Model-Based Edge Selection for 2-D Object Recognition

Hemant D. Tagare
Drew McDermott

Research Report YALEU/DCS/RR-1044
August 1994

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

Model-Based Edge Selection for 2-D Object Recognition

Hemant D. Tagare Drew McDermott

Department of Computer Science
Department of Diagnostic Radiology
Yale University
New Haven, CT 06510.

Abstract

Object recognition requires the selection of appropriate edges from the image. The recognition process can become computationally expensive if edge selection is coupled with recognition in one algorithm. To overcome this, we propose a model-based edge selection algorithm which prefilters the edges from the scene and presents only a small subset of the edges to the object recognition process. We present some design guidelines for model-based edge selection and develop simple algorithms based on these. Model-based edge selection is evaluated on a number of images and the statistics of its performance are presented.

1 Introduction

The classical approach to two-dimensional model-based object recognition is to find a set of edges in the image on which the model can be superposed by a transformation consisting of 2-D translation, rotation, and scaling. The computational complexity of recognition depends on the combinatorics of finding the appropriate edges in the image and the combinatorics of matching them to the corresponding parts of the object. The aim of this paper is to describe a preprocessing technique that selects edges in the image which have a high likelihood of coming from one instance of the object. Selectively choosing a small but potentially fruitful set of

edges is computationally advantageous when a few instances of the model (or none) are present amongst a lot of clutter that is obviously dissimilar.

The human visual system is known to select certain parts of the visual field for early attention. This phenomenon where some edges in the image "pop out" has been investigated at some length [23][12][13]. Although the work in this paper is motivated by pop-out in human vision, it differs from biological pop-out in a significant way. The proposed algorithm is completely model-dependent, whereas pop-out in human vision appears to be based on local features and relatively independent of object models [23]. Our mechanism may correspond more closely to the sort of "tuning" that occurs when a person scans the environment for an instance of a desired object type; if you're looking for your wallet, things of that general shape keep catching your attention, even though most turn out not to be a wallet. Obviously, the human visual system uses several cues besides shape, including color and texture, but we will focus entirely on shape in this paper.

As mentioned above edge selection by visual pop-out mechanism is only a preprocessing step for directing attention. It is possible to construct images where pop-out fails and the only feasible strategy is to examine every part of the image in detail. Camouflage is an example of this. On the other hand, camouflage occurs rarely compared to the situation where there is only one instance of the object present among a large number of dissimilar distractors. In the latter case, a pop-out mechanism almost always leads to quicker recognition.

It should also be noted that a visual pop-out mechanism is fundamentally different from an object-recognition mechanism — there would not be any advantage to using it if the two were the same. In particular, pop-out differs from object recognition in two ways: (1) Pop-out need not recognize the presence of the object, but should strongly localize a few possible instances of the object. In practice, this means that it need not pop out all of the edges of the object that are visible in the image. It is sufficient to pop out just enough edges that look similar to the object and which tightly constrain the possible location, orientation, and scale of the object in the image. (2) Pop-out need not be as stringently selective as exact model matching. A few false positives are acceptable as long as a large percentage of the edges in the image are rejected. Therefore, pop-out may use a less refined, but faster, strategy than recognition.

We assume that pop-out operates in a "bottom-up" fashion. That is, the image is scanned for small features, each of which is then tested to see if it could possibly be part of a match to the model. The initial scan does not take the model into account. A key question is what these small features should be — small linear edge elements, or large sets of edge segments which correspond to large chunks of the model, or some other primitive with intermediate complexity? The most basic property required of a primitive feature is that occlusion should not hamper the detection of primitives in the image. Further, the localization requirement discussed above indicates that a possible match of the primitive to the object model should tightly constrain the range of transformations for the match. Intuitively, this implies that the primitive should be geometrically more complex that a linear edge element. On the hand, a primitive has to be

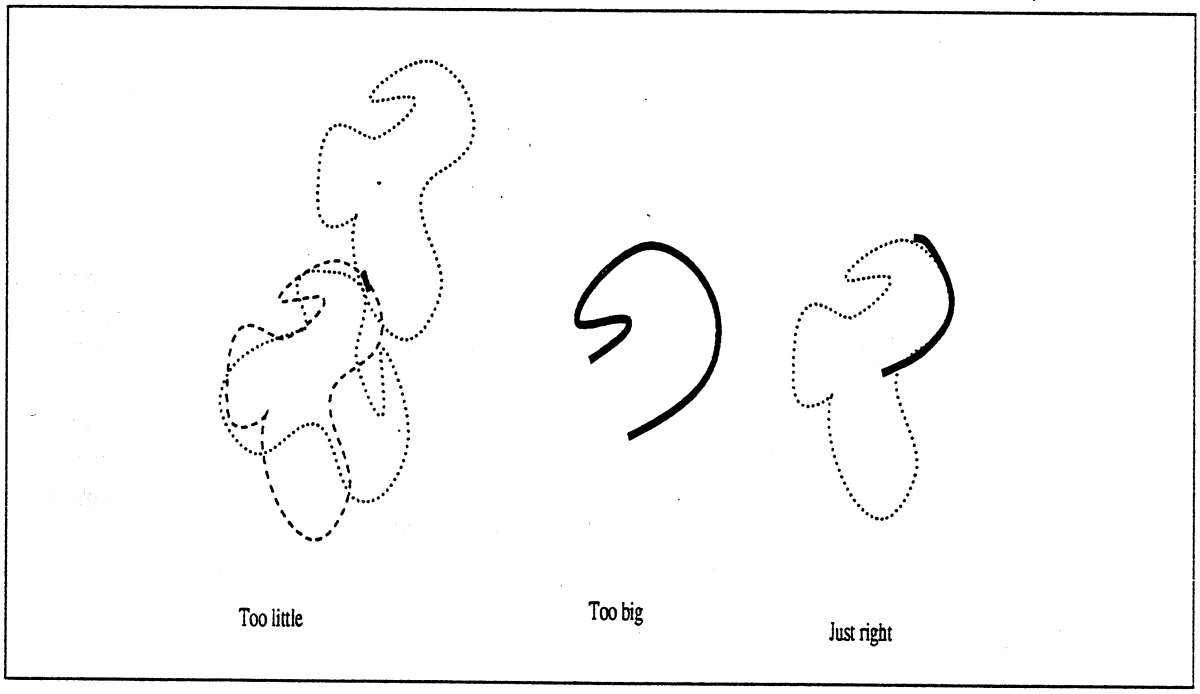

Figure 1: Medium Complexity Primitive.

detected quickly. If it is chosen to be substantially complex and required the selection of a large percentage of the visible edges of the object, the pop-out mechanism would not be any faster than the whole object-recognition mechanism. Therefore, it appears that a useful primitive should have a medium level of geometric complexity (figure 1).

Medium-complexity primitives also have the advantage that they lead to a quick algorithm for comparing image primitives with the object model and for grouping primitives that appear to come from the same instance of the model. This is based on the observation that primitives with medium complexity usually have a number of features that are invariants of the transformation and a primitive arising from the clutter in the image is likely to have *all* of these invariants dissimilar to the object. Hence a quick check of only a few invariants is sufficient to reject such primitives. Detailed matching with the model is not necessary. In fact, we use a single invariant to reject primitives arising from clutter. Further, the same argument can be used to simplify primitive grouping. Since we expect the clutter to be considerably dissimilar to the object, it is unlikely that we will find primitives from clutter with exactly the right transformation parameters so that all of the primitives can be consistently interpreted as coming from a single instance of the model. In fact, since we expect each match of the primitive to narrowly constrain the transformation parameters, two or more primitives arising from clutter are likely to have inconsistent values for *each* of transformation parameters. Thus, primitives may be grouped by checking consistency for only a few of the transformation parameters. The algorithm proposed here checks for consistency of orientation and scale only. Also, if two or more primitives seem to have consistent values of transformation parameters it is likely that they do come from the one instance of the model, so we may group them greedily without further computation.

One candidate control structure for the kind of mechanism we are studying is the Hough transform, perhaps the process used most often for edge selection in computer vision [10][11]. The Hough transform operates by accumulating votes for a model match from low-level feature matches. One can then locate peaks in the transform, and discard all edges that do not participate in the peaks. The surviving edges enter into more detailed matching with the model. Such a strategy is used, for example, by Grimson [7]. Hough transform like winner-take-all networks are also used by Bolle et. al. [6] to limit the combinatorics of search in the model recognition system.

Although the Hough transform can potentially select appropriate edges, it has a number of drawbacks:

1. It does not exploit the perceptual organization of the image. In particular, it does not provide additional weight if the set of edge elements participating in the transform happen to be connected. Connectivity of edge elements is a strong visual clue because it enhances the ability to reject accidental alignment of edges from unconnected curves.

2. If the transformations of the model to the image has a large number of parameters, a large accumulator array is required in the Hough transform. In practice, locating peaks in the

accumulator array can become computationally expensive. To reduce this cost, sometimes an accumulator array is maintained for only a few parameters [7].

3. Since each edge element participating in the Hough transform can match the model almost anywhere, a large number of accumulator cells are incremented by every match and the selectivity of the Hough transform remains low early in the matching process.

   The loss in selectivity is particularly apparent if the accumulator array is maintained only for a few parameters.

The limitations of the Hough transform are clearly seen to be a direct consequence of fact that the primitive used in the transform is an edge element and has very low geometric complexity. We may expect that replacing it with a mechanism that uses medium complexity primitives will be advantageous.

To sum up, we have the following "design rules" for model-based pop-out

1. `Choose a primitive of medium complexity which is easy to find under partial occlusion of the object and which tightly constrains the range of transformation parameters.`

2. `Compare a primitive from the image to the model quickly by checking if a few invariants of the primitive have appropriate values. Expect that most primitives from clutter will be rejected by this.`

3. `Group the surviving primitives by checking if a few of the transformation parameters have consistent values. Greedy grouping is an attractive alternative.`

4. `Pop-out is not recognition. It only serves as a pre-processing stage for increasing the effeciency of more detailed object-recognition.`

Some edge selection strategy is used in almost all previous work on object recognition but there has not been any systematic attempt to compare edge selection strategies or formulate principals or design rules for them. There is considerable literature on object recognition and we do not aim to review all of it here. We review some prototypical recognition systems to illustrate how our edge selection strategy compares with others'.

The arc-length versus turning angle function of the connected edges in the images has been used by a number of researchers as the primitive for selecting the appropriate edges and models. Examples of such studies include McKee and Aggarwal [18]; Turney, Mudge and Volz [24]; the footprint matching strategy of Kalvin et al. [14], Shwartz and Sharir [21], and Mehrotra and Grosky [?]. Stein and Medioni[22] use a similar strategy for recognizing 3-D curves and surfaces. The edge selection strategy proposed here is similar to that of using arc-length turning-angle

functions but differs in that it is much simpler. We parse the connected edges in the images into short segments (the details are in the next section) and only one angle is computed for each segment. Segments which fit the object model in a consistent manner are grouped together as edge segments of interest. What is interesting is that the experiments reported in section 4 indicate that this simple feature is sufficient for effective pop-out. This is consistent with the psychophysical observation that the human visual system uses extremely simple primitives for pop-out [23]. The simplicity of the pop-out feature is also what makes our system different from that of Bolles and Cain [4].

Further, we are not as interested in selecting the appropriate model from a large database of potential objects in the scene but are interested in rapidly selecting a set of edges that are likely to come from a single instance of a single model. Therefore our algorithm for grouping the primitives into consistent subsets is quite different from the algorithms pursed by the authors cited above.

Other than arc-length versus turning angle function the common strategy for edge-selection is the use of Hough transform. We have discussed the Hough transform and some of its limitations above, and as we mentioned our approach differs from the Hough transform in that we use a more complex primitive.

Another similar approach is that of using perceptual organization to choose certain groups of edges for model matching [5][15][16][17][19][20][25]. The perceptual organization approach is similar in that it too uses medium complexity primitives. However, it differs from our approach in that it is model-independent and bottom up. Our approach is very model-dependent and top-down.

2 Choosing the Primitive and Matching it to the Object

The primitive used in our algorithm is called a "U." Edge elements in the image are linked into continuous curves and tokens are placed at the corners and the end points of the curve. (A *corner* is a point of maximum curvature.) A U is the portion of a curve that encompasses three successive tokens. Figure 2a shows an example of a curve parsed into a sequence of U's. Figure 2b shows the corner and arms of a U. By orienting the line of sight from the corner along the bisector of the acute angle between the two arms, we can identify one arm as the left arm and the other arm as the right arm.

The object model as well as image data are parsed into U's. Figure 2b shows an instance of a parsing of an object model. To distinguish between image and model U's, we denote the former as U and the later as $\mathcal{U}$.

It is easy to see that U's satisfy the three requirements of primitives stated above:

1. Occlusion does not inhibit the ability to find U's belonging to the object of interest as long as not all of the corners of the object are occluded. Occlusion creates T shaped corners in

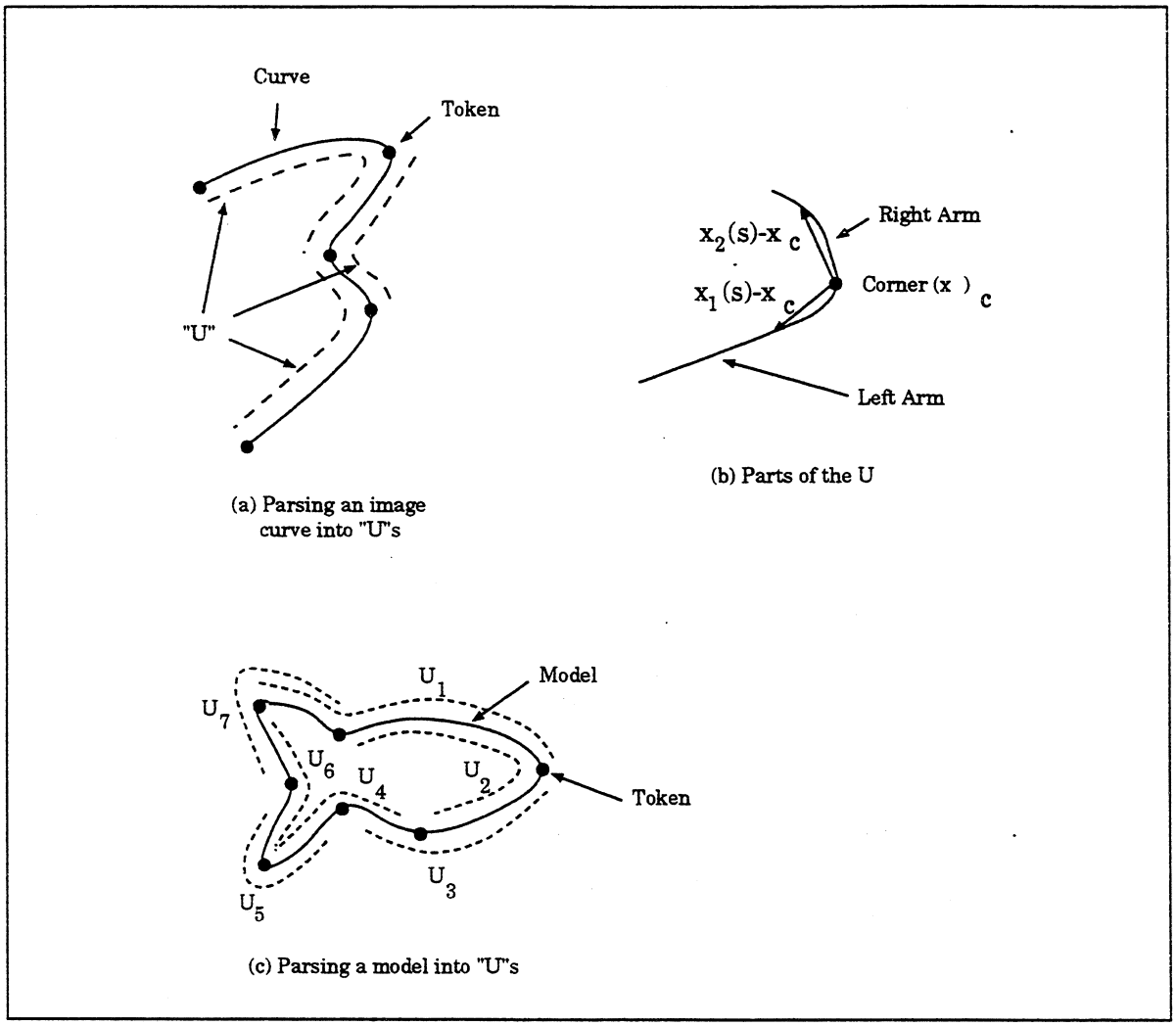

Figure 2: The Primitive.

the image and additional U's arise from such corners. The two arms of such a U do not belong to the same object. However, unless all of the corners of the object are occluded, these arms will also participate in U's that arise from the object of interest (fig 3) and therefore their accidental participation in Us at the occluding corner is not detrimental to the performance of the edge selection algorithm.

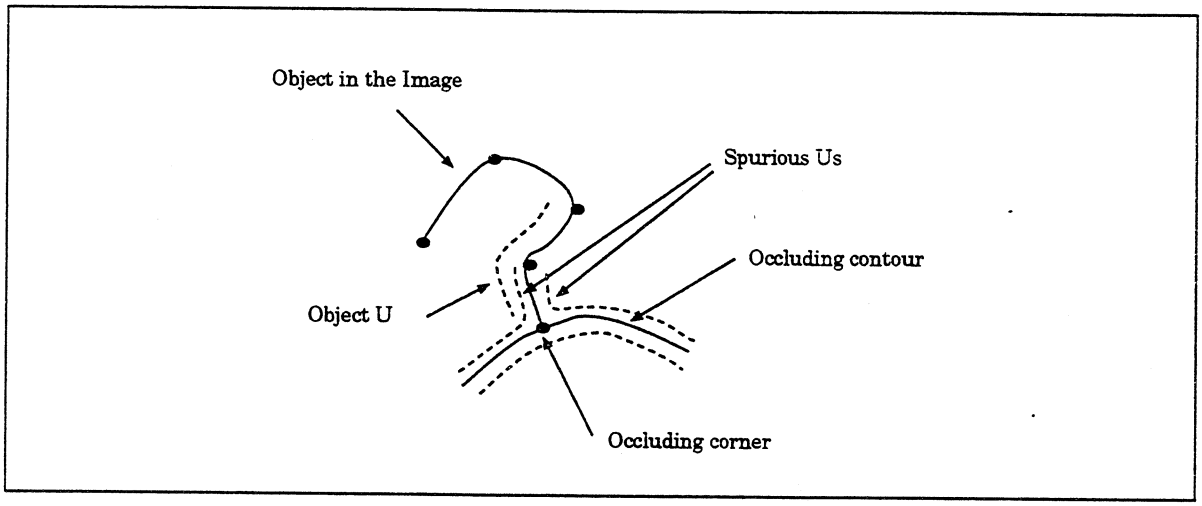

Figure 3: U's from an occluding corner.

2. Matching one U to another tightly constrains the transformation — the corners of the two Us have to line up, which constrains the translation, and the arms of the U's have to line up which constrains the possible rotation and scale.

3. U's arising from clutter are not likely to have the appropriate angle between the two arms to fit most of mode Us. Since the angle is also invariant to translation, rotation, and scale it may be used to quickly check whether an image U matches the model. The notion of angle between arms is made precise below.

4. Finding U's is computationally quick. The edge elements are linked together into continuous curves and the corners of the curves are found as the extrema of curvature. The details are discussed in the experimental results section. The processing time for finding U's is low — typically about 2 secs on a Sun SPARCStation 2 for processing images with 5 to 15 objects in view.

As mentioned in section 1 an image primitive need not be fully matched to a model primitive for accepting it as a feasible instance — if the primitive is rich enough, it suffices to compare

only a few features. We compare U's by the following three features:

1. The angle of the average deviation of its two arms, i.e., if $\mathbf{x}_c$ is the corner of the U, $\mathbf{x}_1(s_1)$ is a point at a distance s_1 from the corner on the left arm, and $\mathbf{x}_2(s_2)$ is a point at a distance s_2 from the corner on the right arm, then (figure 2b)

$$\mathbf{v}_1 = \frac{1}{L}\int_0^L (\mathbf{x}_1(s) - \mathbf{x}_c)\, ds,$$

$$\mathbf{v}_2 = \frac{1}{L}\int_0^L (\mathbf{x}_2(s) - \mathbf{x}_c)\, ds,$$

where L is the length of the smaller arm, are taken to be the "average" arm vectors and

$$\theta = \angle(\mathbf{v}_1, \mathbf{v}_2) \tag{1}$$

where $\angle()$ is the smaller of the two angles between $\mathbf{v}_1$ and $\mathbf{v}_2$, is taken to be the angle of deviation between the two arms. Note that $\mathbf{v}_1$ and $\mathbf{v}_2$ are computed by integrating over the same arc length on both the arms.

Since the angle of deviation is likely to change with different amounts of occlusion, a set of angles of deviation is computed for every model $\mathcal{U}$. For a model $\mathcal{U}$, the "average" arm vectors are computed as

$$\mathbf{w}_{1,n} = \frac{1}{\mu_n L}\int_0^{\mu_n L} (\mathbf{x}_1(s) - \mathbf{x}_c)\, ds,$$

$$\mathbf{w}_{2,n} = \frac{1}{\mu_n L}\int_0^{\mu_n L} (\mathbf{x}_2(s) - \mathbf{x}_c)\, ds,$$

for $\mu_n = 0.1, 0.2, \cdots, 1.0$. Here L refers to the length of the smaller of the two arms, and $1 - \mu_n$ indicates the level of occlusion. The deviation angles ϕ_n are given by

$$\phi_n = \angle(\mathbf{w}_{1,n}, \mathbf{w}_{2,n}).$$

2. The smaller of two arm lengths, and

3. The location of the corner $\mathbf{x}_c$.

Thus, the k^{th} U in the image is characterized by the triple $(\theta_k, L_k, \mathbf{x}_{c_k})$. The k^{th} $\mathcal{U}$ of the model is characterized by a set of 10 triples $(\phi_{k,n}, \mu_n L_{k,n}, \mathbf{x}_{c_k})$, $n = 1, \cdots, 10$.

A model $\mathcal{U}$ (say the k^{th} $\mathcal{U}$) is considered a feasible match of an image U (say the l^{th} U) if for some level of occlusion n, the deviation angles of the two U's are similar and the change of scale required to match the shorter of the two model $\mathcal{U}$ arms to the shorter of the two image U arms is within prerequired bounds, i.e.,

$$\begin{aligned} \|\phi_{k,n} - \theta_l\| &\leq \Delta_\theta, \text{ and} \\ \alpha_{min} - \Delta_\alpha &\leq \frac{L_l}{L_{k,n}} \leq \alpha_{max} + \Delta_\alpha, \end{aligned} \tag{2}$$

where, Δ_θ is the allowable range of angle mismatch and Δ_α is the allowable range of scale mismatch. α_{max} and α_{min} are the maximum and minimum values for allowable scale transformations.

If $\mathcal{U}_k$ can be feasibly matched to U_l, then the ranges of scale and rotation for the match can also be found. The range of the scale is obtained as the interval $S = [\frac{L_l}{L_{k,n}} - \Delta_\alpha, \frac{L_l}{L_{k,n}} + \Delta_\alpha]$. The range of rotation is obtained as the intersection of the range of the rotation parameter for the left arm and the range of the rotation parameter for the right arm, i.e., if $\mathbf{w}_1$ and $\mathbf{w}_2$ are the left and right arms of the model and $\mathbf{v}_1$ and $\mathbf{v}_2$ the left arm and right arms of the image U, the range of rotations is the interval

$$R = [\angle(\mathbf{w}_1, \mathbf{v}_1) - \Delta\theta, \angle(\mathbf{w}_1, \mathbf{v}_1) + \Delta\theta] \bigcap [\angle(\mathbf{w}_2, \mathbf{v}_2) - \Delta\theta, \angle(\mathbf{w}_2, \mathbf{v}_2) + \Delta\theta]. \tag{3}$$

A feasible match is denoted by $(\mathcal{U}_m \overset{[R,S]}{\leftrightarrow} U_n)$.

Multiple feasible matches can be grouped into a feasible interpretation if they have a consistent range of scale and rotation parameters and if no model $\mathcal{U}$ or image U occurs more than once in the interpretation. Note that when feasible matches are grouped together additional information is available from combinations of matches that can further reduce the range of the transformation parameters. In our case, it is easy to see that comparing the distances between the corners of all pairs of model $\mathcal{U}$s in the feasible interpretation to the distances between the corners of corresponding pairs of image Us (fig. 4) gives more information about scale. Also, by comparing triples of model $\mathcal{U}$ corners in the feasible interpretation to corresponding image U corners (fig. 4) further information about the rotation range can be obtained. As more matches are grouped into a feasible interpretation checking all pairs and triples of matches becomes expensive. To combat this complexity we use Grimson's observation that checking constraints amongst combinatorial subsets of a feasible solution does not increase pruning to any appreciable degree [7]. In order to keep the computational cost down, we only compute the scale ranges for distances between corners of certain pairs of Us and check for the consistency of these scale ranges with those obtained from individual feasible matches.

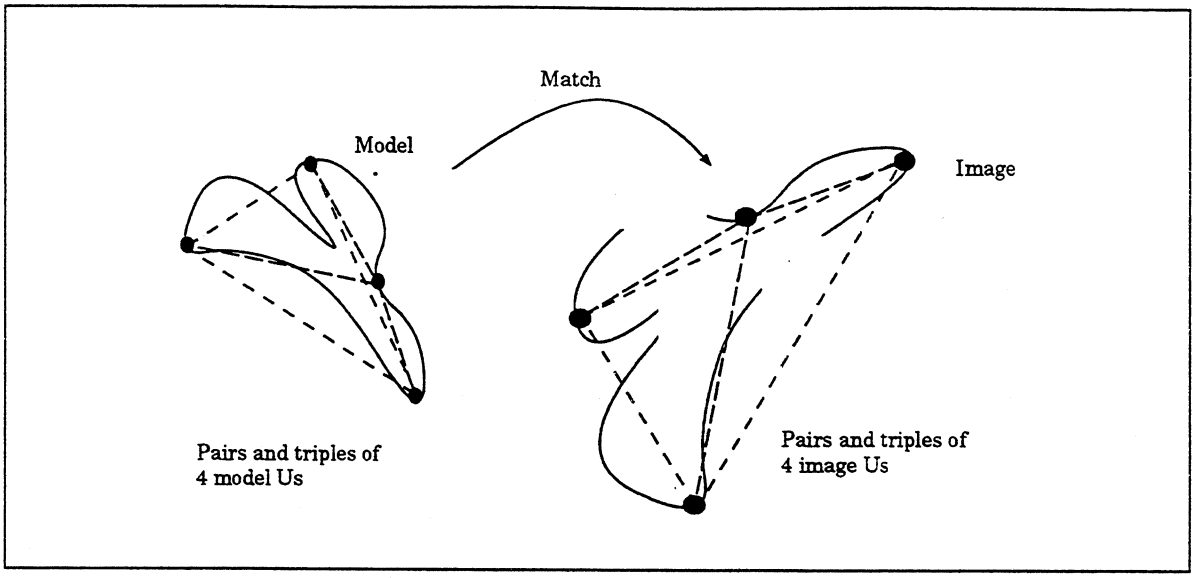

Figure 4: Combinations of Us.

We say that the set

$$. I = \begin{pmatrix} \begin{pmatrix} U_{m_1} & \overset{[R_1,S_1]}{\leftrightarrow} & U_{n_1} \end{pmatrix} \\ \begin{pmatrix} U_{m_2} & \overset{[R_2,S_2]}{\leftrightarrow} & U_{n_2} \end{pmatrix} \\ \cdots \\ \begin{pmatrix} U_{m_p} & \overset{[R_p,S_p]}{\leftrightarrow} & U_{n_p} \end{pmatrix} \end{pmatrix}.$$

of feasible matches is a feasible interpretation iff

- There are no multiple instances of the same model or image U in the interpretation, i.e., $m_j \neq m_k$, for $j \neq k$, and $n_j \neq n_k$ for $j \neq k$, and
- All of the parameter ranges have a non-empty intersection, i.e., $\bigcap_k R_k \neq \emptyset$, and $\bigcap_k S_k \bigcap \tilde{S}_k \neq \emptyset$, where $\tilde{S}_k$ is the scale range of the distances between the corners of the first feasible pair and the k^{th} feasible pair

$$\tilde{S}_k = [\frac{\| \mathbf{x}_{c_{n_1}} - \mathbf{x}_{c_{n_k}} \| - \Delta d}{\| \mathbf{x}_{c_{m_1}} - \mathbf{x}_{c_{m_k}} \| + \Delta d}, \frac{\| \mathbf{x}_{c_{n_1}} - \mathbf{x}_{c_{n_k}} \| + \Delta d}{\| \mathbf{x}_{c_{m_1}} - \mathbf{x}_{c_{m_k}} \| - \Delta d}],$$

where $\mathbf{x}_{c_{n_1}}$ and $\mathbf{x}_{c_{n_k}}$ are the corners of the image U of first and k^{th} feasible match and $\mathbf{x}_{c_{m_1}}$ and $\mathbf{x}_{c_{m_k}}$ are the corners of the model $\mathcal{U}$s of the same feasible matches, and Δd is the expected error in distance calculations due to improper localization of corners.

If a set of matches does not satisfy these constraints, the set is called an infeasible interpretation.

Given the above definition of a feasible algorithm we may develop an algorithm that tries different combinations of matches to find feasible interpretations. However, from a practical point of view, any such algorithm is likely to give undesirable results due to a number of reasons. Experience with experimentation (with other recognition techniques as well) has indicated that the following three reasons appear to be important:

1. Any proper subset of a feasible interpretation is also a feasible interpretation. Hence an algorithm that tries to find as many feasible interpretations as possible is likely to generate a number of feasible interpretations which are subsets of some bigger interpretation. This tendency to generate subsets of a true solution as ostensibly alternate solutions can be noted in other algorithms too. In the Hough transform, for example, accumulator array cells close to the the cell containing the peak often has near peak values derived from model-image pairs that are subsets of the best solution.

2. Since our approach depends on the observation that only a few image Us will potentially match model $\mathcal{U}$s, the presence of any image Us that match a large number of model $\mathcal{U}$s will cause all such Us to be grouped together in different combinations into feasible interpretations. This is particularly disastrous if the image has a large number of Us with short arms since they tend to match many occluded model $\mathcal{U}$ (under severe occlusion all Us tend to look the same as their apparent deviation angle becomes small). In practice, it is common to observe spurious feasible interpretations forming only from matches with severely occluded model $\mathcal{U}$s. This tendency too is present in other recognition algorithms, e.g., it has been noted that small edge fragments can accidentally align in the appropriate way so that they appear to the Hough transform as an instance of the object being present at high occlusion [8].

3. Finally, the definition of a feasible interpretation does not impose any restrictions on the number of feasible matches it should contain. Any feasible match by itself is also a feasible interpretation and any algorithm that seeks more than one feasible interpretation is likely to retain some matches all by themselves as feasible interpretations.

The algorithm used in this paper overcomes these limitations by using explicit post processing to remove feasible interpretations that arise as described above.

3 Forming Feasible Interpretations

We discuss two algorithms for forming feasible interpretations. The first algorithm is an adaptation of the interpretation-tree algorithm and it terminates as soon as an acceptable feasible interpretation is found. The second algorithm is a greedy constraint propagation algorithm that terminates after finding as many feasible interpretations as it can in a greedy manner. Since the second algorithm finds more than one feasible interpretations it is the algorithm of choice for evaluating the utility and limitations of model-based-visual pop-out. All of the experiments reported in section 4 use the second algorithm.

Finding feasible interpretations can be thought as a two-stage process, where all feasible matches are found in the first step and in the second stage the feasible matches are combinatorially grouped into different sets and sets that are not feasible interpretations deleted. The two algorithms presented below differ in the second stage.

The first stage is identical in both algorithms. In it, all possible feasible matches of image and model Us are found and grouped. At the end of this stage, there is a list for every model $\mathcal{U}$ containing all of its feasible matches. The current version of the algorithm creates the lists by comparing every image U with every model $\mathcal{U}$ for a potential match. In our experience this is not a computationally limiting step (details of timing are in the experimental section). However, it would be relatively straightforward to augment this step by creating an indexing scheme based on the invariant (deviation angle).

3.1 Feasible Interpretations by Tree Search

The first algorithm arranges combinations of feasible matches as nodes in a tree and searches the tree for nodes that are feasible interpretations (figure 5). The root node of the tree is the empty set. A node at depth k is a set of feasible matches involving at most the first k model $\mathcal{U}$s, i.e., any node at level k is the following set:

$$\left(\begin{array}{c} \left(\mathcal{U}_{m_1} \overset{[R_1,S_1]}{\longleftrightarrow} U_{n_1} \right) \\ \left(\mathcal{U}_{m_2} \overset{[R_2,S_2]}{\longleftrightarrow} U_{n_2} \right) \\ \cdots \\ \left(\mathcal{U}_{m_p} \overset{[R_p,S_p]}{\longleftrightarrow} U_{n_p} \right) \end{array} \right),$$

with $p \geq 0$ and with $1 \leq m_1, \cdots, m_p \leq k$, and $m_i \neq m_j$ for $i \neq j$. Branches from a node represent the addition of a feasible match of the $k+1$ model $\mathcal{U}$ to the set of the node. There are as many branches as feasible matches of $\mathcal{U}_{k+1}$ plus an extra branch which does not add any feasible match of $\mathcal{U}_{k+1}$ to the set (representing the fact that $\mathcal{U}_{k+1}$ might not be contained in the interpretation).

An effective search strategy to find a feasible interpretation is to begin at the root node and proceed, in a depth first manner, into the tree by expanding those nodes which are feasible interpretations. Subtrees which are rooted at nodes containing infeasible interpretations need not be searched at all. The search can be terminated at a feasible node containing a feasible interpretations soon as a sufficient number of image Us have entered into the feasible interpretation or upon the satisfaction of some other criteria.

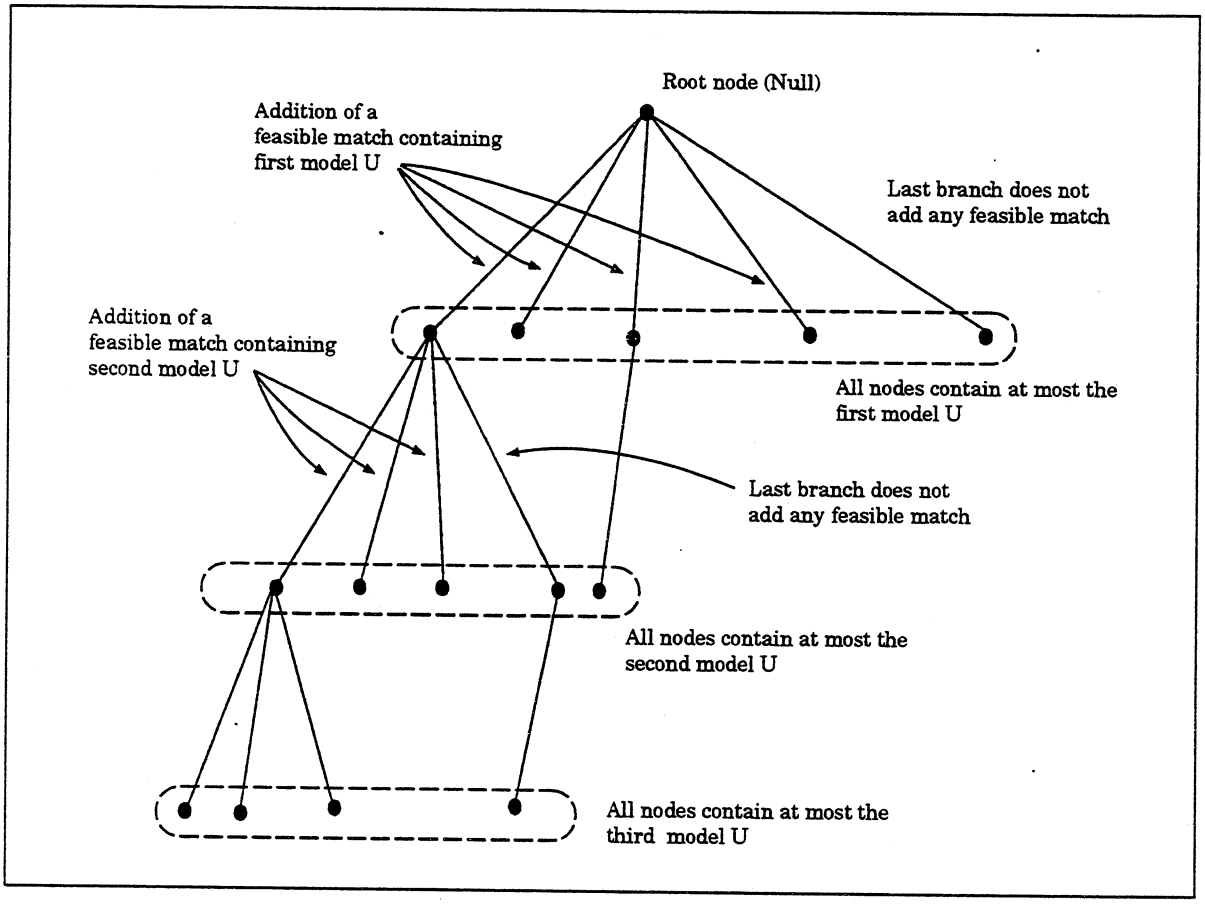

Figure 5: Finding Feasible Interpretations by Tree Search.

This is the algorithm of choice if model-based-edge-selection is used an application where we are interested in finding one instance of the object at a time in a demand-driven fashion. In such applications, the search may be suspended as soon as the first feasible interpretation is found and the feasible interpretation can be passed to a more sophisticated algorithm for recognition.

If the latter indicates that the grouping is inappropriate, the node in the search tree may be treated as infeasible and the search for the next feasible interpretation resumed.

We do not pursue this algorithm further since we are interested in obtaining a whole set of alternate feasible interpretations to experimentally evaluate the overall accuracy and effeciency of model-based-edge-selection.

Previously we have argued that consistent feasible matches may be greedily grouped to obtain feasible interpretations. The second algorithm uses this heuristic.

3.2 Feasible Interpretation by Greedy Grouping

The algorithm has three stages. As before, in the first stage, all possible feasible matches of image and model Us are found and arranged in lists according to the model $\mathcal{U}$ in the feasible match (figure 6).

In the second stage, the feasible match lists are iteratively and greedily merged into a list of feasible interpretations. Here's a sketch of the algorithm:

```
Interps ← {};
For each 𝒰
   (For each I ∈ Interps
      If there is an unused match of the type (𝒰 ↔ U),
            that can be added to I
        then (I ← I ∪ {(𝒰 ↔ U)};
            Mark (𝒰 ↔ U) used in the list of matches with 𝒰);
    For each unused (𝒰 ↔ U)
      Interps ← Interps ∪{(𝒰 ↔ U)})
```

The procedure begins by initializing the feasible-interpretations list `Interps` to the empty list. Then the feasible-interpretations list is merged with each feasible-match list starting from the list for the first model $\mathcal{U}$ and proceeding to the last. Every merge proceeds as follows: Beginning from the first feasible interpretation, each feasible interpretation I is considered for grouping with each feasible match in the feasible-matches list. As soon a grouping which can be merged with I to make an enlarged feasible interpretation, the merge is carried out, and the feasible match is marked "used." Further comparisons of the interpretation with remaining feasible matches (in the current list) are aborted, and comparisons of the next feasible interpretation are begun. Once all feasible interpretations are compared with elements of a feasible matches list, there may remain unused feasible matches in the list for $\mathcal{U}$ that are not compatible with any of the current feasible interpretations. They are added to the end of the list `Interps` as new feasible interpretations.

Note that for the first $\mathcal{U}$ considered, `Interps` is empty, so each $(\mathcal{U} \leftarrow U)$ will be unused and `Interps` will be initialized to the set of all matches to $\mathcal{U}$. On the next iteration, each such "seed" match will be extended, and so forth.

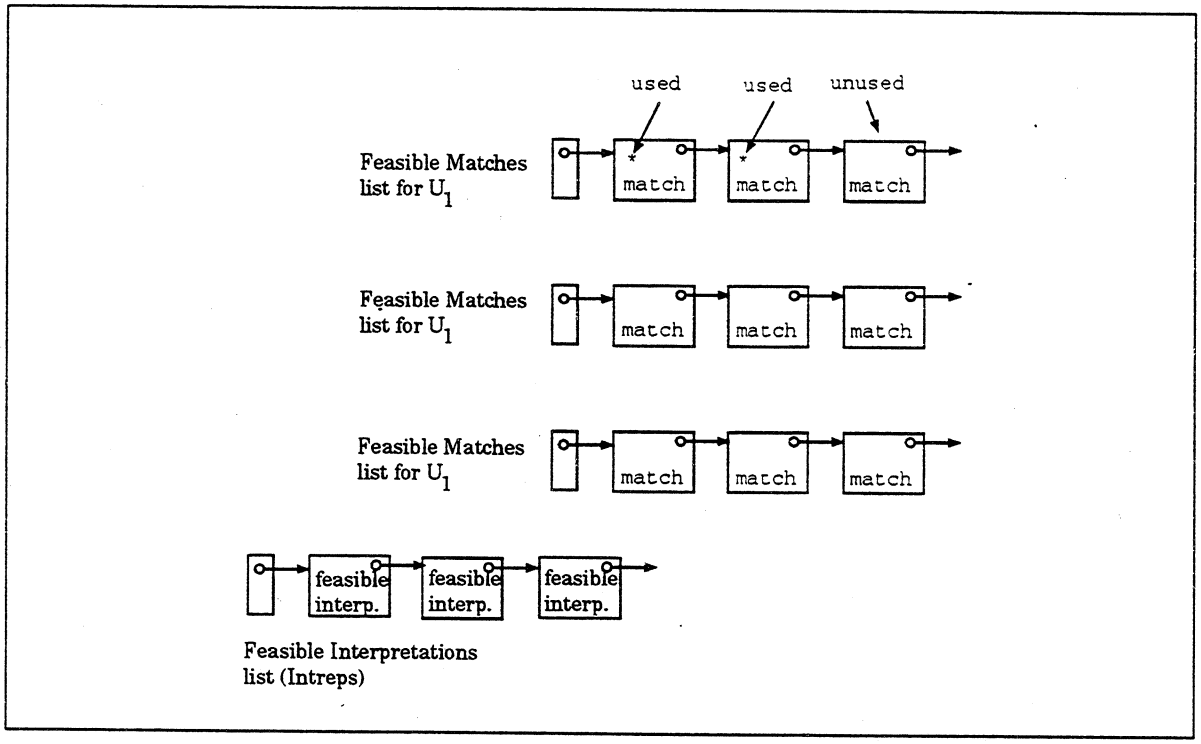

Figure 6: Finding Feasible Interpretations by Greedy Grouping.

The greedy nature of the algorithm lies in the choice to remove a feasible match from its list as soon as it matches with a feasible interpretation; and to abort further comparison of a feasible interpretation once a feasible match is found consistent with it.

At the end of the second stage a set of feasible interpretations is obtained. As mentioned before, post processing is necessary to get rid of undesirable interpretations. Three types of feasible interpretations are deleted: (1) Feasible interpretations which are subsets of other feasible interpretations in the list, (2) Feasible interpretations which indicate that all of the model $\mathcal{U}$s present in it have 80% or higher occlusion, and (3) Feasible interpretations which contained only single feasible matches. All of the surviving feasible interpretations are selected as being appropriate. This is the third stage of the algorithm.

4 Experimental Results

As we discussed before, visual pop-out is useful only on the average it can quickly find the object in the presence of dissimilar clutter. To evaluate the statistical performance of our algorithm we performed experiments on 50 images containing an instance of a model and 25 images containing no instance of the model. The model used was the cardboard cut out of a fish as shown in figure 7. It has 6 Us.

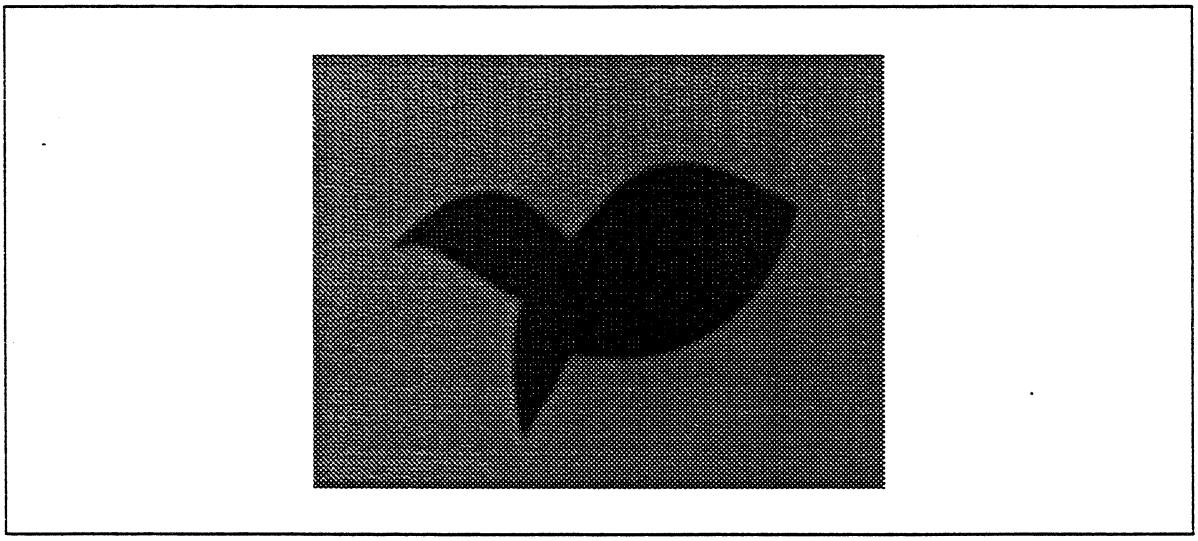

Figure 7: The Model.

4.1 Images with model present

We discuss the images containing the model first. Each image was created as follows: the model was placed on a rectangular area approximately 1 foot long by 1 foot wide. Between 5 and 15 commonly occurring laboratory tools and other objects were randomly tossed onto the above area. These provided the clutter. Since the model was always placed before the clutter was introduced, in all cases the model was partially occluded. The camera zoom was randomly varied so that the apparent magnification of the model in the scene (compared to the size of the model in the image used to acquire the model) was between 2 and 0.5. No special lighting was introduced. A single image of each scene was recorded.

Once an image was obtained it was preprocessed to extract U's as follows:

1. A Canny-like edge detector was used to detect edges.

2. The image was scanned in a raster fashion from the top left to the bottom right. When an edge element was encountered the edge was tracked by looking for connectivity in a 3×3 neighborhood. If the edge element was multiply connected, the branch that had an image gradient most similar to the current edge element was tracked. If the edge appeared to have ended (not connected to new edge elements in a $3X3$ neighborhood), 5×5 and 7×7 neighborhoods were successively scanned for continuation. Thus 1 and 2 pixel gaps in the edges were bridged. If continuation was not found, edge tracking was discontinued and raster scanning for new edge elements resumed.

3. After all the edges in the image were grouped into curves, the corners of the curves were detected by looking for local extrema of curvature. The image gradient direction was chosen as the normal to the curve. A 9 pixel long Gaussian filter smoothed the normal. The curvature was computed as the rate of change of the orientation of the smoothed normal (w.r.t. arc length) and the location of the extrema of the curvature were chosen as corners. If the curve did not have any significant extrema, it was discarded.

4. Every corner along with the curve segments to the previous and next corner were grouped as a U.

5. By starting from the corner and proceeding symmetrically along both arms till the end of the shorter arm, the length of the shortest arm and the average deviation angle were computed for every U.

After image Us were found, they were compared with the model Us and feasible matches were formed. The greedy grouping algorithm was used to find feasible interpretations. The values of all of the parameters used in the algorithm are reported in table 1:

Table 1: Parameter Values Used in Greedy Grouping.

Parameter	Symbol	Value
Deviation Angle Mismatch	$\Delta\theta$	10^o
Scale Mismatch	$\Delta\alpha$	0.1
Maximum Allowable Scale	α_{max}	3.0
Minimum Allowable Scale	α_{min}	0.333
Error in dist. between two corners	Δd	7 pixels

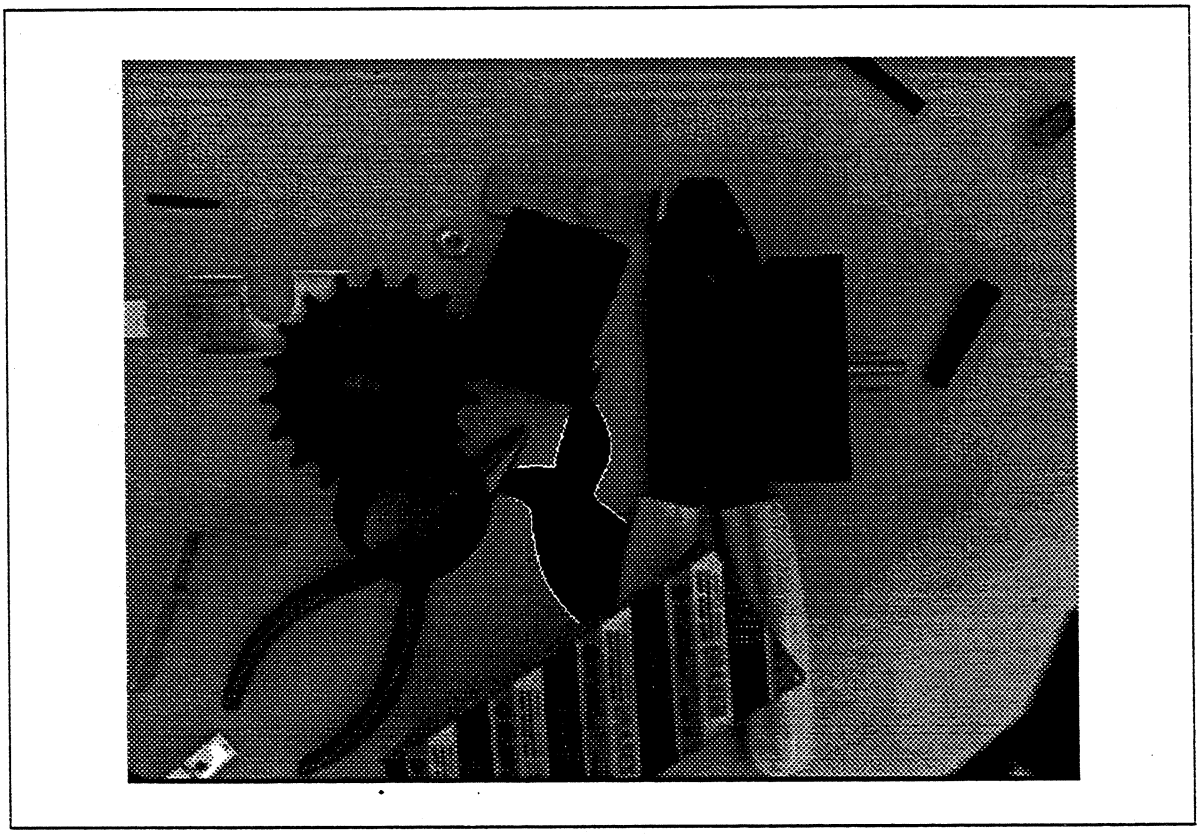

Figure 8: A Correct Feasible Interpretation.

As each image was processed statistics of the different geometric primitives were gathered to validate the assumption that medium complexity features are useful in pop-out. The statistics are reported in Table 2 (a typical image is shown in figure 8).

Table 2: Statistics of Geometric Primitives.

Primitive	Average	± Variance
Image Size	640 × 480 (pixels × pixels)	
Edge Pixels	4316 (pixels)	±1295 (pixels)
Connected Curves	141	±67
Image Us	105	±42.6
Feasible Matches	7.5	±5.0
Feasible Interpretations (after post-processing)	2.16	±1.76

The feature statistics of Table 2 show that on the average there were 105 image Us. Since the model has 6 Us, potentially there are 105 × 6 feasible matches. On the average only 7.5 of these survive validating the argument that medium complexity features can provide the appropriate pruning. As a further validation of the heuristic, we note that on the average only 2.16 feasible interpretations are created.

Each feasible interpretation was visually examined and declared as being correct or incorrect. Correct interpretations were the ones whose image Us came from the occurance of the model in the image. An example of a correct interpretation is shown in figure 8. The figure shows the model in the clutter. The edges of the model which formed the image Us of the correct feasible interpretation are outlined in the figure. In the 50 images we processed, at least one correct interpretation was found in 44 cases giving a success rate of 88%.

We also attempted to validate the use of the greedy algorithm. Once a feasible interpertation is formed by a greedy algorithm, it cannot be dismantled and its feasible matches cannot be used elsewhere. Therefore if a feasible match from the occurance of the model is assigned to an incorrect feasible interpretation, it will not be present in the correct feasible interpretation thereby reducing the "strength" of the correct feasible interpretation. We measured the strength of correct interpretations to evaluate how serious this effect is. We counted the the number of image pixels every feasible interpretation accounted for and ranked all of the feasible interpretations in each image according to the edge pixel count, so that when more than one feasible interpretation was present, the relative strength of any interpretation could be measured by its rank. In the 44 images which contained at least one correct feasible interpretation, the rank of

the correct interpretation is shown in Table 3.

Table 3: Relative Rank of Correct Interpretations.

Rank of Interpretation	No. of Occurences
1	37(37/44 = 84%)
2	6(6/44 = 14%)
3	1(1/44 = 2%)
≥ 4	0

The table shows that 98%(= 84%+14%) of the time the correct interpretation was either the strongest or the next strongest interpretation. This validates the use of the greedy algorithm.

The six images where no correct feasible interpretation was found were also examined. In all cases, the failure could be attributed to one of the following reasons: (1) The object was so heavily occluded so that at most one corner (and hence at most one U) was visible, or (2) The edge detector or the corner detector failed to locate a relevant U. An example of an image where the algorithm failed is shown in figure 9.

For each image, timing statistics were also gathered. These are reported in table 4. The statistics were gathered for the performance on a Sun SPARCStation 2. They were gathered after the edge detection stage.

Table 4: Timing Statistics on a Sun SPARCStation 2.

Processing Step(s)	Average (Secs.)	$\pm$ Variance (Secs.)
Edge grouping, Corner detection, and parsing into Us.	1.67	$\pm$0.55
Finding Feasible Matches, Feasible Interpretation, and Post-processing	1.41	$\pm$0.67

The table shows that on the average the pop-out algorithm took 3.01 seconds to execute

Figure 9: No Correct Feasible Interpretations for this Image.

when the model was present in the image.

4.2 Images with model absent

The procedure for creating the images with the model was repeated additionally 25 times without the model. A typical image is shown in figure 10. The greedy algorithm was executed with these images as data and the same statistics were gathered as before.

Figure 10: A Feasible Interpretation in an Image without the Model.

The statistics of the geometric features in the images are shown in Table 5. Comparing this table to table 2 we see that the ggeometric complexity of the two sets of images was similar.

It was found that the algorithm indicated atleast one feasible interpretation in of the cases). Figure 10 shows one feasible interpretation. Only 16% of cases had more than one feasible interpretation. Visual inspection of all feasible interpretations revealed that most feasible interpretations could easily be rejected by a more detailed model matching stage. In fact, simply

Table 5: Statistics of Geometric Primitives.

Primitive	Average	± Variance
Image Size	640 × 480 (pixels × pixels)	
Edge Pixels	4851 (pixels)	±845 (pixels)
Connected Curves	180	±44
Image Us	120	±27
Feasible Matches	3.24	±2.2
Feasible Interpretations (after post-processing)	1.8	±0.49

checking a description of the U shape which is more sophisticated that the average deviation angle and checking the image for presense of occlusion where the feasible interpretion believes the U is occluded can reject most of these interpretations.

Timing statistics for the performance of the algorithm are given in Table 6. These statistics are comparable to those of table 4.

Table 6: Timing Statistics on a Sun SPARC-Station 2.

Processing Step(s)	Average (Secs.)	± Variance (Secs.)
Edge grouping, Corner detection, and parsing into Us.	1.8	±0.5
Finding Feasible Matches, Feasible Interpretation, and Post-processing	1.1	±0.3

5 Conclusion

A model-based strategy for edge selection is a feasible preprocessing step in 2-D object recognition. Intuitive complexity considerations show that a medium complexity primitive is appro-

priate for use in edge selection. A specific medium complexity primitive called a U is proposed and used in this paper, and the resulting algorithm is shown to have a success rate of 88% in our experiments. All of the failures were due to inadequate performance of the edge and corner detector.

Although a U is a useful primitive for a large class of images, it is by no means the only possible primitive that can be used in model-based edge selection. Other primitives, perhaps along the lines of geons [3], can also be used. The issues of relative effectiveness of primitives and the management of multiple primitives remain open.

References

[1] N. Ayache, O. D. Faugeras, "HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects," IEEE Trans. Pattern Anal. Machine Intell., vol. PAMI-8, no. 1, pp. 44-54, Jan. 1986.

[2] D. H. Ballard, C. M. Brown, *Computer Vision*, Prentice-Hall, 1982.

[3] I. Biederman, "Recognition by Components; A Theory of Human Image Understanding," Psych. Review, vol. 94, no. 2, pp. 115-147, 1987.

[4] Bolles R. C., Cain R. A., "Recognizing and Locating Partially Visible Objects: The local-feature-focus Method," Intl. Journ. Robotics Research, 1(3), 1982.

[5] Brooks R., "Symbolic Reasoning Among 3D models and 2D Images," Artificial Intelligence Journal, Vol. 17, 1982.

[6] R. M. Bolle, Califano A., Kjeldsen, "A Complete and Extendable Approach to Visual Recognition," I.E.E.E. Trans. Pat. Recog. Mach. Intell., Vol. 14, No. 5, May 1992.

[7] W. Eric L. Grimson, T. Lozano-Perez, "Localizing Overlapping Parts by Searching the Interpretation Tree," IEEE Trans. Pattern Anal. Machine Inell., vol. PAMI-9, no. 4, pp. 469-482, July 1987.

[8] W. Eric L. Grimson, D. P. Huttenlocher, "On the Verification of Hypothesized Matches in Model-Based Recognition," IEEE Trans. Pattern Anal. Machine Intell., vol. 13, no. 12, Dec. 1991.

[9] Grosky W. I., Mehrotra R., "Index-Based Object Recognition in Pictorial Data Management," Compt. Vision Graph. Imag. Proc., Vol. 52, 416-436, 1990.

[10] P. V. C. Hough, "A Method and Means for Recognizing Complex Patterns," US Patent No. 3,069,654, 1962.

[11] J. Illingworth, J. Kittler, "A Survey of the Hough Transform," Comput. Vision, Graphics, Image Proc., vol. 44, pp. 87-116, 1988.

[12] B. Julesz, Krose B.,"Features and Spatial Filters," Nature, May 26, Vol. 333, May 1988.

[13] Julesz B., Sagi D., " Where and What in Vision," Science, Vol. 228, June 1985.

[14] A. Kalvin, E. Schonberg, J. T. Schwartz, M. Sharir, "Two-Dimensional, Model-Based, Boundary Matching Using Footprints," Intl. Journ. Robotics Research, Vol. 5, No. 4, Winter 1986.

[15] Kubovy, Pomerantz, *Perceptual Organization*, Lawrence Erlbaum, 1981.

[16] Liou S. -P, Chiu A. P., Jain R. C., "A Parallel Technique for Signal-level Perceptual Organization," I.E.E.E. Trans Pat. Recog. Mach. Intell., Vol. 13, No. 4, April 1991.

[17] D. Lowe, *Perceptual Organization and Visual Recognition*, Kluwer Academic Publishers, 1985.

[18] J. W. McKee, J. K. Aggarwal, "Computer Recognition of Partial Views of Curved Objects," I.E.E.E. Trans. Comput., C-27 (2),126-143, Feb. 1978.

[19] Mohan R., Nevatia R., "Perceptual Organization for Scene Segmentation and Description," I.E.E.E. Trans. on Pat. Recog. Mach. Intell., Vol. 14, No. 6, June 1992.

[20] Nevatia R., Binford T. O., "Description and Recognition of Complex Curved Objects," Art. Intell., Vol. 8, 1977.

[21] Schwartz J. T., Sharir M., "Identification of Partially Obscured Objects in Two Dimensions by Matching of Noisy "Characteristic Curves"," Intl. Journ. of Robotics Research, 6(2):29-44, 1987.

[22] Stein F., Medioni G., "Structural Indexing: Efficient 3-D Object Recognition," I.E.E.E. Trans Pat. Recog. Mach. Intell., Vol. 14, No. 2, Feb 1992.

[23] A. Treisman, "Features and Objects in Visual Processing," Scientific American, Vol. 225, Nov. 1986.

[24] J. L. Turney, T. N. Mudge, R. A. Volz, "Recognizing Partially Occluded Parts," I.E.E.E. Trans. P.A.M.I., Vol. PAMI-7, No. 4, 410-421, July 1985.

[25] Witkin A. P., Tenenbaum J. M., "On the Role of Structure in Vision," in *Human and Machine Vision*, J. Beck, B. Hope and A. Rosenfeld (eds.), Academic Press, 1983.