Physical Querying with Multi-Modal Sensing

Iljoo Baek[1], Taylor Stine[1], Denver Dash[1,2], Fanyi Xiao[1], Yaser Sheikh[1], Yair Movshovitz-Attias[1], Mei Chen[2], Martial Hebert[1], and Takeo Kanade[1]

[1]The Robotics Institute , Carnegie Mellon University, Pittsburgh, PA, USA
[2]Intel Science and Technology Center on Embedded Computing , Carnegie Mellon University, Pittsburgh, PA, USA

Abstract

We present Marvin, a system that can search physical objects using a mobile or wearable device. It integrates HOG-based object recognition, SURF-based localization information, automatic speech recognition, and user feedback information with a probabilistic model to recognize the "object of interest" at high accuracy and at interactive speeds. Once the object of interest is recognized, the information that the user is querying, e.g. reviews, options, etc., is displayed on the user's mobile or wearable device. We tested this prototype in a real-world retail store during business hours, with varied degree of background noise and clutter. We show that this multi-modal approach achieves superior recognition accuracy compared to using a vision system alone, especially in cluttered scenes where a vision system would be unable to distinguish which object is of interest to the user without additional input. It is computationally able to scale to large numbers of objects by focusing compute-intensive resources on the objects most likely to be of interest, inferred from user speech and implicit localization information. We present the system architecture, the probabilistic model that integrates the multi-modal information, and empirical results showing the benefits of multi-modal integration.

1. Introduction

The Internet has revolutionized the availability of information, and mobile technology has revolutionized access to it. Yet, the current model of query and response makes little use of knowledge about the user or the current context. For instance, if someone at a store wants to look up reviews of a shampoo he is looking at, he would need to interrupt his shopping, locate his smartphone, and enter a query. The phone remains entirely unaware of the shampoo the user is looking at and what the user is saying. In contrast, if the user had access to a sales representative at that moment, the representative would be fully in tune with the user's context and could immediately answer his query. This paper presents Marvin, a system that allows users to seamlessly access information about their physical environment by fusing contextual knowledge, giving them an experience much more similar to that of posing queries to a human on the spot.

Marvin combines multiple sensing modalities, multiple high-level sources of information, and collaborative engagement with the user into a personal assistant that understands spoken commands, observes the physical environment, perceives objects of interest, and provides useful information about those entities in real-time. We leverage recent, complementary, advances in computer perception, such as SURF-based feature point matching and HOG-based object recognition [4, 6, 11], as well as established software tools for automatic speech recognition [10], Bayesian reasoning and systems-building [13].

The idea of creating a mobile assistant that uses computer vision to assist the user for various tasks has been proposed before. For example, Google Goggles [7], oMoby [5], and Amazon [1] all provide services where information about objects can be retrieved from the cloud by submitting an image. These services do not make use of speech when performing a query. Our system's addition of speech allows the user to provide important cues which can greatly improve perception accuracy, even when the images are blurry or cluttered. Our method also makes use of complementary visual information, for example a SURF-based component quickly provides information for heavily textured objects, but a more computationally intensive HOG-based retrieval algorithm adds generality and robustness to our system. While the existing methods are proprietary, these services are self-admittedly not well-suited to a range of objects that appear to include mostly non-textured objects. These meth-

ods all rely on cloud-based computations and in some cases can require minutes to return an answer, whereas we focus our use-case on algorithms that can be performed at interactive speeds. OrCam orcam is another proprietary system that accepts both vision and gesture input to help visually-disabled people with basic tasks such as reading, checking buses and recognizing faces. OrCam is also proprietary so it is impossible to compare our approach to theirs, but based on promotional videos it seems that most retail object assistance makes use of optical character recognition (OCR) as opposed to object recognition, so is possibly limited to objects which are explicitely labeled with text.

The idea of combining multiple sources of complementary information is not new: for example, Gupta and Davis [8] combined action recognizers, object recognizers and human pose estimators with a Bayesian network (BN) model to perform more accurate human activity recognition. Barnard et al., [2] used annotated images and built joint models of word/image regions for automated image annotation. Gupta and Davis [9] took this idea further and considered higher order relationships between words and images regions to improve visual classifiers. Cour et al., [3] combined closed-captions of video to perform more accurate video segmentation and action recognition. All of this previous work has combined relatively few sources of information and has focused on offline batch tasks such as image annotation. Our approach contrasts with these in that we present a use-case where the user is present and active during the perception calculation, and we present a system that exploits multiple modalities (speech, SURF and HOG-based vision algorithms). These two facts together enable a user to collaborate in real-time with the perception itself.

The main contributions of our system include:

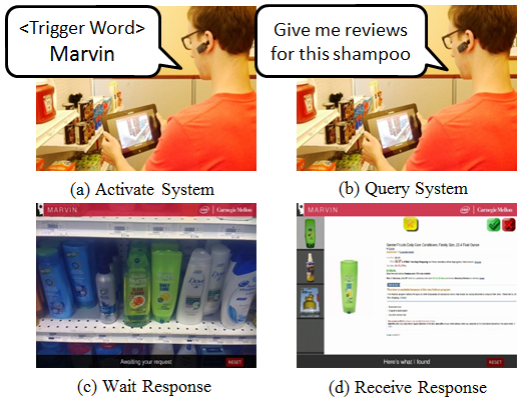
1. A multi-modal, interactive approach to query and response that allows users to seamlessly access information about their physical environment by fusing contextual knowledge with collaborative user-supplied cues.

2. An architecture for Bayesian fusion of multi-modal sensory data using a probabilistic model.

3. A system prototype performing this integration at interactive speeds. Issues of system architecture, scalability, and maintenance are explored.

2. The Marvin System

The usage scenario for the system is depicted in Figure 1: A user holding a mobile or wearable device equipped with a camera and microphone explores the environment by pointing the mobile device toward objects and verbally querying the system about objects of interest. For example, the user may ask: "Can you give me reviews for the shampoo in the green bottle?" The system interprets the most likely user query given all sensor information, and sends the requested information to the user's display.



(a) Activate System (b) Query System

(c) Wait Response (d) Receive Response

Figure 1. The use case: A person holding his mobile device toward the object of interest (a) activates the system via a trigger word, (b) makes a verbal query about the object, (c) waits for a response, and (d) receives the response on his display.

For a system such as this to operate correctly, it must be able to relate visible entities to semantic instances, i.e., it must perform robust object recognition, which is still an open problem in computer vision. However, because this system is used by a human, there are other sources of information that can inform the vision subsystem of the intent of the user. For example:

Trigger Words. To avoid false positives, the user can issue a trigger keyword to inform the system that he/she is about to pose a query (e.g. "Marvin").

View Direction. The pose of the user's camera is often strongly correlated with visual features which belong to objects in the frame or to the background.

Verbal Context. Commands issued by the user provide implicit or explicit cues for the objects of interest. For example the user may describe an object's attributes such as "yellow bottle", or he may even provide specific information such as "Listerine wipes in the yellow bottle" which can provide powerful context to assist the object recognition.

In Section 2.1, we describe in detail the architecture of the system.

2.1. System Architecture

Figure 2 shows the high-level architecture of the Marvin system (shown after the triggering phase). Marvin accepts sensory inputs from both audio and visual modalities. The audio input is assumed to be a speech stream directed

by the user to Marvin. The visual input is assumed to be a frame depicting an object of interest, possibly including many other objects which could be treated both as informative context and as confounding objects which may decrease or increase ambiguity, respectively.

This architecture is motivated by the informativeness and the timing characteristics of each subsystem. Figure 3 shows a typical breakdown of the system timing. The object recognition component (discussed in detail in Section 2.2), while informative, is the most computationally demanding. To apply our object recognition algorithm in brute-force fashion searching for 25 objects typically takes around 7 seconds using a Graphics Processing Unit (GPU) implementation. While this timing may be marginally acceptable for a real-time system, it quickly fails to scale to a larger number of objects (see Figure 10).

Marvin compensates for this bottleneck by integrating speech and "weak" location information (discussed in Section 2.3) into a BN model (discussed in Section 2.5), then using the posterior probability of objects of interest to inform the object recognition subsystem about which objects to look for. The object recognition component can then devote more or fewer resources to finding objects consistent with those preliminary findings. In cases where speech

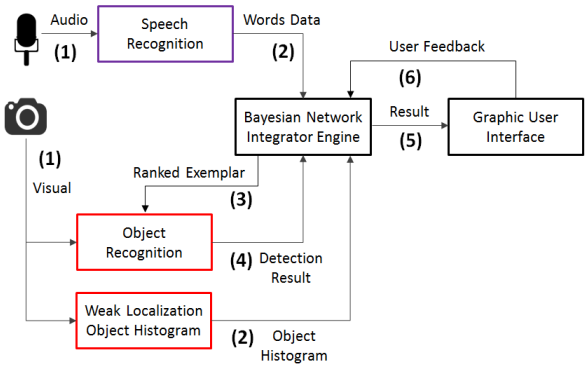


Figure 2. Marvin System Architecture: The purple box refers to audio-based component and the red boxes refer to vision-based components.

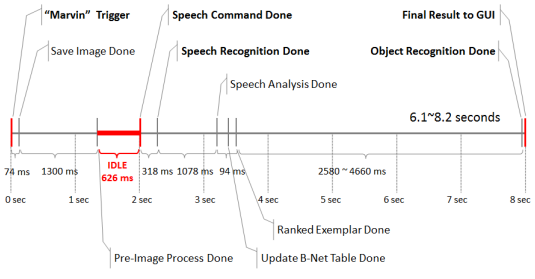


Figure 3. Marvin Timing Profile: shows a typical breakdown of the system timing from triggering the system to receiving results.

Figure 4. Training Data for Objects: On average 85 view-models were trained for each object.

and weak location information lead to high confidence in a particular object, less effort is spent on object recognition; whereas in cases where relatively little information is present, more effort is spent on object recogntion.

After deciding the amount of effort to devote to object recognition, the system integrates object recognition results into the BN which combines those results with the speech and weak localization results again and passes the most likely $\langle command, object \rangle$ pair to the user interface (UI) to be displayed to the user. If there is sufficient uncertainty even after all information has been integrated, the BN reports out a ranked list of likely commands and objects which can be all displayed to the user.

The user has the ability to provide feedback to the system based on the reported results. If the correct object is present in the ranked list, the user can select it and tell Marvin that this object is correct; otherwise, the user can tell Marvin that the results were incorrect. This feedback can then be used for online learning to refine the BN parameters.

2.2. Object Recognition Subsystem

As the object recognizer, we chose to use the Exemplar SVM (eSVM) algorithm and code of [11].

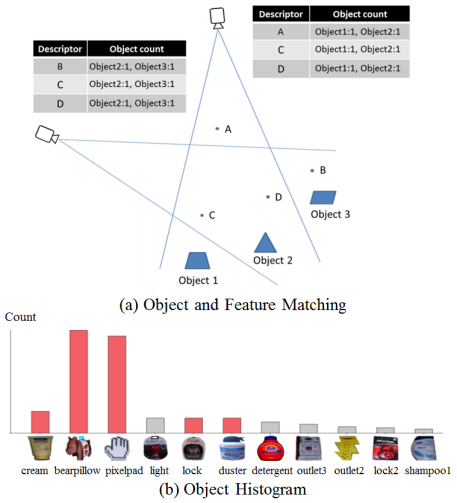
The eSVM algorithm handles view-variation in objects explicitly by creating a separate model for each of many views ("exemplars") of an object. The method learns a large-margin decision boundary between a single exemplar and a large collection of negative samples. At run time, the eSVM classifier performs a sliding window search to find the maximum response of a HOG descriptor trained on a particular view of the target object.

Marvin was trained on 25 retail objects, shown in Figure 9. 16 objects were trained with 72 views and 9 objects were trained with 108 views. See Figure 4 for some example views for two of our objcts.

We use a Matlab version of the real-time recognition portion of the code used by [11], and have reimplemented portions of that code to employ a GPU for matrix computations.

2.3. Weak Localization Subsystem

The purpose of the localization subsystem is to use SURF features in the frame to provide implicit information



(a) Object and Feature Matching

(b) Object Histogram

Figure 5. Weak localization: The counts in the object histogram represent how many features in a test image co-occur with the particular object.

about where the user is looking, and thus to narrow down the set of objects to consider. Because we do not attempt to establish a precise location or pose of the camera, we call this modality "weak localization" because it is using the fact that the user's presence in his location is impacting our beliefs about the possible set of objects that might be present.

This capability allows us to improve system accuracy by exploiting a complementary source of visual information. It also provides a method to scale up object recognition by narrowing down the set of objects that are likely to be in the scene given the background and foreground features present in a test frame, thus providing a source of information that can be used to restrict the more computationally intensive eSVM object recognition to a smaller set of objects.

We accomplish weak localization by first constructing a corpus of images from a physical space which is expected to remain approximately static (such as a retail store). Second we annotate each image in the corpus with a frame-level list of objects present in that image. Third, we extract and cluster all SURF features from interest points in the corpus to form a master list of features, and fourth, we associate each feature with the set of objects which co-occurred with it in the corpus. For example in Figure 5-(a), two images are taken from different view points and have some overlapping objects. An image from the top view has objects 1 and 2, and features A, C and D. An image from the side view has objects 2 and 3, and features B and D. In each image in the corpus, each extracted feature is associated with the objects in that image. Therefore, feature C is associated with objects 1 and 2 in the top view and with objects 2 and 3 in the side view. This relationship between features and objects in the corpus are concatenated into one table. Once a test image is given, extracted features from the image will be matched to corpus features, and the $\langle$`object:count`$\rangle$ pairs associated with each matching feature are added to build an object histogram as shown in Figure 5-(b). A smoothed version of this Location-based object histogram can now be used directly as a prior probability distribuion over objects.

We employ feature extraction and matching code from [12] which has real-time capability to perform an offline 3D reconstruction of the SURF point cloud of a physical environment using GPU acceleration.

2.4. Speech Recognition Subsystem

A speech decoder $\mathcal{S}(A; D, L)$ takes as input a dictionary of words D, a statistical language model L specifying how words are likely to be ordered in sentences, and an audio stream A. It outputs a recovered sentence S which is an ordered tuple of words from D. This sentence must then be mapped onto a set of semantic concepts (for example the object of interest) We use the speech decoder PocketSphinx [10] in our implementation, and we incorporate the mapping functionality into the global BN model which considers other bits of information in addition to the decoded speech. The specific BN we use will be discussed in more detail in Section 2.5.

Prior to directing a query to Marvin, we require a speech-based System Trigger. The System Trigger is a special speech-based classifier that is used to prompt all the Context Modules into action. As such, we need to balance the desire to have a responsive system with the importance of minimizing false positives for this particular classifier. We accomplish this using the following scheme: First, we pick a distinctive trigger word to avoid false positives. Second, we use an *open* language model with a small (124 words) dictionary where there are essentially no constraints in how words appear to match random speech. For other non-trigger speech, we use a language model trained on over 36000 natural english-language sentences involving querying information about retail objects.

2.5. Bayesian Network Integrator

The integration engine collects the outputs of all sources of information and makes a global decision about the most likely $\langle$`command, object`$\rangle$ pair. We use a BN to accomplish this task, the structure of which is shown in plate notation in Figure 6. In this figure, pink nodes indicate latent context variables (the command and object desired by the user) which we would like to infer from the collection of modalities. Gray nodes are latent variables whose values we do not need to infer directly, and green nodes are variables which we may observe directly.

There are a total of three "plates" in this model. For speech modeling, the plates correspond to keywords (N_w

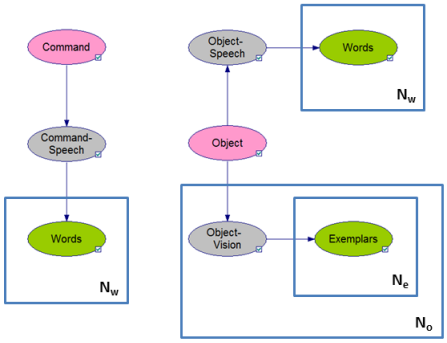


Figure 6. The BN model used for inference in Marvin.

of these) which are informative for each object known by Marvin. For object recognition modeling, the plates represent objects visible in the frame (N_o of these) as well as exemplars trained for each object (N_e of these per object).

In the retail task in which Marvin currently specializes, we assume that the command and the object are independent of one another. If this model were to be expanded to a more general use case with non-retail objects, then this model could be changed to reflect the fact that different objects may be associated with different commands by drawing an arc from *Object* to *Command*.

The BN incorporates three bits of context into its calculation: speech, weak location and object recognition. Text from the speech decoder is instantiated in the model whenever a word that corresponds to a keyword which was preassigned to one or more objects or commands is decoded. The object-histogram generated by our weak localization method enters directly as priors on objects (after applying Laplacian smoothing by adding 1 to all histogram entries), and the object-recognition results enter into the model as nodes representing exemplars which are in the states "present", "absent" if they were checked by the eSVM algorithm or they remain unobserved otherwise.

In addition to combining these inputs into a single result, the BN also serves the purpose of assisting Marvin in resource allocation. This is done by first incorporating the less informative but fast sensors (speech and weak localization) into the model, and then querying the BN about which eSVM exemplars are most likely to be of interest. Depending on how confident the BN is given speech and weak localization, it can choose more or less exemplars to be explored, and it can choose which exemplars are the most likely given historical firing data. This capability of using "cheap" sensors to actively query the more expensive ones, turns out to be critical to enable Marvin to scale to large numbers of objects, as we show in Figure 10.

The BN selects the list of ranked exemplars to evaluate by first calculating the posterior distribution over objects given the speech and possibly localization information. It considers three cases: *clear speech*, *unclear speech* and *partial speech*. The clear speech case is used when the most likely object exceeds the second most likely object by some factor f. The partial Speech case occurs if one or more object probabilities cross some threshold θ but do not satisfy the clear speech criterion. The unclear speech case is the default. After the clarity of speech is determined, we send N_c, N_p or N_u ranked exemplars in the clear, partial and unclear speech cases respectively, where $N_c < N_p < N_u$. In all cases, the exemplars are chosen by multiplying the total number of exemplars by the posterior of each object. If that object does not have sufficient exemplars, we try to maintain our fixed budget by passing the residual number down to less likely objects. Still in some case, when the posterior distribution over some objects is higly peaked, we may end up with a number of exemplars that are less than the target number. This fact can allow us to reduce computation time even further when we are especially confident about certain objects.

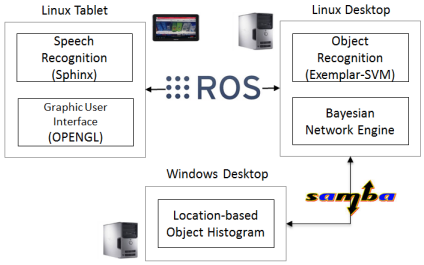


Figure 7. Hardware System Components.

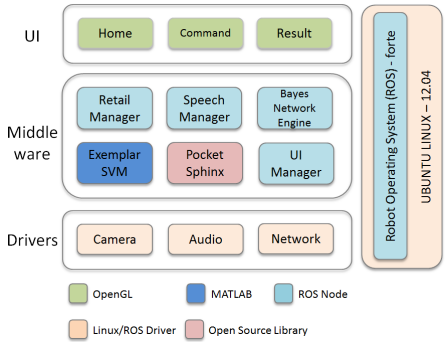


Figure 8. Software Components.

3. Testing and Performance Evaluation

The hardware and software components described in Figure 7 and Figure 8 are used as a prototype of the Marvin system. Marvin was trained to recognize 25 objects retrieved from a university bookstore, as shown in Figure 9.

Based on their physical and semantic properties, these objects were assigned the set of keywords shown in Table 1, and we trained in total 2124 exemplar models for the 25 objects (16 had 72 exemplars and 9 had 108 exemplars).

We systematically explored performance for the clear,

bearpillow : bear, brown, pillow, travel
buglight : bug, light, key, chain, keychain, ladybug
cleanser : cleaner, blue, white, bottle, expo
cream : cream, vaseline, yellow, tube, stuff
clock : clock, digital, advance
dentalfloss : dental, floss, blue, monster
deodorant : speed, stick, deodorant, mennan, green, silver
detergent : tide, red, bottle, blue, detergent
disinfectwipes : disinfecting, disinfectant, wipe, wipes, yellow, tube, bottle, lysol
duckclock : yellow, duck, clock, timer
duster : maxell, blast, away, duster, compressed, air
kitchentimer : kitchen, timer, ladybug, beetle, lady, bug
lock1 : headphone, cord, cords, adapter, maxell
lock2 : red, lock, master, combination
minifan : mini, fan, airplane, usb
outlet2 : outlet, lightning, bolt, yellow, power
outlet3 : outlet, power, center, white
paperpunch : paper, punch, hole
pencilsharpener : pink, pencil, sharpener, titanium, westcott
peppercontainer : salt, shakers, black, white, face, pepper
pixelpad : hand, pad, pixel, tablet
shampoo1 : shampoo, white, blue, bottle, head, shoulders
shampoo2 : fructis, shampoo, green, bottle
sharpener : pepper, grinder, squirrel
stapler : swingline, compact, stapler

Table 1. The seed keywords associated with each retail object. The words on the left are unique IDs used to identify each object. Note that many keywords overlap multiple objects.

unclear and partial speech scenarios mentioned in Section 2.5 by constructing speech queries by hand which activated each scenario. The clear speech scenario includes enough unique keywords of an object for the system to recognize the object by speech alone. For example, the command "give me reviews about this white head and shoulders

Figure 9. Marvin is trained to recognize 25 retail objects, obtained from a university bookstore.

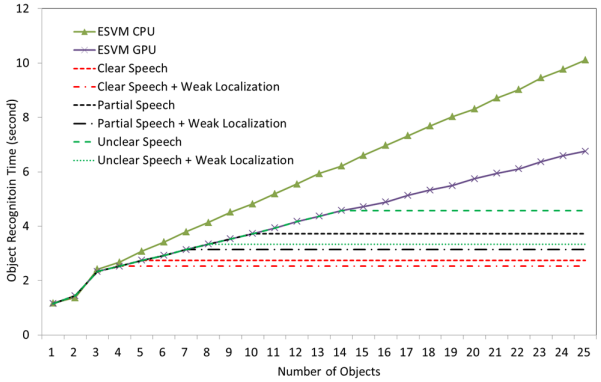


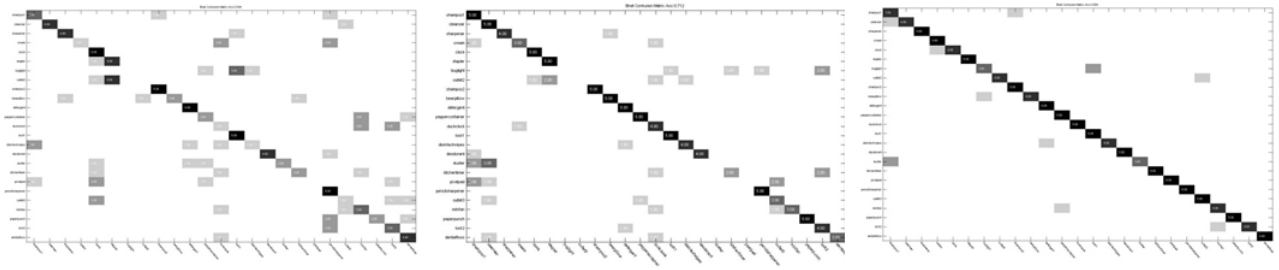
Figure 10. Object Recognition Time vs. Number of Objects: shows the average time spent in the object recognition can remain constant over the large number of objects.

shampoo" triggers a clear speech case for object *shampoo1* because it matches four of the pre-assigned keywords for this object from Table 1. The partial speech case includes fewer keywords which are not unique, for example, "give me reviews about this white bottle" could be referring to several objects in our set. For the unclear case, we use the phrase "What is this?", which has no information to identify an object by speech. For the clear speech case we used $N_c = 400$ exemplars. For the partial speech case, we use $N_p = 800$ exemplars and for the unclear speech case we used $N_u = 1200$. These numbers were chosen based on informal user studies to constrain the response time of the system to "interactive" speeds.

For repeatability, in the quantitative experiments, we used text speech commands for each object during the tests. To evaluate performance in response to cluttered background, each object was tested using two different scenarios: object-in-hand, where the tester holds the object in hand causing the object to dominate the frame and reduce clutter; and object-on-shelf, where the object is on a shelf with other objects nearby. In each scenario, we tested five times per object to have varied background views. In summary, 60 tests were conducted per object: 3 speech cases x 2 scenarios (in-hand, on-shelf) x 5 background views with/without the weak localization.

3.1. Scalability Studies

In Figure 10, we plot the average time spent on object recognition as the number of objects increases. The results from the eSVM algorithm using CPU and GPU were tested with the assumption that the number of exemplar models per object is 85. The vision-only object recognition algorithm searches and compares a given image by exhaustively testing all exemplar models for the trained objects. Therefore, the time required for object recognition increases linearly relative to the number of objects. About 6.71 seconds



(a) Unclear Speech (b) Partial Speech (c) Partial Speech + Weak Localization

Figure 11. Confusion Matrix: shows confusion between target objects of interest and detected objects.

are required to search 2124 exemplar models for 25 objects using GPU-acceleration, and we estimate we would require about 22 seconds if the number of objects were increased to 100, clearly beyond the bar of interactive speeds.

By considering the clear, partial and unclear speech cases, we can allow the system to scale computationally by forgoing exhaustive evaluation of exemplars, and instead bounding the number of exemplars to 400, 800 and 1200, corresponding to maximum times of 2.7s, 3.7s and 4.6s, respectively. These results are also shown in Figure 10. For all results generated by the Marvin integrated system, we only show results for 25 objects in this figure, assuming that those will correspond to an upper bound on smaller numbers of objects.

These numbers (400, 800 and 1200) of exemplars are actually upper bounds on the numbers output by the algorithm for producing ranked exemplars discussed in Section 2.5. The actual numbers produced can be lower than these when the posterior distribution of objects is sharply peaked around a few objects. In this case we may use all the exemplars for those likely objects and there may be insufficient probability mass remaining to generate exemplars for the other objects. Thus in Figure 10, when localization is used in addition to speech, we see shorter times due to the more sharply peaked posteriors. This decrease in number of exemplars due to weak localization information is most apparent when speech itself is the least useful, i.e., the unclear and partial speech cases. In the unclear case, localization reduces the number of exemplars to 651 from 1200, and in the partial speech case, localization reduces the number of exemplars to 579 from 800.

We show in Section 3.2 that for 25 objects, this truncation of exemplars can be done while simultaneously increasing accuracy by considering speech and localization.

3.2. Accuracy Studies

The accuracy of retrieving the target object of interest within the returned results was calculated per the recommended rank orders. We treat this as a retrieval task and measure precision at N [= 1, 2, 3, and 4]. Figure 12 shows the precision of the test results for both the uncluttered "object-in-hand" scenes and the cluttered "object-on-shelf".

The precision results with the clear speech commands are 100% in our test cases because the specific object of interest can be identified using the keywords in the speech commands. In contrast, the precision for the 1st top list drops to about 50% with unclear speech commands, which contains no keywords regarding objects. With the partial speech commands, the precision is increased up to 71% for the 1st top list. The precision for the unclear speech case with the weak location information shows similar performance with the partial speech case regarding the 1st top list and shows even better result for the rest of the top lists in the cluttered scenes (see Figure 12-(b)) because the cluttered scenes provide more vision features associated with the objects in the scene. However, the precision for the clear speech with the weak location information shows a little lower precision performance (see Figure 12-(a)). This is because the weak localization recommends other objects in the scene other than the object of interest the clear speech command specifies.

Generally speaking the object-in-hand results had slightly higher precision than the object-on-shelf results. This is to be expected as the clutter for object-in-hand is dramatically reduced, thus making the problem easier. However, it is interesting to note that the object-on-shelf scenario actually performed better than object-in-hand scenario when weak localization was used. This observation verifies the intuition that SURF features that match the background can augment SURF features matched on the object itself to provide actually more information that can be exploited by Marvin in concert with speech and eSVM.

Figure 11 shows how confusion of finding objects of interest is improved with adopting more modalities; speech and weak location information. The confusion with the unclear speech commands presents 50.4% accuracy and the confusion is improved by achieving 71.2% accuracy with the partial speech and 90% accuracy with both the partial speech and location-dependent object list.

In Figure 13, we show how user feedback can be used

as another modality and improves the recognition accuracy of the object of interest. The user has the ability to dicide whether the reported result is right or wrong. This information is transferred and updated to the integration engine described in Section 2.5, which increases or decreases the probabilities of the used exemplar models for next time. Preliminary experiements with learning show continuous increase in accuracy as the number of repeated tests increased, but more work remains to be done to establish the clear advantage of online learning.

4. Conclusion

We have described Marvin, a system that uses multiple sensing modalities, together with collaborative engagement with the user to understand spoken commands, observes the physical environment, perceives objects of interest, and provides useful information to the user at interactive speed. We have also described Marvin's scalability and accuracy performance tested in a real-world retail store. Marvin has superior recognition accuracy compared to using a vision system alone, especially in cluttered scenes with many objects. It is also able to scale to large numbers of objects by focusing computational resources on the objects most likely to be of interest.

Although not thoroughly tested in this paper, we expect that similar accuracy results can be achieved regardless of the number of objects as long as the speech command and the weak localization provide enough information. We plan to test this hypothesis in future versions of this work. One method to facilitate data collection is to expand our test from the retail store to larger area by deploying Marvin as an android app, using the cloud network for backend computation, and releasing Marvin as open-source.

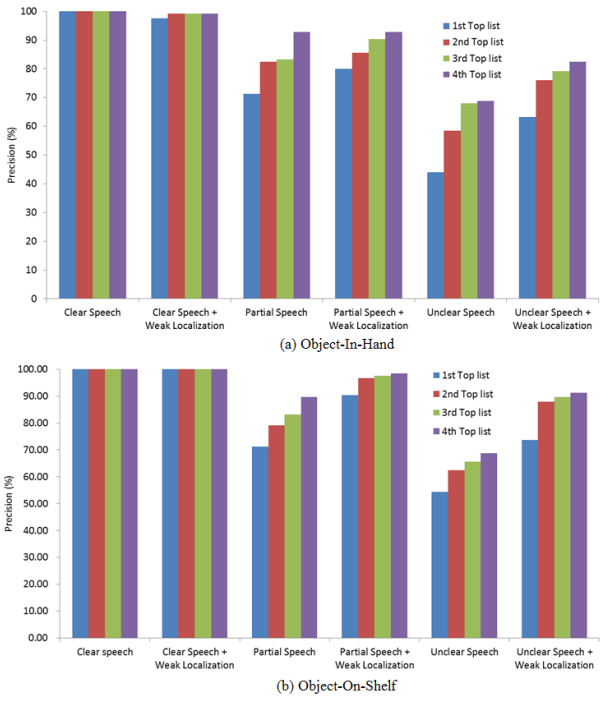


Figure 12. Precision: shows the overall accuray over the 25 objects for the six test cases.

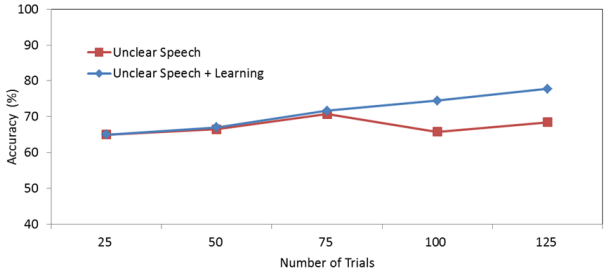


Figure 13. Average Accuary Comparison of Learning vs no Learning: The user feedback can improve the overall accuracy as the user uses the system longer.

We also plan to investigate more strategies for scaling up eSVM. One possibility is to use model recommendation techniques to inform the system of which exemplars are likely to be relevant in a given situation. We plan to integrate more online learning into Marvin to enable new informative keywords to be discovered automatically, and to customize keyword parameters to more accurately reflect their conditional probabilities given the actual objects. Such learning can be accomplished with direct feedback from the user, but also in a semi-supervised framework where some labels are not retrieved from the user.

References

[1] Amazon. User guided object identification, 2012.

[2] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, and M. Jordan. Matching words and pictures. *J. Mach. Learn. Res.*, 3:1107–1135, Mar. 2003.

[3] T. Cour, C. Jordan, E. Miltsakaki, and B. Taskar. Movie/Script: Alignment and Parsing of Video and Text Transcription. In *European Conference on Computer Vision*, 2008.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In C. Schmid, S. Soatto, and C. Tomasi, editors, *International Conference on Computer Vision & Pattern Recognition*, volume 2, pages 886–893, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, June 2005.

[5] I. Engines. omoby, 2012.

[6] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[7] Google. Google goggles, 2010.

[8] A. Gupta and L. Davis. Objects in action: An approach for combining action understanding and object perception. In *CVPR*. IEEE Computer Society, 2007.

[9] A. Gupta and L. Davis. Beyond nouns: Exploiting prepositions and comparative adjectives for learning visual classifiers. In *Proceedings of the 10th European Conference on Computer Vision: Part I*, ECCV '08, pages 16–29, Berlin, Heidelberg, 2008. Springer-Verlag.

[10] D. Huggins-Daines, M. Kumar, A. Chan, A. Black, M. Ravishankar, and A. Rudnicky. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held device. *Acoustics, Speech and Signal Processing*, 1, 2006.

[11] T. Malisiewicz, A. Gupta, and A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *IEEE International Conference on Computer Vision (ICCV)*, pages 89–96, 2011.

[12] H. S. Park, E. Jain, and Y. Sheikh. 3d social saliency from head-mounted cameras. *Neural Information Processing Systems (NIPS)*, 2012.

[13] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, 2009.