Robo-Librarian: Sliding-Grasping Task

Adao Henrique Ribeiro Justo Filho, Rei Suzuki

Abstract**—** Picking up objects from a flat surface is quite cumbersome task for a robot to perform. We explore a simple heuristic for picking up books from a book cart; by sliding the book off the table slightly, we create a grasping surface that the PR2 robot can manipulate. For our learning component, we utilize a support vector machine or the classifying of an object as a book or nonbook as well as an scale-invariant feature transform and nearest neighbor implementation to track the book as we interact with it. We successfully demonstrated in the simulator that the sliding schema would successfully move the book in such a way that we could then pick it up.

I. Introduction

Robots are at their most useful performing repetitive tasks that humans are uninterested in performing. One such task is putting away books. Ideally, the robot would move to a book cart, identify a book on the cart, pick it up, and search its database for the proper location in which to put the book. Then the robot would move to the shelf where the book should be placed, identify the location where the book should be returned and place the book on the shelf. This task while simple and menial for humans is quite complex for robots and can be broken down into a series of smaller sub-tasks that must each individually addressed for the robot to put away the books.

We have taken this complicated problem and decided to address several of the subtasks instead of performing the entire operation. Our goal is to enable the PR2 to perform a simplified version of this task. First, we simplify the problem by removing any motion planning. The PR2 will stand between the book cart and the bookshelf in such a way that it can simply turn between the two of them. Also, it will not have to identify the "correct" location to put the book. Our focus in this project is on the picking and placing of the book mainly and we attempt to abstract any of the other tasks away.

We will attempt to do the following:

1) Identify books from non-books using a support vector machine (SVM)
2) For a given book on the book cart, be able to pick it up using a sliding grasping technique
3) Reorientate the book in gripper to be placed on the shelf
4) Place book on bare / sparsely packed shelf

We chose to tackle these problems as they are the more complex and more widely applicable subtasks of this project. Picking up an object by sliding it on the table and reorientating it while grasping it are heuristics that can be applied to a plethora of other tasks. Ideally, we would like to be able to organize the books in a meaningful way (re: return them onto the shelf in a particular ordering – say alphabetically by author).

The paper is organized as follows: once we finish touching upon related work, we will discuss each subproblem. We will also discuss experimental results for each subtask. Finally, we shall discuss the implications of our work and potential improvements.

II. Related Work

Robotic manipulation is often in a factory setting or a contrived lab scenario where the object of manipulation has an ideal grasping surface readily available to it. For instance, in the Robot Construction Worker project [1], the building bricks are of the ideal size for the gripper to manipulate. The arm can directly pick up the block from the table without needing to position the block in any way. Another example is in "Robotic Grasping of Novel Objects using Vision," where the researchers learned grasping locations for objects [2]. These surfaces had to be at locations where the robot could grasp the object initially without maneuvering the object to create such a grasping surface.

We are attempting to start from a scenario where the robot cannot merely pick up the item of interest, but rather must maneuver it into a position that it can then manipulate.

Object detection is a heavily researched topic in computer vision. We merely seek to implement a sufficient method to allow us to identify and interact with our books. To identify books, we utilize a support vector machine or SVM [3]. For object tracking, we have chosen to utilize the Scale-Invariant Feature Transform or SIFT algorithm [4]. This algorithm detects and describes local features or keypoints in an image, allowing us to track the location of the book that we have identified with the SVM.

III. Identifying Books From Point Cloud Data

Learning books from non-books can be broken into two components. One is the processing of raw point cloud into features which could be learned from; second is the support vector machine (SVM) that learns utilizing the features.

A. Extracting Features

We take raw point cloud data (x, y, z in robot coordinates as well as color data for a given particle) and feed it into Point Cloud Library (PCL). Using PCL, we identify the largest planar surface (re: the table / book cart). We remove the particles that comprise this plane and any particles below the plane. This gives us particle clusters that represent the particles sitting on the table. From this, we extract features like the volume data (from convex hull) and point feature histograms. We then take these features, apply a binary classification (1 for book, 0 for non-book) and pass them into our SVM.

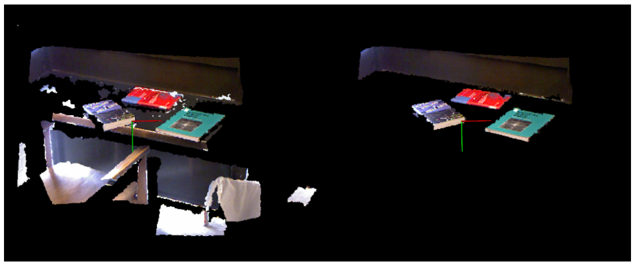
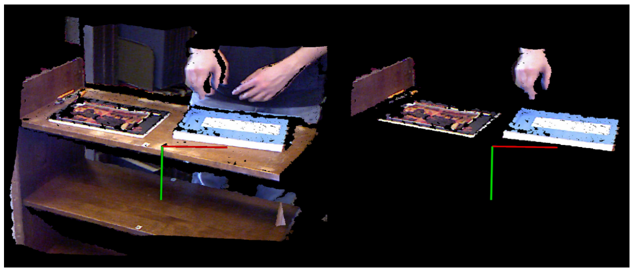

(a) (b)

Fig. 1 (a) the raw point cloud image (b) point cloud after the planar surface is extracted. Each grouping of points will be treated as an object.



B. Classifying with SVM

We chose to use a support vector machine (SVM) as our binary classifier. SVMs are a type of supervised learning algorithm maps data into a richer feature space utilizing nonlinear features, then constructs a hyperplane that will hopefully separate the data. This hyperplane will have a particular margin between the nearest n samples of one particular class and m samples of the other class. The SVM compromises between maximizing this margin and minimizing the error of the classification. Typically it is used for binary classifications, but can be modified for classifying between different types of data. SVMs can perform nonlinear classification utilizing the kernel trick, which implicitly maps inputs into high-dimensional feature spaces. For our SVM, we utilize a simple linear classifier.

We utilize the SVM_Light application created by Professor Joachims at Cornell University [3]. It is a nice, compact and accessible SVM; it is flexible enough to utilize various kernels and large numbers of features.

TABLE 1

SVM Parameters

Parameter description	Value
Trade-off between training error & margin (C)	.003
Epsilon width of tube for regression (w)	10
Value of rho for XiAlpha-estimator (o)	2
Search depth for extended XiAlpha-estimator (k)	100
Kernel type (t)	Linear (0)
Max size of QP-subproblems (q)	40
Number of new variables entering the working set (n)	20

C. Results with SVM

SVM_Light comes conveniently with a leave-one-out form of cross validation. Leave-one-out cross validation (LOOCV) utilizes one observation from the original data as the validation set, utilizing the remaining data as the training data. It does not do a complete LOOCV, instead opting to do it for a subset of the data and to estimate the error, recall and precision from that subset. Accuracy is the proportion of the total number of predictions that were correct. Recall is the proportion of positive cases that were correctly identified. Precision is the proportion of the predicted positive cases that were correct. In general, we would want all three metrics to be relatively high. For this section, we wanted to reach a point where the recall was significantly greater than 50%.

Also SVM_Light performs XiAlpha estimates of the error, recall and precision. To supplement these metrics we also ran a test set and also ran the training set as a test set.

Our results are as follows:

TABLE 2

SVM Performance in Book Identification Task

Data set	Metric	Accuracy	Recall	Precision
362 samples (121 positive, 241 negative) Linear classifier	Test set (41 samples, 14 positive)	85.37% (35 correct)	71.43%	83.33%
	Training set	87.29%	73.55%	86.41%
	SVM_Light leave-one-out estimate	84.25%	70.25%	80.19%
	SVM_Light XiAlpha-estimate	76.52%	65.29%	64.75%

We got significantly improved performance by using a simple linear classifier over more complicated kernels. It only required a little tweaking of parameters to reach a nice level of accuracy, recall and precision.

IV. Object Tracking

Once we have identified the book that we wish to pick up, we need a methodology of tracking the book while we slide it into a position where we can grasp it. To accomplish this, we utilized openCV, in particular feature detection with Speeded Up Robust Features (SURF) and matching descriptor vectors with Fast Library for Approximate Nearest Neighbors (FLANN). The algorithm will in essence find a provided item within a particular scene. SURF is a feature detection technique derived from Scale-Invariant Feature Transform (SIFT). In essence, the technique allows features to be extracted in such a way that scale and orientation are not an issue. The FLANN allows us to match the features we identify in SURF from the target item to the features from the scene itself.

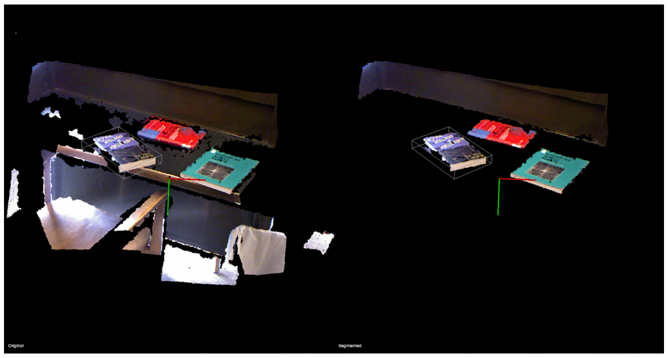


Fig. 3 Bounding box indicating desired book

The final component is to convert the ROS image to openCV using cv_bridge. This enables us to map a location to the pixels that we identify as keypoints for the object that we wish to track.

V. Picking Up Books

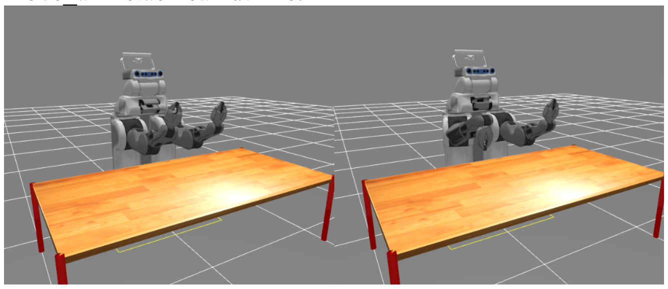
Picking up the book is a complicated task. After we have identified the book, we will use an inverse kinematics controller that avoids collisions to place a gripper behind the book. We then begin a feedback loop that will slide the book some distance delta off the table. We then must move the PR2 base back slightly to allow us to pick up the book from the table. This task is a combination of feedback and perception. The perception is mostly involved in identifying the centroid and edges of the book, while the feedback involves keeping the centroid of the book in line with the gripper off the edge of the table.

This requires the following:

1) Locate the book
2) Place the gripper behind the book
3) Open the gripper
4) Slide the book towards the PR2
5) Stop sliding before the centroid goes over the edge of the table
6) Move the gripper out of the way
7) Move the robot base back
8) Pick up the book
9) Orientate the book
10) Turn to the bookshelf
11) Find an open location on the shelf
12) Place the book

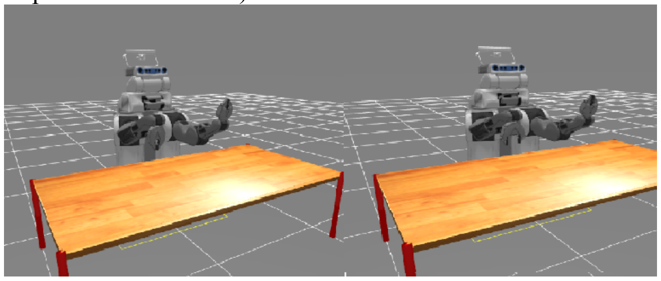
A. *Manipulating the Grippers*

Moving the grippers into position requires an inverse kinematics solver that avoids obstacles. We utilize the move_arm stack to position the arms where we want them to be located. We need to transform the location that we want to place the gripper from global into a reference frame the move_arm stack can utilize.



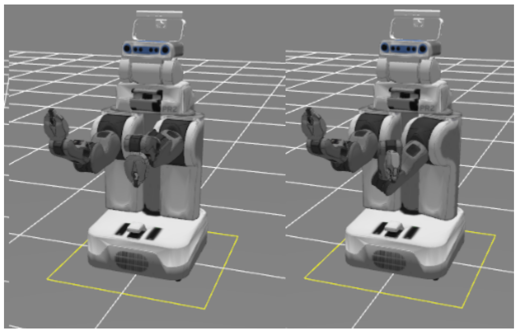
B. *Opening / Closing the Gripper*

For opening and closing the grippers, we make use of the pr2_gripper_action controller. This allows us very rough control (a fully opened and fully closed gripper). It is sufficient for opening the gripper for the final demo, but we would prefer to utilize a more responsive controller when actually grasping the book (more on that in the improvements section).



C. *Sliding Task*

The next component is the sliding component. To accomplish this, we utilize the force controller found in the ee_cart_imped package. This allows us to state a goal location and orientation for the gripper while maintaining a certain force / torque or stiffness / torsional stiffness as the arm moves, which means we can incrementally move the arm towards the edge of the table while applying a constant force to the book.



D. Picking up the Book

Picking up the book turned out to be a more complex task than anticipated. First, we could not simply pick up the book from the position where we originally were manipulating the book from. This is because the PR2's arm would collide with the table, so ultimately it could not find a solution where it could grasp the book. Thus we decided to slide the base station back slightly and then grasp the book from there. We move the base back in proportion to how far the book is from the table. We do not perform any kind of collision checks when backing up, as we decided that this lied out of the scope of the project. We chose to use the pick_and_place stack to accomplish this manipulation task. This created some complications as the stack requires a minimum height from the table from which to grasp an object, and we were essentially placing the gripper below the table to grasp the book. Thus we modified some components of the stack to enable us to pick up the book successfully. This creates some complications further along the process of placing the book on the shelf that is discussed in the conclusions.

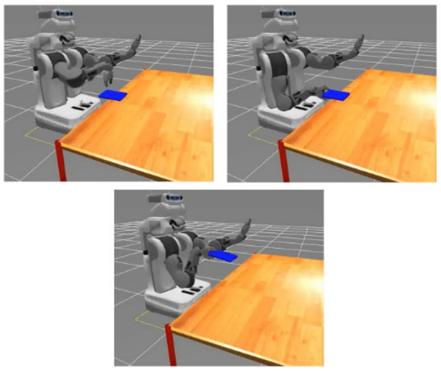


Fig. 7 Move Base (top left), Grasp Book (top right), Pick up Book (bottom)

E. Simulation Results

We successfully integrated each individual motion into a continuous motion. We can now slide and pick up the book in simulation with a relatively high success rate.

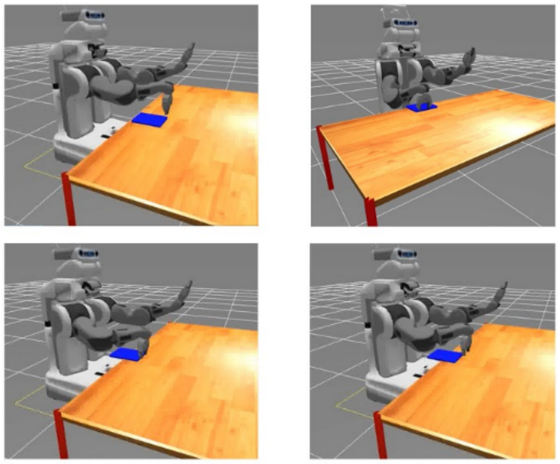


TABLE 3

Sliding and Manipulation Trials

Task	Number of Trials	Successes
Sliding Book	10	80% (failure occurs when book slides off table)
Picking Up Book	10	70% (failure occurs when hitting book while trying to pick it up)

.

VI. Conclusions and Suggested Improvements

We have nearly reached a point where we could exit simulation and begin testing on the real PR2. This would require several modifications. First and foremost, we would integrate the object identification and tracking components with the manipulation components to allow the robot to more dynamically manipulate the books.

Also, we would need to complete the placement component of the manipulation task. This involves incorporating the book into the collision map for the PR2 arm. Since we do not use the pick_and_place stack completely, we do not automatically add the object we pick up to the collision map. Once we have the collision map added, we simply need to turn to the bookshelf, identify an empty shelf and place the book upon it. We have noted that there are some issues in letting the book go once it is upon the shelf, for instance the robot gripper can sometimes knock the book asunder once it has let it go and there are no guarantees that the book remains upright once it has been set down. We may need to consider schemas that would better guarantee that the book indeed stays on the shelf once it has been placed down.

Finally, to get the grasping task to work live, we would like to incorporate the pr2_gripper_sensor_action package, which will enable to hold the book with the minimum amount of force necessary to manipulate it without having it slip. This does not work in simulator since the grip sensors are not included in the simulation.

REFERENCES

[1] Robot Construction Worker, Alycia Gailey, Alex Slover. http://www.cs.cornell.edu/courses/CS4758/2013sp/final_projects/spring_2011/Gailey_Slover.pdf

[2] Robotic Grasping of Novel Objects using Vision, Ashutosh Saxena, Justin Driemeyer, and Andrew Y. Ng. International Journal of Robotics Research (IJRR), vol. 27, no. 2, pp. 157-173, Feb 2008.

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

[4] Rob Hess. 2010. An open-source SIFTLibrary. In Proceedings of the international conference on Multimedia (MM '10). ACM, New York, NY, USA, 1493-1496.