A REPRESENTATION OF MLS DATA AS A BASIS FOR TERRAIN NAVIGABILITY ANALYSIS AND SENSOR DEPLOYMENT PLANNING

Joachim Gehrung[1,2,*], Marcus Hebel[1], Michael Arens[1], Uwe Stilla[2]

[1] Fraunhofer Institute of Optronics, System Technologies and Image Exploitation IOSB,
76275 Ettlingen, Germany - (joachim.gehrung, marcus.hebel, michael.arens)@iosb.fraunhofer.de
[2] Photogrammetry and Remote Sensing, Technical University of Munich (TUM), Germany - stilla@tum.de

ICWG II/III: Pattern Analysis in Remote Sensing

KEY WORDS: Mobile Laser Scanning, Spatial Analysis, Occupancy Grid, Density Function, Heatmap

ABSTRACT:

Recording an ever-changing urban environment in a structured manner requires sensor deployment planning. In case of mobile sensor platforms, this also includes verifying the terrain navigability. Solving both tasks would usually require different application-specific data structures and tools. In this work, we propose a theoretical framework that provides a uniform representation for spatial information as well as the tools required to combine, manipulate and visualize it. We provide an efficient implementation of the framework utilizing octree-based evidence grids. Our approach can be used to solve complex tasks by combining simple spatial information sources, which we demonstrate by providing simple solutions to the aforementioned applications. Despite the use of a volumetric approach, our runtimes are within the range of minutes.

1. INTRODUCTION

The rapid technological development in the last decades has led to the emergence of a variety of geospatial services. The ever-increasing digitization has contributed its part to make them available at any time, anywhere. As a result of these newly created business areas and the low technological entry barriers, it is appealing even for medium and small companies to carry out large-scale data collections. This is possible in a cheap and efficient way, since powerful sensor systems can already be implemented with inexpensive consumer hardware. Besides stationary sensor systems, this also allows the realization of mobile sensor platforms such as person-carried sensor backpacks and measurement vehicles.

The structured acquisition of the environment presents a number of challenges, since recording the entire space of interest requires considering a variety of constraints. In the case of a measuring vehicle this would involve to record all streets and places as completely as possible while keeping the redundancy as low as possible, as this is a waste of measurement time, storage space and later on processing power. It may be required to visit certain points of interest as well as to avoid others, for example due to mandatory privacy regulations. It may also be necessary to exclude areas, which are difficult to reach or leave by the measurement vehicle.

Solving the structured acquisition problem both with respect to sensor deployment and surface navigability requires the fusion of different kinds of spatial information. In this work we propose a theoretical framework that allows to derive a *task specific space of interest* from a multitude of spatial information sources. We define a mathematical representation for spatial data and present the tools required to analyze, manipulate and visualize it. Furthermore, we describe how it can be realized in an efficient way by utilizing octree-based evidence grids. We demonstrate the flexibility of our approach on the freely available **MLS2 - TUM City Campus** dataset.

*Corresponding author

2. RELATED WORK

2.1 Spatial Analysis of Urban Areas

The term *spatial analysis* summarizes a multitude of activities with a common goal of information gain at global, regional to local scales by utilizing different kinds of sensors. In the context of urban environments, a gain in knowledge about the characteristics or the change over time of the area under consideration is derived from the available data. Information like that can be utilized e.g. to support urban planning or development tasks. Herold et al. (2003) use aerial photography and satellite data for the analysis of urban growth on the example of Santa Barbara, California. Based on satellite image time series, Myeong et al. (2006) quantify the carbon storage, distribution and change of urban forests. As summarized in a review by Yan et al. (2015), airborne LiDAR can be used for urban land cover classification. Furthermore, there are approaches that deal with multitemporal data, especially in the context of change detection. Examples include the works of Hebel et al. (2013) and Aijazi et al. (2013), which utilize airborne LiDAR data and mobile LiDAR data, respectively. To the best of the authors' knowledge, there is no work that attempts to store and process spatial information in a generic way such as the one proposed in this work.

2.2 Occupancy Grids

Occupancy grids are able to represent three-dimensional environments in a memory efficient way, even if those contain complex geometries. Moravec et al. (1985) proposed to use a two-dimensional occupancy grid in order to map an indoor environment with ultrasonic sensors. Each grid cell along a plane at sensor level contains a probability in order to describe the degree of occupancy of the encompassed space.

The approach produces a quite accurate representation of the environment, considering the wide angled conical shape of the ultrasonic beam. An approach utilizing an octree to store binary occupancy information in order to represent arbitrary geometries has been proposed by Meagher, (1982). The concept has later been enhanced to comprise probabilistic information by Payeur et al., (1997).

Hornung et al. (2013) extended this approach with a lossless compression strategy based on probability clamping, which allowed for a fast adaption to a changing environment. Due to the versatile usability and an open-source implementation, this approach has gained huge popularity under the name *OctoMap*. Based on the theoretical foundation of this work, Gehrung et al. (2016) proposed a concept for occupancy representation on a global scale. In addition, an algorithm for iterative refinement has been proposed in order to reduce discretization artifacts inherent to the technology as well as to speed up the generation of the occupancy octree (Gehrung et al., 2018).

3. DENSITY FUNCTIONS FOR INFORMATION REPRESENTATION

3.1 Motivation

Information can be classified by the nature of values that are assigned to a variable. For *nominal* information these values have no inherent order, whereas for *ordinal* information an order exists, but the distance cannot be quantified. If both an order of individual values and a distance measure for the values of a variable exist, information is denoted to be of *cardinal* nature. Examples for nominal, ordinal and cardinal information would be colors, letters in an alphabet and regular numbers. Whenever an urban environment is recorded, either by a sensor system or a human being, the resulting information falls into one of the above-mentioned categories.

In order to transform a nominal or ordinal variable into a cardinal one, its actual value is compared to a desired one. If both values match, 1 is assigned to the new cardinal variable, otherwise 0. Once all information has been transformed to be of cardinal nature, otherwise very different information can be combined and processed numerically using a distinct set of algebraic operations.

Combining nominal information such as the visibility from a given point and cardinal information like proximity to a surface can, for example, be condensed into new information that represents the space that can be safely navigated by an UAV while the latter has an uninterrupted line of sight to the area of interest. Such information can than directly be used for tasks such as path planning.

3.2 Formal Definition

Information structured in the way mentioned above can be seen as the value (or intensity) of this information that is in relation to a distinct point in space. Since this is a function of the location, we decided to refer to it as a *density function* or in short, a *density*. The term is inspired by the real-valued density functions used in the mathematical branches of Stochastics. Therefore, we formally define a density function as

$$d\colon \mathbb{R}^3 \to \mathbb{R}, d(x) = i$$

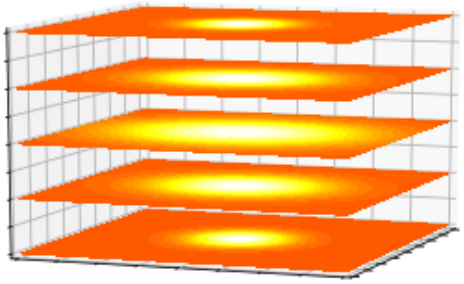


Figure 1. Example of a three-dimensional density function, an information whose intensity is location-specific, visualized by color-coded volume slices.

where $x \in \mathbb{R}^3$ is a location in three-dimensional space in an arbitrary reference frame. The density, or intensity of the information for a given position x is denoted by i. In order to prevent one source of information from superimposing another, it is defined that $i \in [0, 1]$. An illustration for a location-sensitive information with variable intensity can be seen in Figure 1. A selection of density functions that are beneficial to the structured acquisition of the environment are proposed in Section 4.

3.3 Representation

In the context of this work, a density function is always defined over an observed space that has fixed boundaries. Therefore, the latter can also be interpreted as a volume. Assuming that regions with the same density can be grouped together, a good trade-off between computational resources and accuracy is an octree with dynamic resolution. The dynamic aspect is important because it allows the non-discrete function to be resolved as accurately as possible, but as roughly as necessary. Following a well established practice regarding evidence grids used in works such as the one of Hornung et al. (2013), octree cells are considered to be independent of each other. The octree is generated using iterative refinement, such as presented by Gehrung et al. (2018), since it is less resource intensive than normal ray casting.

3.4 Algebraic Operations on Densities

In order to derive new information from existing density functions, a set of basic **algebraic operations** is required. Operations applied to density functions are defined as being point-wise operations. This means that the operation applied to intertwine two densities corresponds to the same operation applied to the intensities from both densities:

$$(d_a \circ d_b)(x) := d_a(x) \circ d_b(x) \tag{1}$$

Therefore combining two non-discrete density functions reduces to applying this combination to a scalar for each point $x \in \boldsymbol{X}$.

3.4.1 Inversion In order to inverse the semantic meaning of a density function, the *invert* operation has been defined as

$$(\overline{d_a})(x) := 1.0 - d_a(x), \tag{2}$$

where the operation corresponds to the inversion of the value at the midpoint of the interval.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-2-W7-39-2019 | © Authors 2019. CC BY 4.0 License.

3.4.2 Sum and Difference The sum and difference between two densities is defined as

$$(d_a \oplus d_b)(x) := d_a(x) + d_b(x) \tag{3}$$

and

$$(d_a \ominus d_b)(x) := d_a(x) - d_b(x) \tag{4}$$

Both operations can lead to results which are outside of the interval of $[0, 1]$, so normalization is required.

3.4.3 Product Two different kinds of product have been specified. The product between density functions is specified as

$$(d_a \otimes d_b)(x) := d_a(x) \cdot d_b(x). \tag{5}$$

Since both values are within the interval of $[0, 1]$, the result also is. The second product is defined between a density function and a scalar s in order to allow weighting:

$$(s \otimes d_a)(x) := s \cdot d_a(x). \tag{6}$$

The result requires normalization if the scalar is less than zero or larger than one.

3.4.4 Maximum In order to create a composite of two densities, the maximum operation is specified. For every location in both densities, the maximum value from either density a or b is chosen. This is specified as

$$max(d_a, d_b)(x) := max\Big(d_a(x), d_b(x)\Big). \tag{7}$$

The result is already normalized.

3.4.5 Normalization Since simple clamping of the value would lead to a potential loss of information, a scaling technique similar to histogram equalization has been chosen in order to normalize density functions.

$$i_{norm} = \frac{i - i_{min}}{i_{max} - i_{min}} \tag{8}$$

The spread between the lowest i_{min} and highest value i_{max} in a density function is determined and used to scale each value i in order to get the normalized value i_{norm}.

3.4.6 Realization based on Octrees Since densities are represented as octrees with dynamic resolution, applying an operation to all points of a density functions reduces to the much more computational effective process of intertwining voxels. However, the structures of two different octrees are rarely identical. Utilizing an approach described by Gehrung et al. (2019), an octree with a structure that represents a combination of both octrees involved in the operation is generated. The result is then stored within this octree. During the operation, whenever a voxel has no counterpart in the other octree, a voxel with a higher resolution is chosen as a substitute.

4. EXEMPLARY DENSITY FUNCTIONS

4.1 Derivation from Occupancy Information

All densities in this work are derived from an occupancy grid loosely based on the theoretical foundation proposed by Hornung et al., (2013), either by analyzing it in a specific way or by applying a form of convolution. It uses an octree with dynamic resolution in order to represent the degree of occupancy for each observed location. Due to the strong similarities between the representation of occupancy and density functions, *occupancy* can also be interpreted as a density and therefore integrates seamlessly into our framework. The following sections demonstrate how densities can be derived from occupancy information.

4.2 Navigable Surfaces

The *navigable surfaces density* d_s that describes planar horizontal surfaces defines where a mobile sensor platform can move. This may not only include streets, but also other surfaces such as parking spaces and undeveloped areas. The density is derived from the *occupancy density* d_o by means of convolution. The structure of both octrees representing d_o and d_s is defined to be identical. The density value of each voxel x of the *navigable surfaces* density is determined as follows:

$$d_s(x) = \frac{\sum\limits_{n \in M(x)} d_o(n)}{|M(x)|} - \frac{\sum\limits_{n \in N(x)} d_o(n)}{|N(x)|} \tag{9}$$

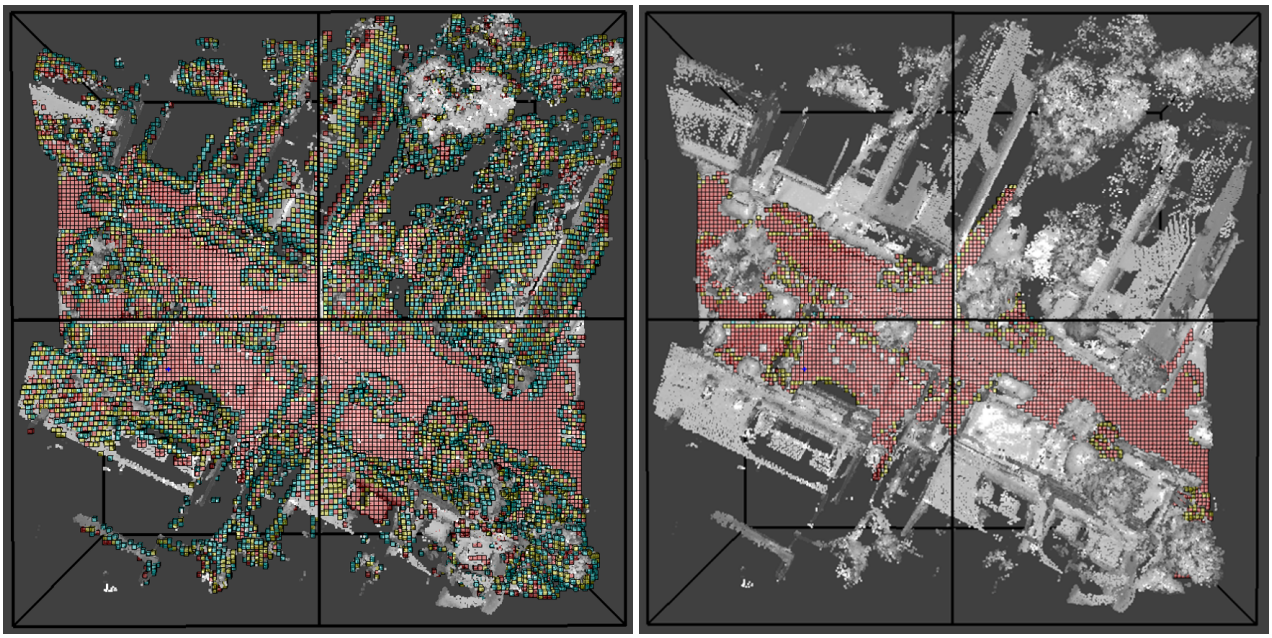
The neighbors M along all four cardinal directions in the occupancy density are determined. In terms of octree neighborhood search, it does not matter whether or not a neighbor is of smaller, equal or bigger size. The occupancy values d_o of these neighbors are added and normalized by the number of neighbors. The density value of the top neighbors N is determined and subtracted from the result in order to penalize surface voxels with occupied space on top. The value of each voxel is already normalized.

The resulting density contains all horizontal planar surfaces, albeit not all of interest, since also floors, ceilings and roofs of building are part of it. An example scene with these surfaces can be seen in Figure 2(a). Based on these, a flood-filling algorithm is used to determine which surfaces are navigable. In order to select the correct subset of interconnected planar surfaces, a starting point is required. The density obtained after applying the flood-filling algorithm can be seen in Figure 2(b).

4.3 Proximity and Distance

The *proximity density* d_p function reflects how close a point in space is to a vertical surface. Voxels are created by sampling space in a three-dimensional grid. The value for a given voxel is determined by sampling a cylindrical space around its position (cf. Figure 3(a)). For each sample, the corresponding value is extracted from the *occupancy density*. If it is higher than the interval midpoint, it is added to the voxel, whereas the sum is afterwards divided by the number of all samples in order to normalize it. An example can be seen in Figure 3(e).

The *proximity density* is required in order to derive the *distance density* d_d, which is simply the inversion of the former one and therefore defined as $d_d = \overline{d_p}$ (cf. Figure 3(b)). Instead



(a) (b)

Figure 2. A street scene demonstrating the navigable surfaces density before (left) and after (right) applying flood filling in order to determine the relevant subset of navigable surfaces. Navigability is encoded in ascending order from blue (low) to green (medium) all the way to red (high).

of highlighting locations close to vertical surfaces, it highlights locations far away from these, as can be seen in Figure 3(f). This operation is used for collision avoidance, to identify areas that are save to navigate even by larger vehicles.

4.4 Invisibility and Visibility

Applications related to sensor management require to determine all areas that can be seen from a given location. Therefore, space is sampled along a three-dimensional grid and ray casting is applied between the given location and the center of each grid voxel. Since ray casting is quite slow, the actual implementation is based on a modified version of the *iterative refinement algorithm* proposed by Gehrung et al. (2018). Starting from the root node of the *occupancy density* octree, only voxels that are intersected by the line-of-sight are further refined. Once a voxel has no children and is considered to be solid, the algorithm is terminated and the voxel at the end of the line-of-sight is marked as invisible.

The result is the *invisibility density* d_i, from which the actual *visibility density* d_v can be derived by inverting the former one using the operation $d_v = \overline{d_i}$. This procedure is computationally cheaper than determining the visibility density directly, since it requires a lot less voxel checks. Figures illustrating both densities can be found in 3(c) and 3(d), examples are shown in Figure 3(g) and 3(h).

5. DENSITY VISUALIZATION USING HEATMAPS

5.1 Heatmaps

Visualizing a density in the same way as shown in Figure 1 may be appropriate to convey the general concept of densities. However, it is not intuitive enough for interpretation by a human observer. Therefore, we suggest a two-dimensional representation that is colloquially called a *heatmap*. It is basically a projection from $\mathbb{R}^3$ to $\mathbb{R}^2$ where values are added along the z-axis of the density. The result can also be considered a density in the sense of Section 3, therefore the same algebraic operations can be applied. An example can be seen in Figure 7.

5.2 Density Projection using Drill Downs

Generating a heatmap from a density is straightforward. It requires iteration over a two-dimensional grid with the intended resolution of the heatmap. For each cell, a *drill down* through the octree representing the density is required (cf. Figure 4). A drill down lists all voxels of an octree along a straight line which pierces the xy-plane in a perpendicular way at a given point. The point is the center point of each heatmap cell. In order to implement the drill down, the octree is traversed from the root node to its leaf nodes. Inner nodes containing the point are further traversed whereas leaf nodes that fulfill the same condition are added to a result list. The values of all nodes within the result list are added, the sum is assigned as the cell value. After calculating each cell value, normalization as described in Section 3.4.5 is applied to the heatmap.

5.3 Interpretation of Heatmaps

The information shown in a heatmap is color-coded. Blue represents low, green medium and red high values. In addition, the point cloud is projected into the image. The value of a heatmap corresponds to the accumulated value along the z-axis of a density function. For the *visibility density* function, a high value means that most of the spots along the axis are visible, whereas a low value means that most of all spots along the axis are invisible. Analogously, this also applies to other density functions.

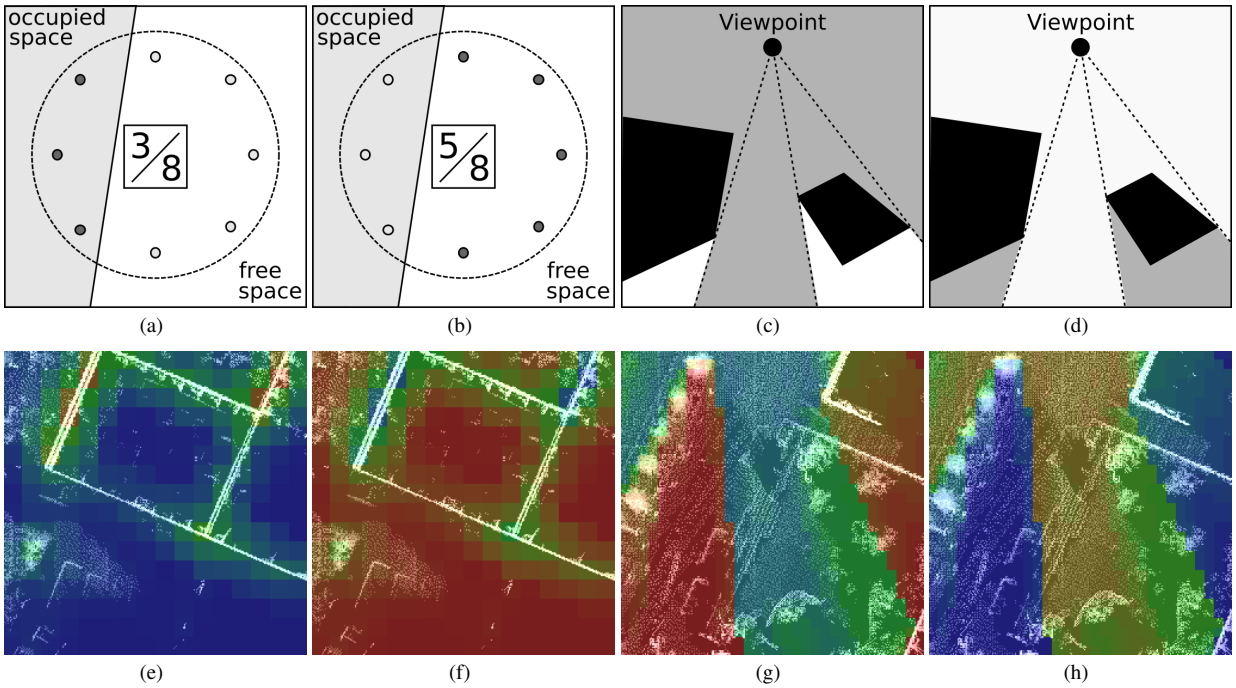


Figure 3. An illustration of each density (top row) and the corresponding heatmaps (bottom row). The densities and heatmaps are (a,e) *proximity*, (b,f) *distance*, (c,g) *invisibility* and (d,h) *visibility*.

6. EXPERIMENTS

6.1 Dataset

The case study is based on the **MLS2 - TUM City Campus**[1] which has been recorded by the Fraunhofer Institute of Optronics, System Technologies and Image Exploitation (IOSB). It is an extension of the **MLS1 dataset** from 2016 by Gehrung et al. (2017), which has also been enriched with class labels by Sun et al. (2018). Both datasets as well as the labels are publicly available under a Creative Commons License. An overview over the one used in this work is given in Figure 5.

Both datasets have been recorded using the measurement vehicle MODISSA (Borgmann et al., 2018). Two Velodyne HDL-64E LiDAR sensors mounted at an angle of 25 ° on the vehicle front roof have been used to record the vehicle's environment as well as building facades. Eight cameras mounted at the roof corners and an IR-camera on a pan-tilt-unit can be utilized to texture the LiDAR measurements. The dataset includes a sequence of individual scans, with each scan covering approximately 0.1 seconds, which roughly corresponds to a full rotation of the sensor head. Each scan is georeferenced based on navigation data from an Applanix POS LV navigation system which utilizes two GNSS antennas, an inertial measuring unit and a distance measuring indicator. The navigation data has been postprocessed to increase accuracy, the dataset itself has been fine-registered utilizing techniques based on graph-based SLAM.

6.2 Scenario A: Sensor Deployment

In the first scenario, four *points of interest (POI)* within the inner yard of the TUM city campus are considered (cf. Figure 7(a)). In the first part of the scenario, a sensor location which offers a clear view of all four POIs is determined. The simulated sensor is assumed to have a field-of-view of 360 °. In the second part, locations for two sensors are chosen in a way that they can perceive a subset consisting of the POIs 1-3. The decision to use a subset instead of all points of interest was made in order to be able to better illustrate the process.

[1] http://s.fhg.de/mls2

6.3 Scenario B: Surface Navigability

In the second scenario, the ability of the system to determine the navigability of terrain is be demonstrated. Therefore, the *navigable surfaces density* is determined and multiplied with the *distance density* in order to describe the space that can be safely navigated by a mobile sensor platform.

6.4 Runtime

To conclude the case study, the runtimes for all steps in the process chain are discussed.

7. RESULTS AND DISCUSSION

7.1 Results of the Case Study

Scenario A In order to determine the sensor location, the fields-of-view of all four points of interest are multiplied with each other, which corresponds to an **and**-operation. The resulting density that contains the common field-of-view can be seen in Figure 7(b). A resolution of 1 m has been chosen. The area close to the location of POI number 2 is highlighted in red, since it is within the field-of-view of all four POIs. On its right side is another area that is also within view of most of the points of interest, but not in the one of POI number 3, since it is shadowed by a row of flag poles. An arbitrary location within the area highlighted in red is chosen as the position of the simulated sensor. Figure 7(c) shows the sensor positions and the visibility of all four points of interest. As expected, all of them are visible from the chosen sensor position. This is not

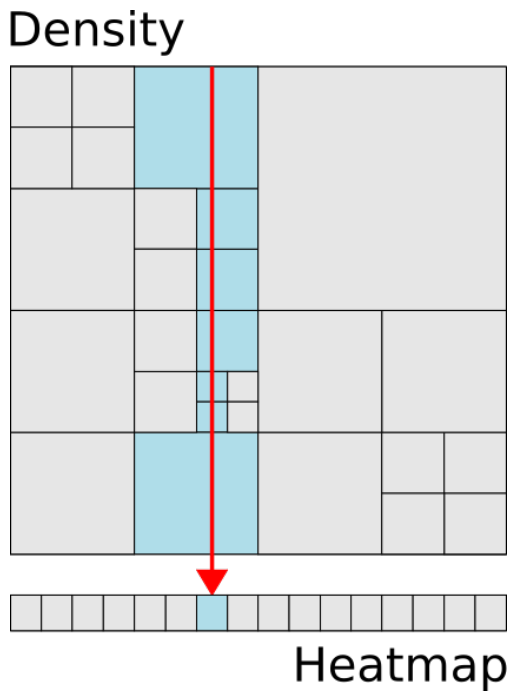


Figure 4. The derivation of a heatmap from a density represented by an octree. A drill down technique is used to determine all octree cells that are projected onto a given heatmap cell.

surprising, since seeing some location A from another location B also means that location B can be seen from location A. However, it underlines the capabilities of this framework, since this information was derived only by combining some very simple spatial information sources using arithmetic operations.

In the second part, a similar approach has been chosen to determine the two sensor locations. Pairs of all points of interest with a direct line of sight are formed. For the subset under consideration, these are 1,2 and 1,3. For each pair, their field-of-view is joined with an **and**-operation by multiplication. The results are joined with an **or**-operation, which in this case is realized by the maximum-operation. The resulting field-of-view can be seen in Figure 7(d). Two arbitrary locations in the area highlighted in red are chosen as sensor positions. Figure 7(e) shows these sensor positions and the visibility of all three points of interest, which are again clearly visible.

In the upper right part of Figures 7(d) it can be seen that despite the rather crude resolution of 1 m, the approach is able to predict the sensor's field-of-view through a window. However, since hardly any geometry of the buildings interior was observed during the recording of the dataset, the visibility calculation inside buildings are not accurate. This may also lead to false positives outside of buildings, if there is another window in the line of sight at the buildings opposite side. The location outside the window may then be marked as visible, while it is not. This is considered a general problem related to observability of the environment using a LiDAR sensor system, but solving it is within the possibilities of the proposed framework, since it is capable of considering information about unseen areas in order to recognize the mentioned situation.

Scenario B. Figure 6(a) shows a heatmap of the *navigable surfaces density*. All locations at least 3-4 m away from a vertical surface are highlighted in red color in Figure 6(b). The

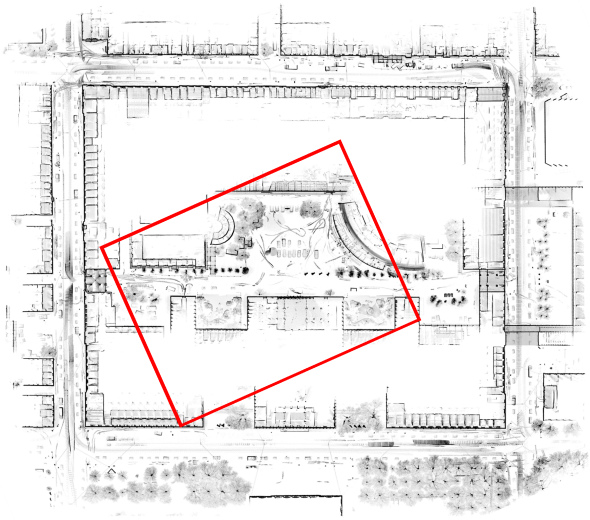


Figure 5. An overview of the MLS2 dataset. The area highlighted in red has been used to conduct the case study.

fusion of both densities can be seen in Figure 6(c). Only parts of the navigable surface that are reasonable far away from an obstacle remain. It can be seen that the area below both rows of trees (along the diagonal, from bottom left to top right) is no longer marked as navigable, as soon as the distance to an obstacle has been considered by the simulation.

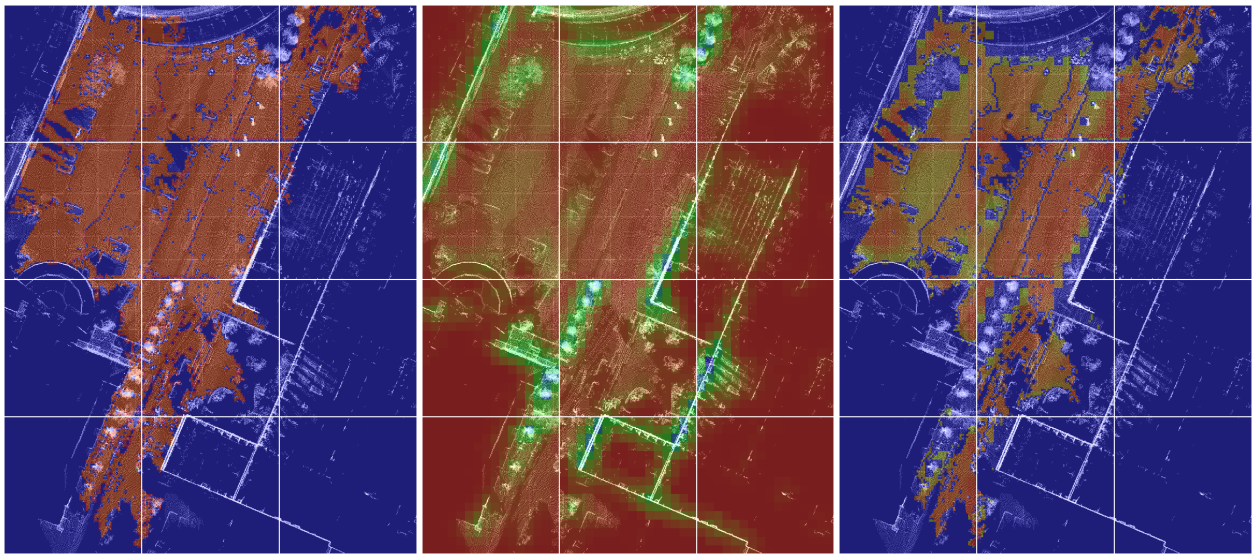
It should be noted, however, that the navigability density has an issue with surfaces rising in a steep angle. As can be seen at the center of the yard, a sudden increase in altitude causes a break in the navigable surface. A similar, related effect can be observed near almost all cars in the center and left part of Figure 6(c). Since the algorithm for surface extraction is designed to handle small changes in height, it is able to follow surfaces rising in a shallow angle. A car as well as other smaller obstacles may be considered to be such a surface, therefore parts of them are marked to be navigable. However, this can be remedied by the fact that the navigability is derived directly from the point cloud, not from an occupancy density.

7.2 Discussion of the Runtime

The dataset has been broken down into 177 3D-tiles, each with an edge length of 32x32x32 m. The distribution of the 2 billion 3D measurements on the tiles required 44 s per tile, the generation of the occupancy density 61 s. Deriving all 26 densities representing intermediate and final results took 93 s per tile, which corresponds to about 3.5 s per tile and density. Generating the 8 heatmaps shown in this paper required 17 s per tile or about 2.1 s per tile and heatmap. All reported results were generated on a machine with 32 gigabyte of RAM, utilizing an Intel Core i7 processor with 3.5 GHz and 12 cores. Most of the algorithms have been parallelized using OpenMP.

8. CONCLUSION AND FUTURE WORK

In this paper we proposed a generic *space of interest* representation for spatial information that can be utilized to solve complex tasks by combining simple spatial information sources. This is made possible by the uniform representation of spatial information that allows the simple combination, manipulation and visualization of the latter. We demonstrated



(a) (b) (c)

Figure 6. The heatmaps for (a) *navigable surfaces* (b) *distance to vertical surfaces* and (c) the resulting *navigable terrain*, which is the product of the two former ones after threshold filtering.

the possibilities of this framework by finding solutions to scenarios from *sensor deployment planning* and *surface navigability analysis*. There may be more specialized solutions to both problems that are more efficient. However, as we have demonstrated, the approach presented here has the clear advantage that it can solve not only one highly specific problem, but a variety of different problems. The method is utilizing a volumetric representation of the world, yet the runtime is low enough that the immediate environment of a LiDAR sensor can be evaluated within a few minutes. Further work will include handling of the uncertainties associated with both measurements and the volumetric representation.

References

Aijazi, A.K., Checchin, P., Trassoudaine, L., 2013. Detecting and Updating Changes in Lidar Point Clouds for Automatic 3D Urban Cartography. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, II-5/W2, 7-12.

Borgmann, B., Schatz, V., Kieritz, H., Scherer-Klöckling, C., Hebel, M., Arens, M., 2018. Data processing and recording using a versatile multi-sensor vehicle. IV-1, 21–28.

Gehrung, J., Hebel, M., Arens, M., Stilla, U., 2016. A Framework for Voxel-based Global Scale Modeling of Urban Environments. *ISPRS International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W1, 45–51.

Gehrung, J., Hebel, M., Arens, M., Stilla, U., 2017. An approach to extract moving objects from MLS data using a volumetric background representation. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-1/W1, 107–114.

Gehrung, J., Hebel, M., Arens, M., Stilla, U., 2018. A voxel-based metadata structure for change detection in point clouds of large-scale urban areas. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2, 97–104.

Gehrung, J., Hebel, M., Arens, M., Stilla, U., 2019. A fast voxel-based indicator for change detection using low resolution octrees. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W5, 357–364.

Hebel, M., Arens, M., Stilla, U., 2013. Change detection in urban areas by object-based analysis and on-the-fly comparison of multi-view ALS data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 86, 52–64.

Herold, M., Goldstein, N.C., Clarke, K.C., 2003. The spatiotemporal form of urban growth: measurement, analysis and modeling. *Remote Sensing of Environment*, 86, 286-302.

Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W., 2013. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 34, 189-206. http://octomap.github.com.

Meagher, D., 1982. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19, 129–147. doi.org/10.1016/0146-664x(82)90104-6.

Myeong, S., Nowak, D.J., Duggin, M.J., 2006. A temporal analysis of urban forest carbon storage using remote sensing. *Remote Sensing of Environment*, 101, 277 - 282.

Payeur, P., Hebert, P., Laurendeau, D., Gosselin, C.M., 1997. Probabilistic octree modeling of a 3D dynamic environment. *Proceedings of the IEEE International Conference on Robotics and Automation*, 2, 1289–1296.

Sun, Z., Xu, Y., Hoegner, L., Stilla, U., 2018. Classification of MLS point clouds in urban scenes using detrended geometric features from supervoxel-based local context. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2, 271–278.

Yan, W.Y., Shaker, A., El-Ashmawy, N., 2015. Urban land cover classification using airborne LiDAR data: A review. *Remote Sensing of Environment*, 158, 295 - 310.

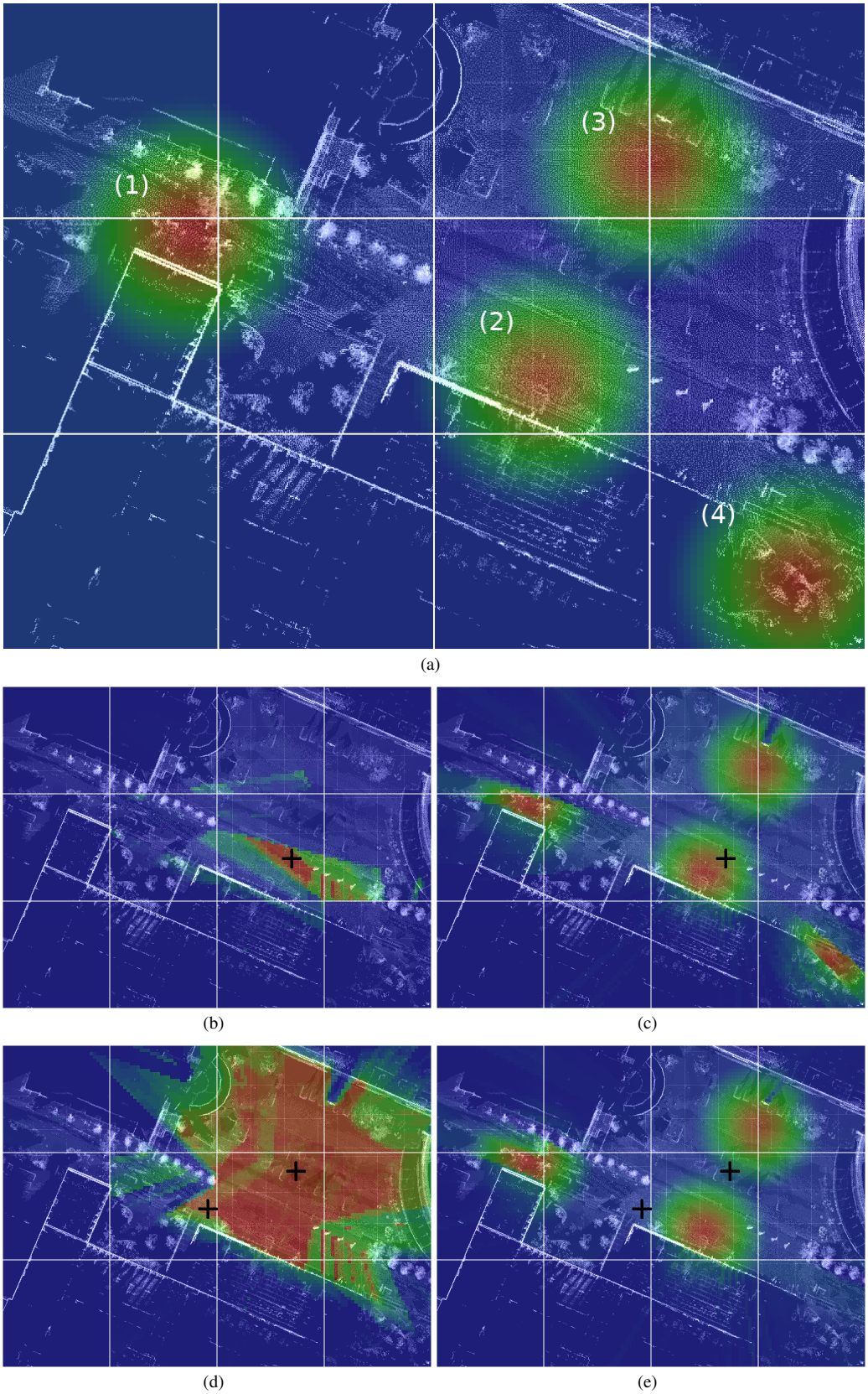


Figure 7. A heatmap showing multiple *points of interests* (a). Applying a multiplication to the fields-of-view of all POIs (b) implies a sensor position (black marker) from which all POIs can be seen (c). In order to deploy two sensors that are able to see only the POIs 1-3, the fields-of-view of the POIs 1,2 and of the POIs 2,3 are multiplied, the maximum-operation is used in order to concatenate both results (d,e).

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-IV-2-W7-39-2019 | © Authors 2019. CC BY 4.0 License.