NEXT-BEST-VIEW METHOD BASED ON CONSECUTIVE EVALUATION OF TOPOLOGICAL RELATIONS

K. O. Dierenbach[a], M. Weinmann[a], B. Jutzi[a]

[a] Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT), Germany
dierenbach.kai@googlemail.com; (martin.weinmann, boris.jutzi)@kit.edu

Commission III, WG III/1

KEY WORDS: LiDAR, Point Cloud, Object Reconstruction, Next-Best-View Problem, Growing Neural Gas

ABSTRACT:

This work describes an iterative algorithm for estimating optimal viewpoints, so called next-best-views (NBVs). The goal is to incrementally construct a topological network from the scene during the consecutive acquisition of several views. Our approach is a hybrid method between a surface-based and a volumetric approach with a continuous model space. Hence, a new scan taken from an optimal position should either cover as much as possible from the unknown object surface in one single scan, or densify the existing data and close possible gaps. Based on the point density, we recover the essential and structural information of a scene based on the Growing Neural Gas (GNG) algorithm. From the created graph representation of topological relations, the density of the point cloud at each network node is estimated by approximating the volume of Voronoi cells. The NBV Finder selects a network node as NBV, which has the lowest point density. Our NBV method is self-terminating when all regions reach a predefined minimum point density or the change of the GNG error is zero. For evaluation, we use a Buddha statue with a rather simple surface geometry but still some concave parts and the Stanford Dragon with a more complex object surface containing occluded and concave parts. We demonstrate that our NBV method outperforms a "naive random" approach relying on uniformly distributed sensor positions in terms of efficiency, i.e. our proposed method reaches a desired minimum point density up to 20% faster with less scans.

1. INTRODUCTION

Acquiring the complete surface of an unknown object with a ranging device (e.g. a range camera or a laser scanner) requires multiple observations taken from different viewpoints around the object. Thereby, the viewpoints of the ranging device are typically selected by the user based on heuristic or empiric knowledge about the scene and/or the data, a process which may be time-consuming and costly. Accordingly, it seems desirable to reduce human effort by automatically selecting appropriate viewpoints, particularly for applications where a larger number of scans is required to adequately cover an object or region of interest.

Given data which have been acquired from one viewpoint or from more viewpoints, an important research topic is addressed by automatically estimating successive viewpoints in a way that the considered scene is appropriately covered and that the number of respective viewpoints is as small as possible. Hence, a new scan taken from an *optimal* position should either cover as much as possible from the unknown object surface in one single scan, or densify the existing data and close possible gaps. This issue has been addressed with view planning methods which are also referred to as *next-best-view* (*NBV*) *methods*. While early NBV methods have already been presented decades ago (Connolly, 1985), more and more attention has been paid to them recently, and respective approaches have been presented for different environments, different tasks and different levels of the desired accuracy. The emerging interest in such techniques is mainly due to technological advancements allowing a faster computation and due to powerful techniques of machine learning. Both of these aspects represent important prerequisites for the development of complex autonomous applications.

Addressing the NBV problem, early investigations focused on turntable-based approaches (Pito, 1999) and on model-based approaches (Scott et al., 2003). However, such approaches are tailored to controlled environments where strong assumptions about either the change in viewpoint or the possibly occurring objects are made. Respective assumptions cannot be made that easily for an autonomous exploration of unknown space (Dornhege and Kleiner, 2013) or for the 3D reconstruction of unknown objects in both indoor and outdoor scenes (Kawashima et al., 2014; Vásquez and Sucar, 2011; Bissmarck et al., 2015). Accordingly, a new generation of model-free approaches for NBV estimation is required, where approaches relying on acquired object surfaces seem to be more promising (Scott et al., 2003), since they are also applicable for unknown environments as well as for objects which exhibit a concave surface geometry.

In this paper, we focus on the use of ranging devices such as range cameras or laser scanners in order to acquire the complete surface of an unknown object within an indoor environment and, given a point-sampled object surface, we intend to find the next-best-view in a data-driven manner. For this purpose, we take into account the point density of the given data. Based on the point density, we recover the essential and structural information of a scene which we call the *topology*. In this context, an undirected graph representing the topology of the considered scene is incrementally constructed by clustering the data in the input space to get the topological relations. For this, the Growing Neural Gas algorithm is applied. Next-best-views are thus derived by consecutively evaluating the density of the point-sampled object surface.

In the following, we first outline related work in Section 2. Subsequently, in Section 3, we explain our methodology in detail. For testing our methodology, we build a simulation framework which is described in Section 4 and allows to generate synthetic, but near-realistic scenarios. For two exemplary scenarios, we provide the experimental results derived by applying our methodology in Section 5. These results are finally discussed in Section 6.

2. RELATED WORK

While the NBV problem has also been addressed for applications focusing on 3D reconstruction based on camera images (Wenhardt et al., 2006; Dunn and Frahm, 2009; Alsadik et al., 2012), we focus on the use of a ranging device such as a range camera or a laser scanner for data acquisition. Accordingly, we provide a glance view on NBV approaches proposed for data acquisition via ranging devices in the following paragraphs.

A rather intuitive way for acquiring the complete surface of an unknown object with a ranging device consists in uniformly placing enough viewpoints around the object of interest in order to reconstruct its surface. However, this is inefficient and also ineffective when dealing with complex object geometries. Instead, it is more efficient to use a method that exploits the available data to make a direct decision about where to place a new viewpoint.

Involving the already available data, a respective approach focusing on the completion of a 3D model of an unknown object has for instance been presented in (Kriegel et al., 2011), where the first step consists in creating a mesh from a real-time data stream. Subsequently, this surface information is used to detect boundaries in the surface which, in turn, serve for estimating a quadratic patch. Finally, possible viewpoints are derived by looking perpendicular to the estimated patch at a certain distance. Furthermore, the completion of a 3D model of an unknown object has been addressed with a NBV approach based on the measure of information gain as a metric for evaluating potential next-view candidates in terms of the respective reduction of uncertainty (Krainin et al., 2011). By maximizing the information gain of a new viewpoint relative to the current reconstruction, the next view to be selected for a range camera looks at the most uncertain surface areas of the considered object.

The measure of information gain is also exploited in the approach presented in (Potthast and Sukhatme, 2014), where the focus is put on the exploration of cluttered environments. In this approach, a belief model of the unobserved space is used to estimate the visibility of occluded space and, thus, also predict the expected information gain of each possible next-view candidate.

The NBV approach proposed in (Wu et al., 2014) relies on observed voxels of an unknown 3D object captured by a range camera from a single view and a finite list of next-view candidates, where each candidate is characterized by a translation and rotation in 3D space. In order to select the most promising view from the list (i.e. the next-best-view), the recognition uncertainty expressed by the measure of conditional entropy is considered for each next-view candidate, and the next-best-view is chosen as the one which reduces recognition uncertainty the most. In this context, the reduction of entropy corresponds to the mutual information, and minimizing the recognition uncertainty therefore corresponds to maximizing the mutual information. As a consequence, the proposed NBV algorithm simply selects the view that can maximize the mutual information.

In (Dornhege and Kleiner, 2013), an approach for the exploration of unknown environments is proposed based on the definitions of *voids* as unexplored volumes in 3D space and *frontiers* as regions on the boundary between voids and explored space. More specifically, 3D point clouds are captured and registered with the existing point cloud data, and subsequently they are integrated into the hierarchical octomap data structure. After the registration, those areas are extracted which are occluded or enclosed by obstacles. These areas correspond to unexplored volumes in 3D space and serve for generating voids. The voids are combined with nearby frontiers in order to estimate next-view candidates for observing the unexplored space. Additionally taking into account visibility constraints, locations are determined from which as many of the void spaces as possible are visible. According to their assigned visibility score, these locations are then visited sequentially until the entire space has been explored.

A further approach for solving the NBV problem has been presented for an autonomous reconstruction of objects (Vasquez-Gomez et al., 2013). This approach focuses on the generation of next-view candidates by uniformly sampling viewpoints around the sensor and a subsequent ranking of these next-view candidates according to their suitability for the reconstruction process. Thereby, the ranking is derived by applying a utility function which, in turn, is defined as a product whose factors rely on constraints addressing the occupancy of the local voxel space, the overlap of a new surface with previous surfaces, the visibility of unknown voxels, and the distance of next-view candidates to the current view. The whole framework also involves a scene representation by octrees with a hierarchical ray tracing that significantly reduces the visibility computation time.

Besides taking into account rather intuitive constraints, it has also been proposed to consider further information which might be valuable for an accurate 3D reconstruction of an unknown 3D object. In this regard, a NBV approach has been presented which also considers the positioning error of the sensor and selects the view that minimizes the effects of this error (Vásquez and Sucar, 2011). More specifically, candidate views are generated via a uniform sampling of 3D space, and each of these views is characterized by (*i*) a configuration formed by the position and the orientation towards the object and (*ii*) a numerical value assigned by a utility function evaluating the respective configuration by exploiting ray tracing based on the current state of the model. Thereby, the numerical value is rather high for views considered as good views. Finally, the utility function is convolved with a Gaussian function taking into account the positioning error. This allows to re-evaluate candidate views according to their neighbors, so that views with lower positioning errors are preferred.

In (Soudarissanane and Lindenbergh, 2011), the selection of optimal scan positions for acquiring an indoor scene with a terrestrial laser scanner is investigated with a particular focus on the influence of both incidence angle and range limitation. Based on these constraints, visibility polygons are defined for a 2D map corresponding to a 2D projection of the scene onto a horizontally oriented plane. Then, potential viewpoint locations are derived by discretizing the scene via a gridding with predefined steps and, by considering the defined constraints, an optimal number of viewpoints is determined which allows to appropriately cover the considered scene.

A NBV approach specialized to the acquisition of piping objects with a laser scanner is presented in (Kawashima et al., 2014). After each scan, piping objects are detected and those spaces are estimated which are occluded by these piping objects but might be occupied by unseen piping objects. Based on this information, the next-best-view is derived in a way that minimizes the occluded spaces.

3. METHODOLOGY

This section starts with a short overview of the methodology proposed in this paper (Figure 1). Subsequently, we provide a detailed description of the single parts in different subsections. As mentioned in previous work from (Bissmarck et al., 2015), we assume an obstacle-free work space where the sensor can move

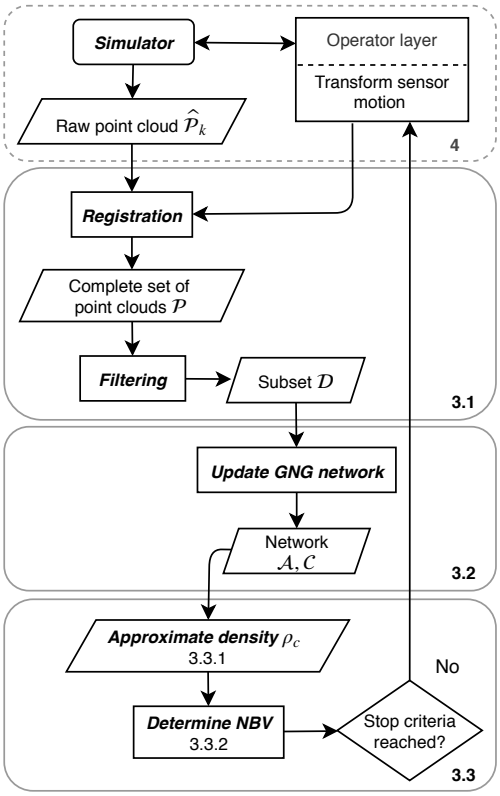

Figure 1. Flowchart of the complete NBV pipeline with the related sections. The chart is divided in four parts, beginning at the top with the dashed grey line which indicates the data generation, described in Section 4. The following solid grey parts belong to the methodology explained in Section 3.

to any position around the object, what makes the sensor space continuous. In addition, the object space lies inside the work space, but is not a part of it. Due the sampled data, the object space is discrete. To show the contrast to other NBV approaches, we follow (Scott et al., 2003) and subdivide the non-model based NBV methods in three categories: *surface-based*, *volumetric* and *global* ones. Our approach principally belongs to the *surface-based* methods, since it is completely data-driven. However, the involved topological model (Section 3.2) and the density approximation (Section 3.3.1) rely on *volumetric* considerations. As a consequence, our approach is a hybrid method with a continuous model space. This allows the *NBV Finder* (Section 3.3) to use almost every position of the model for the NBV.

At the beginning of the NBV pipeline, an initial orientation and position for the sensor is defined. We assume that the first scan tries to acquire as much as possible from the object surface. The initial orientation and position parameters are usually set by the operator in the *Operator layer*. In this work, the *Operator layer* is not active and shows only where the possible user interaction may occur. The actual motion is set and processed only by the *Transform sensor motion* block in Figure 1, top row. This block transforms the orientation and position parameters in a specific format and transfers them to the *Simulator*. This separate step is necessary because the *Simulator* should stay as a standalone module which can be easily replaced by a real sensor if needed. Now, the *Simulator* acquires a single point cloud $\hat{\mathcal{P}}_k$ from the desired location under consideration of the underlying sensor model (see Section 4.2 for details). The raw point cloud is then streamed to the *Registration* block and a subsequent *Filtering* based on an approximate bounding box. Afterwards, a self-organizing vector-based clustering algorithm represented by the Growing Neural Gas (GNG) algorithm (Section 3.2) is updated with a subset of the complete point cloud set $\mathcal{P}$. From the created graph representation of topological relations, the density of the point cloud at each network node is estimated and the NBV Finder now selects that network node as NBV which has the lowest point density $\rho_{\min}$ (Section 3.3). The position and orientation in the sensor space are computed and transferred back to the *Simulator* where the operator or a robot moves the sensor to the new position, respectively in our case the *Transform sensor motion* block which provides the sensor position to the *Simulator*. The next scan of a new point cloud begins. Our NBV method is self-terminating when (*i*) the maximum scan number is reached, (*ii*) all regions reach a pre-defined minimum point density or (*iii*) the change of the GNG error is zero.

3.1 Data Registration and Filtering

The data is given in the form of a set of 3D point clouds $\mathcal{P} = \{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$ containing the single point clouds $\mathcal{P}_k$ at scan time k with $k \in \mathbb{N}$. For further processing, the reference frame has to be changed from the sensor coordinate system to a local world coordinate system. Due the fact that the Simulation Framework (see Section 4 for more details) provides accurate position and orientation parameters $\{\vec{t_k},\ \vec{R_k}\} \in T_k$ for the sensor, this process is simple and straightforward: $\mathcal{P}_k = T_k(\hat{\mathcal{P}}_k)$. This work doesn't deal with the registration of point clouds, we focus mainly on the next-best-view search. For finding the optimal parameters between two sets of point clouds, we refer to (Weinmann, 2016).

After the registration, an approximate bounding box is used to remove points further away which are not of interest. This reduces the spread of the object space what gives a better start configuration. The size of the bounding box can be a problem, since there is no prior information about the size of the object or the whole scene, respectively. Therefore, after every scan, the barycenter and the size of the bounding box are again derived from the geometrical size of $\mathcal{P}$. For increasing the overall performance, every time a new point cloud from the simulator is streamed, the complete set of point clouds $\mathcal{P}$ is shuffled and randomly subsampled to a small amount of the original size. This method preserves the topology and structure of the data. The influence for the clustering is not significant as the overall characteristic of the point cloud will be preserved. This subset $\mathcal{D} \subseteq \mathcal{P}$ (see Figure 1, center part) of the input data has a constant size M.

3.2 Building a Topological Model via the Growing Neural Gas algorithm

The Growing Neural Gas (GNG) algorithm presented in (Fritzke, 1995) is a vector-based learning method which is capable of building (clustering) topological relationships that are present in the input space by applying the Competitive Hebbian Learning (CHL) rule. The neurons in the m-dimensional continuous output space $\mathbb{R}^m$ are competing for the right to respond to a given input signal $\vec{\xi}$ with $\vec{\xi} \in \mathbb{R}^n$ of the discrete input training dataset $\mathcal{D}$. A single input signal contains three coordinates of a point in the input space. Only one neuron (in our case, the first and second place) can win the competition. This is basically called the "winner-takes-all" scheme.

Like Vector Quantization (VQ), Self-Organizing Maps (SOM) or Neural Gas (NG) – for which an overview is provided in (Fritzke, 1998) – the GNG algorithm builds a graph (also referred to as network) from input training samples. This graph contains a set of nodes $\mathcal{A} = \{c_1, \ldots, c_n\}$ with the associated reference vector $\vec{w}_c \in \mathbb{R}^n$ and the local accumulated error e_c. The reference

vector can be seen as the node position in the n-dimensional input space $\mathbb{R}^n$. Furthermore, the graph contains a set of edges $\mathcal{C} = \mathcal{A} \times \mathcal{A}$ which represent the connections between the nodes. These connections are symmetrical, i.e. $(c_i, c_j) \in \mathcal{C} \Leftrightarrow (c_j, c_i) \in \mathcal{C}$, and therefore the graph is undirected. The set of nodes and edges are part of the topological model to map the object.

The difference to SOMs and similar methods is the growing network size N, i.e. an increasing number of nodes. A fixed network size is not practical due the requirement of capturing all kinds of unknown objects and surface geometries without any prior knowledge. The GNG method is self-adaptable to the size of the input data and to the underlying unknown distribution $P(\vec{\xi})$. For point clouds, the input space is discrete and can be described by the finite subset $\mathcal{D} = \{\vec{\xi^1}, \ldots, \vec{\xi^M}\}$ with $\vec{\xi^i} \in \mathbb{R}^n$. The GNG method allows to learn the geometry of the object in an unsupervised manner. No offline training stage is required. The training takes place online and only relies on the available data.

In the following part, the GNG algorithm and the modifications for making it more adjustable for non-stationary input distributions (related to (Wu et al., 2014)) is described. The core GNG algorithm scheme is mainly based on (Fritzke, 1995).

1. Start with two nodes $\mathcal{A} = \{c_1, c_2\}$ from $P(\vec{\xi})$. These two nodes belong to random positions w_1 and w_2 in $\mathbb{R}^n$.
2. Generate a random input signal $\vec{\xi}$ according to $P(\vec{\xi})$ and find the nearest node s_1 and the second nearest node s_2:
$$s_n = \underset{c \in \mathcal{A}}{\arg\min} \|\vec{\xi} - \vec{w}_c\| \tag{1}$$
3. Adapt the winning reference vector and its direct topological neighbor toward $\vec{\xi}$ by a small amount with the learning rate η
$$\begin{aligned} \Delta w_{s_1} &= \eta_1(t)(\vec{\xi} - w_{s_1}) \\ \Delta w_n &= \eta_2(t)(\vec{\xi} - w_n) \end{aligned} \tag{2}$$
for all direct neighbors of s_1 $\forall c \in \mathcal{N}(n)$, given that $\eta_1(t) > \eta_2(t)$ with the training epoch t, $t \in \mathbb{N}$. For faster adaption to the new input signal and for better convergence, annealing the learning rate helps, which means that the learning rate decreases every training step t. This is a common strategy widely used in the field of machine learning because only one hyperparameter more has to be set:
$$\eta(t) = \eta_{\text{in}} \left(\frac{\eta_{\text{fin}}}{\eta_{\text{in}}} \right)^{t/t_{\max}} \tag{3}$$
where η_{in} is the initial learning rate and η_{fin} the final one. In addition, the learning rate η_{in} is bisected every time the GNG network is updated with new data to prevent too much oscillating from the changing data distribution.
4. Create a new edge between the winner s_1 and the second winner s_2 and set the age to zero. Increment the age of all edges that are connecting with s_1. Remove old edges with a larger age than $a_{\max}$ and remove possible remaining nodes without an edge as well.
5. Update the local error by $\Delta e(s_1) = \|\vec{\xi} - \vec{w}_{s_1}\|^2$ with the squared distance between the input signal and the nearest node in the input space.
6. If the number of generated input signals $\vec{\xi}$ is an integer multiple of λ, insert a new node:
   - Search the node q with the maximum accumulated error e_c.
   - Insert a new node r between q and its topological neighbor p by $\vec{w}_r = 0.5(\vec{w}_p + \vec{w}_q)$.
   - Replace the edge $\mathcal{C}(q, p)$ with the new edges $\mathcal{C}(q, r)$ and $\mathcal{C}(r, p)$.
   - Decrease the error of q and p by multiplying it with a constant $\alpha \in [0, 1]$. The error of the new node r is the new error of p.
   - Decrease all error variables by the constant $\beta \in [0, 1]$ with $\Delta e_c = -\beta e_c$.
7. Go back to step 1, until the maximum number of training epochs $t_{\max}$ is reached. One epoch t is one full training cycle on the training set. To get the overall performance of the network, the L2-Norm of the local error e_c followed by a division with the squared root of the current network size N results in the root-mean-square-error (RMSE), which is computed after every training step t: $\text{RMSE}(t) = \|e_c\|_2 / \sqrt{N}$. Once every sample $\vec{\xi}$ from the set is seen, the training set $\mathcal{D}$ is shuffled and the next training epoch starts.

Most of these hyperparameters are robust and can be used for different objects. The hyperparameter set for this work is shown in Table 1. After termination of the GNG algorithm, the nodes $\mathcal{A}$ and the edges $\mathcal{C}$ are stored and used later as start configuration for the update step.

3.3 The NBV Finder

To find the next-best-view, it is necessary to localize where the topological model from Section 3.2 is incomplete, which implies that the reconstructed point cloud $\mathcal{P}$ still contains gaps and that the point density on the object surface is not sufficiently dense. This process of evaluating the topological relations under consideration of low point density regions is described in the following.

For finding regions with low point density, every data point from the input space $\mathbb{R}^n$ is associated with one reference vector $\vec{w}_c$. Every reference vector can been seen as a Voronoi cell V_c. A Voronoi cell V_c defines a region in which a data point is closer to one node c with the reference vector $\vec{w}_c$ than to any other node:
$$V_c = \left\{ \vec{\xi} \in \mathbb{R}^n | c = \underset{c \in \mathcal{A}}{\arg\min} \|\vec{\xi} - \vec{w}_c\| \right\} \tag{4}$$
Accordingly, every Voronoi cell V_c describes a convex region of $\mathbb{R}^n$. Through the partitioning of $\mathbb{R}^n$ by the Voronoi cells, it is equivalent to a Voronoi set $\mathcal{R}_c$ which contains every point from $\mathcal{D}$ that belongs to V_c:
$$\mathcal{R}_c = \left\{ \vec{\xi} \in \mathcal{D} | s(\vec{\xi}) = c \right\} \tag{5}$$
The next goal consists in computing the volume Ω_c of V_c and in estimating the density ρ_c with respect to $\mathcal{R}_c$ for each of the network nodes c_n.

3.3.1 Density estimation from unbounded Voronoi cells For estimating the point density of a surface or 3D volume, the size of the volume is required. The Voronoi cells V_c represent a convex subspace of the input space with the Voronoi set $\mathcal{R}_c$. In the 2D case, it is unambiguous to calculate the surface area if the cell is bounded. In an unbounded case, a simple rectangular bounding box can be used. For higher dimensional cases like 3D space, the Voronoi cell is usually unbounded and the volume computation is much more complex. It is theoretically possible to compute first a Delaunay Triangulation and afterwards a convex hull for each Voronoi cell to get the volume. But the GNG network $\{\mathcal{A}, \mathcal{C}\}$ is not presenting triangles, the result is unpredictable and the convex hull gives mostly a volume that is overestimated.

To avoid these restrictions, we introduce a three step approximation which is more obvious and simpler to overcome the explicit volume computation of 3D Voronoi cells.

1. Simultaneous spherical growing at each network node c_n. The growing for a single sphere Θ_c stops, if the sphere intersects with another neighboring sphere.
2. For each Voronoi cell V_c, do a simultaneous spherical growing at each single point in the Voronoi set $\mathcal{R}_c$. The growing for a sphere $\Theta_{\vec{w}_c}$ stops, if the sphere intersects with another neighboring sphere from $\mathcal{R}_c$.
3. For all spheres $\Theta = \{\Theta_c, \Theta_{\vec{w}_c}\}$ which belong to one Voronoi cell V_c, compute the convex hull (for simplification, named as H_c)
$$H_c = \text{conv}(\Theta) = \bigcap_{\Theta \subseteq C \subseteq \mathbb{R}^n} C \tag{6}$$
whereby C is a convex set, which contains all points from Θ.

The sphere intersections can be computed analytically. The complete growing has an almost linear runtime $\mathcal{O}(n)$, nearly independent of the growing step size which is around 10^{-2} in our case. The convex hull can be computed in an average runtime of $\mathcal{O}(n \log n)$ with the QuickHull algorithm (Barber et al., 1996). With H_c, we also have the volume Ω_c of the hull. With the Voronoi set $\mathcal{R}_c$, the density ρ_c can easily be derived:

$$\rho_c = \frac{\mathcal{R}_c}{\Omega_c} \tag{7}$$

As indicated in Section 3.2, beside the set of nodes $\mathcal{A}$ and the set of edges $\mathcal{C}$, our model also contains the point density ρ_c for each Voronoi cell V_c. With these parameters, the topological model is complete and the density of the object can be evaluated. The NBV determination is explained in Section 3.3.2. The benefit of this approach is that the location of the nodes in $\mathbb{R}^m$ is independent from the location of the data points in $\mathbb{R}^n$. This is more flexible than e.g. Occupancy Grid Maps which divide the model and object space in discrete voxels (Bissmarck et al., 2015).

3.3.2 Determine the NBV To meet the requirements of a dense object surface, the overall intention is to maximize the average point density $\tilde{\rho}_{\text{avg}}$

$$\tilde{\rho}_{\text{avg}} = \frac{1}{N} \sum_{c \in \mathcal{A}} \rho_c \rightarrow \max \tag{8}$$

by summing up the single densities ρ_c and dividing by the network size N. This maximization cannot be solved directly. It is only approximately solvable by selecting possible good sensor positions which probably increase $\tilde{\rho}_{\text{avg}}$. For this, it is necessary to locate the less dense regions. This can be done by sorting all ρ_c in ascending order

$$\Psi = \underset{c \in \mathcal{A}}{\arg\text{sort}}\, \rho_c \tag{9}$$

with the set of sorted node indices $\Psi = \{\Psi_1 < \Psi_2 < \cdots < \Psi_N\}$. Ψ represents the possible next-best-view candidates. The corresponding node with the smallest point density ρ_{min} has the lowest index which is the first element of Ψ:

$$c_{\text{min}} = \mathcal{A}(\Psi_1) \tag{10}$$

The node c_{min} with the associated reference vector $\vec{w}_{c_{\text{min}}}$ is now the start point of the next-best-view position in the object space. To compute the associated position and orientation in the sensor space, the normal from the corresponding Voronoi set $\mathcal{R}_{c_{\text{min}}}$ has to be estimated to get the direction. This can be done by fitting a plane through the points $\mathcal{R}_{c_{\text{min}}}$ and subsequently computing the unit normal vector $\vec{n}_0$ from the plane. For determining five of the six degrees of freedom of the motion parameters (described by the set $\mathcal{X}$) to place the sensor correctly in the local Cartesian coordinate system, three coordinates for the position $\mathcal{X} = \{x, y, z\}$ and three Euler angles for the orientation $\mathcal{X} = \{\varphi, \theta, (\psi)\}$ have to be found, whereby φ is the rotation around the z-axis, θ the rotation around the rotated x-axis and ψ the rotation around the rotated z-axis, assuming a right hand coordinate system with the z-axis up and the x-axis to the right. The last degree of freedom cannot be determined from the normal because the z-axis of the sensor is pointing towards the node c_{min} and corresponds to the normal $\vec{n}_0$. The angle ψ remains indeterminate what is not critical, because the underlying sensor has a quadratic field-of-view (FOV).

With the knowledge of the underlying sensor model, the length of the normal can be adjusted to the correct length $d_{\mathcal{X}}$, to get the optimal distance between the object surface and the NBV sensor position. This distance is important for the resulting point density on the object surface. The closer the sensor is to the object, the denser becomes the recorded part of the surface. Under the assumption of a rectangular FOV and a known size of the imaging sensor S_{im}, an approximation from the two-dimensional case can be made about the required distance to the object surface under a certain object size S_{obj}. With the geometrical intercept theorem assuming a standard single lens construction

$$d_{\text{im}} = \frac{S_{\text{im}}}{S_{\text{obj}}} \cdot d_{\mathcal{X}} \tag{11}$$

and the thin lens equation, the connection between the focal point f, the image distance d_{im} and the object distance $d_{\mathcal{X}}$ is given by

$$\frac{1}{f} = \frac{1}{d_{\text{im}}} + \frac{1}{d_{\mathcal{X}}} \quad \Longrightarrow \quad d_{\mathcal{X}} = f \cdot \left(\frac{S_{\text{obj}}}{S_{\text{im}}} + 1 \right) \tag{12}$$

The object size S_{obj} is derived from the parameter ρ_{user} which is a pre-defined value set by the operator to ensure that the object surface has the required point density. This parameter is defined as the number of points per square meter. There are two possible modes, depending on ρ_{user}:

- *Fast Mode*: If $\rho_{\text{user}} = 0$, the distance between the object surface and the sensor position is approximated, so that the sensor FOV covers as much as possible from the current known object geometry.
- *Densify Mode*: For $\rho_{\text{user}} > 0$, S_{obj} is chosen in a way that $d_{\mathcal{X}}$ is the optimal distance to the object surface for the desired point density per square meter ρ_{user}.

The only difference between these two modes (in short, *FM* and *DM*) is the stop criterion. The *FM* terminates if the slope of the network error RMSE(t) is not decreasing anymore. In other words, RMSE(t) stops lowering, what means that the network $\{\mathcal{A}, \mathcal{C}\}$ reaches a nearly stable state. *DM* terminates if $\rho_{\text{user}} > \rho_{\text{min}}$ which imply that all of the single point densities ρ_c are greater than ρ_{user}. This means that all of the Voronoi cells V_c are dense which leads to the assumption that the object has a dense surface. This simplification can be made, because we assume that the volume Ω_c is getting smaller and a Voronoi cell V_c converges with an increasing network size and the proceeding scan steps to a plane-like surface.

3.4 Evaluation Strategy

For evaluating the results of our proposed NBV method, it is necessary to find significant evaluation parameters E_p. For the reconstruction quality of $\mathcal{P}$, like the completeness, we use the approximated surface coverage c_{ov} (in $\%$) between a reference point cloud $\mathcal{P}_{\text{ref}}$ and $\mathcal{P}$. Both point clouds are subsampled via Poisson Disk Sampling (Corsini et al., 2012) to get equivalent point distances, followed by spherical region growing (Section 3.3.1) for each point and then counting the sphere intersections between both point clouds. For detecting and measuring gaps in the reconstructed point cloud, the averaged absolute "mesh-to-point" distance $\tilde{d}_{\text{abs}}$, between a meshed reference surface and $\mathcal{P}$ is calculated. In addition, the NBV efficiency is measured by the number of scans k_{max} which are required to reach a pre-defined minimum point density ρ_{min}. The number of scans k_{max} is also important to see when the *Fast Mode* terminates, assuming that there are no large gaps in the reconstructed point cloud anymore. These parameters can be summarized to the following tuple

$$E_p = \left(c_{\text{ov}},\ \tilde{d}_{\text{abs}},\ k_{\text{max}},\ \tilde{\rho}_{\text{avg}} \right) \tag{13}$$

which includes $\tilde{\rho}_{\text{avg}}$ as a remarkable parameter for the achieved average object point density. For visualizing the gaps in the reconstructed point cloud, the absolute "point-to-point" distance between $\mathcal{P}_{\text{ref}}$ and $\mathcal{P}$ under the assumption of a complete reconstructed point cloud (see e.g. Figure 5) is more plausible.

To show how our method performs in comparison with another approach, we introduce a "naive random approach". Finding good NBV benchmark sets is very challenging. For example, the benchmark from (Munkelt et al., 2007) was not sufficient in our case, since the test object was a synthetic cube and not a real-world object which has typically a shape of much higher complexity. Other studies consider different tasks and objects or provide no reference data. Therefore, we use our own approach for comparison and place uniformly distributed sensor positions on a sphere around the object, pointing towards the barycenter of the object. This may be naive but is a common straightforward method, especially for locally limited objects like the ones from our test dataset shown in Figure 3. To get these sensor positions in a simple manner, an icosphere can be used. This type of sphere has a surface which only contains triangles of equal size. Also commonly referred to as a geodesic dome, the basic type is a regular polyhedron called icosahedron and has 20 faces F_{20}. The number of faces can be easily extended by subdividing each triangle into four new triangles and we thus get F_{80} and so on. In addition to making the sphere approximation more consistent, the sensor positions are not directly the center points of the faces, but the projected points on a sphere with the same radius.

4. SIMULATION FRAMEWORK

For testing the proposed NBV algorithm, there are different ways to acquire point-sampled data of 3D objects serving as input for the algorithm. On the one hand, a ranging device could be used for acquiring data in a real-world scenario and the data could be sent to a processing unit. While this is the most realistic case, data acquisition can however be challenging and time-consuming. Beside the high effort and the need of real systems for data acquisition, the acquisition of accurate ground truth data required to test and compare algorithms is non-trivial. In this context, ground truth data may not only be required for the sensor movement, but also for the 3D structure of a considered object or scene if the result of 3D reconstruction should be evaluated as well. Instead of considering a real-world scenario, one could also focus on the consideration of recorded data and fixed sensor positions. While this reduces the effort considerably, such a strategy is less flexible and not practical for testing NBV methods. Consequently, it would be desirable to allow for testing a NBV algorithm without involving too much human effort while still being able to preserve flexibility with respect to possible sensor positions.

This can be realized via a simulation framework which can be used to create any kind of virtual scenario from a small statue to a large complex outdoor scenario due to the almost infinite work space and the arbitrary level of detail. The movement of the sensor can be recorded exactly and, accordingly, exact ground truth data is available for evaluating the accuracy of derived registration parameters to reconstruct the scene. The whole data generation process as shown in Figure 1, first block, is bundled in our simulation framework together with the point cloud generator, the sensor motion processing, the data stream for the data and sensor parameters, and the sensor model. Most of these parts are based on Blender (`www.blender.org`), an established open source 3D animation suite which addresses many different tasks and which may easily be extended.

4.1 BlenSor Toolbox

The most important part of the considered simulation is a physically correct ray tracing for single rays which are reflected by an object surface and get back to the sensor array. To the best of our knowledge, the most advanced toolbox in this regard is currently represented by BlenSor (Blender Sensor Simulation Toolbox) (Gschwandtner et al., 2011). BlenSor is a special ray tracer for Blender which modifies the ray-tracing to match the sensor characteristics and thus produces point clouds as obtained with a respective sensor. In this context, the simulated point clouds may rely on the use of different sensors, whereby BlenSor takes into account several physical properties such as backfolding, reflections and sensor noise (if necessary).

For a scene of medium complexity, a full scan can for instance be simulated with approximately $12{,}000$ rays per second when applying the sensor model described in Section 4.2 and testing on a standard desktop computer with six cores and 16 GB of RAM under Linux 4.2.0.

4.2 Sensor Model

For our experiments, we use the model of a Time-of-Flight (ToF) camera exploiting the PMD (Photonic Mixer Device) technology (Jutzi, 2015). Other devices such as a Microsoft Kinect or a Velodyne HDL are also available in the toolbox and could therefore also be used. The main advantages of using a ToF camera consist in the similarity to standard passive sensors like RGB cameras in terms of the behavior of ray paths, and in the fact that a ToF camera can also be applied in outdoor environments and beyond the sensor specifications (Weinmann et al., 2011).

The considered sensor model is defined in analogy to the commonly used PMD[vision] CamCube representing a state-of-the-art range camera. Due to the respective sensor array and configuration, the acquired images have a size of 204×204 pixels which corresponds to a quadratic field-of-view of $40^\circ \times 40^\circ$. Accordingly, the angular resolution of the device is 0.2°.

5. EXPERIMENTS

All experiments were realized with the simulation framework (Section 4) and the two test objects shown in Figure 3. In the following, we provide our parameter settings and a short runtime test (Section 5.1) before we present the derived results (Section 5.2).

5.1 Hyperparameter Adaption

Due the fact that the runtime of the GNG algorithm is $\mathcal{O}(N^2)$, the performance is critically linked with number of nodes N and also slightly affected by the number of training epochs $t_{\max}$ and by the size of the training samples M. Figure 2 shows the influence of the hyperparameters to the elapsed time for one update step, which the GNG algorithm needs to update the network with a new training set. For this performance test, a larger training set with 50,000 input signals is used.

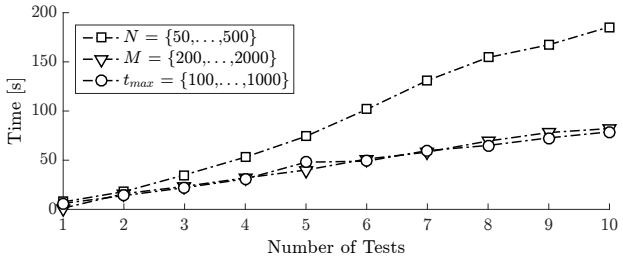

Figure 2. Test of the three most important hyperparameters (number of nodes, number of training samples and number of maximum training epochs) on runtime influence. Every hyperparameter run through ten tests, indicated on the x-axis, in the value range as displayed. During each run, the other two hyperparameters are set on a constant mid-range value, respectively.

Figure 2 (dashed-dot line with square markers) clearly reveals the effect on the runtime if the number of network nodes N increases. Therefore, we use for the experiments the smallest number of nodes which can still represent the topology of the input data. These are at the beginning about 5% of M and then decreasing to 1% of M as shown in Table 1. This method (Equation 3) saturates N after k steps to avoid too many performance dips and to force the network to a consistent distribution of the nodes in $\mathbb{R}^m$. The other hyperparameters which are not listed in the table are set to the default values from (Fritzke, 1995). Nevertheless, a small increase of $t_{\max}$ is necessary to keep the network error low with the increasing amount of data.

Parameter	Value	Parameter	Value
$t_{\max}$	$100 \ldots 500$	η_1	$0.5 \ldots 0.01$
N	$M \cdot (0.05 \ldots 0.01)$	η_2	$0.05 \ldots 0.001$
M	500	λ	$350 \ldots 500$

Table 1. The hyperparameter set for the GNG algorithm. Some of these hyperparameters are directly linked with the current scan number k, the current training epoch t and the size of the input data M.

5.2 Experimental Results

The dataset for testing our NBV approach contains two different objects. A Buddha statue (Figure 3, left side) with a rather simple surface geometry but still some concave parts and the popular Stanford Dragon (Figure 3, right side) with a much more complex object surface containing occluded and concave parts like regions inside and outside the jaws. As mentioned earlier, the initial position and orientation of the sensor is set to a fixed value. We assume that the sensor covers a substantial part of the object to get some evaluable information to start with.

The results for the Buddha in both modes are provided in Table 2 (top). The first column displays the used mode and the target point density per square meter which the object should have in the end. *FM* (*Fast Mode*) is averaged over 20 runs. *DM* (*Densify*

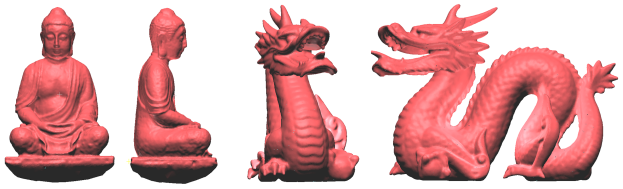

Figure 3. The two test objects: On the left our Buddha statue, scanned with a high resolution 3D scanner and meshed. On the right, the Stanford Dragon meshed by (Rodolà et al., 2013).

Buddha test object					
Mode (ρ_{user})	c_{ov}	d_{abs} [m] / $\sigma_{d_{\text{abs}}}$	$k_{\max}$ / $\sigma_{k_{\max}}$	$\tilde{\rho}_{\text{avg}}$ [pts/m^2]	$\rho_{\min}$ [pts/m^2]
FM (-)	99.97%	0.209 / 0.138	10 / 3	85,076	15,321
DM (20,000)	99.90%	0.232 / 0.157	11 / 2	91,638	23,430
DM (30,000)	100%	0.237 / 0.156	14 / 2	106,695	32,850
DM (40,000)	100%	0.226 / 0.153	17 / 1	129,171	43,515
Dragon test object					
Mode (ρ_{user})	c_{ov}	d_{abs} [m] / $\sigma_{d_{\text{abs}}}$	$k_{\max}$ / $\sigma_{k_{\max}}$	$\tilde{\rho}_{\text{avg}}$ [pts/m^2]	$\rho_{\min}$ [pts/m^2]
FM (-)	99.07%	0.376 / 0.311	11 / 2	105,576	15,989
DM (20,000)	99.49%	0.384 / 0.311	13 / 3	117,103	22,414
DM (30,000)	99.20%	0.399 / 0.319	17 / 2	151,649	32,728
DM (40,000)	99.68%	0.392 / 0.318	20 / 1	177,497	42,766

Table 2. Results for the Buddha test object (top) and the Dragon test object (bottom): the tables show the E_P and the minimal point density $\rho_{\min}$ listed by the modes, for three different user densities in the *DM*.

Mode) is averaged over 10 runs. The next four columns are the evaluation parameters from Section 3.4. Provided that the theoretical maximum for c_{ov} is 100%, for d_{abs} is the lower the better and for $k_{\max}$ also the lower the better. The average point density $\tilde{\rho}_{\text{avg}}$ is calculated by Equation 8. In addition, we show the standard deviation of $k_{\max}$ to see the variance and the minimal reached point density $\rho_{\min}$ which have to be higher than ρ_{user} in the *DM*. The same test was done for the Dragon object, the results are shown in Table 2 (bottom). The hyperparameters were the same for both test objects.

For an expression how the result of a complete NBV finding process looks like, Figure 4 shows exemplarily two final sensor constellations after 14 scans (Buddha, top) and 17 scans (Dragon, bottom) and the corresponding GNG model on the right side. For visualizing possible gaps after a finished run, especially for an operator, the color encoding in Figure 5 indicates the absolute distance in different colors to show parts which are difficult to acquire by the sensor, e.g. occluded parts, whereby every distance greater than 1 cm is treated as a gap and displayed in red.

5.2.1 Comparison to a naive uniform approach

To see how our approach performs in comparison to a different approach, we choose a "naive random" approach based on uniformly distributed positions on a sphere and evaluate the approach in exactly the same experiment as before with the same criteria. The sensor positions were placed uniformly on a sphere around the object. The barycenter of the object fell together with the sphere center. The sphere positions are computed from an icosphere with F_{20} (see Section 3.4 for details). This means, the upper limit of $k_{\max}$ is 20, which is the same limit as for the *FM* and *DM* 20,000 and also *DM* 30,000. For *DM* 40,000, the upper limit is $k_{\max}$ = 25 but was not tested with the "naive random" approach. The initial position and orientation is the same as with our approach, but then integrated in the sphere positions. To make the comparison as fair as possible, after the initial sensor configuration, the next-best-view is chosen randomly from the set of available positions and the correct distance to the object is adjusted as the current point density requires (see Equation 9). The results are averaged over 10 runs and summarized in Table 3 for the Buddha and the Dragon object.

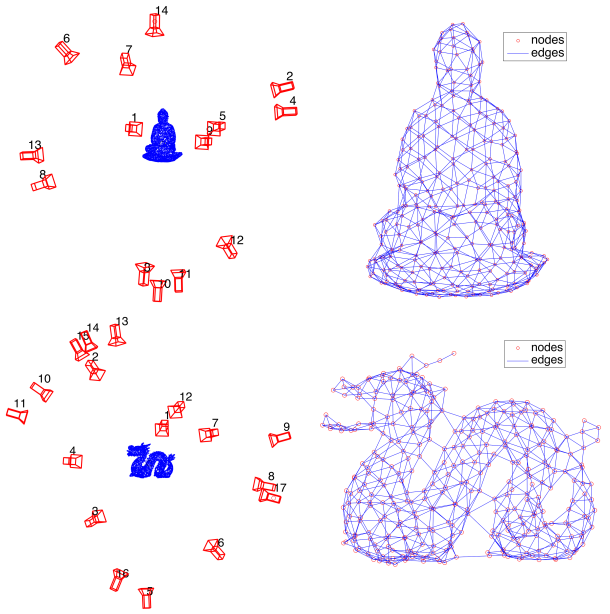

Figure 4. Sensor constellations after a finished run in *DM* ($\rho_{\text{user}} = 30{,}000$) on the left and the corresponding GNG model on the right side: result for the Buddha test object (top, $k_{\max} = 14$) and result for the Dragon test object (bottom, $k_{\max} = 17$).

Buddha test object					
Mode (ρ_{user})	c_{ov}	d_{abs} [m] / $\sigma_{d_{\text{abs}}}$	$k_{\max}$ / $\sigma_{k_{\max}}$	$\tilde{\rho}_{\text{avg}}$ [pts/m^2]	$\rho_{\min}$ [pts/m^2]
FM (-)	99.81%	0.225 / 0.150	9 / 2	80,573	12,138
DM (20,000)	99.97%	0.208 / 0.144	14 / 2	113,061	22,773
DM (30,000)	100%	0.219 / 0.150	15 / 2	122,114	34,837
Dragon test object					
Mode (ρ_{user})	c_{ov}	d_{abs} [m] / $\sigma_{d_{\text{abs}}}$	$k_{\max}$ / $\sigma_{k_{\max}}$	$\tilde{\rho}_{\text{avg}}$ [pts/m^2]	$\rho_{\min}$ [pts/m^2]
FM (-)	99.52%	0.347 / 0.289	11 / 2	103,023	16,910
DM (20,000)	99.58%	0.316 / 0.287	15 / 2	154,946	26,509
DM (30,000)	99.68%	0.347 / 0.303	18 / 1	150,035	32,645

Table 3. Results for the Buddha test object (top) and the Dragon test object (bottom) with the "naive random" approach: the tables show the E_P and the minimal point density $\rho_{\min}$ listed by the modes, for two different user densities in the *DM*.

6. DISCUSSION AND CONCLUSION

One of the main ideas behind our proposed method is the planning of the next-best-view based on an evaluation of the object topology by finding low point density regions. This procedure is compared with a naive random approach and gives reasonable results which are discussed in this section.

A good impression is given in Figure 4 which shows the result of a typical model (right side), build up only from the data. Also small details are represented by at least one node to estimate the point density in this region like small parts on the head of the dragon. For objects with a simpler shape like the Buddha, the sensor positions are placed almost uniformly around it, what is a replicable step. If some positions are close to each other like 13/8 and 2/4 (Figure 4, left, top row), the desired point density was not reached with the first scan in this area or a new part of the object was discovered with a less dense area. If the object is more complex, our algorithm places more scans near the critical parts and thus tries to get the surface more dense. This behavior is normal and can also be observed in Figure 4 (left, bottom row) for the sensor positions 2/13/14/15.

The remaining gaps and less dense regions in the reconstructed point cloud are shown in Figure 5. Red highlights the gaps, blue and green are dense regions and yellow are less dense ones. The

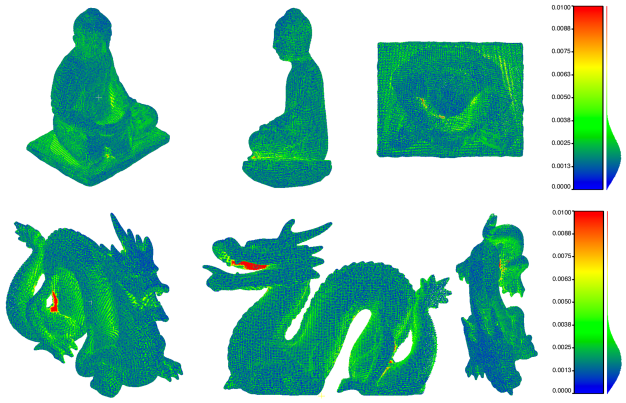

Figure 5. Absolute point-to-point distance encoded by different colors between the reconstructed and reference point cloud for each of the two test objects. All distances greater than 0.01m are marked as gaps in the surface and highlighted in red. In addition to the color map, a histogram showing the distribution of the distance values is provided.

Buddha in the top row was acquired with 14 scans and shows no gaps at all. The Dragon below has two small gaps. One occluded part on the side behind a hind feet and on inside the jaw. The 17 scans were still not enough, but these regions are very challenging for an automatic acquisition and would normally require the intervention from an operator with empiric knowledge about respective regions.

With the increasing number of scans, the point density increases too, but not linearly and only locally. The average point density $\tilde{\rho}_{\text{avg}}$ is not a reliable indicator for the complete object density which can be clearly seen if the number of views and the reached average point density in Tables 2 and 3 are compared. Especially in Table 3 (bottom), it becomes clearly visible that $\tilde{\rho}_{\text{avg}}$ is higher with 15 scans in the second row as with 18 scans in the third row. A look at the minimum point density $\rho_{\min}$ reveals the inconsistency. A high coverage is an indicator for gaps, as also d_{abs}, the point-to-mesh distance is. All of the results have a high coverage near 100% which means that no essential part of the objects were forgotten to scan. The point-to-mesh distance is difficult to interpret, because it is nearly constant for a test object, smaller than 1 mm and the standard deviation has often the same magnitude. Therefore, the coverage is much more intuitive to analyze.

If we first compare the two approaches among themselves separately, an understandable trend is visible, depending on the test object. The number of scans $k_{\max}$ increases with the required point density and with the object complexity. This can be seen if we compare the results of the Buddha and the Dragon from our NBV approach in Table 2. With the object complexity, the obtained coverage decreases what explains the small gaps as shown before in Figure 5. The coverage is not always consistent, but we have to keep in mind that the result of a comparison between two point clouds with several ten thousand points are mapped between 0 and 100. This distorts the result. Furthermore, the standard deviation of $k_{\max}$ is almost consistent and, with a higher parameter ρ_{user}, the standard deviation decreases to one scan which makes the approach more reasonable. Only for the *Fast Mode* (*FM*), $\sigma_{k_{\max}}$ tends to be a little higher. The termination of *FM* depends on the GNG network error which is an indicator for the network stability. As displayed in Figure 6, the red line shows the RMSE for a complete NBV Finding process at every training epoch t, the black line is the slope of the error, indicating the change rate. With increasing k, the slope gets smaller and if the

slope changes the sign, the *FM* stops, assuming a stable network where most of the gaps in the object are closed. This is shown in the top row in Figure 6. Note that both y-axes are in logarithmic scale. In the bottom row are the same graphics, shown for the *DM*. The big spikes in the RMSE at the beginning of each scan process are good indicators that the network gets new data which covers a new part of the object. Newly inserted nodes and older nodes have to move more to fit the new distribution. This indicates that a large new undiscovered region of the object was scanned and fed into the network.

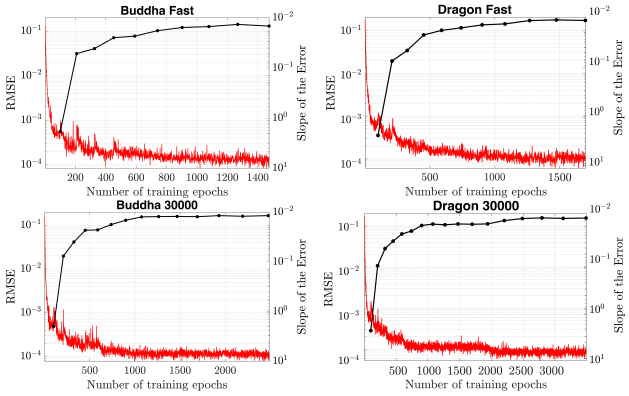

Figure 6. RMSE of the GNG network (red) and slope of the error (black, inverse y-axis) for four different runs: *FM* (top row) with Buddha ($k_{\max} = 10$) on the left side and Dragon ($k_{\max} = 11$) on the right side. The same constellation is provided in the bottom row, ($k_{\max} = 14$ and 17) for *DM* with $\rho_{\text{user}} = 30{,}000$.

Finally, the results of our proposed method and the "naive random" approach can be compared if we consider Table 3. Except for the *FM*, our NBV method outperforms the "naive random" approach in efficiency, i.e. our proposed method reaches the desired minimum point density which is set by the user faster than with uniformly distributed sensor positions. For 20,000 points per square meter, about 20% earlier and for 30,000, about 10% earlier. A possible explanation why the approach does not perform better in the *FM* then the "naive random" approach can be the sensor positions. Since the goal is not a dense surface, uniformly distributed sensor positions around the object are likely better for this case and lead to a faster stable network geometry.

The goal of this proposed work was not to find "optimal" sensor positions and to define an absolute lower limit for $k_{\max}$ for a certain object. It was the maximization of the point density on the surface of unknown objects. Beside this, it is almost impossible to find always an "optimal" sensor configuration for every object type without any prior knowledge, which finally leads to the conclusion that the presented next-best-view method can neither ensure a full model completeness nor that all gaps in the data are closed. But with the focus on next-best-view positions for regions with small point density, e.g. on the borders of the GNG network, it is more likely that the surface of the object is complete and has a certain local minimum point density while acquiring a lower number of scans. To further increase the efficiency, more criteria like the overlapping between scans and the accurate estimation of the normal from complex object geometries are the next best starting points for future work.

REFERENCES

Alsadik, B. S., Gerke, M. and Vosselmann, G., 2012. Optimal camera network design for 3D modeling of cultural heritage. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. I-3, pp. 7–12.

Barber, C. B., Dobkin, D. P. and Huhdanpaa, H., 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software*, 22(4), pp. 469–483.

Bissmarck, F., Svensson, M. and Tolt, G., 2015. Efficient algorithms for next best view evaluation. *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5876–5883.

Connolly, C., 1985. The determination of next best views. *Proc. IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 432–435.

Corsini, M., Cignoni, P. and Scopigno, R., 2012. Efficient and flexible sampling with blue noise properties of triangular meshes. *IEEE Transactions on Visualization and Computer Graphics*, 8(6), pp. 914–924.

Dornhege, C. and Kleiner, A., 2013. A frontier-void-based approach for autonomous exploration in 3D. *Advanced Robotics*, 27(6), pp. 459–468.

Dunn, E. and Frahm, J.-F., 2009. Next best view planning for active model improvement. *Proc. British Machine Vision Conference*, pp. 53.1–53.11.

Fritzke, B., 1995. A growing neural gas network learns topologies. *Advances in Neural Information Processing Systems*, 7, pp. 625–632.

Fritzke, B., 1998. *Vektorbasierte Neuronale Netze*. Shaker, Aachen.

Gschwandtner, M., Kwitt, R., Uhl, A. and Preers, W., 2011. BlenSor: blender sensor simulation toolbox. In: *Advances in Visual Computing*, Springer, Berlin, pp. 199–208.

Jutzi, B., 2015. *Methoden zur automatischen Szenencharakterisierung basierend auf aktiven optischen Sensoren für die Photogrammetrie und Fernerkundung*. Habilitation thesis, Institute of Photogrammetry and Remote Sensing, Karlsruhe Institute of Technology (KIT), Karlsruhe.

Kawashima, K., Yamanishi, S., Kanai, S. and Date, H., 2014. Finding the next-best scanner position for as-built modeling of piping systems. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XL-5, pp. 313–320.

Krainin, M., Curless, B. and Fox, D., 2011. Autonomous generation of complete 3D object models using next best view manipulation planning. *Proc. IEEE International Conference on Robotics and Automation*, pp. 5031–5037.

Kriegel, S., Bodenmüller, T., Suppa, M. and Hirzinger, G., 2011. A surface-based next-best-view approach for automated 3D model completion of unknown objects. *Proc. IEEE International Conference on Robotics and Automation*, pp. 4869–4874.

Munkelt, C., Trummer, M., Denzler, J. and Wenhardt, S., 2007. Benchmarking 3D reconstructions from next best view planning. *Proc. IAPR Conference on Machine Vision Applications*, pp. 552–555.

Pito, R., 1999. A solution to the next best view problem for automated surface acquisition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(10), pp. 1016–1030.

Potthast, C. and Sukhatme, G. S., 2014. A probabilistic framework for next best view estimation in a cluttered environment. *Journal of Visual Communication and Image Representation*, 25(1), pp. 148–164.

Rodolà, E., Albarelli, A., Bergamasco, F. and Torsello, A., 2013. A scale independent selection process for 3D object recognition in cluttered scenes. *International Journal of Computer Vision*, 102(1-3), pp. 129–145.

Scott, W. R., Roth, G. and Rivest, J.-F., 2003. View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1), pp. 64–96.

Soudarissanane, S. and Lindenbergh, R., 2011. Optimizing terrestrial laser scanning measurement set-up. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XXXVIII-5/W12, pp. 127–132.

Vasquez-Gomez, J. I., Sucar, L. E. and Murrieta-Cid, R., 2013. Hierarchical ray tracing for fast volumetric next-best-view planning. *Proc. International Conference on Computer and Robot Vision*, pp. 181–187.

Vásquez, J. I. and Sucar, L. E., 2011. Next-best-view planning for 3D object reconstruction under positioning error. In: *Advances in Artificial Intelligence*, Springer, Berlin, pp. 429–442.

Weinmann, M., 2016. *Reconstruction and analysis of 3D scenes – From irregularly distributed 3D points to object classes*. Springer, Cham.

Weinmann, M., Wursthorn, S. and Jutzi, B., 2011. Semi-automatic image-based co-registration of range imaging data with different characteristics. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. XXXVIII-3/W22, pp. 119–124.

Wenhardt, S., Deutsch, B., Hornegger, J., Niemann, H. and Denzler, J., 2006. An information theoretic approach for next best view planning in 3-D reconstruction. *Proc. International Conference on Pattern Recognition*, pp. 103–106.

Wu, Z., Song, S., Khosla, A., Tang, X. and Xiao, J., 2014. 3D ShapeNets for 2.5D object recognition and next-best-view prediction. *CoRR*, arXiv:1406.5670v2 [cs.CV], pp. 1–9.