Visual-Tactile Geometric Reasoning

Anonymous Author(s)
Affiliation
Address
email

Abstract:

This work provides an architecture which uses a learning algorithm that incorporates depth and tactile information to create rich and accurate 3D models from single depth images. The models are then able to be used for robotic manipulation tasks. This is accomplished through the use of a 3D convolutional neural network (CNN). Offline, the network is provided with both depth and tactile information and trained to predict the object's geometry, filling in the occluded regions of the object. At runtime, the network is provided a partial view of an object. The network then produces an initial object hypothesis using depth alone. A grasp is planned using this hypothesis and a guarded move takes place to collect tactile information. The network can then improve the system's understanding of the object's geometry by utilizing the newly collected tactile information.

Keywords: Sensor Fusion, Grasping, Deep Learning

1 Introduction

Grasp planning based on raw sensory data is difficult due to occlusion and incomplete information regarding scene geometry. Often one sensory modality does not provide enough context to enable reliable planning. For example a single depth sensor image cannot provide information about occluded regions of an object, and tactile information is incredibly sparse spatially. This work utilizes a 3D convolutional neural network to enable stable robotic grasp planning by incorporating both tactile and depth information to infer occluded geometries. This multi-modal system is able to utilize both tactile and RGBD information to form a more complete model of the space the robot can interact with and also to provide a complete object model for grasp planning.

During the runtime stage, a point cloud of the visible portion of the object is captured. As described in section 3 it is voxelized and sent through a CNN to provide an initial hypothesis of the object's geometry. This initial hypothesis is used to plan a grasp. As described in section 4 the hand is then moved to the planned grasp via a guarded move, stopping when contact with the object occurs. At this point, the newly acquired tactile information is combined with the original partial view and sent through the CNN to create an updated object geometry hypothesis. This new hypothesis incorporates both the depth and tactile information.

The contributions of this work include: 1) an open source dataset for training a shape completion system using both tactile and depth sensory information, 2) a framework for integrating multi-modal sensory data to reason about object geometry, and 3) results comparing the completed object models using depth only and combined depth-tactile information.

2 Related Work

The idea of incorporating sensory information from vision, tactile and force sensors is not new [1]. Despite the intuitiveness of using multi-modal data, there is still no agreed upon framework to best integrate multi-modal sensory information in a way that is useful for robotic manipulation tasks. In this work, we are interested in reasoning about object geometry in particular.

Submitted to the 1st Conference on Robot Learning (CoRL 2017). Do not distribute.

Several recent uses of tactile information to improve estimates of object geometry has focused on the use of Gaussian Process Implicit Surfaces(GPIS) [29]. Several examples along this line of work include [7][30] [6][10][16][26][21]. This approach is able to quickly incorporate additional tactile information and improve the estimate of the objects geometry local to the tactile contact or observed sensor readings. There has additionally been several works that incorporate tactile information to better fit planes of symmetry and super quadrics to observed point clouds [15][14][5]. These approaches work well when interacting with objects that confirm to the heuristic of having clear detectable planes of symmetry or are easily modeled as super quadrics.

There has been successful research in utilizing continuous streams of visual information similar to Kinect Fusion[24] or SLAM[28] in order to improve models of 3D objects for manipulation. One example being [20][19] In this work, the authors develop an approach to building 3D models of unknown objects based on a depth camera observing the robots hand while moving an object. The approach integrates both shape and appearance information into an articulated ICP approach to track the robots manipulator and the object while improving the 3D model of the object. Similarly [13] attaches a depth sensor to a robotic hand, and plans grasps directly in the sensed voxel grid. These approaches improve their models of the object using only a single sensory modality, but from many time points.

3 Visual Geometric Reasoning for Robotic Grasping

In previous work, we created a shape completion method using single depth images [2]. The work provides an architecture to enable robotic grasp planning via shape completion. Shape completion is accomplished through the use of a 3D convolutional neural network (CNN). The network is trained on an open source dataset of over 440,000 3D exemplars captured from varying viewpoints. At runtime, a 2.5D point cloud captured from a single point of view is fed into the CNN, which fills in the occluded regions of the scene, allowing grasps to be planned and executed on the completed object. Runtime shape completion is very rapid because most of the computational costs of shape completion are borne during offline training. This work explored how the quality of completions vary based on several factors. These include whether or not the object being completed existed in the training data and how many object models were used to train the network, and the ability of the network to generalize to novel objects allowing the system to complete previously unseen objects at runtime. Below we summarize this method and discuss how we can augment it with tactile data to generate more accurate complete models.

3.1 Data Generation

In order to train a network to reconstruct a diverse range of objects, meshes were collected from the YCB[8] and Grasp Database[17]. The models were run through binvox[23] in order to generate 2563 occupancy grids. In these occupancy grids, both the surface and interior of the meshes are marked as occupied. In addition, all the meshes were placed in Gazebo, and 726 depth images were generated for each object subject to different rotations uniformly sampled (in roll-pitch-yaw space, 11*6*11) around the mesh. The depth images are used to create occupancy grids for the portions of the mesh visible to the simulated camera, and then all the occupancy grids generated by binvox are transformed to correctly overlay the depth image occupancy grids. Both sets of occupancy grids are then down-sampled to 403 to create a large number of training examples. The input set (X) contains occupancy grids that are filled only with the regions of the object visible to the camera, and the output set (Y) contains the ground truth occupancy grids for the space occupied by the entire model. An illustration of this process is shown in Fig. 2.

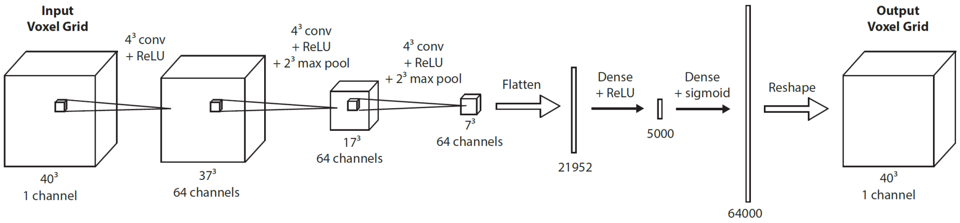


Figure 1: CNN Architecture. The CNN has three convolutional and two dense layers. The final layer has 64000 nodes, and reshapes to form the resulting 40^3 occupancy grid. The numbers on the bottom edges show the input sizes for each layer. All layers use ReLU activations except for the last dense layer, which uses a sigmoid.

3.2 Model Architecture and Training

The architecture of the CNN is shown in Fig. 1. The model was implemented using Keras[9], a Theano[4][3] based deep learning library. Each layer used rectified linear units as nonlinearities except the final fully connected (output) layer which used a sigmoid activation to restrict the output to the range $[0, 1]$. They used the cross-entropy error $E(y, y')$ as the cost function with target y and output y':

$$E(y, y_0) = -(ylog(y') + (1 - y)log(1 - y'))$$

This cost function encourages each output to be close to either 0 for unoccupied target voxels or 1 for occupied. The optimization algorithm Adam[18], which computes adaptive learning rates for each network parameter, was used with default hyperparameters ($\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$) except for the learning rate, which was set to 0.0001. Weights were initialized following the recommendations of [12] for rectified linear units and [11] for the logistic activation layer. The model was trained with a batch size of 32. Each of the 32 examples in a batch was randomly sampled from the full training set with replacement. They used the Jaccard similarity to evaluate the similarity between a generated voxel occupancy grid and the ground truth. The Jaccard similarity between sets A and B is given by:

$$J(A, B) = |\frac{A \cap B}{A \cup B}|$$

[htpb]

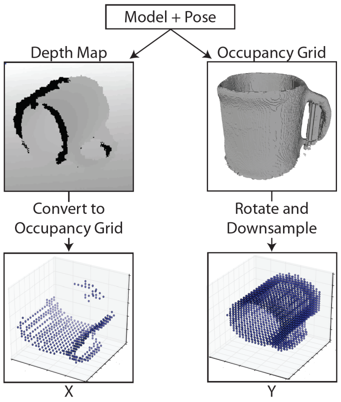
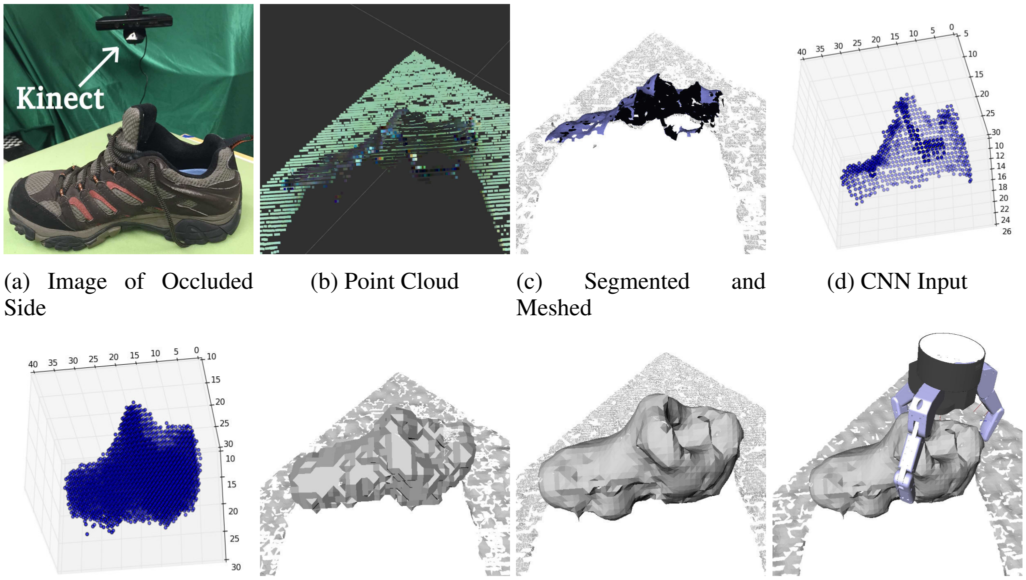


Figure 2: Training Data: In X, the input to the CNN, the occupancy grid marks visible portions of the model. Y, the expected output, has all voxels occupied by the model marked.

The Jaccard similarity has a minimum value of 0, where A and B have no intersection and a maximum value of 1 where A and B are identical. During training, this similarity measure is computed for input meshes that were in the training data (Training Views), meshes from objects within the training data but from novel views (Holdout Views), and for meshes of objects not in the training data (Holdout Models). The CNNs were trained with an NVIDIA Titan X GPU. When we integrated the tactile completion into this pipeline we chose to use the same comparison of training objects and holdout objects as a methodology for evaluating the success of the network.

3.3 Runtime

At runtime the point cloud for the target object is acquired from a 3D sensor, scaled, voxelized and then passed through the CNN. The output of the CNN, a completed voxel grid of the object, goes through a post processing algorithm that returns a mesh model of the completed object. Finally, a grasp can be planned and executed based on the completed mesh model. Fig. 3 demonstrates the full runtime pipeline on a novel object never seen before. With our included tactile process we expand on this process by integrating an additional two steps which are described in section 4.



(a) Image of Occluded Side

(b) Point Cloud

(c) Segmented and Meshed

(d) CNN Input

(e) CNN Output

(f) Fast Mesh

(g) Detailed Mesh

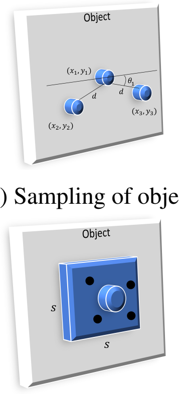
(h) Grasp Planning

Figure 3: Stages to Shape Completion using vision data only. These images are not shown from the angle in which the data was captured in order to visualize the occluded regions. (a): An object to be grasped is placed in the scene. (b): A point cloud is captured. (c): The point cloud is segmented and meshed. (d): A partial mesh is selected by the user and then voxelized and passed into the 3D shape completion CNN. (e): The output of the CNN. (f): The resulting occupancy grid can be run through a marching cubes algorithm to obtain a mesh quickly. (g): Or, for better results, the output of the CNN can be combined with the observed point cloud and preprocessed for smoothness before meshing. (h): Grasps are planned on the smoothed completed mesh. Note: this is a novel object not seen by the CNN during training.

First a targeted point cloud is acquired using a Microsoft Kinect and segmented using PCL's [25] implementation of Euclidian clustering. Then the partial mesh is completed using a CNN with a resolution of 40^3 with an architecture as described in 3.2. The mesh is then smoothed using a marching cubes algorithm and then upscaled using a quadractic programming algorithm as described in [2]. Finally a grasp is calculated using the Graspit! [22] software using the Barrett Hand model. The reachability of the planned grasps are checked using MoveIt![27] and the highest quality reachable grasp is then executed. For the purposes of this paper this last step has been omitted from our data collection step.

3.4 Performance

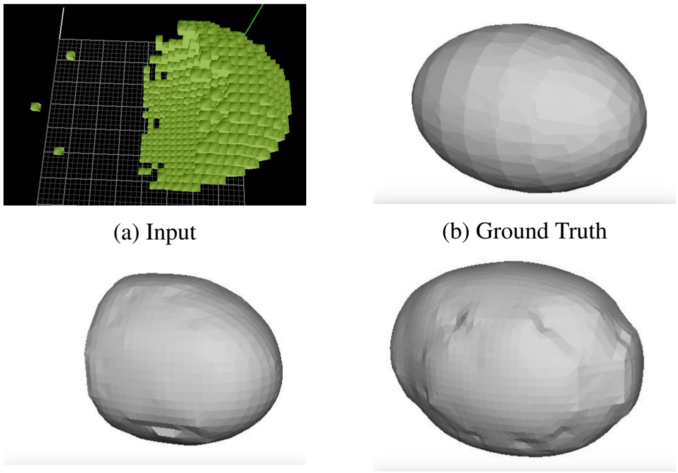
We created a test dataset by randomly sampling 50 training views (Training Views), 50 holdout views (Holdout Views), and 50 views of holdout models (Holdout Models). The Training Views and Holdout Views were sampled from the 14 YCB training objects. The Holdout Models were sampled from holdout YCB and Grasp Dataset objects. We used three metrics to compare the accuracy of the different completion methods: Jaccard similarity, Hausdorff distance, and geodesic divergence. We were able to show improvements over the partial and mirror methods by a significant margin. When comparing our shape completion method to a RANSAC based algorithm we were able to show our algorithm was more generalizable Jaccard (Ours: 0.771, RANSAC: 0.8566), Hausdorff (Ours: 3.6, RANSAC: 3.1), geodesic (Ours: 0.0867, RANSAC: 0.1245). Our approach significantly outperforms the RANSAC approach when encountering an object that neither method has seen before (Holdout Models): Jaccard (Ours: 0.6496, RANSAC: 0.4063), Hausdorff (Ours: 5.9, RANSAC: 20.4), geodesic (Ours: 0.1412, RANSAC: 0.4305). The RANSAC based approachs performance on the Holdout Models is also worse than that of the mirrored or partial completion methods on both the geodesic and Hausdorff metrics.



(a) Sampling of object

(b) Sampling around contact

Figure 4: To generate the simulated tactile data we used an approximation of the Barrett hand's geometry which included finger offset as well as rotation about the z-axis of the camera frame. We then sample around three suggested contact points.



(a) Input

(b) Ground Truth

(c) Depth Only Completion

(d) Tactile + Depth Completion

Figure 5: Egg completion from the YCB and grasp database holdout model set. It is hard to determine how far back the completion actually goes, and it is hard to differentiate what object this is as the dataset contains both eggs and bowls. The Tactile + Depth Completion is better as it uses the tactile information to alleviate both concerns.

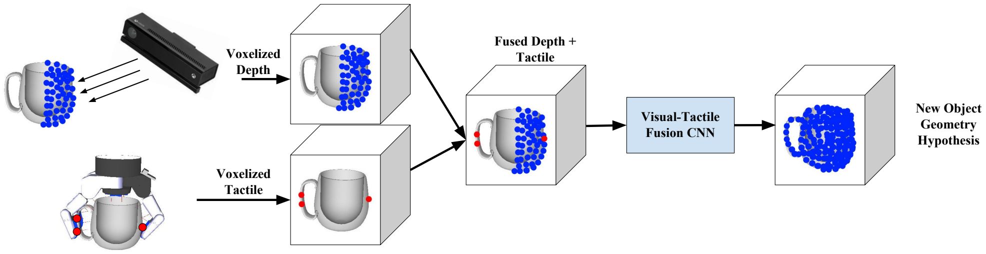


Figure 6: Both Tactile and Depth information are independently captured and voxelized into 40^3 grids. These are merged into a shared occupancy map which is fed into a CNN to produce a hypothesis of the object's geometry.

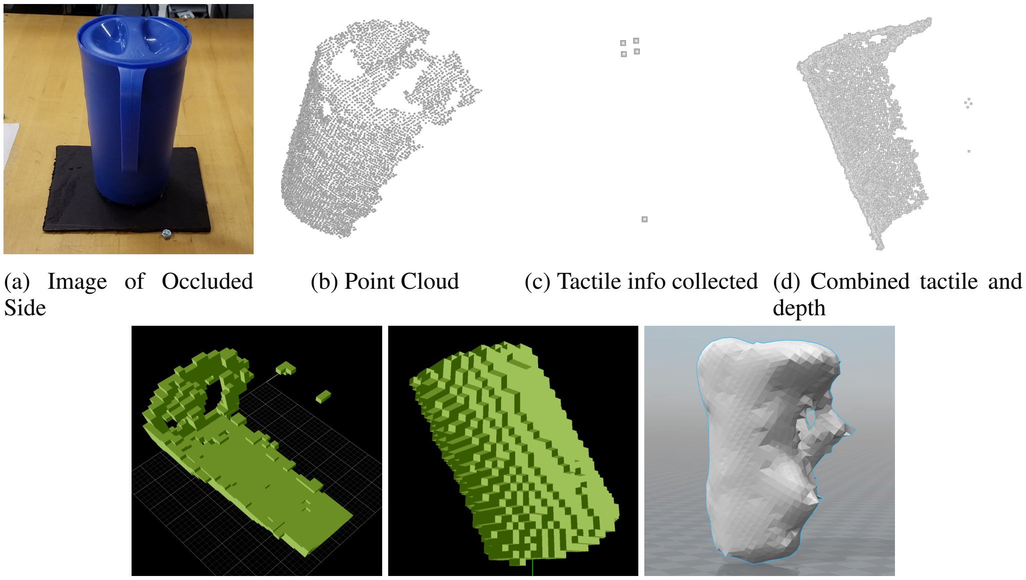
4 Visual-Tactile Geometric Reasoning for Robotic Grasping

The results above provide a series of reasonable shape approximations using a CNN which is trained on a data set of partial views. However a CNN trained on depth alone is not able to account for full range of object geometry that cannot be viewed. To alleviate this, our solution is to add tactile data from tactile probing to the hypothesized shape completed model and generate a new more accurate model incorporating both visual and tactile information. To generate synthetic tactile data we used an approximation of the Barrett hand shown in Fig. 4. This model can then be used for grasp planning and manipulation. An overview of our sensory fusion architecture is shown in Fig. 6.

4.1 Training

The dataset consists of approximately half a million pairs of oriented voxel grids. Where one grid's voxels are marked as occupied if visible to a camera, and the second grid's voxels are marked as occupied if the object intersects a given voxel, independent of perspective. This dataset was augmented with tactile information either from a tactile grasp, or tactile exploration as shown in Fig. 10.

We generated a series of simulated tactile points and combined these points in a 3D voxel grid with a partial view of a ground truth mesh as described in section 3.1. We stored these as a series of


(a) Image of Occluded Side

(b) Point Cloud

(c) Tactile info collected

(d) Combined tactile and depth

(e) CNN input

(f) CNN output

(g) Detailed Mesh

Figure 7: Stages to Shape Completion using vision and tactile data. These images are not shown from the angle in which the data was captured in order to visualize the occluded regions. (a): An object to be grasped is placed in the scene. (b): A point cloud is captured. (c): Tactile information is collected. (d): Tactile information and depth information are merged into one point cloud. (e): The input to the CNN. (f): The output of the CNN. (g): A smoothed mesh of the CNN output using the marching cubes algorithm.

binvox files for the purposes of training a CNN. In order to generate the tactile points we found a set of three points by taking samples across the y-axis of the object in the -z direction. These three points were generated by taking rays through the 3D voxel grid and combining them with the partial view of the object as shown in Fig. 7.

This provided information about up to three additional occupied voxels marking where each finger intersects the object. We then changed our runtime pipeline to incorporate the new tactile information as shown in Fig. 4. A good example of this additional benefit is shown in Fig. 5 where the network was able to complete the given egg voxel grid despite not seeing the back half of the object by incorporating the new tactile information. The tactile information allows the system to correctly predict how far back the completed object should extend and disambiguate between objects used in training that have similar depth maps but very different completions. Fig. 8 shows how completion quality improves as training progresses for two networks one trained using depth alone, and the second trained using depth and tactile information. It is interesting to note that difference in performance between the two networks is much larger on Holdout Models than on Train Views. This can be interpreted to mean that the additional tactile information is more useful on novel objects, while depth alone maybe sufficient for good completions if the object was used during training.

5 Experimental Results

In order to evaluate our system, it was first trained on a simple shape dataset. This dataset consisted of conjoined half shapes. Both front and back halves of the objects were randomly chosen to be either a sphere, cube, or diamond. The front and back halves do match in size. Several example shapes are shown in Fig. 9 (b) half cube half sphere and (d) half sphere half diamond. Next, synthetic sensory data was generated for these example shapes. Depth information was captured from a fixed camera location, and tactile information was collected using both a tactile exploration, and a tactile grasp. The sensory data for two shapes is shown in Fig. 9 (a) and (c) . Fig 10 shows the difference between the tactile grasp and tactile exploration.

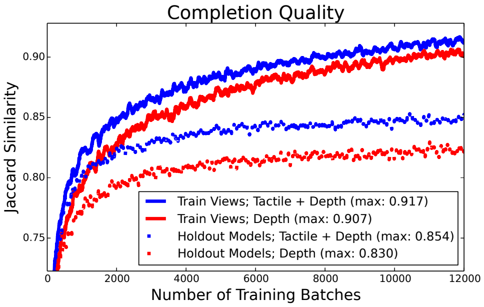


Figure 8: Jaccard similarity for two CNNs, one (Red: Depth) trained with depth alone, the second (Blue: Tactile + Depth) trained with tactile and depth information. While training, the CNNs were evaluated on inputs they were being trained on (Train Views) and novel inputs from meshes they have never seen before (Holdout Models). In both evaluations the network provided with both depth and tactile is able to do a better job, this is especially true for Holdout Models demonstrated by the widened performance gap between the two networks.

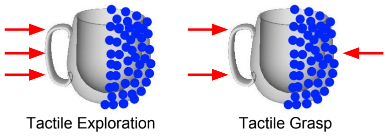
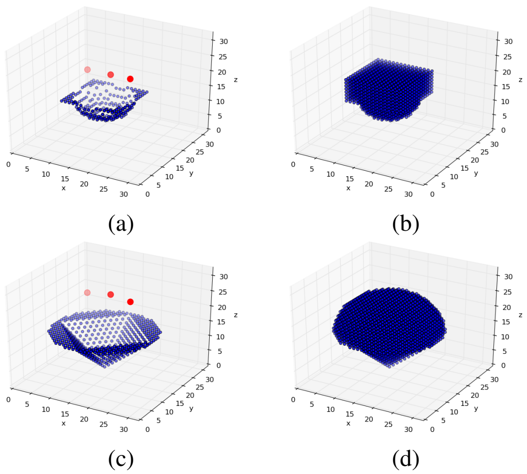


Figure 10: Red arrows show how the fingers approach the object for the tactile exploration case and for the tactile grasp case. Blue dots show points in the depth image captured by the camera.



(a) (b) (c) (d)

Figure 9: Example training pairs from simple shape dataset. The red dots represent the tactile readings from tactile exploration. The blue dots on (a) and (c) represent to occupancy map gathered from the depth image. The blue points in (b) and (d) represent the ground truth 3d geometry.

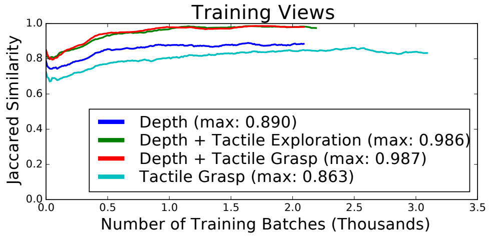
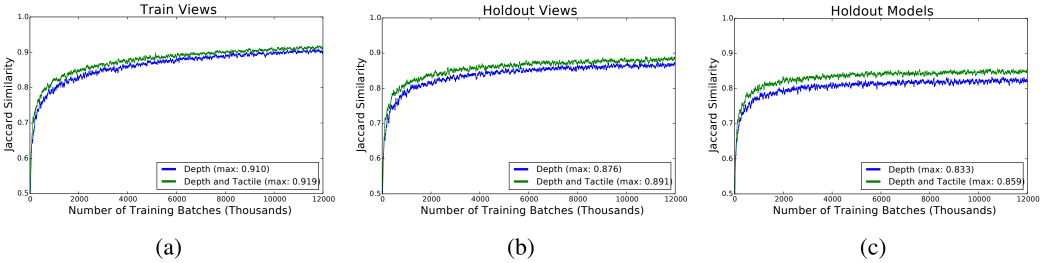


Figure 11: Different runs of the shape completion system where input is provided from: Depth, Depth and Tactile Exploration, Depth and Tactile from Grasp, and only from a Tactile Grasp. When using both tactile and grasp information, the system is able to complete the object almost 100% of the time. While depth or tactile alone are not sufficient to successfully reason about object geometry in all cases.

Four networks with the exact same architecture were trained on this dataset using different sensory data as input. The results are shown in Fig. 11. One network was only provided the tactile grasp information during training, and performed poorly. A second network was given only the depth information during training, and performed better than the first network, but still encountered many situations where it did not have enough information to accurately complete the back half of the object. The other two networks were given the depth and tactile information. One in the form of a tactile grasp and the other from a tactile exploration. These networks were able to learn the task to completion. They successfully utilized the tactile information to differentiate between plausible geometries of occluded regions.

5.1 YCB Live Hardware Experiments

After demonstrating on the simple shape dataset, we trained two additional models using 486 of the grasp and YCB dataset objects, the remaining models were kept for a holdout set. One model was again trained using only the depth information, while a second model was trained using both depth and tactile information provide from a tactile exploration performed in a similar manner as with the

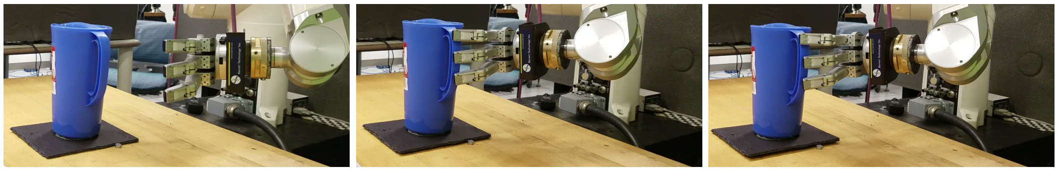


(a)

(b)

(c)

Figure 12: Jaccard similarity for two CNNs, one (shown in blue) trained with depth alone, the second (green) trained with depth and tactile information. For each plot, while training, the CNNs were evaluated on inputs they were being trained on (Training Views, plot a), novel inputs from meshes they were trained on (Holdout Views, plot b) and novel inputs from meshes they have never seen before (Holdout Models, plot c).



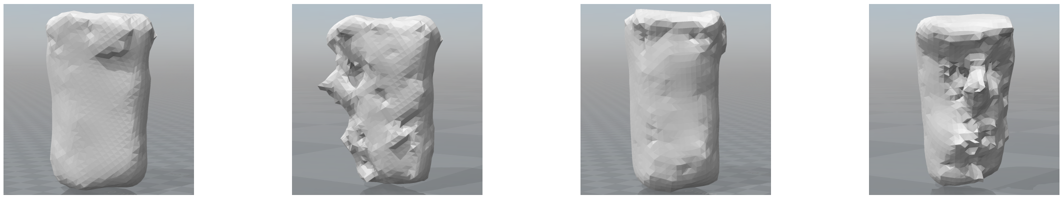
(a) Hand approach

(b) Finger contact

(c) Finger curl

Figure 13: Barrett hand showing contact with the object. The hand is first brought to the position shown in a), followed by and approach as shown in b) and then the fingers are curled towards the object to collect any additional tactile information in c).

simple shape dataset. We then used this model to complete the partial meshes of objects combined in tactile information acquired from the Barrett hand as shown in Fig. 13. This new methodology was tested on the Rubbermaid object from the YCB dataset. The network was able to correctly determine a handle on the back of the object as shown in Fig. 14. An explanatory video is available at `https://rebrand.ly/visualtactilevideo`.



(a) Depth Front

(b) Tactile Front

(c) Depth Side

(d) Tactile Side

Figure 14: a) and c) are both depth only completion which missed the handle on the reverse side of the pitcher. b) and d) however were able to recreate a handle using the tactile information from the robotic hand.

6 Conclusion

We have developed an integrated system for shape modeling and geometric reasoning based upon machine learning from large data sets of 3D models. Both visual and tactile imagery were used to create a CNN that can merge single views of objects with sparse tactile data to create accurate and complete 3D models. The models, once completed, can then be used by a grasp planner to find suitable and stable grasps. Experimental results show that using both tactile data and vision data provides more accurate completed models than using either vision or tactile data alone.

References

[1] P. K. Allen, A. Miller, B. Leibowitz, and P. Oh. Integration of vision, force and tactile sensing for grasping. *Int. Journal of Intelligent Mechatronics*, 4(1):129–149, 1999.

[2] Anonymous. We can't tell you. 1888.

[3] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I., A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio. Theano: new features and speed improvements. *arXiv:1211.5590*, 2012.

[4] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)*, volume 4. Austin, TX, 2010.

[5] A. Bierbaum, I. Gubarev, and R. Dillmann. Robust shape recovery for sparse contact location and normal data from haptic exploration. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 3200–3205. IEEE, 2008.

[6] M. Bjorkman, Y. Bekiroglu, V. Hogman, and D. Kragic. Enhancing visual perception of shape through tactile glances. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 3180–3186. IEEE, 2013.

[7] S. Caccamo, Y. Bekiroglu, C. H. Ek, and D. Kragic. Active exploration using gaussian random fields and gaussian process implicit surfaces. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 582–589. IEEE, 2016.

[8] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *Advanced Robotics (ICAR), 2015 International Conference on*, pages 510–517. IEEE, 2015.

[9] F. Chollet. Keras. `https://github.com/fchollet/keras`, 2015.

[10] S. Dragiev, M. Toussaint, and M. Gienger. Gaussian process implicit surfaces for shape estimation and grasping. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2845–2850. IEEE, 2011.

[11] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the 13th AISTATS*, pages 249–256, 2010.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv preprint arXiv:1502.01852*, 2015.

[13] A. Hermann, F. Mauch, S. Klemm, A. Roennau, and R. Dillmann. Eye in hand: Towards gpu accelerated online grasp planning based on pointclouds from in-hand sensor. In *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*, pages 1003–1009. IEEE, 2016.

[14] J. Ilonen, J. Bohg, and V. Kyrki. Fusing visual and tactile sensing for 3-d object reconstruction while grasping. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3547–3554. IEEE, 2013.

[15] J. Ilonen, J. Bohg, and V. Kyrki. Three-dimensional object reconstruction of symmetric objects by fusing visual and tactile sensing. *The International Journal of Robotics Research*, 33(2): 321–341, 2014.

[16] N. Jamali, C. Ciliberto, L. Rosasco, and L. Natale. Active perception: Building objects' models using tactile exploration. In *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*, pages 179–185. IEEE, 2016.

[17] D. Kappler, J. Bohg, and S. Schaal. Leveraging big data for grasp planning. In *ICRA*, pages 4304–4311. IEEE, 2015.

[18] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] M. Krainin, B. Curless, and D. Fox. Autonomous generation of complete 3d object models using next best view manipulation planning. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 5031–5037. IEEE, 2011.

[20] M. Krainin, P. Henry, X. Ren, and D. Fox. Manipulator and object tracking for in-hand 3d object modeling. *The International Journal of Robotics Research*, 30(11):1311–1327, 2011.

[21] J. Mahler, S. Patil, B. Kehoe, J. van den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg. GP-GPIS-OPT: Grasp planning with shape uncertainty using gaussian process implicit surfaces and sequential convex programming. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[22] A. T. Miller and P. K. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE R&A Magazine*, 11(4):110–122, 2004.

[23] P. Min. Binvox, a 3d mesh voxelizer, 2004.

[24] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[25] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *ICRA*, Shanghai, China, May 9-13 2011.

[26] N. Sommer, M. Li, and A. Billard. Bimanual compliant tactile exploration for grasping unknown objects. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 6400–6407. IEEE, 2014.

[27] I. A. Sucan and S. Chitta. Moveit! *http://moveit.ros.org*, 2013.

[28] S. Thrun and J. J. Leonard. Simultaneous localization and mapping. In *Springer handbook of robotics*, pages 871–889. Springer, 2008.

[29] O. Williams and A. Fitzgibbon. Gaussian process implicit surfaces. *Gaussian Proc. in Practice*, pages 1–4, 2007.

[30] Z. Yi, R. Calandra, F. Veiga, H. van Hoof, T. Hermans, Y. Zhang, and J. Peters. Active tactile object exploration with gaussian processes. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 4925–4930. IEEE, 2016.