Active Perception: Interactive Manipulation for Improving Object Detection

Quoc V. Le
Computer Science Department
Stanford University
Email: quocle@cs.stanford.edu

Ashutosh Saxena
Computer Science Department
Cornell University
Email: asaxena@cs.cornell.edu

Andrew Y. Ng
Computer Science Department
Stanford University
Email: ang@cs.stanford.edu

***Abstract*— In this paper, we address the problem of object detection for robots. Unlike traditional object detection systems, vision in robotics is not limited to passively perceiving the environment. We propose active perception method in which the robot actively manipulates and moves in the environment in order to find an object. Our algorithm chooses actions by greedily maximizing mutual information at each step. In extensive experiments with our robots, we show that perceiving the object while actively manipulating it substantially improves the object detection performance.**

I. Introduction

For household robots, one of the basic abilities is to locate and recognize objects, such as the mug in Fig. 1. State-of-the-art vision algorithms for object recognition are constrained to work with a single image (e.g., [1]). Often, this image gives a view without any distinguishing visual features (for example, handle of a mug); this makes recognition difficult for the vision algorithms. However, when working with robots, we do not have this constraint because robots can examine the object from different views by actively manipulating it. In this paper, we will develop algorithms to enable robots to use manipulation and navigation to improve object recognition performance.

Concretely, consider the task of recognizing staplers in an office. A state-of-the-art classifier [3], trained on stapler images, performs differently on different views of the object. Fig. 2 shows the changes in probabilities of the classifier (i.e., how likely an image patch is a stapler) against 11 viewpoints of a stapler. In some viewpoints, such as the seventh viewpoint, it is very hard for the robot (and even a human) to recognize the stapler. The ninth viewpoint, however, has a high probability because we often see staplers this way, or more formally the viewpoint is *canonical* [2].

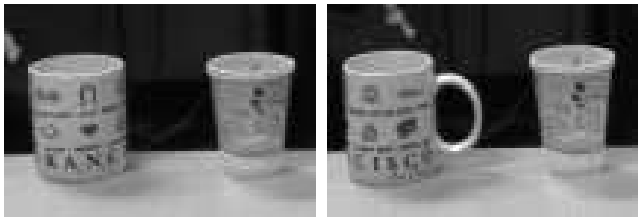


Fig. 1. It is easier to distinguish the mug from the cup in the right picture than in the left picture. The view of the mug in the right picture is called "canonical" view [2]

.

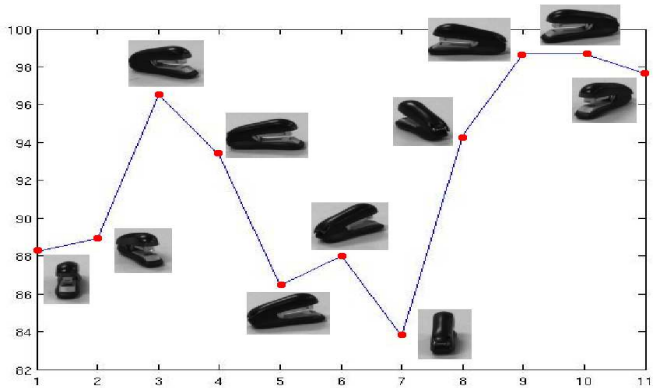


Fig. 2. Probabilities of correctly identifying staplers vs. views. X-axis indicates 11 angles of rotation; Y-axis indicates probabilities.

This large variation in probability scores makes it really hard for the object detection algorithm to perform well. If we lower the threshold (say 82%) such that the classifier can detect staplers in most views, other distracting objects in the scene (such as computer screen's corners, hammers) will be incorrectly detected as staplers. Whereas if we set the threshold to a high value, we will miss a lot of staplers in their non-canonical views.

In a sense, one of the main reasons for the low performance of object detection algorithms is that they are designed to work with static images and constrained to not being able to interact directly with the environment. We recognize objects better not only because our vision algorithm in the brain is better but also usually we have the hands to manipulate the environment. For example, in Fig. 1, a human can just rotate the object and take a second look to decide conclusively whether the object is a mug or a handle-less paper cup.

Likewise, in many cases, the robot has the freedom to change its viewpoint or manipulate the object. Consider, an example of a robot trying to locate and recognize mugs on a table. In order to recognize objects (e.g. for the purpose of fetching them), the robot can move and examine them from different viewpoints to make a more confident decision.

In our work, we show that by using active perception we get significant performance gains in object detection. In detail, we propose an active algorithm that infers a sequence of actions to maximize the *mutual information* between

actions while minimizing the *cost* of the actions. Although optimizing such criterion is a NP-complete problem, we can use a greedy approach which has theoretical guarantees and works well in practice [4].

In our system, the actions are drawn from a very rich set of manipulation and navigation actions. For manipulation, the robot can examine the objects by moving them left, right, diagonally, or rotating them in some angles. For navigation, our robot can move to a specific view towards the objects. This rich set of actions gives the robot more options when it faces a situation when some actions are infeasible or more expensive.

We use this system to enable our STAIR robot to autonomously move into new offices and find cups, mugs and staplers with very high accuracies. Our active perception algorithm substantially improves state-of-the art vision algorithms. More specifically, in our experiments with 35 objects, with pure passive perception, our robot correctly identified only 23 objects. In contrast, using the active perception method, our robot correctly identified 34 objects.

II. Active Perception

Psychologists have studied the interation between manipulation and perception. Goodale [5] describes the visual system as serving two main functions. One function of the vision is the creation of an internal model or percept of the external world, and the other is in the guidance of action, where vision directs our actions with respect to the world by transforming visual inputs into appropriate motor outputs. Many existing vision systems have focused on considering vision as sight (in contrast to being tightly integrated with action), and this is sometimes called "passive perception" [6]. The other, sometimes neglected, application of vision is in predicting what action to take next.

There are a number of examples of such active vision systems, such as foveated vision, in which the eye muscles (or the motors controlling the camera) move reactively to the visual input in a way to improve the performance of the visual task [7]. For example, if the robot finds an area that may contain an object of interest, the camera would pan and zoom-in to take a closer look and hence improve object detection significantly [8]. Paletta and Pinz [9] combined multiple views of the object (obtained by rotating the object on a turn-table without cluttered background) for improving object recognition; in their problem the cost of different actions is uniform and does not apply well to real robotic scenarios. Roy et al. [10] present a survey of such techniques; however, none of these techniques involve robots actually manipulating objects in order to recognize them better.

Other examples of active vision are methods based on multiple views, such as structure-from-motion [11] that takes multiple images (taken by an observer from different views) to construct a 3D model of a scene. SLAM (and visual SLAM) [12] also take multiple laser scans to make a 3D map of the environment; sometimes the map is also tagged with visual information [13].

In humans, this tight coupling between action and perception is seen in many other cases. For example, a child learns about properties of an object when she holds it in her hand and examines it by looking at it from different angles [14]. Metta and Fitzpatrick [15] discuss that vision and manipulation are inextricably intertwined in the primate brain, and Hurley [16] proposes a "shared circuit model" for human brain for considering the dynamics between action and perception. While an experienced adult can interpret visual scenes perfectly well without acting upon them, linking action and perception seems crucial to the developmental process that leads to that competence.

One of the reasons of the tight coupling between action and perception is because a fixed view provides only limited amount of information. The idea that visual recognition performance is view-point specific has been well established [17], [18], [19]. In fact, neurophysiological studies have revealed cells in inferior temporal cortex that are maximally tuned to specific views of objects [20]. This is closely related to the idea of "canonical views" [2] in the field of computer vision. For example, the canonical view of a mug is when its handle is on one side (Figure 1-right); it is not a canonical view when the handle is not visible (Figure 1-left). Canonical views provide unique visual features that are easier for an algorithm (or a human) to recognize [21]. Objects in non-canonical views (or views for which training data is not readily available) are hard to detect [22], [23].

In other related work, Gupta and Davis [24] build a classifier for recognizing objects present in videos using human actions as additional information. Some works in robot manipulation use sensory data other than vision to infer actions. For example, Jain and Kemp [25] monitor the haptic sensory input to decide on strategies for opening doors. Kober and Peters [26] used policy search to learn optimal control policy for robotic manipulation tasks. Hsiao et al. [27] used a belief-based method to robustly choose and execute manipulation actions. Hart, Sen and Grupen [28] use a factorable approach to address generalization and transfer in sensorimotor programs. However, none of these works explicitly consider action and perception together for improving object detection. One notable exception is [29], where Katz et al. use manipulation to infer kinematic information about articulated objects (such as scissors). However, Katz et al. do not choose an optimal manipulation action and it does not apply to improve object detection directly. In our work, we propose a general framework which uses robot capabilities such as navigation and manipulation to improve object detection. We also use a state-of-the-art computer vision algorithm and present empirical results for active perception.

III. Our approach

Our approach focuses on the fact that a mobile robot has the ability to take actions to perceive different views of the object. For example, our robot can do this either by moving to a different location (using its wheels) or by manipulating the object (using its arm).

In our method, the action and perception work in tandem. Given an image of the scene, the goal of our vision algorithm is not only to detect what objects there are in the scene, but also to infer next action such that the object can be identified quickly. In an example, the handle of the mug is not visible in Fig. 1-left; so a good action would be one that rotates it in a way to make the handle visible, as in Fig. 1-right. This way, with only two actions, the robot can tell apart whether the object is a mug or not.

A. Random strategy

One strategy to solve this problem is to perform a series of random actions. Even though this strategy may provide more views of the object, those views of the object may not be very different from the original one and hence not helping much in the detection task.

In fact, in the worst cases, the robot has to perform a large number of actions in order to recognize the object. Our goal, therefore, is to develop a more intelligent strategy that can potentially recognize an object with very few actions. One such strategy is described in the following section.

B. Optimal strategy formulation

Our goal is to select a sequence of actions that gives us most information to recognize an object. Some actions, however, are very expensive to perform by a robot; therefore we will take into account the cost of each action for choosing the optimal action sequence as follows:

$$\text{actions}^* \leftarrow \text{argmax}_{\text{actions}} \frac{\text{information}(\text{actions})}{\text{cost}(\text{actions})} \tag{1}$$

Here, actions* is the optimal series of actions which are selected from a set of feasible actions. The quantity information(actions) represents information obtained by performing a sequence of actions where as cost(actions) represents the cost of performing such sequence.

This optimization problem requires us to know the cost of actions. In normal circumstances, the cost of action can be easily computed based on metrics such as the time to perform an action (or whatever metric is most relevant for the robot). The quantity information(actions), however, is object-dependent and more difficult to compute.

First, we have to simplify its definition. In the task of object detection, the relevant information is the information about views because it is the new views of the object that provide more information helping in the task of object detection, i.e., information(views).

The set of possible actions by our robot are navigation and manipulation actions (such as moving the arm in a particular way). Therefore, in this paper, we assume that an action results in a deterministic consequence. Multiple actions, however, could lead to the same consequence. For example, in order to change the view by 90° the robot could navigate to a new location (which is often not possible because of obstruction by the furniture on which the object is lying) or it could use its hand to rotate the object.

This allows us to divide the inference problem in Eq. 1 in two sub-problems:

$$\text{views}^* \leftarrow \text{argmax}_{\text{views}} \frac{\text{information}(\text{views})}{\min_{\text{actions(views)}} \text{cost}\left(\text{actions}(\text{views})\right)} \tag{2}$$

$$\text{actions}^* \leftarrow \text{argmax}_{\text{actions(views}^*)} \frac{1}{\text{cost}\left(\text{action}(\text{views}^*)\right)} \tag{3}$$

Here, we use actions(views) to mean the actions that can lead to a particular view of the object. This division would enable us to learn the parameters for calculating information(views) from offline data; to compute parameters for information(actions) would have required an actual robot to perform experiments.

Informally, the two equations say that the optimal views are ones that are most informative while having the lowest costs; furthermore, if there are multiple actions that lead to the same consequence, the cheapest actions will be chosen.

This combinatorial optimization problem can be very difficult to solve. One solution is to employ approximation algorithms in [4] which is previously employed to solve the sensor-placement problem. In detail, if we assume a uniform cost for actions, the equation simplifies to

$$\text{views}^* \leftarrow \text{argmax}_{\text{views}} \text{information}(\text{views}) \tag{4}$$

In the sensor-placement problem, the goal is to place a number of sensors such that we can obtain the most information possible. There can be many possible criteria for measuring the information, but a good choice is to use mutual information. More specifically, Krause et al. [4] propose to maximize the *mutual information* between readings from a new location and readings from the already chosen location.

Likewise, in our problem, we would like to obtain a sequence of views such that we can maximize the mutual information between information given at the new views and information at the old views. More specifically, we would like the robot to determine a sequence of actions that maximizes the mutual information to confirm whether an object belongs to an object class . For example, if the robot looks at a vertical stapler, it should aim to find best angles to move with respect to the object such that it can correctly determine that it is a stapler. In robotics, however, it is important that we take into account the fact that some actions take more time and effort than others. Motivated by [4], we will develop an algorithm that takes into account the non-uniform cost of the actions.

C. Near-optimal strategy with mutual information using Gaussian Processes

We will first formalize our problem. Our task is to obtain k views out of a finite set (discretized) $\mathcal{V}$ of possible views. We would like to obtain a set of observed views $\mathcal{O}$ such that

$$\mathcal{O}^* = \arg \max_{\mathcal{O} \subset \mathcal{V}: \|\mathcal{O}\| = k} I(\mathcal{O}; \mathcal{V} \backslash \mathcal{O}) \tag{5}$$

where I is the mutual information. Also $I(\mathcal{O}; \mathcal{V} \backslash \mathcal{O}) = H(\mathcal{V} \backslash \mathcal{O}) - H(\mathcal{V} \backslash \mathcal{O} | \mathcal{O})$, where H is the entropy.

Unfortunately, maximizing this criterion is NP-complete. However, Krause et al. [4] propose a polynomial-time greedy approximation algorithm with approximation factor $(1 - 1/e)$. Their insight is that instead of choosing all viewpoints at once, they can greedy pick one view Y at a time that maximizes

$$H(Y|\mathcal{O}) - H(Y|\mathcal{V}\backslash(\mathcal{O} \cup Y)) \tag{6}$$

where the entropy can be computed from the covariance matrix of Gaussian process regression.

In our problem, we need to take into account the non-uniform cost of obtaining a view. If $\mathcal{S}(Y)$ is the set of actions that result in the view Y, our goal is to find a set of actions $\mathcal{A}$ by maximizing the following:

$$\frac{H(Y|\mathcal{O}) - H(Y|\mathcal{V}\backslash(\mathcal{O} \cup Y))}{\text{cost}(Y)} \tag{7}$$

where $\text{cost}(Y) = \min_{X \in \mathcal{A}(Y)} \text{cost}(X)$. Therefore, the algorithm for obtaining the views $\mathcal{O}$ and actions $\mathcal{S}$ is given below.

Here, the σ and Σ are parameters of Gaussian Processes, learned from labeled data (discussed later) [30]. Let us assume we wish to measure the probability of an object being a stapler given a view. We can perform this task by a Gaussian Process regression over the view with the output being the probability which is real-valued. Gaussian Process regression gives us the mean vector μ and covariance Σ.

D. View initialization

There is a further subtle difference between the sensor-placement problem and our view-selection problem. In our view selection problem, we do not know the orientation of the object with respect to the camera. Therefore, we need to estimate this orientation at the beginning and use the estimate to perform subsequent actions to obtain desired views. We estimate the orientation (represented using a method described in [31]) by training another Gaussian Process on the orientation of object with respect to the camera.

IV. Implementation details

In this section, we will walk through some necessary elements to implement the above ideas on a robot. Specifically, we will discuss our object detection algorithm, our set of manipulation and navigation actions and associated cost functions. We finally explain how to improve the algorithm when the environment is cluttered.

A. Object detection algorithms

We used a state-of-the-art sliding-window method for object detection [3]. In detail, a candidate rectangular area is selected and classified for possible objects, and then this process is repeated for a number of windows over the image. At each step of the sliding window, we compute features from a collection of small patches. We use a learned "patch dictionary" to classify the window.

For each window, an object detection classifier gives a real-valued score. The threshold to detect the object is a tradeoff between two important measures: Precision and Recall. To make this choice fair, we estimated the value of the threshold using a cross-validation set for optimizing the F1-score.

TABLE I

List of actions

Action	Category
Move hand left/right	Manipulation
Move hand back/forth	Manipulation
Move hand diagonal	Manipulation
Grasp and turn	Intricate Manipulation
Rotate wrist by θ	Manipulation
Move robot to new location	Navigation

B. Manipulation and navigation actions and their cost functions

At any stage during object detection, the robot needs to take different actions in order to get try to recognize the object more accurately. A way for the robots to approach this problem is to grasp the objects and then rotate the objects to the desired orientations [32], [33], [34], [35]. This approach, however, is restrictive because grasping sometimes can be very difficult for unknown objects.

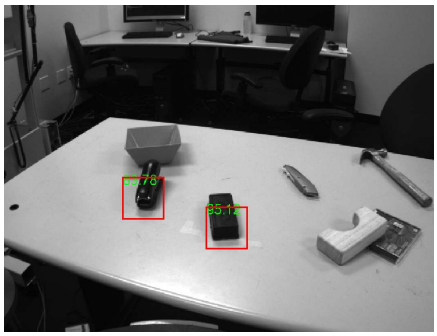
In our method, we consider a rich set of simpler manipulation and navigation actions. Manipulation actions contain primitive actions that can be performed by a robot arm such as moving the arm/hand left, right, diagonally or rotating some specific joints in some desired degrees. We also considers navigation actions such as moving the base to a specific location (using dead-reckoning).

A summary of the actions that we implemented on our robot is listed in Table I. Often some actions are not feasible, e.g., when navigation is not feasible when the environment is blocked by obstacles. Since our robot has a rich set of actions available, it has more chances of succeeding in finding a reasonable action. We discretized these actions to keep the number of actions tractable. We also discretized the views around an object to 11 angles.

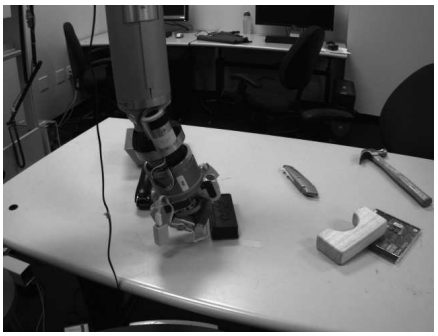
A number of cost functions have been proposed in prior work depending on the task, such as minimizing the total energy, minimizing the total distance moved (more relevant for purely navigational robots), minimizing certain measures based on kinematics/dynamics of the hand (useful for humanoids for example). In our case, we simply associate the cost of actions to time. There are some actions with infinite cost as well, for example, the robot cannot move to a location that is obstructed. We can make of use data from sensors such as camera and laser to obtain such costs.

C. Speed up for multiple objects

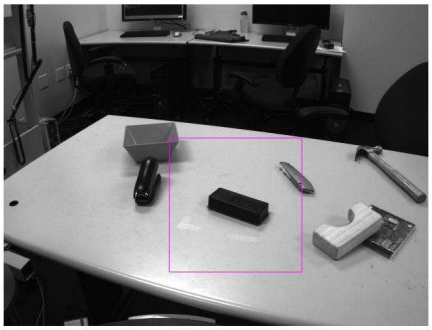
Consider a robot facing a scene with multiple objects, and the goal is to classify each object present in the scene. We can address this problem by maximizing the mutual information in Eq. 7. In actual implementation for running experiments on the robot, we employ two optimization ideas to speed up the process. First, we do not consider any candidate object whose probability score is very low (to make it faster for the algorithm to compute probabilities, etc.). Second, if one



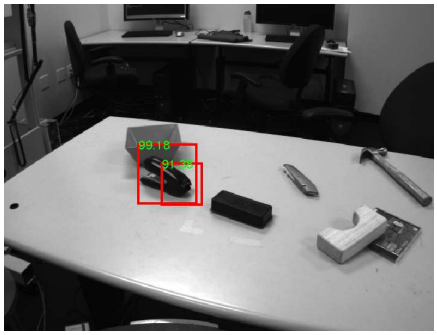
(a) First view of the scene. Which one of the objects shown in red boxes is a stapler?



(b) The robot perform actions to rotate them in order to get better views.

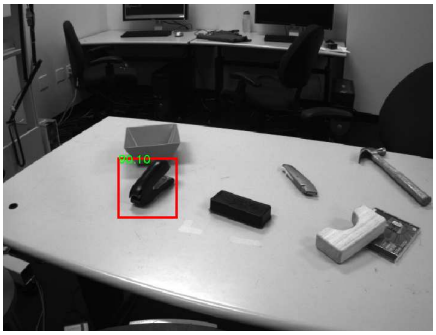


(c) View after first action (using our method).



(d) View after second action (our method)

(e) Robot executes a randomized strategy



(f) Randomized strategy in step 2 results in low probability of detection

Fig. 3. An experiment showing the robot finding an object (stapler) on a table (Best view in color). (a) The probability (given by passive vision) of being a stapler for the object on the right (a eraser) is 95.12% and for the object on the left is 83.78%. The viewpoint detector says they both look like staplers oriented at 90° (7th viewpoint). (b) In Fig. 2, among possible actions that lead to viewpoints (5-9), rotating the object by 45° gives the highest mutual information and has cheapest cost. (c) In the new view, the detector does not find any stapler inside the specified window. (d) Robot performs the second action on the object on the left, and The object detector does find a stapler in the scene with probability 99.18. (e) For comparison, robot executing a randomized strategy. (f) The object detector may or may not recognize the stapler. In this case it works but it gives probability of 90.10.

Algorithm 1 Greedy viewpoint selection

$\mathcal{O} \leftarrow \emptyset$
$\mathcal{S} \leftarrow \emptyset$
for $j = 1$ to k **do**
  $Y^* \leftarrow \arg\max_{Y \in \mathcal{V} \setminus \mathcal{O}, \bar{\mathcal{P}} = \mathcal{V} \setminus (\mathcal{O} \cup Y)} \frac{1}{\text{cost}(Y)} \frac{\sigma_Y^2 - \Sigma_{Y\mathcal{O}} \Sigma_{\mathcal{O}\mathcal{O}}^{-1} \Sigma_{\mathcal{O}Y}}{\sigma_Y^2 - \Sigma_{Y\bar{\mathcal{P}}} \Sigma_{\bar{\mathcal{P}}\bar{\mathcal{P}}}^{-1} \Sigma_{\bar{\mathcal{P}}Y}}$
  $X^* = \arg\min_{X \in \mathcal{A}(Y)} \text{cost}(X)$
  $\mathcal{O} \leftarrow \mathcal{O} \cup Y^*$
  $\mathcal{S} \leftarrow \mathcal{S} \cup X^*$
end for

of the objects gets very high classification probability (i.e., the algorithm is very sure about its prediction), we remove it from our optimization objective.

V. Hardware

We use STanford AI Robot (STAIR) for our experiments. The STAIR platform consists of a mobile base, with four steerable wheel turrets, allowing arbitrary 2D translations and rotations. Its 7-DOF arm (WAM, by Barrett Technologies) can be positioned or torque-controlled. The arm is also equipped with a three-fingered hand and has a positioning error of ± 0.6 mm. It has a maximum reach of 1m and can support a payload of 3kg. The robot is mounted on a omnidirectional mobile Segway base. We use a Flea camera for our vision system.

VI. Experiments

Our goal is to improve object detection in robots, so that they become capable of performing tasks such as fetching objects from homes or offices, and manipulating objects in these environments. We considered three types of indoor objects, namely mugs, staplers, and handleless coffee cups.

For training, we collected 150 images of cluttered office scenes. We labeled the objects of interest and their corresponding views. This labeled data was used to learn the parameters of Gaussian Processes in our algorithm.

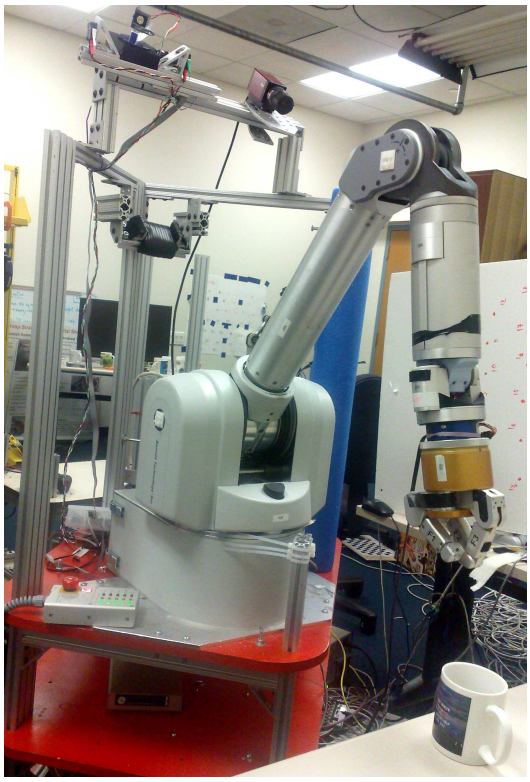


Fig. 4. Our STAIR robot on which we tested our algorithm.

A. Experimental Procedure

In order to evaluate our active perception method, we compare it with the passive method, where the robot sees the environment from a single viewpoint and does not perform any manipulation (the standard object detection used in computer vision). We also compared it against a randomized strategy where the robot performs random actions in order to get more views. The randomized strategy we implemented in our robot is actually smart in the sense that it only considers actions that i) are feasible to the robot at a situation and ii) lead to some changes in the viewpoint.

For each experiment, the robot has the goal of detecting a given object (one of mugs, staplers and cups). The experiments were done in an office with many distracting objects (see Fig. 8 and 9). During the experiments, the algorithm was tested on 15 staplers, 10 mugs and 10 cups, which varied significantly in the shape and appearance and were different from the ones trained on. There were a significant number of distracting objects in the scenes such as white-board erasers, jugs, hammers, scissors, computers, joysticks which share similar looks with our objects of interest.

B. Results

We first test our method on three most commonly-used measures in object detection: Precision, Recall and F1 scores. Tables II, III, IV show the results for staplers, mugs, and cups respectively after three consecutive actions taken by the robot. The results show that our method for active perception performs better than both passive perception and active perception with a randomized strategy. It takes more steps for a randomized strategy to perform reasonably as compared to our method.

TABLE II

STAPLER DETECTION RESULT, A COMPARISON BETWEEN PASSIVE, ACTIVE AFTER THREE STEPS AND RANDOMIZED ACTIVE AFTER THREE STEPS. THE NUMBER OF ACTION STEPS IS SHOWN IN PARENTHESIS.

METHODS (NO. OF ACTION-STEPS)	PRECISION	RECALL	F1
PASSIVE	0.62	0.53	0.57
RANDOMIZED ACTIVE (1)	0.54	0.93	0.68
RANDOMIZED ACTIVE (2)	0.78	0.93	0.85
RANDOMIZED ACTIVE (3)	0.93	0.93	0.93
OUR METHOD (1)	0.70	0.93	0.80
OUR METHOD (2)	0.88	0.93	0.90
OUR METHOD (3)	1.00	0.93	0.97

TABLE III

MUG DETECTION RESULT, A COMPARISON BETWEEN PASSIVE, ACTIVE AFTER THREE STEPS AND RANDOMIZED ACTIVE AFTER THREE STEPS. THE NUMBER OF ACTION STEPS IS SHOWN IN PARENTHESIS.

METHODS (NO. OF ACTION-STEPS)	PRECISION	RECALL	F1
PASSIVE	0.80	0.80	0.80
RANDOMIZED ACTIVE (1)	0.45	1.00	0.63
RANDOMIZED ACTIVE (2)	0.67	1.00	0.80
RANDOMIZED ACTIVE (3)	0.83	1.00	0.91
OUR METHOD (1)	0.63	1.00	0.77
OUR METHOD (2)	0.83	1.00	0.91
OUR METHOD (3)	0.91	1.00	0.95

TABLE IV

CUP DETECTION RESULT, A COMPARISON BETWEEN PASSIVE, ACTIVE AFTER THREE STEPS AND RANDOMIZED ACTIVE AFTER THREE STEPS. THE NUMBER OF ACTION STEPS IS SHOWN IN PARENTHESIS.

METHODS (NO. OF ACTION-STEPS)	PRECISION	RECALL	F1
PASSIVE	0.78	0.70	0.74
RANDOMIZED ACTIVE (1)	0.53	1.00	0.69
RANDOMIZED ACTIVE (2)	0.83	1.00	0.91
RANDOMIZED ACTIVE (3)	1.00	1.00	1.00
OUR METHOD (1)	0.63	1.00	0.77
OUR METHOD (2)	0.83	1.00	0.91
OUR METHOD (3)	1.00	1.00	1.00

To further illustrate how the actions improve object detection measures, we plot the measures as a function of action in active vision and compare with baseline passive vision. This comparison is shown in Figure 5.

In the experiments, a total of 35 objects (staplers, cups, mugs) were placed in an office environment amongst clutter and confusing objects, and the robot had to find all the staplers, cups and mugs.[1] Pure passive vision found 23 objects correctly, but also found 9 incorrect objects. Active vision with randomized strategy found 34 objects correctly, but found 3 incorrect objects. Our active perception method found 34 objects, but found only one incorrect object. We note that the stapler that our robot failed to recognize was rather difficult to detect even by a human (confusing with white-board eraser).

To further analyze the performance of active vision , we picked a stapler (Fig. 2), recorded and showed how

[1]It was not told how many "correct" objects are in the scene.

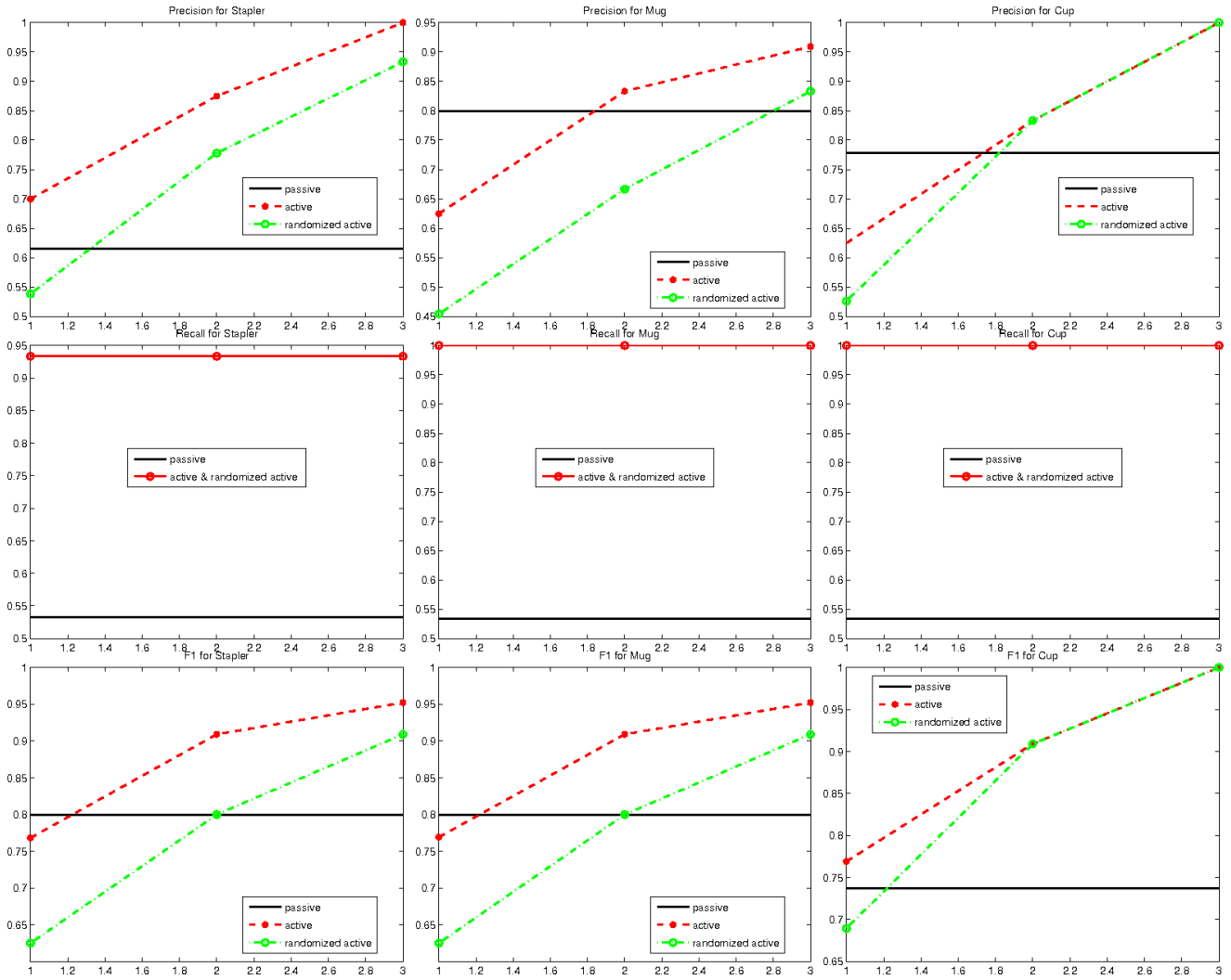


Fig. 5. Precision, Recall and F1 score for Staplers Mugs and Cups. Top row: Precision, Second row: Recall, Bottom row: F1. Left column: Stapler, Middle column: Mug, Right column: Cup

probabilities of correctly detecting a stapler of a function of viewpoint after 3 actions (Fig. 6). As can be seen, after three actions, the probabilities of correctly detecting a stapler are very high.

We also plotted how probabilities of correctly detecting a mug as a function of viewpoint in Figure 7. As can be seen from this figure, for object detection algorithms, orientation matters significantly in terms of its performance. Interestingly, together with Fig 2, Fig 7 shows that probabilities are higher for right-handed orientations than left-handed orientations. This is because the training data have more mugs/staplers in the right-handed orientation, because there are more right-handed mug/cup/stapler users and they tend to place mug with its handle on right-hand side in the training set.

The experiments show that using only passive vision, it is hard to find objects of interest when they are in unusual orientations. Even when we can train one classifier for each orientation, in unusual orientations, such as staplers facing vertically, the objects can be easily confused with other

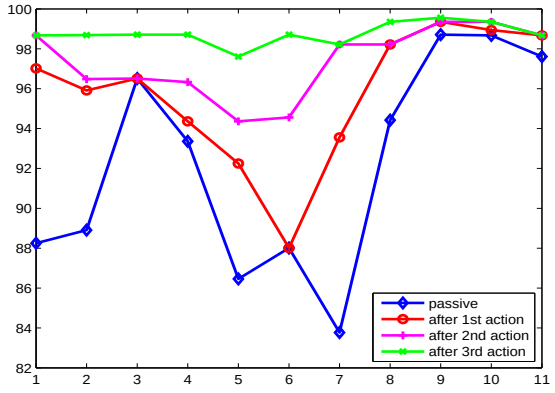


Fig. 6. Changes in probabilities for optimal actions.

objects with similar appearance [36]. More specifically, unusual orientations lead to non-discriminative visual features which cannot be used to classify similar objects. In contrast,

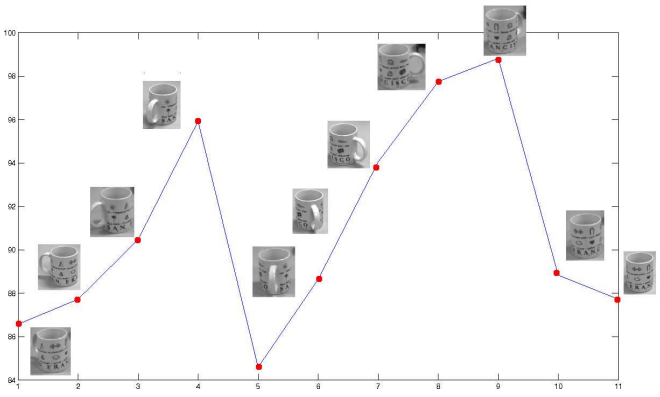


Fig. 7. Probabilities vs viewpoints for mugs.

active vision can solve this problem easily by changing the viewpoint.

An example of sequences of actions performed by robots to remove the confusion are shown in Fig. 3. In detail, these cases show an example that the robot addresses the confusion between a stapler and by moving them with an arm; cases in Fig. 8 show an example that robot moves the base to a different locations to check each stapler. Similarly, cases in Fig. 9 show an instance that robot addresses the confusion for mugs and cups by moving to opposite sides of the table. The robot decides on which actions to take (from a number of available options in manipulation and navigation) depending on what will provide it with better accuracies and the cost of the action.

Furthermore, we would like to emphasize that the randomized strategy requires much more time and hence incurs much more cost than the near-optimal algorithm. In our robot, on average, the randomized strategy spends three times as much time compared to our method. Taking time into an account, we can see that our method does not only achieve superior accuracies in detecting objects but also detect much faster than the competitive randomized strategy.

VII. Conclusion

We addressed the problem of object detection for robots. We proposed active perception methods in which the robot actively manipulates and moves in the environment in order to find an object. Our algorithm chooses actions by greedily maximizing mutual information at each step. In extensive experiments with our robots with manipulation and navigation capabilities, we show that our method substantially improves the object detection performance. Prior to this project, our robot usually misses a lot of objects of interest in the task of "fetch a mug/cup/stapler." Using the algorithm developed in this paper, our robot is able to robustly find and fetch mugs, cups and staplers with very high accuracies.

References

[1] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results," http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html.

[2] S. Edelman, *Representation and recognition in vision*. Oxford University Press, 2001.

[3] W. F. A. Torralba, K. Murphy, "Sharing visual features for multiclass and multiview object detection," in *NIPS*, 2007.

[4] A. Krause, A. Sigh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and emprical studies," *JMLR*, 2008.

[5] M. A. Goodale and G. K. Humphrey, "The objects of action and perception," *Cognition*, vol. 67, pp. 181–207, 1998.

[6] W. Weimer, *A conceptual framework for cognitive psychology: motor theories of mind*. R. Shaw, J. Bransford (Eds), Perceiving Acting and Knowing: Toward an Ecological Psychology. Lawrence Erlbaum Associates, Hillsdale, NJ, 1977.

[7] M. Ebner and A. Zell, "Centering behavior with a mobile robot using monocular foveated vision," *Robotics and Autonomous Systems*, 2000.

[8] 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.

[9] L. Paletta and A. Pinz, "Active object recognition by view integration and reinforcement learning," *Robotics and Autonomous Systems*, vol. 31, pp. 71–86, 2000.

[10] S. Roy, S. Chaudhary, and S. Banerjee, "Active recognition through next view planning: a survey," *Pattern Recognition*, vol. 37, pp. 429–446, 20034.

[11] R. Hartley and A. Zisserman, *Multiview Geometry in Computer Vision*. Cambridge University Press, 2000.

[12] S. Frintrop and P. Jensfelt, "Active gaze control for attentional visual slam," in *ICRA*, 2008.

[13] I. Posner, M. Cummins, and P. Newman, "Fast probabilistic labeling of city maps," in *Robot Science and Systems*, 2008.

[14] P. Rochat, "On the way to literacy: early experiences for visually impaired children," *Developmental Psychology*, vol. 25, 1989.

[15] G. Metta and P. Fitzpatrick, "Early integration of vision and manipulation," *Adaptive Behavior*, 2003.

[16] S. Hurley, "The shared circuits model (scm): How control, mirroring, and simulation can enable imitation, deliberation, and mindreading," *Behavioral and Brain Sciences*, vol. 31, no. 1, pp. 1–22, 2008.

[17] P. Jolicoeur, "The time to name disoriented objects," *Memory and Cognition*, vol. 13, pp. 289–303, 1985.

[18] S. Palmer, E. Rosch, and P. Chase, "Canonical perspective and the perception of objects," *Attention and Performance IX*, 1981.

[19] F. Newell and J. Findlay, "Effects of depth rotation on object identification," *Perception*, vol. 26, pp. 1231–1257, 1997.

[20] N. Logothetis, J. Pauls, and T. Poggio, "Shape representation in the inferior temporal cortex of monkeys," *Current Biology*, vol. 5, pp. 552–563, 1995.

[21] V. Blanz, M. Tarr, H. Buelthoff, and T. Vetter, "What object attributes determine canonical views?" *Perception-London*, 1999.

[22] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool, "Towards multi-view object class detection," in *CVPR*, 2006.

[23] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *CVPR*, 2004.

[24] A. Gupta and L. Davis, "Objects in action: An approach for combining action understanding and object perception," in *CVPR*, 2007.

[25] A. Jain and C. C. Kemp, "Behaviors for robust door opening and doorway traversal with a force-sensing mobile manipulator," in *RSS Manipulation Workshop: Intelligence in Human Environments*, 2008.

[26] J. Kober and J. Peters, "Policy search for motor primitives in robotics," in *NIPS*, 2009.

[27] K. Hsiao, T. Lozano-Perez, and L. P. Kaelbling, "Robust belief-based execution of manipulation programs," in *WAFR*, 2008.

[28] S. Hart, S. Sen, and R. Grupen, "Generalization and transfer in robot control," in *Epirob*, 2008.

[29] D. Katz, Y. Pyuro, and O. Brock, "Learning to manipulate articulated objects in unstructured environments using a grounded relational representation," in *Proceedings of Robotics: Science and Systems*, 2008.

[30] C. E. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[31] A. Saxena, J. Driemeyer, and A. Y. Ng, "Learning 3-D object orientation from images," in *ICRA*, 2009.

[32] ——, "Robotic grasping of novel objects using vision," *IJRR*, 2008.

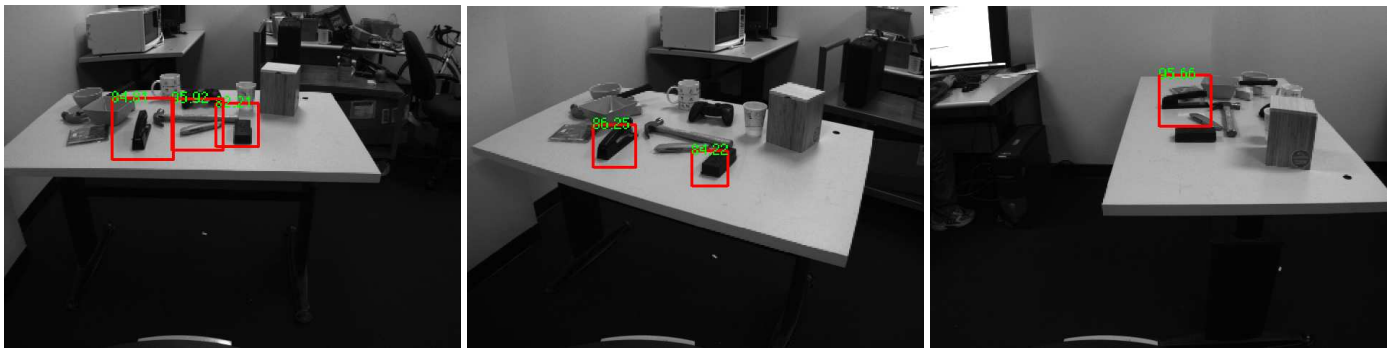


Fig. 8. Left: Robot sees three 'stapler' candidates. Middle: Robot chooses to move right to get more information for the middle object. Other staplers are still recognized but the middle object is correctly classified as not a stapler. Right: Robot move further right. Even though this viewpoint by itself may get less information, it gives a novel viewpoint with the help of which the robot correctly finds the stapler. (Also notice that it is easier for the robot to navigate in this experiment because of the available free space.)

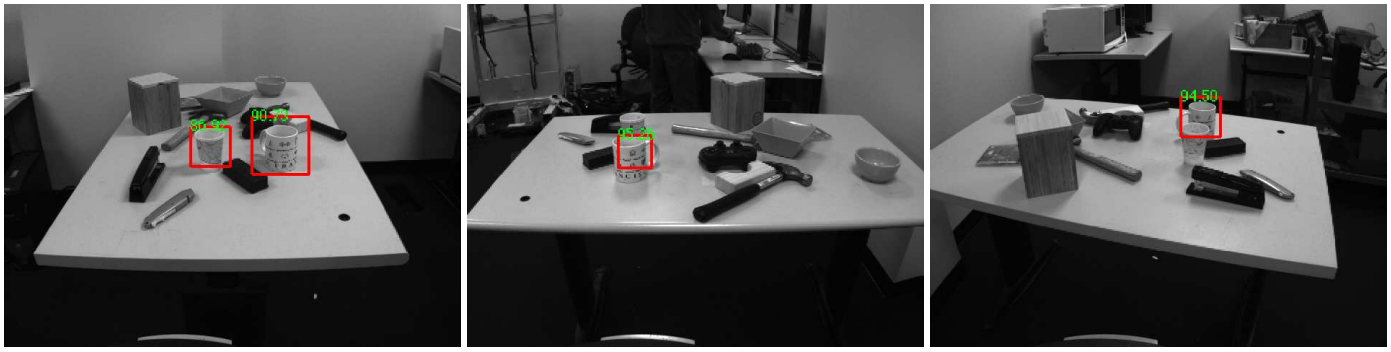


Fig. 9. Left: Robot sees two candidates of mugs. Middle: Robot moves to the right to check the objects and only recognizes the right object as a mug. Right: Robot moves to the other side when the mug is occluded by the cup, but it still correctly classifies it.

[33] A. Saxena, L. Wong, and A. Y. Ng, "Learning grasp strategies with partial shape information," in *AAAI*, 2008.

[34] Q. V. Le, D. Kamm, A. Kara, and A. Y. Ng, "Learning to grasp objects with multiple contact points," in *ICRA*, 2010.

[35] D. Rao, Q. Le, T. Phoka, M. Quigley, A. Sudsang, and A. Ng, "Grasping novel objects with depth segmentation," in *IROS*, 2010.

[36] A. Thomas, V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Gool, "Towards multi-view object class detection," in *CVPR*, 2006.