CHAPTER 3

First contact: tapping into the world

The Bursar shrugged.
"This pot," he said, peering closely, "is actually quite an old Ming vase."
He waited expectantly.
"Why's it called Ming?" said the Archchancellor, on cue.
*The Bursar tapped the pot. It went *ming*.* (Pratchett, 1990)

Vision and action are intertwined at a very basic level in humans (Iacoboni et al., 1999). Researchers in machine vision have found many pragmatic reasons for integrating sensing tightly with motor control on an active vision head. This chapter extends this logic to simple object manipulation, showing how a simple tapping/poking behavior can help figure/ground separation. Poking an object makes it move, and motion is a powerful cue for visual segmentation. Poking itself does not require that an object be accurately segmented, since it can be performed simply as a sweep of the arm through a general neighborhood The periods immediately before and after the moment of impact turn out to be particularly informative, and give visual evidence for the boundary of the object that is well suited to segmentation using graph cuts. Of course, an experienced adult can interpret visual scenes perfectly well without acting upon them, and ideally our robots should do the same. Poking is proposed as a fallback segmentation method when all else fails, and a developmental opportunity for training up a contact-free object segmentation module. This topic is elaborated in later chapters.

One contribution of this work is to clearly formulate a new object segmentation challenge not yet attended to in the machine vision literature, but which will become increasingly important in robotics. That challenge is: how can segmentation best be performed if exploratory manipulation is permissible? Another contribution is to demonstrate an approach that makes use of the most rudimentary manipulation possible to achieve segmentation, establishing a qualitative lower bound on what is possible, and showing that the benefits are non-trivial. Of course, segmentation is just the start of what can ultimately be learned through manipulation – but it is a good start, and given the complexity of dextrous manipulation, it is encouraging that even very simple motor control can lead to worthwhile results.

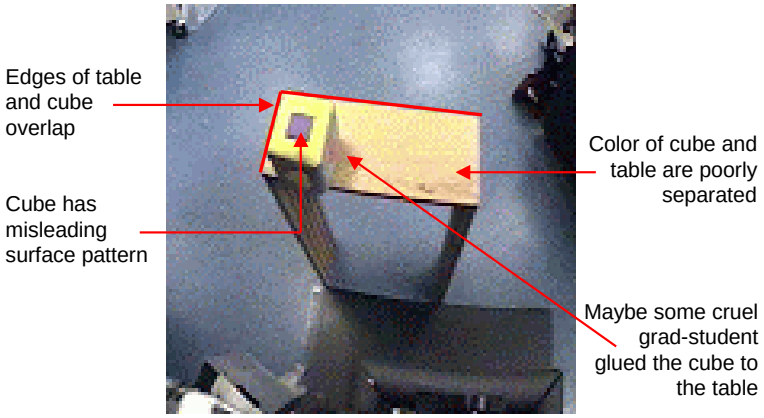


Figure 3-1: A cube on a table, to illustrate some problems in segmentation. The edges of the table and cube happen to be aligned, the colors of the cube and table are not well separated, and the cube has a potentially confusing surface pattern. And even if dense 3D information were available, there is no way to be really be sure the cube is an independently manipulable entity, and not connected to the table below it.

3.1 Active vision

A vision system is said to be *active* if it is embedded within a platform that can change its physical configuration to improve perceptual performance. For example, a robot's cameras might servo a rapidly moving target in order to stabilize the image and keep the target in view. The term is also used when processing is adapted to the current situation. Historically, a number of logically distinct ideas are often associated with active vision. The first is that vision should be approached within the the context of an overall task or purpose (Aloimonos et al., 1987). Another idea is that if an observer can engage in controlled motion, it can integrate visual data from moment to moment to solve problems that are ill-posed statically. Well-chosen motion can simplify the computation required for widely studied vision problems, such as stereo matching (Bajcsy, 1988; Ballard, 1991). These interwoven ideas about active vision are teased apart in Tarr and Black (1994).

This work seeks to add two new threads to the mix. The first is that although active vision is often equated with moving cameras, the entire body of a robot could potentially be recruited to cooperate with the vision system. In this chapter, movement of the robot's arm is recruited to augment its visual system, and in particular to solve the figure/ground separation problem by active means. The second thread is that active systems have the potential to perform experiments that get close to accessing physical ground truth, and so potentially admit of a perceptual system that develops autonomously. The poking behavior gives the robot access to high quality training data that can be used to support object localization, segmentation, and recognition. It also provides a simpler "fall-back" mechanism for segmentation, so that the robot is not entirely at the mercy of the failures of these higher-level competences.

3.2 Manipulation-driven vision

There are at least three clear situations in which it is useful for manipulation to guide vision, rather than the other way around, as is typical in robotics :-

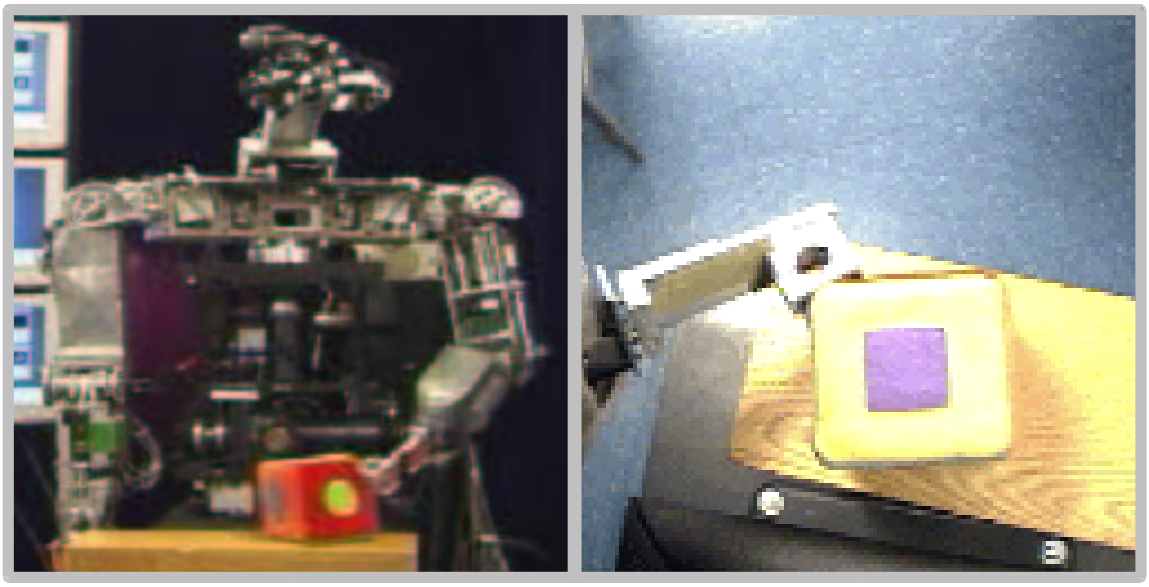


Figure 3-2: Resolving visual ambiguity by active means. The robot (left) reaches towards an object in its environment while fixating it with a camera. The robot's view is shown on the right. The boundary between the cube and the table it is sitting on is clear to human eyes, but too subtle to be reliably segmented by current automatic methods. But once the robot arm comes in contact with the object, it can be easily segmented from the background using the motion due to the impact.

- ▷ **Experimentation:** Making progress when perception is ambiguous.
- ▷ **Correction:** Recovering when perception is misleading.
- ▷ **Development:** Bootstrapping when perception is dumb.

Experimentation

Rather than simply failing in visually ambiguous situations, an active robotic platform has the potential to perform experiments on its environment that resolve the ambiguity. Consider the example in Figure 3-1. Visually, there are difficulties with segmenting this scene, due to some unfortunate coincidences in the alignment and color of the cube and the table. Rather than simply giving up on such situations, we could instead simply dispatch the robot's arm to the ambiguous region and poke around a bit. Several methods for characterizing the shape of an object through tactile information have been developed, such as shape from probing (Cole and Yap, 1987; Paulos, 1999) or pushing (Jia and Erdmann, 1998; Moll and Erdmann, 2001). The work in this chapter exploits the fact that the *visual* feedback generated when the robot moves an object is highly informative, even when the motion is short and poorly controlled, or even accidental. The vocabulary used to describe the robot's motion – "tapping" or poking" as opposed to "probing" – is deliberately chosen to convey the idea of a quick jab (to evoke visual data) instead of an extended grope (for tactile data). Although tactile and visual information could usefully be combined, no tactile or proprioceptive information is assumed in this chapter – not even to determine whether the robot is in contact with an object.

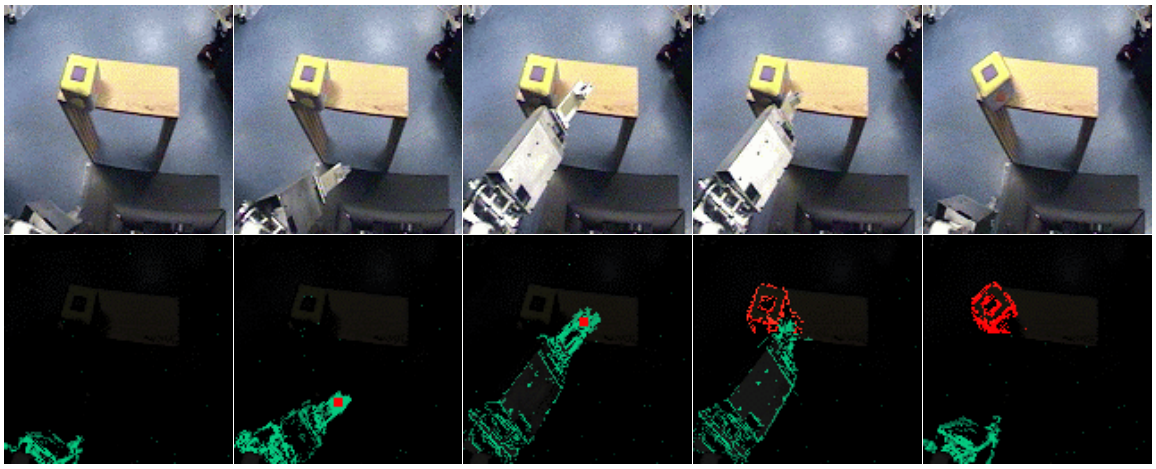


Figure 3-3: The upper sequence shows an arm extending into a workspace, tapping an object, and retracting. This is an exploratory mechanism for finding the boundaries of objects, and essentially requires the arm to collide with objects under normal operation, rather than as an occasional accident. The lower sequence shows the shape identified from the tap using simple image differencing and flipper tracking.

Recovery

How a robot should grasp an object depends on its size and shape. Such parameters can be estimated visually, but this is bound to be fallible – particularly for unrecognized, unfamiliar objects. Failure may result in a clumsy grasp or glancing blow against the object. If the robot does not learn something from the encounter, then it will be apt to repeat the same mistake again and again. As already foreshadowed, this chapter shows how to recover information about an object's extent by poking it, either accidentally or deliberately. This opens the door to extracting information from failed actions such as a glancing blow to an object during an attempt at manipulation, giving the robot the data it needs to do better next time.

Development

It would be cumbersome to always have to poke around to segment an object each time it comes into view. But the cleanly segmented views of objects generated by poking are exactly what is needed to train up an object recognition system, which in turn makes contact-free segmentation possible. So the kind of active segmentation proposed here can serve as an online teacher for passive segmentation techniques. Analogously, while an experienced adult can interpret visual scenes perfectly well without acting upon them, linking action and perception seems crucial to the developmental process that leads to that competence (Fitzpatrick and Metta, 2002).

3.3 Implementing active segmentation

Figure 3-3 shows frames from an early experiment where a robot arm was driven through a preprogrammed open-loop trajectory that swept through an area above a table, watched by a fixed camera. In the course of this trajectory, the arm came into contact with a cube sitting on a table,

and disturbs it. The challenge is to identify and isolate this disturbance, and to use it to segment the cube, or whatever object the arm might encounter. The video stream from this and similar experiments were used to develop a baseline implementation of active segmentation and to clarify the requirements in terms of processing and behavior. The remainder of this chapter then fits this work into an actual behaving robot.

A reasonable way to segment the object would be to track the motion of the arm as it swings outwards, and to look for any motion that is not plausibly associated with the arm itself, but nevertheless appears to be physically adjacent to it. For this simple motivating example, the end-effector (or "flipper") is localized as the arm sweeps rapidly outwards using the heuristic that it lies at the highest point of the region of optic flow swept out by the arm in the image. The reaching trajectory of the robot relative to the camera orientation is controlled so that this is true. The sweeping motion is also made rather gentle, to minimize the opportunity for the motion of the arm itself to cause confusion. The motion of the flipper is bounded around the endpoint whose location we know from tracking during the extension phase, and can be subtracted easily. Flow not connected to the end-effector can be ignored as a distractor.

The sequence shown in Figure 3-3 is about the simplest case possible for segmenting the motion of the object. In practice, we would rather have less constraints on the motion of the arm, so we can approach the object from any convenient direction. It is also desirable to be able to explore areas where an object is likely to be, rather than simply sweeping blindly. But if objects are not already segmented, where can a target for poking come from? This is in fact very straightforward. As described in Section 2, Cog has an attentional system that allows it to locate and track salient visual stimuli. This is based entirely on low-level features such as color, motion, and binocular disparity that are well defined on small patches of the image, as opposed to features such as shape, size, and pose which only make sense on well-segmented objects. If Cog's attention system locates a patch of the image that seems reachable (based on disparity and overall robot pose) that is all it needs to know to reach toward it and attempt to poke it so that it can determine the physical extent of the object to which that patch belongs. A human can easily encourage this behavior by bringing an object close to the robot, moving it until the robot fixates it, and then leaving it down on the table. The robot will track the object down to the table (without the need or the ability to actually segment it), observe that it can be reached, and poke it. Still, there are many things that could go wrong. Here is a list of the many potential pitfalls that could result in an inaccurate segmentation :-

- ▷ Object motion may not be particularly visible – if it is not highly textured, then there may be regions on its projection where optic flow is low or non-existent.
- ▷ The motion of the manipulator might be incorrectly separated from the motion of an object it comes in contact with.
- ▷ Unrelated motion in the background might be mistaken for movement of the manipulator or an impacted object.
- ▷ The manipulator might fail to come into contact with any object.
- ▷ The manipulator might obscure the robot's view of an object as it hits it.
- ▷ The object might not move rigidly, or might move too little or too much to be processed well.
- ▷ Motion of the camera might be mistaken for movement of the manipulator or an impacted object.

The first three points are dealt with by using a segmentation method based on graph cuts that allows diverse local evidence to be factored into making a good approximation to a globally optimal segmentation. Optic flow in textured regions provides evidence of movement, lack of optic flow in textured regions suggests lack of motion, comparing motion before and after the moment of impact

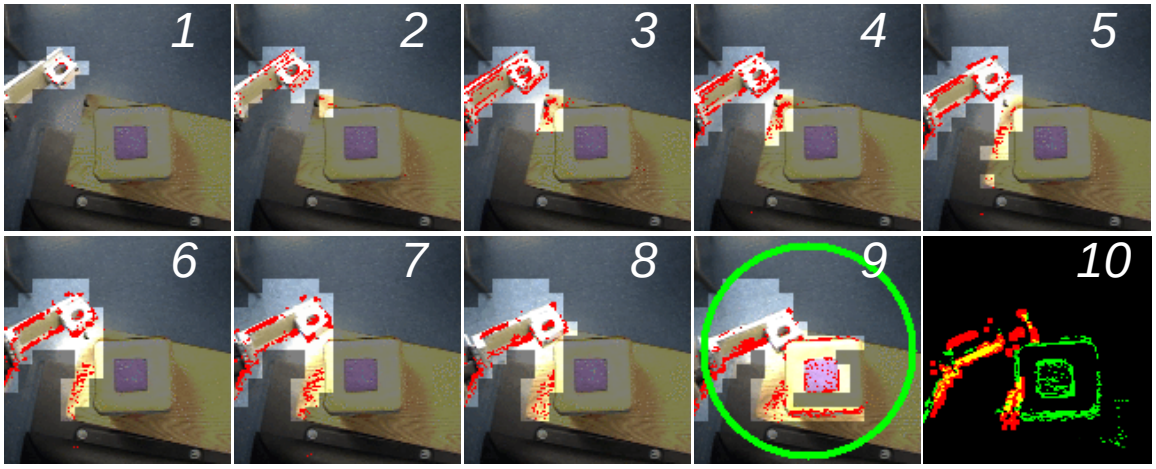


Figure 3-4: The moment of (ground) truth – detecting the point of impact between the robot's arm and an object. As the arm swings in, its motion is tracked frame by frame and aggregated within relatively low-resolution bins (highlighted squares). When an implausibly large spread in motion is detected across these bins, higher resolution processing is activated and segmentation begins.

gives evidence for what part of the image is the manipulator. The next three points are dealt with by careful engineering of the kinds of motion the robot makes. The final point is dealt with simply by keeping the camera fixated during poking. There is no real advantage to moving it, and segmentation based on motion viewed by a fixed camera is well understood and has been explored exhaustively (see for example Ross (2000); Stauffer (1999)).

3.4 First contact

If the object is to be segmented based on motion, we need to differentiate its motion from any other sources in the scene – particularly that of the robot itself. A high-quality opportunity to do this arises right at the moment of first contact between the robot and the object. This contact could be detected from tactile information, but it is also straightforward to detect visually, which is the method described here. The advantage of using visual information is that the same techniques can be applied to contact events about which the robot has no privileged knowledge, such as a human hand poking an object (see Section 6).

For real-time operation, the moment of contact is first detected using low-resolution processing, and then the images before and after the contact are subjected to more detailed (and slower) analysis as described in the following section. Figure 3-4 shows a visualization of the procedure used. When the robot is attempting to poke a target, it suppresses camera movement and keeps the target fixated for maximum sensitivity to motion. A simple Gaussian model is maintained for the (R, G, B) color values of each pixel, based on their value over the last ten frames (one third of a second) received. Significant changes in pixel values from frame to frame are detected and flagged as possible motion. As the arm moves in the scene, its motion is tracked and discounted, along with its shadow and any background motion. Any area that the arm moves through is marked as "clear" of the object for a brief period – but not permanently since the arm may cross over the object before swinging back to strike it. An impact event is detected through a signature explosion of movement that is connected with the arm but spread across a much wider distance than the arm could reasonably have moved

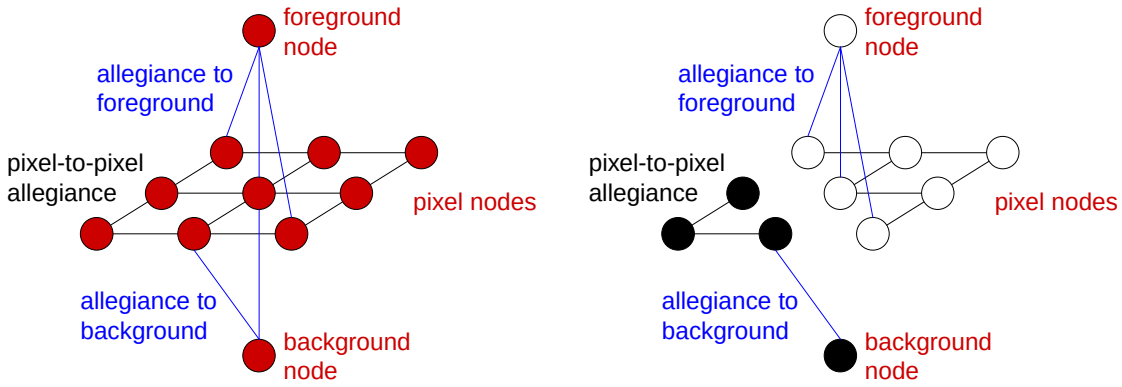


Figure 3-5: For a two-label problem on a 2D image, the input to a minimum-cut algorithm is typically as shown on the left. There is a node for each pixel, and two special nodes corresponding to the labels (foreground/background). Visual evidence is encoded on edges between the nodes. The output of the algorithm is shown on the right. The graph is cut into two disjoint sets, each containing exactly one of the special nodes, such that the total cost of the edges cut is (approximately) minimized.

in the time available. Since the object is stationary before the robot pokes it, we can expect the variance of the Gaussians associated with the individual pixel models to be low. Hence they will be very sensitive to the pixel value changes associated with the sudden motion of the object. Once the impact is detected, we can drop briefly out of real-time operation for a few seconds and perform the detailed analysis required to actually cleanly segment the object based on the apparent motion.

3.5 Figure/ground separation

Once the moment of contact is known, the motion visible before contact can be compared with the motion visible after contact to isolate the motion due to the object. Since we observe pixel variation rather than true motion, we can also factor in how we expect them to relate – for example, a highly textured region with no observed change over time can be confidently declared to be stationary, while a homogeneous region may well be in motion even if there is little observed change. In general, the information we have is sparse in the image and can be framed as probabilities that a pixel belongs to the foreground (the object) or the background (everything else). Let us first look at a simpler version of this problem, where for those pixels that we do have foreground/background information, we are completely confident in our assignments.

Suppose we have some information about which pixels in an image $I(x, y)$ are part of the foreground and which are part of the background. We can represent this as:

$$A(x,y) \quad = \quad \begin{cases} -1, & I(x,y) \text{ is background} \\ 0, & I(x,y) \text{ is unassigned} \\ 1, & I(x,y) \text{ is foreground} \end{cases}$$

We now wish to assign *every* pixel in the image to foreground or background as best we can with the sparse evidence we have. One approach would be to create a cost function to evaluate potential segmentations, and choose the segmentation with minimum cost. If we are willing to accept constraints on the kind of cost function we can use, then there is a family of maximum-flow/minimum-cut al-

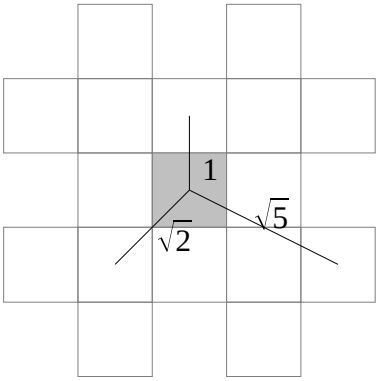


Figure 3-6: The segmentation algorithm is sensitive to the length of the perimeters around foreground regions. It is important that the local pixel connectivity not be so sparse as to introduces artifacts into that perimeter. For example, suppose we just used 4-connected regions. The cost of a zig-zag approximation to a diagonal edge would be $\sqrt{2} = 1.41$ times what it ought to be. 8-connected regions are better, but still distort the perimeter cost significantly, up to a factor of $\frac{1+\sqrt{2}}{\sqrt{5}} = 1.08$. The neighborhood shown here, which is 8-connected plus "knight moves", introduces a distortion of at most $\frac{1+\sqrt{5}}{\sqrt{10}} = 1.02$. Further increases in neighborhood size increases computation time without bringing significant benefit.

gorithms that can provide good approximate solutions to this problem (Boykov and Kolmogorov, 2001). To apply them, we need to translate our problem into the form of a graph, as shown in Figure 3-5. Each pixel maps to a node in the graph, and is connected by edges to the nodes that represent neighboring pixels. There are two special nodes corresponding to the labels we wish to assign to each pixel (foreground or background). The problem the minimum-cut algorithms can solve is how to split this graph into two disjoint parts, with the foreground node in one and the background node in the other, such that the total cost of the edges broken to achieve this split is minimized. So our goal should be to assign costs to edges such that a minimum cut of the graph will correspond to a sensible segmentation.

Let $N(x, y)$ be the node corresponding to pixel $I(x, y)$. Let N_{+1} be the node representing the foreground, and N_{-1} be the node representing the background. If we are completely confident in our classification of pixel $I(x, y)$ into background or foreground, we may encode this knowledge by assigning infinite cost to the edge from $N(x, y)$ to $N_{A(x,y)}$ and zero cost to the edge from $N(x, y)$ to $N_{-A(x,y)}$.

$$\begin{aligned} \mathcal{C}(N(x,y), N_{+1}) &= \begin{cases} \infty, & A(x,y) = 1 \\ 0, & \text{otherwise} \end{cases} \\ \mathcal{C}(N(x,y), N_{-1}) &= \begin{cases} \infty, & A(x,y) = -1 \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

This will force the minimum-cut algorithm to assign that pixel to the desired layer. In practice, the visual information will be more ambiguous, and these weights should be correspondingly "softer".

Costs also need to be assigned to edges between pixel nodes. Suppose we expect foreground information to be available most reliably around the edges of the object, as is in fact the case for motion data. Then a reasonable goal would be to use the minimum cut to minimize the total perimeter

length of segmented regions, and so merge partial boundary segments into their bounding region. To do this, we could simply assign the actual 2D Euclidean distance between the pixels as the cost. This is not quite sufficient if our edge information is noisy, because it permits almost "zero-area" cuts around individual isolated foreground pixels. We need to place an extra cost on cutting around a foreground pixel so that it becomes preferable to group near-neighbors and start generating regions of non-zero area. For this example, we simply double the cost of cutting edges that are connected to pixels known to be foreground or background.

$$\begin{aligned} \mathcal{C}(N(x_0, y_0), N(x_1, y_1)) &= \begin{cases} D, & A(x_0, y_0) = 0, \\ & A(x_1, y_1) = 0 \\ 2D, & otherwise \end{cases} \\ \text{where } D &= \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2} \end{aligned}$$

Edges are only placed between neighboring pixel nodes, to prevent an explosion in connectivity. A neighborhood is defined as shown in Figure 3-6.

Figure 3-7 shows examples of minimum-cuts in operation. The first image (top left) has two (noisy) lines of known foreground pixels, of length w. The minimum cut must place these pixels inside a foreground region. If the regions are disjoint, the total perimeter will be at least $4w$. If the the lines are instead placed inside the same region, the cost could be as little as $2w + 2h$ where h is the distance between the two lines, which is less than w. The figure shows that this is in fact the solution the minimum-cut algorithm finds. The next two examples show what this minimum perimeter criterion will group and what it will leave separate. The fourth example shows that by introducing known background pixels, the segmentation can change radically. The patch of background increases the perimeter cost of the previous segmentation by poking a hole in it that is large enough to tip the balance in favor of individual rather than merged regions. This basic formulation can be extended without difficulty to natural data, where foreground/background assignments are soft.

3.6 Before and after

The previous section showed that if there is some evidence available about which pixels are part of the foreground and which are part of the background, it is straightforward to induce a plausible segmentation across the entire image. Figure 3-8 shows an example of how the necessary visual evidence is derived in practice. The statistical significance of changes in pixel values (the "apparent motion") is measured in the frames directly following the contact event, using the continuously updated Gaussian models. The measurements are combined over two frames to avoid situations where the contact event occurs just before the first frame, early enough to generate enough motion for the contact event to be detected but late enough not to generate enough motion for a successful segmentation. The frames are aligned by searching for the translation that best matches the apparent motion in the two frames (rotation can be neglected for these very short intervals). A similar measurement of apparent motion from immediately before the contact event is also aligned, and is used to partially mask out motion belonging to the robot arm, its shadow, and unrelated movement in the environment. The remaining motion is passed to the segmentation algorithm by giving pixels a strong "foreground" allegiance (high cost on edge to special foreground node). Importantly, the motion mask from before contact is also passed to the algorithm as a strong "background" allegiance (high cost on edge to background node). This prevents the segmented region from growing to include the arm without requiring the masking procedure to be precise. The maximum-flow

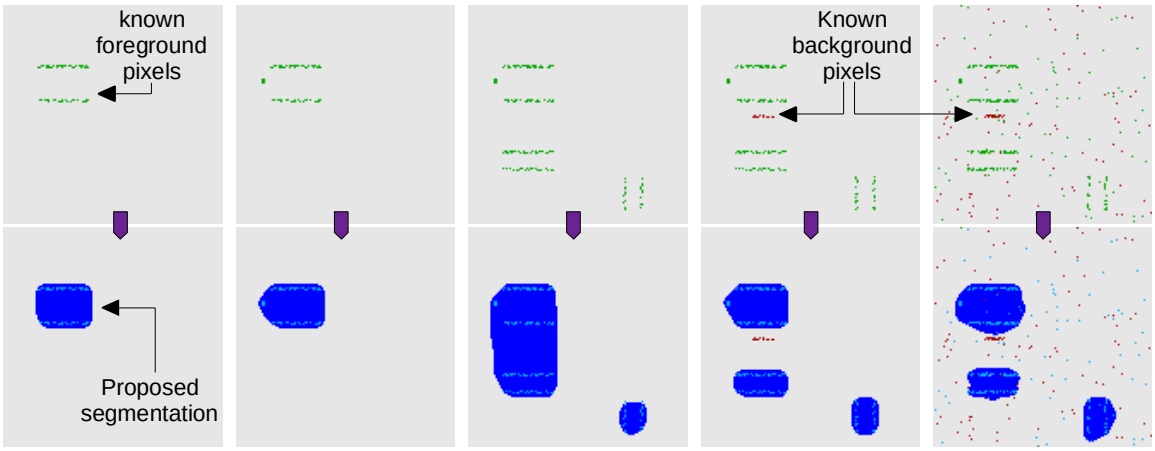


Figure 3-7: Some simple segmentation examples. Input images are shown on the upper row, output is shown as filled regions on the lower row. In the first three cases, the border of the image is set to be background, and the dark pixels are foreground. In the fourth case, a small extra patch of pixels known to be in the background is added, which splits the large segmented region from the previous case in two. The final case shows that the algorithm is robust to noise, where 1% of the pixels are assigned to foreground or background at random. This is in fact a very harsh kind of noise, since we have assumed complete certainty in the data.

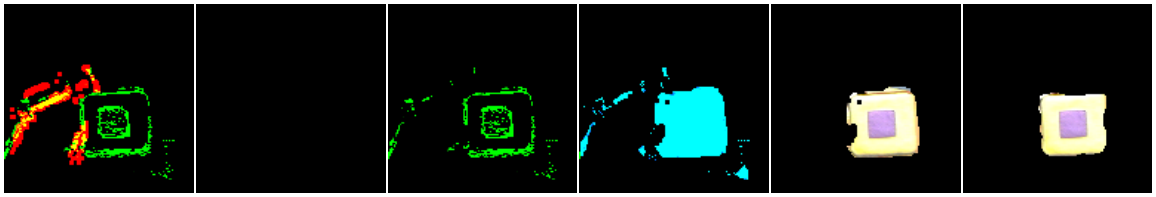
implementation used is due to (Boykov and Kolmogorov, 2001).

Perimeter-minimization seems particularly appropriate for the kind of motion data available, since for textureless objects against a textureless background (the worst case for motion segmentation) motion is only evident around the edges of the object, with a magnitude that increases with the angle that edge makes to the direction of motion. A textured, cluttered background could only make life simpler, since it makes it easier to confidently assert that background regions are in fact not moving.

3.7 Experimental results

How well does active segmentation work? The segmentation in Figure 3-8 is of the object shown in the introduction (Figure 3-2), a cube with a yellow exterior sitting on a yellow table. Active segmentation has a clear advantage in situations like this where the color and texture difference between object and background would be too small for conventional segmentation but is sufficient to generate apparent motion when the object is poked. Figure 3-11 shows poking from different directions. Figure 3-9 shows about 60 successive pokes of the cube, to give a sense of the kinds of errors that occur. Figures 3-10 and 3.7 shows results for particularly difficult situations. Figure 3-13 shows the area plotted against the second Hu moment (a measure of anisotropy) for a set of four objects that were poked repeatedly. The second Hu moment Φ_2 for a region R with centroid (x_0, y_0) and area μ_{00} is:

$$\begin{aligned} \Phi_2 &= (\nu_{20} - \nu_{02})^2 + 4\nu_{11}^2 \\ \nu_{pq} &= \frac{1}{\mu_{00}^2} \int \int_R (x - x_0)^p (y - y_0)^q dx dy \end{aligned}$$



Motion in frame immediately after impact | *Aligned motion from before impact* | *Masking out prior motion* | *Result of segmentation* | *Largest connected region found* | *Refinement of segmentation*

Figure 3-8: Collecting the motion evidence required for segmentation. The apparent motion after contact, when masked by the motion before contact, identifies seed foreground (object) regions. 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. This prevents the region assigned to the object motion from growing to include these fragments. The largest connected region, with a minor post-processing clean-up, is taken as the official segmentation of the object.

If these two features are used to build a simple nearest neighbor classifier, leave-one-out cross validation gives a classification accuracy of 89.8% (chance level is about 25%). So the shape information is a good predictor of object identity.

Qualitatively, poking turned out to be quite a robust procedure, with data gathered opportunistically during the unconstrained interaction of a human with the robot. For example, while the robot was being trained, a teenager visiting the laboratory happened to wander by the robot, and became curious as to what it was doing. He put his baseball cap on Cog's table, and it promptly got poked, was correctly segmented, and became part of the robot's training data.

3.8 Future directions

A unique advantage robots have for object segmentation is that they can reach out and touch the world. Imagine the classical face/vase illusion – this is trivial to resolve if you can simply poke it to see which part is free space and which is not. But poking is simply the lowest-hanging fruit in the set of active strategies a robot could use for achieving object segmentation. If the robot is unsure where the boundaries of an object lie, here are some strategies it can use :-

1. Poke the object gently. Tapping a solid object will induce a small motion of that object. This will result in a coherent region of optic flow on the image plane. If the object is non-rigid, or attached to other objects, then the response will be messy and complicated – but this is in some sense inevitable, since it is in just such cases that the idea of a unique "object boundary" runs into trouble

2. Thump the object savagely. A big disturbance is apt to generate a confusing motion that is hard to process directly. But it will move the object away from its local surroundings, giving another "role of the dice" – an opportunity for the robot to see the object against a new background, perhaps with better contrast. Frequently visual ambiguity is only a local, accidental effect.

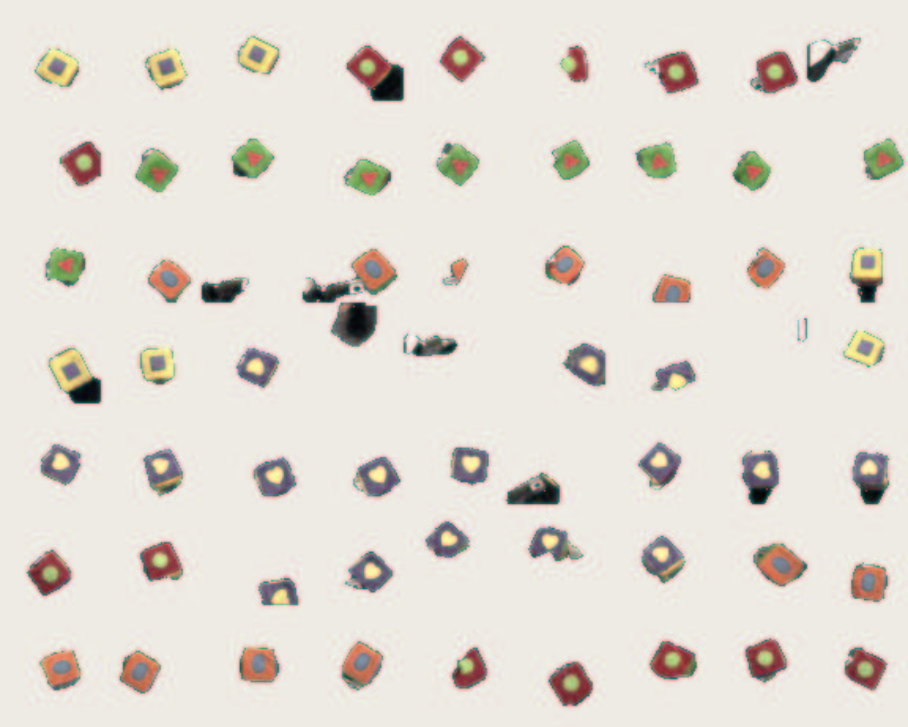


Figure 3-9: Results of a training session, where a toy cube was repeatedly offered to the robot for poking. Each image of the cube corresponds to the segmentation found for it during a single poke. The most common failure mode is inclusion of the robot arm in the segmentation.

3. Try to get the arm's endpoint beside the object. Anywhere the endpoint can reach is presumably free space, constraining the boundary of the object. We can use the arm's endpoint as a mobile reference object to confirm our theories of where free space lies.

4. Try to get the arm's endpoint behind the object. This has the advantage of putting a known background behind the object. Imagine the arm painted bright red to see the advantage of this for identifying the object's boundary.

5. Ask the human to present the object. A human bringing an object near the robot offers the dual advantage of motion cues and a known (if complicated) partial background – the hand.

6. Another alternative is to displace the robot's own head and body, again to get another "role of the dice", or to access three-dimensional information over a longer baseline than is available from the stereo cameras.

This does not even begin to exhaust the space of active strategies that are possible for object segmentation, and at the Humanoid Robotics Group at MIT we are investigating several others (Arsenio et al., 2003).

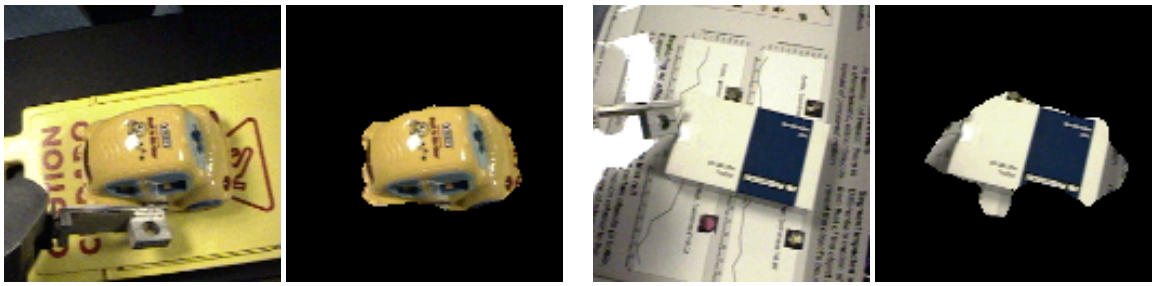


Figure 3-10: Challenging segmentations. The example on the right, a blue and white box on a glossy poster, is particularly difficult since it has complex shadows and reflections, but the algorithm successfully distinguishes both the blue and white part of the box from the background.

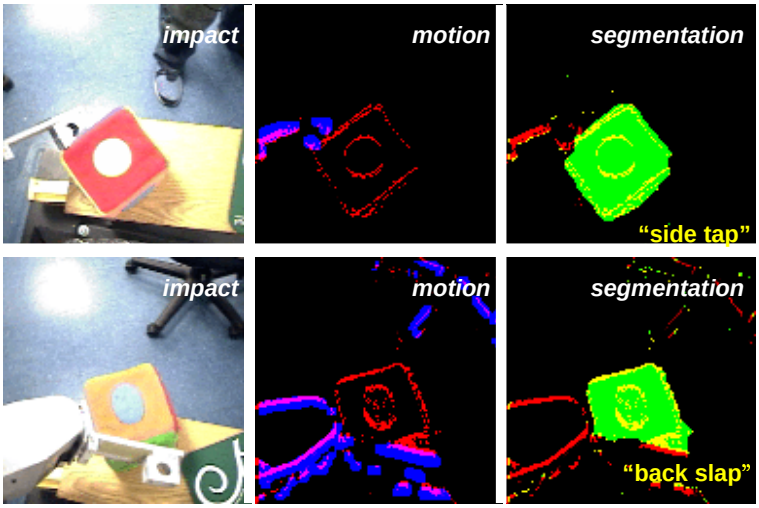


Figure 3-11: Cog batting a cube around from different directions. The images in the first column are from the moment of collision between the arm and the cube, which is detected automatically. The middle column shows the motion information at the point of contact. Red is new motion, purple and blue are pre-existing motion. Notice in the bottom row that the chair is moving. The bright regions in the images in the final column show the segmentations produced for the object.

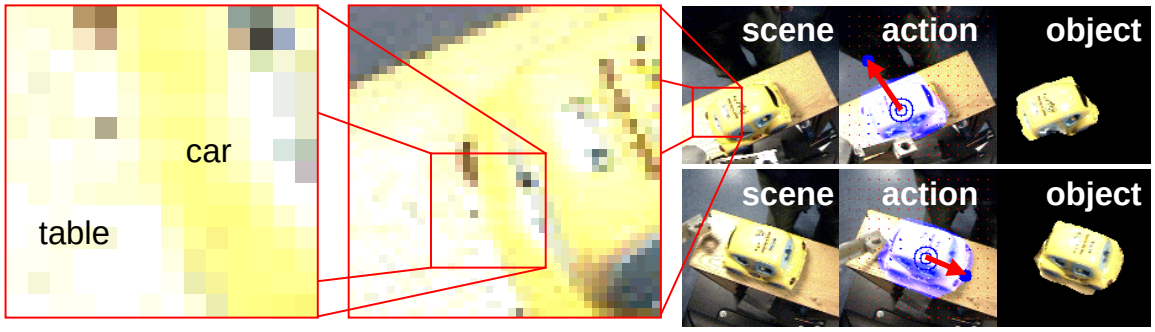


Figure 3-12: An example of the power of active segmentation. The images marked "scene" show two presentations of a yellow toy car sitting on a yellow table. The robot extends its arm across the table. In the upper sequence it strikes from below, in the lower sequence it strikes from the side ("action" images). Once the arm comes in contact with the car, it begins to move, and it can be segmented from the stationary background ("object"). On the left of the figure, a zoomed view of the car/table boundary is shown – the difference between the two is very subtle.

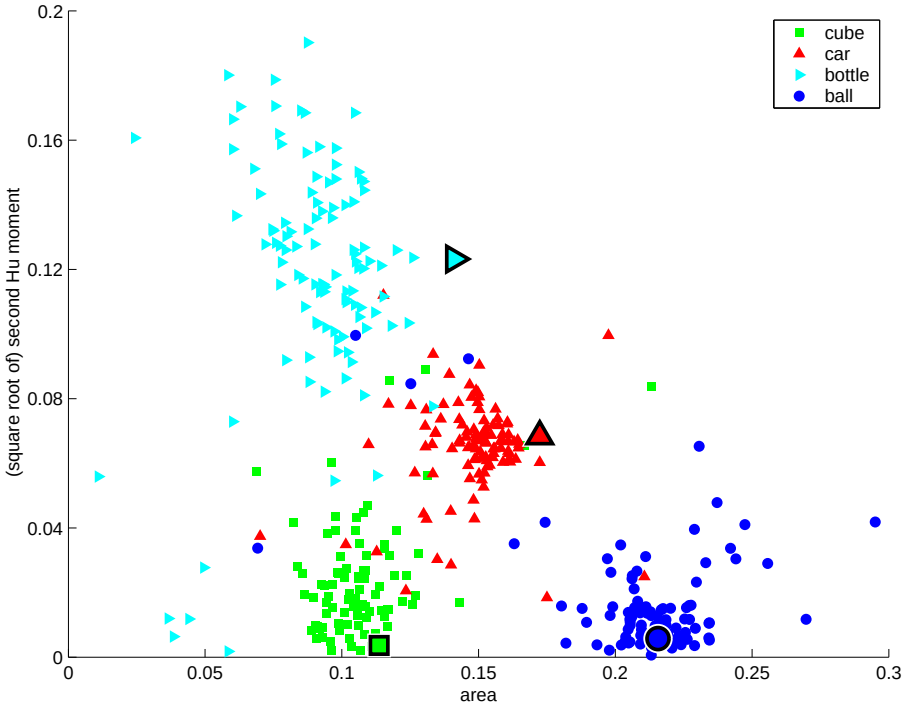


Figure 3-13: A large collection of segmentations are grouped by object identity, and then plotted (area versus second Hu moment). The enlarged markers show hand-segmented reference values. The segmentations are quite consistent, although area tends to be a fraction smaller than in the hand-segmented instances.