TEMPLAR: A Wavelet-Based Framework for Pattern Learning and Analysis

Clayton Scott*, *Student Member, IEEE,* and Robert Nowak, *Member, IEEE*

Abstract

Recovering a pattern or image from a collection of noisy and misaligned observations is a challenging problem that arises in image processing and pattern recognition. This paper presents an automatic, wavelet-based approach to this problem. Despite the success of wavelet decompositions in other areas of statistical signal and image processing, most wavelet-based image models are inadequate for modeling patterns in images, due to the presence of unknown transformations (e.g., translation, rotation, location of lighting source) inherent in pattern observations. In this paper we introduce a wavelet-based framework for modeling patterns in digital images. This framework takes advantage of the efficient image representations afforded by wavelets, while accounting for unknown translations and rotations. In order to learn the parameters of our model from training data, we introduce TEMPLAR (Template Learning from Atomic Representations), a novel template learning algorithm. The problem solved by TEMPLAR is the recovery a pattern template from a collection of noisy, randomly translated and rotated observations of the pattern. TEMPLAR employs minimum description length (MDL) complexity regularization to learn a template with a sparse representation in the wavelet domain. We discuss several applications, including template learning, pattern classification, and image registration.

EDICS Category: 1-TREC

C. Scott is with the Department of Electrical and Computer Engineering, Rice University, 6100 S. Main, Houston, TX 77005; Vox: 713-348-3776; Fax: 713-348-6196; Email: cscott@rice.edu.

R. Nowak is with the Department of Electrical and Computer Engineering, University of Wisconsin, 1415 Engineering Drive, Madison, WI 53706; Vox: 608-265-3194; Fax: 608-262-1267; Email: nowak@engr.wisc.edu.

This work was supported by the National Science Foundation, grant nos. CCR-0310889 and ANI-0099148, the Office of Naval Research, grant no. N00014-00-1-0390, the Army Research Office, grant no. DAAD19-99-1-0290, and the State of Texas ATP, grant no. 003604-0064-2001. Portions of this work appeared previously in [1].

TEMPLAR: A Wavelet-Based Framework for Pattern Learning and Analysis

I. Introduction

Wavelet decompositions often provide parsimonious image representations, and this feature has been exploited to devise powerful compression, denoising and estimation methods [2]. Although wavelets provide sparse representations for many real world images, it is difficult to develop a wavelet-based statistical model for a given pattern based on observations of the pattern. This is because in many (perhaps most) applications, the pattern of interest undergoes an unknown or random transformation during data acquisition (e.g., variations in illumination, orientation, translation, and perspective). Modeling the wavelet expansion of such transformed data leads to distorted components, or even components that model the transformations instead of the structure of the underlying object or pattern.

The objective of this work is to address the problem of recovering a pattern from a collection of randomly translated and rotated, noisy observations of the pattern. Examples of applications where this problem arises include surveillance from satellite imagery, inspection of parts on an assembly line, face recognition, and a variety of image registration problems. We introduce a wavelet-based framework for modeling patterns that have undergone unknown translations and rotations in the image acquisition process. To infer the parameters of our model, we propose TEMPLAR (Template Learning from Atomic Representations), an iterative, alternating-maximization procedure for performing penalized maximum likelihood estimation. TEMPLAR employs minimum description length (MDL) complexity-regularization to automatically learn a low-dimensional pattern template from noisy, randomly transformed observations. The resulting template may be applied to classification or pattern synthesis. The learning algorithm itself may be applied to image registration, or to denoising from multiple, unaligned observations. Our approach is similar in spirit to that of Frey and Jojic [3], although in that work, the dimension of the template is fixed in advance and the basis vectors are adaptive. We work with a fixed (wavelet) basis and allow the number of degrees of freedom in the template to vary.

The essential ingredients underlying TEMPLAR are: (i) an orthonormal basis that is well-matched to the pattern structure (i.e., a basis that provides a parsimonious representation of the pattern template); and (ii) MDL-based complexity regularization to promote a low-dimensional template by allowing only the *significant* basis coefficients to contribute to the template. A low-dimensional template is advantageous

because it facilitates the pattern matching process (in both template learning and classification) by giving more weight to significant template coefficients, which model the defining structure of the pattern, and less weight to surrounding clutter.

In this paper, we emphasize discrete wavelet bases since they tend to provide sparse representations for a wide variety of spatial patterns, especially those patterns characterized by their edges. Thus, wavelet-domain representations will lead to lower-dimensional templates than, for example, pixel or Fourier domain representations. Related to this sparseness property, discrete wavelet transforms of spatial patterns tend to exhibit a strong dichotomy into large (significant) and small (insignificant) coefficients that allows for reasonable and tractable statistical models.

In Section 2 we introduce our pattern-theoretic framework and statistical model for patterns. In Section 3 we present TEMPLAR, an iterative algorithm for supervised learning of a pattern template. We also discuss convergence analysis and initialization. In Sections 4 we illustrate applications of TEMPLAR to a variety of problems. Section 5 concludes with a summary and discussion.

II. Pattern-Theoretic Framework and Statistical Model

When a pattern is observed in an image, it can appear at any number of locations, orientations, scales, etc., in the image, depending on the spatial relationship between the image forming device and the pattern. Further uncertainty in pattern observations can be caused by lighting sources, background clutter, observation noise, and deformations of the pattern itself (if the pattern is not rigid, like a human face). Figure 2 (a) shows several observations of an airplane with variable location, orientation, lighting conditions, and background. We model these uncertainties in pattern observations with a hierarchical framework, based on the notion of deformable templates from pattern theory [4]. For our purposes, a template is a noise-free observation of a pattern that can be transformed into an arbitrary observation of the same pattern by applying a deformation to the template, and adding observation noise. Examples of deformations include global operations such as translations and rotations, as well as transformations with localized support, to represent local perturbations of the pattern.

This observation model is hierarchical in that it attempts to de-couple three different aspects of an observed pattern: observation noise, the unknown transformation, and the pattern itself (we do not explicitly model clutter). Thus, a pattern observation is synthesized by generating a realization of the template, (which we treat as a random vector), applying a randomly selected transformation, and adding a realization of the observation noise. One novel aspect of our approach is that we model the template in the wavelet domain. As we argue in Section 3, this allows us to develop a template learning algorithm

that takes advantage of the properties of the wavelet transform. We now describe in more detail the individual stages of the hierarchical framework.

Assume that pattern observations are $N_1 \times N_2$ images, thought of as N-dimensional real-valued vectors, where $N = N_1 N_2$. Let the random vector $\mathbf{W} = (W_1, \ldots, W_N)^T$ denote the wavelet coefficients of the pattern template. By viewing the template as a random variable, we are able to model variations in the intensity of the patterns, due to changes in illumination, for example. The wavelet-domain template induces a spatial-domain template by the relation $\mathbf{W} = \mathcal{W}\mathbf{Z}$, where $\mathcal{W}$ denotes a wavelet discrete transform. In this work, we restrict attention to 2-D separable, orthonormal wavelet bases. It will be convenient to think of $\mathcal{W}$ as an $N \times N$ orthogonal matrix.[1]

In real-world images, we often observe two "flavors" of wavelet coefficients: large coefficients, corresponding to wavelet basis functions that match edges in the image, and small coefficients, corresponding to smooth regions of the image or noise. We refer to the former as *significant* coefficients, and the latter as *insignificant* coefficients. This dichotomy into large and small coefficients is a useful approximation that has lead to successful wavelet-based signal processing techniques [5], [6]. Our wavelet-domain statistical model is based on this intuition.

Assign to each W_i a state variable s_i taking on the values 0 or 1, with $s_i = 1$ indicating a significant coefficient, and $s_i = 0$ indicating an insignificant coefficient. We model the marginal pdf of the insignificant wavelet coefficients as $f_{W_i|s_i}(w_i|s_i = 0) = \mathcal{N}(w_i|\mu_{i,0}, \sigma^2_{i,0})$; and, we model the marginal pdf of the significant wavelet coefficients as $f_{W_i|s_i}(w_i|s_i = 1) = \mathcal{N}(w_i|\mu_{i,1}, \sigma^2_{i,1})$. Here, $\mathcal{N}(x|\mu, \sigma^2)$ denotes a Gaussian density with mean μ and variance σ^2, as a function of the variable x. For $m = 0, 1$, define $\boldsymbol{\mu}_m \equiv (\mu_{1,m}, \ldots, \mu_{N,m})^T$, and define $\boldsymbol{\Sigma}_m = \text{diag}(\sigma^2_{1,m}, \ldots, \sigma^2_{N,m})$, where $\text{diag}(a_1, \ldots, a_n)$ denotes the $n \times n$ diagonal matrix with diagonal entries $a_1, \ldots, a_n$. To simplify our model, we assume $\mu_{i,0} = 0$ and $\sigma^2_{i,0} = \sigma^2_0$ for each i. Thus, significant coefficients have unconstrained mean and variance, while insignificant coefficients are constrained to have mean zero and common variance σ^2_0. This reflects our belief that the insignificant wavelet coefficients model smooth regions or noise. We collect the means and variances of the significant and insignificant wavelet coefficients together in the parameter vector $\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\}$. Because of our simplifying assumptions, we may also write $\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \sigma^2_0\}$.

We assume that the wavelet coefficients are statistically independent. This is a reasonable approximation because the wavelet basis functions are orthogonal and spatially localized. Therefore the joint distribution

[1]A matrix is *orthogonal* if its inverse is its transpose. Equivalently, the columns of the matrix form an orthonormal basis for $\mathbf{R}^N$.

of the wavelet coefficients of the pattern template is specified by $\boldsymbol{\theta}$, together with a configuration $\mathbf{s} = (s_1, \ldots, s_N)^T$ of state variables. In reality, for most real-world images there are dependencies between wavelet coefficients, such as spatial clustering and persistence across scale [6]. Although we do not treat this issue, a more sophisticated wavelet-domain model could be incorporated into the present work using a hidden markov tree [6].

We model template deformations with a collection of geometric transformations $\boldsymbol{\Gamma}_\ell, \ell = 1, \ldots, L$. Each transformation $\boldsymbol{\Gamma}_\ell$ is a translation, rotation, or composition of the two. While these operations are usually thought of as rigid-body actions on $\mathbf{R}^2$, it is also convenient to think of them as as linear operators on $\mathbf{R}^N$ (representing $N_1 \times N_2$ images, where $N = N_1 N_2$.) As such they are represented by sparse, $N \times N$ orthogonal matrices (see Appendix I for implementation details.) The collection $\{\boldsymbol{\Gamma}_\ell\}_{\ell=1}^N$ is fixed before any analysis is done, and is chosen to cover (or at least reasonably approximate) the suspected range of possible transformations. The number of transformations may be reduced (thus speeding up the learning algorithm) if we assume each training image has been crudely preprocessed to compensate for very gross transformations such as large translations.

As the final stage of our observation model, we assume the observed image is corrupted by zero-mean, additive white Gaussian noise with variance σ^2_{obs}. Thus, to synthesize a pattern observation based on a particular template, we first generate a realization $\mathbf{w}$ of wavelet coefficients according to the joint pdf $f_{\mathbf{W}|\mathbf{s}}(\mathbf{w}|\mathbf{s}) = \prod f_{W_i|s_i}(w_i|s_i)$. Next, we obtain a realization $\mathbf{z} = \mathcal{W}^{-1}\mathbf{w}$ of the template by transforming $\mathbf{w}$ into the spatial domain. A transformation $\boldsymbol{\Gamma}_\ell$ is then selected according to some prior distribution (assumed uniform unless otherwise indicated), and applied to the spatial template to form a transformed pattern $\mathbf{y} = \boldsymbol{\Gamma}_\ell \mathbf{z}$. Finally, the pattern $\mathbf{x}$ is generated by adding white Gaussian noise to $\mathbf{y}$.

The density of the observed image, conditioned on the template parameters and transformation index ℓ, is given by

$$p(\mathbf{x} \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell) \;=\; \mathcal{N}(\,\mathbf{x} \,|\, \boldsymbol{\Gamma}_\ell \mathcal{W}^T \boldsymbol{\mu}\,,\, \boldsymbol{\Gamma}_\ell \mathcal{W}^T \boldsymbol{\Sigma} \mathcal{W} \boldsymbol{\Gamma}_\ell^T + \sigma^2_{\text{obs}} \mathbf{I}\,),$$

where $\boldsymbol{\mu} \equiv (\mu_{1,s_1}, \ldots, \mu_{N,s_N})^T$ and $\boldsymbol{\Sigma} \equiv \text{diag}(\sigma^2_{1,s_1}, \ldots, \sigma^2_{N,s_N})$. Since $\boldsymbol{\Gamma}_\ell$ and $\mathcal{W}$ are orthogonal operators, we have $\sigma^2_{\text{obs}}\mathbf{I} = \boldsymbol{\Gamma}_\ell \mathcal{W}^T (\sigma^2_{\text{obs}}\mathbf{I}) \mathcal{W} \boldsymbol{\Gamma}_\ell^T$, and therefore we may write the covariance matrix of the above density as $\boldsymbol{\Gamma}_\ell \mathcal{W}^T (\boldsymbol{\Sigma} + \sigma^2_{\text{obs}}\mathbf{I}) \mathcal{W} \boldsymbol{\Gamma}_\ell^T$. In light of this fact, we see that it is not necessary to model the observation noise separately from the wavelet coefficients. Henceforth we assume any observation noise is captured by the wavelet domain statistical model, and the likelihood of an observed image is

$$p(\mathbf{x} \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell) \;=\; \mathcal{N}(\,\mathbf{x} \,|\, \boldsymbol{\Gamma}_\ell \mathcal{W}^T \boldsymbol{\mu}\,,\, \boldsymbol{\Gamma}_\ell \mathcal{W}^T \boldsymbol{\Sigma} \mathcal{W} \boldsymbol{\Gamma}_\ell^T\,). \tag{1}$$

III. Complexity Regularized Template Learning Via TEMPLAR

In this section we introduce a method for learning template parameters $\boldsymbol{\theta}$ and $\mathbf{s}$ from training data, i.e., a collection $\mathbf{X} = (\mathbf{x}^1, \ldots, \mathbf{x}^T)$ of statistically independent observations of the same underlying pattern, as in Figure 2 (a). In the process, we also learn the transformations, indexed by $\mathcal{L} = (\ell_1, \ldots, \ell_T)$, giving rise to each observation. Our approach is to perform penalized maximum likelihood (PML) estimation of the parameters $\boldsymbol{\theta}, \mathbf{s}$, and $\mathcal{L}$. In particular, we would like to maximize the objective function

$$F(\boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) = \log p(\mathbf{X}|\boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) + c(\mathbf{s})$$

where $c(\mathbf{s})$ is a complexity penalty term that decreases as the number of significant coefficients increases. We include $c(\mathbf{s})$ to balance the trade-off between fitting the data and model complexity, and thus to promote the learning of a template with a sparse wavelet representation.

We select a minimum description length (MDL) based penalty term of the form

$$c(\mathbf{s}) = -2k \log(N),$$

where $k = \sum s_i$ is the total number of significant coefficients [7]–[9]. The negative of this quantity is interpreted as the number of bits required to encode the location and values of the means and variances of the significant coefficients. In detail, following a derivation similar to that of Saito [7], each significant coefficient requires approximately $\log_2(N)$ bits to encode its location, $\frac{1}{2}\log_2(N)$ bits to encode its mean, and $\frac{1}{2}\log_2(N)$ bits to encode its variance. Since the negative log-likelihood is the Shannon code length required to encode the data, we see that maximizing F is equivalent to minimizing the total code (description) length required to encode both data and model parameters. We remark that a full description of the model parameters would also require encoding the transformations. In our experiments, however, we are assuming the transformations are equally likely. Therefore, encoding the transformations would simply add a constant value to the MDL penalty, and the maximization would be unaffected.

We believe that joint maximization of F over all three parameters is intractable (although we have no proof of this fact.) Our approach is to find an approximate solution by using TEMPLAR (Template Learning from Atomic Representations), an iterative alternating-maximization algorithm. The premise is to maximize over one parameter at a time, while holding the other two fixed. We first initialize estimates $\mathbf{s}_0$ and $\mathcal{L}_0$ of the states and hidden transformations, by stipulating, for example, that all scaling coefficients are significant, all wavelet coefficients are insignificant, and all transformations are the identity

transformation. TEMPLAR then proceeds according to

$$\boldsymbol{\theta}_j = \arg\max_{\boldsymbol{\theta}} F(\boldsymbol{\theta}, \mathbf{s}_{j-1}, \mathcal{L}_{j-1}) \quad (2)$$

$$\mathbf{s}_j = \arg\max_{\mathbf{s}} F(\boldsymbol{\theta}_j, \mathbf{s}, \mathcal{L}_{j-1}) \quad (3)$$

$$\mathcal{L}_j = \arg\max_{\mathcal{L}} F(\boldsymbol{\theta}_j, \mathbf{s}_j, \mathcal{L}) \quad (4)$$

Since each step involves a maximization, this produces a non-decreasing sequence of penalized log-likelihood values. The process continues until $F(\boldsymbol{\theta}_j, \mathbf{s}_j, \mathcal{L}_j) = F(\boldsymbol{\theta}_{j-1}, \mathbf{s}_{j-1}, \mathcal{L}_{j-1})$, which is guaranteed to happen in a finite number of iterations, by Theorem 1 below. When the algorithm stops, the current values of $\boldsymbol{\theta}_j$ and $\mathbf{s}_j$ are estimates for the template, and the current value of $\mathcal{L}_j$ contains the best estimate of the transformations that generated the training images from the learned template.

The update steps (2) and (3) of TEMPLAR are simple and efficient (see Appendix II). The third step, which requires an exhaustive search, is the most time consuming. Recall that pattern deformations are modeled by a discretized collection of rigid-body transformations $\boldsymbol{\Gamma}_1, \ldots, \boldsymbol{\Gamma}_L$. With this setup, one iteration of TEMPLAR requires $O(NLT)$ operations, where N is the total number of pixels in a pattern observation, L is the number of transformations, and T is the number of training images. We discuss techniques for reducing the computational complexity in Section III-C.

A. Convergence Analysis

In this section we characterize the template output by TEMPLAR, and discuss properties that are advantageous for pattern analysis. We begin with some theoretical results, before shifting to a more intuitive discussion of TEMPLAR's convergence. Theorems 1 and 2 assume the transformation space has been discretized and contains L elements. For notational convenience, we define $F_j \equiv F(\boldsymbol{\theta}_j, \mathbf{s}_j, \mathcal{L}_j)$. Also, let $\mathcal{T}_j = (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ be the template corresponding to the j-th iteration, where $\boldsymbol{\mu}_j = (\mu_{1,s_1}, \ldots, \mu_{N,s_N})^T$ and $\boldsymbol{\Sigma}_j = \text{diag}(\sigma^2_{1,s_1}, \ldots, \sigma^2_{N,s_N})$, and μ_{i,s_i} and σ^2_{i,s_i} are determined by the current estimates $\boldsymbol{\theta}_j$ and $\mathbf{s}_j$. As noted earlier, the sequence $\{F_j\}$ is non-decreasing. Note that the particular sequence, as well as the terminal value of $(\boldsymbol{\theta}_j, \mathbf{s}_j, \mathcal{L}_j)$, depends on the initial values of $\mathbf{s}_0$ and $\mathcal{L}_0$.

Theorem 1: The sequence of penalized log-likelihood values $\{F_j\}$ generated by TEMPLAR (according to Equations (2-4)) converges and reaches its limit in a finite number of iterations.

Proof: Recall that the observations are N-dimensional, there are L transformations, and T training images. There are 2^N possible configurations of the state vector $\mathbf{s}$, and L^T possible configurations of $\mathcal{L}$. Given $\mathbf{s}_{j-1}$ and $\mathcal{L}_{j-1}$, the values of $\boldsymbol{\theta}_j, \mathbf{s}_j$, and $\mathcal{L}_j$ are uniquely determined. Thus, there are $2^N L^T$ different possibilities for $(\boldsymbol{\theta}_j, \mathbf{s}_j, \mathcal{L}_j)$, and hence at most $2^N L^T$ different values for F_j. A non-decreasing

sequence that takes on a finite number of values must converge and reach its limit after finitely many terms in the sequence. ■

It follows that TEMPLAR terminates in a finite number of iterations. However, Theorem 1 does not tell us about the convergence of the sequence of template estimates $\mathcal{T}_j$, which is what we are ultimately interested in. The following result allows us to address this issue:

Theorem 2: With probability one, if $F_j = F_{j-1}$, then $\mathcal{T}_j = \mathcal{T}_{j-1}$.

The proof is found in Appendix III. An underlying assumption of the proof is that the training images $\mathbf{x}^1, \ldots, \mathbf{x}^T$ are realizations from a probability measure that is absolutely continuous with respect to Lebesgue measure (i.e, it has a density.) When we say "with probability one," we mean that the set of training images $\mathbf{x}^1, \ldots, \mathbf{x}^T$ such that $F_j = F_{j-1}$ and $\mathcal{T}_j \neq \mathcal{T}_{j-1}$ (for some j) has measure zero with respect to this probability measure.

Since $\mathcal{L}_j$ depends only on $\mathcal{T}_j$ and the data, we also have $\mathcal{L}_k = \mathcal{L}_{k-1}$ with probability one, whenever $F_j = F_{j-1}$. Since each term in the sequence $\{\mathcal{T}_j, \mathcal{L}_j\}$ depends only on the previous term and the data (which is fixed), once two consecutive terms are equal, all terms are equal from that point on. This result eliminates the possibility of "cycling," i.e., having the algorithm endlessly alternate between two or more different configurations of $\mathcal{T}$ and $\mathcal{L}$ that yield the same penalized log-likelihood value.

We are unaware of any result relating the template produced by TEMPLAR to the global maximum of F. Morever, due to the discrete nature of the model parameters, the notion of local optimality is ill-defined. The final estimate produced by TEMPLAR does satisfy a weaker criterion, however; it is a *partial optimal solution*. The parameters $\boldsymbol{\theta}^*, \mathbf{s}^*, \mathcal{L}^*$ are a partial optimal solution to the optimization problem at hand if they satisfy:

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \; F(\boldsymbol{\theta}, \mathbf{s}^*, \mathcal{L}^*) \\ \mathbf{s}^* &= \arg\max_{\mathbf{s}} \; F(\boldsymbol{\theta}^*, \mathbf{s}, \mathcal{L}^*) \\ \mathcal{L}^* &= \arg\max_{\mathcal{L}} \; F(\boldsymbol{\theta}^*, \mathbf{s}^*, \mathcal{L}) \end{aligned}$$

This criterion pairs naturally with alternating-maximization algorithms, and is employed frequently in numerical optimization problems of this sort [10], [11]. That TEMPLAR produces a partial optimal solution is an immediate corollary of Theorems 1 and 2.

As the final stage of our analysis, we give an intuitive explanation for why TEMPLAR works. We have already discussed how wavelets decompose a pattern into a few significant coefficients and several insignificant coefficients, and how the MDL principle encourages the formation of a sparse template involving only the significant coefficients. But how does a sparse template aid in the learning process?

A sparse template improves the pattern matching (i.e., registration) procedure of Equation (4). The template only models significant portions (e.g., edges) of the pattern, and hence when a pattern is matched to the template, the clutter present in that image does not significantly affect the value of the likelihood function.

Moreover, when a training pattern is aligned to the template, the significant coefficients become even more significant (quantified in terms of the variable d_i in Appendix II-B). In other words, the significant coefficients become stronger the next iteration, so other training images will become more likely to align themselves with the template. Once a few training images share a common alignment, other training images are likely to gravitate toward that same alignment. We refer to this phenomenon as "jumping on the bandwagon." This feature of TEMPLAR enables it to converge quickly in most cases (fewer than T iterations).

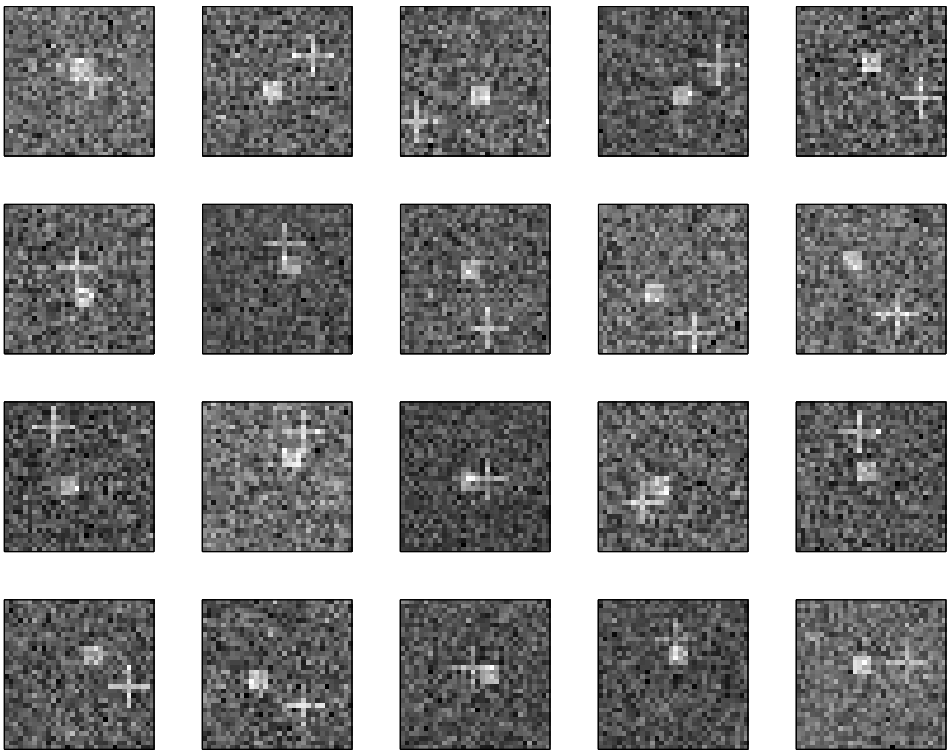
In summary, TEMPLAR generally converges quickly to a template with a sparse wavelet representation. These properties are illustrated in the synthetic example below, and in the applications given in Section IV. In nearly all of our experimentation with TEMPLAR, it converges in 4 to 8 iterations. Of course, the probability of TEMPLAR producing what we would call a good template decreases as the observation noise increases, as the range of transformations in our model decreases, or as the variability of the pattern or clutter increases. In addition, while the sparseness of the template is encouraged by the MDL penalty, it is not guaranteed, being sensitive to the initialization (discussed in III-C).

B. A Synthetic Example

Figure 1 (a) shows twenty training images. Each image is 32×32 pixels, and was formed by adding white Gaussian noise (with variance $\sigma^2_{\text{obs}} = 0.1$) to a particular binary (0 or 1 valued) image consisting of a 4×4 square and a cross. The square occurs within a ± 4 pixel translation (horizontally and/or vertically) of the center of the image and is considered the pattern of interest. The cross occurs at more widely varying translations, independent of the square's location, and is considered clutter.

We apply TEMPLAR to this data using the Haar wavelet transform, and using transformations $\mathbf{\Gamma}_\ell$ that cover translations of up to ± 8 pixels horizontally or vertically, for a total of $(17)^2 = 289$ transformations. The algorithm converges after four iterations for this particular realization of the training data. Figure 1 (b) shows $\mathcal{W}^{-1}\boldsymbol{\mu}^{(j)}$, the template mean transformed into the spatial domain, for each iterarion.

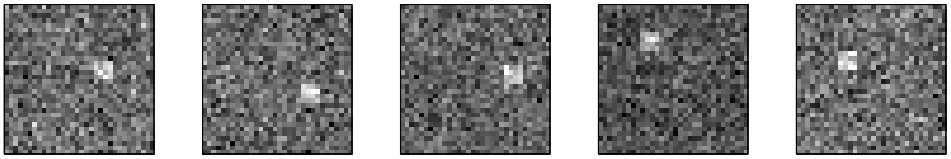
The final template has 10 significant coefficients out of 1024 total coefficients. If we consider a binary image with a 4×4 square of 1s and a background of 0s, then the Haar wavelet transform of such an image can have anywhere from 10 to 56 nonzero wavelet transform coefficients, depending on the location



(a)

$\mathcal{W}^{-1}\boldsymbol{\mu}^{(1)}$ $\mathcal{W}^{-1}\boldsymbol{\mu}^{(2)}$ $\mathcal{W}^{-1}\boldsymbol{\mu}^{(3)}$ $\mathcal{W}^{-1}\boldsymbol{\mu}^{(4)}$

(b)



(c)

Fig. 1. Synthetic example: (a) Training data: randomly translated squares, with randomly translated crosses in the background, plus white Gaussian noise. (b) Sequence of template means, transformed into the spatial domain, for each iteration of the learning algorithm. (c) Observations synthesized from the trained model.

of the square in the image. Thus the sparsest representation of such a square would have 10 nonzero coefficients, compared with 16 coefficients in the pixel domain. The template found by TEMPLAR has the sparsest possible representation for the square, and the 10 nonzero coefficients needed to reconstruct the square at that location are the significant coefficients of the template.

This example illustrates how TEMPLAR converges to a sparse template via jumping on the bandwagon (defined in the previous subsection). In the first iteration $\mathcal{W}^{-1}\boldsymbol{\mu}_1^{(1)}$ is the average of the 20 training images, because the transformations are initialized to be the identity. After it is computed, some wavelet coefficients are "turned on," meaning their states become significant. When the transformations are updated, 12 of the training patterns are aligned at a specific location. At that location, the binary images of the square would have 10 nonzero coefficients, the minimum possible. This is because the square pattern is best represented at this location, given the limited number of significant coefficients. Six more training patterns match this alignment in the second iteration, and two in the third. Upon repeating this experiment 100 times, the average dimension of the template was 10.90, and the average number of iterations needed to reach convergence was 4.65.

We also observe that the final template does not represent any of the clutter present in the training images. Although each training image contains a "cross" pattern in the background, the template does not. This is because the crosses' spatial relationship to the square varies from image to image. It is conceivable that TEMPLAR would try to align the crosses instead of the squares, but this does not happen because the range of transformation used by TEMPLAR does not cover the range expressed by the cross pattern, and because the square is a "stronger" pattern, in that it has more energy.

The common variance of the insignificant coefficients in this example is $\sigma_0^2 = 0.113$, which is slightly larger than the observation noise variance of $\sigma_{\text{obs}}^2 = 0.1$. This discrepancy is to be expected, since background clutter (i.e., the crosses) also contributes to the variance of the insignificant coefficients. Figure 1 (c) shows five pattern observations synthesized from the trained model. The synthesized images closely resemble the training data, except that the clutter (crosses) is no longer present.

C. Initialization and Approximate Search Strategies

In our initial exposition of TEMPLAR, and in the preceding example, the unknown transformations were each initialized to the identity mapping. For more complex scenes, however, this initialization is inadequate for two reasons. First, if the number L of transformations needed is large, the running time of TEMPLAR will be prohibitively large. Second, the algorithm may never "lock on" to a good template, in which case the end result will be an undistinguished blob.

In order to overcome both problems simultaneoulsly, we propose initializing TEMPLAR using an image registration algorithm to align the training patterns. While exact registration will be difficult in the scenarios we have in mind (due to noise, clutter, and pattern variability), a coarse registration may be possible. This will greatly reduce the size of the transformation search space, and allow the first estimate of the template to be at least a rough approximation of the true pattern, thereby encouraging the coarsely aligned patterns to jump on the bandwagon, as discussed in Section III-A.

In our experiments we employ a shape matching methodology based on "shape contexts" [12]. The shape context approach is computationally efficient and has been proven effective at modeling shape information. Registration of two images with shape contexts proceeds in five steps: (i) Extract feature points from both images using, for example, an edge detector; (ii) For each feature point, construct the shape context, which is a histogram of relative locations of the other features points on the same object with respect to a log-polar grid. (iii) By comparing shape contexts, construct a cost matrix that records the similarity of pairs of points in the two images; (iv) Determine corresponding points by solving the linear assignment problem for the cost matrix of the previous step. This can be solved efficiently using well known linear programming techniques; (v) Determine the aligning transformation by minimizing the sum of squared errors over an appropriate class of transformations (e.g., rigid body, affine, thin-plate splines).

After initialization, the optimization in (4) is still the most time consuming step of the learning process. We have found that various approximations can improve this situation in practice. One possibility is to search over translations and rotations independently. This approximation is effective in the latter iterations of TEMPLAR, when the patterns are nearly aligned. Another possibility is to modify the transformation search space $\mathbf{\Gamma}_1, \ldots, \mathbf{\Gamma}_L$ with each iteration. The first iteration would perform a coarse search over a large range of the search space, and subsequent iterations would reduce the range of the search space but increase the precision of the search. This multiresolution search strategy reflects how the template converges in a coarse to fine fashion. We would not lose much by adopting this approach, since the highest resolution coefficients of the template are not learned until later iterations.

Returning to the issue of initialization, if the images are completely free of noise and clutter, shape contexts may align the patterns perfectly, this allowing us to drastically reduce the search space. On the other hand, the shape context approach is somewhat sensitive to clutter. Hence it will only produce a coarse registration of the training patterns. As mentioned above, as long as pre-registration eliminates large transformations, this is enough to significantly decrease the search space and improve the running time of TEMPLAR. From a different viewpoint, TEMPLAR may be seen as one approach to increasing

the robustness of the shape context method and other pattern matching algorithms.

Shape contexts, or more generally, shape matching algorithms based on corresponding feature points, may have a deeper connection to TEMPLAR. Instead of sampling the space of transformations to obtain a finite collection $\mathbf{\Gamma}_1, \ldots, \mathbf{\Gamma}_L$, we could instead maintain a continuously parametrized transformation space, and represent the template by feature points (like those used in the shape context setup). This would allow for extremely rapid transformation updates through the use of well known least-squares techniques (as in step (v) above). Since wavelets are well known for their edge detection capabilities [13], the wavelet domain template model naturally provides a set of feature points that model the pattern but not clutter. We have not developed this idea in this paper, but it would make a sensible extension of the present work for situations where more computational effeciency is required.

IV. Applications of TEMPLAR

All experiments were carried out in MATLAB on a 425 MHz Pentium II processor.

A. Template Learning

In this section we illustrate template learning on real data. Figure 2 (a) shows 20 observations of a randomly translated and rotated toy airplane. These $128 \times 128 = 16,384$ dimensional images were obtained with a digital camera. Note the varying background, lighting intensity, and location of the lighting source. In learning a template for this data, we use transformations $\mathbf{\Gamma}_1, \ldots, \mathbf{\Gamma}_L$ that cover rotations up to one degree accuracy, and translations up to one pixel accuracy within a radius of 50 pixels. Thus the total number of transformations modeled is approximately $\pi(50)^2(360) \approx 2.8$ million. Because this data requires a large number of transformations, we successively refined the transformation search space, as described in section III-C, bringing the total number of transformations per iteration down to around 10,000 or less. Other variations in the observations were not explicitly modeled by transformations.

Using the Haar wavelet transform, TEMPLAR converges to an 853-dimensional template in seven iterations. The mean of the template, transformed into the spatial domain, is shown in Figure 2 (b). The most important observation is that TEMPLAR produces a template that captures the structure of the pattern of interest, while ignoring clutter. The details of the airplane are represented fairly well, while the background is smooth.

Figure 2 (c) is a wavelet-domain map of the significant wavelet coefficients. We see (by looking at the highest resolution subbands) that the significant coefficients model *edges* in the pattern, which is where

the defining information in the pattern is contained. This is what we would expect from complexity-regularized learning: the significant coefficients should be those wavelet coefficients that contribute the most to defining the structure of the pattern. Figure 2 (d) is a map of the variance of the template in the spatial domain. This map was formed by computing the square of each wavelet basis function, multiplying by the variance of the corresponding wavelet coefficient, and summing over all coefficients. Figure 3 displays the registered images produced by TEMPLAR. The template mean in Figure 2 (b) is simply the superposition of these images, reconstructed using only the significant wavelet coefficients.

The size of the transformation search space for this problem is quite large, and correspondingly, the algorithm required about three days to converge. To speed up the algorithm, we reran the experiment after first initializing the transformations using shape contexts as discussed in Section III-C. The patterns were aligned to the 17th training pattern. After initialization, the patterns are roughly aligned to within an accuracy of about 10 pixels. This allowed us to use a much smaller transformation space, and the algorithm ran in about 30 minutes, as opposed to three days. The template that resulted is shown in Figure 2 (e).

B. Pattern Classification

If we are given training images for two or more classes of patterns, we can apply TEMPLAR to produce templates for each class, and use the resulting pattern models for likelihood-based classification. Specfically, if we have trained models $\{\boldsymbol{\theta}_c, \mathbf{s}_c\}_{c=1}^{C}$ for C different classes of patterns, we can use these models to classify an unlabeled test image $\mathbf{x}$ according to a generalized likelihood ratio test (GLRT):

$$c^* = \arg\max_c \left[\max_\ell p(\mathbf{x}|\boldsymbol{\theta}_c, \mathbf{s}_c, \ell) \right]. \tag{5}$$

The GLRT selects the class that has the highest likelihood when evaluated using the most likely transformation for that class. This results in a classifer that is independent of the transformation $\mathbf{\Gamma}_\ell$ that gave rise to $\mathbf{x}$. The number of operations required to classify an image is $O(NM)$.

We illustrate pattern classification by a face recognition example using the Yale Face Database. This database was assembled at the Yale Vision and Robotics Lab, and is available on the Internet at `http://cvc.yale.edu/projects/`. The database consists of 165 images of 15 people, with 11 images of each subject, each having different lighting and/or facial expression. The images for one subject (with the background cropped out) are shown in Figure 4. The same variations in lighting and facial expression were used for each subject. In the data we used, the background was not cropped out and the faces were not aligned.

Belhumeur, Hespanha and Kriegman [14] compare the performance of five different face recognition methods on this data set: the "Fisherface" method, a correlation-based method, the Linear Subspace method, and two variations of the Eigenface method. The performance of each method was evaluated using the "leave-one-out" strategy: to classify an image, that image was removed from the data set and a classifier was constructed for each class using the remaining data. TEMPLAR ranks third out of the six methods, with a misclassification rate of 16.5 % when the background was not cropped out of the picture.

This example is not meant to demonstrate that TEMPLAR is a superior method for face recognition. What is important to notice is that TEMPLAR, while in no way designed for face recognition, performs comparably to these algorithms, *without any pre-processing*. In the experiments reported in [14], each image was manually registered before classification took place. TEMPLAR however requires no pre-processing: the images are automatically registered during the template learning process. Even if the other methods had used an automatic algorithm to register the images beforehand, this registration would by no means align the images in a way that was optimal for each method. With TEMPLAR, however, registration and template learning are jointly optimized under a unified framework.

We would like to emphasize that while TEMPLAR does not achieve the best results for this experiment, it should not be discounted as a methodology for face recognition. The ideas behind TEMPLAR could be incorporated into a system specifically designed for this problem. For example, the wavelet basis could be replaced by a basis tailored for sparse representation of facial features.

As a second example of pattern classification, we consider images containing one of two types of airplanes. Figure 5 (a) shows noise-free observations of these airplanes. Twenty-five training images for each class were artificially generated by applying random transformations and adding Gaussian noise. The transformations considered are horizontal and vertical shifts of up to ± 2 pixels, combined with rotations by multiples of thirty degrees. Figure 5 (b) shows representative training images for each class.

With these training images, we used TEMPLAR to learn a template for each class. The means of these templates are shown in Figure 5 (c). To test the classifier induced by these learned templates, we generated 100 test images for each class in the same way that we generated the training data. The classification rule in (5) yielded no misclassifications.

C. Image Registration

In this section and the next we mention briefly two other applications of TEMPLAR, both of which are interpretations of ideas already discussed. As noted previously, a by-product of TEMPLAR is that the

training patterns are registered (See Figure 3.) TEMPLAR may be especially useful as a tool for image registration in two particular settings.

First, consider the problem of automatically aligning a large number of images all containing the same object. Rather than selecting one of the images as a prototype and aligning the others to it, TEMPLAR procedes in a way whereby all pairs of images interact indirectly through the template. This more unified approach could be especially beneficial when no good prototype exists.

TEMPLAR can also be used as a technique for improving the output of another registration algorithm. Indeed, a first registration of the data may be viewed as the initialization step of TEMPLAR. This is how we used the shape context approach in our experiments. TEMPLAR as a method for improving a registration algorithm is most appropriate when that algorithm makes errors due to clutter or pattern variations.

D. Denoising from Multiple, Unaligned Observations

TEMPLAR can also be used to denoise a collection of misaligned observations of an image or pattern. Previous work on this problem assumes that the pattern is aligned in each of the observations [15]. With our framework, we no longer require this assumption. TEMPLAR automatically registers the observations, and performs denoising by averaging and then setting some wavelet coefficients to zero (although not according to a thresholding rule). The coefficients that are set to zero are precisely the insignificant coefficients. For an example of denoised patterns from previous experiments, see Figures 2 (b) and 5 (c). In the former, TEMPLAR's action is more accurately described as "de-cluttering." This highlights TEMPLAR's ability to remove clutter despite lacking an explicit statistical characterization of the clutter.

V. Conclusion

In this paper we present a wavelet-based approach for modeling observations of patterns with variable location and orientation. We introduce TEMPLAR, an iterative, alternating-maximization algorithm that combines the approximation capabilities of wavelets with MDL complexity-regularization to learn a low-dimensional template from training data. The dimension of the template is automatically inferred from the data. Once the template has been learned, the resulting model can be used for pattern synthesis or pattern classification. TEMPLAR also applies directly to image registration or denoising from unaligned observations. We illustrate these applications with real and synthetic examples. We also discuss approximate search strategies, and initialization based on shape contexts in order to increase TEMPLAR's efficiency.

The primary contribution of this paper is the incorporation of atomic representations into a pattern theoretic framework. In recent years, considerable effort has been invested in finding good atomic representations (e.g., wavelets) for various kinds of data in order to facilitate signal processing tasks. Unfortunately, most such atomic representations are not invariant to transformations of the image, and therefore it is difficult to construct statistical models for observations of a particular pattern using such representations. Pattern theory provides a framework for de-coupling the modeling of transformations from the modeling of the pattern itself. The hierarchical framework presented in this paper, employs this concept to allow a wavelet domain model of pattern observations.

Software for TEMPLAR, in the form of MATLAB m-files, is available at the Rice DSP website: `http://www.dsp.rice.edu/software/`.

APPENDIX I
NOTES ON IMPLEMENTATION OF TRANSLATION AND ROTATION

In order to make translation and rotation as energy-preserving (orthogonal) as possible, the following implementation was used. For translation, the image was treated as a torus. For example, if an image was translated to the right, those pixels on the right edge of the original image would wrap around and appear on the left edge of the translated image. For rotation, when an image was rotated through an angle which was not a multiple of 90 degrees, the corners of the original image that were cropped off by rotation were mapped in a one-to-one manner back to the empty corners of the rotated image.

APPENDIX II
THE DETAILS OF TEMPLAR

In this appendix we provide explicit solutions to the optimization problems in Equations (2-4) and give the associated computational complexities.

A. Estimating Gaussian Mixture Means and Variances

Consider the problem of maximizing F over the parameter $\boldsymbol{\theta} = \{\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1, \sigma_0^2\}$, given training data $\mathbf{X} = (\mathbf{x}^1, \ldots, \mathbf{x}^T)$, with $\mathbf{s}$ and $\mathcal{L}$ fixed. The penalty term $c(\mathbf{s})$ does not depend on $\boldsymbol{\theta}$, so this is equivalent to maximizing the likelihood $p(\mathbf{X} \,|\, \boldsymbol{\theta}, \mathbf{s}, \mathcal{L})$. Denote $\mathcal{W}\boldsymbol{\Gamma}_{\ell_t}^{-1}\mathbf{x}^t$ by $\mathbf{w}^t = (w_1^t, \ldots, w_N^t)^T$. By orthogonality of $\boldsymbol{\Gamma}_{\ell_t}$ and $\mathcal{W}$, it follows from Equation (1) that

$$p(\mathbf{w}^t \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell_t) \;=\; \mathcal{N}(\,\mathbf{w}^t \,|\, \boldsymbol{\mu}\,,\, \boldsymbol{\Sigma}\,). \tag{6}$$

Recall that the i-th wavelet coefficient is modeled as $\mathcal{N}(\mu_{i,s_i}, \sigma^2_{i,s_i})$. Since the wavelet coefficients are independent and the training images are independent, Equation (6) implies

$$\begin{aligned} \hat{\mu}_{i,1} &= \frac{1}{T}\sum_{t=1}^{T} w_i^t \\ \hat{\sigma}^2_{i,1} &= \frac{1}{T}\sum_{t=1}^{T}(w_i^t - \hat{\mu}_{i,1})^2 \end{aligned}$$

are maximum likelihood estimates for the parameters of the Gaussian densities modeling the significant coefficients. If the number of training images T is small, we may wish to normalize by $T-1$ instead of T to achieve unbiased estimates for the variances. To avoid over fitting the model, if $\sigma^2_{i,1} < \epsilon$ for some small ϵ, we set $\sigma^2_{i,1} = \epsilon$.

For the insignificant coefficients, the maximum likelihood estimate for σ_0^2 is

$$\hat{\sigma}_0^2 = \frac{1}{N-k}\sum_{i=1}^{N}(1-s_i)\frac{1}{T}\sum_{t=1}^{T}(w_i^t)^2$$

where $k = \sum s_i$, the number of significant wavelet coefficients. Since $\mathcal{W}$ and $\mathbf{\Gamma}_{\ell_t}^{-1}$ can be computed in $O(N)$ operations, the process of estimating $\boldsymbol{\theta}$ requires $O(NT)$ operations.

B. Determining States of Gaussian Mixtures

In order to maximize F with respect to $\mathbf{s}$, with $\boldsymbol{\theta}$ and $\mathcal{L}$ fixed, it is useful to write

$$\log p(\mathbf{X} \,|\, \boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) = \sum_{i=1}^{N} f_i(s_i)$$

where f_i is the contribution to the overall log-likelihood from the i-th coefficient, as a function of the state variable s_i. The change of variables formula says that

$$p(\mathcal{W}\mathbf{\Gamma}_\ell^{-1}\mathbf{x} \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell)\left|\mathcal{W}\mathbf{\Gamma}_\ell^{-1}\right| = p(\mathbf{x} \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell).$$

By orthogonality of $\mathcal{W}$ and $\mathbf{\Gamma}_\ell$, we have $\left|\mathcal{W}\mathbf{\Gamma}_\ell^{-1}\right| = 1$, so from Equation (6),

$$p(\mathbf{x}^t \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell) = \mathcal{N}(\,\mathbf{w}^t \,|\, \boldsymbol{\mu}\,,\, \mathbf{\Sigma}\,) = \frac{1}{\sqrt{2\pi\,|\mathbf{\Sigma}|}}\exp\left\{-\frac{1}{2}(\mathbf{w}^t - \boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{w}^t - \boldsymbol{\mu})\right\}. \tag{7}$$

Since the observations are assumed statistically independent,

$$p(\mathbf{X} \,|\, \boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) = \prod_{t=1}^{T} p(\mathbf{x}^t \,|\, \boldsymbol{\theta}, \mathbf{s}, \ell_t)$$

Therefore we have

$$\begin{aligned}
\log p(\mathbf{X} \mid \boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) &= \sum_{t=1}^{T} \log p(\mathbf{x}^t \mid \boldsymbol{\theta}, \mathbf{s}, \ell_t) \\
&= \sum_{t=1}^{T} \left[-\frac{1}{2} \log(2\pi\,|\boldsymbol{\Sigma}|) - \frac{1}{2} \sum_{i=1}^{N} \frac{(w_i^t - \mu_{i,s_i})^2}{\sigma_{i,s_i}^2} \right] \\
&= -\frac{1}{2} \sum_{t=1}^{T} \sum_{i=1}^{N} \left[\frac{1}{N} \log 2\pi + \log \sigma_{i,s_i}^2 + \frac{(w_i^t - \mu_{i,s_i})^2}{\sigma_{i,s_i}^2} \right] \\
&= -\frac{1}{2} \sum_{i=1}^{N} \left[\frac{T}{N} \log 2\pi + T \log \sigma_{i,s_i}^2 + \sum_{t=1}^{T} \frac{(w_i^t - \mu_{i,s_i})^2}{\sigma_{i,s_i}^2} \right].
\end{aligned}$$

In the third equality, we use the fact that $|\boldsymbol{\Sigma}| = \prod_{i=1}^{N} \sigma_{i,s_i}^2$. Therefore we may define for $m = 0, 1$

$$f_i(m) = -\frac{1}{2} \left[\frac{T}{N} \log 2\pi + T \log \sigma_{i,m}^2 + \sum_{t=1}^{T} \frac{(w_i^t - \mu_{i,m})^2}{\sigma_{i,m}^2} \right].$$

Now define $d_i = f_i(1) - f_i(0)$. This is the increase in the log-likelihood we achieve when the i-th coefficient is significant as opposed to insignificant. Since our penalty on $\mathbf{s}$ is only a function of the *number* of significant coefficients, if we are going to have any significant coefficients, they should be the coefficients with the largest d_i values. Therefore we order the coefficients so that $d_{i_1} \geq d_{i_2} \geq \cdots \geq d_{i_N}$. The maximum increase in the log-likelihood that can occur from having exactly k significant coefficients is $\sum_{j=1}^{k} d_{i_j}$. But having k significant coefficients carries with it a penalty of $c(\mathbf{s}|k)$. Hence, the number of significant coefficients that maximizes the penalized log-likelihood is

$$k^* = \arg\max_k \left[c(\mathbf{s} \mid k) + \textstyle\sum_{j=1}^{k} d_{i_j} \right],$$

and this maximum is attained by setting $s_{i_j} = 1$ for $j = 1, \ldots, k^*$, and $s_{i_j} = 0$ for $j = k^* + 1, \ldots, N$.

Note that the values w_i^t used to compute d_i were already computed in the previous step, so they do not need to be computed again. The number of operations required to compute the d_i is $O(NT)$, while the number of operations required to compute k^* is $O(N \log N)$, due to the sorting and maximizing.

C. Inferring Hidden Transformations

As in Section II-A, the penalty $c(\mathbf{s})$ does not involve $\mathcal{L}$, so PML estimation of $\mathcal{L}$ given $\boldsymbol{\theta}$ and $\mathbf{s}$ is equivalent to maximum likelihood estimation. Furthermore, since the pattern observations are statistically independent, we may estimate the hidden transformation for each training image independently. The ML estimate $\widehat{\mathcal{L}} = (\hat{\ell}_1, \ldots, \hat{\ell}_T)$ is given by

$$\hat{\ell}_t = \arg\max_{\ell} \log p(\mathbf{x}^t \mid \boldsymbol{\theta}, \mathbf{s}, \ell). \tag{8}$$

We solve this optimization problem by exhaustive search over all L transformations. This log-likelihood can be computed from Equation (8), which requires $O(N)$ operations. We repeat this computation L times for each of the T training images, so computing $\widehat{\mathcal{L}}$ requires $O(NLT)$ operations. This dominates the other two steps of TEMPLAR, so the overall running time of TEMPLAR is $O(NLT)$.

APPENDIX III
PROOF OF THEOREM 2

Let $\mathbf{X} \in \mathbf{R}^{NT}$ denote the collection of N-dimensional training images $\mathbf{x}^1, \ldots, \mathbf{x}^T$, viewed as a single concatenated vector. Set $U \equiv \{\mathbf{X} \mid F_k = F_{k-1} \text{ and } \mathcal{T}_k \neq \mathcal{T}_{k-1} \text{ for some } k\}$. We will show $\lambda(U) = 0$ where λ denotes Lebesgue measure. Then the probabiliy of U will be zero, since the probability measure governing the data is absolutely continuous with respect to the Lebesgue measure.

We may write $F(\boldsymbol{\theta}, \mathbf{s}, \mathcal{L})$ as a function of $\mathbf{X}$:

$$\begin{aligned} F(\boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) &= \log p(\mathbf{X}|\boldsymbol{\theta}, \mathbf{s}, \mathcal{L}) + c(\mathbf{s}) \\ &= -\frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\left[\frac{1}{N}\log 2\pi + \log\sigma_{i,s_i}^2 + \frac{((\mathcal{W}\boldsymbol{\Gamma}_{\ell_t}^{-1}x^t)_i - \mu_{i,s_i})^2}{\sigma_{i,s_i}^2}\right] \end{aligned}$$

Notice that this is a polynomial in the coordinates of $\mathbf{X}$. Now suppose that $\mathbf{s}_k$ and $\mathcal{L}_k$ are fixed, and view $\boldsymbol{\theta}_k = \boldsymbol{\theta}(\mathbf{X}; \mathbf{s}_k, \mathcal{L}_k)$ as a function of the data. From the formulas in section II-A, we see that $\boldsymbol{\theta}_j$, given $\mathbf{s}_j$ and $\mathcal{L}_j$, is a polynomial function of the coordinates of $\mathbf{X}$. Therefore, if $\mathbf{X} \in U$, with $F_k = F_{k-1}$ but $\mathcal{T}_k \neq \mathcal{T}_{k-1}$ for some k, then $\mathbf{X}$ is a zero of the polynomial

$$r(\mathbf{X}) = F(\boldsymbol{\theta}(\mathbf{X}; \mathbf{s}_{k-2}, \mathcal{L}_{k-2}), \mathbf{s}_{k-1}, \mathcal{L}_{k-1}) - F(\boldsymbol{\theta}(\mathbf{X}; \mathbf{s}_{k-1}, \mathcal{L}_{k-1}), \mathbf{s}_k, \mathcal{L}_k).$$

Since $\mathcal{T}_k \neq \mathcal{T}_{k-1}$, this function is nonzero. By the following lemma, the zeros of $r(\mathbf{X})$ comprise a set with measure zero. (See pp. 28-29 of [16] for a proof).

Lemma 1: If $p : \mathbf{R}^n \to \mathbf{R}$ is a nonzero polynomial in n variables, then the set of zeros of p has measure zero with respect to the Lebesgue measure.

Moreover, $r(\mathbf{X})$ is one of a finite family of functions. To see this, observe that $\mathbf{s}$ and $\mathcal{L}$ can take on 2^N and L^T possible values, respectively. Therefore, there are at most $(2^N L^T)^3$ possibilities for $r(\mathbf{X})$, corresponding to the choices for $\mathbf{s}_j$ and $\mathcal{L}_j$, $j = k-2, k-1, k$. Hence, we have shown that if $\mathbf{X} \in U$, then $\mathbf{X}$ belongs to one of a finite number of zero measure sets. Since the finite union of zero measure sets has measure zero, we conclude that U has measure zero. As noted above, this proves the theorem.

REFERENCES

[1] C. Scott and R. Nowak, "Template learning from atomic representations: A wavelet-based approach to pattern analysis," in *Proc. IEEE Workshop on Statistical and Computational Theories of Vision*. Vancouver, CA: Published on the web at `http://www.cis.ohio-state.edu/~szhu/SCTV2001.html`, July 2001.

[2] S. Mallat, *A Wavelet Tour of Signal Processing*. San Diego, CA: Academic Press, 1998.

[3] B. J. Frey and N. J. Jojic, "Transformed component analysis: Joint estimation of spatial transformations and image components," *Proc. IEEE Int. Conf. Comp. Vision*, 1999.

[4] U. Grenander and M. J. Miller, "Representations of knowledge in complex systems," *J. Roy. Stat. Soc.*, vol. 56, no. 3, pp. 1–33, 1994.

[5] H. Chipman, E. Kolaczyk, and R. McCulloch, "Adaptive Bayesian wavelet shrinkage," *J. Amer. Statist. Assoc.*, vol. 92, pp. 1413–1421, 1997.

[6] M. Crouse, R. Nowak, and R. Baraniuk, "Wavelet-based statistical signal processing using hidden Markov models," *IEEE Trans. Signal Processing*, vol. 46, pp. 886–902, 1998.

[7] N. Saito, "Simultaneous noise suppression and signal compression using a library of orthonormal bases and the minimum description length criterion," *Wavelets in Geophysics*, Foufoula-Georgiou and Kumar (eds.), Academic Press, 1994.

[8] J. Rissanen, *Stochastic Complexity in Stastistical Inquiry*. Singapore: World Scientific, 1989.

[9] H. Krim and I. Schick, "Minmax description length for signal denoising and optimal representation," *IEEE Trans. Image Processing*, vol. 45, no. 3, April, 1999.

[10] M. Figueiredo and J. Leitão, "Bayesian esimation of ventricular contours in angiographic images," *IEEE Trans. Med. Imaging*, vol. 11, no. 3, pp. 416–29, 1992.

[11] L. Lakshmanan and H. Derin, "Simultaneous parameter estimation and segmentation of Gibbs random fields using simulated annealing," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 11, pp. 799–813, 1989.

[12] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, 2002.

[13] S. Mallat and W. Hwang, "Singularity detection and processing with wavelets," *IEEE Trans. Inform. Theory*, vol. 38, no. 2, pp. 617–643, 1992.

[14] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Patt. Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–20, 1997.

[15] S. G. Chang, B. Yu, and M. Vetterli, "Wavelet thresholding for multiple noisy image copies," *IEEE Trans. Image Processing*, vol. 9, no. 9, pp. 1631–1635, 2000.

[16] W. M. Wonham, *Linear Multivariable Control*. New York: Springer-Verlag, 1979.

List of figures and captions.

Figure 1 Synthetic example: (a) Training data: randomly translated squares, with randomly translated crosses in the background, plus white Gaussian noise. (b) Sequence of template means, transformed into the spatial domain, for each iteration of the learning algorithm. (c) Observations synthesized from the trained model.

Figure 2 Template learning: (a) Training data: randomly translated and rotated airplanes, with variable background and lighting conditions. (b) Template mean, transformed into the spatial domain. (c) Map of significant wavelet coeficients (indicated by white). (d) Template variance, transformed into the spatial domain. (e) Template mean when TEMPLAR is initialized using the shape context based matching method. This experiment shows that TEMPLAR automatically denoises and declutters a pattern occurring in several noisy, cluttered, and misaligned observations.

Figure 3 As a byproduct of TEMPLAR, the training images are registered.

Figure 4 Images of the first subject in the Yale Face Database, displaying the variety of facial expressions and lighting conditions used. In our expirement, the images were not pre-registered, and the background was not cropped out, as it is in this figure.

Figure 5 Airplane classification experiment: (a) Observations of two airplanes without noise (b) Representative training images (c) Learned airplane templates. This experiment also demonstrates how TEMPLAR produces a denoised image from multiple noisy, misaligned observations. The learned template is a denoised version of the pattern of interest.

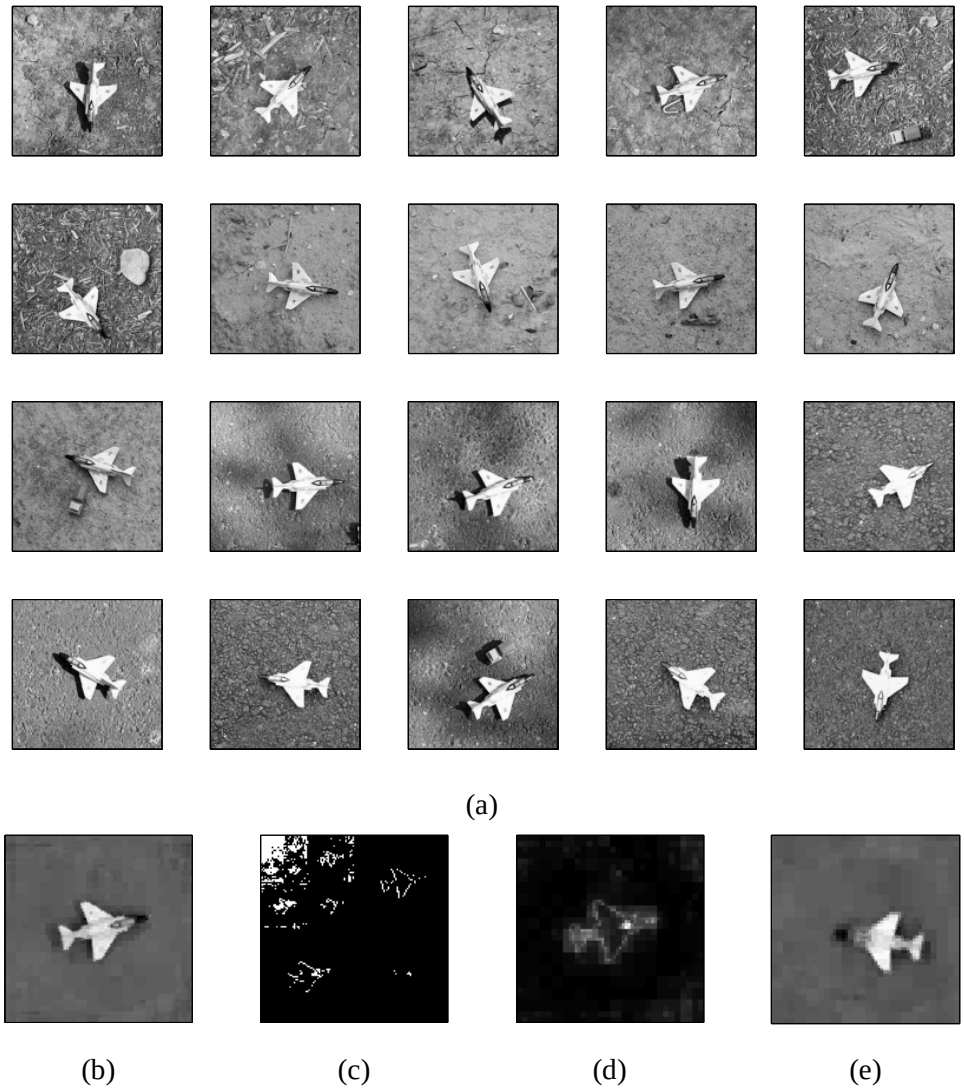


Fig. 2. Template learning: (a) Training data: randomly translated and rotated airplanes, with variable background and lighting conditions. (b) Template mean, transformed into the spatial domain. (c) Map of significant wavelet coeficients (indicated by white). (d) Template variance, transformed into the spatial domain. (e) Template mean when TEMPLAR is initialized using the shape context based matching method. This experiment shows that TEMPLAR automatically denoises and declutters a pattern occurring in several noisy, cluttered, and misaligned observations.

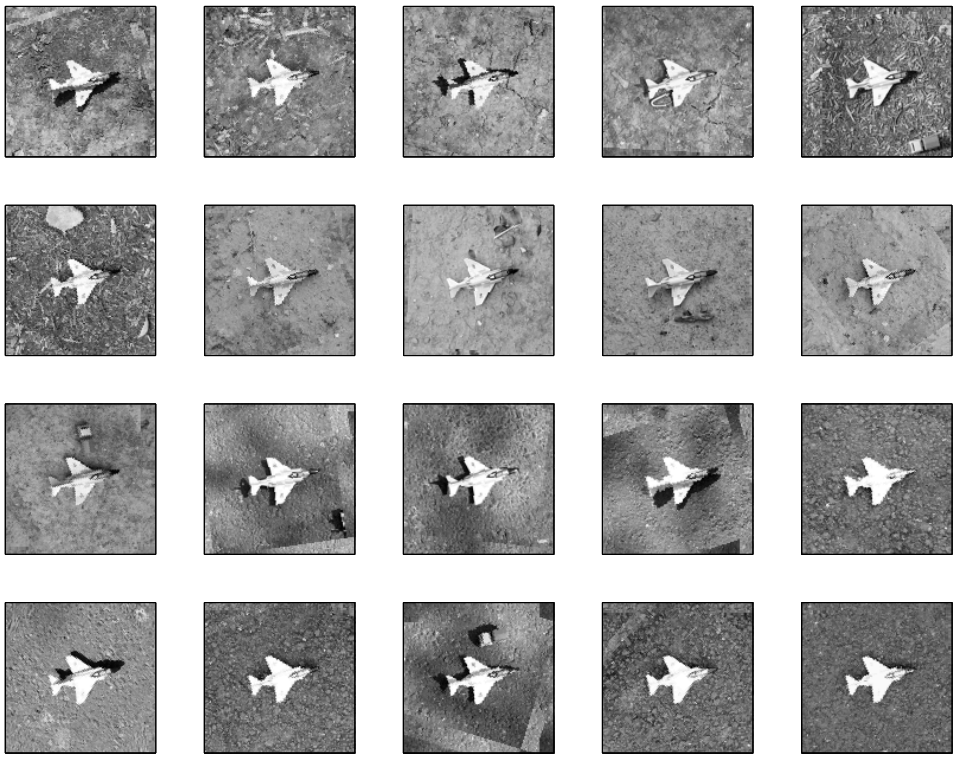


Fig. 3. As a byproduct of TEMPLAR, the training images are registered.

Fig. 4. Images of the first subject in the Yale Face Database, displaying the variety of facial expressions and lighting conditions used. In our expirement, the images were not pre-registered, and the background was not cropped out, as it is in this figure.

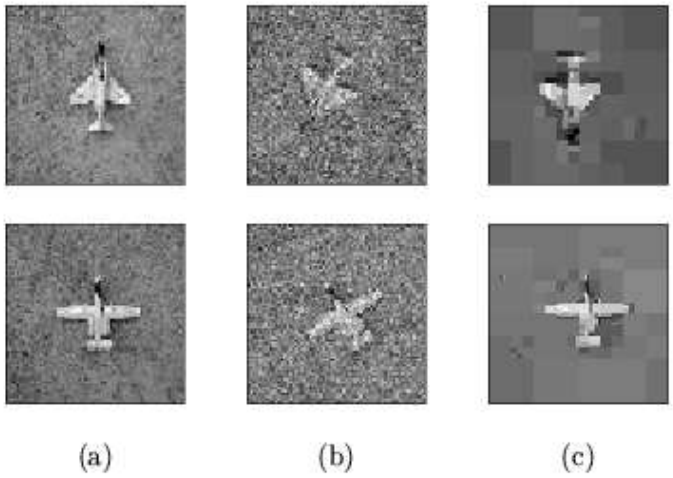


Fig. 5. Airplane classification experiment: (a) Observations of two airplanes without noise (b) Representative training images (c) Learned airplane templates. This experiment also demonstrates how TEMPLAR produces a denoised image from multiple noisy, misaligned observations. The learned template is a denoised version of the pattern of interest.