Object Recognition with Color Cooccurrence Histograms

Peng Chang
The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
peng@ri.cmu.edu

John Krumm
Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
jckrumm@microsoft.com

Abstract

We use the color cooccurrence histogram (CH) for recognizing objects in images. The color CH keeps track of the number of pairs of certain colored pixels that occur at certain separation distances in image space. The color CH adds geometric information to the normal color histogram, which abstracts away all geometry. We compute model CHs based on images of known objects taken from different points of view. These model CHs are then matched to subregions in test images to find the object. By adjusting the number of colors and the number of distances used in the CH, we can adjust the tolerance of the algorithm to changes in lighting, viewpoint, and the flexibility of the object. We develop a mathematical model of the algorithm's false alarm probability and use this as a principled way of picking most of the algorithm's adjustable parameters. We demonstrate our algorithm on different objects, showing that it recognize objects in spite of confusing background clutter, partial occlusions, and flexing of the object.

1. Introduction

Object recognition in images is always based on a model of the object at some level of abstraction. This model is matched to an input image which has been abstracted to the same level as the model. At the lowest level of abstraction (no abstraction at all), an object can be modeled as a whole image and compared, pixel by pixel, against a raw input image. We can abstract away unimportant details by using subtemplates (ignorning background and image position), by using normalized correlation (ignoring illumination brightness), by using edge features (ignoring low spatial frequencies), *etc.* The abstraction itself is embodied in both the representation of the object and in the way it is matched to the abstracted image. For instance, Huttenlocher *et al.*[1] represent objects as simple edge points and then match with the Hausdorff distance. While the edge points form a completely rigid representation, the matching allows the points to move nonrigidly.

The dilemma comes in deciding how much to abstract away. We want to ignore just enough details of the object's appearance to match all anticipated images
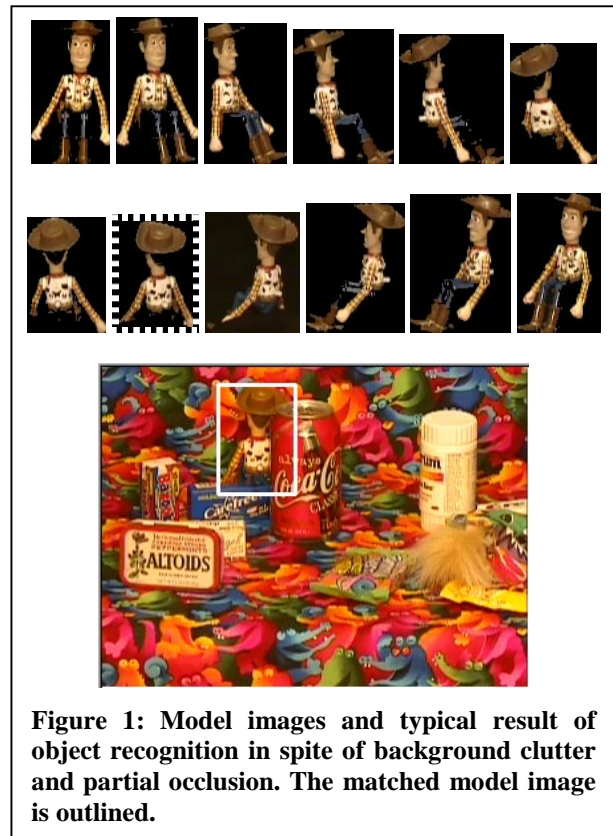

Figure 1: Model images and typical result of object recognition in spite of background clutter and partial occlusion. The matched model image is outlined.

of the object, but not so many details that the algorithm generates false matches.

One interesting dimension of abstraction is rigidity. Near one end of this dimension are the several object recognition algorithms that abstract objects into a rigid or semi-rigid geometric juxtaposition of image features. These include Hausdorff distance[1], geometric hashing[2], active blobs[3], and eigenimages[4, 5]. In contrast, histogram-based approaches abstract away (nearly) all geometric relationships between pixels. In pure histogram matching, *e.g.* Swain & Ballard[6], there is no preservation of geometry, just an accounting of the number of pixels of given colors. The technique of Funt & Finlayson[7] uses a histogram of the ratios of neighboring pixels, which introduces a slight amount of geometry into the representation.

Abstracting away rigidity is attractive, because it allows the algorithm to work on non-rigid objects and because it reduces the number of model images necessary to account for appearance changes due to scaling and viewpoint change. One can start with a geometrically rigid approach and abstract away some rigidity by using geometric invariants [8], loosening the matching criteria [1], or explicitly introducing flexibility into the model[3]. On the other hand, one can start with a simple and effective method like Swain & Ballard's color indexing[6], which ignores all geometry. Our approach is to introduce some geometric representation into the color histogram by using instead a histogram of the cooccurrences of color pixels.

The color cooccurrence histogram (CH) keeps track of the number of pairs of certain colored pixels that occur at certain separation distances in image space. By adjusting the distances over which we check cooccurrences, we can adjust the sensitivity of the algorithm to geometric changes in the object's appearance such as caused by viewpoint change or object flexing. The CH is also robust to partial occlusions, because we do not require that the image account for all the cooccurrences of the model. We show that the CH also performs well in spite of background clutter.

A short summary of our object recognition algorithm begins with Figure 1, where we show a series of model images of an object (Woody from the movie *Toy Story*) taken as the object was rotating on a turntable. We compute CHs from each of these images. To find the object in an input image, we extract overlapping subimages from the input image, compute the CH of each subimage, and match these to the CHs of each of the model images. A typical result is shown in the same figure.

The simplicity of the algorithm allows us to analyze it theoretically, and we devote a significant part of this paper to understanding the algorithm's false alarm probability. This gives us a principled way of choosing the algorithm's adjustable parameters. We believe this is the first theoretical false alarm analysis of histograms (both cooccurrence and regular) for recognizing objects.

The approach we discuss in this paper is similar to other histogram-based approaches, most of which are used to find images in a database rather than to find an object in an image. These approaches share an attempt to add spatial information to a regular color histogram. Huang *et al*.[9] use the "color correlogram" to search a database for similar images. The correlogram is essentially a normalized version of our CH. Pass and Zabih[10] use "color coherence vectors" that represent which image colors are part of relatively large regions of similar color.

We discuss the CH in the next section. We derive an approximation of the false alarm probability in Section

3 which helps us choose the size of the search window, the number of quantized colors, and the number of distances to consider. We discuss the algorithm and show results in Section 4.

2. Matching with Color Cooccurrence Histograms

We represent each model image as a color CH. The color CH holds the number of occurrences of pairs of color pixels $c_1 = (R_1, G_1, B_1)$ and $c_2 = (R_2, G_2, B_2)$ separated by a vector in the image plane $(\Delta x, \Delta y)$, as shown to the right. We can write the cooccurrence histogram symbolically as $CH(c_1, c_2, \Delta x, \Delta y)$. This is essentially the same as the cooccurrence matrix, introduced by Haralick *et al*.[11] for classifying terrain images. We call it a "histogram" rather than a "matrix", since "histogram" better conveys the fact that it is used to keep track of the numbers of some things. In fact for $(\Delta x, \Delta y) = (0,0)$, the color CH reduces to a regular color histogram (where $c_1 = c_2$).
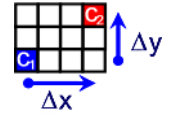
In order to make our CHs invariant to rotation in the image plane, we ignore the direction of $(\Delta x, \Delta y)$ and keep track of only the magnitude $d = \sqrt{(\Delta x)^2 + (\Delta y)^2}$. In addition, we quantize colors into a set of n_c representative colors $C = \{c_1, c_2, \ldots, c_{n_c}\}$, and we quantize the distances into a set of n_d distance ranges $D = \{[0,1), [1,2), \ldots, [n_d - 1, n_d)\}$. We quantize the colors of each model image separately using a standard nearest neighbor k-means algorithm, where the number of means is n_c and where we use Euclidian distance in (R, G, B). This quantization in color and distance means the CH is represented by $CH(i, j, k)$, where i and j index the two colors from set C, and k indexes the distance range from set D. One of the challenges that we address in Section 3 is how to choose the number of colors and distances by analyzing the false alarm rate.

In searching an image for an object, we scan the image for a rectangular window that gives a CH similar to one of the training CHs. The image windows from which the model and image CHs are computed need not be the same size. In fact we always make the image search window as large as we reasonably can to increase the speed of scanning the whole image. We address the search window size in Section 3 using our false alarm analysis.

The image and model CHs, $CH_p(i, j, k)$ and $CH_m(i, j, k)$ respectively, are compared by computing

their intersection, an idea introduced by Swain and Ballard[6]. The intersection is

$$I_{pm} = \sum_{i=1}^{n_c}\sum_{j=1}^{n_c}\sum_{k=1}^{n_d} \min[(CH_p(i,j,k), CH_m(i,j,k)] \quad (1)$$

The interesection indicates how well the image CH accounts for the model CH. If the image accounts for all the entries in the model CH, then the intersection will be equal to the sum of all the entries in the model CH, or I_{mm}. We declare the object to be found in the image if the intersection exceeds some threshold T. For instance, if we want the image to account for at least a fraction α of the cooccurrences in the model, we would set $T = \alpha I_{mm}$. The intersection is an attractive measure of similarity, because if the image CH contains extra entries from the background, they will not reduce the quality of the match. However, if the image CH has large values for all its entries, say due to examining a large region in the image, it could accidentally account for the entries in the model CH, causing a false alarm. We examine the effect of the search window size, as well as other parameters, on the false alarm probability in the next section.

3. False Alarm Probability for Choosing Algorithm Parameters

With enough trial-and-error experimentation, it is possible to discover reasonable values for the number of colors n_c, the number of distances n_d, and the search window size. However, this experimentation is tedious, and it gives only a vague sense of the sensitivity of the algorithm to changes in these parameters. To remedy this, we developed a mathematical model to approximate the false alarm probability and a numerical algorithm to approximate the mathematical model.

The false alarm probability is the probability that the intersection I_{pm} will exceed the match threshold αI_{mm} when the object is *not* actually in the search window. Intuitively, we expect the false alarm rate to increase as the search window grows, because there will be more opportunities for accidentally accounting for all the entries in the model CH. Also, we expect the false alarm probability to decrease as we increase the number of colors n_c and the number of distances n_d, because these increases lead to more specific models that are less likely to be accidentally accounted for by a random background. Pulling these parameters in the opposite direction is our desire for a speedy algorithm and for one that does not required too many model viewpoints of each object. The algorithm would run faster if the search window were larger, because the whole image could then be searched faster. The intersection computation would be faster with fewer colors and distances. Fewer distances would tend to generalize a

model of a particular viewpoint to match those of more disparate viewpoints, and it would match better with objects that flex. The false alarm probability helps us arbitrate between these desires, letting us set the parameters for the fastest possible execution and most tolerant matching without undue risk of false alarm.

The next subsection shows our derivation of the probability of a given CH occurring on a random background. We use this result in subsection 3.2 to compute the probability that a random background will match a model CH. Subsection 3.3 shows how we use the analysis to choose parameters for a real object.

3.1. Probability of a Given CH

We begin by computing the probability of occurrence of a given CH on a random background region of size $h \times w$. The image is quantized into n_c colors, with color i occurring with probability p_i. The cooccurrence histogram looks at pairs of colors, so we observe that the probability of seeing colors i and j at (x_i, y_i) and (x_j, y_j) respectively is $p_i p_j$. The CH would not distinguish the case where the two colors exchanged positions, so we note that the probability of colors i and j being at (x_i, y_i) and (x_j, y_j) respectively *or* (x_j, y_j) and (x_i, y_i) respectively is

$$p_{ij} = \begin{cases} 2p_i p_j \text{ if } i \neq j \\ \qquad\qquad\qquad i,j \in \{1,2,\ldots,n_c\} \\ p_i p_j \text{ if } i = j \end{cases} \quad (2)$$

For lack of any reason to believe otherwise at this stage, we assume that the n_c colors are uniformly distributed in the image background, giving $p_i = 1/n_c$. The probabilities of the unique pairs sum to one as we expect:

$$\sum_{i=1}^{n_c}\sum_{j=1}^{i} p_{ij} = 1 \quad (3)$$

Computing the probability of a given CH is complex, because there is no obvious way to break the probability into a series of independent probabilities that can be multiplied together. We will make an approximation that the probability of a juxtaposition of colors in one distance interval is independent of the probability in any other distance interval. In reality there is some correlation between distances. For instance, if we say many pairs of red pixels at one separation distance, we would expect to see many of the same color pairs at nearby separation distances.

For a given distance interval $[k-1, k)$, there are many different pairs of pixels separated by that amount in an $h \times w$ search window. For instance, the diagram at the right shows some of the
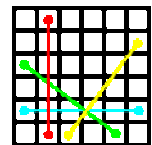
ways that a distance of five pixels can occur in a search window of size 6×6. We designate n_k as the number of ways that a distance interval $[k-1,k)$ can occur in the search window. We compute n_k with a simple Matlab program.

For a fixed distance interval $[k-1,k)$ there are $n_c(n_c+1)/2$ possible unique, nonordered color pairs with corresponding bins in the CH, each pair occurring with probability p_{ij}. Each bin contains n_{ij}^k counts ($n_{ij}^k = CH(i,j,k)$, $i \le j$), with $n_{ij}^k \ge 0$, $n_k = \sum_{i=1}^{n_c} \sum_{j=1}^{i} n_{ij}^k$, and $i, j \in \{1,2,\ldots,n_c\}$. The probability of a partition of particular color pairs into bins with n_{ij}^k in each bin is

$$p_{11}^{n_{11}^k} p_{12}^{n_{12}^k} \ldots p_{ij}^{n_{ij}^k} \ldots p_{n_c n_c}^{n_{n_c,n_c}^k}.$$ This partition can

happen in $\begin{pmatrix} n_k \\ n_{11}^k, n_{12}^k, \ldots, n_{ij}^k, \ldots, n_{n_c,n_c}^k \end{pmatrix}$ ways, each of

which produces an identical CH. Thus, the probability of a partition of the n_k color pairs into $n_c(n_c+1)/2$ bins with n_{ij}^k in each bin is given by the multinomial distribution:

$$f(n_{11}^k, n_{12}^k, \ldots, n_{ij}^k, \ldots, n_{n_c,n_c}^k) =$$

$$\begin{pmatrix} n_k \\ n_{11}^k, n_{12}^k, \ldots, n_{ij}^k, \ldots, n_{n_c,n_c}^k \end{pmatrix} * \quad (4)$$

$$p_{11}^{n_{11}^k} p_{12}^{n_{12}^k} \ldots p_{ij}^{n_{ij}^k} \ldots p_{n_c n_c}^{n_{n_c,n_c}^k}$$

where

$$\begin{pmatrix} n_k \\ n_{11}^k, n_{12}^k, \ldots, n_{ij}^k, \ldots, n_{n_c,n_c}^k \end{pmatrix} = \frac{n_k}{n_{11}^k! n_{12}^k! \ldots n_{ij}^k! \ldots n_{n_c,n_c}^k!}$$

We will abbreviate $f(n_{11}^k, n_{12}^k, \ldots, n_{ij}^k, \ldots, n_{n_c,n_c}^k)$ as $f(n_c, n_k)$.

$f(n_c, n_k)$ is the probability of observing a given set of cooccurrences in a distance range $[k-1,k)$. Our assumption says that these probabilities are independent with respect to k, so the probability of seeing a particular CH in the search window is

$$P(CH) = \prod_{k=1}^{nd} f(n_c, n_k) \quad (5)$$

3.2. False Alarm Probability Approximation

If the intersection of the model CH and the image region CH exceeds the threshold T without the object

being there, then this is a false alarm. Ideally, to compute the probability of a false alarm, we would make a list of all the CHs whose intersection with the model CH exceeds the threshold. We would then use Equation (5) to compute the probability of each CH in the list and sum the probabilities to get the false alarm probability. However, this list is much too long.

We instead approximate the multinomial distribution as a multivariate Gaussian distribution and integrate this Guassian. This simplifies the problem from summing values in an enormous list to integrating a multidimensional Gaussian.

From [12] we find that Equation (4) can be approximated by the ordinate of a multidimensional Gaussian:

$$f(n_c, n_k)$$

$$\approx g_k(\mathbf{n}) = \frac{1}{(2\pi)^{m/2} |\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{n}-\mu)^T \Sigma^{-1}(\mathbf{n}-\mu)\right] \quad (6)$$

where

$$m = n_c(n_c+1)/2 - 1$$

$$\mathbf{n} = \left(n_{11}^k, n_{12}^k, \ldots, n_{ij}^k, \ldots, n_{n_c-1,n_c}^k\right)^T \quad i \le j$$

$$\mu = n_k\left(p_{11}, p_{12}, \ldots, p_{ij}, \ldots, p_{n_c-1,n_c}\right)^T \quad i \le j$$

We note that the dimensionality of these vectors is one less than the number of color pairs $n_c(n_c+1)/2$. This is because once all but the last color cooccurrence bin has been filled, the last bin must contain exactly enough counts to make the total n_k.

To specify the inverse covariance Σ^{-1} we rename the cooccurrence probabilities: $q_1 = p_{11}$, $q_2 = p_{12}$, $\ldots, q_{n_c(n_c+1)/2-1} = p_{n_c-1,n_c}$, and $q^* = p_{n_c,n_c}$. Then the inverse covariance is

$$\Sigma^{-1} = \{a_{rs}\} = \begin{cases} \dfrac{1}{n_k}\left(\dfrac{1}{q_r} + \dfrac{1}{q^*}\right) & \text{if } r = s \\ \dfrac{1}{n_k q^*} & \text{if } r \ne s \end{cases} \quad (r,s) \in [1,2,\ldots,m]$$

The Gaussian $g_k(\mathbf{n})$ can be integrated to give probabilities of sets of CHs occurring in the image background. The integration limits are given in terms of the number of cooccurrences of particular color pairs. The Guassian only applies to a single specified distance range $[k-1,k)$.

We are still faced with the problem of listing all the CHs that could cause a false alarm. These would ideally be represented in terms of integration limits for the Guassian approximation. However, this list is too complex to characterize in this way, so we simplify our definition of a false alarm from "any background CH whose intersection with the model exceeds the threshold $T = \alpha I_{mm}$" to "any background CH, each of whose

entries n_{ij}^k exceeds a threshold $T_{ij}^k = \alpha m_{ij}^k$." Here $m_{ij}^k = I_m(i,j,k)$ is the model CH, $n_{ij}^k = CH_p(i,j,k)$ is the search window CH, and $0 < \alpha \le 1$ is the fraction of the histogram counts that we require to declare the model found in the image. This tends to underestimate the probability of a false alarm, but our experiments show that it is still approximately correct.

Integrating the Gaussian gives the probability of false alarm for all color pairs at one distance range $[k-1, k)$. From Equation (5), we consider cooccurrences at different distances to be independent. Thus we approximate the probability of a false alarm as

$$\text{P(false alarm)} \approx \prod_{k=1}^{n_d} \int_{\Omega_k} g_k(\mathbf{n}) \qquad (7)$$

where Ω_k is the integration region for CH distance range $[k-1, k)$. Ω_k is unbounded above and bounded below as $n_{11}^k \ge \alpha m_{11}^k, n_{12}^k \ge \alpha m_{12}^k, \ldots, n_{ij}^k \ge \alpha m_{ij}^k, \ldots, n_{n_c-1, n_c}^k \ge \alpha m_{n_c-1, n_c}^k$. We perform the actual integration using a Monte Carlo technique.

The next section shows plots of the false alarm probability to demonstrate how we used the analysis to pick some of the algorithm's operating parameters.

3.3. Using False Alarm Probability to Choose Parameters

Our algorithm requires the specification of three parameters: the number of colors n_c, the number of distances n_d, and the size of the search window $h \times w$. This section shows how we used our false alarm

analysis to guide our choice of these parameters for the first model image of Woody, shown in Figure 1. We show plots of the false alarm probability as the parameters of interest vary. In reality we had to iterate through the parameters and associated plots a few times to find a mutually good-working set of values.

We start by deciding the size of the search window. Setting $n_c = 8$ and $n_d = 12$, Figure 2 shows the effect of scaling the search window size from the original size of the model image (100×60) up to 5 times this size in each dimension. The false alarm rate increases as we expect, staying essential zero if the scaling factor is less than three. Thus our search window size is 300×180. The three curves in this figure also show how the false alarm rate decreases with increasing number of colors, as we expect.

We next choose the number of colors n_c that we use for the k-means color quantization. Using a search window size of 3 times the model window (as chosen above) and $n_d = 12$ distances, the false alarm probability as a function of n_c is shown in Figure 3. The false alarm probability decreases with more colors, because more colors make the model more specific and less prone to matching a random background. We chose $n_c = 8$ based on this plot, because that is where the false alarm rate first goes to nearly zero.

We created similar plots for the number of colors n_d which led us to choose $n_d = 12$.

This analysis gives a reasonable set of parameters to use for the first model image. Ideally we would go through a similar analysis for the model images of other
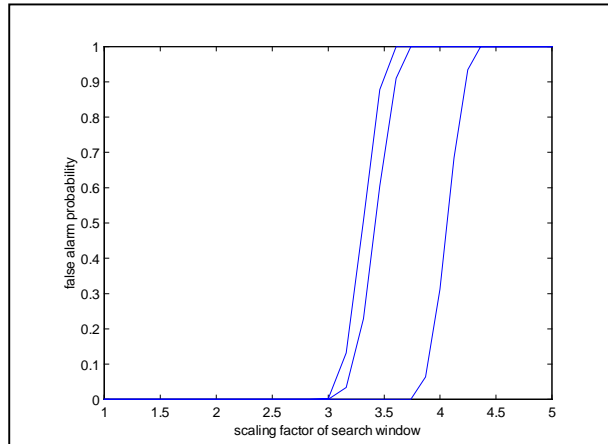

Figure 2: False alarm probability as a function of search window size. All curves are based on 12 distances. The left, middle, and right curves represent 6, 8, and 10 colors respectively. Based on this plot, we picked a search window scaling factor of 3.
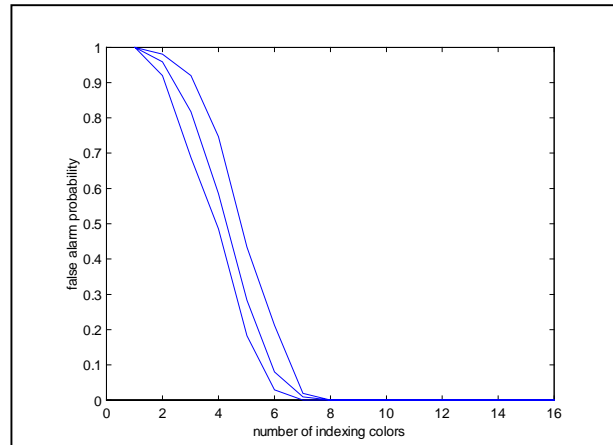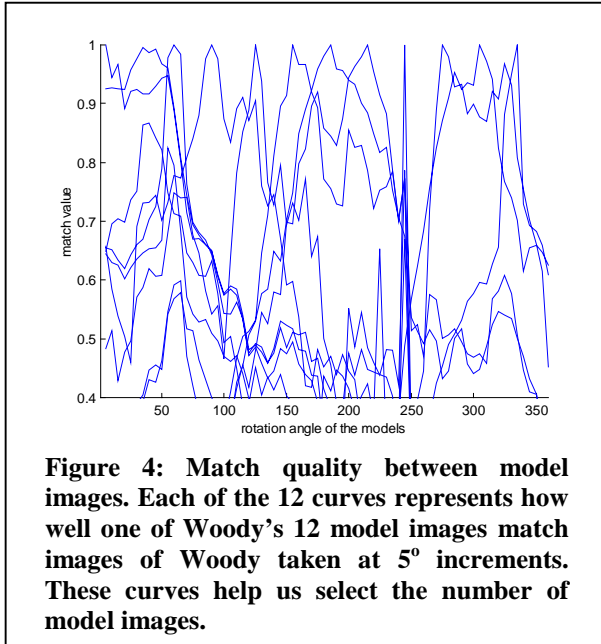

Figure 3: False alarm probability as a function of the number of colors. All curves are based on a window scaling factor of 3. The left, middle, and right curves represent 14, 12, and 10 distances respectively. We chose to use 8 colors based on this plot.

Figure 4: Match quality between model images. Each of the 12 curves represents how well one of Woody's 12 model images match images of Woody taken at 5° increments. These curves help us select the number of model images.

viewpoints of Woody. However, we used the same parameters for all our experiments with satisfactory results. It is interesting to note that for the autocorrelograms used in [9], the authors used 64 colors and 4 distances, compared to our 8 colors and 12 distances. In our application, using this many colors would likely make the algorithm miss many good matches due to unavoidable color shifts induced by interreflections with surrounding objects.

Although it falls outside of our false alarm analysis, another important number to choose for our algorithm is the number of image models to use for a given object. For Woody, we used 12 images to cover 360° of rotation. This choice was based on an analysis of how well the model images match themselves. We began by taking model images of Woody every 5°. Figure 4 shows how well the 12 model images we chose match with the other model images. As long as the 12 model images maintain a good degree of matching with the other model images, then the 12 are adequate for modeling the full set of viewpoints around 360°.

4. Algorithm and Results

We made image models by rotating the object on a motorized turntable. We chose a long focal length lens (75mm on a C-mount camera) that allowed us to put the camera far away from the objects (about eight meters). This reduced the effects of distance-induced scale change as we moved the objects around in the scene. We disabled the camera's automatic gain control and automatic white balance to help make the images consistent. We took model images with a black background to make segmentation easier. For test images, we used a bright, multi-colored background to make clutter. We found that interreflections from the

Figure 5: Result on Woody. Model images show Woody sitting with arms at his sides, but algorithm finds him standing with one arm outstretched.

Figure 6: Algorithm finds toothpaste box in spite of partial occlusion and background clutter.

Figure 7: Algorithm finds bottle of vitamins.

colored background caused a significant color shift on the objects, which our algorithm overcame by using a rough quantization of colors ($n_c = 8$).

While we only varied one angle of the object's point of view for testing, we could easily generalize to an arbitrary point of view using more model images. Given that we used 12 model images around the viewing circle, we would likely need about 12^2 model images around the viewing sphere. We could also generalize to more scales by using wider, overlapping distance ranges. For instance, if we designate the model images to have a scale factor of one, we could accommodate scales ranging from one to two in the test images by changing the distance set $D = \{[0,1],[1,2],\ldots,[n_d - 1, n_d)\}$ to $D^* = \{[0,2],[2,4],\ldots,[2(n_d - 1), 2n_d)\}$.

To find an object in a test image, we scan the image using each model image separately. For each model image, we first quantize it into n_c colors using the k-means nearest neighbor algorithm. We quantize the test image using the same n_c colors. We begin with a coarse scan for the object by extracting overlaping search windows from the test image. These windows are of the size we selected in our false alarm analysis, which for the Woody object is 300×180 , and they overlap by half. This reduces the search area for our pixel-by-pixel hill-climbing search. This high-resolution search starts at the center of the best window in the coarse search, centers a new search window on each of the four neighboring pixels, moves to the best one, and repeats until it reaches a local maximum.

Some typical results are shown in Figures 5, 6, and 7 and also in Figure 1. Figure 1 shows that the algorithm finds Woody despite significant occlusion. In Figure 5, the algorithm was able to find Woody standing with one arm outstretched, while the model images show him sitting with his arms at his sides. This demonstrates that the algorithm can tolerate some flexing in the object. In Figures 6 and 7 the algorithm has found a box of toothpaste and a bottle of vitamins despite partial occlusion and background clutter.

5. Conclusion

The color CH is an effective way to represent objects for recognition in images. By keeping track of pairs of pixels, it allows a variable amount of geometry to be added to the regular color histogram. We can analyze the algorithm to approximate its false alarm rate and use this to set some of the algorithm's adjustable parameters. Our results show that the algorithm works in spite of confusing background clutter and moderate amounts of occlusion and object flexing.

References

[1] D. P. Huttenlocher, G. A. Klanderman, and W. J. Ricklidge, "Comparing Images Using the Hausdorff Distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, pp. 850-863, 1993.

[2] Y. Lamdan and H. J. Wolfson, "Geometric Hashing: A General and Efficient Model-Based Recognition Scheme," presented at Second International Conference on Computer Vision, Tampa, Florida, 1988.

[3] S. Sclaroff and J. Isidoro, "Active Blobs," presented at Sixth International Conference on Computer Vision, Bombay, India, 1998.

[4] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience*, vol. 3, pp. 71-86, 1991.

[5] H. Murase and S. K. Nayar, "Visual Learning and Recognition of 3-D Objects from Appearance," *International Journal of Computer Vision*, vol. 14, pp. 5-24, 1995.

[6] M. J. Swain and D. H. Ballard, "Color Indexing," *International Journal of Computer Vision*, vol. 7, pp. 11-32, 1991.

[7] B. V. Funt and G. D. Finlayson, "Color Constant Color Indexing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, pp. 522-529, 1995.

[8] D. Forsyth, J. L. Mundy, A. Zisserman, C. Coelho, A. Heller, and C. Rothwell, "Invariant Descriptors for 3-D Object Recognition and Pose," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 971-991, 1991.

[9] J. Huang, S. R. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih, "Image Indexing Using Color Correlograms," presented at IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, 1997.

[10] G. Pass and R. Zabih, "Histogram Refinement for Content-Based Image Retrieval," presented at IEEE Workshop on Applications of Computer Vision, Sarasota, Florida, 1996.

[11] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-3, pp. 610-621, 1973.

[12] O. Kempthorne and L. Folks, in *Probability, Statistics, and Data Analysis*. Ames, Iowa: Iowa State Univeristy Press, 1971, pp. 108-121.