Object Placement using the Adept Arm

Kevin Yang, Michael Lyons, and Paul Kiernan, *Cornell University*

***Abstract*— In this paper, we focus on the use of the Adept Arm Robot for determining the proper placement of a number of objects by using point cloud data from a Microsoft Kinect mounted on the arm. In this work, we also present a significant data set consisting of 9585 labels across 35 objects placed on 6 flat surfaces. This work also looks at 3711 labels across 4-35 objects on 3 non-flat surfaces. Building SVM models from features extracted from these labels, we were able to attain performances in excess of 80% for both precision and recall for our test sets for both flat and non-flat surfaces. In our robot experimentation, we demonstrated the usefulness of our best-performing model by having the robot place objects on a number of surfaces.**

I. Introduction

Object placement plays a crucial role in many applications. Stable object placement ensures that an object is undamaged and remains at the same location, making future localization of this object easier. However, it remains a difficult task for robots because stable placements of objects are not always clear. Robots often lack the kinds of a priori knowledge of objects that humans usually have. This knowledge ranges from previous experience of how similar objects should be placed, material properties of the object which may affect its stability, to knowledge of the role that physics would play on certain objects. Furthermore, robots often lack the fine tactile feedback that humans can use to determine if a placed object is stable and not leaning or rotating.

In this paper, we consider the particular challenge of object placement of a number of known objects on a variety of surfaces. We focused on the construction of a significant dataset as well as determining which SVM models most accurately predict good and bad placements. We also compared our models with a synthetic dataset as a benchmark as part of our offline experiments.

II. Hardware Platform

For our project, we are using an Adept Viper s850 arm with a gripper installed. We used a robot-mounted Microsoft Kinect to obtain object point cloud data. Since the gripper does not have tactile feedback, all information regarding the initial grip of an item and its orientation must be given beforehand, usually hard coded. It could also lead to unstable placement of objects if the object shifts while being held. Our project uses Robot Operating System (ROS)'s Diamondback distribution running on an installation of Ubuntu 10.10 (Maverick Meerkat).

Adept Viper s850 arm with Gripper and mounted Microsoft Kinect

III. Approach

A. Data

For our project, we needed a large dataset of labels of good and bad object placements in order to train our model so that it would be able to make a good hypothesis regarding stable object placements. However, we found that there were no significantly large datasets that would be suitable for our particular task. This proved problematic as our initial work relied on relatively small sets that we had gathered over a limited number of objects-surface combinations. These datasets proved to be of insufficient size to produce a good model. Furthermore, as our initial placement labeling had many more bad placements than good placements, our results were affected in that our model believed that any object-surface placement was likely to be bad. Since our combination of features also varied significantly in magnitude, lack of feature normalization likely played a role.

Thus, we realized the need to obtain point cloud data for a larger set of objects and surfaces to create a dataset containing a range of good to bad placements. This was accomplished by using a preexisting script to capture point cloud data from a variety of angles using the Adept arm. We cleaned this point cloud data and combined them in order to create a more complete model of the object. Additionally, when labeling, we also made sure to have a sufficient

The authors are students with Cornell University, Department of Electrical and Computer Engineering, Ithaca, NY 14850 USA. They are enrolled in ECE/CS 4758: Robot Learning for the Spring 2011 semester.
{ky238, mpl56, pak79} @ cornell.edu

number of good placements in our data.

1) *Flat Surfaces*

We gathered point cloud data and labeling information for a variety of object and surfaces (Table 1). The first 20 object-environment combinations have around 20 labels each; later ones had around 50-80 labels each.

TABLE I
OBJECT AND FLAT SURFACES IN DATASET

Object	
Air Pump	Purple Barbell Squeaky Toy
Yellow Martini Glass	Orange Highlighter
Green Mug	Tissue box
Black Bowl	Soap Dispenser
Orange Hippo	Rubik's Cube
Dish Rack	3-hole punch
Red Plastic Cup	Stapler
Foam Stack	Eyeglass Case
Orange Cone	Hairspray
Pink Bee Plate	Toilet Paper Roll
Light Blue Barbell	Vitamin Bottle
Rabbit Cup	Blue water bottle
Black Bookend	Computer Mouse
Purple Seal Toy	Headphones
Duct Tape	Binder
Black Plastic Holder	Graphing Calculator
Green Ethernet Cable	Travel Mug
Rovio Robot	Tissue box
Environment	
Aerial Robot Box	Rovio Box
Air Robot Case	Kinect Box
Striped Padded Surface	Ground
Open Box	

The placements were manually labeled with a value between 0 and 10, with 10 being a perfect or close to perfect placement and 0 being a poor/impossible placement.

We defined good placements as placements where the object would remain stable in the environment. This means that the object would not fall once released, would remain on top of the surface, and should remain in its placed position. We define bad placements as placements where the object intersects the environment (impossible placements), would fall or tip onto the environment, or would not remain on top of the surface.

Although the labels contained values between 0 and 10 for their class, for this project, we treated them as a binary classification. Placement scores above 5 would be considered good and those 5 and below would be considered poor. Intermediate placements of 3, 5, or 7 were used depending on the shape of the objects. The favorability of placements was based on the human perception of good placement. For example the martini glass would have a placement of 10 if it was right side up on its base. A placement of 7 would be used if the martini glass were upside down on its head. A rating of 3 would be used to dictate the placement of glass on its rounded side showing a semi-favorable position with a reasonable amount of instability. Finally a 0 would be used if the glass was placed on the tip of its base or head or with part of the glass in the surface. With the variety of objects we used the labeling varied depending on the various positions deemed favorable via human perception and stability.

The dataset as a whole contains 9585 labels, distributed between 2536 good and 7049 bad placements.

2) *Non-Flat Surfaces*

In addition to creating a large data set for flat surfaces, the problem of placing objects on more complex surfaces was also pursued. Not wanting to weaken placement performance on any one type of surface, different models were created for different types of surfaces in the hopes of increasing ideal placements for each different type of surface. The first non-flat surface that was modeled was a Wooden V-shaped or Slanted Dish Rack in which dishes are placed vertically. Unfortunately, within our wide array of objects, only a few could be placed on this surface. The object surface combination can be seen in the table below in Table II.

A similar methodology was used in creating the labels for the wooden dish rack but the ideal positioning was one where vertically placed objects were ideal - which is very much counter to the original flat surface labeling data. The most ideal positions with a score of 10 would be placed in between the thin wooden slits being well supported in the dish rack. Poor placements of 5 and below were ones in which the object would fall into a random spot in the V-shape of the dish rack. For the wooden dish rack the data of 369 labels was split between 179 good placements and 190 bad placements.

TABLE II
OBJECT AND WOODEN DISH RACK

Object	
Yellow Martini Glass	Black Bowl
Green Mug	Pink Bee Plate
Environment	
Wooden Slanted Dish Rack	

The second non-flat surface that was modeled was a gray studded dish rack common in most dining halls and large kitchens. The flatness of the rack allowed for more objects to be placed inside of it but difficulties were encountered with large and awkwardly shaped objects as they did not fit between the studs. The object surface combinations can be seen below in Table III.

Creating labels for the gray studded dish also had most ideal positions being vertical placements in between the studs. Some objects could be place similarly to the flat surface data in between the studs due to the objects small size. The most ideal positions with a score of 10 were objects that were well supported by the studs. Poor placements of 5 and below were used for placements on top of the studs or placements in which the object would fall or roll out of position. Bad placements were quite numerous in this data. For the gray studded dish rack the data of 1532 labels was split between 328 good placements and 1204 bad placements.

TABLE III
OBJECT AND GRAY STUDDED DISH RACK

Object	
Air Pump	Purple Barbell Squeaky Toy
Yellow Martini Glass	Orange Highlighter
Green Mug	Tissue box
Black Bowl	Soap Dispenser
Red Plastic Cup	Stapler
Foam Stack	Eyeglass Case
Orange Cone	Hairspray
Pink Bee Plate	Vitamin Bottle
Rabbit Cup	Binder
Black Bookend	Graphing Calculator
Purple Seal Toy	Tissue box
Green Ethernet Cable	
Environment	
Gray Studded Dish Rack	

The final non-flat surface is inside a small open cardboard box ("Amazon box"). While it is flat, it makes for an interesting placement scheme. In this case, all objects could be placed inside the box except for a few. What makes the placement inside a box so interesting is avoidance of the walls during placement. The object box combinations can be seen above in Table III and it can be seen that all objects are compatible with it.

Good placements, those with label of 10, were ones in which the object was placed in the center of the box. Bad placements in this case were numerous as with the gray dish rack. Bad placements included against the walls, on top of the walls, on the top flaps of the box, and outside the box. For Amazon Box, the dataset of 1756 labels was split between 184 good placements and 1572 bad placements.

B. Features

We explored a number of features of this placement dataset that would be able to accurately predict the placement of the object. These features gave mixed results.

Object Centroid to Environment Distance: This feature attempted to capture the idea that objects which are stable should minimize the distance between its centroid and the environment. Unstable objects, such as tipped objects and objects that would fall would have higher distances. However, we realized that this would result in the classifier preferring non-ideal good placements over ideal good placements. For example, an inverted martini glass would have a shorter object centroid to environment distance than if it were placed properly. While both are good placements in that they are stable, the inverted martini glass is considered a less favorable placement.

Sliding Window: This feature generated a box around the point cloud data of the object and subdivided this box into a number of user defined grid cells. It would then count the number of points within each grid cell. This attempted to capture the idea that stable object placements would have more points along the bottom grid cells and a lower number of them for a certain placement would indicate instability. However, we found that this feature did not result in good performance. Possible reasons are the incomplete nature of the collected point cloud data. For the cups, not as many points that can be seen on the cup bottoms, so that even with good placements, not a lot of points are in the lower grid cells.

Supporting Contacts: This feature finds the number of supporting contact points between the environment and the object. Objects that have more supporting contacts with the environment should be more stable since objects that are tipped to the side or hovering over the environment would have less supporting contacts. This feature is good because it gives a second level of stability on top of our favorable human perceived placements.

Histogram: This feature is a histogram of the points in the object point cloud in relation to the lowest point on the object. This captures similar data to the sliding window, although instead of being separated by grid cell, it is separated in two different dimensions. This feature gives a good distinction between object shapes, which is critical in stable object placement.

We found that supporting contacts and histogram features gave the best results when used to train the support vector machine. Thus, we used those features in further experimentation.

C. Evaluation

1) Flat surface – good features

Using both preexisting and our own feature generators, we generated a feature set from this dataset. We then tested the resulting set using 5-fold cross-validation. By adjusting various parameters of SVM-Light, an implementation of the support vector machine, we tested 605 different combinations of kernel and learning options for the classifier [2]. The kernel options used were: linear, polynomial, radial bias, sigmoid hyperbolic tangent, and as well hyperbolic tangent kernel functions. In addition to trying these kernel functions, we also adjusted their parameters to improve performance. The learning options used were: classification, regression, preference ranking, trade between training error and margin, epsilon width of tube for regression, cost-factor by which training errors on positive examples outweigh errors on positive examples, using biased hyperplanes instead of unbiased hyperplanes, and removing inconsistent training examples. We eventually narrowed down our focus to the models that gave the best recall, precision, and F_1 metrics. These metrics are defined in (1), (2), and (3) respectively.

$$Recall = \frac{TP}{TP + FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (3)$$

The average metrics of the best performing models and various parameters are in Table IV for flat surfaces.

TABLE IV
BEST PERFORMING MODELS FOR FLAT SURFACES

Model Parameters	Precision	Recall	F_1
Polynomial	0.822	0.696	0.754
Polynomial, 0.1 trade-off b/w training & margin	0.817	0.756	0.786
Polynomial, Biased Hyperplane	0.788	0.875	0.829
Radial Basis, Biased Hyperplane	**0.852**	**0.946**	**0.897**

Metrics obtained by averaging the results from the 5-fold cross-validation.

ROC curves, or receiver operating characteristic curves, are a common tool used to evaluate learning algorithms. The ROC curves for the default SVM, 0.1 trade-off SVM, the biased hyperplane SVM, and the biased hyperplane with radial basis kernel SVM can be seen in figures (1), (2), (3), and (4), respectively. The four ROC curves plotted against each other for evaluation can be seen in (Figure 5).

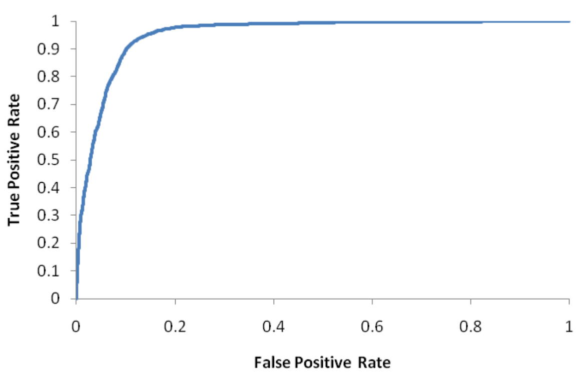

Figure 1. Default SVM Model using a Polynomial Kernel Function for flat surfaces

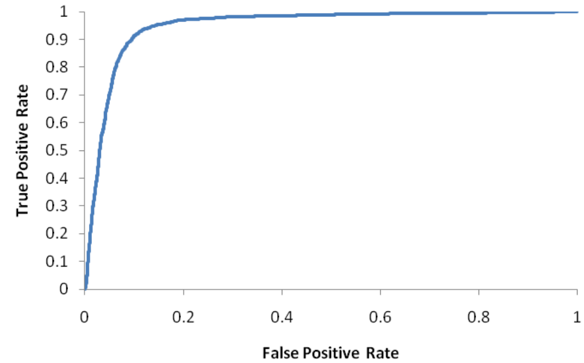

Figure 2. SVM Model using a 0.1 trade-off between training and margin and a polynomial kernel function for flat surfaces

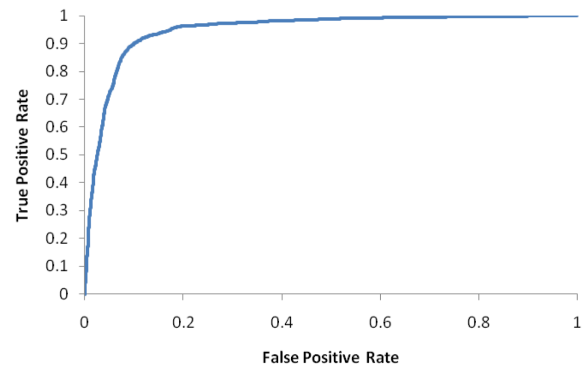

Figure 3. SVM Model using a biased hyperplane and a polynomial kernel function for flat surfaces

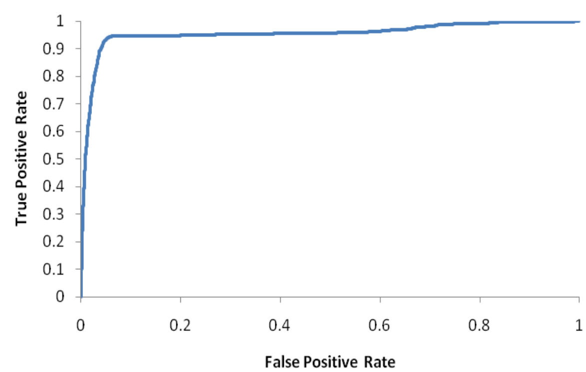

Figure 4. SVM Model using a biased hyperplane and a radial basis kernel function for flat surfaces

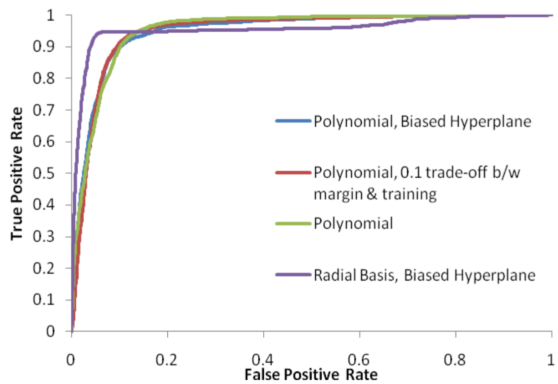

Figure 5. Comparison of ROC Curves (from Figures 1, 2, 3 and 4) for flat surfaces

We see from (Figure 5) that the three models using the polynomial kernel function exhibit very similar ROC curves, while the radial basis kernel function with biased hyperplane displays significantly better true positive rates for lower false positive rates, and thus, has a slightly higher AUC (Table V). The best model found was the Radial Basis kernel with Biased Hyperplane.

TABLE V
AREA UNDER CURVE FLAT SURFACE MODEL COMPARISON

Model Parameters	Area Under Curve
Polynomial	0.952
Polynomial, 0.1 trade-off b/w training & margin	0.948
Polynomial, Biased Hyperplane	0.948
Radial Basis, Biased Hyperplane	**0.954**

Metrics obtained by averaging the results from the 5-fold cross-validation.

2) *Non-flat Surfaces*

After extensively modeling the flat surface data it became time to view the non-flat surfaces performance as well. Support Vector Machine modeling was also used and a bit of experimentation was used to figure out the ideal models for each non-flat surface. For the wooden slanted dish rack two good models were found, which were Biased Hyperplane with the Radial Basis kernel and Polynomial kernel. The results can be seen in Table VI.

TABLE VI
BEST PERFORMING MODELS FOR WOODEN SLANTED DISH RACK

Model Parameters	Precision	Recall	F_1
Polynomial	0.875	1.00	0.966
Radial Basis, Biased Hyperplane	**0.922**	**0.994**	**0.961**

Metrics obtained by averaging the results from the 5-fold cross-validation.

The respective ROC curves for these models can be seen in Figure 6.

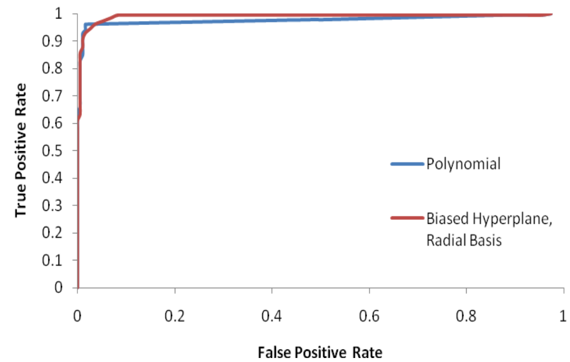

Figure 6: Comparison of ROC Curves for Wooden Slanted Dish Rack

The gray studded dish rack was then sent into the modeling and was also used with the Support Vector Machine. The best models found for this surface were the polynomial kernel and the radial basis kernel paired off with the biased hyperplane. The results of precision, recall, and F_1 score can be seen in Table VII.

TABLE VII
BEST PERFORMING MODELS FOR GRAY STUDDED DISH RACK

Model Parameters	Precision	Recall	F_1
Polynomial	0.842	0.417	0.558
Radial Basis, Biased Hyperplane	**0.829**	**0.968**	**0.893**

Metrics obtained by averaging the results from the 5-fold cross-validation.

The respective ROC Curves for the gray studded dish rack can be seen in Figure 7.

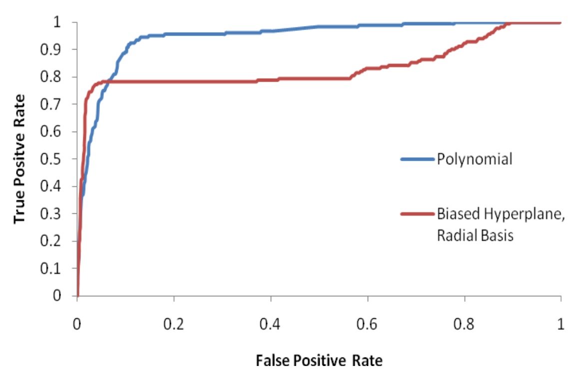

Figure 7: Comparison ROC Curves fo the Gray Studded Dish Rack

The final non-flat surface modeled is the Amazon Box, which actually had a wider array of possible models to choose from. Only three models stood out and they were the polynomial kernel, the radial basis kernel itself, and the radial basis kernel with a biased hyperplane. The results of precision, recall, and F_1 score can be seen in Table VIII.

TABLE VIII
BEST PERFORMING MODELS FOR OPEN AMAZON BOX

Model Parameters	Precision	Recall	F_1
Polynomial	0.820	0.805	0.812
Radial Basis, Biased Hyperplane	**0.797**	**0.937**	**0.861**

Metrics obtained by averaging the results from the 5-fold cross-validation.

The respective ROC Curves for the Amazon Box can be seen in Figure 8.

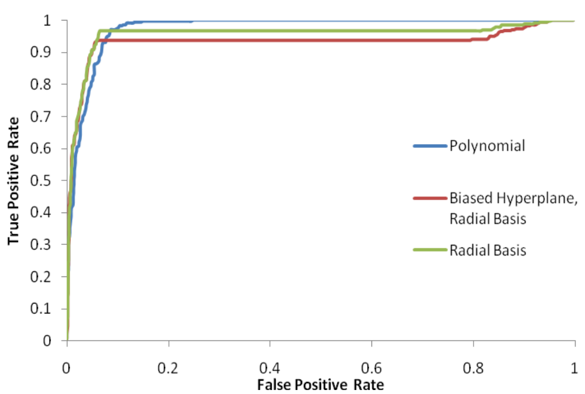

Figure 8: Comparison of ROC curves for the Amazon box

After looking at the various models for the various surfaces, it became hard to know which surface actually performed the best. The area under the curve (AUC) benchmark proved to be a good measure with which model performed the best. The results of this comparison can be seen in Table IX.

TABLE IX
AREA UNDER CURVE FOR NON-FLAT MODELS

Model Parameters	Wood Rack	Open Box	Gray Rack
Radial Basis	N/A	0.951	N/A
Biased Hyperplane, Radial Basis	0.963	0.927	0.662
Polynomial	**0.969**	**0.971**	**0.812**

Metrics obtained by averaging the results from the 5-fold cross-validation.

D. Synthetic Data

To properly evaluate our data, we obtained synthetic data generated from a physics-based model. We converted this synthetic data from their TET labeling format to the labeling format used by our data. We generated features for this data and used 5-fold cross-validation in order to generate the SVM models. We used the same models as our dataset. The performance of these models can be seen in (Table VI).

TABLE VI
MODELS FOR SYNTHETIC DATA

Model Parameters	Precision	Recall	F_1
Polynomial	0.580	0.203	0.300
Polynomial, 0.1 trade-off b/w training & margin	0.248	0.448	0.319
Polynomial, Biased Hyperplane	0.570	0.379	0.455
Radial Basis, Biased Hyperplane	0.488	0.426	0.455

Metrics obtained by averaging the results from the 5-fold cross-validation.

The performance of this model is rather surprising, as we expected that the synthetic model would give good results due to both the dataset size (32000 labels in total, meaning that it is being trained by more labels) and the fact that our hand-marked placements is more subject to error (due to the incomplete nature of the point cloud data and human error). The Synthetic model performed so surprisingly poor that it was not pursued in any further detail.

E. Model Comparison

After extensively modeling nine different surfaces using the contact and histograms features and comparing them through various calculations and measurements it became necessary to see which models actually placed objects in correct spots. Classification of features is one thing but having those features actually identify good placements is the true end to end test of our data.

In order to accurately test our end to end results, we used test_main, a preexisting program developed by Yun Jiang and Marcus Lim, which visualized an ideal placement on the surface. The first group of surfaces, the flat surfaces, had decent classification using the support vector machine and also performed well in the placement tests. Figure 9 demonstrates one example of a predicted good placement.

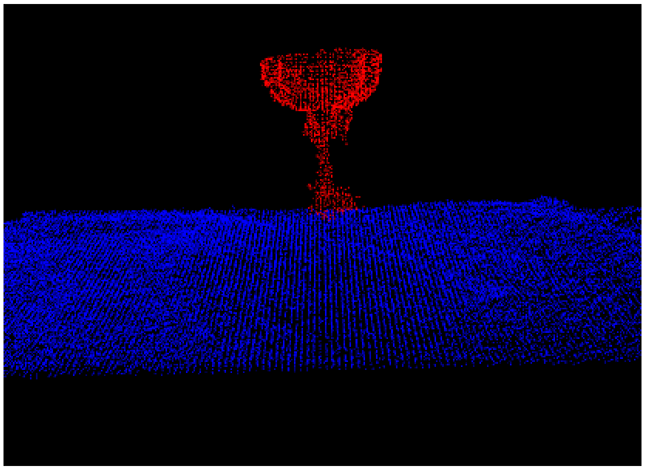

Figure 9: Good Placement on Flat Surface (Ground)

The non-flat surface models were also tested using the same program. The Wooden Slanted Dish Rack performed the best overall. There are various explanations for this but the main explanation that can explain this difference in performance is the skew in the data. The Wooden Slanted Dish Rack had a 50/50 split of good and bad placements while the Gray Studded Dish Rack and Amazon Box had very few good placements. With the random sampling of test main the skew caused the program to choose less ideal positions. Below you can see an excellent placement in the Wooden Slanted Dish Rack in Figure 10 and bad placements in both the Gray Studded Dish Rack and the Amazon Box in Figures 11 and 12.

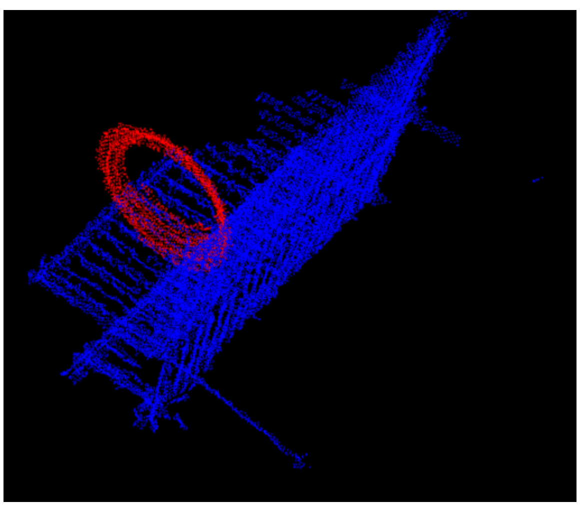

Figure 10: Good placement found for the Wooden Slanted Dish Rack

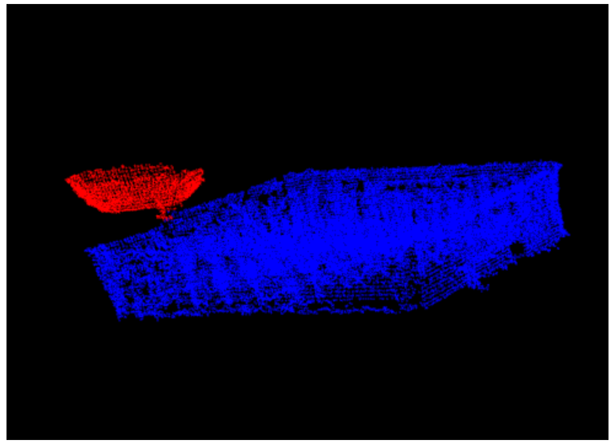

Figure 11: Bad Placement found for the Gray Studded Dish Rack

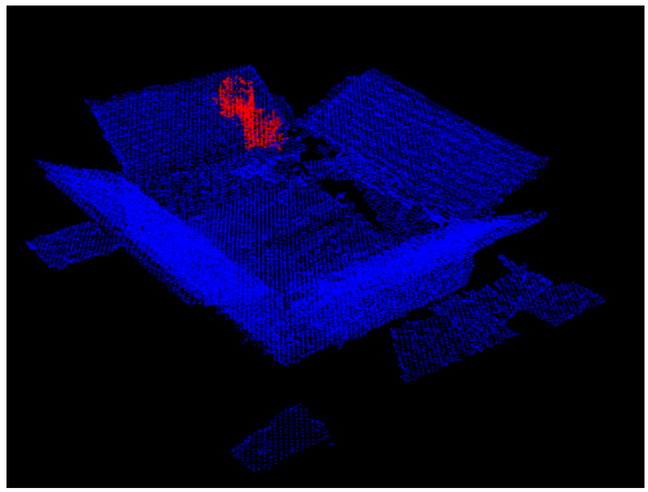

Figure 12: Bad Placement found for the Amazon Box

F. *Robot Experimentation*

To demonstrate the usefulness of our model in actual object placement, we wrote code to move our object to a surface and release them at a stable placement determined by our model.

We had the robot grasp an object at a predefined location in the same orientation given by its point cloud data. This was done so that we would know the initial orientation of the object for placement purposes. This is needed due to the how the Kinect is mounted on the Adept Arm. The Kinect is unable to see what the arm is grasping so there is no automated way of discovering this information other than by using another Kinect mounted in such a way that it can see the object that the gripper is holding. However, this is beyond the scope of our project.

Then, we gave it the locations of our various surfaces on which we wanted to place the object. Using our SVM model, we would then determine which surface was predicted to be the best placement by which one resulted in the highest discrimination threshold value, meaning that the model predicts that one particular placement is the most likely to be stable. It then placed the object in the appropriate orientation on the favored surface. The surface choosing was only done for flat surfaces.

Figure 13: Adept Arm placing Orange Cone on the Kinect Box

Figure 14: Adept Arm placing Pink Bee Plate on the Rovio Box

In addition, the placement of a single object and multiple objects was done in the Wooden Slanted Dish Rack due to its superior performance in the end to end testing. In this case it was tested to see if it was possible to place one object with a known orientation into the dish rack which was successful. It was then tested to place multiple objects into the Wood Slanted Dish Rack which was also successful after some anticipation of placement location but the model was used entirely for orienting the objects.

Figure 15: Adept Arm placing dishes into Wooden Slanted Dish Rack

IV. Conclusion and Future Work

We have constructed a significant dataset for use in any future projects that may require object placements on a number of environments. Furthermore, we have generated a model that shows relatively strong performance in our testing data. We compared our model against synthetic data generated by a physics model for comparison. Finally, we demonstrated the usefulness of the model through robot experimentation where the robot was able to successfully place most of the objects that we had given it.

As data collection took a significant amount of time, future projects should be able to save a great deal of time and effort and focus more on robot experimentation instead. One clear area of future work would be using an externally mounted Kinect to automatically determine the orientation at which an object is held, eliminating the need for a human input in robot experimentation as well as experimentation with additional features.

More future work could be done in improving the poor performing non-flat surfaces. This would involve improving the skew in the data which would allow for the testing program we used to have more samples of good positioning, allowing it choose favorable positions and orientations. More work could also be done in selecting more features. As our data set combined is in the thousands of labels we could legitimately include more features in the hope of improving model performance.

Future work would also allow us to do robot experiments using all of our surfaces and combining all of the models together to find ideal placements for different objects. This could prove interesting as hopefully the model could choose to place a dish in the dish rack and a barbell on a flat surface. In addition to improved modeling it would also be useful to further explore planners for the Adept Arm. Better planning would allow for easier placement of objects. Our experimentation showed good positioning of the objects but the arm could use better planning to place the objects in those locations.

Acknowledgments

The authors would like to thank Yun Jiang and Marcus Lim for their valuable guidance throughout the course of this project. We would also like to thank Dr. Ashutosh Saxena for organizing this course and providing the resources used for this project.

References

[1] Y. Jiang, C. Zheng, M. Lim, and A. Saxena. “Learning to Place New Objects,” Cornell University Technical Report, May 2010.

[2] T. Joachims, “Making large-Scale SVM Learning Practical,” in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.