ACTA UNIVERSITATIS AGRICULTURAE ET SILVICULTURAE MENDELIANAE BRUNENSIS

Volume LXI 271 Number 7, 2013
http://dx.doi.org/10.11118/actaun201361072415

# POINT CLOUD PROCESSING FOR SMART SYSTEMS

Jaromír Landa, David Procházka, Jiří Šťastný

**Received: April 11, 2013**

## Abstract

LANDA JAROMÍR, PROCHÁZKA DAVID, ŠŤASTNÝ JIŘÍ: *Point cloud processing for smart systems.* Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis, 2013, LXI, No. 7, pp. 2415–2421

High population as well as the economical tension emphasises the necessity of effective city management – from land use planning to urban green maintenance. The management effectiveness is based on precise knowledge of the city environment. Point clouds generated by mobile and terrestrial laser scanners provide precise data about objects in the scanner vicinity. From these data pieces the state of the roads, buildings, trees and other objects important for this decision-making process can be obtained. Generally, they can support the idea of "smart" or at least "smarter" cities.
Unfortunately the point clouds do not provide this type of information automatically. It has to be extracted. This extraction is done by expert personnel or by object recognition software. As the point clouds can represent large areas (streets or even cities), usage of expert personnel to identify the required objects can be very time-consuming, therefore cost ineffective. Object recognition software allows us to detect and identify required objects semi-automatically or automatically.
The first part of the article reviews and analyses the state of current art point cloud object recognition techniques. The following part presents common formats used for point cloud storage and frequently used software tools for point cloud processing. Further, a method for extraction of geospatial information about detected objects is proposed. Therefore, the method can be used not only to recognize the existence and shape of certain objects, but also to retrieve their geospatial properties. These objects can be later directly used in various GIS systems for further analyses.

point cloud, object recognition, GIS

Laser scanning is a technique widely used for many years. It is possible to find many applications from airborne laser scanning to terrestrial laser measurements (see overview in Vosselman (2010) and many others). The usual goal of these measurements is to obtain a shape of a selected object. In case of the airborne scanning, it is frequently a digital elevation model. In case of the terrestrial measurement, it is possible to reconstruct a shape of a house, bridge, even a complex machine. Generally, the goal is to measure as precise as possible the shape of a target structure and create a computer model of its surface. If we simplify the problem, we can say that the whole process is composed of several steps: Firstly, a measurement is done. The output is a set of reflected laser beams – so called point cloud. These points have position, refraction index and potentially other information pieces. Further, the point cloud is loaded into a specialized application and the object surface is reconstructed. In case of terrain measurements, geospatial approximation methods are usually used (see Lloyd, 2010). In case of artificial objects (e.g. buildings), many mathematical approximation methods are used (B-Splines or NURBS approximation etc.) (Robin, McLeod, Baart, 1998). These methods are not designed solely for point cloud processing. We can also apply them for path approximation for CNC machines (Procházková, Sedlák, Procházka, 2007), reconstruction of surfaces from optical data (e.g. microscopes in Matoušek, Martišek, Procházková (2007)) or other curve or surface approximations.

All mentioned applications are based on an assumption that there is a known specific object, and the process itself is just about its modelling.

This article is focused on another problem: we have a complex point cloud with unknown objects (a street with different objects), and it is necessary to identify both the surface and the kind of measured objects (e.g. what part of the point cloud is a tree, car or house). This object identification within the point clouds has substantial potential for many fields. We can compare it to expansion of computer vision that brought many revolutionary applications from automatic traffic signs identification in cars to searching using images in Google search engine. One of the promising applications is automatic passportization of state/municipal property (green on public places and alongside roads, traffic signs etc.). Therefore, we focused our article on identification of objects that can be found at these public places. The goal of the described process is to simplify management of this property.

This paper presents an overview of key steps that must be performed to fulfil this task, as well as common methods, tools and file formats. The problem is explained in pole-like object recognition case study.

## MATERIALS AND METHODS

This section describes selected methods for point cloud processing. These methods are divided into two parts: point cloud preprocessing and object recognition. The first category describes widely used methods, such as segmentation, outliers removal and sampling. The second category describes methods used to recognize certain objects found in urban environments, e.g. roads and pole-like objects.

### Point Cloud preprocessing

Preprocessing is a very important step in the point cloud processing pipeline. Usually the number of points in a scene representing an urban environment is very high. It is not uncommon for the point cloud to have more than million points. It is very difficult to process this amount of points in a reasonable amount of time. This subsection describes three most common groups of preprocessing methods: outliers removal, sampling and segmentation.

One of the key problems in point cloud processing is a noise. If the cloud contains a lot of noise, the entire processing process can be at risk. Noise represents points that are not at the scanned surface. These points are called outliers. The basic and widely used methods use the points neighbourhood to determine if the point is an inlier or outlier. They use either the distance from the point to other points or the point density (Sotoodeh, 2006). The point that does not meet certain criteria is characterized as an outlier. More about removing outliers in Schall, Belyaev, Seidel (2005) and Sotoodeh (2006).

Sampling, or rather downsampling is a process of lowering the number of points while preserving the properties of the scanned scene. Xiaohui (2007) divides sampling into three categories: re-sampling, iterative simplification and clustering. Re-sampling computes a new sampling point set based on given rules. Iterative simplification successively picks point-pairs and collapses them into a single point. The idea of clustering is to partition the cloud into subsets and unify points within each subset. There are also methods for sampling the surface created from the point cloud rather the sampling the cloud itself (Pauly, Gross, Kobbelt, 2002).

The last very common preprocessing task is segmentation. Segmentation is a process of identification of points with similar or same attributes. These points subsequently form smaller clouds. These clouds usually represent certain objects or part of objects, such as trees, buildings, cars (Xiaojuan, 2009). Rabbani and Vosselman divide segmentation algorithms into three categories. Edge-based segmentation detects the edges outlining different regions and groups together the points inside these regions. Surface-based segmentation uses local surface properties to merge points which are close and have similar surface properties. Scanline-based threats each line in a point cloud as a scan line. At each scan line, different segments are identified and then grouped together through all scan lines.

There are many other preprocessing methods (Škorpil and Šťastný (2008), Šťastný and Škorpil (2007)) besides approaches mentioned in this article. The example is the registration technique that is used to find correspondences in a point cloud and combines them to create a large cloud.

### Object recognition

This section deals with detection and recognition of objects in point clouds. Object recognition is used to identify a certain object in point clouds. These objects range from poles and trees, to facades, cars and even humans. Object recognition can be used not only to identify objects, but also to detect changes in them, such as changes in vegetation (Wen *et al.*, 2012). This section describes selected approaches for detection of two common objects in urban environments, pole-like objects and roads.

There are two main approaches for pole-like objects recognition. The first approach tries to evaluate the points neighbourhood (El-Halawany, 2011). Points on poles have certain properties, such as high density in the z axis, but low in all other axis. Monnier, Vallet and Soheilian (2012) evaluate the direction of point's neighbourhood, where one main direction means linear neighbourhood and the point is classified as belonging to the pole. The second approach tries to fit the circle to the poles cross section (Bienert, 2007).

Most of the approaches for road detection from the point cloud use curbstones to separate road from the environment (Inrahim, Lichti (2012) and Vosselman). The height difference between the road and pavement is larger than 5 cm. This can be used to identify the roadside. However, sometimes in urban environments the curbstone is missing which

can cause gaps in the data. These gaps are usually approximated to form a full road extraction. More about road detection can be found in Inrahim and Lichti (2012).

## Common point cloud formats and processing tools

There are many formats that can be used to store a point cloud. Basically, any format capable of storing three numbers representing the *x*, *y* and *z* coordinate can be used. However, there are some formats commonly used for point cloud storing. These formats can be divided into two categories, binary and ASCII. The ASCII file format has the advantage of human readability. This section describes three most common formats and common tools for their processing.

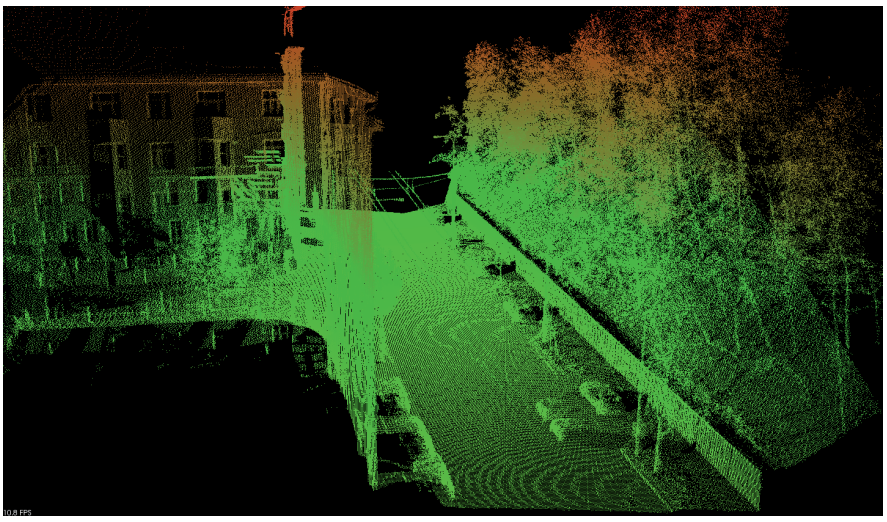
Common format point cloud storage is the PLY[1,2] (*PoLYgon*) file. It is an ASCII text file used primarily for 3D mesh storage. The PLY file consists of two parts. The first part is a header. The header defines what kind of information does the file carry, or rather what kind of information is stored for each point, eg. *x*, *y*, *z* coordinates, normals etc. The second part of the file is a list of points. PLY files can be read by many programs and because it is a text file it is very easy to write your own file handler.

Another example of an ASCII file format is PCD (*Point Cloud Data*). The PCD is very similar to the PLY and is used in an open source library for point cloud processing called *Point Cloud Library*[3] (PCL). The library contains methods for point clouds operations (filtering, segmentation, surface reconstruction, visualization, features and keypoints detection etc.). The PCL is also a cross-platform library and supports 32b and 64b systems (Landa, Procházka, 2012). The PCL has also a huge community of users. That makes it currently the best open-source library for point cloud processing.

The most common file format for point clouds is the LAS (*LASer*). The LAS file format is a public file format for the exchange of 3-dimensional point cloud data. The LAS falls into a category of binary files. The format contains binary data consisting of a *public header block*, optionally any number of *Variable Length Records*, the *Point Data Records*, and optionally any number of *Extended Variable Length Records*. All data are in the little-endian format[4]. The advantage of the LAS file format over ASCII format is the processing performance. The LAS format also retains other information relevant to the each point (e.g. intensity, strength, classification). There are several tools for working with LAS files, such as *libLas*[5] *LASTools*[6]. *LibLAS* is an open-source library written in C++ language capable of reading and writing LAS files. *LASTools* is a set of tools for reading, writing viewing and processing of LAS files. Many commercial applications are also capable of reading LAS files (ArcGIS, AutoCAD and others).

## RESULTS

The results are demonstrated on our object recognition framework that is used to recognize pole-like objects in a point cloud of a street. The raw cloud has 2 580 809 points (Fig. 1). Each is represented by its coordinates in Krovak geospatial coordinate system and the intensity associated with



1: *Raw input cloud of a city street composed of 2 580 809 points*

1 http://people.sc.fsu.edu/~jburkardt/data/ply/ply.html
2 http://graphics.stanford.edu/data/3Dscanrep/
3 http://www.pointclouds.org/
4 http://asprs.org/a/society/committees/standards/LAS_1_4_r12.pdf
5 http://www.liblas.org/
6 http://www.cs.unc.edu/~isenburg/lastools/

it. The framework uses geometric properties of the poles to identify them. We are using *libLAS* to load LAS files and *Point Cloud Library* for point cloud processing operations.

## Ground Removal

The presented algorithm works only with poles that are not standing on the ground. Therefore, the first step is a ground removal. There are two common approaches for this task. The first approach uses the point normals to determine which points are on the ground. The second approach uses height information and removes the lowest part of the cloud. Usually it is 10 per cent. Both approaches have one disadvantage. If the ground is not horizontal (e.g. the scanned area is on the hill), the whole ground will not be removed. The next part describes our ground removal algorithm. The algorithm is designed to adapt to the surface curvature:

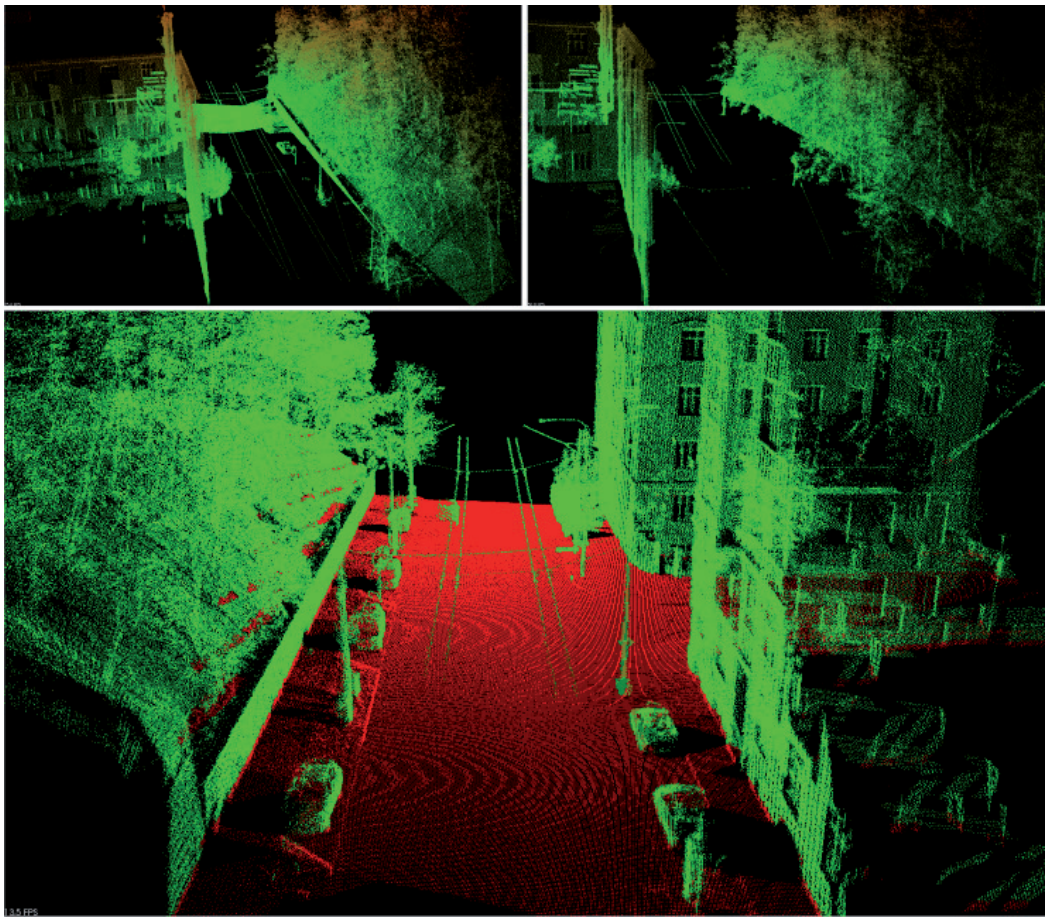
1. Compute cloud bonding box
2. Divide bounding block into 2500 small blocks
3. FOR EACH Block
   - Compute minimum and maximum height in the block
   - Compute height elimination coefficient
   - IF Points height <= height elimination coefficient
     - mark point as ground point
   - ELSE
     - mark point as above ground point

Our height elimination coefficient used in the described algorithm can be calculated:

$$C_{HE} = \min_l H + \Delta H \left( 1 - \frac{\max_g H - \max_l H}{\Delta H} \right) + C_E, \qquad (1)$$

where:
$\min_l H$ ..... local minimum height
$\max_l H$ ..... local maximum height
$\max_g H$ ..... global maximum height
$\Delta H$ ..... difference between global minimum and global maximum of height
$C_E$ ..... height coefficient.

The height coefficient can be set to any number and defines the minimum height for the removal. Usually the coefficient is set to 0.015, which



2: *Ground removal. Top, from left: 10% of lower points removed (distant part of the ground was not removed), 20% of lower points removed. Bottom: ground removal with proposed algorithm (red indicates ground points).*

eliminates ground and all points approximately 0.5 meter above ground.

The advantage of this approach is that it adapts to the terrain height variations and can eliminate ground on very steep terrains. Because it always eliminates only lower parts of the cloud, it never eliminates any points on the pole (see Fig. 2). However, the disadvantage is that it also removes part of the terrain, not just the immediate ground.

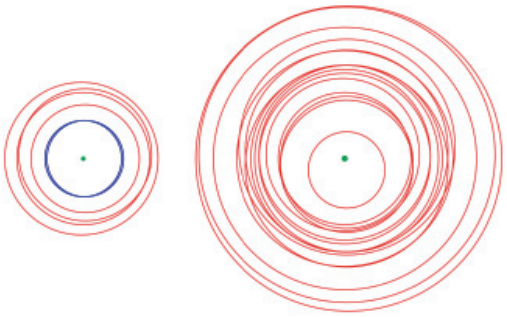
Once the ground is removed, three preprocessing operations are used. The first operation is outliers removal. The algorithm is set to remove all outliers plus many points with lower density neighbourhood. This eliminates most of the points that can be problematic in a later phase, such as leaves and small branches on trees. The following step is sampling. Sampling is used to downsample the cloud to make it more processing friendly (it reduces the number of points).

The last step is segmentation. We are using Euclidian Cluster Extraction algorithm[7] implemented in Point Cloud Library. This algorithm segments the cloud with constraints, such as maximal distance from a point to the segment and minimum/maximum segment size. Each segment it then treated as a potential pole-like object.

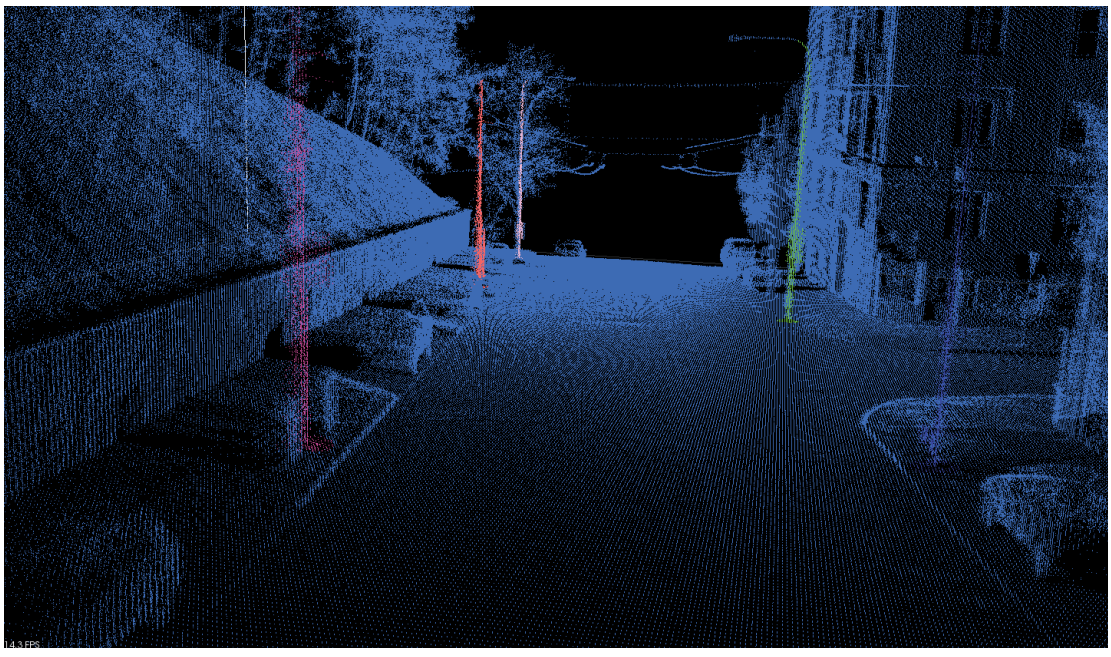
The algorithm for pole recognition is based on a cut algorithm where each segment is divided into n vertical cuts and each cut is treated as a 2D circle. The algorithm is as follows:

- `Divide each segment into n vertical cuts`
- `FOR EACH cut`
  - `compute centroid and 2D circle radius around centroid`
  - `Store circle radius in a group`
- `FOR EACH circle group`
  - `IF (maximum height - minimum height) > 4`
    - `Compute median and variance for x,y and radius`
    - `IF variance < varianceThreshold && median radius > median radius Threshold`
      - `mark group as pole object`

The second part of the algorithm is designed to identify evenly distributed points inside a cylinder. We compute variance and compare the variance in x and y axis with our variance threshold values. If the variance is large enough, it means the circles are not evenly distributed. For instance, if the variance in x axis is large, it usually means there was an



3: *Detected circles. Left: pole where blue circles represent 12 circles with same radius. Right: Tree.*



4: *Final cloud with recognized poles. Poles are highlighted with different colours.*

7 http://pointclouds.org/documentation/tutorials/cluster_extraction.php#cluster-extraction

inclined tree. This leaves us with only large pole-like objects. However, these objects can be either poles of street lighting or tall trees. To distinguish the tree from pole, we compare the variance in radius. This comparison can be seen in Fig. 3.

Slight deviations from the radius are caused by objects on the poles, such as traffic signs and advertisements. However, the centroids and circles retain their even distribution (they have small variance around the *x*, *y* axis and *radius*).

## DISCUSSION

Our algorithm described in this article is able to remove ground points much better than commonly used ground removal algorithms. Most of the algorithms do not consider substantial changes in surface elevation. Nonetheless, these are very common in urban areas. Our algorithm is able to remove effectively around 0.5 m of lowest points from the input point cloud. Substantial advantage of this algorithm is that it is very computationally friendly because it uses only basic mathematical equations (e.g. average) and does not detect any objects. Hence, it can be used even on large point clouds. However, the disadvantage of this algorithm is that it can remove points that are not necessarily the ground points (points around the road). Even with this disadvantage, the algorithm never removes any points on the pole-like objects. Therefore, it is ideal for applications where the objects such as trees, poles and facades are detected.

The second algorithm, described in this article is used to identify pole-like objects. The advantage of this algorithm is that it anew uses only statistical analysis, therefore, it is very fast. Another advantage is that it is able to recognize even poles that have signs and advertisements on them. Nonetheless, the disadvantage of this algorithm is that it cannot recognize poles that are occluded with another object, such as trees.

## SUMMARY

This paper introduces the key issues of object recognition in point clouds. The whole problem is illustrated on street lighting poles recognition task. As it is obvious from the pictures, our proposed algorithm is able to clear the point cloud, remove unnecessary points and distinguish the poles and trees. Nevertheless, this particular case was relatively simple. A substantial amount of research must be done before it will be able to detect automatically all common objects that can be found on the street. Nonetheless, even ability to recognize such simple objects as traffic signs, street lighting, etc. brings significant simplification into their management. It is no longer necessary to control their state visually by manpower, it can be done automatically using generally available data.

Acknowledgement

This work was supported by grant IGA FBE MENDELU 18/2013 (Enhancement of property management effectiveness using point clouds). Data used in this article were provided by GEODIS BRNO, spol. s r. o.

## REFERENCES

BIENERT, A., 2007: *Tree Detection and Diameter Estimations by Analysis of Forest Terrestrial Laserscanner Point Clouds*, ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007, 3: 50–55. Cited from www.dresden-uni.com/die.../2007_Bienert_Helsinki2007.pdf.

EL-HALAWANY, S. I., 2011: Detection of Road Poles from Mobile Terrestrial Laser Scanner Point Cloud. *Multi-Platform/Multi-Sensor Remote Sensing and Mapping (M2RSM), 2011 International Workshop*. pp. 1–6.

IBRAHIM, S., LICHTI, D., 2012: *Curb-Based Street Floor Extraction From Mobile Terrestrial Lidar Point Cloud*, International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XXXIX-B5, 2012. Cited from http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XXXIX-B5/193/2012/isprsarchives-XXXIX-B5-193-2012.pdf.

LANDA, J., PROCHÁZKA, D., 2012: Usage of Microsoft Kinect for augmented prototyping speed-up, *Acta Univ. Agric. et Silvic. Mendel. Brun.*, 60, 2: 175–180. ISSN 1211-8516.

LLOYD, CH. D., 2010: *Local Models for Spatial Analysis*, 2. ed., Boca Raton: CRC Press, 352 p. ISBN 978-1-4398-2919-6.

MARTIŠEK, D., MARTIŠEK, K., PROCHÁZKOVÁ, J., 2007: New methods for space reconstruction of inside cell structures. *Journal of Applied Biomedecine*, 5, 3: 151–155, ISSN 1214-0287.

MONNIER, F., VALLET, B., SOHEILIAN, B., 2012: Trees Detection from Laser Point Clouds Acquired in Dense Urban Areas by a Mobile Mapping System, *XXII ISPRS Congress, Technical Commission III*, Boca Raton: CRC Press, I-3, pp. 245–250. ISBN 978-1439829196.

PAULY, M., GROSS, M., KOBBELT, L. P., 2002: Efficient simplification of point-sampled surfaces, *Proceedings of the conference on Visualization '02*, Washington, DC: IEEE Computer Society, pp. 163–170. ISBN 0-7803-7498-3.

PROCHÁZKOVÁ, J., SEDLÁK, J., PROCHÁZKA, D., 2007: Direct B-Spline Interpolation of CNC Path

from Cloud of Points. *Proceedings of symposium on computer geometry SCG 2007*, Bratislava: Slovenská technická univerzita v Bratislave, pp. 104–110, ISBN 978-80-227-2734-1.

RABBANI, T., VOSSELMAN, C., 2013: Segmentation of Point Clouds Using Smoothness Constraint, *ISPRS Commission V Symposium Image Engineering and Vision Metrology*. [cit. 2013-2-14]. Cited from http://www.isprs.org/proceedings/XXXVI/part5/paper/RABB_639.pdf.

ROBIN, J. Y., MCLEOD, M., BAART, L., 1998: *Geometry and Interpolation of Curves and Surfaces*. Cambridge: Cambridge University Press, 430 p. ISBN 9780521159395.

SCHALL, O., BELYAEV, A, SEIDEL, H-P., 2005: Robust Filtering of Noisy Scattered Point Data. *Eurographics Symposium on Point-Based Graphics*, Aire-la-Ville: Eurographics Association, pp. 71–77. ISBN 3-905673-20-7.

SOTOODEH, S., 2006: Outlier Detection in Laser Scanner Point Clouds. International Archives of Photogrammetry, *Remote Sensing and Spatial Information Sciences*, 35, 5: 297–302. Cited from http://www.isprs.org/proceedings/XXXVI/part5/paper/SOTO_653.pdf.

ŠKORPIL, V., ŠŤASTNÝ, J., 2008: Comparison of Learning Algorithms. In: *24 the Biennial Symposium on Communications*. Kingston, Canada: Queens University Kingston, pp. 231–234. ISBN 978-1-4244-1945-6.

ŠŤASTNÝ, J., ŠKORPIL, V., 2007: *Analysis of Algorithms for Radial Basis Function Neural Network*, Personal Wireless Communications, Springer, issue 1, pp. 54–62. ISBN 978-0-387-74158-1.

VOSSELMAN, G., *Advanced Point Cloud Processing*. [cit. 2013-2-14]. Cited from http://www.ifp.uni-stuttgart.de/publications/phowo09/160Vosselman.pdf.

VOSSELMAN, G., 2010: *Hans-Gerd Maas: Airborne and Terrestrial Laser Scanning*. 1st ed. Boca Raton: CRC Press, 336 p. ISBN 1439827982.

WEN, X. et al, 2012: *Change detection of trees in urban areas using multi-temporal airborne lidar point clouds*, Proc. SPIE 8532, Remote Sensing of the Ocean, Sea Ice, Coastal Waters, and Large Water Regions 2012. Cited from: http://recherche.ign.fr/labos/matis/pdf/articles_conf/2012/SPIEproceedingWenXiao.pdf.

XIAOHUI, D., 2007: Adaptive Out-of-Core Simplification of Large Point Clouds, *Multimedia and Expo, 2007 IEEE International Conference*, Beijing: IEEE, pp. 1439–1442. ISBN 1-4244-1016-9.

XIAOJUAN, N. et al, 2009: Segmentation of architecture shape information from 3D point cloud, *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry*, New York: ACM New York, pp. 127–132. ISBN 978-1-60558-912-1.

Address

Ing. Jaromír Landa, Ing. David Procházka, Ph.D., prof. RNDr. Ing. Jiří Šťastný, CSc., Department of Informatics, Mendel University in Brno, Zemědělská 1, 613 00, Brno, Czech Republic, e-mail: jaromir.landa@mendelu.cz, david.prochazka@mendelu.cz, jiri.stastny@mendelu.cz