MTA SZTAKI
INSTITUTE FOR COMPUTER SCIENCE AND CONTROL HUNGARIAN ACADEMY OF SCIENCES
MAGYAR TUDOMÁNYOS AKADÉMIA SZÁMÍTÁSTECHNIKAI ÉS AUTOMATIZÁLÁSI KUTATÓINTÉZET

Urban Traffic Monitoring from LIDAR Data with a Two-Level Marked Point Process Model

Attila Börcs — Csaba Benedek

Technical Report

N° i4D-2

January 2013

Theme VISION

MTA SZTAKI
H-1111 Budapest, Kende u. 13-17, Hungary
H-1518 Budapest P.O.B. 63, Hungary
Phone: (+36 1) 279 6000
Fax: (+36 1) 466 7503

Urban Traffic Monitoring from LIDAR Data with a Two-Level Marked Point Process Model

Attila Börcs[*], Csaba Benedek[†]

Theme VISION — Computer Vision
Divison: Distributed Events Analysis Research Laboratory

Research report — January 2013 — 20 pages

Abstract: In this report we present a new object based hierarchical model for joint probabilistic extraction of vehicles and coherent vehicle groups – called *traffic segments* – in airborne and terrestrial LIDAR point clouds collected from crowded urban areas. Firstly, the 3D point set is segmented into terrain, vehicle, roof, vegetation and clutter classes. Then the points with the corresponding class labels and intensity values are projected to the ground plane. In the obtained 2D class and intensity maps we approximate the top view projections of vehicles by rectangles. Since our tasks are simultaneously the extraction of the rectangle population which describes the position, size and orientation of the vehicles and grouping the vehicles into the traffic segments, we propose a hierarchical, Two-Level Marked Point Process (L^{2}MPP) model for the problem. The output vehicle and traffic segment configurations are extracted by an iterative stochastic optimization algorithm. We have tested the proposed method with real aerial and terrestrial LiDAR measurements. Our aerial data set contains 471 vehicles, and we provide quantitative object and pixel level comparions results versus two state-of-the-art solutions.

Key-words: rotating multi-beam Lidar, MPP, vehicle detection, traffic monitoring

This work is connected to the i4D project funded by the internal R&D grant of MTA SZTAKI.

[*] The author was supported by the Grant #101598 of the Hungarian Research Fund (OTKA)
[†] The author was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences and by the Grant #101598 of the Hungarian Research Fund (OTKA)

Institute for Computer Science and Control
Hungarian Academy of Sciences
H-1111, Budapest Kende utca 13-17, Hungary
Telephone: +36 1 279 6000, Fax: +36 1 466 7503
http://www.sztaki.hu

Városi forgalomfelügyelet kétszintű jelölt pontfolyamat modellel LIDAR felvételeken

Kivonat : Riportunkban egy új objektum alapú hierarchikus valószínűségi modellt mutatunk be, melynek célja távérzékelt városi LiDAR pontfelhőkben lévő járművek észlelése és a forgalmi szempontból összetartozó járműcsoportok, *forgalmi szegmensek*, kinyerése. Első lépésként a háromdimenziós ponthalmazt szegmentáljuk, megkülönböztetve a *növényzet*, *járműjelölt*, *épületek tetőszerkezetei*, illetve *ritka ponthalmaz* osztályokat. Ezután az egyes pontokhoz tartozó osztálycímkéket és a LiDAR eszköz által mért intenzitás (visszaverődés erősség) értékeket a talaj síkjára vetítjük. Az így kapott 2D címke- és intenzitásképen a felülnézetből látszódó járműveket téglalapokkal közelítjük. Mivel feladatunk egy időben a járművek elhelyezkedését és dimenzióit leíró téglalap populáció megtalálása, valamint az objektumok csoportosítása forgalmi szegmensekbe, egy hierarchikus, kétszintű jelölt pontfolyamat modellt (L^2MPP - Two-Level Marked Point Process) dolgoztunk ki a probléma megoldására. Az optimális jármű és forgalmi szegmens konfigurációt iteratív sztochasztikus algoritmussal határozzuk meg. A módszert valódi, összesen 471 járművet tartalmazó légi LiDAR adathalmazokon teszteltük, kvantitatív módon kiértékeltük, és eredményességét két szakirodalmi módszerrel összehasonlítva igazoltuk. Kiterjesztést mutatunk be földi LiDAR mérések kezelésére is.

Kulcsszavak : Lidar, jelölt Markovi pontfolyamatok, jármű detekció, forgalom figyelés

Contents

1 Introduction

Automatic traffic monitoring is a central goal of urban traffic control, environmental protection and aerial surveillance applications. Complex traffic analysis needs a hierarchical modeling approach: at low level *individual vehicles* should be detected and separated, meanwhile at a higher level we need to extract *coherent traffic segments*, by identifying groups of corresponding vehicles, such as cars in a parking lot, or a vehicle queue waiting in front of a traffic light. Here, we introduce a joint probabilistic model for vehicle detection and traffic segmentation in airborne LIDAR data, which contains point position, intensity and echo information.

1.1 Marked Point Processes

We model a traffic scene by a Marked Point Process (MPP) [1], which is an efficient Bayesian tool to characterize object populations, through jointly describing individual objects by various data terms, and using information from entity interactions by prior geometric constraints. However, conventional MPP models offer limited options for hierarchical scene modeling, since they usually exploit pairwise object interactions, which are defined on fixed symmetric object neighborhoods. In a traffic situation we often find several groups of regularly aligned vehicles, but we must also deal with junctions or skewed parking places next to the roads (Fig. 7), where many differently oriented cars appear close to each other. In addition, the coherent car groups may have thin, elongated shapes, therefore concentric neighborhoods are less efficient.

For this reason, we propose here a Two-Level MPP (L^{2}MPP) model, which partitionates the complete vehicle population into vehicle groups, called *traffic segments*, and extracts the vehicles and the optimal segments simultaneously by a joint energy minimization process. Object interactions are differently defined within the same segment and between two different segments, implementing adaptive object neighborhoods. This model extends our single level MPP method [2] proposed for vehicle detection. In addition, we present here an improved point cloud segmentation algorithm, and provide a detailed quantitative evaluation on four datasets of 471 vehicles, considering two reference methods [3, 4].

1.2 Related Work

Vehicle detection on urban roads is a crucial task in automatic traffic monitoring and control, environmental protection and surveillance applications [5]. Beside terrestrial sensors such as video cameras and induction loops, airborne and spaceborne data sources are frequently exploited to support the scene analysis. Some of the existing approaches rely on aerial photos or video sequences, however in these cases, it is notably challenging to develop a widely applicable solution for the recognition problem due to the large variety of camera sensors, image quality, seasonal and weather circumstances, and the richness of the different vehicle prototypes and appearance models [6]. The Light Detection and Ranging (LiDAR) technology offers significant advantages to handle many of the above problems, since it can jointly provide an accurate 3-D geometrical description of the scene, and additional features about the reflection properties and compactness of the surfaces. Moreover the LiDAR measurements are much less sensitive on the weather conditions and independent on the daily illumination. On the other hand, efficient storage, management and interpretation of the irregular LiDAR point clouds require different algorithmic methodologies from standard computer vision techniques.

LiDAR based vehicle detection methods in the literature follow generally either a grid-cell- or a 3-D point-cloud-analysis-based approach [7]. In the first group of techniques [3, 8], the obtained LiDAR data is first transformed into a dense 2.5-D Digital Elevation Model (DEM), thereafter established image processing operations can be adopted to extract the vehicles. On the other hand, in point cloud based methods [5], the feature extraction and recognition steps work directly on the 3-D point clouds: in this way we avoid loosing information due to projection and interpolation, however time and memory requirement of the processing algorithms may be higher. We propose a hybrid model, where the initial point cloud is classified via 3D features, but the optimal object configuration is extracted in a 2D lattice, after ground plane projection.

Another important factor is related to the types of measurements utilized in the detection. A couple of earlier works combined multiple data sources, e.g. [9] fused LiDAR and digital camera inputs. Other methods rely purely on geometric information [4, 8], emphasizing that these approaches are independent on the availability of RGB sensors and limitations of image-to-point-cloud registration techniques. Several LiDAR sensors, however, provide an intensity value for each data point, which is related to the intensity of the given laser return. Since in general the shiny surfaces of car bodies result in higher intensities, this feature can be utilized as an additional evidence for extracting the vehicles.

The vehicle detection techniques should also be examined from the point of view of object recognition methodologies. Machine learning methods offer noticeable solutions, e.g. [8] adopts a cascade AdaBoost framework to train a classifier based on edgelet features. However, the authors also mention that it is often difficult to collect enough representative training samples, therefore, they generate more training examples by shifting and rotating the few training annotations. Model based methods attempt to fit 2-D or 3-D car models to the observed data [5], however, these approaches may face limitation for scenarios where complex and highly various vehicle shapes are expected.

We can also group the existing object modeling techniques whether they follow a *bottom-up* or an *inverse* approach. The *bottom-up* techniques usually consist in extracting *primitives* (blobs, edges, corners etc.) and thereafter, the objects are constructed from the obtained features by a sequential process. To extract the vehicles, [3] introduces three different methods with similar performance results, which combine surface warping, Delaunay triangulation, thresholding and Connected Component Analysis (CCA). As main bottlenecks here, the Digital Terrain Model (DTM) estimation and appropriate height threshold selection steps critically influence the output quality. [4] applies three consecutive steps: geo-tiling, vehicle-top detection by local maximum filtering and segmentation through marker-controlled watershed transformation. The output is a set of vehicles contours, however, some car silhouettes are only partially extracted and a couple of neighboring objects are merged into the same blob. In general, bottom-up techniques can be relatively fast, however construction of appropriate primitive filters may be difficult/inaccurate, and in the sequential work flows, the failure each step may corrupt the whole process. In addition, we have limited options here to incorporate a priori information (e.g. shape, size) and object interaction.

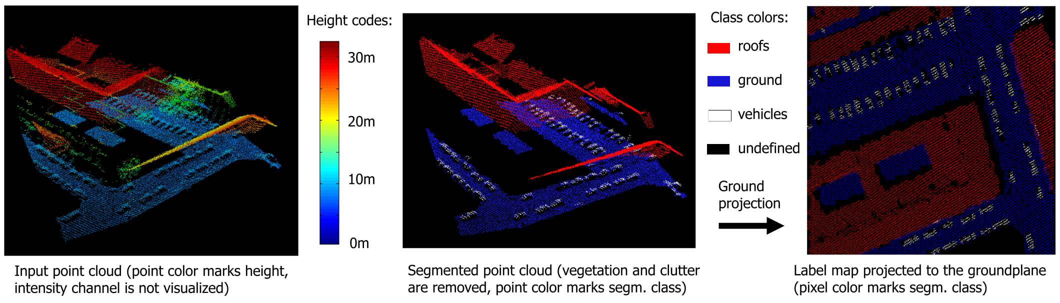

Height codes:
30m
20m
10m
0m
Class colors:
roofs
ground
vehicles
undefined
Ground projection
Input point cloud (point color marks height, intensity channel is not visualized)
Segmented point cloud (vegetation and clutter are removed, point color marks segm. class)
Label map projected to the groundplane (pixel color marks segm. class)

Figure 1: Workflow of the point cloud filtering, segmentation and projection steps. Test data provider: Astrium GEO-Inf. Services©

Inverse methods, such as Marked Point Processes, MPPs, [1, 10], assign a fitness value to each possible object configuration, thereafter an optimization process attempts to find the configuration with the highest confidence. In this way complex object appearance models can be used, it is easy to incorporate prior shape information (e.g. only searching among rectangles) and object interactions (e.g. penalize intersection, favor similar orientation). However, high computational need is present due searching in the high dimension population space. Therefore, applying efficient optimization techniques is a crucial need.

We propose an MPP based vehicle detection method with the following key features. (i) Instead of utilizing complex image descriptors and machine learning techniques to characterize the individual vehicle samples, only basic radiometric evidences, segmentation labels and prior knowledge about the approximate size and height of the vehicle bounding boxes are exploited. (ii) We model interaction between the neighboring vehicles by prescribing prior non-overlapping, width similarity and favored alignment constraints. (iii) Features exploited in the recognition process are directly derived from the segmentation of the LiDAR point cloud in 3-D. However, to keep the computational time tractable, the optimization of the inverse problem is performed in 2-D, following a ground projection of the previously obtained class labels. (iv) During the projection of the LiDAR point cloud to the ground (i.e. a regular image), we do not interpolate pixel values with missing data, but include in the MPP model the concept of *pixel with unknown class*. In this way we avoid possible artifacts of data interpolation.

2 Segmentation of aerial point clouds

In this section, we introduce our point cloud segmentation method for aerial LiDAR measurements.

The input of the proposed framework is a remotely sensed LiDAR point cloud $\mathcal{L}$. Let us assume that the cloud consists of l points: $\mathcal{L} = \{p_1, \ldots, p_l\}$, where each point, $p \in \mathcal{L}$, is associated to geometric position, intensity and echo number parameters, as detailed in Table 1. Let us denote by

Parameter	Domain	Description
x_p, y_p, z_p	$\mathbb{R}^3$	coordinates of the 3-D geometric location of the point p
g_p	[0,255]	intensity (or gray level) value associated to the point p
n_p	$\{1, 2, 3, 4\}$	number of echoes (or returns) from the direction of p
r_p	$\{1, 2, 3, 4\}$	index (ordinary number) of the echo associated to point p from its direction (i.e. $r_p \leq n_p$)

Table 1: Parameters associated to a point p of the input cloud $\mathcal{L}$

$\mathcal{V}_\epsilon(p)$ the ϵ neighborhood of p:

$$\mathcal{V}_\epsilon(p) = \{q \in \mathcal{L} : ||q - p|| < \epsilon\},$$

where $||r - p||$ marks the Euclidean distance of points r and p. Then with using $|\mathcal{V}_\epsilon(p)|$ for the cardinality of a neighborhood:

$$\mu(p) = \text{clutter iff } |\mathcal{V}_\epsilon(p)| < \tau_\mathcal{V},$$

where ϵ and $\tau_\mathcal{V}$ threshold parameters depend on the point cloud resolution and density. For efficient neighborhood calculation, we need to divide the point cloud into smaller parts by making a nonuniform subdivision of the 3-D space using a *k*-d tree data structure.

For point cloud segmentation we have proposed an energy minimization method in the 3D space, which utilizes various 3D descriptors to identify the different point classes. In our model, we distinguish *terrain*, *vegetation*, *roof*, *vehicle* and *sparse* regions, and we denote by $\xi(p)$ the class label of a given point p.

To classify the point cloud, we define for each class ξ a $\mu_\xi(p) \in [0, 1]$ inverse membership function, which evaluates the hypothesis that $p \in \mathcal{L}$ belongs to the ξ segmentation class, marking high quality matches with lower μ values. For deriving the membership functions we use ζ sigmoid functions, which can be considered as *soft thresholds*:

$$\zeta(x, \tau, m) = \frac{1}{1 + \exp(-m \cdot (x - \tau))}.$$

where $x \in \mathbb{R}$ is a scalar valued fitness descriptor, τ is the soft threshold corresponding to x, a m is a steepness parameter used for normalization.

We identify the *terrain* points, by estimating the the best plane $\mathcal{P}$ in the cloud $\mathcal{L} \setminus \mathcal{L}_{\text{cv}}$ using a RANSAC-based algorithm of [11]. This technique selects in each iteration three points randomly from the input cloud, and it calculates the parameters of the corresponding plane. Then it counts the points in $\mathcal{L} \setminus \mathcal{L}_{\text{cv}}$ which fit the new plane and compares the obtained result with the last saved one. If the new result is better, the estimated plane is replaced with the new candidate. The process is iterated till convergence is obtained. Since the ground is usually not planar in a greater area, large point clouds are first be divided into smaller segment, and the ground plane is estimated within each segment separately. Thereafter the points are evaluated based on their $d_p^T = \mathrm{dist}(p, T)$ distance measured from the local ground plane:

$$\mu_{\text{terrain}}(p) = \zeta\left(d_p^T, \tau_{\text{terrain}}, m_{\text{terrain}}\right),$$

where τ_{terrain} is a height threshold depending on the geometric accuracy of the LiDAR data and m_{terrain} is a normalizing parameter. We set these factors in a supervised way by training regions, since they highly depend on the noise level and point density of the measurement.

For estimating the *vegetation*, we analyzed the return (echo) numbers of the points. As detailed in Table 1, the LiDAR system provides apart from the 3D point position coordinates, the number of laser returns from the direction of point p (n_p), and the reflection index corresponding to p (r_p). Typically, in regions covered by vegetation we can observe multiple lase returns ($r_p < n_p$ i.e. $n_p - r_p \geq 1$) which gives as evidences to filter trees and bushes:

$$\mu_{\text{vegetation}}(p) = 1 - \zeta\left(n_p - r_p, 0.5, m_{\text{vegetation}}\right).$$

Regarding the *roof* class, we assume that the d_p^T height parameter of the points exceeds a τ_{roof} threshold, and the points form dense regions, so that $|\mathcal{V}_\epsilon(p)| > \tau_{\mathcal{V}}$. The corresponding data term is:

$$\mu_{\text{roof}}(p) = \Big(1 - \zeta\left(d_p^T, \tau_{\text{roof}}, m_{\text{roof}}\right)\Big) \cdot \Big(1 - \zeta\left(|\mathcal{V}_\epsilon(p)|, \tau_{\mathcal{V}}, m_{\mathcal{V}}\right)\Big)$$

In *sparse regions*, in contrast with the previous case, we expect at most a few neighbors around each point

$$\mu_{\text{sparse}}(p) = \zeta\left(|\mathcal{V}_\epsilon(p)|, \tau_{\mathcal{V}}, m_{\mathcal{V}}\right)$$

Finally, for points corresponding to vehicles, we expect that the height from the local terrain plain segment were between a minimal (τ_{jmin}) and maximal (τ_{jmax}) height value, and the should correspond to the last reflection from the direction corresponding to them:

$$\mu_{\text{vehicle}}(p) = \zeta\left(d_p^T, \tau_{\text{jmax}}, m_{\text{vehicle}}\right) \cdot \Big(1 - \zeta\left(d_p^T, \tau_{\text{jmin}}, m_{\text{vehicle}}\right)\Big) \cdot \zeta\left(n_p - r_p, 0.5, m_{\text{n}}\right)$$

After constructing the membership functions, we define an E energy function on the space of the possible global point cloud labellings, which uses the Potts model to describe the neighborhood interactions similarly to [12].

$$E(\{\xi(p)|p \in \mathcal{L}\}) = \sum_{p\in\mathcal{L}} \mu_{\xi(p)}(p) + \sum_{p\in\mathcal{L}} \sum_{r\in\mathcal{V}_\epsilon(p)} \kappa \cdot \mathbf{1}\left\{\xi(p) \neq \xi(r)\right\} \tag{1}$$

where $\kappa > 0$ is the weight of the intrraction term and $\mathbf{1}\{.\}$ is an indicator function: $\mathbf{1}\{\mathrm{true}\} = 1$, $\mathbf{1}\{\mathrm{false}\} = 0$.

For the minimum of (1), we can get an efficient approximation by graph-cut based techniques, which we have tested using the implementation of [13]. However, we have also experienced that compared to the point-by-point segmentation (which ignores the Potts smoothing terms), the quick

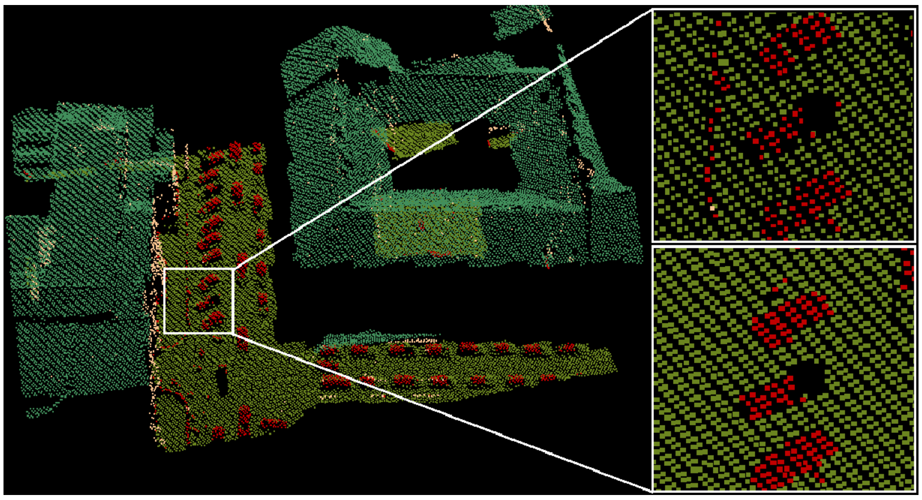


Figure 2: Results of point cloud segmentation in a data sample. Top right: result of point-by-point classification. Bottom right: classification obtained by the minimization of (1) with the *ICM* algorithm

Iterated Conditional Modes (ICM) optiomization can also provide significant improvements which is demonstrated in Fig. 2.

After the 3-D segmentation process, we stretch a 2-D pixel lattice S (i.e. an image) onto the ground plane, where $s \in S$ denotes a single pixel. Then, we project each LiDAR point to this lattice, which has a label of ground, vehicle or building roof. This projection results in a 2-D class label map and an intensity map, where multiple point projections to the same pixel are handled by a point selection algorithm, which gives higher precedence to vehicle point candidates. On the other hand, the projection of the sparse point cloud to a regular image lattice results in many pixels with undefined class labels and intensities. In contrast to several previous solutions, we do not interpolate these missing points, but include in the upcoming model the concept of unknown label at certain pixels. In this way, our approach is not affected by the artifacts of data interpolation.

Let us denote by $\chi(s) \subset \mathcal{L}$ the set of points projected to pixel s. After the projection (Fig. 4), we distinguish $\mathrm{vehicle}$, $\mathrm{background}$ and $\mathrm{undefined}$ classes on the lattice as follows:

$$\nu(s) = \begin{cases} \mathrm{vehicle} & \text{if } \exists p \in \chi(s) : \mu(p) = \mathrm{vehicle} \\ \mathrm{background} & \text{if } \forall p \in \chi(s) : \begin{cases} \mu(p) = \mathrm{roof} \\ \quad \mathrm{OR} \\ \mu(p) = \mathrm{ground} \end{cases} \\ \mathrm{undefined} & \text{if } \chi(s) = \emptyset. \end{cases}$$

Note that for easier visualization, in Fig. 1 and 4 we have distinguished pixels of roof (red) and ground (blue) projections, but during the next steps, we consider them as part of the $\mathrm{background}$

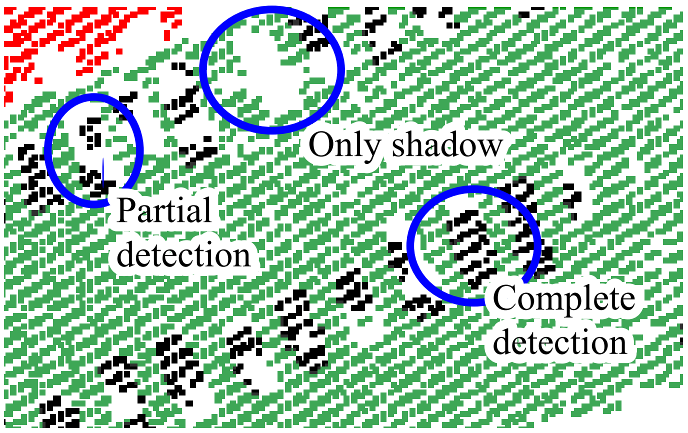


Figure 3: Challenges of vehicle detection in the label map

class. We also assign to each pixel s and intensity value $g(s)$, which is 0, if $\nu(s) = \text{undefined}$, otherwise we take the average intensity of points projected to s.

Note that we may face further challenges regarding vehicle detection from the projected point cloud data. As shown in Fig. 3, we must expect several missing or only partially detected vehicles due to missing data or segmentation errors. An interesting case is shown in the top of Fig. 3, where a car was parking below a tree thus the vehicle points were hidden from the Lidar, but we can observe an appropriately sized hole on the ground which gives evidence for the presence of a car. To estimate the true vehicle shapes, we can exploit some prior expectation such as regular alignment of vehicles, i.e. similar sizes and orientations are expected in local neighborhoods. For this reason we have chosen a population level traffic description approach, where prior information is exploited about vehicle geometry and interaction In the following part of the algorithm, we purely work on the previously extracted label and intensity images. The detection is mainly based on the label map, but additional evidences are extracted from the intensity image, where several cars appear as salient bright blobs due to their shiny surfaces.

3 L^2-Marked Point Process Model

The inputs of this step are the label and intensity maps over the pixel lattice S, which were extracted in the previous section. We will also refer to the input data jointly by $\mathcal{D}$. We assume that each vehicle from top view can be approximated by a rectangle, which we aim to extract by the following model. A vehicle candidate u is described by five parameters: c_x and c_y center coordinates, e_L, e_l side lengths and $\theta \in [-90°, +90°]$ orientation (Fig. 5(c)). Note that with replacing the rectangle shapes for parallelograms, the "shearing effect" of moving vehicles may also be modeled [7], but in the considered test data this phenomenon could not be reliably observed.

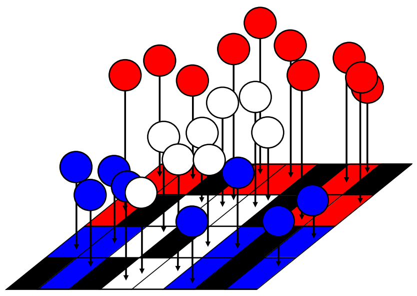


Figure 4: Demonstration of the projection step (best viewed in color). LiDAR points are denoted by spheres, and pixels on the image lattice by cells, with the following color codes: red - roof, blue - ground, white - vehicle. Roof and ground pixels represent the *background* class in the lattice, while black cells correspond to pixels with class label *undefined*.

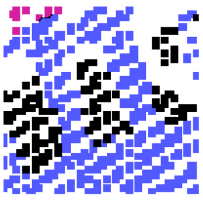
Let $\mathcal{H}$ be the space of u objects. We define a neighborhood relation $\sim$ in $\mathcal{H}$: $u \sim v$ iff the distance of the object centers is smaller than a threshold. We describe the scene by a Two-level Marked Point Process (L^{2}MPP) model: a global configuration ω is a the set of k traffic segments, $\omega = \{\psi_1, \ldots, \psi_k\}$, where each traffic segment ψ_i ($i = 1 \ldots k$) is a configuration of n_i vehicles, $\psi_i = \{u_1^i, \ldots, u_{n_i}^i\} \in \mathcal{H}^{n_i}$. Here we prescribe that $\psi_i \cap \psi_j = \emptyset$ for $i \neq j$, while the k set number and $n_1, \ldots, n_k$ set cardinality values may be arbitrary (and initially unknown) integers. We mark with $u \prec \omega$ if u belongs to any ψ in ω, i.e. $\exists \psi_i \in \omega : u \in \psi_i$. Ω denotes the space of all the possible ω global configurations.

$$\Omega = \cup_{k=0}^{\infty} \left\{ \{\psi_1, \ldots, \psi_k\} \in [\cup_{n=1}^{\infty} \Psi_n]^k \right\} \quad \text{where } \Psi_n = \{\{u_1, \ldots, u_n\} \in \mathcal{H}^n\}$$

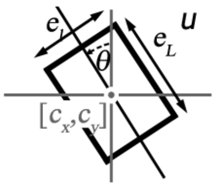
Taking an inverse approach, an energy function $\Phi(\omega)$ is defined, which can evaluate each $\omega \in \Omega$ configuration based on the observed data and prior knowledge. The above neighborhood-energies are constructed by fusing various data terms and prior terms, as introduced in the following subsections in details. Therefore, the energy can be decomposed into a data term and a prior term: $\Phi(\omega) = \Phi_d(\omega) + \Phi_p(\omega)$, and the optimal ω is obtained by minimizing $\Phi(\omega)$.

3.1 Data-dependent energy terms

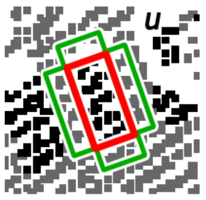
Data terms evaluate the proposed vehicle candidates (i.e. the $u = \{c_x, c_y, e_L, e_l, \theta\}$ rectangles) based on the input label- or intensity maps, but independently of other objects of the population. The data modeling process consists of two steps. *First*, we define different $f(u) : \mathcal{H} \to \mathbb{R}$ features which evaluate a vehicle hypothesis for u in the image, so that 'high' $f(u)$ values correspond to efficient vehicle candidates. In the *second step*, we construct $\varphi_d^f(u)$ *data driven* energy subterms for each feature f, by attempting to satisfy $\varphi_d^f(u) < 0$ for real objects and $\varphi_d^f(u) > 0$ for false candidates. For this purpose, we project the feature domain to $[-1, 1]$ with a monotonously decreasing function:



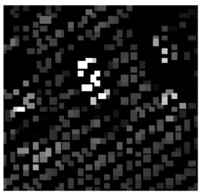
(a) Label map (Colors: vehicle class: black, undefined class: white)



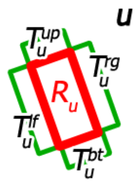
(c) Rectangle parameters of a vehicle candidate *u*



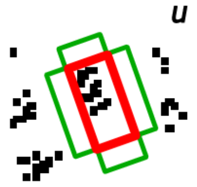
(e) Vehicle model + label map (background class = roof + terrain: gray color)



(b) Intensity map



(d) Internal *(R)* and external *($T^{(\cdot)}$)* regions for *u*



(f) Vehicle candidate + binarzed intensity map

Figure 5: Demonstration of the (a)-(b) input maps (c) object rectangle parameters and (d)-(f) dataterm calculation process

$\varphi_d^f(u) = \mathcal{Q}\big(f(u), d_0^f\big)$, where

$$\mathcal{Q}(x, d_0) = \begin{cases} \left(1 - \frac{x}{d_0}\right), & \text{if } x < d_0 \\ \exp\left(-\frac{x-d_0}{0.1}\right) - 1, & \text{if } x \geq d_0. \end{cases} \tag{2}$$

Observe that the $\mathcal{Q}$ function has a key parameter, d_0^f, which is the object acceptance threshold for feature f: u is acceptable according to the $\varphi_d^f(u)$ term iff $f(u) > d_0^f$.

We used four different data-based features. To introduce them, let us denote by $R_u \subset S$ the pixels of the image lattice lying inside the u vehicle candidate's rectangle, and by T_u^{up}, T_u^{bt}, T_u^{lt}, and T_u^{rg} the upper, bottom, left and right object neighborhood regions, respectively (see Fig. 5). The feature definitions are listed in the following paragraphs.

The *vehicle evidence* feature $f^{\text{ve}}(u)$ expresses that we expect several pixels classified as vehicle within R_u:

$$f^{\text{ve}}(u) = \frac{1}{|R_u|} \sum_{s \in R_u} \mathbf{1}\left\{\nu(s) = \text{vehicle}\right\},$$

where $|R_u|$ denotes the cardinality of R_u, and $\mathbf{1}\{.\}$ marks an indicator function: $\mathbf{1}\{\text{true}\} = 1$, $\mathbf{1}\{\text{false}\} = 0$.

The *external background* feature $f^{\mathrm{eb}}(u)$ measures if the vehicle candidate is surrounded by background regions:

$$f^{\mathrm{eb}}(u) = \operatorname*{min2nd}_{i\in\{\mathrm{up},\mathrm{bt},\mathrm{lt},\mathrm{rg}\}} \left(\frac{1}{|T_u^i|} \sum_{s\in T_u^i} \mathbf{1}\left\{\nu(s) = \mathrm{background}\right\} \right),$$

where the $\mathrm{min2nd}$ operator returns the second smallest element from the background filling ratios of the four neighboring regions: with this choice we also accept vehicles which connect with at most one side to other vehicles or undefined regions.

The *internal background* feature $f^{\mathrm{ib}}(u)$ prescribes that within R_u only very few background pixels may occur:

$$f^{\mathrm{ib}}(u) = \frac{1}{|R_u|} \sum_{s\in R_u} 1 - \mathbf{1}\left\{\nu(s) = \mathrm{background}\right\}.$$

Demonstration of the f^{ve}, f^{eb} and f^{ib} feature calculation can be followed in Fig. 5(e).

Finally, the *intensity* feature provides additional evidence for image parts containing high intensity regions (see Fig. 5(b) and (f)).

$$f^{\mathrm{it}}(u) = \frac{1}{|R_u|} \sum_{s\in R_u} \mathbf{1}\left\{g(s) > T_g\right\},$$

where T_g is an intensity threshold.

After the feature definitions, the data terms $\varphi_d^{\mathrm{it}}(u)$, $\varphi_d^{\mathrm{ve}}(u)$, $\varphi_d^{\mathrm{ib}}(u)$, $\varphi_d^{\mathrm{eb}}(u)$ can be calculated with the $\mathcal{Q}$ function by appropriately fixing the corresponding d_0^f parameters for each feature. We set the parameters based on manually annotated training data, which step can be further optimized by Maximum Likelihood Estimators (MLE) as detailed in [14].

Once we obtained the subterms, the joint data energy of object u is derived as

$$\varphi_d(u) = \max(\min(\varphi_d^{\mathrm{it}}(u), \varphi_d^{\mathrm{ve}}(u)), \varphi_d^{\mathrm{eb}}(u), \varphi_d^{\mathrm{ib}}(u)).$$

Here the min and max operators are equivalent to the logical OR resp. AND operations for the different feature constraints in the negative fitness domain. We do not prescribe simultaneously the *vehicle evidence* and *intensity* constraints, since usually not all vehicles appear as bright blobs in the intensity map. The data term of the ω configuration is obtained as the sum of the individual object energies: $\Phi_d(\omega) = \sum_{u\prec\omega} \varphi_d(u)$.

3.2 Prior terms

The prior terms implement geometric constraints between different objects and traffic segments of ω.

$$\Phi_p(\omega) = \sum_{\substack{u,v\prec\omega \\ u\sim v}} I(u,v) + \sum_{u\prec\omega,\psi\in\omega} A(u,\psi) \tag{3}$$

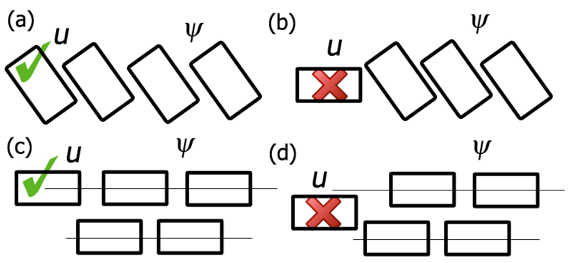


Figure 6: Favored ($\surd$) and penalized ($\times$) sub-configurations within a traffic segm.

where $I(u,v)$ penalizes any overlapping rectangles within the ω configuration:

$$I(u,v)=\frac{\text{Area}\{R_u\cap R_v\}}{\text{Area}\{R_u\cup R_v\}}$$

.

To measure if a vehicle u is appropriately arranged with respect to a traffic segment ψ, we define an alignment distance measure $d_\psi(u)\in[0,1]$ which is the average of two terms: *firstly*, the normalized angle difference between u and the mean angle within ψ (see Fig. 6(a)-(b)), *secondly*, with using RANSAC, we fit one or a couple of parallel lines to the object centers within ψ, and calculate the normalized distance of the center of u from the closest line (Fig. 6(c)-(d)). For prescribing spatially connected traffic segments, we use a constant high difference factor, if u has no neighbors within ψ w.r.t. relation $\sim$. Thus we derive a modified distance:

$$\hat{d}_\psi(u)=\begin{cases}1 & \text{if } \nexists v\in\psi\backslash\{u\}: u\sim v\\ d_\psi(u) & \text{otherwise}\end{cases}$$

We define the $A(u,\psi)$ arrangement term of (3) in the following way. We slightly penalize vehicle groups which only contain a single vehicle: with a small $0<c\ll 1$ constant $A(u,\psi)=c$ iff $\psi=\{u\}$. Otherwise, large $\hat{d}_\psi(u)$ is *penalized* if $u\in\psi$; and *favored* if $u\notin\psi$:

$$A(u,\psi)=\mathbf{1}_{u\in\psi}\cdot\hat{d}_\psi(u)+\mathbf{1}_{u\notin\psi}\cdot(1-\hat{d}_\psi(u))$$

where $\mathbf{1}_E\in\{0,1\}$ is an indicator function of event E.

4 Optimization

To estimate the optimal object configuration, we have proposed a two-level modification of the Multiple Birth and Death Algorithm [1], as follows:

Initialization: start with empty population $\omega=\emptyset$, set the birth rate b_0, initialize the inverse temperature parameter $\beta=\beta_0$ and the discretization step $\delta=\delta_0$.

Main program: alternate the following three steps:

• *Birth step*: Visit all pixels on the image lattice S one after another. At each pixel s, with probability δb_0, generate a new object u with center s and random e_L, e_l and θ parameters. For each new object u, with a probability $p_u^0 = \mathbf{1}_{\omega=\emptyset} + \mathbf{1}_{\omega\neq\emptyset} \cdot \min_{\psi_j\in\omega} \hat{d}_{\psi_j}(u)$, generate a new ψ empty traffic segment, add u to ψ and ψ to ω. Otherwise, add u to an existing traffic segment $\psi_i \in \omega$ with a prob. $p_u^i = (1 - \hat{d}_{\psi_i}(u))/\sum_{\psi_j\in\omega}(1 - \hat{d}_{\psi_j}(u))$.

• *Death step*: Consider the actual configuration of all objects within ω and sort it by decreasing values depending on $\varphi_d(u) + A(u,\psi)\big|_{u\in\psi}$. For each object u taken in this order, compute $\Delta\Phi_\omega(u) = \Phi_\mathcal{D}(\omega/\{u\}) - \Phi_\mathcal{D}(\omega)$, derive the *death rate* $d_\omega(u)$ as

$$d_\omega(u) = \Gamma(\Delta\Phi_\omega(u)) = \frac{\delta \exp(-\beta \cdot \Delta\Phi_\omega(u))}{1 + \delta \exp(-\beta \cdot \Delta\Phi_\omega(u))},$$

and delete object u with probability $d_\omega(u)$. Remove empty traffic segments from ω, if they appear.

• *Group re-arrangement*: Propose randomly group merge, group split and vehicle re-clustering moves. For each proposed move $\mathbf{M}$, calculate the corresponding energy cost $\Delta\Phi_\omega^\mathbf{M}$, and apply the move with a probability $\Gamma(\Delta\Phi_\omega^\mathbf{M})$, similarly to the case in the death step.

Convergence test: if the process has not converged yet, increase β and decrease δ with a geometric scheme, and go back to the birth step.

Convergence test: if the process has not converged yet, increase the inverse temperature β and decrease the discretization step δ with a geometric scheme, and go back to the birth step.

5 Evaluation

We evaluated our method in four aerial LIDAR data sets (provided by Astrium GEO-Inf. Services - Hungary), which are captured above crowded urban areas and contain in aggregate 471 vehicles. The parameters of the method were set based on a limited number of training samples, similarly to [1]. For accurate Ground Truth (GT) generation, we have developed an accessory program with graphical user interface, which enables us to manually create and edit a GT configuration of rectangles. We have performed quantitative evaluation both at object and at pixel levels. At object level, we have measured how many vehicles are correctly or incorrectly detected in the different test sets, by counting the Missing Objects (MO), and the Falsely detected Objects (FO). These values are compared to the Number of real Vehicles (NV), and the F-rate of the detection (harmonic mean of precision and recall) is also calculated [1]. At pixel level, we compared the vehicle silhouette mask to the GT mask, and calculated the F-rate of the match [1]. We have also measured the correct Group Classification Rate (GR, %) among the true positive samples, considering GT classification of human observers.

5.1 Reference Methods

For comparison, we have selected two algorithms. The first is a *bottom-up* grid-cell-based algorithm from [3], called later as *DEM-PCA*, which consists of three consecutive steps: (1) Height map (or *D*igital *E*levation *M*odel) generation by ground projection of the elevation values in the LiDAR point

Table 2: Obj. and pix. level F-rates (in %) by the DP [3], hX [4] and the proposed L^{2}MPP (^{2}M) methods, and the Group Classification Rate (GR) of the L^{2}MPP model.

Set	NV*	Object level %			Pixel level %			GR
		DP	hX	**^{2}M**	DP	hX	**^{2}M**	**^{2}M**
#1	78	78	68	**96**	64	46	**89**	**94**
#2	91	90	93	**98**	77	77	**88**	**93**
#3	132	70	74	**83**	61	46	**66**	**86**
#4	170	85	87	**89**	77	76	**64**	**92**
All	471	83	82	**91**	70	61	**80**	**91**

*NV = Number of real Vehicles in the test set

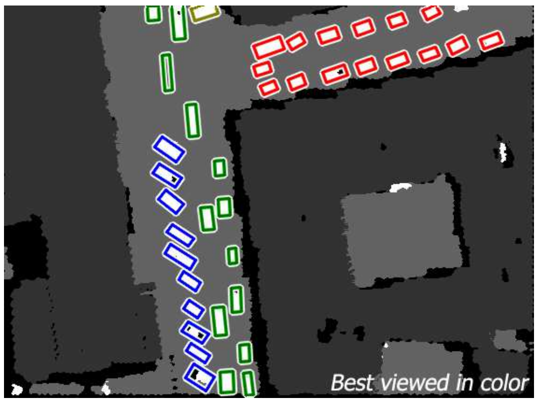


Figure 7: Detection result with four clusters. Vehicles of different segments are displayed with different colors, background is interpolated for visualization.

cloud, and missing data interpolation. (2) Vehicle region detection by thresholding the height map followed by morphological connected component extraction. (3) Rectangle fitting to the detected vehicle blobs by *P*rincipal *C*omponent *A*nalysis.

The second is a recent state of the art method [4], which uses h-maXima (hX) transform followed by watershed segmentation. Some qualitative results are shown in Fig. 7 and 8 (best viewed in color), and the quantitative evaluation is provided in Table 2. Since the reference methods do not deal with vehicle grouping, only the car detection rates are compared: the proposed L^{2}MPP model surpasses the references both at object and at pixel levels.

6 Model extension for terrestrial LiDAR data

The previously discussed model can be extended in order to use for vehicle detection in terrestial LiDAR data (see figure 9). The terrestial data provided by the *Velodyne HDL-64E* LiDAR sensor.

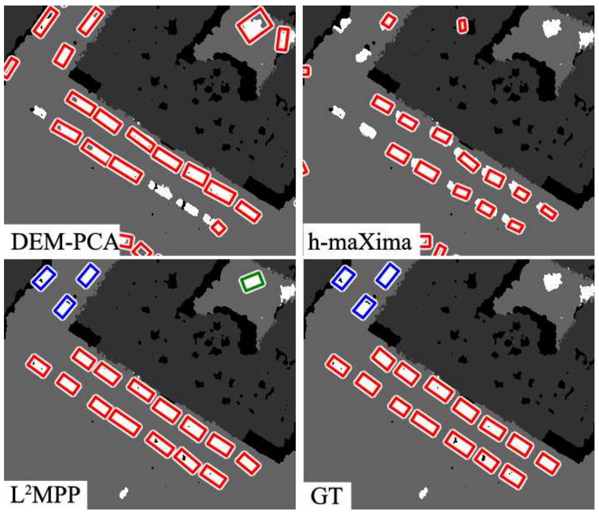


Figure 8: Method comparison on a sample

To achieve this goal we developed a method to preprocess and segment urban scenes in terrestial LiDAR point clouds. The segmented classes are the followings: *road surface*, *short street objects (such as cars and people)*, *Wall and tall static objects (such as lamps posts, traffic lights)*. This classification is based on local point properties. Using some statistical descriptors, we segment the data into one of these semantic classes which later can be used together or separately for various tasks [15]. In many cases, the old data-dependent energy term not sufficient enough to complete vehicle detection in terrestial point clouds due to data occlusion and shape deformation. For this reason hereby we present two new data-dependent energy terms to achieve good detection results:

- The *Unlabelled data allowance* feature $f^{\mathrm{mi}}(u)$ expresses that we exept small proportion of the unlabelled pixels besides vehicle pixels $\mathrm{vehicle}$ within R_u:

  $$f^{\mathrm{ve}}(u) = \frac{1}{\sigma|R_u|} \sum_{s\in R_u} \mathbf{1}\left\{\nu(s) = \mathrm{unlabelled}\right\},$$

  where $|R_u|$ denotes the cardinality of R_u, σ is a proportion coefficient of the unlabelled data (we used $\sigma = 0.3$ here) and $\mathbf{1}\left\{.\right\}$ marks an indicator function: $\mathbf{1}\{\mathrm{true}\} = 1$, $\mathbf{1}\{\mathrm{false}\} = 0$.

- *elevation* feature provides additional evidence for image parts containing elevation values within a certain range

  $$f^{\mathrm{el}}(u) = \frac{1}{|R_u|} \sum_{s\in R_u} \mathbf{1}\{g(s) > T_l \ \wedge g(s) < T_u\}$$

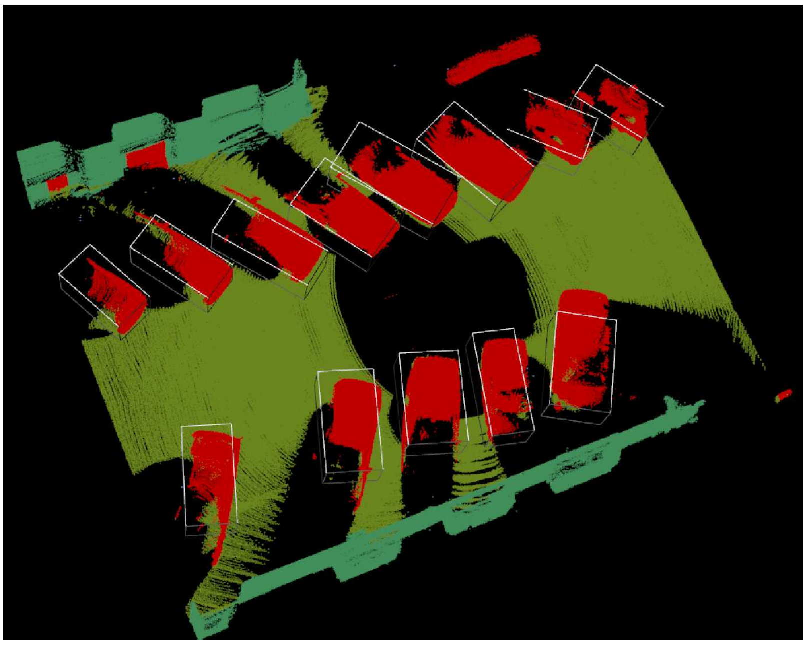


Figure 9: Vehicle detection result on terrestial Velodyne data

where T_l is a lower and T_u is an upper elevation thresold.

7 Conclusions

This paper has proposed a novel Two-Level MPP model for joint extraction of vehicles and traffic segments in aerial and terrestrial laser point cloud data. The efficiency of the approach has been tested with real-world LIDAR measurements, and its advantages versus two reference methods have been demonstrated. Note that in the proposed model, the vehicles are grouped based on similar orientation, but we have experienced that the method can deal with car groups on slightly curved roads as well. As future work, we plan to extend the prior terms of our method to handle more complex vehicle arrangement patterns such as strongly curved exit ramps or roundabouts.

References

[1] C. Benedek, X. Descombes, and J. Zerubia. Building development monitoring in multitemporal remotely sensed image pairs with stochastic birth-death dynamics. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(1):33–50, 2012.

[2] A. Börcs and C. Benedek. A marked point process model for vehicle detection in aerial lidar point clouds. In *ISPRS Annals of Photogrammetry, Remote Sensing and the Spatial Information Sciences*, 2012.

[3] Á. Rakusz, T. Lovas, and Á. Barsi. Lidar-based vehicle segmentation. *International Archives of Photogrammetry and Remote Sensing*, XXXV(2):156–159, 2004.

[4] W. Yao, S. Hinz, and U. Stilla. Automatic vehicle extraction from airborne lidar data of urban areas aided by geodesic morphology. *Pattern Recogn. Letters*, 31(10):1100 – 1108, 2010.

[5] W. Yao, S. Hinz, and U. Stilla. Extraction and motion estimation of vehicles in single-pass airborne lidar data towards urban traffic analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66:260–271, 2011.

[6] S. Tuermer, J. Leitloff, P. Reinartz, and U. Stilla. Automatic vehicle detection in aerial image sequences of urban areas using 3D HoG features. In *ISPRS Photogrammetric Computer Vision and Image Analysis*, page B:50, Paris, France, 2010.

[7] W. Yao and U. Stilla. Comparison of two methods for vehicle extraction from airborne lidar data toward motion analysis. *IEEE Geoscience and Remote Sensing Letters*, 8(4):607–611, 2011.

[8] B. Yang, P. Sharma, and R. Nevatia. Vehicle detection from low quality aerial LIDAR data. In *IEEE Workshop on Applications of Computer Vision (WACV)*, pages 541 –548, 2011.

[9] C.K. Toth and D. Grejner-Brzezinska. Extracting dynamic spatial data from airborne imaging sensors to support traffic flow estimation. *ISPRS Journal of Photogrammetry and Remote Sensing*, 61(3-4):137 – 148, 2006.

[10] X. Descombes, R. Minlos, and E. Zhizhina. Object extraction using a stochastic birth-and-death dynamics in continuum. *J. Mathematical Imaging and Vision*, 33:347–359, 2009.

[11] M. Y. Yang and W. Foerstner. Plane Detection in Point Cloud Data. Technical Report TR-IGG-P-2010-01, Department of Photogrammetry, University of Bonn, 2010.

[12] F. Lafarge and C. Mallet. Creating large-scale city models from 3D-point clouds: A robust approach with hybrid representation. *Int. J. of Computer Vision*, 2012. in press.

[13] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[14] F. Chatelain, X. Descombes, and J. Zerubia. Parameter estimation for marked point processes. application to object extraction from remote sensing images. In *Energy Minimization Methods in Comp. Vision and Pattern Recogn.*, Bonn, Germany, August 2009.

[15] O. Józsa and C. Benedek. Reconstruction of 3D Urban Scenes Using a Moving Lidar Sensor. Technical Report i4D-1, MTA SZTAKI, 2013.