Autonomous 3D reconstruction, mapping and exploration of indoor environments with a robotic arm

Yiming Wang[1], Stuart James[2], Elisavet Konstantina Stathopoulou[3], Carlos Beltrán-González[4]
Yoshinori Konishi[5] and Alessio Del Bue[1]

***Abstract*—We propose a novel information gain metric that combines hand-crafted and data-driven metrics to address the next best view problem for autonomous 3D mapping of unknown indoor environments. For the hand-crafted metric, we propose an entropy-based information gain that accounts for the previous view points to avoid the camera to revisit the same location and to promote the motion toward unexplored or occluded areas. Whereas for the learnt metric, we adopt a Convolutional Neural Network (CNN) architecture and formulate the problem as a classification problem. The CNN takes as input the current depth image and outputs the motion direction that suggests the largest unexplored surface. We train and test the CNN using a new synthetic dataset based on the SUNCG dataset. The learnt motion direction is then combined with the proposed hand-crafted metric to help handle situations where using only the hand-crafted metric tends to face ambiguities. We finally evaluate the autonomous paths over several real and synthetic indoor scenes including complex industrial and domestic settings and prove that our combined metric is able to further improve the exploration coverage compared to using only the proposed hand-crafted metric.**

***Index Terms*—Computer Vision for Automation; Range Sensing; Deep Learning in Robotics and Automation**

I. Introduction

AUTONOMOUS systems need to perceive their surrounding 3D world in order to navigate and operate safely. Even if 3D reconstruction is a mature technology, less attention has been posed to the problem of efficiently mapping and covering the 3D structure of an unknown space. This task has strong relations to the longstanding *Next Best View* (NBV) problem [3], [14], [22], [19] but with the additional issue of not having any prior knowledge of the environment where the autonomous system is moving. In particular, in this work the autonomous agent is embodied by a robotic arm equipped with a RGBD camera providing both depth and pose. The proposed approach is general and can be extended to other mobile systems [25], [12] and aerial 3D mapping [18].

Regardless various tasks, such as active object recognition [17], [13], [10] and 3D reconstruction [16], [15], [24], [12], [4], the main methodology of NBV is similar: modelling the information gain and selecting the next view with the most informative measurement where additional constraints, such as the motion-related cost and safety, can be applied [16], [25], [24], [12], [4]. Our system follows such paradigm: at each time step the system selects a new pose among a set of candidates that satisfies certain constraints regarding: i) navigation, i.e. the new pose should be reachable from the current pose, and ii) feasible 3D reconstruction, i.e. the new pose should guarantee a minimal Field of View (FoV) overlapping with current pose. The pose with the highest metric that quantifies the information gain will be the next pose.

Modelling the information gain plays a key role in the NBV problem. In this work we propose a novel hand-crafted metric together with a data-driven metric to address the severely ill-posed NBV problem for 3D mapping of an unknown environment. For the hand-crafted information metric, we propose an entropy-based volumetric utility that accounts for historical view points to avoid the camera revisiting the same pose, while attracting the camera to visit voxels with a lower confidence that can be caused by occlusion. We achieve this via a view-dependent visibility descriptor that records the accumulated information gain distributed by viewing directions. Different from inferring the occlusion from spatial hints as in other works [15], [4]; we try to infer the occlusion via the temporal-spatial hint provided by the proposed descriptor. For the data-driven metric, we introduce an approach that avoids to formulate the task as a regression problem or training a 3D CNN, which commonly require large numbers of parameters to optimize along with extensive training data. Instead, we learn the information utility function in a classification setting which needs less training data and computation. We propose to use a more efficient CNN architecture, that takes as input the current depth image and outputs the ranked motion directions as a indication for where to explore. We further propose various strategies to combine the learnt metric with the proposed hand-crafted metric in order to drive the robot out of ambiguous situations when using only the hand-crafted metric. This likely happens in the beginning of the exploration when moving any

Manuscript received: February, 24, 2019; Revised May, 27, 2019; Accepted June, 19, 2019.

This paper was recommended for publication by Editor Cesar Dario Cadena Lerma upon evaluation of the Associate Editor and Reviewers' comments.

[1]Y. Wang and A. Del Bue are with Visual Geometry and Modelling (VGM) Lab, Istituto Italiano di Tecnologia (IIT), Genova, Italy `yiming.wang@iit.it`

[2]S. James is with Center for Cultural Heritage Technology (CCHT), Istituto Italiano di Tecnologia (IIT), Marghera, Italy

[3]E. K. Stathopoulou contributed to this paper while she was a member of the Visual Geometry and Modelling (VGM) Lab, Istituto Italiano di Tecnologia (IIT), Genova, Italy

[4]C. Beltrán-González is with Pattern Analysis and Computer Vision (PAVIS) Department, Istituto Italiano di Tecnologia (IIT), Genova, Italy

[5]Y. Konishi is with OMRON research, Kyoto, Japan.

Digital Object Identifier (DOI): see top of this page.

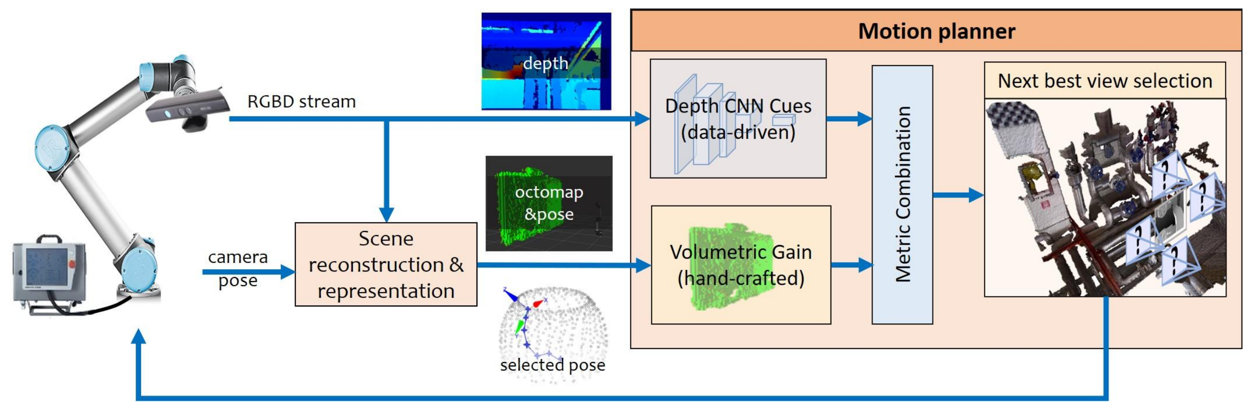


Fig. 1: At each step the pose and depth are provided to the motion planner. The Depth CNN provides a data-driven metric for suggesting the direction while the volumetric gain is a hand-crafted metric. The two metrics are combined to produce the next movement. This process is iterated taking the best of both metrics to maximize surface coverage in as few steps as possible.

directions can be equally good in terms of volumetric gain.

To summarize, our contributions are: 1) A novel entropy based view dependent information gain function, 2) A simple and efficient data-driven CNN that learns the information gain function over depth images, 3) a combined hand-crafted and data-driven technique over time and 4) a synthetic and real dataset with ground truth for NBV problems.

The rest of the paper is organized as follow. Section II discusses the related works on selecting the NBV for 3D exploration and mapping. Section III describes our hand-crafted information gain utility, Section IV describes the CNN structure for learning the motion hints, and Section V details how we combine both utilities. Section VI provides the experimental evaluation and finally Section VII concludes the paper providing perspectives for future work.

II. Related Work

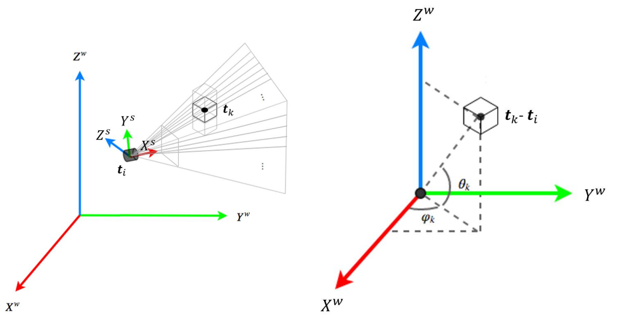
In this section, we will cover related works on NBV for 3D exploration and mapping with particular focus on the modelling of information gain.

NBV without 3D knowledge is an ill-posed problem and how to model the information gain plays an essential role in addressing this task. The modelling of information gain greatly depends on how the 3D environment is represented, which can be categorized as surface-based [2] and volume-based representation [20]. The volumetric representation is often employed for online motion planing for its compactness and efficiency in visibility operations [4]. The status of each voxel can be either occupied, free or unknown. The candidate camera poses for the next move should be in the free voxels and satisfy any kinematic or navigation constraints.

Multiple volumetric information metrics have been proposed for selecting the NBV, often using ray tracing. A common idea is to provide statistics on the number of unknown voxels [16], [25], [24], where one can either count all the unknown voxels encountered [16], or count only the frontier voxels, which are the voxels on the boundary between the known free space and the unexplored space [25]. Occlusion is further taken into account by counting the *occuplane* (a contraction for occlusion plane) voxels that are defined as voxels bordering free and occluded space [24].

In addition to counting-based metrics, there are also metrics based on probabilistic occupancy estimation that accounts for the measurement uncertainty [6], [12],[11]. The main method for computing probabilistic information metrics is based on information entropy [15], [7], [12], [4]. As a ray traverse the map, the information gain of each ray is the accumulated gain of all visible voxels in the form of either a sum [4] or an average [7]. The sum favours views with rays traversing deeper into the map, while the average favours more on the uncertainty of voxels regardless the ray depth. Moreover, inaccurate prediction of the new measurement probability during ray tracing can be an issue for computing the information gain if occlusion is not considered. To address this issue, Potthast and Sukhatme [15] utilize a Hidden Markov Model to estimate the likelihood of an unknown voxel being visible at any viewpoints. The work in [4] accounts for the probability of a voxel being occluded via weighting the information gain by the product of the emptiness probability of all voxels before reaching that voxel. Although the computation can be different, the heuristic behind both [15], [4] is similar, i.e. a voxel with a large unobserved volume between its position and the candidate view position is more likely to be occluded and therefore contributes less information gain.

In addition to the hand-crafted information gain metrics, there is also a recent effort to train a 3D CNN that aims to learn the information utility function [5]. 3D CNN is trained with the known 3D models of the scene, and the utility is defined as the decrease in uncertainty of surface voxels with a new measurement. The goal is that, at run time, the network works as an oracle that can predict the information gain for any input pose given the current occupancy map. Hepp et al. [5] trained and tested the learnt metric with both outdoor synthetic dataset and indoor dataset, showing better mapping performance and faster processing compared to the hand-crafted information gain metrics. While [5] trains a 3D CNN and requires as input the 3D reconstruction status, we instead propose a 2D CNN to learn the information gain function from a single-shot depth image and combine the learnt metric with the hand-crafted



(a) (b)

Fig. 2: Illustration of visibility modelling. (a) A sensor with the frustum-shaped FoV located at $\mathbf{t}_i$ views the k^{th} voxel at position $\mathbf{t}_k$ in the world coordinate $W = (X^w,\ Y^w,\ Z^w)$. (b) The horizontal view angle φ_k and vertical view angle θ_k for the k^{th} voxel with respect to the sensor position.

metric which encodes the reconstruction status.

III. Hand-crafted volumetric metric

Our hand-crafted utility is designed based on octomap [6], a probabilistic 3D volumetric representation. The possible space is quantized into voxels that are associated with an occupancy probability $p_k \in [0,\ 1]$, at each time step p_k is updated with the new range measurement coming from the depth image ($\mathbf{D}$). Based on the occupancy probability, each voxel is thus expressed as having one of three states: Unknown, Free and Occupied. A higher p_k value indicates that the voxel is more likely to be occupied, while the lower indicates a higher likelihood to be empty.

Our reconstruction and mapping framework requires as input a RGBD frame (e.g. as given by a Kinect sensor) and a calibrated sensor pose that can be given either by a Simultaneous Localization and Mapping (SLAM) method or from a robotic arm position encoders. We prefer the latter in this work since it provides a high accuracy positions for testing the motion policy.

For each candidate pose $\mathbf{p}_i = [\mathbf{t}_i, \mathbf{R}_i]$ with $\mathbf{t}_i$ for position and $\mathbf{R}_i$ for rotation, we compute a information gain utility by tracing rays from the candidate pose to the space in a discretized manner within the FoV. The utility of candidate pose aggregates the utility of all rays as an average, where the utility of each ray is computed as the averaged utility of all visible voxels along the ray.

A. View-dependent descriptor

Each voxel in the space is associated with two descriptors, $\mathbf{O}_k$ and $\mathbf{C}_k$, that record the status of voxels over the viewing directions. The value $\mathbf{O}_k$ stores the probability of the voxel being occupied and $\mathbf{C}_k$ stores the accumulated confidence of the voxel's status being either occupied or empty.

The descriptors will only be updated when the voxel is visible to the sensor at a given pose. A voxel is considered visible if it lies within the sensor's FoV (modelled as a frustum) by checking its distance and angle difference [23] (see Fig. 2(a)). We discretize the horizontal and vertical angles to index the view direction [23]. If k_{th} voxel is visible from the sensor at a position with φ_k and θ_k (see Fig. 2(b)), the two-dimensional index $\mathbf{i}_k = [i_k^h,\ i_k^v]$ is computed as $i_k^h = \lfloor \frac{\varphi_k}{\delta} \rfloor$ and $i_k^v = \lfloor \frac{\theta_k}{\delta} \rfloor$, where δ is the angle step that is set to $\frac{\pi}{18}$ and $\lfloor \cdot \rfloor$ gives the floor value. We can update our descriptors at a certain sensor pose indexed by $\mathbf{i}_k$ as:

$$\mathbf{O}_k[\mathbf{i}_k] = p_k \tag{1}$$

$$\mathbf{C}_k[\mathbf{i}_k] = \mathbf{C}_k[\mathbf{i}_k] + c(p_k), \tag{2}$$

where the function $c(p_k) = 1 + p_k \log(p_k) + (1-p_k)\log(1-p_k)$ computes the confidence based on information entropy.

B. View-aware utility

We compute the information gain utility of each candidate pose based on the view-dependent descriptors. We aim to assign the highest utility to voxels that have never been visited at any poses, and to decrease the utility as the accumulated confidence of the voxel increases.

For the visible k_{th} voxel that has not been seen before, the utility $u_k[\mathbf{i}_k]$ corresponding to the sensor pose indexed by $\mathbf{i}_k$ is set to be 1. For the visible k_{th} voxel that has been seen before, we define the utility $u_k[\mathbf{i}_k]$ as:

$$u_k[\mathbf{i}_k] = \begin{cases} (\sum_{\mathbf{C}_k})^{-1}, & \mathbf{C}_k[\mathbf{i}_k] = 0 \\ \frac{c(\mathbf{O}_k[\mathbf{i}_k])}{\sum_{\mathbf{C}_k}}, & otherwise \end{cases} \tag{3}$$

where $\sum_{\mathbf{C}_k}$ is the sum of the accumulated confidences at all view directions. The utility $u_k[\mathbf{i}_k]$ is set to 1 if voxel v_k is not viewed at the current pose and discounted by the sum of confidences.

The final information gain utility given a certain sensor pose, $u_g(\mathbf{p}_i)$, is computed by tracing rays in the 3D space that discretize the FoV as shown in Fig. 2 (a). The angle step between consecutive rays is set to $\frac{\pi}{400}$ for both horizontal and vertical directions. Each ray is associated with a visibility utility that is the *average* of the utilities of voxels that are within the visibility range. The visibility range starts at the minimum sensor range and ends at the first occupied voxel or the maximum range. In the case when a ray traverses an unexplored area, the maximum utility, i.e. 1, is associated to the ray. The final utility $u_g(\mathbf{p}_i)$ is the *average* of the utilities associated to all rays.

IV. Learning motion directions

Our data-driven metric is inspired by prior works on objectness [1], [9] that identify possible meaningful structure in an image. In contrast, we utilize *depth* images to learn a function u_h to maximize the information gain and in turn predict a direction label for the next move. Although this does not maintain a history of the space being reconstructed, it is efficient to compute at test time and avoids duplicating the memory over ahead already stored within the octomap. We therefore pose this problem as a classification task on the direction labels using the architecture of AlexNet [8] over depth images. This strategy makes the problem tractable than regressing the movements, it also removes the need for modelling kinematics within the possible solution space, as invalid direction can simply be removed afterwards.

A. CNN structure

The model consists of five convolutional layers with three max pool and nonlinear layers, as per the original AlexNet model. We adjust the two fully connected and classification layers to output the six directional classes with soft max. To learn u_h, we define a function f as the AlexNet model learnt from a constructed synthetic training set on SUNCG [21]. The SUNCG dataset consists of over 45K synthetic indoor scenes and each scene has rooms and furniture layouts that are manually created in a realistic manner, which was validated by mechanical turk. All the scenes are semantically annotated at the object level.

During training, we create a set of views and render their corresponding depth images. In addition we render the six adjacent views corresponding to the directional labels (see Fig. 3), creating a set of candidate poses $\mathbf{p}_j \in \Omega_i$. As SUNCG provides the 3D models of objects, we can compute the volumetric information gain as the number of visible surface voxels from each view. The label y, therefore corresponds to $y_i = \arg\max(G(\mathbf{p}_i))$, where G is the information gain function.

We opt not to use an initialized model, for example from ImageNet, to avoid semantic class bias and only use the synthetic object surfaces captured by depth during training. We therefore minimize the cross-entropy loss over the training set $\mathbf{D} = 1...N$ where N is the number of training samples formulated as:

$$\mathcal{L}(w) = \sum_{i=0}^{N}\sum_{c=0}^{6} -y_{ic}\log f_c(\mathbf{D}_i). \tag{4}$$

At test time given $\mathbf{D}$, we compute the direction label as $y(\mathbf{D}) = \arg\max(f(\mathbf{D}))$.

It is worth noting that this model will contain no temporal information, but only an indication to the direction to maximize the information gain. Although including a temporal model, e.g. LSTM or Markov Model, would allow this information to be propagated over, our goal is only to provide an indication to the path planning and avoid any domain specific training of a motion category.

Implementation Details: We follow the standard hyper parameter settings for learning on an AlexNet model; learning rate $1e^{-2}$, 10 epochs, momentum 0.9 and weight decay $5e^{-4}$. Note that given the more limited training data, convergence occurs earlier than when training on ImageNet data. We also have reduced number of weights with only one input channel.

B. Dataset Generation for training/validation

In order to train the 2D CNN, we created a dataset based on the SUNCG [21]. To train our model, we first created 7920 camera poses and their corresponding RGB and depth images which contain a minimum of three objects in each view. We constrain the generation based on normalized best motion direction, this removes the influence of a singular motion (most commonly *backwards* due to the label function) being selected and resulting in training bias.

We compute the corresponding candidate movement poses from each of the generated sensor poses by translating the camera to six directions by a distance d_δ. We set the translation distance d_δ to $0.1\ m$ in order to maintain 80% overlap between the current and next depth images[1]. As only translation is of concern, we only need to compute the position vector of the candidate poses:

$$\begin{cases} \mathbf{t}_i^u &= \mathbf{t}_i + d_\delta \mathbf{e}_i^u \\ \mathbf{t}_i^d &= \mathbf{t}_i - d_\delta \mathbf{e}_i^u \\ \mathbf{t}_i^f &= \mathbf{t}_i + d_\delta \mathbf{e}_i^f \\ \mathbf{t}_i^b &= \mathbf{t}_i - d_\delta \mathbf{e}_i^f \\ \mathbf{t}_i^r &= \mathbf{t}_i + d_\delta \mathbf{e}_i^f \times \mathbf{e}_i^u \\ \mathbf{t}_i^l &= \mathbf{t}_i - d_\delta \mathbf{e}_i^f \times \mathbf{e}_i^u \end{cases} \tag{5}$$

where the symbol $\times$ represents the cross product operation and $\mathbf{e}_i^f \times \mathbf{e}_i^u$ gives the unit vector for moving right.

The final step is the generation of the motion label(y) for each camera pose. We exploit the ground-truth knowledge of the scene volume and select the next camera pose with the most occupied volume. We quantify the volumetric gain as the difference of the number of visible occupied voxels between the current camera pose and the next pose. One example for demonstrating the dataset generation is shown in Figure 3. Given a camera pose in a scene, we show the resulted depth (the first row) and RGB (the second row) images by moving the current camera pose to the six directions. By checking the resulted volumetric gain, the motion label is set to 'Backward'.

CNN evaluation: We evaluate on the SUNCG synthetic dataset by performing an 80-20 split between training and validation, therefore measuring the performance of the data-driven given the perfect case and non-continuous. For fair comparison, we perform 3-fold cross-validation and average the training and validation accuracy. We achieve an average accuracy for the training of 82.4% and a validation accuracy of 64.3%.

V. Combining Metrics

Let $\mathbf{p}_j \in \Omega_i$ be the set of reachable candidate poses of current pose $\mathbf{p}_i$. For selecting the NBV, we compute a final utility, $u(\mathbf{p}_j, \mathbf{D}_i)$ for each candidate pose that combines both the utility $u_g(\mathbf{p}_j)$ based on the hand-crafted gain metric and the utility $u_h(\mathbf{D}_i)$ computed based on data-driven direction indications.

The utility $u_h(\mathbf{p}_j, \mathbf{D}_i)$ includes the suggested direction label that is generated by CNN. To quantify the utility in range $[0,\ 1]$, we compute the projection between the unit vector of the suggested direction and the unit vector from the current pose $\mathbf{p}_i$ to the candidate pose $\mathbf{p}_j$. The pose with the most similar direction to the suggested one has the highest utility.

Let $\mathbf{e}_i^* = \mathbf{R}_i\mathbf{e}^*$ be the unit vector of the CNN-suggested direction at the current pose, where the superscript $*$ corresponds to one of the six directions. Define $\Delta\mathbf{e}_{j,i} = \frac{\mathbf{t}_j - \mathbf{t}_i}{|\mathbf{t}_j - \mathbf{t}_i|_2}$ as the unit difference vector of the position of candidate pose $\mathbf{t}_j$ and

[1]The overlap of consecutive depth camera FoV is necessary to obtain 3D reconstruction with our method described in Sec VI-A

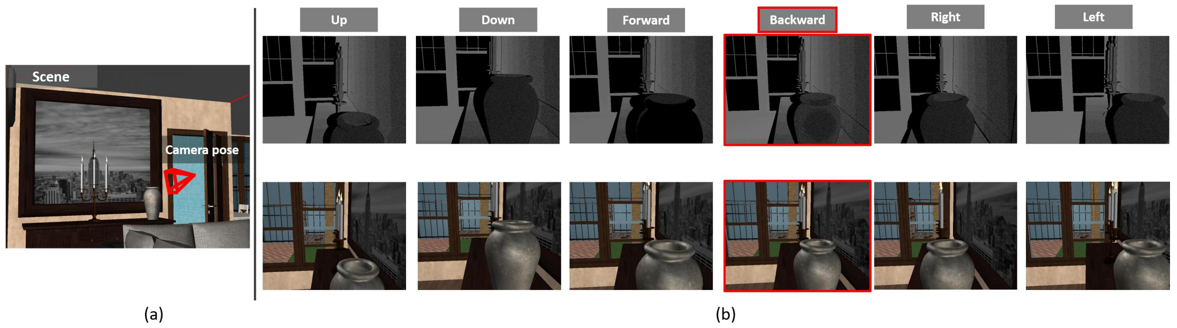


Fig. 3: Example of a depth camera in a SUNCG scene. (a) The scene and the camera pose. (b) The resulting depth (the first row) and RGB (the second row) images given by moving the camera pose to the six directions. The motion label is selected as 'Backward' as this motion results in the highest volumetric gain (best viewed in color).

the position of current pose $\mathbf{t}_i$. The smaller angle difference between $\Delta\mathbf{e}_{j,i}$ and $\mathbf{e}_i^*$ should indicate a higher utility, we therefore design the utility function based on the dot product as

$$u_h(\mathbf{p}_j, \mathbf{D}_j) = \frac{\Delta\mathbf{e}_{j,i}\mathbf{e}_i^* + 1}{2}. \qquad (6)$$

When $\Delta\mathbf{e}_{j,i}$ coincides with $\mathbf{e}_i^*$, $u_h(\mathbf{D}_j)$ is valued 1, whereas when $\Delta\mathbf{e}_{j,i}$ is at the opposite direction of $\mathbf{e}_i^*$, $u_h(\mathbf{D}_j)$ is valued 0.

We attempted to combine the utilities in four different manners. The guiding design principle in this combination was that the CNN should influence the NBV only in cases where using the information utility alone might be ambiguous. We avoid being dependent on the learnt motion indications while not necessary as the learning process does not encode temporal information.

Naive Switching: The combined utility $u(\mathbf{p}_j, \mathbf{D}_i)$ follows a switching policy which is rather naive. We anticipate that the gain utility may encounter difficulties at the initial exploration stage because a similar gain may be resulted from moving in any direction. In such cases, the system selects the first poses purely based on the learnt utility $u_h(\mathbf{p}_j, \mathbf{D}_i)$ while afterwards switching to the gain utility $u_g(\mathbf{p}_j)$ as:

$$\begin{cases} u(\mathbf{p}_j, \mathbf{D}_i) = u_h(\mathbf{D}_i), & t \leq \tau \\ u(\mathbf{p}_j, \mathbf{D}_i) = u_g(\mathbf{p}_j), & t > \tau. \end{cases} \qquad (7)$$

Weighted Average: Following the similar motivation that the gain utility may encounter difficulties at the initial stage, we combined utility as the weighted average of $u_g(\mathbf{p}_j)$ and $u_h(\mathbf{p}_j, \mathbf{D}_i)$ with the weight decaying with time t:

$$u(\mathbf{p}_j, \mathbf{D}_i) = (1 - \alpha(t))u_g(\mathbf{p}_j) + \alpha(t)u_h(\mathbf{p}_j, \mathbf{D}_i), \qquad (8)$$

where $\alpha(t)$ is the weight with a decaying function that is the inverse of t.

Temporal Conditioning: With temporal conditioning, we switch between the gain utility and learnt utility autonomously by detecting the ambiguous cases. We determine whether to switch by checking the tendency of the gain utility and the difference between the next pose suggested by the two utilities individually within a small time window. In the cases when 1) gain utility continuously decreases and in the mean time 2) the learnt utility continuously suggests a different direction compared to the gain utility (c_2), we switch to using learnt utility otherwise the gain utility will be used. The temporal window is set to 3 time steps. We set the boolean variable c_1 to 1 when condition 1) is satisfied otherwise to 0. The same rule applies to boolean variable c_2.

The switching based on temporal conditioning is given by:

$$\begin{cases} u(\mathbf{p}_j, \mathbf{D}_i) = u_h(\mathbf{p}_j, \mathbf{D}_i), & c_1 = 1 \ \& \ c_2 = 1 \\ u(\mathbf{p}_j, \mathbf{D}_i) = u_g(\mathbf{p}_j), & \textit{otherwise}. \end{cases} \qquad (9)$$

Weighted Average & Temporal Conditioning: We further combine the Weighted Average and Temporal Conditioning in the following manner: At initial steps, the NBV is selected based on the combined utility computed using Weighted Average strategy, and after the combined utility will be computed with the Temporal Conditioning strategy.

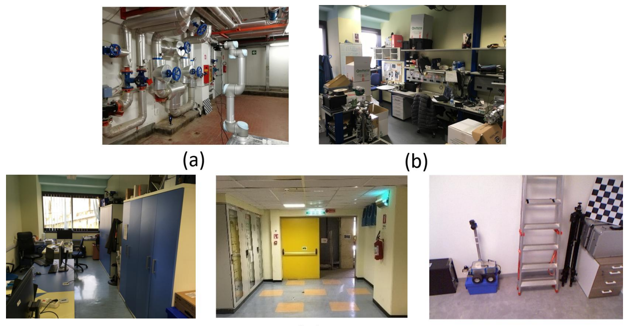


Fig. 4: Five real indoor rooms for the experiments. (a) maintenance room (mroom), (b) laboratory (lab), (c) corridor, (d) office and (e) storage room (storage).

VI. Experiments

We perform two types of experiments based on synthetic data (Fig. 4) and real world data captured in four complex environments (Fig. 5). We outline the creation of the dataset and then perform studies on the exploration performance using different strategies based on the coverage of the space.

A. Dataset creation

We create a real world dataset containing representative data of three indoor environments: industrial, laboratory and office.

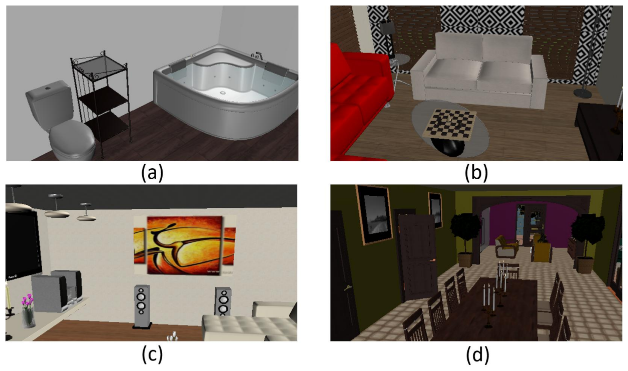


Fig. 5: Four synthetic indoor rooms for the experiments. (a) room1 (b) room2 (c) room3 and (d) room4.

Within these general environment categories, we collected data in five concrete scenarios: (i) a maintenance room (mroom), (ii) an electronics laboratory (lab), (iii) an office (office), (iv) a corridor with control panels (corridor) and (v) a small storage room (storage) (see Fig. 4).

We adopted the same pipeline for data acquisition throughout all scenarios. The RGBD data with poses, provided by the robotic arm were collected using a Kinect v1 mounted on the end-effector of a UR5 robotic arm. By following a dense dome-shaped path the captured data of the whole indoor environment was reconstructed using Intel Open3D [26] libraries to generate a 3D point cloud. Laser scanning data were also acquired as ground truth (GT) information to evaluate the reconstruction accuracy while mapping the scene. A Leica C10 laser scanner was placed in multiple positions along the scenes in order to acquire dense 3D point clouds from multiple viewpoints. From the laser scan data the GT was generated by aligning and clean 3D mesh through merging the point clouds of multiple acquisitions. When compared against GT point clouds, the 3D reconstruction error, i.e. the mean cloud to cloud absolute distance, of the fixed dome-shape path lies within the range of $[4, 7]$ cm.

The four synthetic indoor housing environments are rendered using the SUNCG dataset (see Fig. 5). Depth and RGB images are rendered following the same dome-shaped path as in the real room dataset. The initial height of the camera is chosen at 1.25 m which is similar to the real configuration of our table-mounted robotic arm. Regarding GT point clouds, the SUNCG dataset has CAD models available for each indoor environment, thus the evaluation of reconstruction accuracy can be performed.

B. Evaluation and analysis

We evaluate our four attempted Combined Gain (*CGain*) metrics against Random, i.e. randomly selecting a next pose among candidate neighbouring poses, a probability-based information gain using the standard octomap structure [11] (*BaseGain-P*), our entropy-based information gain using the standard octomap structure without view-dependent descriptors (*BaseGain-E*), Information Gain Metric (*InfoGain*), Learnt Metric (*CNN*) and *InfoGain* combined with *Random* using Weighted Average (*CGain-Rand*) on both the synthetic and real scenes. Our four Combined Gain metrics presented in Sec. V are (i) Weighted Average (*CGain-WA*), (2) Naive Switching (*CGain-NS*) with $\tau = 10$, (3) Temporal Conditioning (*CGain-TC*) and Weighted Average & Temporal Conditioning (*CGain-WATC*).

The exploration performance is quantified as the coverage ratio of the number of the surface voxels generated using autonomous methods against the number of surface voxels produced using the predefined dense dome-shaped path, i.e. an exhaustive sampling of the camera pose space. We consider that the dome-shaped path explores the complete room, i.e. the coverage ratio is 100% with 672 poses and implicitly contains the constraints regarding the NBV reachability and FoV overlapping for 3D reconstructions.

As *Random*, the Learnt Metric and some Combined Gains have no suitable stopping criteria due to no memory of current status of exploration, we evaluate for all methods until a fixed number of steps (i.e. 130). The number is set according to *InfoGain* metric which in all cases was invoked earlier when the camera starts looping within a small area (i.e. ending up in a local minima). The metric that achieves a larger coverage ratio within the fixed number of steps is considered more efficient.

We present in Table I the results of all metrics and Fig. 7(a) shows the exploration coverage ratio over time. The results of real rooms and synthetic rooms are averaged over 5 independent runs. The coverage ratio given by *InfoGain* outperforms Random and the two *BaseGain* on average. The view-dependent descriptors weight the metric based on the view history thus being able to drive the robot with constant coverage improvement, while the two *BaseGain* methods saturate due to being stuck at a local area (see Fig. 7(a)). With *InfoGain* only, the coverage ratio can already reach a satisfactory performance of 85% on average and with the largest coverage ratio of 99% and the lowest of 52%. *InfoGain* tends to end up in local minima when the camera revisits the already explored area. In general, with *InfoGain* only, the average coverage ratio achieved in synthetic rooms is 86%, which is higher than the coverage ratio in real rooms is 84%. A similar trend is observed when using other *InfoGain*-based metrics which shows that the effectiveness of the *InfoGain* can be affected when the scene is more complex. The strategy *CNN* only relies on the motion hints which often results in a much smaller coverage ratio as the *CNN* does not encode any temporal memory, thus tends to being stuck in a local area as shown in Fig. 7(a).

In general, we do observe coverage improvements in most rooms when applying the Combined Gain with *CGain-WA* and *CGain-NS*. This proves our hypothesis that the learnt utility can help in the initial steps when the hand-crafted gain can be similar moving at any directions as the scene is mostly unexplored. The *CGain-TC* metric shows some potential in outperforming *InfoGain* with a small margin while the improvement is less dominant compared to *CGain-WA* and *CGain-NS*. While we noticed that *CGain-Rand* can achieve the best coverage in two rooms among all metrics, the inclusion of the learnt utility do tend to provide a higher coverage improvement for the remaining rooms (six over nine).

TABLE I: Exploration 3D coverage ratio (with standard deviation) achieved with a fixed number of steps (i.e. 130) using various information metrics under different synthetic and real indoor scenes (higher is better).

	Random	*BaseGain-P*	*BaseGain-E*	*InfoGain*	*CNN*	*CGain-Rand*	*CGain-TC*	*CGain-WATC*	*CGain-NS*	*CGain-WA*
room1	0.49 (0.20)	0.22 (0.15)	0.52 (0.10)	0.89 (0.20)	0.26 (0.10)	0.91 (0.02)	0.89 (0.20)	0.94 (0.04)	0.93 (0.04)	**0.99 (0.03)**
room2	0.53 (0.15)	0.28 (0.04)	0.45 (0.11)	0.83 (0.10)	0.15 (0.04)	0.83 (0.03)	0.83 (0.10)	0.86 (0.04)	0.84 (0.08)	**0.87 (0.07)**
room3	0.52 (0.10)	0.24 (0.09)	0.48 (0.01)	0.89 (0.05)	0.19 (0.07)	0.80 (0.02)	0.86 (0.05)	0.87 (0.04)	**0.90 (0.03)**	0.83 (0.04)
room4	0.58 (0.18)	0.23 (0.13)	0.53 (0.04)	0.84 (0.13)	0.24 (0.08)	0.81 (0.02)	0.85 (0.13)	**0.90 (0.11)**	0.82 (0.26)	**0.90 (0.10)**
mroom	0.39 (0.11)	0.32 (0.08)	0.42 (0.06)	0.78 (0.04)	0.29 (0.10)	0.80 (0.02)	0.78 (0.06)	0.82 (0.05)	0.79 (0.02)	**0.84 (0.03)**
lab	0.35 (0.16)	0.49 (0.07)	0.45 (0.11)	0.86 (0.04)	0.25 (0.10)	0.85 (0.02)	**0.89 (0.05)**	0.79 (0.06)	0.83 (0.06)	0.84 (0.05)
office	0.49 (0.11)	0.20 (0.09)	0.20 (0.11)	0.87 (0.03)	0.31 (0.09)	**0.89 (0.02)**	**0.89 (0.04)**	0.85 (0.09)	**0.89 (0.04)**	**0.89 (0.04)**
corridor	0.44 (0.21)	0.10 (0.03)	0.16 (0.11)	0.83 (0.06)	0.43 (0.06)	**0.86 (0.04)**	0.81 (0.04)	0.83 (0.03)	0.84 (0.03)	0.72 (0.24)
storage	0.43 (0.18)	0.29 (0.15)	0.44 (0.04)	0.87 (0.20)	0.40 (0.07)	**0.99 (0.01)**	0.85 (0.18)	0.97 (0.03)	0.81 (0.10)	0.84 (0.22)

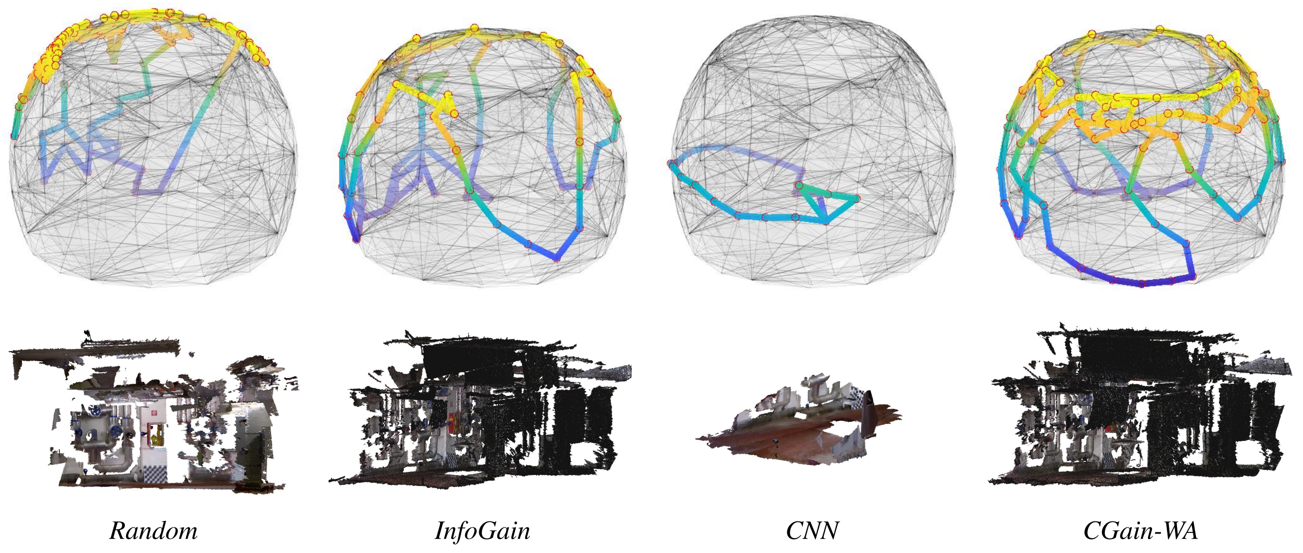


Fig. 6: The selected poses and their corresponding 3D reconstruction in maintenance room. Row 1: the selected poses on the dome surface; Row 2: 3D reconstruction result in colored point cloud (best viewed in color).

When using *GGain-WA* and *GGain-WATC*, the coverage ratio can be improved by up to 10% compared to using *InfoGain* only. Note that when the scene is small with simple structure, e.g. the synthetic room1 and the real storage room, the coverage ratio can reach 100% when using the combined gain *CGain-WA* and *GGain-WATC* (see the supplementary video for the demo with a UR5 robotic arm). For more complex rooms, although we observe the coverage improvement is lower, the combined *CGain-WA* can still achieve up to 6% coverage improvement in the maintenance room which has challenging structures and reflective surfaces.

The reconstruction error of the autonomous paths on average lies within the range of $[4, 7]$ cm which is similar to the error achieved when following the fixed dense dome-shape path. In terms of computation time analysis, we performed experiments using a Dell Alienware Aurora with core i7, the processing time break-down for 3D reconstruction including volume integration and point cloud extraction, octomap update and NBV selection is shown in Fig. 7(b). The processing time of all modules increases as the scene gets completed with the voxel size of 2.5 cm. The NBV module takes around 0.8 s for computing the hand-crafted and learnt utilities and selecting the next pose using standard parallelization techniques. The processing time for CNN-learnt metric is 0.005 s, which is negligible.

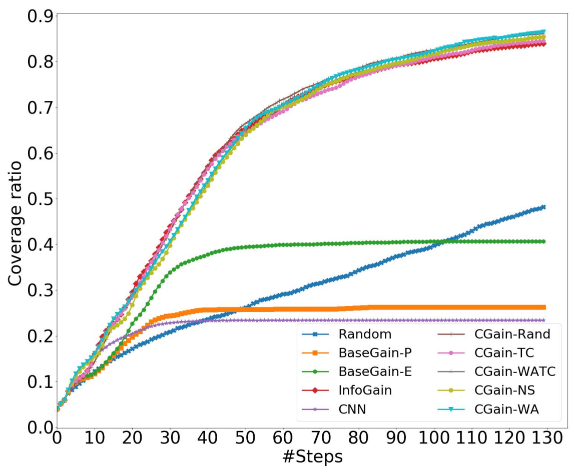
As an example, we show in Fig. 6 the resulted paths on a dome-surface in the maintenance room together with their corresponding 3D reconstruction results in colored point clouds. The path generated with Random metric fails to scan the complete space, where the *InfoGain* metric is able to drive the robotic arm to visit most of the space, although it gets stuck easily in local minima. Instead, the combined gain *CGain-WA* is able to continuously drive the arm to visit more space resulting in a better coverage.

We have implemented an online system on ROS that takes RGBD stream and performs the NBV selection among candidate poses in real time. Existing MoveIt[2] libraries are exploited to update octomap and ensure path feasibility by checking collisions with the robot itself and its surroundings. At run time, the 3D reconstruction and octomap update can take up to 2 s depending on the reconstruction density, while the NBV module maintains around 0.8 s.

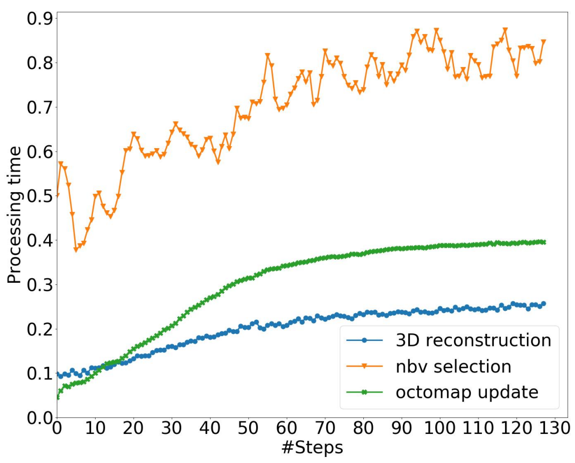
VII. Conclusion

In this paper we have proposed a solution that takes advantage of hand-crafted and data-driven metric to address the NBV problem within unconstrained environments. We have shown by experiments with synthetic and real dataset that the data-driven metric is more advantageous in the beginning of exploration while later the hand-crafted metric is able to

[2]https://moveit.ros.org/



(a)



(b)

Fig. 7: Coverage ratio (a) and processing time break-down (b) over time averaged over all rooms (best viewed in color).

complete the scene more effectively. With the proposed hand-crafted metric only, the system achieves on average 85% coverage. The combined metric can further improve the coverage achieved with the hand-crafted metric up to 10% with the improvement margin depending on the complexity of rooms. The data-driven metric is not aware of the reconstruction status, i.e. the metric cannot guarantee to help the system out of ambiguous situations or local minima. As future work, we will further improve the learnt metric by encoding the status of local reconstruction into the network.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, Nov. 2012. 3

[2] R. Border, J. D. Gammell, and P. Newman. Surface edge explorer (see): Planning next best views directly from 3d observations. In *Proc. of IEEE Int'l Conf. on Robotics and Automation (ICRA)*, pages 1–8, May 2018. 2

[3] C. Connolly. The determination of next best views. In *Proc. of IEEE Int'l conf. on Robotics and automation (ICRA)*, volume 2, pages 432–435, March 1985. 1

[4] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza. A comparison of volumetric information gain metrics for active 3d object reconstruction. *Autonomous Robots*, 42(2):197–208, Feb. 2018. 1, 2

[5] B. Hepp, D. Dey, S. N. Sinha, A. Kapoor, N. Joshi, and O. Hilliges. Learn-to-score: Efficient 3d scene exploration by predicting view utility. In *Proc. of European Conf. on Computer Vision (ECCV)*, pages 455–472, Sep. 2018. 2

[6] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, April 2013. 2, 3

[7] S. Kriegel, C. Rink, T. Bodenmüller, and M. Suppa. Efficient next-best-scan planning for autonomous 3d surface reconstruction of unknown objects. *J. Real-Time Image Proc.*, 10(4):611–631, Dec. 2015. 2

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. of Int'l Conf. on Neural Information Processing Systems (NIPS)*, pages 1097–1105, Dec. 2012. 3

[9] L. Li, W. Feng, L. Wan, and J. Zhang. Maximum cohesive grid of superpixels for fast object localization. In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3174–3181, June 2013. 3

[10] M. Malmir and G. W. Cottrell. Belief tree search for active object recognition. In *Proc. of IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems (IROS)*, pages 4276–4283, Sep. 2017. 1

[11] Z. Meng, H. Qin, Z. Chen, X. Chen, H. Sun, F. Lin, and M. H. A. Jr. A two-stage optimized next-view planning framework for 3-d unknown environment exploration, and structural reconstruction. *IEEE Robotics and Automation Letters*, 2(3):1680–1687, July 2017. 2, 6

[12] E. Palazzolo and C. Stachniss. Effective exploration for mavs based on the expected information gain. *Drones*, 2(1), March 2018. 1, 2

[13] T. Patten, W. Martens, and R. Fitch. Monte Carlo planning for active object classification. *Autonomous Robots*, 42(2):391–421, Feb. 2018. 1

[14] R. Pito. A solution to the next best view problem for automated surface acquisition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(10):1016–1030, Oct. 1999. 1

[15] C. Potthast and G. S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *J. of Visual Communication and Image Representation*, 25(1):148 – 164, Jan. 2014. 1, 2

[16] P. Quin, G. Paul, A. Alempijevic, D. Liu, and G. Dissanayake. Efficient neighbourhood-based information gain approach for exploration of complex 3d environments. In *Proc. of IEEE Int'l Conf. on Robotics and Automation (ICRA)*, pages 1343–1348, May 2013. 1, 2

[17] A. Roberti, M. Carletti, F. Setti, U. Castellani, P. Fiorini, and M. Cristani. Recognition self-awareness for active object recognition on depth images. In *Proc. of British Machine Vision Conf. (BMVC)*, page 15, Sep. 2018. 1

[18] M. Roberts, S. Shah, D. Dey, A. Truong, S. N. Sinha, A. Kapoor, P. Hanrahan, and N. Joshi. Submodular trajectory optimization for aerial 3d scanning. In *Proc. of IEEE Int'l Conf. on Computer Vision (ICCV)*, pages 5334–5343, Oct. 2017. 1

[19] K. Schmid, H. Hirschmüller, A. Dömel, I. Grixa, M. Suppa, and G. Hirzinger. View planning for multi-view stereo 3d reconstruction using an autonomous multicopter. *J. of Intelligent & Robotic Systems*, 65(1):309–323, Jan. 2012. 1

[20] W. R. Scott, G. Roth, and J.-F. Rivest. View planning for automated three-dimensional object reconstruction and inspection. *ACM Comput. Surv.*, 35(1):64–96, March 2003. 2

[21] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[22] H. Surmann, A. Nchter, and J. Hertzberg. An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3–4):181–198, Dec. 2003. 1

[23] M. Tamassia, A. Farinelli, V. Murino, and A. Del Bue. Directional visual descriptors and multirobot strategies for largescale coverage problems. *J. of Field Robotics*, 33(4):489–511, July 2015. 3

[24] J. I. Vasquez-Gomez, L. E. Sucar, R. Murrieta-Cid, and E. Lopez-Damian. Volumetric next-best-view planning for 3d object reconstruction with positioning error. *Int'l J. of Advanced Robotic Systems*, 11(10):159, Oct. 2014. 1, 2

[25] J. Wettach and K. Berns. Dynamic frontier based exploration with a mobile indoor robot. In *Proc. of Int'l Symposium on Robotics (ISR) and German Conf. on Robotics (ROBOTIK)*, pages 1–8, June 2010. 1, 2

[26] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, Jan. 2018. 6