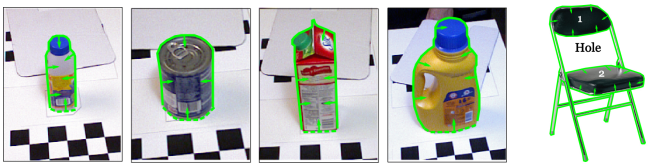
Visual Segmentation of "Simple" Objects for Robots

Ajay K. Mishra and Yiannis Aloimonos
University of Maryland, College Park
Maryland, 20742

***Abstract*—The ability to automatically segment a "simple" object of any size from its background is important for an active agent (e.g. a robot) to interact effectively in the real world. Recently, we proposed an algorithm [12] to segment a "simple" object in a scale invariant manner, given a point anywhere inside that object. However, in [12], a strategy to select the point inside a "simple" object was not provided. In this paper, we propose a new system that automatically selects the points inside different "simple" objects in the scene, carries out the segmentation process for the selected points, and outputs only the regions corresponding to the "simple" objects in the scene. The proposed attention mechanism for the segmentation problem utilizes, for the first time, the concept of border ownership [17].**



(a) (b)

Figure 1: (a) and (b) show "simple" objects and a complex object respectively. In (a), depth and contact boundaries are shown by the solid and dotted lines respectively while the border ownership of the boundary pixels are shown by the arrows. In (b), a hole is shown as well which is a "simple" object with reverse border ownership.

I. Motivation

For the robots interacting with their surroundings, object perception is as important a capability as navigation. Robots need the navigation capability to reach objects of interest which are the outcome of object perception. Without object perception, the robots would not have any dynamic target to navigate to except the static targets such as doors in a room or a fixed location using GPS. In this respect, object perception and navigation are complementary capabilities.

However, unlike navigation, object perception is not well defined. In fact, the exact definition of an object and what constitutes perception are both open questions. Perception is an intricate phenomenon involving not only visual inputs but also other cognitive modalities. To simplify, we define a minimal form of object perception: knowing the boundary of an object. This minimal object perception might even be sufficient for some basic interactions with the object such as picking, moving, and pushing it. But in order to complete the definition of object perception, we have to define an object.

We define a "simple" object in terms of only low-level visual cues without using any high-level semantics as they are hard to quantify. A "simple" object is a compact region in the scene enclosed by the edge pixels at depth and/or contact boundaries with "correct" border ownership. The border ownership of a boundary pixel is the information about the side containing the object. Since the boundary pixels of both types and their border ownership can be determined using low-level visual cues (read sections V,VI), "simple" objects can be extracted in a truly bottom-up fashion without any prior knowledge. Figure 1a shows the examples of "simple" objects. Figure 1b shows a complex object consisting of multiple "simple" objects. In the rest of the paper, we are going to use the term "object" in place of "simple" object for better readability.

Segmenting objects in a purely bottom-up fashion has two main advantages: first, a recognition module can take advantage of the known shape and scale of the segmented entities to reliably assign them semantic labels; second, the number of segmented entities to be processed by a recognition module is only a few more than actual number of objects in the scene. While recognition following segmentation seems logical, it is in contrast with the standard approach wherein objects are recognized before being localized and segmented. The standard techniques, although successful in some multimedia systems [5], are challenged when used in robotics. For instance, localization, using standard techniques, usually means a bounding box around a detected object which is not enough for robotics purposes. Our proposed segmentation approach will first identify different parts of visual space as candidate "objects" to which we then devote our efforts to recognize and interact.

II. Introduction

The system, proposed in this paper, builds upon our previously proposed point-based segmentation strategy [12] that finds the "optimal" closed contour around a given point in the scene. If that point lies inside an object, the resulting closed contour is the boundary of that object. However, if that point does not lie inside any object, the output is still a closed contour that does not correspond to the boundary of any object in the scene. We will call this a non-object closed contour. Thus, we propose strategies to select points inside objects, and to reject the non-object closed contours due to the points selected outside of objects.

The key concept used in this paper is the **border ownership** [17] of the boundary pixels, which means knowing their object side. Besides helping select points inside the objects, the border ownership information also helps differentiate between the closed contours corresponding to the objects and the non-object closed contours. A closed contour corresponding to

an object is made up of the boundary pixels whose border ownership points to the interior of that closed contour whereas the pixels on a closed contour corresponding to the boundary of a hole will have border ownership pointing outside of it. See figure 1b for an example of a hole.

We also introduce the concept of **contact boundary** to identify the portion of the boundary of a static object touching the surface such as a table or floor. A significant part of the boundary of a static object in the scene has a depth discontinuity across it. But, across the portion of the boundary where the object meets with the resting surface, depth varies smoothly. We call this portion of the boundary contact boundary. Figure 1a shows the contact boundary of the objects as the dotted line.

The likelihood of a pixel to be at the boundary of an object is stored in a probabilistic boundary edge map of the scene. The pixels on the boundary of a moving object can be identified as they also lie on the motion boundaries in the scene. Identifying the pixels on the boundary of a static object, however, depends upon the state of the camera. If the camera is moving, depth boundary can be located using discontinuity in the optical flow map. If a stereo pair of cameras is used, depth boundary can be located using discontinuity in disparity map. While the pixels at a depth discontinuity can be determined using optical flow or disparity maps, the pixels at a contact boundary needs some additional information. In this paper, we assume to know the color distribution of the surface, the objects are resting on. With this information, an edge pixel at contact boundary has significantly different color distribution on its two sides and the color distribution on one of the side is similar to that of the surface. More about this later in section IV-C.

In this paper, we explain the system assuming that there is a moving camera which is looking at the static objects on a table of known color distribution. Although this is just one of different scenarios possible considering the state of camera and the objects, the principles illustrated in the paper can be easily applied to any of the other cases to segment the objects. An overview of our system is as follows: A probabilistic boundary edge map is generated using color, texture and motion cues (see section IV). For the pixels with significant boundary likelihood, the border ownership is also estimated. Using the border ownership information, a set of points inside different objects are automatically selected (see section V). Around each point, [12] finds the closed contour in the probabilistic boundary edge map (a brief overview is given in section VI). A subset of resulting closed contours that uniquely corresponds to the objects is finally selected as the output of the system (section VII). To purge any spurious closed contour still remaining in the output, we also enforce temporal persistence as explained in section VII-C.

Our two main contributions are:

- A new attention mechanism, designed for the segmentation problem. The only requirement on this attention system is that the points should lie inside the objects for the point-based segmentation strategy[12] to successfully segment them.
- A method to select only those closed contours that correspond to the objects in the scene while rejecting duplicate and non-object closed contours.

III. Related Work

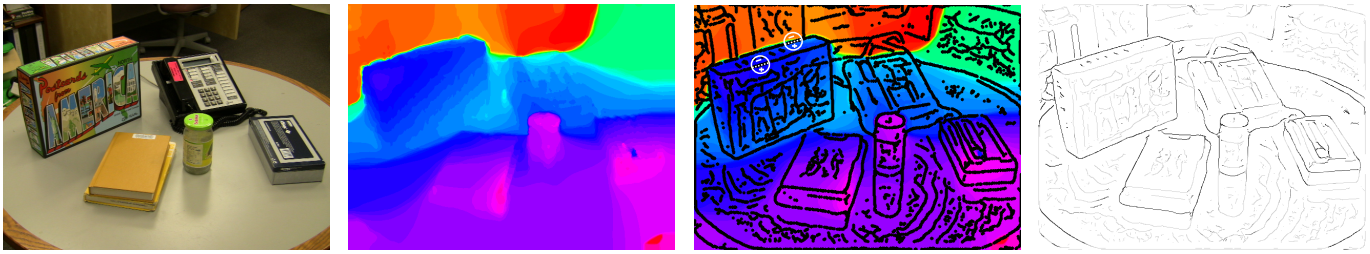
Attention is classified into two categories based on whether its deployment over a visual scene is guided by scene features or by intention: the first is called bottom-up and is driven by low-level processes; the second refers to top-down processes. Most work has happened in bottom-up attention. The feature integration theory [15] has inspired many bottom-up models of visual attention [8, 1]. The most popular is the one proposed by L. Itti et al. [10] and it has become a standard model of bottom-up visual attention, in which saliency due to primitive features such as intensity, orientation and color are computed independently. A model like this is not appropriate for our robots, because it often fails when attention needs to be focused on an object.

Less known are two recent works in fixation and attention [9, 16], that do not follow the main stream of thought. [9] shows, using systematic experiments, that humans are looking at objects as if they knew them before they became aware of their identity. That is, humans look as if they know, before they know! How could that be possible? [16] offers an alternative to the traditional saliency theories built on the assumption that "early" features (color, contrast, orientation, motion, and so forth) drive attention directly. Specifically, through meticulous experiments, it suggests that observers attend to objects, and this hypothesis has a better predictive power in the data than any other theory. But how can we look at an object without knowing about it, since we haven't looked at it yet?

The border ownership has been reported to be registered by the neurons in a primate's visual cortex [17, 4]. Zhou et al. show that the depth information is most important in determining the border ownership of a pixel at the boundary of an object. They also report that the determination of the border ownership happens as a result of local processing of visual cues. In computer vision literature, a popular term for border ownership is figure/ground assignment. Fowlkes et al. [6] use local shape information to determine the figure/ground side for an edge pixel. But, compared to static monocular cues, depth and motion information are stronger cues in determining border ownership of a boundary edge pixel.

IV. Probabilistic Boundary Edge Map

In the probabilistic boundary edge map, the intensity of a pixel is set to be the probability to be either depth or contact boundary in the scene. The probability to be at a depth boundary can be determined by checking for a discontinuity in the optical flow map at the corresponding pixel location because depth discontinuity introduces discontinuity in the optical flow map as well. But the exact location of discontinuities in optical flow maps often do not match with the true object boundaries, a well known issue for optical flow algorithms. To account for this, we use static cues such as color and texture to, first, find all possible boundary locations in the image which are the edge pixels with positive color and/or texture gradient. Then, the probability of these edge pixels to be on depth and contact boundary is determined. The maximum of two



(a) (b) (c) (d)

Figure 2: (a) The first frame of the image sequence. (b) 2D optic flow map. (c) All the edge pixels with non-zero color and texture gradient overlaid on the flow map. (d) The final probabilistic boundary edge map (the darkness of an edge pixel is proportional to its probability of being at an object boundary).

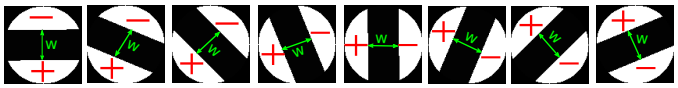


Figure 3: The oriented disc filters with opposite polarity. The corresponding orientation values (in degrees) are shown at the bottom of the figure. The middle zone (of width w) in the filters are suppressed to tolerate the misalignment between the optical flow boundary, and the actual boundary of the objects. The radius of the disc is 0.015 times the image diagonal and w is 0.2 times the disc radius.

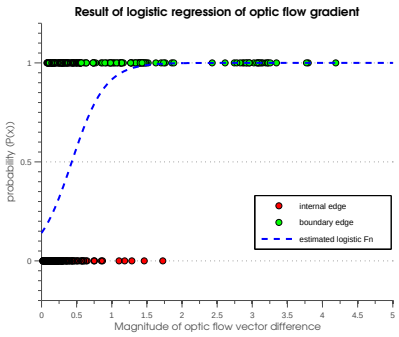


Figure 4: The estimated logistic function for converting the optic flow gradients across edge pixels into their probability of being at a depth discontinuity.

probability is assigned as the probability of an edge pixel to be on object boundary, $P_B(x, y)$. Note that color and texture gradient values do not participate in determination of the boundary probability.

A. *Boundary localization using static cues*

Using color and texture cues, all locations in frame 1 with positive color and/or texture gradient are detected and stored in a binary edge map [11]. See Figure 2c for an example of the binary edge pixels overlaid on the optical flow map. At all the edge pixels, the dominant tangential direction, which can be one of eight quantized values between 0 and π is calculated as well. The binary edge map, by selecting only a subset of all pixels, effectively assigns zero probability to the pixels from inside smooth areas in the scene to be on the boundary of an object.

However, the boundary probability of the edge pixels can not be estimated using the color or texture gradient at their locations. An edge pixel with a high color or texture gradient value can be both inside and on the boundary of an object. We use motion cues and color information of the surface to determine the probability of the edge pixels to be depth boundary and contact boundary respectively.

B. *Probability to be depth boundary*

Firstly, we compute the optical flow flow gradient at binary edge pixels. [3] is used to calculate optical flow map using two consecutive frames, an example of a color-coded flow map is shown in figure 2b. Eight disc filters for the eight possible orientations of the edge pixels are designed (Figure 3.) By placing the appropriately oriented disc filter in the optical flow map at the location corresponding to an edge pixel, we compute the optical flow gradient as the magnitude of the difference in the mean optical flow vectors $||\vec{V_+} - \vec{V_-}||$, where $\vec{V_+}$ and $\vec{V_-}$ are the mean optical flow vectors in the two halves of opposite polarity indicated by indicated by " $+$ " and " $-$ " in Figure 3. Figure 3(c) shows the binary edge pixels overlaid on the flow map.

Secondly, the optical flow gradient is converted into probability. The relation between the magnitude of the gradient and the probability is not linear because larger depth discontinuity does not mean proportionally higher likelihood of the edge pixels to be depth boundary. That is why we use a logistic function g given below to map the gradient value to $[0, 1]$:

$$g(x) = \frac{1}{1 + e^{-\beta_1(x-\beta_2)}} \tag{1}$$

To determine $\beta_1$and β_2, we select 200 edge pixels at regular intervals on the boundary and inside the objects and calculate the optical flow gradients for these edge pixels. This will form a training set where the gradient values for the boundary pixels are mapped to 1 and for the internal edge pixels to 0. Using logistic regression, we fit the logistic function to this data as shown in Figure 4. $\beta_1$and β_2 are found to be 4.519 and 0.4313 respectively. All our experiments were performed with these values.

C. Probability to be contact boundary

An edge pixel at a contact boundary contains the pixels with color distribution close to the known color distribution of the surface on only one of its two sides. For any binary edge pixel, the pixels from its two sides are gathered using the appropriately oriented disc filter to form the 3D color histograms of its two sides. The χ^2 distances between the computed color distribution with that of the surface are represented by d_+ and d_-. Using these distances, the probability of the edge pixel to be at a contact boundary is $|g(d_+) - g(d_-)|$ where g is a logistic function given in equation (1). The parameters of the logistic function is determined empirically to be $\beta_1 = 8$ and $\beta_2 = 0.3$. The color histogram of the surface is built using an image of the surface alone.

The illumination invariant color representation (c_1, c_2, c_3), described in [7], is used to represent color, and the three color axes are divided into 10 equally spaced bins to form $10 \times 10 \times 10$ color histogram.

V. Fixation Strategy

The goal of the fixation strategy is to select points inside objects so that the fixation-based segmentation method [12] takes those points and generates the closed boundaries enclosing those points. To select the points, we first pick the edge pixels in the probabilistic boundary edge map with the boundary probability greater than 0.5, and assume that they lie on the boundary of some object in the image. We represent this subset of boundary edge pixels by I_B as given below:

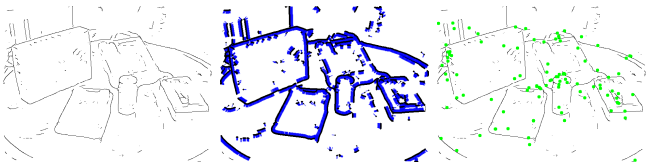
$$I_B(x,y) = \begin{cases} 1 & \text{if } P_B(x,y) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Additionally, we estimate the border ownership (object side) of the edge pixels in I_B. A boundary edge pixel in I_B can either be a depth or a contact boundary. For a pixel at depth discontinuity, the object side is the one that is closer to the camera which means will have bigger mean optic flow value. For a pixel at contact boundary, the object side is the one with the color distribution different than the known color distribution of the surface. We compute the difference between the properties of the two sides of an edge pixel using the appropriately oriented disc filter from figure 3 and assign $+1$ and -1 as their border ownership depending upon whether the positive or the negative side of the filter is found to be object side. The object side (border ownership) information is calculated as:

$$O_B(x,y) = \begin{cases} +1 & \text{if } d_+ < d_- \ \vee \ ||\vec{V_+}|| > ||\vec{V_-}|| \\ -1 & \text{if } d_- < d_+ \ \vee \ ||\vec{V_-}|| > ||\vec{V_+}|| \\ 0 & \text{if } I_B(x,y) = 0 \end{cases}$$

See figure 5a and 5b for an example of I_B and O_B respectively for the table top scenario shown in figure 2.

Using O_B, we could select the points inside objects by just moving a fixed distance toward the "object" side in the normal direction at all boundary edge pixels in I_B. But this would result in as many points as the number of edge pixels, causing redundancy. Many points will lie inside the same object meaning a large number of duplicate segmentation. To reduce redundancy, we break the contours of I_B into fragments interrupted by either an end point or a junction. Instead of an edge pixel, we now select a point for each edge fragment by moving again towards the "object side" in the normal direction at the middle of the fragment. The point is selected at a fixed distance of 20 px from the edge pixel. For instance, see Figure 5cfor all the points selected in this case.



(a) (b) (c)

Figure 5: (a) The known boundary edge map I_B. (b) Stubs extended towards the object side calculated using the border ownership O_B of the pixels in I_B. (c) The set of points selected on the object side.

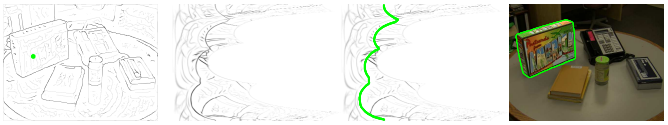
This leads to a more efficient fixation strategy, but there is still some redundancy left. In fact, the object will roughly get as many fixation points selected inside it as its sides. So, once the segmentation is carried out for all the fixation points, the output has some duplicate instances. Thus, we need a post-segmentation step that will sift through the regions to extract the regions corresponding to the objects in the scene. This step is described in section VII.

VI. Fixation Based Segmentation

For each selected point found in section V, the fixation-based segmentation approach [12] finds the closed boundary around the given point by combining the edge pixels in the probabilistic boundary edge. The segmentation for each point has two intermediate steps: first, the probabilistic boundary edge map P_B is converted from the Cartesian to the Polar space using the given point as the pole for the conversion, in order to achieve scale invariance. Following this, a binary segmentation of the polar edge map generates the optimal path through the polar edge map such that when the curve is mapped back to the original Cartesian space, it encloses the point. The two-step segmentation process is repeated for all fixation points found in section V using the same P_B. Figure 6 shows an example of the entire segmentation starting from the probabilistic boundary edge map and the point in (a) to the final segmentation in (d). The source code for the segmentation step is available at the URL:http://www.umiacs.umd.edu/~mishraka/code.html.

VII. Selecting Closed Contours Corresponding to Objects

We have as many closed contours as the number points selected by the fixation strategy. Since the selection of points depends on the contour fragments in I_B , the fragments that are part of the same object boundary generate multiple



(a) (b) (c) (d)

Figure 6: Using the given point (a dot) as the pole, the probabilistic boundary edge map, shown in (a), is transformed to the polar space as shown in (b). The optimal path through the polar space shown in (c) is transferred back to the Cartesian space to find the region, shown in (d), containing that point.

points lying inside the same object; these points give rise to duplicate closed contours. Also, sometimes due to error in the estimation of border ownership of the edge fragment, the corresponding point lies outside of any object in the image which will lead to a closed contour that does not correspond to an object. So, we need a process that sifts through all the closed contours to pick only the ones that uniquely correspond to the objects in the scene. We require a method to differentiate between any two closed contours which will be described in section VII-A. Following this, we describe our minimum set cover formulation to select the subset of closed contours that correspond to the objects in section 6.2.

Notation: I_C^i is the binary mask representing the i^{th} closed contour whose interior is represented by another binary mask I_R^i. If I is a 2D matrix with binary entries, X_I is the set of 2D coordinates of the non-zero pixels in I.

A. Coverage of a Closed Contour

We define the coverage of a closed contour such that high coverage means the closed contour is more likely to correspond to the boundary of an object in the scene. A closed boundary of an object is composed of the edge pixels in I_B such that the border ownership of these edge pixels points to the interior of the contour (see Figure 7(a) for an example of an object). The opposite of this is the closed boundary of a hole wherein the object side of all the boundary edge pixels lie outside of the hole (see Figure.7(b) for an example of a hole). So, the coverage of a closed contour is defined as:

$$Coverage(I_C, I_R) = \frac{1}{|X_{I_C}|} \sum_{x \in X_{I_C}} I_B(x)\Pi(x) \quad (2)$$

$$\Pi(x) = \begin{cases} +1 & \text{if } I_R(x + \lambda O_B(x) u_\perp) \neq 0 \\ -1 & \text{otherwise} \end{cases}$$

$$u_\perp = \begin{bmatrix} cos(\theta - \frac{\pi}{2}) \\ sin(\theta - \frac{\pi}{2}) \end{bmatrix}$$

where x is a 2D coordinate, θ is the orientation of the tangential direction at the edge pixel x and λ is the distance from the selected point on the "object" side to the edge pixel. In our experiment, we keep λ to be 5 px.

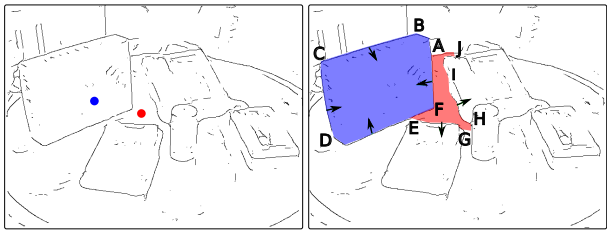


Figure 7: Left: A probabilistic boundary edge map with two fixation points: one inside an object, other in a hole. Right: The segmentation results for the fixations are shown along with the border ownership (shown by arrows) for the pixels along the region boundaries. The arrow showing border ownership is pointing inward for the closed contour corresponding to the object and outward for the closed contour corresponding to the hole.

B. Selecting Closed Contours Corresponding to Objects

With the definition of the coverage of a closed contour given above, simply looking at the sign of the coverage value differentiates between the boundary of an object and of a non-object. A closed contour outside of any object traces the edge pixels in I_B with "object" side lying outside of the contour and thus has a coverage that is negative or close to zero if positive. To handle the duplicate closed contours of the same object, we can pick the one resulting in the highest coverage with the object boundary. We formulate the process of selecting the closed boundaries of objects from the set of all closed contours as a type of a min-cover problem, whose standard definition is:

Definition 1: Given an Universal set U and a set of subsets S, find $S^{'} \subseteq S$ such that $S^{'}$ contains all the elements of U and $|S^{'}|$ is minimum.

In this case, X_{I_B} is the universal set U. $\{X_{I_C^i}\}_{i=1}^n$ is the set of subsets since $X_{I_C^i}$, the pixels along each closed contour, contains a subset of pixels in X_{I_B}. n is the total number of closed contours. Our objective is to find the minimum number of closed contours that together trace almost all the pixels in I_B. Since the minimum set cover problem is NP-Complete, we propose a greedy solution. The pseudo-code is given in Algorithm 1. The selected closed contours at the end of the process are the boundaries of the different objects in the scene.

The greedy solution works iteratively. It starts with computing coverage of all closed contours and then selects the best contour in each iteration. At the end of the iteration, the universal set is updated by eliminating all the pixels traced by the current best contour. After updating, the coverage of the remaining contours are recomputed for the next iteration. The selection process repeats until the "best" closed contour in the current iteration has a coverage below a certain threshold.

An important step in the proposed greedy solution is the the update of the remaining closed contours at the end of each iteration. To handle the case of occluding contours, when one of them is selected as the best contour in an iteration, the remaining closed contours are updated such that the pixels on the shared boundary do not affect their coverage as those pixels have already been traced by the current best contour.

Algorithm 1 Selecting objects, $t_{coverage} = 0.5$, $t_o = 0.05$

Input:
I_B ▷ edge pixels predicted to be on object boundaries
O_B ▷ object side information
$S_{in} = \{I_R^i, I_C^i\}_{i=1}^n$ ▷ closed contours for all n fixations
Output:
$S_{out} = \{I_R^j, I_C^j\}_{j=1}^m$ ▷ final closed contours
Intermediate variables:
I_T^k ▷ all closed contours traced until iteration (k-1)
I_M^k ▷ accumulated region mask until iteration (k-1)

begin
Initialize $k \leftarrow 0$; $I_B^0 \leftarrow 0$; $I_A^0 \leftarrow 0$
while $|S_{in}| > 0$ **do**
  Compute coverage of all closed contours $\in S_{in}$;
  Let b be the index of the closed contour with maximum coverage;
  if $Coverage(I_R^b, I_C^b) < t_{coverage}$ **then**
    Exit the loop;
  else
    Move the closed contour from S_{in} to S_{out};
    $I_A^k \leftarrow I_A^k + I_R^b$;
    $X_{I_B} \leftarrow X_{I_B} - (X_{I_B} \cap X_{I_C^b})$;
    $X_{I_T^k} \leftarrow (X_{I_T^k} \cup X_{I_C^b})$;
    for $I_C^i \in S_{in}$ **do**
      overlap = $|X_{I_R^i} \cap X_{I_A}|/|X_{I_R^i}|$;
      if overlap $< t_o$ **then**
        $X_{I_C^i} \leftarrow X_{I_C^i} - (X_{I_C^i} \cap X_{I_T^k})$ ▷ to handle occlusion
      end if
    end for
    Increment k;
  end if
end while

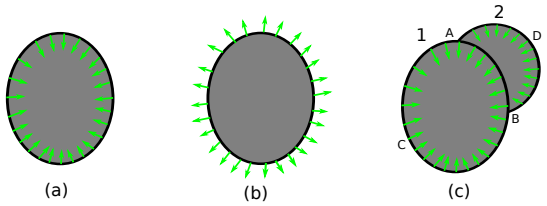


Figure 8: (a) and (b) are the closed contours likely to a moving object and a hole respectively. (b) A closed contour hole. (c) Two occluding closed contours. Note that the arrows indicate the "object" side of the boundary pixels.

Consider, for instance, the two occluding closed contours $ABCA$ and $ADBA$ in Figure 8(c). They share the pixels along the segment AB and the "object" side indicates it will contribute positively to the coverage of $ABCA$ and negatively to that of contour $ADBA$. After the selection of $ABCA$, in the next iteration, we would like to make $ADBA$ prominent because the boundary portion AB that was contributing negatively has already been traced by region $ABCA$. So, the new coverage of $ADBA$ will include the contribution from the remaining pixels in the segment ADB. But, this could have unintended consequence for overlapping duplicate contours where once the best contour for the object is selected, all the duplicate contours with a slight change will become important just as the occluding contour case. To avoid this situation, we check for the overlap of the inside of the remaining closed contours with that of already selected closed contours. Only the closed contours with no overlap are updated.

C. Temporal Persistence

The regions selected by the greedy solution may still contain spurious regions arising due to wrongly detected strong edges in the probabilistic boundary edge map. One simple way to get rid of the spurious regions without having to use high level knowledge is based on the observation that the noisy edges in the probabilistic boundary edge map is caused by the noise in the flow map as we have used motion cues to determine the object boundaries in our experiments. Since the effect of noise is not localized but changes its locations in the flow map over time, the wrongly detected boundary edge fragments will change too. This means, the spurious regions formed by these edge fragments would change as well. All of this suggests that if we repeat the entire process for frame 2 and 3 and match the selected regions with the regions obtained for the frames 1 and 2, and accept only the regions that occur in both cases, we end up discarding most of the spurious regions. The regions that persisted in time are more likely to be actual objects.

VIII. Experiments

We evaluate the performance of the proposed system for the scenario when the camera is moving and objects are static. There are a number of motion segmentation algorithms[14] to segmenting objects. But segmenting static objects using motion cues is rather uncommon and is a distinguishing factor for the proposed system. We choose to evaluate the efficiency and robustness of the system in segmenting these static objects.

Data

The input is a sequence of three consecutive frames captured using a camera moving roughly parallel to the objects in the scene, and an image of the surface (e.g table or floor) on which the objects are kept. The camera displacement between the consecutive frames is less than 15 pixels for the optic flow algorithm [3] to compute the optical flow map accurately. We generate a dataset of 45 such test sequences with 35 distinct objects of different shapes and sizes. (The ratio between the biggest and the smallest object sizes is ~ 10). Also, the shape of the objects in the dataset ranges from a spherical ball to an elongated pen to a cone. The appearance inside the objects varies from smooth (apple) to a very textured pattern (a box with a logo). The average number of objects present in each test sequence is 5. The maximum and minimum number of objects in a test sequence is 9 and 3 respectively. To check for the robustness of using the color information of the surface to locate contact boundary, we choose 6 different types of surfaces, namely a wooden table, a smooth table cloth, a textured table cloth, green grass, bricked floor. Finally, to make the dataset representative of the typical scenarios faced by an autonomous robot, we captured the dataset in both indoor and outdoor conditions. Since the segmentation process does not depend upon the actual appearance of the pixels but on the probabilistic boundary edge map, the illumination condition does not affect the accuracy in any significant manner.

	Precision	Recall
After TP	0.72 ± 0.21	0.85 ± 0.18
Before TP	0.42 ± 0.23	0.85 ± 0.19

Table I: Recall is the ratio between the number of objects segmented by our system and the total number of objects in the scene. Precision is the ratio between the number of segmented contours corresponding to any object and the total number of segmented contours. TP stands for temporal persistence.

Results

The performance is measured in terms how many of the objects correspond to one of the segmented contours (recall) and how many of segmented contours correspond to the actual objects (precision). To decide about the correspondence between a segmented closed contour and an object, we use the manually segmented binary mask of the object. If the overlap $\frac{R_1 \bigcap R_2}{R_1 \bigcup R_2}$ between the object mask R_1 and the interior of the closed contour R_2 is more than 0.9, they are declared to correspond to each other. The recall and precision values are averaged over all test sequences, and the resulting mean values are given in Table I. The first and second row of the table corresponds to the performance of the system after and before enforcing temporal persistence on the segmented contours for two pairs of consecutive image frames. The temporal persistence helps weed out closed contours arising out of noisy edge pixels in the probabilistic boundary edge map. It improves the precision of the system by almost 25%. Further improvement in precision can be achieved by using a recognition system which will match the contours with its knowledge base to reject any unknown contours.

An important performance metric is the recall in Table I which is about 85%. This means the proposed system will, on average, segment 17 out of 20 objects kept on a table given a three consecutive frames of the scene captured using a moving camera. For a robot carrying the segmentation system, repeating the segmentation process from different viewpoints around the scene might result in the segmentation all objects in at least one of the viewpoint.

To evaluate the efficiency of the fixation strategy, we calculate the percentage of all selected fixation points that actually lie inside an object and it is 85%. To measure the redundancy in the fixation strategy, we compute the average number of fixations lying inside each object which is 12.5. The final evaluation is about the robustness of predicting the border ownership (or object side) which is computed as the mean of the percentage of the pixels on any object boundary with correct object side prediction. It is 93.5%.

Computational complexity

The image size is 640×480. The timing analysis of the system is done using a quad-core 32-bit processor. 1) Cue computation step: our multithreaded C++ implementation of the Berkeley edge detector takes about $6s$ to compute the the edge map; [3] takes about $24s$ to compute the optical flow map. 2) The segmentation for a given fixation point takes about $2s$. 3) The selection process in the end and the probabilistic boundary edge detection process combined takes about $8s$.

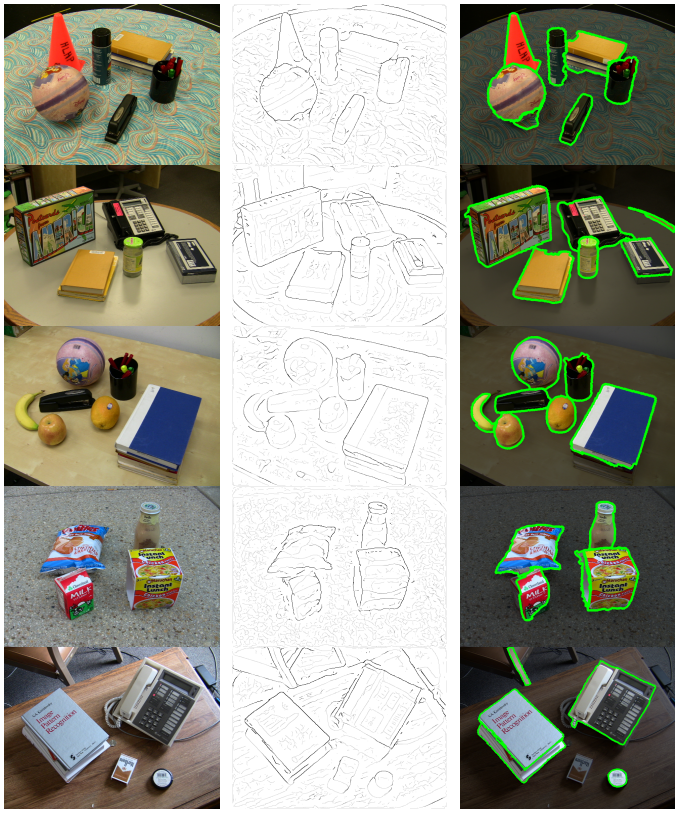


Figure 9: Left column: the first frame of three frame sequences. Middle column: the probabilistic boundary edge map. Right column: The output of the proposed segmentation system after enforcing temporal persistence.

Since the system is completely modular, we can improve the speed of each step dramatically. Instead of optic flow, we can use the new device from Microsoft, Kinect, to compute the disparity map which takes less than a second. Also, the segmentation step can be carried out for all the selected points in parallel as they do not depend on each other.

IX. Discussion & Conclusion: Toward Semantic Active Vision

We described a system that segments objects of any size or shape automatically. The system is based on the idea that segmentation should produce a closed contour surrounding the fixation point of attention, introduced in [12]. Two novel contributions of this work are a fixation strategy (basic attention mechanism) to select the points on the objects, and a strategy to select only the regions corresponding to the objects.

The input to the system is a minimum of three images of the scene. Any robot with a camera, which is capable of moving in its environment and acquiring images from different views, can use this system. In fact, the system is particularly suitable to robotic applications for the two main reasons: first, it is independent from any user parameter; second, it can easily incorporate more cues even non-visual cues. The second attribute is due to the modular structure of the system.

The cues are responsible only for generating the probabilistic boundary edge map; the segmentation step is immune to how the probabilistic boundary map is created. So, if a robotic system has, say, a depth map coming from a Laser sensor, it can be used to refine the boundary edge map. This provides the flexibility of using different sensory inputs without changing the basic algorithm.

Finally, there are ways to improve the proposed system. A robust recognition step can help eliminate spurious regions besides assigning the high-level semantics to the remaining regions. It should be noted that if the object is segmented prior to recognition, so the recognition process can proceed on the basis of attributes (properties). Although the idea of attribute based object recognition has existed for quite a while, it has never been implemented because of the lack of segmentation (recent work on attribute based object recognition[13] was applied to a database of faces and segmentation was not an issue). The ability to estimate object attributes leads naturally to Semantic Active Vision, where robots can find objects.

The second important addition can be an efficient fixation system that, instead of outputting all possible fixation points, would sequentially suggest the next fixation point such that it reduces the computational cost of segmenting the same object multiple times. It is also possible to integrate the proposed system with other systems[2], where the areas detected by learning could be used to provide the initial fixation points. Lastly, an interesting addition could be to incorporate shape cues in addition to color, texture and motion/stereo cues to segment the objects even in the absence of sufficiently strong visual cues (e.g. in single images).

REFERENCES

[1] C.C. Williams A. Hollingworth and J.M. Henderson. To see and remember: Visually specific information is retained in memory from previously attended objects in natural scenes. *Psychonomic Bulletin and Review*, 8:761–768, 2001.

[2] J. Kosecka B. Micusik. Semantic segmentation of street scenes by superpixel co-occurrence and 3d geometry. In *IEEE Workshop on Video-Oriented Object and Event Classification, ICCV*, 2009.

[3] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. pages 25–36. Springer, 2004.

[4] Edward Craft, Hartmut Schütze, Ernst Niebur, and Rüdiger von der Heydt. A neural model of figure-ground organization. *Journal of Neurophysiology*, 6(97):4310–4326, 2007.

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[6] C.C. Fowlkes, David R. M., and J. Malik. Local figure/ground cues are valid for natural images. *JV*, 7(8):1–9, 2007.

[7] T. Gevers and A. Smeulders. Color based object recognition. *Pattern Recognition*, 32:453–464, 1997.

[8] J. M. Henderson. Human gaze control during real-world scene perception. *Trends in Cognitive Sciences*, 7:498–504, 2003.

[9] J. Eriksson L. Holm and L. Andersson. Looking as if you know: Systematic object inspection precedes object recognition. *Journal of Vision*, 8(4):1–7, 2008.

[10] C. Koch L. Itti and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *T-PAMI*, 20:1254–1259, 1998.

[11] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cues. *T-PAMI*, 26(5):530–549, May 2004.

[12] A. Mishra, Y. Aloimonos, and L. F. Cheong. Active segmentation with fixation. In *ICCV*, 2009.

[13] P.N. Belhmeur S. K. Nayar N. Kumar, A.C. Berg. Attribute and simile classifiers for face verification. In *ICCV*, 2009.

[14] D. Sinclair. Motion segmentation and local structure. *ICCV*, 93:366–373, 1993.

[15] A.M. Treisman and G. Gelade. A feature-integration theory of attention. *Cognitive Psychological*, 12:97–136, 1980.

[16] M. Spain W. Einhäuser and P. Perona. Objects predict fixations better than early saliency. *Journal of Vision*, 8(14):1–26, 2008.

[17] H. Zhou, H.S. Friedman, and R. von der Heydt. Coding of border ownership in monkey visual cortex. *The Journal of Neuroscience*, 20:6594–6611, September, 2000.