On quantitative evaluation of vision-based 3D object pose estimation

Chin-Chia Wu[a,#], Nelson Yen-Chung Chang[b], Kai-Chieh Chuang[c], Yu-Chen Cheng[d], and Jwu-Sheng Hu[e]

Industrial Technology Research Institute, Taiwan, ROC

[a]ChinChiaWu@itri.org.tw, [b]NelsonChang@itri.org.tw, [c]kaichiehc0412@gmail.com, [d]itri451627@itri.org.tw, [e]hujwusheng@itri.org.tw

[#]corresponding author

Keywords: Pose estimation, Evaluation, Point cloud, Simulator.

Abstract. In recent years, many approaches have been proposed for estimating 6 degree-of-freedom object pose. Usually, each approach has been tested on a different (small) database of 3D models, and thus direct performance comparison among different methods has not been available. Moreover, since there is no ground truth could be referred, the performance usually lacks quantitative result in clutter scenes. In this paper, we describe the ITRI Pose Estimation Benchmark, a publicly available database for vision based 3D object pose estimation in both single object and clutter scenes. A physics-based simulator is used to acquire point clouds from multi-view object and clutter scenes of each polygonal model. The setting of 3D vision sensor and experiment setup is defined such that comparison of performance can be done based on the same experiment condition. As a result, the proposed data sets can be used as ground truth to evaluate the performance of different pose estimation algorithms quantitatively. Also, a framework that performs the benchmark in terms of pose estimation error is provided. Furthermore, the database allows future comparison with novel algorithms.

Introduction

Vision recognition, localization, measurement, and inspection have been widely applied in fields such as intelligent vehicle, intelligent automation, and intelligent robotics. During the development of a vision algorithm, it is common to first understand the characteristic of a target object and the functional goal in terms of performance specification. Only when such understanding and specification are defined can an efficient and effective vision algorithm be derived. However, vision application nowadays is facing the challenge of parts variety as manufacture trends turn toward low-volume high-variety. Vision algorithms are required to be ever more robust and flexible to accommodate different objects or scenarios. As a result, vision algorithms are changing from application and object specific algorithm to more flexible and generalized algorithms.

Developing generalized vision algorithms are difficult because of different feature and constraint posed by the variety of objects or scenes. In other words, it's always possible to find an object or scene that will fail a vision algorithm. To reduce the failure possibility, developer would try out as many objects or scenes as possible under as many testing conditions and parameter settings to quantitatively evaluate the flexibility and robustness of a vision algorithm. However, many vision algorithm researches, such as image classification or object pose estimation, often used different test data sets for performance evaluation. Since the test data sets are different, it would be difficult to objectively compare the quantitative performance of different vision algorithms.

To remedy this issue of different comparison basis, many academic or research institutes provide various test image databases in hope to provide a general and standardized test data sets. These databases often consist of thousands to millions of images. These images are classified and tagged based on their genre, textures, object position or orientation, and other properties. Some of these databases are designed for specific applications, such as CMU PIE [1] database for facial recognition, FVC2004 [2] database for fingerprint recognition, and BSDS500 [3] database for foreground/background segmentation. Other databases are designed as collections of more

generalized images. These databases are often intended to be used in object detection, classification, searching, or recognition. There are some small databases such as PASCAL [4] and MSRC [5]. There are also large database like ImageNet [6], which is one of the most commonly used image database with a goal of providing challenges for vision algorithms that are found in real life scenarios. By verifying on these common test data sets, an objective and quantitative performance comparison of different vision algorithm can be conducted.

In contrast to 2D image database, the development of 3D object database for vision algorithm performance evaluation had been limited by sensing technology availability and object size scale variety. Therefore the number of common 3D object databases has always been fewer than that of 2D image databases. The test data set in a 3D object database is often consisted of 3D mesh models, CAD models, or 3D point clouds. Similar to 2D image databases, the content of a 3D object database is often categorized and tagged according to predefined properties. An example of early 3D object database is PSB [7], which is mainly consisted of CAD models and designed for performance evaluation of 3D object recognition and searching. Recently, with the mature of commodity 3D sensors, such as Kinect sensor, the barrier for collecting 3D data has been drastically lowered [8]. Some database which combines RGB 2D and depth 3D data are also available for 3D facial recognition [9]. For large scale scenario, BigBIRD [10] database is available for 3D object recognition and classification.

Despite all these existing databases, all of these are designed to verify performance on recognition, classification, segmentation, or searching. These databases were not designed for evaluating the performance of 3D object pose estimation algorithms, which is the core for the challenging robotic random bin picking application. Random bin picking has been a long developing technology that has wide potential applications in industrial automation. Many 3D object pose estimation algorithms [11-14] have been proposed over the past two decades, but each with their own performance evaluation based on different test data sets. Some of these performances are quantitative, some are not. The problem being that it is difficult to acquire absolutely authentic pose of a 3D object in 3D space based on measurement. Measurement often involves measurement error, calibration error, and coordinate frame error. Therefore, this work proposed and 3D object database designed for 3D object pose estimation based on digital simulation approach. The test data includes 3D sensor's depth images of randomly stacked objects and each object's ground truth 3D pose with respect to the 3D sensor. 3D object pose estimation can be tested by estimating the pose of an object in a depth image and compare pose estimation error with the ground truth pose. This will be demonstrated with a 3D pose estimation [14] to demonstrate how objective quantitative pose estimation performance could be done.

3D Stacked Objects Database

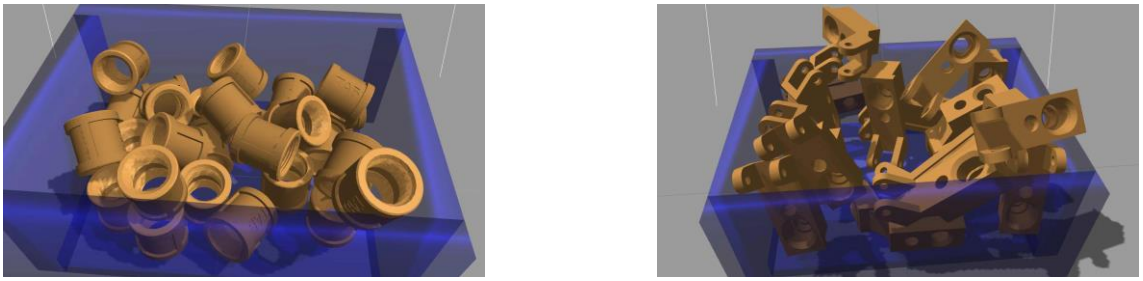
In random bin picking applications, the input of 3D object pose estimation are often represented as depth image or point clouds. These input data of an object's surface geometry are often acquired by stereo vision camera or laser scanner. Therefore, the database was constructed based on how a real 3D sensor works.

GazeboSim. We construct the 3D stacked object database using GazeboSim [15]. GazeboSim is an authentic 3D simulator that supports multiple robots. GazeboSim supports up to four different physics simulation engine. Therefore GazeboSim can be used to simulate how a robot move and collide in indoor or outdoor environment. GazeboSim use OGRE3D [16] to perform 3D rendering. Therefore GazeboSim can be used to simulate basic lighting, texture, shadow effect. It also enables visualization of a simulation process. GazeboSim also provide powerful plug-in interface that allows user to develop custom sensor model. Given these capability, we adopted GazeboSim to simulate objects stacked together, then generate depth images by using a custom modeled virtual 3D sensor.

Randomly Stacked Objects. The most common way to describe a 3D object is by triangular mesh. Triangular mesh describes and object's surface geometry information, and has been commonly used

in various 3D object manipulation and display. There are various sources of triangular mesh models, such as export from CAD modeling softwares or 3D scan reconstruction.

In RBP application, objects are randomly stacked in a box or bin. We constructed a box with suitable size within our simulation environment. Several objects with the same shape and size would be randomly placed above the box initially. Then they are dropped into the box in a free fall condition by simulation. The objects are affected by gravity and collision during their fall. These physical impacts are modeled by the use of Bullet physics engine [17]. As a result, a box with objects randomly stacked would be generated as shown in Fig. 1.



(a) (b)

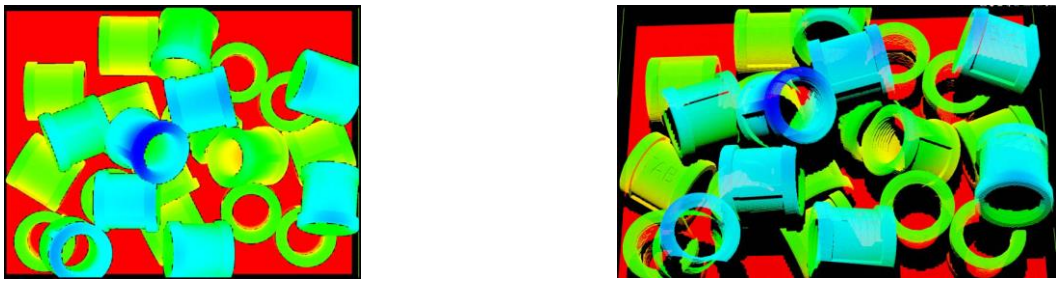
Fig. 1. Objects randomly stacked in a bin after free falling simulation. (a) CYLINDER and (b) BRAZO workpieces.

Virtual 3D camera. Once a scene of randomly stacked objects was available, we took depth image using a virtual 3D camera. The virtual 3D camera was developed using GazeboSim's open sensor interface. We based our virtual 3D camera design on a real ITRI's depth camera specification. Table 1 shows the sensor performance specification. We implemented the virtual 3D sensor by extracting the depth information of each pixel in a depth image. This is done by writing a shader program to extract information from 3D graphics rendering pipeline. After the acquisition of a depth image, a corresponding point cloud can be computed by using the virtual camera's projection parameters.

Table 1. ITRI's depth camera specification.

Sensor	ITRI Depth Sensor
Resolution	1280 x 960 (pixels)
Field of View	240x180 (mm)
Working Distance	800 (mm)
Working Range	±150 (mm)

Similar to how a real depth camera would be used in an RBP scenario. We setup the virtual 3D camera right above the bin with stacked objects. An example of depth images is shown in Fig. 2. The image on the left is the raw 2D depth image and the image on the right is a 3D view of the point cloud.



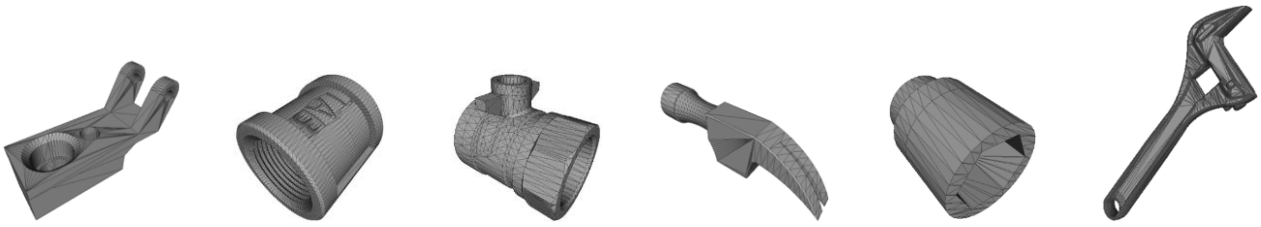
(a) (b)

Fig. 2. An example of a perfect depth image grabbed by the virtual sensor, (a) 2D view and (b) 3D view.

Retrieving Ground-truth Pose. Defining the actual pose of a 3D object in real world is difficult. Although not impossible, it requires to define a principal point of a reference coordinate system first. Then can the pose, which includes both the position and orientation, of an object be defined. To do so, one must rely on accurate and precise measurement tools and measurement steps to accomplish such task. However, the measurement error would arise during measurement process and would thus compromise the accuracy and precision of the measurement result.

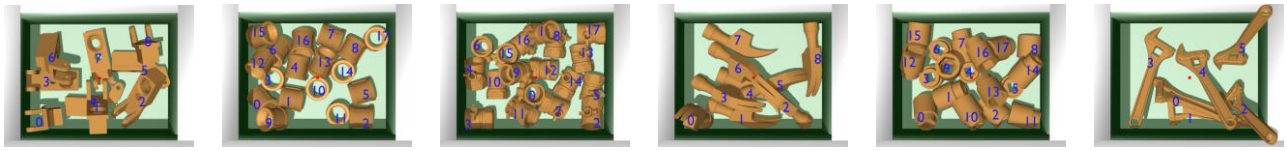
In contrast to real world measurement, the ground-truth pose of an object is obvious. The pose of each object within a simulated scene is recorded by a 3D scene manager. Therefore, the pose of each object and the virtual 3D camera could be extracted. By transforming the pose of each object with respect to world coordinates into pose with respect to virtual 3D camera coordinates. This pose of an object with respect to the virtual 3D camera coordinates is defined as the ground-truth pose of this object. Each object within a box was indexed according to the order of ground-truth pose generation. This allows user to find the corresponding ground-truth pose of an object of interest easily.

Database Structure. The proposed stacked-object database (SOD) consisted of 6 objects of various shape. These objects are named BRAZO, CYLINDER, FINEFOOD, HAMMERHEAD, SOCKET, and WRENCH, and shown in Fig. 3. For each object, 800 randomly stacked scenarios were generated. According to the object size, each scenario consisted of from 6 to18 identical objects. Each scenario was represented using a 3D point cloud with each object indexed and ground-truth recorded. Each scenario corresponded to a test data entry. To help users understand the relation between objects and index numbers, the test data of each scenario included an image take from the virtual 3D camera with the index number marked next to each object as shown in Fig. 4. In other words, each test data entry was consisted of a scenario point cloud file, a 3D ground-truth pose file, and an object index image file.



(a) (b) (c) (d) (e) (f)

Fig. 3. Six objects contained in the SOD are (a) BRAZO, (b) CYLINDER, (c) FINEFOOD, (d) HAMMERHEAD, (e) SOCKET, and (f) WRENCH.



(a) (b) (c) (d) (e) (f)

Fig. 4. Object index images of 6 scenarios with objects (a) BRAZO, (b) CYLINDER, (c) FINEFOOD, (d) HAMMERHEAD, (e) SOCKET, and (f) WRENCH.

Experiment Result

The stacked objects database is used to test a 3D object pose estimation algorithm in [14]. The input to the algorithm under test is point clouds of stacked objects from the database. To evaluate the performance of the 3D object pose estimation only, stacked objects must be segmented into separate independent objects before conducting pose estimation. An ideal segmentation was developed based on known prior information from the scene of a simulation. That is during GazeboSim simulation, the point cloud points of an object were all labeled with the same index number. This can be intuitively done by extracting information from the scene manager. As a result, an ideal segmentation of a point

cloud can be done. By only sending the point cloud points of a single object, we could run the pose estimation algorithm under test to conduct pose estimation on this single object properly.

Pose Estimation Success Rate. We defined pose estimation success rate (PESR) to quantitatively evaluate the performance of a pose estimation algorithm. PESR was defined as the percentage of number of correct pose estimation result over total number of pose estimation trial. A pose estimation result is a pose represented by a position and an orientation angle. A correct pose estimation result was defined as an estimated pose with a pose estimation error, which is with respect to a corresponding ground truth pose, which falls within a predefined correct pose tolerance. A correct pose tolerance was defined as the maximal pose estimation error that can be tolerated to assume the estimation result is correct. Therefore, each PESR is subject to a given correct pose tolerance. Correct pose tolerance should be chosen based on how much pose accuracy is needed for a target pose estimation algorithm. For instance, our evaluation was targeted for industrial bin picking application where pose estimation error could be compensated by a gripper with an opening of 2.5 mm, hence a correct pose tolerance of 2.5 mm position error and 10 degree rotation error was chosen.

Table 2. Pose estimation success rate of each object.

Object	BRAZO	CYLINDER	FINEFOOD	HAMMERHEAD	SOCKET	WRENCH
Success Rate (%)	80.0	82.85	75.25	63.62	87.44	56.21

Table 2 listed the PESR of each object. The number of pose estimation trials must be sufficiently large for PESR to reflect the true performance of a pose estimation algorithm. Fig. 5 depicts the relation between the number of pose estimation trials and PESR. During the first hundred trials, PESRs fluctuated significantly. After more than 600 trials, PESRs started to converge. Take object CYLINDER for example, after 530 trials, the PESR variation dropped within 0.2%. According to this experiment result, we believed that only PESRs with more than 600 trials could truly reflect the performance of the 3D object pose estimation algorithm.

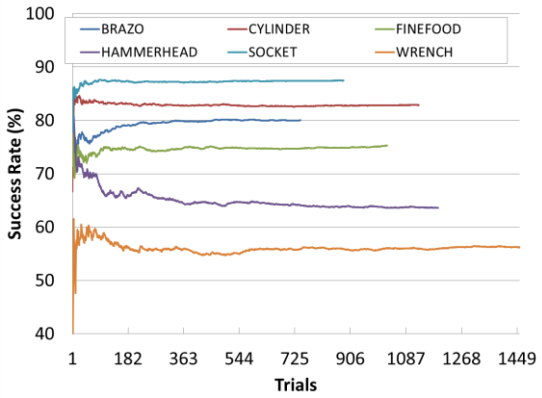
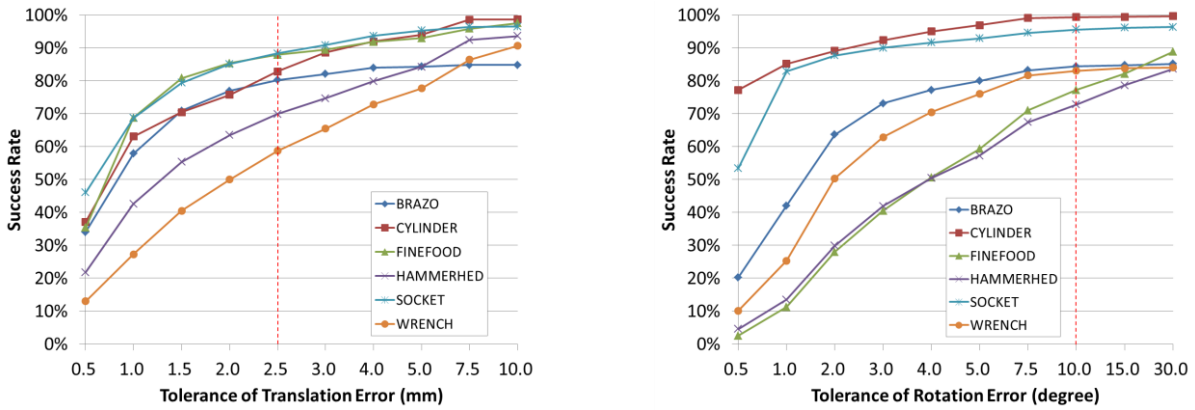


Fig. 5. The relation between the number of pose estimation trials and PESR.

Accumulated Success Rate and Correct Pose Tolerance. Fig. 6 shows the effect of different correct pose tolerance on PESR for each object. Note that a smaller correct pose tolerance may result in lower PESR while larger tolerance may result in higher PESR. From Fig. 6(a), it can be seen that object CYLINDER, FINEFOOD, and SOCKET had similar PESR-position tolerance curve. This may due to the fact the 3 objects were all principally cylindrically shaped. For object WRENCH, it had lower PESR-position tolerance performance compared to others. This may be due to lack of effective feature points on the thin and long handle of the WRENCH object. Similarly, object HAMMERHEAD also had lower PESR-position tolerance performance due to having more flat feature-less surface.

From Fig. 6(b), object CYLINDER and SOCKET had better PESR-orientation tolerance performance. This may be a result of object shape symmetry. In contrast, object BRAZO, WRENCH, HAMMERHEAD were affected by lack of feature points or feature points being unreliable, thus resulted in inferior PESR-orientation tolerance. Object FINEFOOD was the most peculiar one in terms of PESR-orientation tolerance performance. Object FINEFOOD had a cylindrical symmetric shape but resulted in the worst PESR-orientation tolerance performance. After analyzing the pose estimation process for object FINEFOOD, we found that the small bumping part of the object were too small in size for the virtual 3D sensor to properly detect difference on orientation. This may also suggest the object being observed using a sensor with an unsuitable specification.



(a) (b)

Fig. 6. The effect of different correct pose tolerance on PESR for each object. (a) Translation tolerance and (b) rotation tolerance.

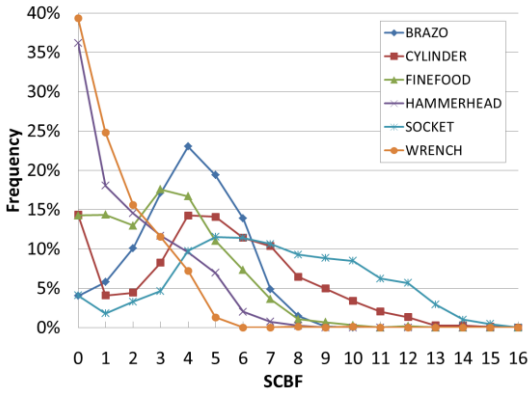


Fig. 7. SCBF of each objects.

Table 3. SCBF average and standard deviation of each object.

Object	BRAZO	CYLINDER	FINEFOOD	HAMMERHEAD	SOCKET	WRENCH
Avg. SCBF	4.5	5.33	3.54	2.25	7.46	1.78
Standard Deviation of SCBF	1.86	3.11	2.15	1.81	3.34	1.40

Successive Success Count. In real industrial RBP application, it would be better to conduct continuous successive success pose estimation. We define an evaluation metric called success count before failure (SCBF) to evaluate the possibility of pose estimation failure. Larger SCBF suggests that fewer pick retry would happen and higher picking throughput can be expected. Smaller SCBF suggests more pick retry would happen and more time will be wasted. Therefore, a good pose estimation algorithm should be expected to yield a large average SCBF.

Fig. 7 shows the frequency-SCBF of each object. This figure shows the distribution of different SCBF value over all the pose estimation trials. The average and standard deviation of each object's SCBF is listed in Table ZZ. Object SOCKET, CYLINDER, and BRAZO had larger average SCBF than others did. Object HAMMERHEAD and WRENCH had smaller average SCBF as expected from previous PESR result and PESR-tolerance analysis.

In summary, we used the test data in the proposed stacked object database that covered 6 different objects to quantitatively evaluate the performance a 3D object pose estimation. The evaluation resulted in an average PESR under a given correct pose tolerance. It also provided us insight on how different correct pose tolerance affected the PESR. Finally, we propose SCBF as a performance evaluation metric that matters to industrial RBP application. Most important of all, this test data enabled us to subjective evaluate the performance of an algorithm as well as to understand the limitation of the algorithm.

Conclusion

A quantitative performance evaluation method for 3D object pose estimation algorithms is proposed. A simulation-based stacked object database was constructed to model scenes of randomly stacked objects in an RBP application. This allowed a ground-truth pose to be easily acquired and enabled comparison of the ground-truth pose and an estimated pose. Average PESR, PESR-pose tolerance, and SCBF metrics were used to evaluate pose-estimation performance quantitatively. Experiment result demonstrate how effective quantitative performance evaluation could be carried out based on the propose method and database.

Future work includes improving simulation authenticity and expanding database. On improving authenticity, incorporating authentic measurement noise model would be necessary. This involves modeling and experimental verification. On expanding database, adding new objects with different shape characteristic would enhance the completeness of the proposed database. A more complete database can be used to further find out the limitation of a pose estimation algorithm. We hope this database can serve as a common test data set for comparing the performance of different pose estimation algorithms. The database can also be used as big-data for learning-based pose estimation algorithm.

Acknowledgment

The authors are grateful to the anonymous referees for their helpful comments and suggestions to improve the presentation of the paper. This work was financially supported by the Department of Industrial Technology, Ministry of Economy, Taiwan.

References

[1] T. Sim, S. Baker, and M. Bsat, "The CMU pose, illumination, and expression database," IEEE Trans. Pattern Anal. Mach. Intell., vol. 25, no. 12, pp. 1615–1618, 2003.

[2] D. Maltoni, D. Maio, A. K. Jain, S. Prabhakar, Handbook of Fingerprint Recognition, 2nd edition, Springer, London, 2009.

[3] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation.," IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 5, pp. 898–916, May 2011.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," Int. J. Comput. Vis., vol. 88, no. 2, pp. 303–338, 2010.

[5] J. Shotton, J. Winn, C. Rother, and A. Criminisi, "TextonBoost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation," in European Conference on Computer Vision (ECCV), 2006.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conf. Comput. Vis. Pattern Recognit., pp. 2–9, 2009.

[7] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, “The Princeton Shape Benchmark,” Proc. of Shape Model Int’l SMI 2004, vol. 08540, pp. 167–178, 2004.
[8] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, “A category-level 3-D object dataset: Putting the Kinect to work,” Proc. IEEE Int. Conf. Comput. Vis., pp. 1168–1174, 2011.
[9] S. Gupta, K. R. Castleman, M. K. Markey, and A. C. Bovik, “Texas 3D Face Recognition Database,” Proc. IEEE Southwest Symp. Image Anal. Interpret., pp. 97–100, 2010.
[10] A. Singh, J. Sha, K. S. Narayan, T. Achim, and P. Abbeel, “BigBIRD: A Large-Scale 3D Database of Object Instances,” 2014 IEEE Int. Conf. Robot. Autom., no. 4, 2014.
[11] S. Savarese and L. Fei-Fei, “3D generic object categorization, localization and pose estimation,” in Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007.
[12] C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam, “Voting-based pose estimation for robotic assembly using a 3D sensor,” 2012 IEEE Int. Conf. Robot. Autom., pp. 1724–1731, May 2012.
[13] M.-Y. Liu, O. Tuzel, A. Veeraraghavan, Y. Taguchi, T. K. Marks, and R. Chellappa, “Fast object localization and pose estimation in heavy clutter for robotic bin picking,” Int. J. Rob. Res., vol. 31, no. 8, pp. 951–973, May 2012.
[14] C.-C. Wu, Y.-F. Hsu, and J.-S. Hu, “Robust Pose Estimation of Randomly Stacked Rigid Objects through Virtual Space Modeling and Simulation,” Automation 2013, The 12th International Conference on Automation Technology, Nov. 2, 2013.
[15] Gazebo. [Online]. Available: http://gazebosim.org/. [Accessed: Apr. 30, 2015].
[16] OGRE–Open Source 3D Graphics Engine. [Online]. Available: http://www.ogre3d.org/. [Accessed: Apr. 30, 2015].
[17] Real-Time Physics Simulation. [Online]. Available: http://bulletphysics.org/wordpress/. [Accessed: Apr. 30, 2015].