

University of Parma Research Repository

Surfel-Based Next Best View Planning

This is the peer reviewd version of the followng article:

Original

Surfel-Based Next Best View Planning / Monica, Riccardo; Aleotti, Jacopo. - In: IEEE ROBOTICS AND AUTOMATION LETTERS. - ISSN 2377-3766. - 3:4(2018), pp. 3324-3331. [10.1109/LRA.2018.2852778]

Availability: This version is available at: 11381/2849407 since: 2021-10-11T10:59:27Z

*Publisher:* Institute of Electrical and Electronics Engineers Inc.

Published DOI:10.1109/LRA.2018.2852778

Terms of use:

Anyone can freely access the full text of works made available as "Open Access". Works made available

Publisher copyright

note finali coverpage

(Article begins on next page)

# Surfel-Based Next Best View Planning

Riccardo Monica D and Jacopo Aleotti

Abstract-Next best view (NBV) planning is a central task for 3 4 automated three-dimensional (3-D) reconstruction in robotics. The most expensive phase of NBV computation is the view simulation 5 step, where the information gain of a large number of candidate 6 sensor poses are estimated. Usually, information gain is related 7 to the visibility of unknown space from the simulated viewpoint. 8 A well-established technique is to adopt a volumetric representa-9 10 tion of the environment and to compute the NBV from ray casting by maximizing the number of unknown visible voxels. This letter 11 explores a novel approach for NBV planning based on surfel repre-12 13 sentation of the environment. Surfels are oriented surface elements, such as circular disks, without explicit connectivity. A new kind of 14 surfel is introduced to represent the frontier between empty and 15 16 unknown space. Surfels are extracted during 3-D reconstruction, with minimal overhead, from a KinectFusion volumetric repre-17 sentation. Surfel rendering is used to generate images from each 18 simulated sensor pose. Experiments in a real robot setup are 19 20 reported. The proposed approach achieves better performance than volumetric algorithms based on ray casting implemented on 21 22 GPU, with comparable results in terms of reconstruction quality. Moreover, surfel-based NBV planning can be applied in larger 23 environments as a volumetric representation is limited by GPU 24 25 memory.

*Index Terms*—Autonomous agents, range sensing, motion and path planning, computer vision for other robotic applications.

# I. INTRODUCTION

NEXT Best View (NBV) algorithm computes the best 29 viewpoint of a depth sensor, mounted on a robot, to im-30 prove the knowledge of the environment by maximizing the 31 expected information gain. Typical NBV algorithms operate it-32 33 eratively in two phases: viewpoint generation and viewpoint evaluation. In the first phase, the free configuration space of the 34 robot is explored to retrieve a set of candidate sensor poses. 35 In the second phase, a view is simulated from each candidate 36 sensor pose, given the current model of the environment, and 37 38 then the most promising viewpoint is selected. The goal of the simulation is to estimate the amount of unknown space visible 39 from the view pose, which in turn predicts the information gain. 40 The view simulation phase is usually the most computationally 41 expensive operation, which may limit the number of poses that 42 can be evaluated in a reasonable time. 43

44 Unlike most works that address the NBV problem by using 45 a volumetric representation of the environment, in this work a

Manuscript received February 23, 2018; accepted June 17, 2018. Date of publication; date of current version. This letter was recommended for publication by Associate Editor A. Argyros and Editor T. Asfour upon evaluation of the reviewers' comments. (*Corresponding author: Riccardo Monica.*)

The authors are with the RIMLab, Department of Engineering and Architecture, University of Parma, Parma I-43100, Italy (e-mail: riccardo.monica@ unipr.it; jacopo.aleotti@unipr.it).

Digital Object Identifier 10.1109/LRA.2018.2852778

1

Fig. 1. (a) RGB image of the scene. (b) Incomplete surfel-based representation, with surface surfels (gray shading) and frontier surfels (frontels, black). (c) A closeup view. (d) Closeup view with surfel size reduced to 20%.

novel NBV evaluation strategy is proposed that exploits a surfel-46 based representation. A surfel is a surface element represented 47 as a circular disk without explicit connectivity, described by 48 geometric attributes such as position, radius, normal and color 49 [1]. Multiple surfels may be assembled to describe a surface. 50 Research interest on surfels has risen significantly as a set of 51 surfels can be processed like a point cloud but it contains more 52 information. A surfel cloud is also simpler than a polygon mesh 53 data structure and it requires less memory than a volumetric 54 representation. As an example, the ElasticFusion algorithm [2] 55 has been proven capable of GPU-accelerated real-time 3D re-56 construction on a surfel cloud. 57

In this letter, a novel kind of surfel is introduced, named 58 frontel, in order to represent the frontier between empty and 59 unknown space (Fig. 1). We exploit previous work [3] where the 60 KinFu Large Scale algorithm, an open source implementation 61 of KinectFusion [4] in the Point Cloud Library, was modified to 62 reliably keep track of both empty and unknown voxels. Further 63 changes have been made in this work to generate surfels and 64 frontels in real-time during the KinFu reconstruction process, 65 with minimal overhead. In the view simulation phase, frontels 66 and surfels are rendered to simulate a depth image. 67

The proposed NBV approach was evaluated in a real setup including a robot arm with a Kinect sensor in eye-in-hand configuration. Results are compared with volumetric methods, where viewpoint evaluation was accelerated on GPU thanks to KinFu internal ray casting. The experimental evaluation shows a

2377-3766 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications\_standards/publications/rights/index.html for more information.

2



28

Q2

87

significant performance improvement for the surfel-based NBV 73 algorithm, with a similar quality in 3D reconstruction. Most of 74 the performance improvement is due to the opportunity to fit the 75 76 whole 3D representation on GPU memory at once, which is not possible with a volumetric representation. Indeed, in volumetric 77 NBV approaches, when observing different regions of space, 78 parts of the environment must be loaded from CPU to GPU 79 memory before viewpoint evaluation. Similarly, due to limited 80 GPU memory, another advantage of surfel-based NBV planning 81 82 against volumetric approaches is that it can be applied in larger environments. The paper is organized as follows. Related work 83 is reported in Section II. In Section III, the proposed surfel-84 based NBV algorithm is described. Experiments and results are 85 reported in Section IV. Finally, Section V concludes the work. 86

## II. RELATED WORK

Next best view planning is a well established area of re-88 89 search in robotics [5]–[7]. Here we mainly review recent results, with the aim of showing that the use of volumetric approaches, 90 91 requiring computationally intensive ray casting operations for viewpoint evaluation, remains prevalent. Most works focus on 92 non model-based methods that do not assume an a priori model 93 of the scene, while model based NBV planning requires both 94 95 modeling and object recognition [8]. In [9] an approach was proposed which takes into account the position uncertainty of 96 the sensor and that adopts a utility function that considers sev-97 eral factors like perception of unseen areas, navigation distance, 98 reconstruction quality and fast occlusion estimation. Potthast 99 et al. [10] presented a variant of the NBV problem that builds a 100 belief model of the unobserved space for cluttered environments 101 with occlusion. In [11] a frontier-oriented algorithm based on 102 volumetric hierarchical ray tracing was introduced to speed up 103 NBV evaluation. 104

In mobile robotics the view planning problem is constrained 105 in the 2D plane [9], [12]-[18]. Senarathne et al. [12] presented 106 a method based on surface frontiers, i.e. the boundary voxels 107 of mapped surfaces adjacent to unmapped space. In [14] an 108 adaptable and probabilistic object reconstruction approach was 109 proposed to evaluate the information gain. An optimization was 110 also introduced, based on a lookup table, for the evaluation of 111 multiple view orientations from the same viewpoint. Isler et al. 112 [15] exploited a probabilistic volumetric map and investigated 113 an algorithm that estimates whether a ray is expected to hit the 114 backside of already observed surfaces. Delmerico et al. [16] 115 evaluated different information gain metrics for NBV planning 116 using a probabilistic voxel map in terms of completeness and 117 entropy. The comparative analysis indicated that the utility func-118 tion defined in [9] achieved the best performance, as well as a 119 120 metric that gives higher weights to unobserved voxels close to already observed surfaces. Patten et al. [17] developed a model 121 122 based system for outdoor active object classification using a mobile robot. Monte Carlo methods for planning new observations 123 were adopted with time and distance constraints together with 124 a non-parametric Bayesian regression classifier. 125

NBV planning has also been applied to aerial robotics. In [19]a receding horizon method was introduced based on sampling a

random tree of candidate viewpoints. The method scales better 128 than a frontier-based planner in large environments. In [20] a 129 2-stage planning solution was proposed that aims at achieving 130 full coverage of the environment and global optimality of the 131 exploration path. 132

Surface based approaches for NBV planning were inves-133 tigated using triangular meshes [21]-[23]. Viewpoints were 134 generated by examining the boundaries of the reconstructed 135 surfaces. These methods do not model the surface between 136 empty and unknown space, and, therefore cannot select the 137 next best view by estimating information gain from unknown 138 volume. Volumetric representations are also preferred for 139 probabilistic approaches [15]. 140

## III. METHOD

The robot task is to perform a 3D reconstruction of the volume 142 around a set of Points Of Interest (POIs), in a tabletop scenario, 143 by taking a sequence of observations. It is assumed that the set 144 of POIs is given as input to the system. For example, a POI 145 may indicate the location of an object or a group of objects 146 to be scanned. At the beginning of the task the robot has an 147 initial, possibly incomplete, representation of the environment 148 and the volume around each POI is cleared (set to unknown), as 149 described in Section III-B. The NBV planning procedure is then 150 executed. At each iteration candidate viewpoints are sampled 151 on a view sphere with fixed radius around each POI, oriented 152 towards the POI itself. In the view simulation phase, surfels and 153 frontels are rendered to a virtual view (Section III-C). Then a 154 score is computed from the rendered image (Section III-D). The 155 score represents the expected information gain for the simulated 156 sensor pose. The poses are attempted using a motion planner in 157 decreasing order of scores. The first feasible solution is executed 158 by the robot. 159

In the proposed method surfels and frontels are generated 160 on GPU in real-time during robot observations from the KinFu 161 internal representation. Both surfels s and frontels f are circu-162 lar disks in 3D space. Surfels separate empty from occupied 163 space, while frontels separate empty from unknown space. In 164 particular, surfel/frontel generation is performed during trun-165 cated signed distance function (TSDF) volume shifting opera-166 tions to optimize performance (Section III-A). The TSDF vol-167 ume is the volumetric representation of the environment used by 168 the KinFu algorithm. As KinFu is able to keep in GPU memory 169 only a limited volume, if the whole environment does not fit 170 inside the TSDF volume KinFu Large Scale shifts the TSDF 171 volume by unloading and loading parts of the TSDF volume 172 from and to GPU memory according to sensor movements. 173

#### A. Real-Time Surfel and Frontel Generation 174

The TSDF volume is organized as a regular voxel grid. The 175 TSDF assigns to each voxel the distance to the nearest surface, negative in occupied space and positive in empty space. 177 The distance is truncated in the interval  $[-v_{\text{max}}, v_{\text{max}}]$ . Each 178 voxel  $c(x_c, y_c, z_c)$  holds a TSDF value  $v_c$ , i.e. the value of the 179 TSDF at the center of the voxel, and a weight  $w_c$  which contains the number of times the voxel has been observed, up to a 181

maximum. KinFu operates on a cubic TSDF volume with edge 182 length  $E = c_{\max}e$ , where  $c_{\max}$  is the volume resolution and e is 183 the voxel size. Whenever a point is observed by the sensor, the 184 voxels on the ray between the observation and the sensor po-185 sition are updated. However, only the volume inside the TSDF 186 is updated by KinFu. The Large Scale extension of KinFu per-187 forms a shifting operation when the distance between the TSDF 188 volume center and a virtual point, at about E/2 distance from the 189 sensor along the z axis, is greater than E/3. A shifting operation 190 191 translates the TSDF volume such that it represents a new region in space, centered at the the virtual point. Shifting ensures that 192 the region currently observed by the sensor is the one represented 193 on GPU and that it can be updated. Information contained in the 194 intersection between the old TSDF volume and the new one is 195 kept in GPU memory. Voxels with  $w_c > 0$  and  $v_c < v_{max}$  outside 196 the new volume are downloaded from GPU memory to RAM. 197 Then, voxels are converted into points, described by position c198 and TSDF value  $v_c$ , and added to a TSDF point cloud. Points 199 are reloaded to TSDF volume if a subsequent shifting operation 200 moves the volume back in the old region. Due to memory con-201 202 straints, empty voxels with  $v_c = v_{max}$  are not downloaded by the aforementioned procedure. Therefore, in our previous work 203 [3] an octree  $\Omega$  was added to store  $w_c$  when  $v_c = v_{\text{max}}$ . 204

Surfels s are circular disks in 3D space, with position  $p_s$ , 205 206 radius  $r_s$  and normal  $n_s$ . Color is not available in KinFu, but it is not required for the proposed NBV score computation. 207 Similarly, frontels f have position  $p_f$ , radius  $r_f$  and normal  $n_f$ . 208 The surface between the outside and the inside of objects is 209 located in the TSDF volume between voxels having a different 210 sign of v. That is, an empty voxel c is near the surface of an 211 212 object if the following conditions hold:

$$\begin{cases} v_c \ge 0 \quad \land \quad w_c > 0, \\ \exists \ c' \in N_6 \ (c) \ | \ v_{c'} < 0, \ w_{c'} > 0 \end{cases}$$
(1)

where  $N_6(c)$  is the 6-neighborhood of voxel c. Similarly, the frontier between empty and unknown space is located where wchanges from 0 to a positive value, i.e.

$$\begin{cases} v_c \ge 0 \land w_c > 0, \\ \exists c' \in N_6(c) \mid w_{c'} = 0 \end{cases}$$
(2)

Before a TSDF volume shifting operation occurs, surfels and 216 frontels are generated from the old TSDF volume. A surfel 217 centered on voxel c is generated, i.e.  $p_s = c$ , if condition (1) 218 holds, whereas a frontel  $p_f = c$  is generated if condition (2) 219 holds. Generated surfels/frontels are added to two sets on CPU 220 RAM, here named  $\Sigma$  and  $\Phi$  respectively. To avoid duplication, 221 existing elements of  $\Sigma$  and  $\Phi$  inside the region represented by the 222 old TSDF volume are cleared. These elements may have been 223 generated by previous shifting operations in the same region. 224

The surfel local normal is estimated from the TSDF local gradient, according to the signed distance function properties, as follows (normalization omitted):

$$n_s(c) = \sum_{c' \in N_6} v_{c'} \frac{c' - c}{\|c' - c\|}$$
(3)



Fig. 2. The current position of the TSDF volume is marked with a black dashed square. The sensor observes towards the right, clearing the yellow unknown area. The green line indicates where frontels are generated (a) without and (b) with padding.

Conversely, a frontel normal is estimated from the 26neighborhood  $N_{26}$  of the voxel c, as follows (normalization 229 omitted): 230

$$K_w(c) = \begin{cases} 1 & \text{if } w_c > 0 \\ -1 & \text{if } w_c = 0 \end{cases}$$
(4)

$$n_f(c) = \sum_{c' \in N_{26}} K_w(c') \frac{c' - c}{\|c' - c\|}$$
(5)

The 26-neighborhood was chosen over the 6-neighborhood to 231 reduce the quantization error. Indeed, unlike  $v_c$ ,  $K_w(c)$  as in 232 (4) can assume only two values. Normal  $n_f$  is oriented from un-233 known space towards known space. Equations (1) and (2) place 234 the surfel at the center of an empty voxel next to an occupied 235 or unknown voxel. Therefore, occupied and unknown volumes 236 are enlarged by  $\delta_e = \frac{1}{2}e$  per side, i.e half the KinFu voxel edge 237 e. To generate a surface without holes, the surfel/frontel radius 238 must be set as half the distance between the two farthest vertices 239 of a voxel, i.e.  $\frac{\sqrt{3}}{2}$   $(e+2\delta_e)$ . Therefore, the surfel/frontel radius 240 may be approximated as  $r_s = r_f = \sqrt{3}e$ . 241

The surfel/frontel generation procedure [(1) and (2)] requires 242 knowledge about the neighboring voxels. Voxels in contact with 243 the TSDF volume surface are considered as a special case cov-244 ered by reducing the active TSDF volume by a one-voxel-wide 245 padding. The padding is downloaded and uploaded from/to GPU 246 as usual, but it is not updated with new data acquired by the sen-247 sor. An example of this issue is provided in Fig. 2. In case (a), 248 without padding, the TSDF volume has been set to empty up 249 to its right surface. Therefore, frontels are not generated in that 250 area, leaving a hole in the surface enclosing unknown space. In 251 case (b), the unknown values near the TSDF volume surface are 252 not removed and frontels are correctly generated. 253

#### B. Initialization of the Regions of Interest for the NBV Task 254

Each POI defines a spherical region with origin  $p_{poi}$  and radius  $R_{poi}$ . At the beginning of the task each spherical region 256 around a POI is cleared and set to unknown. To this purpose 257 the TSDF volume and the surfel/frontel-based representation 258 must be managed in a consistent way. In the part of the sphere 259 currently inside the TSDF volume,  $w_c$  is set to 0 whenever  $\|c - p_{poi}\| \le R_{poi}$ .

Outside the TSDF volume, the update procedure is executed as follows. First, surfels and frontels inside the sphere are deleted from  $\Sigma$  and  $\Phi$ . Then, frontels must be generated in  $\Phi$  between the cleared volume and the empty voxels in octree  $\Omega$ . Let  $c \in \Omega$ be a voxel outside the sphere, c has a known neighbor inside the sphere if the following conditions hold:

$$\begin{cases} ||c - p_{\text{poi}}|| > R_{\text{poi}} & \land \quad w_c > 0, \\ \exists c' \in N_6(c) \mid w_{c'} > 0, \; ||c' - p_{\text{poi}}|| \le R_{\text{poi}} \end{cases}$$
(6)

Therefore, a new frontel is created at position  $p_f = c$ , with radius  $r_f$  and normal  $n_f = (p_f - p_{\text{poi}})/||p_f - p_{\text{poi}}||$  when conditions in (6) hold. Finally, the voxels inside the sphere are set to unknown in  $\Omega$ .

#### 272 C. Surfel Rendering

The standard pinhole model is used for simulating the Kinect sensor. The depth image of a real sensor contains, for each pixel, the distance of the first intersection of a ray cast from the origin of the camera center passing through that pixel. Some pixels are left undefined, since the intersection may go below (or beyond) the minimum (or maximum) sensor range.

A sensor which follows the pinhole model can be simulated by 279 rendering. For each surfel or frontel, position  $(3 \times 32$ -bit floats), 280 normal  $(3 \times 32$ -bit floats) and radius (32-bit float) are provided 281 to the rendering pipeline. In total, 28 bytes are required per surfel 282 or frontel. In our approach, rendering generates an image that 283 284 includes all information needed for NBV score computation. Unlike the real sensor, three possible results may occur for the 285 virtual sensor pixels, i.e. a pixel is marked: a) occupied, if a 286 surfel is rendered; b) unknown, if a frontel is rendered; c) out-287 288 of-range, if nothing is rendered or the surfel/frontel is out of the sensor range. A single real value  $o_{ij}$  is used for each pixel as the 289 290 output of the rendering pipeline. Given the  $z_d$  depth in camera coordinates of the rendered surfel/frontel, the output value is set 291 292 as:

$$o_{ij} = \begin{cases} z_d & \text{if (a)} \\ -z_d & \text{if (b)} \\ 0 & \text{if (c)} \end{cases}$$
(7)

Therefore, when rendered, surfels and frontels produce positive and negative depths respectively.

Image resolution, center point and focal length are known 295 from the sensor intrinsic calibration. The maximum range of 296 the sensor is used to set the frustum far plane, so that farther 297 points are discarded. By setting background to 0, case (c) is auto-298 matically handled for points beyond maximum range. However, 299 since surfels between the camera and minimum range occlude 300 surfels behind them in the real sensor, they must be rendered 301 with output value 0. 302

In the view evaluation phase TSDF volume shifting operations must be simulated. The predicted TSDF volume is assumed centered on the POI which generated the view. Surfels/frontels outside the predicted position of the TSDF volume are rendered with  $o_{ij} = 0$ , as when the real sensor measures range data



Fig. 3. (a) Surfel s, with position  $p_s$  and radius  $r_s$ , is projected to the view plane. Square size  $l \ge \overline{AB}$  must be estimated. (b) A case in which l is underestimated. Distance  $\overline{OP}$  is reduced to highlight the effect.

outside the TSDF volume such data do not contribute to the 3D 308 reconstruction. 309

A point-based rendering technique is adopted to efficiently 310 render surfels as points [24]. The default OpenGL pipeline ren-311 ders points as squares of size l facing the camera. A vertex 312 shader is executed in parallel on GPU for each point sent to 313 the rendering pipeline. In particular, each point is transformed 314 into camera coordinates and the distance  $z_d$  is computed. The 315 OpenGL pipeline is instructed to draw squares with size l by 316 setting the variable gl\_PointSize in the vertex shader. Each 317 square may span multiple fragments, each corresponding to 318 a pixel. For each fragment, a fragment shader is run by the 319 OpenGL pipeline. The fragment shader computes the 3D dis-320 tance of each fragment from the center of the surfel which 321 generated it. The fragment is discarded if the distance is greater 322 than the radius  $r_s$ . 323

Estimation of a suitable square size *l* in the vertex shader 324 is a non-trivial task. Indeed, the size must be large enough 325 to encompass the entire surfel. However, setting a too large 326 square size would cause generation of many useless fragments, 327 thus deteriorating performance. If the surfel is roughly parallel 328 to the image plane [Fig. 3(a)], l can be estimated as follows. 329 Triangles  $\triangle OAB$  and  $\triangle OA'B'$ , and also  $\triangle OAP$  and  $\triangle OA'P'$ 330 are similar, i.e.: 331

$$\frac{\overline{OP'}}{\overline{OP}} = \frac{\overline{OA'}}{\overline{OA}} = \frac{\overline{A'B'}}{\overline{AB}}$$
(8)

Therefore, given focal length F, edge l is estimated as:

1

$$=\frac{2Fr_s}{z_d}\tag{9}$$

A similar estimate was performed in [24]. However, by using 333 such formula, *l* may be underestimated as shown in Fig. 3(b), in 334 a 2D example. This results in incompletely rendered surfels, cut 335 by the bounding square. In the following, we provide an upper 336 bound for this under-estimation. 337

Surfel *s* is rotated by angle  $\alpha$  so that it appears slightly bigger, 338 i.e. the projection of its top-most point onto the plane through 339  $p_s$ , parallel to the view plane, is in *C'* instead of *A'*. A similar 340

effect happens for the lower-most pixel, which now projects 341 in D'. However, the error for D' is smaller, due to the minor 342 distance from the sensor principal axis. The circle displayed 343 344 around the surfel in Fig. 3(b) represents the possible positions of the top-most and lower-most points, by varying  $\alpha$  values. The 345 maximum distance  $\overline{A'C'}$  occurs when the surfel is perpendicular 346 to C'O, i.e.  $\beta = \pi/2$ . In this case,  $\alpha = \angle P'OC'$ . Therefore, the 347 projected surfel radius  $r'_s = \overline{p_s C'}$  is upper bounded as follows: 348

$$r'_s \le \frac{r_s}{\cos\left(\angle P'OC'\right)} \tag{10}$$

However,  $\angle P'OC'$  can be at most equal to the sensor Half Field Of View (HFOV). Therefore, for a Kinect-like sensor, with HFOV  $\approx 30^{\circ}$ , cos (HFOV) =  $\sqrt{3}/2 \approx 0.866$ . Hence, in the worst case a surfel is reduced by less than 14% its size.

#### 353 D. NBV Score Computation

Standard NBV score computation methods assign a score 354 equal to the number of unknown voxels visible from each pose. 355 The more unknown voxels are visible, the higher is the infor-356 mation gain expected from that pose. In this letter we propose 357 a novel score function where the total area of visible frontels 358 is used to approximate the number of unknown voxels. Since 359 frontels are generated to cover the area of the frontier between 360 empty and unknown space, the total area of visible frontels is 361 (approximately) proportional to the number of voxels in the 362 frontier. The number  $\gamma_r$  of sensor view rays which intersect a 363 frontier is computed from the rendering output (7) as: 364

$$U_{o}(o_{ij}) = \begin{cases} 1 & \text{if } o_{ij} < 0\\ 0 & \text{otherwise} \end{cases}$$
(11)  
$$\gamma_{r} = \sum_{ij} U_{o}(o_{ij})$$
(12)

However, each frontel f contributes to  $\gamma_r$  proportionally to 365 its projected area  $\mathcal{A}_{f}^{p}$ . Therefore, frontels close to the camera 366 contribute excessively. This high contribution is not consistent 367 with the standard score function that counts the number of un-368 known visible voxels. Indeed, in the standard score function a 369 370 voxel close to the camera counts as 1, like any other voxel. In the following, a more appropriate score function is defined by 371 introducing a weighting factor. 372

Equation (9) in Section III-C relates distance  $z_d$  and radius  $r_f$ of a frontel (roughly parallel to the view plane) to the diameter lof its projection on the view plane. Also, due to (7),  $z_d = |o_{ij}|$ . Hence, the projected frontel area in pixels can be estimated as:

$$\mathcal{A}_f^p = \pi \left(\frac{l}{2}\right)^2 = \pi \left(\frac{1}{2} \frac{2Fr_f}{z_d}\right)^2 = \left(\frac{F}{|o_{ij}|}\right)^2 \mathcal{A}_f^v \quad (13)$$

where  $A_f^v = \pi r_f^2$  is the actual frontel area. The weighted NBV score function is then defined as:

$$\gamma = \sum_{ij} U_o(o_{ij}) \gamma_{ij} \tag{14}$$



Fig. 4. The experimental setup.

where the weighting factor  $\gamma_{ij}$  is

$$\gamma_{ij} = \frac{\mathcal{A}_f^v}{\max\left\{\mathcal{A}_f^p, 1\right\}} \tag{15}$$

For frontels not too far away from the camera, (15) can be 380 simplified by using (13) so that: 381

$$\gamma_{ij} = \frac{\mathcal{A}_f^v}{\mathcal{A}_f^p} = \left(\frac{|o_{ij}|}{F}\right)^2 \quad \text{if } A_f^p \ge 1 \quad (16)$$

i.e.  $\gamma_{ij}$  is set as inversely proportional to the projected surfel 382 area  $\mathcal{A}_f^p$ . Conversely, for distant frontels where the projection 383 area  $\mathcal{A}_f^p$  is smaller than a pixel, gain  $\gamma_{ij}$  becomes 384

$$\gamma_{ij} = \mathcal{A}_f^v = \pi r_f^2 \qquad \text{if } \mathcal{A}_f^p < 1 \tag{17}$$

to prevent overestimation. From (15), in the general case, both 385 distance  $o_{ij}$  and frontel radius  $r_f$  are required to compute area 386  $\mathcal{A}_f^v$  and then gain  $\gamma_{ij}$ . In this work, the frontel radius is constant, as shown in Section III-A. However, our approach can be 388 extended to a variable frontel radius by computing gain  $\gamma_{ij}$  in 389 the fragment shader, where the radius is available. 390

#### IV. EXPERIMENTS 391

#### A. Experimental Setup

The experimental setup (Fig. 4) includes an industrial robot 393 arm (Comau SMART SiX) with six degrees of freedom. A 394 Kinect sensor is mounted on the end-effector and calibrated 395 with respect to the robot wrist. KinFu egomotion tracking is 396 disabled and the sensor position is computed from robot for-397 ward kinematics. The workspace is a square of about 2.5  $\times$ 398 2.5 m that encompasses two tables in front of the robot (high-399 lighted in yellow in Fig. 4). KinFu TSDF volume side is set to 400 to E = 1.5 m and the edge resolution is set to  $c_{\text{max}} = 512$  cells. 401 Therefore, the voxel edge length is set to e = 2.9 mm, and thus 402  $r_f = 5.1$  mm. As E is lower than the workspace size, shifting 403 operations must be performed while scanning different regions 404 of interest. The same shifting operations must be simulated in 405 the view evaluation phase, to accurately predict information 406 gain. Each experiment is performed as follows. First, the robot 407 scans the environment on a predefined path using KinFu. POIs 408 are given as input to the system, located near the objects to be 409 scanned. Regions of interest are initialized as unknown around 410 each POI as described in Section III-B. Viewpoints are sampled 411

379



Fig. 5. Top: scenarios 1 and 2. Bottom: the spherical regions of interest centered at each POI, surrounded by black frontels.

TABLE I Experiment Initial Data

Scenario	1	2
Initial surfels	$\sim 519 \mathrm{k}$	$\sim 489 \mathrm{k}$
Initial frontels	$\sim 275 \mathrm{k}$	$\sim 302 k$
Initial unknown	$\sim \! 4309 \mathrm{k}$	$\sim 7608 \mathrm{k}$

on spheres around each POI, at a fixed distance of 0.8 m. Each 412 POI generates 960 candidate views, sampled at regular intervals, 413 414 by varying latitude (10 intervals), longitude (12 intervals) and rotation around the sensor axis (8 intervals). Then, the NBV task 415 is started. At every iteration, all viewpoints generated for each 416 POI are evaluated by the NBV algorithm. The MoveIt! ROS 417 framework is used for planning a collision free path towards 418 the viewpoint with the highest predicted information gain. A 419 collision map with precision 5 cm is extracted from KinFu to 420 update the MoveIt! planning scene. The collision map considers 421 both occupied and unknown voxels as obstacles. Viewpoints are 422 attempted in decreasing score order, until the planner succeeds. 423 The robot then moves the sensor in the selected NBV pose and 424 new information is acquired. The Kinect is slightly tilted  $(\pm 5^{\circ})$ 425 to remove artifacts that otherwise would appear due to the emit-426 ted IR pattern. Indeed, if the sensor is fed multiple images from 427 the same viewpoint in a static environment ripples appear in the 428 3D reconstruction. The Kinect depth image has a resolution of 429  $640 \times 480$  pixels, with a focal length of about 528 pixels. How-430 ever, when the sensor is simulated during NBV computation a 431  $560 \times 540$  resolution is adopted, with the same focal length. 432 The simulated camera height is increased to account for the 433 tilting motion of the real sensor during acquisition, while the 434 simulated camera width is reduced due to sensor calibration. 435 The software runs on an Intel i7-6700 CPU at 3.40 GHz, 32 GB 436 RAM, NVIDIA GeForce GTX 980 Ti GPU, 6 GB RAM. 437

Experiments were performed in two different scenarios, 438 shown in Fig. 5. In the first scenario, three POIs were defined, 439 one for each object, with radius 0.2 m. In the second scenario, 440 four POIs of different sizes were defined, one for each group 441 of objects. Table I reports the initial number of surfels, frontels 442 and unknown voxels inside the POIs. In each scenario, the ex-443 periment was repeated three times, by changing the NBV score 444 function. The first score function, named RCV (Ray Casting 445 Voxel count), is the standard NBV algorithm that maximizes 446

TABLE II AVERAGE EXECUTION TIME (SECONDS)

Phase	RCV	RCP	SRP
Initialization	-	-	$0.02 \pm 0.01$
Shifting	$2.83 \pm 0.17$	$2.81 \pm 0.18$	-
Rendering	-	-	$0.25\pm0.03$
Ray casting	$1.49\pm0.18$	$1.11 \pm 0.12$	-
GPU download	$1.29\pm0.04$	$0.34 \pm 0.00$	$0.71 \pm 0.04$
Score computation	$3.53\pm0.15$	$0.56\pm0.04$	$0.46\pm0.02$
Total	$9.14 \pm 0.41$	$4.83\pm0.25$	$1.43\pm0.06$

the number of unknown visible voxels. KinFu ray casting was 447 exploited to obtain  $o_{ij}$  and voxel index  $c_{ij}$  for each pixel. To 448 prevent the same voxel index to be counted more than one time, 449 a support boolean 3D matrix is used to track which voxels have 450 already been counted. The matrix has the same size of the TSDF 451 volume and it is indexed by  $c_{ij}$ . The second NBV score func-452 tion (RCP: Ray Casting Pixel gain) exploits KinFu ray casting 453 to compute the pixel values  $o_{ij}$ . The distance  $|o_{ij}|$  is obtained 454 as the depth of the intersection between the ray and the first non 455 empty voxel. Gain  $\gamma$  (14) is used to evaluate the next best view. 456 RCP only requires knowledge of the pixel values  $o_{ij}$ . Finally, 457 the third score function (SRP) uses pixel values  $o_{ij}$  obtained by 458 surfel rendering as proposed in Section III-C. The NBV is again 459 estimated by  $\gamma$ . The robot task terminates either when the NBV 460 predicts a gain lower than a threshold or after 10 robot poses. 461 The threshold was set as  $\gamma_{\text{th}} = 0.002 \text{ m}^2$  for RCP and SRP. In 462 the RCV case, as the area of a voxel face is approximately 463  $(2.9 \text{ mm})^2$ , the threshold was set equal to  $0.002 \text{ m}^2/(2.9 \text{ mm})^2$ 464  $\approx 237$  voxels. 465

# B. Results

Average execution times of each phase of NBV evaluation are 467 reported in Table II, as well as standard deviations. Results were 468 obtained by averaging the evaluation time of each POI (960 view 469 poses). More than 50 POI evaluations were executed for each 470 approach in the reported experiments. It can be noticed that the 471 total time to evaluate 960 poses is 9.14 s for RCV, 4.83 s for RCP 472 and 1.43 s for SRP. That is, surfel rendering (SRP) outperforms 473 both volumetric approaches (RCV and RCP) since rendering 474 is faster than ray casting. The proposed NBV approach based 475 on surfel rendering (SRP) requires a short initialization phase, 476 since clouds  $\Sigma$  and  $\Phi$  are reloaded from RAM to GPU memory. 477 For the methods based on KinFu ray casting, instead, the TSDF 478 volume must be shifted and centered on the current POI. There-479 fore, the shifting operation delays NBV computation by 2.81 s 480 on average. SRP improvement is partially offset by the shorter 481 data download time of RCP, probably due to better CUDA opti-482 mization. In general, the RCV approach is the slowest, since it 483 also downloads voxel index  $c_{ij}$ . Moreover, in RCV each voxel 484 must be counted only once, thus requiring a more complex score 485 computation method. 486

466

The surfel/frontel generation procedure introduces a low overhead at each KinFu shifting operation during 3D reconstruction. 488 Indeed, shifting lasts for additional 132 ms on average, over a total time of about 781 ms. The amount of unknown voxels 490



Fig. 6. Top: the first, third, fifth and seventh NBV reached by the robot in the SRP experiment of scenario 1. Bottom: the surfel- and frontel-based representation after each NBV.



Fig. 7. Number of unknown voxels inside regions of interest, after each NBV.

TABLE III NBV ITERATIONS, COMPLETENESS AND RECONSTRUCTION QUALITY

Scenario	1			2		
Method	RCV	RCP	SRP	RCV	RCP	SRP
Iterations	8	7	8	10+	10+	9
Completeness	96%	94%	96%	92%	93%	90%
Average $Q_t$	425.2	411.3	408.5	531.0	514.7	477.3
$Q_t$ Std. dev.	294.7	249.3	295.5	495.6	385.7	428.7



Fig. 8. Surfel- and frontel-based reconstruction at the end of the SRP experiments, for each scenario.

inside regions of interest decreases with the number of NBV 491 iterations, as the robot explores the unknown regions. Fig. 6 492 shows images of some next best views for SRP in scenario 1, as 493 well as the progressive reduction of unknown space surrounded 494 by frontels (in black). The graphs in Fig. 7 show the number of 495 unknown voxels as the NBV task progresses. All three methods 496 have a similar trend. Table III reports the number of NBV itera-497 tions for each experiment. Symbol 10+ marks experiments that 498 where stopped after 10 iterations. The final 3D environment re-499 construction for the surfel-based experiments is shown in Fig. 8. 500 Reconstruction quality was assessed by comparing results of the 501 NBV task with the initial scan of the environment. The initial 502 robot scan was performed along a predefined path to scan all 503 POIs and, therefore, it acts as ground truth. We define T the ini-504



Fig. 9. Effect of the volume padding on frontel generation, during first NBV of scenario 2. Left: KinFu output, with the TSDF volume limit highlighted in red. Center and right: surfels/frontels before and after the NBV.

tial point cloud produced by KinFu marching cubes algorithm 505 after the initial scan, inside the regions of interest. Moreover, we 506 define G the set of all points acquired by Kinect during an ex-507 periment. Quality  $Q_t$  of point  $t \in T$  is defined as the number of 508 points in G whose distance to t is lower than 2 cm. The meaning 509 of  $Q_t$  is that automated NBV reconstruction quality is higher the 510 more points are closer to ground truth points obtained in a full 511 scan of the POIs. Completeness of reconstruction was evaluated 512 as the fraction of points where  $Q_t > 0$ . Completeness, average 513 quality across all points in T and quality standard deviation are 514 reported in Table III. In general, completeness is comparable for 515 the three methods. Quality is slightly lower for RCP and SRP 516 than for RCV, which is probably due to the use of an approxi-517 mated gain, as presented in Section III-D, instead of the exact 518 number of voxels. Fig. 9 shows the effect of volume padding 519 on frontel generation, as described in Section III-A. The sensor 520 is oriented towards a POI centered on the small horse object. 521 However, other POIs are also visible. No data can be acquired 522 outside the TSDF volume, as highlighted by the red line near 523 the boxes. Therefore, the unknown regions surrounding the two 524 POIs on the boxes are only partially carved. Frontels are cor-525 rectly generated on the surface of the TSDF volume inside the 526 spheres. The methods SRP and RCP generate a slightly differ-527 ent simulated image  $o_{ij}$ , due to the different approximations 528 made by the two algorithms. An example is shown in Fig. 10. 529 RCP shows some drawback: the step of the KinFu ray casting 530 algorithm is larger than the size of a single voxel, therefore, thin 531 unknown volumes may be skipped during ray casting, as in the 532 area highlighted with yellow squares. Moreover, some borders 533 appear jagged (yellow circle), as the algorithm may detect the 534 unknown space up to one step deep inside the surface. On the 535



Top: visible points simulated by RCP (left) and SRP (right) for the Fig. 10 third NBV of the SRP experiment in scenario 1. Blue colors represent positive values of oij, while orange colors negative values. Bottom left: difference between the two simulated images. Bottom right: the surfel-based representation observed from a different viewpoint.

other hand the surfel-based approximation of SRP causes ob-536 jects to expand slightly, since the surfel/frontel sizes  $r_s = r_f$ 537 were over-estimated in Section III-A. 538

539

554

03

#### V. CONCLUSION

540 In this work a novel method for robot next best view planning has been proposed. The approach is based on a surfel repre-541 sentation of the environment. Surfel rendering is used for NBV 542 evaluation, instead of complex ray casting operations. Results 543 indicate that a score function based on surfels is more efficient 544 545 to compute and that it achieves comparable results in terms of reconstruction quality and completeness. Another advantage is 546 that surfel-based NBV planning can be applied in larger envi-547 ronments. A limitation of the proposed method is that it still 548 requires an initial voxel based volume representation, which is 549 obtained through KinectFusion, from which surfels and frontels 550 are extracted. In future work we plan to investigate algorithms 551 for surfel and frontel generation that do not require an initial 552 voxel representation of the environment. 553

## REFERENCES

- 555 [1] H. Pfister, M. Zwicker, J. van Baar, and M. Gross, "Surfels: Surface 556 elements as rendering primitives," in Proc. 27th Annu. Conf. Comput. 557 Graph. Interactive Techn., 2000, pp. 335-342.
- 558 T. Whelan, S. Leutenegger, R. S. Moreno, B. Glocker, and A. Davison, [2] "Elasticfusion: Dense SLAM without a pose graph," in Proc. Robot., Sci. 559 560 Syst., Rome, Italy, Jul. 2015.
- R. Monica, J. Aleotti, and S. Caselli, "A KinFu based approach for robot 561 spatial attention and view planning," Robot. Auton. Syst., vol. 75, Part B, 562 563 pp. 627-640, 2016.

- [4] R. A. Newcombe et al., "KinectFusion: Real-time dense surface map-564 ping and tracking," in Proc. IEEE Int. Symp. Mixed Augmented Reality, 565 Oct. 2011, pp. 127-136. 566
- C. Connolly, "The determination of next best views," in *Proc. IEEE Int.* [5] 567 Conf. Robot. Automat., 1985, vol. 2, pp. 432-435. 568
- [6] P. Whaite and F. Ferrie, "Autonomous exploration: Driven by uncertainty," 569 IEEE Trans. Pattern Anal. Mach. Intell., vol. 19, no. 3, pp. 193-205, Mar. 1997.
- [7] R. Monica and J. Aleotti, "Contour-based next-best view planning from 572 point cloud segmentation of unknown objects," Auton. Robots, vol. 42, 573 no. 2, pp. 443-458, 2018. 574
- [8] S. Kriegel, M. Brucker, Z. C. Marton, T. Bodenmüller, and M. Suppa, 575 "Combining object modeling and recognition for active scene explo-576 ration," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2013, pp. 2384-577 2391.
- [9] I I Vasquez-Gomez L E Sucar R Murrieta-Cid and E Lopez-Damian 579 "Volumetric next-best-view planning for 3D object reconstruction with 580 positioning error," Int. J. Adv. Robot. Syst., vol. 11, no. 10, p. 159, 2014. 582
- [10] C. Potthast and G. S. Sukhatme, "A probabilistic framework for next 583 best view estimation in a cluttered environment," J. Vis. Commun. Image 584 Representation, vol. 25, no. 1, pp. 148-164, 2014. 585
- [11] F. Bissmarck, M. Svensson, and G. Tolt, "Efficient algorithms for next 586 best view evaluation," in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 587 2015, pp. 5876-5883. 588
- [12] P.G.C.N. Senarathne and D. Wang, "Towards autonomous 3D exploration 589 using surface frontiers," in Proc. IEEE Int. Symp. Safety, Secur., Rescue 590 Robot., 2016, pp. 34-41 591
- [13] K. Wu, R. Ranasinghe, and G. Dissanayake, "Active recognition and pose 592 estimation of household objects in clutter," in Proc. IEEE Int. Conf. Robot. 593 Automat., May 2015, pp. 4230-4237. 594 595
- [14] J. Daudelin and M. Campbell, "An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-D objects," IEEE Robot. 596 Automat. Lett., vol. 2, no. 3, pp. 1540-1547, Jul. 2017. 597
- [15] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza, "An information 598 gain formulation for active volumetric 3D reconstruction," in Proc. IEEE 599 Int. Conf. Robot. Automat., May 2016, pp. 3477-3484. 600 601
- [16] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza, "A comparison of volumetric information gain metrics for active 3D object reconstruction," Auton. Robots, vol. 42, no. 2, pp. 197-208, Feb. 2018.
- [17] T. Patten, W. Martens, and R. Fitch, "Monte Carlo planning for active 604 object classification," Auton. Robots, vol. 42, no. 2, pp. 391-421, Feb. 605 2018.606
- [18] T. Patten, M. Zillich, R. Fitch, M. Vincze, and S. Sukkarieh, "Viewpoint 607 evaluation for online 3-D active object classification," IEEE Robot. Au-608 tomat. Lett., vol. 1, no. 1, pp. 73-81, Jan. 2016. 609
- [19] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Re-610 ceding horizon "next-best-view" planner for 3D exploration," in Proc. 611 IEEE Int. Conf. Robot. Automat., May 2016, pp. 1462-1468. 612
- [20] Z. Meng et al., "A two-stage optimized next-view planning frame-613 work for 3-D unknown environment exploration, and structural recon-614 struction," IEEE Robot. Automat. Lett., vol. 2, no. 3, pp. 1680-1687, 615 Jul. 2017. 616
- [21] R. Pito, "A solution to the next best view problem for automated surface 617 acquisition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 10, 618 pp. 1016-1030, Oct. 1999. 619
- [22] S. Chen and Y. Li, "Vision sensor planning for 3-D model acquisition," 620 IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 35, no. 5, pp. 894–904, 621 622 Oct. 2005.
- S. Kriegel, T. Bodenmüller, M. Suppa, and G. Hirzinger, "A surface-623 [23] based next-best-view approach for automated 3D model completion of 624 unknown objects," in Proc. IEEE Int. Conf. Robot. Automat., May 2011, 625 pp. 4869-4874. 626
- [24] M. Botsch, M. Spernat, and L. Kobbelt, "Phong splatting," in Proc. First 627 Eurograph. Conf. Point-Based Graph., Aire-la-Ville, Switzerland, 2004, 628 pp. 25-32. 629

570 571

578

581 04

602

•	Authors: We cannot accept new source files as corrections for your paper. If possible, please annotate the PDF proof	631		
	we have sent you with your corrections and upload it via the Author Gateway. Alternatively, you may send us your	632		
corrections in list format. You may also upload revised graphics via the Author Gateway.				
•	Authors: If you need an invoice or have any other billing questions, please contact reprints@ieee.org as they handle these	634		
	requests.	635		
	QUERIES	636		
Q	1. Author: Please provide the expansion of the acronym "GPU," if required.	637		
$Q^2$	2. Author: Please confirm or add details for any funding or financial support for the research of this article.	638		
Q	3. Author: Please provide page range in Ref. [2].	639		
Q4	4. Author: Please provide full page range in Ref. [9].	640		

630

GENERAL INSTRUCTIONS