AN ABSTRACT OF THE THESIS OF

Matthew Paul Nishita Unrath for the degree of Honors Baccalaureate of Science in Computer Science in Computer Science presented on June 2, 2014.

Title: Predicting Robotic Grasps Using Surrogate Datasets

Abstract approved: _____

Weng-Keen Wong

One of the tasks that continues to prove difficult in robotics is the ability to grasp objects of varying shapes. It is time-consuming to acquire large amounts of real-world data in order to train accurate classifiers that can predict the success or failure of a grasp. To solve this issue, we examine using artificially generated surrogate, or substitute, datasets as replacement training data for more expensive physically-tested training data. By dividing up the grasp space using kd-trees, we demonstrate that surrogate datasets can be efficiently leveraged to produce high-precision data in specific areas of the grasp space. This greatly eases the burden of collecting data by only requiring physical testing in areas where surrogate datasets have little expertise.

Key Words: robotic grasping, kd-tree, Gaussian Process, Logistic Regression, classification

E-mail address: unrathm@onid.oregonstate.edu

Predicting Robotic Grasps Using Surrogate Datasets

by

Matthew Paul Nishita Unrath

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Computer Science (Honors Scholar)

Presented June 2, 2014
Commencement June 2014

Honors Baccalaureate of Science in Computer Science thesis of Matthew Paul Nishita Unrath presented on June 2, 2014.

APPROVED:

Mentor, representing Computer Science

Committee Member, representing Mechanical Engineering

Committee Member, representing Mechanical Engineering

Dean, University Honors College

I understand that my thesis will become part of the permanent collection of Oregon State University, University Honors College. My signature below authorizes release of my thesis to any reader upon request.

Matthew Paul Nishita Unrath, Author

ACKNOWLEDGEMENTS

I would like to thank my mentor Dr. Weng-Keen Wong, for his encouragement, patience, and guidance throughout the year. I've truly learned a lot and couldn't have done it without your support. A big thank-you to Dr. Ravi Balasubramanian and his graduate student team, who laid the groundwork for this research and who were very helpful throughout the process. And thank you to my parents, Mark and Nancy Unrath, who raised me to play with Legos, taught me to think critically about the world, indulged many hours of programming, and who have always pushed me to do my best.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDIX TABLES

Table Page

Chapter 1: Introduction

Robotics has advanced greatly in recent history, and automation has allowed us to increase our productivity in industry, as well as aid humans in executing hazardous or difficult tasks such as interplanetary exploration. Nevertheless, certain tasks continue to be a challenge to the robotics field, one of which is the robust grasping of objects. Predicting the success or failure of a grasp, given an object and a specific mechanical hand, is nontrivial; large amounts of training data are required to achieve acceptable accuracy. Obtaining this training data is expensive, as it requires a human operator to manually position the hand and record the results of testing over several runs.

In this thesis, we explore the hypothesis that surrogate datasets can be used in place of real-world physical testing data to train grasp prediction classifiers. One such source of surrogate data is data generated by Columbia University's GraspIt! simulator. This thesis explores the effectiveness of training a classifier on GraspIt! data and testing the accuracy of its predictions against ground truth success and failures obtained from physical testing. We also investigate the areas of feature space in which the classifier trained on surrogate data is accurate enough to replace more costly physical trials.

Chapter 2: Previous Work

This paper builds upon previous work on this same grasping problem. Grasp success prediction using Gaussian Processes was done by Goins et al.[4] using physical testing data. Balasubramanian et al.[1] use human input to evaluate how automated grasp planners differ from grasps in which humans position a hand manually. Better performance for the human grasps than the generated grasps was observed, suggesting a new metric to use in grasp prediction (skewness).

Chapter 3: Methodology

3.1 Dataset

Our working dataset is 522 points selected by humans as plausibly good grasps of objects. These objects are varied in shape; the set is composed of a CD case, cereal box, coil, pitcher, remote, soap dispenser, soda can, water bottle, and martini glass. The grasp is described in terms of 11 different metrics listed in Table 3.1; each attempts to capture a quality that indicates a successful grasp. The baseline class label for the dataset is obtained from physical testing. To compute this, an object is grasped, then lifted into the air and shaken vigorously. This is repeated over 10 trials; if at least eight of them succeed, the grasp is considered to be a success. Training a classifier using this physical shake test as a class label has been done by Goins et al.[4] with some success. The drawback to continuing to use physical testing is the high cost of gathering it, as each grasp must be manually run and entered into a database. For this reason, we would like to explore whether another surrogate dataset could be used to replace the real thing.

Our initial experiments were driven by a specific hypothesis: surrogate datasets, such as energy estimates from programs such as GraspIt! or OpenRAVE, or human voting assessments, will not be as accurate as physical testing training. However, they could perform very well in certain areas of the space and serve as a low-cost replacement to more expensive physical testing.

Most of our experimental work was carried out using the GraspIt! energy metric as a surrogate class label. GraspIt! attempts to minimize this value, which is calculated by creating friction cones for a set of points on the robotic hand. These cones are scaled depending on the distance of the points from the object's surface; a lower energy value ideally corresponds to a more stable grasp [4].

Metric	Description	Min	Max	Source
Contact Point Equilateralness[a]	Equilateralness of the triangle made by the contact points of the fingertips	0	1	[2]
Grasp Volume[a]	Volume of the triangular prism consisting of the fingertips and the palm	0	669cm^3	[2]
Finger Extension[b]	Average finger flexion	0	1	[2]
Finger Spread[a]	Amount of spread of the fingers	0	1	[2]
Finger Limit[c]	Total flexion of all the fingers	0	1	[2]
Parallel Symmetry[b]	Distance between center of mass of object and contact point parallel to the object's principal axis	0	0.5	[6]
Perpendicular Symmetry[b]	Distance between center of mass of object and contact point perpendicular to the object principal axis	0	0.5	[6]
Object Volume Enclosed[a]	Normalized volume of the object enclosed by the hand	0	1	[6]
Skewness[c]	Alignment of the hand principal axis parallel to the object principal axis	0	180	[1]
Grasp Wrench (Epsilon)[a]	Minimum disturbance wrench that can be resisted	0	1	[3, 5]
Grasp Wrench Volume[a]	Volume of grasp wrench space	0	2^6	[3, 5]

Table 3.1: Description of metrics. a - Larger = Better grasp; b - Smaller = Better grasp; c - Mid-range = Better grasp.[4]

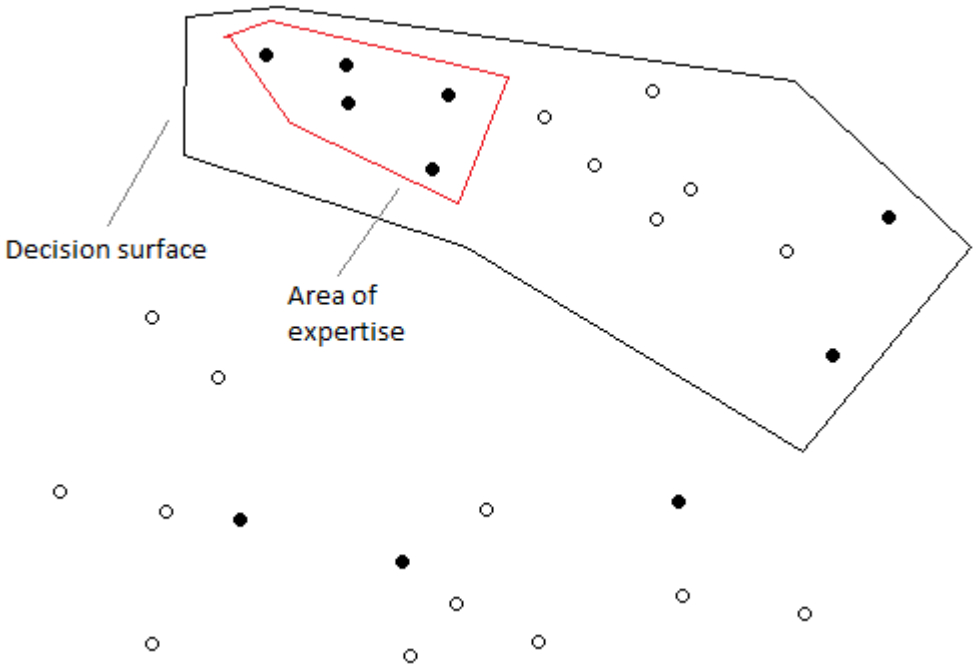

Figure 3.1: Visualization of Surrogate Classifier Expertise

3.2 Algorithm

3.2.1 kd-tree

In this paper we compare three different types of classifiers: a globally-fitted Gaussian Process, a globally-fitted Logistic Regression, and a tree of locally-fit Logistic Regressions. For each classifier we measure the performance of (1) an unfiltered version, which is given all test points, and (2) a precision-filtered version, which attempts to predict only in regions where the classifier has a history of high precision. To achieve this precision filtering, we must divide up the grasp space into regions. One approach to doing this is to use a uniformly spaced grid and filter poorly performing sections. This approach has two major drawbacks. First, the number of grid cells grows exponentially with the number of dimensions and secondly, the data is not not uniformly distributed in this space. Another alternative is to find a lower dimensional representation of the data using Principal Components Analysis (PCA). Unfortunately, even with PCA, the

majority of the data is densely packed into a region of low-dimensional space when using only a few principal components to represent the data. Using a uniform grid results in the majority of datapoints falling into only a few bins. A much better approach is to use a multi-resolution space partitioning data structure such as a kd-tree.

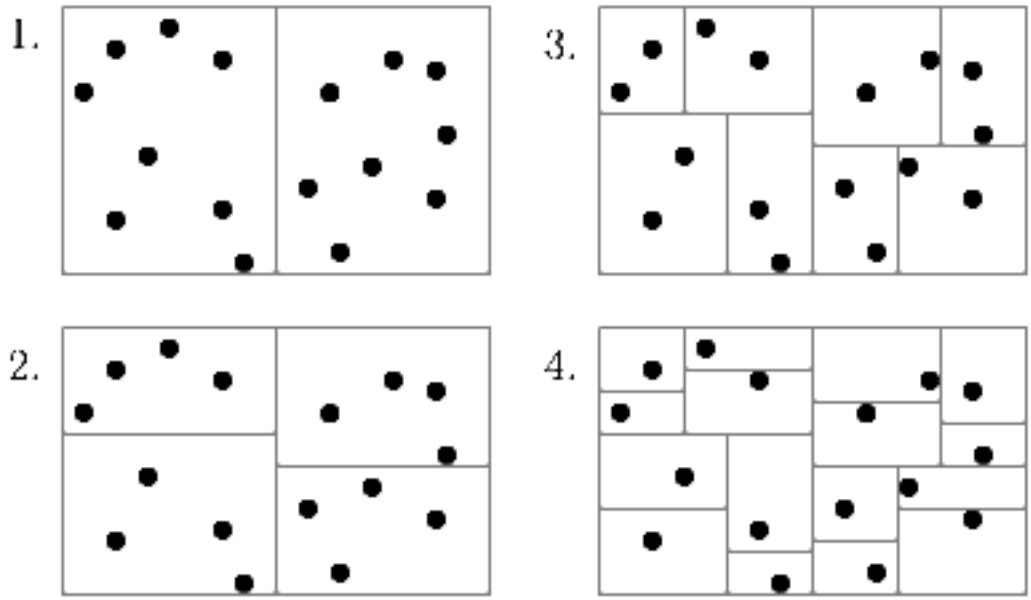

Figure 3.2: Building a kd-tree

A kd-tree is a binary tree that acts as a multi-resolution representation of all points in a space, partitioning dense areas well without consuming cumbersome amounts of memory. It is constructed using the following recursive algorithm:

```
function kd-tree(x, maxPoints, dim)
if |x| < maxPoints:
    return node(x)
else
    pivot = median(x[dim])
```

```
x1 = all xi in x where xi[dim] < pivot
x2 = all xi in x where xi[dim] >= pivot
n = new node
Add kd-tree(x1, maxPoints, dim+1) as child of n
Add kd-tree(x2, maxPoints, dim+1) as child of n
return n
```

After partitioning is complete, every node in the tree contains at most maxPoints datapoints. The kd-tree can be queried with new points and return its containing node.

3.2.2 Precision Filtering

To improve the performance of the classifiers, we look to abstain from predicting in nodes with low precision. Precision takes into account two attributes of the test set - the number of true positives (TP), where the classifier predicts success and the ground truth is also success; and the number of false positives (FP), where the classifier predicts success but the ground truth is a failure. Precision is defined as TP / (TP+FP), representing the proportion of positive predictions that were actually successful. The classifier outputs a probability that a grasp is a success. Typically, probabilities above 0.5 are considered successes (ie. positives) while those falling below are considered failures. The threshold value of 0.5 can be adjusted to any value, which in turn makes the classifier's performance sensitive to this threshold. We will refer to this threshold as a *prediction cutoff* and report values for various threshold values corresponding to different false positive rates. Intuitively, a classifier with high precision does not necessarily have to find every successful grasp, but when it predicts positive, it is very accurate.

Our procedure to eliminate low-precision areas involves a three-step process. First, classifiers are trained on the surrogate class label - in the case of global classifiers, a single classifier for the entire space; for the local LR classifier, one per kd-tree node. This step is followed by a second training phase using an independent second training dataset to identify high precision kd-tree nodes. The precision of each kd-tree node is calculated by comparing their predictions to ground truth, then stored for use during testing. In the final testing stage, every point is checked for whether it falls into a low-precision area. If it does, it is eliminated from the test set; this is equivalent to the classifier giving up and saying it is not knowledgeable about that specific area. The remaining test points

lie only in high-precision areas, and should presumably yield more accurate predictions.

3.2.3 Global Classifiers

The two global classifiers are a Gaussian Process and Logistic Regression. They are trained over the entire training set, and output the predictions for all test/query points. The only additional behavior we add here is the filtering by nodes before predicting on the final test set.

3.2.4 Local Logistic Regression

Training a Logistic Regression over the entire dataset can feasibly limit its performance, as it attempts to encompass nuances of certain parts of the space. Local classifiers, by contrast, could tailor themselves only to their immediate datapoints without accommodating outlying areas. This is the motivation behind a tree of local Logistic Regressions. After building the kd-tree, we train a separate classifier on each node of the tree. Predictions for new points are routed to the regression corresponding to that area. Since each node is only a small number of points, this creates the danger of each local classifier grossly overfitting the data in 11 dimensions. However, our hope is that the predictions remain reliable within the small area of the node.

We do not attempt the same process with local GPs in the interest of time. Minimizing the hyperparameters associated with a Gaussian Process is expensive, and the typical application of grasp prediction would be attempting to pick an object up in real time. Spending large amounts of time predicting grasps, even if accurate, would not meet the requirements of the problem.

3.3 Experimental Setup

Our experiment has several degrees of freedom, requiring us to choose only certain areas to explore in the interest of time. As mentioned previously, Energy, our surrogate class label, is spread very evenly across a wide range, making selection of a threshold point difficult; the data spread is plotted in Figure 3.3. We test with energy threshold values of 20, 30, and 40. Any datapoint with an energy value of less than the threshold value

is given a class label of "success"; any value higher than the threshold corresponds to a "failure" class label. Our empirical findings show that the results are not overly sensitive to this threshold and we will use 30 as the cutoff.

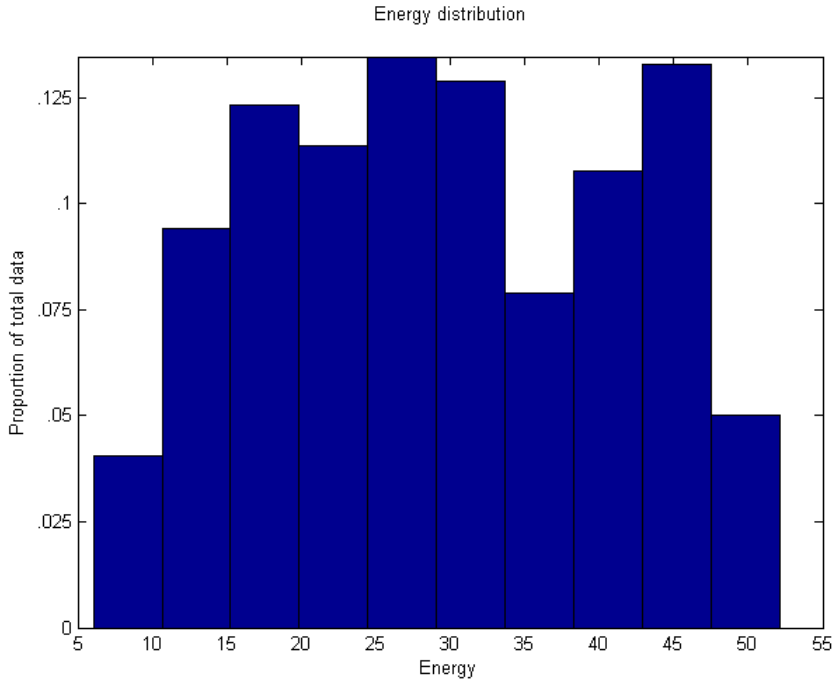

Figure 3.3: Distribution of GraspIt! Energy Values, plotted over 522 grasps.

The use of kd-trees in filtered predictions adds another degree of freedom to our experiments - it is not clear what node size is optimal for maximizing precision. Too small of a bin size means the filtering step will overfit the precision of different spaces; too large of a bin size means the space is not meaningfully partitioned. In practice, we have found that bin sizes larger than 40 does not make a significant impact compared to no filtering. For our experimental procedure we run our tests with several node sizes under this value, ranging from 10 to 40. One thing to note is that in practice, due to the small number of datapoints, the kd-tree does not typically split on all 11 dimensions; by four to five splits, all the data is partitioned into suitably-sized nodes. It is also necessary when filtering to define a precision value for what is a "bad" node. Any nodes with a precision of less than this value are not predicted on at test time. We test with precision

cutoff values ranging from 0.5 to 0.9 in 0.1 increments.

Gaussian Processes can consume significant computation time when predicting on high-dimensional data. For this reason, we reduce the data for our global GP to the first two principal components from PCA. This does not affect the building of the kd-tree; query points are still mapped to a node in 11-dimensional space before being given to the GP. Our Logistic Regression classifiers use ridge regression in both the global and local versions.

To enable comparison, both filtered and non-filtered classifiers break the datasets into three parts: 1/3 is used for training, 1/3 is reserved for the train2 phase, and 1/3 is used for testing. In the case of non-filtering classifiers, the train2 data is simply discarded. This split is performed 100 times on the data.

Predictions and calculating precision requires a cutoff separating success from failure, both in terms of ground truth and classifier prediction. We define any physical testing result ≥ 0.8 to be a success. For classifier predictions, we consider any value ≥ 0.5 to be a predicted success.

Chapter 4: Results

Below in Figure 4.1 are contour plots for each of the three classifiers, using no filtering and filtering with a 0.8 precision-filtering cutoff. This threshold typically gave the best results in terms of precision and recall. Data for precision and recall are given in the Appendix in Table A.1.

Precision threshold	**Global GP**	**Global LR**	**Local LR**
No filtering	172	172	172
0.5	146.11	161.7	138.95
0.6	135.4	151.73	125.91
0.7	125.25	137.05	109.7
0.8	113.05	113.03	95.8
0.9	91.69	72.84	72.03

Table 4.1: Number of overall prediction points. binSize=30, energy=30.

Analysis of the results in Figure 4.3 shows that up until high precision cutoffs, the global GP generally has the highest precision and recall, with the global LR only slightly lower. Local LR performs disappointingly, with more restrictive filtering and lower precision than the two global classifiers. This being said, the error for all three classifiers is large relative to their differences and there is significant room for overlap. All three classifiers benefit from the kd-tree filtering strategy, usually with a 5-6% gain in precision at a precision-filtering cutoff of 0.9. Improvements above .8 a precision-filtering cutoff are marginal and do not give a large amount of return for the reduction in the number of predicted points.

Overall, the local LR strategy offers little over the global LR. Global LR has better precision values for equivalent precision thresholds, and removes fewer points during the filtering process as shown in Table 4.1. The largest advantage to LR is its speed; a GP grows cubically with the size of its input, so especially as more datapoints are gathered,

a global LR will greatly outperform a GP in terms of computation time. With only 1/3 of the 522 points to train on, a GP takes 1.07 sec per split, versus LR's .171 sec. This could be a significant advantage when looking to predict grasps in real time. As shown by the contour plots, all three of the classifiers share some regions of very high ($>.9$) precision. For generated grasps similar to these, GraspIt! can be considered an extremely trustworthy surrogate label.

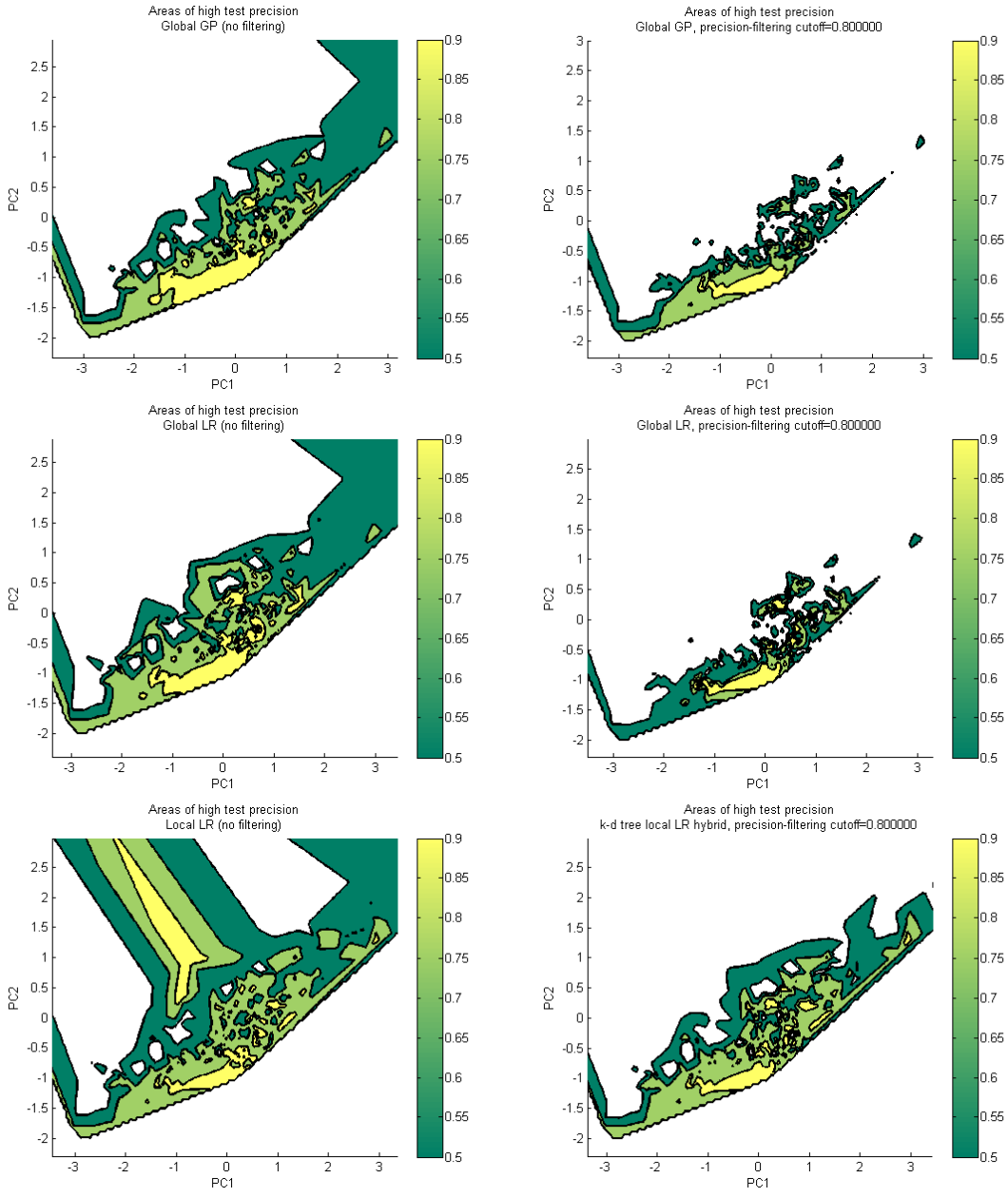

Figure 4.1: Contour plots of all three classifiers, with no filtering and filtering with .8 precision-filtering cutoff

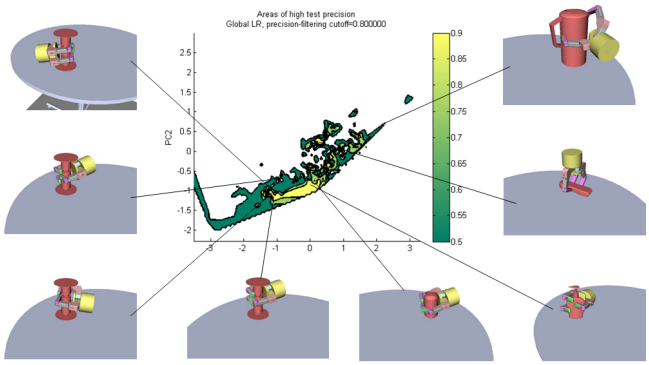

Figure 4.2: Various grasps and their location on the contour.

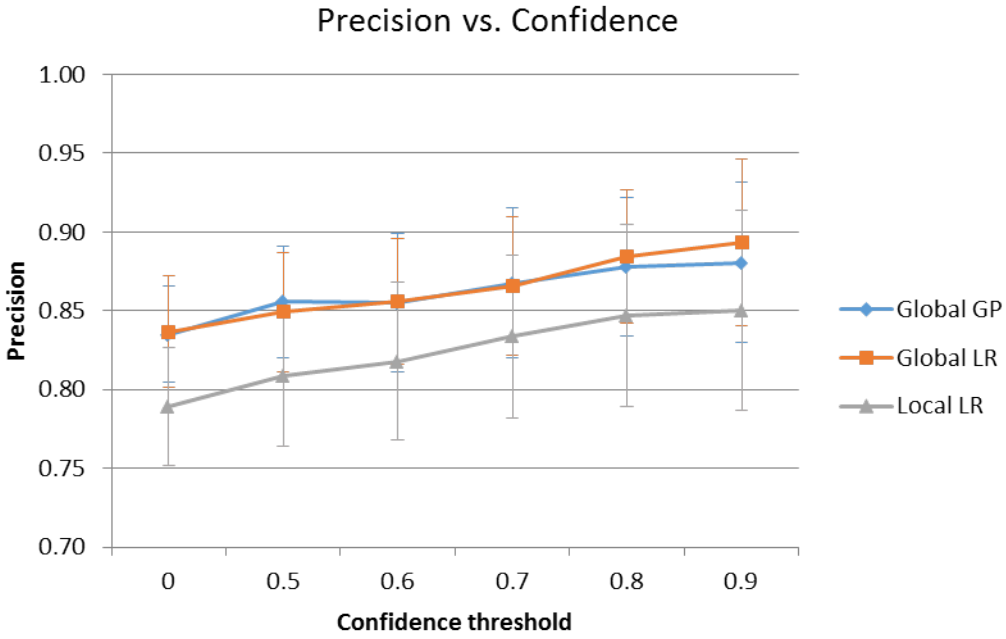

Figure 4.3: Precision vs. filtering threshold. binSize=30, energy=30. Error shown in terms of standard deviation.

Chapter 5: Conclusions and Future Work

We have examined multiple strategies for training a classifier on surrogate data that can predict well in terms of actual ground truth. Our results show that multiple local classifiers add little to the overall performance of a classifier, and global GP and LR classifiers are better in terms of precision and TPR for given FPR thresholds. Filtering areas of prediction can be seen to significantly improve the accuracy of all classifiers, typically providing a 5-6% increase to precision and improving TPR even further when a low FPR is required. With filtering, all three classifiers cull down their predictions to a few areas where GraspIt! energy corresponds well to real-world testing. These experiments support the conclusion that, in a few special areas, energy can be relied upon as an acceptable substitute to more expensive physical testing.

We have evaluated our classifiers using only one surrogate class label, the GraspIt! energy metric. Using other datasets, such as human-generated votes on grasp success, could yield better results in different areas of the space. Another dataset to try might be the energy heuristic from OpenRAVE, a grasp prediction program with different metric calculations.

Although we have shown GraspIt! to be a competent surrogate label in certain areas of the space, it is not well understood what these grasps have in common. Further research could be done on what relationship these grasps might have in common, and whether other forms of local classification might be viable options.

One important feature not yet taken into account is the shape of different objects. This could indirectly map to different metrics, such as grasp volume and finger extension, but this has not yet been conclusively shown. Various shapes call for different grasping strategies, so separating these could yield better results than attempting a composite classifier.

Bibliography

[1] Ravi Balasubramanian, Ling Xu, Peter Brook, Joshua R. Smith, and Yoky Matsuoka. Physical Human Interactive Guidance: A Simple Method to Study Human-Planned Grasping. *IEEE Transactions on Robotics*, 2012.

[2] E. Chinellato, A. Morales, R. B. Fisher, and A. P. del Pobil. Visual quality measures for characterizing planar robots grasps. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 35(1):30 – 41, 2005.

[3] C. Ferrari and J. Canny. Planning optimal grasps. *Proceedings - IEEE International Conference on Robotics and Automation*, 3:2290 – 2295, 1992.

[4] Alex Goins. Improving Robotic Grasping Performance Using Machine Learning Techniques. *IROS*, 2014.

[5] A. T. Miller and P. K. Allen. Graspit: A versatile simulator for robotic grasping. *IEEE Robotics and Automation Magazine*, 11(4):110 – 122, 2004.

[6] A. Saxena, L. Wong, M. Quigley, and A. Y. Ng. A vision-based system for grasping novel objects in cluttered environments. *Springer Tracts in Advanced Robotics*, 66.

APPENDICES

Appendix A: Precision and Recall

	Global GP	Global LR	Local LR
No filtering	.8348 / .6372	**.8367 / .6382**	.7892 / .5630
Conf=0.5	**.8556 / .5838**	.8492 / .5765	.8088 / .5070
Conf=0.6	**.8854 / .5590**	.8560 / .5444	.8180 / .4634
Conf=0.7	**.8676 / .5103**	.8656 / .4983	.8336 / .4000
Conf=0.8	.8782 / **.4672**	**.8845** / .4395	.8471 / .3538
Conf=0.9	.8804 / **.3780**	**.8934** / .3729	.8500 / .2690

Table A.1: Precision and recall at the conventional prediction cutoff of 0.5. binSize=30, energy=30.