EXTRACTION OF SPATIAL OBJECTS FROM LASER-SCANNING DATA USING A CLUSTERING TECHNIQUE

Bin Jiang

Division of Geomatics, Dept. of Technology and Built Environment, University of Gävle, SE-801 76 Gävle, Sweden - bin.jiang@hig.se

Commission III, WG III/3

KEY WORDS: DEM/DTM, LIDAR, Laser-scanning, Algorithms, Classification

ABSTRACT:

This paper explores a novel approach to the extraction of spatial objects from the laser-scanning data using an unsupervised clustering technique. The technique, namely self-organizing maps (SOM), creates a set of neurons following a training process based on the input point clouds with attributes of xyz coordinates and the return intensity of laser-scanning data. The set of neurons constitutes a two dimensional planar map, with which each neuron has best match points from an input point cloud with similar properties. Because of its high capacity in data clustering, outlier detection and visualization, SOM provides a powerful technique for the extraction of spatial objects from laser-scanning data. The approach is validated by a case study applied to a point cloud captured using a terrestrial laser-scanning device.

1. INTRODUCTION

Laser-scanning has been proven to be as an effective 3D data acquisition means for extracting spatial object models such as digital terrain models and building models and it has been widely used nowadays in geospatial information industry (Ackermanm 1999). However what the laser-scanning can acquire is a digital surface model, which captures all points from treetops, buildings and ground surface depending on the circumstance. For many practical applications, we often expect spatial object models such as digital terrain models (DTM) of the bare earth surface and 3D building models. To this end, various research efforts have been made over the past years in an attempt to processing the original captured datasets for the derivation of various spatial objects. For instance, in order to get a DTM of the bare earth surface, we have to remove those non-terrain and undesired points.

This process of deriving spatial objects involves a range of operations such as filtering, interpolation, segmentation, classification, modelling and possible human interaction if no complete automatic way is reached (Tao and Hu 2001). Most filtering algorithms are targeted to the derivation of DTM, thus assumptions on the spatial distributions of points or geometric characteristics of a point relative to its neighbourhood in a terrain surface are used to construct various filtering strategies. For instance, through using TopScan, Petzold et al. (1999) used the lowest points found in a mowing window to create a rough terrain model. And it is used to filtering out those points higher than a given threshold. Then repeat the procedure several times with smaller sizes of moving window, and finally lead to a DTM. Kraus and Pfeifer (1998) used an averaging surface between terrain points and vegetation points to derive residuals of individual points, and then use the residuals to determine the weights of individuals to be selected or eliminated. Maas and Vosselman (1999) adopted approaches based on the ideas of moment invariants and the intersection of planar faces in triangulated points for extracting building models. The slope-based filter (Vosselman 2000) considers an observed fact that a large height difference between two nearby points is unlikely to be caused by a steep slope in the terrain. These algorithms together with many others as reviewed in a recent comparison study (Sithole and Vosselman 2003) are proven to be effective and efficient in the studies, noting that they sometimes require interpolation of a point cloud into regular grid format in order to carry out the post-processing. When it is too complex to distinguish different object points, additional information is needed for classification or to achieve better results (e.g. Haala and Brenner 1999, McIntosh and Krupnik 2002). However, these filtering algorithms are all based one way or another on supervised classifications with prior knowledge or assumptions about different spatial objects. The supervised classification solutions show various constraints in the sense of efficiency, e.g. sensitivity to varying point densities, limited applicability for certain kinds of spatial objects or under a certain circumstance, and difficulty in dealing with stripe etc.

The supervised classification solutions rely much on the human understanding or prior knowledge of the point geometric characteristics of spatial objects. However, it is very difficult in reality to get a true understanding, in particular when many objects are involved in a point cloud. It also depends on our specific task: e.g. to derive one single object or all objects with one point cloud. It is probably an easy task to derive one object rather than to distinguish all objects from an input point cloud. For the case of single object, we can investigate the point cloud and try to figure out the characteristics of its point distribution and further design an appropriate algorithm. Furthermore, many assumptions about the point characteristics do not always hold true, and they depend on the circumstances of laser-scanning. When come to the situation where all objects should be derived, we believe unsupervised clustering seems a more appropriate way.

One of the major reasons why unsupervised methods are so important in the post-processing is that it is very difficult to assume some characteristics of a certain object. Instead of figuring out the assumption, unsupervised methods put these characteristics aside and adopt a simple assumption, i.e. same objects should have the same similarities in terms of their xyz coordinates and intensity. This assumption seems to apply to all kinds of circumstance. In this paper, we attempt to use self-organizing maps (SOM) (Kohonen, 2001), an unsupervised clustering technique to make a classification of points with a point cloud. We adopt SOM training algorithm to group all points into different categories according to their xyz coordinates and intensity. Through the trained SOM – a two dimensional grid of neurons, the similarity of points can be interactively explored and visualized. Thus we are able to distinguish different points belonging to different objects.

As a well-developed technique, SOM has found many applications in various fields such as data classification, pattern recognition, image analysis, and exploratory data analysis (for an overview, see Oja and Kaski 1999). In the domain of GIS, Openshaw and his colleagues have used the approach in spatial data analysis to carry out the classification of census data (Openshaw 1994, Openshaw et al. 1995). It has been applied to cartographic generalization for building typification (e.g. Højholt 1995), street selection (Jiang and Harrie 2004), and line simplification (Jiang and Nakos 2003). All these studies rely on the SOM's ability in data clustering and pattern recognition. This paper will look at how it can be used for filtering laser-scanning data in deriving spatial object models from laser-scanning datasets. The remainder of this paper is structured as follows. Section 2 introduces the basic principle and algorithm of SOM. Section 3 presents a SOM-based approach for deriving different clusters within a laser scanned point cloud. Section 4 presents a case study for validation of the approach. Finally section 5 concludes the paper and points out future work.

2. SELF-ORGANIZING MAP

SOM is a well-developed neural network technique for data clustering and visualization. It can be used for projecting a large data set of a high dimension into a low dimension (usually one or two dimensions) while retaining the initial pattern of the dataset. That is, data samples that are close to each other in the input space are also close to each other on the low dimensional space. In this sense, SOM resembles a geographic map concerning the distribution of phenomena, in particular referring to first law of geography: everything is related to everything else, but near things are more related to each other (Tobler 1970). Herewith we provide a brief introduction to the SOM; readers are encouraged to refer to more complete descriptions in literature (e.g. Kohonen 2001).

2.1 Basic principle

Let's represent a d-dimensional dataset as a set of input vectors of d dimensions, i.e. $X = \{x_1, x_2, \ldots x_n\}$, where n is the size of the dataset or equally the number of input vectors. The SOM training algorithm involves essentially two processes, namely vector quantization and vector projection (Vesanto 1999). Vector quantization is to create a representative set of vectors, so called output vectors from the input vectors. Let's denote the output vectors as $M = \{m_1, m_2, \ldots m_k\}$ with the same dimension as input vectors. In general, vector quantization reduces the number of vectors, and this can be considered as a clustering process. The other process, vector projection, aims at projecting the k output vectors (in d-dimensional space) onto a regular tessellation (i.e., a SOM) of a lower dimension, where the regular tessellation consists of k neurons. In the vector projection each output vector is projected into a neuron where the projection is performed as such, "close" output vectors in d-dimensional space will be projected onto neighbouring neurons in the SOM. This will ensure that the initial pattern of the input data will be present in the neurons.

The two tasks are illustrated in figure 1, where both input and output vectors are represented as a table format with columns as dimension and rows as ID of vectors. Usually the number of input vectors is greater than that of output vectors, i.e. $n \succ k$, and the size of SOM is the same as that of output vectors without exception. In the figure, the SOM is represented by a transitional color scale, which implies that similar neurons are being together. It should be emphasized that for an intuitive explanation of the algorithm, we separate it as two tasks, which are actually combined together in SOM without being sense of one after another.

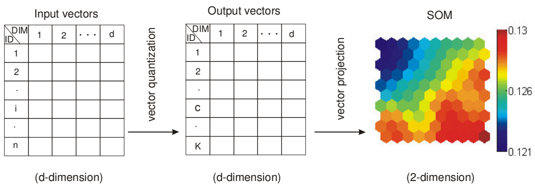


Figure 1: Illustration of SOM principle

2.2 The algorithm

The above two steps, vector quantization and vector projection, constitute the basis of the SOM algorithm. Vector quantization is performed as follows. First the output vectors are initialized randomly or linearly by some values for its variables. Then in the following training step, one sample vector x from the input vectors is randomly chosen and the distance between it and all the output vectors is calculated. The output vector that is closest to the input vector x is called the Best-Matching Unit (BMU), denoted by m_c:

$$\|x - m_c\| = \min_i \{\|x - m_i\|\}, \qquad [1]$$

where $\|.\|$ is the distance measure. Second the BMU or winning neuron and other output vectors in its neighbourhood are updated to be closer to x in the input vector space. The update rule for the output vector i is:

$$\begin{aligned} m_i(t+1) &= m_i(t) + \alpha(t) h_{ci}(t)[x(t) - m_i(t)] && \text{for } i \in N_c(t) \\ m_i(t+1) &= m_i(t) && \text{for } i \notin N_c(t) \end{aligned} \qquad [2]$$

where $x(t)$ is a sample vector randomly taken from input vectors, $m_i(t)$ is the output vector for any neuron i within the neighbourhood $Nc(t)$, and $\alpha(t)$ and $h_{ci}(t)$ are the learning rate function and neighbourhood kernel function respectively.

The algorithm can be described in a step-by-step fashion as follows.

Step 1: Define input vectors in particular their multiple variables that determine an attribute space.

The input vectors are likely to be in a table format as shown in Figure 1, where d variables determine a d-dimensional attribute space. Based on the input vectors space, an initialized SOM will be imposed for training process (c.f. step 3).

Step 2: Define the size, dimensionality, and shape of a SOM to be used.

The size is actually the number of neurons for a SOM. It can be determined arbitrarily, but one principle is that the size should be easy enough to detect the pattern or structure of SOM (Wilppu 1997). The number of neurons can be arranged in a 1- or 2-dimensional space (dimensionality). Three kinds of shape are allowed, i.e. sheet, cylinder or toroid, but usually sheet as default shape.

Step 3: Initialize output vectors *m* randomly or linearly.

At the initialisation step, each neuron is assigned randomly or linearly by some values for the *d* variables. Thus an initial SOM is imposed in the input vectors space for the following training process.

Step 4: Define the parameters that control the training process involving map lattice, neighbourhood, and training rate functions.

The number of neurons defined can be arranged in two different map lattices, namely hexagonal and rectangular lattices. However, hexagonal lattice is usually preferred because of better visual effect according to Kohonen (2001). Neighbourhood function has different formats such as 'bubbs', 'gaussian', 'cutgauss' and 'ep' (see Vesanto et al. 2000, pp. 10), but gaussian function is usually adopted and it is defined by:

$$h_{ci}(t) = e^{-d_{ci}^2 / 2\sigma_t^2} \qquad [3]$$

where σ_t is the neighbourhood radius at time t, d_{ci} is the distance between neurons *c* and *i* on the SOM grid. It should be noted that the size of the neighbourhood $N_c(t)$ reduces slowly as a function of time, i.e. it starts with fairly large neighbourhoods and ends with small ones (see figure 2).

The training rate function can be linear, exponential or inversely proportional to time t (see Vesanto et al. 2000, pp. 10). For instance, $\alpha(t) = \alpha_0 /(1+100t/T)$ is the option we adopted in the following case study, where T is the training length and α_0 is the initial learning rate. Usually the training length is divided into two periods: t_1 for the initial coarse structuring period and t_2 for the fine structuring period.

Step 5: Select one input vector x, and determine its Best-Matching Unit (BMU) or winning neuron using equation [1].

Although Euclidian distance is usually used in equation [1], it could be various other measures concerning 'nearness' and 'similarity'. Depending on the form of data measurement, other measures are allowed as long as they represent the distance between input and output vectors.

Step 6: Update the attributes of the winning neuron and all those neurons within the neighbourhood of the winning neuron, otherwise leave alone (c.f. equation [2]).

Step 7: Repeat steps 5 to 6 for a very large number of times (training length) till a convergence is reached.

The convergence is set like this, $m_i(t+1) = m_i(t), for\ t \to \infty$. In practice, the training length in epochs is determined by the size of SOM (*k*) and the size of training data (*n*), for instance for coarse period $t_1 = \frac{4 \times k}{n}$.

After the above steps, all output vectors are projected on to a 1- or 2-dimensional space, where each neuron corresponds to an output vector that is the representative of some input vectors. A 2-dimensional hexagonal map lattice grid is shown in Figure 2 where each hexagonal cell has a uniform neighbourhood.

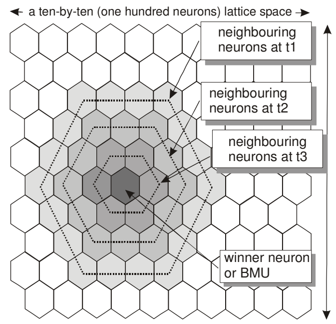


Figure 2: The characteristics of a 10x10 SOM (t1<t2<t3 with $h_{ci}(t)$ in equation 3)

3. SOM-BASED CLUSTERING ANALYSIS FOR LASER SCANNED POINT CLOUDS

3.1 Principle and overall procedure

Laser scanned point clouds are usually defined in a four dimensional space with xyz coordinates and the return intensity. For a given point cloud, all points constitute input vectors which can be used for clustering analysis in order to distinguish different points belonging to different objects. Depending on the size of input cloud, an appropriate SOM will be decided together with other parameter settings. Once all these are determined, a SOM will be derived to represent the pattern or structure of a point cloud. The SOM is organised in a grid in which nearby neurons are more similar to those which are widely separated. With the SOM, various clusters can be identified and they in essence represent different sets of points with a certain similarity in terms of their coordinates and the intensity. The process of the clustering analysis can be described as follows. For a given point cloud, all the points constitute input vectors in a four dimensional space. This space is defined by three coordinates and the return intensity. These vectors then are used to train a SOM as described in the above section. With the SOM, points with similar attributes will correspond to neurons that are grouped together. Various clusters can be identified from the SOM, and finally the derivation of various spatial object models is based on these clusters.

From a more practical perspective, the points and their corresponding attributes are used for creating input vectors in Matlab. Then training process is performed on SOM Toolbox with Matlab 6 (Vesanto et al. 2000). Although the number of output vectors (neurons) of a SOM can be arbitrarily determined, usually we choose a number that is smaller than that of the input vectors. Through the training process, each point is supposed to have a BMU from the set of neurons within the SOM. It helps to set up a linkage between a SOM and the corresponding point cloud. The specific procedure for setting up such a linkage in ArcView GIS platform is as follows (Figure 3):

- Create a polygon theme in which each polygon has a hexagonal shape, representing a neuron with output vectors as attributes in a table (SOM table)
- Create a link table (LINK table) with two fields, namely BMU and point ID
- Link the SOM table and LINK table (note fields SOM-ID and BMU are equivalent)
- Link the LINK table and NETWORK table through the common field street-ID

Through the above procedure, a linkage that is set up between a SOM and corresponding point cloud will help to select points belonging to different spatial objects.

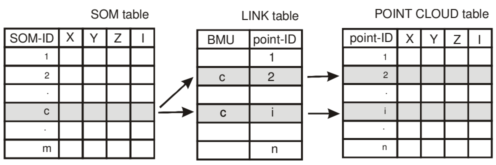


Figure 3: Linkage between a SOM and point cloud

3.2 An interactive environment for clustering and selection

Based on the above procedure, an interactive environment for clustering and selection can be built in a GIS platform. The trained SOM is imported into a GIS to setup a linkage to the point cloud that is represented as both 2D theme and 3D scene. In order to detect various clusters with the SOM, a unified distance matrix between a neuron and its neighbouring neurons (Ultsch and Siemon 1990) is computed. The distance matrix reflects the level of similarity between a neuron and its neighbouring neurons. With color scales for representing the distance matrix, we can easily detect clusters, i.e. those neurons tied closely. From the view entitled as SOM4029 in figure 4, we note that those neurons with light colors are supposed to be clusters, while those neurons with dark colors are neurons that are far from various centres of clusters. With the same figure, a cluster is selected with yellow, and the corresponding set of points is highlighted in both 2D view and 3D scene, from which we note the points are those from forest rather than from the ground.

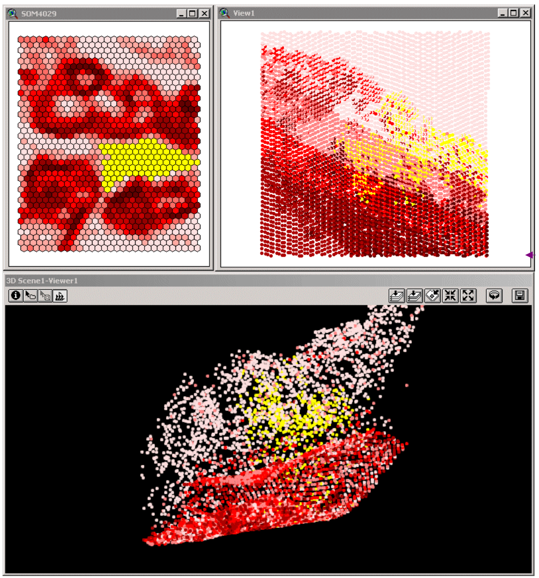


Figure 4: An interface with three connected visual components: SOM, 2D view and 3D scene of a point cloud

Formatted: Bullets and Numbering

4. A CASE STUDY

To validate the approach, we carried out a case study applied to a dataset that consists of 9072 points (figure 5). The dataset was a part of a larger dataset captured using a terrestrial laser scanner by the GIS institute at the University of Gävle. The reason why we choose the dataset is that the GIS institute has already manually filtered the dataset. Different spatial objects such as clay-road, stones, spruce and ground are extracted. Thus it provides a base to validate the model.

Using a heuristic way, we decided a SOM with the size of 40×29 to train the dataset. The process is performed in the way as follows with reference to the description in section 2. The 1160 neurons are initialised by randomly giving some values of xyz coordinates and the return intensity; and each of the neurons compares to the individual points with the point cloud to determines its best match unit using equation [1]; Then the winning neurons and its neighbourhood are adjusted their values of xyz coordinates and the intensity according to equation [2]. Details on parameter settings for the training process are shown in table 1. Once a pre-determined convergence is reached, the training process is finished with a trained SOM. The trained SOM is supposed to retain the initial structure of the point cloud. Figure 6 is the component visualizations of the SOM, and the smooth color transitions reflect the fact that similar neurons are being closer than those dissimilar.

Table 2: Parameter settings for the SOM training

Parameter	Value
Size (m)	1160
Dimensionality	2

Neighbourhood	Gaussian
Learning rate (α)	$\alpha(t) = \alpha_0 /(1+100t/T)$
Initial learning rate (α_0)	0.5 for the coarse period 0.05 for the fine period
Training length in epochs (T)	0.51 epochs for the coarse period 2.05 epochs for the fine period
Initial neighbourhood radius (σ_0)	20
Final neighbourhood radius	5 for the coarse period 1 for the fine period

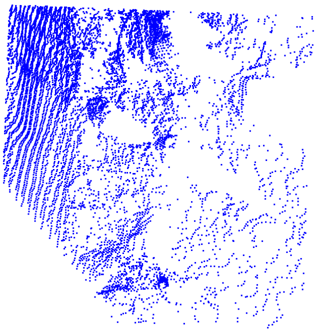
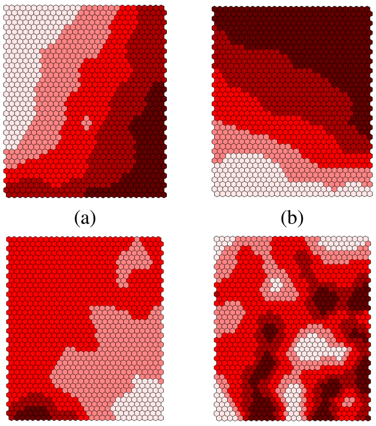


Figure 5: The original point cloud consisting of 9072 points



(a) (b)

(c) (d)

Figure 6: Component visualizations of the SOM: (a) x coordinate, (b) y coordinate, (c) z coordinate, and (d) intensity

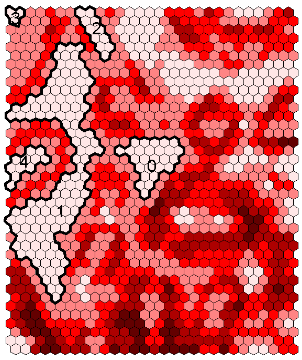
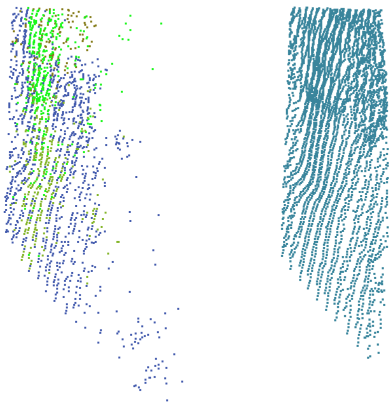


Figure 7: Five clusters detected from Umatrix of the SOM

In order to detect different spatial objects, we derive a unified distance matrix (U-matrix) between the adjacent neurons (Ultsch and Siemon 1990). Figure 7 illustrates the distance from each neuron to its neighbouring neurons. We can note those neurons that are surrounded by darker colours tend to be clusters. We tried to select those points that best match to the clusters in the SOM, and it ends up with 5 meaningful clusters as indicated in figure 7. The cluster 0 match to the stones quite well, while the rest four clusters match to clay-road. Figure 8 illustrates those points associated with clusters 1-4 (a) and points representing clay road (b). Visual inspection suggests the model is a useful tool for filtering scanning datasets. In the meantime, cautious should be taken for the model, as other spatial objects such as spruce and ground are not clearly shown with the clusters in the umatrix of the SOM. This suggests further work is needed with the training process, probably by introduction of a weight among xyz coordinates and return intensity.



(a) (b)

Figure 8: Points associated with detected clusters 1-4 (a) and points representing clay road

5. CONCLUSIONS

This paper explores a new approach to filtering laser-scanning dataset for the extraction of spatial objects based unsupervised clustering technique. It presents an advantage in the sense that there is no prior knowledge is needed for such learning processes, i.e. data samples group themselves in terms of similarity. We develop an interactive environment integrated a SOM view, 2D and 3D views of the dataset, thus it facilities detections of clusters associated with different spatial objects. Despite the preliminary nature of the case study, it does illustrate the powerfulness of unsupervised methods in general and SOM in particular in extracting spatial objects from a laser-scanning dataset. It is important to note that SOM training process is much dependent on the parameter settings as reported in table 1. This issue deserves further research, in particular in terms of how the parameter settings have impact on the extraction of spatial objects from a point cloud.

ACKNOWLEDGEMENTS

The author would like to thank Mikael Östlund from the GIS institute at the University of Gävle who provides the datasets for the case study.

REFERENCES

Ackermanm, F. (1999). Airborne laser-scanning - present status and future expectations. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54: 64-7.

Axelsson P. (1999), Processing of Laser Scanner Data – Algorithms and applications, *ISPRS Journal of Phtotogrammetry & Remote Sensing*, 54, pp. 138 – 147.

Haala N. and Brenner C. (1999), Extraction of buildings and trees in urban environments, *ISPRS journal of photogrammetry & remote sensing*, 54, pp. 130 – 137.

Højholt P. (1995). Generalization of build-up areas using Kohonen-networks, *Proceedings of Eurocarto XIII*, 2-4 October, Ispra, Italy.

Jiang B. and Harrie L. (2003), Selection of streets from a network using self-organizing maps, ICA Generalization Workshop, Paris, 28 – 30 April 2003 (a revised version of this paper is to appear in *Transactions in GIS*, Blackwell Publishers, Vol. 8, No.3)

Jiang B. and Nakos B. (2003), Line simplification using self-organizing maps, a working paper presented at *ISPRS workshop on spatial analysis and decision making*, 3 – 5 December 2003, Hong Kong.

Kohonen T. (2001), *Self-Organizing Maps* (third edition), Springer, Berlin, Heidelberg, New York.

Kraus K. and Pfeifer N. (1998), Determination of terrain models in wooded areas with airborne laser scanner data. *ISPRS Journal of Photogrammetry & Remote Sensing*, 53:193-203.

Maas H. and Vosselman G. (1999), Two algorithms for extacting building models from raw laser altimetry data, *ISPRS journal of photogrammetry and remote sensing*, 54, pp. 153 – 163.

McIntosh K. and Krupnik A. (2002), Integration of laser-derived DSMs and matched image edges for generating an accurate surface model. *ISPRS Journal of Photogrammetry & Remote Sensing*, 53(3):167-176.

Oja E. and Kaski S. (editors. 1999), *Kohonen Maps*, Elsevier.

Openshaw S., Blake M. and Wymer C. (1995), Using Neurocomputing Methods to Classify Britain's Residential Areas, (html paper), available at http://www.geog.leeds.ac.uk/papers/95-1/ (accessed on 2002-11-15).

Openshaw, S. (1994), Neuroclassification of spatial data, in Hewitson, B. C. and Crane, R. G. (eds), *Neural Nets: Applications in Geography*, Dordrecht: Kluwer Academic Publishers, 53-70.

Petzold B., Reiss P., and Stössel W. (1999), Laser-scanning - surveying and mapping agencies are using a new technique for the derivation of digital terrain models. *ISPRS Journal of Photogrammetry & Remote Sensing*, 54(2-3):95-104.

Sithole G. and Vosselman G. (2003), *Report: ISPRS Comparison of Filters*, Department of Geodesy, Faculty of Civil Engineering and Geosciences, Delft University of Technology, available at http://www.geo.tudelft.nl/frs/isprs/filtertest/Report05082003.pdf (accessed on 2003-11-12)

Tao C. V., Hu Y., 2001: "A review of post-processing algorithms for airborne LIDAR Data". *Proceedings ASPRS conference* April 23-27, 2001. St. Louis Missouri. CD-ROM, 14 pages.

Tobler W. R. (1970), A Computer Movie Simulating Urban Growth in Detroit Region, *Economic Geography*, 46, pp. 234-240.

Ultsch A. and Siemon H. P. (1990), Kohonen's self organizing feature maps for exploratory data analysis, In *Proc. INNC'90, int. neural network conf.,* page 305-308, Dordrecht, Netherlands, Kluwer.

Vesanto J. (1999), SOM-based data visualization methods, *Intelligent data analysis*, Elsevier Science, Volume 3(2), pp. 111-126.

Vesanto J., Himberg J., Alhoniemi E. and Parhankangas J. (2000), *SOM toolbox for Matlab 5*, Report A57, Helsinki University of Technology, Libella Oy, Espoo.

Vosselman, G., 2000. Slope based filtering of laser altimetry data, *International Archives of Photogrammetry and Remote Sensing*, Volume XXXIII, Amsterdam, pp. 935-942.