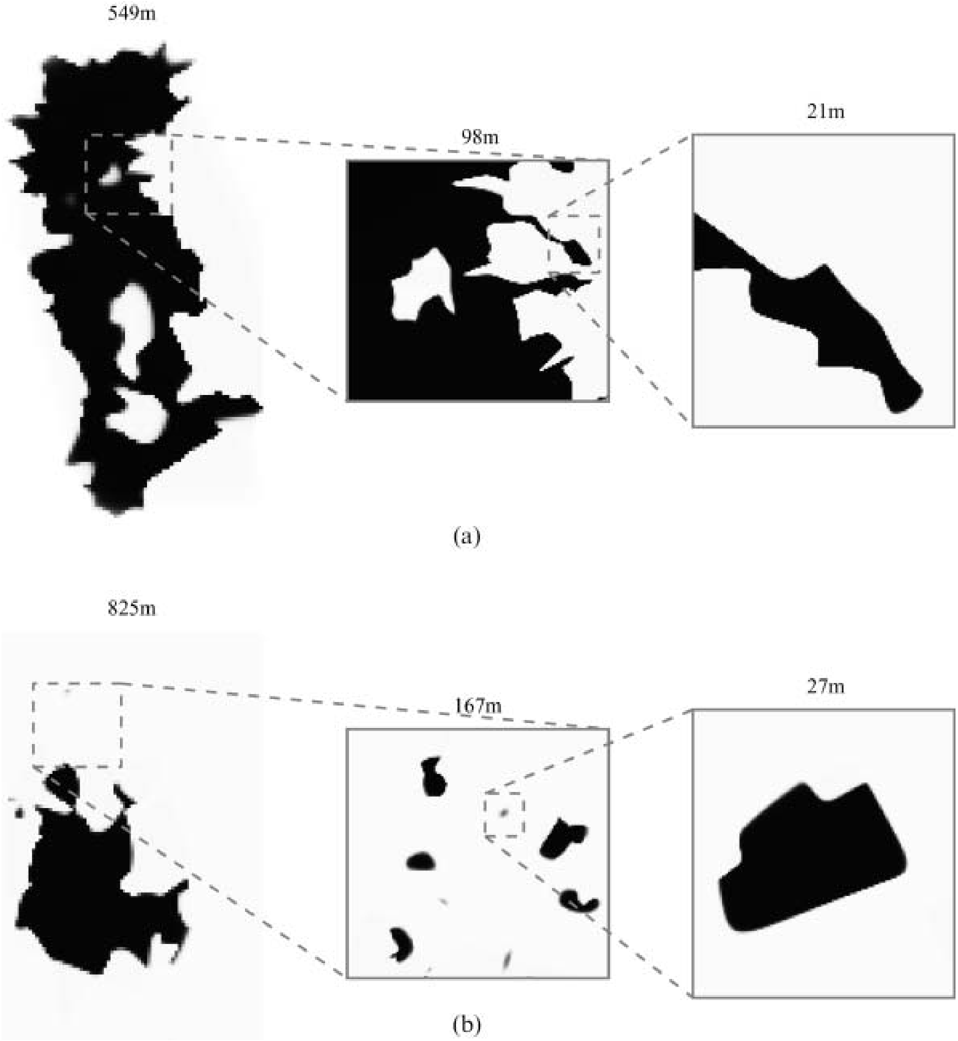


Figure 11. Spatial filtering effect of the scaling parameter for (a) an urban feature and (b) a forest feature. All the scaling parameters were set to the point spacing of the simulation grid.

represents the metropolitan area of Mexico City. This area includes satellite cities located mostly north of Mexico City. Figure 11(b) represents a temperate forest stand located just two hours south east from Mexico City. Because increasing the resolution beyond the critical resolution, as given by the scaling parameter, does not provide more details of the membership function, the cell size of each simulation grid was determined from the scaling parameter itself—whereas the scaling parameter is automatically computed by dividing the selected area by a fixed number of cells. To this end, it is important to stress the fact that the network representation, like a vector representation, is free of resolution and extent. Yet the selection of a cell size and domain is required for displaying purposes and, more generally, for querying the data model through simulation of the neural network, in which case the scaling parameter can be used for selecting an optimal cell size.

5.2 *Representation of multi-resolution data*

An example that demonstrates the representation of multi-resolution raster data used the data from figure 1. Specifically, the lagoon map at the highest resolution (1 m) was vectorized and converted to its neural network representation. The network representation was then used to simulate the maps at coarser resolutions by letting the scaling parameter of the first hidden layer equal the pixel resolution. Figure 12 shows the simulations and the absolute error images. Although the root

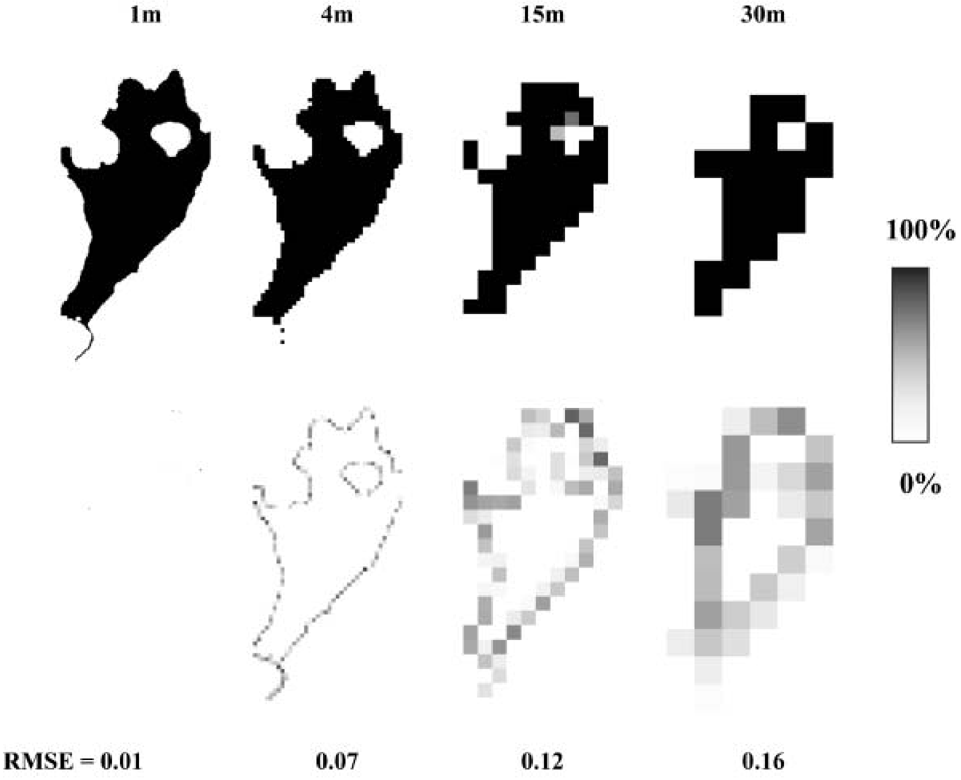


Figure 12. Model-based simulations of sub-pixel fraction maps (top) and absolute error images (bottom) from example of figure 1. The network construction was based on the finest-detail map, and then all the scaling parameters were set to the point spacing of the simulation grid.

mean square error (RMSE) is reasonably low, it tends to increase as the scaling parameter increases. A closer look at this effect revealed that the distortions start to appear when the transition region width (specified by twice the value of the scaling parameter) goes beyond the length of the segment. It should be noted that network parameters were not optimized to fit the multi-resolution data, yet the approximations are fairly good. An iterative optimization method, such as the well known back-propagation algorithm (Rumelhart *et al.* 1986), could be further developed for a better estimation of the scaling parameters.

5.3 *Storage requirements*

There is an apparent overhead introduced by the neural network with respect to vector representations, which results in an increase in storage requirements. Memory requirements by the network model can be roughly estimated in terms of the number of neural units. A simple polygon with n vertices requires one unit per edge plus one unit per operator. The worst-case scenario, which occurs when there are only binary operators (with exception of the output unit, which must have at least three operands), yields a number of $2n-2$ units. On the other hand, the best-case scenario, which occurs when the polygon is convex, yields a number of $n+1$ units. A conservative estimate of the expected number of neural units would be the mid point between these two extremes, i.e. $3n/2$ units. Table 4 compares the characteristics of the vector and neural network representations for each feature used above. The conservative estimates of neural units for the urban, forest and lagoon features are 11,892, 3,165, 1,275, respectively, which are very close to the actual values (table 4). In general, the storage requirement for a network structure is much higher than that of its vector counterpart, yet it grows linearly with the number of vertices. This apparent overhead is worthy if we consider the ability of the network to reproduce the same feature at various levels of detail without introducing much redundant information. It is also worth noting that there is one fundamental difference with respect to vector data models: while a vector model encodes the boundary by carrying implicitly some semantic information in the order of vertex appearance, the network model encodes such semantic information more explicitly in a tree-like structure of constructive operations.

6. Discussion and conclusions

The uncertainty of several geographical entities is generally related to the scale of their representation and observation (Couclelis 1996, Bian 1997). A typical example is the sub-pixel mapping of geographic features through remote sensors of varied

Table 4. Comparison between vector and neural network representations of three geographic features. The number of layers of a neural network is defined as the largest number of units an input point undergoes until it is transformed onto a membership value.

	Vector			Network		
Feature	Vertices	Edges	Rings	Layers	Units	Weights
Urban	7,928	7,888	40	13	11,086	37,947
Forest	2,110	2,103	7	10	3,203	10,411
Lagoon	850	848	2	9	1,270	4,235

spatial resolutions: the coarser the spatial resolution, the more uncertain the feature boundary results. Several fuzzy models have been developed for handling the boundary phenomena (e.g. Usery 1996, Wang and Hall 1996, Schneider 1999). Nevertheless, any practical implementation of them has generally confronted the limitations from traditional data structures, which are currently implemented in most GIS: vector and raster. The issue is further aggravated when handling multiresolution or multiscale data.

In this study, a previously introduced neural network-based data model (Silván 2006) was adapted for representing geographical entities with scale-induced indeterminate boundaries. The model has been mainly developed at an intermediate level of conceptual computer model, as it is both hardware and application independent (Worboys 1995). The general structure of the model consists of a feed-forward neural network of up to eight types of neural units. Each type of neural unit is built using one of two kinds of weight functions, namely dot product or Euclidian distance, with one of two types of activation functions, namely point-spread or cumulative, which, in turn, can be hard or soft. The model builds upon traditional spatial databases (with crisp boundaries), while allowing hierarchical construction of complex objects from basic ones. A claim is made that the interconnection of several computational units allows representing the membership function of virtually any geometrical entity, provided that enough processing units are used, wherein the connectivity of the network is dictated by the geometric structure of the object being represented.

The neural representation of points and lines was illustrated, and a method for building the neural representation of simple polygons was developed. The latter is based on the CSG representation, which consists of a Boolean formula (or *n*-ary tree) on the half-spaces defined by the polygon edges. The neural network efficiently implements the Boolean formula, while allowing soft activation functions to define soft boundary transitions. A scaling parameter is used to control the smoothness of each activation function. In the conventional treatment of the neural network paradigm, the scaling parameter corresponds to the inverse norm of each weight vector at the first hidden layer. In the present model, the scaling parameter posses physical units of distance and, thus, can be linked to the inner scale and spatial uncertainty. A few examples were presented where all scaling parameters were set to a unique value. This value was linked to the level of (spatial) detail or to the spatial resolution of a simulation grid. These representations yielded boundaries of constant degree of fuzziness or detail, similar to those produced by shift-invariant linear filters. In practice, there is no apparent reason for the whole boundary of a region to be sharp or to have a constant degree of fuzziness (Schneider 1999). There are a lot of geographical application examples illustrating that the boundaries of spatial objects can be determinate at one place, but indeterminate at another place. For example, boundaries of geological, soil, and vegetation units are often sharp in some places and vague in others. The neural representation is potentially useful for representing geographical entities of this kind. Potential applications envisaged include multiple point-in-polygon tests (Leung 1997, Walker and Snoeyink 1999), as well as the evaluation of complex overlay functions that can be expressed as set operations. In principle, any set operation expressed in the form of Boolean or fuzzy logic operations can be implemented with a neural network. As a matter of fact, non-simple polygons (e.g. polygons with holes) are constructed using this principle (Silván 2005).

Overall, the proposed model encodes a geographic feature through a set of network parameters, while providing a computational structure for complex set-based operations. It shares with the vector model the idea that boundaries are the only spatial elements that need to be coded for object's spatial embedding definition (Worboys 1995). It also shares the idea with the field view of a mapping between the locational reference frame and the attribute domain (Worboys 1995), wherein the attribute is a membership value. In this sense, ANN provides a means for linking the field and object views for geographical phenomena, a direction that is recognized as an important research need (Cova and Goodchild 2002). As with all new models, the real importance of the proposed data model for many practical applications is still to be seen. Before that can happen, a number of issues need to be addressed. For example, proving equivalence of two given networks is hard, if not impossible (a limitation that is inherited from the CSG representation). This issue could be better tackled if a formal link with the fuzzy set theory is established. In addition, the decoding of relationships and the complexity analysis of scale-dependent queries from neural representations are also important topics that deserve a more formal treatment in future research.

Acknowledgements

The authors would like to thank the anonymous referees for their comments that helped to reach the final shape of this article. The first author acknowledges the financial support given by CentroGeo.

References

Ahlqvist, O., Keukelaar, J. and Oukbir, K., 2000, Rough classification and accuracy assessment. *International Journal of Geographical Information Science*, **14**(5), pp. 475–496.

Ahlqvist, O., Keukelaar, J. and Oukbir, K., 2003, Rough and fuzzy geographical data integration. *International Journal of Geographical Information Science*, **17**(3), pp. 223–234.

Anderson, J.A., 1990, Data representations in neural networks. *AI Expert*, **5**(6), pp. 30–37.

Bian, L., 1997, Multiscale nature of spatial data in scaling up environmental models. *In*: D.A. Quattrochi and M.F. Goodchild (Eds). *Scale in remote sensing and GIS* (Boca Raton: Lewis), pp. 71–85.

Blakemore, M., 1984, Generalization and error in spatial databases. *Cartographica*, **21**(2/3), pp. 131–139.

Burrough, P.A., 1996, Natural objects with indeterminate boundaries. *In*: P.A. Burrough and A.U. Frank (Eds). *Geographic objects with indeterminate boundaries* (London: Taylor & Francis), pp. 3–28.

Cheng, T., Molenaar, M. and Lin, H., 2001, Formalizing fuzzy objects from uncertain classification results. *International Journal of Geographical Information Science*, **15**, pp. 27–42.

Cheung, C.K., Shi, W. and Zhou, X., 2004, A probability-based uncertainty model for point-in-polygon analysis in GIS. *GeoInformatica*, **8**, pp. 71–98.

Cohn, A.G. and Gotts, N.M., 1996, The 'Egg-Yolk' Representation of regions with indetermined boundaries. *In*: P.A. Burrough and A.U. Frank (Eds). *Geographic objects with indeterminate boundaries* (London: Taylor & Francis), pp. 171–187.

Couclelis, H., 1992, People manipulate objects (but cultivate fields): beyond the raster-vector debate in GIS. *In*: A.U. Frank, I. Campari and U. Formentini (Eds). *Theories and methods of spatio-temporal reasoning in gegraphic space* (New York: Springer-Verlag), pp. 65–77.

Couclelis, H., 1996, Toward an operational typology of geographic entities with ill-defined boundaries. *In*: P.A. Burrough and A.U. Frank (Eds). *Geographic objects with indeterminate boundaries* (London: Taylor & Francis), pp. 71–85.

Cova, T.J. and Goodchild, M.F., 2002, Extending geographical representation to include fields of spatial objects. *International Journal of Geographical Information Science*, **90**, pp. 509–532.

Dovkin, D., Guibas, L., Hershberger, J. and Snoeyink, J., 1988, An efficient algorithm for finding the CSG representation of simple polygon. *In*: J. Dill, (Ed). *Computer Graphics (SIG-GRAPH '88 Proceedings vol. 22)*, Atlanta, GA, August 1988 (New York: ACM), pp. 24–46.

Dowla, F.U. and Rogers, L.L., 1995, *Solving problems in environmental engineering and geosciences with artificial neural networks* (Cambridge, MA: The MIT Press), p. 339.

Egenhofer, M.J., Glasgow, J., Günther, O. and Herring, J.R., 1988, Progress in computational methods for representing geographical concepts. *International Journal of Geographical Information Science*, **13**(8), pp. 775–796.

Florack, L.M.J., Ter Haar romeny, B.M., Koenderink, J.J. and Viergever, M.A., 1994, Linear scale-space. *Journal of Mathematical Imaging and Vision*, **4**, pp. 325–335.

Foody, G.M., 2004, Relating the land-cover composition of mixed pixels to artificial neural network classification output. *Photogrammetric Engineering & Remote Sensing*, **62**, pp. 491–499.

Goodchild, M.F., 2001, Models of scale and scales of modelling. *In*: N.J. Tate and P.M. Atkinson (Eds). *Modelling scale in geographical information science* (Chichester: Wiley), pp. 3–10.

Leung, M., 1997, Point-in-polygon analysis under certainty and uncertainty. *GeoInformatica*, **1**, pp. 93–114.

Looney, C.G., 1997, *Pattern recognition using neural networks* (New York: Oxford University Press), p. 458.

Mark, D.M. and Frank, A.U., 1996, Experiential and formal models of geographic space. *Environment and Planning B: Planing and Design*, **23**, pp. 3–24.

Mennis, J.L., Peuquet, D.J. and Qian, L., 2000, A conceptual framework for incorporating cognitive principle into geographical database representation. *International Journal of Geographical Information Science*, **14**(6), pp. 501–520.

Oosterom, P. and Van Schenkelaars, V., 1995, The development of an interactive muilt-scale GIS. *International Journal of Geographical Information Systems*, **9**(5), pp. 489–507.

Openshaw, S. and Openshaw, C., 1997, *Artificial intelligence in geography* (New York: Wiley), p. 329.

Peuquet, D.J., 1988, Representations of geographic space. *Annals of the Association of American Geographers*, **78**, pp. 375–394.

Rumelhart, D.E., Hinton, G.E. and Williams, R.J., 1986, Learning representations by back-propagating errors. *Nature*, **323**, pp. 533–536.

Sarjakoski, T., 1996, How many lakes, islands and rivers are there in Finland? *In*: P.A. Burrough and A.U. Frank (Eds). *Geographic objects with indeterminate boundaries* (London: Taylor & Francis), pp. 299–312.

Schneider, M., 1999, Uncertainty Management for Spatial Data in Databases: Fuzzy Spatial Data Types. *In*: R.H. Güting, D. Papadias and F.H. Lochovsky (Eds). *Proceedings of the 6th International symposium on advances in spatial databases*, LNCS 1651, July 1999, Hong Kong, China (New York: Springer-Verlag), pp. 330–351.

Schuurman, N., 1999, Critical GIS: theorizing an emerging science. *Cartographica*, **36**(4), pp. 1–108.

Shi, W., 1998, A generic statistical approach for modeling errors of geometric features in GIS. *International Journal of Geographical Information Science*, **12**, pp. 131–143.

Silván, J.L., 2005, Neural representation of polygon layers. UCGIS Summer Assembly 2005. Available online at: http://www.ucgis.org (accessed 11 January 2006).

Silván, J.L., 2006, Multiscale modeling of fuzzy spatial objects by means of neural networks. *In*: J.Y. Gong and J.X. Zhang (Eds). *Proceedings of SPIE conference Geoinformatics 2006: Geospatial Information Science*, 28 October 2006, Wuhan, China (SPIE). Vol. 6420, p.14.

Usery, E.L., 1996, A conceptual framework and fuzzy set implementation for geographic features. *In*: P.A. Burrough and A.U. Frank (Eds). *Geographic objects with indeterminate boundaries* (London: Taylor & Francis), pp. 71–85.

Walker, R.J. and Snoeyink, J., 2003, Practical point-in-polygon tests using CSG representations of polygons. Technical Report TR-99-12, University of British Columbia.

Wang, F. and Hall, G.B., 1996, Fuzzy representation of geographical boundaries in GIS. *International Journal of Geographical Information Science*, **10**, pp. 573–590.

Worboys, M.F., 1995, *GIS: a computing perspective* (London: Taylor & Francis), p. 413.

Zhan, F.B., 1998, Approximate analysis of topographical relations between geographic regions with indeterminate boundaries. *Soft Computing*, **2**(2), pp. 28–34.

Zhan, F.B. and Lin, H., 2003, Overlay of two simple polygons with undetermined boundaries. *Transaction in GIS*, **7**, pp. 76–81.