entropy

Article

A Generic View Planning System Based on Formal Expression of Perception Tasks [†]

Yanzi Kong [1,2,3,4], Feng Zhu [1,2,3,*], Haibo Sun [1,2,3,5], Zhiyuan Lin [1,2,3,4] and Qun Wang [1,2,3,4]

[1] Key Laboratory of Opto-Electronic Information Processing, Chinese Academy of Sciences, Shenyang 110169, China; kongyanzi@sia.cn (Y.K.); sunhaibo@sia.cn (H.S.); linzhiyuan@sia.cn (Z.L.); wangqun@sia.cn (Q.W.)
[2] Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110169, China
[3] Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China
[4] University of Chinese Academy of Sciences, Beijing 100049, China
[5] Faculty of Robot Science and Engineering, Northeastern University, Shenyang 110819, China
[*] Correspondence: fzhu@sia.cn

[†] This paper is an extended version of our paper published in 2021 IEEE International Conference on Robotics and Biomimetics.

Abstract: View planning (VP) is a technique that guides the adjustment of the sensor's postures in multi-view perception tasks. It converts the perception process into active perception, which improves the intelligence and reduces the resource consumption of the robot. We propose a generic VP system for multiple kinds of visual perception. The VP system is built on the basis of the formal description of the visual task, and the next best view is calculated by the system. When dealing with a given visual task, we can simply update its description as the input of the VP system, and obtain the defined best view in real time. Formal description of the perception task includes the task's status, the objects' prior information library, the visual representation status and the optimization goal. The task's status and the visual representation status are updated when data are received at a new view. If the task's status has not reached its goal, candidate views are sorted based on the updated visual representation status, and the next best view that can minimize the entropy of the model space is chosen as the output of the VP system. Experiments of view planning for 3D recognition and reconstruction tasks are conducted, and the result shows that our algorithm has good performance on different tasks.

Keywords: active perception; view planning; information expression; next best view; entropy reduction

check for updates

Citation: Kong, Y.; Zhu, F.; Sun, H.; Lin, Z.; Wang, Q. A Generic View Planning System Based on Formal Expression of Perception Tasks. *Entropy* **2022**, *24*, 578. https://doi.org/10.3390/e24050578

Academic Editors: Qiang Zhang and Yifeng Zeng

Received: 24 February 2022
Accepted: 18 April 2022
Published: 20 April 2022

1. Introduction

Perception is a crucial operation for robots to understand and express the surrounding environment. It is defined as a task of collecting and processing data obtained by various hardware sensors and generating real-time perception results, which can be divided into two stages, sensing and comprehension. Generally, intelligent perception means perception that employs a smart algorithm in the comprehension stage. However, we can also improve the intelligence level of the perception task in its sensing stage. This can be achieved in two ways. (a) Enhancing the sensor's ability to feel the environment. For example, Wan et al. [1] develop a bimodal artificial sensory neuron (BASE) based on ionic/electronic hybrid neuromorphic electronics to implement visual–haptic fusion, which helps to build a highly integrated perceptual system to access massive sensory data for improving current cyborg technologies and artificial intelligence. (b) Automatically adjusting the parameters of the sensing system during the sense process based on the current perception result. This links the sensing and comprehension stages into a closed-loop system, which can improve the system's ability to interact with uncertain environments. This concept was first proposed by Connolly in 1985 [2].

Visual perception is the perception task where vision sensors such as a visible light camera or radar are used to collect the images or point clouds of the objects. It generates a visual representation of the environment, and finally reaches the goal of the perception task. For example, the reconstruction task aims to create a full digital model of an object, and the recognition task is to identify the candidate class to which the object belongs. Since data obtained at one view are not always enough to receive the target, we need to adjust the sensors' postures to sense the environment at multiple views. Visual sensors installed on the robotic platforms explore the circumstances, following the movement of their backbones. Traditionally, the movement of the sensors is set before perception, so it is a passive way to obtain information. We focus on improving the self-adjustment ability of the perception system, as (b) in the previous paragraph. View planning (VP) is employed for the multi-view tasks. It automatically adjusts the postures of sensors, to improve the efficiency of perception based on the prior information and the observed data [3]. We call the perception task with VP active perception.

For an active perception task, the next best view (NBV) is determined according to several factors of the VP system, including the goal of the perception task, the capabilities of the sensor system, the perceived objects, the optimization goal of view planning and the status of the perception result. The existing literature is designed for a particular task. For example, Chen et al. [4] proposed a VP algorithm for 3D reconstruction. They used the ductility of the smooth surface of the object to predict the curve of an unknown area, and then calculated the NBV by the predicted curve. Wu et al. [5] provided 3D ShapeNets to generate the NBV for recognition. In these studies, the factors are fixed in the VP algorithm, which prevents the algorithm from being extended to other perception tasks.

With the existing algorithms, when we deal with a new perception task, the VP system needs to be redesigned, and there is a big price to pay for doing this. To deal with this problem, we are committed to building a framework of a VP system that can handle different active visual perception tasks. The structure of the generic VP system is shown in Figure 1. When faced with a specific perception task, it only needs to formally describe the task as input to the view planning module according to the system requirements. Part of the status of the system will be set before a perception task, and others will be autonomously updated with the received data during the iterate perception. The NBV is planned by the view planning module based on the formal description of the task. The proposed system can improve the efficiency and reduce the energy consumption of robots during the visual perception process. Our contributions are listed as follows:

1. A generic view planning system is proposed. It is suitable for multiple visual tasks with a single object in the scene.
2. We define a series of statuses that can express the key elements of different perception tasks, and describe these statuses in a formal way.
3. Statuses are initialized before a given perception task, and they are updated to represent the real-time state of the VP system. The NBV is chosen based on the current state of the system.

Experiments on reconstruction tasks and recognition tasks were executed, which proved the effectiveness of our algorithm for different tasks. The rest of this paper is organized as follows: the second section reviews the literature. The third section introduces our generic VP algorithm. It explains the formal description of tasks, the update method of each status and the NBV calculation method. Experiments are represented in the fourth section with the discussion. Conclusions are given in the last section.

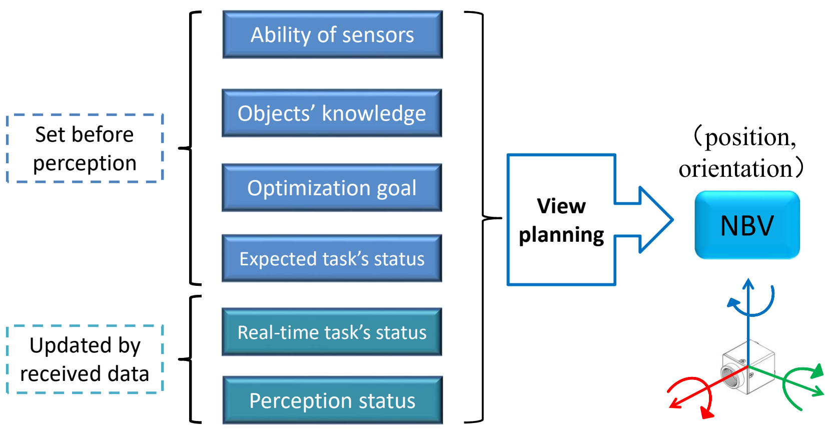


Figure 1. Structure of the generic view planning system. Ability of sensors, objects' prior knowledge, optimization goal of view planning and the expected status of a perception task are set before the perception process. With data receiving at the calculated views, real-time task status and the status of perception result are updated. View planning is based on these formal descriptions and the NBV is determined for the next iteration of data collection.

2. Related Work

To our knowledge, there is no framework that is specifically designed for view planning in different kinds of visual perception tasks. Active reconstruction and active recognition are the two scenarios that have received the most attention with the view planning process. Kriegel et al. [6] proposed a VP method for active scene exploration. It improves the speed of reconstruction and recognition. However, this VP method actually acts on active reconstruction. When it is applied independently to 3D recognition tasks, the improvement in the efficiency could not fulfil expectations.

The optimization goal of VP is to cover more surface with fewer views in reconstruction tasks. As the global optimization is an NP-hard problem, the greedy algorithm is usually employed instead of the global optimization. The view that can minimize the spatial uncertainty is selected as the NBV. When the objects' models are known before reconstruction, the NBV is calculated based on the known surface information. VP methods that deal with the above reconstruction tasks are called model-based methods [7,8]. Scott W R. [9] used the greedy method to obtain the NBV, which can be used to explore more of the unknown surface of the object. Chen et al. [10] proposed a view map generation method based on a genetic algorithm and the min–max criterion, which achieves the goal of covering the target surface with a smaller view set, and they traversed the view set with the shortest path by Christopher's algorithm. Kaba et al. [11] solved the model-based reconstruction problem by a non-greedy algorithm. The parameters in the evaluation function are modified at different perception statuses, and they are learned by a reinforcement learning method.

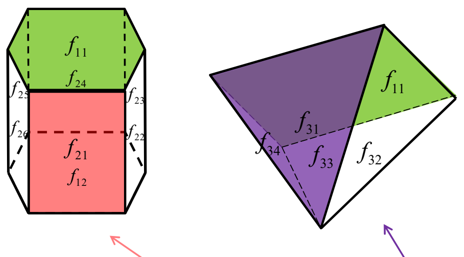
Since the information is complete enough for model-based methods, which results in a small research space, VP methods used for non-model reconstruction have received more attention. Surface-based methods analyze the surface trend to make sure that the entropy of the unknown model space will decrease the most after the detection at the next view. Chen et al. [4] utilized a point cloud to represent the model of the object. The boundary points are ranked by an evaluation function and the search direction of the top-ranked point is calculated to find the NBV. However, since only the smooth surface has a continuation trend, the surface-based methods are only suitable for the objects without rapid changes in curvature. Search-based methods make use of ray casting to calculate the information gain of each view [12–15]. They evaluate each candidate view by accumulating the gain of voxels that were passed through by rays of the camera. In these papers, voxel gain is defined in different ways. In general, the authors assign higher gains to unknown voxels that are close to the measured surface, because they consider that there is a higher probability of the presence of the surface at these voxels' locations, making it easier to reduce the uncertainty of the unknown space. This assumption is based on general knowledge of all objects, but for some objects, they have special prior knowledge. The rough shape of an object can be predicted by the use of its partial prior information, which provides more reliability for NBV determination. To make full use of the prior information, semi-model methods are proposed by Wei et al. [16] and Kong et al. [17]. They describe the partial prior information of the object, and plan the NBV by invoking the formal prior information.

For active recognition tasks, the VP method is used to find the view that can cover the features that minimize the ambiguity of the object. As concluded in some reviews for the active recognition methods [18,19], the information gain-based, aspect graph-based and the machine learning methods are the mainstream methods to deal with the VP issues in recognition tasks. The information gain-based methods generate a set of hypotheses based on the current data, and choose the view that can maximally disambiguate the initial set of hypotheses. For example, Potthast et al. [20] choose the candidate view that may receive more features as the NBV. The relationship between candidate views and their reachable features must be known for this kind of method. The aspect graph-based methods build an aspect graph of the candidate geometries, and they use the reasoning method to uniquely determine the object's type. Roy et al. [21] established an aspect graph as the knowledge system and inferred the NBV with a search tree. Dickinson et al. [22] proposed an aspect hierarchy to express the candidate geometries, and they chose the NBV based on the Bayesian network. In the situation where the scale of aspects or features is too large to be built as a search tree, learning methods are proposed to calculate the NBV. Wu et al. [5] represent a convolutional deep belief network to represent geometric 3D shapes as probability distributions of binary variables on a 3D voxel grid. When the recovery result is uncertain, they generate the observable images of the candidate views, and the distinctive view is chosen as the NBV. Johns et al. [23] decompose an image sequence into a set of image pairs, and select the view that achieves the maximum classification accuracy associated with the current view as the NBV. Sun et al. [24] employed a reinforcement learning method to obtain a globally optimal view.

As we focus on regular objects, whose prior information can be expressed formally, the deep learning methods are too expensive for us on large datasets. For the purpose of building a generic VP system for different kinds of tasks, the formal description method in our approach draws inspiration from the semi-model methods for reconstruction [17], which describe the structures of objects. It also draws on the active recognition method based on an aspect graph [21], which describes the status of features used to distinguish different categories of objects.

3. Method

The prior information of a task contains the ability of the sensor, which describes the sensor's capabilities to sense the environment. It also contains some deterministic information about the object, which can be used to limit the category and shape of the incomplete object. We describe the prior information of objects from the feature level. Each feature contains several local surfaces with the same kinds of characteristics. The presence of features is used to distinguish candidate classes, and the attributes of features determine the rough shape of an instance. As is shown in Figure 2, f_2 only exists in the prism and f_3 only exists in the pyramid. When one of them is detected, we can determine whether the object is a prism or pyramid. The attributes of f_1 are vertices arranged counterclockwise. The attributes of f_2 are four vertices of a rectangle, and the edge of the rectangle begins with a bottom edge. The attributes of f_3 are three vertices of a triangle, and the first vertex is the apex of the pyramid. If the object's category has been determined as a pyramid, vertices of f_1 are obtained, a local surface of f_3 is detected, and then the rough shape of the object can be predicted.



Features / Local surfaces / Class	f_1(*bottom*)	f_2(*rectangular side*)	f_3(*triangle side*)
prism	f_{11}, f_{12}	$f_{21}, f_{22}, f_{23}, f_{24}, f_{25}, f_{26}$	X
pyramid	f_{11}	X	$f_{31}, f_{32}, f_{33}, f_{34}$

Figure 2. Features of different classes of objects. The prism has features f_1 and f_2, while the pyramid has features f_1 and f_3. The rough shape of an instance can be defined by the parameters of its features.

The task's completion status is defined, which reflects the progress of an ongoing perception task. The expected completion status is given for a specific task, and the real-time completion status is updated during the perception. When the real-time completion status reaches the expected one, it means a termination of the VP procedure. With the acquisition of data in the iterative perception process, the state of the perceived space will change, which describes the data acquired during the entire perception process and is used to calculate the NBV. We describe the perception result by the visual representation status in two aspects. One is from the perspective of perceived space, which includes the voxel status, feature status and the candidate class status ("candidate status" for short). They describe the perceived space from concrete to abstract. The other one is from the perspective of candidate views, which includes the view status. It describes the information that can be observed from each candidate view. As a summary, the formal description of a visual task includes the formal expression of the task's completion status, the prior information of the perceived object and the visual representation status.

Figure 3 illustrates the update flow chart of each state. The relevant descriptions need to be initialized before a given perception task. During the iterative measurements, the voxel status and feature status are renewed when fresh data are received. The candidate status will be changed when some classes are excluded from the candidate library according to the feature status, and the feature status is affected by the candidate status in turn. Unknown features are predicted based on the prior information library. The voxel status is renewed when the voxel is at the location of the predicted features. If a feature is detected to be nonexistent based on the voxel status, the feature status and the candidate status will be updated again. The view status is updated with the voxel status, based on the given optimal function. The completion status is calculated with the view status and the candidate status. The algorithm is terminated when the prospective task status is reached. Otherwise, the NBV will be determined on the basis of the view status, and the robot will move to the NBV for the next sense. Details of the formal description and status update of perception tasks are introduced below.

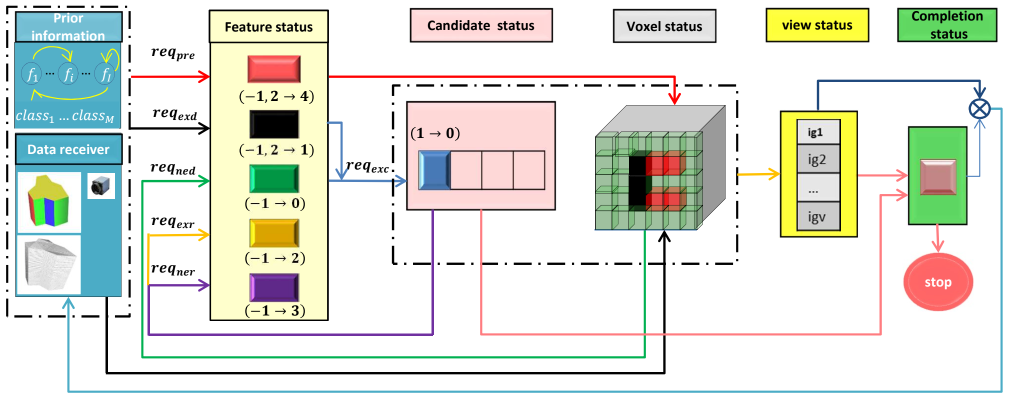


Figure 3. The update flow chart of each state. Boxes are basic description modules of our system and arrows represent the data streams when a new measurement is received.

3.1. Formal Description of Tasks

The task's completion status s_{cmp} is described in two aspects, the class confidence status and the surface integrity status.

$$s_{cmp} : (s_{cfd}, s_{ail})$$

where s_{cfd} is the class confidence state, which is utilized to indicate if the object belongs to a certain class with an ideal probability. s_{ail} is a state that shows whether the available surface information about the model is left.

The prior information library contains knowledge about the sensor and the object of a task. For the description of sensors' ability, we simply need to take the parameters of the camera into the prior information library, such as the field of view, depth of field, etc. For the prior information of the object, it includes the features' existence in each candidate class, description of these features and the positional relationship between features. The first part is to determine which class the object belongs to; the second part is to detect the latent features, and the last part is to infer the unknown surface of an object.

For a perception task, the candidate library is built by all geometry classes that the object may belong to. The library S_{class} is described as follows:

$$S_{class} : (class_1, \dots class_m, \dots class_M)$$

There are M classes in the library. We add all the features, which can distinguish the candidate classes and define the rough shape of an object, into the feature set. The feature set is noted as S_f, which has I members. Each candidate class $class_m$ is defined by the presence of features in S_f.

$$class_m : (ext_1^m \dots ext_i^m \dots ext_I^m)$$

ext_i^m indicates the presence of the feature f_i in $class_m$. If the feature exists in this class, ext_i^m is assigned a value of 1; otherwise, it is set to 0.

We simplify the local surfaces of features into polygons to express their rough shapes. Prior information about features promotes the acquisition of an object's surfaces. According to the degree of the information's certainty, we divide the information into three levels, including definite information, strong information and weak information. The definite information is actually the detection method of a feature. Strong and weak information are prediction methods of unknown features by the positional relationships between features. The positional relationship of two local surfaces can be defined by their common edges and the angle between them. For two features, if only the positional relationship between them is known, the relationship is called the weak information, while if the locations between the common edge and other edges in surfaces are known as well, the information is called the strong prior information.

In order to facilitate the detection and inference of features, we unify the description of all kinds of prior information into the same expression. Inspired by [17], we express the prior information by rules. There are K rules in the rule set, each of which is represented as:

$$rule_k : \{S_k^{ant} \rightarrow f_k^{cons}, proof_k(), cal_k(), pred_k()\}$$

$rule_k$ is the information that helps us to obtain the surface points of feature f_k^{cons}. If it describes definite information, we set S_k^{ant} as an empty one, and $proof_k()$ stores the detection function to check the existence of the feature f_k^{cons}. Its input is the obtained RGBD images, and the output is a Boolean value that indicates whether the feature is detected. f_k^{cons}'s attributes are calculated by $cal_k()$ and stored in the attributes set S_a. $pred_k()$ is a void function that has no operation.

When we describe the strong or weak information about position relationships between the consequence feature f_k^{cons} and several antecedent features, S_k^{ant} contains the antecedent features, $f_{k1}^{ant} \sim f_{kR}^{ant}$. The rule can only be used when the consequence is unknown and all antecedent features in S_k^{ant} have been detected. $proof_k()$ stores the detection function to check if the feature that has the given relationship with the antecedent features exists. $cal_k()$ has the same function as that in rules with definite information. $pred_k()$ stores the function that generates the predicted points of f_k^{cons}.

We give an example in Figure 4. In this situation, f_1 and f_3 are detected, and their vertices and edges are calculated by the definite rules about them:

$$\{\phi \rightarrow f_1, det\_f_1(), cal\_f_1(), pred\_void()\}$$

$$\{\phi \rightarrow f_1, det\_f_3(), cal\_f_3(), pred\_void()\}$$

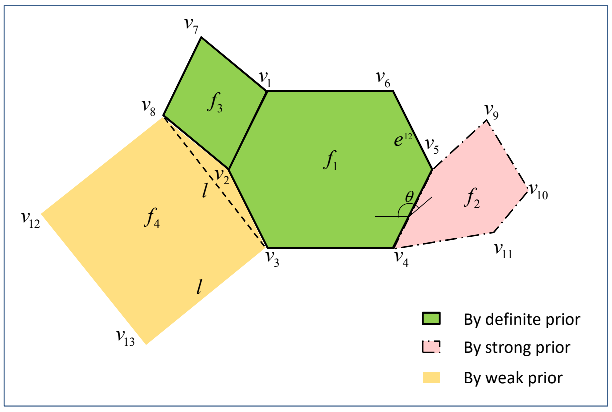


Figure 4. Positional relationship of local features. The green surfaces are detected by definite prior information, the pink one is predicted by strong prior information, and the yellow is predicted by weak prior information.

The common edge of f_1 and f_2 is known as $E(v_4v_5)$, the angle between them is θ, and the relationships of the common edge and other edges in f_2 are known before. Then vertices of f_2 can be calculated, and surface points of f_2 are generated by uniform sampling of the polygon in $pred_k()$ of strong rules:

$$\{f_1 \rightarrow f_2, det\_f_1\_f_2(), cal\_f_2(), pred\_f_1\_f_2()\}$$

The common edge of f_1 and f_4 is known as $E(v_2v_3)$, and the common edge between f_3 and f_4 is known as $E(v_2v_8)$. Then, unknown edges of f_4 are predicted to make up a square

$S(v_3v_8v_{12}v_{13})$ together with the line $\overline{v_3v_8}$, which connects the start point and the end point of the detected common edges. This prior information is stored in the weak rule:

$$\{f_1, f_3 \to f_4, det\_f_1f_3\_f_4(), cal\_f_4(), pred\_f_1f_3\_f_4()\}$$

The visual representation status includes the voxel status, the feature status, the candidate status and the view status.

We use octomap [25] to represent the model space. States of voxel nodes in the model space are divided into four types, as Figure 5 shows. The state of voxel v_n refers to ps_n, which is initialized as "unknown". With further observation, it turns to "occupied" or "free". f_{ij} is the jth local area of the feature f_i. For the unknown voxels where predicted areas are located, we give them a label of "preOccupied", and the serial numbers of their corresponding features are tagged on them. $-1, 1, 0, 2$ are used to represent the "unknown", "occupied", "free" and "preOccupied" states, respectively.

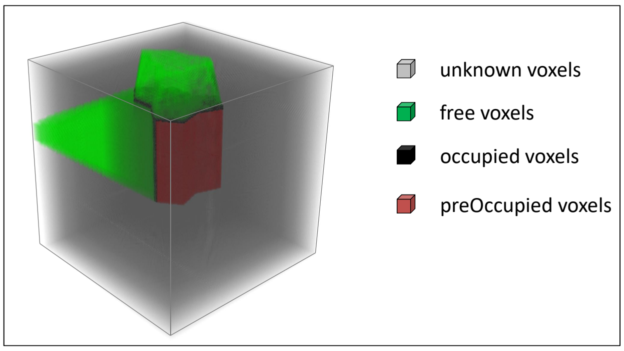


Figure 5. Voxels' status in the model space.

Based on the observed data, a feature's existence state in the perceived object can be calculated by the stored prior information. The existence state of f_i is called $es(f_i)$, es_i for short. All features' states form an array S_{es}, which is called the feature status. Members of S_{es} are set to -1 before perception, and es_i will turn to 0, 1, 2 or 3 with the acquisition of perceived data. Moreover, 2 is an interim state of a feature, which means that the feature exists but still needs to be observed to confirm its attributes. Meanwhile, 0, 1 and 3 are the final states of a feature. If the feature is predicted, its will be added with a prediction state, which is noted as 4. The attribute set S_A contains the attributes of all sub-features.

- $-$**1** unknown state, where we cannot determine whether the feature is existent or not;
- **0** nonexistent by detection, which means that the potential location of the feature has been detected, but this feature is still not found;
- **1** existent by detection, which means that the feature has already been detected from the obtained data;
- **2** existent by reasoning, which is the signal that the feature is existent by reasoning the candidate library;
- **3** nonexistent by reasoning, which is a state that the feature is nonexistent by reasoning the candidate library;
- **4** predicted, which means that the potential location of the feature is predicted.

Candidate status S_{cand} is a vector that shows whether each candidate class in S_{class} is still a possible category of the object.

$$S_{cand} : (cs_1, \dots cs_m, \dots cs_M)$$

where the element cs_m is initialized to true before the perception task. When the feature status is calculated iteratively, $class_m$ may be excluded from the candidate classes, and its corresponding status cs_m changes to false.

Candidate view set S_v is generated uniformly around the object space before perception tasks. For each view v_c in S_v, its state vs_c reflects the information gain that can be received when the sensor moves to it as the next view. The state is calculated by the optimization function that is defined manually.

3.2. Status Update

The feature status is updated after each measurement. The transformation between existence states of a feature is shown in Figure 6. The received data are put into the detection functions in the prior information library to analyze the feature's existence. es_i is set to 1 when req_{exd} is met, which means that the feature has been detected. Then, local areas of f_i are extracted and their attributes are calculated and stored in the attribute set.

$$req_{exd}: \ \exists k, k \in [1, K], (f_k^{cons} = f_i) \& (proof_k() = 1) \tag{1}$$

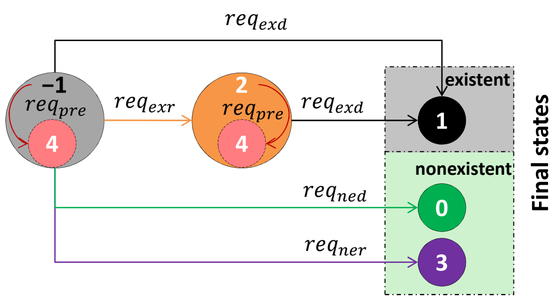


Figure 6. Transformation between the existence states of a feature.

If the acquired information is fully used, and f_i is still in non-final state, then es_i is attached to a prediction state 4 when req_{pre} is met, which means that the feature's location can be inferred by its related features.

$$\begin{aligned} req_{pre}: \, !req_{exd} \& (\exists k, k \in [1, K], f_k^{cons} = f_i) \\ \& (\forall f_{kj}^{ant}, f_{kj}^{ant} \in S_k^{ant}, es(f_{kj}^{ant}) = 1) \end{aligned} \tag{2}$$

es_i is set to 0 when the detected proportion of f_i's predicted voxels has exceeded the threshold and the feature still has not been detected. The requirement req_{ned} is inferred from the voxel status.

$$req_{ned}: \ \frac{1}{|PV_i|} \sum_{pv_i \in PV_i} U(\xi - \min_{p \in PC_t} ||p - pv_i||_2) \geq \xi_{ned} \tag{3}$$

where U is the Heaviside step function, PV_i is the predicted voxels of f_i, PC_t is the point cloud observed until time t, ξ is a distance threshold and ξ_{ned} is a percentage threshold.

The candidate library is employed to verify if es_i should be renewed from -1 to 2. The conversion condition req_{exr} is satisfied if f_i exists in every class.

$$req_{exr}: \ \forall m, m \in [1, M], ext_i^m = 1 \tag{4}$$

When f_i is nonexistent in every candidate class, es_i will be renewed from -1 to 3. The condition req_{ner} is described as:

$$req_{ner}: \ \forall m, m \in [1, M], ext_i^m = 0 \tag{5}$$

The voxel status is updated when a new observed point cloud is transformed and inserted into the model space. If an unknown voxel is located on the detected surface, its state converts to "occupied". The voxel's state will convert to "free" when it is passed through by rays. ps_n is set to "preOccupied" when v_n is on the location of predicted local surfaces.

$$ps_n = \begin{cases} 2, & \sum_{i=1}^{I} U(\xi - \min_{pv_i \in PV_i} ||v_n - pv_i||_2) > 0 \\ 0, & otherwise \end{cases} \tag{6}$$

After the feature f_i obtains its final state through detection, the candidate status will be updated with the feature status. cs_m will turn to false when $class_m$ conflicts with the feature status S_{es}^t at time t. The exclusion requirement req_{exc} is:

$$\begin{aligned} req_{exc} : & P(class_m | S_{es}^t) \\ & = \frac{P(S_{es}^t | class_m) P(class_m)}{\sum_{j=1}^{M} P(S_{es}^t | class_j) P(class_j)} = 0 \end{aligned} \tag{7}$$

$P(class_j)$ is the prior probability of $class_j$. At time t, if there are some features whose existence states are final states and are the opposite of the relevant presence states in $class_j$, then the probability $P(S_{es}^t | class_j)$ is 0; otherwise, it is 1.

The view status is updated when the voxel status update has been done. When views have been reached, they have no contribution to the perception task any more, so their states are set to 0. Other views' states are calculated by the evaluation function:

$$vs_c = G_{ovlp}(v_c) \cdot (G_{fd}(v_c) + W \cdot G_{so}(v_c)) \tag{8}$$

The evaluation function takes three factors into account, consisting of the overlap constraint, the feature detection and the unknown surface observation. The first component $G_{ovlp}(v_c)$ is to guarantee an overlap between the new and the existing point cloud for registration [13]. The feature detection factor $G_{fd}(v_c)$ is the kernel of the function to detect the features with higher importance, for the reason that the view with a high value at this part can not only speed up the process of recognition, but also promote the acquisition of a complete model.

$$G_{fd}(v_c) = \max_{i \in [i,I], j \in [1,J]} w_i \cdot \frac{n^{ij}(v_c)}{n_t^{ij}} \tag{9}$$

where w_i is a parameter that reflects the contribution of f_i to identify the object. If the existence state of f_i is -1, its weight w_i is defined as the conditional entropy. If es_i is 2, which indicates that f_i is of no use to the identification but contributes to the object reconstruction, we give w_i a value smaller than the minimum weight of all features that have an existence state of -1. w_i is set to 0 otherwise. All predicted voxels of f_{ij} form a voxel set, n_t^{ij} is the number of voxels in the set, and $n^{ij}(v_c)$ is the number of reachable voxels at view v in the set.

$G_{so}(v_c)$ provides assistance to $G_{fd}(v_c)$, in the situation where no preoccupied voxel exists and the model is not complete. It is assigned by counting up all reachable unknown voxels behind the surface of the model. W is set to 1 when the result of $G_{fd}(v_c)$ is 0 for all candidate views.

s_{cfd} is calculated and it will be set when the maximum confidence of all species exceeds the set threshold; otherwise, it is false. ξ_{cfd} is the confidence threshold.

$$s_{cfd} = \begin{cases} 1, & \max_{m=1}^{M} P(class_m | S_{es}^t) \geq \xi_{cfd} \\ 0, & otherwise \end{cases} \tag{10}$$

The state s_{ail} will turn to 1 from the default 0 when all candidate views could not receive available information from the environment.

$$s_{ail} = \begin{cases} 1, & \forall v, G(v) = 0 \\ 0, & otherwise \end{cases} \tag{11}$$

When s_{cmp} has obtained its expected values, the perception task will come to an end. Otherwise, the view whose state has the maximum value is selected as the NBV to obtain the next measurement.

$$NBV = \arg\max_{v_c \in S_v} vs_c \tag{12}$$

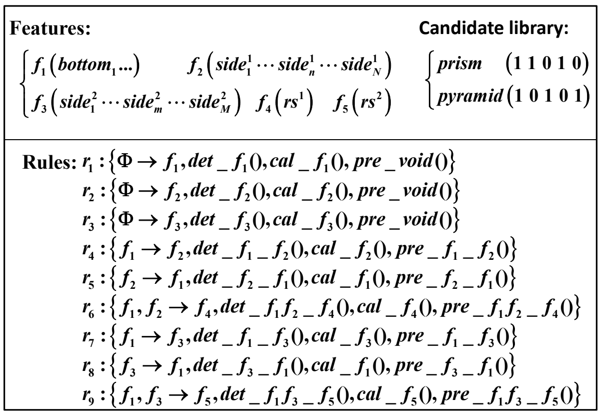
4. Experiments

To prove the efficiency of our methods for different kinds of perception tasks, experiments for reconstruction tasks and recognition tasks are performed in a simulated environment. The robotic operating system [26] is used for data transmission and virtual camera controlling. Deformed prisms and pyramids are employed as objects to be reconstructed, and 12 tanks that can be identified by 7 features are used for recognition tasks [27]. In order to focus on the VP process, we choose noticeable colors and textures as the information to detect features. The positions of the candidate views are evenly distributed on the sphere surrounding the model space, and the camera's optical axis always points to the center of the sphere. For each experiment, we select one of the candidate views as the initial view randomly, and execute the VP method until the termination is reached. The maximum iteration is set to 20. As we expect to complete a perception task with fewer views, the number of views requested for complete visual tasks is an indicator to evaluate the efficiency of the VP algorithm. Therefore, we record the number of views required to complete each task and compare it with that for other methods to verify the effectiveness of our algorithm.

4.1. Experiment for Reconstruction

Experiments on reconstruction tasks were executed with the simulated objects. The candidate library and the prior information about features are generated as in Figure 7. There are 5 features in the feature set, $f_1 \sim f_5$, including the bottom of the object, the side of the prism, the side of the pyramid, the rough shape of the prism and the rough shape of the pyramid. The vector representing the existence states of features for the prism is (1, 1, 0, 1, 0), while it is (1, 0, 1, 0, 1) for the pyramid. $bottom_i$ is the sub-feature of f_1 defined by the vertices circling the bottom counterclockwise and the normal of the bottom. $side_n^1$ is one side of the prism, which is a sub-feature of f_2. Its attributes are four vertices circling it clockwise and the normal of the side. $side_n^2$ is a sub-feature of f_3. It is a side of the pyramid. f_3 has three vertices, and it is the only difference between attributes of f_2 and f_3. rs^1 is the rough shape of the prism, defined by the attributes of the bottom and the height of the prism. rs^2 is the rough shape of the pyramid, which has the same parameters as rs^1, except for the substitution of the height to the apex of the pyramid. $r_1 \sim r_3$ stockpile descriptions of $f_1 \sim f_3$ and $r_4 \sim r_9$ store relationships between $f_1 \sim f_5$.

(0, 1) is the target for completion status. ξ is set to 0.01; ξ_{ned} and ξ_{cfd} are set to 0.99. Eight other methods were used as control groups, including seven methods compared by Delmerico et al. [28] and the two-step active reconstruction method (T-S) proposed by Kong et al. [17]. The reference methods do not take the overlap constraint into consideration, which will cause a decrease in the required views, so we executed the experiments with (GNO) and without (GN) the overlap constraint separately to evaluate the efficiency of our algorithm on reconstruction tasks.



Features:

$$\begin{cases} f_1(bottom_1 \ldots) \quad f_2(side_1^1 \cdots side_n^1 \cdots side_N^1) \\ f_3(side_1^2 \cdots side_m^2 \cdots side_M^2) \quad f_4(rs^1) \quad f_5(rs^2) \end{cases}$$

Candidate library:

$$\begin{cases} prism \ (1\ 1\ 0\ 1\ 0) \\ pyramid\ (1\ 0\ 1\ 0\ 1) \end{cases}$$

Rules:

$r_1 : \{\Phi \to f_1, det\_f_1(), cal\_f_1(), pre\_void()\}$

$r_2 : \{\Phi \to f_2, det\_f_2(), cal\_f_2(), pre\_void()\}$

$r_3 : \{\Phi \to f_3, det\_f_3(), cal\_f_3(), pre\_void()\}$

$r_4 : \{f_1 \to f_2, det\_f_1\_f_2(), cal\_f_2(), pre\_f_1\_f_2()\}$

$r_5 : \{f_2 \to f_1, det\_f_2\_f_1(), cal\_f_1(), pre\_f_2\_f_1()\}$

$r_6 : \{f_1, f_2 \to f_4, det\_f_1 f_2\_f_4(), cal\_f_4(), pre\_f_1 f_2\_f_4()\}$

$r_7 : \{f_1 \to f_3, det\_f_1\_f_3(), cal\_f_3(), pre\_f_1\_f_3()\}$

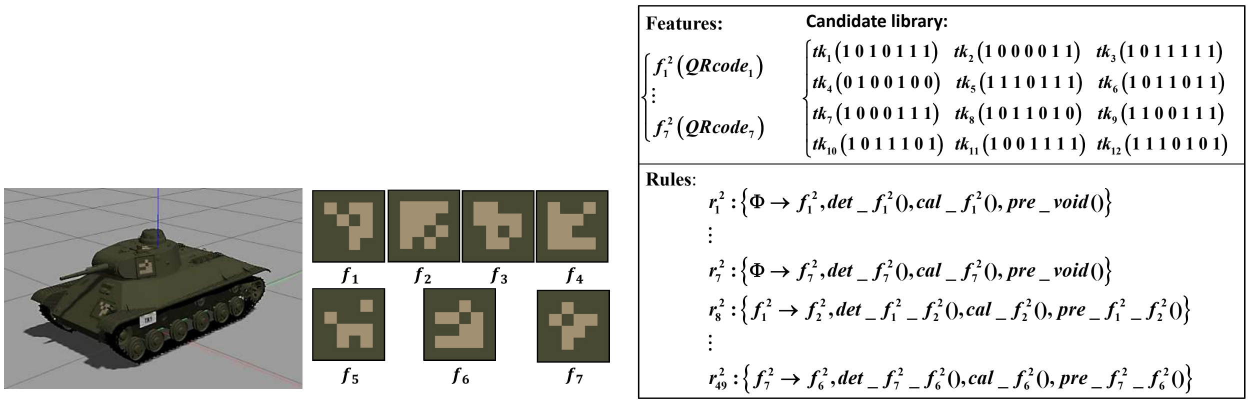
$r_8 : \{f_3 \to f_1, det\_f_3\_f_1(), cal\_f_1(), pre\_f_3\_f_1()\}$

$r_9 : \{f_1, f_3 \to f_5, det\_f_1 f_3\_f_5(), cal\_f_5(), pre\_f_1 f_3\_f_5()\}$

Figure 7. Prior knowledge library of the objects for reconstruction.

4.2. Experiment for Recognition

We conducted experiments to plan the NBV in tank recognition. The candidate library is shown in Figure 8. There are 7 features in the feature set. Each feature has only one sub-feature and they are distributed on the surface of the tank. The sub-feature $QRcode_i$ is located in a box. Rules $r_1^2 \sim r_7^2$ store description methods of each feature, while $r_8^2 \sim r_{49}^2$ are relationships between features.



Features:

$$\begin{cases} f_1^2(QRcode_1) \\ \vdots \\ f_7^2(QRcode_7) \end{cases}$$

Candidate library:

$$\begin{cases} tk_1(1\,0\,1\,0\,1\,1\,1) \quad tk_2(1\,0\,0\,0\,0\,1\,1) \quad tk_3(1\,0\,1\,1\,1\,1\,1) \\ tk_4(0\,1\,0\,0\,1\,0\,0) \quad tk_5(1\,1\,1\,0\,1\,1\,1) \quad tk_6(1\,0\,1\,1\,0\,1\,1) \\ tk_7(1\,0\,0\,0\,1\,1\,1) \quad tk_8(1\,0\,1\,1\,0\,1\,0) \quad tk_9(1\,1\,0\,0\,1\,1\,1) \\ tk_{10}(1\,0\,1\,1\,1\,0\,1) \quad tk_{11}(1\,0\,0\,1\,1\,1\,1) \quad tk_{12}(1\,1\,1\,0\,1\,0\,1) \end{cases}$$

Rules:

$r_1^2 : \{\Phi \to f_1^2, det\_f_1^2(), cal\_f_1^2(), pre\_void()\}$

$\vdots$

$r_7^2 : \{\Phi \to f_7^2, det\_f_7^2(), cal\_f_7^2(), pre\_void()\}$

$r_8^2 : \{f_1^2 \to f_2^2, det\_f_1^2\_f_2^2(), cal\_f_2^2(), pre\_f_1^2\_f_2^2()\}$

$\vdots$

$r_{49}^2 : \{f_7^2 \to f_6^2, det\_f_7^2\_f_6^2(), cal\_f_6^2(), pre\_f_7^2\_f_6^2()\}$

Figure 8. Prior knowledge library of the tanks for recognition.

(1,0) is the target for completion status. ξ is set to 0.03, ξ_{ned} is set to 0.95, ξ_{cfd} is set to 0.99. For each model, we tested our method against the random method and the Prior Feature Distribution Table method (PFDT). The NBV is stochastically specified among the candidate views by the random method. In the PFDT method, the features correspond to the candidate views one-to-one, and the NBV is determined according to the relationship between features.

4.3. Results and Discussion

The number of views required for each active reconstruction is recorded and shown in Table 1. We find that our algorithm can complete each reconstruction with a relatively small and steady number of views. According to Figure 9, which represents the average views among all models for each method, it is shown that the GN, GNO method and the T-S method have distinct advantages over other algorithms. Among them, the proposed GN method uses the lowest average number of views. When the overlap constraint is taken into consideration, the efficiency of VP is slightly affected. This result is reflected in the figure, where the average number of views needed by the GNO method is slightly higher than that by the GN method, and levels with that by the T-S method. This is caused by the overlap constraint that limits the size of unknown surface obtained from a new view.

However, the overlap constraint is necessary in the real world, for the reason that the robot movement will have errors that need to be eliminated by point cloud registration.

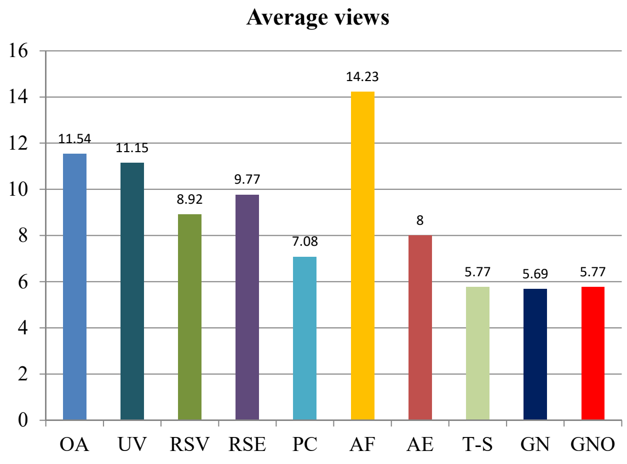
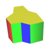


Figure 9. Average number of views needed by each method in reconstruction experiments.

Table 1. Comparison of each model for reconstruction. T-S, GN and GNO achieve better performance.

Models	**m1**	**m2**	**m3**	**m4**	**m5**	**m6**	**m7**	**m8**	**m9** 	**m10**	**m11**	**m12**	**m13**
OA	8	20	20	8	8	8	5	9	9	20	10	8	17
UV	8	20	13	8	8	8	5	9	9	20	14	8	15
RSV	6	17	4	5	9	6	4	16	8	**5**	11	13	12
RSE	6	16	6	**4**	8	**4**	4	14	12	11	8	14	20
PC	5	9	5	5	**5**	5	3	6	**7**	8	5	16	13
AF	9	10	5	20	17	20	18	7	20	10	9	20	20
AE	**4**	8	7	6	6	8	5	6	8	10	7	14	15
T-S	5	**7**	4	5	**5**	**4**	5	6	**7**	**5**	**4**	8	10
GN	5	8	**3**	5	**5**	5	3	**5**	**7**	7	5	9	**7**
GNO	8	8	**3**	6	**5**	6	**2**	6	8	**5**	5	**6**	**7**

For the active recognition tasks, the number of views needed for each tank is shown in Figure 10. The random method always needs more views for the recognition of each object. It shows that the efficiency of recognition can be promoted by a reasonable view planning method. Our method requires fewer views than the other two methods in most cases, which proves that our method is suitable for recognition tasks as well. The average number of views required by our method is less than that by the PFDT method. The reason for this result may be that the PFDT method only focuses on the area of one feature at a view, which makes the probability of other features being detected very low, and the discriminative features can only be detected view-by-view.

Through these experimental results, we can find that the proposed method plans the NBV effectively in different perception tasks, where the perception goals and measured objects are different. Furthermore, the efficiency of the generic VP is relatively higher than other methods. The results show that our method can be applied to increase the intelligence of robots or unmanned platforms. When the robot equipped with our VP system is assigned the task of multi-view visual perception, it can automatically choose the next view with the greatest contribution to the overall task. These view sequences are smaller than that selected randomly or by other methods, so that the control system of the robot does not need to sense and process the data of the environment frequently, which reduces the consumption of movement and calculation. In addition, for some exploration tasks, the smaller the number of views that need to be detected, the less the disturbance to the target in the environment.

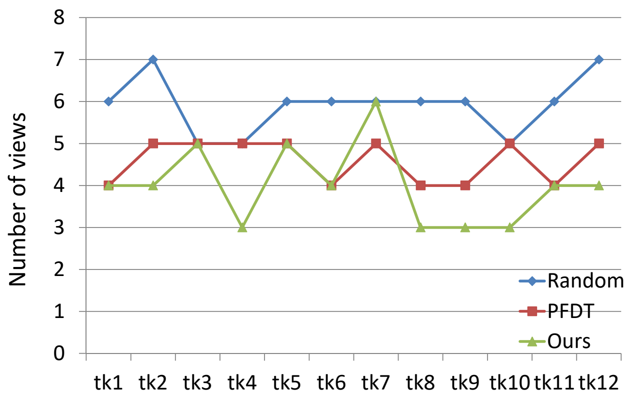


Figure 10. Number of views needed in recognition experiments.

5. Conclusions

We propose a generic view planning method that is suitable for multiple visual tasks. A formal description of tasks, including the prior information library, the visual representation status and the completion status, is raised to build the basis of VP for a specific task. A task begins with the initialization of its descriptions. The states are updated after each measurement. The NBV is chosen based on the current task status, until the task is ended with a signal that the completion status has received its target. Experiments for reconstruction and recognition were executed and the results show that our algorithm is effective for NBV planning in multi-tasks. However, our method will be limited when the feature's local surfaces are too complex to be expressed with parameters, for the reason that the information of a complex object's features is too difficult to be described manually. A deep neural net has the ability to autonomously learn the features of objects by adjusting its weights through backpropagating the error of samples in the large dataset. In the future, we will consider using deep learning methods to solve the prediction problem of complex surfaces. Moreover, as the Hebbian synapse-based reinforcement learning [29] can continuously adjust the weight between nodes in an unsupervised manner, it also provides us with a novel mean of generating the NBV end-to-end.

Author Contributions: Conceptualization, Y.K.; methodology, F.Z. and Y.K.; software, Y.K. and H.S.; validation, F.Z. and Y.K.; formal analysis, Y.K.; investigation, Y.K.; resources, F.Z.; data curation, Y.K.; writing—original draft preparation, Y.K.; writing—review and editing, Z.L. and Q.W.; visualization, Y.K.; supervision, Z.L. and Q.W.; project administration, F.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset of this paper can be found at https://github.com/Miss-Katherine/DAE-model-of-prisms, accessed on 17 April 2022.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wan, C.; Cai, P.; Guo, X.; Wang, M.; Matsuhisa, N.; Yang, L.; Lv, Z.; Luo, Y.; Loh, X.J.; Chen, X. An artificial sensory neuron with visual-haptic fusion. *Nat. Commun.* **2020**, *11*, 4602. [CrossRef] [PubMed]
2. Connolly, C. The determination of next best views. In Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, USA, 25–28 March 2003.
3. Scott, W.; Roth, G.; Rivest, J.F. View planning for automated three-dimensional object reconstruction and inspection. *ACM Comput. Surv.* **2003**, *35*, 64–96. [CrossRef]
4. Chen, S.Y.; Li, Y.F. Vision sensor planning for 3-D model acquisition. *IEEE Trans. Syst. Man. Cybern. B* **2005**, *35*, 894–904. [CrossRef] [PubMed]

5. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In Proceedings of the Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015.
6. Kriegel, S.; Brucker, M.; Marton, Z.C.; Bodenmuller, T.; Suppa, M. Combining Object Modeling and Recognition for Active Scene Exploration. In Proceedings of the International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013.
7. Mavrinac, A.; Chen, X.; Alarcon-Herrera, J.L. Semiautomatic model-based view planning for active triangulation 3-D inspection systems. *IEEE/ASME Trans. Mechatron.* **2015**, *20*, 799–811. [CrossRef]
8. Wakisaka, E.; Kanai, S.; Date, H. Model-based next-best-view planning of terrestrial laser scanner for HVAC facility renovation. *Comput.-Aided Des. Appl.* **2018**, *15*, 353–366. [CrossRef]
9. Scott, W.R. Model-based view planning. *Mach. Vis. Appl.* **2009**, *20*, 47–69. [CrossRef]
10. Chen, S.Y.; Li, Y.F. Automatic sensor placement for model-based robot vision. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 393–408. [CrossRef]
11. Devrim Kaba, M.; Gokhan Uzunbas, M.; Nam Lim, S. A Reinforcement Learning Approach to the View Planning Problem. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
12. Isler, S.; Sabzevari, R.; Delmerico, J.; Scaramuzza, D. An Information Gain Formulation for Active Volumetric 3D Reconstruction. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016.
13. Irving, V.; Enrique, S.L.; Rafael, M.C. Tree-based search of the next best view/state for three-dimensional object reconstruction. *Int. J. Adv. Robot. Syst.* **2018**, *15*, 1–11
14. Potthast, C.; Sukhatme, G.S. A probabilistic framework for next best view estimation in a cluttered environment. *J. Vis. Commun. Image Represent.* **2014**, *25*, 148–164. [CrossRef]
15. Daudelin, J.; Campbell, M. An adaptable, probabilistic, next-best view algorithm for reconstruction of unknown 3-D objects. *IEEE Robot Autom. Lett.* **2017**, *2*, 1540–1547. [CrossRef]
16. Wei, J.; Polden, J.; Tao, P.; Lin, W.; Shimada, K. View planning for 3D shape reconstruction of buildings with unmanned aerial vehicles. In Proceedings of the International Conference on Control, Automation, Robotics and Vision (ICARCV), Phuket, Thailand, 13–15 November 2016; pp. 1–6.
17. Kong, Y.; Zhu, F.; Hao, Y.; Sun, H.; Xie, Y.; Lin, Z. An active reconstruction algorithm based on partial prior information. *Int. J. Adv. Robot. Syst.* **2020**, *17*, 1–8. [CrossRef]
18. Chen, S.Y.; Li, Y.F.; Kwok, N.M. Active vision in robotic systems: A survey of recent developments. *Int. J. Robot Res.* **2011**, *30*, 1343–1377. [CrossRef]
19. Zeng, R.; Wen, Y.; Zhao, W.; Liu, Y.J. View planning in robot active vision: A survey of systems, algorithms, and applications. *Comput. Vis. Media* **2020**, *6*, 225–245. [CrossRef]
20. Potthast, C.; Breitenmoser, A.; Fei, S.; Sukhatme, G.S. Active multi-view object recognition: A unifying view on online feature selection and view planning. *Robot. Auton. Syst.* **2016**, *84*, 31–47. [CrossRef]
21. Roy, S.D.; Chaudhury, S.; Banerjee, S. Isolated 3D object recognition through next view planning. *IEEE Trans. Syst. Man Cybern.-Part A Syst. Hum.* **2000**, *30*, 67–76. [CrossRef]
22. Dickinson, S.J.; Christensen, H.I.; Tsotsos, J.K.; Olofsson, G. Active object recognition integrating attention and viewpoint control. *Comput. Vis. Image Underst.* **1997**, *67*, 239–260. [CrossRef]
23. Johns, E.; Leutenegger, S.; Davison, A.J. Pairwise Decomposition of Image Sequences for Active Multi-View Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
24. Sun, H.B.; Zhu, F.; Kong, Y.Z.; Wang, J.; Zhao, P. Continuous viewpoint planning in conjunction with dynamic exploration for active object recognition. *Entropy* **2021**, *23*, 1702. [CrossRef]
25. Hornung, A.; Wurm, K.M.; Bennewitz, M.; Stachniss, C.; Burgard, W. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Auton. Robot.* **2013**, *34*, 189–206. [CrossRef]
26. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source robot operating system. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 17 May 2009; pp. 1–6.
27. Sun, H.; Zhu, F.; Hao, Y.; Fu, S.; Kong, Y.; Xu, C.; Wang, J. Unified Optimization for Multiple Active Object Recognition Tasks with Feature Decision Tree. *J. Intell. Robot. Syst.* **2021**, *103*, 31. [CrossRef]
28. Delmerico, J.; Isler, S.; Sabzevari, R.; Scaramuzza, D. A comparison of volumetric information gain metrics for active 3D object reconstruction. *Auton. Robot.* **2018**, *42*, 197–208. [CrossRef]
29. Dresp-Langley B. From Biological Synapses to "Intelligent" Robots. *Electronics* **2022**, *11*, 707. [CrossRef]