The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

Class-Specific Grasping of 3D Objects from a Single 2D Image

Han-Pang Chiu, Huan Liu, Leslie Pack Kaelbling, Tomás Lozano-Pérez

Abstract— **Our goal is to grasp 3D objects given a single image, by using prior 3D shape models of object classes. The shape models, defined as a collection of oriented primitive shapes centered at fixed 3D positions, can be learned from a few labeled images for each class. The 3D class model can then be used to estimate the 3D shape of a detected object, including occluded parts, from a single image. The estimated 3D shape is used as to select one of the target grasps for the object. We show that our 3D shape estimation is sufficiently accurate for a robot to successfully grasp the object, even in situations where the part to be grasped is not visible in the input image.**

I. INTRODUCTION

Our ultimate goal is to be able to recognize objects from known classes from a single image and to be able to grasp them. Robust monocular sensing could be a valuable complement or even substitute for more expensive three-dimensional sensors. However, given only a single 2D image, even a calibrated image, it can be challenging to reconstruct the 3D shape with sufficient precision to perform effective grasping, especially if the variation in shape of instances of the class is substantial.

In this paper, we describe an approach to reconstruct full 3D shapes from a single 2D image, based on 3D class models that are an extension of the Potemkin model [1], [2]. An object class is defined as a collection of parts, which have an arbitrary arrangement in 3D; our approach assumes that, from any viewpoint, the parts themselves can be treated as being nearly planar. This 3D model can be efficiently learned from a few part-labeled 2D views of instances of an object class from different, uncalibrated viewpoints. It does not require any 3D training information.

Once a model is learned, the reconstruction mechanism can be built on top of any 2D view-specific recognition system that returns a bounding box for the detected object. Within the bounding box, we use a model-based segmentation method to obtain an object contour. We then deform projections of the 3D parts of the class to match the object contour. Based on this fit of the visible parts to the oriented 3D primitive shapes, we can obtain an approximate reconstruction of the full 3D object.

The novelty of our approach is that it computes a reasonably accurate qualitative 3D shape model for novel instances of a known class, including its occluded parts, from only a single 2D image of the object in general position. With a fixed calibrated camera, the 3D estimation is sufficiently accurate for a robot to successfully grasp the object, even in situations where the part to be grasped is not visible in the input image.

This work was supported under DARPA IPTO Contract FA8750-05-2-0249, "Effective Bayesian Transfer Learning".

Han-Pang Chiu is with the Vision and Robotics Laboratory, Sarnoff Corporation, Princeton, NJ 08540, USA hchiu@sarnoff.com

Huan Liu, Tomás Lozano-Pérez, and Leslie Pack Kaelbling are with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA {hliu,tlp,lpk}@csail.mit.edu

II. RELATED WORK

Existing systems for planar grasp planning [4] extract the contour of the object from a single image, and detect contact points for grasping directly from the image, without any 3D information. However, these systems have limited applicability; robust manipulation of complex objects requires reliable 3D information. To obtain this 3D information, many systems [8], [9] detect and reconstruct surfaces of objects with the aid of stereo vision systems or laser scanners. Even when the sensors perform effectively, there still remains a recognition problem, especially when the objects can vary in shape, and a need to reconstruct hidden surfaces.

To avoid reconstructing 3D model for grasping, several recent approaches [16], [17], [19] identify grasp points of objects directly from monocular images. Saxena et al. [16], [17] use a supervised learning algorithm to learn 2D grasp points in monocular images, by using synthetic images as training data. They then use two or more images to triangulate the 3D location of the grasp point. This approach [18] can also be extended to involve 3D scene information obtained by depth sensor. Stark et al. [19] identify grasp regions of functional objects as a set of object pixels that has been occluded by the human in the course of an interaction. They use 2D features extracted from the identified regions to map grasp areas on detected objects. The 3D location of the grasp area is then obtained by 3D triangulation from a calibrated stereo rig.

These approaches can transfer grasping experience across similar objects by mapping 2D grasp points from training objects to test objects. However, they are still limited to grasping based on the visible surfaces.

More recently, two approaches to grasping objects from only a single image, including reconstruction of hidden surfaces, have been described.

Glover et al. [6] learn generative probabilistic models of 2D object geometry that capture the shape variability of a specific deformable object. The learned models can be used to detect objects based on visible portion of each object, and then to recover occluded object contours in a single image. However, their approach doesn't capture variability of objects within a class and is limited to planar grasp planning.

Collet et al. [3] build a metric 3D model for an object by using a set of calibrated training images. Given a single test image, they can detect multiple instances of the same object,

match instances to the stored model, and return accurate 6-DOF pose estimates, suitable for manipulation. While this approach provides reliable 3D information of known objects for grasping, it does not handle new instances that vary substantially from the stored model.

There has been recent work in the computer vision community on estimating 3D poses of class-specific instances from single images [14], [20]. However, it hasn't been shown to work for robotic grasping.

III. 3D CLASS MODELS

In this paper, we use a 3D extension of the Potemkin model [1] to represent object classes. The original Potemkin model was made up of a set of vertical planar parts, and was primarily used to transform images of objects from one view to several other views, generating virtual data for many viewpoints for multi-view recognition. In previous work [2], we have extended the Potemkin model to allow parts to be selected from a library of orientations, and demonstrated that the new model was more effective for image viewpoint transformation. In this paper, we further augment the model to support reconstruction of the 3D shapes of object instances.

A. Definition

Informally, the 3D Potemkin (3DP) *class model* can be viewed as a collection of 3D planar shapes, one for each part, which are arranged in three dimensions. The model specifies the locations and orientations of these parts in an object-centered 3D reference frame. In addition, it contains canonical images with labeled parts, which allow recognition results to be decomposed into parts. The view space is divided into a discrete set of *view bins*, and an explicit 3D rotation from the object-centered 3D reference frame to the view reference frame is represented for each view bin.

The recognition process produces a 3DP *instance model*, which is also a collection of 3D planar shapes arranged in three dimensions, corresponding to the parts of the particular 2D instance from which it was constructed.

More formally, a 3DP object class model with N parts is defined by:

- k *view bins*, which are contiguous regions of the view sphere. Each view bin is characterized by a *rotation matrix*, $T_\alpha \in R^{3\times3}$, which maps object-centered 3D coordinates to 3D coordinates in each view reference frame α;
- k *part-labeled images*, specifying the image regions of parts of an instance in each view bin α;
- a *class skeleton*, $S_1, \ldots, S_N$, specifying the 3D positions of part centroids, in the object-centered reference frame; and
- N *3D planes*, $Q_i, i \in 1, \ldots, N$, specifying the 3D plane parameters for each planar part, in the object-centered reference frame;

$$Q_i : a_i X + b_i Y + c_i Z + d_i = 0. \quad (1)$$

In addition, the 3DP class model contains an estimated bounding polygon to represent the extent of the 3D part graphically, but this polygon plays no role in reconstruction. Instead, the part shapes in the part-labeled images for each viewpoint are used for reconstruction.

B. Estimating a 3DP model from data

In broad outline, the part centroids are obtained by solving for 3D positions that best project into the observed part centroids in the part-labeled images in at least two views. The 3D planes are chosen so as to optimize the match between the 2D transformations between the boundaries of corresponding parts in the part-labeled images. Below, we give a brief overview of this estimation process; further details can be found in [2].

- The view bins are selected. The choice of view bins is arbitrary and guided by the demands of the application. In our applications, we have used 12 views bins equally spaced around a circle at a fixed elevation. The view bins determine the associated rotation matrices.
- The part-labeled images in each viewpoint should be for similarly-shaped instances of the class (though they can be significantly deformed during the recognition process) and two of them must be for the same actual instance.
- The skeleton locations S_j are estimated, using Power-Factorization [7], from the mean and covariance of the coordinates of the centroids of labeled part j in the set of part-labeled images.
- Learning the 3D planes is more involved. The process is trained in two phases: one generic, and one object-class specific. These phases are described below.

The first phase is class-independent and carried out once only. In it, the system learns, for each element of a set of oriented 3D shape primitives, what 2D image transformations are induced by changes of viewpoint of the shape primitive. The necessary data can be relatively simply acquired from synthetic image sequences of a few objects rotating through the desired space of views. Transforms for each primitive between each view bin pair are learned by establishing correspondences between points on these synthetic training images using the shape-context algorithm [15], and then using linear regression to solve for a 2D projective transform that best models the correspondence data.

The second phase is class-specific. The shape-context algorithm is used again to match points on the boundaries of each part; these matched points are then used to construct the cross-view transforms for the part across the labeled views. For each part, the oriented planar primitive that best accounts for observed cross-view transforms of the parts in the training set is selected to represent the part.

In previous experiments [2], we ran a greedy selection algorithm to select a small set of primitives that would effectively model four test object classes (chair, bicycle, airplane, car), which together have 21 separate parts. Four primitive orientations suffice to model all of the parts of

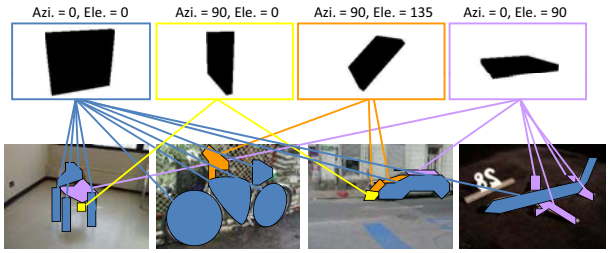

Fig. 1. 3D shape primitives selected for each part of each class.

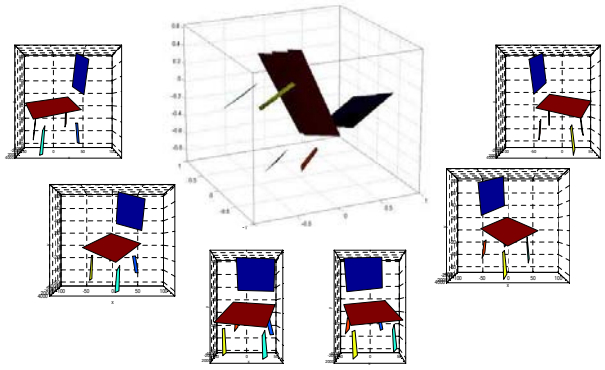

Fig. 2. Learned 3DP class model for four-legged chairs in the object-centered reference frame, and in each view reference frame.

these classes effectively. The primitives chosen for each part of each class are shown in Figure 1.

Once the primitives are selected, a small set of images, which are a subset of the k part-labeled images in the model, of the same object instance, from any set of views, as long as each part is visible in at least two views, are used to estimate the positions and orientations of the parts for this class. By finding a similarity transform between the actual part outlines and the projections of the primitives in two different views, and having computed correspondences between the outlines of the projections of the primitives in phase 1, we can solve for 3D positions of points on the outline of the shape. This allows us to estimate a rough extent and planar model of the part in 3D, even when there is very little data available. We compute $Q_1, \ldots, Q_N$ based on these planar parts.

Figure 2 shows an estimated 3DP class model for chairs. It was constructed from two part-labeled images of the same object instance, knowing the view bins but with no further camera calibration.

These easily-obtained 3DP class models may not be able to capture highly detailed shape information or all of the variability within a class, but each provides adequate information to represent the basic 3D structure shared by instances of a class. Figure 3 shows two views of the learned 3DP class model of toy cars.

IV. AUTOMATIC SINGLE-VIEW RECONSTRUCTION

In this section we will describe how to use 3DP object class models to reconstruct 3D objects from a single image.

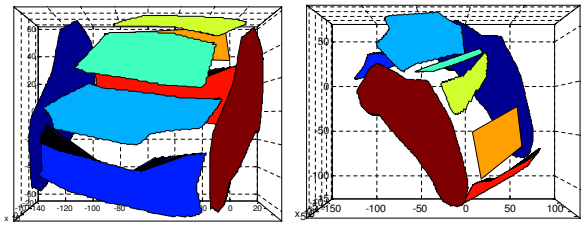

Fig. 3. 3DP class model of toy cars, constructed from four part-labeled views.

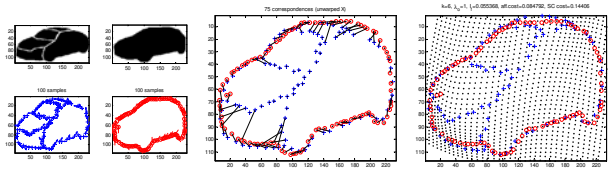

Fig. 4. Given a model instance with labeled parts (blue), the parts of another instance (red) in the same view can be found by matching points along the boundaries of the instances (middle) and by deforming the model instance into the target instance (right).

To achieve complete automation of the reconstruction process for manipulation, we developed a vision-based system involving several steps: detection, segmentation, part registration, and model creation. We will address the details of each step below.

A. Detection and segmentation

Given the input image, we need to detect the object, identify the viewpoint, and obtain the contour of the object. In theory, this step can be carried out by using any existing multi-view object-class recognition system. For example, Leibe et al.'s car detection system [11], composed of a set of seven view-dependent ISM detectors [12], provides robust results on localizing cars (a bounding box and a coarse object segmentation for each detected car) and identifying their viewpoints on test images.

In our system, we used the detection method developed by Wang et al. [21]. One advantage of this detection method is it needs only a few training instances for each viewpoint of each object class. To make the detection process more robust and efficient, we stored a background image taken by the same fixed camera in advance and used this stored image to filter foreground regions in the test image. Then our system only searches over these regions for detecting objects.

The detection system is able to determine a bounding box for the detected object and to identify the viewpoint bin. Within the bounding box, the outline of the detected object can be obtained by existing model-based segmentation techniques [13], [10]. We use the part-labeled outline for the identified view bin in our model to initialize the segmentation process. The segmented contours in our system were obtained by using the publically available implementation of level-set evolution by Li et al. [13].

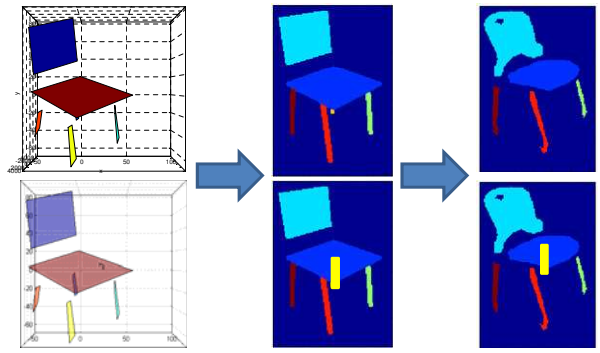

Fig. 5. Given the 3DP model of chairs in the view-reference frame (left), the whole region of the partially-occluded leg on the model instance (middle) in the same view can be registered based on visible portion. The total region of the partially-occluded leg on the target instance (right) then can be obtained by deforming the model instance into the target instance. The first row only shows visible portion on the model and the instances in the same view.

B. Part registration

Once an object outline is available, we need to obtain the part regions corresponding to the individual parts in the model. Our approach is based on the fact that objects in the same class, seen from the same view, have similar 2D arrangements of parts. That is, the centroids of the projected parts have characteristic arrangements.

We use the shape context algorithm [15] to match and deform the boundaries of the stored part-labeled image for the detected view bin into the corresponding boundary of the detected instance, as shown in figure 4. This match induces a deformation of the part-labeled image that is used to predict internal part boundaries for the detected instance. We then get the regions of non-occluded parts on the detected instance.

C. Partially-occluded part registration

For those parts that are partially-occluded in the part-labeled image, we use the 3DP model in the view-reference frame to register the whole regions of the parts based on visible portion. Then we apply the deformation on those parts from the part-labeled image to the detected instance, and get the corresponding regions of parts, as shown in figure 5.

D. Creating the 3D model

Now we are able to generate a 3D instance model from the segmented parts of the detected object in the input image using our 3D model of the class.

In our controlled environment, we calibrated a fixed camera $M \in R^{3\times 4}$ in advance, using the Matlab camera calibration toolbox. Then all objects are randomly placed on the known 3D ground plane $Q_g(a_gX+b_gY+c_gZ+d_g=0)$, a table, within a 1m by 1.2m area, visible from the camera.

We proceed in the following stages:

- Recover 3D coordinates of each image point (x_{im}, y_{im}) on the ground region by solving for X, Y, and Z in the following projection equations.

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix}. \quad (2)$$

$$x_{im} = \frac{m_{11}X + m_{12}Y + m_{13}Z + m_{14}}{m_{31}X + m_{32}Y + m_{33}Z + m_{34}}. \quad (3)$$

$$y_{im} = \frac{m_{21}X + m_{22}Y + m_{23}Z + m_{24}}{m_{31}X + m_{32}Y + m_{33}Z + m_{34}}. \quad (4)$$

$$a_gX + b_gY + c_gZ + d_g = 0. \quad (5)$$

- For each planar part i of the 3DP class model, compute the parameters $(a_{i\alpha}, b_{i\alpha}, c_{i\alpha})$ of the 3D plane $Q_{i\alpha}$ in the 3D reference frame of view bin α (identified by the detector) by applying the 3D rotation matrix T_α to Q_i. Note that the scale of parameter $d_{i\alpha}$ is unknown.
- Fit a line l_g through image points where the detected object touches the ground region in the image, and get the 3D coordinates of those ground points.
- For each object part j that includes points along the line l_g, estimate $d_{j\alpha}$ based on the recovered 3D coordinates of points on that ground line. Then, solve for the 3D coordinates of all 2D points of part j using equations (2)–(4) and $Q_{j\alpha}$ (the plane supporting part j).
- For each part k connected via adjoining pixels in the image to some previously recovered part j, estimate $d_{k\alpha}$ based on the recovered 3D coordinates of those points on the intersection of part j and part k. Then solve for the 3D coordinates of all the 2D points of part k using equations (2)–(4) and $Q_{k\alpha}$ (the plane supporting part k). Repeat this process until all parts are reconstructed.

E. Estimating locations of totally-occluded parts

After we reconstruct a 3D model for the visible parts of the detected instance in the source image, we are able to further predict approximate 3D coordinates for the totally occluded parts. We compute a 3D transformation (over translation, rotation and scale) from the 3D class model to the reconstructed 3D instance. The transformation is chosen to minimize the sum of squared distances between matching points an on the recovered 3D parts of the instance and the corresponding 3D primitive parts in the class model. Then for each totally-occluded part of the instance in the source image, we apply this 3D transformation to the corresponding part in the class model.

Figure 6 shows one example of a completely automated reconstruction. It involves detection [21], segmentation [13], part registration, and finally the reconstructed 3D instance model on the ground plane.

The ability to estimate the 3D shape and extent of the entire instance, including parts that are not visible in the source image, is very important for robot manipulation, as demonstrated in the next section.

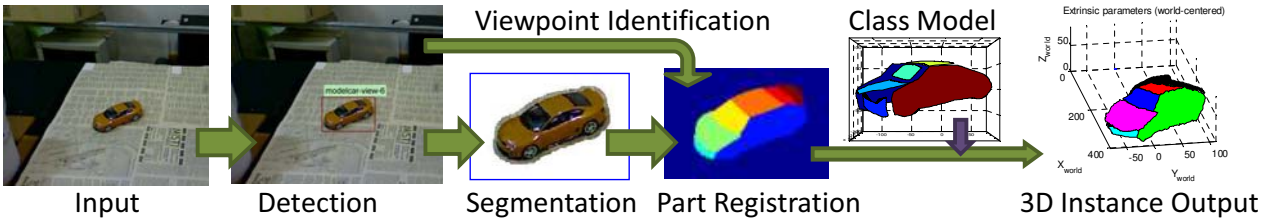

Fig. 6. The processing pipeline for automatic single-view 3D reconstruction.

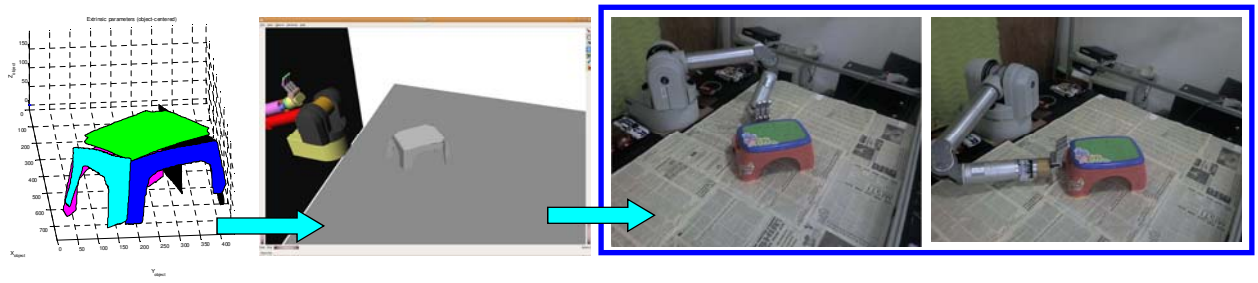

Fig. 7. From left to right: The estimated 3D instance is imported into the OpenRave [5] system and used to select a grasp from a pre-stored table that is kinematically feasible.

V. GRASPING EXPERIMENTS

Ultimately, the quality of these reconstructions can be tested by whether they support robust behavior in the real world. We do this by placing novel instances of known classes on a table in front of a robot manipulator and using the 3DP reconstruction as the basis for grasping the object and picking it up. The goal of our experiments is to evaluate whether the reconstruction accuracy for 3DP instance models is sufficient for grasping.

All grasping experiments in this section were conducted as follows (Figure 7):

- Place an object to be grasped on a known 3D ground plane, a table, within a 1m by 1.2m area. Acquire an image from a calibrated fixed camera.
- Generate a 3DP instance model of the object, as described in Section 4.
- Pick a feasible grasp for the object from a table of pre-taught grasps.
- Execute the grasp using a 7-DOF Barrett arm and 4-DOF Barrett hand.
- Count grasp as successful if the robot can lift the object from the table.

We conducted two sets of experiments. The first set tests the accuracy of the estimated 3D locations and orientations on small toy cars, which require precise 3D localization for successful grasping. The second set tests the effectiveness of our 3DP instance models for three classes of larger objects (coolers, stools, and water cans). Figure 8 shows the training instance, the constructed 3DP class model, and test instances for each of the four classes used in our experiments. The training instance is used to construct the 3DP class model, to train detectors, and to initialize the segmentation/part-registration process of detected objects.

A. Grasping with a single target grasp

In the first set of experiments, we used a single class model for a car. We placed each of the 5 test toy cars in different positions and orientations on the table, and reconstructed the 3DP instance model from each input image. Because all the toy cars are small and are grasped with the same target grasp, the grasp is determined only by the estimated orientation and 3D location of the center. The robot successfully grasped the car in 38 of 44 trials. The successful rate was around 86%. Figure 9 shows four grasps on different toy cars.

B. Grasping with multiple target grasps

In the second set of experiments, we tested the quality of the face reconstructions on three object classes (stools, coolers and watering cans) that are physically larger. We placed each instance of the three classes in different positions and orientations on the table, and reconstructed the 3DP instance model from each input image. The grasps for these classes are targeted at specific faces of the object rather than the object center. All parts, including both visible and occluded parts, of the 3D instance model are used as grasp candidates. The reconstruction for these classes needs to adapt to substantial size and shape variations among the instances so as to recover reasonably accurate face descriptions.

To simplify the process of choosing grasps, we took one object instance in each class to teach grasps. For example, figure 7 shows two demonstrated grasps for two different parts of one instance of the stool class. For each demonstrated grasp we recorded a set of grasp transforms, each

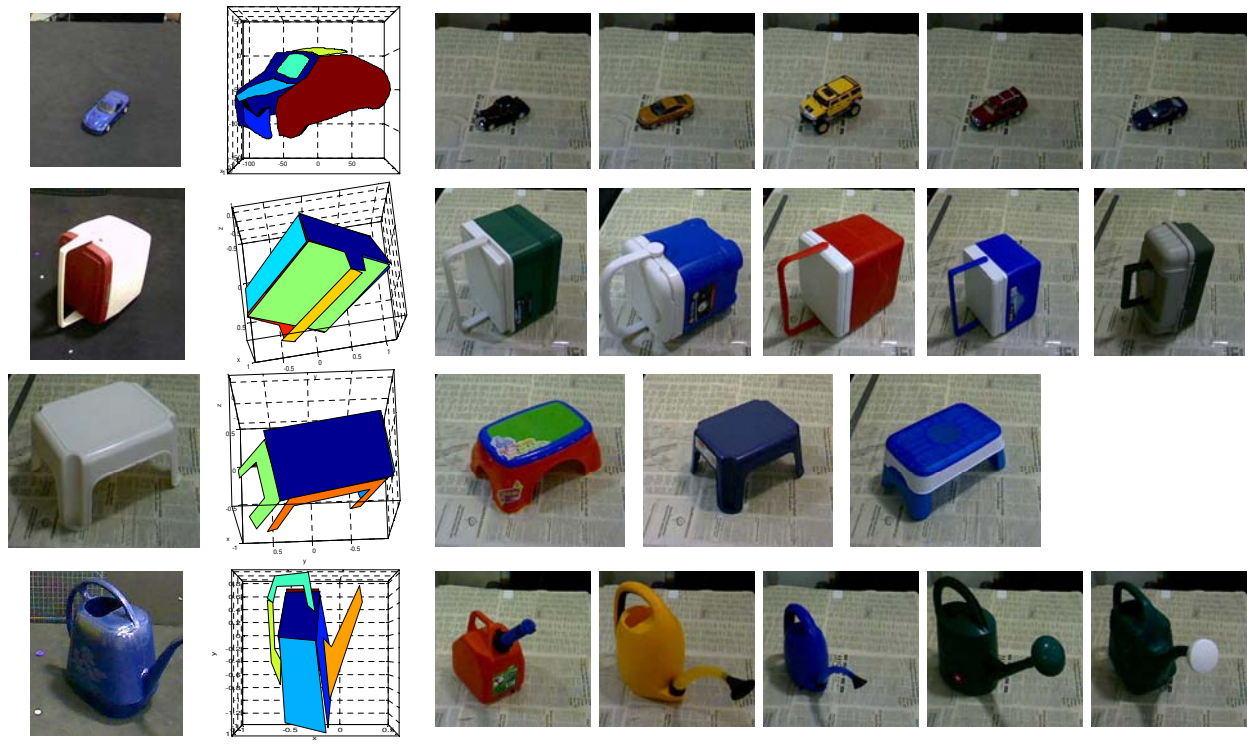

Fig. 8. For each class, there is one training instance (From Left: the first column), one 3DP class model (the second column) constructed using the training instance, and 3-5 test instances used in experiments.

Fig. 9. Four grasps on different toy cars.

one relative to one part (a planar face) of the reconstructed model of the object instance. These demonstrated grasps serve as the basis for choosing grasps for other instances of the same class. All the grasps are executed "open-loop", that is, the robot moves to the grasp pose and closes the fingers. Generally, the object will accommodate to the grasp somewhat, sometimes leading to success and other times to failure.

Given a reconstructed 3DP model from a test image, we find a face and corresponding grasp that is reachable by the robot, move there, grasp and lift. Figure 10 shows four grasps for each of the three classes. We found that grasps on the wide side of the stools, the handle of the coolers and the handle of the watering cans succeeded in lifting the object in approximately 79% of the cases (101 of 128). Each of the objects had approximately the same success rate. However, attempts to grasp the stools on the narrow end fared much worse: only 30% (8 of 26) of the attempts were successful.

One interesting question is the performance in grasping when the grasp face was one of the occluded faces. For the wide end of the stools and the cooler handle, the success rate was around 80% (46 of 53) while the narrow end of the stools yielded 30%. The watering-can handle was always visible in our experiments. When an occluded face was being grasped, we experienced improved success using grasps expressed relative to the reconstructed occluded face (21 of 22) compared to when the grasp is expressed relative to a visible face (25 of 31). This demonstrates the value of reconstructing full 3D models of objects, which support prediction of positions of occluded faces (see our accompanying video).

Fig. 10. Four grasps for each of the three classes (from top to bottom: coolers, stools, and watering cans).

VI. CONCLUSIONS

We have demonstrated an approach for reconstructing the three-dimensional structure of instances from several object classes from a single image. The reconstructions, although not perfect, are accurate enough to enable simple open-loop grasping and can be used as the starting point for more sophisticated sensor-based grasping strategies.

REFERENCES

[1] H. Chiu, L. P. Kaelbling, and T. Lozano-Perez. Virtual training for multi-view object class recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[2] H. Chiu, L. P. Kaelbling, and T. Lozano-Perez. Learning to generate novel views of objects for class recognition. *Computer Vision and Image Understanding*, 2009.

[3] A. Collet, D. Berenson, S. Srinivasa, and D. Ferguson. Object recognition and full pose registration from a single image for robotic manipulation. In *Proceedings of International Conference on Robotics and Automation*, 2009.

[4] C. Davidson and A. Blake. Error-tolerant visual planning of planar grasp. In *Proceedings of International Conference on Computer Vision*, 1998.

[5] R. Diankov and J. Kuffner. Openrave: A planning architecture for autonomous robotics. Technical Report CMU-RI-TR-08-34, Robotics Institute, CMU, 2008.

[6] J. Glover, D. Rus, and N. Roy. Probabilistic models of object geometry for grasp planning. In *Proceedings of Robotics: Science and Systems*, 2008.

[7] R. Hartley and F. Schaffalitzky. PowerFactorization: 3D reconstruction with missing or uncertain data. In *Australia-Japan Advanced Workshop on Computer Vision*, 2003.

[8] A. Hauck, J. Rttinger, M. Song, and G. Frber. Visual determination of 3d grasping points on unknown objects with a binocular camera system. In *Proceedings of International Conference on Intelligent Robots and Systems*, 1999.

[9] H. Jang, H. Moradi, S. Lee, and J. Han. A visibility-based accessibility analysis of the grasp points for real-time manipulation. In *Proceedings of International Conference on Intelligent Robots and Systems*, 2005.

[10] M. Kumar, P. Torr, and A. Zisserman. Obj cut. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[11] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Dynamic 3D scene analysis from a moving vehicle. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[12] B. Leibe, E. Seemannand, and B. Schiele. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[13] C. Li, C. Xu, C. Gui, and M. Fox. Level set evolution without re-initialization: a new variational formulation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[14] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[15] G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[16] A. Saxena, J. Driemeyer, J. Kearns, C. Osondu, and A. Ng. Learning to grasp novel objects using vision. In *International Symposium on Experimental Robotics*, 2006.

[17] A. Saxena, J. Driemeyer, and A. Ng. Robotic grasping of novel objects using vision. *International Journal of Robotics Research*, 27(2), 2008.

[18] A. Saxena, L. Wong, and A. Ng. Learning grasp strategies with partial shape information. In *Proceedings of AAAI conference on Artificial Intelligence*, 2008.

[19] M. Stark, P. Lies, M. Zillich, J. Wyatt, and B. Schiele. Functional object class detection based on learned affordance cues. In *Proceedings of International Conference on Computer Vision Systems*, 2008.

[20] M. Sun, H. Su, S. Savarese, and L. Fei-Fei. A multi-view probabilistic model for 3d object classes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[21] L. Wang, J. Shi, G. Song, and I. Shen. Object detection combining recognition and segmentation. In *Proceedings of Asian Conference on Computer Vision*, 2007.