Utilizing object-object and object-scene context when planning to find things

2009 IEEE International Conference on Robotics and Automation
Kobe International Conference Center
Kobe, Japan, May 12-17, 2009

Utilizing *object-object* and *object-scene* context when planning to find things

Thomas Kollar and Nicholas Roy

Abstract— **In this paper, our goal is to search for a novel object, where we have a prior map of the environment and knowledge of some of the objects in it, but no information about the location of the specific novel object. We develop a probabilistic model over possible object locations that utilizes *object-object* and *object-scene* context. This model can be queried for any of over 25,000 naturally occurring objects in the world and is trained from labeled data acquired from the captions of photos on the Flickr website. We show that these simple models based on object co-occurrences perform surprisingly well at localizing arbitrary objects in an office setting. In addition, we show how to compute paths that minimize the expected distance to the query object and show that this approach performs better than a greedy approach. Finally, we give preliminary results for grounding our approach in object classifiers.**

I. Introduction

The goal of this work is to understand natural language interactions where a person asks the robot to find a novel object, and the robot must search through the environment in order to find the object. In principle, the novel object can be any of thousands of types and could be located in hundreds of places in a given environment. Reasoning about the location of objects usually relies on specialized object detectors that perform well at detecting the goal object. In order to find the object, the robot might search through the environment using a chosen exploration strategy, building a map as it goes. At the same time, the robot passively or actively uses an object detector until it finds the object. Finally, it might register the location of the query object to a global map of the environment.

However, a search that does not take into account the structure of natural environments will be inefficient and arbitrary. Instead of having a single object detector, if the robot has an array of detectors for different objects, it can utilize the fact that some objects tend to co-occur, or reside in certain kinds of places in the environment but not in others. For example, given that the robot has detected a *sofa*, this may make a *remote control* much more likely, and vice-versa. Similarly, given that the robot has detected a *sofa* and a *remote control*, this increases the likelihood that the scene is a *living room*, which in turn increases the likelihood of detecting a *television*. We use the term "scene" to denote the type of an environment, such as a kitchen, a living room, etc. Using this idea, we can use a set of object detectors in order to recognize scenes and predict the location of objects that the robot has never encountered before.

T. Kollar and N. Roy were supported by the Office of Naval Research under MURI N00014-07-1-0749. Their support is gratefully acknowledged. Thomas Kollar is a PhD candidate at CSAIL and Nicholas Roy is Faculty in the Department of Astronautics and Aeronautics, Massachusetts Institute of Technology, Cambridge, MA, 02139; United States. {tkollar, nickroy}@mit.edu

In this work, we will assume access to a number of object detectors. Given these detectors, we will show how to use *object-object* and *object-scene* context in order to localize any of the 25,000+ objects scenes in the English language in natural environments[1]. We will show both simulated and real-world experiments that use just a small subset of detectable objects and scenes in order to robustly predict the location of a significant number of goal objects. We will additionally show preliminary results that use category-level visual object detectors instead of simulated ones. Finally, we will propose a method to search for a novel object by minimizing the expected length of the path to a goal object.

The contribution of this work is twofold. First, we will show that by using *object-object* and *object-scene* context learned from captions attached to photos on Flickr, we can robustly predict the locations of a wide variety of other objects and scenes. Secondly, incorporating these predictions into the search process and choosing paths that minimize the expected length to the goal object, we are able to find the object more quickly than a greedy approach. Throughout this paper, our primary thesis is that strong positive or negative correlations between objects in an environment give strong priors on the locations of other objects.

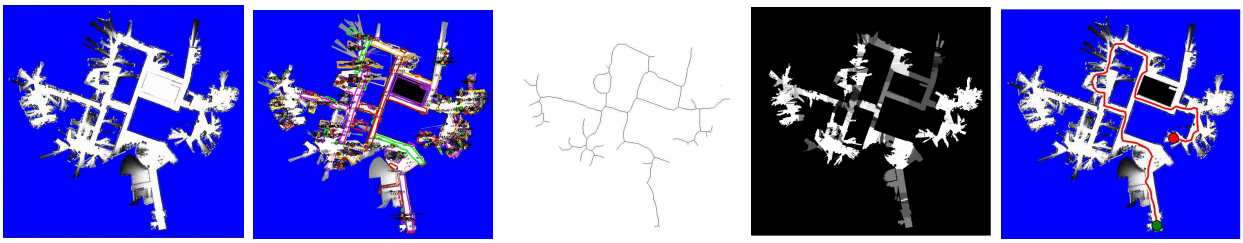
The paper is organized as follows. In section II we will give an overview of our approach to finding novel objects. In section III, we formalize the problem of inferring object location and describe a probabilistic model that utilizes context in order to predict the existence of novel objects. In section IV, we formalize the problem of searching for an object and show how to optimize the path of the robot in order to find this object. In section V, we show the results from a number of experiments, both simulated and using real-world data. Finally, in sections VI and VII, we will give the related work, conclusions and future work.

II. Overview of approach

Our proposed approach is to utilize objects and scenes that the robot knows about in order to predict the existence of new objects. Our underlying assumptions are that we have a robot that is taking odometry measurements, laser measurements and camera images as it travels through its environment. Our proposed algorithm is given in figure 1.

In step 1, the robot will build a map and using the associated object detections at each location in its trajectory, it will place annotations in the map where each object was

[1]The number of objects was computed by taking all scenes, objects, and animals from the WordNet database [8]. In [1], the authors estimate one to two thousand of *concrete* nouns, but this does not take into account any ambiguity in the way people communicate. Although this ambiguity might be resolved with a carefully constructed semantic network, we expect unstructured queries where this resolution may not be possible.



(a) Step 1: Build a map (b) Step 2: Recognize objects (c) Step 3: Medial-axis transform (d) Step 4: Likelihood map (e) Step 5: Plan a path

Fig. 1. This figure illustrates the proposed algorithm. In step 1 and step 2 a map is built of the environment and objects are registered in it (alternatively, a person labels the of the objects). In step 3 the map is skeletonized using the medial-axis transform. In step 4, for each gridcell in this skeletonized map, the likelihood of finding the query object is computed. Finally, in step 5, the likelihood map is used along with a start location in order to plan a path to the goal object. In this map, the labeled objects include: *stairs, desk, kayaks, door, printer, couch, bike, whiteboard, fish, computer, secretary, workarea, trashcan, coffeemug, stapler, pencil, book, wine, keyboard, helicopter, chair, drill, espresso, coffeemaker, sink, trash, microwave, tv, paper, towel, cellphone, robot, copier, plant, remotecontrol, refrigerator, soap, solder*.

observed (step 2). Given a query object, in step 3 the robot will compute a skeleton of the map using the medial-axis transform to reduce the size of the inference and planning problems. In step 4, the likelihood of finding the object at each location in the skeletonized map is computed using the labeled locations of the known objects. Finally, in step 5, given a starting location, a breadth-first search can be performed in order to search for the best path to the query object.

Due to space limitations, we leave out the details of steps 1-3. An interested reader should refer to [12] and [16]. In the next section we will formalize how to compute the likelihood map (step 4) and in section IV we will describe our path-planning objective and breadth-first search algorithm (step 5).

III. Inferring object locations

In order to compute the object likelihood function over the map (i.e., the likelihood map), we propose to use the *object-object* and *object-scene* relationships inherent in the environment. For a location on the map skeleton, we want to compute the probability that a novel query object is visible given our object detections. Formally, given a location l on the map skeleton, we can compute a distribution over the existence $o_{s,l}$ of the novel object s at location l given the detections of objects $c_{i,l}$, that is,

$$p(o_{s,j}|c_{1,1},\dots c_{M,1},\dots c_{1,N}\dots c_{M,N}), \tag{1}$$

where i indexes the M detectable objects and l indexes the N locations on the skeleton.

A. Inferring object locations

In this section we will describe how to compute equation 1, which corresponds to step four in figure 1. For the purposes of this work, we make the simplifying assumption that other locations $l^{'}$ have no effect on whether object o_s is visible from the current location. By making this simplifying assumption, we have the following distribution, which we call the *Markov Random Field* (MRF) model:

$$p(o_{s,l}|c_{1,l}\dots c_{n,l}) = \sum_i p(o_{s,l}, o_{1,l}, \dots, o_{m,l}|c_{1,l}\dots c_{n,l}) \tag{2}$$

In order to define the distribution in the sum, we want to take into account noisy measurements and contextual relationships. Thus, we have:

$$p(o_{s,l}, o_{1,l}, \dots, o_{m,l}|c_{1,l}\dots c_{n,l}) = \frac{1}{Z}\prod_{i,j}\psi(o_{i,l}, o_{j,l})\prod_i \phi(o_{i,l}, c_{i,l}) \tag{3}$$

The local MRF can be seen graphically in figure 2. In the first term ψ, we have the likelihood of two objects co-occurring while in the second term ϕ, we have the likelihood of an object detector being correct. Note that there are two types of "objects": those that we can observe directly and those which are latent, e.g., scene variables. In the case of all our experiments, some subset of these variables will be observed based on the categories that we are able to obtain. However, there are nevertheless latent variables that are not observed, but which have an effect on the solution.

We have also explored a simpler model that does not take into account the fact that classifiers are noisy. Here, we assume that the classifiers are always correct, and we can therefore take them as observations of the objects themselves, in which case, equation 1 becomes:

$$p(o_{s,l}|c_{1,l}\dots c_{n,l}) = p(o_{s,l}|o_{1,l}\dots o_{n,l}) \tag{4}$$

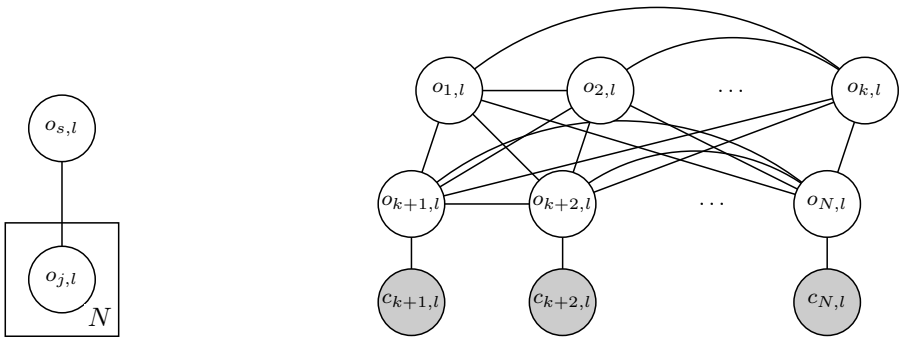
$$= \frac{1}{Z}\prod_j \psi(o_{s,l}, o_{j,l}), \tag{5}$$

which is a *Naive-Bayes* model. While the compatibility matrices currently only utilize two objects, in the future we plan to use models of higher order.

In the experiments presented in section V, loopy belief propagation was used in order to perform the inference. Thus, the next challenge is to determine the compatibility matrices $\psi(o_{i,l}, o_{s,l})$ for objects $o_{1,l}, \dots, o_{n,l}$ and the query $o_{i,l}$.

B. Learning the compatibility matrices

In order to learn the functions ψ and ϕ, we use co-occurrence statistics $n_{i,j}$, that is, the count for how often object o_i occurs with object o_j. Assuming that we have these statistics, we learn these functions as:



(a) Naive Bayes model (b) Local Markov Random Field

Fig. 2. The two models we have used in our experiments.

$$\psi(o_s = T, o_i = T) = \frac{n_{s,i}}{\sum_j n_{s,j}} \quad (6)$$

$$\psi(o_s = F, o_i = T) = \frac{\sum_{m \neq s} n_{n,i}}{\sum_{m \neq s} n_m} \quad (7)$$

$$\psi(o_s = T, o_i = F) = 1 - p(o_s = \mathrm{T}, o_i = \mathrm{T}) \quad (8)$$

$$\psi(o_s = F, o_i = F) = 1 - p(o_s = \mathrm{F}, o_i = \mathrm{T}) \quad (9)$$

In the first term we have in the numerator the frequency for how often object s and object i co-occurred together and in the denominator is the sum of the counts in category s. In the second term the numerator has the sum over all the times that object o_i has been seen in any other category divided by the sum of the elements in all the other categories.

C. Large co-occurrence databases

In order to determine the co-occurrence counts n_{ij}, we require a large database of information about which objects tend to be spatially co-located, and which objects tend to occur in which scenes. The specific database we use is Flickr, which has image data as well as tags that have been given to the images by millions of users. Our insight in using a photo database is that the captions generally describe objects in the image, and objects in the same image are in the same location. While there is some amount of bias in the dataset (or irrelevant tags), the words that people use to describe their images often actually do correspond to the object classes present in the image (e.g. *computer, desk, keyboard... etc.*). We can see the co-occurrence counts in figure 3 for the *desk* and the *mac* classes. On the vertical axis are the top 20 object classes that co-occur with the base category and on the horizontal axis are the frequency with which they co-occur. Near the top of the list for *desk* are *computer, keyboard, mouse, printer, lamp*, all things that humans would expect to find with a *desk*. In addition, we are not limited to a strict vocabulary. This can be seen by looking at *mac* in figure 3(b). *Mac* refers to a Macintosh computer, and as expected we find that it co-occurs with desks, computers, chairs, printers, etc.

Instead of hard-coding the set of objects that we can query, we perform a dense sampling of images over all the locations that exist in the English language (e.g. hallway, office... etc.). In other words, we use all of the *objects*, *scenes* and *animals* defined in the WordNet database and search for images on the Flickr photosharing site. For the top 1000 hits, we download these images and use the associated tags to derive co-occurrence counts from these images.

IV. Planning to find objects

We want to be able to compute a path through the environment that minimizes the expected travel distance to the object (step 5 of figure 1). Here we again leverage the medial-axis transform in order to reduce the search space. Using breadth-first search, we expand locations on the medial axis to the immediate connected neighbors (of which there are at most 4). The goal is to find an object as soon as possible, which means that we want to minimize the expected length of the path to the object $E[L_p]$. The expectation is taken with respect to the distribution over objects,

$$\operatorname*{argmin}_{p \in \text{paths}} E[L_p] = \quad (10)$$
$$\sum_{l=1}^{M} p(o_{s,l}{=}\mathrm{T}, o_{s,l-1}{=}\mathrm{F} \ldots o_{s,1}{=}\mathrm{F} | c_{1,1} \ldots c_{M,N}) \times l.$$

Here, $p(o_{s,l} = \mathrm{T}, o_{s,l-1} = \mathrm{F} \ldots o_{s,1} = F | c_{1,1} \ldots c_{M,1} \ldots c_{M,N})$ is the likelihood of finding object s along the path up to the current location given the likelihood of finding object (or not) along the way and the classifications derived from the map.

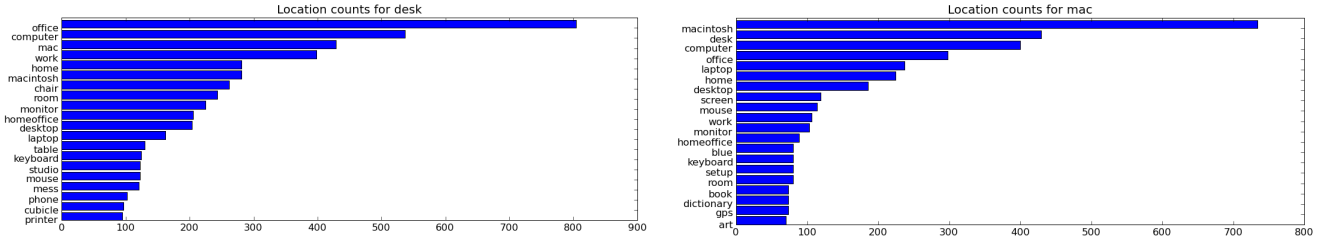
Given a start location in the map, we retract the start location onto the medial axis and perform a breadth-first search from there. Thus, at each node, we will expand the node as follows, recursively computing the expected length of the path:

$$E[L_n] = E[L_{n-1}] + \left[\prod_{l=1}^{n-1} p(o_{s,l} = F | c_{1,l} \ldots c_{M,l}) \right] \times p(o_{s,n} = T | c_{1,l} \ldots c_{M,l}) \times n \quad (11)$$

It is straightforward to keep track of the expected length of the path as well as the probability that the robot did not see the object at any previous location on the path. In addition, we can compute the likelihood of having found the object after n timesteps as:

$$p(o_{s,n} = T \text{ or } o_{s,n-1} = T \ldots \text{ or } o_{s,1} = T) \quad (12)$$
$$= \sum_{l=1}^{k} p(o_{s,l} = T, o_{s,l-1} = F \ldots o_{s,1} = F | c_{1,1} \ldots c_{M,N})$$

We perform a breadth-first search out to a specified horizon. We then determine which paths have a likelihood of finding the object greater than threshold t (equation 12), and we sort these paths by their expected length (equation 11),



(a) desk (b) mac

Fig. 3. We can see on the *x*-axis above are the raw counts for the number of times *desk* or *mac* appeared with the categories on the *y*-axis in the Flickr dataset. In (a) we can see that a *desk* is often in an office, with a chair, a laptop, a lamp, etc. In (b) we can see the flexibility of our approach. Here we are able to query *mac* (a computer made by Apple), which are often found near desks, screens, keyboards, etc., as one might expect in natural environments.

picking the minimal expected length. For the purposes of our search, we allow backtracking only when the robot has explored to an end point of the medial axis. Also, the path is only allowed to traverse a particular location twice, and all paths are constrained to have equal length. After computing the path, the robot will then execute this path and search out the object, utilizing either classifiers to find the object once it is in high-likelihood areas or using a human to close the loop.

V. Results

In this section, we show that our model robustly predicts the location of novel objects. In addition, we show that the proposed solution to the planning problem results in shorter expected paths than a greedy strategy.

A. The effect of context

Let us first assume that we have a perfect object detector: when an object is in view it will be seen. We will perform a simulated experiment where we take a map of the third floor of a building at MIT and by walking around the environment label all the locations of a limited number of objects in this environment (for a partial list of the objects see figure 1). We remove one object type from this list and let the other objects predict the location of this query object. In this particular environment, we labeled approximately one hundred object types. Based on the labels already present in the Flickr dataset, we can query any of the 25,000 object and scene types in the English language.

In figure 4, we can see queries for a number of semantic categories. Of particular note is that we can query many types of objects (even ones that are not labeled in the map). For example, in figure 4 we can see the *mac* object. Also, we can see that there is almost zero likelihood of finding a *cow* in the environment, as expected. However, *zebra* shows a slight likelihood of appearing in the lounge because a zebra couch is apparently a somewhat popular type of striped couch.

We can also use our approach to compute the most likely scene (e.g. office, hallway, conference room... etc.). Referring to figure 5, we find the most likely place to find a kitchen is near the refrigerators, toaster ovens, and espresso machine. There are three kitchen areas and in each of these, we can see that the *soap* appears next to the sink and refrigerator. The most likely places to find *offices* are away from any hallways and near desks, computers, and monitors. Of particular note is that the monitor, desk, computer, and all occur at similar areas, indicating that they tend to occur together in office environments. Exits are most likely near the stairs and in hallway areas and there is likelihood for the existence of a lounge near the television. Overall, the places with high likelihood match our intuitions.

In order to quantitatively evaluate the performance of the object inference, we divided the likelihood maps from figures 4 and 5 into 21 topological regions. Treating the likelihood map as a classifier, if the probability of seeing an object anywhere in this region was over a threshold t, then we classified this region as having the object in it, otherwise we classified it as not having the object present.

With $t = 0.7$, we have a precision of 82% of the objects in figure 4, with a recall of 93% . With $t = 0.99$, then a precision of 95% was attained, with a recall of 87%. On a per-class basis, most objects were predicted well, with the exception of the bottle class, which we believe to be because bottles can appear many places, leading to a moderate likelihood over the entire environment.

B. Application to real-world data

In addition to the simulated experiment, we evaluated our techniques on another floor of a building at MIT using a real object detector from [3]. We detected three objects as the robot moved around the floor: *chairs*, *bicycles*, and *monitors*. Some examples of the classifier output are shown in figure 6. We added the object detections to the map, according to where the robot was located as shown in figure 7(a). Out of a trajectory of approximately 5000 images, there were 13 false positives, and 64 true positives. The chair detector incorrectly detected 13 chairs, while the bicycle detector missed no bicycles (there were two in the environment) and the monitor detector falsely detected no monitors (there were two detections).

Based on these three detectors alone, we were able to predict the location of a number of objects. The predicted location was qualitatively reasonable for a number of categories. In figure 7(a) we can see the locations where a monitor is known to be visible based on the object detections and in figure 7(b) we can see the resulting search path for a novel object, specifically a computer; the path goes past locations where the monitor is likely to be. Thus, we have demonstrated the feasibility of applying our approach to real-world data and plan to include more comprehensive results in the future.

C. Path optimization results

Finally, we did a study in order to optimize the path from a random location to a number of query objects (e.g. the ones from figure 4). To perform this experiment, we used

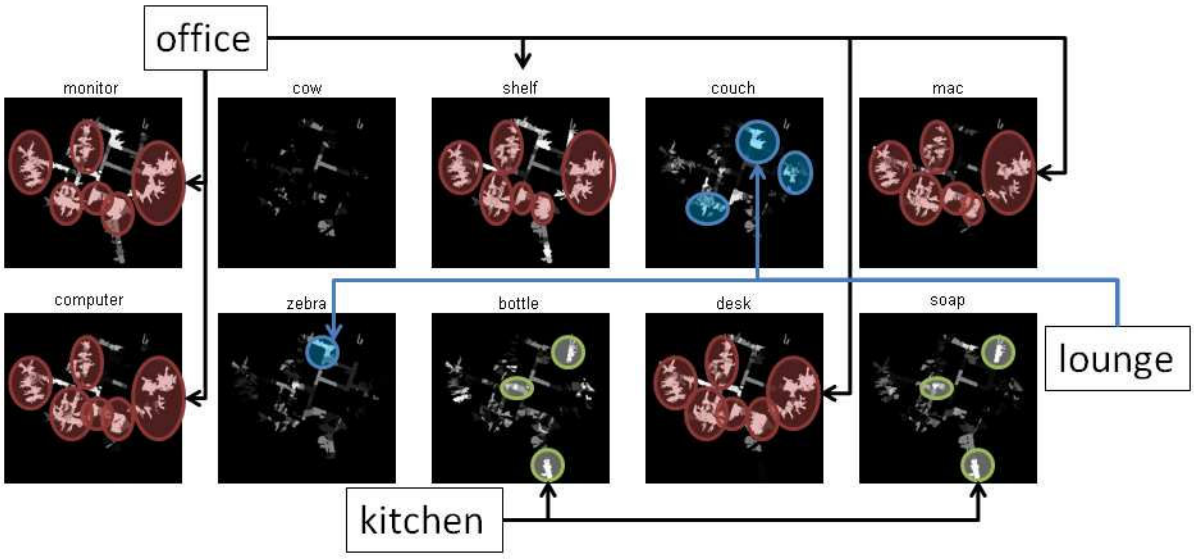


Fig. 4. Above are the likelihoods of finding objects over the entire environment. White is higher likelihood, darker is lower likelihood and these are computed according to equation 4. In addition, we have labeled some salient scene types by highlighting them.

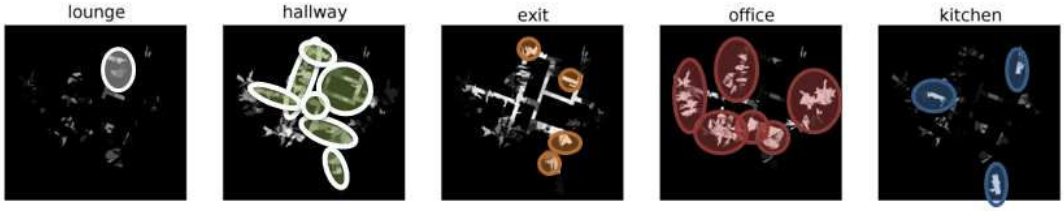


Fig. 5. Above are the likelihoods for various locations over the entire environment. White is higher likelihood, darker is lower likelihood and these are computed according to equation 4. Note that the circled areas correspond to the actual location of each scene in the environment.

a baseline approach where the path was generated to the nearest location with a probability over the threshold t of having the object visible (this is the *greedy* approach). We compared this solution to the paths generated according to our objective function. In order to normalize for length, we extended the greedy path through a series of locations selected according to the greedy strategy.

In our experiments, we computed 30 random start locations in the map. Over these locations, we found that our approach had a shorter expected length to the object from 13% (for desks) to 68% (for refrigerators) of the time. In the rest of the cases the *greedy* approach and our approach had equal objective values due to the fact that the greedy approach would go through the same location as our approach.

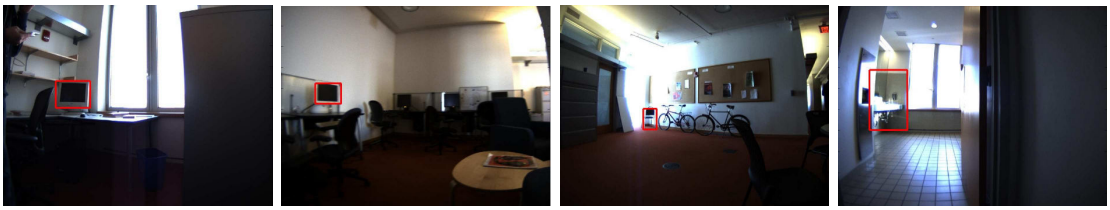
VI. Related Work

There has been considerable interest in utilizing the structure of the environment when interacting with humans. By characterizing space as a hierarchy of elements [4], [2], [15] are able to capture the relationships of scenes and objects and communicate with humans. In contrast to our approach, each of these works utilizes ontologies that have been created by hand and are deterministic. In [4], the authors additionally propose a means by which to learn these semantic representations from sensor data.

In terms of communicating about tasks semantic level, [11], [6] use local commands and extract spatial relationships from maps. In [6], the authors describe a robotic wheelchair that can follow directions over an extended period but do not perform a systematic evaluation of their assertions. In [9], the authors describe directing a semi-autonomous wheelchair, where commands take the form of "enter right door" and "follow corridor." There has also been work on utilizing an object-based representation of the environment, although the extent to which the authors have applied this to real-world problems is unclear [14]. Our work, in contrast to these approaches utilizes the notion of *object-object* and *object-scene* context in order to reason about the environment.

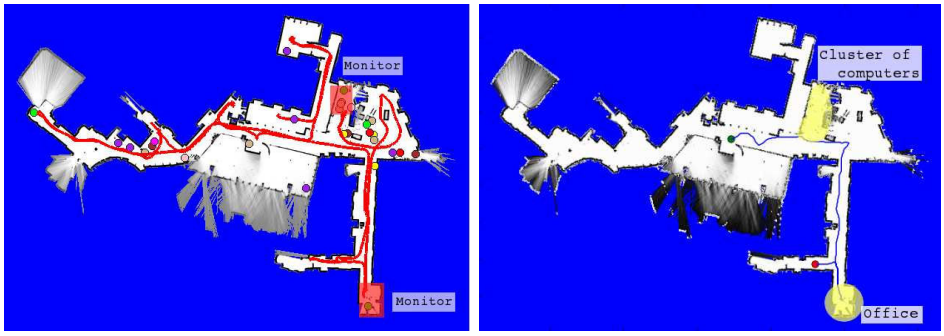
From the scene understanding community context is used in order to perform object detection or localization [13], [5], [10], [7]. Probably the most related to our work is [13], where the authors use a hidden Markov model to estimate the scene type (e.g. hallway or office) and then use this as a prior for improving object detection. In contrast, we include object and scene types in the same framework, and are able to use the geometric structure of the environment. In [7], the authors utilize viewpoint and image geometry in order improve classification accuracy, which is complementary to our approach and a cue that we believe could be useful.

Finally, there is the Semantic Robot Vision Challenge (SRVC), where competitors use keywords to download images from the Internet, train a model of an object and find it in a competition arena using a robot. Most approaches to this competition use active vision to search locally for the objects and map candidate locations. Our approach is



(a) True positive (Monitor) (b) True positive (Monitor) (c) True positive (Chair) (d) False positive (Chair)

Fig. 6. Using the approach in [3], we classify a number of categories in the an office building on MIT. Above are some of the images classified correctly and one instance of a false positive. In (a/b) are the two locations where a monitor was detected in the environment. In (c) is a true positive of a chair while in (d) is a false positive of a chair.



(a) Location of objects registered to the map (b) Planned path for a *computer*

Fig. 7. In (a) we can see the location of the detected objects registered to the map. The two locations where monitors were detected are highlighted in red. In (b) we can see the path to follow in order to find a computer. The path passes a cluster of computers and goes to an office where a computer resides in the real environment, passing by a number of the query objects in the process. Note that the green circle is the start location and the red circle is the destination.

different from these in that we are looking at the structure of the environment in order to predict the location of objects, while the SRVC is using a local search in order to find a query object.

VII. Conclusions and Future Work

In conclusion, we have developed a model that accurately predicts novel objects in the scene based on context. In order to improve this model's accuracy and robustness, we plan to incorporate smoothing terms that will allow the inferences to be propagated across space as well. In addition, we also plan to incorporate information other than co-occurrence information, such as object size, height above the ground, disparity in depth, and others in order to improve our inference.

We have also demonstrated that given a limited amount of prior information, we can compute the best path to find this novel object. One future direction we would like to explore is viewpoint planning so that when the robot arrives at a location likely to contain a novel object, we might use weaker object detectors in order to search for the novel object by planning its viewpoint.

References

[1] Irving Biederman. Recognition-by-components: a theory of human image understanding. *Psychological Review*, 94(2):115–147, April 1987.

[2] Krieg B. Bruckner, U. Frese, K. Luttich, C. Mandel, T. Mossakowski, and R. Ross. Specification of an Ontology for Route Graphs. *Spatial Cognition IV: Reasoning, Action, Interaction. International Conference Spatial Cognition*, pages 390–412, 2004.

[3] P. Felzenszwalb, D. Mcallester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) Anchorage, Alaska, June 2008.*, June 2008.

[4] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, Fernandez J. A. Madrigal, and J. Gonzalez. Multi-hierarchical semantic maps for mobile robotics. *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 2278–2283, 2005.

[5] Y. Gao and J. Fan. Incorporating concept ontology to enable probabilistic concept reasoning for multi-level image annotation. *Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 79–88, 2006.

[6] W. S. Gribble, R. L. Browning, M. Hewett, E. Remolina, and B. J. Kuipers. Integrating Vision and Spatial Reasoning for Assistive Navigation. *LECTURE NOTES IN COMPUTER SCIENCE*, pages 179–193, 1998.

[7] D. Hoiem, A. A. Efros, and M. Hebert. Putting objects in perspective. *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2006.

[8] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to WordNet: An On-line Lexical Database*. *International Journal of Lexicography*, 3(4):235–244, 1990.

[9] R. Muller, T. Rofer, A. Lankenau, A. Musto, K. Stein, and A. Eisenkolb. Coarse Qualitative Descriptions in Robot Navigation.

[10] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in Context. *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8, 2007.

[11] M. Skubic, D. Perzanowski, S. Blisard, A. Schultz, W. Adams, M. Bugajska, and D. Brock. Spatial language for human-robot dialogs. *Systems, Man and Cybernetics, Part C, IEEE Transactions on*, 34(2):154–167, 2004.

[12] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, Cambridge, MA, 2005.

[13] A. Torralba, K. P. Murphy, W. T. Freeman, and M. A. Rubin. Context-based vision system for place and object recognition. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 273–280, 2003.

[14] Shrihari Vasudevan, Stefan Gachter, Viet Nguyen, and Roland Siegwart. Cognitive maps for mobile robots – an object based approach. *Robotics and Autonomous Systems*, 55:359–371, 2007.

[15] H. Zender, Martinez O. Mozos, P. Jensfelt, G. J. M. Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56:493–502, 2008.

[16] T. Y. Zhang and C. Y. Suen. A fast parallel algorithm for thinning digital patterns. *Communications of the ACM*, 27(3):236–239, 1984.