Blind Grasping: Stable Robotic Grasping Using Tactile Feedback and Hand Kinematics

Hao Dang, Jonathan Weisz, and Peter K. Allen

Abstract— **We propose a machine learning approach to the perception of a stable robotic grasp based on tactile feedback and hand kinematic data, which we call *blind grasping*. We first discuss a method for simulating tactile feedback using a soft finger contact model in *GraspIt!*, which is a robotic grasping simulator [10]. Using this simulation technique, we compute tactile contacts of thousands of grasps with a robotic hand using the Columbia Grasp Database [6]. The tactile contacts along with the hand kinematic data are then input to a Support Vector Machine (SVM) which is trained to estimate the stability of a given grasp based on this tactile feedback and also the robotic hand kinematics. Experimental results indicate that the tactile feedback along with the hand kinematic data carry meaningful information for the prediction of the stability of a blind robotic grasp.**

I. Introduction

Grasp planning is a fundamental problem in the field of robotics that has been attracting an increasing number of researchers [4], [15], [9], [13], [12]. One fact about many existing grasp planning algorithms is that they all require some information about the object to be grasped, i.e. either a 2D image as required in [15], [12] or a full 3D geometric model as in [4]. This, to a large extent, limits the application of these methods across a wide range of different working situations. On the one hand, obtaining 3D geometry information is expensive and may even be impossible without certain devices such as 3D laser scanners. On the other hand, acquiring images with cameras also poses constraints on the working environment. For example, cameras work differently in different lighting conditions, even though we expect them to perform consistently across these different lighting conditions.

What if the robot is blind, by which we mean it is not able to obtain visual or geometric information of the object beforehand? Can a robotic hand still apply a stable grasp to an object without geometric or visual information? We define this robotic grasping as ***blind grasping***. An example that describes the situation could be a robotic hand reaching into a gym bag and getting an object out of it. As human beings, it is intuitive and straightforward for us to grasp objects even when we cannot see them. Lederman and Klatzky have shown [8] that humans have the ability to "blindly" recognize objects with a high degree of accuracy. Humans can also create stable grasps on unknown objects in the total absence of any visual feedback. This ability is sorely lacking with current robots that are performing grasping tasks. Using tactile sensing for object recognition has been explored by many researchers. Some recent work includes [16], [7], [11]. Using tactile exploration for grasping, Bierbaum *et al.* proposed a method to generate grasp affordances based on reconstructed faces of an object through tactile exploration [1]. In our paper, we are attempting to use non-visual tactile and kinematic feedback to predict stable grasps on an unknown object.

We take a machine learning approach to this problem. A problem with this approach is to generate datasets of reasonable size that encode tactile and kinematic information on robotic grasps. To overcome this, we use a tactile simulator on a large database of objects to be grasped. The Columbia Grasp Database (CGDB) [6] consists of over 300,000 stable grasps over 7,256 objects and for several robotic hands, including Barrett hands of different surface materials and a simulated human hand. It provides us with a pool of robotic grasp data from which we can simulate a tactile sensor system and collect useful tactile feedback. It then allows us to use this simulated tactile feedback data to learn the stability of a robotic grasp. In Section II, we describe a soft finger contact model used to simulate tactile feedback. In Section III, we describe the procedure to collect simulated tactile feedback from grasps in CGDB. In Section IV, we talk about the feature vectors we used to represent each grasp and build an SVM that accepts these feature vectors as input to predict the stability of the corresponding grasp. Experiments are described and discussed in Section V, followed by conclusions in Section VI.

II. Soft Finger Contact Model

Tactile sensors play an important role in representing the contacts between the surface of the hand and the object that are touching each other. The output of the tactile sensors around each contact is characterized by the forces applied at each sensor cell. So, a reasonable contact model that approximates the contact region and the pressure distribution is necessary for simulating a reasonable tactile feedback. Pezzementi et al. [18] used a point spread function model to simulate the response of a tactile sensor system. In our approach, we build our tactile simulation system based on a soft finger contact model proposed by Ciocarlie et al. [3]. We briefly introduce this model as follows. Interested readers please refer to the original paper for more details.

This work was funded in part by NIH BRP grant 1RO1 NS 050256-01A2 and NSF Grant IIS-0904514.

All authors are with Department of Computer Science, Columbia University, 450 Computer Science Building, 1214 Amsterdam Avenue, New York, NY, 10027, USA, {dang, jweisz, allen}@cs.columbia.edu

A. Contact Region Approximation

In *GraspIt!* each contact is initially considered as a point contact since the two bodies, i.e. the hand and the object, are assumed as rigid bodies. In the real world, however, the hand and the object in contact are actually deformable to some extent, resulting in an area in contact rather than a point. A point contact assumption then no longer holds reasonably. To simulate the contact region between the two bodies touching each other, we use a soft finger contact model as is developed in [3]. This model takes into account the local geometry and structure of the objects in contact and captures frictional effects such as coupling between tangential force and frictional torque. It locally approximates the surfaces of the two touching bodies as

$$z_i = A_i x^2 + B_i y^2 + C_i xy, \quad i \in \{1, 2\} \tag{1}$$

where the local contact coordinate system has its origin at the center of the contact and the z axis aligned with the contact normal. The subscript i distinguishes the contacting bodies from each other.

The seperation between the two surfaces h is

$$h = (A_1 - A_2)x^2 + (B_1 - B_2)y^2 + (C_1 - C_2)xy \tag{2}$$

By choosing the orientation of the x and y axes so that the term in xy vanishes, we can end up with the separation h between the two surfaces in the form of

$$h = \frac{1}{2R'}x^2 + \frac{1}{2R''}y^2 \tag{3}$$

where R' and R'' are the relative radii of curvature of the objects in contact, depending only on their local geometry.

B. Pressure Distribution

After a contact region is determined, we consider how the forces are formed within the contact region so that the response of the corresponding tactile sensor cells can be analyzed and evaluated. To express the pressure distribution inside a contact region using non-planar models that take into account the local geometry of the objects involved, we choose a Hertzian model as used in [3]. In this model, the ratio of frictional torque to contact load which is used to compute the eccentricity parameter of the friction ellipsoid can be obtained from

$$\frac{max(\tau_n)}{P} = \frac{3\pi}{16}\mu\sqrt{ab} \tag{4}$$

where μ is the frictional coefficient, τ_n is a frictional moment about the contact normal, P is the contact load, and a and b are the lengths of the semi-axes.

III. Tactile Sensor Simulation

A. Tactile Sensor Configuration of a Barrett Hand

The robotic hand used in the tactile simulation experiment is a Barrett hand with tactile sensors attached on the surfaces of the palm, the mid-digit links and the fingertip links. The PPS RoboTouch system [14], which is widely used on Barrett hands, was simulated. The tactile sensor system contains seven major sensor pads which cover many possible hand-object contact points of the Barrett hand. Each pad has 22 to 24 tactile sensor cells which results in a 162-cell sensor system. The configuration of these sensor pads and sensor cell arrangement is described in Table I. Figure 1 shows a real PPS RoboTouch system attached to a Barrett hand.

(a) Palm (b) Finger

Fig. 1. Tactile Sensor Configuration (from PPS spec sheet [14])

TABLE I
CONFIGURATION OF SENSOR PADS ON A BARRETT HAND

Location	Num. of Cells	Res. (mm)	Grid
Palm	24	10 × 10	4 × 6
Mid-Digit	24	6 × 6	8 × 3
Fingertip	22	6 × 6	7 × 3 + 1 tip

Algorithm 1: Computing tactile feedback

Input: A robotic grasp with a list of point contacts between the hand and the object
Output: Seven 2D arrays that carry the simulated tactile sensor values of the corresponding sensor cells

```
Initialize the output tactile sensor cell arrays to zero's
foreach point contact do
    Calculate the relative radii at the contact
    Calculate the contact region
    Discretize the contact region to 10 × 10 sub-regions
    Calculate the forces within each discretized
    sub-region according to the pressure distribution
    foreach discretized contact sub-region do
        foreach tactile sensor cell do
            if the discretized contact sub-region
            overlays on the tactile sensor cell
            then
                Accumulate the force of the discretized
                contact sub-region onto the overlaying
                tactile sensor cell
            end
        end
    end
end
Return the sensor cell arrays
```

B. Generating Tactile Feedback from the CGDB

Based on the soft finger contact model, we can compute the contact region for a hand-object contact as well as the pressure distribution within the contact region. Since a tactile

sensor cell performs as an atomic sensing unit, we discretize the soft finger contact region so that we can accumulate the total forces within each discrete part and use this to compute the forces sensed on each corresponding tactile sensor cell. We summarize the procedure to generate the tactile feedback of a robotic grasp in Algorithm 1. Figure 2 shows an example of a simulated tactile feedback.

IV. Learning the Stability of a Robotic Grasp

Following the explanation of the simulation process, we now discuss the way we built an SVM classifier that predicts the stability of a given robotic grasp. Once trained, this classifier can be used efficiently to estimate the stability of a robotic grasp without even knowing the geometric or visual information about the object to be grasped.

A. Grasp Dataset

Our grasp data is from the CGDB database. This database contains hundreds of thousands of grasps constructed from several robotic hands and thousands of object models.

Object models used in the CGDB are from the Princeton Shape Benchmark (PSB) [17]. The PSB provides a repository of 3D models which span across many objects that we encounter everyday. One fact about the PSB model set is that the models were not originally selected with an eye towards robotic grasping, and so some of the models are not obvious choices for grasping experiments. For example, the model set contains insects, which are often outside our everyday grasping range. Although the CGDB provides grasps for all these object models in the PSB, instead of using the full set of grasps in the CGDB, we choose to select grasps computed on a smaller set of objects that are more frequently grasped and manipulated by us in our everyday life. In total, we collected about 15,000 robotic grasps from 936 objects across 23 different classes.

B. Labeling Grasps

In the CGDB, all of the grasps are good in terms of their physical properties, i.e. they either have good Ferrari-Canny grasp metric volume qualities or good epsilon qualities [5]. The epsilon quality, ϵ, measures the minimum relative magnitude of the outside disturbances that could destroy the grasp. So, when we take into account the limit of the maximum forces a robotic hand can apply, a grasp would be less stable if it has a smaller epsilon quality. This is because the smaller epsilon quality indicates that a relatively smaller outside disturbance can break this grasp even when the robotic hand has already applied the maximum forces it supports.

Another consideration is from the perspective of the environment uncertainty. Due to the uncertainty of the environment, objects may move away slightly from their original position during a grasp execution. A fragile grasp may fail to fully grasp the object in this situation while a stable one may display its robustness and still succeed in grasping the object in the perturbation. We have experimentally found a strong correlation between this robustness and the epsilon quality. We have found that grasps with epsilon quality $\epsilon > 0.07$ tend to be more robust in uncertain object perturbations.

Based on the above two considerations, for grasps that are form-closure, they may also differ from each other in the sense of being more stable or less stable. So, we treat those more stable grasps as good grasps and the less stable ones as bad grasps. Using a threshold $t_\epsilon = 0.07$ as the boundary, we label $grasp_i$ as a good (1) or a bad (0) grasp as follows,

$$label(grasp_i) = \begin{cases} 0 & \text{if} \quad \epsilon(grasp_i) \leq t_\epsilon \\ 1 & \text{if} \quad otherwise \end{cases} \tag{5}$$

C. Feature Vector

For a human grasp, two properties are usually perceived by us. One is the tactile sensing which specifies the contact configuration between our hands and the object. The other is the hand kinematics which indicates how our hands are shaped around the object for the manipulation. These two pieces of information help us predict whether this is a stable grasp. In the robot domain, we use the same idea to synthesize a feature vector for a robotic grasp: we use tactile and kinematic information to characterize a grasp.

Given a robotic grasp, the output of the simulated tactile sensor system is a group of seven 2D arrays. Each array is corresponding to one sensor pad. Each element of an array stores the value of the force sensed in the corresponding sensor cell. We vectorize each of these 2D arrays into one-dimensional vectors and concatenate them together. This gives us a 162-dimensional vector. In addition to the tactile feedback, we also obtain the values of the seven joint angles of a Barrett hand when a grasp is applied and append them to the end of the 162-dimensional tactile feedback vector. This makes the final feature vector 169-dimensional.

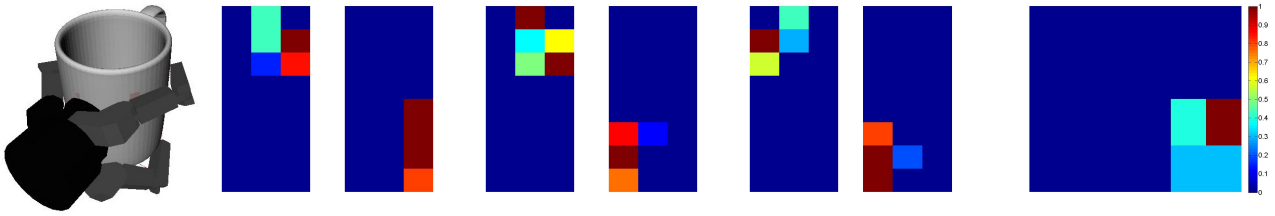
D. Scaling

A feature vector contains both the 162-dimensional tactile feedback and the 7-dimensional joint values. They are from two totally different domains and of two different dimensions. Instead of feeding them directly into a training algorithm which will treat each dimension in both the tactile feedback and the joint angle equally, we scale each dimension of the tactile data and the joint data differently so that these two major parts can be weighted equally on the whole.

For each tactile feedback T_i, we first normalize the 162-dimensional tactile feedback such that they sum up to one. Specifically we write it down as follows,

$$T_i = \frac{T_i}{\sum_{j=1}^{162} T_i^j} \tag{6}$$

Then, for a feature vector, the scaling approach we used is to scale the dimensions of the tactile part such that the standard deviation across all the samples is one while to scale the dimensions of the joint angle part such that the standard deviation is w, where w is considered as a weight factor that balances the weights between the tactile feedback



(a) Grasp (b) F1 (c) F2 (d) F3 (e) Palm

Fig. 2. Tactile Sensor Simulation, Figure 2(a) show a robotic grasp of a Barrett hand on a mug, Figure 2(b), 2(c), and 2(d) show the tactile responses on finger F1, F2, F3 respectively. In each group, the left one is the tactile pad on the finger tip and the right one is the tactile pad on the mid finger. Figure 2(e) shows the tactile readings on the palm. The 8-bit scale (0 - 255) goes from blue (no response) to red (saturation).

and the joint angle values. More specifically, a feature vector $x = [T \quad J], T \in \mathcal{R}^{162}, J \in \mathcal{R}^{7}$ is scaled as follows

$$x'^{\,i} = \begin{cases} \frac{x^i}{std^i} & \text{if} \quad 1 \leq i \leq 162 \\ w\frac{x^i}{std^i} & \text{if} \quad 163 \leq i \leq 169 \end{cases} \tag{7}$$

where x^i denotes the i^{th} dimension of the feature vector. std^i denotes the standard deviation in the i^{th} dimension. In our scaling process, we used the same std^i computed from the training set to scale the test data.

V. Experiments and Results

A. Training and Testing Dataset

Tactile feedback is a core component in our feature vector. Inside *GraspIt!*, the physical simulation system is able to capture all the possible contacts between a robotic hand and the object. Although the current PPS tactile sensor system captures many possible hand-object contacts, there are still locations the system does not cover. For example, Figure 3 shows a grasp when a Barrett hand is grasping a wrench. The contacts on the edges of the fingertips, F1 and F2, cannot be captured by the current tactile sensor system. However, it is detected and considered in the physical simulation system inside *GraspIt!* for the computation of the epsilon quality. For such kind of grasps which involve contacts in uncovered regions, they cannot be fully represented by the tactile sensor system, either the simulated or the real one. Therefore, there is an inconsistency in these grasps between the physical simulation and the real world and this inconsistency may influence the performance of the classification.

Since grasps with fewer number of non-zero tactile responses have more potential to contain un-captured contacts, we first filter out grasps from the grasp dataset, $\mathcal{D}$, obtained in Section IV-A based on the number of sensor pads that have non-zero responses. Since each sensor pad is a flat plane, a stable grasp must have at least two sensor pads in contact with the object being grasped, resulting in at least two sensor pads with non-zero responses. In our experiment, we first divide the grasp dataset $\mathcal{D}$ into two subsets, $\mathcal{D}_1$ and $\mathcal{D}_2$. $\mathcal{D}_1$ contains all the grasps that have only one tactile sensor pad with non-zero responses, while $\mathcal{D}_2$ contains all the grasps with at least two tactile sensor pads having non-zero responses.

TABLE II

Test Accuracy on 5641 Grasps (%)

Classification Accuracy	False Neg. Predct.	False Pos. Predct.
70.4	19.4	10.2

We then choose $2/3$ grasps from $\mathcal{D}_2$ that evenly distributed among all the objects to generate our training set. We put the remaining $1/3$ of the grasps from $\mathcal{D}_2$ into the test set. Although contacts of grasps in $\mathcal{D}_1$ are not fully captured by the simulated tactile system, their tactile feedback would be realistic considering a real Barrett hand with a real tactile system. In order to keep this potential considered as in a real working environment, we do not want to rule them out for testing. Thus, we put all the grasps from $\mathcal{D}_1$ in the testing set, and the test set is a union of $\mathcal{D}_1$ and $1/3$ of $\mathcal{D}_2$.

B. Results

During scaling, we tried different w's. $w = \sqrt{\frac{162}{7}}$ worked best in our experiment. To train an SVM classifier, we used libsvm [2]. We used a RBF kernel with 5-fold cross validation to determine the best cost parameter C and RBF parameter γ.

After we trained our SVM classifier, we fed the grasps in the testing set to the classifier. Experiment results are shown in Table II. In total, the best overall accuracy we get is 70.38% with $w = \sqrt{\frac{162}{7}}$. In this experiment, only 10.2% grasps are incorrectly classified as positive. This means that a grasp will be classified incorrectly as a good one with only a small probability. It also indicates the conservativeness of this classifier which is very necessary in a blind grasping context. Because usually we can just keep searching for another grasp if we mis-classify a good grasp to be a bad one, but the cost or risk is usually too high to tolerate if we execute a grasp that is actually bad but mis-classified as a good one.

In Table III, we show more detailed statistics on the accuracy in each object class based on our best result. In the context of blind grasping, the false positive predictions are more important in our consideration because of the high cost we have to pay for a false positive prediction. We summarized the percentages of false positive predictions of each object class.

In Figure 4, 5, and 6, we show some example test grasps. Based on these test results, we find some interesting points.

For grasps whose contacts are fully captured, they have more potential to be classified correctly. Figure 4 shows an example grasp whose contacts are fully captured by the tactile sensor system. This is an example grasp that is classified correctly. In contrast, Figure 5 is a false negative prediction. As is shown in the tactile sensor response, the contacts on the fingers are not fully captured by the current tactile sensor system. This makes the tactile feedback incapable to carry enough information to represent the grasp. Thus, losing tactile information increases the potential of a false prediction.

Contacts on the edges of the finger may confuse the classifier and result in more false predictions. Figure 6 shows a false positive example. Although all the tactile pads have non-zero responses, many of the contacts are on the edges of the fingers. In this situation, the contact normals on the object surface differ dramatically from that of the surface of the finger. But a tactile sensor can only record the normal forces leaving the tangential forces un-captured. The tactile sensor representation does not fully capture the grasp feature.

When contacts are not fully captured, hand kinematics may confuse the classifier. As we can see in Figure 6, two of the three fingers of the Barrett hand are shaped towards the other one. In general, a grasp with a set of contacts that are facing each other is more likely to be a good one and having fingers facing towards each other increases the potential of such contacts. In this sense, without further distinction obtained from the tactile sensor feedback, the hand kinematics may confuse the classifier to make it consider this grasp as a good one.

VI. Conclusion

In this paper, we proposed a method to utilize tactile feedback to predict the stability of a robotic grasp. Different from most current grasp planning related methods, our approach does not require either the geometric or the visual information of the object to be grasped. This approach could be applied alone in working environments where geometric or visual data is not obtainable, such as blind grasping as mentioned in Section I. Or it can work with other grasp planning algorithms as an enhancement to boost their overall performance.

We tested our algorithm on the simulated tactile feedback from grasps on a small set of objects that are frequently grasped and manipulated in our everyday life. The experiments validated the capability of our method to distinguish stable grasps from non-stable ones. One feature worth noting is that this method preserves a small false positive percentage which makes it useful when we need to avoid taking potentially large risk in a false positive grasp prediction.

For the next step, we are planning to do experiments on a real robotic hand with a real tactile sensor system. Since our current feature vector only contains the raw output from the tactile sensor system and the hand kinematics, some properties of a grasp might not be fully displayed directly from this raw data, and different weights between tactile feedback and hand kinematics may influence the prediction process in different ways. We will be exploring more sophisticated features as well as the weights between data from different domains to increase the prediction performance. Due to the flexibility of the simulation system, we will also simulate different patterns of a tactile sensor system. Then, we will examine different tactile configurations and see how they could improve the prediction performances. This may give us more insights into a better design of a tactile sensor system. To further utilize this new approach, we are planning to integrate this grasping perception method into *GraspIt!* so that it can be used as a metric in grasp planning, which could result in a more efficient planning method.

References

[1] Er Bierbaum, M. Rambow, T. Asfour, and R. Dillmann, *Grasp affordances from multi-fingered tactile exploration using dynamic potential fields*, Humanoids, IEEE Intl. Conference on, 2009, pp. 168 –174.

[2] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM: a library for support vector machines*, 2001, Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[3] Matei Ciocarlie, Claire Lackner, and Peter Allen, *Soft finger model with adaptive contact geometry for grasping and manipulation tasks*, World Haptics Conference (2007), 219–224.

[4] Matei T. Ciocarlie and Peter K. Allen, *Hand Posture Subspaces for Dexterous Robotic Grasping*, The International Journal of Robotics Research **28** (2009), no. 7, 851–867.

[5] C. Ferrari and J. Canny, *Planning optimal grasps*, Robotics and Automation, IEEE Intl. Conference on, May 1992, pp. 2290 –2295.

[6] C. Goldfeder, M. Ciocarlie, H. Dang, and P.K. Allen, *The columbia grasp database*, Robotics and Automation, IEEE International Conference on, May 2009, pp. 1710 –1716.

[7] Nicolas Gorges, Stefan Escaida Navarro, Dirk Göger, and Heinz Wörn, *Haptic object recognition using passive joints and haptic key features*, Robotics and Automation, IEEE International Conference on, May 2010, pp. 2349 –2355.

[8] Susan J. Lederman and Roberta L. Klatzky, *Hand movements: A window into haptic object recognition*, Cognitive Psychology **19** (1987), no. 3, 342 – 368.

[9] Zexiang Li and S. Sastry, *Task oriented optimal grasping by multi-fingered robot hands*, Robotics and Automation, IEEE International Conference on, 1987, pp. 389–394.

[10] A. T. Miller and P. K. Allen, *Graspit! a versatile simulator for robotic grasping*, Robotics & Automation **11** (2004), no. 4, 110–122.

[11] A. Petrovskaya, O. Khatib, S. Thrun, and A.Y. Ng, *Bayesian estimation for autonomous object manipulation based on tactile sensors*, Robotics and Automation, IEEE International Conference on, may. 2006, pp. 707 –714.

[12] Mila Popovic, Dirk Kraft, Leon Bodenhagen, Emre Baseski, Nicolas Pugeault, Danica Kragic, Tamim Asfour, and Norbert Krüger, *A strategy for grasping unknown objects based on co-planarity and colour information*, Robotics and Autonomous Systems **58** (2010), no. 5, 551 – 565.

[13] M. Prats, P.J. Sanz, and A.P. del Pobil, *Task-oriented grasping using hand preshapes and task frames*, Robotics and Automation, IEEE International Conference on, april 2007, pp. 1794 –1799.

[14] Pressure Profile, *Robotouch*, `http://www.pressureprofile.com/products-robotouch`.

[15] Ashutosh Saxena, Justin Driemeyer, Justin Kearns, and Andrew Y. Ng, *Robotic grasping of novel objects*, Advances in Neural Information Processing Systems 19 (B. Schölkopf, J. Platt, and T. Hoffman, eds.), MIT Press, Cambridge, MA, 2007, pp. 1209–1216.

[16] Alexander Schneider, Jürgen Sturm, Cyrill Stachniss, Marco Reisert, Hans Burkhardt, and Wolfram Burgard, *Object identification with tactile sensors using bag-of-features*, IEEE/RSJ international conference on Intelligent robots and systems, 2009, pp. 243–248.

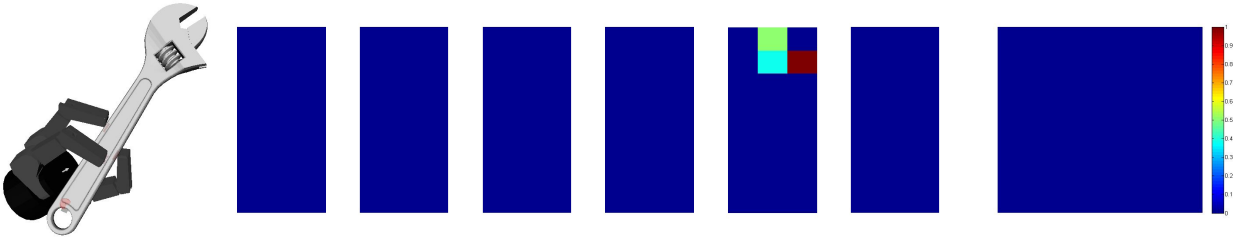
[17] Philip Shilane, Patrick Min, Michael Kazhdan, and Thomas Funkhouser, *The princeton shape benchmark*, In Shape Modeling International, 2004, pp. 167–178.

[18] Zachary Pezzementi, Erica Jantho, Lucas Estrade, Gregory D. Hager, *Characterization and Simulation of Tactile Sensors*, Haptics Symposium, March 2010.

TABLE III

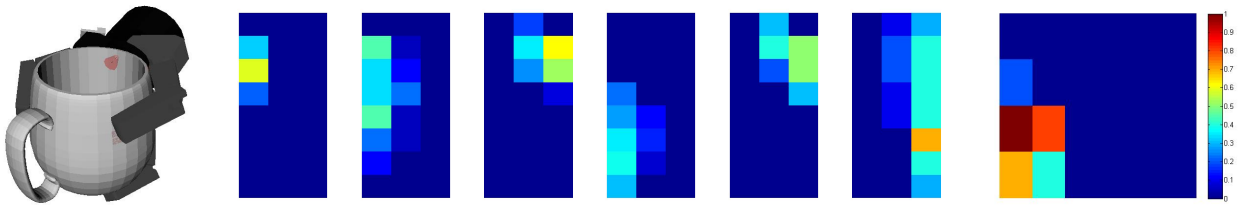
CLASSIFICATION ACCURACY (EACH LEFT COLUMN) AND PERCENTAGES OF FALSE POSITIVE PREDICTIONS ON GRASPS OF EACH OBJECT CLASS(%).

axe		book		bottle		butcher knife		gear		wine glass		guitar		hammer	
85.11	10.63	57.57	6.81	63.90	8.57	71.18	2.54	63.05	12.85	72.54	5.73	79.04	18.56	69.90	9.70
handgun		helmet		ice cream cone		knife		lamp		microscope		mug		phone handle	
62.28	9.29	68.72	12.02	57.49	5.81	88	6.85	65.83	10.30	69.65	9.65	82.29	9.37	50.84	15.25
axe		screw driver		shovel		skateboard		sword		vase		wrench		-	
76.17	11.06	90.97	3.75	71.27	15.95	54.42	5.44	82.69	15.06	75.75	10.77	65.45	9.09	-	-



(a) Grasp (b) F1 (c) F2 (d) F3 (e) Palm

Fig. 3. A robotic grasp whose contacts are not fully captured by the current tactile sensor syetem. Figure 3(b) to 3(e) show the tactile sensor outputs in each of the seven sensor pads. Only contacts on the tip of the Finger3 are captured in the system as shown in 3(d). Contacts that are on the other two fingers, Finger1 and Finger2, are not captured by the tactile system.

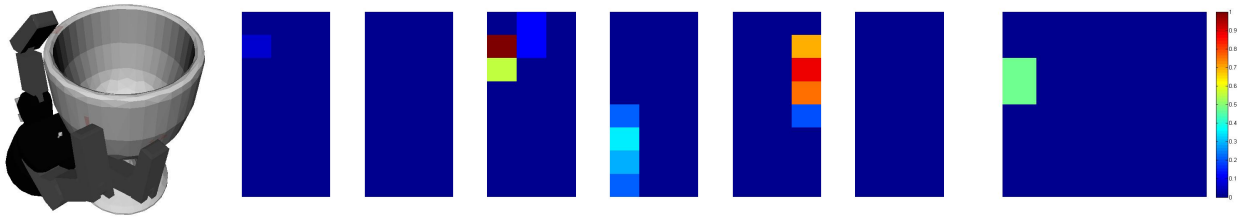


(a) Grasp (b) F1 (c) F2 (d) F3 (e) Palm

Fig. 4. A good robotic grasp which is classified correctly. Figure 4(b) to 4(e) show the tactile sensor outputs in each of the seven sensor pads. All of those contacts are captured by the seven tactile sensor pads.

(a) Grasp (b) F1 (c) F2 (d) F3 (e) Palm

Fig. 5. False negative: A good robotic grasp which is classified incorrectly as a bad one. Figure 5(b) to 5(e) show the tactile sensor outputs in each of the seven sensor pads. The problem here is that the contacts on the edges of the fingers are not fully captured.



(a) Grasp (b) F1 (c) F2 (d) F3 (e) Palm

Fig. 6. False positive: A bad robotic grasp which is classified incorrectly as a good one. Figure 6(b) to 6(e) show the tactile sensor outputs in each of the seven sensor pads. All of those contacts are captured by the seven tactile sensor pads. However, they are on the edges of the fingers and the dramatic surface normal differences between the object and the robotic hand at the contact location may confuse the classifier.