A Vision-based System for Grasping Novel Objects in Cluttered Environments

Ashutosh Saxena, Lawson Wong, Morgan Quigley, Andrew Y. Ng

Computer Science Department, Stanford University, Stanford, CA 94305, USA.
{asaxena,lsw,mquigley,ang}@cs.stanford.edu

Summary. We present our vision-based system for grasping novel objects in cluttered environments. Our system can be divided into four components: 1) decide where to grasp an object, 2) perceive obstacles, 3) plan an obstacle-free path, and 4) follow the path to grasp the object. While most prior work assumes availability of a detailed 3-d model of the environment, our system focuses on developing algorithms that are robust to uncertainty and missing data, which is the case in real-world experiments. In this paper, we test our robotic grasping system using our STAIR (STanford AI Robots) platforms on two experiments: grasping novel objects and unloading items from a dishwasher. We also illustrate these ideas in the context of having a robot fetch an object from another room in response to a verbal request.

1 Introduction

In this paper, we present our vision-based system for grasping novel objects in cluttered environments.

In the past, most approaches to robotic grasping [1–4] assume availability of a complete 3-d model of the object to be grasped. In practice, however, such a model is often not available—the 3-d models obtained from a stereo system are often noisy with many points missing, and 3-d models obtained from a laser system are very sparse (see Fig. 2 and Fig. 4). This makes grasping a hard problem in practice. In [5, 6], we developed a learning algorithm that enabled a robot to grasp novel objects, even ones that it perceived for the first time through vision. Although we were able to grasp a wide variety of novel objects with an overall accuracy of 87.8%, those experiments were performed in uncluttered environments with objects placed against a uniform background. Grasping in cluttered environments, such as a dishwasher or a normal home kitchen (Fig. 1), is a harder problem both from a planning as well as a perception point of view. Further, in such environments, most straightforward approaches, such as visual servoing [7], become more difficult because the arm now needs to execute more complicated paths for grasping,

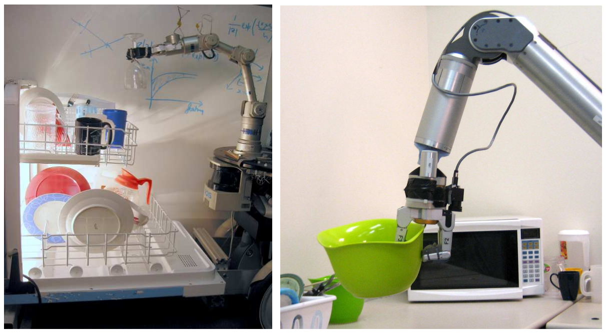

(a) (b)

Fig. 1. (a) Our robot (STAIR 1) unloading items from a dishwasher, (b) Our robot (STAIR 2) grasping an object in a normal kitchen environment.

and occlusions (by the objects or by the arm itself) make it harder to track the arm.

In this paper, we will describe our vision-based robotic grasping system, which takes into account uncertainty in the location of obstacles when planning and when choosing where to grasp. Our algorithm consists of several components: first, to decide where to grasp the object, second, to perceive the obstacles, third, to plan a path to the object while avoiding obstacles, and fourth, to actually execute the grasp.

We also describe our two robotic platforms, one having a 5-dof arm and a two-fingered hand, and the other having a 7-dof arm and a three-fingered hand. These robotic platforms are built as a part of a project whose long-term goal is to build a useful, general-purpose household robot that can navigate in indoor environments, pick up and interact with objects and tools, and carry out tasks such as tidying up a room or preparing simple kitchen meals.

We tested our vision-based robotic grasping system in picking up novel objects and unloading items from a dishwasher. As part of a larger team effort, we also combined our grasping system with tools from various other sub-fields of AI to have a robot fetch an object from another room in response to a verbal request.

2 Prior Work

Most work in robotic manipulation assumes a known 3-d model of the object and the environment, and focuses on designing control and planning methods to achieve a successful and stable grasp in simulation environments. In practice, however, such a model is often not available. For example, the point clouds obtained from a stereo system are noisy and/or missing, and points obtained from a laser system are very sparse (see Fig. 2). Therefore, approaches

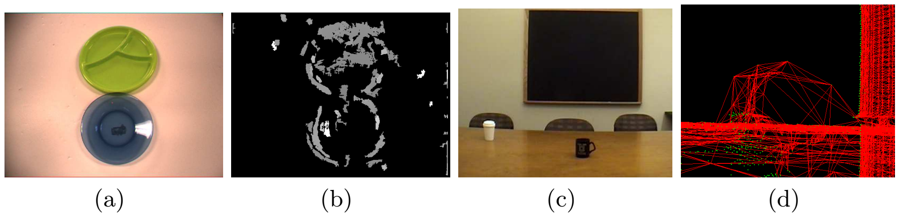


Fig. 2. (a) An image of textureless/transparent/reflective objects. (b) Depths estimated by our stereo system. The grayscale value indicates the depth (darker being closer to the camera). Black represents areas where depth-finding failed. (c) An image of a cup lying on a table, (d) Its 3-d model calculated from a laser scanner (view from the right). The sparsity of the points makes perception of the 3-d model difficult.

that assume availability of a complete 3-d model (to compute surface normals, etc.) would not apply. Here, we will focus on prior work that has performed real world grasping experiments, and refer the reader to [1–4] for a more general survey of past work in robotic manipulation.

For grasping 2-d planar objects, most prior work focuses on finding the location of the fingers given the object contour, which one can find quite reliably for uniformly colored planar objects lying on a uniformly colored table top. Using local visual features (based on the 2-d contour) and other properties such as force and form closure, the methods discussed below decide the 2-d location at which to place the fingertips (two or three) to grasp the object. Piater et al. [8, 9] estimated 2-d hand orientation using K-means clustering for simple objects (specifically, square, triangle and round blocks). Morales et al. [10, 11] calculated 2-d positions of three-fingered grasps from 2-d object contours based on feasibility and force closure criteria. Bowers and Lumia [12] also considered the grasping of planar objects and chose the location of the three fingers of a hand by first classifying the object as a circle, triangle, square or rectangle from some visual properties, and then using pre-scripted rules based on fuzzy logic.

In more general grasping, Kamon et al. [13] used Q-learning to control the arm to reach towards a spherical object to grasp it using a parallel plate gripper. Edsinger and Kemp [14] grasped cylindrical objects using a power grasp by using visual servoing. Platt, Grupen and Fagg [15] used schema structured learning to learn control policies for power grasps for objects. These methods apply to power grasps for simple objects (spherical and cylindrical) and do not apply to grasping for general shapes (e.g. grasping a cup by its handle) or to grasping in cluttered environments.

In other related work, Hsiao et al. [16] used Partially Observable Markov Random Process (POMDP) for whole body grasps; however only in simulation. Simeona [17] assumed perfect knowledge of the scene and the objects to plan a more global path (i.e. to go from point A to point B for grasping) using Probabilistic Roadmaps (PRM).

3 Algorithm

Our robotic grasping system consists of the following components:

1. **Inferring grasping points**: This component infers a 3-d point and an orientation at which to grasp the object. It takes into account the arm and hand kinematics.
2. **Perception of the environment**: The robot needs to find the object, the obstacles, and the destination where the object is to be placed.
3. **Path planner**: This component finds a path that takes the arm from the initial position to the grasp position, and then from the grasp position to the destination, while avoiding obstacles.
4. **Control**: Low level control for the arms, the hand, and the mobile base.

Besides the above, various other components are needed, such as an object recognizer to find an object, and a calibration system that calibrates different sensors and actuators in the same coordinate system.[1]

3.1 Inferring Grasping points

There are some visual features that indicate good grasps and are consistent across objects, e.g., cups could be grasped by their "handles," and bowls and glasses could be grasped at their "lips." We proposed a learning algorithm [5,6,18] that learns these visual features for identifying a 3-d point and an orientation at which to grasp the object, given its image. (Fig. 3b,c) This algorithm, trained using labeled synthetic images of objects (Fig. 3a), gives a set of candidate points (and their orientation) that are good grasps. (We have made the code available at the URL given in Section 6.2.)

However, this algorithm does not take into account the robot kinematics or the shape of the robot's workspace. Among the candidate points, some points could be better reached by a robot; for example, in Fig. 6a it is easier for the robot to pick up the martini glass by its neck rather than by its lip. In fact, some of the candidate grasping points (and orientations) predicted by the algorithm might not even be reachable because of the robot kinematics.

Therefore, we select the best grasp from the set of candidate grasps G as follows:

[1] Note that for vision based grasping systems, a good calibration of the vision sensors and the robot arm in a fixed coordinate system is essential, because the errors in the calibration directly show up as errors in the 3-d location of the predicted grasping point.

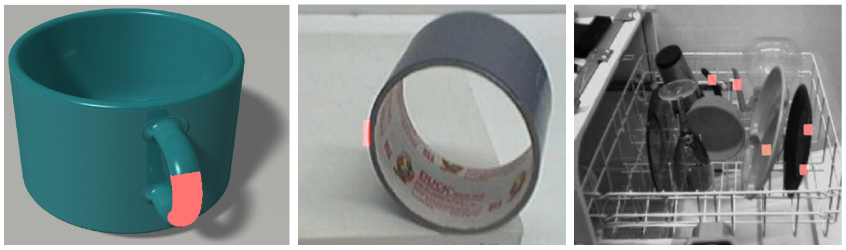

(a) (b) (c)

Fig. 3. Grasping point classification. (a) A synthetic image of a coffee mug with the grasp labels shown in red, (b,c) Test on new real objects: The red points in each image show the locations most likely to be the grasping point, as predicted by our algorithm. (Best viewed in color.)

$$\psi^* = \arg \min_{\psi \in \Psi,\ g \in G} ||\psi_x - g_x||_2 - \mu |\psi_q \cdot g_q| \tag{1}$$

where Ψ is the set of all end-effector poses (3-d location and 3-d orientation) that are possible because of robot kinematics, $\psi_x \in \mathbb{R}^3$ represents the 3-d location of the end effector, and $\psi_q \in \mathbb{R}^4, ||\psi_q||_2 = 1$ represents the 3-d orientation of the end effector in quaternion representation. (The quantities g_x and g_p are the 3-d position and orientation of the candidate grasp respectively.) This method gives a location and an orientation of the end-effector that is closest to one of the grasping points (and its orientation)—the weight μ decides how much weight we give to the orientation versus the location. This optimization problem is solved efficiently by choosing Ψ to be a small neighborhood around g.

On STAIR 1, we have a 5-dof arm, therefore, knowing the location/orientation of the end-effector, we can easily compute the goal joint angles in the configuration space using inverse kinematics. However, STAIR 2 is equipped with a 7-dof arm; therefore instead of one, we will have a *set* of goal joint angles corresponding to the extra degree-of-freedom. As discussed in Section 3.3, we will attempt to choose the goal for which the planned path has maximum distance from the obstacles.

3.2 Perception of the environment

An environment consists of a variety of objects, such as the robot itself, walls, floor, tables, objects to be grasped, etc. In order to successfully move the arm without hitting an obstacle, and to make certain decisions (e.g., where to place an object after picking it up), a robot needs to build a 3-d model of the environment.

We use a stereo system for this purpose; however, this has some problems. First, these models are usually incomplete because many of the depths computed by the stereo system are missing. It is hard to complete the models

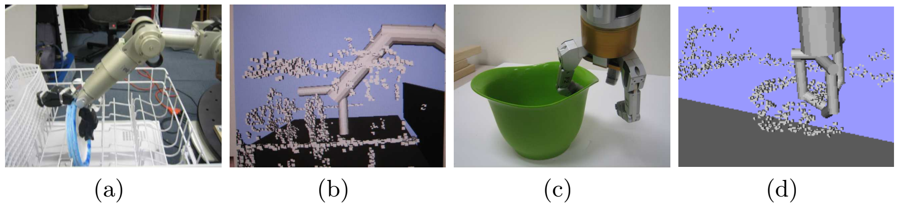


Fig. 4. (a) STAIR 1 grasping a plate from a dishwasher, (b) Imperfect 3-d structure perceived by the robot using stereovision. Determining where to grasp and planning becomes hard because of missing 3-d points and the clutter in the environment. (c) STAIR 2 grasping a bowl, (d) Only a few 3-d points are perceived by the robot on the textureless bowl using the stereo system.

because the objects are unknown. (Even if they are known by using object recognition algorithms, it is difficult to complete the model because of the unknown pose of the object.) Second, the limited view of the camera provides only a partial view of the scene. Such incomplete models sometimes result in a failure to grasp. For example, a robot might not correctly perceive a textureless table (for which a stereo system usually fails to find depths), and therefore hit the table while attempting to grasp an object lying on it.

We address this problem by noticing that many objects, such as walls, floor, tables, etc. are fixed, and hence the robot can pre-store their 3-d structure. We can use various 3-d mapping algorithms for building 3-d models depending on the sensors available, such as laser-based algorithms [19], or vision-based algorithms [20–23].[2] In our experiments, we localize the robot relative to these fixed objects by finding a few known template structures.

3.3 Path Planner

Once a grasp has been identified, the robot needs to plan a path to reach the grasp, and then from the grasp to the destination, while avoiding obstacles.

We used a probabilistic roadmap (PRM) planner from the Motion Planning Kit (MPK) [24] for this purpose. The MPK requires the start and goal positions in configuration space, which is parametrized by the arm's joint angles. We use inverse kinematics to convert the start and goal positions to joint angles. The path planning includes the hand joints as well, which significantly increases the accuracy of the grasping system because many failures are caused by the hand hitting the object to be grasped. For computing the goal configuration of the fingers (hand pre-shape), we used a criterion that attempts to minimize the opening of the hand while leaving a small margin (about 0.5cm) between the fingers and the object to be grasped.

[2] Because of the resolution of the laser, one would still miss objects that have size less than a few inches.

The algorithm above works well, except that in presence of noisy obstacles the planner often fails to find a good path. We modified the PRM to have a "soft" tolerance by allowing the path to hit fewer than a number of points. (We empirically chose $M = 3$ as the number of points.) The intuition is that 3-d points corresponding to obstacles tend to be clustered, and a single isolated point is more likely to be noise. This allows our method to robustly find paths in cluttered environments with noisy data.[3]

3.4 Control

The output of the path planner is a set of joint angles in configuration space (milestone configurations) that the robotic arm must traverse through smoothly in order to get from the start to the goal without hitting any obstacles. We control the arms in position-controlled mode to follow the milestones.

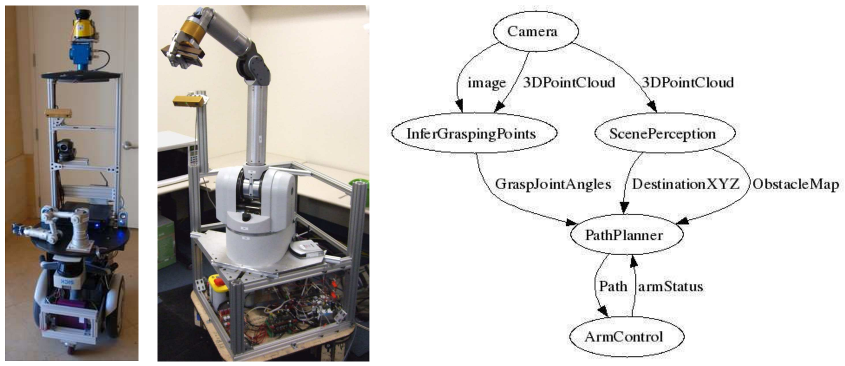
In our experiments, since the hand pre-shape is such that the fingers are just short of touching the object, the actual grasp control is simply closing the fingers of the manipulator, until they stop moving. This method works very well in practice for non-deformable objects; however there were some instances where the grasp failed because the hand knocked the object off. In those cases, using various complementary methods described in prior work (Section 2) that use haptic (force) feedback [25] or visual feedback [7] would quite likely improve the performance of our grasping system.

4 Robot Platforms

Our experiments were performed on two robots built for the STAIR (STanford AI Robot) project. Each robot has an arm and other equipment such as cameras, lasers, computers, etc. (See Fig. 5a,b.)

STAIR 1 consists of a harmonic arm (Katana, by Neuronics) mounted on a Segway robotic mobility platform. Its 5-dof arm is position-controlled and has a parallel plate gripper. The arm has a positioning accuracy of ± 1 mm, a reach of 62cm, and can support a payload of 500g. Our vision system used a low-quality webcam (Logitech Quickcam Pro 4000) mounted near the end effector, a stereo camera (Bumblebee, by Point Grey Research) and pan-tilt-zoom (Sony DV100) cameras mounted on a frame behind the arm. In addition, the robot has a laser scanner (SICK LMS-291) mounted approximately 1m above the ground for navigation purposes, and an additional laser scanner (SICK LMS-200) on a pan-tilt motor (Amtec Powercube) atop the sensor

[3] If the goal joint angles are such that a path could not be found without hitting fewer than M obstacle points, then we take the next best grasp from Eq. 1. In case of STAIR 2, we have an extra degree of freedom; therefore from the set of equally good goal configurations, we choose the one for which the path has a large distance from the obstacles.



(a) STAIR 1 (b) STAIR 2 (c) Switchyard Graph

Fig. 5. (a) STAIR 1 is equipped with a 5-dof arm and a parallel plate gripper, (b) STAIR 2 is equipped with 7-dof Barrett arm and three-fingered hand, (c) The graph for grasping experiments in Sections 6.1 and 6.2.

frame. The second laser was used to gather the point cloud images shown previously. For our robotic grasping experiments in Section 6.1 and 6.2, we used only the webcam and the stereo camera.

STAIR 2 sits atop a holonomic mobile base designed and constructed by Reuben Brewer of the Stanford Biorobotics Laboratory. The base has four steerable wheel turrets, allowing arbitrary 2-d translations and rotations. Its 7-dof arm (WAM, by Barrett Technologies) can be position or torque-controlled, is equipped with a three-fingered hand, and has a positioning accuracy of ± 0.6 mm. It has a reach of 1m and can support a payload of 3kg. The vision system consists of a stereo camera (Bumblebee2, by Point Grey Research).

5 Software architecture

We are using a purpose-built distributed software framework, which we call Switchyard, to route messages between the different computers and peripherals that make up the STAIR platform. This section contains a high-level overview of the framework, focusing on the aspects important to this paper. For more details, please see [26].

Switchyard supports distributed computation through TCP message passing, providing inter-process communication across computers and operating systems. The framework is based upon modular software design, where a large software system (such as the STAIR robot) is constructed at run-time by launching many small processes and connecting them together via message passing. Thus, Switchyard software systems are essentially directed graphs. Each node in the graph is a process running on some machine, and the edges represent TCP streams between processes. The graph used for the grasping experiments is shown in Fig. 5c.

Although all nodes in the graph have a "control" connection to the master server, data transmitted between nodes flows on peer-to-peer TCP sockets. This drastically improves system performance, particularly since large robots such as STAIR often have a wide variety of connection speeds between various computers comprising the system (e.g., offboard machines are connected to each other via Ethernet, but the mobile robot is only connected via 802.11). In comparison, we found that architectures in which all data flows through a central server may route significant traffic across the slowest link in the system, particularly as the system grows in data load and number of machines connected.

There are many message-passing frameworks in existence, such as [27–29]. We developed our own framework to address the previously-mentioned issue of cluster traffic routing across heterogeneous networks, and our own practical desires for a cross-platform framework that minimizes client boilerplate code.

6 Experiments

6.1 Grasping Novel Objects

In these experiments, we asked a person to place several objects in front of the STAIR 2 robot. The bowls were placed upright at a random location on a table (with height unknown to the robot), and the plates were stacked neatly in a rack (also in a random location). Using our robotic grasping system that was trained on five types of objects (mugs, martini glasses, eraser, book and pencil), STAIR 2 achieved a grasping success rate of 60% for cereal bowls, and 80% for plates (5 trials for each object). In our earlier experiments [5,30] on STAIR 1, the grasp success rate was 87.8%.

6.2 Unloading Items from dishwasher

The goal of the STAIR project is to build a general purpose household robot. As a step towards one of STAIR's envisioned applications, in this experiment we considered the task of unloading items from dishwashers (Fig. 1a and 6). This is a difficult problem because of the presence of background clutter and the occlusion between objects—one object that we are trying to unload may physically block our view of a second object.

In this experiment, we asked a person to randomly arrange several objects neatly in the upper tray of the dishwasher. STAIR 1 used the image from the stereo camera and used our robotic grasping system to unload items from a dishwasher. In these experiments, we did not use color information, i.e., the images fed to the algorithm were grayscale. We evaluated the algorithm quantitatively on five different objects from each of four object classes: plates, bowls, mugs and glasses. (We have successfully unloaded items from multiple dishwashers; however, we performed quantitative experiments only on one

Table 1. Grasp-success rate for unloading items from a dishwasher.

Objects Tested on	Grasp-Success-rate
Plates	100%
Bowls	80%
Mugs	60%
Wine Glass	80%
Overall	80%

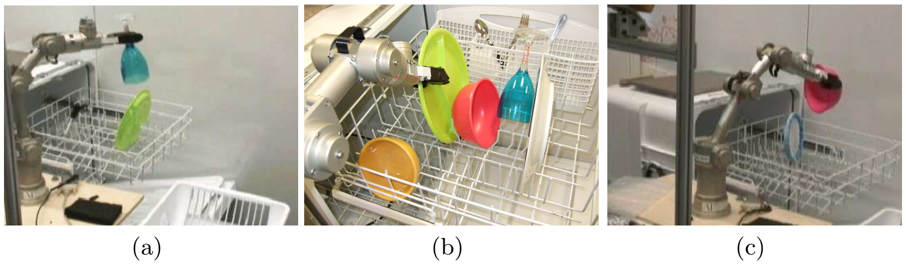


Fig. 6. Dishwasher experiments (Section 6.2): Our robotic arm unloads items from a dishwasher.

dishwasher.) In five trials for each object class (each trial used a different object), the robot was able to successfully pick up objects 80.0% of the time on average (see Table 1). Our algorithm was able to successfully pick up objects such as plates and wine glasses most of the time. Videos of our robot grasping objects and unloading items from a dishwasher are available at:

http://ai.stanford.edu/∼asaxena/learninggrasp

6.3 Fetch an object in response to verbal request

As part of a larger team effort, we demonstrate how our robotic grasping system was used, along with other tools from various fields of Artificial Intelligence—an object recognizer from computer vision [31], motion planning, and a spoken dialogue system [32]—to accomplish the task of having a robot fetch an item in response to verbal request from another room.

In particular, the following video shows one of us verbally asking a robot to fetch a stapler, in response to which the robot drives around in the lab while avoiding obstacles, finds a stapler in another room, picks it up, and brings it back to the person.

http://www.cs.stanford.edu/group/stair/multimedia.php

7 Conclusions

We presented our vision-based system for grasping novel objects in cluttered environments. Our algorithms are robust in that they can grasp objects of types not seen before, in a wide variety of new environments. We used our algorithms in the tasks of unloading items from a dishwasher and of fetching an object in response to a verbal request from another office.

The ability to pick up novel objects represents a first step towards our larger goal of enabling robots that can perform a large variety of household tasks, such as tidying up a living room after a party, and interacting with objects and tools for preparing simple meals in a kitchen. In future work, we plan to develop further perception algorithms for robotic manipulation, and thereby hopefully bring robots into more human environments; for example, we are currently working on applying variations on the algorithms described in this paper to enable STAIR to prepare simple meals using a normal home kitchen.

Acknowledgments

We give warm thanks to Brad Gulko and Justin Driemeyer for useful discussions and for help in running the experiments. We also thank Patrick Gallagher for help with the robotic hardware. This project was supported by DARPA transfer learning program under contract number FA8750-05-2-0249, Barrett Technologies, and Willow Garage.

References

1. A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," in *ICRA*, 2000.
2. M. T. Mason and J. K. Salisbury, "Manipulator grasping and pushing operations," in *Robot Hands and the Mechanics of Manipulation*. Cambridge, MA: The MIT Press, 1985.
3. K. Shimoga, "Robot grasp synthesis: a survey," *IJRR*, vol. 15, pp. 230–266, 1996.
4. A. M. Okamura, N. Smaby, and M. R. Cutkosky, "An overview of dexterous manipulation," in *ICRA*, 2000.
5. A. Saxena, J. Driemeyer, J. Kearns, and A. Y. Ng, "Robotic grasping of novel objects," in *NIPS 19*, 2006.
6. A. Saxena, J. Driemeyer, J. Kearns, C. Osondu, and A. Y. Ng, "Learning to grasp novel objects using vision," in *ISER*, 2006.
7. D. Kragic and H. I. Christensen, "Robust visual servoing," *IJRR*, vol. 22, pp. 923–939, 2003.
8. J. H. Piater, "Learning visual features to predict hand orientations," in *ICML Workshop on Machine Learning of Spatial Knowledge*, 2002.
9. J. Coelho, J. Piater, and R. Grupen, "Developing haptic and visual perceptual categories for reaching and grasping with a humanoid robot," *Robotics and Autonomous Systems*, vol. 37, pp. 195–218, 2001.
10. A. Morales, P. J. Sanz, and A. P. del Pobil, "Vision-based computation of three-finger grasps on unknown planar objects," in *IEEE/RSJ Intelligent Robots and Systems Conference*, 2002.

11. A. Morales, P. J. Sanz, A. P. del Pobil, and A. H. Fagg, “An experiment in constraining vision-based finger contact selection with gripper geometry,” in *IEEE/RSJ Intelligent Robots and Systems Conference*, 2002.
12. D. Bowers and R. Lumia, “Manipulation of unmodeled objects using intelligent grasping schemes,” *IEEE Trans on Fuzzy Systems*, vol. 11, no. 3, 2003.
13. I. Kamon, T. Flash, and S. Edelman, “Learning to grasp using visual information,” in *ICRA*, 1996.
14. A. Edsinger and C. C. Kemp, “Manipulation in human environments,” in *IEEE/RAS Int’l Conf on Humanoid Robotics (Humanoids06)*, 2006.
15. R. Platt, R. Grupen, and A. Fagg, “Improving grasp skills using schema structured learning,” in *ICDL*, 2006.
16. K. Hsiao, L. Kaelbling, and T. Lozano-Perez, “Grasping POMDPs,” in *ICRA*, 2007.
17. T. Simeona, J. Laumond, J. Cortes, and A. Sahbani, “Manipulation planning with probabilistic roadmaps,” *IJRR*, 2003.
18. A. Saxena, J. Driemeyer, and A. Y. Ng, “Learning 3-D object orientation from images,” presented in NIPS workshop on Principles of Learning Problem Design, 2007.
19. S. Thrun and M. Montemerlo, “The graphslam algorithm with applications to large-scale mapping of urban structures,” *IJRR*, vol. 25, pp. 403–430, 2005.
20. A. Saxena, S. H. Chung, and A. Y. Ng, “Learning depth from single monocular images,” in *NIPS 18*, 2005.
21. A. Saxena, J. Schulte, and A. Y. Ng, “Depth estimation using monocular and stereo cues,” in *IJCAI*, 2007.
22. A. Saxena, M. Sun, and A. Y. Ng, “Learning 3-d scene structure from a single still image,” in *ICCV workshop on 3D Representation for Recognition (3dRR-07)*, 2007.
23. ——, “3-d reconstruction from sparse views using monocular vision,” in *ICCV workshop on Virtual Representations and Modeling of Large-scale environments (VRML)*, 2007.
24. F. Schwarzer, M. Saha, and J.-C. Latombe, “Adaptive dynamic collision checking for single and multiple articulated robots in complex environments,” *IEEE Trans on Robotics and Automation*, vol. 21, pp. 338–353, 2005.
25. A. Petrovskaya, O. Khatib, S. Thrun, and A. Y. Ng, “Bayesian estimation for autonomous object manipulation based on tactile sensors,” in *ICRA*, 2006.
26. M. Quigley, E. Berger, and A. Y. Ng, “Stair: Hardware and software architecture,” in *AAAI Robotics Workshop*, 2007.
27. B. Gerkey, R. T. Vaughan, and A. Howard, “The player/stage project: Tools for multi-robot and distributed sensor systems,” in *ICAR*, 2003.
28. M. Montemerlo, N. Roy, S. Thrun, D. Haehnel, C. Stachniss, and J. Glover, “Carmen: Robot navigation toolkit,” 2000, http://carmen.sourceforge.net/.
29. Microsoft, “Microsoft robotics studio,” 2006, msdn.microsoft.com/robotics/.
30. A. Saxena, J. Driemeyer, and A. Y. Ng, “Robotic grasping of novel objects using vision,” *IJRR*, 2008.
31. S. Gould, J. Arfvidsson, A. Kaehler, B. Sapp, M. Meissner, G. Bradski, P. Baumstarch, S. Chung, and A. Y. Ng, “Peripheral-foveal vision for real-time object recognition and tracking in video,” in *IJCAI*, 2007.
32. F. Krsmanovic, C. Spencer, D. Jurafsky, and A. Y. Ng, “Have we met? MDP based speaker ID for robot dialogue,” in *InterSpeech–ICSLP*, 2006.