An Object Recognition System for Complex Imagery that Models the Probability of a False Positive

Clark F. Olson and Daniel P. Huttenlocher
Computer Science Department
Cornell University
Ithaca, NY 14853
clarko@cs.cornell.edu,dph@cs.cornell.edu

Abstract

This paper describes an object recognition system for use in complex imagery that can perform recognition adaptively by setting the matching threshold such that the probability of a false positive is low. In order to accurately model small, irregularly shaped objects, we represent the objects using dense sets of edge pixels with associated local orientations. Three-dimensional objects are modeled by a set of two-dimensional views of the object. We allow translation, rotation, and scaling of the views to approximate full three-dimensional motion of the object. We use a version Hausdorff measure to determine which positions of an object model are good matches to an image. These positions are determined efficiently through the examination of a hierarchical cell decomposition of the transformation space, which allows large volumes of the space to be pruned quickly. Additional techniques are used to decrease the computation time of the method when matching is performed against a catalog of object models. We then describe a new model of the matching process that allows the probability of a false positive to be estimated efficiently at run-time. Finally, we give results of this system recognizing objects in infrared and intensity images.

1 Introduction

This paper describes techniques to perform object recognition in complex images. Since we often wish to recognize objects that are small and irregularly shaped, it is important to capture as much of the structure of the objects as possible. We thus represent objects using sets of individual edge pixels describing the complete, two-dimensional boundary of the object (and possibly important interior edges), rather than approximating the model as straight edge segments. Images are represented similarly and matching is performed in this domain. While the use of edge maps implies the use of two-dimensional object models, we can recognize three-dimensional objects by representing each of the objects as a set of two-dimensional views of the object. Since we explicitly model translation, rotation in the plane, and scaling of the objects (i.e. similarity transformations), we approximate the full, six-dimensional, transformation space.

One problem that edge matching techniques can have is that images with considerable clutter can lead to a significant rate of false positives. We reduce this problem by considering, not only the location of each edge pixel, but, in addition, the local orientation at each of the edge pixels. This greatly reduces the rate at which false positives are found. An additional benefit of this information is that it helps to prune the search space and thus leads to improved running times.

A primary concern in this work has been the development of a model of the matching process that allows the probability of finding a false positive in an image to be efficiently estimated at run-time. To accomplish this, we model the matching process as a Markov process. This model allows the use of an adaptive algorithm, where the recognition threshold is set such that the probability of a false positive is small. Alternately, we can use this model to rank competing hypotheses.

In this paper, we first review a modified Hausdorff distance that incorporates both location and orientation information. This matching measure is used to determine which positions of the object model are good matches to an image. We then describe a search strategy by which we can efficiently determine the positions that satisfy this matching measure. Additional techniques that reduce the computation time when multiple object models are considered in the same image are also given. Next, we describe our model of the matching process and techniques by which we can estimate the probability of a false positive efficiently at run-time. Finally, we demonstrate the use of these techniques in infrared and intensity imagery and test the accuracy of the estimation model.

2 Matching measure

This section first reviews the definition of the Hausdorff measure and how a generalization of this measure can be used to decide which object model positions are good matches to an image. This generalization of the Hausdorff measure yields a method for comparing edge maps that is robust to object occlusion, image noise and clutter. A further generalization of the Hausdorff measure that can be applied to sets of oriented points is then described.

2.1 The Hausdorff measure

The directed Hausdorff measure from M to I, where M and I are point sets, is:

$$h(M, I) = \max_{m \in M} \min_{i \in I} \|m - i\|$$

where $\|\cdot\|$ is any norm. This yields the maximum distance among the points in M to their closest points in I. In the context of recognition, the Hausdorff measure is used to determined the quality of a match between an object model and an image. If M is the set of (transformed) object model pixels and I is the set of image edge pixels, the directed Hausdorff measure determines the distance of the worst matching object pixel to its closest image pixel. Of course, due to occlusion, it cannot be assumed that each object pixel appears in the image. The partial Hausdorff measure [4] between these sets is thus often used. It is given by:

$$h_K(M, I) = \underset{m \in M}{K^{\text{th}}} \min_{i \in I} \|m - i\| \tag{1}$$

This yields the Hausdorff measure among the K object pixels that are closest to image pixels. K can be set to the minimum number of object pixels that are expected to be found in the image if the object model is present or K can be set such that the probability of a false alarm occurring is small. Since this measure does not require that all of the pixels in the object model match the image closely, it is robust to partial occlusion. Furthermore, noise can be withstood by accepting models for which this measure is non-zero, and this measure is robust to clutter that may appear in the image, since it measures only the quality of the match from the model to the image and not vice versa.

Typically, we are interested in whether there exists a match with a size of at least K with Hausdorff measure below some threshold, δ. It is useful to conceptualize this as a set containment problem. Let $S_1 \oplus S_2$ denote the Minkowski sum of sets S_1 and S_2 (or dilation of S_1 by S_2). The statement $h(M, I) < \delta$ is equivalent to $M \subset (I \oplus E_\delta)$, where E_δ is a "disk" of radius δ centered at the origin in the appropriate L_p norm:

$$E_\delta = \{x \mid \|x\| \leq \delta\}.$$

Similarly, $h_K(M, I) < \delta$ and $|M \cap (I \oplus E_\delta)| \geq K$ are equivalent, where $|\cdot|$ denotes cardinality.

One method of determining whether a match of size K exists is to dilate the image pixels, I, by E_δ and probe the result at the location of each of the model pixels, M. Each time a probe hits a pixel in the dilated image, a match for a pixel in the object model has been found. A count on the number of these matches is kept. If the count surpasses K, then we have found a match which surpasses the threshold.

When there is a combination of a small set of object features and a complex image, this measure can yield a significant number of false alarms, particularly when the transformation space is large [6]. This problem can be solved, in part, by using local orientation information in addition to location information in determining the proximity between pixels in the transformed object model and the image.

2.2 The generalization to oriented points

The Hausdorff measure can be generalized to include the orientation of each of the points by representing each point as a vector in $\mathbb{R}^3$:

$$p = \begin{bmatrix} p_x \\ p_y \\ p_o \end{bmatrix}$$

where (p_x, p_y) is the location of the point and p_o is the orientation of the point (e.g. the direction of the gradient, edge normal or tangent). Typically, we are concerned with edge points on a pixel grid and the x and y values thus fall into discrete sets. The orientations can be mapped into a discrete set in a similar manner. Let's call a set of image points that have been extended in this fashion an *oriented image edge map*, I_o, and similarly, let's call such an extended set of points in the object model an *oriented model edge map*, M_o.

We now need a measure to determine how well these oriented edge maps match. Among pixels with the same orientation, we would like the measure to reduce to the previous Hausdorff measure. Furthermore, the previous measure should be a lower bound on the new measure. One measure that fulfills these conditions is:

$$h_\alpha(M, I) = \max_{m \in M} \min_{i \in I} \max \left\{ \left\| \begin{bmatrix} m_x - i_x \\ m_y - i_y \end{bmatrix} \right\|, \frac{|m_o - i_o|}{\alpha} \right\}$$

This has the same general form as the previous Hausdorff measure, but the distance between two points is now measured by taking the maximum of the distances in translation and orientation. In this measure, α is a normalization factor that makes the orientation values implicitly comparable with the location values. In practice, this allows the specification of a maximum deviation in translation and in orientation for two pixels to match, and thus a count of the number of model pixels that match image pixels according to both conditions can be kept. The parameters, α and δ, can be set arbitrarily to adjust the required proximities. A partial measure for oriented points that is robust to occlusion can also be formulated similar to Equation (1).

Our system discretizes orientations such that $\alpha = 1$ and uses the L_∞ norm. In this case, the measure for oriented points simplifies to:

$$h(M, I) = \max_{m \in M} \min_{i \in I} ||m - i||_\infty$$

3 Searching for matches

Recent work [4, 5, 6, 7, 8] has shown that we can search a discretized transformation space efficiently by using information from one possible transformation of the object model to rule out many other possible transformations. This section discusses the use of such techniques in this system and also describes techniques that can be used to reduce the running time when multiple models are considered.

3.1 Hierarchical cell decomposition

Let's first examine the case of unoriented pixels. Consider the distance transform of an edge image. This transform measures the distance from each location in the image to the nearest edge pixel according to some metric. If the distance transform at the location of a pixel in the transformed object model is large, then no close position of the object model can result in a match for this pixel.

We can use this fact to rule out many positions of an object model with the examination of only a single position in the distance transform. First, the image is dilated by a disk of radius δ and the distance transform of this dilated image is computed. We probe this distance transform at the transformed locations of the model pixels. If the Kth largest probe into this distance transform is zero, then we have found a match with a size of at least K. Otherwise, the Kth largest probe yields the distance to the closest possible position of the object model that could produce a match with a size of at least K. We can thus rule out any transformation that does not move any object pixel at least this distance.

To speed up the search, the transformation space is discretized, but to ensure that no good matches are missed, this discretization is such that two adjacent transformations move any object pixel by at most one pixel (Euclidean distance). Now, if d is the value of the Kth largest probe, we can rule out at least those transformations with a city-block distance (L_1 norm) less than d from the current transformation in the discretized transformation space, since such transformations are guaranteed to move each object pixel less than d pixels (Euclidean distance) from the current location and we use the L_∞ norm to measure distances in the image.

Now, since our oriented object and image points are represented by three parameters, we require a three-dimensional distance transform, but we must carefully consider how rotations of object models will be treated, since such rotations change the orientations of the object pixels. If we wish to rule out nearby transformations that change the orientations of object pixels, then this must be accounted for the distance transform, but this is problematic since the discretization of the rotations in the transformation space will, in general, be very different from the discretization of the orientations of the edge pixels. To avoid this problem, each rotation of an object model is treated independently (essentially as separate object models). This allows each orientation plane of the distance transform to be treated independently.

We must also consider how we will rotate and scale the models to compare them to the image. If we have a CAD model from which we determine the edges of our objects, we can rotate these models before performing the edge detection stage, since we treat different rotations of the model as (essentially) separate models. On the other hand, if our original model has been extracted from another image and consists only of a set of edge points, we simply rotate each point around the center of the model. Similarly, scaling of the model is performed by scaling each point with respect to the center of the model.

We can now use Hausdorff matching techniques similar to those for unoriented points to perform efficient recognition. This is accomplished by examining a hierarchical cell decomposition of the transformation space [5, 8]. We first discretize the transformation space as above and divide it into a set of rectilinear cells. Since the orientations are treated independently, the cells of the transformation space have three dimensions: scale, and translation in x and y. For each such cell, we consider the discrete transformation that is closest to the center of the cell. If the match at this transformation is poor enough that we can rule out the entire cell using the techniques described above, we prune the cell. Otherwise, we divide the cell into subcells and recursively consider each of the subcells. If we reach a cell containing only one transformation that we cannot rule out, then we have found a possible object location.

Let's consider the processing of a single cell. First, a discrete transformation close to the center of the cell is chosen and the maximum difference in the transformed location of a model pixel between the center transformation and any other transformation in the cell must be computed. This is given by the sum of the distance in the scale direction (by counting the number of discrete scales) between the transformations and the maximum of the distances in the x and y directions, since we use the L_∞ norm in the image space. The distance transform is then probed at the locations of the model pixels after transforming them by the transformation at the center of the cell. If the Kth largest probe into the distance transform is greater than the maximum distance any other transformation in the cell can move an object pixel from its current position, then the entire cell can be pruned. This is determined simply by counting the number of probes that yield a greater value than the computed distance. Otherwise, the cell is divided, into either two subcells by cutting at the midpoint of the range of scales in the cell, or into four subcells by cutting in both the x and y translations, based on whether the distance in scale is greater than the distance in translation in both x and y.

The examination of a single cell in the transformation space can be performed very quickly if some preprocessing is performed. The index into the array storing the distance transform can be computed in advance for each pixel of each model at every rotation and scale. For a particular translation, the pointer into the distance transform array need only be offset by a constant amount and these indexes can be used directly to probe at the locations of pixels of the object model.

3.2 Searching for multiple models

When there are multiple object models that we wish to search for in a single image, there are methods that can be used to make the search faster than examining each object model sequentially. This section describes one such method. Note that these object models need not come from separate objects, they may be alternate views of the same object.

The first step in this process is to determine a canonical position for each model with respect to the other models and to construct a hierarchical representation of the model set. This step is performed off-line, prior to recognition. For our multiple model search strategy, it is desirable to maximize the number of pixels between the edge maps of various models that overlap in their canonical position, in both position and orientation.

We first determine the best relative position between each pair of individual models

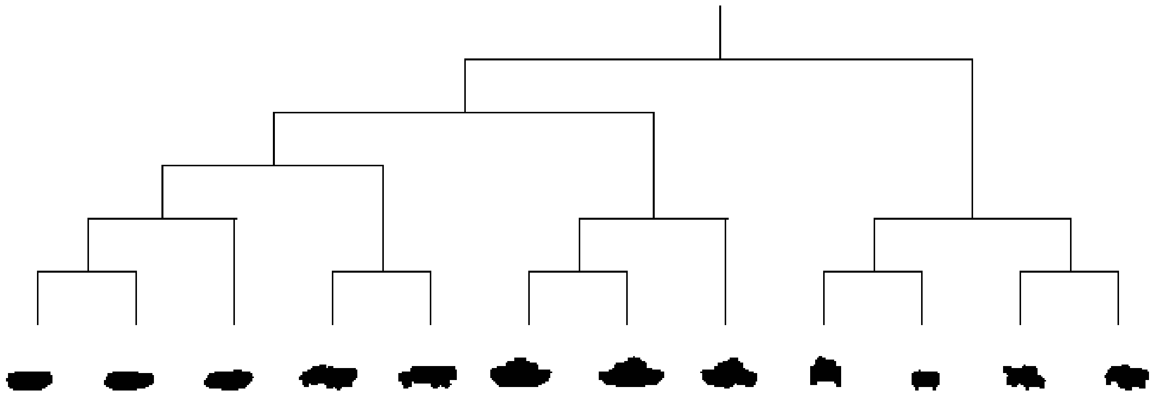


Figure 1: A hierarchical clustering of the models is performed as the canonical positions of the models relative to each other are determined. This figure shows an example of the hierarchy produced by these techniques for 12 model views. The full silhouettes are shown rather than the edge maps for visual purposes.

according to the chamfer measure [1] using search techniques similar those described above and in [6]. The chamfer measure sums the distances from each of the pixels in one image to their closest neighbors in the other. This measure is asymmetric, since the chamfer measure from some model, M_i, to another, M_j, is not necessarily the same as the reverse measure from M_j to M_i. A symmetric version is used that takes the maximum of the two measures. This measure is used as a score indicating how well each pair of models match.

We then build a tree of models using hierarchical clustering techniques [2]. At each step, the two closest models are determined and *agglomerated* into a single cluster. This yields a canonical position for these models with respect to each other and a new set of model points replacing the two previous models. We then compare the new 'model' to the remaining models as above and repeat the process until all of the models belong to a single hierarchically constructed model tree. At this point, canonical positions for each model with respect to the others have been computed and a model hierarchy represented by a binary tree has been determined, where the leaves of the tree are individual models and the remaining nodes correspond to the set of models below them in the tree. Figure 1 shows a small example.

It should be noted that this procedure can be time consuming if there are a large number of models, since the clustering procedure requires $O(M^2 \log M)$ time with a significant constant factor, where M is the number of model views. This is usually acceptable since this step is performed off-line. For very large model sets, there are a number of heuristics that can be used to reduce the time that this process requires.

For each node in the tree, we determine and store the model pixels that overlap at the canonical positions of all of the models below the node in the tree, except for those that are stored at ancestors of the node. We can now reduce the amount of redundant computation among the object models. At each transformation that is considered, we search the tree starting at the root, performing the probes for the model points that are stored at each node. A count on the number of probes that yield a distance greater than the distance to the edge of the cell in the transformation space is kept and propagated to the children of each node. If this count reaches a large enough value, we can prune this subtree of the model hierarchy for this cell of the transformation space and all of its subcells. This search is continued until we have pruned all of the object models or we have determined that we cannot prune all of the object models and we must therefore subdivide the cell or output a match.

4 Probability of a false positive

In this section, we consider the probability of a false positive occurring when matching using the matching measure described in the Section 2. We are primarily interested in methods by which this probability can be estimated efficiently at run-time and how this estimate can be used to improve the output of the recognition system.

4.1 A simple model for matching oriented pixels

Let's consider matching a single connected chain of oriented object pixels to the image at some position. For some pixel, p, in the object chain, we will say that it results in a hit if the transformed object pixel matches an image pixel in both location and orientation according to our measure. Otherwise, we will say that it results in a miss. If we map the object chain to a sequence of such hits and misses, this yields a stochastic process.

Note that if some pixel in the object chain maps to a hit, this means that, locally, the object chain aligns with an image chain closely in both location and orientation. It is thus likely that the next pixel will also map to a hit, since we match connected chains of such pixels. Let S_i be a random variable describing whether the ith object pixel is a hit or a miss and let s_i be the value taken by this variable for a specific object chain. If the probability of being in each state at each pixel is dependent only upon i and the previous state:

$$\Pr[S_i = s|(S_{i-1} = s_{i-1}) \wedge \ldots \wedge (S_0 = s_0)] =$$

$$\Pr[S_i = s|S_{i-1} = s_{i-1}]$$

then the process is said to be a *Markov process*. If, furthermore, the probability does not depend on i, then the process is a *Markov chain*. To determine the probability distribution of the number of hits over the entire object model, we must explicitly count j, the number of hits so far in our chain. We thus use a separate state in the chain for each member of:

$$\{hit, miss\} \times \{j \mid 0 \le j \le m\}$$

where m is the number of object pixels. When we are only interested in whether a false positive of size K occurs, we can use a Markov chain with $2K + 1$ states. See Figure 2. If we reach the final state of this chain due to matches with random edge chains in the image, then a false positive has occurred.

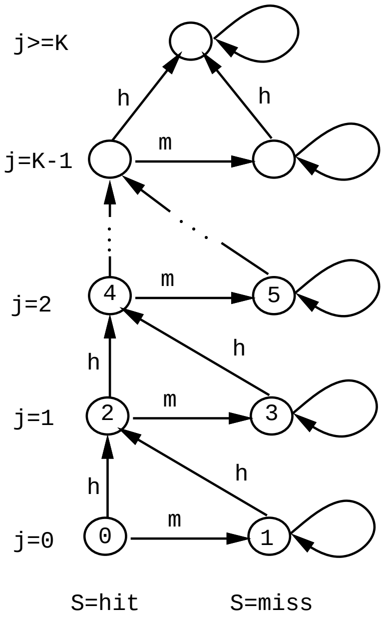


Figure 2: A Markov chain that counts the number of object pixels that match image pixels.

Let's number the states in the Markov chain as follows:

$$\begin{array}{ll}
0: & (S_i = hit) \wedge (j = 0) \\
1: & (S_i = miss) \wedge (j = 0) \\
2: & (S_i = hit) \wedge (j = 1) \\
3: & (S_i = miss) \wedge (j = 1) \\
 & \vdots \\
2k-2: & (S_i = hit) \wedge (j = K-1) \\
2k-1: & (S_i = miss) \wedge (j = K-1) \\
2k: & (j \geq K)
\end{array}$$

Abbreviate $P(S_i = h | S_{i-1} = m)$ as P_{mh}. We then have the following state transition matrix for the Markov chain in Figure 2:

$$\mathrm{T} = \begin{bmatrix}
0 & 0 & 0 & 0 & & 0 & 0 & 0 \\
P_{hm} & P_{mm} & 0 & 0 & & 0 & 0 & 0 \\
P_{hh} & P_{mh} & 0 & 0 & & 0 & 0 & 0 \\
0 & 0 & P_{hm} & P_{mm} & & 0 & 0 & 0 \\
0 & 0 & P_{hh} & P_{mh} & \cdots & 0 & 0 & 0 \\
 & & & \vdots & & & & \\
0 & 0 & 0 & 0 & & P_{hm} & P_{mm} & 0 \\
0 & 0 & 0 & 0 & & P_{hh} & P_{mh} & 1
\end{bmatrix}$$

Let p_0 be a vector containing the probability of the chain starting in each state. The probability distribution among the states after examining the entire object chain is:

$$p_m = T^m p_0$$

The last element of p_m is the probability that a false positive with a size of at least K will occur at this position of the model. We can determine the probability that a false positive of any other size, $K' \leq K$, will occur by summing the appropriate elements of p_m.

4.2 An accurate model for matching

If we want to model the matching process accurately, we cannot treat the state transition probabilities as independent of which pixel in the chain we are examining. Consider the probability of a hit following another hit for two cases. In the first case, the two object pixels have the same orientation and lie along the line perpendicular to the gradient. In the second case, there is a significant change in the orientation or the line between the pixels is not perpendicular to the gradient. The first case has a significantly higher probability of the second pixel resulting in a hit given that the first pixel was a hit, since we expect the chain of image pixels to continue in the direction perpendicular to the gradient with approximately the same gradient direction.

This means that the stochastic process of pixel hits and misses is not a Markov chain, but is is still a Markov process. Let T_i be the state transition matrix for the ith object pixel in such a process. The state probability vector, p_m, is now given by:

$$p_m = \left(\prod_{i=0}^{m-1} T_i \right) p_0 \tag{2}$$

Furthermore, we should not treat all hits the same. In the Hausdorff measure, an image pixel may match more than one pixel in an object chain, since we dilate the image prior to matching. This causes an effect such that after a pixel in the object chain hits a pixel in the oriented image edge map, the following pixels in the object chain are likely to hit the same image pixel, especially if there is no orientation change between the object pixels. This effect dies off after a few pixels, but it means that the probability of an object pixel resulting in a hit is not dependent upon only the previous state. We can still use a Markov process, if we encode the necessary information in the states of the process. If we use $\delta = 1$, which is sufficient for most applications, then the following states can be used:

- m: the object pixel did not hit an image pixel.
- n: the object pixel hit a new pixel in the oriented image edge map.
- o: the object pixel hit the same pixel in the oriented image edge map as the previous object pixel.

- p: the object pixel hit the same pixel in the oriented image edge map as the previous two object pixels.

It is possible for an object pixel to hit both a new pixel and an old pixel. In this case, state n takes precedence. To determine the probability distribution over the number of hits, we must now use a Markov process that consists of the cross-product of these states with the count of the number of hits so far:

$$\{m, n, o, p\} \times \{j \mid 0 \leq j \leq K\}$$

Empirical results indicate that this model of the matching process is sufficient to achieve accurate results in determining the probability of a false positive at a single transformation of the object model, if accurate estimates for the transition probabilities are used.

4.3 State transition probabilities

The state transition probabilities must now be determined. These probabilities will be different in locations of the image that have different densities of edge pixels. Consider, for example, the probability of hitting a new pixel following a miss. The probability will be much higher if the location is dense with edge pixels rather than having few edge pixels. To model this, let's consider the window of the image that the object model overlays at some position. This is simply the rectangular subimage covered by the object model at this position. Each of these windows in the image will enclose some number, d, of image pixels, which we'll call the density. The state transition probabilities are closely approximated by linear functions of the number of edge pixels present in the image window and belong to one of two classes:

1. Probabilities that are linear functions passing through the origin (i.e. $\Pr = k_i d$). The probability of an object model pixel hitting a new image pixel, when the previous object model pixel did not hit a new pixel, is well approximated by such a function. We thus model the following state transition probabilities in this manner: $P_{mn}(i)$, $P_{on}(i)$, and $P_{pn}(i)$. Note that each has a different constant, k_i.

2. Probabilities that are constant (i.e. $\Pr = c_i$). When the previous object model pixel hit an image pixel, the probability that the current object model pixel will hit the same image pixel is essentially constant. In addition, when the object model chain is following an image chain (i.e. the previous object model pixel hit a new image pixel), the probability that the object model chain continues to follow the image chain is approximately constant. The state transitions that are modeled in this manner are thus: $P_{op}(i)$, $P_{no}(i)$, and $P_{nn}(i)$.

These probabilities are determined by sampling possible positions of the object model and comparing the object model to the image at these positions. This is performed by examining the pixels of the object model chain, in order, and determining whether each object model pixel hits an image pixel or not, and if so, whether the previous object model pixel(s) hit the same image pixel. In addition, for each case we record what the next state is. The appropriate constant, given by $c_i = \Pr(i)$ or $k_i = \frac{\Pr(i)}{d}$, is then averaged over each of the positions to estimate the correct value of the constant.

The remaining probabilities can be determined as a function of these probabilities as follows:

$$P_{nm}(i) = 1 - P_{no}(i) - P_{nn}(i)$$
$$P_{om}(i) = 1 - P_{on}(i) - P_{op}(i)$$
$$P_{pm}(i) = 1 - P_{pn}(i)$$
$$P_{mm}(i) = 1 - P_{mn}(i)$$

If we consider the state at $i = 0$ to be m, this will yield the correct result for the first pixel in the object chain (i.e. $i = 1$). In this case, there are no previous object model pixels to compare against and we simply want the probability of an object pixel resulting in a hit at random. Similarly, if the object model consists of more than one chain of pixels, we simply reset the state to m when we start a new chain.

4.4 Probability of a false positive over a set of transformations

Let's now consider the probability that there exists a false positive at any translation of the object model. As with the search strategy, only translations on the integer grid are considered. While this may miss the optimal translation for our matching measure, this can increase the size of the minimum Hausdorff measure over the space of possible translations by at most $\frac{1}{2}$ when using the L_∞ norm.

While the probability that a false positive occurs at some transformation is not independent of whether a false positive occurs at a close transformation, previous work [3] has indicated that approximating these events as independent yields accurate results. We will thus treat these events as if they are independent here and check the performance of the model on real data to ensure that this assumption is realistic.

We do not assume that an object will be consistently either brighter or darker than the background, but we do assume that each object in the image is uniformly one or the other. This means that each transformation must be considered twice, once for the case when the object is brighter than the background and once for the case when the object is darker, since the orientation of the gradient in these two cases will be shifted by π. If $P_K(t)$ is the probability of a false positive of size K at transformation t, we can determine the probability of a false positive existing over all transformations as follows:

$$1 - \prod_t (1 - P_K(t))$$

This can be computed more efficiently if we determine the pixel density of each image window and use the histogram of these values, since the probability of a false positive at two positions with the same density is the same. Let d_i be the number of windows with density i, for $0 \leq i \leq W$, where W is the size of the window in pixels, and let $P_K(i)$ be the probability of a false positive of size K in a window with density i. The probability of a false positive is now given by:

$$1 - \prod_{i=0}^{W} (1 - P_K(i))^{d_i} \tag{3}$$

To estimate the probability of a false alarm when scaled and rotated versions of the object models are allowed in the matching process, the discretization of the transformation space must be considered. Rotating and scaling the object model does not move every pixel a uniform distance as translation does, but discrete rotations and scales can be considered such that two adjacent transformations move the farthest moving object pixel by no more than one pixel in the image (Euclidean distance), as in the search strategy. If the probability that each of these transformations yields a false positive is treated as being independent of the others, an estimate of the probability of a false alarm can be obtained over all similarity transformations by sampling the state transition probabilities over the possible translations, scales, and rotations of the object model and following the above equations.

The overall steps in the estimation of the probability of a false positive are as follows. First, we sample possible locations of the object model in the image to estimate the probabilities in the state transition matrices, T_i, as a function of the density of the image window. We also compute a histogram of the edge densities for each image window using dynamic programming. For each density, we then estimate the probability of a false positive occurring at a window with that density by computing Equation (2). Equation (3) is used to estimate the probability of a false positive occurring over the entire image. To improve the speed of this process, we consider only every tenth density value in the histogram and perform interpolation to estimate the remaining values.

The expected number of false alarms can also be estimated, if desired, as follows:

$$E(N_F) = \sum_{i=0}^{W} d_i P_K(i)$$

In addition, we can estimate the *a priori* probability that any particular image window yields a false positive by examining the result of Equation (2) for the density of that image window.

4.5 Using the probability estimate

Now that we have a method to estimate the probability of a false positive for any particular matching threshold, let's consider how we should use this estimate to improve the output of a recognition system that matches dense, oriented edge pixels.

We can use an adaptive algorithm for recognition by setting the matching threshold such that the probability of a false positive is below some selected probability. This ensures that when we find a match it is likely to be a correct match. In some cases, though, this may

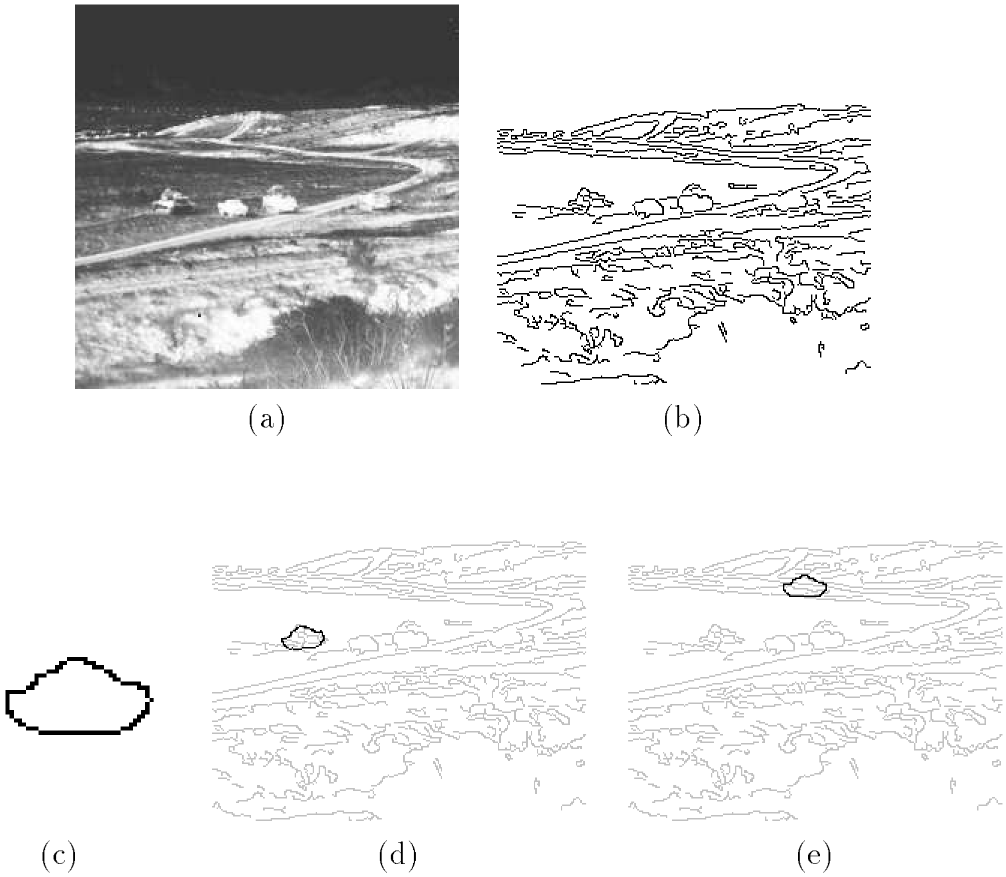


Figure 3: Automatic target recognition example. (a) A FLIR image after histogram equalization. (b) The edges found in the image. (c) The smoothed edges of a tank model. (d) The detected position of the tank. (e) A false alarm.

cause us to miss true instances of objects that we wish to detect, particularly when the image clutter is significant.

Alternatively, we can set our matching threshold such that we expect to detect most or all of the true object instances that are present in the image. In this case, our estimation model allows us to determine the likelihood that each hypothesis that we find is a false positive by considering the *a priori* probability that the image window of the hypothesis yields a false positive as described above. These likelihoods can be used to rank the hypotheses.

5 Results

Figure 3 shows an example of the use of these recognition techniques. The image is a low contrast infrared image of an outdoor terrain scene. After histogram equalization, a tank can be seen in the left-center of the image, although, due to the low contrast, the edges of

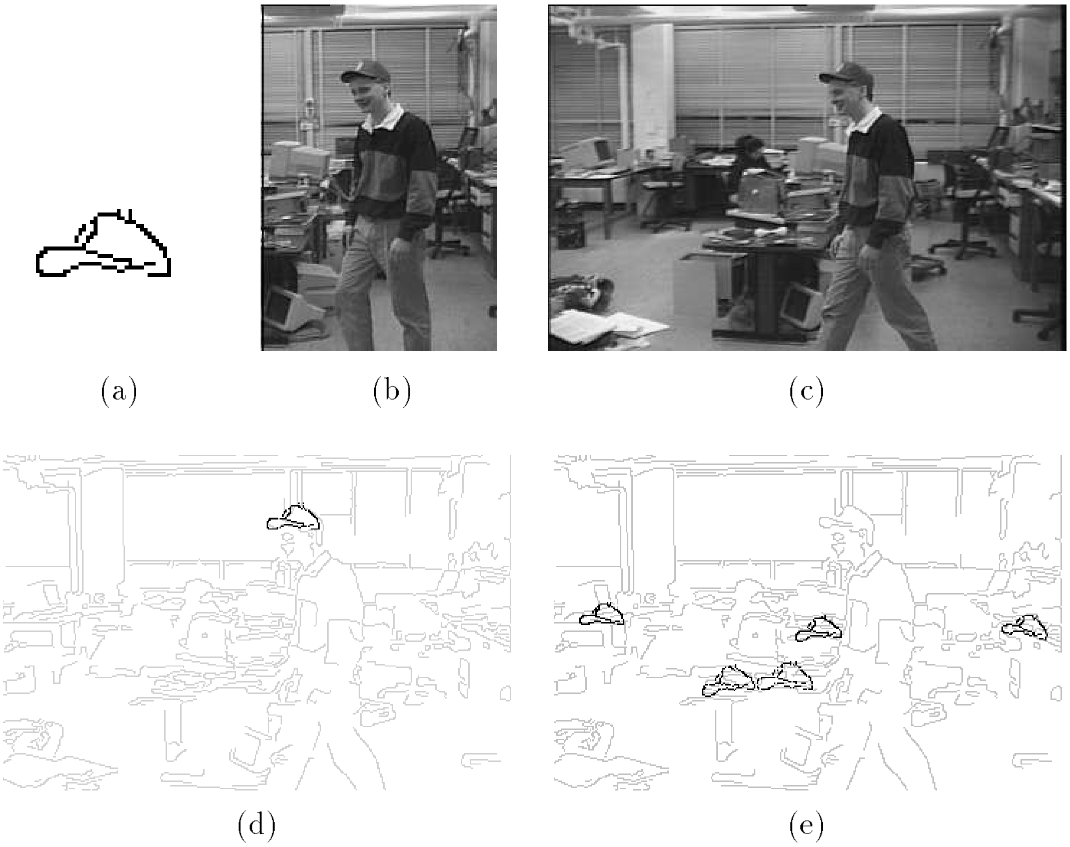


Figure 4: Image sequence example. (a) The object model. (b) Part of the image frame that the model was extracted from. (c) The image frame in we are searching for the model. (d) The position of the model located using orientation information. No false alarms were found for this case. (e) Several false alarms that were found when orientation information was not used. These each yielded a higher score than the correct position of the model.

the tank are not clearly detected. Despite the mediocre edge image, a large match was found at the correct location of the tank. It should be noted, however, that this was not the only match reported. Figure 3 also shows a false positive that was found. The image window for this false positive is more dense with edge pixels than the correct location. We can use the estimation techniques to rank these hypotheses based on their likelihood of being a false positive.

This implementation uses 16 discrete orientations and $\delta = \alpha = 1$ (each discrete orientation thus corresponds to $\frac{\pi}{8}$ radians, but matches are also allowed with neighboring orientations). In these experiments, we limited the allowable orientation and scale change of the object views to $\pm\frac{\pi}{18}$ and $\pm 10\%$, respectively, since we expect to have prior knowledge of the approximate range and orientation of the object in this application.

These techniques are not limited to automatic target recognition. Figure 4 shows an

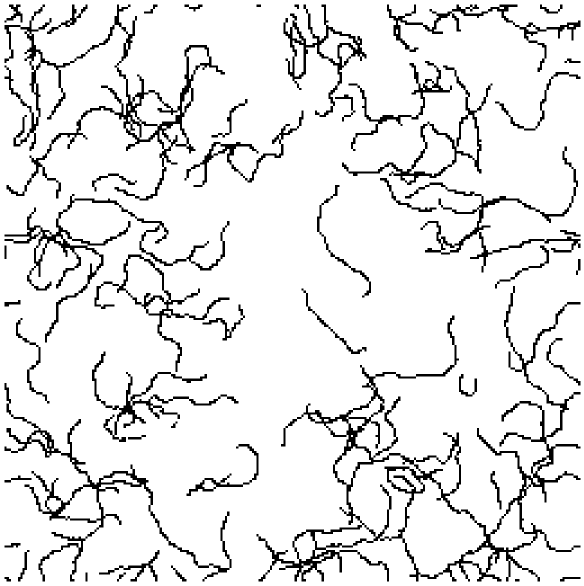
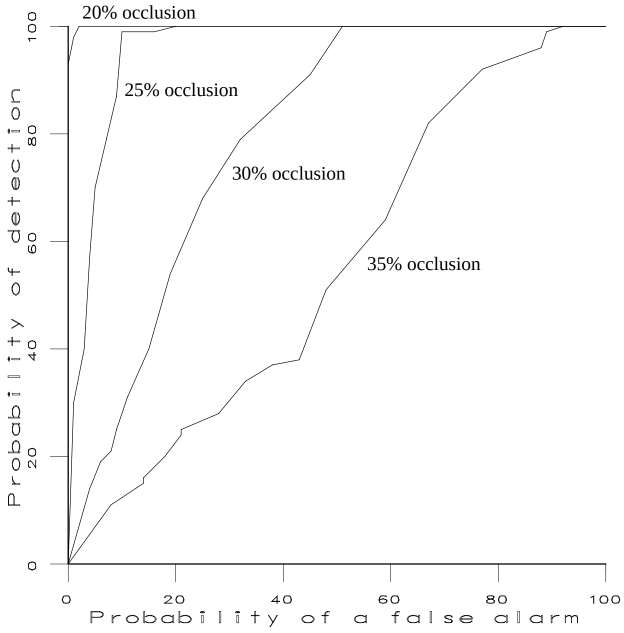


Figure 5: A synthetic image used to generate ROC curves.

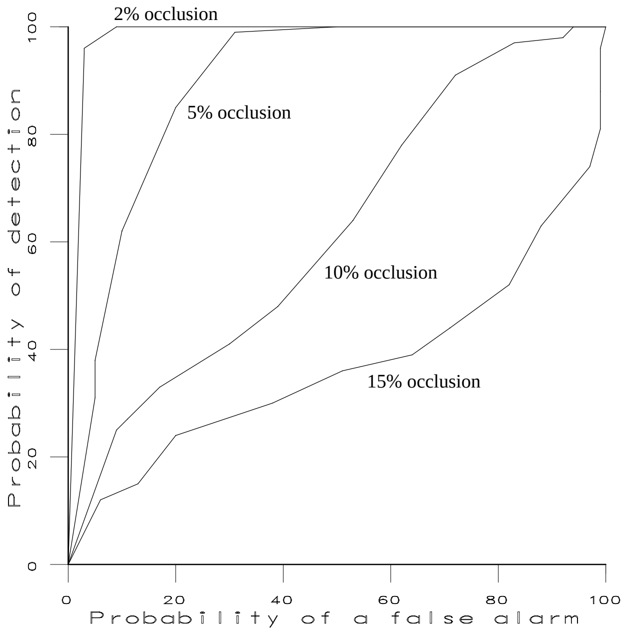
example of the use of these techniques in a complex indoor scene. In this case, the object model was extracted from a frame in an image sequence and matching is performed against a later frame in the sequence (as in tracking applications). The position of the object was correctly located when orientation information was used. No false alarms were found for this case. When orientation information was not used, several positions of the object were found that yielded a better score than the correct position of the object.

We have generated ROC curves for these techniques using synthetic edge images. Each synthetic edge image was generated with 10% of the pixels filled with random image clutter (curved chains of connected pixels). An instance of an object was placed in each image with varying levels of occlusion generated by removing a connected segment of the object boundary. Random Gaussian noise was added to the locations of the pixels corresponding to the object. Figure 5 gives an example of such a synthetic image. Figure 6 shows ROC curves generated for cases when orientation information was used and when it was not. These ROC curves show the probability that the object was located versus the probability that a false alarm of this object model was reported for varying levels of the matching threshold. When orientation information was used, the performance of the system was very good in these images up to 25% occlusion of the object . On the other hand, when orientation information was not used, the performance degraded significantly before 10% occlusion of the object was reached.

The estimation techniques were tested on real imagery. In these tests, the largest threshold at which a false positive was found was determined for each object model and image in a test set. In addition, the estimation techniques were used to determine the probability that



(a)



(b)

Figure 6: Receiver operating characteristic (ROC) curves generated using synthetic data. (a) ROC curves when using orientation information. (b) ROC curves when not using orientation information.

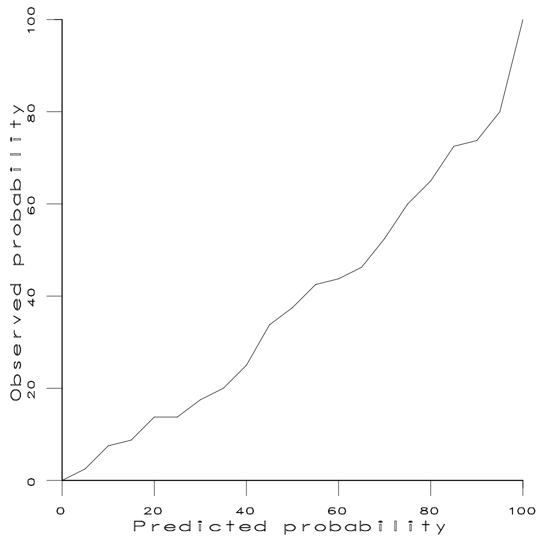


Figure 7: Predicted probability of a false alarm versus observed probability of a false alarm in trials using real images.

a false positive of at least this size would be determined in each case. From this information, we can obtain the observed probability of a false positive when the matching threshold is set to yield any predicted probability of a false positive by determining the fraction of tests that yielded a false positive with the matching threshold set to yield the predicted rate. See Figure 7. In the ideal case this would yield a straight line between (0,0) and (100,100). Since the plot that was produced by these tests lies slightly below this line for the most part, the estimation techniques described here predict false positives that are slightly larger than those actually observed in these tests, but the prediction performance is otherwise quite good.

The time required by the system is low. The preprocessing stage requires approximately 7 seconds on a Sparc-5 for a 256×256 image. This stage performs the edge detection on the image, creates and dilates the oriented image edge map, and computes the distance transform on each orientation plane of the oriented image edge map. This step is performed only once per image. The running time per object view varies with the size of the object model and the matching threshold used, but we have observed times ranging from 0.5 seconds to 4.5 seconds. See Table 1 for example times and counts on the number of transformations that were probed in each case. The prediction stage required approximately an additional 1.0 second per model to estimate the false alarm rate.

In addition to reducing the false alarm rate, the use of orientation information has significantly improved the speed of matching. Table 1 indicates that, in a small sample of the trials, the search time is reduced by approximately a factor of 10 when everything else was held constant. The techniques to reduce the search time when multiple models were consid-

			Using orientations			No orientations		
	Points	*Thresh*	*Probes*	*Time*	*Biggest*	*Probes*	*Time*	*Biggest*
Sample	67	53	122K	1.1s	63	2263K	11.0s	67
FLIR	67	60	49K	0.5s	62	1367K	5.9s	67
images	95	60	318K	4.5s	65	4396K	34.6s	95
	95	76	83K	1.1s	-†	2383K	17.3s	95
Int. Image	123	98	78K	1.3s	99	1832K	17.2s	120

† No match was found surpassing the threshold for this case.

Table 1: Performance comparison. *Points* is the number of points in the model. *Thresh* is the threshold that was used to determine hypotheses. *Probes* is the number of transformations of the object model that were probed in the distance transforms and is in thousands. The time given is for matching a single object model and neglects the image preprocessing time. *Biggest* is the size of the largest false alarm found.

ered in a single image also helped to speed the search. When 27 different object models were considered in the same image using the multi-model techniques, 0.86 seconds were necessary per model to perform the matching when 80% of the model edge pixels were required to match the image closely and 0.34 seconds were necessary per model with when 90% of the model edge pixels were required to match closely.

6 Summary

This paper has discussed techniques to perform object recognition by matching sets of oriented edge pixels. We first reviewed a generalization of the Hausdorff distance that allows the determination of good matches between an oriented model edge map and an oriented image edge map. We then described a search strategy that allowed the full space of possible transformations to be examined quickly in practice using a hierarchical cell decomposition of the transformation space. Additional techniques for reducing the overall time necessary when we search for any of several object models in a single image were also given.

We have considered in detail the probability that these matching techniques will yield a large false positive due to random chains of edge pixels in the image using a model of the matching process and have give a method to estimate this probability efficient at run-time. Methods by which this information can be used to improve the system were then discussed. Empirical results indicate that the use of orientation information considerably reduces the number of false positives and the computation time and that our false positive estimation model performs well in practice.

The techniques described here yield a very general method to perform object recognition that is robust to changes in lighting and contrast, occlusion, and image noise, and that can be applied to a wide range of imaging modalities. Since efficient techniques exist to determine good matches, even when a large space of transformations are considered, and to determine the likelihood that a false positive will be found or that any particular hypothesis is a false positive, these methods are useful and practical for identifying objects in images.

Acknowledgments

This work was supported in part by ARPA under ARO contract DAAH04-93-C-0052 and by National Science Foundation PYI grant IRI-9057928.

References

[1] H. G. Barrow, J. M. Tenenbaum, R. C. Bolles, and H. C. Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 659–663, 1977.

[2] W. H. E. Day and H. Edelsbrunner. Efficient algorithms for agglomerative hierarchical clustering methods. *Journal of Classification*, 1(1):7–24, 1984.

[3] W. E. L. Grimson and D. P. Huttenlocher. Analyzing the probability of a false alarm for the Hausdorff distance under translation. In *Proceedings of the Workshop on Performance versus Methodology in Computer Vision*, pages 199–205, 1994.

[4] D. P. Huttenlocher, G. A. Klanderman, and W. J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[5] D. P. Huttenlocher and W. J. Rucklidge. A multi-resolution technique for comparing images using the Hausdorff distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 705–706, 1993.

[6] C. F. Olson and D. P. Huttenlocher. Recognition by matching dense, oriented edge pixels. In *Proceedings of the International Symposium on Computer Vision*, pages 91–96, 1995.

[7] D. W. Paglieroni, G. E. Ford, and E. M. Tsujimoto. The position-orientation masking approach to parametric search for template matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16:740–747, July 1994.

[8] W. J. Rucklidge. Locating objects using the Hausdorff distance. In *Proceedings of the International Conference on Computer Vision*, pages 457–464, 1995.