International Journal of Computer Vision, 7:1, 11-32 (1991)

Color Indexing

MICHAEL J. SWAIN
Department of Computer Science, University of Chicago, Chicago, IL 60637

DANA H. BALLARD
Department of Computer Science, University of Rochester, Rochester, NY 14627

Received January 22, 1991. Revised June 6, 1991.

Abstract
Computer vision is embracing a new research focus in which the aim is to develop visual skills for robots that allow them to interact with a dynamic, realistic environment. To achieve this aim, new kinds of vision algorithms need to be developed which run in real time and subserve the robot's goals. Two fundamental goals are determining the location of a known object. Color can be successfully used for both tasks.

This article demonstrates that color histograms of multicolored objects provide a robust, efficient cue for indexing into a large database of models. It shows that color histograms are stable object representations in the presence of occlusion and over change in view, and that they can differentiate among a large number of objects. For solving the identification problem, it introduces a technique called *Histogram Intersection,* which matches model and image histograms and a fast incremental version of Histogram Intersection, which allows real-time indexing into a large database of stored models. For solving the location problem it introduces an algorithm called *Histogram Backprojection,* which performs this task efficiently in crowded scenes.

1 Introduction

In recent years a new set of ideas about the goals and the methods of computer vision has gained prominence, and may be on its way to becoming the dominant paradigm, because it promises the quickest route to constructing working vision systems. The term used to describe this set of ideas is *animate vision*, introduced by Ballard (1989, 1991). Similar ideas have recently been expressed using the terms *active perception* (Bajcsy 1985, 1988), *active vision* (Aloimonos et al. 1988; Aloimonos 1990), *qualitative vision* (Nelson 1989, 1991), *inexact vision* (Thompson 1986), and *dynamic vision* (Dickmanns 1988). As in many other areas of science and technology, one of the driving forces behind the change in research strategy has been the availability of new research tools. In this case it has been the advent of powerful real-time imaging-processing equipment, light-weight video cameras, and off-the-shelf computer-controlled motors which allowed movable camera setups to be constructed.

The real-time constraints of *animate vision* require fast algorithms that enable the robot to achieve its goals. Two such goals are determining the identity of an object with a known location, and determining the location of a known object. Color, because it is an identifying feature that is local, and largely independent of view and resolution, can be efficiently used for both tasks. The locality of color information leads to an efficient algorithm for recognizing three-dimensional objects from a variety of viewpoints. The color-identification algorithm can be used without figure-ground segmentation, a task difficult to do without first recognizing the object. The algorithm can be used to identify deformable objects and substances described by mass nouns, something that most other recognition algorithms cannot be used for because they are based on shape.

1.1 The Role of Color in Vision

The ease of recognition using color strands in contrast to the neglect given recently to color as a recognition cue, although it has been used in earlier work (Feldman & Yakimovsky 1974; Garvey 1976; Ohlander et al. 1978). Instead, much more attention has been given to geometric algorithms that extract shape from stereo, motion, and lighting cues. The fundamental reason that color has not been used may be that it is not intrinsically related to the object's class identity in the way that these other cues are. This view is well represented in Biederman (1985):[1]

> Surface characteristics such as color and texture will typically have only secondary roles in primal access. . .we may know that a chair has a particular color and texture simultaneously with its volumetric description, but it is only the volumetric description that provides efficient access to the representation of CHAIR.

The implicit claim made in the quotation above is that form follows function: Geometrical cues will be the most reliable of object identity. While this may be generally true, it may not be true for routine behavior (Chapman 1990). In such behavior, wherein familiar objects are interacted with repeatedly, color may be a far more efficient indexing feature.

Color may be used in other situations as well. There are many examples in nature where color correlates with the class identity of an object, because of pigments which form part of the function of the object, or because species use it to send messages of enticement or warning. Similarly, color is used as a trademark or identifying feature in objects that occur in artificial environments, such as packaged goods, advertising signs, road signs, etc. Shape cues, in contrast to color, are highly resolution dependent, include only a highly restricted set that is view invariant (e.g., corners, zeros of curvature), and may require elaborate processing to extract them from an image.

Robotic vision systems can also use representations that are heavily personalized to achieve efficient behaviors. For example, it may not be helpful to model coffee cups as being red and white, but *yours* may be, and that color combination is very useful in locating it and recognizing it. Recognition of a particular object is a task that is probably carried out as often as classification; and while classification may in some cases precede recognition of the individual this need not be true in general.

Another reason why color has not been used may have been the lack of good algorithms for *color constancy*, that is perceiving a stable perception of color over varying light conditions, as poeple do in most circumstances. However, recently there has been great progress in correcting both for the chromaticity of the illuminant (Maloney & Wandell 1986; Forsyth 1990; Rubner & Schulten 1989; Brainard et al. 1989; Novak & Shafer 1990) and for geometric effects such as specularity (Klinker et al. 1988). So there is good reason to believe that color can be used as an identifying invariant of object surfaces, even under varying light conditions.

1.2 What vs. Where

Eye traces of human observers suggest that we do not build categorical databases of the world around us independent of the task we are carrying out but that, instead, only highly selective regions of the scene are examined in detail, and these are highly dependent on the task being carried out. Furthermore, the sequential nature of the eye movement traces suggests that the visual architecture cannot analyze the entire picture at a glance but must break the analysis up into smaller sequential components. One gross distinction that we make is between *identification algorithms* that analyze the foveated area during fixation and *location algorithms* that direct the eyes to new targets.

Support for this *what/where* distinction comes from studies of human and primate brains. A significant feature of the gross organization of the primate visual brain is the specialization of the temporal and parietel lobes of visual cortex (Mishkin & Appenzeller 1987; Maunsell & Newsome 1987). The parietal cortex seems to be subserving the management of locations in space whereas the temporal cortex seems to be surbserving the identification of objects in the case where location is not the issue. In a striking experiment by Miskin (Miskin & Appenzeller 1987), monkeys with parietel lesions fail at a task that requires using a relational cue, but have no trouble performing a very similar task that requires using a pattern cue. The reverse is true for temporal lesions. Why should the primate brain be specialized in this way? If we think generally about the problem of relating internal models to objects in the world, then one way to interpret this "What/Where" dichotomy is as a suggestion that image interpretation, the general problem of associating many models to many parts of the image simultaneously, is either too hard or unnecessary, or both (see table 1). In order to build vision systems that function in real time, perhaps the problem must be simplified. In sections 3 and 4, approaches to the identification and location problems are presented and tested.

Table 1. The biological organization of cortex into What/Where modules may have a basis in computational complexity. Trying to match a large number of image segments to a large number of models at once may be too difficult. (From Ballard (1991)).

		Object to Match Against	
		One	Many
Image Portions	One		Identification: trying to identify an object whose location can be fixated
	Many	Location: trying to find a known object	Image interpretation: Too hard?

1.3 Outline

Section 2 introduces the multidimensional color histogram. Given a discrete color space, a color histogram counts how much of each color occurs in the image. Color histograms are invariant to translation, rotation about an axis perpendicular to the image, and change only slowly with rotation about other axes, occlusion, and change of distance to the object. On the other hand, histograms for different objects can differ markedly, and there are a huge number of possible histograms (exponential in the number of different colors in the color space). Therefore, the color histogram is an excellent representation to use for identifying objects.

Section 3 introduces a method of comparing image and model histograms called *Histogram Intersection*, which is especially suited to comparing histograms for recognition because it does not require the accurate separation of the object from its background or occluding objects in the foreground. Experiments show that Histogram Intersection can distinguish models from a large database, that it is robust to occlusion as well as image and histogram resolutuion, and that only a small number of histograms are needed to represent a three-dimensional object. They also show that an effective color-constancy algorithm will be needed for Histogram Intersection to work under variable light conditions. The section also describes a modification of Histogram Intersection called *Incremental Intersection* that allows efficient indexing into a very large database.

Section 4 shows how a model histogram can be used to find the location of a known object (the *target*). The algorithm to solve this problem is called *Histogram Backprojection.* It finds the region in the image where the colors in the model histogram show up together, relying more on those colors that show up about as much as expected than those which show up much more, and therefore occur in other objects besides the target.

The experiments in section 4.1.2 show that Histogram Backprojection works well even when the objects containing the same colors occur in the image and when the object is partially occluded.

2 Color Histograms

Given a discrete color space defined by some color axes (e.g., red, green, blue), the color histogram is obtained by discretizing the image colors and counting the number of times each discrete color occurs in the image array. The image colors that are transformed to a common discrete color are usefully thought of as being in the same 3D histogram bin centered at that color. To illustrate, figure 1 (see color figures on page 29) shows the output from a color camera together with a color histogram obtained from the image.

Histograms are invariant to translation and rotation about the viewing axis, and change only slowly under change of angle of view, change in scale, and occlusion (see color figures 2 and 3 on page 30). Because histograms change slowly with view, a three-dimensional object can be adequately represented by a small number of histograms, corresponding to a set of canonical views (Koenderink & van Doorn 1976; Feldman 1985).

Histograms define an equivalence function on the set of all possible colors, namely that two colors are the same if they fall into the same bin. This equivalence function is not ideal for recognition, because the relative range of colors that are considered the same as a given color depend on where the given color is located within the bin. Ideally, the colors considered the same would be in a region centered on the color, or in some region whose shape depends on knowledge of the possible variations introduced by changes in lighting or noise in the sensors. It should be determined by the random effect of how the color happens to link up with respect to the tesselation of the discrete color space. Another problem is that the equivalence is all or nothing. Presumably, as the difference in color of two object patches increases the probability of them being the same object patch decreases smoothly. The binary threshold used to define the tesselation serves as a crude approximation to the probability function.

Histograms whose bins define overlapping bell-shaped (e.g., Gaussian) functions of color space would address some of the concerns of the previous paragraph, as would interpolation coding (Ballard 1987). Extensions such as these have not been considered because histograms in their simplest form have worked well. Why do histograms works, despite their inherent problems? Objects tend to have surfaces that are made up of regions of color. Because of shading and camera noise these regions are blurred in color space, and so span more than one bin in a historgram. When image and model histograms of the same object are matched, a high match value is obtained because the regions match well, even if point-by-point matches on the object's surface are not always reliable.

Strat (1990) has matched *cumulative histograms* with a match algorithm similar to Histogram Intersection to make a system that is robust to lighting changes. In a three-dimensional color space (x, y, z) the cumulative histogram is defined:

$$C(x, y, z) = \sum_{i=1}^{x} \sum_{j=1}^{y} \sum_{k=1}^{z} H(x, y, z)$$

where $H(x, y, z)$ is the non-cumulative histogram discussed above.

Both the object identification and object search implementations described in the following sections use color histograms to represent objects.

3 Identification

This section describes how to use the color histogram to identify an object whose approximate location is known, the "Identification" problem of table 1. To identify objects based on their color histogram, we must be able to judge the similarity of the color histogram of an image to the color histograms in the database. Section 3.1 introduces a method of comparing image and model histograms called *Histogram Intersection*, which tells how many of the pixels in the model histogram are found in the image. This method is especially suited to comparing histograms for recognition because it does not require the accurate separation of the object from its background of occluding objects in the foreground, often a difficult task to perform before the object has been recognized. The results of the experimental section show that:

- Histogram Intersection can distinguish objects from a large database (66 objects).
- The Histogram Intersection match value is insensitive enough to rotation and moderate changes in distance so that only a small number of views is needed to represent a three-dimensional object (about 6).
- The range of colors that occur in the world need only be split into about 200 different discrete colors to distinguish a large number of objects, so color constancy to the degree demanded by the algorithm should be feasible. However, without transforming the input by a color-constancy algorithm, Histogram Intersection is sensitive to lighting changes.
- Identification can be done even when a significant amount of the object is *occluded* (not visible).
- Recognition accuracy is typically extremely insensitive to the histogram resolution used.

Section 3.2 describes an incremental version of Histogram Intersection, called *Incremental Intersection*. By matching the largest bins from the image and the models, Incremental Intersection allows extremely fast indexing into a large database. Experiments show that Incremental Intersection does not sacrifice accuracy because most of the information is carried by the largest bins of the histograms.

3.1 Histogram Intersection

Because the model database may be large, we can afford only a highly restricted amount of processing per model, but at the same time we must be able to overcome the problems that hinder recognition, most importantly

- distractions in the background of the object,
- viewing the object from a variety of viewpoints,
- occlusion,
- varying image resolution,
- varying lighting conditions.

The matching method proposed here, called *Histogram Intersection*, is robust to the first four problems; the last is left to a color-constancy module that operates on the input prior to the histogram stage.

3.1.1. Description. Given a pair of histgrams, I and M, each containing n bins, the intersection of the histograms is defined to be

$$\sum_{j=1}^{n} \min (I_j, M_j).$$

The result of the intersection of a model histogram with an image histogram is the number of pixels from the model that have corresponding pixels of the same color in the image. To obtain a fractional match value between 0 and 1 the intersection is normalized by the number of pixels in the model histogram. The match value is then

$$H(I, M) = \frac{\Sigma_{j=1}^{n} \min(I_j, M_j)}{\Sigma_{j=1}^{n} M_j}$$

The normalized histogram intersection match value is not reduced by distracting pixels in the background. This is the desired behavior since complete segmentation of the object from the background cannot be guaranteed. Segmentation is still a topic of active reserach, but the indications from the large amount of research done in the past are that complete, efficient, and reliable semgnetation cannot be done prior to recognition. The histogram intersection match value is only increased by a pixel in the background if

- the pixel has the same color as one of the colors in the model, and
- the number of pixels of that color in the object is less than the number of pixels of that color in the model.

There are a number of ways of determining the approximate depth of an object, from laser or sonar range finders, disparity, focus, or touching the object with a sensor. The depth value combined with the known size of the object can be used to scale the model histogram. Alternatively, if it is possible to segment the object from the background and if it is not significantly occluded, the image histogram can be scaled to be the same size as the model histogram. Appendix A shows that when it is possible to segment the object from the background and thus scale the image histogram to be the same size as the model histogram, that Histogram Intersection is equivalent to the use of the sum of absolute differences or *city-block* metric. That is, if

$$\sum_{i=1}^{n} M_i = \sum_{i=1}^{n} I_i$$

then if we let T equal this value, we have

$$1 - H(I, M) = \frac{1}{2T} \sum_{i=1}^{n} |I_i - M_i|$$

If images are scaled by depth then Histogram Intersection does not define a metric, since there is an asymmetry between the model and the image. That is, the model and image historgrams are not constrained to contain the same number of pixels, so the normalization factor in the denominator will differ matching image to model as matching model to image. This assymetry is a natural result of expecting background pixels in the image but not in the model.

Histogram Intersection is capable of differentiating among a large number of different objects. Appendix B shows that the fraction of the multidimensional space defined by the bins of the histogram occupied by a single model is at most

$$\frac{(2\delta)^{n-1}}{\sqrt{n}}$$

where $1 - \delta$ is the minimum Histogram Intersection match value allowed and n is the number of bins in a histogram.

For $\delta = 0.4$ (a reasonable number based on our experiments) and $n = 512$, the fraction is 1×10^{-51}. If histograms were distributed evenly throughout color space, the reciprocal of this number would approximate the carrying capacity of the histogram. But histograms are not distributed evenly throughout histogram space, as shown by figure 4 on page 30. For a back-of-the envelope calculation we can try to account for the unequal distribution by reducing the number of histogram bins to, say 100, as figure 2 would suggest. The fraction of histogram space is still very small, 3×10^{-11}. A more accurate analysis might consist of generating a Monte-Carlo distribution of histograms throughout color space and measuring their overlap.

3.1.2. Experiments. Experiments were performed to see if a large number of objects can be distinguished and to test the senstitivity of the recognition technique to changes in view, image resolution, and occlusion.

An experimental test of histogram intersection shows that the technique is capable of indexing into a large database, that is, eliminating most of the possible matches leaving only a small number for further consideration. The 32 images in figure 6 were matched to each of the 66 models in the database in figure 5 (see color figures on page 31). The color axes used for the histograms were the three *opponent color* axes, defined as follows (Ballard & Brown 1982):

$$rg = r - g$$

$$by = 2 * b - r - g$$

$$wb = r + g + b$$

Here r, g, and b represent red, green, and blue signals, respectively. The rg, by, and wb axes are analogous to the opponent color axes used by the human visual system (Lennie & D'Zmura 1988). They were used here simply to allow the intensity (wb) axis to be more coarsely sampled than the other two, because the intensity axis is more sensitive to lighting variation from shadows and distance from the light source. The wb axis was divided into 8 sections, while the rg and by bins were each divided into 16 sections, for a total of 2048 bins. Because the total intensity limits the color differences possible, only a fraction of them can actually receive counts. For example, suppose the camera outputs a maximum M on each channel. Then if $wb = 0$ ($r = g = b = 0$) or $wb = 3M$ ($r = g = b = M$) then the variables by and wb must both take on the value 0 (see (Swain 1990a) for more details). Because most colors we experience are fairly unstaturated (i.e., close to the wb axis), even for objects such as ones in the database shown in figure 5, only about 200 ($5\times5\times8$) of the 512 receive an appreciable number of counts, as is discussed further below. So we are dividing up color space fairly coarsely.

For the 66-object database shown in Figures 5–8, the correct model is the best match 29 of 32 times and is always one of the top two matches. The three cases when the correct model was the second highest match are, listed in the format (*model: object receiving larger response*).

1. Crunchberries: Campbell's Special Request soup
2. Raisin Bran: Campbell's Chicken with Rice soup
3. Windsurfer shift: Ivory detergent bottle

Other, more expensive, matching techniques can be used to verify which of the top scoring models is the correct one, so it is not crucial that the correct model is always the best match. In the experiment the models were segmented from the background prior to generating the model histograms. No segmentation was performed on the images of the unknown objects.

In addition to using methods other than color to resolve ambiguous cases, there are steps that can be taken to improve the use of color information in the histogram intersection algorithm. All three of the models that received larger match values than the correct models had smaller numbers of pixels in their

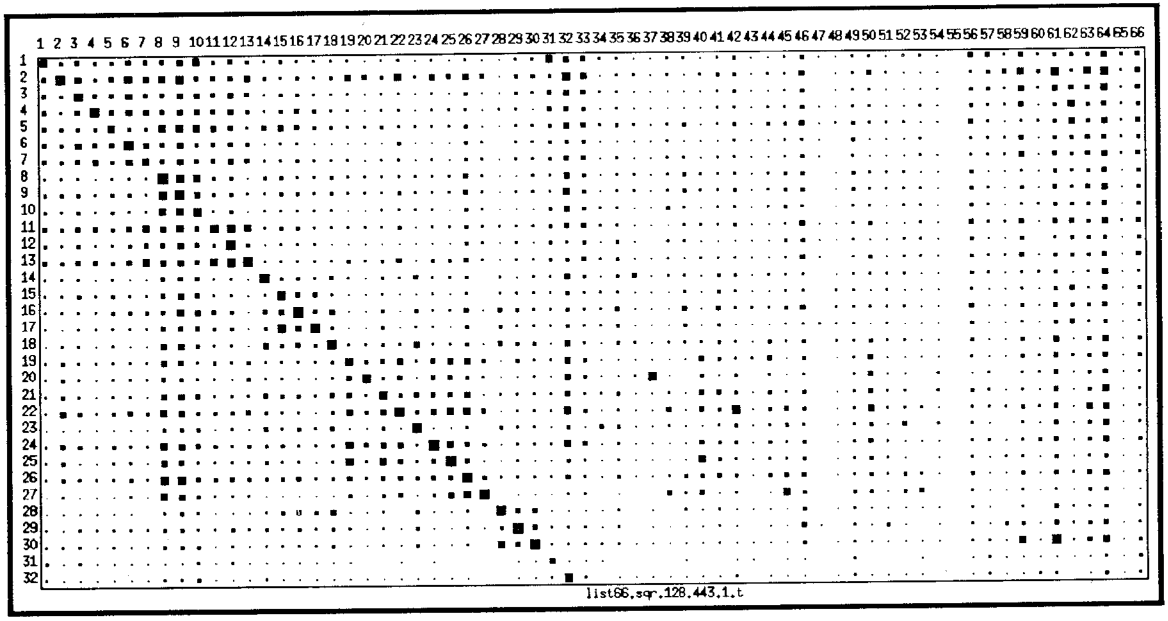

Fig. 7. The results of matching all combinations of image and database histograms displayed pictorially where the size of the squares are proportional to match values. The dominance of the diagonal values shows that the correct match is almost always selected. Twenty-nine of thirty-two matches are correct; in three cases the correct model received second-highest score. Models are along the horizontal axis; unknown objects along the vertical axis.

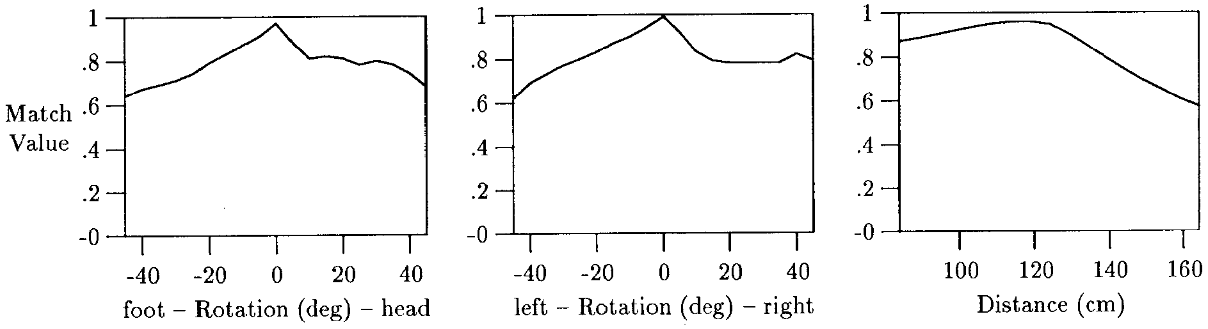

Fig. 8. Variation of the Histogram Intersection match value (see section 3 for definition) as the camera is moved with respect to a Snoopy doll. In the Distance graph the model image was taken at a distance of 124 cm. The match value changes slowly with changes in angle and distance.

histograms. It is easier to find evidence for a smaller object in any given image region that it is to find evidence for a larger object. A recognition system could choose to verify the large objects with high match values before the smaller ones. Alternatively, if the distance is known, objects could be categorized by their size before indexing using color.

One important claim is the insentivitity of the matching process to variations in view. To test this, the variation in match value with respect to view changes in a single model, the Snoopy Doll, was studied further. Figure 9 shows how the Histogram Intersection match value changes as the camera is rotated about the Snoopy Doll shown in figure 5, and moved closer and further from the doll. Compare these match values to the ones in figure 10. Even at 45 degrees rotation or 1⅓ times the original distance the match value (about 0.6) is higher than 99 percent of the false matches.

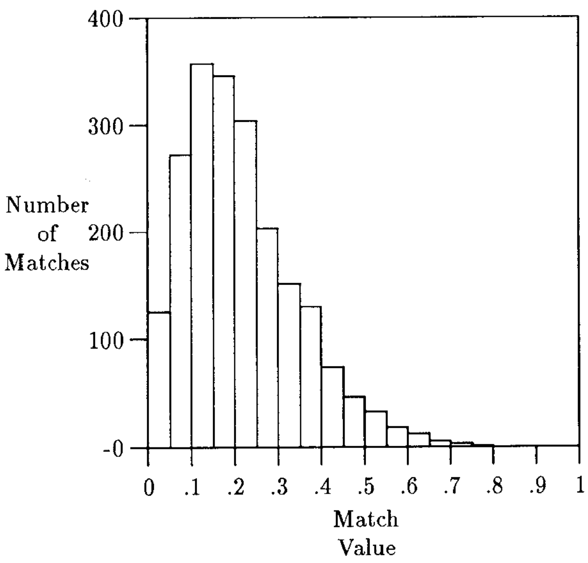

Fig. 9. Distribution of match values for incorrect image-model matches for the models and images shown in figures 4 and 5. The values counted here are all the off-diagonal elements of the matrix shown in figure 7. The values for incorrect matches rarely exceed the values for different views of the same object (see figure 8), even when they are obtained from significantly different angles and distances.

Another important claim is that recognition accuracy is fairly insensitive to image resolution. Table 2 shows how match success is affected by reducing the resolution of the images. The images were reduced in resolution by averaging the values of the pixels to be combined, and the histograms were taken from the reduced images. The model histograms were obtained from images of size 128×90 and scaled by the appropriate multiplicative factor prior to matching. For each image the match values to each model are sorted; the *rank* is the position of the correct match in this list. The match percentile for each image matched is then calculated as (n-models − rank) / (n-model − 1), where n-models is the number of models in the database (66). The match percentile is averaged over all 32 images in the experiment. A value of 100 indicates perfect matching; a value of 99 indicates that, on average, the correct match scored a higher match value that 99 of 100 of the other models, and so on. An average match percentile of 50 indicates the match selection is no better than random.

This experiment simulates matching to an object that covers only a small region of the image array. The match values are reasonable even for images of size 16×11, which is fewer than 200 pixels! The success of matching under low resolution can be explained by the fact that the images being matched to the database have fairly large regions of constant color. The color

Table 2. Image resolution and match accuracy. The *Correct Match Placement* columns show the rank of the correct match for each of thirty-two images in figure 5 being matched to the sixty-six models in figure 4. The model histograms were obtained from images of 128×90, and scaled appropriately. See the text for the definition of *average match percentile.*

Image Size	Correct Match Placement				Average Match Percentile
	1st	2nd	3rd	> 3rd	
128 x 90	29	3	0	0	99.9
64 x 45	27	5	0	0	99.8
32 x 22	24	7	1	0	99.6
16 x 11	15	6	4	0	97.8
8 x 5	4	4	3	21	78.1

Table 3. Match accuracy with scaled image and model resolution. Both model and image histograms were generated from images of the same indicated size. Interpret the match data as in table 2.

Image Size	Correct Match Placement				Average Match Percentile
	1st	2nd	3rd	> 3rd	
128 x 90	29	3	0	0	99.9
64 x 45	29	3	0	0	99.9
32 x 22	28	4	0	0	99.8
16 x 11	23	5	3	1	99.3
8 x 5	17	7	2	6	97.7

histograms of more highly textured objects could change more dramatically over scale. Images obtained from typical cameras contain 512×485 (about ¼ million) pixels, so these results suggest that color matching could be performed reliably on regions that cover as little as 1/1000 of the total image area, provided the camera was focused well enough that camera blur did not destroy detail in the image.

Table 3 shows that match success is improved for extremely low-resolution images by obtaining the model histograms from low-resolution images as well. Since it is unlikely that matching will be done to such small parts of the image, scaling the model histogram will do in most circumstances. What this does suggest is that a hypothetical extremely inexpensive system that operated on very low-resolution images would be able to recognize the dominant object in the image using color about as well as a full-resolution system. Examples of the reduced-resolution images are shown in color figure 6.

Recognition accuracy is also fairly insensitive to occlusion. To test this, subparts of the images were matched to the database. First, the bottom third of each image was removed (see color figure 11 on page 31) before matching to the database. Then, the right-hand third of the image was removed, leaving only 4/9th of the original image (color figure 12). This set was also matched to the database. The results are shown in table 4. There is only slight degradation in the match accuracy with occlusion. The correct matches are still among the top three match values (out of sixty-six), even for the most severe occlusion.

These results demonstrate how match values will degrade under occlusion when the occluding object can be segmented from the object of interest. Matching in the presence of occlusion will be more difficult when segmentation cannot be achieved, because colors from the occluding object may also match to the models in-the database. Nevertheless, other experiments have shown that matching can be achieved even with some occlusion of this sort (see Swain (1990b)).

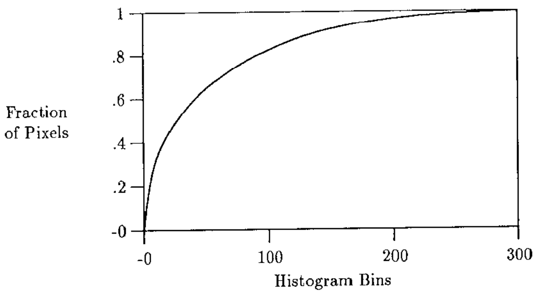

Fig. 12. Distribution of pixels across histogram bins for the database shown in figure 4 (black background removed). A point (x, y) on the curve indicates that fraction y of pixels fall into the x largest bins. There are a total of 512 bins in the entire histogram.

Table 4. Occlusion and match accuracy.

Occluded Region	Correct Match Placement				Average Match Percentile
	1st	2nd	3rd	> 3rd	
None (Figure 4)	29	3	0	0	99.9
Bottom (Figure 10)	27	4	1	0	99.7
Bottom, Side (Figure 11)	22	5	5	0	99.3

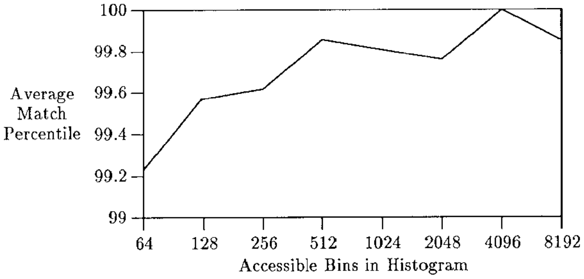

Fig. 13. Effects of changing histogram resolution on match success.

The number of bins in the opponent color histogram that receive a significant number of counts is much smaller than the total number of bins that could possibly receive counts. Figure 4 shows the distribution of counts for the histograms representing the models in figure 5. (Remember the background black is subtracted before creating the model histograms.) Sixty-five percent of the counts lie in the top fifty bins, eighty-three percent lie in the top one hundred bins, and ninety-six percent lie in the top two hundred. The bins that receive the most counts lie on the white-black (*wb*) axis. The numbers drop off with distance away from this axis.

The results of Histogram Intersection were extremely insensitive to the number of bins in the histogram used in the image and model histograms. Figure 13 shows the effects of varying the size of the histogram over two orders of magnitude, from 64 accessible bins (8×8×4 bins total) to 8125 accessible bins (40×40×20 bins total). There are only small changes in the match effectiveness over the entire range of histogram sizes. Note that matches in the high resolution histograms rely on the fact that there are smooth distributions of colors on the objects. These distributions arise from the large regions of constant color being blurred in color by shading and camera noise. In the highest resolution histograms the same pixels are not matching each other, but different pixels from the same color region.

In a set of real-time experiments employing a Datacube pipelined image processor, 8×8×8 (red, green, blue) histogram were used instead of the 16×16×8 opponent color histograms, with good results, so the choice of color axes is not crucial either. The color camera was mounted in our mobile robot platform and panned across a floor containing the database of colored shirts. The shirts were spread out on the floor, but no special effort was made to lay them perfectly flat or approximate the view in the database. Since this experiment tests the "what" or identification problem, the panning is done so that the image serves as a fovea, that is, each shirt, when centered, occupies the majority of the image. Nevertheless, the shirts are close enough that there is often another shirt in the background when a match is being done. The shirt occupying the major portion of the image is invariably the top notch.

As discussed at the beginning of this section, it is expected that a color-constancy algorithm be used before histogramming. Nonetheless, we tested Histogram Intersection in the presence of changing light intensity without color constancy. One aim is to see how necessary a color-constancy algorithm is. As we expected, changes in lighting conditions affect the match value considerably. More importantly, this experiment can be also used to test how well a color-constancy algorithm must work. Changing light intensity was simulated by multiplying the image pixel values by a constant factor ranging from 0.4 to 1.6. The resulting pixel values were constrained to be no greater than 255, as would occur in a camera. The transformed images were matched to the original models. The results are displayed in figure 14.

It appears that with 16×16×8 opponent color histograms, where 8 is the number of bins along the white-black axis, good pruning of the possible matches can be achieved if the intensity is recovered to within about plus or minus 15 percent of the true values. Since one bin in the white-black direction only represents at most 12.5 percent of the maximum value in the histogram, the matching process can still work even if many of the counts fall in neighboring bins. This is because regions tend to cover a number of neighboring bins in color space, the same explanation for why fine-grain

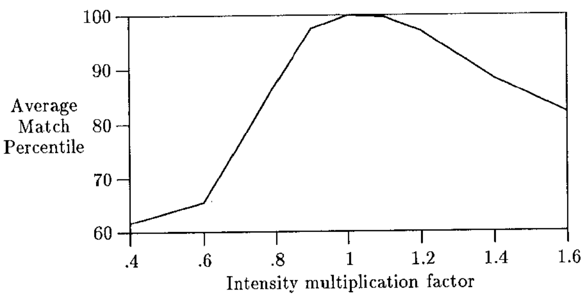

Fig. 14. Effects of changing intensity on match success.

histograms can be used for accurate matching (see figure 13). When the multiplying factor is 0.4, a 60-percent decrease in light levels, the average match percentile is 61.7, not much better than random (50).

The simplest color constancy algorithm simply normalizes the red, green, and blue responses by their sum, that is:

$$r' = r/(r + g + b)$$

$$g' = g/(r + g + b)$$

$$b' = b/(r + g + b)$$

These new axes provide only two degrees of freedom, since the values of any two define the third. With 8×8 (r', g') histograms, an average match percentile of 98.0 was achieved (see table 5), with the worst ranking of an object matched to itself being a seventh place for the Raisin Bran box. The normalized intensity values are unaffected by changes in lighting intensity (assuming a linearized camera), making them much more effective than the 3D histograms in variable lighting conditions. By placing an object of known reflectance in the image and normalizing the responses with respect to it, a third axis of color information could be recovered under variable intensity lighting.

Table 5. Matching with normalized color signals.

Correct Match Placement				Average Match Percentile
1st	2nd	3rd	> 3rd	
15	7	3	7	98.0

In summary, Histogram Intersection can successfully prune the number of candidates in a large database to a small number of possible matches. Because color histograms change only slowly over view, a small number of them can be used to represent a three-dimensional object. Histogram Intersection is robust to occlusion and changes in image and histogram resolution. With three sensors, three-dimensional histograms are sensitive to changing light conditions; and so, when the lighting is variable, it must be used after the pixel array has been transformed by an effective color-constancy algorithm. We demonstrated the effects of using the simplest color-constancy algorithm, scaling the color axes by the total intensity. The sensitivity to intensity variations was eliminated, as the cost of a moderate decrease in the ability to prune objects from the database under changing lighting conditions.

Histogram Intersection is an efficient way of matching histograms. Its complexity is linear in the number of elements in the histograms. Two 16×16×8 histograms can be matched in 2 milliseconds on a SUN Sparcstation 1 (a 12 MIP RISC machine). The histograms themselves are efficient to compute using parallel image processing hardware. For instance, generating a 16×16×8 histogram from a 512×485 image takes about 40 milliseconds using a Datacube FeatureMax board, including the time needed to transfer the histogram to the host.

Histogram intersection is efficient compared to most recognition schemes. Nevertheless, for large databases the linear dependence of the recognition scheme on database size will add up. Parallel processing is one way of attacking this problem, since the match over different models is easily parallelizable. Another way, which reduces the recognition complexity to constant time for a broad range of databases, is described in the next section.

3.2 Efficient Indexing into a Large Database

We introduce an algorithm, called *Incremental Intersection*, for indexing into a large database efficiently on a sequential computer. In this scheme, only the largest bins from the image and model histograms are compared, and a partial histogram intersection value is computed. The computation is incremental, so that the algorithm can be interrupted at any time after the sort with as good results as one could expect with the amount of time used. This last feature could prove to be important in a system that interacts with a dynamic world, in which the times that actions are taken are often dictated by outside events.

Incremental intersection is split into two phases, an off-line phase in which the data structure representing the database is generated and an on-line matching phase. In the off-line phase:

Table 6. Recognition times. Each histogram contains 2048 bins. Times (in milliseconds) were measured on a SUN SPARCstation 1.

	Database Size		
	19	37	70
Histogram Intersection	38	73	150
Incremental Intersection (B=10)	15	15	15

1. Assign to each bin in each model histogram a key which is the fraction of the total number of pixels in the histogram that fall in that bin.
2. Group the bins by index (color).
3. Sort each group by key.

In the on-line phase:

1. Sort the image histogram bins by size.
2. For the B largest image bins, starting with the largest, match the image bin to all the model bins with the same index whose key is larger. If previously unmatched model bins are matched, match them to all larger image bins.

The efficiency of Histogram Intersection and Incremental Intersection is compared in table 6. A complexity analysis shows that the time for Histogram Intersection depends linearly on the product of the size of the histogram and the size of the database, i.e., the algorithm is $O(nm)$, where n is the total number of bins in the histogram and m is the number of models in the database. The asymptotic complexity of Incremental Intersection is $O(n \log n + cm)$, where c is the number of image bins used for indexing into the database. The complexity of Incremental Intersection is also linear in the size of the database. However, the constant factor is so low that for most databases the complexity is dominated by the sort of the image histogram bins.

Incremental Intersection only computes an approximation of the Histogram Intersection value, unless it is allowed to match every bin in the image histogram. One would think, therefore, that its performance would only be a fraction as good as complete Histogram Intersection. Figure 15 shows that the match effectiveness climbs very quickly and even surpasses that of Histogram Intersection using small numbers of image histogram bins. After examining 10 bins, Incremental Intersection matches the images to the models without error, whereas in three cases the Histogram Intersection match value for the correct match was the second highest. How can this happen? Most of the surface of each object in the test database consists of at most five or six different colors, so then histogram bins capture a good percentage of the uncorrupted signal coming from the object, while the smaller bins are more likely to be noise.

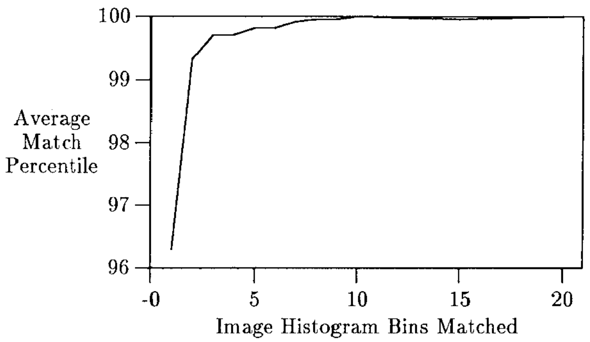

Fig. 15. Effectiveness of Incremental Intersection as function of the number of image histogram bins matched. For comparison, the average match percentile for Histogram Intersection is 99.86.

4 Location

The previous section discussed recognizing an unknown object whose location is known, the "Identification" box in table 1. This section discusses the complementary task, locating a known object, the "Location" box in the same table. Determining the location of an object is necessary when executing many tasks, not only looking for a "lost" object. Fixating a moving object, or a stationary object when the robot is moving, also requires keeping track of the location of the object (Coombs 1989). Verging a pair of cameras upon an object requires determining the location of an object in both images so that they can be registered (Olson & Coombs 1991). All these tasks can be accomplished using color histograms and an algorithm called *Histogram Backprojection.*

4.1 Histogram Backprojection

Histogram Backprojection answers the question "Where in the image are the colors that belong to the object being looked for (the *target*)?" The answer is given in such a way so that the colors that appear in other objects besides the target are deemphasized so that they are less likely to distract the search mechanism. Experiments show that the technique works for objects in cluttered scenes under realistic conditions.

In Histogram Backprojection the model (target) and the image are represented by their multi-dimensional color histograms M and I as in Histogram Intersection. A *ratio histogram* R, defined as

$$R_i = \min\left(\frac{M_i}{I_i}, 1\right)$$

is computed from the model and image histograms. It is this histogram R that is backprojected onto the image, that is, the image values are replaced by the values of R that they index. The backprojected image is then convolved by a mask, which for compact objects of unknown orientation could be a circle with the same area as the expected area subtended by the object. The peak in the convolved image is the expected location of the target, provided the target appears in the image.

4.1.1. Description. More precisely, let $h(c)$ be the histogram function that maps a color c (a three-dimensional value) to a histogram bin (another three-dimensional value). Let D^r be a disk of radius r:

$$D^r_{x,y} = \begin{cases} 1 & \text{if } \sqrt{x^2 + y^2} < r \\ 0 & \text{otherwise} \end{cases}$$

Define the "loc" function to return to pixel (x, y) with the value of its argument, and let the $*$ symbol denote convolution. Then Histogram Backprojection can be written

```
1. for each histogram bin $j$ do
       $R_j := \min\left(\frac{M_j}{I_j}, 1\right)$
2. for each $x, y$ do
       $b_{x,y} := R_{h(c_{x,y})}$
3. $b := D^r * b$
4. $(x_t, y_t) := \mathrm{loc}(\max_{x,y} b_{x,y})$
```

4.1.2. Experiments. As a demonstration of Histogram Backprojection, we consider figure 6 as a single crowded scene, and look for objects within it using the models from figure 5. The results are shown in figures 16 and 17 (see color figures on page 32). In all cases but four the largest peak in the convolved image corresponds to the correct object. The four cases in which it doesn't are listed below in the format (*target: objects receiving larger response*).

1. Wheaties: Manischewitz matzo farfel.
2. Campbell's clam chowder: red and white shirt, Campbell's chicken soup.
3. Charmin: orange White Cloud.
4. Mickey Mouse underwear: red and white shirt, USA Flyer.

The success rate is shown more graphically in table 7.

Because the convolution can be carried out on a reduced resolution image, Histogram Backprojection is very efficient. Its complexity on a sequential computer would be $O(I + c * I')$ where I is the number of pixels in the full-resolution image, I' is the number of pixels in the reduced resolution image, and c is the number of pixels in the convolution mask applied to the reduced resolution image.

Table 7. Performance of Histogram Backprojection. The number in each square is the rank of peak that falls into the corresponding square in figure 5 when looking for the model whose image is in that square. A "1" means the object has been correctly located, a "2" indicates the object created the second largest peak in the convolved backprojected image, etc.

1	1	1	1	1	1
2	1	3	1	1	1
1	1	2	1	1	1
1	1	1	1	1	1
1	3	1	1	1	1
1	1				

Histogram Backprojection has been implemented in a Datacube image processor, with a Sun 4/260 workstation as its host. The Datacube can do histograms, subsample, and convolutions with 8×8 masks within a frame time. Using 8×8×8 (512 total) size histograms and a reduced image of size 32×32 (1024 pixels) for the convolutions, the algorithm can be executed four times a second. The real-time experiments show that because Histogram Backprojection is extremely efficient it is useful not only for locating an object but also for tracking an object moving relative to the robot.

Histogram Backprojection, like Histogram Intersection, is robust to occlusion. If instead of using figure 6 as the crowded scene, we use figure 12 in which only four-ninths of the image of each object remains, the location of each object can be found almost as well as in the image with no occlusion. They only target for which the effectiveness of Histogram Backprojection suffers badly is that of Charmin paper towels, the object that already had a portion occluded in the original image. As for the previous experiment, we show the backprojected ratio histogram for the blue and white striped shirt (see figure 18 on color page 32) and the combined results of looking for each of the objects in the image (see figure 19 on color page 32). In all cases but six the largest peak in the convolved image corresponds to the correct object. The six cases in which it doesn't are listed below in the format (*target: objects receiving larger response*).

1. Campbell's clam chowder: red and white shirt.
2. Manischewitz chicken soup: Manischewitz bakit, Manischewitz matzo farfel.
3. Angelsoft: Charmin.
4. Charmin: orange White Cloud, Bakit, Northern, purple White Cloud, Campbell's Special chicken soup, Manischewitz chicken soup.
5. Balloons shirt: white with pink border shirt.
6. Mickey Mouse underwear: red and white shirt.

The success rate is shown diagrammatically in table 8.

Table 8. Performance of Histogram Backprojection under occlusion. The number in each square is the rank of peak that falls into the corresponding square in figure 11 when looking for the model whose image is in that square.

1	1	1	1	1	1
1	1	2	1	1	3
1	2	7	1	1	1
2	1	1	1	1	1
1	2	1	1	1	1
1	1				

While the effects of occlusion are different for Histogram Intersection and Histogram Backprojection because the algorithms differ in how they process spatial information, the effects of changing image and histogram resolution are similar: Both algorithms will be successful if and only if the colors in the object stay in matching bins to the colors in the model. Since Histogram Intersection is very insensitive to changing the image or histogram resolution it is expected that Histogram Backprojection will be as well. Likewise, Histogram Backprojection is expected to be about as sensitive to failures of color constancy as Histogram Intersection.

5 Conclusion

The advent of real-time image-processing hardware is changing the research focus of computer vision in a fundamental way. Instead of attempting to build elaborate representations of the environment from static images, a new objective is to construct visual skills which allow a robot to interact with a dynamic, realistic environment. To achieve this objective, new kinds of vision algorithms need to be developed that are capable of running in real time and subserving the robot's goals. Two important skills for interacting with the environment are identifying an object in a known location and locating a known object. We have shown here how robust, extremely efficient algorithms for achieving these objectives can be designed using color histograms as their model and image representations. The robustness of the algorithms is directly related to their real-time performance. Since the algorithms function at or near video frame rates of 30 frames per second, they can fail on several frames per second and still achieve the overall goals of identification and location.

In the past research has concentrated on geometric cues. The shift toward real-time systems requires faster algorithms. For instance, in Rosenfeld's comprehensive 1989 bibliography, there are thirty-seven articles on recognition of three-dimensional objects, all of which use shape.[2] In contrast, there are no articles that use color for object identification. Color-based algorithms fulfill the requirement of such systems, because of their fast performance and capability of dealing with changes in view, object deformations, and inaccurate segmentation of objects from their backgrounds.

Because of color's important applications and ease of use, color cameras and digitizing facilities should be a feature of robotic systems that have to operate in typical human surroundings. There are applications of computer vision being considered in which color could play an important role. For instance, manufacturers of automated check-out devices in grocery stores are considering automating the identification of fruits and vegetables. Color would be an important identifying feature in this situation. As well, an aquarium is investigating installing equipment to automatically identify the fish swimming by a visitor to an aquarium. Again, the coloration of the fish could be an important identifying feature.

Color could also be used for vision systems in manufacturing environments provided the environment is color coded. Identification, location, and tracking in such an environment using color would be straightforward. Color may be the easiest way to label objects and locations for a robot. Robots could pick up the tools they need based on color, follow lines on the floor of various colors to go to specific destinations, dangerous objects could be color coded in certain ways, the boundary of the workspace could be denoted by a colored band. The possibilities are numerous.

One especially convenient aspect of manufacturing environments is that the lighting can often be carefully controlled. Consistent lighting would avoid the necessity of solving the color constancy problem. Lighting which can be described by a small set of basis functions is easier to discount than lighting which may be from a variety of sources with different spectra. Only one basis function is needed when the light is from only one type of bulb. Similarly, color constancy is easier to achieve if the surface reflectances come from a known distribution which can be described by a small number of basis functions. This objective may be more easily achieved in a constrained manufacturing environment than in an unconstrained environment.

5.1 Future Work

One problem that requires much more careful analysis is the appliction of color recognition under varying light conditions. In addition to the simple normalization scheme described in the text, Novak and Shafer's the "guided" color-constancy algorithm of Novak and Shafer (1990) is a good one to try, because the large numbers of constraints introduced by the color chart they place in the scene should give it better color constancy than the algorithms that deduce the lighting from less reliable information. We conjecture that if the matching techniques introduced here don't work under variable lighting using Novak and Shafer's algorithm, they probably won't work with any of the other algorithms.

The light source does not have to be beyond the control of the robot or vision system. For instance, a robot could carry its own light source with it. If this light source were significantly stronger than the other lights in the room, only the intensity would change from image to image. If the distance to the object were known, even the intensity could be calculated. Some time of flight-range laser-range sensors also generate reflectance images. If a collection of lasers at different wavelengths were used, the reflectance images could be analyzed just as the color images are here.

A second challenging problem is identifying the region from which to extract the histogram for histogram intersection. Histogram intersection is fairly insensitive to pixels in the background, nonetheless, since they can cause mismatches and since cropping a portion of the object will diminish the match value, delimiting the object of interest is an interesting problem. The answer to this problem will probably not be a single solution, but a number of different ones each of which works better under a different situation. For isolating a moving object, motion cues should be used; for isolating an object separated from its background by depth, disparity cues or cues from an active depth sensor should be used. The visual motion and disparity cues or cues from an active depth sensor should be used. The visual motion and disparity cues may only be reliable at sparse points on the surfaces, and so they must be either extended by surface models or enclosed within a bounding region. A simple technique for eliminating background pixels based on verging on the object of interest and ignoring those pixels that do not register in the two eyes was demonstrated by Swain (1990a). Other techniques such as desensitization to a commonly occuring background could be used.

The algorithms have not been tested in cooperation with a method for estimating the depth of the object. Provided the object is within a fairly small distance, let's say 5 meters, the problem should be solvable using standard techniques. In this range, vergence (stereo), laser-range finders, and focus can recover approximate depth. A demonstration of this capability would be interesting. Estimating depth at long distances is more problematic, and may only be possible if other objects at a similar depth are first recognized and provide scale.

It may be possible to use other surface properties besides color for identification and location. The most obvious one to try is texture. Instead of histogramming colors, outputs of nonoriented and oriented spatial filters could be historgrammed. Malik and Perona (1990) have demonstrated a scheme for finding texture boundaries based on the output of such filters, but they have not investigated how to recognize texture. There is one problem to be overcome in texture recognition that does not occur in color recognition, viz., that the directionality of the filters does not make this scheme naturally orientation invariant. One approach to the solution of this problem would be to extract direction invariant measures from the output of the filters; another would be to attempt to align the image and model before comparison of the histograms. One drawback to the use of texture in machine vision systems is the large amount of computation needed. For instance, Malik and Perona employ the output of 192 different filters. Since Malik and Perona were proposing a biological model, cost of implementation in image processing hardware was not an issue. There may be ways of economizing on the use of filters using approaches such as suggested by Freeman and Adelson (1990).

Is it possible that a histogram-style approach can work for shape recognition? Pigeons can recognize Charlie Brown pictures in a variety of positions, orientations, and scales (Hernstein 1982). They do not distinguish, however, between a correct Charlie Brown figure, and a "jumbled up" version where the figure has been cut in half, and the two halves rearranged. It is possible, therefore, that they are using some sort of histogram-like data structure for recognition which counts the local features that show up but does not consider their relative orientation.

The *sum* and *minimum* and *division* operations needed for Histogram Intersection and Histogram Backproduction could easily be implemented in neural hardware. Could it be possible that algorithms similar to these are used in the brain? There are wavelength-sensitive cells in monkey Visual Area 4 that have large receptive fields and which could be loosely described as histogram cells. On the other hand, the work by Treisman (1985) and others on preattentive ("pop-out") phenomena suggests that people may have trouble searching for a conjunction of colors, as is done in the Histogram Backproduction algorithm. Until now, there has been no work in how color is used for identification or location in biological systems. This article provides computational models whose presence could be explored in biological systems.

Acknowledgments

Lambert Wixson, Chris Brown, Randal Nelson, and the Rochester vision group were a source of influential discussions and ideas. Thanks to Ketan Mulmuley for help with the theorem in Appendix B. Ray Rimey and the reviewers made careful comments on the manuscript. This work was supported by NSF research grant DCR-8602958 and ONR research grant N00014-91-J-1185.

Appendix A: Relation to Pattern Recognition

Each of the different bins of the histogram can be considered a different feature, as is done in pattern recognition (Young & Fu 1986). This approach to recognition has been studied extensively, and so it is important to discuss Histogram Intersection in relation to the approaches used in this discipline.

In pattern recognition, the set of features are designated to be axes in a *feature space*, in which the object is defined to be a point $(f_1, \ldots, f_n)$. A *metric* is defined on the space, and identification is done by finding the nearest object in feature space to the set of features extracted from the image. Recall that *metric space* is defined as follows:

Definition 1: A set X, whose elements we shall call *points*, is said to be a *metric space* if with any two points p and q of X there is associated a real number $d(p, q)$, called the *distance* from p to q, such that

1. $d(p, q) > 0$ if $p \neq q$; $d(p, p) = 0$;
2. $d(p, q) = d(q, p)$;
3. $d(p, q) \leq d(p, r) + d(r, q)$ for any $r \in X$.

Any function with these three properties is called a *distance function*, or a *metric*.

When the image and model histograms are scaled to be the same size, as can be done when the object in the image can be segmented from its background, then using Histogram Intersction is equivalent to using the sum of absolute differences or *city-block* metric, as is shown below.

Theorem 1. *If*

$$\sum_{i=1}^{n} M_i = \sum_{i=1}^{n} I_i$$

then

$$1 - H(I, M) = \frac{1}{2n} \sum_{i=1}^{n} |I_i - M_i|$$

Proof. The key to the proof is the identity shown in equation (1). To derive this identity, we note that

$$I_i = \min(I_i, M_i) + |I_i - M_i| \quad \text{if } I_i > M_i$$

$$I_i = \min(I_i, M_i) \text{ if } I_i \leq M_i$$

and

$$M_i = \min(I_i, M_i) \qquad \text{if } I_i > M_i$$
$$M_i = \min(I_i, M_i) + |I_i - M_i| \qquad \text{if } I_i \leq M_i$$

In either case

$$I_i + M_i = 2\min(I_i, M_i) + |I_i - M_i| \tag{1}$$

The proof follows easily. Let

$$\sum_{i=1}^{n} M_i = \sum_{i=1}^{n} I_i = k$$

Then, using equation (1),

$$\begin{aligned} k &= \frac{1}{2}\sum_{i=1}^{n}(I_i + M_i) \\ &= \sum_{i=1}^{n}\min(I_i, M_i) + \frac{1}{2}\sum_{i=1}^{n}|I_i - M_i| \qquad (2) \end{aligned}$$

By definition,

$$1 - H(I, M) = \frac{k - \Sigma_{i=1}^{n}\min(I_i, M_i)}{k}$$

and so

$$1 - H(I, M) = \frac{k - \Sigma_{i=1}^{n}\min(I_i, M_i)}{k}$$

Replacing the k in the numerator by the expression in equation (2) we have

$$1 - H(I, \mathrm{M}) = \frac{1}{2k}\sum_{i=1}^{n}|I_i - M_i|$$

and the theorem is proven. Q.E.D.

If the model and image histograms do not contain the same number of pixels, that is, if

$$\sum_{i=1}^{n} M_i \neq \sum_{i=1}^{n} I_i$$

then the symmetry relation (axiom number 2) does not hold and Histogram Intersection is not a metric.

Appendix B: Representing a Large Database with Color Histograms

Consider the multidimensional space E defined by the bins of the histogram. That is, points in E are n-tuples $(c_1, c_2, \ldots, c_n)$ where n is the number of bins in the histogram, and c_1 is the count in the ith bin. We ignore the discrete nature of histograms obtained from discrete images, and assume that the c_i are continuous values in the range $[0, n]$. We assume that image histograms are scaled to contain the same number of counts I as the model histograms, and so Histogram Intersection is equivalent to the use of the *city-block* metric (see Appendix A).

Define a *city-block metric n-ball* to be the following n-dimensional geometric figure:

$$\sum_{i=1}^{n}|x_i| \leq 1 \tag{3}$$

The theorem we wish to prove relies on the following lemma.

LEMMA 2. *The volume of the intersection of a city-block metric n-ball of radius r and any n − 1 dimensional hyperplane through the origin is less than or equal to the area of the city-block n − 1 ball of radius r.*

Proof. From elementary multidimensional geometry we know that the intersection of the geometrical figure defined by equation (3) with an $n - 1$ dimensional hyperplane passing through the origin is of the form

$$\sum_{i=1}^{n-1}|a_i x_i'| \leq 1$$

where the x_i' are defined with respect to a natural orthonormal coordinate system for the hyperplane in which the nth axis is perpendicular to it. Using the triangle inequality we have for all points in the n-ball

$$\left(\sum_{i=1}^{n}|x_i'|^2\right) = \left(\sum_{i=1}^{n}|x_i|^2\right)^{1/2} \leq \sum_{i=1}^{n}|x_i| \leq 1$$

and so it follows that for all i, $|a_i| \geq 1$. Therefore, the intersection of the n-ball with the hyperplane could be contained within an $n - 1$ ball (in the coordinate system of the hyperplane), and so its is of smaller size. Q.E.D.

We can now show

THEOREM 2. *The fraction of the volume of E occupied by a single model is at most*

$$\frac{(2\delta)^{n-1}}{\sqrt{n}}$$

where $1 - \delta$ *is the minimum Histogram Intersection match value allowed and n is the number of bins in a histogram.*

Proof. The points in E for which

$$\sum_{i=0}^{n} c_i = I$$

form an $n - 1$ dimensional subset of E, which we will call P. We can find the $n - 1$ dimensional volume of P by differentiating the n-dimensional volume of the set V, in which

$$\sum_{i=0}^{n} c_i \leq I$$

By induction, it can be shown that the volume of V is

$$\mathcal{V}_V = \frac{I^n}{n!} \tag{4}$$

The volume of E is then

$$\mathcal{V}_P = \frac{(d/dI)\ \mathcal{V}_V(I)}{(d/dI\ \mathcal{D}_P(I)} \tag{5}$$

where $\mathcal{D}_P$ is the distance from the origin to P.

To understand this formula think of the numerator multiplied by δI as the differential change in volume of V and the denominator multiplied by δI as the differential width of the volume.

Since the closest point to the origin in P is $(I/n, I/n, \ldots, I/n)$, we have

$$\mathcal{D}_P = \frac{I}{\sqrt{n}} \tag{6}$$

Differentiating equations (4) and (6) we have

$$\frac{d}{dI}\ \mathcal{V}_V(I) = \frac{I^{n-1}}{(n-1)!}$$

and

$$\frac{\mathrm{d}}{dI}\ \mathcal{D}_P(I) = \frac{1}{\sqrt{n}}$$

Therefore, from (5),

$$\mathcal{V}_P = \frac{I^{n-1}\sqrt{n}}{(n-1)!} \tag{7}$$

We have found the volume of P. Now we need to find an upper bound on the volume occupied by a single model.

Under the city-block metric an n-ball has the shape of the region E in each quadrant. Since there are 2^n quadrants in n-dimensional Euclidean space, the volume of an n-ball of radius r is—using equation (4)

$$\frac{(2r)^n}{n!} \tag{8}$$

Using lemma 1 and equations (5) and (8), we have that the ratio $\mathcal{V}_m/\mathcal{V}_P$ of the volume occupied by a model and the total volume is bounded by

$$\frac{\mathcal{V}_m}{\mathcal{V}_P} \leq \frac{\dfrac{(2I\delta)^{n-1}}{(n-1)!}}{\dfrac{I^{n-1}\sqrt{n}}{(n-1)!}} = \frac{(2\delta)^{n-1}}{\sqrt{n}}$$

which is the required result. Q.E.D.

Notes

[1] A similar opinion is expressed by Ullman (1986): "For many objects color, texture, and motion play only a secondary role. In these cases, the objects are recognized by their shape properties. This is probably the most common and important aspect of visual recognition and therefore 'object recognition' is often taken to mean the visual recognition of objects based on their shape properties."

[2] Resenfeld's bibliographies are available by anonymous FTP from ADS.COM (in the VISION-LIST-ARCHIEVE directory).

References

Aloimonos, J. 1990. "Purposive and qualitative active vision." *Proc. Int. Conf. Pat. Rec.*, pp. 346–360.

Aloimonos, J., Weiss, I., and Bandyopadhay, A. 1988 "Active vision." *Intern. J. Comput. Vision* 1:436–440.

Bajcsy, R. 1985. "Active perception vs. passive perception," *Workshop on Computer Vision: Representation and Control,* pp. 55–59.

Bajcsy, R. 1988. "Active perception." *Proc. IEEE* 76:996–1005.

Ballard, D.H. 1987. "Interpolation coding: A representation for numbers in neural models." *Biological Cybernetics,* 57:389–402.

Ballard, D.H. 1989. "Reference frames for animate vision." *Intern. Joint Conf. Artif. Intell.*, pp. 1635–1641.

Ballard, D.H. 1991. "Animate vision." *Artificial Intelligence* 48:57–86.

Ballard, D.H., and Brown, C.M. 1982. *Computer Vision.* Prentice Hall: New York.

Biederman, I. 1985. "Human image understanding: Recent research and a theory." *Comput. Vision, Graph. Image Process.* 32(1):29–73.

Brainard, D.H., Wandell, B.A., and Cowan, W.B. 1989. "Black light: How sensors filter spectral variation of the illuminant." *IEEE Trans. Biomed. Engineer.* 36:140–149.

Chapman, D. 1990. "Vision, instruction, and action." Technical Report 1204, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, Cambridge, MA.

Coombs, D.J. 1989. "Tracking objects with eye movements." *Proc. Topical Meet. Image Understand. Mach. Vision.*

Dickmanns, E.D. 1988. "An integrated approach to feature based dynamic vision." *Proc. IEEE Conf. Comput. Vision and Patt. Recog.*, pp. 820–825.

Feldman, J.A. 1985. "Four frames suffice: A provisional model of vision and space." *Behav. Brain Sci.* 8:265–289.

Feldman, J.A., and Yakimovsky, Y. 1984. "Decision theory and artificial intelligence: I. A semantics-based region analyzer." *Artificial Intelligence* 5:349–371.

Forsyth, D.A. 1990. "A novel algorithm for color constancy." *Intern. J. Comput. Vision* 5:5–35.

Freeman, W.T., and Adelson, E.H. 1990. "Steerable filters for early vision, image analysis, and wavelet decomposition." *Proc. 3rd Intern. Conf. Comput. Vision*, Osaka, pp. 406–415.

Garvey, T.D. 1986. "Perceptual strategies for purposive vision." SRI International, Technical Note 117.

Hernstein, R.J. 1982. "Objects, categories, and discriminative stimuli." *Animal Cogn. Proc. Harry Frank Guggenheim Conf.*

Klinker, G.J., Shafer, S.A., and Kanade, T. 1988. "The measurement of highlights in color images." *Intern. J. Comput. Vision*, 2:7–32.

Koenderink, J.J., and van Doorn, A.J. 1976. "The singularities of the visual mapping." *Biological Cybernetics* 24:51–59.

Lennie, P., and D'Zmura, M. 1988. "Mechanisms of color vision." *CRC Crit. Rev. Neurobiol.* 3:333–400.

Malik, J., and Perona, P. 1990. "Preattentive texture discrimination with early vision mechanisms." *J. Opt. Soc. Amer. A* 7: 923–932.

Maloney, L.T., and Wandell, B. 1986. "Color constancy: A method for recovering surface spectral reflectance." *J. Opt. Soc. Amer. A* 3(1):29–33.

Maunsell, J.H.R., and Newsome, W.T. 1987. "Visual Processing in monkey extrastriate cortex." *Annu. Rev. Neurosci.* 10:363–401.

Mishkin, M., and Appenzeller, T. 1987. "The anatomy of memory." *Scientific American*, June, pp. 80–89.

Nelson, R.C. 1989. "Obstacle avoidance using flow field divergence." *IEEE Trans. Patt. Anal. Mach. Intell.* 11:1102–1106.

Nelson, R.C. 1991. "Qualitative detection of motion by a moving observer." In this issue.

Novak, C.L., and Shafer, S.A. 1990. "Supervised color constancy using a color chart." School of Computer Science, Carnegie Mellon University, Technical Report CUM-CS-90-140.

Ohlander, R., Price, K., and Reddy, D.R. 1978. "Picture segmentation using a recursive region splitting method." *Comput. Graph. Image Process.* 8:313–333.

Olson, T.J., and Coombs, D.J. 1991. "Real-time vergence control for binocular robots." In this issue.

Rubner, J., and Schulten, K. 1989. "A regularized approach to color constancy." *Biological Cybernetics* 61:29–36.

Strat, T.M. 1990, personal communication.

Swain, M.J. 1990a. "Color indexing." Department of Computer Science, University of Rochester, Technical report 360.

Swain, M.J. 1990b. "Companion videotape to 'color indexing' ".

Thompson, W.B. 1986. "Inexact vision." *Workshop on Motion, Representation, and Analysis*, pp. 15–22.

Treisman, A. 1985. "Preattentive processing in vision." *Comput. Vision, Graph. Image Process.* 31:156–177.

Ullman, S. 1986. "An approach to object recognition: Aligning pictorial descriptions." Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Technical Report 931.

Yarbus, A.L. 1967. *Eye Movements and Vision.* Plenum Press: New York.

Young, T.Y., and Fu, K.S. eds. 1986. *Handbook of Pattern Recognition and Image Processing.* Academic Press: San Diego, CA.

COLOR FIGURES

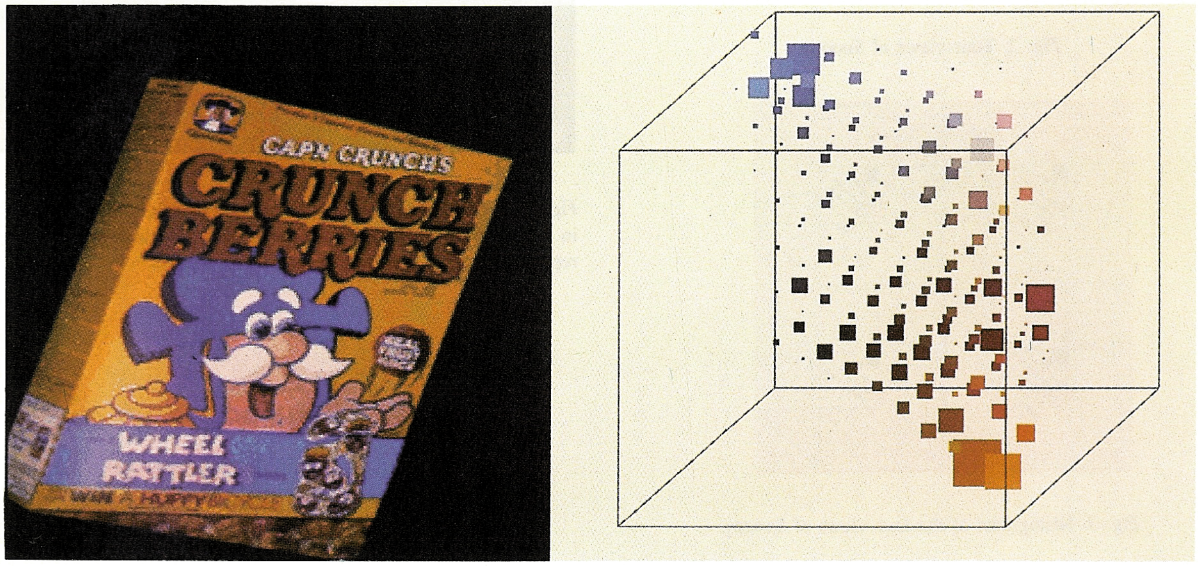

Fig. 1. Left: Image of a Crunchberries cereal box. Right: Three dimensional color histogram of the Crunchberries image with the black background substrated.

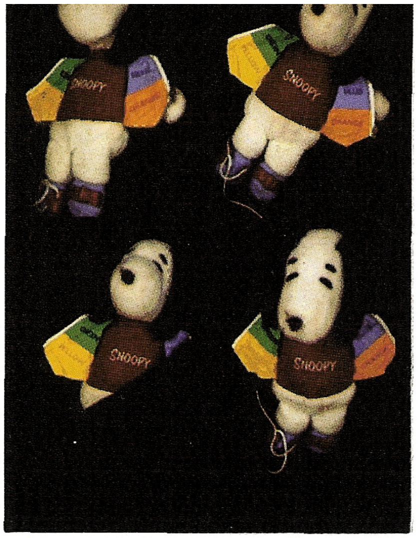

Fig. 2. Four views of Snoopy.

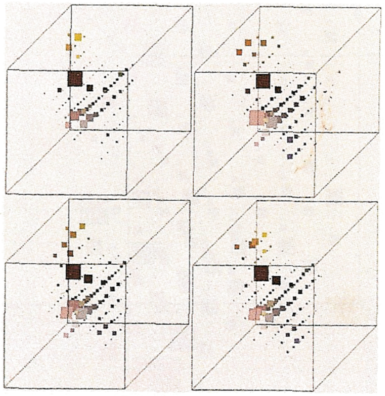

Fig. 3. Histograms of the four views of Snoopy.

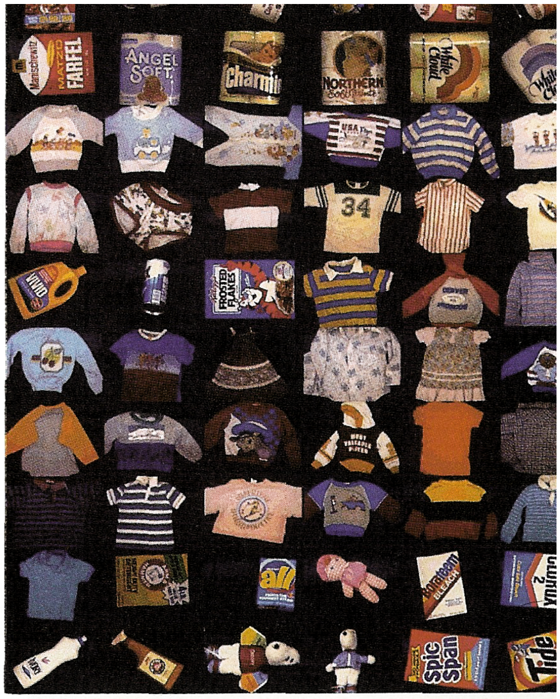

Fig. 4. Modeling indexing experiment based on color cues (continu
in figures 6 and 8). Each of the sixty-six models shown here
represented by its color histogram.

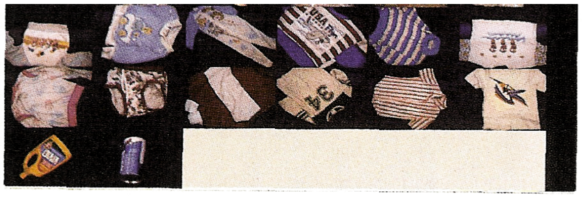

Fig. 5. The unknown objects. Each is identified with the model color histogram that best matches its own color histogram. Compared to the models the unknown objects are translated (Ajax), rotated about various axes (Frankenberry, Ajax) scaled (USA Flyer), occluded (Charmin), partly outside of the field of view (red, white striped shirt), and deformed (Mickey Mouse underwear).

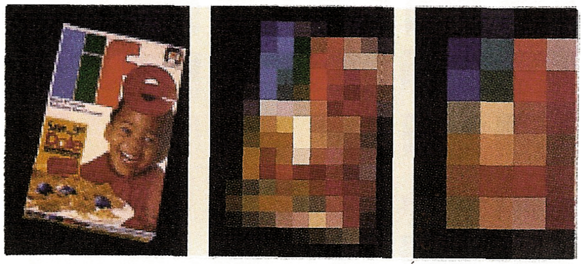

Fig. 6. Life cereal box image and reduced resolution copies. Left: 128×90 (1); Middle: 16×11 (2); Right 8×5 (30). The numbers in parenthesis indicate the rank of the match value for the Life cereal model. The middle image matches effectively, but the one on the right does not.

Fig. 10. Images from figure 5, each with the bottom third removed These images and the images below are used in the occlusion experiment (see table 4).

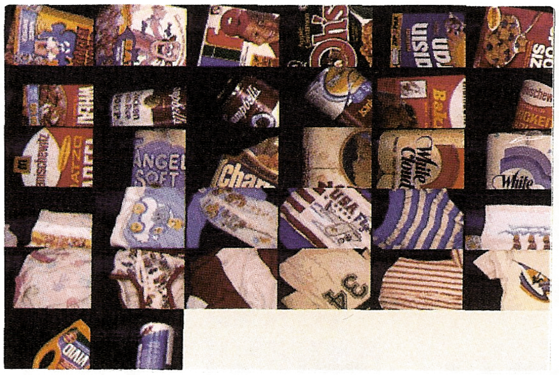

Fig. 11. Images from figure 10, each with the right-hand third removed. The upper left-hand corner (four ninths of the original image) is left.

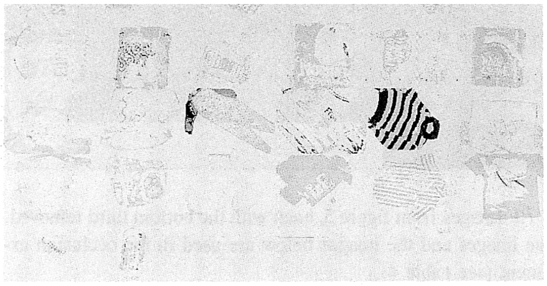

Fig. 16. Results of the ratio histogram backprojection step (2) of Histogram Backprojection, using figure 6 as the image and the striped blue and white shirt as the target. The blue hue is found in only a small area outside of the target, so it gives a strong response. White is found in many objects so it gives a weak response.

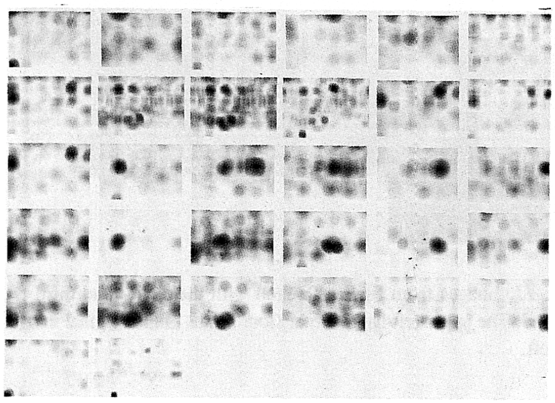

Fig. 17. Results of the convolution step (3) of Histogram Backprojection, for the same image as above. The results for all the models are shown in the image, each in the rectangle corresponding to the location of that model in the composite photo. When the algorithm successfully finds the object, the darkest black dot in the small image is in the same location within that image as the image in the component.

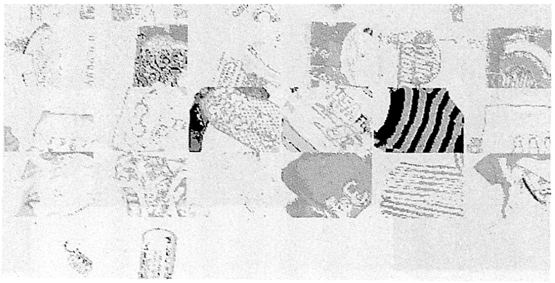

Fig. 18. Results of the ratio histogram backprojection step (2) Histogram Backprojection when the objects are occluded. Figure is the image and the striped blue and white shirt is the target.

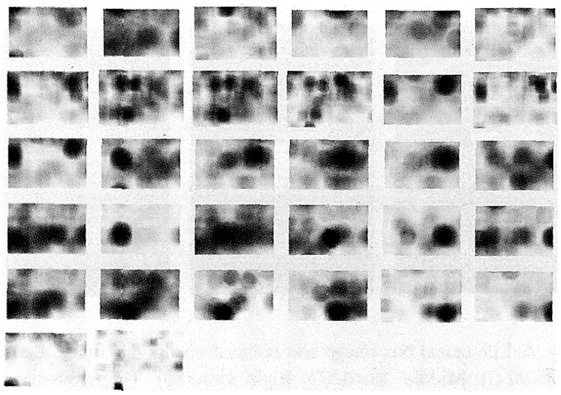

Fig. 19. Results of the convolution step (3) of Histogram Bac projection, for the same image as above. The results of all the mode are shown in the image, each in the rectangle corresponding to tl location of that model in the composite photo. When the algorith successfully finds the object, the darkest black dot in the small imag is in the same location within that image as the image is in the cor posite. Compare with figure 17.