The Whole World in Your Hand: Active and Interactive Segmentation

Artur Arsenio[*] **Paul Fitzpatrick**[*] **Charles Kemp**[*] **Giorgio Metta**[**]

[*]MIT AI Lab
Cambridge, Massachusetts, USA

[**]Lira Lab, DIST, Uiversity of Genova
Genova, Italy

Abstract

This paper presents three approaches to object segmentation, a fundamental problem in computer vision. Each approach is aided by the presence of a hand or arm in the proximity of the object to be segmented. The first approach is suitable for a robotic system, where the robot can use its arm to evoke object motion. The second method operates on a wearable system, viewing the world from a human's perspective, with instrumentation to help detect and segment objects that are held in the wearer's hand. The third method operates when observing a human teacher, locating periodic motion (finger/arm wagging or tapping) and using it as a seed for segmentation. We show that object segmentation is a key resource for development. We demonstrate that, once high-quality object segmentation is available, it is possible to train up both high-level visual modules (object recognition and localization) and to enhance low-level vision (orientation detection).

1. Introduction

The presence of a body changes the nature of perception. The body is a source of constraint on interpretation, opportunities for experimentation, and a medium for communication. Hands in particular are very revealing, since they interact directly and flexibly with objects. In this paper, we demonstrate several methods for simplifying visual processing by being attentive to hands, either of humans or robots. Our first argument is that in a wide range of situations, there are many cues that can be used to make object segmentation an easy task. Object segmentation or figure/ground separation is a long-standing problem in computer vision, due to the fundamental ambiguities involved in interpreting the 2D projection of a 3D world. No matter how good a passive system is at segmentation, there will be times when only an active approach will work, since visual appearance can be arbitrarily deceptive.

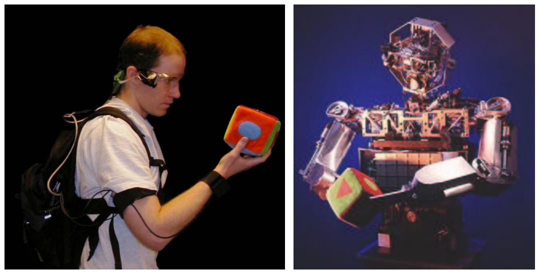
Figure 1: The platforms.

Our second argument is that object segmentation is truly a key ability worth getting excited about. We support this by demonstrating that when segmentation is available, several other important vision problems can be dealt with successfully – object recognition, object localization, edge detection, etc.

Here are the three situations we look at:

(i) Active segmentation for a robot viewing its own actions. A robot arm probes an area, seeking to trigger object motion and then identify the boundaries of an object through its motion.

(ii) Active segmentation for a wearable system viewing its wearer's actions. The system monitors human action, issues requests, and uses active sensing to detect grasped objects held up to view.

(iii) Protocol-based segmentation for a robot viewing a human's actions. Segmentation is achieved by detecting and interpreting natural human showing behavior such as finger tapping, arm waving, or object shaking.

2. Segmentation on a robot

The most well-known instance of active perception is active vision. The term "active vision" is essentially synonymous with moving cameras. Active vision work on Cog is oriented towards opening up the potentially rich area of manipulation-aided vision, which is still largely unexplored. But there is much to be gained by taking advantage of the fact that robots are actors in their environment, not simply passive observers. They have the opportunity to examine the world using causality, by performing probing actions and learning from the response. In conjunction with a developmental framework, this could allow the robot's experience to expand outward from its sensors into its environment, from its own arm to the objects it encounters, and from those objects both back to the robot itself and outwards to other actors that encounter those same objects.

As a concrete example of this idea, Cog was given a simple "poking" behavior, whereby it selects locations in its environment, and sweeps through them with its arm (Fitpatrick and Metta, 2002,). If an object is within the area swept, then the motion signature generated by the impact of the arm with that object greatly simplifies segmenting that object from its background, and obtaining a reasonable estimate of its boundary (see Figure 2). The image processing involved relies only on the ability to fixate the robot's gaze in the direction of its arm. This coordination is easy to achieve either as a hard-wired primitive or through learning (Fitpatrick and Metta, 2002,). Within this context, it is possible to collect excellent views of the objects the robot pokes, and the robot's own arm.

Since the robot had a limited reach, this activity required the cooperation of a human companion to bring the robot interesting objects to poke. The behavior could also be preempted by the companion; when the robot fixated an object and was about to reach for it, the companion could choose to poke the object instead, in which case the robot would refrain from acting.

This choice of activity has many benefits. *(i)* The motion signature generated by the impact of the arm with a rigid object greatly simplifies segmenting that object from its background, and obtaining a reasonable estimate of its boundary (see Figure 2). This "active segmentation" procedure is key to automatically acquiring training data of sufficient quality to support the many forms of learning described in the remainder of this paper. *(ii)* The poking activity also leads to object-specific consequences, since different objects respond to poking in different ways. For example, a toy car will tend to roll forward, while a bottle will roll along its side. *(iii)* The basic operation involved, striking objects, can be performed by either the robot or its human companion, creating a controlled point of comparison between robot and human action.

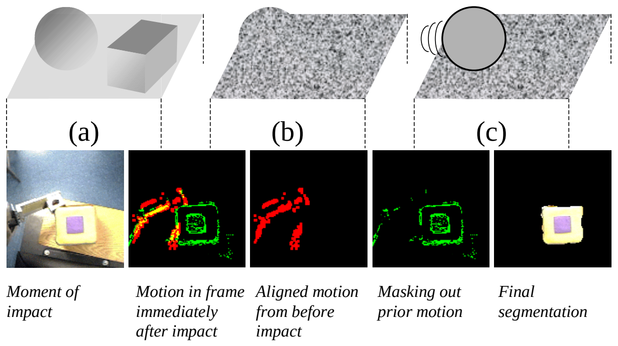
(a)
(b)
(c)
Moment of impact
Motion in frame immediately after impact
Aligned motion from before impact
Masking out prior motion
Final segmentation

Figure 2: Cartoon motivation (top) for active segmentation (bottom). Human vision is excellent at figure/ground separation (top left), but machine vision is not (top center). Coherent motion is a powerful cue (top right) and the robot can invoke it by simply reaching out and poking around. The lower row of images show the processing steps involved. The moment of impact between the robot arm and an object, if it occurs, is easily detected – and then the total motion after contact, when compared to the motion before contact and grouped using a minimum cut approach, gives a very good indication of the object boundary. Active segmentation. The robot arm is deliberately driven to collide with an object. The apparent motion after contact, when masked by the motion before contact, identifies a seed foreground (object) region. Such motion will generally contain fragments of the arm and environmental motion that escaped masking. Motion present before contact is used to identify background (non-object) regions. An optimal object region is computed from the foreground and background information using graph cuts (Boykov and Kolmogorov, 2001).

3. Segmentation on a wearable

Cite research abstract (Kemp, 2002).

Wearable computing systems have the potential to measure a great deal of the sensory input and physical output of a person as he or she experiences everyday activities. Much can be learned through passive observation of these measurements. However, if we can also find ways for the wearable system to control the behavior of the person wearing the system, many learning tasks can be made easier.

We are designing a wearable creature that exploits a cooperative human for its own learning purposes. Essentially, the human wearing the system becomes the host for a parasitic creature that wishes to learn about the world by watching and sometimes controlling the more experienced host as he or she goes through common human activities in the day. By using the same sensory input as the host and co-opting the output behaviors of the host, the wearable creature serves as a top layer of control in a subsumption architecture In essence, by controlling the human in an effort to learn the wearable creature changes the human into a humanoid robot.

As an initial exploration into this class of wearable applications, we are creating a wearable system that attempts to learn commonsense about everyday actions as they relate to objects and changes to the environment. As shown in the figure at the top of the page, the system currently consists of a glasses-mounted camera from which the creature watches the world and 3 Intersense devices, each of which provides an absolute orientation, with which the creature measures the kinematic configuration of the host's dominant arm. The creature will also serve as a high level controller that attempts to co-opt the host's behaviors by requesting actions through headphones. For example, the creature might request that its host repeat an action by uttering, "do that again!", which with a cooperative host should help the creature segment the activity into meaningful parts. Likewise the creature might ask to see an object of interest better, thereby influencing the person to inspect the object more closely. More generally, by requesting actions the wearable creature can test hypotheses it has made about actions and their effect in the world.

We hope to develop a viable system for the acquisition of commonsense related to everyday human activities. A successful creature would be able to learn and control a set of common behaviors performed by a cooperative human and would be able to relate common action patterns to the visual appearance of objects and to observed changes in the world.

4. Segmentation through periodicity

Cite research abstract (Arsenio, 2002).

4.1 Waving the object

4.2 Waving the hand/arm/finger

Through direct embodied actuation

Teacher waves objects to the robot

Human arm/hand/finger waving for segmentation

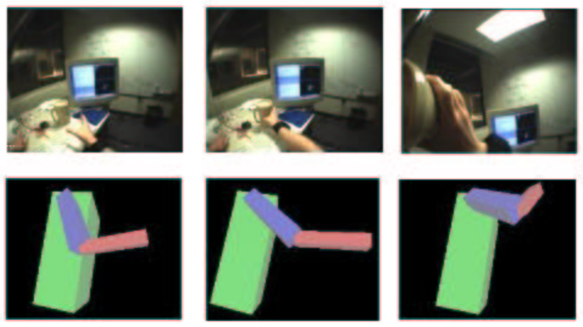

Figure 3: cckemp 1

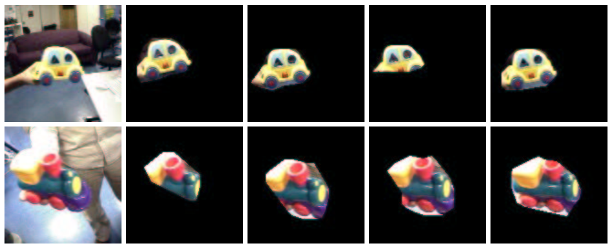

Figure 4: artur 1

Robust: to humans, other objects moving in the background – ignored if they are non-periodic

Powerful technique for segmenting stationary objects e.g. Objects painted in a book

Heavy, stationary objects: table, sofa

Using relative size of the arm, can get relative size of objects

Through indirect embodied actuation

Teacher waves arm/hand/finger to create a pop-out effect finger trajectory groups segmented regions together

Method:

1) Detect Motion ¿ Th

Detect Skin tones for arm/hand/finger ¿ Th

2) Initialize grid of points in moving region

3) Track these points over the window interval (32 and 64 frames)

- using optical flow

4) Compute WFFTs for the temporal signal of each point

5) Select signals with a single peak of energy over all frequencies

5. Building on segmentation

As a specific example of development, the segmented views provided by poking of objects and actors by poking can be collected and clustered as shown in Figure 7. Such views are precisely what is needed to train up an ob-

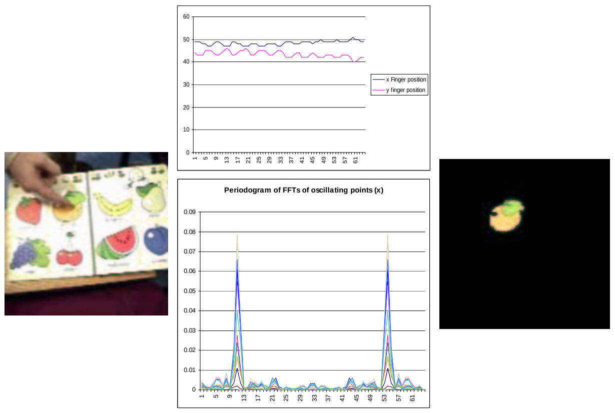

Figure 5: artur 2

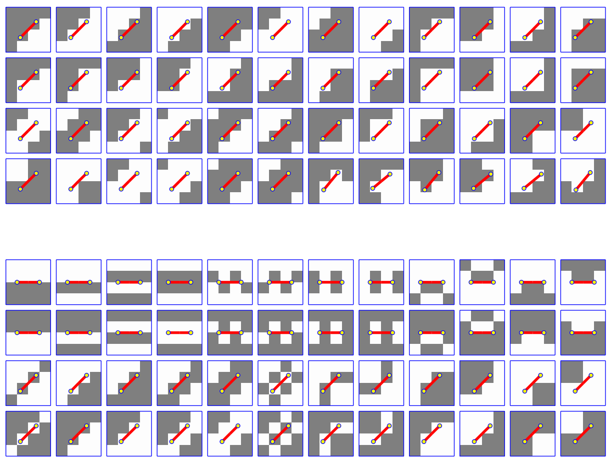

Figure 6: The empirical appearance of edges. Each 4×4 grid represents the possible appearance of an edge, quantized to just two luminance levels. The dark line centered in the grid is the average orientation that patch was observed to have in the training data. The upper set of patches are the most frequent ones that occur in training data consisting of about 500 object segmentations. The lower set of patches are a selection of patterns chosen to illustrated the diversity of possible patterns that can occur. The oriented features represented include edges, thin lines, thick lines, zig-zags, corners etc. It is difficult to imagine a set of conventional filters that could respond correctly to the full range of features seen here – all of which appeared multiple times in object boundaries in real images.

ject detection and recognition system, and follow those objects and actors into other, non-poking contexts.

As well as giving information about the appearance of objects, the segmented views of objects can be pooled to train up detectors for more basic visual features – for example, edge orientation. Once an object boundary is known, the appearance of the edge between the object and the background can be sampled along it, and labelled with the orientation of the boundary in their neighborhood. Figure 6 shows an orientation filter trained up from such data that can work at much finer scales than normally possible when the filter is derived from an ideal edge model such as that of (Chen et al., 2000,). The "catalog" of edge appearances found shows that the most frequent edge appearances is an "ideal" straight, noise-free edge, as might be expected (top of Figure 6) – but a remarkable diversity of other forms also occur which are far less obvious (bottom of Figure 6).

Can build on segmentation to learn about cool and useful stuff.

Learning about orientation

A low-level example.

Learning to recognize objects

A higher-level example.

6. Learning about orientation

Orientation is an important visual cue for many purposes, such as object segmentation, recognition, and tracking. It is associated with neighborhoods rather than individual points in an image, and so is inherently scale dependent. At very fine scales, relatively few pixels are available from which to judge orientation. Lines and edges at such scales are extremely pixelated and rough. Orientation filters derived from analytic considerations, with parameters chosen assuming smooth, ideal straight lines or edges (for example, (Chen et al., 2000)) are more suited to larger neighborhoods with more redundant information. For fine scales, an empirical approach seems more promising, particularly given that when the number of pixels involved is low, it is practical to sample the space of all possible appearances of these pixels quite densely.

The data collected during segmentation allows us to explore how edges truly look in "natural" images, by simply building up a catalog of edge fragments seen around the boundaries of objects. Of course, such a catalog is only practical at small scales. We worked with 4×4 pixel windows, a size is chosen to be large enough to be interesting, but small enough for the complete range of possible appearances to be easily visualized. Even at this scale, manual data collection and

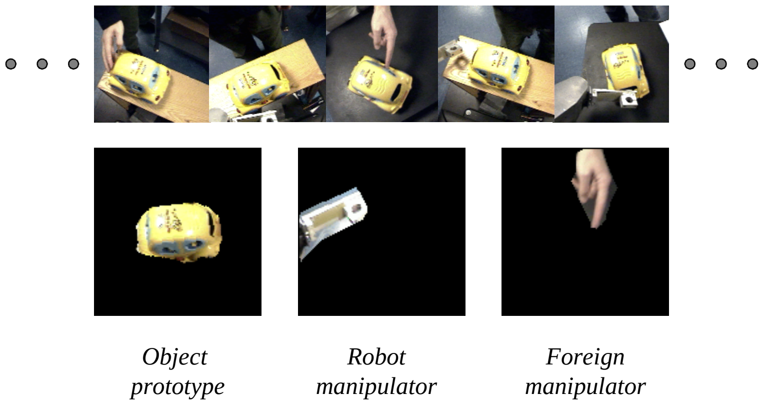

Figure 7: The top row shows sample views of a toy car that the robot sees during poking. Many such views are collected and segmented as described in (Fitzpatrick, 2003,). The views are aligned to give an average prototype for the car (and the robot arm and human hand that acts upon it). To give a sense of the quality of the data, the bottom row shows the segmented views that are the best match with these prototypes. The car, the robot arm, and the hand belong to fundamentally different categories. The arm and hand cause movement (are actors), the car suffers movement (is an object), and the arm is under the robot's control (is part of the self).

labelling would be extremely tedious, so it is definitely advantageous to have a robotic system to automatically compile and label a database of the appearance of oriented features. These features were extracted by sampling image patches along object boundaries, which were in turn determined using active segmentation. The resulting catalog of edge appearances proved remarkably diverse, although the most frequent appearances were indeed the "ideal" straight, noise-free edge.

7. Learning to recognize objects

With any of the active segmentation behaviors introduced here, the system can familiarize itself with the appearance of nearby objects. This section is concerned with learning to locate and recognize those objects whenever they are present, even when the special cues used for active segmentation are not available. Based on segmentations from the poking method.

7.1 Approaches to object recognition

Physical objects vary greatly in shape and composition. This variety is reflected in their visual appearance. Unfortunately, it is not a straightforward matter to recover object properties from appearance, since there are many other factors at work – illumination, relative pose, distance, occlusions, and so on. The central challenge of object recognition is to be sensitive to the identity of objects while maintaining invariance in the face of other incidental properties. There are at least two broad approaches to recognition, geometry-based and appearance-based.

Geometry-based recognition

Image formation is a geometric process, so one way to approach recognition is to model invariant geometric relationships that hold for a particular class of object. These relationships can be between points, lines, surfaces or volumes. They may be known for many possible views of the object, or just one. When a new scene is presented, geometric relationships in it are measured and matched against the model. There are many details in what to measure and how to do the matching (there is a good review in (Selinger, 2001)). The main difficulty is the combinatorics of the search involved. There are a lot of free parameters to search over when we try to match an unsegmented scene to an object model – in particular, which elements in the scene correspond to the object, and what the transformation is between those elements and the object. For high-speed performance, geometric hashing is a useful technique (for a review see (Wolfson and Rigoutsos, 1997)). In this method, geometric invariants (or quasi-invariants) are computed from points in model (training) images, then stored in hash tables. Recognition then simply involves accessing and counting the contents of hash buckets. One possibility for the geometric invariants is to take a set of points selected by an interest operator, and use each pair of points in turn to normalize the remainder by scale and rotation. The position of the normalized points can be stored in a 2D hash table. Invariants in 3D are more difficult to achieve, but various solutions have been proposed.

Appearance-based recognition

While the world is geometric in nature, geometric features are not particularly easy to extract reliable from images. In appearance-based recognition, the focus is shifted from the intrinsic nature of an object to properties that can be measured in images of that object, including geometric properties but also surface properties such as color or texture.

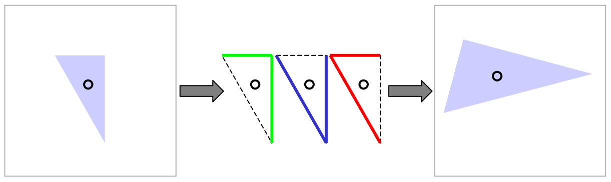

Figure 8: Rich features for hashing.

For example, (Swain and Ballard, 1991) proposed using the set of colors present in segmented views of an object as their representation. Regions of an image that contain the same color mix (as determined by histogram intersection) could contain the object. Histogram back-projection can be used to quickly filter out regions unlikely to contain the object, by assigning each pixel a weight based on the frequency of its color in the histogram. Some form of region-growing method then accumulates this evidence to find plausibly colored regions. This method is very fast, and for objects with distinctive colors it can be useful as a prefilter to more computationally expensive processing. Color is also intrinsically somewhat robust to scale and rotation, but is sensitive to changes in the spectral profile of illumination.

Many appearance-based methods are window-based. A classifier is built that operates on a rectangular region of interest within an image. That window is moved across the entire image, at multiple scales, and sometimes multiple orientations. Responses of the classifier across locations and scales are combined using various heuristics. Variation in orientation (rotation in depth) is typically dealt with by training up multiple recognizers for various poses. These poses can be sampled quite sparsely, but still each pose requires iteration of the search procedure. Ther are ways to speed all this up, for example using a cascade of classifiers that reject clearly non-target windows early, devoting full analysis only to plausible targets. The actual classifier itself can be based on eigenspace methods or many other possibilities.

Probabilistic methods like Schiele. Version with Roy.

7.2 Hashing with rich features

The approach used here is like geometric hashing, but uses richer features that include non-geometric information. Geometric hashing works because pairs of points are much more informative than single points. An ordered pair of points defines a relative scale, translation, and orientation (in 2D). Further points may be needed for more general transformations, such as affine or projective, or to move to 3D (Wolfson and Rigoutsos, 1997,). But, even staying with the 2D case, using just two points is somewhat problematic. First of all, they are not at all distinctive – any two points in an image could match any two points in the model. Hashing doesn't require distinctiveness, but it would be a useful prefilter. Secondly, there is no redundancy; any noise in the points will be directly reflected in the transformation they imply.

One possibility would be to use triplets of points, or more. Then we have distinctiveness and some redundancy. But we also have an explosion in the number of possible combinations. Pairs of points are just about manageable, and even then it is better if they are drawn from a constrained subset of the image. For example, the work of (Roy and Pentland, 2002) uses histograms of the distance and angles between pairs of points on the boundary of an object for recognition.

In this work, pairs of edges (or more generally, any region with well-defined and coherent orientation) are used instead of pairs of points (Figure 8). Pairs of edges are more distinctive than pairs of points, since they have relative orientation and size. And if used carefully during matching, they contain redundant information about the transformation between image and model. A disadvantage is that edges are subject to occlusion, and edges/regions found automatically many be incomplete or broken into segments. But in all but the most trivial objects, there are many pairs of edges, so this approach is at least not doomed from the start.

The orientation filter developed earlier is applied to images, and a simple region growing algorithm divides the image into sets of contiguous pixels with coherent orientation. For realtime operation, adaptive thresholding on the minimum size of such regions is applied, so that the number of regions is bounded, independent of scene complexity. In "model" (training) views, every pair of regions belonging to the object is considered exhaustively, and entered into a hash table, indexed by relative angle, relative position, and the color at sample points between the regions (if inside the object boundary).

A useful extension of geometric hashing is coherency, where each match implies a par-

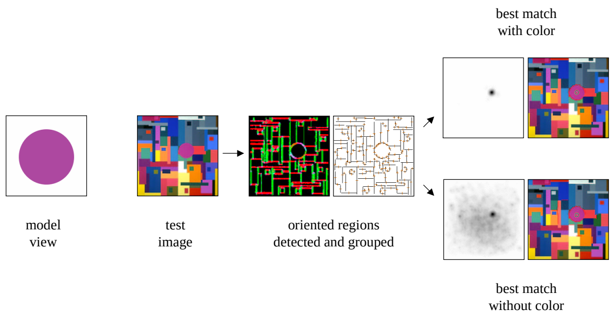

Figure 9: A simple example of object localization: finding the circle in a Mondrian.

ticular transformation, and only "coherent" matches are aggregated (for example, see (Lamiroy and Gros, 1996)). This could be applied in the present instance. For speed, an abbreviated version is used here, where we filter by the centroid location each match implies for the object (this is information we would like to have anyway). There is no coherence checking by scale and orientation at the matching stage. This procedure means that we perform object localization simultaneously with matching.

Oriented regions are relatively sparse. Experiments showed that on a fast machine (800MHz) and at low resolution (128×128) it is possible to use triplets of regions as features at close to real-time. These can be very distinctive, and very redundant, and non-trivial objects have very very many possible triplets. But the frame-rate was just too slow (approximately 1 Hz) to be worth using here.

At the other extreme, another possibility would be just to use single edges. But they are not very distinctive, and sampling colors at an edge (generally a high-variance area) is problematic.

7.3 Searching for a synthetic object in a synthetic scene

As a simple example of how this all works, consider the test case shown in Figure 9. The system is presented with a model view of the circle, and the test image. For simplicity, the model view in this case is a centered view of the object by itself, so no segmentation is required. The processing on the model and test image is the same – first the orientation filter is applied, and then regions of coherent orientation are detected. For the circle, these regions will be small fragments around its perimeter. For the straight edges in the test image, these will be long. So finding the circle reduces to locating a region where there are edge fragments at diverse angles to each other, and with the distance between them generally large with respect to their own size. Even without using color, this is quite sufficient for a good localization in this case. The perimeter of the circle can be estimated by looking at the edges that contribute to the peak in match strength. The algorithm works equally well on an image of many circles with one square.

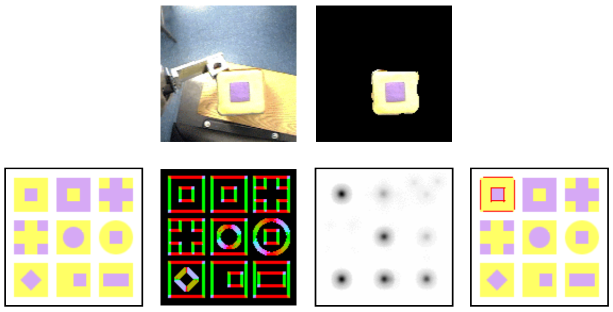

Figure 10: Looking for the best match to a cube found through poking in a synthetic image of various approximations and distractors. The superimposed red lines on the rightmost image indicate the detected position of the object and the edges implicated. The image on its immediate left shows the strength of evidence for the object across the entire image, which lets us rank the distractors in order of attractiveness.

7.4 Searching for real objects in synthetic scenes

In Figure 10, we take a single instance of an object found through poking, and search for it in a synthetic image containing an abstract version of it along with various distractors. The algorithm picks out the best match, and lets us rand the distractors in order of salience. It is clear that a yellow square with anything in it is a good match, and having the internal purple square adds another boost. The closest distractor is a yellow square with a purple square inside it, rotated by 45°.

7.5 Recognizing real objects in real images

Figure 11 shows examples of the cube being resegmented in real images.

Testing on a set of 400 images of four objects (about 100 each) being poked by the robot, with half the images used for training, and half for testing, gives a recognition error rate of about 2%, with a median localization error of 4.2 pixels in a 128×128 image

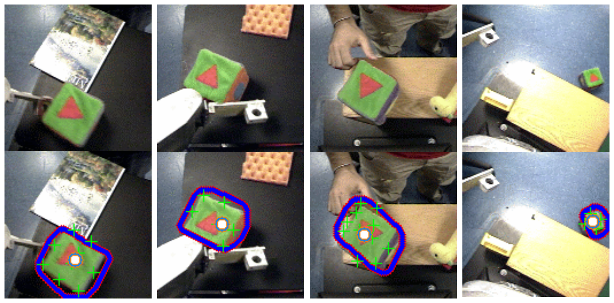
Figure 11: The cube being recognized, localized, and segmented in real images. The image in the first column is one the system was trained on. The image in the remain columns are test images. Note the scale invariance demonstrated in the final image.

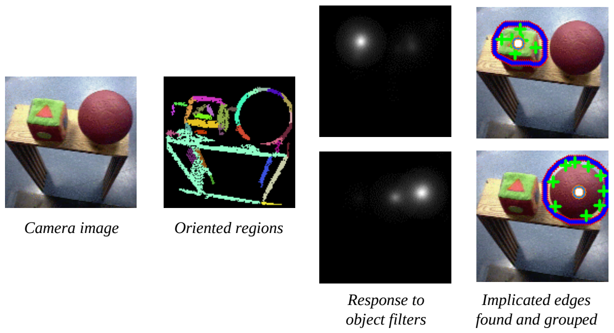
Camera image
Oriented regions
Response to object filters
Implicated edges found and grouped

Figure 12: resegment multiple

(as determined by comparing with the center of the segmented region given from poking). By segmenting the image by grouping the regions implicated in locating object, and filling in, a median of 83.5% of the object is recovered, and 14.5% of the background is mistakenly included (again, determined by comparison with the results of poking).

7.6 Multiple objects simultaneously

See Figure 12. Don't actually do this normally, deal with foveation and egomap (described later).

7.7 Online training

Finally, Figure 13 shows recognition and training occurring online.

8. Discussion and conclusions

Acknowledgements

Funds for this project were provided by DARPA as part of the "Natural Tasking of Robots Based on Human Interaction Cues" project under contract number DABT 63-00-C-10102, and by the Nippon Telegraph and Telephone Corporation as part of the NTT/MIT Collaboration Agreement.

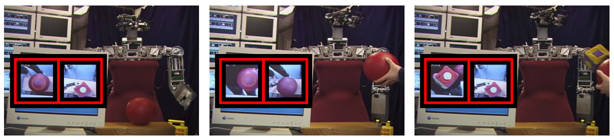
Figure 13: This figure shows stills from a short interaction with Cog. The area of the first frame highlighted with a square shows the state of the robot – the left box gives the view from the robot's camera, the right shows an image it associates with the current view. If the object confuses another object with what it has already seen, such as the ball in the first frame, this is easily to fix by poking. IMPROVE THIS DESCRIPTION

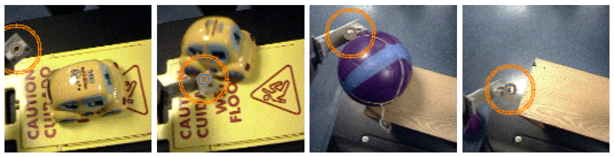
Figure 14: This doesn't really need to be in the paper.

References

Arsenio, A. (2002). Boosting vision through embodiment and situatedness. In *MIT AI Laboratory Research Abstracts*.

Boykov, Y. and Kolmogorov, V. (2001). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 359–374.

Chen, J., Sato, Y., and Tamura, S. (2000). Orientation space filtering for multiple orientation line segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(5):417–429.

Fitpatrick, P. and Metta, G. (2002). Towards manipulation-driven vision. In *IEEE/RSJ Conference on Intelligent Robots and Systems*.

Fitzpatrick, P. (2003). First contact: Segmenting unfamiliar objects by poking them. submitted to IROS.

Kemp, C. C. (2002). Humans as robots. In *MIT AI Laboratory Research Abstracts*.

Lamiroy, B. and Gros, P. (1996). Rapid object indexing and recognition using enhanced geometric hashing. In *Proceedings of the 4th European Conference on*

Computer Vision, volume 1, pages 59–70, Cambridge, England.

Roy, D. and Pentland, A. (2002). Learning words from sights and sounds: A computational model. *Cognitive Science*, 26(1):113–146.

Selinger, A. (2001). *Analysis and Applications of Feature-Based Object Recognition*. PhD thesis, University of Rochester, Rochester, New York.

Swain, M. J. and Ballard, D. H. (1991). Color indexing. *International Journal of Computer Vision*, 7(1).

Wolfson, H. and Rigoutsos, I. (1997). Geometric hashing: an overview. *IEEE Computational Science and Engineering*, 4:10–21.