Looking for Shapes in Two-Dimensional, Cluttered Point Clouds

Anuj Srivastava and Ian H. Jermyn, *Member, IEEE,*

Abstract—We study the problem of identifying shape classes in point clouds. These clouds contain sampled contours and are corrupted by clutter and observation noise. Taking an analysis-by-synthesis approach, we simulate high-probability configurations of sampled contours using models learnt from the training data to evaluate the given test data. To facilitate simulations, we develop statistical models for sources of (nuisance) variability: (i) shape variations within classes, (ii) variability in sampling continuous curves, (iii) pose and scale variability, (iv) observation noise, and (v) points introduced by clutter. The variability in sampling closed curves into finite points is represented by positive diffeomorphisms of a unit circle and we derive probability models on these functions using their square-root forms and the Fisher-Rao metric. Using a Monte Carlo approach, we simulate configurations from a joint prior on the shape-sample space and compare them to the data using a likelihood function. Average likelihoods of simulated configurations lead to estimates of posterior probabilities of different classes and, hence, Bayesian classification.

Index Terms—Shape classification, clutter model, Fisher-Rao metric, planar shape models, diffeomorphisms

1 Introduction

The classification and recognition of objects in images is an important problem in machine vision, biometrics, medical image analysis, and many other branches of science. A common approach is to represent the objects of interest with certain discriminant features, and then use some statistical models on these feature spaces for classification. An important feature of many objects is their **shape** and, as a consequence, shape analysis has become an integral part of object classification [1], [2]. One way to use shape analysis is to estimate the boundaries of the objects (in images) and to analyze the shapes of those boundaries in order to characterize the original objects. Towards that end, there have been several papers in the literature on analyzing the shapes of continuous, closed, planar curves (see for example [3], [4] and others referenced therein). While such continuous formulations are fundamental in understanding shapes and their variability, practical situations mostly involve heavily under-sampled, noisy, and cluttered discrete data, often because the process of estimating boundaries uses low-level techniques that extract a set of primitives (points, edges, arcs, etc.) in the image plane. (We will restrict attention to only the points in this paper—some examples of point sets derived from real images are shown in Figures 17, 21 and 22—but the method generalizes to more complex primitives.) Therefore, an important problem in object recognition is to (probabilistically) relate a given set of primitives to pre-determined (continuous) shape classes and to classify the shape of this set using a *fully statistical framework*.

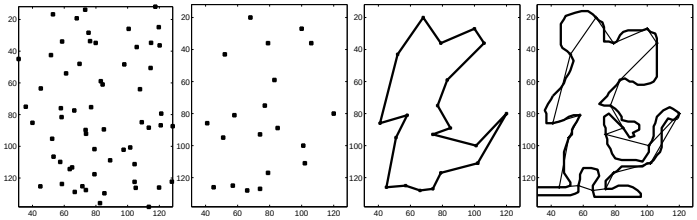
- *A. Srivastava is with the Department of Statistics, Florida State University, Tallahassee, FL 32306.*
- *I. H. Jermyn is with the Ariana project-team, Project ARIANA, INRIA, Sophia Antipolis, France.*

Manuscript received April 19, 2005; revised January 11, 2007.

1.1 Problem Challenges

The biggest challenge is to select and organize a large subset of the given primitives into shapes that resemble the shapes of interest. The number of permutations for organizing primitives into possible shapes is huge. For example, if we take the primitives to be points, the number of possible polygons using 40 distinct points is of the order of 10^{47}. If we select only 20 points out of the given 40 and form a polygonal shape, the number of possibilities is still approximately 10^{29}. To form and evaluate all these shape permutations is impossible. Similar to [5], our solution is to analyze these configurations through synthesis, *i.e. to synthesize high-probability configurations from known shape classes and then to measure their similarities with the data*. Although this approach has far smaller complexity than the bottom-up combinatoric approach, the joint variability of all the unknowns is still enormous. To go further, one must use the structure of the problem to break down the variability into components, and then probabilistically model the components individually. Through an example presented in Figure 1, we will try to explain these components.

1. **Clutter Rejection**: It is not just the object boundary that will generate primitives: the background and the object interior will too. From the perspective of the shape analysis of object boundaries, these background and interior points are labelled as clutter. Perhaps the most difficult issue is to determine which primitives belong to the object boundary and which belong to the clutter. Discarding clutter takes us from (a) to (b) in Figure 1.
2. **Ordering**: Even if the primitives belonging to the object boundary were known, their ordering along the boundary is most probably unknown. If n primitives are used to form a polygonal shape, there are $n!$ orderings. Having a specific ordering moves us from (b) to (c) in Figure 1.
3. **Classification**: Even for an ordered set of primitives,



(a) Data (b) No Clutter (c) Ordering (d) Classification

Fig. 1. Problem Challenges: The point cloud in (a) contains clutter as well as the shape of interest. The removal of clutter leads to points in (b) which when ordered result in a polygon in (c). Subsequently, this polygon can be used for shape classification as in (d).

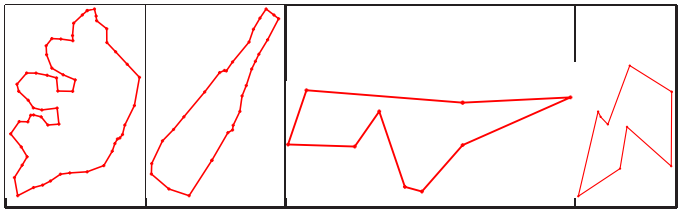


Fig. 2. Examples of ordered point sets to be classified into given shape classes. The cardinality of these point sets decreases from left to right making the classification more challenging.

all of them belonging to the boundary, the task of shape (class) determination, that is going from (c) to (d), is still challenging, although not as difficult as going from (a) to (d). Depending upon where the primitives are placed on the curve, the resulting polygons can have very different shapes. To reach a statistical framework for this classification, we have to develop models for the variabilities associated with shapes, the generation of primitives (*i.e.* sampling in the case of points), and the observation noise.

Given these challenges, we will address the general problem in two steps. First, we will study the classification problem in the absence of clutter and assuming a known ordering. Then, we will extend that solution to the more general case with clutter and an unknown ordering.

Problem I—Baseline Problem: we assume that all the observed points belong to the boundary of interest and that an ordering of these points is known. Thus, the goal is to develop a statistical framework to classify *an ordered set of primitives* into pre-determined shape classes. Some examples of ordered point sets are shown in Figure 2. Given shape classes, such as crown, glass, bottle, carriage, etc., we seek to classify the observed points, or polygons, into these classes. In the figure, the number of points is high on the left and decreases towards the right. For any observer it will be relatively easier to classify the polygons on the left than those on the right.

Problem II: Extension to General Problem: in this more general case, not only do we not know the ordering of the points generated by the object boundary, but clutter points are generated by the background and the object interior are also present. We do not know how many or which of the data points fall on the boundary.

For the experiments described in this paper, we will utilize one of the Kimia databases (see for example [6]) consisting of 16 classes of shapes: bone, bird, bottle, brick, cat, carriage, car, chopper, crown, fountain, man, rat, fork, tool, fish, and glass, with approximately 400 total training shapes. Figure 9 shows the mean shapes from these 16 shape classes.

In the past literature, the search for parametric shape models (lines, circles, cylinders, etc) in cluttered data has been performed using the RANSAC algorithm [7], [8]. However, the multiplicity of shape classes and the non-parametric nature of shape variability makes it difficult to apply RANSAC in this context. Also, note that the goal here is different from reconstruction of curves from point cloud data. A related problem is the shape analysis of objects, most commonly 3D, using discrete representations of their surfaces, for instance using point clouds as in Memoli and Shapiro [9]. Similarly, Glaunes et al. [10] represent curves and surfaces as measures in $\mathbb{R}^n$ and compare the shapes by comparing their associated measures. Although such solutions, proposed for comparing point clouds to point clouds can also be applied to the current problem, the presence of clutter is a problem. Peter and Rangarajan [12] impose a very different structure, originating from a mixture of Gaussian, to analyze shapes of point clouds. Felzenszwalb and Schwartz [11] propose a hierarchical, tree-like representation of curves using a triplet of points at each node and compare the trees by comparing shapes of the triangles formed by triplets. The specific problem of classifying shapes of 2D contours using cluttered points provides an additional structure, coming from variability in shapes and their samplings into finite points, that is not exploited by some of these general methods.

1.2 Problem formulation and overview

The classification problem is described by the probability $P(C|\mathbf{y})$, where $C \in \mathcal{C}$ is the class of the object represented by the data set, and $\mathbf{y} \subset \mathcal{Y}$ is the data, *i.e.* a finite set of primitives. (Because we are restricting attention to primitives that are simply points in $\mathbb{R}^2$, we have $\mathcal{Y} = \mathbb{R}^{2m}$ for m primitives.) We fix an arbitrary enumeration of these points for convenience. Classification can then be performed by maximizing the probability: $\hat{C} = \text{argmax}_C\, P(C|\mathbf{y})$. The construction of $P(C|\mathbf{y})$ is most easily performed by first rewriting it using Bayes' theorem: $P(C|\mathbf{y}) \propto P(\mathbf{y}|C)P(C)$.

In what follows, we will take the prior probability over classes to be uniform, but including a non-uniform prior is trivial. The difficulty of the problem is contained in $P(\mathbf{y}|C)$, which describes the formation of the data starting from the object class. To make any further progress, this probability must be broken down into components corresponding to simpler stages in the data formation process. Here we will provide a schematic overview of these stages, and the algorithm to which they give rise. The various quantities used below will be defined precisely in the following sections. First, we introduce some variables:

- Let $g \in \mathcal{G}$, where $\mathcal{G} \equiv (SO(2) \ltimes \mathbb{R}^2) \times \mathbb{R}_+$, be a similarity transformation that includes rotation, translation, and scale. The symbol $\ltimes$ denotes the semi-direct product.

- Let $q \in \mathcal{Q}$ be a shape, *i.e.* an object boundary modulo similarity transformations and reparametrizations. Thus, a specific boundary is given by gq.
- Let $s \in \mathcal{S}$ represent the generation of n point-primitives on the shape boundary; among other variables s contains n. We will call this a "sampling". Then qs will be a set of n point primitives modulo a similarity transformation, while a specific set of point primitives is given by $\mathbf{x} = gqs$.
- Let $\mathcal{I} \ni \iota : [0, \ldots n] \rightarrow [0, \ldots m]$ be a one-to-one map, i.e. an injection, for relating each element of $\mathbf{x}$ to a unique element of $\mathbf{y}$.

Then we can write (making certain independence assumptions, to be discussed later)

$$P(\mathbf{y}|C) = \sum_{\iota \in \mathcal{I}} \iiint_{\substack{g \in \mathcal{G} \\ s \in \mathcal{S} \\ q \in \mathcal{Q}}} P(\mathbf{y}|\iota, gqs)\ P(\iota|s)\ P(g|q, C) \times P(s|q, C)\ P(q|C) dg\ ds\ dq\ . \quad (1)$$

In this paper, we will take $P(\iota|s)$ and $P(g|q, C)$ to be uniform, a point we discuss in Section 3.3. With these assumptions, g and ι appear solely in the first factor in the integrand, $P(\mathbf{y}|\iota, gqs)$.

The difficulties of the problem can now be seen in mathematical terms. In order to compute the posterior probability of a class, one must in some way (at least approximately) sum over all possible injections ι, corresponding to the first and second challenges; and integrate over all possible transformations g, samplings s, and shapes q, corresponding to the third challenge. The simplified baseline problem, Problem I, corresponds to knowing ι ($P(\iota) = \delta(\iota, \iota_0)$), so that the sum over it trivializes. Note, however, that $P(\mathbf{y}|\iota_0, gqs)$ still must model observation noise.

Our algorithmic strategy for dealing with this great complexity is based on two approximate methods for evaluating the integrals and sums: Monte Carlo integration and the saddle point approximation (also called the Laplace's method). We use the first for the integrals over q and s, generating realizations from their probability distributions and then summing the values of the integrand evaluated at these realizations. We use the second for the integral over g and the sum over ι. For Problem I, the latter is trivial, and so the maximization problem reduces to a Procrustes alignment of two 2D point sets under the likelihood $P(\mathbf{y}|\iota_0, gqs)$ describing the observation noise, which we take to be white and Gaussian. For Problem II, we have also to find the best injection ι in addition to the best transformation g. Using a combination of the Hungarian algorithm and the Procrustes alignment, we solve the joint registration-transformation problem.The cost function for this optimization is the likelihood $P(\mathbf{y}|\iota, gqs)$, which must now include a stochastic model of the clutter points. The result of these procedures is an approximation to the value of $P(\mathbf{y}|C)$ for each value of C, *i.e.* each class, and thus, after a trivial normalization, to the value of $P(C|\mathbf{y})$. Classification is then immediate.

To construct a fully statistical framework, then, we have to develop probability models and computational methods for the variability in shape ($P(q|C)$), sampling ($P(s|q, C)$), and observation noise and clutter ($P(\mathbf{y}|\iota, gqs)$). We now discuss each of these in more detail, beginning with sampling, since our approach here is novel.

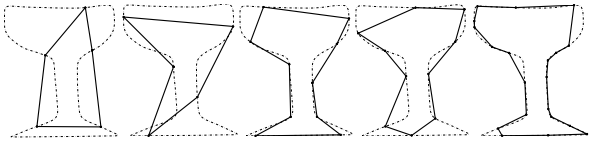


Fig. 3. Illustration of sampling variability for a curve.

2 Modeling Sampling Variability

By a sampling of a continuous curve, we mean selecting an ordered finite number of points on that curve. (We underline the distinction between our use of "sampling a continuous curve" and the phrase "sampling from a probability". To avoid this confusion, we will use "simulating from a probability" for the latter.) The sampling step results in a significant loss of information about the original shape. Figure 3 shows some examples of samplings of a single shape. Since the sampled points are ordered, we can draw a polygon for improving the visualization of the sampled points.

A sampling is intended to represent the generation of primitives by particular types of sensor, or, more commonly, by simple image processing techniques such as edge detection. As such, it is heavily dependent on the procedure used to generate the primitives. To avoid a presumption on the image processing technique, we must treat the generation of these primitives in a generic probabilistic way. Before we can go on to describe the probability distribution $P(s|q, C)$, however, we have to specify on what space it will be defined.

2.1 Representation

How can we mathematically represent a sampling? The process of sampling, by itself, is seldom studied in the literature, although the related problem of matching sampled shapes has received a lot of attention, see e.g. [6]. A sampling involves two elements: a certain number of points, n, and their placement on the curve. The latter can be expressed by parameterizing the curve in terms of its arc length, and then selecting n values in the interval $[0, L]$, where L is the length of the curve. Since we will be sampling the points from shapes, we can assume that $L = 1$. Note that this assumes that the probability of a sampling does not depend on the position, orientation, and scale of a curve, which was implicitly written into Eqn. 1.

If we know n, then sampling a curve amounts to partitioning a circle into n subintervals. This process simplifies if we place the origin on the curve at the position of the first sample, and thereby consider the sampling problem as that of partitioning the unit interval $[0, 1]$ into n subintervals. The position of the origin now becomes an element of the representation: we will denote it by τ. Any partition of $[0, 1]$ by n points can be identified with a probability mass function with n elements. Therefore, if n is fixed, one can represent a sampling as a

point in the $(n-1)$-simplex $\Delta^{(n-1)}$. However, for unknown n, one would like to allow all possibilities in a model and this motivates a broader representation. In particular, one would like there to be some consistency between the probabilities of samplings with different numbers of points, which suggests separating the choice of number of points and their placement. This can be achieved as follows.

Let Γ be the set of increasing, differentiable functions from $[0,1]$ to itself, such that for all $\gamma \in \Gamma$, $\gamma(0) = 0$ and $\gamma(1) = 1$, or, in other words, the group Γ of positive diffeomorphisms of the unit interval. Now let $U = [0 \ldots n]/n$ be a uniform partition of the interval $[0,1]$ into n sub-intervals. Then any element of $\Delta^{(n-1)}$ can be represented by $\gamma(U)$, for some $\gamma \in \Gamma$. In fact, there are an infinity of elements γ all of which give rise to the same point in $\Delta^{(n-1)}$. A sampling s will thus be represented by an equivalence class of triples $\langle n, \tau, \gamma \rangle \in \mathbb{N} \times \mathbb{S}^1 \times \Gamma$. The advantage of this representation is that we can change n without changing γ, and vice-versa.

We still have to decide, however, how to represent γ. The functions in Γ can be thought of as cumulative distribution functions for probability densities on $[0,1]$, with which they are in bijective correspondence, and this gives rise to a number of possibilities for representing such functions:

Diffeomorphism: An element of Γ is represented as itself, *i.e.* as an increasing function from $[0,1]$ to itself, such that $\gamma(0) = 0$ and $\gamma(1) = 1$. The advantage of this representation is that the action of the group of diffeomorphisms on itself is particularly simple, by composition.

Probability density: An element of Γ is represented by its derivative, denoted $\mathcal{P} \ni p = \dot{\gamma}$, which is an everywhere positive probability density on $[0,1]$, *i.e.* a positive function that integrates to 1.

Log probability: An element of Γ is represented by the logarithm of a probability density, $\mathcal{N} \ni \nu = \ln(p)$. It is an arbitrary function whose exponential integrates to 1. The advantage of this representation is that the values of the function ν are unconstrained apart from the normalization.

Square-Root Form: An element of Γ is represented by the square root of a probability density, $\Psi \ni \psi = p^{\frac{1}{2}}$. This is a positive function whose square integrates to 1, *i.e.* its $\mathbb{L}^2$ norm is 1. The set of these functions thus forms the positive orthant of the unit sphere in the space $\mathbb{L}^2([0,1])$. The advantage of this representation is that it greatly simplifies the form of the most natural Riemannian metric one can place on Γ, as we will now discuss.

2.2 Riemannian Structure on Γ

We wish to construct probability distributions on Γ, perform inferences, compute statistics, and so on. The difficulty is in performing calculus on this space while maintaining the underlying nonlinear constraints on the functions involved. A natural solution is to work on the nonlinear manifold formed by these functions and to utilize the intrinsic geometry of this manifold to perform statistics. This requires computing geodesic paths between points on the manifold, which in turn requires a Riemannian structure. We must thus make a choice of Riemannian metric, as well as a choice of one of the above representations in which to express it.

Fortunately, while there are clearly a large number of Riemannian metrics one could place on Γ, one is selected uniquely by invariance requirements, as follows. It is a remarkable fact, proved by Čencov [13], that on spaces of probability distributions on finite sets, there is a unique Riemannian metric on the space of probability distributions that is invariant to probabilistic mappings. This Riemannian metric is the so called *Fisher-Rao* (F-R) metric. (In finite dimensions, it has been used previously in computer vision [12], [14].) The F-R metric extends naturally to the space of probability measures on continuous spaces such as $[0,1]$, where it is invariant to the (re-parameterization) action of the diffeomorphism group. Since Γ is isomorphic to $\mathcal{P}$, we can view the F-R metric as a metric on Γ too. Because of its invariance properties, this is the metric we choose to use. In terms of the probability density representation, it takes the following form: the inner product between tangent vectors δp and $\delta' p$ to the space of probability distributions on $[0,1]$ (here tangent vectors are functions that integrate to zero) at the point $p \in \mathcal{P}$ is $\langle \delta p, \delta' p \rangle_p = \int_0^1 \delta p(s)\, \delta' p(s)\, \frac{1}{p(s)}\, ds$. It turns out, however, that the F-R metric simplifies greatly under the half-density representation. Indeed, it becomes $\mathbb{L}^2$, because $\psi^2 = p$ means that $2\psi\delta\psi = \delta p$, and thus that $\langle \delta\psi, \delta'\psi \rangle_\psi = \int_0^1 \delta\psi(s)\, \delta'\psi(s)\, ds$. We have already seen that Ψ is the positive orthant of the unit sphere in $\mathbb{L}^2([0,1])$, and now we see that the F-R metric is simply the $\mathbb{L}^2$ Riemannian metric on $\mathbb{L}^2([0,1])$ restricted to Ψ. The space Ψ endowed with the F-R metric is thus the positive orthant of the unit sphere in $\mathbb{L}^2([0,1])$ with the induced Riemannian metric.

As a consequence of this analysis, geodesics under the F-R metric are nothing but great circles on this sphere, while geodesic lengths are simply the lengths of shortest arcs on the sphere. Arc-length distance on a unit sphere has been used to measure divergences between probability density functions for a long time [15]. This metric also plays an important role in information geometry as developed by Amari [16].

We now prove the invariance property of the F-R metric. This is important because using this metric, the probability model that we construct on the space of sampling functions will be invariant to reparameterizations of curves in a shape class.

Theorem. *The Fisher-Rao metric is invariant to the action of* Γ.

Proof: We show this using the square-root form but the proof is similar for the other representations. The action of Γ on Ψ is easily deduced from its action on Γ by composition: $(\gamma^*\psi)(s) = \dot{\gamma}^{\frac{1}{2}}(s)\psi(\gamma(s))$. This is linear, and so the action on tangent vectors is analogous: $(\gamma^*\delta\psi)(s) = \dot{\gamma}^{\frac{1}{2}}(s)\delta\psi(\gamma(s))$. Therefore, the inner product $\langle \gamma^*\delta\psi, \gamma^*\delta'\psi \rangle_{\gamma^*\psi}$ becomes

$$\int_0^1 (\gamma^*\delta\psi)(s)\, (\gamma^*\delta'\psi)(s)\, ds = \int_0^1 \dot{\gamma}(s)\delta\psi(\gamma(s))\, \delta'\psi(\gamma(s))\, ds$$
$$= \int_0^1 \delta\psi(t)\, \delta'\psi(t)\, dt = \langle \delta\psi, \delta'\psi \rangle_\psi \ . \square$$

2.2.1 Geodesic, exponential maps, etc

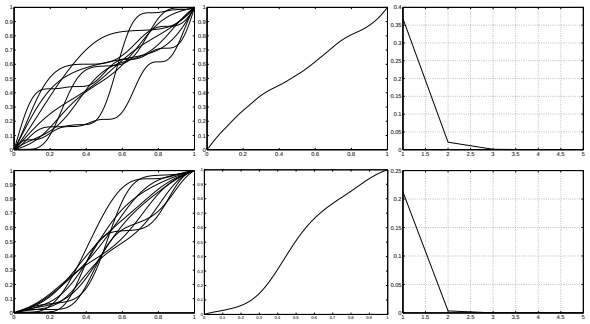
In this section we list some analytical expressions that are useful in a statistical analysis on Ψ and, thus on Γ. As Ψ is an infinite-dimensional sphere inside $\mathbb{L}^2([0,1])$ (see e.g. Lang [17]), the length of the geodesic in Γ between any two functions γ_1 and γ_2 under the F-R metric is given by $d(\gamma_1,\gamma_2) = \cos^{-1}(\langle \dot{\gamma}_1^{\frac{1}{2}}, \dot{\gamma}_2^{\frac{1}{2}} \rangle)$, where the inner product is $\mathbb{L}^2$. The geodesic between two points γ_1 and γ_2 of Γ is similarly derived. For $\psi_i = \dot{\gamma}_i^{\frac{1}{2}}$, the corresponding geodesic in Ψ is given by $\psi(t) = \frac{1}{\sin(\theta)}[\sin((1-t)\theta)\psi_1 + \sin(t\theta)\psi_2]$, where $\cos(\theta) = \langle \psi_1, \psi_2 \rangle$. The desired geodesic in Γ is then given by $\gamma(t)$, where $\gamma(t)(s) = \int_0^s \psi(t)(\tau)^2 \, d\tau$. Due to this additional integration step, it is sometimes easier to perform the Riemannian analysis in Ψ and to map the final result back to Γ. This is especially true for computing means and variances of sampling functions, for constructing probability densities on Γ, and for simulating from these probability densities.

In Ψ, the geodesic starting from a point ψ, in the direction $v \in T_\psi(\Psi)$, can be written as: $\cos(t)\psi + \sin(t)\frac{v}{\|v\|}$ (with the $\mathbb{L}^2$ norm). As a result, the exponential map, $\exp : T_\psi(\Psi) \to \Psi$, has a very simple expression: $\exp_\psi(v) = \cos(\|v\|)\psi + \sin(\|v\|)\frac{v}{\|v\|}$. The exponential map is a bijection between a tangent space and the unit sphere if we restrict $\|v\|$ so that $\|v\| \in [0,\pi)$, but for large enough $\|v\|$, $\exp_\psi(v)$ will lie outside Ψ, *i.e.* ψ may take on negative values. We will discuss this further when we define prior probabilities on Γ. For any $\psi_1, \psi_2 \in \Psi$, we define $v \in T_{\psi_1}(\Psi)$ to be the inverse exponential of ψ_2 if $\exp_{\psi_1}(v) = \psi_2$; we will use the notation $\exp_{\psi_1}^{-1}(\psi_2) = v$. This can be computed using the following steps: $u = \psi_2 - \langle \psi_2, \psi_1 \rangle \psi_1, \quad v = u \cos^{-1}(\langle \psi_1, \psi_2 \rangle) / \langle u, u \rangle^{\frac{1}{2}}$.

2.3 Statistics on Γ

Consider the task of computing the statistical mean of a set of sampling functions $\{\gamma_1, \gamma_2, \ldots, \gamma_k\}$ intrinsically in Γ. As mentioned earlier, we will use the square-root forms of these functions to perform such calculations. Let the corresponding set of square-root forms be given by $\{\psi_1, \psi_2, \ldots, \psi_k\}$, $\psi_i = \dot{\gamma}_i^{\frac{1}{2}}$. We define their Karcher mean as: $\mu = \text{argmin}_{\psi \in \Psi} \sum_{i=1}^k d(\psi, \psi_i)^2$, where d is the geodesic distance on Ψ. The minimum value $\sum_{i=1}^k d(\mu, \psi_i)^2$ is called the Karcher variance of that set. The search for μ is performed using a gradient approach where an estimate is iteratively updated according to: $\mu \to \exp_\mu(\epsilon v), \; v = \frac{1}{k}\sum_{i=1}^k \exp_\mu^{-1}(\psi_i)$. Here, $\exp$ and $\exp^{-1}$ are as given in the previous section, and $\epsilon > 0$ is a small number. The gradient process is initialized to $\bar{\psi}/\sqrt{\langle \bar{\psi}, \bar{\psi} \rangle}$, where $\bar{\psi} = \frac{1}{k}\sum_i \psi_i$.

In Figure 4, we show two examples of computing Karcher mean. The column (a) shows examples of sampling functions $\gamma_1, \gamma_2, \ldots, \gamma_{10}$, and the column (b) shows their Karcher means μ_γ (the sampling function obtained by squared integration of $\mu \in \Psi$). We remark that one can extend this framework to define a full covariance structure on the tangent space $T_\mu(\Psi)$ (or equivalently $T_{\mu_\gamma}(\Gamma)$) by mapping the observed sampling functions to that tangent space [18].



(a) (b) (c)

Fig. 4. Examples of Karcher mean in Γ: In each case, (a) shows ten γ_i, (b) shows their Karcher mean μ_γ, and (c) shows the cost functions vs. iterations.

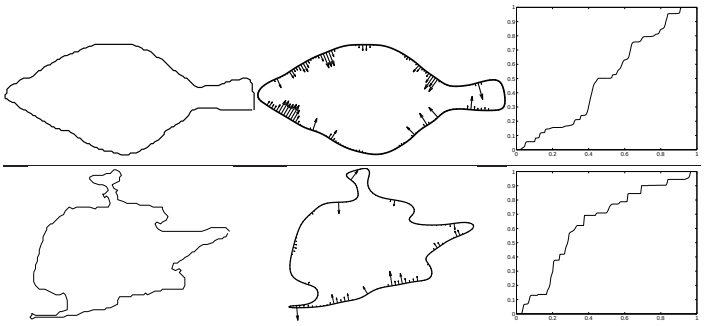
2.4 Probability distributions & Simulations

Having established a representation and a Riemannian metric on the space Γ of sampling functions, we now turn to the question of constructing a probability distribution. Recall that a sampling s is a triple $\langle n, \tau, \gamma \rangle \in \mathbb{N} \times \mathbb{S}^1 \times \Gamma$, and we have deliberately chosen our representation so that we can write the probability for s as $P(s|C) = P(n)P(\tau|C)P(\gamma|C,\tau)$ *i.e.* $P(n)$ does not depend on the shape; we will use a geometric distribution for n. The most interesting part of the distribution is the factor $P(\gamma|C,\tau)$. Clearly the possibilities here are enormous. We will restrict ourselves to "Gaussian" distributions of the form

$$P(\gamma|C,\tau) = Z^{-1} e^{-\frac{1}{2\sigma_s^2} d^2(\dot{\gamma}^{\frac{1}{2}}, \psi_0)}, \qquad (2)$$

where d is the geodesic distance under our chosen Riemannian metric, and where $\psi_0 = \dot{\gamma}_0^{\frac{1}{2}}$ is, in consequence, the mode of the distribution. We discuss two possibilities for γ_0 and σ_s.

The simplest possibility is to emphasize the samplings of a curve that are uniform with respect to its arc-length parametrization, independently of C, by choosing $\gamma_0(s) = s$, or equivalently $\psi_0 \equiv 1$. This case is simple. Alternatively, γ_0 may depend on local geometrical properties. e.g. have samplings whose density increases with increasing curvature of the underlying curve. One could define γ_0 in a way that depends on the shapes in C. Let $E(s) = \int_0^s \exp(-|\kappa(s')|/\rho) ds'$, where $\kappa(s)$ is the curvature of q at arc-length parameter point s and $\rho \in \mathbb{R}^+$ is a constant. (Note that to define E, an origin τ must be chosen and the resulting distribution will be dependent on this choice.) Then define $\gamma_q(s) = E(s)/E(1)$.
We wish to define a single γ_0 for each class C based on the γ_q values for that class. We do this based on training curves from that class, as follows. First we compute γ_q for each training curve, and then, using the techniques presented in Section 2.3, we compute the Karcher mean, which we use as γ_0, and σ_s^2 the Karcher variance for that class. For this computation, the placement of the origin is aligned for all curves in a class, so that the observed γ_q all use the same τ. (This alignment is performed during the computation of geodesics between shapes of curves; this shape analysis is summarized in the next section.) We now illustrate these ideas with some examples.



(a) (b) (c)

Fig. 5. Curvature-driven sampling: (a) a curve; (b) a smoothed version, with $e^{-|\kappa(s)/\rho|}$ displayed as a normal vector field; (c) γ_κ.

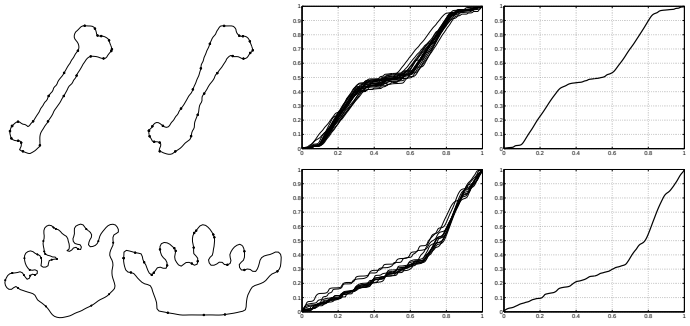


Fig. 6. Each row shows two examples of training curves in a class, the sampling functions γ_κ for that class, and their Karcher mean.

Shown in Figure 5, column (a) are two shapes q. We smooth these curves using Gaussian filters: their smoothed versions are shown in column (b). For these smoothed curves, we compute κ and then $E(s)$. This function is displayed as a normal vector field on the smoothed curve in (b). Finally, γ_q is computed; it is shown in column (c). Figure 6 shows some examples of class-specific means of the γ_q for two classes. By using these means as γ_0 for each class, we can form class-specific priors of the form given in Eqn. 2.

To simulate from probability densities of the form in Eqn. 2, we first randomly generate a function $f \in T_{\psi_0}(\Psi)$ such that $|f| = 1$, where, as before, $\psi_0 = \dot{\gamma}_0^{\frac{1}{2}}$. Then, we generate a normal random variable $x \sim N(0, \sigma_s^2)$, and compute a point $\psi = \cos(x)\psi_0 + \sin(x)f/\|f\|$. The random sampling function is then given by $\gamma(s) = \int_0^s \psi(s')^2\, ds'$. Figure 7 shows some examples of random simulations from such a class-specific prior density for increasing values of σ_s^2. If σ_s is too large, then many of the sampled points will lie outside Ψ, *i.e.* ψ will take on negative values. Including such samples still defines a probability density on Γ, but its interpretation is complex due to the "folding back" effect of taking the square of ψ. Such points may, however, simply be rejected from the samples, thus preserving the form of the density given above. For efficiency's sake, though, the proportion of such points should not be too large, and this implies a constraint on σ_s.

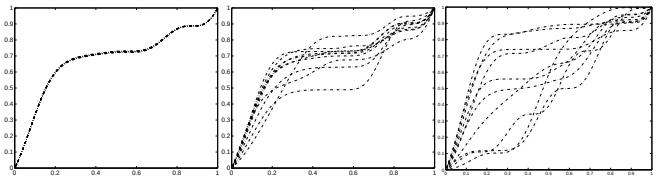


Fig. 7. Random samples from $P(\gamma|C)$ with σ_s^2 increasing from left to right.

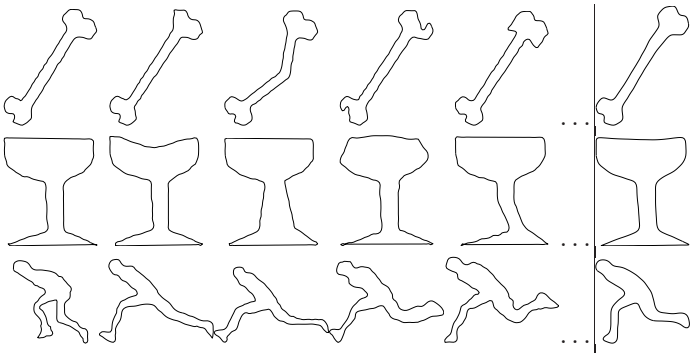


Fig. 8. Each row shows examples of the training shapes with their Karcher means shown in the rightmost panels.

3 Shape and Shape Variability

We now turn to the construction of the shape model, $P(q|C)$. While objects of a given class are similar in their shapes, there is naturally also variability within each class. It is this commonality and variability that $P(q|C)$ must describe. Figure 8 illustrates shape variability for three classes in the Kimia database.

There have been several recent papers that develop tools for analyzing the shapes of planar closed curves [3], [4] and others. The main differences amongst these articles lie in the choice of representation for the curves and of the metric used to compare shapes. An emerging choice of metric for comparing the shapes of curves is the *elastic metric* [19], under which curves are allowed to stretch, compress, and bend in order to reach an optimal matching. Although this metric has been studied in several forms, two recent papers [20], [21] present an efficient representation under which the elastic metric becomes a simple $\mathbb{L}^2$ metric, with the result that shape analysis simplifies considerably. This has been called the square-root elastic framework, and we describe it in the next section.

3.1 Representation

Consider a closed, parameterized curve: a differentiable mapping β from $\mathbb{S}^1$ to $\mathbb{R}^2$, and we want to analyze its shape. There are two invariances we have to include in our analysis. One is that the notion of "shape" is independent of the size, orientation, and position of the curve. Secondly, it is invariant to the reparameterizations of the curve. The variability generated by changing these variables can be written as actions of appropriate groups on the spaces of closed curves and, thus, can be "removed" from the representation using quotients.

Before we present shape analysis in more details, we consider an important question: Why do we use parameterized curves to represent boundaries or regions? It is possible to analyze the shapes of regions using representations that do not involve explicit parameterizations. For instance, one can use a level set of a function to represent a region [22], [23], or one can view a region as a subset of $\mathbb{R}^2$ and use set-theoretic metrics, *e.g.* the Hausdorff metric, to compare shapes [24]. Since there is no parametrization involved in these representations, one does not have to "remove" it in shape analysis. However, this becomes a disadvantage when the goal is to associate arbitrarily sampled points to given shape classes: it is simply more difficult to associate sampled points in these representations than by using an explicit parameterization. On the other hand, approaches that represent shapes by a small subset of points on the boundary selected *a priori*, *e.g.* active shape models [1], cannot introduce arbitrary samplings. Hence, the choice of parameterized curves for shape analysis of boundaries is important.

As described in [20], [21], we will represent a curve β by its square-root velocity function: $q : \mathbb{S}^1 \rightarrow \mathbb{R}^2$, where $q(t) = \frac{\dot{\beta}(t)}{|\dot{\beta}(t)|^{\frac{1}{2}}}$, $|\cdot|$ is the Euclidean norm in $\mathbb{R}^2$, and t is an arbitrary coordinate on $\mathbb{S}^1$. Note that the use of the derivative already eliminates translations. To eliminate scalings, we restrict ourselves to the space of unit length closed curves. The resulting space is a unit sphere $\mathcal{B} = \{q | \int_{\mathbb{S}^1} (q(t) \cdot q(t))\, dt = 1\}$, where $(\cdot)$ is the Euclidean inner product in $\mathbb{R}^2$. The transformations that remain are rotations $SO(2)$ and reparameterizations $\mathrm{Diff}(\mathbb{S}^1)$. Since the actions of these two groups on $\mathcal{B}$ are isometric, with respect to the $\mathbb{L}^2$ metric, we can define the shape space to be the quotient space $\mathcal{Q} = \mathcal{B}/(\mathrm{SO}(2) \times \mathrm{Diff}(\mathbb{S}^1))$ and inherit the $\mathbb{L}^2$ metric from $\mathcal{B}$. In other words, for a point $q \in \mathcal{Q}$ the Riemannian metric takes the form $\langle \delta q_1, \delta q_2 \rangle_q = \int_{\mathbb{S}^1} \delta q_1(t) \cdot \delta q_2(t)\, dt$. To perform statistical analysis in $\mathcal{Q}$, however, which is our goal, one needs to construct geodesics in $\mathcal{Q}$. Joshi *et al*. [21] describe a gradient-based technique for computing geodesics in $\mathcal{Q}$. The technique uses path-straightening flows: a given pair of shapes is first connected by an initial, arbitrary path that is then iteratively "straightened" so as to minimize its length [20]. The length of the resulting path is then the geodesic distance between the shapes. Since one of the effects of $\mathrm{Diff}(\mathbb{S}^1)$ is different placements of the origin on closed curves, its removal results in an alignment of shapes in that regard.

3.2 Statistics and Probabilities on $\mathcal{Q}$

One can define and compute the mean of a collection of shapes using the Karcher mean, now based on the geodesic distance defined in the previous section [18]. Three sets of examples of shapes and their Karcher means are shown in Figure 8, while the Karcher means for all the 16 classes used in this paper are displayed in Figure 9. Figure 10 shows a dendrogram clustering of these mean shapes using the geodesic distance. We make two observations from this clustering. Firstly, this clustering agrees with our human inference in that similar shapes have been clustered together. Secondly, later on when we study classification of shapes, we anticipate that the algorithms will have more difficulty separating similar classes. For example, classes 1 and 15 — bones and tools – will be harder to distinguish than say bones and glasses.

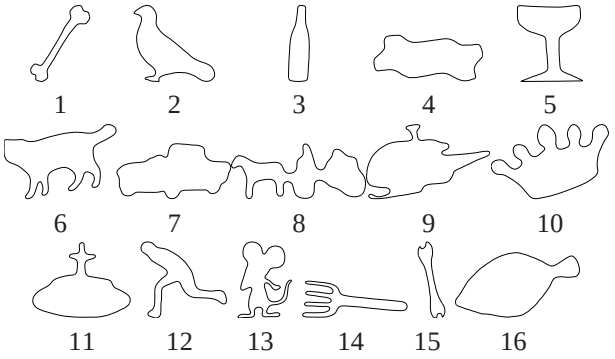


Fig. 9. Karcher means of the 16 shape classes used.

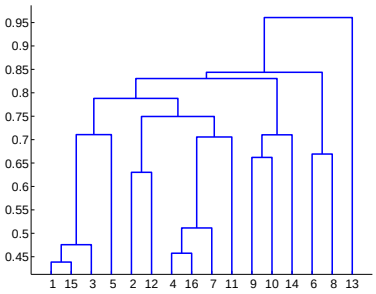


Fig. 10. A dendrogram plot of the Karcher means in Figure 9 using geodesic distances.

The next step is to impose a probability model on $\mathcal{Q}$. Perhaps the simplest model is the one used for Γ, Eqn. 2. As was suggested in [18], it is much easier to express this distribution using the tangent space $T_{q_0}\mathcal{Q}$ to $\mathcal{Q}$ at the mean shape q_0 than using $\mathcal{Q}$ itself, because the former is a vector space. In that space, one can use the principal component analysis (PCA) and impose a standard Gaussian distribution on the PCA coefficients, and use the exponential map to "push forward" these tangent vectors to $\mathcal{Q}$ itself. Empirical study shows, however, that the histograms of these tangent principal coefficients are often far from Gaussian. We therefore use kernel estimates of the underlying densities to capture this more complex behavior. This is illustrated in Figure 11. The essential methodology is unaltered, and indeed applies to any distribution on $\mathcal{Q}$ that we can simulate. For the purpose of simulating from this model, we treat the tangent principal coefficients as independent random variables. In practice we use approximately 10 tangent principal coefficients per shape class.

To simulate from $P(q|C)$ described above, we first simulate from the estimated density of the tangent principal coefficients, and then use the exponential map to generate corresponding elements of $\mathcal{Q}$. Figure 12 shows some examples of simulations from one such non-parameteric model.

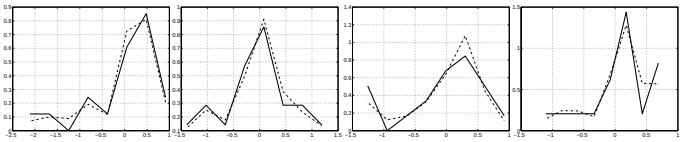


Fig. 11. Empirical distributions (solid lines) and kernel estimates (broken lines) of densities of four tangent principal coefficients in shape class 1.

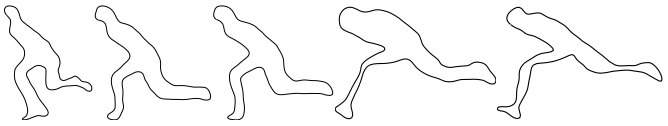


Fig. 12. Some randomly generated shapes from a TPCA model on a shape class.

3.3 Probability distribution for $\mathcal{G}$

We have described a representation for shapes $q \in \mathcal{Q}$, and some possible models $P(q|C)$. In order to describe a set of points with a particular position, orientation, and scale, however, we have to transform the q using a similarity transformation $g \in \mathcal{G}$ and then sample it. $P(g|q, C)$ is the corresponding probability distribution. In this paper, we will assume a uniform prior on $\mathcal{G}$, suitably truncated for large enough scales or translations to allow normalization.

4 Observation Model

Depending upon the technique used to extract primitives from the image data, the actual observations will often differ from the corresponding points on the curves. This may be due to low quality, coarse resolution, and quantization of images. A standard way to treat this variability is to introduce an independent observation noise that perturbs the sampled points according to some probability model. In this paper, we take this noise to be additive, white, and Gaussian, but the use of Gaussian noise is purely for convenience; more sophisticated noise models can similarly be included in the solution. The deterioration of data due to obscuration of shapes is not included in the observation noise.

In addition to the perturbation of the primitives generated by the object boundary, we expect to have primitives from the background and the object interior, creating "clutter". Our likelihood term needs to model these points as well. So, given n unperturbed points $\mathbf{x} = gqs$ generated by the curve, what is the probability of a given dataset of m points $\mathbf{y}$ ($m \geq n$)? If we know the injection ι relating $\mathbf{x}$ to n unique elements of $\mathbf{y}$, then we can divide $\mathbf{y}$ in two sets: a set of n points, named $\mathbf{y}_s$, related to $\mathbf{x}$ and the remaining $m - n$ points, named $\mathbf{y}_c$, attributed to clutter. The first set is modeled using additive, white-Gaussian noise and the second is modeled using a homogeneous Poisson process with intensity λ_b. The likelihood function for the complete data is given by:

$$\begin{aligned} P(\mathbf{y}|\iota, gqs) &= P(\mathbf{y}_s|gqs)\, P(\mathbf{y}_c) \\ &= \frac{1}{Z} e^{\frac{-1}{2\sigma_y^2}\sum_{k=1}^{n} \|\mathbf{y}_{\iota(k)} - \mathbf{x}_k\|^2} \frac{\lambda_b^{m-n}}{(m-n)!} \ . \end{aligned} \quad (3)$$

The probability $P(\mathbf{y}_c)$ thus depends solely on $(m - n)$. Note this likelihood also applies Problem I, except there $\mathbf{y} = \mathbf{y}_s$ and the likelihood consists only of the first term $P(\mathbf{y}_s|gqs)$.

5 Problem I Solution

For Problem I, n is fixed to be the number of points in $\mathbf{y}$, and s is reduced to the pair (τ, γ). In terms of Figure 1, our task is to go from (c) to (d). So we take up the problem of evaluating the posterior $P(C_i|\mathbf{y})$ and note that the Bayes' integral in Eqn. 1 is too complicated to solve analytically. It is therefore approximated using numerical techniques. There are several ways of approximating such an integral.

One possibility is to use the Laplace's approximation by maximizing the integrand over the variables of integration: $P(C_i|\mathbf{y}) \approx \frac{P_0(C_i)}{P(\mathbf{y})} P(\mathbf{y}|\iota_0, g_i^* q_i^* s_i^*) P(q_i^*|C_i) P(g_i^*|C_i) P(s_i^*|C_i)$, where (g_i^*, q_i^*, s_i^*) are the maximizers of the function $P(\mathbf{y}|\iota_0, gqs) P(q|C_i) P(g|C_i) P(s|C_i)$. Such an approximation is reasonable when the integrand has a single mode with a support that remains similar from class to class.

A more classical approximation is the Monte Carlo approach where one independently simulates values from the prior probabilities, evaluates the likelihood function and averages the likelihoods to estimate the required posterior. That is, generate $q_j \sim P(q|C_i)$, $g_j \sim P(g|C_i)$ and $s_j \sim P(s|C_i)$, for $j = 1, 2, \ldots, J$ independently and form the Monte Carlo estimate: $P(C_i|\mathbf{y}) \approx \frac{P_0(C_i) \sum_{j=1}^{J} P(\mathbf{y}|\iota_0, g_j q_j s_j)}{\sum_i P_0(C_i)(\sum_{j=1}^{J} P(\mathbf{y}|\iota_0, g_j q_j s_j))}$.

Sometimes it is more efficient to use a combination of these two ideas. For instance, since the use of white Gaussian observation noise leads to a quadratic likelihood energy, the optimal value of g for matching a $\mathbf{y}$ to an $\mathbf{x} = gqs$ can be found easily using the standard point registration. Similarly, of the two variables making up s – (τ and γ) – one can also optimize over τ while randomly simulating γ from the prior $P(\gamma|C_i)$. Since τ decides which element of the circular set $\mathbf{x}$ is the starting point, there are only n possibilities and they can be searched exhaustively. Thus, it is easier to remove g and τ from the integration using optimization. Let q_j and γ_j be the simulated values from $P(q|C_i)$ and $P(\gamma|C_i)$, and let

$$(g_j^*, \tau_j^*) = \operatorname*{argmax}_{g,\tau} P(\mathbf{y}|\iota_0, g q_j s_j), \quad s_j = (\tau, \gamma_j) \ . \quad (4)$$

Define a point set $\mathbf{x}_{j,i}^*$ to be the one resulting from taking the shape q_j, sampling function γ_j, registration τ_j^*, and the alignment g_j^*, all generated from models for class C_i. Then, an estimate of the posterior is given by

$$P(C_i|\mathbf{y}) \approx \frac{P_0(C_i) \sum_{j=1}^{J} P(\mathbf{y}|\mathbf{x}_{j,i}^*)}{\sum_i P_0(C_i)(\sum_{j=1}^{J} P(\mathbf{y}|\mathbf{x}_{j,i}^*))}. \quad (5)$$

Here, the likelihood is given by the first term in Eqn. 3.

5.1 Joint Registration And Alignment

The subproblem we address here is given in Eqn. 4: Given two sets of ordered points in $\mathbb{R}^2$, call them $\mathbf{x}$, $\mathbf{y} \in \mathbb{R}^{n \times 2}$, we want to rotate, scale, translate, and circularly shift $\mathbf{x}$ so as to minimize its Euclidean distance squared from $\mathbf{y}$. Define $\mathbf{x}^\tau$ to

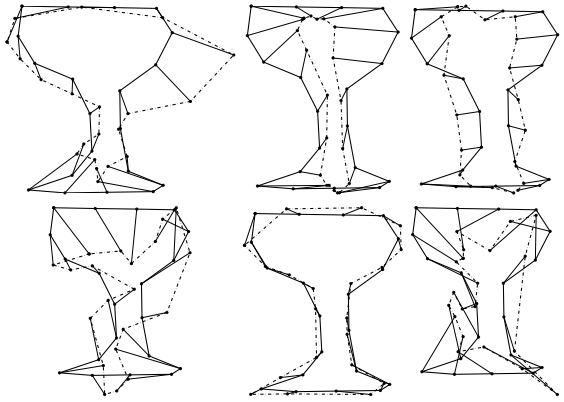


Fig. 13. Examples of several $\mathbf{x}^*$ (broken line), corresponding to different shape classes, for the same $\mathbf{y}$(solid line).

be a circular shift of the elements of $\mathbf{x}$ such that τ^{th} element becomes the first element now, $\tau \in \{1, 2, \ldots, n\}$.

If τ is fixed, then the two sets of points are completely registered and their alignment is performed using the Procrustes method as follows. Compute the 2×2 matrix $A = (\mathbf{y} - \bar{\mathbf{y}})^T(\mathbf{x}^\tau - \bar{\mathbf{x}}^\tau)^T$, where $\bar{\mathbf{y}}$ and $\bar{\mathbf{x}}^\tau$ are means of $\mathbf{y}$ and $\mathbf{x}^\tau$, respectively. Let $A = U\Sigma V^T$, the SVD of A. The optimal rotation, scaling, and translation of $\mathbf{x}^\tau$ are given by:

$$O^* = \begin{cases} UV^T & \text{if } \det(A) > 0 \\ U \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} V^T & \text{otherwise.} \end{cases}$$

$$\rho^* = \frac{\mathrm{Tr}((\mathbf{y} - \bar{\mathbf{y}})^T(\mathbf{x}^\tau - \bar{\mathbf{x}}^\tau))}{\mathrm{Tr}((\mathbf{x}^\tau - \bar{\mathbf{x}}^\tau)^T(\mathbf{x}^\tau - \bar{\mathbf{x}}^\tau))}, T^* = \bar{\mathbf{y}} - \bar{\mathbf{x}}^\tau .$$

The search for optimal τ is exhaustive. That is, for each possible shift, and there are n such possibilities, we compute the best alignment of the resulting $\mathbf{x}$ to $\mathbf{y}$ and keep the closest one. This is the optimal $\mathbf{x}^*$ for the given pair $(\mathbf{x}, \mathbf{y})$. Some examples of this registration/alignment process are displayed in Figure 13.

Here is a summary of steps needed to approximate the posterior $P(C_i|\mathbf{y})$ for a given $\mathbf{y}$.

Algorithm 1. *For $j = 1, 2, \ldots, J$:*

1) *Randomly generate a shape class C_i and simulate a shape $q_j \sim P(q|C_i)$.*
2) *Generate a sampling function $\gamma_j \sim P(\gamma|C_i)$.*
3) *Solve for g_j^*, τ_j^*, and then $\mathbf{x}_j^*$ using the Procrustes method.*
4) *Evaluate the likelihood function $P(\mathbf{y}|\mathbf{x}_{j,i}^*)$ using Eqn. 3.*

Approximate the posterior $P(C_i|\mathbf{y})$ using Eqn. 5.

The noise variance σ_y^2 is a free parameter here. Its value affects the shape of the posterior histogram but not the posterior mode.

5.2 Experimental Results

We now describe some experimental results on estimating $P(C_i|\mathbf{y})$. In this experiment, we simulate the data $\mathbf{y}$ according to the data model and apply Algorithm 1. Figure 14 presents six examples of computing posterior using simulated data under Problem I. In each block, the left panel shows the true underlying curve and the points sampled on it to form $\mathbf{y}$ (elements of $\mathbf{y}$ are joined to form a polygon). The middle panel shows a bar chart of the estimated posterior probability $P(C_i|\mathbf{y})$ for each of the 16 classes. The last panel shows the simulated configuration $\mathbf{x}^*$ (dotted line) that results in the maximum likelihood, along with the hypothesized curve q and the data $\mathbf{y}$ (solid lines). As these examples demonstrate, the algorithm is quite successful in generating high-likelihood candidates from the correct shape classes, even when $\mathbf{y}$ is generated for a relatively small value of n. Of all these nine cases, only the top row has the highest posterior for the incorrect classes. This is expected as $n = 3$ is clearly too small to distinguish shape classes.

Once the posterior $P(C_i|\mathbf{y})$ is approximated, it can be used for classifying $\mathbf{y}$ into a shape class. Since the data $\mathbf{y}$ here has been simulated with known shape classes, we can evaluate the algorithm's performance by comparing the estimated class with the true class. To estimate the posterior for each $\mathbf{y}$, we have used $J = 300$ realizations from the posterior, and to estimate probability of correct classification, we have used 150 runs (simulations of $\mathbf{y}$) for each value of n and σ_y. For these simulations, the underlying shape class is picked randomly with equal probability. The results are shown in the left panel of Figure 15 where the probability of correct classification is plotted versus n, for three different observation noise levels. The noise levels are: $\sigma_y = 0.01$, $\sigma_y = 0.025$, and $\sigma_y = 0.05$, expressed in terms of the arc-lengths of the curve. For example, $\sigma_y = 0.01$ implies that $\mathbf{y}$ was simulated by adding noise at standard deviation 0.01 times the length of the true curve to each component of $\mathbf{y}$. This plot suggests that, in case of low noise, the sampling of shapes by $n = 6$ points results in approximately 50% classification rate. To reach over 90%, one will need more than 20 points in this setting. Even at a very high noise level $\sigma_y = 0.05$, the algorithm can classify more than 45% of observations with only 15 points. If we use a k-nearest neighborhood classifier (kNN), with increasing k, we get the result shown in the middle panel of Figure 15. The right panel shows the classification performance for each class individually, for the case $n = 12$ and $\sigma_y = 0.01$. In this plot the classification performance was estimated by averaging over 100 simulations of $\mathbf{y}$ generated from only one class at a time. As the dendrogram in Figure 10 shows, shapes in classes 1, 3, and 15, and 4, 16, and 7 are quite similar, respectively, and this naturally affects the classification rate for these classes. Their classification rates increase drastically when we go from 1-NN to 3-NN classifier. For example, the classification rate for class 3 jumps from 0.64 to 0.97 and for class 15 from 0.62 to 1.0. This supports the argument that the classification is closely tied to distinctiveness of shapes across classes. Another interesting point is the low classification rate of classes in which shapes are more complicated – cat (6) and mouse (13). We believe this is because the shape variability within the class is more complex and the shape model used here does not completely capture this variability.

In terms of the computational cost, the time taken to estimate $P(C_i|\mathbf{y})$ for each $\mathbf{y}$ using Algorithm 1 is approx-

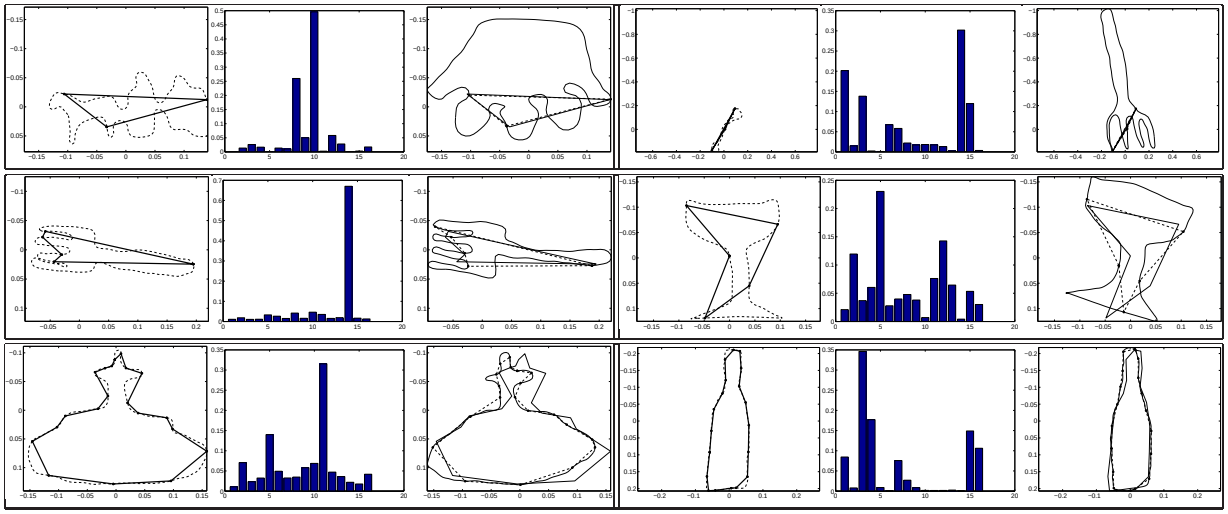


Fig. 14. Each block shows — Left panel: the data $\mathbf{y}$ (solid polygon) superimposed on the underlying true curve (broken line); Middle panel: the posterior $P(C_i|\mathbf{y})$; Right panel: highest likelihood sample $\mathbf{x}^*$ (broken polygon) drawn over the hypothesized curve β (solid line). Data polygon $\mathbf{y}$ is drawn in solid lines for comparison here. The top row has $n = 3$, the middle $n = 5$, and the bottom row $n = 20$. The numbering of classes in the bar chart is same as the order in Figure 9 and the correct classes (from top left to bottom right) are 8, 1, 14, 5, 11, and 3.

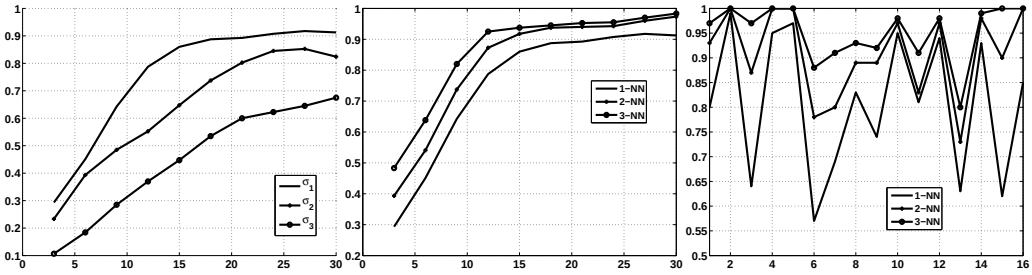


Fig. 15. Classification performance versus n. Left: Shape classification performance of nearest neighbor classifier for four different noise levels. Middle: Classification performances of one-, two-, and three-nearest neighbors classifiers, versus n, when $\sigma_y = 0$. Right: Classification performance by the class for $n = 12$ and $\sigma_y = 0$.

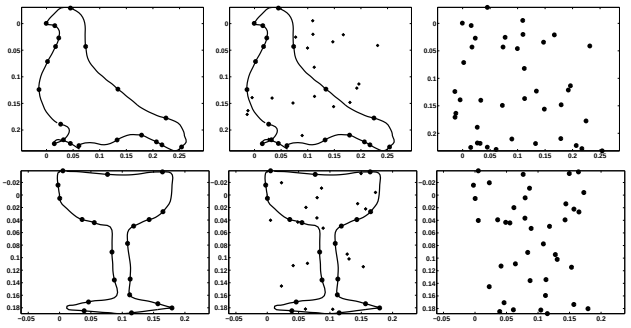
imately 20 seconds in MATLAB when $J = 300$. Since we estimate the probability of correct classification using 150 such evaluations of the classifier, for each value of σ_y and n, it takes approximately 50 minutes to estimate each point on the performance curves shown in Figure 15.

6 Problem II Solution

Now we return to the more general problem of finding shape classes in given point clouds, where the given points are: (i) unordered and (ii) may or may not lie on the object boundary. In terms of the problem description in Figure 1, our goal in this section is to go from the data (a) to the inference (d). Two sets of results are presented: one from the simulated data and one from primitives extracted from real images.

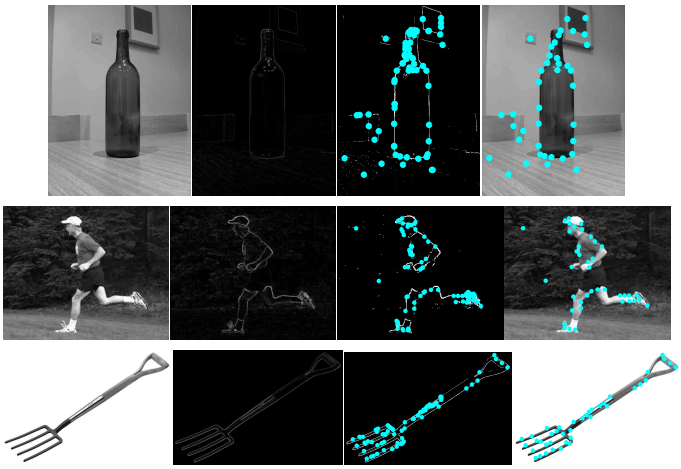
We start by describing the formation of the simulated data. As shown in Figure 16, we start by picking a class C_i, generating a shape $q \sim P(q|C_i)$ and sampling it according to a randomly generated sampling function $s = \langle n, \tau, \gamma \rangle$. Here $n \sim$ Geometric(n_0), τ is random in $\{1, 2, \ldots, n\}$, and $\gamma \sim P(\gamma|C_i)$. Next, we introduce additive, Gaussian noise to these sampled points. So far, the data formation is similar to the baseline problem studied earlier. Then we introduce background clutter by simulating from a homogeneous Poisson process with mean λ_b. The result is shown in panel (b) of this figure. Finally, we take all the points: sampled with noise from q and simulated from Poisson, and randomly permute them to result in the set $\mathbf{y}$ of observed data points, as shown in panel (c).

The second set of experiments in this section involves primitives derived from the image data using a simple processing step demonstrated in Figure 17. For an image I (left panel), we have used $I_w \equiv |\frac{\partial I}{\partial x}| + |\frac{\partial I}{\partial y}|$ to isolate (vertical and horizontal) edges in I (second panel). Then, we threshold I_w using three standard deviations from the mean value in I_w, to obtain a binary edge map (third panel). To obtain point primitives from the binary map, we randomly select a predetermined number, say m_0, from the points with value 1 (also shown in the third panel). Finally, we use a thinning procedure to discard $(m_0 - m)$ points to results in a set $\mathbf{y}$ of m points (last panel). This thinning basically computes all pairwise distances between points and iteratively discards those points that are



(a) (b) (c)

Fig. 16. Simulated data. (a) The original curve β and its sampling gqs, (b) with Poisson clutter, (c) the resulting $\mathbf{y}$.



(a) (b) (c) (d)

Fig. 17. Examples of pre-prcessing of images: (a) I, (b) I_w, (c) a random selection from binary image (d) thinning step to result in $\mathbf{y}$ drawn over I.

associated with the smallest distances. In the experimental results presented here we used $m_0 = 70$ and $m = 40$.

6.1 Registration Problem

The key step to handle Problem II is to solve a *registration problem*: given two sets of points $\mathbf{x} \in \mathbb{R}^{n\times 2}$ and $\mathbf{y} \in \mathbb{R}^{m\times 2}$, $n \leq m$, associate to each element of $\mathbf{x}$ a unique element of $\mathbf{y}$ so as to minimize a certain cost function. Using an injection $\iota : \{1 \dots n\} \mapsto \{1 \dots m\}$ each hypothesis point $\mathbf{x}_k$ has to be associated with a data point $\mathbf{y}_{\iota(k)}$. This results in a subset $\mathbf{y}_s$ of points that are assigned to the shape and a subset $\mathbf{y}_c$ of remaining points assigned to the background clutter. The likelihood energy function for this model is given by: $-\log(P(\mathbf{y}|\iota, gqs))$, where $P(\mathbf{y}|\iota, gqs)$ is given in Eqn. 3. Similar to the hybrid approach taken in Problem I, we would like to solve for the pair (g, ι) explicitly using:

$$(g^*, \iota^*) = \underset{g\in\mathcal{G},\iota\in\mathcal{I}}{\operatorname{argmin}} \left(\sum_{k=1}^{n} \|\mathbf{y}_{\iota(k)} - \mathbf{x}_k\|^2 \right), \quad \text{for } \mathbf{x} = gqs \,. \tag{7}$$

The minimization problem over ι, for a fixed g, is one version of the famous optimal assignment problem. The solution is given by the Kuhn-Munkres or the Hungarian algorithm and their MATLAB implementations are readily available. Hence, we do not reproduce that algorithm here but directly present our experimental results. Once the optimal mapping ι^* is found, it solves the two original issues: background rejection and point ordering. Note that the ordering of points in $\mathbf{x} = gqs$ is known and this ordering, in turn, imposes an ordering on the corresponding elements of $\mathbf{y}$. Shown in Figure 18 are some examples of registering a given $\mathbf{y}$, top-left panel, with several hypothesis of $\mathbf{x}$, shown in the remaining panels. For each hypothesis, we use the Hungarian algorithm to find optimal ι^* (for $m = 40$, $n = 20$) and an ordering on *automatically selected* elements of $\mathbf{y}$ (solid polygon) inherited from the corresponding elements of $\mathbf{x}$ (broken line polygon).

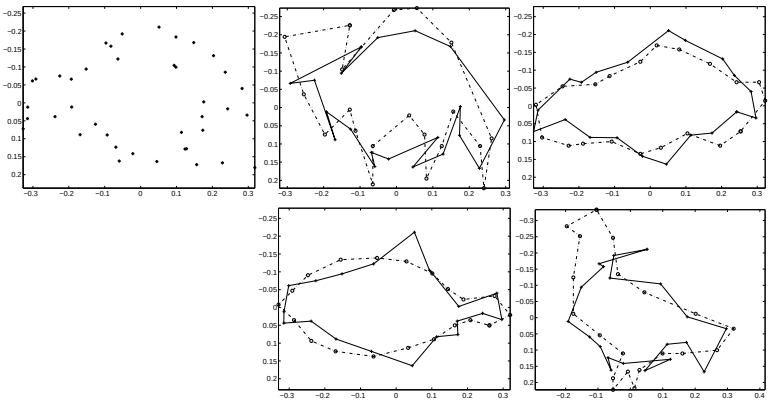


Fig. 18. Association Problem: For the data set $\mathbf{y}$ shown in top left, we show four examples of: $\mathbf{x}$ (thicker points), the selected $\mathbf{y}_s$, and an estimated ordering of $\mathbf{y}_s$ (solid lines) inherited from corresponding ordering in $\mathbf{x}$ (broken lines).

6.2 Joint Registration and Alignment

In addition to the registration ι^*, we also need to solve for the optimal transformations g^* in Eqn. 7. The transformation g consists of a rotation $O \in SO(2)$, scale $\rho \in \mathbb{R}_+$ and a translation $T \in \mathbb{R}^2$, as was the case in Problem I. For a fixed ι, we have a registration between elements of $\mathbf{x}$ and $\mathbf{y}_s$ and we can solve for the optimal transformation g^* directly (using Eqns. in Section 5.1).

Now we have a situation that is familiar to problems in registration/alignment of point clouds. For a given registration ι, we can solve for the optimal transformation and for a given transformation g we can solve for the optimal registration. However, we need a joint solution. This we accomplish by initializing a transformation of g and iterating back and forth between the two conditional optimizations. The result is a local solution to the joint optimization problem; we will label the final values of g and ι as g^* and ι^*, respectively. The initial value of T is taken to be $\bar{\mathbf{y}} - \bar{\mathbf{x}}$ while the initial rotations of $\mathbf{y}$ and $\mathbf{x}$ are obtained using the SVD of matrices $\sum_k (\mathbf{y}_k - \bar{\mathbf{y}})(\mathbf{y}_k - \bar{\mathbf{y}})^T$ and $\sum_k (\mathbf{x}_k - \bar{\mathbf{x}})(\mathbf{x}_k - \bar{\mathbf{x}})^T$. The scale ρ is initialized by scaling $\mathbf{x}$ and $\mathbf{y}$ in such a way that the Frobenious norm of $\mathbf{y}$ is $\sqrt{m/n}$ times the Frobenious norm of $\mathbf{x}$. The logic for this choice is that a subset of size n from

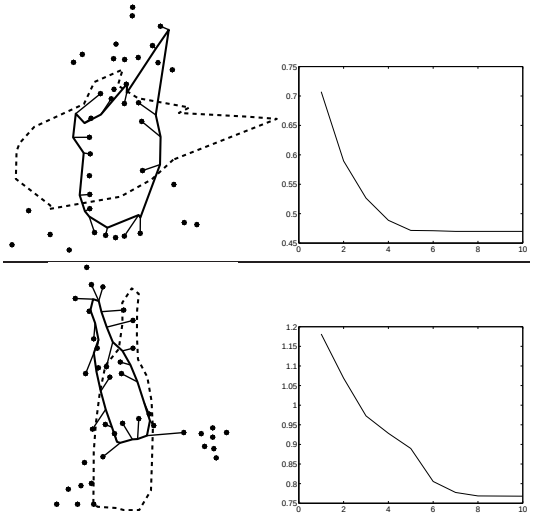


Fig. 19. Left panels show $\mathbf{y}$ (points) , $\mathbf{x}$ before (broken line) and $\mathbf{x}$ after (solid line) the joint registration and alignment. Right panels show the cost function in Eqn. 7.

$\mathbf{y}$, although we don't yet know which particular subset, has to be matched to $\mathbf{x}$. Two examples in Figure 19 illustrate this iterative optimization. Once the optimal association and transformation of $\mathbf{x}$ are found, we have the optimal version of the hypothesized configuration $\mathbf{x}^*$. Using a large number of simulated hypothesis, we can estimate the posterior using Eqn. 5.

Here is a summary of steps for approximating the posterior distribution in Problem II for a given $\mathbf{y}$.

Algorithm 2. *Same as Algorithm 1 except these two steps:*

2) *Generate a sampling function* $\gamma_j \sim P(\gamma|C_i)$ *and a sample size* $n \sim min(Geometric(n_0), m)$.
3) *Solve for* g_j^*, *and* ι_j^* *using Section 6.2. This gives rise to an optimal version of the hypothesis,* $\mathbf{x}_{j,i}^*$.

The parameters λ_b, σ_y and n_0 are free parameters here.

6.3 Experimental Results

Two sets of results, corresponding to the simulated data and the real image-based data, are presented here.

Shown in top three rows of Figure 20 are experimental results on the simulated data with $m = 40$ and $n_0 = 20$. In each case, the left panel shows the true underlying curve which was sampled to generate the data set $\mathbf{y}$ which are also shown there. The next panel displays a bar chart of estimated $P(C_i|\mathbf{y})$ for this $\mathbf{y}$, $i = 1, 2, \ldots, 16$ using $J = 300$ samples. The last figure shows a high probability polygon formed using the subsets $\mathbf{y}_s$ using Algorithm 2. In each of the three cases, the amount of clutter is quite high – the number of points on the curve equals the number of clutter points. Still, the algorithm puts the highest probability on the correct class for all cases. The bottom left chart is the estimated average performance of Algorithm 2 plotted against the ratio ν, where $\nu = \frac{\text{number of points on curve}}{\text{total number of points in } \mathbf{y}}$. Low values of ν denote a larger amount of clutter and the related classification performance is expectedly low. It is interesting to note that the performance of the nearest-neighbor classifier is more than 50% even when $\nu < .5$. As these experiments suggest, the algorithm is able to put high probability on the correct shape class despite the presence of clutter.

As a comparison, we have studied the performance of classification using the Hausdorff metric and the Iterated Closest Point (ICP) algorithm. In both cases hypothesis $\mathbf{x}$ are generated as earlier but the likelihood is computed differently. In the case of Hausdorff metric it is computed using $e^{-d_h(\mathbf{y},\hat{\mathbf{x}})^2}$, where d_h is the classifical Hausdorff distance, $d_h(\mathbf{y},\mathbf{x}) = \max_i(\min_j \|\mathbf{x}_i - \mathbf{y}_j\|)$ and $\hat{\mathbf{x}} = \text{argmin}_{O\mathbf{x}|O\in SO(2)}\, d_h(\mathbf{y}, O\mathbf{x})$. The scale and the translation of $\mathbf{x}$ is initialized as previously and kept fixed. The classification performance for this metric, for different levels of clutter, is shown in the right panel of Figure 20. Similarly, ICP algorithm is another commonly used procedure for registering and aligning arbitrary point clouds. The basic idea is to iterate between the Procrustes alignment and nearest-neighbor registration until convergence. We have used ICP to register elements of $\mathbf{x}$ to the elements of $\mathbf{y}$, resulting in $\tilde{\mathbf{x}}$, and use the resulting squared distance $d_{icp} = \sum_i(\min_j \|\tilde{\mathbf{x}}_i - \mathbf{y}_j\|)^2)$ to compute the likelihood $e^{-d_{icp}^2}$. The results for recognition based on this likelihood are also shown in the right panel. These general-purpose methods do not account for the clutter model and do not ensure that a unique element of $\mathbf{y}$ is assigned to each element of $\mathbf{x}$. Consequently, their recognition performance is lower than the structured approach proposed in this paper.

Figures 21-22 show several examples of inferences on shape classes in real images. In each row, the left panel shows the original image and the data $\mathbf{y}$ drawn over it. The next panel shows the posterior probability estimated using Algorithm 2, and the remaining two panels show examples of high probability $\mathbf{y}_s$ drawn over the image. In this experiment, we have used $m = 40$ and $n_0 = 20$. The examples of $\mathbf{y}_s$ can viewed as most likely polygons that can be constructed using the primitives present in the corresponding $\mathbf{y}$. Several observations can be made from these results. Firstly, the algorithm finds it easy to detect distinct, elongated objects (bottle, tools, bone, etc) but not so easy to distinguish between them. The first and the last examples in Figure 21 all show high posterior probability on those three related classes (1,3 and 15). Secondly, the algorithm is sensitive to the difference between training shapes and the test shapes. The test glass in Figure 21 is quite different in height from the glasses used in training shape priors for class 5. Similarly, the helicopter in Figure 22 is different from the training helicopters in class 9. This adversely affects Algorithm 2's ability to discriminate between classes. Lastly, the clutter present in this data is much more structured that in the simulated data (where clutter came from the Poisson model). Therefore, the algorithm is not as immune to clutter as it was in the simulated case. In the third example of Figure 22, where we get points from both the fishes, the algorithm tries to fit shapes using points from both the fishes. In the last panel of this row, the algorithm does succeed in ignoring clutter and finding the fish contour.

In terms of the computational cost, the time taken to

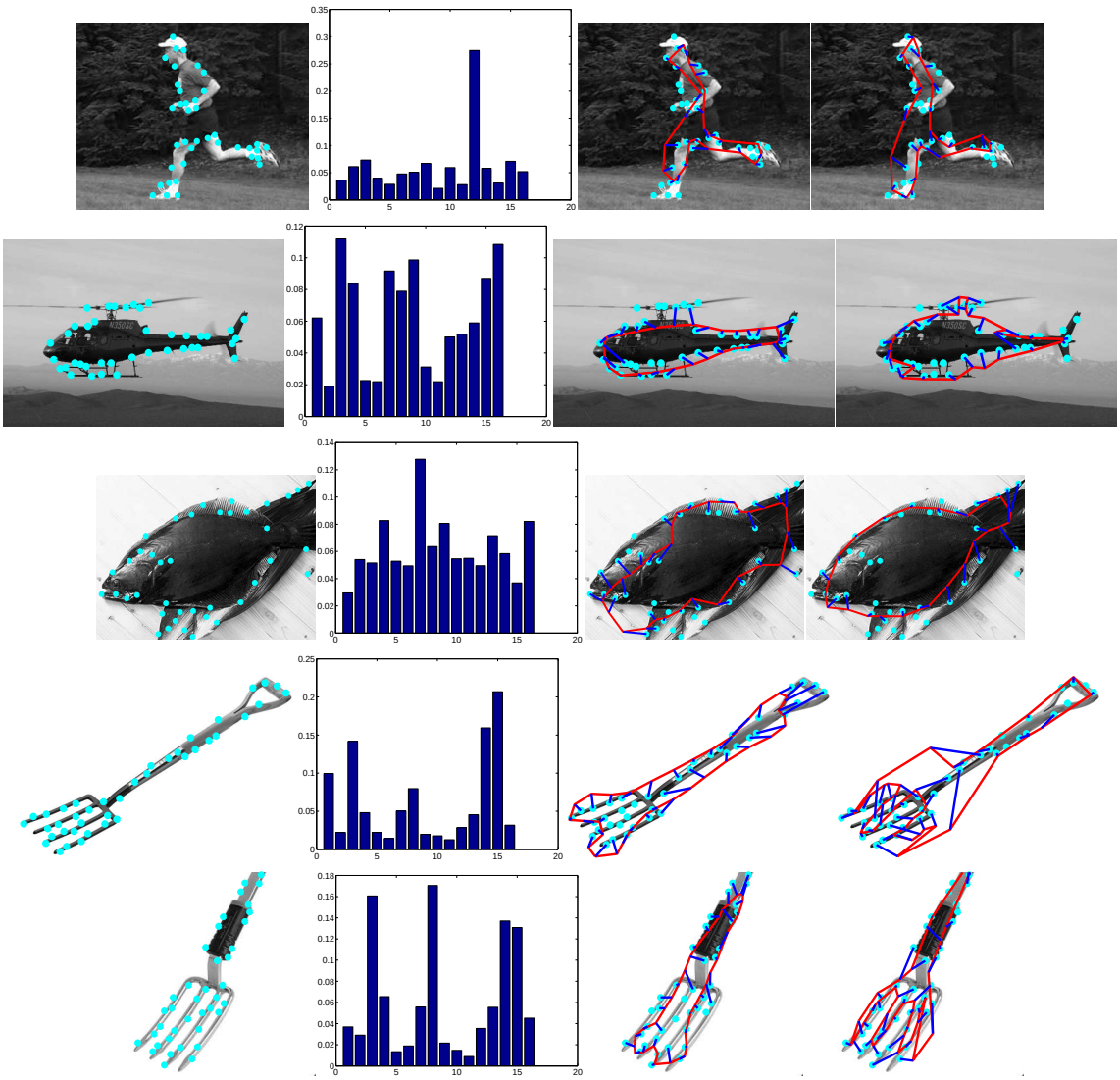


Fig. 22. Same as Figure 21. The correct classes in these examples are: 12, 9, 16, 14, and 14.

estimate $P(C_i|\mathbf{y})$ for each $\mathbf{y}$ (Algorithm 2) is approximately 60 seconds in MATLAB when $J = 300$. The corresponding time for estimating $P(C_i|\mathbf{y})$ in Problem I was 20 seconds. This increase in computational cost is attributed to the need to solve a more general registration problem in Problem II.

7 Conclusion

We have presented a Bayesian approach for finding shape classes in a given configuration of points that is characterized by under sampling of curves, observation noise, and background clutter. Rather than trying all possible permutations of points, we take a synthesis approach and simulate configurations using prior models on shape and sampling. The class posterior is estimated using a Monte Carlo approach. The strengths and the limitations of this framework depend squarely on the strengths and the limitations of the models used, especially $P(q|C_i)$ and $P(\gamma|C_i)$. In this paper, we have restricted to points, but additional primitives, including lines (first order) and arcs (second order) can be also be used.

Acknowledgments

We thank the producers of Kimia database. The research presented here was partially supported by ARO W911NF-04-01-0268, AFOSR FA9550-06-1-0324, and Northrop-Grumman Innovation Alliance grants, and by the INRIA/Florida State University Associated Team “SHAPES” grant.

References

[1] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, “Active shape models: Their training and application,” *Computer vision and image understanding*, vol. 61, no. 1, pp. 38–59, 1995.

[2] T.F.Cootes, G. Edwards, and C.J.Taylor, “Active appearance models,” in *in Proc. ECCV, H.Burkhardt & B. Neumann Ed.s*, vol. 2, 1998, pp. 484–498.

[3] E. Klassen, A. Srivastava, W. Mio, and S. Joshi, “Analysis of planar shapes using geodesic paths on shape spaces,” *IEEE Patt. Analysis and Machine Intell.*, vol. 26, no. 3, pp. 372–383, March, 2004.

[4] P. W. Michor and D. Mumford, “Riemannian geometries on spaces of plane curves,” *Journal of the European Mathematical Society*, vol. 8, pp. 1–48, 2006.

[5] A. Srivastava, M. I. Miller, and U. Grenander, “Bayesian automated target recognition,” *Handbook of Image and Video Processing, Academic Press*, pp. 869–881, 2000.

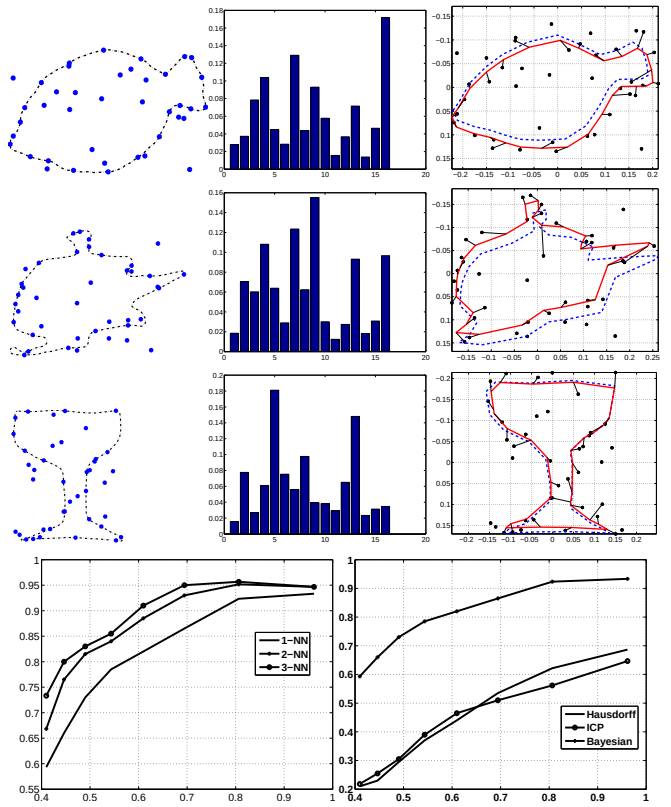
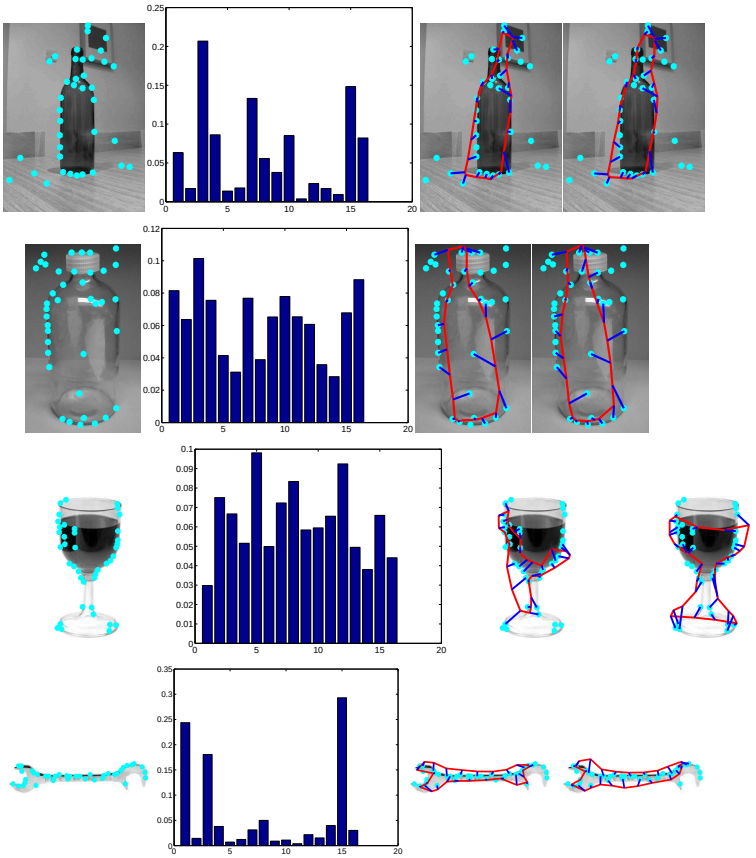


Fig. 20. Top three rows – The original curve and the simulated dataset $\mathbf{y}$ (left), and the estimated posterior $P(C_i|\mathbf{y})$ (middle), and a high-probability configuration (last). The correct classes in these examples are: 16, 9, and 5. The bottom left plot shows the average classification performance versus ν for the Bayesian approach, while the bottom right compares this approach with a classification that uses the Hausdorff metric and an ICP algorithm.



(a) (b) (c) (d)

Fig. 21. (a) Original image and detected primitives, (b) estimated $P(C_i|\mathbf{y})$ for 16 shape classes, (c)-(d) two examples of high-probability $\mathbf{x}$ and optimal correspondences with $\mathbf{y}$. The correct classes are 3, 3, 5, and 15

[6] T. B. Sebastian, P. N. Klein, and B. B. Kimia, "On aligning curves," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 1, pp. 116–125, 2003.

[7] M. A. Fischler and R. C. Bolles, "Random samples consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, pp. 381–395, 1981.

[8] R. Schnabel, R. Wahl, and R. Klein, "Efficient RANSAC for point-cloud shape detection," *Computer Graphics Forum*, vol. 26, no. 2, p. 214226, 2007.

[9] F. Memoli and G. Sapiro, "A theoretical and computational framework for isometry invariant recognition of point cloud data," *Foundations and Computational Mathematics*, vol. 5, no. 3, pp. 313–347, 2005.

[10] J. Glaunes, A. Trouvé, and L. Younes, "Diffeomorphic matching of distributions: A new approach for unlabelled point-sets and sub-manifolds matching." in *CVPR (2)*, 2004, pp. 712–718.

[11] P. Felzenszwalb and J. Schwartz, "Hierarchical matching of deformable shapes," in *Proceedings of CVPR*, 2007.

[12] A. Peter and A. Rangarajan, "A new closed-form information metric for shape analysis," in *Proc. of MICCAI, Copenhagen*, 2006.

[13] N. N. Čencov, *Statistical Decision Rules and Optimal Inferences*, ser. Translations of Mathematical Monographs. Providence, USA: AMS, 1982, vol. 53.

[14] S. J. Maybank, "Detection of image structures using the Fisher information and the Rao metric," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 12, pp. 1579–1589, 2004.

[15] A. Bhattacharya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bull. Calcutta Math. Soc.*, vol. 35, pp. 99–109, 1943.

[16] S. Amari, *Differential Geometric Methods in Statistics*, ser. Lecture Notes in Statistics, Vol. 28. Springer, 1985.

[17] S. Lang, *Fundamentals of Differential Geometry*. Springer, 1999.

[18] A. Srivastava, S. Joshi, W. Mio, and X. Liu, "Statistical shape analysis: Clustering, learning and testing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 4, pp. 590–602, 2005.

[19] W. Mio, A. Srivastava, and S. Joshi, "On shape of plane elastic curves," *International Journal of Computer Vision*, vol. 73, no. 3, pp. 307–324, 2007.

[20] S. H. Joshi, E. Klassen, A. Srivastava, and I. H. Jermyn, "A novel representation for efficient computation of geodesics between n-dimensional curves," in *IEEE CVPR*, 2007.

[21] ——, "Removing shape-preserving transformations in square-root elastic (SRE) framework for shape analysis of curves," in *EMMCVPR, LNCS 4679*, A. Y. et al., Ed., 2007, pp. 387–398.

[22] M. Leventon, W. E. L. Grimson, and O. Faugeras, "Statistical shape influence in geodesic active contours," in *Proc. IEEE Conf. Comp. Vision and Pattern Recognition*, 2000, pp. 316–323.

[23] M. Rochery, I. H. Jermyn, and J. Zerubia, "Phase field models and higher-order active contours," in *ICCV*, Beijing, China, 2005.

[24] G. Charpiat, O. Faugeras, and R. Keriven, "Approximations of shape metrics and application to shape warping and empirical shape statistics," *Journal of Foundations of Computational Mathematics*, vol. 5, no. 1, pp. 1–58, 2005.