A Global Max-Flow-Based Multi-Resolution Next-Best-View Method for Reconstruction of 3D Unknown Objects

Sicong Pan and Hui Wei

***Abstract*—Many robot tasks, such as grasping and inspection, may require complete 3D models with enough surface details. Fully autonomous environment exploration and 3D reconstruction of unknown objects are challenging for a robot when little or no knowledge about an object is known as a priori. Previous work updated sensor measurements from a greedy ordered set of views into a map with an unchangeable voxel size, leading to the lack of details on the object surface, sampling problems, and no adaption to small objects. We propose a global max-flow-based multi-resolution next-best-view (NBV) method to improve performance on these problems. In particular, it utilizes a max-flow-based global view quality function to obtain optimal NBVs, and a multi-resolution strategy to optimize the reconstruction quality and efficiency. Results of comparative simulation experiments with state-of-the-art (SOTA) methods show that our method achieves a higher voxel coverage under the same resolution. Parametric and ablation studies confirm that the global view function and multi-resolution strategy are effective. Results of real-world experiments show that our method achieves complete reconstruction of small objects with only 6–8 views.**

***Index Terms*—Autonomous agents, motion and path planning, probability and statistical methods.**

I. Introduction

IT IS an essential ability of autonomous robots to interact with the environment. For example, when carrying out grasping tasks with precise manipulation, a robot must robustly perceive the environment and the surface details of an object to avoid the failure of subsequent tasks due to a lack of precision. Under most circumstances, we do not know the object as a priori, which may have different sizes, locations, and surface details. The autonomous robot requires data collected from sensors at different positions and poses to reason and reconstruct the object and the environment.

Manuscript received September 24, 2021; accepted November 25, 2021. Date of publication December 3, 2021; date of current version December 14, 2021. This letter was recommended for publication by Associate Editor Berk Calli and Editor Markus Vincze upon evaluation of the reviewers' comments. This work was supported in part by the NSFC Project under Grant 61771146, and in part by the National Thirteen 5-Year Plan for Science and Technology under Grant 2017YFC1703303. *(Corresponding author: Hui Wei.)*

The authors are with the Laboratory of Algorithms for Cognitive Models, Shanghai Key Laboratory of Data Science, School of Computer Science, Fudan University, Shanghai 200438, China (e-mail: 18210240033@fudan.edu.cn; weihui@fudan.edu.cn).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3132430, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3132430

Such an exploration can be accomplished by a next-best-view (NBV) planning task, whose goal is to generate a complete 3D model of an object. Most methods of the task rely on stochastic state analysis [1], [2] or pre-learned knowledge [3] to iteratively obtain the NBV after each robotic scan. However, the greedy iterative selection leads to bad performance on view evaluation [4], lacking details on the object surface. Moreover, the fixed resolution of the volumetric occupancy grid leads to sampling problems and no adaption to small objects under a large number of candidate views. To sum up, a more effective NBV method is necessary.

We seek global optimality for overcoming the bad performance of the greedy selection of NBV planning. This is usually solved as a set cover optimization problem with an object prior 3D model [4]. However, it is not suitable for NBV planning since there is no prior knowledge.

To design a global optimal model with uncertainty, we propose a max-flow-based multi-resolution NBV method to autonomously explore and reconstruct an object without prior knowledge of the object or its bounding box (BBX). The key is to use a global view quality function, which combines a probabilistic local information gain and the flow-network information gain. We use max flow to obtain the global optimal views that can cover all interested voxels.

Any view that contains part of the object can be the start of reconstruction. Our algorithm can adaptively choose multi-resolution to optimize the sampling problem and adapt to small objects. It iteratively selects NBVs, navigates the sensor to them, and repeats the process until reconstruction is accomplished. A proposed convergence criterion determines whether the algorithm is finished.

II. Related Work

The NBV planning method can autonomously determine the NBV from the sensor data by maximizing the amount of view information gain. Earlier approaches required a priori knowledge to evaluate information about the views [5]. The size and position of the object are regarded as a priori knowledge [6]. However, this is not easily obtained in advance. Thus, to work without prior knowledge, recent algorithms for 3D object reconstruction can be categorized [7] as data-driven or search-based (probabilistic).

In data-driven algorithms, Mendoza and Vasquez-Gomez [3] *et al.* proposed their method (NBV-Net) to solve NBVs as a

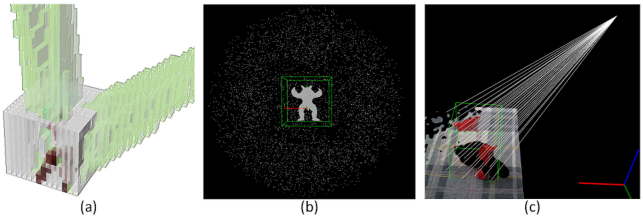

Fig. 1. (a) OctoMap voxels: occupied (red), empty (green), and unknown (grey). (b) Candidate views (white points in the sphere) and the object 3D model for simulation (gray) with its BBX (green). (c) Ray-cast process from a candidate view: rays (white).

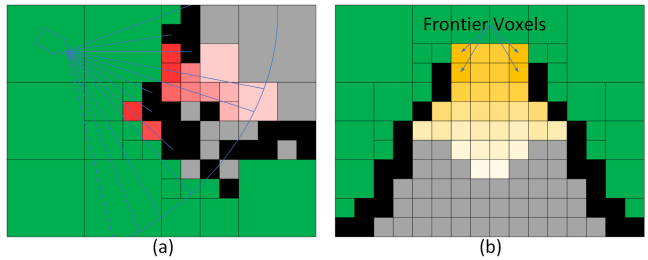

Fig. 2. 2D octomaps: occupied voxels (black), unknown voxels (grey), empty voxels (green), a candidate view (blue camera), rays (blue), and maximal ray range (blue circle). (a) Red shading means visibility. (b) Yellow shading means the probability of belonging to the object.

classification problem. The dataset was expanded, and the regression problem was proposed in their recent work [8]. Rui Zeng [9] *et al.* proposed a point cloud-based trained network (PC-NBV) to predict NBVs. Another data-driven approach utilizes reinforcement learning to converge the evaluation function [10]. Although these methods do not require a prior model, they adapt with difficulty to tasks without appropriate object datasets or sufficient training time.

In probabilistic algorithms, Krainin [11] *et al.* could predict NBVs based on the entropy function. Potthast [12] *et al.* determined NBVs according to the 3D occupancy grid. Vasquez-Gomez [13] *et al.* used the number of visible unknown voxels to calculate NBVs. Isler [14], Vasquez-Gomez [15] and Delmerico [2] *et al.* used the voxel information gain method and proposed rear side entropy (RSE). Andreas Bircher [16] *et al.* proposed a receding horizon NBV method for fast exploration on a random tree. Zehui Meng [17] *et al.* proposed a two-stage optimization method for NBV planning. Daudelin [1] *et al.* proposed the adaptive probabilistic object reconstruction algorithm (APORA) without prior knowledge of the object. Their method can adapt to objects of large sizes.

These algorithms use the fixed resolution of OctoMap so that it is difficult to adapt to small objects and the sampling problem. Another important issue is that current algorithms consider only the predicted NBVs from partial reconstruction, leading to the lack of details on the object surface.

III. Global View Quality Function

We introduce the global view quality function, i.e., view information gain. The goal is to define global optimality, finding a view subset that covers all the voxels. We establish a network flow with the aid of local view information gain to approximately solve the global optimality.

A. Local View Information Gain

A volumetric occupancy grid called OctoMap [18] is chosen to store a probabilistic representation of the environment and the object, which is composed of layered voxels, including occupied, empty, and unknown voxels, as shown in Fig. 1(a). The value of voxel node x in OctoMap is probabilistic and represents the occupancy. Let V be the set of candidate views sampled in 3D space, for example in a sphere as shown in Fig. 1(b). Let R_v be the set of rays cast through a series of pixels sampled from the virtual image plane of view v. The endpoint 3D coordinates of the ray are reprojected by the sampled pixels using the pinhole camera imaging principle. To facilitate understanding, we draw the rays in point cloud space as shown in Fig. 1(c).

Each ray r traverses through a set of voxels X_r in the OctoMap before it hits those occupied voxels or reaches the maximum range. OctoMap contains some partial measurements, and each voxel should have different weights. The local view quality function can be formulated as the weighted amount of information gain:

$$I_{local}(v) = \sum_{\forall r \in R_v} \sum_{\forall x \in X_r} H(x) * P\left(vis_x\right) * P\left(obj_x\right) \tag{1}$$

where $H(x)$ is the entropy function, vis_x is a random binary variable of visibility, and obj_x is a random binary variable belonging to the object surface (object probability). Note that obj_x models the continuity of surfaces based on the fact that most object surfaces are closed or nearly closed.

The probabilistic definitions of the above-mentioned weights are defined in these studies [1], [2]. However, we improved the object probability by defining the K (by default 6) nearest frontier voxels instead of the nearest one. Frontier voxels are the unknown voxels whose adjacent voxels contain at least one empty voxel and one occupied voxel. The three weights can be given as:

$$\begin{aligned} H(x) &= -P_o(x)\ln P_o(x) - \bar{P}_o(x)\ln \bar{P}_o(x) \\ P\left(vis_x\right) &= \prod_y \left(1 - P_o(y)\right) \\ P\left(obj_x\right) &= \prod_y P\left(obj_x^y\right) = \prod_y e^{-\left(\alpha * d_y^x\right)^2} \end{aligned} \tag{2}$$

where $P_o(x)$ is the occupancy of voxel x in OctoMap, $\bar{P}_o(x)$ is the complement probability, voxels y are those traversed along a ray from view v before it hits voxel x, and α is a variable related to the OctoMap resolution (by default 0.1/resolution), and d_y^x is the Euclidean distance between voxel x and voxels y. Fig. 2 shows a 2D example of a partially observed object and these three weights.

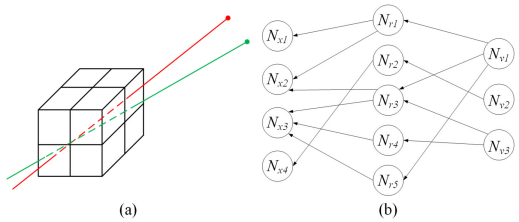

Fig. 3. (a) Relationship between views and rays: a red view and a green view share the same ray because their original rays (red and green) travel the same voxels. (b) Tripartite graph G_{tri}: view (N_{vi}), ray (N_{ri}), and voxel (N_{xi}).

B. Global Optimality

With the local view quality function, a simple greedy strategy is to iteratively choose the view that maximizes (1) until termination [1], [2]. Although the greedy algorithm is complete, there may be a large intersection between two voxel sets belonging to two chosen views.

To solve this global problem, it is common to define global optimality as covering all voxels with the minimum set of views [4]. Unfortunately, this problem is equivalent to the set cover problem, which was first introduced in Karp's 21 NP-complete problems and was later proved to be NP-hard [19]. The equivalent transformation of this problem is as follows (here we ignore the rays). Given a set of voxels $U = 1, 2, 3, \ldots, n$ (called the universe) and a collection S of view sets whose union equals the universe, the set cover problem is to identify the smallest m sub-collections of S whose union equals the universe, i.e., $S_1 \cup S_2 \cup \ldots \cup S_m = U$. For example, consider the voxel set $U = \{1, 2, 3, 4, 5\}$ and the collection of view sets $S = \{\{1, 2, 3\}, \{2, 4\}, \{3, 4\}, \{4, 5\}\}$. Clearly, the union of S is U. However, we can cover all of the voxels with a smaller number of sets, $m = 2 : \{\{1, 2, 3\}, \{4, 5\}\}$.

Since the direct solution is NP-hard, we seek the approximate global optimality. We review the process of solving for local view information gain and find a structure among the data. If we regard the view, ray, and voxel as three kinds of vertex, then a classic tripartite directed graph $G_{tri} = (N, E)$ can be formed. If we mark a vertex as N, then we have three types of vertices, N_v for the view, N_r for the ray, and N_x for the voxel; and two edges, $< N_v, N_r > \in E$ for the relationship between the view and ray, and $< N_r, N_x > \in E$ for that between the ray and voxel. Fig. 3(a) shows how different views can share the same ray. Fig. 3(b) shows a tripartite graph with three views, five rays, and four voxels.

We define approximate global optimality as finding a view subset V_{global}, covering all voxels with maximum information gain. Note that here we relax the constraint of minimum size so that V_{global} can be solved in polynomial complexity time. Thus, we can transform the demand of approximate global optimality to a graph optimization problem, as follows.

Definition 1: Graph optimization for approximate global optimality. Given a view-ray-voxel tripartite graph $G_{tri} = (N, E)$ with information $H(x) * P(vis_x) * P(obj_x)$ on edge $< N_r, N_x > \in E$, the graph optimization problem for approximate global optimality is to find a subset of N_v (i.e., V_{global}) covering all voxels N_r by the corresponding paths whose total information is maximized.

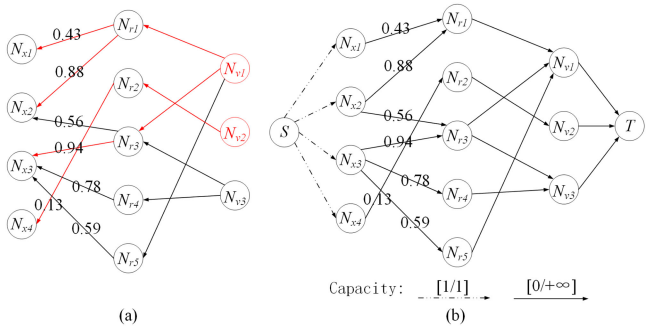

Fig. 4. (a) Graph optimization for approximate global optimality: red vertices and edges indicate the covering with maximum information. (b) Flow network: nonzero quality values of edges and capacities of edges with the double-dotted and solid line.

To cover on graph means that there is at least one directed path between two graph vertices. We put the information gain on the edges to force it to choose the paths of maximum information to cover all voxel vertices. As shown in Fig. 4(a), we may choose N_{v1} and N_{v2} to cover all the voxels with the maximum information of all chosen paths (the sum of information values on the red edges). For example, the path used to cover N_{x2} is $\{< N_{v1},\ N_{r1} >, < N_{r1},\ N_{x2} >\}$ instead of $\{< N_{v1},\ N_{r3} >, < N_{r3},\ N_{x2} >\}$.

C. Network Flow Modeling

We generate a new flow network, $G_{flow} = (N, E)$, to solve this graph optimization, i.e. to "flow" the information of all the voxels into a subset of views. The graph vertex N in G_{flow} takes the vertex in G_{tri} unchanged, and edge E must take an edge from G_{tri} and reverse it, i.e., $< N_r, N_v > \in E$, $< N_x, N_r > \in E$. Each edge has a nonnegative integer capacity with an upper bound and a lower bound. If $< N_i, N_j > \in E$, then $C_{min}(N_i, N_j) = 0$ and $C_{max}(N_i, N_j) = +\infty$; if $< N_i, N_j > \notin E$, then $C_{min}(N_i, N_j) = 0$ and $C_{max}(N_i, N_j) = 0$. In addition, since there is no clear source or sink in G_{flow}, we add two vertices, super source S and super sink T. We also must add the directed edge $< S, N_x >$ between S and N_x with capacity $C_{min}(S, N_x) = 1$ and $C_{max}(S, N_x) = 1$, and the directed edge $< N_v, T >$ between N_v and T with capacity $C_{min}(N_v, T) = 0$ and $C_{max}(N_v, T) = +\infty$. Each edge also may have a quality value. If the edge is between the voxel and the ray, then $Q(N_x, N_r) = H(x) * P(vis_x) * P(obj_x)$, and otherwise $Q(N_i, N_j) = 0$. Fig. 4(b) shows an example of G_{flow}.

Network flow is a integer function $f : N \times N \to Z$, $f(N_i, N_j)$ represent the flow from N_i to N_j, which must satisfy three requirements [20]. Based on the flow, we can formulate flow f_S from the source and Q_T as the quality that flows into the sink as:

$$f_S = \sum_{n \in N} f(S, n), Q_T = \sum_{< N_i, N_j > \in E} Q(N_i, N_j) * f(N_i, N_j) \tag{3}$$

Obviously, we can use $f_S = |N_x|$ to ensure that all voxels will be covered by a view subset, i.e., each voxel will eventually find a path to a view through a flow, and we can use $maxQ_T$ to ensure that all voxels will be covered with maximum information,

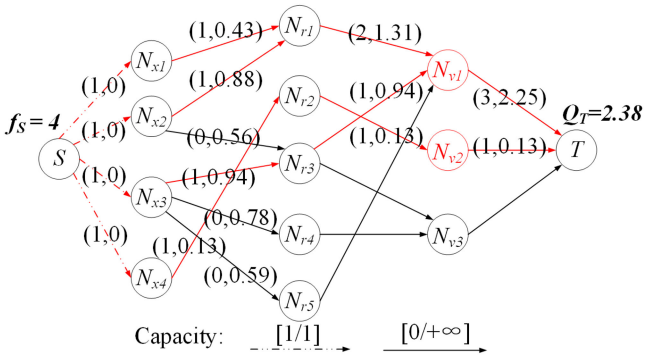

Fig. 5. Feasible maximum flow: red vertices in V_{min} and red edges with a pair (f, Q) of nonzero flow and cumulative quality value that added from previous red edges or itself.

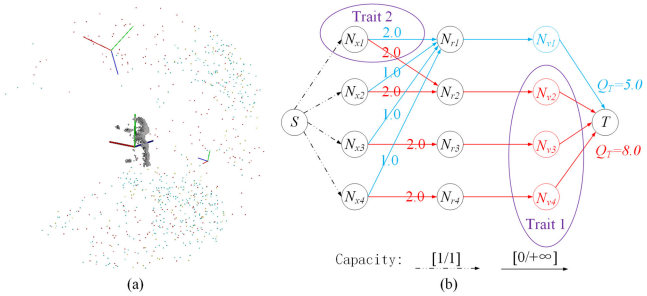

Fig. 6. (a) Evaluated candidate views: in V_{global} (red), in top 10% local-quality view set (cyan), and in both (yellow). (b) Traits of flow: flow for V_{global} (red), flow for ideal view subset with minimum size (blue), and intuitive explanation of traits (purple).

i.e., the total information of flows will be maximized. Thus, we define the feasible flow as $f_S = |N_x|$, and the maximum flow as $maxQ_T$, where $||$ is the size of the set. Feasible flow always exists because the worst case is to select all views to cover all voxels. So, to solve for $|V_{global}|$ is now to solve for $f_S = |N_x| \wedge maxQ_T$ in G_{flow}. Fig. 5 shows a solution of the feasible maximum flow in G_{flow}. We can see that the feasible flow covers all voxels ($f_S = 4$) with maximum information ($maxQ_T = 2.38$).

For such a network flow model with upper and lower bounds, the successive shortest path [21] is used to solve this. We use the shortest path faster algorithm (SPFA) [22] for finding the shortest path. In the worst case, it has complexity $O(|N| * |E| * F)$, where $|N|$ is the size of vertices, $|E|$ is the size of edges, and F is the value of flow.

D. Flow-Network Information Gain

Fig. 5 shows the ideal solution of the network flow model, i.e., the size of V_{global} is also a minimum. However, in the early iterations, the feasible maximum flow may obtain V_{global} with a size of 100 or more if there are 5,000 candidate views, as shown in Fig. 6(a). Thus, in the case that all views in V_{global} cannot be traversed at once, we have to consider an iterative approach, i.e., to give weight to those views for view information gain.

A simple way to give weight is to consider the local information gain for views in V_{global}, as defined below in subsection III-E. However, this does not fully consider the traits of the flow network. We aim to design the weight function by considering the traits of the flow itself. Fig. 6(b) shows the two main traits of flow: (1) each voxel finds its own optimal view by its flow, i.e., the measurement of this view contains the maximum information gain of this voxel; and (2) each voxel may have multiple optimal views, but only one of them will be randomly obtained by its flow. Hence, a view in V_{global} is the optimal view belonging to some voxels.

Let it be the current number of iterations, and V^k_{global} the subset of views solved at iteration k, $k = 0, 1, 2, \ldots, it$. Based on the traits of flow, if a view in V^{it}_{global} is also in the previous solutions, it is more likely to be the optimal view belonging to more voxels. However, as the iteration continues, some voxels in a previous OctoMap may have been covered. Hence, the earlier set of views is solved, and less weight is given to it. For convenience, an exponential function is chosen to provide the history attribute:

$$\text{his}\,(v,k) = \begin{cases} e^{-(it-k)} & \text{if } v \in V^k_{global} \\ 0 & \text{if } v \notin V^k_{global} \end{cases} \tag{4}$$

So, we can formulate the flow-network quality function as:

$$I_{\text{flow}}\,(v) = E\left[\sum_{k=0}^{it} \text{his}(v,k)\right] \tag{5}$$

E. Global View Quality Function

To comprehensively consider the local and flow-network information, we normalize them and formulate them as:

$$I_{\text{global}}\,(v) = (1-\gamma) * \frac{I_{\text{local}}\,(v)}{\sum_v I_{\text{local}}\,(v)} + \gamma * \frac{I_{\text{flow}}\,(v)}{\sum_v I_{\text{flow}}\,(v)} \tag{6}$$

where $\sum_v I_{local}(v)$ and $\sum_v I_{flow}(v)$ are the total local information and network flow information, respectively, predicted for the current iteration over all view candidates, and $\gamma \in [0, 1]$ is a network flow weight (by default 0.5). The NBV v^* at the current iteration can be found from $I_{global}(v)$ as:

$$v^* = \arg\max_{v \in V} I_{\text{global}}\,(v) \tag{7}$$

IV. Global Max-Flow-Based Multi-Resolution Next-Best-View Method

In the real world, it is hard to obtain the size of the object in advance. We analyzed multi-resolution for small objects and the sampling problem of ray casting. A novel NBV method is proposed, and computation is optimized.

A. Dynamic BBX and Highest Resolution Analysis

Adapting object size in the real world has long been discussed in APORA algorithm [1]. It adopted a method which we call it Dynamic BBX. As shown in Fig. 7(a) and (b), as more point clouds of the cup object are observed, the BBX surrounding the cup object expands automatically. This helps to reconstruct large objects. However, this method does not work well on small objects. As OctoMap is a voxelization, the bigger the voxel (i.e., the lower the resolution), the less information is contained, and

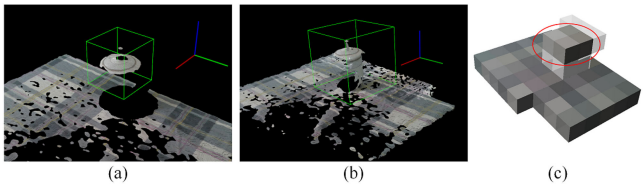

Fig. 7. Dynamic BBX and low-resolution OctoMap. (a) The point cloud of the first iteration and BBX (green). (b) The point cloud of the second iteration and BBX (green). (c) The cup object with four voxels (red) in OctoMap.

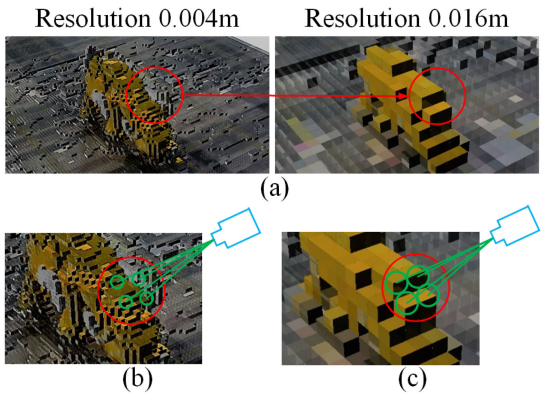

Fig. 8. Illustration of sampling problem and multi-resolution: distribution of unknown voxels (grey), labeled regions (red circle), the emergence of unknown voxels (red line), candidate views (blue), rays (green line), and hit voxels (green circle). (a) Multi-resolution OctoMaps. (b) Sampling problems. (c) The comprehensiveness of low resolution.

vice versa. As shown in Fig. 7(c), there is a worst-case scenario where we only use four voxels of information to reconstruct the cup object in low resolution. Therefore, it is necessary to use a high OctoMap resolution for small objects.

B. Sampling Problem and Multi-Resolution Analysis

However, with high resolution comes sampling problems, i.e., the loss of information gain. Fig. 1(c) shows a visual ray-cast process from a candidate view. The real ray-cast stage is time-consuming, so we have to use a down-sampling method [2]. This will result in the inability to obtain the full voxel information. As shown in Fig. 8(b), four rays uniformly casting from a view on a high resolution hit four unknown voxels, but more unknown voxels in the labeled region were not hit, thereby losing the information gain.

As shown in Fig. 8(a), the distribution of unknown voxels varies with the resolution. Small unknown voxels of higher resolution may be merged with surrounding voxels of lower resolution. Hence, the information gain of a low resolution will be less precise but more comprehensive than that of a high resolution. As shown in Fig. 8(c), these rays hit all unknown voxels in the labeled region on a low resolution. A multi-resolution strategy is proposed to deal with the sampling problem, balancing precision and comprehensiveness.

C. Multi-Resolution Strategy and Convergence Criterion

Our multi-resolution strategy is to choose an initial low resolution according to the size of the object (making sure that there are at least 12 voxels on the edge of object BBX) and to adaptively increase it until the highest resolution (user-defined or conforming to sensor accuracy). Because the OctoMap is already a multi-resolution data structure [18] (an octree), it is convenient to switch among multiple resolutions.

Algorithm 1: Global Max-flow-based Multi-resolution Next-best-view Method.

Require: P_0
$M \leftarrow$ update with P_0
Generate V and initial resolution res
while res *is lower than highest resolution* **do**
  $v^* \leftarrow \arg\max_{v \in V} I_{\text{global}}(v)$
  Navigate to v^*
  $M \leftarrow$ update with sensor data
  while $Entropy(M) < q_{thresh} * Full(M)$ **do**
    increase resolution res
  end while
  switch M to resolution res
end while

Map entropy is used to provide the attribute of adaptive increase:

$$Entropy(M) = \sum_{x \in BBX} H(x), Full(M) = \sum_{x \in BBX} H(0.5) \tag{8}$$

where BBX is the predicted bounding box of the object on OctoMap M. A convergence criterion is proposed to determine whether the OctoMap is convergent:

$$Entropy(M) < q_{thresh} * Full(M) \tag{9}$$

where q_{thresh} is a threshold for OctoMap convergence (by default 0.5). Algorithm 1 shows the architecture of the global max-flow-based multi-resolution NBV method, which requires initial point cloud P_0 of the sensor. Such an algorithm can reconstruct a priori unknown objects and adapt to the problems of size, sampling, and the lack of surface details.

D. Optimization of Computation Efficiency

We optimize the two main time-consuming parts of our algorithm: (1) ray casting; and (2) tripartite graph generation. For the ray casting stage, a hash table is used to store the correspondence between ray and view so that we can separate their calculation. If the time complexity of voxel information function is $O(1)$, the native complexity $O(it * |V| * |R_v| * |X_r|)$ is reduced to $O(lv * |V| * |R_v| + it * |R_{all}| * |X_r|)$, where $|R_{all}|$ is the number of all rays, it is the number of iterations, lv is the number of used OctoMap resolutions. For tripartite graph generation, a pruning strategy is used to merge the graph vertices and delete useless edges according to the relationship shown in Fig. 3(a).

V. Experiments

To evaluate the effectiveness of the improvements, we compared them with three SOTA methods (APORA [1], RSE [2],

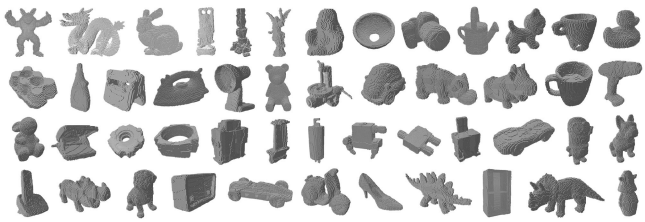

Fig. 9. 3D models of objects from three datasets.

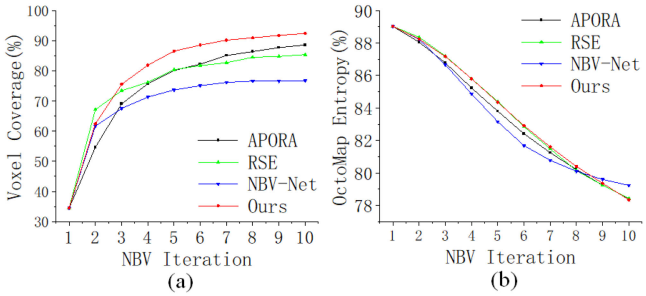

Fig. 10. Comparison of reconstruction process: (a) voxel coverage; (b) OctoMap entropy.

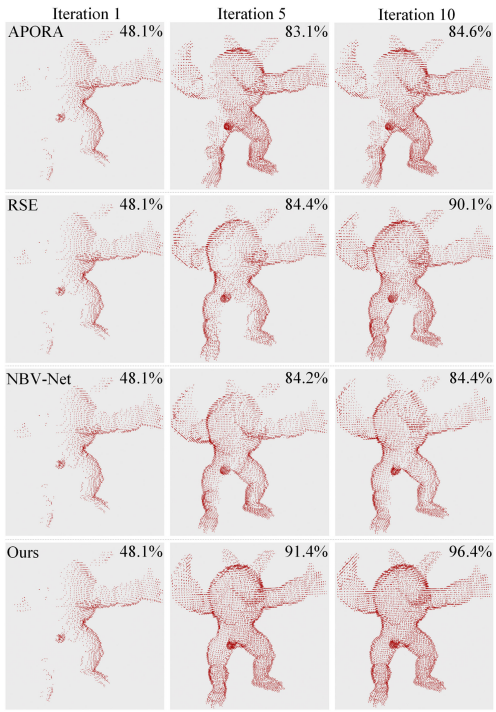

Fig. 11. Reconstruction progress comparison on Armadillo model of iterations 1, 5, and 10, with voxel coverage on the upper-right corner.

and NBV-Net [3]). APORA and RSE are search-based probabilistic methods while NBV-Net is a data-driven method. Experiments were conducted on a PC with an Intel Xeon CPU E5-2640. The code of our system will be made publicly available (github.com/psc0628/NBV-Simulation). A video of the simulation and real-world experiments can be viewed at youtu.be/PWenwK0m9-0.

Scanned complex 3D models for simulation are from Stanford 3D Scanning Repository (graphics.stanford.edu/data/3Dscanrep/), Linemod, and HomebrewedDB (bop.felk.cvut.cz/datasets/); for simplicity, we call them Stanford, LM, and HB, respectively. Fig. 9 shows six object models from Stanford which have their own names, 12 models from LM, and 31 models from HB.

A. Experiments on View Quality Function

1) Setup and Indicators: For a fair comparison, we tested view quality functions of all methods under the same OctoMap resolution of 0.01 m and started the reconstruction from the same view. We sampled 5,000 views and iterated the algorithms 10 times for each object. We used two indicators to evaluate the effectiveness of the algorithms. The voxel coverage is defined as the ratio of the number of discovered voxels (observed by the virtual camera) and the number of voxels in the ground truth (sampled by the object 3D model). The OctoMap entropy is defined as the sum of the entropy of all voxels in object BBX, as defined in (8).

2) Results of Reconstruction Process: Fig. 10 shows the reconstruction process of all four methods on average. Fig. 10(a) shows that our method can reconstruct the complete object with higher voxel coverage than all other methods. Fig. 10(b) shows that as the iteration goes on, our method will eventually achieve lower OctoMap entropy. This is because our algorithm does not use the greedy rule as the comparison algorithms do. To give more intuitive reconstruction results, we show a comparison of the reconstruction process on the Armadillo model as shown in Fig. 11. The full results of each iteration can be seen in our video.

3) Results of Objects: Because of a constraint in the number of pages, we cannot show statistical figures of every object here. Instead, we show the voxel coverage of the final iteration of each object in Table I. Results show that our method performs best for most of the models and achieves the highest score for voxel coverage.

4) Computational Efficiency: In Table II, averaged extra memory, training time, and computing time per iteration are reported for four methods. Note that the time for computing the hash-table is divided by 10 (number of iteration) and added into computing time. Compared with the data-driven method, our method does not need to spend a lot of time for training. Compared with the search-based method, our method requires a little more memory and time, but it achieves satisfactory object reconstruction results.

B. Multi-Resolution and Parametric Studies

1) Multi-Resolution Setup and Results: Most configurations were the same as those of experiments on the view quality function. We scaled the object 3D models of different sizes to the same size (about 0.1 m) so that the lowest resolution could be the same (0.04 m). Note that only search-based methods can be applied to the multi-resolution strategy. Table III shows the total averaged voxel coverage and computing times of each method. The ablation results confirm that the multi-resolution strategy can improve reconstruction quality and efficiency. The

TABLE I
VOXEL COVERAGE OF EACH OBJECT AND METHOD

Datasets	Objects \ Methods	APORA [1]	RSE [2]	NBV-Net [3]	Ours
Stanford	Armadillo	84.6	90.1	84.4	**96.4**
	Dragon	88.5	85.9	75.4	95.1
	Bunny	88.6	83.7	78.0	**92.7**
	Buddha	91.8	**92.5**	85.2	92.0
	Thai	89.1	88.4	89.1	**90.9**
	Lucy	92.1	93.1	86.6	**93.1**
	Average	89.1	88.9	83.1	**93.4**
LM (Linemod)	LM1	89.6	81.1	83.9	**93.6**
	LM2	96.3	**97.3**	89.4	91.8
	LM3	**95.0**	80.8	76.7	93.4
	LM4	95.2	90.5	66.2	**95.2**
	LM5	58.4	92.7	76.7	**94.3**
	LM6	**90.2**	74.3	86.6	89.3
	LM7	72.9	92.5	78.4	**92.5**
	LM8	73.7	81.3	60.4	**90.4**
	LM9	93.0	94.4	90.4	**94.7**
	LM10	97.2	86.2	65.2	**98.5**
	LM11	**93.8**	80.9	70.4	91.6
	LM12	95.8	95.7	87.8	**98.7**
	Average	87.6	87.3	77.7	**93.7**
HB (HomebrewedDB)	HB1	79.2	85.9	89.8	**89.8**
	HB2	84.8	87.1	81.7	**90.9**
	HB3	94.7	87.1	76.8	**94.9**
	HB4	89.3	66.0	67.8	**89.3**
	HB5	93.1	80.4	57.7	**95.2**
	HB6	**96.3**	79.4	84.5	90.0
	HB7	86.8	94.3	86.9	**95.8**
	HB8	90.5	79.4	82.7	**92.4**
	HB9	76.5	92.6	57.8	**94.6**
	HB10	**88.4**	65.7	75.1	86.7
	HB11	**96.5**	74.8	59.0	95.4
	HB12	95.6	89.2	86.0	**96.7**
	HB13	96.6	88.3	73.9	**96.6**
	HB14	85.4	**88.7**	34.7	85.5
	HB15	95.4	94.1	66.9	**96.7**
	HB16	**89.3**	83.6	68.6	88.7
	HB17	96.9	95.5	85.4	**97.6**
	HB18	**94.8**	82.6	79.5	94.1
	HB19	83.8	58.7	**89.5**	87.0
	HB20	**86.7**	79.4	77.8	83.5
	HB21	82.3	**87.1**	81.1	84.1
	HB22	59.0	74.7	61.8	**87.8**
	HB23	87.7	93.5	81.5	**95.7**
	HB24	91.8	83.9	55.8	**97.9**
	HB25	**96.5**	92.9	77.1	95.2
	HB26	89.7	71.6	74.6	**93.0**
	HB27	89.2	84.2	78.1	**90.0**
	HB28	93.5	87.8	**93.7**	93.0
	HB29	92.4	89.6	82.4	**93.9**
	HB30	90.8	**91.7**	67.5	89.3
	HB31	80.4	93.6	92.8	**94.2**
	Average	88.8	84.0	75.1	**92.1**
Total Average		88.6	85.4	76.7	**92.6**

TABLE II
COMPUTATIONAL EFFICIENCY

Methods	APORA	RSE	NBV-Net	Ours
Extra Memory (MByte)	85.62	0	6.038	133.1
Training Times (seconds)	0	0	5400 [3]	0
Computing Times (seconds)	6.416	262.3	0.011	6.972

sampling problem and adaption to small objects are addressed for real-world reconstruction.

2) Parametric Studies: The results of the parametric study of the history weight function $his(v, k)$ and network flow weight γ are shown in Table IV. No history function means that the weights of views in V_{global} are 1. The results confirm that the history weight function is effective, and the value of γ is suitable. The parametric study of the threshold q_{thresh} for OctoMap convergence is shown in Table V. The results confirm that the value of q_{thresh} is suitable for each search-based method.

TABLE III
MULTI-RESOLUTION ABLATION STUDY OF EACH METHOD

Method	APORA	APORA +multi	RSE	RSE +multi	Ours	Ours +multi
Voxel Coverage(%)	88.6	**88.9**	85.4	**85.8**	92.6	**93.0**
Computing Times (s)	6.416	**3.922**	262.3	**138.8**	6.972	**4.558**

TABLE IV
PARAMETRIC STUDY OF NETWORK FLOW ON VOXEL COVERAGE

γ	0	0.25	**0.5**	0.75	1
Ours+no-his	**89.6**	88.4	88.6	89.0	60.6
Ours	89.6	91.8	**92.6**	92.0	91.7

TABLE V
PARAMETRIC STUDY OF MULTI-RESOLUTION ON VOXEL COVERAGE

q_{thresh}	0	0.25	**0.5**	0.75	1
APORA	88.7	88.8	**88.9**	88.8	88.6
RSE	85.2	85.4	**85.8**	85.5	85.4
Ours	92.6	92.7	**93.0**	92.7	92.6

TABLE VI
TOTAL RUNNING TIME OF EACH METHOD

Method	APORA	RSE	NBV-Net	Ours
Running Time (minutes)	18.9	25.4	12.4	13.6

C. Real-World Experiments

1) Robot System: Our lab is equipped with a STEP 6-DOF robotic arm with a RealSense D435i RGBD camera mounted on its end-effector. In our robot sensor processing, since there is no calibration process in the initial stage, we regress the point clouds to reduce the error. Therefore, like Sfm [23], our robot stops many times to acquire multiple views and collect data from the last view for reconstruction. In our robot controller processing, since the pose is not important for object reconstruction [8], we assume that the camera always points to the object. The depth camera needs a distance (about 0.3 m) from the object to correctly collect the point clouds, making a sphere of object BBX an obstacle. We plan the path straight at first, and if the robot encounters obstacles, we let it go around them.

2) Setup: Most parameters were the same as in the simulation experiment. We used 0.005 m as the highest resolution. Our output was a 3D model registered by multi-view point clouds. Since the data of the depth camera will have a large error where the object contacts the table, we filtered out points below 0.002 m. Our method considers all information from accessible views. In our video, the missing surfaces are due to the absence of accessible views.

3) Reconstruction Results: As shown in Fig. 12, we rearranged the positions of the results and tried to display the point cloud from the direction observed. Our approach adaptively chose the OctoMap resolutions. As shown in Fig. 13, our approach had better and faster reconstruction, while the comparison methods lost the object surface details. Table VI reports the running time of each method. Assuming a task time limit of 15 minutes, our method can complete the task well.

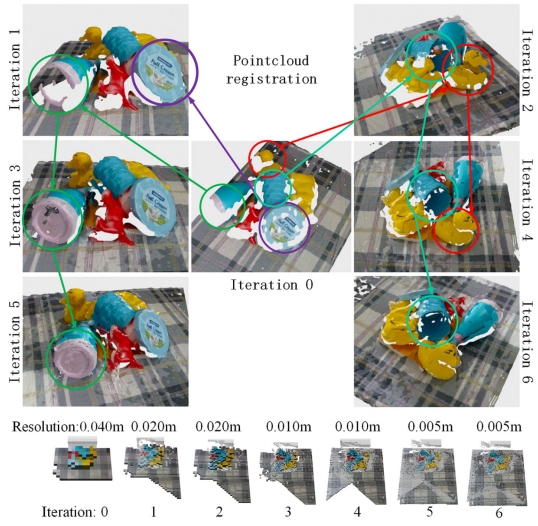

Fig. 12. Reconstruction progress of point cloud registration (red, green, purple, and cyan) and multi-resolution OctoMaps.

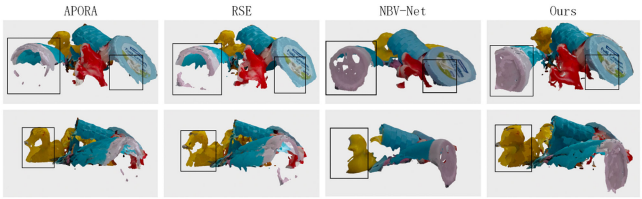

Fig. 13. Comparison of reconstruction result: the major surface differences (black box).

TABLE VII
ROBOT MOVEMENT STUDY

θ	0	0.25	0.5	0.75	1
Voxel Coverage (%)	**93.0**	92.8	92.2	90.4	48.7
Movement Cost (m)	4.568	4.221	3.850	3.276	**0.251**

D. Robot Movement Analysis

As shown in Fig. 12, the view positions of adjacent iterations may differ greatly, but the views that have short distances to the visited views may be visited in the following iterations. Thus, we analyze this problem by a view quality function with the robot movement cost [14]:

$$I_{Robot}(v) = (1-\theta) \times \frac{I_{Global}(v)}{\sum_v I_{Global}(v)} - \theta \times \frac{I_{Move}(v)}{\sum_v I_{Move}(v)}$$

where θ is a weight of robot cost and $I_{Move}(v)$ can be calculated as the Euclidean distance from v to the current NBV. Table VII reports on the robot movement study of our method and shows that as the weight increases, the voxel coverage decreases.

VI. CONCLUSION

We presented a global max-flow-based multi-resolution NBV object reconstruction method, including a global optimality formulation. Although NBV algorithms have been discussed in the literature, to our knowledge, we are the first to use network flow to reconstruct an object and actively adapt to surface details, object sizes and sampling problems without prior knowledge. Experiments confirmed that the proposed method improves these problems. Improvements remain to be addressed in future work. One is to use a Markov random process for probabilistic modeling. Another is to study the non-iterative flow-based method, which could be used to balance the total movement distance and surface details.

REFERENCES

[1] J. Daudelin and M. Campbell, "An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3D objects," *IEEE Robot. Automat. Lett.*, vol. 2, no. 3, pp. 1540–1547, Jul. 2017.
[2] 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, 2018.
[3] M. Mendoza, J. I. Vasquez-Gomez, H. Taud, L. E. Sucar, and C. Reta, "Supervised learning of the next-best-view for 3D object reconstruction," *Pattern Recognit. Lett.*, vol. 133, pp. 224–231, 2020.
[4] M. D. Kaba, M. G. Uzunbas, and S. N. Lim, "A reinforcement learning approach to the view planning problem," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5094–5102.
[5] E. Dunn, J. Van DenBerg, and J.-M. Frahm, "Developing visual sensing strategies through next best view planning," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2009, pp. 4001–4008.
[6] S. Wenhardt, B. Deutsch, E. Angelopoulou, and H. Niemann, "Active visual object reconstruction using D-, E-, and T-optimal next best views," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–7.
[7] R. Zeng, Y. Wen, W. Zhao, and Y.-J. Liu, "View planning in robot active vision: A survey of systems, algorithms, and applications," *Comput. Vis. Media*, vol. 6, pp. 225–245, 2020.
[8] J. I. Vasquez-Gomez, D. Troncoso, I. Becerra, E. Sucar, and R. Murrieta-Cid, "Next-best-view regression using a 3D convolutional neural network," *Mach. Vis. Appl.*, vol. 32, no. 2, pp. 1–14, 2021.
[9] R. Zeng, W. Zhao, and Y.-J. Liu, "Pc-Nbv: A point cloud based deep network for efficient next best view planning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 7050–7057.
[10] D. Peralta, J. Casimiro, A. M. Nilles, J. A. Aguilar, R. Atienza, and R. Cajote, "Next-best view policy for 3D reconstruction," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 558–573.
[11] M. Krainin, B. Curless, and D. Fox, "Autonomous generation of complete 3D object models using next best view manipulation planning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 5031–5037.
[12] C. Potthast and G. S. Sukhatme, "A probabilistic framework for next best view estimation in a cluttered environment," *J. Vis. Commun. Image Representation*, vol. 25, no. 1, pp. 148–164, 2014.
[13] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "Hierarchical ray tracing for fast volumetric next-best-view planning," in *Proc. Int. Conf. Comput. Robot Vis.*, 2013, pp. 181–187.
[14] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza, "An information gain formulation for active volumetric 3D reconstruction," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 3477–3484.
[15] J. I. Vasquez-Gomez, L. E. Sucar, and R. Murrieta-Cid, "View/state planning for three-dimensional object reconstruction under uncertainty," *Auton. Robots*, vol. 41, no. 1, pp. 89–109, 2017.
[16] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "Next-Best-View" planner for 3D exploration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1462–1468.
[17] Z. Meng *et al.*, "A two-stage optimized next-view planning framework for 3-D unknown environment exploration, and structural reconstruction," *IEEE Robot. Automat. Lett.*, vol. 2, no. 3, pp. 1680–1687, Jul. 2017.
[18] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robots*, vol. 34, no. 3, pp. 189–206, 2013.
[19] B. H. Korte, J. Vygen, B. Korte, and J. Vygen, *Combinatorial Optimization*. Berlin, Germany: Springer, vol. 1, 2011.
[20] D. Gale *et al.*, "A theorem on flows in networks," *Pacific J. Math.*, vol. 7, no. 2, pp. 1073–1082, 1957.
[21] H. S. Wilf, *Algorithms and Complexity*. Boca Raton, FL, USA: AK Peters/CRC Press, 2002.
[22] D. Fanding, "A faster algorithm for shortest-path-SPFA," *J. Southwest Jiaotong Univ.*, vol. 2, no. 9, pp. 207–212, 1994.
[23] O. Ozyesil, V. Voroninski, R. Basri, and A. Singer, "A survey of structure from motion," *Acta Numerica*, vol. 26, pp. 305–364, 2017.