Connectionist Networks for Feature Indexing and Object Recognition

Clark F. Olson
Department of Computer Science
Cornell University
Ithaca, NY 14853
clarko@cs.cornell.edu

Abstract

Feature indexing techniques are promising for object recognition because of their ability to eliminate many feature set matches from consideration without much computation. This work exploits another property of such techniques. They have inherently parallel structure and connectionist network formulations are easy to develop. Once indexing has been performed, a voting scheme such as geometric hashing [16] can be used to generate object hypotheses in parallel. We give a framework for the connectionist implementation for such indexing and recognition techniques. With sufficient processing elements, recognition can be performed in a small number of time steps. The number of processing elements necessary to achieve peak performance and the fan-in/fan-out required for the processing elements is determined. These techniques have been simulated on a conventional architecture with good results.

1 Introduction

Techniques that use sets of image features to index sets of model features that may match them (e.g. [6, 7, 8, 9, 14, 15, 16, 17, 19, 21, 22, 23, 24]) are promising for object recognition because of their ability to eliminate many matches from consideration without much computation. In this work, we exploit another beneficial property of such indexing systems. They have inherently parallel structure and can be implemented using very fine-grain parallelism. To this end, we give a framework for the connectionist implementation of feature set indexing and object recognition techniques. This framework uses a very large number of simple processing elements communicating on a fixed pattern. In such a connectionist framework, eliminating groups from consideration is not necessary for fast recognition, since the work on separate feature sets can be done in parallel. Matches can simply be given varying levels of likelihood, as in Bayesian formulations of indexing [8, 21], which are then used to determine which objects are likely to be present in the image.

This work assumes that the image features have already been extracted, and it uses these features as the primary tool for recognition. Most feature detection techniques can be implemented without difficulty in a connectionist manner, but the examination of such techniques is outside the scope of this work. While the use of sparse features can be limiting on the performance of such a recognition system [11, 12], additional verification techniques can be used to discard incorrect hypotheses.

Once features have been extracted from the image, recognition can be performed in a small number of time steps. When we have sufficient processing elements (with fixed fan-in and fan-out) to achieve maximum parallelization, the time required is $O(\log n)$. If the processing elements can broadcast a value on $O(n)$ dedicated connections in $O(1)$ time and maximize and sum $O(n)$ inputs on dedicated connections in $O(1)$ time then the running time is $O(1)$. Simulation of this system on sequential hardware has yielded good results.

If we use randomization concepts to limit the number of image feature sets that must be considered, maximum parallelization can be achieved with $O(mn^k)$ processing elements, where m is the number of features in the object model, n is the maximum number of image features at which high accuracy is maintained, and k is the number of matches between model and image features necessary to perform indexing. Note that this number of processing elements is not required, but processing time and/or accuracy will suffer as the number of processing elements is decreased. Alternatively, we can use $O(n^k + I)$ processing elements and achieve maximum parallelization, where I is the number of processing elements necessary to cover the indexing space.

It should be noted that parallel implementations of indexing systems (particularly geometric hashing) have been previously studied (e.g., [4, 20, 21]), but these works have primarily considered general-purpose parallel-processing systems and have not achieved full parallelization since they retain sequential examination of image feature sets.

The remainder of this paper is organized as follows. Section 2 reviews the use of indexing and voting to perform object recognition. Section 3 gives a framework for the connectionist implementation of indexing systems for object recognition. The ideal number of processing elements and their required fan-in and fan-out is determined in Section 4. Section 5 describes how some current indexing methods can be implemented in this framework. Simulations that were performed to test the techniques described here are described in Section 6. Section 7 discusses some interesting issues related to connectionist indexing and object recognition. Finally, Section 8 summarizes the paper.

2 Object recognition using an election

Feature set indexing systems [6, 7, 8, 9, 14, 15, 16, 17, 19, 21, 22, 23, 24] attempt to quickly determine which sets of object features could have projected to specific sets of image features under certain noise assumptions. Such systems consist of two phases. First, a preprocessing phase is performed in which an index table is created. In this phase, the feature sets of some predetermined cardinality are stored in one or more locations in the index table according to parameters describing their geometry (and possibly other attributes). Ideally, these parameters are invariant to the transformations and projections that may be applied to the object in forming its image. Recognition is performed during the second phase. In this phase, the parameters of the sets of image features are used to index sets of object features in the index table that may match them.

One of the primary uses of indexing has been in voting schemes like the geometric hashing system of Lamdan *et al.* [16]. This system uses indexing to determine the number of additional feature matches that are brought into alignment by the same transformation that aligns various basis sets of matches and is thus conceptually similar to the alignment method [13]. The geometric hashing method allows much of the work to be moved off-line and uses randomization to achieve improved efficiency. This shift of work off-line and the natural parallelism of the technique make such methods excellent for connectionist implementation, since we wish to keep our parallel processing elements as simple as possible.

Let k be the number of feature matches necessary to perform indexing. We'll call a set of k image features an *image group* and a set of k object features a *object group*. Similarly, a set of $k-1$ features will be called an *object basis* or an *image basis*. If we restrict ourselves to conventional indexing methods using point features, then $k = 2$ for two-dimensional translation only, $k = 3$ for two-dimensional translation, rotation, and scale, and $k = 4$ for full three dimensional transformations under weak-perspective.

The basic procedure for recognizing an object using an election is:

1. Generate an index table storing the relevant information from each possible

object group.

2. Detect the features in the image.

3. Choose a random image basis.

4. Form all image groups comprised of the image basis and one additional feature.

5. Index possible matches for each image group that was formed in step 4.

6. For each model group that is indexed, record a vote for the model basis that is matched to the random image basis.

7. If some model basis receives enough votes then an object hypothesis has been found. Otherwise, Steps 3-7 are repeated until enough image bases have been examined to rule out the presence of the object in the image with high probability.

Step 1 is a preprocessing step and is performed prior to runtime. Step 2 is performed once per image. This step will not be examined in this paper. Steps 3-7 form the recognition phase of the algorithm. The criteria for stopping the procedure will be discussed further in Section 4.

3 Connectionist framework

Feature set indexing and object recognition by election translate naturally into a connectionist framework, where we can use massive parallelism to examine possible matches at the same time. Our framework breaks the process into layers of processing elements that correspond to the following conceptual entities.

Layer 1: Image features
Layer 2: Image feature groups
Layer 3: Matches between image groups and model groups
Layer 4: Matches between image bases and model bases
Layer 5: Model hypotheses

The first three layers perform the indexing and the final two layers perform the election. Here we assume that the preprocessing has already occurred and that the model group parameters have been loaded into the appropriate processing elements. The computations that take place in each of these layers and the communication pattern between them are as follows:

Layer 1: Image features. The parameters of the image features comprise the input layer. This layer does not perform any computation, the parameters are simply fed to the appropriate processing elements in the following layer.

Layer 2: Image feature groups. The processing elements in the second layer correspond to the image groups (feature sets of size k). These processing elements generate the indexing parameters, which are typically easily computed functions of the feature parameters. Each of the processing elements in this layer receives input from the processing elements in the previous layer corresponding to the k features that make up the image group, and sends output to each of the group matches that contain this image group.

Layer 3: Group matches. The processing elements in the third layer represent matches between image groups and model groups. They take input from the processing element corresponding to a particular image group and they are tuned to the indexing parameters of a particular model group such that they output a high value if the indexing parameters of the input image group are close to the parameters of the model group. This value is ideally some likelihood function of the group match being correct. These processing elements send output to the processing elements corresponding to the basis matches contained in the group match.

Layer 4: Basis matches. The fourth layer performs the voting for each possible match between a model basis and an image basis. Each processing element at this layer receives output from each of the processing elements at the previous layer that matches the given image basis and model basis. The sum (or some other combination) of these likelihoods is used as the score for the basis match. A large score at this stage indicates that the basis match is likely to be correct.

Layer 5: Model hypotheses. The final layer determines which objects are present in the image by considering the information from each of the model bases for the object. These processing elements receive input from each of the basis matches for this model and output the maximum of the inputs. If more than one instance of an object model may be present, this layer may be omitted and the basis matches can be used to indicate where the multiple object models are present in the image.

Figure 1 shows the flow of information in this process. The image feature parameters feed into image group processing elements, which determine the indexing parameters. These are then passed to the group match processing elements, which determine the likelihoods of each model group matching each image group. The basis match processing elements take the scores from each of the group matches that contain the appropriate basis match and perform a summing operation to get the score for the basis match. Finally, the model processing elements output a high score if any of their basis matches have a high score.

It should be noted that we do not select a subset of the possible matches for a particular image group be indexed in this framework. Each possible match is assigned a score which, is then propagated through the network. This is similar to Bayesian formulations [8, 21], except that we do not eliminate matches in the connectionist framework even if they have a small likelihood of being correct in the Bayesian formulation. In addition, we note that the pattern of connections does

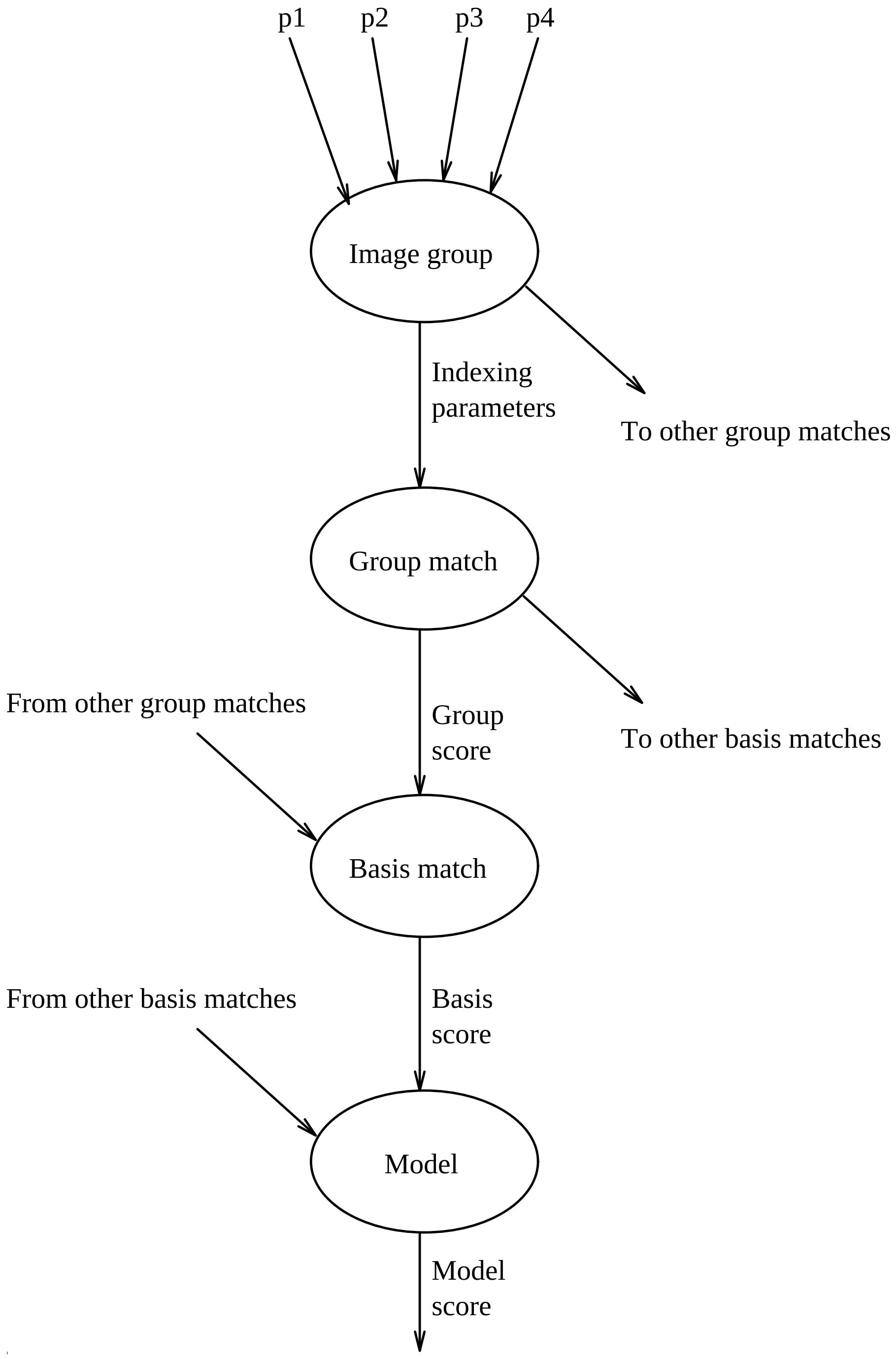


Figure 1: The flow of information in the recognition process.

not change for different images. We need only feed the feature parameters into the processing elements in the first layer.

4 Ideal number of processing elements

For any object that has at least some fraction, f, of its features appearing in the image, it is possible to achieve any fixed probability of examining a correct image basis while examining far less than each of the $O(n^{k-1})$ image bases. If we choose some number, x, of image bases with cardinality $k-1$ at random and examine only those bases, the probability of not examining any correct bases for a particular object is bounded by approximately:

$$p \leq \left(1 - \left(\frac{fm}{n}\right)^{k-1}\right)^{x}$$

This is true since the probability of any particular image feature being from this object is at least $\frac{fm}{n}$ in this case. If we require this probability to be less than some small constant, δ, (e.g. 0.001), we have:

$$\left(1 - \left(\frac{fm}{n}\right)^{k-1}\right)^{x} \leq \delta$$

Taking the logarithm of both sides yields:

$$x \ln\left(1 - \left(\frac{fm}{n}\right)^{k-1}\right) \geq \ln \delta$$

Solving for x, we get:

$$x \geq \frac{\ln \delta}{\ln\left(1 - \left(\frac{fm}{n}\right)^{k-1}\right)}$$

We can use the approximation $\ln(1+\alpha) = \alpha$:

$$x \geq \left(\frac{n}{fm}\right)^{k-1} \ln \frac{1}{\delta}$$

We can thus examine $O(\frac{n^{k-1}}{m^{k-1}})$ image bases and achieve high accuracy (f and δ are constants). For each image basis, we consider each of the $\binom{m}{k-1}$ object bases as possible matches and for each possible match, we examine each of the $(n-k+1)(m-k+1)$ additional feature matches to determine if the match between the bases is correct. In total, we examine $O(\frac{n^{k-1}}{m^{k-1}}) \cdot O(m^{k-1}) \cdot O(mn) = O(mn^{k})$ group matches and achieve probability $1-\delta$ that a correct image basis is examined for a particular object.

Layer	Description	Number	Fan-In	Fan-Out
1	Image features	n	1	$\binom{n-1}{k-1}$
2	Image groups	$\binom{n}{k}$	k	$\sim \alpha m$
3	Group matches	$\sim \alpha m n^k$	1	1
4	Basis matches	$\sim \alpha n^{k-1}$	$\sim mn$	1
5	Models	1	$\sim \alpha n^{k-1}$	1

Table 1: The number of processing elements and fan-in/fan-out necessary per object model at each layer.

In the connectionist implementation, we can simply select the appropriate number of random image bases to examine in parallel and achieve probability $1 - \delta$ of examining a correct image basis. The processing elements corresponding to the remainder of the matches are unnecessary. We thus gain maximum parallelization when we use $O(mn^k)$ processing elements in this case. Of course, in practice, the number of processing elements available will set some limit on the number of image features that can be handled with accuracy δ. Performance will be good on images of this complexity or less and will degrade gracefully as the image complexity rises past this level. Alternatively, we can let processing elements perform more than one task to maintain a high performance at the cost of additional processing time.

Note that perceptual organization can be performed to determine sets of features that are likely to come from the same object. In this case, we can use these sets as the image bases that we examine. This will result not only in better performance in finding objects, but it will allow the number of image bases that we need to examine to be reduced.

We can now consider the ideal number of processing elements at each level, as well as the fan-in and fan-out required for each processing element. We need to examine approximately αn^{k-1} basis matches (where $\alpha = \frac{\ln \frac{1}{\delta}}{f^{k-1}}$, and thus is constant when f, δ, and k are set), since for each image basis we must examine each of the $(k-1)!$ permutations of each of the $\binom{m}{k-1}$ model bases that may match it. This implies that the number of group matches that must be examined is approximately $\alpha(m-k+1)(n-k+1)n^{k-1}$. The required fan-in and fan-out of the processing elements in each level can be determined from the connection pattern described in the previous section. See Table 1. If we don't have individual processing elements with this fan-in/fan-out capability, we can build up the fan-in/fan-out using a tree of simple elements.

It is possible to reduce the ideal number of processing elements at the group match stage by combining processing elements for matches that share the same (or very similar) indexing parameters. These processing elements perform the same function, since they are matching image groups against the same set of parameters. This would then require $O(n^k + I)$ processing elements, where I is the number of processing elements necessary to cover the space of indexing parameters.

This analysis of the ideal number of processing elements is on a per object basis. As the number of objects in the database increases, the ideal number of processing elements increases linearly. Alternatively, we can process the objects sequentially, using the same processing elements for each object.

5 Connectionist indexing

This section discusses how some recent indexing schemes can be implemented in the connectionist framework that we have described.

5.1 Geometric hashing

The geometric hashing system [16] uses the property that the relative coordinates of a point, with respect to three other coplanar points, are invariant to affine transformations and orthographic projection. This is used to build an indexing system.

The geometric hashing system can be easily implemented in the framework discussed above. The processing elements corresponding to the image point groups simply determine the relative coordinates of the fourth point in terms of the first three. The processing elements corresponding to the group matches are assumed to know the relative coordinates of the correct model group. These processing elements output a high value if the relative coordinates input to them are close to the model group relative coordinates held there.

While Lamdan *et al.* perform binning to determine which image and model groups may match, a better way to determine the output for the processing elements corresponding to group matches is to use a Bayesian formulation as is done by Costa *et al.* [8] and Rigoutsos and Hummel [21]. In these works, the geometric hashing problem is recast in probabilistic terms. Rather than indexing being a discrete event where each image group either indexes or doesn't index each specific model group, each possible match is given a score signifying the likelihood that it is correct. In these formulations, only the matches with large scores are examined. In a connectionist implementation, we need not ignore any matches that include a particular basis match and we can thus gain additional accuracy.

In our connectionist framework, the processing elements corresponding to the group matches determine the likelihood of each group match being correct using the image group information derived in the previous layer. The next layer, corresponding to the basis matches, sums (or otherwise combines) the results from the group matches. Finally, we can use a processing element to perform a maximum operation on the basis matches from each model to determine if the model is likely to be present in the image.

5.2 Indexing 3D models (Clemens and Jacobs)

Research has shown that there are no invariants for single views of general three-dimensional point sets in two-dimensional images [5, 7], but Clemens and Jacobs [7] have built an indexing system that is capable of indexing three-dimensional point sets under weak-perspective by representing groups of four model points over a two-dimensional surface of a four-dimensional index table. The sphere of possible viewing directions is discretized and each model group is represented in the table from each of the viewing directions.

While much of the connectionist implementation of this work would be similar to that of geometric hashing, this indexing technique involves the additional complication that each model group is represented from many directions on the viewing sphere. This can be handled by including, in Layer 3, a processing element corresponding to each of the viewing directions for each match. An additional layer is included (Layer 3.5) in which the maximum value over all of the viewing directions for the matches is determined as the output for each group match.

Jacobs [14] has modified this work for affine transformations such that sampling the viewing sphere is not necessary. In this system, each group is represented by a line in each of two two-dimensional parameter spaces. To index model groups that may match an image group, the image group parameters are determined and model groups are indexed in both spaces. The intersection of the two sets of model groups correspond to the possibly matching groups. To implement this system using a connectionist network, we use one or two processing elements to store the two lines and these calculate the distance of their input from these lines. A score for the match is then generated using the combination of these two distances. It should be noted that this formulation of the indexing problem uses $k = 5$ and thus more processing elements would be necessary to achieve maximum parallelization.

5.3 Indexing 3D models (Weinshall)

Weinshall [24] describes an indexing method for weak-perspective using model-based invariants. The basic idea is that a function can be constructed for each model group that, given some image group, evaluates to zero (in the noiseless case) if and only if the model group could project to the image group. While Weinshall describes a method of implementing an indexing system using these ideas, we can use the model-based invariants directly in a connectionist implementation. Rather than viewing each model group as representing some space in an index table, we can simply compute the model-based invariant that Weinshall describes, and use a function of this invariant as a score for the match. A single processing element can thus easily determine the score for a group match.

The model-based invariant is given directly in terms of image feature point coordinates, so no parameters specific to an image group need be generated. The network implementation of these ideas thus does not require a layer of processing elements corresponding to the image point groups. The processing elements corresponding to the group matches compute the invariant functions and output a high value if they are close to zero (the value a noiseless correct match would yield.)

An alternative metric that could be used is the transformation metric defined in [25], which is similar to the model-based invariants. Our experiments indicate that a normalized version of this metric is more accurate in discriminating between correct and incorrect matches in the presence of noise.

5.4 Probabilistic indexing

Olson [19] describes a probabilistic indexing method. This method uses the probabilistic peaking effect [1, 2, 5] to create an indexing system. This effect is that the probability densities of angles and ratios of distances of image features peak strongly at the values taken by the model features that correspond to them. Thus, these parameters change little over a substantial range of viewing directions. This effect is used to build an indexing system capable of indexing three-dimensional point groups with cardinality three under general weak-perspective transformations. The disadvantage to this system is that false negatives (cases where the correct group of model features are not indexed) must be allowed. This system determines probabilistically which model points groups are most likely to have projected to specific groups of image points. Thus, the probability of indexing the correct groups is considerably higher than the probability of indexing any particular incorrect group.

While probabilistic indexing systems are capable of indexing using groups of size three, this is probably not suitable for an election method, due to the significant percentage of false positive matches indexed for this case. Probabilistic indexing has been extended to handle point groups larger than three points [18]. When using point groups of size four, we can index many fewer false positives than when using groups of size three, while still indexing a significant fraction of the correct groups. To perform probabilistic indexing using point sets with cardinality greater than three, we perform indexing on each of the subgroups of three points. If a model group is indexed by enough of the subgroups then the group is considered indexed by the entire group. For four point groups we thus need either four processing elements for each group match to perform the probabilistic indexing on the four subgroups of three points or a single processing element capable of performing all four calculations. An additional calculation must then be performed to combine the results of the previous calculations. An additional processing element can be used to perform this task.

The connectionist implementation of the probabilistic indexing system in this framework is straightforward. The processing elements corresponding to the basis matches still receive information from each of the group matches that contain that basis match, but these processing elements will now output a high value based on a lower number of high input values, since not all of the correct matches will be indexed. This may be acceptable, since a considerably smaller percentage of the incorrect matches will also be indexed for most cases.

6 Results

These techniques have been simulated using a conventional computer architecture. To determine which model groups may have projected to each image group, a variant of the transformation distance metric described by Weinshall and Basri [25] was used. These techniques determine how well a set of point correspondences can be brought into alignment by a rigid transformation under weak-perspective. The transformation metric is:

$$N_{tr} = \frac{1}{2}(x^T Bx + y^T By - 2\sqrt{x^T Bx \cdot y^t By - (x^T By)^2})$$

where x and y are vectors of the x- and y-coordinates of the image points after translating them such that their center of mass is at the origin and B is what Basri and Weinshall call the *characteristic matrix* of the model points and is given by $B = (P^+)^T P^+$ where

$$P = \begin{pmatrix} X_1 & Y_1 & Z_1 \\ & \vdots & \\ X_n & Y_n & Z_n \end{pmatrix}$$

P^+ is the pseudo-inverse of P and (X_i, Y_i, Z_i) are the coordinates of the ith point of the model group after translating the model group points such that their center of mass is at the origin.

The transformation metric, N_{tr}, will be zero if the points can be brought into alignment exactly by a rigid transformation and otherwise measures how far the optimal affine transformation is from rigid.

We normalize N_{tr} using:

$$N_{norm} = \frac{N_{tr}}{\|x\| \cdot \|B\| \cdot \|y\|}$$

To perform an indexing-like function using this metric, we must have $k = 4$. The processing elements corresponding to group matches should then output a high score if the points can be brought into alignment closely using a rigid transformation. We achieve this using the following function:

$$\frac{\lambda}{\lambda + N_{norm}}$$

Here λ is an empirically determined constant. If the points can be brought into alignment perfectly, then the output is 1, otherwise the output falls off to a low score depending on how closely they can be brought into alignment by a rigid transformation.

In experiments on random object groups that were projected using the perspective projection with bounded noise added ($\epsilon = 1.0$) and using $\lambda = .0003$, correct matches received an average score of .701, while incorrect matches received an average score of .0036. This procedure thus performs well in discriminating between correct and incorrect matches.

The entire system has been simulated on recognition problems consisting of 20 models points projected onto the image using the perspective projection and 30 noise points. Once again bounded noise was added ($\epsilon = 1.0$) to the locations of the projected object points. In these experiments, an average of 6.16 correct basis matches were found for each correct object, thus objects were detected when they were present in the image. In addition, out of the 1.3 million incorrect basis matches examined per object, an average of 3.13 false positives were found that required verification to discard (a false alarm rate of $2.41 \cdot 10^{-6}$).

Figure 2(a) shows an example recognition problem. For this example, the simulation found four basis matches that scored highly. Three of these were correct. Figure 2(b) shows the best alignment according the transformation metric of Basri and Weinshall [25] of the highest scoring match, which was one of the correct matches. All of the object points are transformed close to their corresponding image point. Most of the inaccuracy in these experiments is due to perspective distortion, which is not modeled by Basri and Weinshall's transformation metric, rather than image noise.

Figure 2(c) shows the best transformation aligning the basis match that results in a false positive for this image. There are two reasons why a false positive resulted from this match. First, a single transformation brings several of the model points very nearly into alignment with image points as shown in the figure. Second, both the image basis and the model basis in the match were close to colinear. This means that we can rotate the model around this line and these points will remain in close alignment. For each other model point, one of these rotations is likely to bring it into alignment with some image point, and thus each group containing this basis is likely to get a high score. False positives could be eliminated from this system by not considering the image bases that are nearly colinear.

Figure 3 shows the results of using these techniques to recognize a stapler in two real images. In this case, feature points were determined using an interest operator [10] and their locations were used in the recognition process.

7 Discussion

This section discusses the overall running time of the system and the possibility of learning the object models.

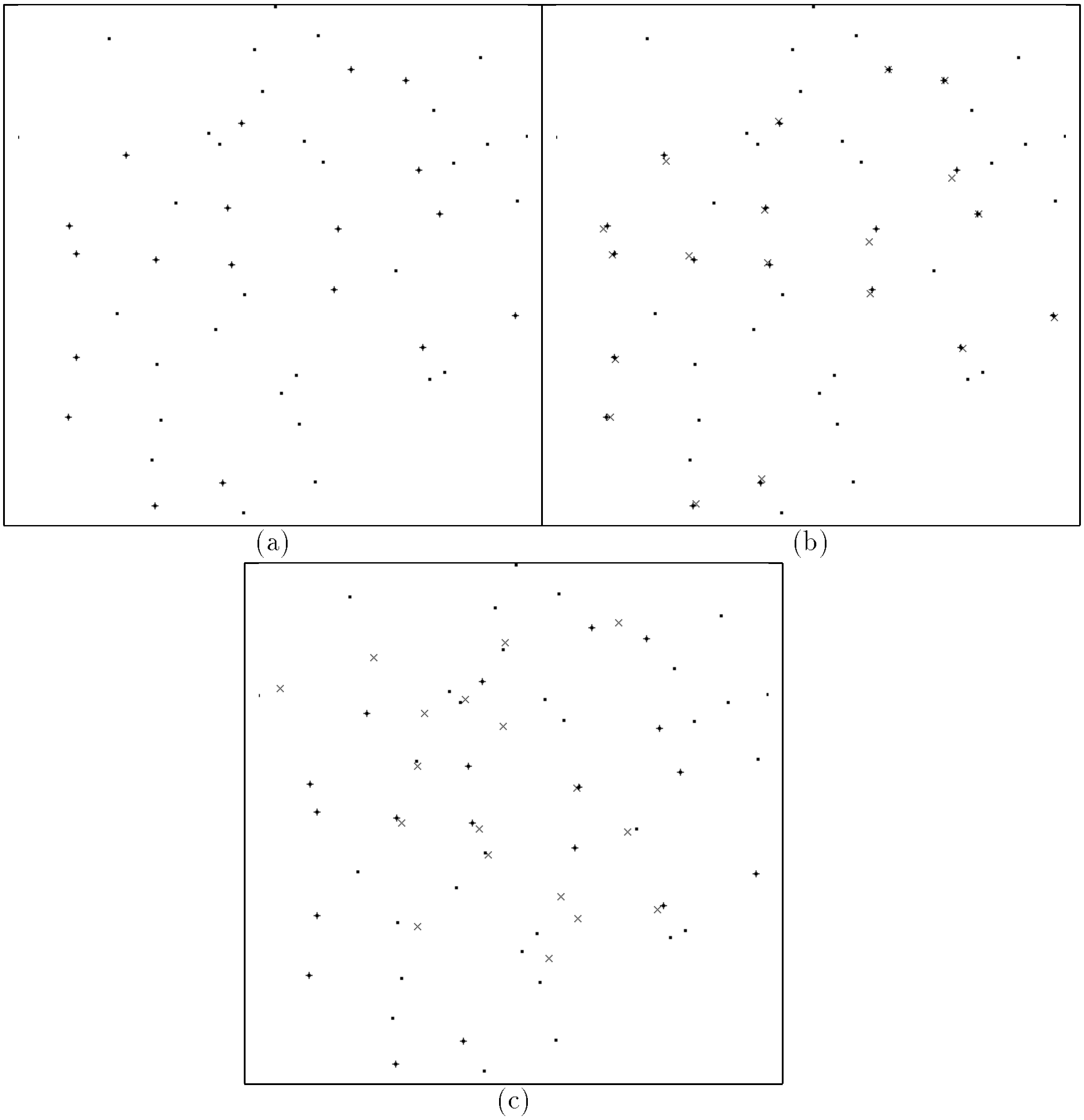
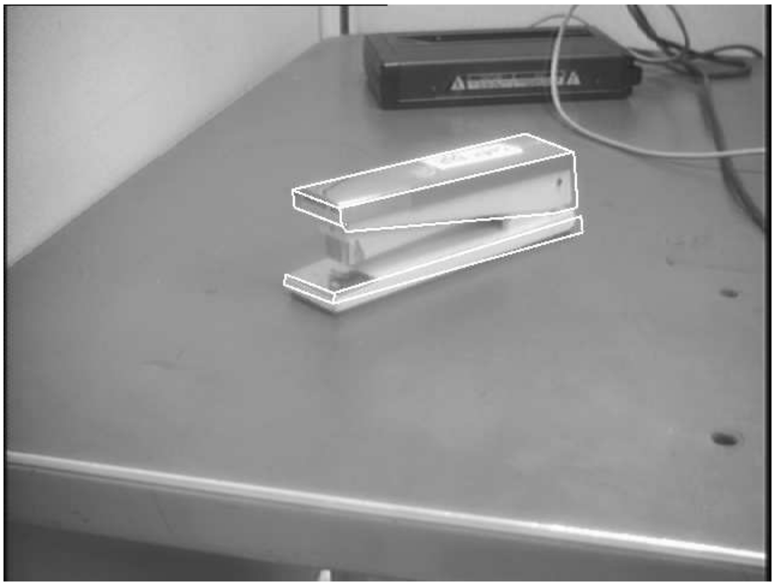
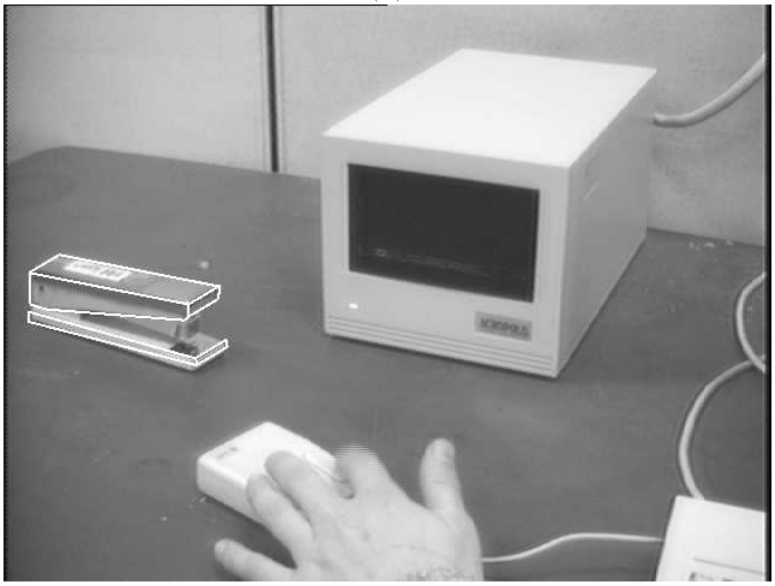


Figure 2: A sample recognition problem. (a) The image. The stars are the object points and the dots are the noise points. (b) The best transformation of the object points. The X's are the transformed locations of the object points. (c) The false positive that was found for this image.



(a)



(b)

Figure 3: Two examples recognizing a three-dimensional object in a real image.

7.1 Running time

First, note that we must have $n \geq fm$ to recognize a model in an image, so we can assume $m = O(n)$. If we have sufficient processing elements with $O(n)$ fan-in and fan-out capabilities[1], then the running time is $O(1)$, since the computation required at each layer can be computed in constant time and there are a constant number of layers. (The $O(n^{k-1})$ fan-in and fan-out capabilities can be simulated using a tree of processing elements of a constant height proportional to k.) For conventional processing elements with limited fan-in and fan-out capabilities, we can chain processing elements in a tree with $O(\log n)$ height to provide the necessary capability. Thus, in this case, the running time of the system is $O(\log n)$. Such a system would be fast in practice, in either case, since the computations performed by each processing element are simple.

7.2 Learning

In the framework described above, we have assumed that the necessary indexing parameters are loaded into the correct processing elements prior to recognition time. An interesting possibility to consider is whether these parameters could be learned in a supervised or unsupervised manner through the examination of examples.

Note that this system does not use weights associated with the connections. Given the number of object and image features that we wish to handle, the communication pattern is completely fixed, regardless of the specific object and image. The parameters that vary in this system are the indexing parameters stored at the processing elements corresponding to group matches. These parameters could be trained by a learning algorithm.

While it is conceivable that the indexing parameters for an entire object could be learned concurrently, a much superior learning strategy would be to isolate each of the processing elements corresponding to the group matches and train them separately. If this can be achieved, learning the parameters would be simple. Otherwise, learning the parameters would be a formidable task.

At this time, we have made no attempt to train such a system.

8 Summary

This paper has considered connectionist methods for performing feature set indexing and object recognition. We first summarized the basic algorithm for performing object recognition using an election. Such systems use indexing to determine which sets of model features could have projected to the sets of image features that contain a basis set of features. Votes are recorded for each of the basis sets of model features that are

[1] This assumes the ability to find the maximum and sum the inputs on these dedicated connections in $O(1)$ time.

indexed and such basis sets that accumulate many votes are considered recognition hypotheses.

We then described a connectionist framework for the implementation of indexing and voting techniques to perform object recognition. This framework uses a large number of simple processing elements performing in parallel and communicating on a fixed connection network. The ideal number of processing elements and the fan-in and fan-out required by the processing elements was determined to be $O(mn^k$, where m is the number of model features, n is the number of image features, and k is the number of feature matches necessary to perform indexing. When the ideal number of processing elements are present the system can perform object recognition in $O(\log n)$ time, even when the fan-in and fan-out of the processing elements is limited. If the processing elements can broadcast on $O(n)$ connections and select the maximum and sum of $O(n)$ inputs on dedicated connections in $O(1)$ time, then object recognition can be performed in $O(1)$ time.

We next described how several current indexing systems can be implemented in this framework to perform fast object recognition. Finally, we have described the results of simulating the performance of these techniques on a conventional architecture using real and synthetic images.

References

[1] J. Ben-Arie. The probabilistic peaking effect of viewed angles and distances with application to 3-d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(8):760–774, August 1990.

[2] T. O. Binford, T. S. Levitt, and W. B. Mann. Bayesian inference in model-based machine vision. In *Uncertainty in Artificial Intelligence 3*, pages 73–95. Elsevier Science Publishers B.V., North-Holland, 1989.

[3] R. M. Bolle, A. Califano, and R. Kjeldsen. A complete and extendable approach to visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(5):534–548, May 1992.

[4] O. Bourdon and G. Medioni. Object recognition using geometric hashing on the connection machine. In *Proceedings of the International Conference on Pattern Recognition*, volume 2, pages 596–600, 1990.

[5] J. B. Burns, R. S. Weiss, and E. M. Riseman. View variation of point-set and line segment features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(1):51–68, January 1993.

[6] A. Califano and R. Mohan. Multidimensional indexing for recognizing visual shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):373–392, April 1994.

[7] D. T. Clemens and D. W. Jacobs. Space and time bounds on indexing 3-d models from 2-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1007–1017, October 1991.

[8] M. Costa, R. Haralick, T. Phillips, and L. Shapiro. Optimal affine-invariant point matching. In *Proceedings of the SPIE, volume 1095, Applications of Artificial Intelligence VII*, pages 515–530, 1989.

[9] P. J. Flynn. 3d object recognition using invariant feature indexing of interpretation tables. *CVGIP: Image Understanding*, 55(2):119–129, March 1992.

[10] W. Förstner. Image matching. Chapter 16 of *Computer and Robot Vision*, Vol. II, by R. Haralick and L. Shapiro, Addison-Wesley, 1993.

[11] W. E. L. Grimson and D. P. Huttenlocher. On the sensitivity of geometric hashing. In *Proceedings of the International Conference on Computer Vision*, pages 334–338, 1990.

[12] W. E. L. Grimson, D. P. Huttenlocher, and D. W. Jacobs. A study of affine matching with bounded sensor error. *International Journal of Computer Vision*, 13(1):7–32, 1994.

[13] D. P. Huttenlocher and S. Ullman. Recognizing solid objects by alignment with an image. *International Journal of Computer Vision*, 5(2):195–212, 1990.

[14] D. W. Jacobs. Space efficient 3d model indexing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 439–444, 1992.

[15] T. F. Knoll and R. C. Jain. Recognizing partially visible objects using feature indexed hypotheses. *IEEE Journal of Robotics and Automation*, 2(1):3–13, 1986.

[16] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson. Affine invariant model-based object recognition. *IEEE Transactions on Robotics and Automation*, 6(5):578–589, October 1990.

[17] R. Mehrotra and W. I. Grosky. Shape matching utilizing indexed hypotheses generation and testing. *IEEE Transactions on Robotics and Automation*, 5(1):70–77, 1989.

[18] C. F. Olson. Probabilistic indexing: Recognizing 3d objects from 2d images using the probabilistic peaking effect. Technical Report UCB//CSD-93-733, Computer Science Division, University of California at Berkeley, May 1993.

[19] C. F. Olson. Probabilistic indexing for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(5):518–522, May 1995.

[20] I. Rigoutsos and R. Hummel. Massively parallel model matching: Geometric hashing on the connection machine. *Computer*, pages 33–41, February 1992.

[21] I. Rigoutsos and R. Hummel. Distributed Bayesian object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 180–186, 1993.

[22] I. K. Sethi and N. Ramesh. Local association based recognition of two-dimensional objects. *Machine Vision and Applications*, 5:265–276, 1992.

[23] F. Stein and G. Medioni. Efficient two dimensional object recognition. In *Proceedings of the International Conference on Pattern Recognition*, pages 13–17, 1990.

[24] D. Weinshall. Model-based invariants for 3-d vision. *International Journal of Computer Vision*, 10(1):27–42, 1993.

[25] D. Weinshall and R. Basri. Distance metric between 3d models and 2d images for recognition and classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 220–225, 1993.