Point Cloud Completion Using Extrusions

Oliver Kroemer, Heni Ben Amor, Marco Ewerton, and Jan Peters
Intelligent Autonomous Systems
Technische Universitaet Darmstadt
Email: {oli, amor, peters}@ias.tu-darmstadt.de

***Abstract*— In this paper, we propose modelling objects using extrusion-based representations, which can be used to complete partial point clouds. These extrusion-based representations are particularly well-suited for modelling basic household objects that robots will often need to manipulate.**

In order to efficiently complete a partial point cloud, we first detect planar reflection symmetries. These symmetries are then used to determine initial candidates for extruded shapes in the point clouds. These candidate solutions are then used to locally search for a suitable set of parameters to complete the point cloud. The proposed method was tested on real data of household objects and it successfully detected the extruded shapes of the objects. By using the extrusion-based representation, the system could accurately capture various details of the objects' shapes.

I. Introduction

In the future, service robots working in everyday environments will need to grasp and manipulate a wide range of different objects. Given these unstructured environments, many of the encountered objects will be novel to the robot, and their complete shapes will initially be unknown. This shape information is however vital for successfully and efficiently manipulating the objects. Hence, the robot will need to autonomously determine the shapes of novel objects.

One approach to acquiring a suitable 3D model is to scan the object from different perspectives by either shifting the object or moving around the object [1]. The information from these multiple perspectives can then be accumulated to form a 3D model. Although this approach can acquire accurate object models, the process of acquiring multiple images is a time-consuming and non-trivial task.

Alternatively, the robot can attempt to predict the shape of an object from only a single perspective. The partial model acquired from one perspective can be completed by detecting patterns in the observed shape and extending these patterns into the occluded regions [2]. For example, planes of symmetry can be detected and, subsequently, used to complete the point cloud accordingly [3].

In this paper, we show how the partial point clouds of objects can be completed by detecting extruded shapes. Extruded 3D shapes are 2D shapes that have been extended into the third dimension along a particular path, such as a line segment (linear extrusion) or circle (rotational extrusion). For example, a cube is a linearly extruded square, and a sphere is a rotationally extruded circle.

Our overall approach to detecting extruded shapes is similar to the *shape from symmetry* framework proposed by Thrun and Wegbreit [4]. The robot first searches for planes of symmetry in the partial point cloud, which are entailed by both linear and rotational symmetries. The detected symmetries are then used to initialize local searches for suitable extrusion parameters. Finally, the detected extrusions are evaluated according to a scoring system, and used to complete the point cloud when applicable. The main contribution of this paper is therefore the use of extrusion-based, rather than symmetry-based, representations.

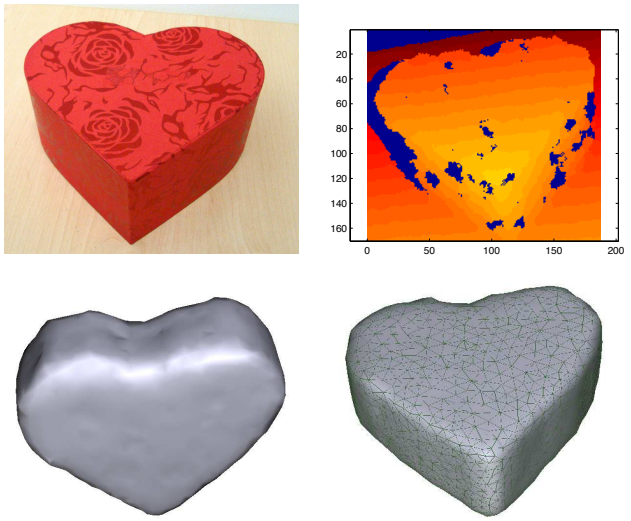

Figure 1. The top left image shows a heart-shaped box. The top right image shows the depth data collected from the heart-shaped box. The bottom images show the object models obtained using the extrusion-based approach to point cloud completion.

Extrusion-based representations are well-suited for robots working in everyday environments, wherein many objects are manufactured. Not only are linear and rotational extrusions often used to design objects, but they are also common in the manufacturing process. As a result, many everyday objects have basic extruded shapes. In this paper, we will be focusing on completing the point clouds of basic objects that can be described by single extrusions.

Although the proposed approach is similar to the shape from symmetry method, extrusion-based representations can complete some point clouds that the symmetry-based representations cannot. For example, the extrusion-based approach can complete surfaces by extruding edges, as illustrated in Fig. 1. The heart-shaped box has two curved surfaces that were not visible in the original point cloud. However, by

extruding the curved edges observed along the top of the box, the extrusion-based approach could complete these parts of the point cloud. The symmetry-based approach relies on projecting observed surfaces into occluded regions. Given that all of the observed surfaces are flat, this approach cannot complete the curved surfaces.

A similar problem can occur when completing the point cloud of a rectangular box when only two of the six sides are observed. The extrusion-based approach can extrude one surface according to the other in order to complete the entire box. The symmetry-based approach would, however, have problems completing the third pair of opposing surfaces. There are obviously also situations in which a symmetrical object cannot be modeled by an extrusion-based representation. Thus, these two representations complement each other and could be used together.

The proposed method is explained in Section II, which also includes an overview of related work in completing point clouds for robot applications. The applicability of the extrusion-based approach is demonstrated in Section III, wherein the robot successfully completes the shapes of various household objects from a single perspective.

II. Extrusion-based Point-Cloud Completion

In this section, we explain how the observed point clouds can be completed by detecting extruded shapes in the partial point cloud. We begin by giving a brief overview of methods used in robotics applications to complete partial point clouds. In Section II-B, we detail the extrusion-based representation, which is flexible enough to model a wide range of objects shapes. Sections II-C to II-F explain how one can search for extrusions in a partial point cloud.

The proposed method assumes that the robot's vision system acquires structured point clouds; i.e., each point cloud corresponds to a pixel in a 2D grid. This form of data can be acquired from dense stereo, time-of-flight camera, Kinect, or other active stereo cameras.

A. Point Cloud Completion in Robotics

Determining the shape of an object from a single view is a common problem in robotics. One approach to solving this problem is to provide the robot with a library of previously scanned models, which it can then fit into the observed scene [5, 6, 7, 8]. This approach allows the robot to accurately reconstruct the scene. However, it also relies on the robot having a model of the object, and requires searching through a large library of known objects. In the field of computer vision, Pauly et al. [9] presented a method for completing the shape of objects using the models of objects with similar shapes. Hence, a smaller library of objects could be used, as the objects generalize to novel objects.

Another approach is to fit primitive shapes, such as cubes and cylinders or superquadrics, to the partial view [10]. Primitive shapes cannot only be used to represent simple objects, but also parts of more complex objects [11]. The primitive shapes are generally parameterized such that they can be adapted to model a range of similar object parts. The ability to adapt the primitives in this manner is important, as the additional flexibility allows the model to capture more details of the object.

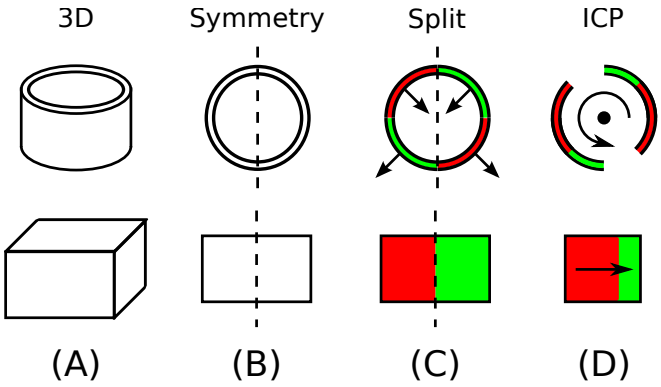

Figure 2. The figure illustrates how the initial extrusion hypotheses are generated. The top row corresponds to rotational extrusions, and the bottom row demonstrates linear extrusions. (A) The 3D objects of a tube and a box are shown. (B) A top view of the objects, as well as the detected plane of symmetry indicated by the dashed line. (C) The point clouds are divided into two regions, indicated by red and green. The arrows in the top image indicate the observed surface normals for these regions, which are used to divide the points. (D) The ICP algorithm is used to slide one region into the other. The rigid body transformation found by the ICP algorithm is then used to compute the extrusion parameters.

Point cloud completion can also be performed by predicting the full shape of an object from symmetry [4, 3]. As already mentioned, this approach is the most similar to the one presented in this paper. Thrun and Wegbreit proposed a hierarchy of symmetries that can be detected from a partial view and, subsequently, used to complete the point cloud [4]. They begin by performing a grid search over the entire object for valid local symmetries, followed by a local optimization of the symmetry parameters using the hill climbing algorithm. The resulting candidate symmetries are evaluated using a scoring system based on the probability of observing the completed point cloud. The symmetry-based approach to point cloud completion was extended to robot manipulation by Bohg et al. [3].

B. Extrusion-based Object Representations

The goal of the work presented in this paper is to detect extruded shapes of objects from a single perspective. In this manner, the robot can attempt to complete the shapes of the objects in occluded region, such that the model can be used for manipulating the object.

An extruded shape consists of two components: the profile and the path. The profile is the basic 2D shape that the 3D extruded shape is based on. The path is the line indicating how the profile is extended into the third dimension. In this paper, we focus on paths defined by straight line segments (linear extrusions) and circles (rotational extrusions). This family of shapes allows us to represent a wide range of different primitive shapes, including spheres, rectangular prisms, cylinders, and cones.

However, the robot will also encounter more complex extruded shapes. Hence, we need the representation to be flexible enough to capture these shapes in detail. We achieve this goal by representing the profile of the object as a 2D point cloud. This low-dimensional point cloud can be used to represent a wide range of shapes. The profile can, thus, also be directly obtained from the observed 3D point cloud.

The path of a linear extrusion is defined by a 3D coordinate system and the length of the extrusion. The direction of the extrusion is always in the coordinate frame's z-direction, and the 2D profile defines the shape in the x-y plane. For a rotational extrusion, we define the axis of rotation as a 3D line. The 2D profile points define the location along the axis of rotation, as well as the radial distance from this axis.

C. Detecting Planes of Symmetry

One important characteristic of both linear and rotational extrusions is that they result in symmetric shapes; i.e. the extrusions entail a mirror symmetry. Therefore, to find extruded shapes in the point cloud, we begin by first searching for planar reflection symmetries. Instead of using a grid search to detect these symmetries [4, 3], we adopt the fast voting-based approach proposed by Mitra et al. [12].

First, the normal vector and curvature are computed for each point in the point cloud. The normal vector can be obtained by computing the eigenvectors for a local neighborhood of points. The normal direction is given by the eigenvector with the smallest eigenvalue, which points towards the camera. The eigenvectors can then be computed for the local neighborhood of normal directions. The resulting two largest eigenvalues are used to approximate the local curvature.

Subsequently, each point is compared with every other point in the cloud in order to find pair-wise symmetries. The plane of symmetry for two points $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ is located at the midpoint $0.5(\boldsymbol{x}_a + \boldsymbol{x}_b)$, with a normal aligned with the direction $\boldsymbol{x}_b - \boldsymbol{x}_a$. However, this pair-wise symmetry is only considered valid if the points' normals are also reflected in this plane, and the difference in curvature values between the two points is below a given threshold.

The parameters of each valid plane of symmetry are treated as one vote. The goal is therefore to find parameter settings with many votes, which correspond to large symmetric regions in the point clouds. In order to find these planes of symmetries, the distribution of symmetry plane parameters is modelled as a kernel density estimate [12]. All of the modes of this distribution can then be found using mean-shift clustering [13]. The corresponding symmetry parameters form the basis for local searches for extruded shapes.

D. Computing Initial Extrusion Parameters using ICP

Given a plane of symmetry, a set of extrusion parameters needs to be computed. The steps used to perform this computation are outlined in Fig. 2.

We begin by dividing the point cloud into two parts according to the plane of symmetry. To detect extrusions, we divide the points according to which side of the plane of symmetry the point lies on. To detect rotations, we separate the points according to their normals $\boldsymbol{n}$ and the symmetry plane's normal $\boldsymbol{p}$. If the inner product $\boldsymbol{p}^T\boldsymbol{n}$ is greater than zero, the point is assigned to the first region and otherwise it is assigned to the second region. The dividing of the point cloud into two regions is illustrated in Fig. 2C.

Once the point cloud has been divided into two parts, we need to find a rigid-body transformation that shifts one of the parts to the pose of the other. We use the iterative closest point (ICP) algorithm to compute this transformation [14]. The computed transformation should have the same effect as sliding the part along the extrusion's path. Hence, a linear extrusion should result in a translation, and a rotational extrusion should result in a rotation about an axis (see Fig. 2D). Given the transformation computed using ICP, we can compute the direction of the path for the linear extrusions, and the axis of rotation for rotational extrusions.

It should be noted that the symmetry-detection algorithm proposed by Mitra et al. [12] also uses ICP to detect symmetries more accurately. However, their approach searches for symmetries, and then uses ICP to find the same type of symmetry more accurately. Instead, our approach first detects planar reflection symmetries, and then uses ICP to find extrusions, which are a different type of pattern. In this manner, we exploit the self-similarity property of extruded shapes.

E. Local Search

Given an initial hypothesis for an extrusion, the parameter and profile can usually be further improved using a local search. The 2D point cloud for representing the extrusion's profile also needs to be determined at this stage.

For linear extrusions, the profile will be a plane of points that is orthogonal to the path of the extrusion. Hence, we must align the direction of the path $\boldsymbol{d}$ with the normal of the profile plane. We perform this local alignment using an iterative procedure. We first define the set of profile points as those points that have a normal $\boldsymbol{n}$ such that $-\boldsymbol{d}^T\boldsymbol{n} > \tau$, where $1 > \tau > 0$ is a threshold value. Given this set of profile points, the direction of the path is updated as the negative mean of the profile points' normals. The set of profile points can then again be updated according to the update path direction. In order to improve the robustness of this process, we begin with a low threshold value, e.g., $\tau = 0.5$, and increase the value in each iteration up to a maximum value, e.g. $\tau = 0.95$. After the path direction has been aligned with the normal of one of the object's sides, we need to fit a plane to this side. We first find the location of this plane along the path direction by computing the position of the current profile points in this direction and selecting the mode. The points that are near to this plane are then used to define the final set of profile points. The direction of the extrusion path is given by the normal of this plane. The length of the extrusion path is set according to the length of the surfaces that are orthogonal to the direction of the extrusion.

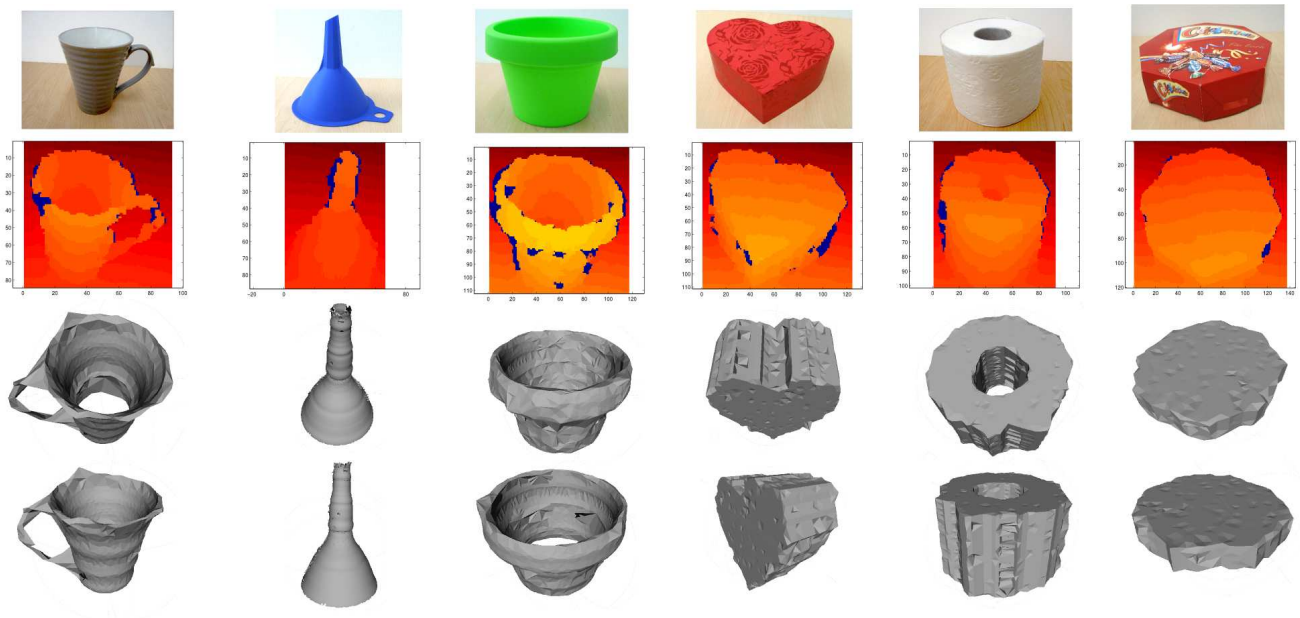

Figure 3. The columns correspond to the results for a cup, a funnel, a pot, a heart-shaped box, a roll of toilet paper, and a box respectively. The first row shows a picture of each object. The second row shows the depth image taken of the object, which corresponds to the partial point cloud. Darker red regions are further away than yellow regions, and blue regions have a depth of zero. The bottom two rows show the reconstructed object shape from different angles. These reconstructions were made from one perspective of the object and the 3D meshes were not post-processed.

For rotational extrusions, we want to set the rotational axis such that many of the profile points are mapped onto each other. We begin by projecting the data into the 2D profile space according to the initial hypothesis, and marking points in dense regions as profile points. For each of the m profile points, located at 3D points $\boldsymbol{x}_{1:m}$, we compute the position along the rotation axis $a_{1:n}$ and the distance from the rotation axis $r_{1:m}$ to the point. We also compute the normalized direction $\boldsymbol{d}_{1:m}$ from each point to the closest point on the axis. For the i^{th} data point, we now compute the locally weighted mean radius $\hat{r}_i = \sum_{j=1}^{m} w_{ij} r_j / \sum_{k=1}^{m} w_{ik}$ where the weight w_{ij} is given by $w_{ij} = \exp(-(a_i - a_j)^2 / v^2)$ and v is a length scale parameter. This radius value $\hat{r}_i$ is an approximation of the desired radius at this point along the axis. Using this desired radius, we create a new 3D point $\boldsymbol{y}_i = \boldsymbol{x}_i + \hat{r}_i \boldsymbol{d}_i$. Once all $\boldsymbol{y}_{1:n}$ have been computed, we fit a line to these points, which then becomes the new axis of rotation. Note that the point $\boldsymbol{y}_i$ will be closer to $\boldsymbol{x}_i$ than the axis of rotation if $r_i > \hat{r}_i$ and, thus, will draw the axis of rotation closer to $\boldsymbol{x}_i$. Similar to the linear extrusion case, we iterate over these steps multiple times to acquire a suitable axis of rotation. The final profile for the rotational extrusion can be smoothed by using local averaging over the radii component.

F. Visibility Score

In the final stage of the extrusion detection process, we assign each candidate extrusion a score in order to select the one that best represents the robot's observations. Similar to the scoring systems used in symmetry-based approaches [4, 3], the score is defined according to how well the completed point cloud matches the observed scene. Our basic scoring system is based on the depth images obtained from the camera, such as the one shown in Fig. 3. This data structure is similar to a z-buffer in computer graphics, and indicates which regions in space are occluded. Using the 3D positions of the observed points, and their pixel location in the z-buffer, we can compute a projection matrix $\boldsymbol{P}$ between the 3D camera space and the 2D z-buffer pixel space.

The partial point cloud is first completed according to the extrusion parameters currently being evaluated. In order to compare different extrusions in a fair manner, each point in the profile should be used to generate the same number of extruded points. The completed point cloud is then projected into the z-buffer space using the projection matrix $\boldsymbol{P}$. Each projected point is assigned to the nearest pixel in the z-buffer.

A projected point is assigned a score according to its z value in the camera coordinate frame z_c and the depth value of the assigned z-buffer pixel z_p. We also define a length scale parameter h. If the depth values are close together $\|z_p - z_c\| < 2h$, then the point provides evidence for the extrusion and, hence, is assigned a positive score of $\exp(-0.5(z_p - z_c)^2/h^2)$. If the point has a depth greater than the z-buffer $z_p - z_c > 2h$, its location corresponds to an occluded region and, hence, it is assigned a score of zero. Finally, if the point has a depth that is less than the z-buffer $z_p - z_c < -2h$, then it contradicts the observed scene and, hence, it is assigned a negative score, e.g. -3. The score for the entire point cloud is given by the sum of the scores obtained by the individual points, divided by the number of points in the profile.

The extrusion with the largest score is used to complete

the partial point cloud, which can then be used to create a 3D model for manipulating the object.

III. Experiment

The proposed method was implemented and applied to a set of common household objects. The results show that the method can detect the extruded shapes in the point clouds, and even capture details of the objects' shapes.

A. Setup and Results

In this experiment, we evaluated the accuracy of the extrusion paths found by the proposed method. In particular, we measured the errors in the radii of rotational extrusions and the path lengths of linear extrusions.

Using a standard Kinect camera, we collected 30 partial point clouds of common household objects. The objects were placed individually on a table, and the table was segmented out of the point cloud. The segmented point cloud was then subsampled to obtain around 3000 points. The approach presented in Section II was then used to complete each of the partial point clouds. The desired type of extrusion was pre-specified for each object.

The actual lengths and radii of the extrusions were also measured manually using the point cloud. By measuring these distances directly from the point clouds, camera-specific calibration errors do not affect our results. The measured distances were then compared to those found by the point cloud completion method.

The robot found suitable extrusions for 28 of the 30 images (93% success rate). In one of the failed trials, the robot extruded an incorrect surface. In the other failed trial, the robot did not find a suitable axis of rotation for describing the shape of the object. In the successful trials, the error in the computed distances could be measured. The distribution over these errors is shown in Fig. 4. The mode and mean of the distribution are at -1.36 mm and -1.19 mm respectively. The standard deviation of the distribution is 2.23 mm. Detecting the planar symmetries for each object in the experiments took on average 19.9 seconds.

Fig. 3 shows a set of models obtained using the proposed method. For the rotationally extruded objects, a stochastic optimization was used to improve the alignment of the axis of rotation and increase the visibility score. The 3D models were generated from the completed point clouds using the ball-pivot algorthim [15]. Some of the objects may seem shorter than the actual object, which is a result of the table segmentation. Standard post-processing methods, such as smoothing, were *not* applied to the point clouds, in order to display the quality of the profiles more clearly. For a real application, we recommend post-processing the 3D model.

B. Discussion

The results show that the extrusion-based approach could accurately complete the objects' shapes in most of the trials. In practice, the accuracy of the model would decrease due to other sources of error, such as the camera calibration.

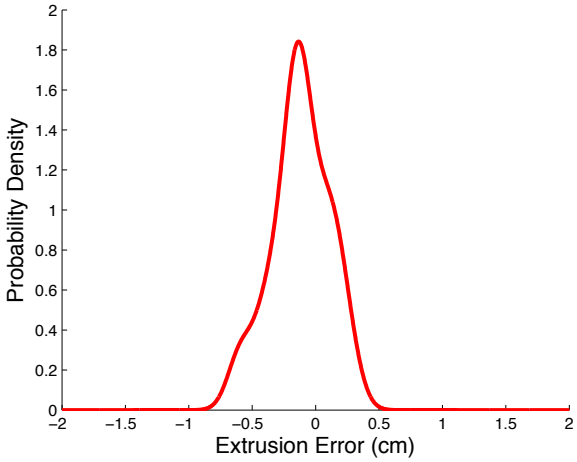

Figure 4. The distribution over errors in extrusion lengths and radii. The distribution was modelled using a kernel density estimate with a Gaussian kernel with a width of $\sigma = 0.1$cm. A negative value indicates that the extruded shape found by the proposed method was smaller than the actual size.

However, the computed models should still be suffficiently accurate for performing coarse manipulations with the objects.

The results also show that the voting-based symmetry detection method performs well even when applied to noisy data. The detected planes of symmetry allowed the robot to find valid extrusions in most of the objects used in this experiment.

The computed extrusions are slightly biased towards being too small. This may be a result of the relatively large penalization for overestimating the size of the extrusion. However, a slight bias is also to be expected for some objects, such as the cup. The point cloud of the cup contains points from both the outside and the inside of the cup. Hence, a rotational extrusion that maps the front-outer points onto the back-inner points would achieve a higher visibility score, but would also result in a smaller cup radius than the actual radius.

The models in Fig. 3 show the importance of using a flexible profile representation. The extrusions are capable of modelling details, such as the lip of the pot and the hole in the middle of the toilet paper roll. The quality of the profiles could be improved by reincorporating more points from the original point cloud, once a valid set of extrusion parameters has been found. Fine details, such as the texture of the cup, are obviously lost due to noise in the data.

One shortcoming of the current method is that it can sometimes detect degenerate extrusions when a linear extrusion is applied to a rotationally extruded object, or vice versa. For example a rotational extrusion applied to a box may detect a cylinder that fits into the shape of the box. We plan to address this problem in the future in order to render the method more robust.

The method is also able to cope with additional parts of

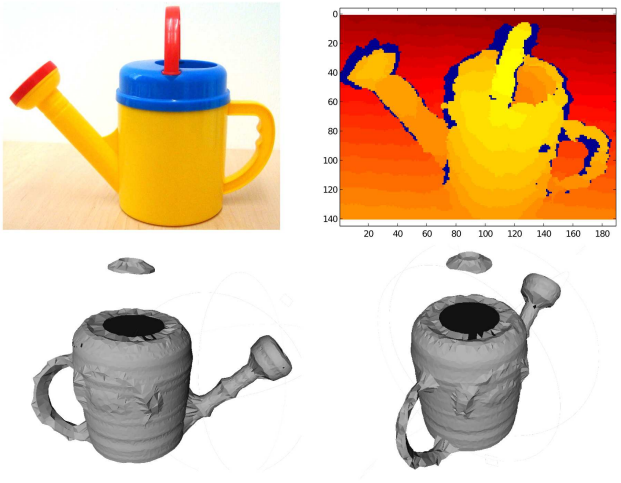

Figure 5. The top left image show the watering can. The top right image shows the depth image taken of the watering can. Darker red regions are further away than yellow regions, and blue regions have a depth of zero. The bottom images show the results of applying the proposed point cloud completion approach to the watering can's partial point cloud. The watering can is an example of an object that consists of multiple extruded parts.

the object that are not extruded, as shown by the handle of the cup. In the future, we will investigate how a robot can robustly decompose more complex objects into multiple extruded parts, using the method proposed by Mitra et al. [12]. An early result of this approach applied to a watering can is shown in Fig. 5. Although the system failed to complete the top handle, and incorrectly completed the side handle, these initial results are promising.

Overall, the experiment has demonstrated that the proposed method could detect most of the extruded shapes and, thus, accurately complete the point clouds.

IV. Conclusion

In this paper, we investigated how point clouds of basic objects can be represented and completed by linear and rotational extrusions. These extrusions are represented in a flexible manner, which allows them to accurately model a wide range of shapes. By detecting local symmetries in partial point clouds, we can search for extrusions in an efficient manner, and use these extrusions to complete the point cloud. In the experiment, the proposed method was applied to point clouds obtained from real household objects, and successfully completed most of the partial point clouds. In the future, we plan to use the proposed method in order to plan grasps on novel objects. In particular, the proposed method allows us to compute contact points for occluded regions of the object.

Acknowledgements

The project receives funding from the European Community's Seventh Framework Programme under grant agreement n° ICT- 248273 GeRT.

References

[1] M. Krainin, P. Henry, X. Ren, and D. Fox, "Manipulator and object tracking for in-hand 3d object modeling.," *I. J. Robotic Res.*, vol. 30, no. 11, pp. 1311–1327, 2011.

[2] T. P. Breckon and R. B. Fisher, "Amodal volume completion: 3d visual completion," *Comput. Vis. Image Underst.*, vol. 99, pp. 499–526, Sept. 2005.

[3] J. Bohg, M. Johnson-Roberson, B. León, J. Felip, X. Gratal, N. Bergström, D. Kragic, and A. Morales, "Mind the Gap - Robotic Grasping under Incomplete Observation," in *ICRA 2011*, May 2011.

[4] S. Thrun and B. Wegbreit, "Shape from symmetry," in *ICCV 2005*, pp. 1824–1831, IEEE, 2005.

[5] S. Savarese and F.-F. Li, "3d generic object categorization, localization and pose estimation," in *ICCV 2007*, pp. 1–8, 2007.

[6] R. Detry, N. Pugeault, and J. Piater, "A probabilistic framework for 3D visual object representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 10, pp. 1790–1803, 2009.

[7] A. Aldoma, N. Blodow, D. Gossow, S. Gedikli, R. Rusu, M. Vincze, and G. Bradski, "Cad-model recognition and 6 dof pose," in *ICCV 2011, 3D Representation and Recognition (3dRR11)*, 11/2011 2011.

[8] G. Biegelbauer and M. Vincze, "Efficient 3d object detection by fitting superquadrics to range image data for robot's object manipulation," in *ICRA 2007*, pp. 1086 –1091, 2007.

[9] M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. Guibas, "Example-based 3d scan completion," in *Symposium on Geometry Processing*, pp. 23–32, 2005.

[10] Z. C. Marton, L. C. Goron, R. B. Rusu, and M. Beetz, "Reconstruction and Verification of 3D Object Models for Grasping," in *ISRR 2009*, (Lucerne, Switzerland), 2009.

[11] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, "Grasp planning via decomposition trees," in *ICRA'07*, pp. 4679–4684, 2007.

[12] N. J. Mitra, L. Guibas, and M. Pauly, "Partial and approximate symmetry detection for 3d geometry," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 25, no. 3, pp. 560–568, 2006.

[13] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603 –619, may 2002.

[14] Y. Chen and G. Medioni, "Object modeling by registration of multiple range images," in *ICRA 1991*, pp. 2724–2729, 1991.

[15] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, and S. Member, "The ball-pivoting algorithm for surface reconstruction," *IEEE Trans. Visualization and Computer Graphics*, vol. 5, pp. 349–359, 1999.