The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

View Planning for 3D Object Reconstruction

Juan Irving Vásquez-Gómez, Efraín López-Damian and Luis Enrique Sucar

***Abstract*— For manipulating an unknown object a robot needs a 3D model of it. Given the limited field of view of a camera and self occlusions, a set of views is required to build a complete 3D model, so an important problem is how to select these views optimally according to certain criteria. We propose a novel algorithm to select the next-best-view (NBV) for a range camera to model 3D arbitrary objects. We use a volumetric representation and voxel labeling. We propose a new utility function based on factors considering voxel, quality and navigation information. We also propose two novel strategies to make faster the search of the NBV, one based on a hierarchical decomposition of the search space and other based on a multi-resolution of ray tracing. We have tested our planner in simulation with 7 different 3D objects, showing good results in terms of quality of the models and computation time required, and at the same time reducing the distance that the sensor has to travel to obtain the set of views.**

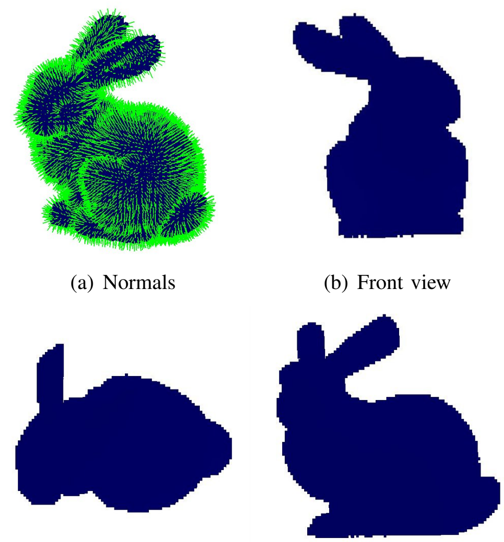
I. INTRODUCTION

Autonomous 3D object reconstruction has an important place in service robotics. It is desirable that a service robot could interact with the objects in the environment. The robot needs to acquire a 3D model of such objects in order to manipulate them. Vision offers a huge amount of information and several techniques to scan surfaces. However, due to the limited camera's field of view and object's self-occlusions, a set of scans from different view points is needed. A fast and easy solution can be a predetermined set of view points. However, due to the size and shape variations of the objects this simple strategy could lead to incomplete models or to many more views that those required. So an important problem is how to plan these view points.

There are some criteria that must be considered in an object reconstruction: overlap with previous scans (making easy the information register stage) and quality scans (we consider the best scan quality when the sensor orientation is perpendicular to the surface). Besides, the robot must navigate to each view point in order to capture the images, spending time and energy. Then, a set of views that minimize the distance traveled by the robot, that maximize the quality of the reconstructed model and that provides overlap with previous scans is desirable. How to obtain this set is the problem we address in our view planning method.

A view planning algorithm for object reconstruction is an incremental model construction method composed of a number of observing-and-planning loops [1]. Based on a partial model, this algorithm determines the next view that provides an optimal amount of information, or NBV, from a set of candidate views.

J. I. Vásquez-Gómez, E. López-Damian and L. E. Sucar are with the Department of Computer Science, INAOE, Tonantzintla, Puebla, Mexico {ivasquez,eldamian,esucar}@ccc.inaoep.mx



(a) Normals (b) Front view

(c) Top view (d) Left view

Fig. 1. Output model from our view planner for the bunny object. All the subfigures show the occupied voxels. Subfigure a) also shows the voxel normals.

In this paper we present a view planning method which can reconstruct arbitrary 3D objects with relative few views and an acceptable quality, it successfully reduces the distance that the robot has to travel, saving energy. The main contributions of our method are: (i) a novel utility function that allow us to evaluate how good a view can be based on area percentage, quality and navigation distance; and (ii) two search strategies that reduce the computation time required to determine the NBV. We have tested our method in simulation, with promising results in terms of quality of the models and computation time, and at the same time reducing the distance that the sensor has to travel to scan the object surface. Fig. 1 shows the reconstruction of a bunny object by our planner.

II. RELATED WORK

Since the 80's many view planning algorithms for object reconstruction have been developed, in [2] there is an extensive review of these works.

Many authors simplify the general problem by reducing the search space using discrete candidate views around an approximate sphere. This sphere is defined as a set of views which are at a certain distance from the center of the object, the sensor always is pointing to the center of the sphere [3], [4].

In order to evaluate how desirable is a view, several utility functions have been proposed. The simplest of those is to measure the amount of unknown information. Others, include overlap with previous scans and quality. In [5] a function that measure the overlap was proposed. The function was based on the percentages of voxels that are visible from a candidate view. In [4] a way to improve the quality of the model was proposed, but it does not take into account the overlap. In [6] they defined two different functions. The problem of both functions is that they do not consider the case when there is no voxels of some kind in the image, giving good evaluation for the view.

So far, the work done for minimizing the traveling distance needs to know a priori the set of views to reconstruct the object. For instance in [7], they make, in a first stage, a minimization of the amount of views and traveling distance. But the method needs a previous 2D map of the objective. Therefore, previous work do not consider all the mentioned criteria.

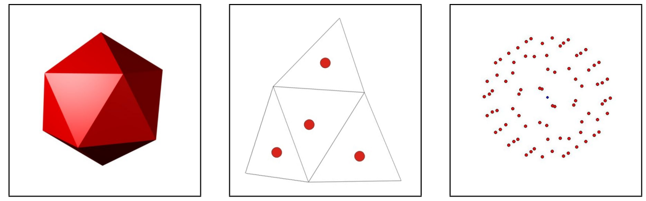
III. VIEW PLANNER OVERVIEW

In this section we present a general overview of our method. We use a voxel map to represent the scene and store the partial model of the object at every iteration. This voxel map is conformed by a three-dimensional matrix of labeled voxels. Each voxel stores information about the space that it represents by means of three attributes [5]: a label indicating its type, a surface normal and a quality value. Each label has a color associated with it for display purposes. The labels are defined as follows:

- Unmarked. A voxel that has not been observed yet by the range camera.
- Occupied. A voxel which position matches with acquired points in the 3D range image.
- Empty. A voxel in a seen area but for which there are no acquired points in this position.
- Occluded. A voxel in the sensor field of view but not seen because it is behind of an occupied voxel.
- Occplane. An occluded voxel that is adjacent with any of its six faces to an empty voxel.

Additionally, the normal of the surface and a quality value are attached just to occupied voxels. The normal of the surface is computed as the average of all others occupied voxel normals that are preset in the image adjacent to the voxel that has been touched by the sensor ray. The quality is determined according to the algorithm shown in [4].

Our algorithm starts by taking a first range image of the object from a known free-space sensor position in the voxel map. The sensor takes a range image. The points acquired from the range image are converted to the coordinate system of the voxel map generating a partial model of the object by labeling the voxels. Then, the search strategy uses the utility function to evaluate a set of views from a view sphere around the object. The function considers the amount of voxels in the view, their surface normals and the distance from the current sensor position to evaluate each view of the set. After such evaluation the NBV is selected. The next step consists on moving the range camera (robot) to this position and we proceed to capture a range image to obtain a more complete object model. The process is repeated until at least one of the termination criteria is achieved: there are no occplane voxels in the voxel map or the search strategy does not find a next view (the possible views does not provide new information). We summarize our iterative method in Algorithm 1.



(a) Icosahedron (b) View points (c) View sphere

Fig. 2. Icosahedron tesselation and view positions

Algorithm 1: View Planner Algorithm for 3D Object Reconstruction

Data: Initial Robot Position
Result: Object model M
$Im \leftarrow$ Take range image;
Update model M with Im;
while *true* **do**
  $nbv \leftarrow$ Compute the next best view;
  if nbv *does not provide new information* **then**
    Return M and exit;
  end
  Pose robot in nbv;
  $Im \leftarrow$ Take range image;
  Update model M with Im;
end

The search space consists of a set of views around the object (view sphere). The views are created by sphere tessellation; using an icosahedron (20 views) as the lowest sphere resolution (Fig. 2(a)). To increase the sphere resolution, each face of the icosahedron is divided in four sub-faces. The search strategy uses the centroid of each face as a candidate view (Fig. 2(b)). Fig. 2(c) shows a sphere with a resolution of 80 faces.

In order to evaluate a candidate view, a set of rays is traced in the voxel map simulating a range sensor. Once the rays have been traced the gathered information is used by the utility function to rank the candidate view.

IV. THE UTILITY FUNCTION

The utility function quantifies the desirability of a view. A new view must: include unseen areas, provide a percentage of overlap with previous range images, improve voxel quality and reduce the distance the sensor needs to travel. We propose a set of factors that quantify each characteristic of a desirable view, the combination of these factors form the utility function.

A. Area Factor

The objective of the area factor is to perceive unseen areas and to provide at the same time some overlap with

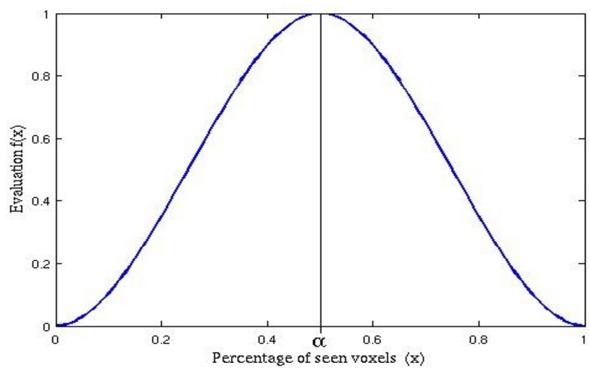


Fig. 3. The graph shows the desired behavior for the function f_i for each type of voxel considered in the area factor.

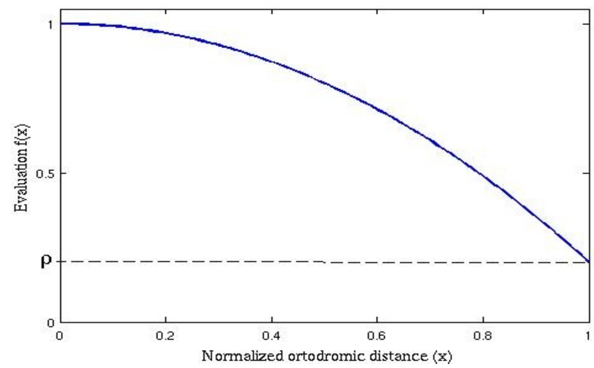


Fig. 4. The graph shows the desired behavior for the navigation factor

previous scans. Therefore, this factor is based on measuring the percentage of each voxel type that is visible from a candidate view. Mathematically, the area factor is represented by the sum of functions f_i. Each function f_i gives a value in the range $[0, 1]$ based on the percentage of its corresponding voxel type. Fig. 3 shows the desirable behavior of these functions. Each function f_i reaches a maximum when the percentage x_i of voxels i is equal to an optimum percentage, α_i, specified by the user ($f_i = 1$ when $x_i = \alpha_i$). The function reaches a minimum when the percentage x_i is either 1 or 0, that is to say, the sensor only perceives voxels of this type or it does not perceive these kind of voxels at all in the image. We can not use a Gaussian function because it is not always symmetric, its shape depends on α.

The function f is subject to the following constraints:

$$\begin{array}{ll} f(0) = 0 & f'(0) = 0 \\ f(\alpha) = 1 & f'(\alpha) = 0 \\ f(1) = 0 & f'(1) = 0 \end{array}$$

$$f(x) > 0 \text{ for all } x \in [0, 1]$$

$$f'(x) > 0 \text{ for all } x \in [0, \alpha]$$

$$f'(x) < 0 \text{ for all } x \in [\alpha, 1]$$

We use a general third degree polynomial as our function, divided in two parts, otherwise the function gives negative values:

$$\begin{array}{l} A_1x^3 + B_1x^2 + C_1x + D_1, \ 0 < x \leq \alpha \\ A_2x^3 + B_2x^2 + C_2x + D_2, \ \alpha < x < 1 \end{array} \quad (1)$$

Applying the constraints to the function and its derivative, we solve the system of equations (1) to obtain two expressions (3),(4) depending only on α and x:

$$f(x, \alpha) = \begin{cases} f_1(x, \alpha), & x \leq \alpha \\ f_2(x, \alpha), & x > \alpha \end{cases} \quad (2)$$

where

$$f_1(x, \alpha) = -\frac{2}{\alpha^3}x^3 + \frac{3}{\alpha^2}x^2 \quad (3)$$

and

$$f_2(x, \alpha) = -\frac{2}{(\alpha - 1)^3}x^3 + \frac{3(\alpha + 1)}{(\alpha - 1)^3}x^2 - \frac{6\alpha}{(\alpha - 1)^3}x + \frac{3\alpha - 1}{(\alpha - 1)^3} \quad (4)$$

To guarantee an overlap in the next view we consider the percentage of occupied voxels and to perceive unseen areas we consider occplane voxels. The percentage of occplane voxels is x and the percentage of occupied voxels y, the optimal percentage for occplane voxels is α and the optimal percentage for occupied voxels is β, then the area factor is:

$$f_{area} = f(x, \alpha) + f(y, \beta) \quad (5)$$

The percentage of a voxel type is calculated as the amount of voxels of this type divided by total number of voxels in the ray tracing, empty voxels are not considered. In the experiments we used $\alpha = 0.8$ and $\beta = 0.2$

B. Navigation Factor

The objective of the navigation factor is to evaluate a candidate view based on the navigation distance between the current view point and a candidate view. To measure the distance between view points, a metric in the configuration space of the robot should be used. Also, the distance should include collision avoidance. As we assume that our sensor (robot) has no motion constraints (free-flyer), it can be positioned at any point of the view sphere around the object. Then, the orthodromic distance, defined as the shortest distance between two points on a sphere, seems a reasonable option. The minimum distance in this case is when the camera does not move, it only changes its orientation. The maximum distance is defined by two opposite points on the sphere, so we can define a normalized orthodromic distance, x. In a future work, this distance metrics could be changed according to the robot kinematics constraints.

The navigation factor $f_{navigation}$ must satisfy the next constraints: (i) the candidate view with the shortest distance has the maximum value; (ii) the candidate view with the largest distance has a minimum value given by ρ; (iii) a candidate view with larger distance than others must have smaller value; and (iv) at short distances the value of the function decreases more slowly than at large distances. Fig. 4 shows the desired behavior for the navigation factor. Therefore, the constraints for the function are:

$$\begin{array}{ll} f(0) = 1 & \\ f(1) = \rho, & 0 \leq \rho < 1 \\ f'(x) < 0, & x \epsilon [0,1] \\ f'(0) = 0 & \end{array}$$

We use a second grade polynomial (6) to model the behavior in fig. 4:

$$\begin{array}{c} f(x) = Ax^2 + Bx + C \\ f'(x) = 2Ax + B \end{array} \tag{6}$$

Similar to the area factor, we apply the constraints to the polynomial to solve the system of equations to find the expression of the navigation factor (7) in function of the normalized ortodromic distance x and ρ:

$$f_{navegation} = (1 - \rho) x^2 + 1 \tag{7}$$

C. Quality Factor

The quality of a sensed voxel is the cosine of the angle formed between the surface normal and the sensor optical axis. For a candidate view, this factor measures the quality of the overlap area (occupied voxels). With the measured quality, it makes a prediction of the quality that is going to be taken from that candidate view. Then, the quality factor gives preference to the views that have a better quality prediction, in other words, it gives preference to views that are orthogonal to the overlap area. This idea was used previously by [6]:

$$f_{quality} = \frac{\sum i{=}1{,}n\_occu \cos(\alpha_i)}{n\_occu} \tag{8}$$

where α_i is the angle formed between the surface normal of the occupied voxel i and the sensor optical axis, and n_occu is the amount of occupied voxels in the ray tracing.

D. Occlusion Factor

The occlusion factor aims to solve occluded areas (e.g. object self-occlusions). This factor gives a higher value to the views that see more occplane voxels:

$$f_{occlusion} = \frac{n\_occlusion}{pts\_h * pts\_v} \tag{9}$$

where $n\_occlusion$ is the amount of occplane voxels, pts_h is the amount of horizontal points and pts_v is the amount of the vertical points from sensor resolution.

E. Utility Function

The utility function is a combination of the previous factors. This combination considers as a basic factor the area factor and the others as secondary factors (the reason is that the main objective of the view planner is to reconstruct, as much as possible, the object while an overlap is provided in each view):

$$f_{utility} = f_{area} * (f_{quality} + f_{navigation} + f_{occlusion}) \tag{10}$$

V. SEARCH STRATEGY

An exhaustive search tests each candidate view with full sensor resolution. So a densely approximated sphere and a high resolution guarantees that a good view will be find. However the computational cost is extremely high and these evaluations must be done in each iteration, so a more efficient alternative is required. We propose two strategies to find the next best view: a recursive hierarchical strategy and a multi-resolution strategy.

A. Recursive Hierarchical Strategy

This strategy exploits a hierarchical discretization of the sphere and it consists of two steps. In the first step, the faces of an icosahedron are taken as candidate views. Each view is evaluated with full sensor resolution and ranked according to the value of the utility function. Then, the best view is selected. In the second step, we apply a recursive tessellation of the face that corresponds to the selected view, giving as a result a new set of candidate views. The new views are tested and the best evaluated view is selected. If the final resolution level (l) is reached the search ends, otherwise the second step is repeated. The process is shown in algorithm 2.

Algorithm 2: Recursive Hierarchical Strategy

```
Data: l (final view sphere resolution level)
Result: nbv
i ← 0;
V ← Icosahedron views;
Evaluate V with full sensor resolution;
v ← view with the best evaluation of V;
while i < l do
    H ← Set of tesselated views from the face v;
    Evaluate H with full sensor resolution;
    v ← Best evaluated view from H;
    Increase n;
end
nbv = v;
```

B. Ideal Ray Tracing Resolution

The ideal ray tracing resolution (R_{ideal}) is closely related to the voxel map resolution. The traced rays should be separated from each other by a distance equal to the dimensions of the voxels. For a voxel map with length len and a sensor with horizontal aperture α and vertical aperture β positioned at a distance r from the center of the voxel map, the ideal ray tracing resolution is computed as follows:

$$pts\_h = \frac{\alpha}{\arcsin\left(\frac{len}{r}\right)}; \; pts\_v = \frac{\beta}{\arcsin\left(\frac{len}{r}\right)}$$

Where pts_h is the amount of horizontal points and pts_v is the amount of vertical points. The ideal ray tracing resolution is the lowest required for the ray tracing to distinguish the model details from the voxel map.

C. Multi-Resolution Strategy

The multi-resolution strategy is based on the variability of the sensor resolution while the amount of candidate views remains unchanged. This strategy is formed by k stages. In each stage a set of views is evaluated with a sensor resolution for that stage. In the first stage, all views from view sphere (V) are evaluated with a minimal ray tracing resolution (R_{min}) (set by the user). The best evaluated views are selected for the next stage. In the next stage, the subset of selected views is evaluated with a higher resolution. Again, a subset with the best evaluated views is selected. The process continues until the k stage is reached, where a reduced subset of views is evaluated with the ideal ray tracing resolution. Algorithm 3 presents this search strategy.

Algorithm 3: Multi-resolution Strategy

Data: V, k, R_{min}, R_{ideal}
Result: nbv
$R_1 = R_{min}$;
for *i=1:k-1* **do**
  Evaluate V with resolution R_i;
  $V \leftarrow$ Select the n_i best evaluated views;
end
Evaluate V with resolution R_{ideal};
$nbv \leftarrow$ Best evaluated view from V;

VI. SIMULATION RESULTS

Our algorithm was tested in simulation. The range camera was simulated and the objects were taken from virtual models, except for the bunny and dragon objects that were taken from the Stanford repository.

The simulated range camera returns a range image, similar to a real range camera with 45 degrees for horizontal and vertical apertures and a resolution of 320 x 320 points. The system was implemented in C language. The machine used for the tests was an AMD Turion64 with 2 GB of RAM memory.

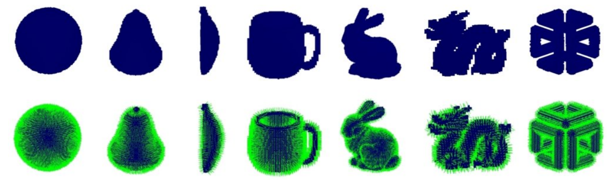
The objects are showed in fig. 5(a). These objects have different shapes and present incremental difficulty for the reconstruction process: sphere is a convex simple object; pear and banana are nonconvex; mug, bunny and dragon are non convex with self-occlusions; and finally sgi-logo has holes and self-occlusions.

A. Search Strategy

In this test we evaluate each search strategy. The voxel map dimensions for the scene were 205 x 205 x 205 (8 615 125) voxels and 55 x 55 x 55 (166 375) voxels for the object enclosure. The voxel size was 0.02m. We used a sphere with a radius of 2m.

The exhaustive strategy was tested with 20 and 80 views, using a full sensor resolution of 320x320. The hierarchical strategy started with 20 views and finished with 80 views in the second step, each candidate view was evaluated with full sensor resolution. The multi-resolution, was tested with:

(a) List of synthetic objects



(b) View planner outputs

Fig. 5. Reconstructed 3D objects. In the top row are the CAD object models, at the bottom we find the reconstructed models with surface normals

$k = 2$, $R_{min} = 40\text{x}40$, $R_{ideal} = 158\text{x}158$, $n_1 = 10$ and 80 candidate views.

Using the hierarchical and multi-resolution strategies we reconstructed every object in the test set, despite the complexity of their shape. Fig. 5(b) shows the reconstructed object models.

The time required for each iteration was 25.6s for exhaustive search with 20 candidate views, 135.5s for exhaustive with 80 views, 30.5s for hierarchical strategy (80 views) and 6.2s for multi-resolution strategy (80 views). We note that the multi-resolution strategy is the fastest. Moreover, if we decrease the number of selected views (n_1) to 5 we can get a computation time of 4.9s, but we prefer 10 selected views to increase the strategy efficacy. Since this time is required for each iteration, the time saving is considerable.

Table I presents the results for the multi-resolution strategy: column *Views* shows the number of views needed to complete the model; *Occup.* shows the amount of occupied voxels; *Qual.* shows the voxel map quality (quality mean of all occupied voxels); and *Dist.* shows the total traveled orthodromic distance measured in meters. For the hierarchical method the results are similar (omitted due to space limitations).

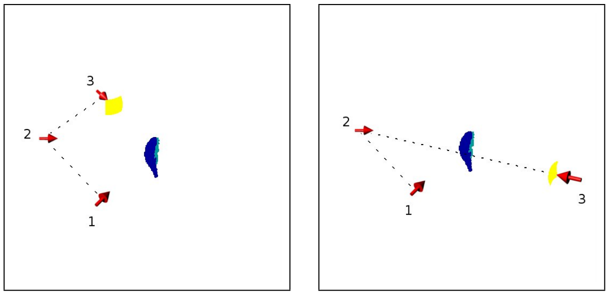
TABLE I

RECONSTRUCTION DATA FROM THE MULTI-RESOLUTION STRATEGY

Object	Views	Occup.	Qual.	Dist.
Sphere	6	7200	0.757	17.70
Pear	7	4342	0.803	16.40
Banana	4	1049	0.765	7.15
Mug	8	7736	0.718	21.76
Bunny	9	5395	0.821	23.56
Dragon	9	3639	0.798	22.14
SGI-Logo	12	11900	0.832	32.53

B. Navigation Factor Behavior

With this test we show the effect of the navigation factor in the reconstruction. We used the multi-resolution search strategy. We run the algorithm with and without the navigation factor in the utility function. Table II shows the results.



(a) Path with the nav. factor (b) Path without the nav. factor

Fig. 6. Views until the 3rd iteration. The paths are seen from a top view perspective that gives the wrong impression that all the views are in a horizontal plane. The sensor positions (views) are at different heights.

The columns *w* show the results with the navigation factor and the columns *wo* show the results without it. There is a significant reduction in the distance traveled by the sensor in 5 of the 7 objects. In Fig. 6 we show an example of how the navigation factor affects a reconstruction. In that figure we can see that in the 3rd iteration a closer view is selected when the navigation factor is used.

TABLE II

COMPARISON WHEN THE NAVIGATION FACTOR IS PRESENT (W) OR NOT (WO) IN THE UTILITY FUNCTION

Object	Views		Qual.		Dist.	
	w	wo	w	wo	w	wo
Sphere	6	6	0.75	0.75	17.7	20.9
Pear	7	7	0.80	0.82	16.4	19.7
Banana	4	4	0.76	0.81	7.15	9.50
Mug	8	8	0.71	0.72	21.7	22.7
Bunny	9	9	0.72	0.81	23.5	23.8
Dragon	9	9	0.79	0.78	22.1	28.4
SGILog.	12	12	0.83	0.83	32.5	37.0

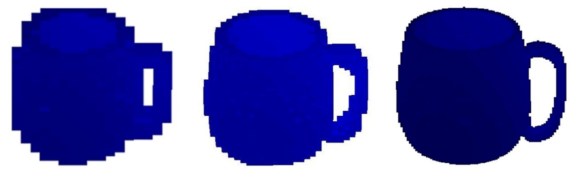
C. Voxel Resolution

Our system is able the operate with different scales and voxel map resolutions. In table III and fig. 7 we show the result from the use of different voxel sizes (voxel resolution) in the reconstruction of the object mug. The column *Time* shows the time required by iteration, in seconds; column *V* shows the number of views needed to complete the model. We can see that when the voxel size is smaller the number of views increases, because more details and occlusion areas have to be seen. On the other hand, the computation time increases since the ideal ray tracing resolution used by the strategy increases according to the voxel map dimensions.

TABLE III

RESULTS OF USING DIFFERENT VOXEL RESOLUTION.

Res.	Vxls. in Map.	Occup.	V.	Time	Qual.	Dist.
0.5	$614,125$	$1,063$	5	1.27	0.70	14.3
0.3	$2,628,072$	$3,408$	7	2.81	0.72	18.8
0.2	$8,615,125$	$7,909$	8	6.90	0.72	21.7
0.1	$66,430,125$	$30,374$	11	44.80	0.72	27.1



(a) Res. 0.5 (b) Res. 0.3 (c) Res. 0.1

Fig. 7. Comparison between different voxel resolution models.

VII. CONCLUSIONS AND FUTURE WORKS

We have presented a view planning algorithm that determines the next best view to reconstruct any arbitrary unknown 3D object. Based on a voxel map representation and a view sphere to position a range sensor, the algorithm plans a series of views according to a novel utility function and using an efficient search strategy. The planner has been evaluated in simulation, reconstructing 7 different objects of different complexities. The results show good quality in the reconstructed models, with significant savings in computation time and the distance traveled. The main contributions are: (i) a utility function that takes into account the the navigation distance; and (ii) two novel search strategies, one based on a hierarchical tessellation and other in a multi-resolution ray tracing.

Given the promising results obtained in simulation, our future work is to test it in a mobile manipulator with a stereo vision system.

VIII. ACKNOWLEDGMENTS

This work was supported in part by Conacyt grant No. 47968 and the National Institute of Astrophysics, Optics and Electronics (INAOE)

REFERENCES

[1] S. Chen, Y. Li, J. Zhang, and W. Wang, *Active Sensor Planning for Multiview Vision Tasks*. Springer-Verlag, 2008.

[2] W. R. Scott, G. Roth, and J.-F. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," *ACM Comput. Surv.*, vol. 35, no. 1, pp. 64–96, 2003.

[3] L. M. Wong, C. Dumont, and M. A. Abidi, "Next Best View system in a 3-d Object Modeling Task," in *Computational Intelligence in Robotics and Automation, 1999. CIRA '99. Proceedings. 1999 IEEE International Symposium on*, 1999, pp. 306–311.

[4] N. A. Massios and R. B. Fisher, "A Best Next View Selection Algorithm Incorporating a Quality Criterion," in *British Machine Vision Conference BMVC98, Southampton*, 1998, pp. 780–789.

[5] J. Sanchiz and R. Fisher, "A next-best-view algorithm for 3d scene recovery with 5 degrees of freedom," in *British Machine Vision Conference, Nottingham, UK*, 1999, pp. 163–172.

[6] M. T. Lozano Albalate, M. Devy, and J. M. Sanchiz Martí, "Perception Planning for an Exploration Task of a 3d environment," in *ICPR '02: Proceedings of the 16 th international conference on pattern recognition (ICPR'02)*, vol. 3. Washington, DC, USA: IEEE Computer Society, 2002, pp. 704– 707.

[7] P. Blaer and P. Allen, "Data acquisition and view planning for 3-D modeling tasks," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, Nov. 2007, pp. 417–422.