# A Bayesian Framework for Deformable Pattern Recognition With Application to Handwritten Character Recognition

Kwok-Wai Cheung, *Student Member, IEEE,* Dit-Yan Yeung, *Member, IEEE,* and Roland T. Chin, *Member, IEEE,*

**Abstract**—Deformable models have recently been proposed for many pattern recognition applications due to their ability to handle large shape variations. These proposed approaches represent patterns or shapes as deformable models, which deform themselves to match with the input image, and subsequently feed the extracted information into a classifier. The three components—*modeling, matching,* and *classification*—are often treated as independent tasks. In this paper, we study how to integrate deformable models into a Bayesian framework as a unified approach for modeling, matching, and classifying shapes. Handwritten character recognition serves as a testbed for evaluating the approach. With the use of our system, recognition is invariant to affine transformation as well as other handwriting variations. In addition, no preprocessing or manual setting of hyperparameters (e.g., regularization parameter and character width) is required. Besides, issues on the incorporation of constraints on model flexibility, detection of subparts, and speed-up are investigated. Using a model set with only 23 prototypes without any discriminative training, we can achieve an accuracy of 94.7 percent with no rejection on a subset (11,791 images by 100 writers) of handwritten digits from the NIST SD-1 dataset.

**Index Terms**—deformable models, Bayesian inference, handwriting recognition, expectation-maximization, NIST database.

## 1 Introduction

### 1.1 Deformable Pattern Recognition

MODEL-BASED recognition is a process in which a prior model is searched for in an input image, its occurrence and location are determined, and subsequently its identity is classified. With the use of *deformable models* (DM) which possess shape-varying ability, the approach can be applied to nonrigid patterns, such as human faces, cells, gestures, and handwritten characters. To extract nonrigid shapes by deformable matching, *model deformation* and *data mismatch* are quantified by two criterion functions: one measuring the degree by which the model is deformed and the other measuring how much the data differ from the deformed model. Optimal matching is achieved by minimizing a weighted sum of the two criteria. The weighting factor is the so-called *regularization* parameter, which provides a trade-off between model deformation and data mismatch. Multiclass classification is achieved by defining a set of such models, each containing its own pertinent shape information with an allowed range of deformation specified using a priori information or by training. In the literature, these various steps of the recognition process are often treated separately as if they are independent components.

### 1.2 Previous Works on Deformable Model-Based Handwriting Recognition

Due to the availability of a vast amount of real-world data and the high variability of handwriting styles, handwriting recognition has been used as an excellent testbed for DM-based recognition and is also used in this paper for evaluating our proposed system.

In the literature, there already exist some good studies on DM-based handwritten digit recognition. Wakahara [9] proposed *local affine transform* (LAT) for matching skeleton shapes of characters, each of which is represented by interpolating a set of points. Shape deformation is measured by the smoothness of neighboring local affine transform parameters, and such a measure is invariant to global affine transform. Data mismatch is measured by the sum of the minimum feature distance from each data point to the set of model points. Least-squares fitting is used for minimization, and the regularization parameter is set manually. Classification is based on a dissimilarity measure. The number of prototypes per class is one. Based on a test set with 2,400 digit images, the achieved recognition, substitution, and rejection rates were 96.8 percent, 0.2 percent, and 3 percent, respectively. Another study was conducted by Revow et al. [8], where digits are modeled using elastic spline models. Model deformation is measured by the Mahalanobis distance of the spline control points from a reference vector. The input is assumed to be binary, and the distribution (likelihood) of black pixels is modeled by a mixture of Gaussians with their means uniformly placed along the spline. Data mismatch is defined as the negative log likelihood function. Minimization is performed using the expectation-maximization (EM) algorithm [3], with the regularization parameter manually set. Classification is performed by a back-propagation neural network, where some extracted measures, such as model deformation, data mismatch and affine transform parameters, are the network inputs. The number of prototypes per class is one. Based on the CEDAR database, the best result achieved was a subsitution rate of 1.5 percent for the test set of *goodbs* and 3.14 percent for *bs*, at 0 percent rejection. In a separate study, Jain et al. [5] modeled digits by pixelwise digit boundary templates. Model deformation is measured by the sum of the squared values of a set of displacement function coefficients. Data mismatch is defined by an edge dissimilarity measure between the model template and the input. Minimization is done by a deterministic gradient algorithm, again with the regularization parameter manually set. Classification is based on a weighted sum of two dissimilarity measures. The number of prototypes per class is around 200, which is significantly large to give this method a nonparametric flavor characteristic of nearest neighbor classifiers. Based on a subset of the NIST SD-1 dataset with 2,000 digit images, the lowest substitution rate achieved at 0 percent rejection was 0.75 percent.

The short summary above is by no means exhaustive, but it does show that

1) the DM-based approach is promising for such applications as handwriting recognition and
2) the different components of DM-based recognition are often treated separately as independent components, instead of being integrated into a complete, unified computational framework.

### 1.3 Paper Summary

In this paper, we use the DM-based recognition system proposed by Revow et al. [8] as a base and study how DMs can be integrated seamlessly into a Bayesian framework to give a complete, unified computational framework for modeling, matching, and classifying isolated handwritten characters. To differentiate our system from that of Revow et al., our newly introduced integration does not require any preprocessing of input and manual setting of hyperparameters. The parameter values are determined automatically as part of the integrated framework. Such a modification can make our system more adaptive and portable to other applications. Also, instead of using discriminative classifiers like back-propagation neural networks, the model likelihood (or later called *evidence*)

- *The authors are with the Department of Computer Science, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong. E-mail: {william, dyyeung, roland}@cs.ust.hk.*

*Manuscript received 19 May 1997; revised 15 Sept. 1998. Recommended for acceptance by R. Plamondon.*
*For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 108059.*

$p(\mathbf{D} \mid H_i)$ is also used as the metric for classification which fits naturally into the Bayesian framework. Besides, issues on the incorporation of constraints on model flexibility, detection of subparts, and speed-up are also further investigated.

The rest of the paper is organized as follows. Details of the Bayesian framework are described in Section 2. The procedure of applying the framework to character recognition can be found in Section 3. Section 4 shows the experimental results. The strengths and limitations of our approach are discussed in Section 5. Section 6 concludes the paper.

## 2 Bayesian Framework for Deformable Pattern Recognition

The following provides a brief overview of the Bayesian framework in the context of deformable pattern recognition for handwritten character recognition.

### 2.1 Three Levels of Inference

Let $H_i$ denote the model of the $i$th character class, $\mathbf{D}$ the input image, $\mathbf{w}$ the model parameter vector describing character shape, $\alpha$ the regularization parameter, and $\beta$ the character stroke width. The parameters $\alpha$ and $\beta$ are referred to as hyperparameters.

**Level 1. Modeling:** A number of reference models $\{H_i\}$, one for each class $i$, are constructed based on some model representation scheme that requires prior knowledge.[1] Training is typically involved in model specification.

**Level 2. Matching:** Optimal parameters $\{\mathbf{w}^*, \alpha^*, \beta^*\}$ for each model $H_i$ are estimated by a best match of $H_i$ with the input image $\mathbf{D}$. The process is equivalent to first maximizing the posterior probability density $p(\alpha, \beta \mid \mathbf{D}, H_i)$ and then maximizing $p(\mathbf{w} \mid \mathbf{D}, \alpha, \beta, H_i)$, resulting in a maximum of $p(\mathbf{w}, \alpha, \beta \mid \mathbf{D}, H_i)$.

**Level 3. Classification:** The best model is determined by selecting the model $H_i$ with maximum posterior probability $Pr(H_i \mid \mathbf{D})$ among all the possible $i$.

According to Level 3, $Pr(H_i \mid \mathbf{D})$ of each model has to be computed for classification. Using the Bayes rule and assuming equal prior probabilities $Pr(H_i)$,

$$\begin{aligned}\arg\max_i Pr\left(H_i \mid \mathbf{D}\right) &= \arg\max_i p\left(\mathbf{D} \mid H_i\right) Pr\left(H_i\right) \\ &= \arg\max_i p\left(\mathbf{D} \mid H_i\right), \qquad (1)\end{aligned}$$

where $p(\mathbf{D} \mid H_i)$ is called the *evidence*[2] of model $H_i$.

Expanding $p(\mathbf{D} \mid H_i)$ according to the Bayes rule again and assuming that $\mathbf{D}$ is independent of $\alpha$ and $\mathbf{w}$ is independent of $\beta$,[3]

$$p\left(\mathbf{D} \mid H_i\right) = \iint \frac{p\left(\mathbf{D} \mid \mathbf{w}, \beta, H_i\right) p\left(\mathbf{w} \mid \alpha, H_i\right)}{p\left(\mathbf{w} \mid \mathbf{D}, \alpha, \beta, H_i\right)} p\left(\alpha, \beta \mid H_i\right) d\alpha d\beta, \qquad (2)$$

where $p(\mathbf{w} \mid \alpha, H_i)$ is the prior parameter distribution, $p(\mathbf{D} \mid \mathbf{w}, \beta, H_i)$ is the likelihood function, and $p(\mathbf{w} \mid \mathbf{D}, \alpha, \beta, H_i)$ is the posterior parameter distribution given the data $\mathbf{D}$.

By Laplacian approximation, (2) becomes

$$p\left(\mathbf{D} \mid H_i\right) \simeq \frac{p\left(\mathbf{D} \mid \mathbf{w}, \beta^*, H_i\right) p\left(\mathbf{w} \mid \alpha^*, H_i\right)}{p\left(\mathbf{w} \mid \mathbf{D}, \alpha^*, \beta^*, H_i\right)} p\left(\alpha^*, \beta^* \mid H_i\right) 2\pi \Delta \log \alpha \Delta \log \beta, \qquad (3)$$

where $\Delta\log\alpha$ and $\Delta\log\beta$ are the *effective* ranges of $\alpha$ and $\beta$, respectively, and the maximum a posteriori (MAP) estimates $\{\mathbf{w}^*, \alpha^*, \beta^*\}$ are computed in Level 2 inference,[4] using the models derived as a result of training in Level 1.

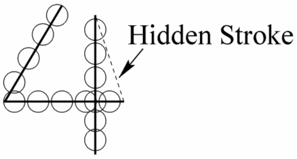


Fig. 1. A “4” digit model with one hidden stroke. There is no pixel on the hidden stroke.

## 3 Deformable Model-Based Character Recognition

In this section, DMs are formulated under a Bayesian framework to yield a unified computational approach to modeling, matching, and classification for deformable pattern recognition.

### 3.1 Model Representation

As in [8], handwritten digits are represented as cubic B-splines, each of which is parameterized by a small set of $k$ control points and the corresponding model parameter vector $\mathbf{w} \in \Re^{2k}$ is formed by concatenating the $x$ and $y$ coordinates of all the $k$ control points, i.e., $\mathbf{w} = (x_1, y_1, x_2, y_2, ..., x_k, y_k)^t$. To achieve affine invariance, each character model in the *model frame* is mapped to the *image frame* of the input character image by an affine transform with parameters represented as $\{\mathbf{A}, \mathbf{T}\}$, where $\mathbf{A}$ is a $2 \times 2$ matrix and $\mathbf{T}$ is a two-dimensional vector.

To represent digits with separate strokes like $\angle$ and $\mid$ for the digit “4,” the above single-spline model can still be used by connecting the disjoint strokes together using *hidden* strokes, along which *no* black pixels are placed. Fig. 1 shows a “4” digit model with one hidden stroke.

Using the spline representation, at least one reference model is constructed for each class. Different people often write very differently even for the same digit, let alone digits from different classes. The variation is sometimes morphological and cannot be satisfactorily represented by elastic deformation of a single digit model, e.g., “7” and “7̵” for the digit class “seven.” Moreover, the distribution of the model parameters for a class may not be represented well by a single mean reference vector. Both suggest that using multiple reference prototypes for each class is inevitable for getting better results. Deriving such categorization automatically from the training data is nontrivial. In this study, we examined the common variations found in real-world handwriting data and constructed the initial models manually (see Section 5.1 for further discussions).

The model parameters to be learned (or estimated) for characterizing a deformable spline include the number of control points $k$ and the mean vector and covariance matrix of $\mathbf{w}$. Using a priori knowledge, a fixed value of $k$ is carefully chosen for each digit model so that the digit shape can be readily represented. Training based on *maximum likelihood* (ML) methods, as in [8], then follows to refine the model parameters using real handwriting data. To categorize the training data automatically to multiple within-class prototypes, we match each training example with all the within-class prototypes and assign it to the prototype with the highest value of model evidence $p(\mathbf{D} \mid H_i)$. Fig. 2 shows all the digit models after training.

[1] In general, there can be more than one model for each digit class, especially if the within-class shape variation is morphological (see Section 3.1).

[2] The evidence $p(\mathbf{D} \mid H_i)$ obtained at Level 2 is referred to as the likelihood for Bayesian classification at Level 3.

[3] These assumptions can be easily justified by their definitions.

[4] The MAP estimate $\mathbf{w}^*$ is needed for approximating $p(\mathbf{w} \mid \mathbf{D}, \alpha, \beta, H_i)$.

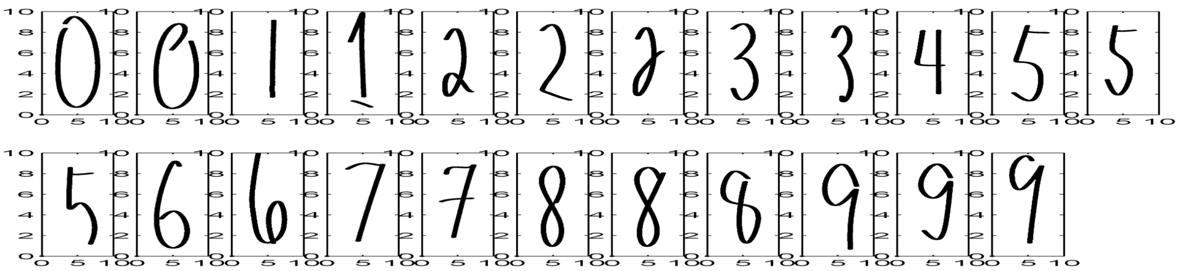


Fig. 2. Digit models after training.

## 3.2 Formulation of Optimization Criteria

### 3.2.1 Model Deformation Criterion

The degree of deformation, quantified by the model deformation criterion $E_w(\mathbf{w})$ of the $i$th model $H_i$, is defined as the Mahalanobis distance of the vector $\mathbf{w}$ of control points from a predefined mean vector $\mathbf{h} \in \Re^{2k}$ as follows:

$$E_w(\mathbf{w}) = \frac{1}{2}(\mathbf{w}-\mathbf{h})^t\,\Sigma^{-1}(\mathbf{w}-\mathbf{h}), \tag{4}$$

where $\Sigma$ is the $2k \times 2k$ covariance matrix of $\mathbf{w}$ for $H_i$ and $\mathbf{w}^t$ denotes the transpose of $\mathbf{w}$. Subsequently, the prior probability distribution of $\mathbf{w}$ is given by

$$p(\mathbf{w} \mid \alpha, H_i) = \frac{1}{Z_w(\alpha)}\exp(-\alpha E_w(\mathbf{w})) \tag{5}$$

where

$$Z_w(\alpha) = \left(\frac{2\pi}{\alpha}\right)^k |\Sigma|^{1/2}, \tag{6}$$

$|\Sigma|$ is the determinant of $\Sigma$, and $\alpha$ is the regularization parameter. The components of $\mathbf{h}$ and $\Sigma$, as discussed in Section 3.1, are computed by ML estimation during the training stage (Level 1 inference).

### 3.2.2 Data Mismatch Criterion

Let the input image be binary. The distribution of black pixels is modeled using a uniformly weighted mixture of Gaussians with their means uniformly placed along the visible portions of the spline.[5] Mismatch between the model and the data is measured by the data mismatch criterion, defined as

$$E_D(\mathbf{w},\mathbf{A},\mathbf{T};\mathbf{D}) = -\sum_{l=1}^{N}\log\left[\frac{1}{N_g}\sum_{j=1}^{N_g}\exp\left(-\beta\frac{\left\|\mathbf{m}_j(\mathbf{w},\mathbf{A},\mathbf{T})-\mathbf{y}_l\right\|^2}{2}\right)\right]. \tag{7}$$

The likelihood function is then given by

$$p(\mathbf{D} \mid \mathbf{w},\mathbf{A},\mathbf{T},\beta,H_i) = \frac{1}{Z_D(\beta)}\exp(-E_D(\mathbf{w},\mathbf{A},\mathbf{T};\mathbf{D})) \tag{8}$$

where

$$Z_D(\beta) = \left(\frac{2\pi}{\beta}\right)^N, \tag{9}$$

$\mathbf{S}_j$ is a $2k \times 2$ matrix containing the corresponding cubic B-spline coefficients, $\mathcal{A}$ and $\mathcal{T}$ are a $2k \times 2k$ block diagonal matrix with $k$ $\boldsymbol{A}$ submatrices placed on its diagonal and a $2k \times 1$ vector formed by concatenation of $k$ $\mathbf{T}$ subvectors, respectively, $\mathbf{m}_j(\mathbf{w},\mathbf{A},\mathbf{T}) = \mathbf{S}_j^t(\mathcal{A}\mathbf{w}+\mathcal{T})$ is the mean of the $j$th Gaussian, $N$ is the number of black pixels in the image, $N_g$ is the number of Gaussians along the spline,[6] $\beta$ is the inverse of the variance of the Gaussians for modeling the character stroke width, $\mathbf{y}_l$ is the location vector of an individual black pixel, and $\mathbf{D}$ denotes the set $\{\mathbf{y}_l \mid 1 \le l \le N\}$. The use of a single global $\beta$ for all the Gaussians results in an implicit assumption that the character stroke is of uniform width.

For simplicity, the prior distribution of the affine transform parameters is assumed to be uniform throughout the paper, except that those affine transform parameters that can lead to very large shearing or shrinking (i.e., illegible characters) are prohibited and the corresponding model configuration is rejected before classification. This avoids the models from degenerating into a line segment which then often matches well with the character "1." Such excessive shearing or shrinking is not commonly found in real handwriting.

### 3.2.3 Combined Criterion Function

Combining the model deformation criterion and the data mismatch criterion, the overall criterion function is given by

$$E_M(\mathbf{w},\mathbf{A},\mathbf{T};\mathbf{D}) = \alpha E_w(\mathbf{w}) + E_D(\mathbf{w},\mathbf{A},\mathbf{T};\mathbf{D}) \tag{10}$$

where $\alpha$ is the regularization parameter. The joint posterior distribution of $\mathbf{w}$ and $\{\mathbf{A}, \mathbf{T}\}$ is defined as

$$p(\mathbf{w},\mathbf{A},\mathbf{T} \mid \mathbf{D},\alpha,\beta,H_i) = \frac{1}{Z_M(\alpha,\beta)}\exp(-E_M(\mathbf{w},\mathbf{A},\mathbf{T};\mathbf{D})) \tag{11}$$

where

$$Z_M(\alpha,\beta) \simeq \int \exp\left(-E_M\left(\mathbf{w},\mathbf{A}^*,\mathbf{T}^*;\mathbf{D}\right)\right)d\mathbf{w} \tag{12}$$

with the assumption that $p(\mathbf{w}, \mathbf{A}, \mathbf{T} \mid \mathbf{D}, \alpha, \beta, H_i) \simeq p(\mathbf{w}, \mathbf{A}^*, \mathbf{T}^* \mid \mathbf{D}, \alpha, \beta, H_i)$ and $\mathbf{A}^*$ and $\mathbf{T}^*$ are the ML estimates of $\mathbf{A}$ and $\mathbf{T}$.

## 3.3 Matching

### 3.3.1 Estimation of Optimal Control Points and Affine Transform Parameters

The MAP estimates of the spline control point vector $\mathbf{w}$ and the affine transform $\{\mathbf{A}, \mathbf{T}\}$ are obtained by maximizing $p(\mathbf{w}, \mathbf{A}, \mathbf{T} \mid \mathbf{D}, \alpha, \beta, H_i)$ in (11) (or equivalently by minimizing $E_M(\mathbf{w}, \mathbf{A}, \mathbf{T}; \mathbf{D})$ in (10)). The EM algorithm [3], similar to the one in [8] but with an affine transform initialization step added, is used here. Applying the EM algorithm to our application, the E-step and the M-step are given by (13), (14), (15), and (16), respectively:

[5] Note that in Revow et al.'s study, an additional uniform noise process is used to model some structure noises caused by bad segmentation. As the dataset we used is relatively well-segmented, whether to introduce the noise process or not does not make a difference. For a more detailed study on badly segmented cases, readers are referred to [2].

[6] Note that the value of $N_g$ will change accordingly as the value of $\beta$ (hence the stroke width estimate) changes.

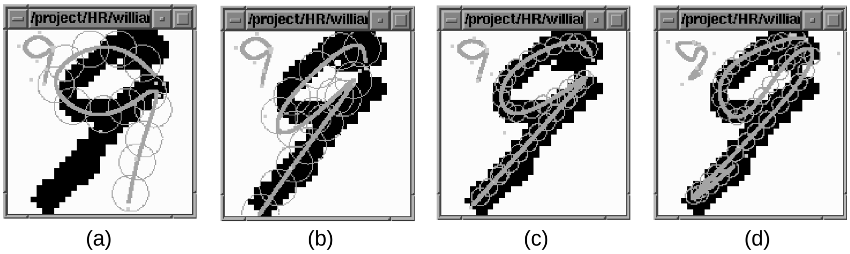


Fig. 3. Illustration of the importance of affine transform initialization. The small character near the upper left corner in each figure is the model before affine transformation. (a) Initial position of the model. (b) Model initialization using the proposed EM procedure for the affine transform parameters. (c) and (d) Final match with and without the proposed affine transform initialization step.

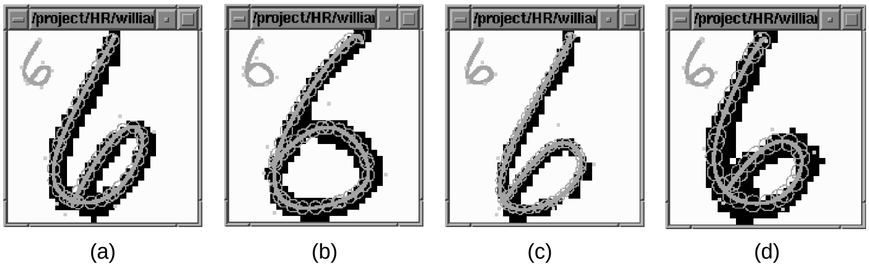


Fig. 4. (a), (b) The value of $\alpha^*$ is estimated automatically based on the degree of deformation of the input character, where $\beta^* \simeq 0.9$ for both cases. (a) $\alpha^* = 3.54$. (b) $\alpha^* = 0.89$. (c), (d) The stroke width of the character increases as the estimated value $\beta^*$ (inversely related to the square of the stroke width) decreases. (c) $\beta^* = 1.72$. (d) $\beta^* = 0.52$.

$$h_j^l\left(\hat{\mathbf{w}}_n, \hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n; \mathbf{y}_l\right) = \frac{\exp\left(-\beta\frac{\left\|\mathbf{m}_j(\hat{\mathbf{w}}_n)-\mathbf{y}_l\right\|^2}{2}\right)}{\sum_p \exp\left(-\beta\frac{\left\|\mathbf{m}_p(\hat{\mathbf{w}}_n)-\mathbf{y}_l\right\|^2}{2}\right)} \tag{13}$$

$$E'_D\left(\mathbf{w}, \mathbf{A}, \mathbf{T}, \hat{\mathbf{w}}_n, \hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n, \mathbf{D}\right) = \sum_{l=1}^{N}\sum_{j=1}^{N_g} \frac{h_j^l\left(\hat{\mathbf{w}}_n, \hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n; \mathbf{y}_l\right)\left\|\mathbf{m}_j(\mathbf{w}) - \mathbf{y}_l\right\|^2}{2} \tag{14}$$

$$Q\left(\mathbf{w}, \mathbf{A}, \mathbf{T}, \hat{\mathbf{w}}_n, \hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n, \mathbf{D}\right) = -\alpha E_w(\mathbf{w}) - \beta E'_D\left(\mathbf{w}, \mathbf{A}, \mathbf{T}, \hat{\mathbf{w}}_n, \hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n, \mathbf{D}\right) \tag{15}$$

$$\left\{\hat{\mathbf{w}}_{n+1}, \hat{\mathbf{A}}_{n+1}, \hat{\mathbf{T}}_{n+1}\right\} = \arg\max_{\mathbf{w},\mathbf{A},\mathbf{T}} Q\left(\mathbf{w}, \mathbf{A}, \mathbf{T}, \hat{\mathbf{w}}_n, \hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n, \mathbf{D}\right) \tag{16}$$

where $\hat{\mathbf{w}}_n$ and $\left\{\hat{\mathbf{A}}_n, \hat{\mathbf{T}}_n\right\}$ are the estimates of the control point vector and the affine transform obtained in the *n*th EM iteration. Fig. 3 illustrates the advantage of using the added affine transform initialization step with which global deformation can be better detected, and subsequently a better final match results.

### 3.3.2 Estimation of Regularization and Stroke Width Parameters

By maximizing the posterior probability density $p(\alpha, \beta \mid \mathbf{D}, H_i)$, the MAP estimates $\alpha^*$ and $\beta^*$ can be determined. As in [7], it relies on the approximation of $Z_M(\alpha, \beta)$, which can be approximated as

$$Z_M(\alpha^*, \beta^*) \simeq \exp(-E_M(\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*; \mathbf{D}))(2\pi)^k \left|\nabla_\mathbf{w}\nabla_\mathbf{w} E_M(\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*; \mathbf{D})\right|^{-1/2}, \tag{17}$$

where $\nabla_\mathbf{w}\nabla_\mathbf{w} E_M(\mathbf{w}, \mathbf{A}, \mathbf{T}; \mathbf{D}) = \alpha\Sigma^{-1} + \nabla_\mathbf{w}\nabla_\mathbf{w} E_D(\mathbf{w}, \mathbf{A}, \mathbf{T}; \mathbf{D})$. By approximating $\nabla_\mathbf{w}\nabla_\mathbf{w} E_D(\mathbf{w}, \mathbf{A}, \mathbf{T}; \mathbf{D})$ by $\beta\nabla_\mathbf{w}\nabla_\mathbf{w} E'_D\left(\mathbf{w}, \mathbf{A}, \mathbf{T}, \hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{D}\right)$ and assuming that the value of $h_j^l\left(\hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{y}_l\right)$ remains constant for all *j* and *l* when $\mathbf{w}$ is near its MAP estimate $\mathbf{w}^*$, it can be shown that the MAP estimates $\alpha^*$ and $\beta^*$ must satisfy

$$\alpha^* = \frac{\gamma}{2E_w\left(\mathbf{w}^*\right)} \quad \beta^* = \frac{2N - \gamma}{2E'_D\left(\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*, \hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{D}\right)}, \tag{18}$$

where

$$\gamma = 2k - \alpha Trace\left(\nabla_\mathbf{w}\nabla_\mathbf{w} E'_M\left(\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*, \hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{D}\right)^{-1}\right) \tag{19}$$

$$\nabla_\mathbf{w}\nabla_\mathbf{w} E'_M\left(\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*, \hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{D}\right) = \alpha\mathbf{I} + \beta\nabla_\mathbf{w}\nabla_\mathbf{w} E'_D\left(\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*, \hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{D}\right). \tag{20}$$

Since there exist no closed-form solutions for $\alpha^*$ and $\beta^*$, the $\{\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*\}$ estimation step and the $\{\alpha^*, \beta^*\}$ estimation step are implemented in an iterative fashion, with (18) serving as the convergence criteria. Some initial values of $\alpha$ and $\beta$ are required.[7] The overall matching algorithm is summarized in Fig. 5.

Fig. 4 illustrates the effect of different degrees of deformation resulting in different values of $\alpha^*$ and the effect of different stroke widths resulting in different values of $\beta^*$. Note that a smaller value of $\alpha^*$ is the result of a higher degree of deformation. This is consistent with the notion that a smaller weighting factor for the model deformation criterion gives the model greater flexibility for a better match with the image data. Also, a smaller value of the automatically estimated $\beta^*$ implies a wider stroke.

### 3.3.3 Model Flexibility Constraints

The flexibility of a deformable spline model is controlled by both the covariance matrix $\Sigma$, which is obtained via training, and by the regularization parameter $\alpha$, which is estimated adaptively based on the input. In the framework, $\alpha$ is assumed to have a uniform prior distribution, i.e., all the values of $\alpha$ are equally probable. This however is undesirable as extremely small values of $\alpha$ may

[7] From our experiments, the convergence of the algorithm was found to be not very sensitive to the initial values of $\alpha$ and $\beta$.

```
For each character model from the candidate model set:
1. Set the spline control points $\mathbf{w}$ to some predetermined (via training) locations.
2. Compute the character image frame and hence a rough initial guess of the affine transform $\{\mathbf{A}, \mathbf{T}\}$ by scaling the model accordingly.
3. Initialize $\{\mathbf{A}, \mathbf{T}\}$ using an EM procedure.
   a) E-step: Compute $h_j^l(\hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{y}_l)$ as defined in (13) for all $j$ and $l$,
   b) M-step: Fix $\mathbf{w}$ in the model frame and compute $\{\hat{\mathbf{A}}, \hat{\mathbf{T}}\}$ by maximizing the $Q$-function defined in (15),
   c) Iterate this initialization process until convergence.
4. Match the model with the image data using an EM procedure.
   a) E-step: Compute $h_j^l(\hat{\mathbf{w}}, \hat{\mathbf{A}}, \hat{\mathbf{T}}, \mathbf{y}_l)$ for all $j$ and $l$,
   b) M1-step: Fix $\{\mathbf{A}, \mathbf{T}\}$ and compute $\hat{\mathbf{w}}$ by maximizing the $Q$-function,
   c) M2-step: Fix $\mathbf{w}$ in the image frame and compute $\{\hat{\mathbf{A}}, \hat{\mathbf{T}}\}$ by maximizing the $Q$-function with respect to $\{\tilde{\mathbf{A}}, \tilde{\mathbf{T}}\}$ where $\tilde{\mathbf{A}} = \hat{\mathbf{A}}^{-1}$ and $\tilde{\mathbf{T}} = \hat{\mathbf{A}}^{-1}\hat{\mathbf{T}}$,
   d) Iterate this matching process until convergence.
5. Compute $\alpha^*$ and $\beta^*$ according to (18).
6. Iterate Steps 4 and 5 for the particular character model until convergence.
```

Fig. 5. The matching algorithm.

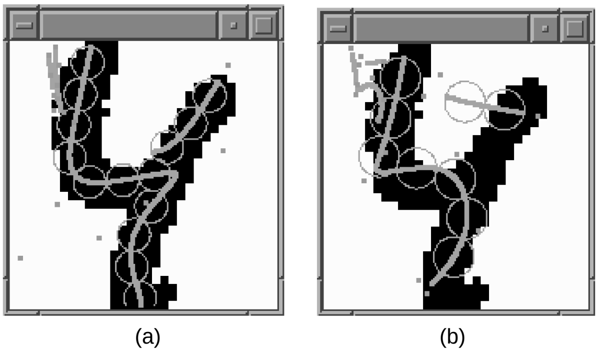


Fig. 6. Avoiding an unfavorable match by imposing model flexibility constraints. (a) Unconstrained match. (b) Constrained match.

```
For any input character: ($h$ = image height; $w$ = image width)
1) create a vertical projection profile $p[i]$ of black pixels, where the profile is computed by counting the number of black pixels in the first continuous black pixel segment for each top-to-bottom vertical scan;
2) compute $ll$ and $rl$ by detecting the left and right margins where $p[i] > 0.6 \times h$;
3) if $ll > 0.5 \times w$, return "Not thick ONE"; /* To avoid confusion with "7" */
4) else
   a) $thickness$ := 0,
   b) for each location from $ll$ to $rl$,
      i) if $p[i] > 0.6 \times h$, increment $thickness$ by one; else break;
5) if $thickness > 6$, return "Thick ONE"; else return "Not thick ONE".
```

Fig. 7. The thick “1” filtering algorithm.

result in good matches between severely deformed models and input characters that do not belong to the model classes (see Fig. 6). This observation implies that the uniform prior assumption for $\alpha$ is inappropriate, allowing too much flexibility for the models. While obtaining an accurate prior distribution for $\alpha$ is in general not easy and may result in a more complicated matching procedure, constraining the value of $\alpha$, according to (18), can be indirectly achieved by constraining the value of the model deformation criterion $E_w(\mathbf{w})$. This implies that the flexibility restriction can be imposed by putting a hard constraint directly on $E_w(\mathbf{w})$ for each individual model. Any matching iteration that results in a value of $E_w(\mathbf{w})$ greater than the threshold will be forbidden. For each individual model, such a threshold can be precomputed as the upper bound of $E_w(\mathbf{w})$ based on its training data. Fig. 6 illustrates how the incorporation of constraints on model flexibility can avoid an unfavorable match of a “5” model to a digit image of “4.”

## 3.4 Classification

### 3.4.1 Evidence Comparison

Classification involves approximating the evidence $p(\mathbf{D} \mid H_i)$ based on $\{\mathbf{w}^*, \mathbf{A}^*, \mathbf{T}^*, \alpha^*, \beta^*\}$ obtained for each of the candidate models. By substituting (5) and (8) into (3), it can be shown that

$$p(\mathbf{D} \mid H_i) \propto \frac{Z_M(\alpha^*, \beta^*)}{Z_w(\alpha^*) Z_D(\beta^*)} \sqrt{2/\gamma}\sqrt{2/(2N-\gamma)}. \tag{21}$$

The quantities $Z_w(\alpha^*)$, $Z_D(\beta^*)$, and $Z_M(\alpha^*, \beta^*)$ can be computed according to (6), (9), and (17), respectively. Finally, classification is determined by finding $i^* = \arg\max_i p(\mathbf{D} \mid H_i)$, and the character is classified as $H_{i^*}$. Ambiguous input rejection is done by computing the posterior class probability $P(H_i \mid \mathbf{D})$, given by

$$P(H_i \mid \mathbf{D}) = \frac{p(\mathbf{D} \mid H_i) P(H_i)}{\sum_{j=1}^{M} p(\mathbf{D} \mid H_j) P(H_j)} \tag{22}$$

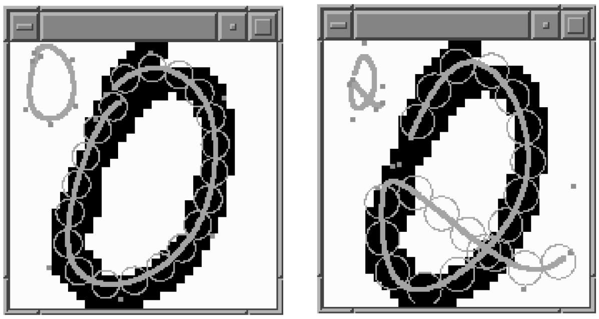
and comparing it with a predefined confidence threshold.

### 3.4.2 Likelihood Inaccuracy

The success of Bayesian inference greatly relies on model accuracy. In our experiments, it is found that any inaccuracy in $\beta$ estimation, and, hence, the likelihood estimation, can easily confuse the evidence comparison among the best few candidates. To correct such an inaccuracy, the classification rule can be modified by first computing the maximum evidence value $P(\boldsymbol{D} \mid H_{i^*})$ and then forming a short-list of model candidates, each with its value of $P(\boldsymbol{D} \mid H_i)$ close enough (determined by a predefined threshold) to $P(\boldsymbol{D} \mid H_{i^*})$. To come up with the short list, we assume that the difference in data mismatch among the model candidates is negligible and, hence, the candidate with the greatest value of the prior $p(\mathbf{w} \mid H_i)$ is then the classified output.

### 3.4.3 Filtering Normalized “1”

According to the report by the NIST group [4], all the segmented character images in the NIST SD-1 dataset are normalized first to 20 × 32 and then put to the center of a 32 × 32 image. This leads to the existence of many thick “1” digits in the database and causes serious misclassification as all models can find good fits to them. As the normalization step causes the above-mentioned difficulty and normalization is in fact not required at all for our approach, instead of collecting new data for class “1,” we derived a simple filter to preclassify all the thick “1” digits. The algorithm is described in Fig. 7.



(a) (b)

Fig. 8. Illustration of the subpart problem. See Section 3.4.4 for explanation. (a) Model "0." (b) Model "2."

### *3.4.4 Subpart Detection*

The subpart problem arises when some models in the model set are subparts of some others. For example, the model "0" and "2" sometimes can fit almost equally well to a "0" digit image (see Fig. 8). By noting the obvious difference that the "2" model has several Gaussians resting on the white space, the situation can be detected by incorporating the following detection rule: *If the first ranked class is "2" but with Gaussians on the white space while the next ranked class is "0" without any Gaussians on the white space, then the output class is "0."* In our study, using some prior knowledge, we create a rule base containing four rules to distinguish between the following pairs of digits:

1) "0" and "2";
2) "4" and "9";
3) "7" and "9"; and
4) "3" and "8,"

where each former character model is a subpart of the latter.

## 4 Experimental Results on the NIST Handwritten Digits

The proposed framework has been applied to recognize isolated handwritten digits in the NIST Special Database 1 for performance evaluation. Three subsets of the NIST data, denoted as *S*1, *S*2, and *S*3, respectively, are used in our experiment. *S*1 is the training set which contains 11,660 digits (32 × 32 binary pattern each) written by 100 different individuals (`f0000-f0099` in NIST SD-1). *S*2 and *S*3 are two test sets which contain digits written by another group of 100 individuals (`f0100-f0199` in NIST SD-1). Their sizes are 1,000 and 11,791 respectively. The testing result is summarized in Table 1.

The proposed methods increase the recognition accuracy to different extents, where the model flexibility constraint incorporation is the most effective one based on our experience. By combining all of them, we achieve an accuracy of 94.7 percent at 0 percent rejection.

## 5 Limitations and Future Work

### 5.1 Model Set Construction

Although the proposed framework is generic for any shape recognition applications, porting it to other applications requires a manual and intelligent process of creating the class reference shapes. In order to automate the process, we are still lacking

1) an algorithm to construct shape representations (cubic B-splines in our case) for different classes and
2) an algorithm to create an optimal set of reference models.

For the extreme case with all the training data used as reference models, a 99.25 percent accuracy has been achieved by Jain et al. [5] on a subset of handwritten digits from NIST. However, this nearest-neighbor-type approach is computationally too expensive for practical applications. For our case, by using only 23 models (which is, of course, by no means optimal), a 94.7 percent accuracy is achieved (though based on another subset of NIST data but of much larger size than that in Jain et al. [5]).

### 5.2 Fast Implementation

The iterative deformable matching procedure is known to be computationally expensive. Also, if a multiclass DM-based recognition system is implemented directly on sequential computers, it is apparent that this approach will further suffer due to the scale-up problem, i.e., computation increases linearly with the number of candidate models. Other than hardware solutions like parallelization or special-purpose hardware, some efficient software techniques such as geometric hashing [6] have been proposed to tackle this problem. However, most of these techniques require the object to be represented by a set of pre-extracted salient points, like corners, and the deformation allowed is, so far, very restricted.

For fast matching, by noting the information redundancy in the input image, subsampling techniques are expected to help. We have tested two subsampling techniques:

TABLE 1
Recognition Accuracy Obtained Based on Combinations of Different Methods

| Methods | "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" | Overall |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Training set: S1 (11,660 digits) | | | Test set: S2 (1,000 digits) | | | | | | | | |
| B | 99% | 54% | 79% | 84% | 82% | 83% | 76% | 66% | 82% | 84% | 78.8% |
| B+R | 99% | 69% | 96% | 96% | 95% | 94% | 96% | 90% | 92% | 94% | 92.1% |
| B+R+O | 99% | 91% | 96% | 96% | 95% | 94% | 96% | 90% | 92% | 94% | 94.3% |
| B+R+O+P | 99% | 91% | 96% | 96% | 95% | 94% | 98% | 94% | 93% | 95% | 95.1% |
| Training set: S1 (11, 660 digits) | | | Test set: S3 (11,791 digits) | | | | | | | | |
| B+R+O | 98.5% | 95.1% | 94.9% | 94.9% | 92.7% | 94.8% | 93.0% | 92.5% | 90.1% | 91.5% | 93.8% |
| B+R+O+P | 99.3% | 94.6% | 95.6% | 94.7% | 93.2% | 95.7% | 94.8% | 92.9% | 91.4% | 92.9% | 94.4% |
| B+R+O+P+S | 99.4% | 94.6% | 95.5% | 94.6% | 94.0% | 95.7% | 94.8% | 93.3% | 92.5% | 92.6% | 94.7% |
| B+R+O+P+S+Rj-4.9 | 99.4% | 97.5% | 95.9% | 95.2% | 96.1% | 95.7% | 95.6% | 95.3% | 94.7% | 93.8% | 95.9% |
| B+R+O+N2 | 99.5% | 97.3% | 98.2% | 97.7% | 96.5% | 98.5% | 98.2% | 96.8% | 94.0% | 97.5% | 97.4% |
| B+R+O+N3 | 99.7% | 98.1% | 98.8% | 98.7% | 97.8% | 99.3% | 99.5% | 98.1% | 97.5% | 99.1% | 98.7% |
| B+R+O+N4 | 100% | 98.2% | 99.3% | 99.3% | 98.2% | 99.6% | 99.8% | 98.9% | 98.8% | 99.6% | 99.2% |

*The abbreviations stand for: B—basic framework (Section 3.4.1), R—restriction on model flexibility (Section 3.3.3), O—thick "1" filtering (Section 3.4.3), P—considering prior in final decision (Section 3.4.2), S—subpart penalty (Section 3.4.4), Rj-4.9—rejection at 4.9 percent, Nn—correct class within best* n *models. The thresholds used in method R are obtained via training.*

1) uniform random sampling (50 percent of data sampled) and
2) same uniform random sampling plus all boundary pixels.

The achieved speed-up factors are 1.69 and 1.2 with approximately 0.9 percent and 0.2 percent accuracy sacrificed, respectively.

To alleviate the scale-up problem, we have also tested a competitive mixture of DMs which is basically using the early elimination approach to save unnecessary computation resulting from the matching with irrelevant models. For a particular experiment [1], implementing this idea where seven of the 10 models are eliminated after the affine initialization step, we have achieved a speed-up factor of 1.9 at the expense of 1.2 percent accuracy drop. It is believed that some better competitive process should be worth investigating to achieve higher speed-up and lower accuracy drop.

## 6 Conclusion

A unified framework based on Bayesian inference is proposed for modeling, matching, and classifying patterns which exhibit large variations in shape. DMs are incorporated as an important component in this Bayesian framework. Handwritten character recognition is used to provide a meaningful and realistic testbed for this DM framework. For handwritten digits from the NIST SD-1 dataset, by using only 23 prototypes, we have achieved an accuracy of 94.7 percent on 11,791 test examples. No discriminative training is used at all in the whole framework, and the same approach can readily be applied to other shape recognition problems. Developing an automatic model set construction algorithm and a fast implementation of the matching and classification step will be of interest to further research.

Using this approach, the obvious next step is to formulate character segmentation of cursive handwritten words [2] as a component of the overall framework so that character segmentation and isolated handwritten character recognition can be tightly coupled together for better interaction and feedback to achieve a higher level of performance.

## Acknowledgment

This research is supported in part by the Hong Kong Research Grants Council under grants HKUST 614/94E and HKUST 746/96E, and the Sino Software Research Centre under grant SSRC 95/96.EG12.

## References

[1] K.W. Cheung, D.Y. Yeung, and R.T. Chin, “Competitive Mixture of Deformable Models for Pattern Classification,” *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 613-618, San Francisco, Calif., June 1996.

[2] K.W. Cheung, D.Y. Yeung, and R.T. Chin, “Robust Deformable Matching for Character Extraction,” *Proc. Sixth Int’l Workshop Frontiers in Handwriting Recognition*, Taejon, Korea, Aug. 1998.

[3] A.P. Dempster, N.M. Laird, and D.B. Rubin, “Maximum-Likelihood From Incomplete Data Via the EM Algorithm,” *J. Royal Statistical Soc., Series B*, vol. 39, pp. 1-38, 1977.

[4] J. Geist, R.A. Wilkinson, S. Janet, P.J. Grother, B. Hammond, N.W. Larsen, R.M. Klear, M.J. Matsko, C.J.C. Burges, R. Creecy, J.J. Hull, T.P. Vogl, and C.L. Wilson, “The Second Census Optical Character Recognition Systems Conference,” Technical Report NISTIR 5452, U.S. Nat’l Inst. of Standards and Technology, 1994.

[5] A.K. Jain and D. Zongker, “Representation and Recognition of Handwritten Digits Using Deformable Templates,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 12, pp. 1,386-1,390, Dec. 1997.

[6] Y. Lamdan and H.J. Wolfson, “Geometric Hashing: A General and Efficient Model-Based Recognition Scheme,” *Proc. Second Int’l Conf. Computer Vision*, pp. 238-249, Tampa, Fla., Dec. 1988.

[7] D.J.C. MacKay, “Bayesian Interpolation,” *Neural Computation*, vol. 4, no. 3, pp. 415-447, 1992.

[8] M. Revow, C.K.I. Williams, and G.E. Hinton, “Using Generative Models for Handwritten Digit Recognition,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 6, pp. 592-606, June 1996.

[9] T. Wakahara, “Shape Matching Using LAT and Its Application to Handwritten Numeral Recognition,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 618-629, June 1994.