2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Crowdsourcing the Construction of a 3D Object Recognition Database for Robotic Grasping

David Kent and Morteza Behrooz and Sonia Chernova
{davidkent, mbehrooz, soniac}@wpi.edu
Worcester Polytechnic Institute, Worcester, MA, USA

Abstract— **Object recognition and manipulation are critical in enabling robots to interact with objects in a household environment. Construction of 3D object recognition databases is time and resource intensive, often requiring specialized equipment, and is therefore difficult to apply to robots in the field. We present a system for constructing object models for 3D object recognition and manipulation made possible by advances in web robotics. The database consists of point clouds generated using a novel iterative point cloud registration algorithm, which includes the potential to encode manipulation data and usability characteristics. We validate the system with a crowdsourcing user study and object recognition system designed to work with our object recognition database.**

I. INTRODUCTION

The abilities to recognize and manipulate a wide variety of objects are critical for robots operating autonomously in household environments. In order to be able to pick up a wide range of objects, the robot must either make use of a database of known objects, or automatically determine useful properties of unknown objects. Automatic detection can lead to difficulties in determining usability characteristics for arbitrary objects, but this type of information can easily be encoded into object recognition databases.

A major disadvantage of object recognition databases is that they require a significant amount of work to construct. Large object recognition databases do exist, but researches are then limited to only the objects included in the database, which in turn limits the environments in which a robot can operate. Adding new objects to a database requires obtaining a detailed model of each new object. The current state of the art is to use a turntable with sensors mounted around it in a specific setup [1], [2]. This is a time consuming approach, and it cannot be used for learning in a real-world environment, where such a detailed setup is impractical. Learning object models based on real-world observations encountered during operation in the target environment is a more challenging problem since the object angles and relationships between multiple views are unknown. Additionally, the challenges of collecting sufficient training data and object labels must also be overcome.

The developing field of web robotics provides new solutions for the collection of training data. Crowdsourcing allows for repetitive tasks, such as collecting the data required to model an object in a recognition database, to be completed with little effort per individual participant. One successful method of taking advantage of the Internet for object recognition is in using large annotated datasets from online image search engines, such as Flickr or Google [3], [4]. We present a different method of incorporating crowdsourcing into an object recognition system, in which participants connect remotely to a robot and label objects while demonstrating grasps through teleoperation. This allows for robot-specific data beyond object labels, such as successful grasp examples, to be collected via the web. With participants demonstrating both object recognition and manipulation techniques, the data can automatically reflect usability characteristics.

This paper makes two contributions. First, we present a web-based system for the collection of point cloud data for a desired set of objects. We designed this system specifically to allow inexperienced participants to remotely collect object data while controlling a sophisticated robotic manipulation platform through a web browser. The system also collects manipulation data for each object, and requires no further equipment than the robot itself.

The second contribution, and central focus of this work, is a system for constructing a functional object recognition system using only point cloud data gathered through crowdsourcing. The system constructs 3D object models given a set of point cloud representations of objects collected from various object views, by determining disparate sets of overlapping point clouds and merging each set of point clouds with common features. We demonstrate that when combined with object labels provided by either a researcher or through crowdsourcing, these 3D object models can be used for object recognition. We present results of using this object recognition system with the object database constructed from a crowdsourcing user study. We also include further analysis using supplemental data collected from a larger and more varied set of objects.

II. RELATED WORK

Crowdsourcing, or obtaining information from large amounts of people through the Internet, is a technique with growing relevance for robotics. In particular, it has the potential to change the way researchers conduct user studies. This idea is central to the development of the Robot Management System (RMS) [5], which provides tools for researchers to create, manage, and deploy user study interfaces that allow direct control of robots through a web browser. RMS allows anyone with an Internet connection

978-1-4799-3684-7/14/$31.00 ©2014 IEEE

to remotely participate in a user study from the comfort of their own home. Along with its ability to reach large groups of people, RMS includes automatic user scheduling, which allows user study sessions to be run with little to no break in between. These properties make RMS an excellent system for implementing a crowdsourcing experiment.

Sorokin et al. conducted previous work using crowdsourcing for novel object grasping [6]. In this work, the researchers divided object modeling into subproblems of object labeling, human-provided object segmentation, and final model verification. Workers on Amazon Mechanical Turk (AMT), a microtask crowdsourcing marketplace, completed these simpler tasks. Sorokin demonstrated successful object recognition and grasping using the results obtained from AMT workers. Microtask crowdsourcing marketplaces do have some limitations, however. The system does not allow for direct control of a robot, and researchers must design tasks to be simple and quick to ensure for high quality data.

An alternative method for crowdsourcing object recognition involves the use of online image search databases. Chatzilari et al. demonstrate that a database consisting of annotated images downloaded from Flickr is sufficient to recognize objects [3]. A downside to this approach is that it is limited to the set of keywords used during download. Generalizing this type of approach to work for any potential object, Torralba et al. constructed a dataset by downloading 760 Gigabytes of annotated images using every non-abstract noun in the English language over the course of 8 months [4]. Because of the limitations that arise from keyword selection, we avoided the use of online databases for the object recognition system presented in this paper. Furthermore, there are no large databases comparable to online image searches that also contain object manipulation data (e.g. grasp points).

There are many alternative methods to creating object recognition databases other than crowdsourcing. Lai et al. provide an example of a common approach to building 3D models [1]. Using a constantly rotating turntable and a set of cameras mounted at multiple angles from the table horizon, accurate 3D object models can be constructed for use in object recognition. Kasper et al. also used a turntable to construct 3D object models for the KIT object models database, although they required a larger and more expensive array of sensors [2]. A disadvantage of these approaches is that they require an accurate setup of sensors and turntables to produce an accurate model, and each new object must be scanned individually using such a process. As with using online databases, these techniques also do not allow for the collection of grasp information.

Computer vision researchers have demonstrated many approaches that construct 3D object models from ordered [7], [8] and unordered [9] sets of 2D images. These approaches are effective for generating 3D models, but similar problems arise as with the turntable method, as they do not allow for the collection of grasping information about the objects.

Willow Garage designed the household objects database [10] to allow researches to add their own objects to the database. This can be a difficult process, however, as they note that there is no cost- or time-effective off-the-shelf tool for creating the required object models. They constructed the original database by defining surfaces of rotation for rotationally symmetric objects, and building object models for non-symmetric models using 3DSOM, which does not handle object concavities. Because of the lack of an effective method for generating object models, adding new objects to the database is difficult.

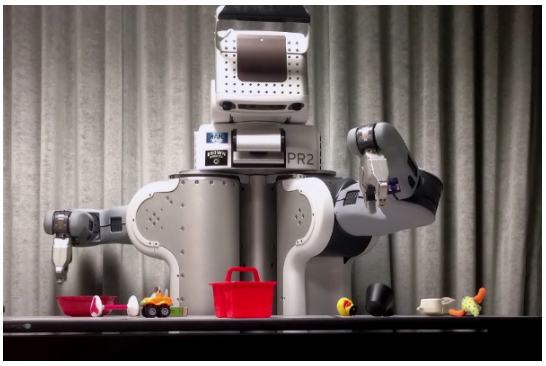

Fig. 1. Physical setup of the user study

III. EXPERIMENT

We collected initial data by conducting a user study. We recruited 42 participants through advertising on the campus of Worcester Polytechnic Institute and in the surrounding area. The study setup consisted of a PR2 robot placed in a fixed position. The robot faced a table containing ten household objects arranged in random positions and orientations, as shown in Figure 1. The table itself consisted of six sections marked with boundary lines. Participants remotely connected to the PR2 through the RMS web interface, which consisted of two main components, shown in Figure 2. In the view component, participants could switch between video feeds from the cameras in each of the PR2's arms, as well as the RGB video stream from a head-mounted Microsoft Kinect sensor. Participants could also control the direction the head was pointing to adjust the video feed view. In the control component, participants used interactive markers on a simulated robot model to change the position and orientation of the physical robot's end effectors.

Participants were instructed to pick up as many objects as possible over the course of a twenty minute session. After grasping an object, the interface asked participants to label the object. The interface then highlighted a randomly selected section of the table surface, where participants were to place the object. In this manner, the user study was automatically resetting, in that each new participant began the study with a different arrangement of objects, determined by how the previous user placed each object. Participants who successfully completed the required actions were rewarded with points and encouraging statements in the interface to maintain user engagement in the study.

Upon successfully picking up an object, the system stored a segmented point cloud of the grasped object using the head-mounted Kinect, the position and orientation of the PR2's

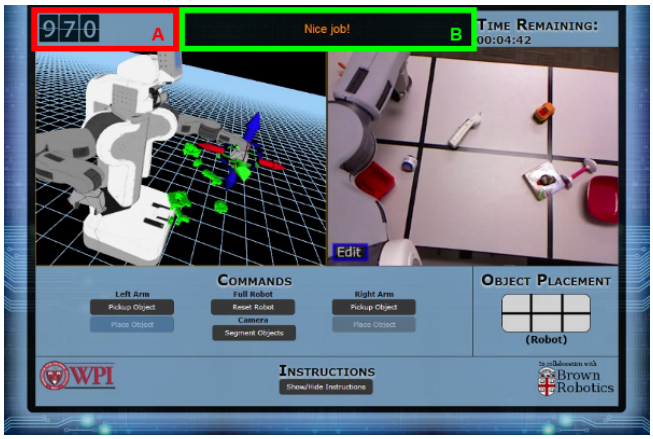

Fig. 2. An example of the web interface used in the remote user study. (A): participant's current score. (B): feedback based on participant's performance.

gripper, force data from the sensor array on each gripper finger, and the object label provided by the participant. The system stored this data in a database for later use in constructing 3D object models. Furthermore, the randomized object placement allowed for the collection of data for multiple common orientations of each object, without the need for a researcher to change the object orientations between each user session. Figure 3 shows an example of point cloud data gathered in the user study, as well as an object model constructed from that data.

Fig. 3. Left: An example of point cloud data gathered upon a successful object pickup. Right: An object model constructed from the database generated by the user study.

IV. OBJECT MODEL CONSTRUCTION

Given the set of individual segmented point clouds of each object collected using the above method, the next step was to create more complete object models from these individual views. First, we used the Point Cloud Library (PCL) [11] to implement a pairwise point cloud registration pipeline. This process allows the merging of any two point clouds into a single point cloud according to correlated SIFT features.

Theoretically, we could construct a complete object model by iteratively performing this pairwise point cloud registration process for all of an object's data. In practice, however, pairing point clouds arbitrarily in this way is highly susceptible to error propagation. Aside from minor errors propagating, a single incorrect merge early on in the process can result in highly inaccurate models. It is therefore critical to distinguish *successful merges* from *failed merges*, where we define a successful merge as one in which all overlapping points common to both point clouds are correctly aligned, so that a human being would verify it as a single object.

We present a novel approach for building object models through iterative point cloud registration. For each pair of point clouds, we first calculate a set of metrics that characterize the potential merge with respect to different properties. Using these metrics, and a training set of hand labeled successful and failed pairwise point cloud registrations, we then train a decision tree to predict successful merges. Finally, we leverage the decision tree to construct a graph representation of candidate pairwise merges that are then iteratively performed to generate the object model.

In considering a candidate merge, we define one point cloud model as the base point cloud B, and the other as the target point cloud T. R represents the point cloud that results from registering T onto B, and we define b_n, t_n, and r_n as the individual points within B, T, and R, respectively. We let $distanceXYZ(i,j)$ and $distanceRGB(i,j)$ represent the Euclidean distance in XYZ space and RGB color space between points i and j, respectively; $nn(p,X)$ represents the nearest neighbor of point p in point cloud X. Finally, parameter δ represents a predefined Euclidean distance threshold between two points in a point cloud.

Based on this representation, we use the following set of metrics to describe a candidate point cloud resulting from a pairwise registration:

- **Overlap** (m_o) - percentage of overlapping points after registration

$$m_o = \frac{|\{i|b_i - nn(b_i,T) < \delta, i \in B\}|}{|B|} \quad (1)$$

- **Distance Error** (m_{dErr}) - the mean of the distance error between merged point pairs

$$m_{dErr} = \frac{1}{|B|}\sum_i distanceXYZ(b_i, nn(b_i,T)) \quad (2)$$

- **Color Error** (m_{cErr}) - the mean of the color difference between overlapping points

$$m_{cErr} = \frac{1}{n}\sum_n distanceRGB(b_n, nn(b_n,T)), \quad n \in \{i|b_i - nn(b_i,T) < \delta, i \in B\} \quad (3)$$

- **Average Color Difference** (m_{cAvg}) - average color difference between point clouds B and T

$$m_{cAvg} = abs(avg(b_i.r + b_i.g + b_i.b) - avg(t_i.r + t_i.g + t_i.b)), i \in B, j \in T \quad (4)$$

- **Color Deviation Difference** (m_{cDev}) - difference in standard deviation of color between point clouds B and T

$$m_{cDev} = abs(stdev(b_i.r + b_i.g + b_i.b) - stdev(t_i.r + t_i.g + t_i.b)), i \in B, j \in T \quad (5)$$

- **Size Difference** (m_{size}) - difference in the number of points between point clouds B and T

$$m_{size} = abs(|B| - |T|) \quad (6)$$

- **Spread Difference** (m_{spread}) - difference in the spread (i.e. the distance between the two most distant points within a point cloud) of point clouds B and T

$$m_{spread} = abs(max(distanceXYZ(b_i, b_j)) - max(distanceXYZ(t_k, t_l))), \quad (7)$$
$$i, j \in B; k, l \in T; i \neq j; k \neq l$$

We designed these metrics to represent different characteristics of the data. m_o and m_{dErr} measure differences in physical shape of the two point clouds. Furthermore, m_o provides an indication of how much new data is added to cloud B by registering it with cloud T. m_{cErr}, m_{cAvg}, and m_{cDev} measure differences in point cloud color; where m_{cAvg} indicates the overall similarity in color of the point clouds, m_{cDev} indicates whether the two point clouds have similar range of color, and m_{cErr} measures differences of color between the points of clouds B and T with relation to their spatial positions in cloud R. The remaining metrics represent a comparison of the overall size of the point clouds, with m_{size} relating their total number of points and m_{spread} relating their maximum physical lengths.

Next, we leveraged these metrics to train a decision tree to predict successful merges. We constructed a training set by applying the seven metrics to 174 instances of pairwise point cloud registrations selected from the user study data. This included both successful and failed merges, and each instance was appropriately labeled. Using this dataset, we used the C4.5 algorithm [12] to generate a decision tree to classify whether a pairwise registration was successful or failed based on the registration metrics. The final decision tree correctly classified 81% of the training data, with a false positive rate of 10%. Minimizing the false positive rate as much as possible was the most important factor in selecting the final decision tree, as accepting a failed merge propagates error throughout the rest of the object model construction process.

To construct the object model, we implemented an algorithm (Algorithm 1) that first generates a graph stucture representing all of the point clouds for each individual object in a graph, and then selectively registers pairs of point clouds to generate one or more object models. The algorithm initializes a graph with a node representing each point cloud (line 3). It next iteratively considers all pairs of nodes, and constructs edges between them for which the decision tree predicts a successful pairwise merging of the two point clouds represented by the nodes (lines 4-8). The resulting graph structure represents the similarities between the various views of the object.

The object model is constructed by collapsing the graph by merging nodes until no edges remain. The algorithm selects a random edge of the graph and performs pairwise registration on the point clouds connected by the selected edge (lines 10-11). The algorithm then removes the nodes for the two point clouds, and replaces them with a single node representing the new merged point cloud (lines 12-14). Again using the decision tree, the algorithm constructs edges from the new node to the remainder of the nodes in the graph (lines 15-19). The process then repeats until there are no graph edges left. Figure 4 provides a visual demonstration of this process.

Algorithm 1 Object model construction by graph

Require: List<PointCloud> P
1: List<PointCloud> $nodes$;
2: List<Edge> $edges$;
3: $nodes$.addAll(P);
4: **for all** n_i, n_j in $nodes$ **do**
5: **if** isSuccessfulRegistration(n_i, n_j) **then**
6: $edges$.add(Edge(n_i, n_j));
7: **end if**
8: **end for**
9: **while** $edges$ is non-empty **do**
10: Edge e = $edges$.pop();
11: PointCloud p = pairwiseRegister(e.node$_1$, e.node$_2$);
12: $nodes$.remove(e.node$_1$, e.node$_2$);
13: $edges$.removeEdgesContaining(e.node$_1$);
14: $edges$.removeEdgesContaining(e.node$_2$);
15: **for all** n_i in $nodes$ **do**
16: **if** isSuccessfulRegistration(p, n_i) **then**
17: $edges$.add(Edge(p, n_i));
18: **end if**
19: **end for**
20: **end while**

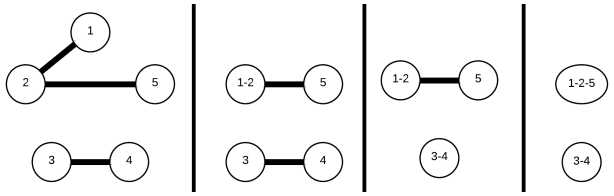

Fig. 4. A visual example of the model construction graph. Graph nodes represent point clouds, graph edges represent a possible successful registration. The example begins with 5 individual point clouds [1, 2, 3, 4, 5], and ends with the two merged models [1, 2, 5] and [3, 4]

The result is that the process can represent each object by multiple object models composed of merged point clouds from different views. For objects that look significantly different depending on the viewing angle (e.g. the front and back cover of a book), this removes the risk of poorly merging point clouds taken from significantly different angles and propagating that error through future iterative registration. This method also provides flexibility for adding new data; if any new data is obtained, it can be merged into the existing data by initializing graph nodes for both the previously merged object models and the new data.

The entire model construction algorithm takes a set of point clouds as input, and produces a set of object models for each object to be used in object recognition. The algorithm can be run either per object, where a separate graph is constructed for the point clouds associated with each individual object, or for the entire dataset, where a single graph is constructed using all of the point clouds as nodes. The latter approach has a significantly higher runtime, but as this is

essentially a training algorithm, it does not have to run in real-time.

A. Object Recognition

In order to test the usefulness of the models created by the process described above, we implemented an object recognition system based on processes similar to those used in the model construction. Specifically, using the same pairwise point cloud registration pipeline implemented for the model construction, the object recognition algorithm attempts to register a point cloud from an unknown object to each object model in the recognition database. By calculating the metrics listed above, the algorithm assigns a measure of error to each attempted pairwise registration. The algorithm then recognizes the unknown object as the database object that resulted in the lowest error.

We define the registration error, s_{err}, as a linear combination of the normalized distance error and the normalized color error.

$$s_{err} = \alpha * norm(m_{dErr}) + (1 - \alpha) * norm(m_{cErr}) \quad (8)$$

The parameter α can be set within the range $[0, 1]$ to adjust the relative weighting of the distance error and color error. Setting α closer to 0 causes the algorithm to prioritize differences in color rather than shape; likewise setting α closer to 1 causes the algorithm to prioritize differences in shape rather than color. For all experiments presented in this paper, we set α to 0.5, since we consider differences in shape and color to be equally important in our object sets.

V. RESULTS

The data collected from the crowdsourcing user study provided a set of point clouds for each object; Figure 5 shows the object set used in the study. The amount of point clouds collected varied in the range of 6 to 30 point clouds per object, as participants could pick up any of the objects as many times as they chose.

As mentioned in section IV, the object model construction algorithm could have generated models without first organizing the data by object. This works in theory, since the learned decision tree can determine whether two point clouds belong to different objects as they will result in a failed merge. In practice, however, the potential for an incorrect registration between two different but similar looking objects creates a dangerous risk of error propagation, thus we deem the approach not robust enough for practical use. Further refinement to the decision tree could reduce this risk, but that is beyond the scope of this paper.

To better demonstrate the generalizability of the object construction and recognition system, we evaluated the data using repeated random sub-sampling validation. For each iteration of testing, the point clouds for each object were randomly split into a training set (used for model construction) and a test set (used for object recognition). We performed five iterations of testing. Overall, the recognition system correctly classified objects at a rate of 88.7% +/- 5.1%. Figure 7 shows the confusion matrix for the set of objects, where entries

Fig. 5. The ten household objects used in the user study.

along the main diagonal represent correct classifications and any other cell represents a misclassification. Figure 6 provides examples of object models generated by the object model construction algorithm.

Object \ Classification	Ball	Basket	Bone	Book	Bowl	Cup	Dragon	Duck	Phone	Truck
Ball	0.680	0	0	0	0	0	0.040	0	0.280	0
Basket	0	1.000	0	0	0	0	0	0	0	0
Bone	0	0	0.743	0	0	0	0.029	0	0.229	0
Book	0	0	0	1.000	0	0	0	0	0	0
Bowl	0	0.053	0	0	0.905	0.042	0	0	0	0
Cup	0	0	0	0	0	1.000	0	0	0	0
Dragon	0	0	0	0	0	0	1.000	0	0	0
Duck	0	0	0	0	0	0	0.400	0.600	0	0
Phone	0	0	0	0	0	0	0.050	0	0.950	0
Truck	0	0	0	0	0	0	0.300	0	0	0.700

Fig. 7. The confusion matrix for the user study data. The vertical axis consists of the actual object labels, and the horizontal axis represents the labels determined by the object recognition system.

In addition to the data collected from the user study, we collected supplemental data on a larger set of thirty-four objects, shown in Figure 8. This data did not include any grasping information, as it was collected to further test the object recognition system over a larger set of objects. We designed this data collection procedure to mimic the data collected in the user study. We placed each object in randomized poses and locations on the table, and stored 20 segmented point clouds captured from an Asus Xtion Pro depth sensor mounted above a table at a similar height and angle as the PR2's head-mounted Kinect. We analyzed the performance of the algorithm on this larger dataset to determine whether our system generalized well to a larger and more varied object set. Again after organizing the point clouds by object type, we tested the object construction pipeline and object recognition system on this dataset.

Before gathering results, however, we removed a few objects from the dataset when it became apparent that they could not be properly registered. This case occurred with small, reflective objects, e.g. a metal tape measure, or a glossy paperback novel. The issue with reflective objects is that depending on the lighting and their position and orientation on the table, the point clouds at each view were too significantly different to allow for accurate registration. We determined that five objects had to be removed from the set, leaving a total of 29 objects with which to complete the evaluation.

We tested this dataset using holdout validation, in which the dataset was split evenly into a training and testing set. The object model construction algorithm used the training set as an input to construct object models, and the object

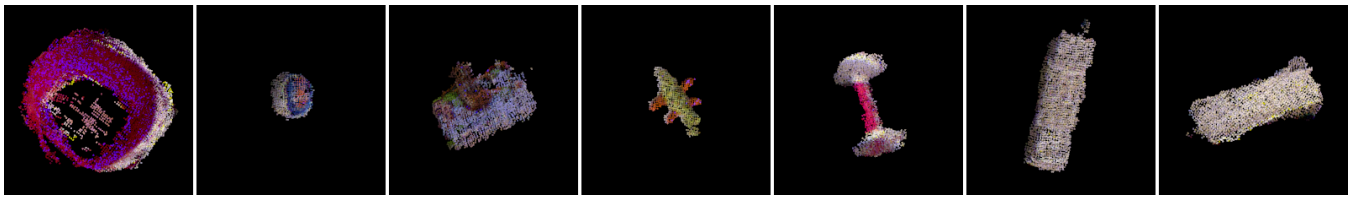

Fig. 6. Examples of object models constructed from the user study data. Left to right: bowl, hacky sack ball, dinosaur book, dragon toy, plastic dog bone, phone (front view), phone (back view)

Fig. 8. The 34 objects used to form the supplemental dataset.

recognition system used the testing set to determine the recognition accuracy. Following this test, we switched the training and testing set so that all of the data could be used for both object construction and object recognition testing. Overall, the recognition system correctly classified objects at a rate of 83.93%, showing a comparable result to the recognition rate of the 10 object dataset from the user study.

VI. CONCLUSIONS AND FUTURE WORK

Our work seeks to overcome the drawbacks of existing techniques for generating 3D object recognition data sets for mobile manipulation. We presented a novel approach for constructing an object recognition database from crowd-sourced information, contributing a framework for collecting data through the web and a novel computational method for construction and recognition of 3D object models through iterative point cloud registration. We validated this approach with data collected from a remote user study experiment, as well as with a larger supplemental dataset. The developed object recognition system was successful, correctly classifying objects at a rate of 88.7% for the experimental data and 83.93% for the larger supplemental dataset.

The presented techniques form a strong foundation for future research into mobile manipulation in real world environments. Our data collection framework records grasp data in addition to visual object descriptions, and the transforms provided by the iterative point cloud registration process enable the grasp points to be mapped onto the final object model. Our future work will explore the use of grasps in conjunction with object recognition. The user study and object recognition system presented here is but one of many possible ways of using the new techniques that web robotics has to offer. With the future inclusion of the manipulation data collected in the user study, the object recognition and grasping database built by this system is a strong example of the power of applying crowdsourcing to robotics.

ACKNOWLEDGMENT

This work was supported by the NSF award number 1149876, CAREER: Towards Robots that Learn from Everyday People, PI Sonia Chernova, and ONR Grant N00014-08-1-0910 PECASE: Tracking Human Movement Using Models of Physical Dynamics and Neurobiomechanics with Probabilistic Inference, PI Odest Chadwicke Jenkins.

REFERENCES

[1] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 1817–1824.

[2] A. Kasper, Z. Xue, and R. Dillmann, "The KIT object models database: An object model database for object recognition, localization and manipulation in service robotics," *The International Journal of Robotics Research*, vol. 31, no. 8, pp. 927–934, 2012.

[3] E. Chatzilari, S. Nikolopoulos, S. Papadopoulos, C. Zigkolis, and Y. Kompatsiaris, "Semi-supervised object recognition using Flickr images," in *Content-Based Multimedia Indexing (CBMI), 2011 9th International Workshop on*, 2011, pp. 229–234.

[4] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 11, pp. 1958–1970, 2008.

[5] R. Toris and S. Chernova, "RobotsFor.Me and Robots For You," in *Interactive Machine Learning Workshop, Intelligent User Interfaces Conference*, March 2013, pp. 10–12.

[6] A. Sorokin, D. Berenson, S. S. Srinivasa, and M. Hebert, "People helping robots helping people: Crowdsourcing for grasping novel objects," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 2117–2122.

[7] T. Azevedo, J. Tavares, and M. Vaz, "3D Object Reconstruction from Uncalibrated Images Using an Off-the-Shelf Camera," in *Advances in Computational Vision and Medical Image Processing*, ser. Computational Methods in Applied Sciences, J. Tavares and R. Jorge, Eds. Springer Netherlands, 2009, vol. 13, pp. 117–136. [Online]. Available: http://dx.doi.org/10.1007/978-1-4020-9086-8_7

[8] C. H. Esteban and F. Schmitt, "Silhouette and stereo fusion for 3D object modeling," *Computer Vision and Image Understanding*, vol. 96, no. 3, pp. 367 – 392, 2004, Special issue on model-based and image-based 3D scene representation for interactive visualization. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077314204000542

[9] M. Brown and D. Lowe, "Unsupervised 3D object recognition and reconstruction in unordered datasets," in *3-D Digital Imaging and Modeling, 2005. 3DIM 2005. Fifth International Conference on*, 2005, pp. 56–63.

[10] W. Garage. (2010, June) The household_objects SQL Database. [Online]. Available: http://wiki.ros.org/household%20objects

[11] R. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 1–4.

[12] J. R. Quinlan, *C4.5: programs for machine learning*. Morgan Kaufmann, 1993, vol. 1.