Abstract

A novel method for computer recognition of occluded objects using eigenspace projection techniques and a subspace graph search is described. Examples using both non-occluded and occluded aircraft images are demonstrated.
Overview

• Introduction

• The Eigenspace Approach to Object Recognition

• Computer Recognition of Non-occluded Objects

• Computer Recognition of Occluded Objects

• Conclusions
Introduction

Creating an accurate, unsupervised object recognition system is a high priority goal of both the military and industry. The focus here is recognition of aircraft, but the method can be used for any type of object.

• A robust general object recognition system requires:
  1. extracting necessary object features from sensed data, and
  2. classifying the extracted features for object recognition.

• A major problem with typical feature extraction process (in addition to computational complexity): validity of extracted features depends on sensor noise, sensor resolution, and lighting conditions.
  – Accurate object feature extraction is rare in natural scenes, adversely affecting subsequent object classification procedure as well as final recognition decision.
  – Use of image eigenvectors as features avoids many of these problems—eigenvectors provide “global” image features.
In computer-based object recognition work, two different approaches have been used for representing object models:

1. CAD-based models (this name is derived from the similarity of such models to a Computer Aided Design (CAD) drawing of the object), and

2. non-CAD-based models, which include subspace appearance representations such as eigenspace models.

Systems using CAD-based models match a selected set of object features with the associated model features for recognition.

- Requires high computational complexity: system must process sensed data, extract desired features, sort out relationships between features, estimate sensor pose to project the model onto a 2-D plane, and try to match model and image features based on some reasoning scheme.
Accurate CAD model generation of complex objects cannot be easily created due to nonlinear curves and numerous corners such objects possess.

Even if an accurate model can be generated, the computation time required to extract necessary features from such a model is another challenging task.

Many complex objects have certain regions altered due to an event such as opening a hatch on a tank, adding a ski rack to an automobile, etc. Currently, such variations can only be represented coarsely in CAD models.
The alternative method of object model representation is generically called “non-CAD-based.”

- Gathers all possible appearances (or rather as many as practical) of an object captured from various sensor angles and lighting conditions.
  - Requires an enormous amount of memory and storage space.

- For recognition, the matching scheme compares all known appearances in the model with the actual appearance of an image object (similar to work by Turk and Pentland for recognizing human faces).

- This model representation helps avoid the need to create elaborate CAD object models, but presents new problems: storing large amounts of data and the difficulty of template matching this large amount of data with an image for all possible appearances.
The primary drawback to non-CAD-based models is the huge amount of data that must be stored and compared to some new image.

- Significant data reduction can be obtained by representing objects in a subspace using the eigenvectors of an objects appearance.

- This alternative representation accomplishes two objectives:
  1. compact model representation, and
  2. elimination of complex data processing for feature extraction.

This subspace will be called an eigenspace.
Mathematical Methods

To explain the mathematics behind the eigenspace approach, we first turn to the Karhunen-Loeve (K-L) expansion, sometimes referred to as principle component analysis, which transforms data into a set of orthonormal vectors and corresponding coefficients.

- The optimal set of orthonormal vectors and coefficients which minimize data representation error (as in the Euclidean $\ell^2$ norm) are the eigenvectors and their corresponding eigenvalues.

- Consider a gray-scale image as matrix $A$. Now create the matrix

$$C = AA^T$$

which will always be a square and symmetric matrix. For $C$, the eigenvectors $x$ and eigenvalues $\lambda$ must satisfy the equation

$$Cx = \lambda x.$$
Lets use a quick, simplified example. Suppose matrix $C = \begin{bmatrix} 4 & -2 \\ 1 & 1 \end{bmatrix}$.

Then we can see that

\[
\begin{bmatrix} 4 & -2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \end{bmatrix} = 3 \begin{bmatrix} 2 \\ 1 \end{bmatrix}
\]

(3)

and

\[
\begin{bmatrix} 4 & -2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix} = 2 \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\]

(4)

Thus the two distinct eigenvectors of $C$ are $x_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$ and $x_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, which have corresponding eigenvalues of $\lambda_1 = 3$ and $\lambda_2 = 2$, respectively. The eigenvectors of $C$ span the column space of the original matrix $A$. The magnitude of an eigenvalue $\lambda$ represents the dominance of the corresponding eigenvector $x$ in representing the null space of matrix $(C - \lambda I)$. Note we could have just as well used $C = A^T A$. 
A transformation which factorizes a rectangular matrix into a multiplication of three matrices is called singular value decomposition (SVD). Consider matrix $A$ with dimension $n \times m$ (i.e., $n$ rows and $m$ columns).\footnote{Following common practice, matrix dimension is given as “rows \times columns,” but image resolution (in pixels) will be given as “columns \times rows.”}

- If we apply SVD to matrix $A$ we obtain

$$A = U \Sigma V^T = [u_1 \quad u_2] \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix} [v_1^T \quad v_2^T]$$

where $S$ is a diagonal matrix with its diagonal elements representing the singular values $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \ldots \sigma_k$. Furthermore, $u_1$ is $n \times k$, $u_2$ is $n \times (n - k)$, $v_1^T$ is $k \times m$, and $v_2^T$ is $(m - k) \times m$, and $k \leq n, m$.

- The decomposition provides matrix $U$ containing a set of orthonormal eigenvectors for $AA^T$, where the eigenvalues of $AA^T$ are $\sigma_1^2, \sigma_2^2, \sigma_3^2, \ldots, \sigma_n^2$. On the other hand, matrix $V$ contains an orthonormal set of eigenvectors for matrix $A^T A$. 
Implementation Details

For this object recognition system, an object model in eigenspace must be constructed. The process is

1. obtain the required collection of object images from a variety of sensor poses and illumination conditions,

2. compress the image data as much as possible without losing important detail,

3. assemble all the images into a matrix A, and

4. obtain the eigenvectors need to create the model.

Each of these steps will now be addressed.
Figure 1: A typical setup for capturing images using a camera.
Collecting Object Images

Obviously, collecting images for all combinations of object pose and lighting conditions is not practical. We avoid most lighting variations by first thresholding to a binary image of the aircraft.

- Assumption: images to be recognized will be high altitude photos of aircraft on the ground.

- We acquire a set of images for an appearance model by quantizing the possible yaw angle variations into $k$ segments and taking images of an object of interest as we vary the yaw angle.
  - We scale the object in the $z$ direction such that it occupies a majority of the viewing image, collecting a set of images $\{\hat{a}_1, \hat{a}_2, \hat{a}_3, \ldots, \hat{a}_k\}$ where each $\hat{a}_i$ represents an uncompressed image of $p \times q$ pixels captured by the sensor with a specific yaw angle $\phi_i$.
  - Now we must compress the $k$ individual images.
Image Compression

Image data compression is performed by resizing the image multiple times. At each step, we reduce the image to one half of its original dimension for both the $x$ and $y$ axes, resulting each time in an image requiring only $1/4$ the storage space.

- We apply a low-pass spatial filter and a simple decimation technique to an image, retaining key features of the original image while significantly reducing the data size.

- Fig. 2 on the next page shows a sequence of images for the F22 aircraft where we first applied a threshold operation to create a binary image, then at each step a low-pass filter was applied to the current image before scaling the image size by factor of $1/4$. 
Figure 2: A sequence of compressed images of an F22 aircraft.

Visually one can see that the compressed images contain similar “information” as the original image. The lower right image has been reduced to a $5 \times 5$ matrix.
Each iteration results in an approximation of the original image but with less and less detail compared to the original image; there exists a point where we cannot further reduce the size of data without losing critical information.

- We empirically determined this point to be when the size of an image becomes $10 \times 10$ pixels.

- Global features such as eigenvectors are quite robust in the presence of this type of compression.
Creating Matrix $A$

After compression, we form a 1-D column vector of each compressed image by concatenating each row of an image starting from the left top corner and ending at the right bottom corner. Each 1-D column vector $a_i$ of $A$ will thus have $pq/L$ elements where $L$ is the compression factor.

We now form an image data set called matrix $A$, where the column space of $A$ is made up of the $k$ compressed image vectors described earlier such that $A = [a_1 \ a_2 \ a_3 \ \ldots \ a_k]$. Each column of $A$ is composed of a 1-D image vector, and the dimension of $A$ is $pq/L \times k$ (i.e., $pq/L$ rows and $k$ columns).

- The central idea is to describe $A$ using a succinct representation in a new space spanned by a set of eigenvectors.
  - This space is where the actual recognition is performed, thereby reducing both the computational burden as well as the storage requirements for the recognition system.
Obtaining the Eigenvectors

Next we use SVD (as described earlier) to obtain the eigenvectors, which represent an orthonormal basis derived from object appearance matrix $AA^T$.

- The transformation operator between the original appearance space and the eigenspace is a vector inner-product which takes each eigenvector $e_i$ and each compressed image $a_j$ (in its 1-D column vector form) and computes the vector product $e_i^T a_j$.
  
  - This results in a scalar value which represents the projection of image $j$ on to the $i$-axis of the eigenspace.

- Computing all inner-products for this image, $e_i^T a_j$ for all $(i = 1, 2, \ldots, r)$, yields the vector $n_j = [e_1^T a_j \ e_2^T a_j \ \ldots \ e_r^T a_j]$ in the eigenspace, which represents the projection of image $j$ on to all dimensions of the eigenspace.
Performing this same mapping for all images \((j = 1, 2, \ldots, k)\) provides a set of \(k\) vectors in the \(r\)-dimensional (i.e., \(\mathbb{R}^r\)) eigenspace.

- We smoothly connect the end points of the vectors using a spline method to form a differential manifold in the eigenspace, which forms the appearance model of the object whose images are contained in matrix \(A\).

- Fig. 3 on the next page shows the 24 images of a P51 model aircraft which were used to construct the 24 vector-point appearance model in the eigenspace shown in Fig. 4.
Figure 3: Collection of P51 model aircraft images to generate an eigenspace appearance model.
Figure 4: Eigenspace model for the aircraft P51. Note that the actual eigenspace is 24-dimensional space (since in this instance $r = k$), but for visual clarity only three dimensions are shown by using the three most dominant eigenvectors and simply connecting the $R^3$ points with straight lines.
Computer Recognition of Non-occluded Objects

Once we have an object model, recognition is performed in the eigenspace.

- A new test image is transformed into a 1-D vector with \( \frac{pq}{L} \) elements as described earlier, and we then perform vector inner-products of the test image with the \( r \) eigenvectors which span the eigenspace.

- This generates a vector point in the eigenspace which is then compared with the object model residing in the same space.

- The final task: establish a metric to determine how close the projection of the test image is to an object model, yielding a recognition decision.
Figure 5: Five different aircraft used for the recognition experiments, left to right: F22, F14, F4, Tornado, and P51.
Figure 6: Various poses of the five different aircraft used for our experiment, top to bottom: F22, F14, F4, Tornado, and P51.
Figure 7: Eigenspace representations of images in Fig. 6, UL to LR: F22, F14, F4, Tornado, and P51. Note that the actual eigenspaces are 24-dimensional space, but for visual clarity we only show three dimensions by using the three most dominant eigenvectors and simply connecting the $R^3$ points with straight lines.
Figure 8: Ten test images of five aircraft, UL to LR: F4(1), F4(2), F14(1), F14(2), Tornado(1), Tornado(2), F22(1), F22(2), P51(1), and P51(2).

These images were tested for recognition against the stored eigenspace models.
Figure 9: Ten test images projected into the F22 eigenspace, UL to LR: two P51 test images projected, two F4 images projected, two F14 images projected, two F22 images projected, and two Tornado images projected. For each frame, the clustered points represent the appearance model of the F22 and the ‘+’ and ‘∗’ symbols represent projection points for the first and the second test images, shown in Fig. 8, of a particular aircraft, respectively.
Table 1: Results of the first experiment, showing Euclidean distance to nearest eigenspace model point. Smaller numbers represent “better” recognition. This experiment tested for basic recognition.

<table>
<thead>
<tr>
<th></th>
<th>P51</th>
<th>F4</th>
<th>F14</th>
<th>F22</th>
<th>Tornado</th>
</tr>
</thead>
<tbody>
<tr>
<td>P51(1)</td>
<td>0.6609</td>
<td>2.8469</td>
<td>5.2513</td>
<td>5.8356</td>
<td>10.4256</td>
</tr>
<tr>
<td>P51(2)</td>
<td>0.5057</td>
<td>3.0310</td>
<td>6.0694</td>
<td>6.2056</td>
<td>6.7346</td>
</tr>
<tr>
<td>F4(1)</td>
<td>3.6857</td>
<td><strong>1.0907</strong></td>
<td>4.1222</td>
<td>4.9266</td>
<td>11.1408</td>
</tr>
<tr>
<td>F4(2)</td>
<td>2.1117</td>
<td><strong>0.2660</strong></td>
<td>3.9323</td>
<td>6.2060</td>
<td>12.3135</td>
</tr>
<tr>
<td>F14(1)</td>
<td>6.0293</td>
<td>5.9249</td>
<td><strong>2.4245</strong></td>
<td>2.0832</td>
<td>11.1057</td>
</tr>
<tr>
<td>F14(2)</td>
<td>5.3308</td>
<td>4.3390</td>
<td><strong>2.7189</strong></td>
<td>2.0294</td>
<td>11.7778</td>
</tr>
<tr>
<td>F22(1)</td>
<td>7.8852</td>
<td>7.3462</td>
<td>1.4564</td>
<td><strong>5.8664</strong></td>
<td>14.1029</td>
</tr>
<tr>
<td>F22(2)</td>
<td>6.0394</td>
<td>4.5550</td>
<td>3.1053</td>
<td><strong>2.9425</strong></td>
<td>14.6169</td>
</tr>
<tr>
<td>Tornado(1)</td>
<td>4.6805</td>
<td>5.9688</td>
<td>6.0660</td>
<td>8.6051</td>
<td><strong>8.3254</strong></td>
</tr>
<tr>
<td>Tornado(2)</td>
<td>4.2696</td>
<td>4.7761</td>
<td>5.0415</td>
<td>6.3260</td>
<td><strong>7.2840</strong></td>
</tr>
</tbody>
</table>
Figure 10: Test images of three non-aircraft: a toy car, a Lego® toy, and a Mickey Mouse figure. These images were used to test for a false positive tendency of the recognition system.
Table 2: Results of the second experiment, showing Euclidean distance to nearest eigenspace model point. Smaller numbers represent “better” recognition. This experiment tested for false positives.

<table>
<thead>
<tr>
<th></th>
<th>P51</th>
<th>F4</th>
<th>F14</th>
<th>F22</th>
<th>Tornado</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toy car</td>
<td>4.7595</td>
<td>6.2721</td>
<td>2.9862</td>
<td>4.4988</td>
<td>6.2571</td>
</tr>
<tr>
<td>Lego toy</td>
<td>7.3371</td>
<td>9.0815</td>
<td>6.6650</td>
<td>10.0532</td>
<td>15.7431</td>
</tr>
<tr>
<td>Mickey</td>
<td>6.3180</td>
<td>5.0366</td>
<td>6.7925</td>
<td>56.6789</td>
<td>11.1999</td>
</tr>
</tbody>
</table>

Comparing these results with Table 1 results shows some promising capabilities of the recognition system.

- The non-aircraft objects had on average a minimum distance to aircraft appearance models three times as great as the minimum distance of the aircraft projected into their respective eigenspaces.

- One can make a preliminary conclusion that the recognition method can be used to classify aircraft objects from non-aircraft objects.
Computer Recognition of Occluded Objects

When the system as described was tested with occluded objects, performance suffered. As a result, we modified the technique.

- The method we chose to mitigate the occlusion problem is to project the test image on to lower dimensional subspaces of the original eigenspace.

- A subspace of a subspace in this context may be called a sub-eigenspace or be referred to simply as another subspace.

- The technique is to measure how close a match can be achieved when the occluded target image is projected on to a given sub-eigenspace.
  
  - We have found that this technique can result in improved object recognition of occluded objects.

A schematic representation of a simple three dimensional (i.e., $R^3$) eigenspace and its sub-eigenspaces is shown in Fig. 11. The actual experiments for this system utilized an $R^{24}$ eigenspace.
A practical issue arises: how to manage the combinatorial “explosion” associated with selecting the number of subspaces required and then projecting images on to those sub-eigenspaces for measurement.

We use a graph search technique to resolve this problem.
**A* Search Method**

The A* (called “A-star”) search method has been shown to produce optimal solutions to problems in areas such as robotics and artificial intelligence.

- However, the application of this method to image processing has been limited due to its high computational cost.

- Use of a suitable heuristic evaluation function can significantly increase the efficiency of such a search.

- Typically, an evaluation function \( f(n) \) has the following form: 
  \[
  f(n) = g(n) + h(n)
  \]
  where the argument \( n \) represents a particular node of interest, \( g(n) \) is the “cost” from the root node to node \( n \), and \( h(n) \) is an estimated cost from the current node \( n \) to a leaf node.

  - The selection of \( h(n) \) and \( g(n) \) determines the type of a search. For example, if \( h(n) \) is set to zero and \( g(n) = d \) where \( d \) denotes the depth in a search tree, the search becomes a breadth-first search.
In selecting an evaluation function, the following factors should be balanced to maximize the heuristic power:

1. The actual cost of the desired path,

2. the number of child nodes that must be generated to find the desired path, and

3. the computational cost needed to find the function $h(n)$.

- Unfortunately, no concrete algorithm exists for selecting an optimal evaluation function; typically informed intuition proves to be effective.
  - To reduce computational complexity, we skew the graph search technique toward a depth-first search by rewarding a path with most number of nodes.
  - At each node, the $g(n)$ score receives the total cost for the parent node, $f(n - 1)$, where index $n$ specifies a node along the path.
– We define $h(n)$ to represent how well the current subspace (i.e., sub-eigenspace) projection of the object image matches the subspace model.
– Thus $h(n)$ will be the minimum Euclidean distance from the object image projection to the sub-eigenspace model. The closer the projection is to the model, the better the evaluation function score for $f(n)$ should be.
– We therefore define $h(n)$ as

$$h(n) = \frac{1}{\sqrt{\min \left( \sum_{i=1}^{q} (x_i - x_{im})^2 \right)}}$$

where $x_i$ is the $i^{th}$ vector component of a test image in the subspace, $x_{im}$ is the $i^{th}$ vector component of a closest model point from the test image in the same subspace, $q$ represents the dimensionality of the subspace, and $n$ represents a particular node of interest.
Graph Generation and Search Strategy

If any child node has no match (i.e., a projection yields 0) or if any other child node contains a superior match (i.e., $h(n)$ is less than some small threshold value), then terminate the search along that path and try again.

- The actual algorithm to carry out the search follows.

  1. Start with the root node and place it in empty set $N$.
  2. Find a node with minimum evaluation function of $f(n)$ in set $N$.
  3. Remove that node from set $N$.
  4. Expand that node (i.e., generate the child nodes).
  5. Include the newly created nodes into set $N$.
  6. If the current level for expanding node is not a leaf node, or if a pre-determined level is not reached, or if a superior match has not been found, then go to step 2. Otherwise continue.
  7. Collect the nodes on the selected path and terminate.

- The heuristic approach described here requires $O(n)$ node operations compared $O(n!)$ for the brute force breadth-first search.
Results

To demonstrate the system performance with occluded objects, Fig. 12 on the next page shows the images and the associated error profiles.

- To quantify the improvement for occluded objects, observe the top middle image in Fig. 12 on the next page.
  - The minimum Euclidean distance measure for the projection of the occluded F22 image using the full eigenspace was 3.0845, a relatively large value for correct identification.
  - Observe that when we apply the sub-eigenspace scheme, the minimum distance shown in the error profile decreases to 0.0890 using the proposed graph search method thus providing excellent recognition!

- It appears from Fig. 12 that the iterative sub-eigenspace projection method provides an effective means to identify the occluded object as an F22 aircraft.
Figure 12: A set of occluded-object test images of the F22 aircraft and the resulting error profiles.
In all six occluded F22 images in Fig. 12, notice the steep reduction in error at the start of each error profile, corresponding to eigenspaces $R^{24}$ through $R^{14}$ followed by gradually decreasing error. One might use the error profile rate of change to distinguish occluded versus non-occluded F22 aircraft.

- The error profiles in Fig. 12 suggest we use the minimum error value at the end of the iteration.
  - Initial error values using the full eigenspace for the six figures were 40.1475, 3.0845, 2.7136, 2.5316, 3.5441, and 2.0688.
  - Using the sub-eigenspace search, the final error values are shown as 36.5495, 0.0890, 0.0788, 0.0115, 0.0961, and 0.0267, respectively.
    - Top left image showed only a modest decrease in error.
    - Only a small portion of the F22 nose tip was visible, suggesting the degree of occlusion for that image surpasses any reasonable means to recognize the object as an F22 aircraft.
  - All other final error values indicate that a threshold value of 0.1 can be used to determine whether an occluded object corresponds to the F22 aircraft.
False Positives?

To test for false positives, we applied the same technique to four images shown at the top of Fig. 13 on the next page, yielding the error profiles shown at the bottom of Fig. 13.

- The initial error values using the full eigenspace for the four non-F22 images was 2.2013, 40.1475, 4.4705, and 5.0762.

- Using the sub-eigenspace search, the final error values are shown as 0.1233, 36.5495, 0.86, and 0.5078 for F14, P51, Tornado, and M6, respectively.

- If the recognition threshold was 0.1 as suggested earlier, no false positives would have fooled the system.
Figure 13: A set of non-F22 aircraft test images (F14, P51, Tornado, and M6) and the resulting error profiles. This tested for false positives.
Conclusions

The benefits of a subspace projection-based object recognition system over a CAD-model based system were described and demonstrated.

- The eigenspace provides a global and robust object model.

- The eigenspace works well even when high image compression is used.
  - The proposed method provides reduced computational complexity and decreased memory storage requirements; the aesthetic beauty of a simple object model representation is an added attraction!
  - A variation, based on the $A^*$ graph search method, was shown to mitigate many problems associated with occluded objects.
  - The feasibility of this recognition method was demonstrated using both non-occluded and occluded aircraft images.
  - Preliminary results demonstrate the applicability of the proposed technique to a practical target recognition system.
  - Further analysis of the proposed technique is planned, to include variations on $x$, $y$, $z$, $\theta$, and $\varphi$. 
Questions?