3D Object Reconstruction with a Single RGB-Depth Image

Silvia Rodríguez-Jiménez, Nicolas Burrus and Mohamed Abderrahim
Department of Systems Engineering and Automation, Carlos III University of Madrid, Leganés, Spain

Keywords: 3D Reconstruction, Robotics, Vision, RGB-D, Kinect.

Abstract: This paper presents a fast method for acquiring 3D models of unknown objects lying on a table, using a single viewpoint. The proposed algorithm is able to reconstruct a full model using a single RGB + Depth image, such as those provided by available low-cost range cameras. It estimates the hidden parts by exploiting the geometrical properties of everyday objects, and combines depth and color information for a better segmentation of the object of interest. A quantitative evaluation on a set of 12 common objects shows that our approach is not only simple and effective, but also the reconstructed model is accurate enough for tasks such as robotic grasping.

1 INTRODUCTION

The objective of this work is to acquire 3D models of unknown objects lying on a table, using a single viewpoint. This is of particular interest for applications that have to deal with new objects constantly, such as augmented reality or general-purpose robotic manipulation, which is the context of this paper (Figure 1). With the availability of inexpensive RGB-Depth (RGB-D) cameras such as the Microsoft Kinect (Microsoft, 2010), dense color and depth information about the scene can be acquired in real-time with a good precision at short distances. Thus, a RGB-D image already contains a lot of information, but a single image only provides the geometry of the visible parts (Figure 2). Due to self-occlusions, the hidden parts create empty gaps that have to be estimated using a priori knowledge.

The literature on object reconstruction from multiple views is large, but single view modeling has received a significant interest only recently, mostly motivated by robotic grasping applications. A first category of methods assumes that the objects to be modeled have a simple enough shape, and try to fit a predefined set of shape primitives (Kuehnle et al., 2008) (spheres, cylinders, cones or boxes) or a combination of them (Miller and Allen, 2004). This approach was made more general in other works such as (Sun et al., 2011) and (Thomas et al., 2007) by using a database of objects with known shapes and a recognition module.

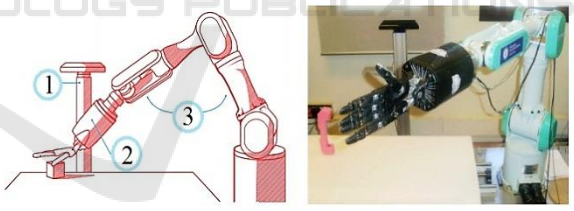


Figure 1: Robotic platform which is the scenario of this paper: (1) The Kinect camera is located on the side, oriented to get a top view of the objects; (2) a 20-DoF, five-fingers anthropomorphic hand from Shadow; (3) a 7-DoF PA-10 arm.

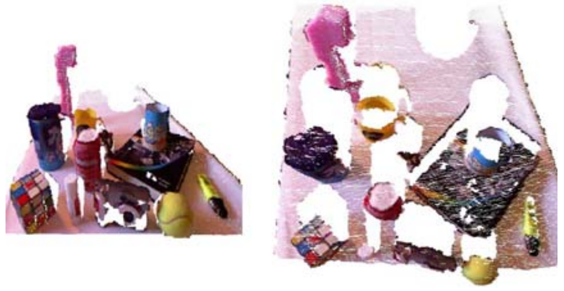


Figure 2: Example of a point cloud from the Kinect camera. Left: view of the visible parts from everyday objects lying on a table. Right: same point cloud from top view, where empty gaps belong to occluded parts.

When an extensive database of object models is not available or practical, more generic a priori assumptions are required. The most common one is to rely on the symmetries of real-life objects (Thrun and Wegbreit, 2005). The problem then becomes to

Rodríguez-Jiménez S., Burrus N. and Abderrahim M..
3D Object Reconstruction with a Single RGB-Depth Image.
DOI: 10.5220/0004286401550163
In *Proceedings of the International Conference on Computer Vision Theory and Applications* (VISAPP-2013), pages 155-163
ISBN: 978-989-8565-48-8
Copyright © 2013 SCITEPRESS (Science and Technology Publications, Lda.)

find the nature of the symmetries in the partial point cloud. These are hard to estimate in practice because of the large search space and limited data, leading e.g. to limit the set of hypotheses to a vertical plane axis in a restricted range (Bohg et al., 2011), or to focus on rotational symmetries (Marton et al., 2010).

Modeling 3D objects by symmetry is a common approach because many objects are symmetric, but also, a large class of everyday objects, especially when manufactured, can be generated by extruding a 2D shape through an extrusion axis. The extrusion process is widely used by designers and engineers to generate 3D models from 2D sketch input. This approach is particularly adapted to the fast reconstruction of objects lying on a flat table, which is a common scenario in robotics, because the table plane normal provides a natural extrusion axis. Thus, this paper proposes to leverage this property by reconstructing the hidden parts with an extrusion of the top view of the objects.

The contributions of this paper are three-fold. First, we propose a new technique to extrude an initial sparse point cloud output by a tabletop object detector. Second, we propose a refinement step that takes advantage of the complementarity of the depth and color images by carefully initializing a graph-cut based color segmentation with the depth data. Finally, a quantitative evaluation of the accuracy of the reconstructed meshes is performed on a set of 12 common use objects, showing that its effectiveness is comparable to the most recent approach using symmetries (Bohg et al., 2011). Some preliminary experiments for grasping applications are also conducted using the OpenRAVE simulator (Diankov, 2010).

2 GLOBAL OVERVIEW

For achieving our aim of the acquisition of 3D models using a single RGB-D image, we propose an algorithm which can be divided into two main stages: computation of the initial volume (Section 3) and its completion through color-based model refinement (Section 4). These stages include several steps which are illustrated in Figure 3.

In the first stage, a table-top object detector identifies and extracts a cluster of 3D points belonging to the object. Then, existing points are extruded along the table plane normal to fill a voxelized volume around the cluster of interest. Object concavities may get filled during the extrusion step, which we compensate by checking the voxel consistency against the depth image.

Depth images output by low-cost RGB-D cameras are usually imprecise around the object borders, and frequently have holes due to reflections or other optical effects. Since the color image does not suffer from these issues, in the second stage, we refine the object boundaries using color segmentation. The refined set of voxels is then given as an input to the final meshing algorithm.

3 COMPUTATION OF THE INITIAL VOLUME

3.1 Cluster Extraction

A table top object detector similar to (Rusu et al., 2010) is run on the depth image. The dominant 3D plane is first fitted to the depth data using RANSAC, then points lying outside of a prism around the table plane are eliminated. Remaining points are then clustered using Euclidean distances with fixed thresholds. Clusters that are too small or do not touch the table are eliminated. The cluster of interest is then determined in a task-dependent way, e.g. by choosing the most central one. To make 3D processing faster and get a natural neighborhood between 3D points, a voxelized volume of fixed size is then initialized around the cluster, and the voxels corresponding to a cluster point are labeled as "object". The voxel size is a user-defined parameter depending on the desired precision/speed tradeoff. All reconstructions shown in this paper are with 3mm voxels.

3.2 Voxel Filling by Extrusion

The objective of this step is to "fill" the occluded parts by relying on the assumption that the object can be approximated by an extrusion process. Taking into account that the table plane normal provides the natural extrusion axis for most objects, it is not necessary to calculate the object axis to get the extrusion direction. Instead, we consider the table plane normal as the extrusion direction of the top face of the object. The proposed algorithm is the following:

1. For each voxel which is considered as "object", compute the line segment going from the voxel to the plane along the plane normal.
2. Label all voxels intersecting a line segment as "maybe object".

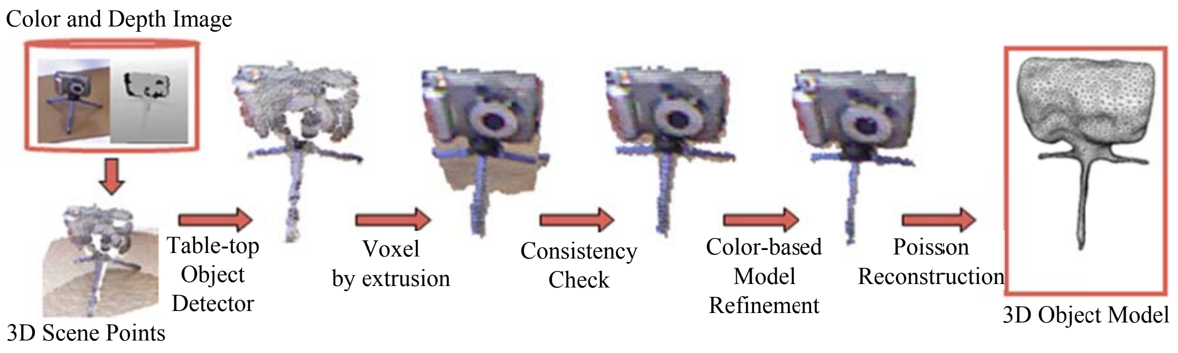


Figure 3: Overview of our model acquisition process. The Kinect camera provides a depth and color image, which are used to obtain the initial volume. This stage includes three key points: table-top object detector, voxel filling by extrusion and consistency check. The computed model is the input of the color-based refinement model stage.

The result of this step is a rough estimation of the object volume. The model is then slightly smoothed by running a morphological closing to cope with the uncertainties around object borders in the depth image. The optimal structuring element size depends on the voxel size and the properties of the depth data. For voxels of 3mm and a Kinect camera, we empirically found that a 3x3x3 cube is a satisfying structuring element. An example of output is given in Figure 4.

3.3 Consistency Check

The extrusion step may fill regions of the object that correspond to holes or concavities. This can be corrected by checking the consistency of "maybe object" voxels against the depth image. This is done by reprojecting each voxel onto the depth image, and comparing the projected depth with the depth image.

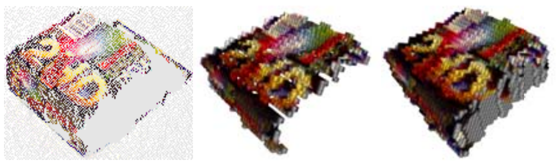


Figure 4: Voxel filling by extrusion. Left: raw point cloud of a box. Middle: voxelized mesh of the raw object cluster. Right: voxelized mesh after extrusion towards the table plane. Gray voxels correspond to unseen parts due to self-occlusions.

If the difference is greater than a threshold δd, the voxel is labeled as "background". The threshold depends on the estimated accuracy of the depth sensor, and is set to 3mm in all experiments with Kinect. The output of this process is illustrated in Figure 5.

4 COLOR-BASED MODEL REFINEMENT

After the above steps, the obtained 3D object model may still have missing parts and irregularities due to missing or incorrect depth information in the RGB-D frame. Incorrect pixels in the depth image usually belong to object borders and areas of specular, transparent or reflective objects. Observing that the color image do not suffer from these issues, we propose to improve the quality of the model by first refining the object segmentation using the color image and then filling-in incorrect depth values using image inpainting.

4.1 Improvement of the Object Segmentation

There are many existing techniques for color-based segmentation, but this is still an open problem in the general case. However, when a good initialization is available, graph-based techniques (Boykov and Jolly, 2001) have proven very effective for foreground/background segmentation (Lombaert et al., 2005). In particular, the GrabCut variant (Rother et al., 2004) combines graph cuts with Gaussian mixture models and is designed to take advantage of a user provided mask. It is thus particularly adapted to the refinement of an initial segmentation. Recently, GrabCut has been extended in the work described in (Vaiapury et al., 2010) to use depth information by combining the RGB and depth channels with a weighting factor. Instead of merging both information in a single energy, we propose to run GrabCut only on the color image, but using depth information for the initialization of an accurate mask. This approach takes a greater advantage of the complementarity of the techniques, since the depth image is misleading near the object borders, and the color information is not necessary and more sensitive to background clutter for the initial segmentation. The initialization is thus taken from the initial model output by the algorithm of Section 3 using depth information only, re-projecting every 3D point from the volume onto the depth image. Then, pixels are labelled as object (foreground), background or unknown (if their projected depth is not consistent or they do not have depth information). Taking this initialization as a starting point, the mask is created. GrabCut can take four different initialization values according to pixels belong to foreground, background, most probably foreground or background. Pixels which have been considered as foreground and background will not be changed by the algorithm and thus ensure a good robustness to segmentation errors. To handle the uncertainty associated to edge pixels in the depth image, only pixels which are not on a boundary are marked using those definitive labels. The GrabCut algorithm on the color image is then run using the computed mask for one iteration.

Thanks to the accuracy of the initial mask, Grab-Cut performs well even if the object and the background have a similar color distribution or if the background is cluttered, as shown in Figure 6.

4.2 Hole Filling through Depth Inpainting

The obtained object segmentation is accurate but some pixels which have been classified as object after the color refinement do not have depth. Most of the hole filling methods use image interpolation or inpainting techniques to fill up the remaining holes using neighbouring pixels. Recently, to improve the depth map output by Kinect, a cross-modal stereo vision approach has been presented in (Chiu et al., 2011). However, it does not benefit from a foreground/background segmentation. Furthermore, a hole-filling method using depth-based inpainting for 3D video was proposed in (Oh et al., 2009).

Following this, image inpainting is proposed in this work to fill missing depth values, but using the segmentation mask to fill pixels with only surrounding values of the same kind. Thus, object holes are filled only with depth information coming from the other "object" pixels and background holes are filled only with depth coming from surrounding "back-ground" values.

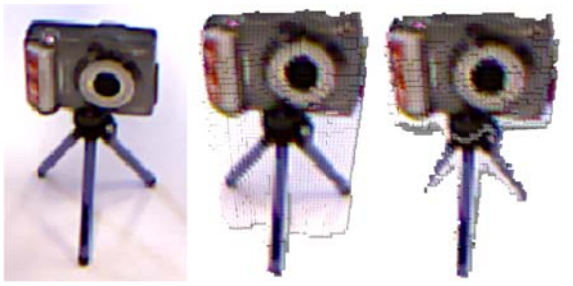


Figure 5: Consistency check to carve holes and concavities. Left: color image. Middle: colorized voxels after extrusion. Right: remaining voxels after consistency check. Holes and concavities that were wrongly filled by the extrusion algorithm are removed if they are visible.

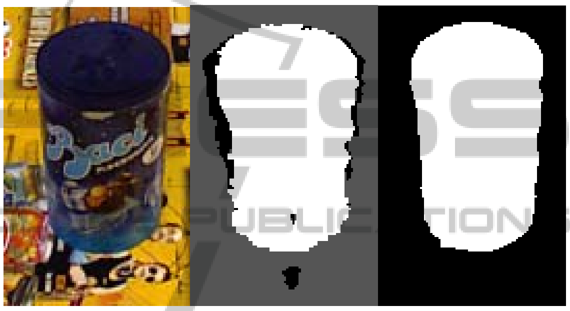


Figure 6: Snapshot of a refined object segmentation even when the foreground is similar to background. Left: color image, the object of interest is a storage jar on a color poster. Middle: initial segmentation. Pixels are marked as: unknown (black), object (white) and background (gray). Right: final object segmentation, after Grabcut. Pixels are marked as: object (white) and background (black).

The OpenCV implementation of Telea (Telea, 2004), a fast inpainting technique based on fast marching, is the used method. It takes as input the original depth image and an inpainting mask specifying the pixels to be filled.

To fill the depth image, two masks are used depending on the pixel class:

1. Object: the target area to be filled corresponds to "object" pixels without depth value or with inconsistent depth values determined in Section 4.1. Pixels which belong to background are also marked as target area to prevent them from influencing the inpainting.
2. Background: the target area corresponds to pixels labeled as “background” without depth value. Similarly to the previous case, "object" pixels are also marked as target area.

Once the depth image has been refined and filled, the algorithm of Section 3.3 is run again. The improvement obtained after segmentation refinement and depth inpainting is shown in Figure 7, filling pixels without depth information and border pixels whose depth was not correct. The final object model is obtained using Poisson surface reconstruction (Kazhdan et al., 2006) on the voxelized point cloud to create a smooth mesh of the object.

5 EXPERIMENTS

5.1 Evaluation of the Accuracy of the Reconstructed Mesh

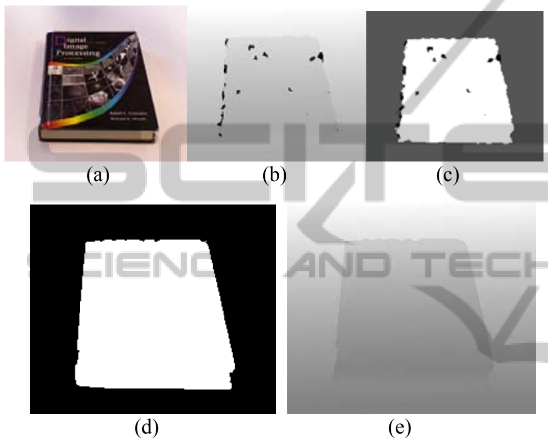


Figure 7: 2D images results of the color-based model refinement using a book as illustrative example. Images from the Kinect camera: (a) color image and (b) depth image. (c) Initial segmentation re-projecting every 3D point from the volume of Section 3.3. Pixels are marked as: unknown (black), object (white) and background (gray). (d) Final object segmentation after GrabCut according to Section 4.1. Pixels are marked as: object (white) and background (black). (e) Depth image after hole filling through depth inpainting.

The proposed algorithm has been tested on a set of 12 real objects with very different sizes and shapes, which are shown in Figure 8. For each of the object, between 5 and 9 meshes have been acquired using our algorithm in the scenario showed in Figure 1, where the objects lie off the table in different orientations and places. Therefore, the data set contains 72 reconstructed models which have been calculated from a single view of Kinect camera. For the evaluation, the geometric difference between reference and reconstructed meshes using our proposed algorithm has been calculated. The reference models have been acquired with a commercial laser scanner.

The processing time of the whole algorithm is currently less than 2 seconds on a 2Ghz computer for a point cloud with less than 30000 points, significantly improving computation time achieved in (Bohg et al., 2011) with a similar number of points. Although this computation time is suitable for the current application, optimization is considered as future work.

A free 3D mesh processing software, MeshLab (MeshLab, 2011), has been used to compute the geometric difference between reference and reconstructed 3D models which should be well aligned in the same space. Iterated Closed Point (ICP) is used to align the meshes and the Hausdorff distance to measure the geometrical distance between them.

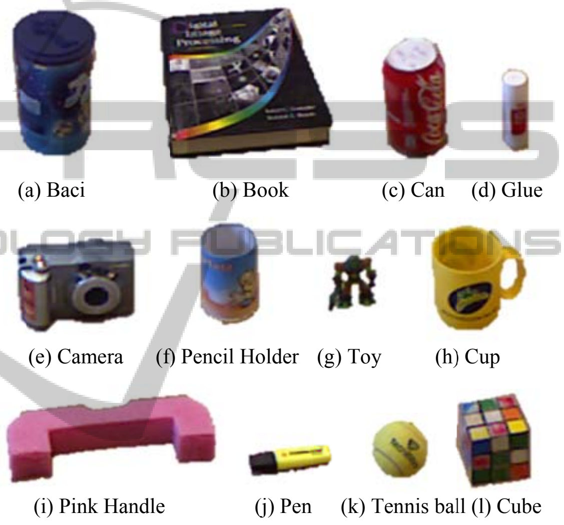


Figure 8: The 12 real objects of the database. For each object, at least 5 images have been acquired from the object in different orientations and places on the table.

Figure 9 shows the mean and the standard deviation between reference and reconstructed meshes for all objects. The average error for all meshes is 3.87mm and the standard deviation is 0.96mm. Taking into account that the objects are similar to the set used in (Bohg et al., 2011), our extrusion approach provides a similar effectiveness in comparison with earlier symmetry method and a significant improvement for large objects. With our method, the mean error is less than 5mm in all objects, independently of their size while in (Bohg et al., 2011) the average error is less than 7mm and 20mm for bigger objects.

It is important to note that the experimental measurement gathered is statistically very rich in the sense that each object image was captured in different orientations and different locations on the table, as it is shown in Figure 10. Such set takes into consideration most of the possible sources of errors,

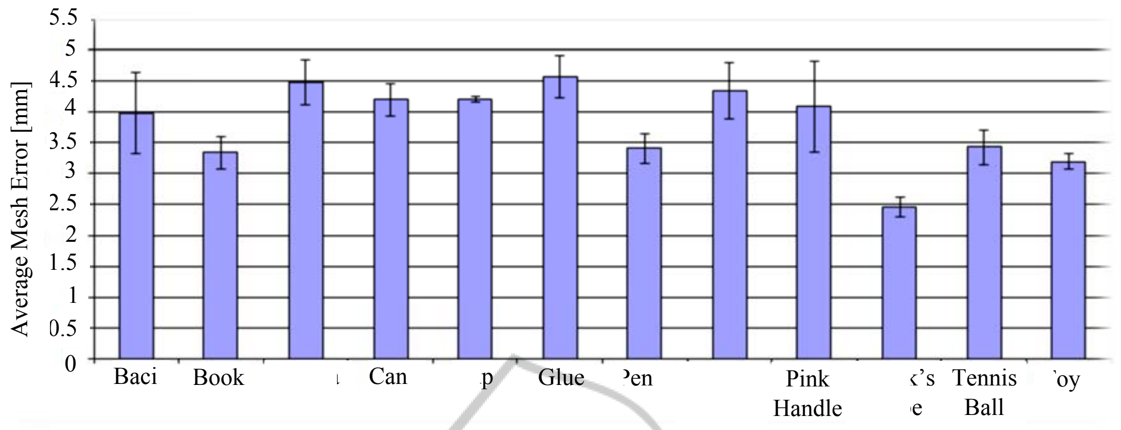
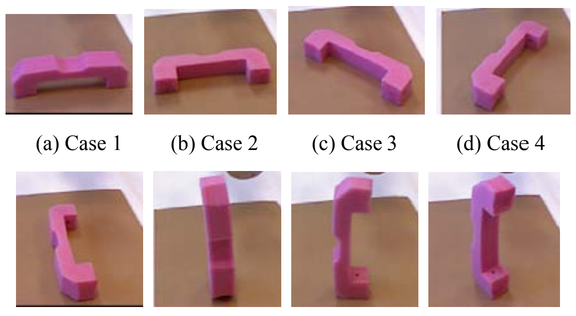


Figure 9: Evaluation of the mean error and standard deviation between reference and reconstructed meshes for all objects of the database. The mean error is less than 5mm in all objects, being the average error less than 4mm.

such as hiding different geometric details, reflections or other optical effects, which affect the obtained results and increase the error.

Figure 11 shows as, due to the position of the "pink handle" object, the error is lower when the visible parts provide enough information to approximate the geometry by an extrusion of its top-view (Case 2 and 3), but the error increases when the top-view is not very informative (Case 6).

Taking into account that the voxel size chosen in this work is 3mm (Section 3.1) then it is fairly obvious that we cannot obtain reconstructions with errors less that the mentioned voxel value. This can be seen in the table of the average errors of the different objects, where only the Rubik's cube average error is 2,5mm while the rest are above 3mm. If more precision is required, we may scarify the speed and improve the accuracy of the reconstructed object models. This can be done if a particular task requires it.



(a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

(e) Case 5 (f) Case 6 (g) Case 7 (h) Case 8

Figure 10: The "pink handle" object in the 8 evaluated orientations on the table.

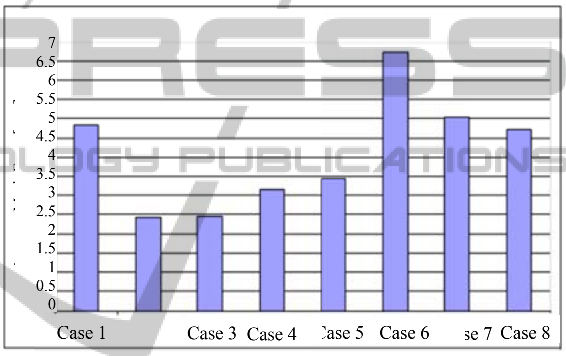


Figure 11: Evaluation of the error for 8 orientations on the table of the "pink handle" object, shown in the Figure 10. Comparing to its reference model, the mean error is 4.09mm and the standard deviation is 1.49mm.

5.2 Model Reconstruction Results

Figure 12 shows some meshes acquired using our algorithm for the tested set of 12 real objects. Figure 13 shows objects for which a very good model could be obtained despite of a very sparse initial point cloud. The quality of the top view is essential for the approach, but it was made significantly more robust thanks to the segmentation and depth filling steps. Reconstruction is even possible in some cases where almost no information was present in the original image.

Figure 14 gives examples of objects which have a geometry that cannot be roughly approximated by an extrusion of their top-view. Note that even if the obtained models are not very accurate, useful estimations for grasping are still obtained. Adding another camera with a different point of view would be enough to obtain a good model in these cases.

5.3 Application to Grasping

Since grasping itself is not the main scope of this paper, the suitability of the acquired meshes for grasping has been tested on a single object as a representative way. Both planning and grasping experiments have been performed within OpenRAVE simulator (Diankov, 2010). It simulates grasps in many positions to determine a set of stable grasps for a given object, as illustrated in Figure 15.

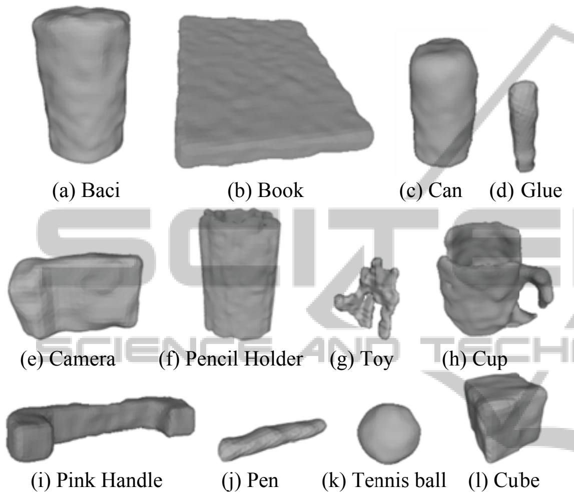


Figure 12: Model reconstruction results of the 12 real objects of the database, shown in Figure 8.

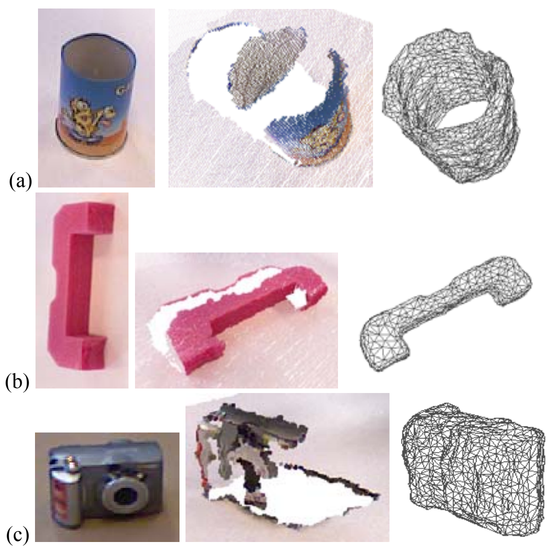


Figure 13: Model reconstruction results of a pencil holder (a), a pink handle (b) and a camera (c). Left: color image. Middle: initial point cloud (white points correspond to the table). Right: final mesh using Poisson reconstruction.

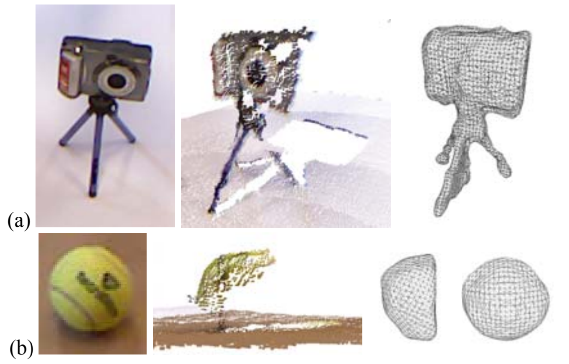


Figure 14: Model reconstruction results of a camera on a tripod (a) and a tennis ball (b). Left: color image. Middle: initial cluster. Right: final mesh using Poisson reconstruction.

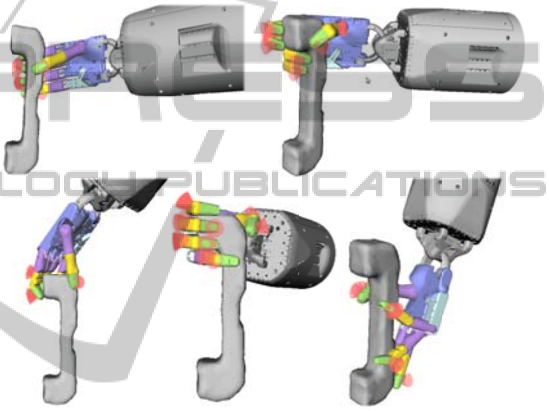


Figure 15: Five grasps of the grasp table generated by OpenRAVE for a pink handle whose mesh has been generated using the proposed algorithm.

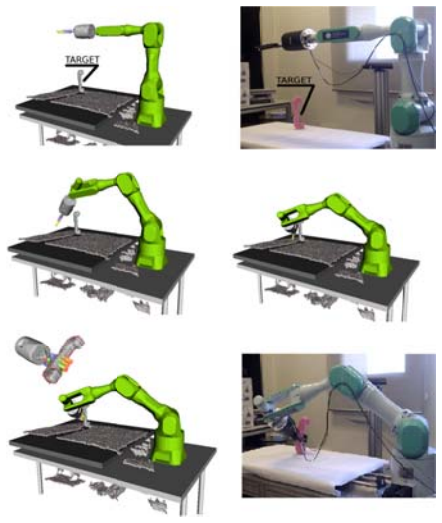


Figure 16: Simulated and real sequence of the trajectory toward the selected grasping position, which has been calculated off-line previously. Both planning and grasping have been performed within OpenRAVE.

Then it can be used for online path planning in a given scene, where the object is recognized and its pose estimated to perform the suitable grasp, which has been calculated off-line previously.

Figure 16 shows the sequence of the trajectory in simulation and on the real robot of our scenario (Figure 1), suggesting that the acquired mesh is suitable for grasping. A more exhaustive evaluation of grasping from a single viewpoint in simulation and on our robotic platform is considered as future work.

6 DISCUSSION AND FUTURE WORK

In this paper, a method that reconstructs a model of everyday, man-made objects from a single view has been proposed. We have validated the precision evaluating the difference between the reference and the reconstructed model for 12 real objects. The average error for all meshes is less than 4mm and the standard deviation is less than 1mm. Furthermore, compared to earlier methods, our approach provides 3D models improving run-times significantly with a similar accuracy and even, a significant improvement both in run-time and accuracy for bigger objects.

Experimental results with different objects demonstrate that the obtained models are precise enough to compute reliable grasping points. Thus, the current system is an easy and effective approach but it has some limitations when objects have very thin structures, or with objects whose top-view is not very informative. However, thanks to the generality of the proposed algorithm, this could be compensated by adding more cameras as needed, applying the same technique on each view and finally merging the resulting voxels. Furthermore, symmetry and extrusion could complement one another.

In the future, to handle a wider range of objects, rotational symmetries exploitation is planned through the combination with techniques of shape estimation such as the work described in (Marton et al., 2010). Moreover, for manipulation applications, the integration of single view estimation with the incremental model refinements techniques of e.g. (Krainin et al., 2010) and (Krainin et al., 2011) would be interesting. Finally, the combination of this approach with an online grasp planner is also planned to enable fast online grasping and manipulation of unknown objects.

ACKNOWLEDGEMENTS

The research leading to these results has been funded by the HANDLE European project (FP7/2007-2013) under grant agreement ICT 231640 – http://www.handle-project.eu.

REFERENCES

Bohg, J., Johnson-Roberson, M., León, B., Felip, J., Gratal, X., Bergstrom, N., Kragic, D., and Morales, A. (2011). Mind the gap-robotic grasping under incomplete observation. In *Proceedings of the IEEE International Conference on Robotics and Automation,* pages 686–693, Shanghai, China.

Boykov, Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary amp; region segmentation of objects in n-d images. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, pages 105–112, Vancouver, Canada.

Chiu, W., Blanke, U., and Fritz, M. (2011). Improving the kinect by cross-modal stereo. In *Proceedings of the British Machine Vision Conference*, pages 116.1–116.10. Dundee, UK.

Diankov, R. (2010). *Automated Construction of Robotic Manipulation Programs*. PhD thesis, Carnegie Mellon University, Robotics Institute.

Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Symposium on Geometry Processing*, pages 61–70.

Krainin, M., Curless, B., and Fox, D. (2011). Autonomous generation of complete 3d object models using next best view manipulation planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 5031–5037, Shanghai, China.

Krainin, M., Henry, P., Ren, X., and Fox, D. (2010). Manipulator and object tracking for in-hand model acquisition. In *Proceedings of the Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation at the IEEE International Conference on Robotics and Automation*, Anchorage, Alaska.

Kuehnle, J., Xue, Z., Stotz, M., Zoellner, J., Verl, A., and Dillmann, R. (2008). Grasping in depth maps of time-of-flight cameras. In *International Workshop on Robotic and Sensors Environments*, pages 132–137.

Lombaert, H., Sun, Y., Grady, L., and Xu, C. (2005). A multilevel banded graph cuts method for fast image segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, pages 259–265, Beijing, China.

Marton, Z., Pangercic, D., Blodow, N., Kleinehellefort, J., and Beetz, M. (2010). General 3d modelling of novel objects from a single view. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3700–3705.

MeshLab (2011). Visual Computing Lab-ISTI-CNR, <http://meshlab.sourceforge.net/>

Microsoft (2010). Kinect for Xbox 360, <http://www.xbox.com/en-US/kinect/>

Miller, A. and Allen, P. (2004). Graspit! a versatile simulator for robotic grasping. *IEEE Robotics Automation Magazine,* 11(4):110–122.

Oh, K.-J., Yea, S., and Ho, Y.-S. (2009). Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-d video. In *Proceedings of the Picture Coding Symposium,* pages 1–4.

Rother, C., Kolmogorov, V., and Blake, A. (2004). "Grab-Cut": interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23:309–314.

Rusu, R., Bradski, G., Thibaux, R., and Hsu, J. (2010). Fast 3d recognition and pose using the viewpoint feature histogram. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162.

Sun, M., Kumar, S. S., Bradski, G., and Savarese, S. (2011). Toward automatic 3d generic object modeling from one single image. In *Proceedings of the 3DIMPVT*, Hangzhou, China.

Telea, A. (2004). An Image Inpainting Technique Based on the Fast Marching Method. *Journal of graphics, gpu, and game tools*, 9(1):23–34.

Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., and Van Gool, L. (2007). Depth-from-recognition: Inferring meta-data by cognitive feedback. In *Proceedings of the IEEE International Conference on Computer Vision,* pages 1–8.

Thrun, S. and Wegbreit, B. (2005). Shape from symmetry. In *Proceedings of the IEEE International Conference on Computer Vision,* volume 2, pages 1824–1831, Beijing, China.

Vaiapury, K., Aksay, A., and Izquierdo, E. (2010). Grab-cutd: improved grabcut using depth information. In *Proceedings of the ACM workshop on Surreal media and virtual cloning,* pages 57–62, New York, NY, USA.