Multi-scale Superquadric Fitting for Efficient Shape and Pose Recovery of Unknown Objects

Kester Duncan, Sudeep Sarkar, Redwan Alqasemi, and Rajiv Dubey

Abstract— **Rapidly acquiring the shape and pose information of unknown objects is an essential characteristic of modern robotic systems in order to perform efficient manipulation tasks. In this work, we present a framework for 3D geometric shape recovery and pose estimation from unorganized point cloud data. We propose a low latency multi-scale voxelization strategy that rapidly fits superquadrics to single view 3D point clouds. As a result, we are able to quickly and accurately estimate the shape and pose parameters of relevant objects in a scene. We evaluate our approach on two datasets of common household objects collected using Microsoft's Kinect sensor. We also compare our work to the state of the art and achieve comparable results in less computational time. Our experimental results demonstrate the efficacy of our approach.**

I. Introduction

For autonomous or semi-autonomous robotic systems to effectively interact with and manipulate objects in their surroundings, accurate and robust perception is necessary. In order to correctly manipulate objects in a scene, accurate position and orientation information is required. To this end, object pose estimation and shape recovery have been well addressed over the years in the computer vision and robotics literatures. In this paper, we consider the problem of rapidly estimating the 3D shape and pose characteristics of unknown objects from single views for robotic manipulation. Having knowledge of these object properties is of paramount importance for grasp planning and manipulation maneuvers across multiple domains including but not limited to assistive devices, rehabilitation, and industrial automation [1].

The task of determining the pose of unknown objects from single views was commonly addressed by finding correspondences between 2D image features and model features from a database of known objects. The next step was to estimate the model pose that best agreed with that set of correspondences. This has been the de facto standard for years. For instance, Collet et al. presented an online method for recognizing objects and their poses from single views using a combination of RANSAC and Mean Shift clustering [2] of keypoints. They extended this work in [3] and [4] to utilize multiple object views in an effort to achieve scalability and low latency. Unfortunately, this method relies on 2D information and an offline modeling stage that learns metric 3D models from multiple object views. Similarly, Sun et al. outlined an approach to jointly detect objects, estimate their pose, and recover their 3D shape information from a single 2D image using a generalized Hough voting-based scheme [5]. However, this work is also dependent on 2D data and a training phase. In our work, there is no training involved and we utilize single view 3D data of unknown objects. Over the last few years, there has been a great resurgence of works handling 3D data. For instance, using point cloud data, Ye et al. estimate human body poses from single 3D views [6].

With compact, low-cost 3D sensors such as the Microsoft Kinect [7] becoming more available, it is more appropriate and more accurate to utilize 3D data because it provides important geometrical information. Knowing the geometry of an object is important for pose estimation. However, using 3D data brings its challenges. Massive amounts of 3D data must be processed rapidly in order for robotic systems to be responsive. Autonomous robotic systems equipped with 3D sensors can acquire point cloud data at an increasingly high rate. Executing common tasks such as scene segmentation and 3D reconstruction on massive amounts of 3D data is computationally expensive and requires a lot of computational time, which is unacceptable for real-time or near real-time robotics [8].

In this paper, we address the issue of finding the shape and pose information of unknown objects in a rapid manner. This is done from single view point clouds where only the front part of the object is visible and assumptions must be made about the back side in order to correctly manipulate the object [9]. We attempt to handle this issue by employing superquadrics, which are compact parametric shapes with tri-axis symmetry that are appropriate for modeling frequently encountered objects in domestic settings.

Over the years, different models have been introduced for 3D shape recovery such as spherical harmonics and geometric icons, but superquadrics are conceivably the most appropriate for our tasks [10]. Their compact shape can be described with a small set of parameters thereby facilitating the description of a wide variety of different basic shapes such as spheres, cylinders, and cuboids. Superquadrics have been used for object approximation [11], [12], object detection [10], novelty detection [13], object segmentation [14], [15], [16], and collision detection [17]. Our goal is to quickly find the superquadric that best fits an unorganized point cloud representing an object hypothesis. This is a major issue inherent with employing superquadrics. Their parameters must be minimized in a least-squares fashion and this process can be computationally expensive if point clouds are large. We address this by proposing a multi-scale voxelization strategy. With this strategy, we are able to estimate the pose of an

K. Duncan and S. Sarkar are with the Computer Science and Engineering Department, University of South Florida, Tampa, FL 33620 U.S.A. {kkduncan, sarkar}@cse.usf.edu

R. Alqasemi, and R. Dubey are with the Mechanical Engineering Department, University of South Florida, Tampa, FL 33620 U.S.A. {alqasemi, dubey}@usf.edu

object using superquadrics in a robust and computationally-efficient manner without sacrificing accuracy. We provide adequate results to support our claims.

This paper is organized as follows. In Section II, we present a short introduction to superquadrics, followed by the description of our multi-scale voxelization approach for superquadric fitting in Section III. Section IV shows our experimental results and evaluations and Section V gives our conclusion.

II. Superquadrics

Superquadrics are a family of parametric shapes that include superellipsoids, supertoroids, and superhyperboloids with one and two parts. They are appealing for robotic applications by nature of their definition. In this work, we focus on the superellipsoid which is useful for a volumetric part-based description. Given the parameters that define a superquadric, the shape and pose information can be easily extracted as well as volumes and moments of inertia. They are compact in shape and have a closed surface. Moreover, superquadrics exhibit tri-axis symmetry, which is a characteristic well approximated by many household objects [9].

Superquadrics can be defined in an object centered coordinate system with five variables and in a general coordinate system by eleven independent variables. The implicit form of the superquadric equation used for optimization in this work is given by

$$F(x_w, y_w, z_w) = \Bigg[\left(\frac{n_x x_w + n_y y_w + n_z z_w - p_x n_x - p_y n_y - p_z n_z}{a_1}\right)^{\frac{2}{\epsilon_2}} + \left(\frac{o_x x_w + o_y y_w + o_z z_w - p_x o_x - p_y o_y - p_z o_z}{a_2}\right)^{\frac{2}{\epsilon_2}}\Bigg]^{\frac{\epsilon_2}{\epsilon_1}} + \left(\frac{a_x x_w + a_y y_w + a_z z_w - p_x a_x - p_y a_y - p_z a_z}{a_3}\right)^{\frac{2}{\epsilon_1}} \quad (1)$$

where the variables (a_1, a_2, a_3) are the scaling dimensions along the x, y, and z axes of the superquadric, (ϵ_1, ϵ_2) are the factors which determine the superquadric's shape ranging from from 0.1 to 1, and $(n_x, n_y, n_z, o_x, o_y, o_z, a_x, a_y, a_z, p_x, p_y, p_z)$ are the twelve parameters of the homogeneous transformation matrix that is a result of a rotation and translation of the world coordinate frame. Equation 1 is commonly referred to as the inside-outside function F. For a given point in the world coordinate system $\mathbf{x}$, $F(\mathbf{x}) = 1$ if the point is on the superquadric surface, $F(\mathbf{x}) < 1$ if the point is inside the superquadric, and $F(\mathbf{x}) > 1$ if the point is outside the superquadric. Therefore, the eleven variables that define a superquadric in general position and orientation are $\Lambda = \{a_1, a_2, a_3, \epsilon_1, \epsilon_2, \phi, \theta, \psi, p_x, p_y, p_z\}$. To handle global superquadric deformations, the tapering parameters k_x and k_y are used but we do not exploit this feature in this work.

The Levenberg-Marquardt algorithm [18] is used to recover the parameter set Λ that best fits a given set of points $\mathbf{x_k}$ in a least-squares minimization. The following expression must be minimized:

$$\min_k \sum_{k=0}^{n} (\sqrt{a_1 a_2 a_3}(F^{\epsilon_1}(\mathbf{x_k}; \Lambda) - 1))^2 \quad (2)$$

where the multiplier $\sqrt{a_1 a_2 a_3}$ enforces the recovery of the smallest superquadric and the exponent ϵ_1 promotes faster convergence as it makes the error metric independent of the shape factor [11]. An important aspect to this minimization is the initial parameter set used. A good initialization is crucial to the success of the superquadric fitting process. Therefore, we use the initial pose given via the eigenvalue decomposition of the point cloud. The initial shape used is an ellipsoid and the superquadric scale factors are based on the dimensions of the cloud itself.

III. Multi-scale Voxelization for Superquadric Fitting

Autonomous robotic systems equipped with 3D sensors can acquire point cloud data at an increasingly high rate. Performing common tasks such as scene segmentation and 3D reconstruction on massive amounts of 3D data is computationally expensive and requires a lot of computational time for processing, which is unacceptable for real-time or near real-time robotics. For a responsive robotic system, the latency of frequently executed tasks should be low. For example, as the number of points in a point cloud increase, the computational time required for processing greatly increases. For a system requiring user interaction, this is unacceptable.

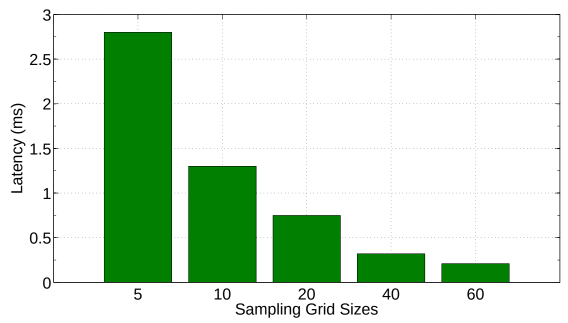

Fig. 1. The computational time required for superquadric fitting using regular sampling grids. Every ith data point is chosen, such as every 5 points or every 20 points and so on.

We address this issue by introducing an automatic coarse-to-fine voxelization scheme for superquadric fitting. Fitting superquadrics to point cloud data is a computationally expensive task and the bottle neck of this process is the iterative Levenberg-Marquardt [18] algorithm minimizing eleven parameters. This is a time consuming algorithm that is heavily dependent on the number of points to be fitted. Simple sampling grids can be used to alleviate this issue, whereby every ith point is chosen, but this may not be sufficiently effective for reducing latency, as shown in Figure 1. Our proposed scheme significantly reduces the size of

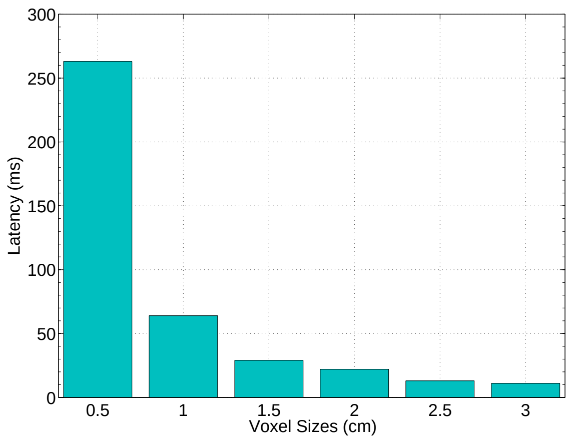

Fig. 2. Execution times for Superquadric fitting in milliseconds for different voxel sizes. The voxel sizes range from 3.0cm. to 0.5 cm. The larger the voxel size, the lower the computational time.

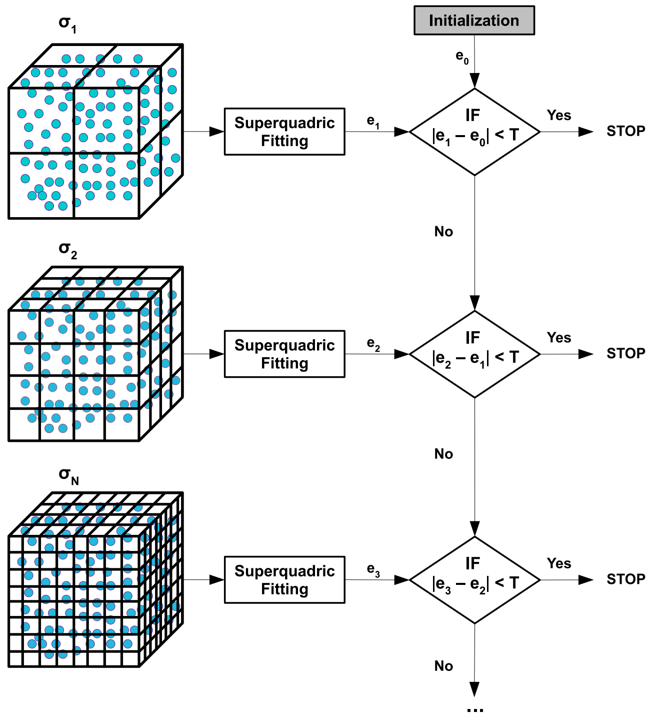

Fig. 3. Overview of Point Cloud Downsampling.

point clouds while maintaining their general shape and space relations. To execute this task, we use voxels which are volumetric pixels that divide the 3D space into uniform 3D cells, typically cubes. In each voxel, all the points present will be approximated or downsampled using their centroid. By doing this, the surface is represented more accurately. Voxels are perfectly justifiable as a means of downsampling data, having been extensively used in the medical imaging and computer graphics communities (see Figure 2).

By the same token, setting the dimensions of a voxel beforehand may not be the best solution to this problem. If the dimensions of the voxel are set too high relative to the point cloud, certain shape details can be lost. On the other hand, if the dimensions are too low, the benefits of downsampling may not be harnessed. To this end, our scheme performs a multi-scale voxelization of the point cloud data so that there is a good balance between speed of computation and accuracy. Figure 3 gives an overview of this process.

For our scheme, there are four main parameters:

- s_{max}, which represents the maximum voxel size,
- s_{min}, representing the minimum voxel size,
- N, denotes the number of scales, and
- τ, which represents the threshold for error change respectively.

The input to our scheme is a point cloud representing a segmented object from a 3D scene. At the first scale σ_1, the cloud is voxelized using a voxel size of s_{max} which significantly reduces its size. A superquadric fitting at this scale is performed. Its fitting error e_1 is compared to that of an initial error e_0 that is computed using unoptimized superquadric parameters acquired via the eigenvalue decomposition of the original point cloud as mentioned in Section II. If the difference between these error values is less than τ, the process stops and the superquadric parameters recovered at this scale are accepted. Likewise, if this difference is greater than τ, we proceed down to the next scale initializing the fit using the acquired superquadric parameters of the previous scale. At this level, the voxel size s_i to be used is determined according to Equation 3.

$$\begin{aligned} s_i &= s_{max} - [(i-1)*\delta], \\ \delta &= \frac{s_{max}-s_{min}}{N}, \\ i &= 1, ..., N \end{aligned} \tag{3}$$

The same process continues until there is no significant change in the error, whereby the process stops or it would proceed until the σ_N scale is encountered.

$$\sum_{k=0}^{n}((F^{\epsilon_1}(\mathbf{x_k};\Lambda)-1))^2 \tag{4}$$

The error metric we use is given by Equation 4 which is similar to Equation 2 without the constraint for the smallest superquadric. If the parameters of this scheme are set appropriately, this may never be necessary. However, including a minimum downsampling stage ensures that some degree of data downsampling is done so that latency is lowered. This scheme can be employed for much more than superquadric fitting by replacing the fitting step, the error metric, and the initialization components.

IV. Experiments and Evaluation

In this section, we present experiments to demonstrate the efficacy of our proposed multi-scale voxelization scheme for fitting superquadrics. The algorithms used in this work were all developed in C++ and the evaluations were performed on a PC equipped with a 2.13GHz Intel Core Duo processor and 4.00GB of memory. All of the point cloud data used in this work were acquired with the Microsoft Kinect sensor[7]. It is important to note that the 3D point clouds are captured from a single view and the segmentation is not perfect. This makes the shape and pose recovery of an object difficult because of the lack of information. Our algorithm recovers the superquadric that best fits the data that it is given, and for the majority of cases it performs well despite noisy or spurious data. Nonetheless, we first outline the datasets used in this work, followed by our evaluations.

A. Datasets

We use two datasets to perform our evaluations; our own dataset comprised of 15 common household objects in different poses which we refer to as *Dataset 1* and Lai's RGBD Dataset [19], [20] which consists of RGB and depth images of 300 common everyday objects taken from multiple view angles organized into 51 categories which we refer to as *Dataset 2*. These objects are primarily cylindrical, spherical, or box-like in shape thereby rendering the superquadric as the ideal parametric model for recovering their shapes because of its tri-axis symmetry characteristic. We provide the principal axis orientation of the objects in our dataset (*Dataset 1* as the ground truth pose information. We believe that this is a more appropriate metric for determining pose when objects are to be handled. For spherical objects where there may be more than principal axis, we choose the one that is closer to the z-axis. Examples of images from both datasets are shown in Figure 4.

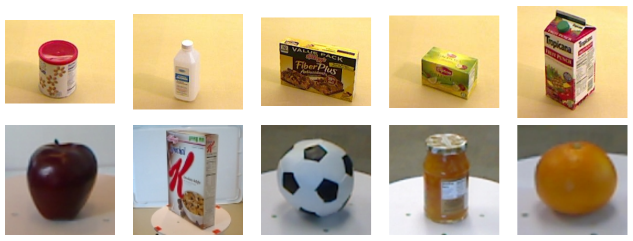

Fig. 4. Examples of objects from our dataset (top row) and the RGBD Dataset [19] (bottom row).

B. Effect of Changing Voxel Size

In this section, we present quantitative results on *Dataset 1* demonstrating the effect of changing the voxel size for point cloud downsampling on pose estimation accuracy. We measure this accuracy by calculating the Euclidean distance between the ground truth location of the centroid of an object and the centroid position values recovered by the superquadric (i.e. p_x, p_y, p_z). We also investigate the accuracy with regards to principal axis estimation. This is measured by calculating the angle between the ground truth principal axis vector and the recovered one. Table I shows the results of this experiment. The first row shows the average distance between the recovered object centroid position and ground truth, the second row shows the median distance, the third row shows the average angle differences between the recovered principal axes and ground truth, and the fourth

TABLE I
THE EFFECT OF VOXEL SIZE ON THE RECOVERED POSE ESTIMATES.

	3.0cm.	**2.5cm.**	**2.0cm.**	**1.5cm.**	**1.0cm.**	**0.5cm.**
Average location distance	3.35cm.	3.36cm.	3.33cm.	3.36cm.	3.31cm.	3.29cm
Median location distance	2.2cm.	2.52cm.	2.7cm.	2.5cm.	2.6cm.	2.24
Average angle difference	31.2°	22.9°	23.1°	23.7°	22.5°	27.2°
Median angle difference	2.1°	3.02°	3.02°	3.80°	3.1°	3.6°

shows the median. Notice that the average location distance is relatively the same across the different scales, which demonstrates that for these voxel sizes, pose estimation is not severely affected. This also tells us that for this dataset, it may not be necessary to use all of the point cloud information to determine the approximate location of an object. Conversely, determining the exact principal axis orientation may require using more data in this case. We report the median angle difference due to the severe effect that a miscalculated orientation value can have on the overall average error. One advantage of our multi-scale voxelization approach is that it can implicitly determine the appropriate downsampling scale for 3D data, hence reducing the adverse effects of having one predetermined value.

C. Pose Estimation Results

In this section, we present the pose estimation results of our proposed approach in comparison to two other methods: the classical superquadric fitting approach introduced by Solina et al. [11], and the approach of Biegelbauer et al. [10]. In the classical algorithm, the full point cloud is used for recovering the parameters to the superquadric model. Therefore, the computational time of this method is directly proportional to the amount of object points provided. Conversely, Biegelbauer and Vincze's method uses a hierarchical RANSAC search and a sorted quality-of-fit criteria to find superquadric models for objects in a scene[10].

TABLE II
POSE ESTIMATION RESULTS OF OUR PROPOSED ALGORITHM, BIEGELBAUER ET AL. (*HS*), AND SOLINA ET AL. (*Classic*) ON *Dataset 1*.

	Proposed	**Solina**	**Biegelbauer**
Median location distance	2.23cm.	2.24cm.	3.85cm.
Median Absolute Deviation (distance)	0.89	0.86	0.461
Median angle difference	2.85°	3.23°	13.46°
Median Absolute Deviation (angles)	1.77°	2.52°	10.52°
Average time (s)	0.04	2.1	7.9

1) Experiments using Dataset 1: Table II displays the results of pose estimation on *Dataset 1* using all three algorithms. We include the median pose accuracies as well as the median absolute deviation because the distribution across objects is irregular. For instance, if the estimated principal axis is incorrect, the angle difference from the ground truth can in some cases be approximately 90° producing massive error. Also, we use the median absolute deviation because it is a robust statistic that is resilient to data irregularities.

Nonetheless, it can be seen that our algorithm is most similar in performance to the classical approach even though the computational time required by our algorithm is significantly lower as a result of the multi-scale downsampling (see Figure 5). Beigelbauer and Vincze's approach [10] produced less accurate but relatively similar results, but at great computational expense, which may be unacceptable for a responsive robotics system.

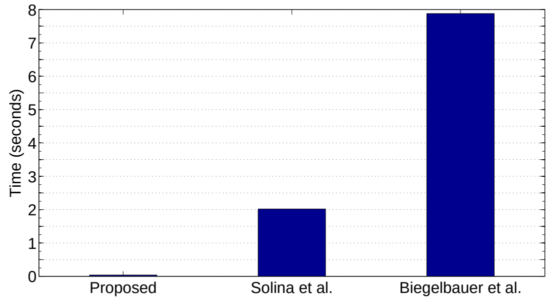

Fig. 5. Average computational times for the three algorithms tested work on *Dataset 1* - Proposed approach: 40ms, Solina et al.: 2.02s, Biegelbauer et al.: 7.88s.

TABLE III

POSE ESTIMATION RESULTS OF OUR PROPOSED ALGORITHM AND THE CLASSIC SUPERQUADRIC FITTING ALGORITHM [11] ON A SUBSET OF *Dataset 2*.

		Standard Deviation			
		x	y	z	**Time (s)**
Ball_1	Proposed	0.29	0.30	0.30	0.09
	Solina et al.	0.33	0.34	0.33	0.23
Ball_2	Proposed	0.29	0.31	0.33	0.09
	Solina et al.	0.26	0.32	0.32	0.26
Cereal_box_2	Proposed	0.33	0.31	0.32	1.1
	Solina et al.	0.26	0.25	0.29	4.1
Food_can_3	Proposed	0.30	0.28	0.28	0.18
	Solina et al.	0.26	0.29	0.29	0.52
Soda_can_3	Proposed	0.29	0.30	0.31	0.18
	Solina et al.	0.21	0.09	0.22	0.61

2) Experiments using Dataset 2: Due to the similarity in performance of our approach and the one of Solina et al. [11], we perform a more detailed analysis of the pose estimation performance of both algorithms. We conducted pose estimation experiments on a subset of *Dataset 2* consisting of over 3500 total image views of 5 different objects. We show the results in Table III. We proceed with this experiment from the standpoint that for different views of the same object, the principal axis orientation should relatively be the same. The RGBD dataset used in this test contains multiple views of the same object as it rotates on a turntable, hence only changing the orientation of the object around the upright axis. Therefore, an appropriate test for pose accuracy in this case is to calculate both the standard deviation σ and median absolute deviation of the principal axis orientation of multiple views of the same object. Ideally, these values should be low, indicating that the pose estimations are less dispersed and mostly like the mean. Our subset consists of the varying views of a ball, cereal box, soda can, and food can. These results confirm that we are not sacrificing accuracy for computational savings. Rather, we obtain comparable accuracy to the state of the art and achieve it in considerably less time. We achieved less speedup with this dataset as compared to *Dataset 1* because the object point clouds were smaller. With larger point clouds, the computational savings become more noticeable and vice versa.

D. Shape fitting estimation

In this section, we report the effect of multi-scale downsampling on the quality of the recovered superquadric model. Table IV demonstrates the shape fitting quality of our approach in comparison to that of Solina et al. [11]. These values were calculated for objects of *Dataset 1* according to Equation 4. They are given in terms of the average fitting error, the average number points per object point cloud, and the average number of points processed per cloud. As can be seen, with downsampling the fitting error is remarkably similar to the value acquired by using the complete object point cloud. Using this dataset, we discovered that only approximately 1% of the point cloud is necessary for determining the relative pose of an object.

TABLE IV

AVERAGE SHAPE FITTING PERFORMANCE

	Error	No. of Points per Cloud	No of Points Processed
Our approach	15.22	15641	110
Solina et al.	14.1	15641	15641

V. CONCLUSION AND FUTURE WORK

In this paper, we have presented an approach for rapidly acquiring the shape and pose information of unknown objects from single view point cloud data. Our proposed approach uses a low latency multi-scale voxelization strategy that is capable of accurately estimating the shape and pose parameters of relevant objects in a scene. A reconstructed 3D model of the object is computed by fitting superquadrics to the data which provides us with the underlying shape and pose. We obtain results comparable to the state of the art and do so in significantly less time. We evaluated our approach on two datasets of common household objects collected using Microsoft's Kinect sensor. Our experimental results demonstrate the efficacy of our approach.

For future work, we intend to use the recovered pose of relevant objects in a scene in a novel robotic grasping system for a wheelchair-mounted robotic arm (see Figure 7). This system is designed to assist the physically challenged in manipulating objects in their living environments without the assistance of other human beings. Additionally, we intend to use non-uniform voxels as well as adjusting our system to estimate the pose and shape of more complex objects. This would enable us to more accurately process the wide variety of shapes present in human environments.

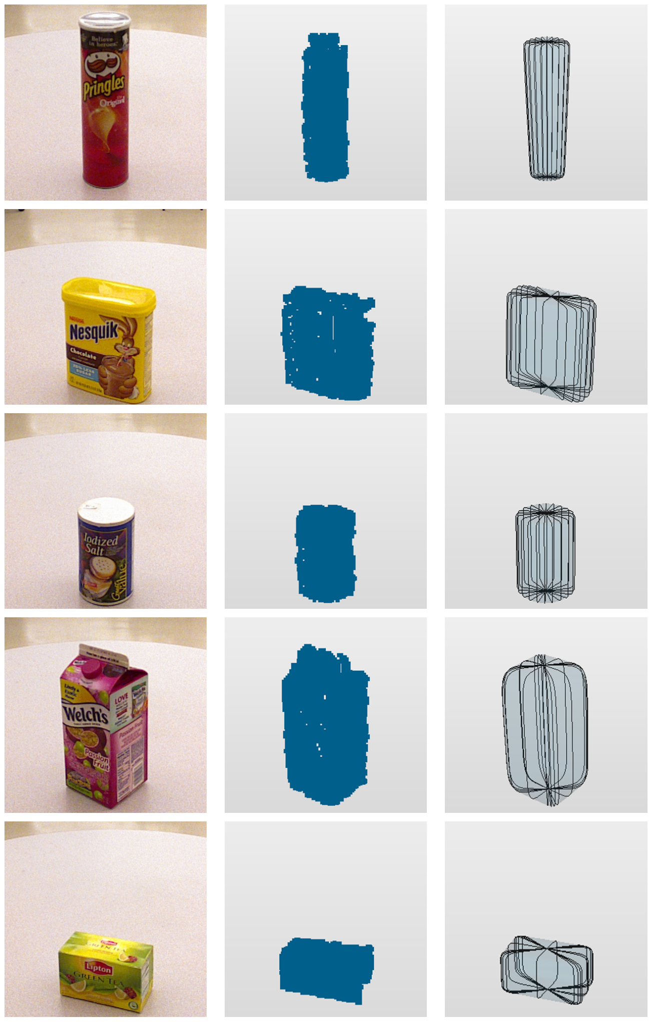

Fig. 6. Visual results of the superquadric fitting. Left column: RGB Image, Middle column: Noisy segmented point cloud, Right Column: Recovered superquadric

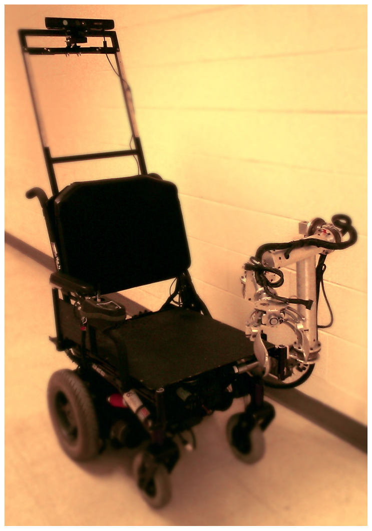

Fig. 7. The Wheelchair-Mounted Robotic Arm system (WMRA) that would be used to assist physically-challenged individuals.

Acknowledgments

We would like to thank Mustafa Mashali, Georg Biegelbauer, and Kevin Lai for their insight and wisdom.

References

[1] S. Colbert, R. Alqasemi, and R. Dubey, "Efficient shape and pose recovery of unknown objects from three camera views," in *2010 7th International Symposium on Mechatronics and its Applications (ISMA)*, April 2010, pp. 1 –6.

[2] A. Collet, D. Berenson, S. S. Srinivasa, and D. Ferguson, "Object recognition and full pose registration from a single image for robotic manipulation," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 48–55.

[3] A. Collet, "Efficient multi-view object recognition and full pose estimation," in *International Conference on Robotics and Automation*, May 2010, pp. 2050–2055.

[4] A. Collet, M. Martinez, and S. S. Srinivasa, "The MOPED framework: Object recognition and pose estimation for manipulation," *The International Journal of Robotics Research*, vol. 30, no. 10, pp. 1284–1306, Apr 2011.

[5] M. Sun, G. Bradski, B. Xu, and S. Savarese, "Depth-encoded hough voting for joint object detection and shape recovery," in *European Conference on Computer Vision*, 2010, pp. 658–671.

[6] M. Ye, R. Yang, and M. Pollefeys, "Accurate 3D pose estimation from a single depth image," in *International Conference on Computer Vision*, Nov 2011, pp. 731–738.

[7] "Microsoft Kinect," http://www.xbox.com/en-us/kinect.

[8] R. Rusu, N. Blodow, and Z. Marton, "Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments," in *International Conference on Intelligent Robots and Systems*, 2009, pp. 3–8.

[9] Z. Marton, D. Pangercic, N. Blodow, J. Kleinehellefort, and M. Beetz, "General 3D modelling of novel objects from a single view," in *International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 3700–3705.

[10] G. Biegelbauer and M. Vincze, "Efficient 3D object detection by fitting superquadrics to range image data for robot's object manipulation," in *International Conference on Robotics and Automation*, no. April, 2007, pp. 10–14.

[11] F. Solina and R. Bajcsy, "Recovery of parametric models from range images: the case for superquadrics with global deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 2, pp. 131–147, Feb 1990.

[12] M. Strand, Z. Xue, M. Zoellner, and R. Dillmann, "Using superquadrics for the approximation of objects and its application to grasping," in *International Conference on Information and Automation (ICIA)*, June 2010, pp. 48–53.

[13] P. Drews, P. Nunez, R. Rocha, M. Campos, and J. Dias, "Novelty detection and 3d shape retrieval using superquadrics and multi-scale sampling for autonomous mobile robots," in *IEEE International Conference on Robotics and Automation*, May 2010, pp. 3635–3640.

[14] A. Leonardis, A. Jaklic, and F. Solina, "Superquadrics for segmenting and modeling range data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 11, pp. 1289–1295, Nov 1997.

[15] D. Katsoulas, C. Bastidas, and D. Kosmopoulos, "Superquadric segmentation in range images via fusion of region and boundary information," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 5, pp. 781 –795, May 2008.

[16] M. Strand and R. Dillmann, "Segmentation and approximation of objects in pointclouds using superquadrics," in *International Conference on Information and Automation*, June 2009, pp. 887–892.

[17] K. Moustakas, D. Tzovaras, and M. Strintzis, "Sq-map: Efficient layered collision detection and haptic rendering," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 1, pp. 80–93, jan.-feb. 2007.

[18] J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," in *Numerical Analysis*, ser. Lecture Notes in Mathematics. Springer Berlin / Heidelberg, 1978, vol. 630, pp. 105–116.

[19] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *International Conference on Robotics and Automation*, May 2011, pp. 1817–1824.

[20] K. Lia, L. Bo, X. Ren, and D. Fox, "A scalable tree-based approach for joint object and pose recognition," in *AAAI Conference on Artificial Intelligence*, 2011.