International Journal of Computer Vision 36(2), 131–147 (2000)

Object Detection and Localization by Dynamic Template Warping

APARNA LAKSHMI RATAN, W. ERIC L. GRIMSON AND WILLIAM M. WELLS III
Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 545 Technology Square, Cambridge, MA 02139, USA
aparna@ai.mit.edu
welg@ai.mit.edu
sw@ai.mit.edu

Received October 16, 1998; Revised July 8, 1999

Abstract. A simple method is presented for detecting, localizing and recognizing instances of classes of objects, while accommodating a wide variation in an object's pose. The method utilizes a small two-dimensional template that is warped into an image, and converts localization to a one-dimensional sub-problem, with the search for a match between image and template executed by dynamic programming. For roughly cylindrical objects (like heads), the method recovers three of the six degrees of freedom of motion (2 translation, 1 rotation), and accommodates two more degrees of freedom in the search process (1 rotation, 1 translation). Experiments demonstrate that the method provides an efficient search strategy that outperforms normalized correlation. This is demonstrated in the example domain of face detection and localization, and can extended to more general detection tasks. An additional technique recovers rough object pose from the match results, and is used in a two stage recognition experiment in conjunction with maximization of mutual information.

Keywords: object recognition, object detection, localization, dynamic programming, mutual information

1. Introduction

Many recent approaches to object recognition are really verification and localization methods, because they address the question: "Is there an instance of a specific object in this image, and if so, where is it?" Thus, such methods verify or refute the hypothesis that a particular object exists in the image,[1] but often don't address:

- Detecting and localizing instances from classes of objects (e.g., faces or cars), rather than specific objects.
- Constructing object models that support object detection from 2d images while allowing for three dimensional pose changes. One could use an explicit 3D model, or a large set of 2D images (real or virtual). Ideally, one would like to use a very small set, e.g., one image.
- Efficiently seeding the search for an instance of an object in an image. Many recognition methods find the alignment of model with data by minimizing some error criterion. Such methods often have narrow capture radii. An alternative is to provide multiple seed hypotheses, e.g., by considering matches of minimal pairings of data and model features, but this runs the danger of being overwhelmed by the search combinatorics. We need methods that will converge to optimal solutions from highly inaccurate initial estimates.

To address these issues, we present a method that:

- detects instances of a class of objects (heads), while allowing for object class variation, and pose changes;
- extracts approximate pose data that can seed more detailed verification, and

- if a 3D model is available, the method verifies the existence and pose of an object by the Mutual Information (MI) (Viola and Wells, 1995) approach to recognition, which supports a very general matching of model and image data, without direct feature correspondence.

The technique we develop has several critical properties that make it robust in extracting the 2D image location and 3D pose estimate of an object whose pose in the image is unknown. It differs from the view-based correlation methods in that the model is a single, simple, flexible template extracted from a nominal view of the object and it is used in a matching method that can handle a wide range of pose variations. This low resolution template model together with the flexible matching technique allows the method to accommodate within-class variations caused by scaling and shifts of individual regions.

Key properties of our approach include:

- The model for an object class is a single, simple template that may be stretched or warped in one dimension.
- This warped template can be matched to instances of an object under widely different poses, accommodating scaling, translation and out of plane rotations about the vertical axis.
- The method converts the search for object instances and poses into one (or a few) one dimensional matching problem(s).
- Dynamic programming is used to establish a mapping from the deformable template representing the object class into the image.
- While the method can recover good estimates of three degrees of freedom of object motion (x and z translation, y rotation), it can also accommodate up to two more DOFs of motion in the search process (x rotation, y translation). Extensions allow one to also search for the final rotation. Depending on the implementation, some information about y translation may be available from knowledge of the actual match of template to image.

Our approach combines the general detection flavor of patch correlation with the ability to accommodate more degrees of freedom of object motion that are traditionally associated with feature-based recognition systems. We demonstrate our method on face detection, as faces naturally suit the approach (heads are usually oriented upwards, but appear facing in varying directions). We show performance comparable to some current face detectors and show that the detection results can be used for model-based pose verification if needed.

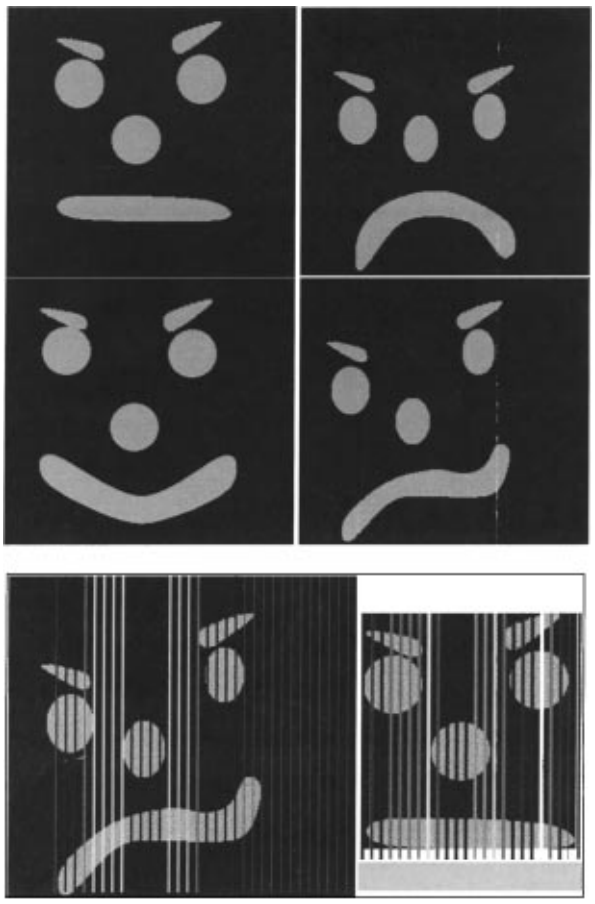

Figure 1. This class of cartoon faces shows the 2D deformations handled by the system. The last row shows the resulting of matching the template (nominal view) to a novel image.

The flexible shape model we propose deals with 2D deformations that are more severe than traditional transformations. Figure 1 shows an example of 2D transformations that the method can handle. The first image in the set is the prototype (nominal) image and the rest of the images illustrate warps due to y-translation, vertical shifts, scaling and shearing variations that are accommodated as long as the overall ordering of the features are maintained. This method can be used to accommodate class variations due to variation of features across a population or variations of an individual over time (e.g. facial expressions).

1.1. Background

A primary difficulty in object detection is accommodating the wide range of variations in object appearance due to pose, lighting and class variations, while maintaining good detection performance. Existing approaches for dealing with these variations are described below.

In view-based methods, one computes the similarity between a template and image locations and returns all matches above a threshold. Since most classes of objects cannot be modeled by a single template, these techniques typically use a set of templates to accommodate variations in pose, lighting and shape, by synthesizing views from a set of examples (Beymer, 1993), or by using networks to learn an object class from several examples with varying pose and lighting (Sung and Poggio, 1994; Rowley et al., 1995). Betke (Betke and Makris, 1995) uses simulated annealing for fast 2D object recognition (traffic signs) in noisy images by matching a new scene to a set of templates generated by transforming model images. Most of the existing methods for face detection perform well on upright, frontal faces and do not handle 3D pose variations. Recently, some systems like Rowley et al. (1998), Schneiderman and Kanade (1998) detect faces with arbitrary rotations in the image plane. However, these systems are not as robust with off-plane rotations of faces (profiles). In this paper, we propose a method that can handle off-plane rotations (rotations about the vertical axis for faces) efficiently and reliably.

In deformable template matching (Yuille et al., 1992; Cootes et al., 1993), templates are constructed to model the non-rigid features of the object, and at recognition time the templates are aligned to the image by minimizing an energy function for the individual features. These methods work well when the deformations are small—they provide a detailed analysis of the image in terms of the template, but do not address non-local search issues.

Eigenspace methods (Murase and Nayar, 1995; Pentland et al., 1994; Huttenlocher et al., 1996) represent the varying appearances of objects by using principal components analysis (PCA) on a set of sample views to identify a low-dimensional subspace of the view-space. Images are accepted or rejected based on their distance from the pre-computed subspace. These methods need a large set of views of the object to be sampled under varying pose and lighting conditions.

The ratio-template (Sinha, 1994) detects faces under varying illumination conditions, by capturing changes in luminance between fixed image patches. While this method performs well for frontal faces under varying lighting conditions, it is not as robust under 3D pose variations.

Hornegger (1995) converts feature based object recognition under orthography into several 1D search subproblems. The technique constructs 1D images from the more conventional 2D images by a process of *marginalization*, which projects image features onto the x axis (and discards their y coordinates). This effectively discharges two degrees of freedom (DOF) of object motion, namely x rotation and y translation.[2] The remaining three DOFs are then searched exhaustively. The resulting candidate partial solutions are then completed with a full 5 DOF localization.

In this paper, we advocate the use of dynamic programming to address the 3 DOF search that remains after conversion to 1D matching subproblems. Dynamic programming (DP) has been used successfully in speech recognition to produce an optimal time alignment of an actual speech stream to be recognized to an acoustic model, via the Viterbi algorithm (Viterbi 1967; Forney, 1973; Sakoe and Chiba, 1978), and we were motivated to apply it to search problems in object recognition. DP is well established as a methodology for solving the stereo matching problem, e.g., Baker and Binford (1981) and Ohta and Kanade (1985). It has also been used for contour adjustment and other boundary oriented scoring or identification tasks e.g., Ballard and Brown (1982), Barrow (1976) and Shashua and Ullman (1988).

DP matching techniques typically will accommodate fairly general 1D warpings in the matches that are found, subject to order constraints. This might seem to be a drawback because in object recognition, it would allow distortions that can not be generated by a rigid object moving in front of a camera. However, within the warps that DP accepts are embedded some useful mappings, for example x translation and the overall scale changes that can occur as an object moves away from a camera. In addition, y rotations of a vertically oriented cylindrical object will induce (primarily) stretches and shrinks along x, which may be accommodated in a DP match. Figure 2 illustrates the 1D-warps induced by rotations and scale changes. We have found in our experiments that this feature can be used to accurately recover the y rotation when the object in question is approximately cylindrical.

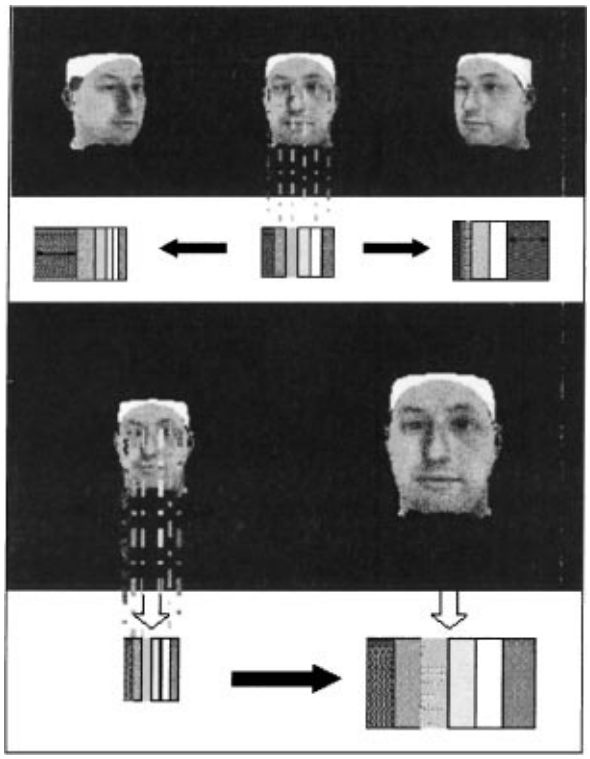

Figure 2. Rotations and scale changes induce 1-D warps.

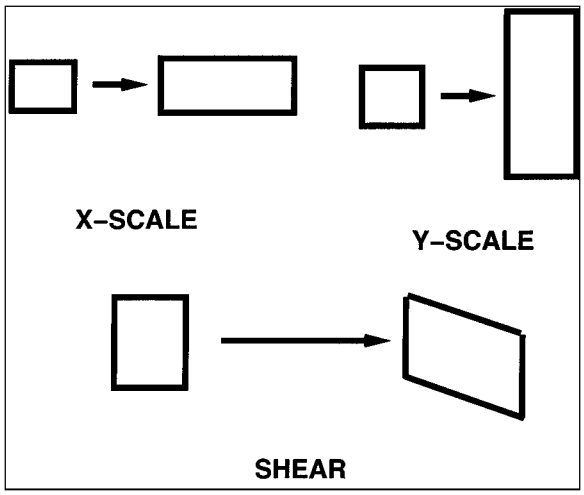

Figure 3. 2D affine warps handled by the system (x,y translation, x,y scale and shear).

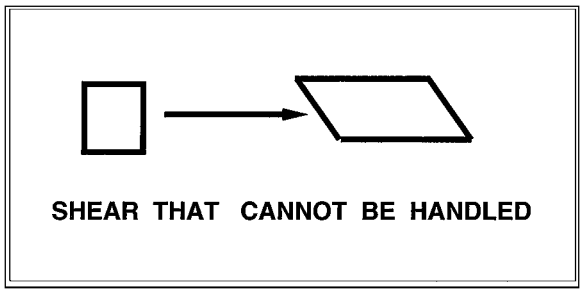

Figure 4. 2D shear that cannot be handled by the system.

2. Comparison with Affine Models

There has been considerable work in the computer vision literature on recognition systems that accommodate affine transformations on objects (e.g. Huttenlocher and Ullman (1995) and Rucklidge (1994) among others). Part of the motivation is that for planar objects, affine transforms are equivalent to rigid body motion under orthography. The flexible shape model that we propose handles five of the six DOF of affine transformations. Figure 3 shows the allowable set of affine deformations and Fig. 4 shows a shear that is not handled by the system. Our technique finds a match between members of a class by relating new images in the class to the prototype (template) through the set of allowable deformations based on stretches and shifts in the template features, subject to order constraints. The ability of our system to handle these affine warps is similar to the constrained-affine shape model proposed by Syeda-Mahmood in (Mahmood and Zhu, 1998). The constrained-affine shape model is a region-topology based model that captures the spatial layout similarity between members of a class by a set of affine deformations from a prototypical member. The constrained-affine shape model accommodates individual 2D-deformations of every region in the model as long as constraints on the residual translation, scaling and the overall ordering of the regions is satisfied. The technique we propose in this paper is related in that it also accommodates 2D-warps by deforming model regions (columns) but our technique extends this framework to handle out-of-plane rotations (y-rotations) as well.

3. Dynamic Template Warping

The information flow used in our system including 3D pose recovery is outlined in Fig. 5. In brief, the class model consists of many 3D surface patches derived form a 3D model if it is available. The 3D model is not essential for the detection but if present we can use it to verify and refine the pose of the object in

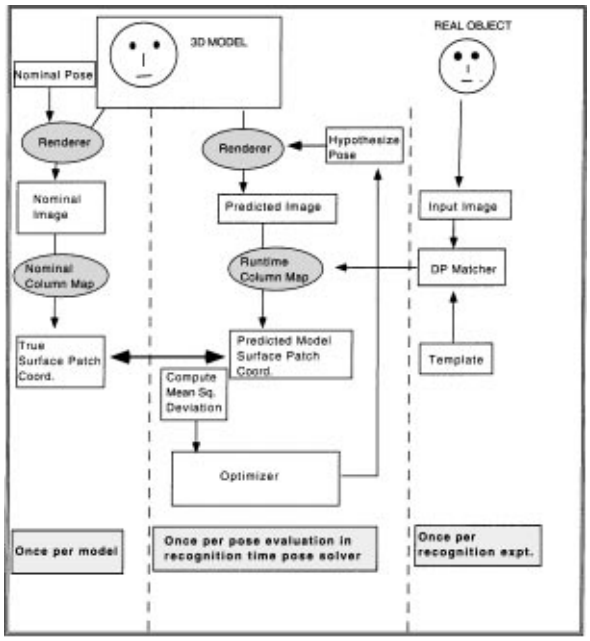

Figure 5. Information flow in the DTW pose solver. The template is derived from the nominal image.

the image. A template (whose details are provided below) is constructed from a characteristic (typically frontal) view of a representative object in the class. The actual mapping from surface patches to elements of the template is determined once for the model. At recognition time, dynamic programming (DP) is used to solve one-dimensional matching sub-problems and thus determine mappings from elements of the image to corresponding elements of the template. A partial solution for the unknown object pose is obtained by solving an optimization in which the projection of the model surface patches into template columns, via the hypothesized pose and the runtime column mapping determined by DP, are made to best agree with their true values.

Below, we provide more details about the detection and localization process.

3.1. Representing Objects

Our goal is to represent classes of objects in a flexible manner, capturing 3D pose changes and class variations in a simple system. Initially, we assume that the images we will be processing are taken in an upright position, i.e., the gravity vector runs vertically through the image. As a consequence, we choose to represent a class of objects by a sequence of image columns, each column consisting of a set of intensity patterns. To construct our model template, we begin with an image of the object rendered in the nominal pose (Fig. 6)—e.g., for the case of faces, we use an image of a face model in a roughly frontal view. This image is smoothed, downsampled and intensity-normalized, then separated into columns. Each column is described as a sequence of intensity classes obtained by coarsely quantizing the

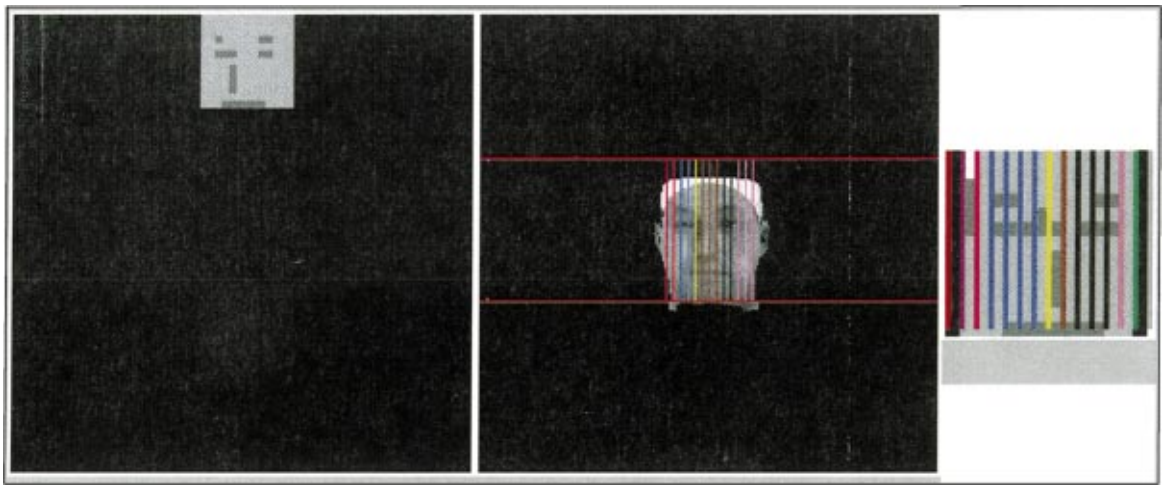

Figure 6. Results of the DP matcher on the nominal pose image. The left image shows the low resolution model columns copied into the appropriate matching image columns. The images in the middle and right show the column mappings. The color code on the model template and the image lines show the mapping. The red box on the image indicates the horizontal strip in which the best match was found. The middle image shows the nominal view (frontal face) from which the template is extracted.

intensities present in the smoothed, downsampled and intensity-normalized image. The smoothing, downsampling and rough quantization blurs out the effects of small intensity variations in the class, and as we shall see shortly, the use of separated columns will allow us to accommodate 3D pose variation.

3.2. Finding Template Matches in the Image

To find matches of such a template in an image, we need to consider the variations that a template may undergo as a function of class variation and pose variation. Class variations will be discussed shortly, when we describe the measurement of similarity. We break pose variation into several pieces, initially focusing on three of the six degrees of freedom. Translation of the template in the two image dimensions can be straightforwardly handled through a search process. Rotation about the vertical axis will have the effect of inducing a warping on the columns of the template. In particular, some of the template's columns will be stretched out to cover several image columns, and other columns will be foreshortened and thus compressed into fewer columns. This implies that our matching of template columns to image columns is no longer a one-to-one map, and cannot be dealt with by simple techniques such as correlation. On the other hand, such a warping of the template naturally lends itself to dynamic programming methods.

Dynamic Programming (DP) is a search technique that has been applied to many problems that are formulated as the minimization of a cost function over paths in a graph. It may also be applied to find optimal matchings among features if certain monotonicity constraints are met. The dynamic programming algorithm we have used is derived from the longest common subsequence algorithm described in Corman, Leiserson and Rivest (1990). In the present application, sequences of image and template features are compared by a measure of their degree of match rather than the binary-valued measure used in the standard subsequence algorithm.

For a given model template, a set of overlapping horizontal strips is extracted that cover the input image (Fig. 7). The strips are 1.5 times higher than the template height, and overlap by three fourths of their height—they have been designed so that some strip will entirely cover an instance of the object with nominal height. Within each strip, the template column features are matched with the image column features by

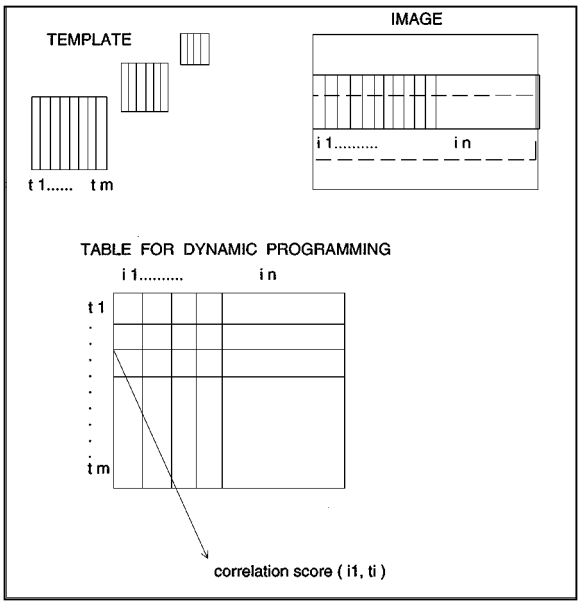

Figure 7. DP matcher data structures.

a correlation-like process that locates the best matched section of the image column feature. This embedded problem of measuring the agreement of template and image column features is somewhat similar to the surrounding problem of matching image and template, and it is similarly useful to address it with a mechanism that is able to tolerate vertical shifts, and moderate scaling and stretching.

3.3. Column Matcher

The horizontal strips of the template and the image are sequences of pixel column vectors $T(i)$ and $I(j)$. Pixels in the template and the image column are indexed as follows: $T(i, m), 0 < m < M$ where M is the template column height and $I(j, n), 0 < n < N$ where N is the image column height. The largest set of corresponding pixels that preserve relative ordering can be formulated as the problem of finding the longest common subsequence (LCS) of the ordered pixels in the template column $T(i)$ that is also found in the image column $I(j)$. We adapted the dynamic programming algorithm for computing an LCS described in Corman, Leiserson and Rivest (1990) to give us the correspondence between the template and image columns.

3.3.1. Local Cost. The local cost $d(m, n)$ of matching a template column pixel $T(i, m)$ with an image column

pixel $I(j, n)$ is given by the squared difference between $T(i, m)$ and $I(j, n)$.

3.3.2. Global Column Matching Cost. The accumulated cost function $D(i, j)$ is the sum of the local cost $d(n, m)$ of matching $T(i, m)$ with $I(i, n)$ over an optimal path for $0 < m < M$ and $0 < n < N$. The intermediate results are recorded in a dynamic programming table $Score(n, m)$, $0 < m < M$ and $0 < n < N$ which records the current length of the sub-sequence obtained by matching $T(i, m)$ with $I(j, n)$.

3.3.3. Details. Our search algorithm returns a set of matched sequences $S = (s_1, s_2, \ldots)$ where each s_k is a sequence which contains the following information. s_k.length gives the length of that matched sequence, s_k.score gives the score of that matched sequence and s_k.matches has the list of matched feature indices (n, m).

$d_{model}(m)$ $0 < m < M$, tracks the score of each model pixel in the image allowing for one-to-many mapping of model pixels in the image. If many consecutive pixels map to the same model pixel $d_{model}(m)$, the final match score for that pixel is the average over all the pixel match scores to that model pixel.

The first step is to build a table $d(n, m)$ for $m = (0 - M)$ and $n = (0 - N)$. $d(n, m)$ given the squared difference between $I(j, n)$ and $T(i, m)$. The search for score of the longest common subsequence proceeds as follows:

3.3.4. Initialization.

$$Score(0, m) = d(0, m)\ 0 \leq m < M$$

$$C(0) = \min_{0 \leq m \leq M}(Score(0, m))$$

$$In(0) = \arg\min_{0 \leq m \leq M}(Score(0, m))$$

3.3.5. Recursive Step.

$$Score(n, m) = d(n, m) + C(n - 1, m)$$

$$C(n, m) = \min_{0 \leq m \leq M}(Score(n, m))\ 1 \leq n < N$$

$$In(n) = \arg\min_{0 \leq m \leq M}(Score(n, m))\ 1 \leq n < N$$

if n matches two consecutive model features $(m1, m2)$ equally well, $In(n)$ gets set to the model feature with the larger index.

If there is an acceptable match (i.e. $C(n) < T$), the current matched sequence s_k is updated based on the ordering constraint as follows.

- If $In(n) < In(n - 1)$, start new match sequence s_{k+1} since the ordering constraint is violated.
- If $In(n) > In(n)$, update current match sequence s_k since the ordering constraint is preserved. The sequence is updated as given in Section 3.4.
- If $In(n) = In(n)$, update current match sequence s_k since the ordering constraint is preserved if $MHits(In(n) < \delta$. The sequence is updated as given in Section 3.4.

3.4. Updating the Match Sequence

We would like to update the model match score accounting for multiple consecutive matches in the image to the same model pixel i.e. accounting for some limited deformation of the model column in the image.

- If the best matched model pixel is the same as the previous best matched model pixel i.e. $In(n) == In(n - 1)$, update the score for that model pixel $d_{model}(m)$ by adding the new match score and incrementing the number of hits to that model pixel $Mhits(m)$. Update s_k.length and s_k.match.
- If the best matched model pixel is not the same as the previous matched model pixel, update the score for the previous matched model pixel $d_{model}(m - 1)$ by averaging the total score for the previous model pixel by the number of hits. Update s_k.length, s_k.match and $s_k.score$ with the computed average score for the model pixel $d_{model}(m - 1)$. This avoids giving a larger score for a scaled/stretched instance of the model in the image. Update the score for the current matched model pixel as before.
- If there is no match, and the number of consecutive non-matched pixels in the current sequence s_k is within some limit, the current sequence score is updated, but the length of the current matched sequence remains the same. This mechanism provides a way of handling some noise in the image since the occluding patches will be consecutive image column pixels with no matching model column pixels.
- Result of column matcher: Given the matched subsequences $S = (s_0, s_1, \ldots, s_K)$, we can find the score $((D(i, j))$ of the longest matched subsequence for model column $M(i)$ and image column $I(j)$.

To summarize, the column matcher provides a way of finding a many to one matching between elements

of the image and template column features. The process operates by searching for equivalent structure in terms of uniform intensity class subsequences while accommodating some variation in subsequence length (up to a factor of two). In more detail, each template and image column is a vector of pixels given by $(T(i,0),\ldots,T(i,M))$ and $(I(j,0),\ldots,I(j,N))$. Starting from all pairs of pixels in the template and image column vector, match the content of pairs of pixels based on their intensity. The match score between pairs of pixels $(T(i,m), I(j,n))$ are used to fill a cost table. Dynamic programming can then be used to find the optimal subsequence that satisfies the following constraints: (1) the overall spatial ordering of elements in the template column is preserved in the image column and (2) the extent of stretch of a template column element $(T(i,m))$ in the image column is bounded by a fixed threshold δ.[3] This column matching algorithm gives the longest common set of corresponding elements between the template and image columns which match in their content, have the same overall spatial ordering while allowing for some bounded vertical shifts and scale variations.

3.5. *Column Mapper*

To match across the sequence of column matches, we use the standard subsequence algorithm for dynamic programming, which assumes that matches are order-preserving and have scores that are independent of one another. The algorithm constructs two tables, the cost table, and the index table, that we index by every pair of template $(T(0),\ldots,T(m))$ and image $(I(0),\ldots,I(n))$ column-features (the cost table is illustrated in (Fig. 7)). The dynamic programming approach to finding the optimal subsequence proceeds from left to right. By consulting the cost table, and by examining previously computed solutions to problems ending in the previous position, the algorithm efficiently computes and stores in the tables the scores and sub-sequence predecessor information for the optimal solutions to the problem at the current position.

Our algorithm extends the standard one to return all subsequences having scores above a threshold (rather than a single best subsequence), by preserving information defining the previously most promising subsequence when a new subsequence is initiated.

The algorithm needs more space to store this information, but the time complexity remains the same $(O(MN))$ for M model column features and N image column features and k strips. In addition, we employ a look-ahead technique similar to those used in DP stereo-matching systems to accommodate partial occlusions, by requiring several columns of match failure before the initiation of a new subsequence. When combined with the column matcher described above, and run at a single scale, the DP template matching algorithm has equivalent complexity to simple template correlation while accommodating larger rotations about the y-axis.

The DP matching algorithm described above is used to find ordered image strip columns that are deemed to be explained by the template by having scores greater than a given threshold. The matches found by the DP algorithm are ranked according to score.

We can repeat this process for each horizontal strip of the image. Note that using multiple strips covers the y translation, and finding the optimal column match using DP directly covers the x translation and rotation about the y axis. Note that the flexibility of the matching—both the roughly quantized intensity sequence matching within each column, and the flexible matching across columns—allows for a range of class variations as well.

When the DP matcher has finished, the following information is available for use by the partial pose solver (which will be described below):

- The "active patch" in which the match was found if the model was present in the image i.e. the horizontal position together with the vertical position given by the horizontal strip.
- A detailed column mapping between image columns and template columns where many image columns can map to the same model column (to account for horizontal stretching caused by rotations about the y-axis and scaling). Image columns may also map to the "no" model column (this accommodates partial occlusions).

3.6. *Scale Hierarchy*

Our initial strategy uses a fixed template height of 64 pixels. Because the column matcher is able to accommodate some variation in scale, and the DP matching algorithm can accommodate scale variation as a "stretching" of the match, this mechanism can accommodate object instances with apparent heights ranging from roughly 40 to 80 image pixels, and the scale may be later recovered from information recorded in the detailed column mapping.

A hierarchical implementation was used for the rest of the experiments. A set of three templates were constructed with heights of 16, 32, and 64 pixels by pixel replication, and three sets of image strips were prepared, as described above. The DP matching algorithm was then run at the three scales, in order to accommodate objects instances with apparent heights ranging from roughly 10 to 80 image pixels. Note that this implicitly allows us to solve for the third translational component of the pose.

3.7. Partial Pose from DP Match Results

In this section we describe a method for recovering a rough, or partial, object pose from the detailed column mapping that is obtained as a result of the DP match from image to template columns. The center column of Fig. 5 shows the flow of information that is used in the pose determination process given a 3D model which has been rendered in a nominal pose to form the template (left column of Fig. 5) and the results of the DP matcher (right column of Fig. 5).

Our pose determination subsystem may be used in those situations where a concrete 3D model of the object is available. In our experiments we have used 3D data from a cyberware scan of a person's head. This data consists of a collection of small surface patches that are each described by their 3D position, surface orientation, and albedo.

We utilize a rendering algorithm that is able to simulate perspective projection and generate realistic images of the object, as well as a mapping from image pixels back to the 3D patch (if any) that is imaged at the pixel. The column mappings generated by the matcher map from image columns onto template columns, and by using these mappings, we can obtain, for each image pixel imaging a surface patch, an association between the patch and its "image" in template space. The pose solver makes use of the runtime column map and operates by minimizing the disagreement among the true values of template space coordinates of the surface patches, and those values obtained via renderings at hypothesized poses. This mechanism is described in more detail below.

We seek a pose such that a rendering of the model by that pose looks like the input image. We assume that the DP matcher has determined the correct column map (the "runtime column map") that maps image columns to their corresponding template columns. Thus we want to find a pose such that when the model is rendered by that pose, the resulting synthetic image will (like the input image) be correctly related to the template columns via the runtime column map. This suggests the following approach for evaluating a hypothesized model pose:

- Project the model surface patches into the image by rendering them according to the hypothesized pose, and send the resulting image coordinates through the runtime column mapping. This process will produce a mapping from model surface patches to their corresponding template columns.
- Evaluate the consistency of the hypothesized pose by comparing these predicted surface patch coordinates (template indices) with their true values. The true values of the template coordinates of the surface patches was obtained as follows. The template itself was derived from a rendering of the 3D model at the nominal pose, by a quantization process. A nominal column map was obtained by running the matcher (using the template) on the nominal image. As described above, an association between the model surface patches and template coordinates was obtained from the output of the rendering algorithm and the nominal column map, and this association was taken to define the "true" values of the surface patch coordinates in template space.

An objective function may be constructed to measure this consistency—we have used the mean squared difference between the predicted and true template coordinates of the surface patches.

As it happens, there is a simple method available for establishing an approximation to the true mapping from object surface patches to template columns—we simply render the model at a nominal (in this case, frontal) pose, and use the DP matcher to establish a nominal column map from the nominal image to the template. We then establish the mapping from surface patches to template columns in the manner described above.

The mechanism outlined above provides detailed information relating to x and z translations, and also the y rotations of the model.

Useful, but less precise, information related to the x and y translations is available from our knowledge of the "active patch" in the image (the section of the image strip this is involved in the match that just includes the columns at the beginning and of the DP match to the template).

In our experiments, the partial pose solution is determined in the following way.

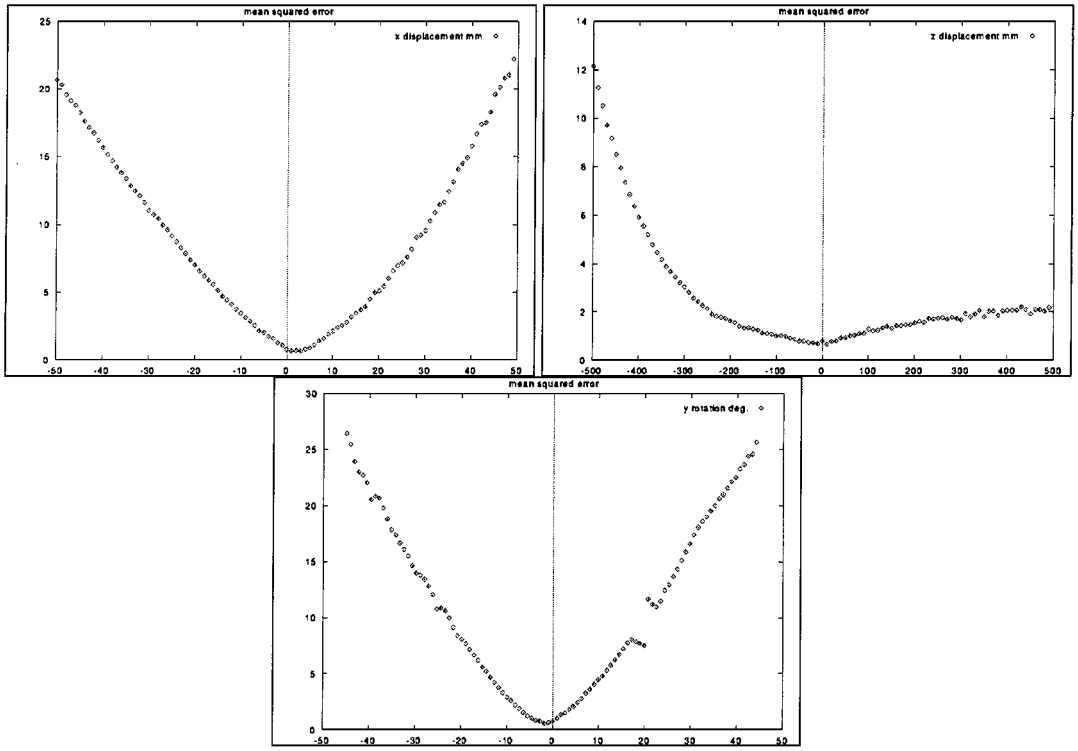

Figure 8. Probed values of the objective function used in the recognition-time pose estimation (the mean squared difference between predicted and true template columns for projected model surface patches). The plots display the objective function against the difference between the three pose adjustment parameters and their known true values.

- Starting with the nominal pose, an initial pose estimate is obtained by applying a motion to the object that adjusts the apparent azimuth and elevation of the object in order to center it in the active patch. While this motion is applied to the object, in terms of the relation of the object and the camera, it is equivalent to a rotation of the camera about x and y.
- The initial pose estimate is used as a starting value for a second stage of pose estimation. An additional motion is applied to the object that consists of a rotation of the object about y (measured in degrees), and translations along x and z (in millimeters). These parameters are optimized with respect to the objective function described above, using the first stage estimate as a starting value.

The effect of the first stage described above is to move the object from the nominal pose in order to place it in approximately the right part of the image, while the second stage provides relatively precise adjustment of some of the pose parameters from the match.

After this process, x and z translation and y rotation have been determined relatively precisely from the match, and y translation has been approximately determined from knowledge of the active patch. The remaining parameters of motion, x and z rotation are not recovered by the method, and their values will be whatever is obtained from the application of the two stages of adjustment to the nominal pose (the adjustments have minor effects in this regard).

Some plots of the pose objective function are shown in Fig. 8, for a match done against a synthetic image (top left image in Fig. 9) where the true pose is known. The plots suggest a prominent minima near the true value for the adjustment parameters, although some noise and local minima are apparent.

4. Image Matching Results using DP

In this section we show some simple experiments to demonstrate that dynamic programming with a simple, stretchable template can be used as an efficient

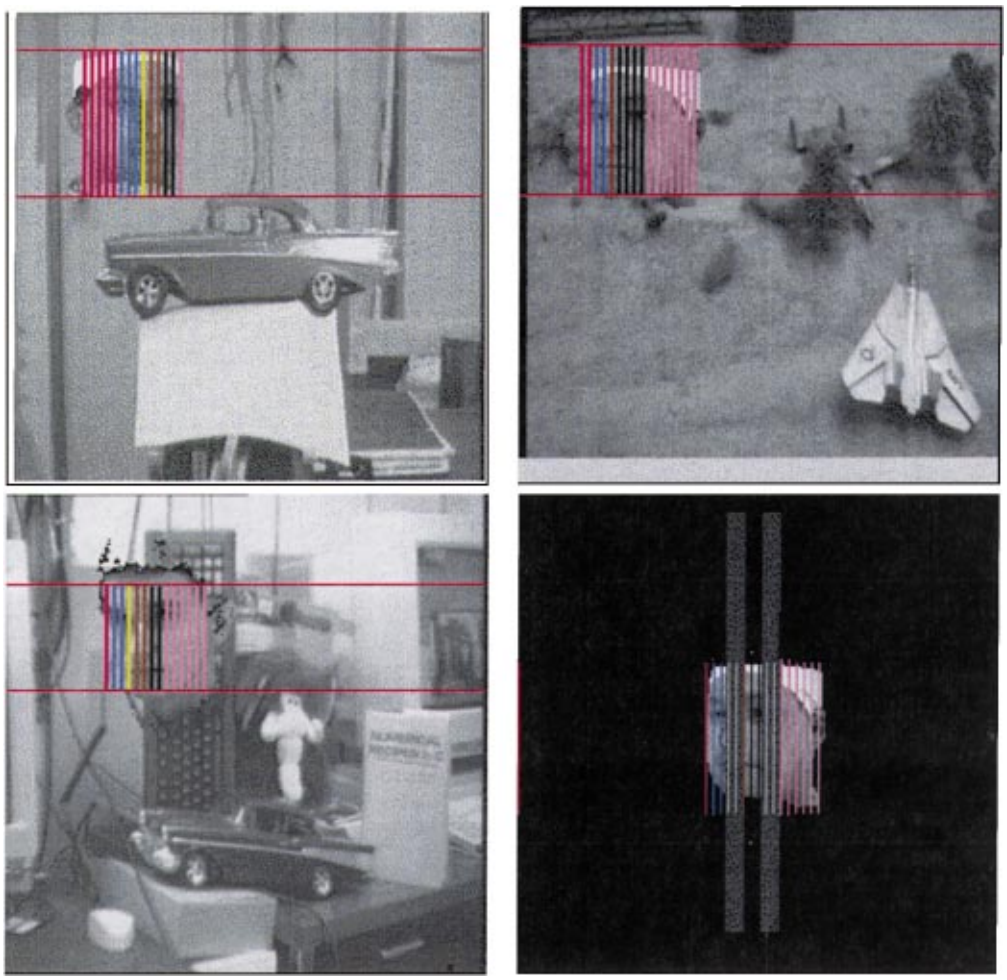

Figure 9. Results of DP matcher on images with varying backgrounds. The colored vertical lines indicates the mapping from template columns to input image columns. The box on the image indicates the strip in which the best match was found.

search strategy that can perform better than normalized correlation for detection and localization tasks under translations, y-rotations and partial vertical occlusions. The method also provides a detailed column mapping from matched image columns onto model columns which can then be used for 3D pose estimation. We will describe experiments that address these issues below.

We used five Cyberware (Cyberware Incorporated) scans of heads rendered at various poses using a graphics package. The Cyberware scanner is a structured light scanning device that simultaneously obtains 3D surface and surface texture information. The rendered heads were then embedded in a variety of backgrounds as shown in Fig. 9. The cyberware models were rendered with varying horizontal and vertical translations, a large range of y-rotations (-50 to 50 degrees from the nominal pose) and a small range of x-rotations (-5 to 5 degrees from the nominal pose). There were no z-rotations of the model. In addition to the cyberware scans, we tested the system on real images of heads and compared detection performance with normalized correlation (Figs. 10 and 11).

For roughly cylindrical and textured objects (like heads), this technique can handle rotations about the Y-axis since they are approximated by stretches and shrinks of the model columns in the image. The model can be thought of as a flexible template that consists of a set of ordered columns that can be stretched and shifted to fit the image.

Figure 6 describes the process of getting the nominal column mapping used to approximate the true column

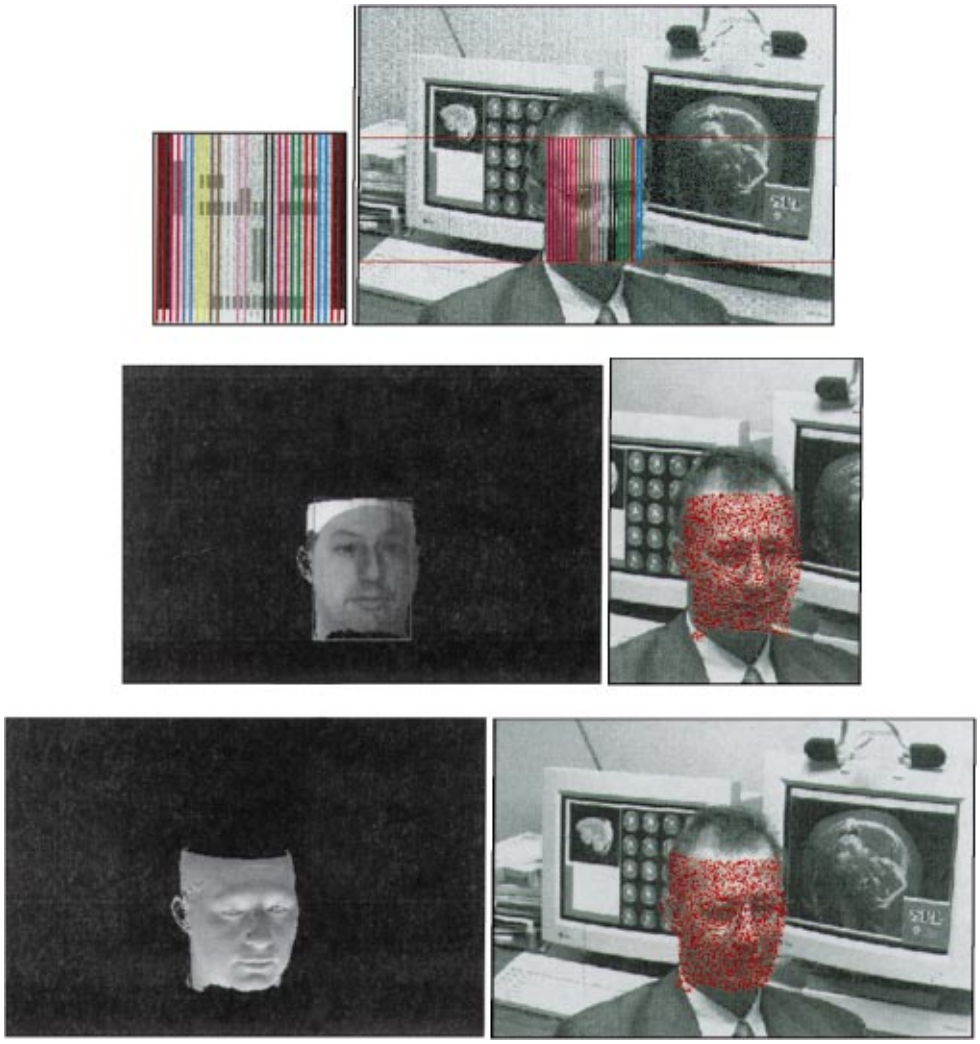

Figure 10. End to end system running on a real image. The template and results of the DP column match are shown in the top row. Results of the partial 3D pose solution using the DP-method, which is used as the initial pose given to the MI recognition system are shown in the next row. This pose is illustrated by a rendering and by an overlay of points subsampled from the object. The final pose is illustrated by a rendering of the 3D model, and by an overlay of points subsampled from the object surface in the next row. From the figure, we see that the final solution given by the MI alignment has corrected the x-rotation error, and the pose in the final rendered image agrees with the input pose.

mapping for the model surface patches. The template is a set of columns extracted from the quantized frontal face (model face at the nominal pose) at low resolution. The color code on the model face (right) and the input image lines (left) show the mapping. The red box on the image indicates the horizontal strip in which the best match was found. A single template at 3 different scales is used for all the experiments. The results (Fig. 9) illustrate how rotations about the Y-axis are handled as horizontal deformations of model columns. The best match finds the longest common subset of ordered model columns that explain an image region. Figure 9 also shows that simple vertical occlusions can be naturally accommodated by the matcher and that a single template can be used to locate heads in general. Figure 10 shows an example of scale changes handled by the system. This figure also illustrates that the system can localize the head and get accurate column mapping information in real images.

4.1. Comparison to Normalized Correlation

These experiments compare the performance of the DP search to normalized correlation in a domain with rotations about the Y-axis and clutter.

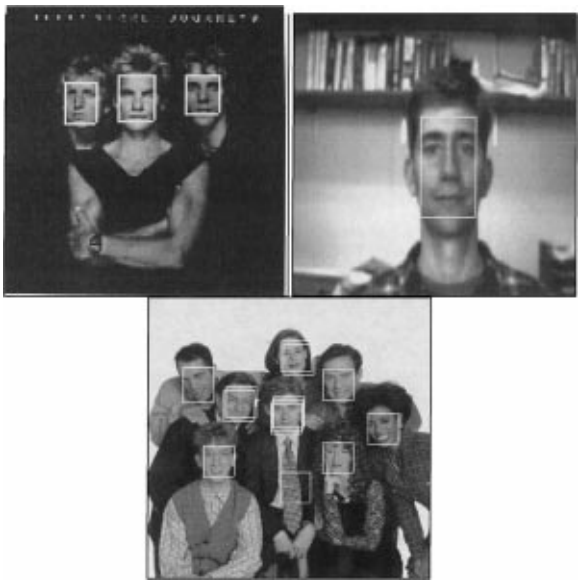

Figure 11. Some examples showing the detection performance of the system. The template used for detection was a frontal view of a rendered synthetic face at 3 different scales.

Normalized correlation has been used in a variety of detection tasks to find instances of an object in the image. It works well when the object is not rotated or scaled much in the image and the lighting varies uniformly. Variations of this basic technique have been used successfully in the face detection literature, for example in Brunelli and Poggio (1993), to locate instances of face-like patterns with low computational cost. More complicated systems like Sung and Poggio (1994) and Rowley et al. (1995) use a view-based correlation approach to detect faces by representing variations in pose using many examples. These systems operate in a view-based feature space of masked and normalized face patterns. They build a distribution based model of the class of all these canonical face patterns in the feature space and learn similarity measures for matching new patterns against the model.

In this experiment we compare the DP matcher with normalized correlation with a single frontal template extracted from the nominal pose shown in Fig. 6 (rendered face facing forward). For each threshold value, we ran normalized correlation on 20 images with varying backgrounds and varying rotations of the face (-50 to 50 degrees) about the Y-axis and rotations of (-5 to 5 degrees) about the X-axis. The template used was extracted from the image of the face facing forward with no rotation about the Y-axis as shown in the figure. The search window size was the size of the template. The search was centered at pixels (i, j) for ($i = 0$, $i < \mathit{ImageWidth}$, $i = i + 2$) and ($j = 0$, $j < \mathit{ImageHeight}$, $j = j + 5$) in all the images.

The detection performance of the two methods are compared using a "Receiver Operating Characteristic" (ROC) shown on the top in Fig. 12. The ROC is frequently used in the detection literature to evaluate detection performance by plotting the percentage of true positives detected as a function of the false alarms due to clutter. A higher curve indicates better performance. We see that for detecting faces with cluttered

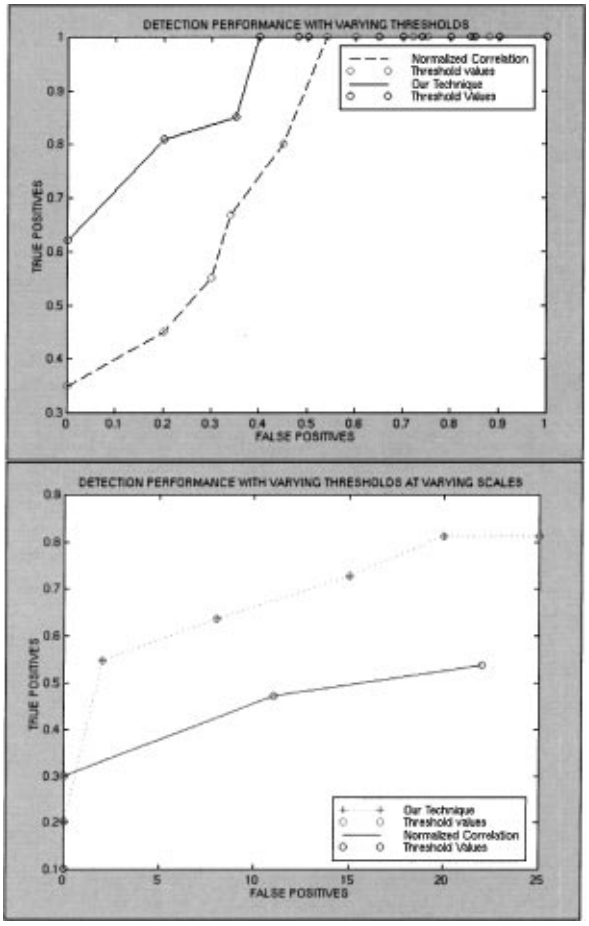

Figure 12. ROC Curves: Plot of False Positives vs True Positives for varying threshold values. Each point represents the (False Positive, True Positive) pair for a particular threshold value. The threshold values increase from 0 to 1 from the upper right corner to the lower left. TOP: The curves show the detection performance of the DP matcher (solid line) and Normalized correlation (dotted line) on the synthetic dataset with rotated images of the head in cluttered backgrounds. BOTTOM: The curves show the detection performance of the DP matcher (dotted line) and Normalized correlation (solid line) on a synthetic dataset with images where the scale varies upto a factor of two.

backgrounds and varying rotations about the Y-axis this method performs significantly better than normalized correlation with a single template. Techniques like Sung and Poggio (1994) and Rowley et al. (1995) use several templates extracted from several views of the object to accommodate y-rotations. The advantage with the DP matcher is that a wide range of y-rotations can be accommodated with a single stretchable template.

The lower plot in Fig. 12 compares the DP-matcher with normalized correlation with a single frontal template on a dataset of synthetic images of varying scales (variations upto a factor of two in scale). Here again we see that the DP matcher is able to accommodate larger variations in scale using a single template than traditional normalized correlation.

4.2. Comparison with Romano's Facefinder

We compared the detection performance of the DP matcher to a robust facefinder (Romano et al., 1996) technique in a domain where normalized correlation is known to perform well.

The input consisted of a dataset of 50 images which contained frontal views of faces. The template is extracted from one of the faces in the dataset by specifying the coordinates of the locations of the left and right eye, the left and right nose lobes and the left and right extremities of the mouth. This template was then matched against the images in the dataset using normalized correlation for different thresholds.

Our algorithm was run on the same dataset of 50 images. Our template was extracted from the same face image that was used to extract the template for Romano's method. The input images were split into overlapping strips as described earlier and the output consisted of image regions with column mapping scores that were greater than the given threshold.

The ROC-curve in Fig. 13 shows that the DP matcher performs slightly worse for this dataset for low thresholds but is comparable to normalized correlation for thresholds in the middle to high range.

4.3. Detection Experiments

We also ran the system on a set of face and non-face images, to determine its performance as a class detection system. Figure 11 shows examples of faces detected by the system. The database had a total of 188 faces. The true positive fraction of faces detected by the system was 0.91. The false positive fraction was 0.23.

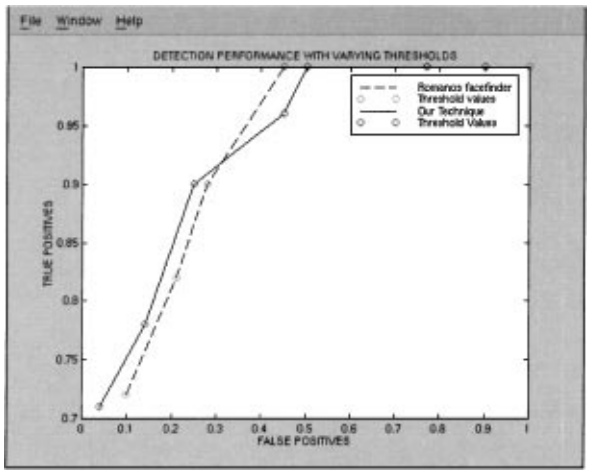

Figure 13. ROC Curves: Plot of False Positives vs. True Positives for varying threshold values. Each point represents the (False Positive, True Positive) pair for a particular threshold value. The threshold values increase from 0 to 1 from the upper right corner to the lower left. The curves compare the detection performance of the DP matcher (solid line) and Romano's facefinder (dotted line) on a dataset of frontal faces in cluttered backgrounds with varying lighting.

4.4. Pose Solving Experiment

Partial Pose Solution. Figures 14 and 10 shows examples of running the pose solver. For the experiment shown in Fig. 14, we have optimized the adjustment parameters by using the downhill simplex method (Press et al., 1990). The initial simplex was established with displacements of 1 and 5 millimeters in x and z, and 3.5 degrees in y rotation. These values allow the optimization to step past small local minima in the initial movements of the optimization. For this (synthetic) test image, the ground truth for the object pose is available, and the resulting errors in the estimated pose were 3.3 mm in x position, 12.8 mm in z position, and 1.2 degrees in y rotation. Note that the error in depth is slight, considering that the object is 1000 mm from the camera. The x and z positions, and y rotation, have been well determined from the information in the DP match, while the approximate y position has been determined from the vertical position of the "active patch" in the image.

Figure 10 shows the results of the pose solver running on a real image.

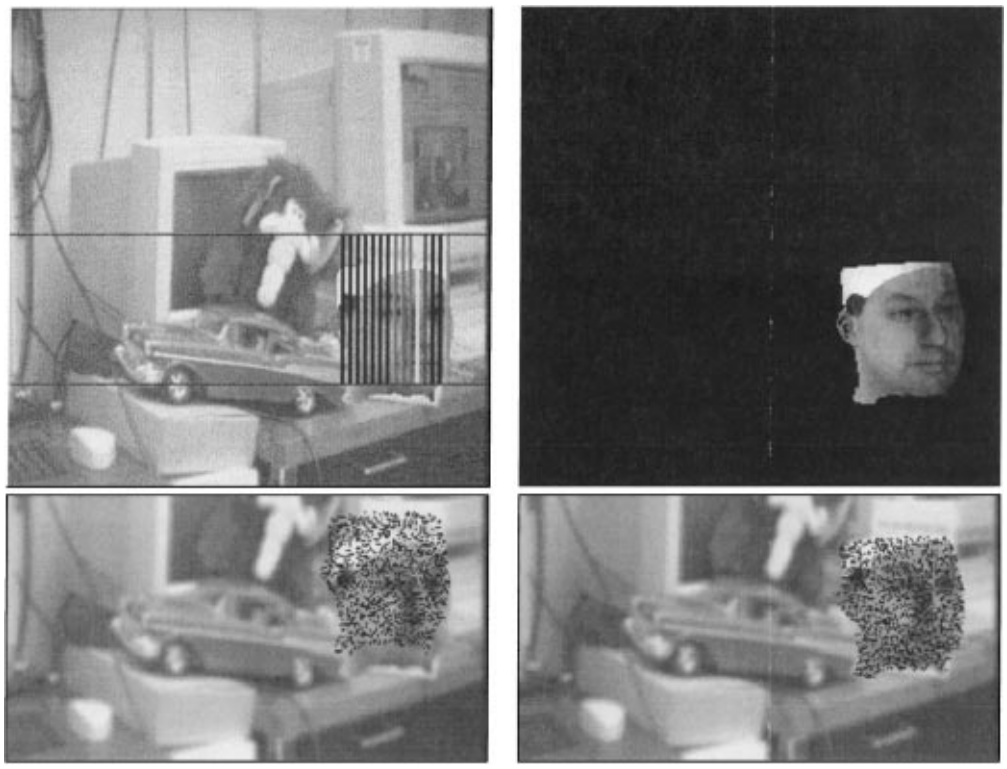

Figure 14. Complete System: Detection and Pose estimation. The input image with a 20 deg. rotation, and the DP match are shown at top left. Results of the partial 3D pose solution using the DP-method, which is used as the initial pose given to the MI recognition system are shown at bottom left. This pose is illustrated by an overlay of points subsampled from the object. The partial pose estimate is too high—the y estimate is off by a few cm. The y information was derived only from knowledge of which image strip was involved in the match. The estimated y-rotation is off by about 2 degrees from the ground truth pose. The final pose is illustrated by a rendering of the 3D model in the top right, and by an overlay of points subsampled from the object surface in the bottom right. The figure shows that the final solution given by the MI alignment has corrected the y-displacement, and the pose shown in the final rendered image agrees with the input pose.

Two Stage Pose Determination using Mutual Information. The pose determined by the pose solver was used as an initial pose for a final pose refinement by the method of Alignment by Maximization of Mutual Information (MI) (Viola and Wells, 1995). The final pose determined by MI is illustrated on the right in Figs. 14 and 10. This refinement was carried out as a local minimization over the six degrees of freedom of motion of the model. Upon visual inspection, the refinement has effectively corrected errors in the initial pose and aligned the model with the image.

5. Conclusions

We have demonstrated a simple visual search technique, based on dynamic programming, that is useful as a first stage for model based object recognition. The method has some of the characteristics of template correlation, but can solve for more DOF of object motion, while retaining similar complexity.

We have shown experiments to demonstrate that the method is nearly as robust as template correlation, in a domain where template correlation is successful. We have also shown that our method is more robust than simple template correlation in a domain having significant out of plane rotations. This method is most effective for objects with textured patterns (e.g. faces/heads) that can be approximated by an upright cylinder.

While the system can recover good estimates of three of the six degrees of freedom of motion (x and z translation and y rotation), it is very sensitive to rotations about the z-axis and cannot handle more than a ± 5 degree rotation about this axis. This is a limitation of the current system. However, this is the only rotation that we can synthesize reliably from a single example

in order to try the same DP-localization using a few in-plane rotations.

The run-time efficiency of dynamic programming does not come for free—there are constraints that must be met in the design of the objective function, in order to get the guaranteed optimality properties in the solutions, along with the economy. One of these constraints implies that the matches between image and template columns be evaluated independently. This weakens the overall match strictness, by allowing column matches that have inconsistent vertical positions. One may ask: is the method able to maintain adequate robustness despite this relaxation. The empirical evidence shown in Section 4 suggests that the answer is a qualified "yes", at least in our test domain.

An additional benefit of our method is that partial pose information may be obtained for downstream use by local search and verification methods.

In speech recognition using DP, learning methods are well established for automatically deriving the templates (in the form of hidden Markov models) from training examples, and such a technique might be applicable here as well.

Acknowledgments

Some inspiration for this work originated in discussions with J. Hornegger in Erlangen. We thank R. Romano for providing access to her facefinding system and Dr. Kikinis for serving as the subject of the recognition experiments.

Research supported in part by ARPA under ONR contract N00014-95-1-0600.

Notes

1. Of course not all current recognition methods are so restricted. For example, Murase and Nayar and related methods attempt to find the best object from a library that accounts for the data.
2. To make the descriptions concrete, we use the convention that the 1D searches are carried out along the x-axis in the image.
3. This stretching is caused by many image column elements matching the same template element.

References

Baker, H.H. and Binford, T.O. 1981. Depth from edge and intensity based stereo. In *Proc. 7th IJCAI*, pp. 631–636.

Ballard, D. and Brown, C. 1982. In *Computer Vision*. Prentice Hall.

Barrow, H.G. 1976. Interactive aids for cartography and photo interpretation. SRI Tech. Report, SRI International.

Betke, M. and Makris, N. 1995. Fast object recognition in noisy images using simulated annealing. In *Proc. Int. Conf. on Computer Vision*, pp. 523–530.

Beymer, D. 1993. Face recognition under varying pose. AI Memo 1461, Artificial Intelligence Lab at MIT, Cambridge, MA.

Brunelli, R. and Poggio, T. 1993. Face recognition: Features versus templates. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 15(10):1042–1052.

Cootes, T.F., Taylor, C.J., Lanitis, A., Cooper D.H., Graham, J. 1993. Building and using flexible models incorporating gray level information. In *Proc. Int. Conf. on Computer Vision*, Berlin, pp. 242–246.

Corman,T., Leiserson, C., and Rivest, R. 1990. *Introduction to Algorithms*. McGraw Hill.

Cyberware Incorporated Monterey, CA.

Forney, G.D. 1973. The Viterbi algorithm. In *Proceedings IEEE*, Vol. 61, pp. 268–278.

Hornegger, J. 1995. Statistical learning, localization and identification of objects. In *Proc. Int. Conf. on Computer Vision*, Cambridge, MA, pp. 914–919.

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

Huttenlocher, D.P., Lilien, R., and Olson, C. 1996. Object recognition using subspace methods. In *Proc. European Conf. on Computer Vision*, pp. 537–545.

Mahmood, S.T.F. and Zhu, W. 1998. Image organization and retrieval using a flexible shape model. In *Proc. of Content Based Access of Image and Video Libraries*.

Murase, H. and Nayar, S. 1995. Learning and recognition of 3-d objects from brightness images. *AAAI Fall Symposium Series Working Notes*, AAAI.

Press, W. and Flannery, B. et al. 1990. *Numerical Recipes in C*. Cambridge University Press.

Ohta, Y. and Kanade, T. 1985. Stereo by intra and inter-scanline search using dynamic programming. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 7(2).

Pentland, A., Moghaddam, B., and Starner, T. 1994. View-based and modular eigenspaces for face recognition. In *Proc. Computer Vision and Pattern Recognition*, pp. 84–91.

Romano, R., Beymer, D., and Poggio, T. 1996. Face verification for real time applications. *ARPA, IU Workshop*, Vol. 1.

Rowley, H., Baluja, S., and Kanade, T. 1995. Human face detection in visual scenes. *CMU-CS-95-158R*, Carnegie Mellon University, Pittsburg, PA.

Rowley, H., Baluja, S., and Kanade, T. 1998. Rotation invariant neural-network based face detection. In *Proc. Computer Vision and Pattern Recognition*.

Rucklidge, W.J. 1994. Locating objects using the hausdorff distance. *Proc. Int. Conference on Computer Vision*, pp. 457–464.

Schneiderman, H. and Kanade, T. 1998. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *Proc. Computer Vision and Pattern Recognition*.

Sakoe, H. and Chiba, S. 1978. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Trans. Acoustics, Speech, and Signal Proc.*, Vol. ASSP-26, pp. 43–49.

Shashua, A. and Ullman, S. 1988. Structural saliency: The detection of globally salient structures using a locally connected network. In *Proc. Int. Conference on Computer Vision*, pp. 321–327.

Sinha, P. 1994. Object recognition via image invariants: A case study. In *Investigative Opthamology and Visual Science*, Florida.

Sung, K. and Poggio, T. 1994. Example based learning for view-based human face detection. AI Memo 1521, MIT. Cambridge, MA.

Turk, M. and Pentland, A. 1991. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1).

Ullman, S. and Basri, R. 1991. Recognition by linear combination of models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(10).

Vaillant, R., Monrocq, C., and Le Cun, Y. 1994. Original approach for the localization of objects in images. *IEEE Proc. on Vision, Image and Signal Processing*, Vol. 141, No. 4.

Viola, P. and Wells, W.M. 1995. Alignment by maximization of mutual information. In *Proc. Int. Conference on Computer Vision*, Cambridge, MA.

Viterbi, A.J. 1967. Error bounds for convolution codes and an asymptotically optimal decoding algorithm. *IEEE Trans. on Information Theory*, IT-13:260–269.

Yuille, A., Hallinan, P., and Cohen, D. 1992. Feature extraction from faces using deformable templates. *Int. Journal of Computer Vision*, 8(2):99–111.