

Grasping Class-Specific 3D Objects in a Single 2D Image

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

Abstract—Our goal is to grasp 3D objects from a single image, by using prior 3D shape models of 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 input to a robot motion planner, which results in the robot grasping the object. We provide grasp experiments demonstrating our approach estimates precise 3D locations of detected objects, constructs effective 3D shapes for instances vary from the class model, and generates complete 3D object shape from a single image. 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

Experience with instances of a class of 3D objects can yield enough information to generate effective 3D models of individual objects from a single image. Robot manipulators can rely on these generated 3D models to plan their motion and compute successful grasps.

A grasp can be computed directly based on complete 3D shape information of objects. If an object appears in a single 2D image, manipulations based on only visible parts of the object may fail. As a result, object recognition is required to identify the object’s location and pose, and a grasp can be computed by fitting a prior shape model to the object. However, novel instances of a known class of objects may vary from the prior model. Obtaining accurate 3D reconstruction of these new instances becomes a very challenging problem in robot manipulation.

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 [2], [3]. An object class is defined as a collection of parts, which have an arbitrary arrangement in 3D, but it assumes that, from any viewpoint, the parts themselves can be treated as being nearly planar.

This 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.

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

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 to compute a reasonably accurate qualitative 3D shape model for a novel class-specific instance, including its occluded parts, from only a single 2D image. With a fixed calibrated camera, the 3D estimation is sufficiently accurate for a robot to estimate the pose of the object and successfully grasp it, even in situations where the part to be grasped is not visible in the input image.

II. RELATED WORK

Some systems [5], [15] on planar grasp planning extract the contour of the object from a single image, and detect contact points for grasping directly from the image without 3D information. Although in some special cases, when grasping is done perpendicular to the image plane, grasping can be done without a 3D understanding of the scene, robust manipulation of complex objects requires reliable 3D information.

To obtain 3D information for guiding manipulations, many grasping methods [9], [10] focus on detecting and reconstructing surfaces of objects with the aid of stereo vision systems or laser scanners. These methods are only able to reconstruct visible portions of objects. Moreover, stereo systems work poorly for objects without texture.

Some researchers attempt to avoid reconstructing 3D models by identifying grasp points of objects directly from many monocular images across different viewpoints. For example, Saxena et al. [17] use supervised learning algorithms 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. The focus of these works is to simplify the data source for more generalized manipulation tasks. They do not consider any 3D shape or 2D geometry information of objects.

More recently, there are two promising works on grasping objects from only a single image. Glover et al. [7] learn generative probabilistic models of 2D object geometry, which captures the variability of instances within a known class. 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, without 3D

information, their approach cannot manipulate complicated objects.

Collet et al. [4] build a metric 3D model for a real 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 estimations for manipulation. While their work provides reliable 3D information of known objects for grasping, they cannot handle new instances that vary from the stored model.

III. 3D CLASS MODELS

In this paper, we use a 3D extension of the Potemkin model [2] 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 [3], 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*, $T_\alpha \in R^{3 \times 3}$, 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, \dots, S_N , specifying the 3D positions of part centroids, in the object-centered reference frame; and
- N *3D planes*, $Q_i, i \in 1, \dots, N$, 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 [1].

- 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 [8], 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.

The generic 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 [16], 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 [1], 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.

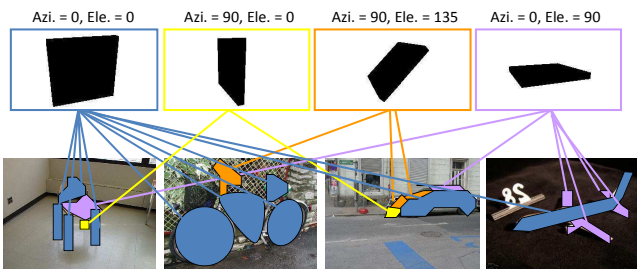


Fig. 1. 3D shape primitives selected for each part of each class.

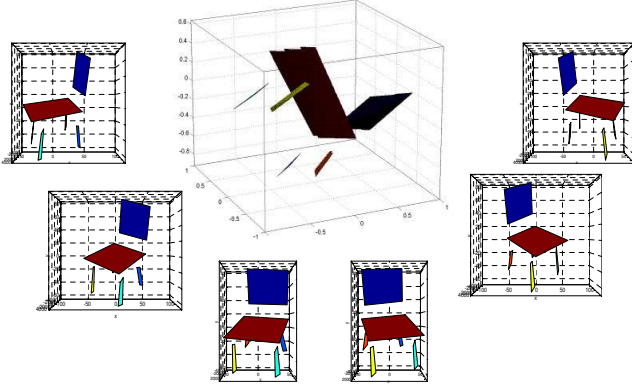


Fig. 2. Learned 3DP class model for four-legged chairs in the object-centered reference frame, and in each view reference frame.

Four primitive orientations suffice to model all of the parts of 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, \dots, 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.

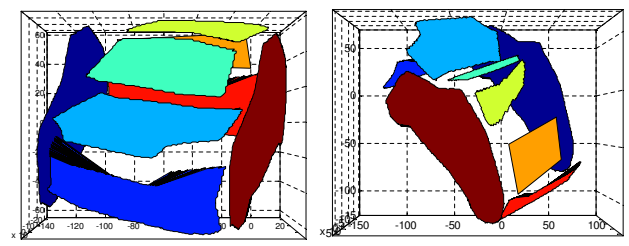


Fig. 3. 3DP class model of toy cars, constructed from four part-labeled views.

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. 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 [12], composed of a set of seven view-dependent ISM detectors [13], 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. [18]. 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.

Our 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 [14], [11]. 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. [14].

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

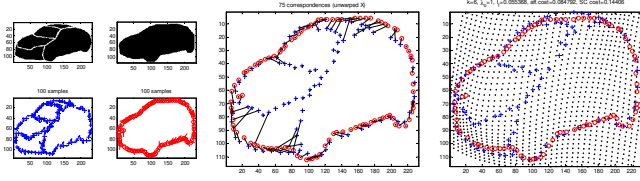


Fig. 4. Given a model instance with labeled parts (blue), the parts of another instance (red) in the same view can be found by matching points along the boundaries of the instances (middle) and by deforming the model instance into the target instance (right).

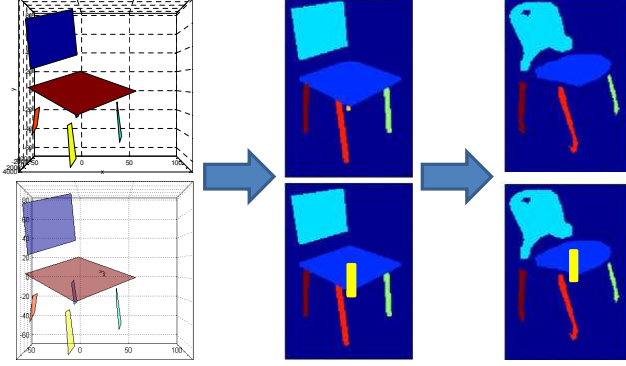


Fig. 5. Given the 3DP model of chairs in the view-reference frame (left), the whole region of the partially-occluded leg on the model instance (middle) in the same view can be registered based on visible portion. The total region of the partially-occluded leg on the target instance (right) then can be 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.

arrangements of parts. That is, the centroids of the projected parts have characteristic arrangements.

We use the shape context algorithm [16] 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 of the totally-occluded parts. We compute a 3D transformation from the 3D class model to the reconstructed 3D instance, by mapping 3D coordinates between the recovered 3D parts of the instance and corresponding 3D primitive parts in the class model. Then for each totally-occluded part i of the instance in the source image, we apply this 3D transformation to part i in the class model.

Figure 6 shows one example of a completely automated reconstruction. It involves detection [18], segmentation [14], 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 next section.

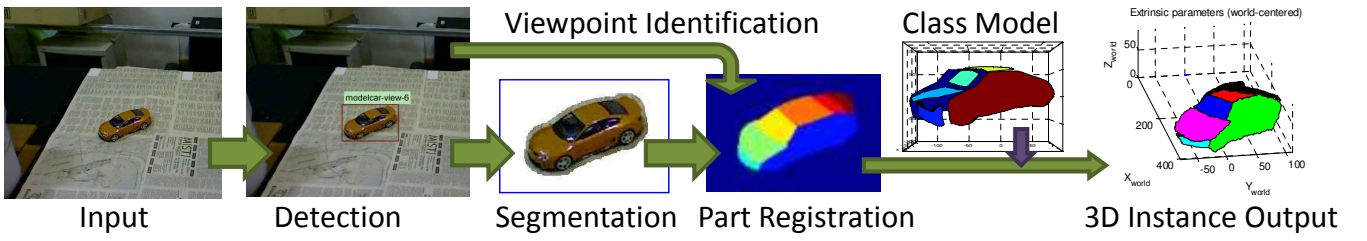


Fig. 6. The processing pipeline for automatic single-view 3D reconstruction.

V. MANIPULATION EXPERIMENTS

We use estimated 3D poses of the parts of objects as input to a robot motion planner, which calculates a motion trajectory for a robot arm and hand, which should result in the robot grasping the object. We have used a 7-DOF Barrett robot arm and 4-DOF Barrett robot hand, together with the OpenRave robot motion planning system [6] together with the 3DP model estimation outputs (Figure 7), to build a manipulation system that demonstrates hand-eye coordination in picking objects.

More formally, all manipulation experiments in this section were conducted according to the following steps:

- Place an object to be grasped on a known 3D ground plane, a table, within a 1m by 1.2m area, visible from a calibrated fixed camera. Then take a picture of the object using the camera;
- Generate a 3D instance model of the object using the picture as described in Section 4;
- Find a grasp of the object based on the 3D instance model and the robot motion planning system; and
- Execute the grasp using the robot arm and hand;

To test the utilities of 3DP instance models, we have conducted two sets of experiments. The first set demonstrates the accuracy of the estimated 3D locations and orientations on small toy cars, which require very precise 3D localization for successful grasping. The second set shows the effectiveness of our 3D instance models for three complex classes (coolers, stools, and watercans). For novel instances vary from the class models, successful grasps can be achieved even in situations where the part to be grasped is not visible in the input image.

In sum, the goal of our experiments is to show:

- The 3DP instance model estimates an object’s center position and orientation with high accuracy;
- The 3DP instance model provides accurate position and orientation estimations for visible parts of objects that can vary from the 3DP class model; and
- The 3DP instance model provides accurate position and orientation estimations for occluded parts of objects that can vary from the 3DP class model;

Figure 8 shows the training instance, the constructed 3DP class model, and the 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. 3D localization

In the first set of experiments, we placed each of the 5 test toy cars in 10 different positions and orientations on the table, and reconstructed the 3D car from each input image. Because all toy cars are small and have relatively the same shape, we only take the 3D center location and orientation for planning the grasp. As the result, the robot successfully grasped the car in all these 50 trials,

B. Grasping novel class-specific objects

In the second set of experiments, we test our approach on three object classes which require more complicated grasping strategies. All parts, including both visible and occluded parts, of the 3D instance model need to be used for grasp generation. For each class, we placed each test instance in a set of poses (locations/orientations). We took the first test instance as the instance to generate demonstrated grasps. For each pose of the first test instance, we recorded several demonstrated grasps. For example, figure 7 shows two demonstrated grasps relative to two different parts of the first test instance of the stool class. Such demonstrated grasps were ready to be applied to any instance of the same class. We recorded the following information for each demonstrated grasp:

- A set of relative grasp transform to each part (a planar face) of the object; and
- An ordered list of part indices. For example, [2,4,1,0,3] means that the relative grasp transform to Part 2 is more preferable than those to other parts.

Then for a 3D instance model of an object reconstructed from a test image, we execute a grasp experiment as following steps:

- Get the 3D position/orientation of the most preferable part from the 3D instance model;
- Move the robot hand to the grasp position relative to that part based on the corresponding demonstrated grasp;
- Grasp; and
- If the grasp fails, repeat the above steps with the next preferable part.

We did the above grasp experiment for each pose of each test instance of each object class.

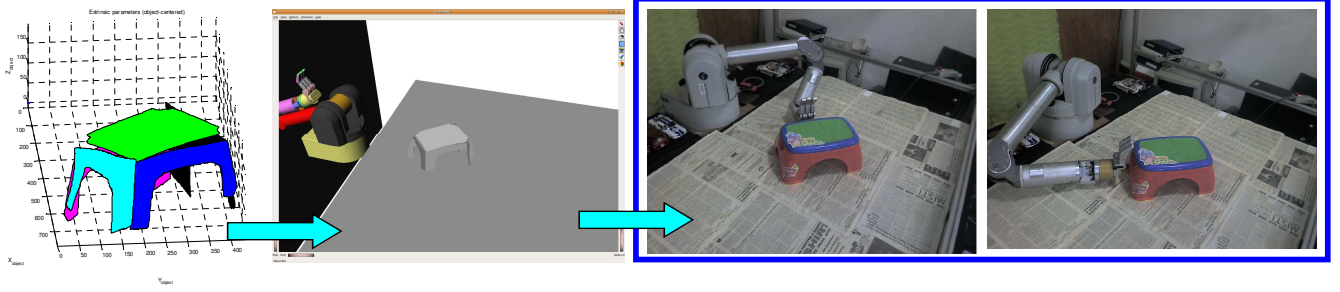


Fig. 7. From left to right: The estimated 3D instance is used as input to the OpenRave robot motion planning system, which results in two grasps from different directions.

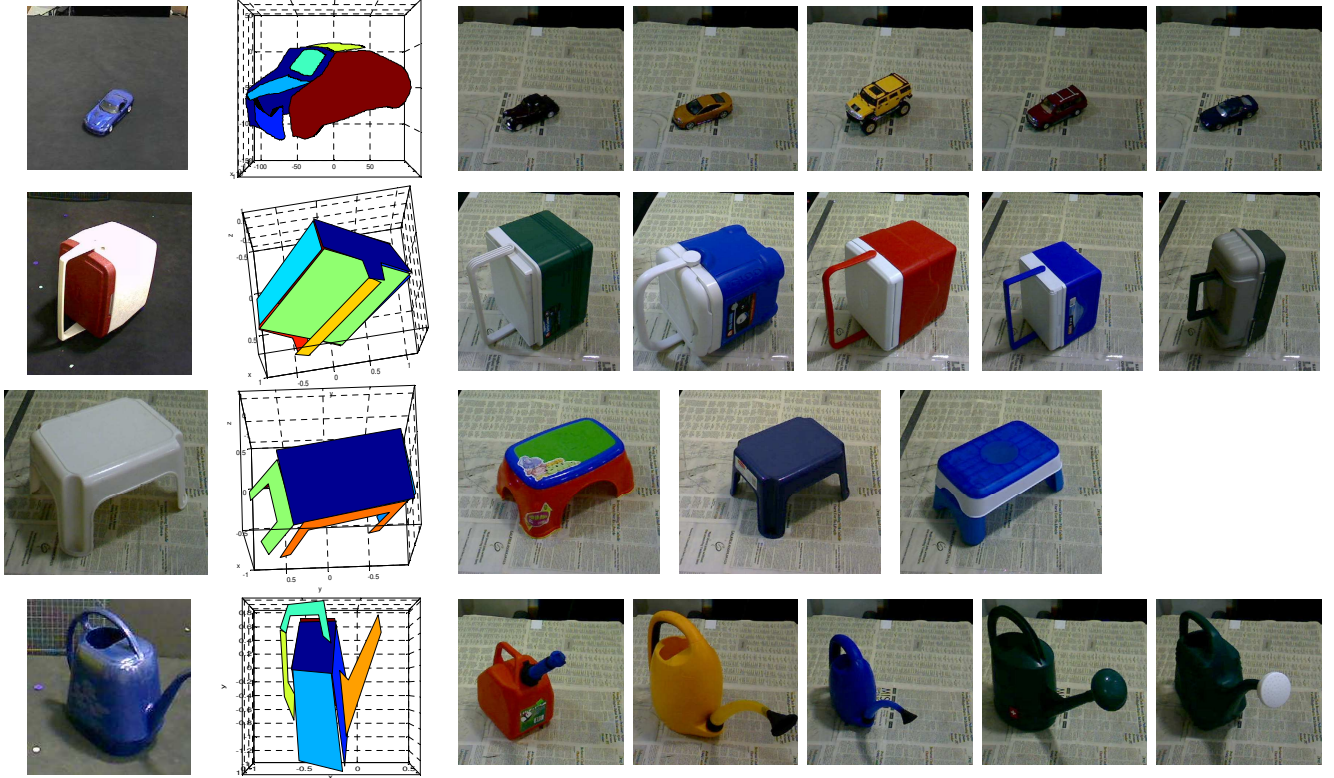


Fig. 8. For each class, there is one training instance (From Left: the first column), one 3DP class model (the second column) constructed using the training instance, and 3-5 test instances used in experiments.

We also propose two variations of our 3DP model as control methods for the same grasp experiments.

One is called 3DP-Visible models, which are constructed without the step in Section 4.5 and consist of only visible parts of the 3DP models. The grasp process is the same as that for 3DP models, except that the grasp only applies to parts that can be seen in the input image. We predicted that if the most preferable part did not exist in the 3DP-Visible model, the grasp derived from a less preferred part might not be successful.

The other control method is 3DP-Stretch models, which are transformed 3DP class models. To construct a 3DP-Stretch model, instead of the part registration step (Section 4.2), we compute a scale factor that matches the seen 2D

parts of the training instance with the same viewpoint to the segmented test instance. We then apply this scale factor to transform the 3DP class model, which is constructed using images of the training instance. The transformed 3DP class model becomes the resulting 3DP-Stretch model, which contains all the parts (including occluded parts) as a 3DP model. The grasp strategies for 3DP models can also be applied for 3DP-Stretch models, since both methods reconstruct entire 3D shapes of objects. However, 3DP-Stretch models cannot deal with shape variation within a class. If the shape of the test instance is very different from the training instance, grasps derived from 3DP-Stretch models may fail.

VI. CONCLUSIONS AND FUTURE WORK

REFERENCES

- [1] Anonymous. Learning to generate novel views of objects for class recognition. *Computer Vision and Image Understanding*, under review.
- [2] H. Chiu, L. P. Kaelbling, and T. Lozano-Perez. Virtual training for multi-view object class recognition. In *Proc. CVPR*, 2007.
- [3] 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.
- [4] A. Collet, D. Berenson, S. Srinivasa, and D. Ferguson. Object recognition and full pose registration from a single image for robotic manipulation. In *International Conference on Robotics and Automation*, 2009.
- [5] C. Davidson and A. Blake. Error-tolerant visual planning of planar grasp. In *Proceedings of International Conference on Computer Vision*, 1998.
- [6] R. Diankov and J. Kuffner. Openrave: A planning architecture for autonomous robotics. Technical Report CMU-RI-TR-08-34, Robotics Institute, CMU, 2008.
- [7] J. Glover, D. Rus, and N. Roy. Probabilistic models of object geometry for grasp planning. In *Proceedings of Robotics: Science and Systems*, 2008.
- [8] R. Hartley and F. Schaffalitzky. PowerFactorization: 3D reconstruction with missing or uncertain data. In *AJAWCV*, 2003.
- [9] A. Hauck, J. Rittinger, 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.
- [10] 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.
- [11] M. Kumar, P. Torr, and A. Zisserman. Obj cut. In *Proc. CVPR*, 2005.
- [12] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool. Dynamic 3D scene analysis from a moving vehicle. In *Proc. CVPR*, 2007.
- [13] B. Leibe, E. Seemannand, and B. Schiele. Pedestrian detection in crowded scenes. In *Proc. CVPR*, 2005.
- [14] C. Li, C. Xu, C. Gui, and M. Fox. Level set evolution without re-initialization: a new variational formulation. In *Proc. CVPR*, 2005.
- [15] A. Miller, S. Knoop, P. Allen, and H. Christensen. Automatic grasp planning using shape primitives. In *Proceedings of International Conference on Robotics and Automation*, 2003.
- [16] G. Mori, S. Belongie, and J. Malik. Shape contexts enable efficient retrieval of similar shapes. In *Proc. CVPR*, 2001.
- [17] A. Saxena, J. Driemeyer, J. Kearns, C. Osundu, and A. Ng. Learning to grasp novel objects using vision. In *International Symposium on Experimental Robotics*, 2006.
- [18] L. Wang, J. Shi, G. Song, and I. Shen. Object detection combining recognition and segmentation. In *Proc. ACCV*, 2007.