Probabilistic Models of Object Geometry with Application to Grasping

Jared Glover, Daniela Rus and Nicholas Roy

***Abstract*—Robot manipulators typically rely on complete knowledge of object geometry in order to plan motions and compute grasps. But when an object is not fully in view it can be difficult to form an accurate estimate of the object's shape and pose, particularly when the object deforms.**

In this paper we describe a generative model of object geometry based on Mardia and Dryden's "Probabilistic Procrustean Shape" which captures both non-rigid deformations and object variability in a class. We extend their shape model to the setting where point correspondences are unknown using Scott and Nowak's COPAP framework. We use this model to recognize objects in a cluttered image and to infer their complete 2-D boundaries with a novel algorithm called OSIRIS. We show examples of learned models from image data and demonstrate how the models can be used by a manipulation planner to grasp objects in cluttered visual scenes.

1. Introduction

Robot manipulators largely rely on complete knowledge of object geometry in order to plan their motion and compute successful grasps. If an object is fully in view, its shape can be inferred from sensor data and a grasp computed directly. If the object is occluded by other entities in the scene, manipulations based on the visible part of the object may fail; to compensate, object recognition is often used to identify the location of the object and compute the grasp from a prior model. However, new instances of a known class of objects may vary from the prior model, and known objects may appear in novel configurations if they are not perfectly rigid. As a result, manipulation planning can pose a substantial challenge when objects are not fully in view.

Consider the camera image[1] of four toys in a box in figure 1(a). Prior knowledge of each object's geometry is extremely useful in that the geometry of the visible segments (such as the three parts of the stuffed bear) can be used to recognize each object, and a grasp can then be planned using the known geometry as in figure 1(b). However, having such a prior model of the geometry of every stuffed bear in the world is not only infeasible but unnecessary. Although the bear may change shape as it is handled and placed in different configurations, the general shape in terms of a head, limbs, etc. are roughly constant. Regardless of configuration, a single robust model for each class of objects which accounts for deformations in shape should be sufficient for recognition and grasp planning for most object types.

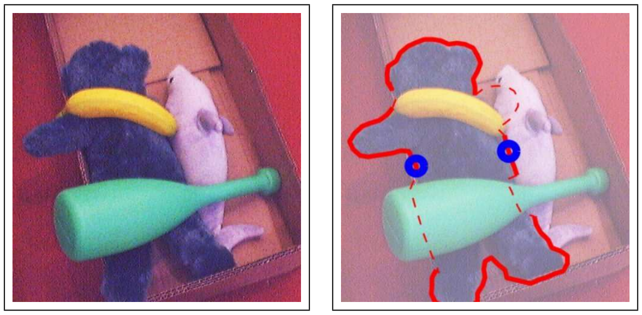

(a) Original Image (b) Recovered Geometry

Fig. 1. (a) A collection of toys in a box. The toys partially occlude each other, making object identification and grasp planning difficult. (b) By using learned models of the bear, we can identify the bear from the three visible segments and predict its complete geometry (shown by the red line; the dashed lines are the predicted outline of the hidden shape). This prediction of the complete shape can then be used in planning a grasp of the bear (planned grasp points shown by the blue circles).

In this paper we describe an algorithm for learning probabilistic models of visual object geometry. Statistical models of shape geometry have recently received attention in a number of domains, including computer vision and robotics (Felzenszwalb 2005; Elidan et al. 2006), but existing techniques have largely been coupled to tasks such as shape localization (Elidan et al. 2006), recognition and retrieval (Mokhtarian and Mackworth 1992; Belongie et al. 2002). On the other hand, many effective recognition and retrieval algorithms are discriminative in nature and create representations of the shape that make it difficult to perform additional inference such as recovering hidden object geometry.

Since we are specifically interested in using object geometry for manipulation planning, in section 3 we describe the representation of shapes as dense 2-D contours using *Procrustean shape models* (Dryden and Mardia 1998; Kendall et al. 1999) which provide invariance to translation, scale and rotation. In section 4 we present an algorithm for learning a Procrustean shape model from a set of complete object contours for a known object class. One challenge in using these shape models is that to compute the likelihood of a particular shape given a model, we must *a priori* know which points on the contour of the observed shape correspond to which points on the contour of the learned model. Thus, as part of the model learning process, we describe in section 4.2 how to solve the data association problem between points on two contours by extending Mardia and Dryden's model learning algorithm to the setting where correspondences are unknown. To infer correspondences between shapes, we use Scott and Nowak's

[1] Note that for the purposes of reproduction, the images have been cropped and modified from the original in brightness and contrast. They are otherwise unchanged.

COPAP framework (2006) which relies on the cyclic ordering of points around 2-D object boundaries to generate point-to-point matchings.

In the second technical section of the paper, we describe an algorithm for using the learned models to recognize occluded objects and complete the occluded geometry, an algorithm we call OSIRIS (Occluded Shape Inference Routine for Identification of Silhouettes, section 5). Given a set of learned models, we adapt COPAP to allow recognition and inference of partially-hidden, deformable shapes using two modifications. We extend the Procrustean model to incorporate "wildcard" points that match the hidden parts of partially-occluded shapes and secondly, we provide a novel point-assignment cost function based on a local Procrustean shape distance which we call the *Procrustean Local Shape Distance* (PLSD). We conclude with an experimental analysis of the algorithm on a large data set of object contours, a data set of real images and a demonstration of using the learned models to compute grasps.

The goal of this work is to provide estimates of geometry that allow a grasp to be planned for an object in a cluttered scene given a single image of the scene. The input to the algorithm is therefore a single image which is first segmented into perceptually similar regions. Although image segmentation is a challenging research problem, it is outside the scope of this paper and we rely on existing segmentation algorithms such as that of Shi and Malik (2000). The boundaries or contours of the image segments are extracted, and it is these representations of object geometry that are used throughout this paper. All of the techniques in this paper in principle extend to 3-D, but following the observation of Bone and Du (2001) that "grasp planning is much simpler in 2D, and 2D grasps are applicable to many 3D objects", we concentrate on the 2-D representations required to grasp objects (Shimoga 1996; Mirtich and Canny 1994) with a planar manipulator capable of supporting the weight of the object.

This paper extends the methods presented in Glover et al. (2006) to include extensions of the shape completion algorithm and a complete description of the model learning and shape inference algorithms. In addition to the preliminary experiments on a small set of shapes reported in Glover et al. (2006), we present results on the larger MPEG-7 shape dataset (Latecki et al. 2000).

2. RELATED WORK

Point-based statistical shape modeling began with the work of Kendall (1984) and Bookstein (1984) on landmark data in the 1980s. However, algorithms for finding Procrustean mean shapes (Kristof and Wingersky 1971; Gower 1975; Berge 1977) were developed long before the topology of shape spaces were well-understood (Kendall et al. 1999; Small 1996). In the classical computer vision literature, there has been considerable work on recognizing occluded objects, e.g., Lin and Chellappa (1987); Koch and Kashyap (1987); Grimson and Lozano-Pérez (1987). Recognizing and localizing occluded objects when the objects are rigid is known as the "bin-of-parts" or "bin-picking" problem. Despite being described by early vision researchers as the most difficult problem in automatic assembly (Gottschalk et al. 1989), there were many successful systems developed in the 1980's which solved the bin-of-parts problem in controlled environments. Most systems used greedy, heuristic search in a "guess-and-test" framework. Thus, in order to scale well with the number of object classes (and the number of objects in the image) they needed to be *especially* greedy, pruning away as much of the search space as possible to avoid an exponential running time. As a result, these approaches were especially sensitive to variations in shape.

An explosion of interest in object detection and segmentation in recent years has led to many advances in modeling shape variability (Cremers et al. 2003; Cootes et al. 1995; Felzenszwalb 2005; Blake and Isard 1998; Elidan et al. 2006). However, most of these shape deformation models have been applied in constrained environments, detecting only a handful of prescribed object types–for example in medical imaging (McInerney and Terzopoulos 1996) or face detection (Cootes et al. 1995). We believe our work is one of the first to perform probabilistic inference of deformable objects from partially occluded views. In terms of shape classification, shape contexts (Belongie et al. 2002) and spin images (Johnson and Hebert 1999) provide robust frameworks for estimating correspondences between shape features for recognition and modelling problems. Our work is very related but initial experiments with these descriptors motivated development of a better shape model for partial views of objects. In addition to the Procrustean shape model, Hu moments (Hu 1962) also provide invariance to position, scale, and rotation (as well as skew); however, our emphasis on boundary completion and manipulation made us choose the point-based, Procrustean approach over invariant moment, image-based approaches. Classical statistical shape models require a large amount of human intervention (e.g., hand-labelled landmarks) in order to learn accurate models of shape (Dryden and Mardia 1998); only recently have algorithms emerged that require little human intervention (Felzenszwalb 2005; Elidan et al. 2006).

The goal of our shape inference algorithm is to infer object geometry required for traditional grasp planning. Our presentation of object grasping in section 7 is intended as a demonstration of shape inference *in situ*, and could be used by any grasping algorithm that guaranteed geometrically closure properties Nguyen (1989); Pollard (1996). Our approach is in contrast to recent work by Saxena et al. (2006) that learns manipulation strategies directly from monocular images. While this technique shows promise, the focus has been generalizing as much as possible from as simple a data source as possible, rather than reasoning about multiple occluding objects. More recently, Katz and Brock (2008) showed that manipulation strategies could be learned from changes in object geometry; in occluded scenes their work would complement ours.

3. Probabilistic Models of 2-D Shape

We begin with a summary of the Procrustean shape model[2]. Formally, we represent an object $\mathbf{z}$ in an image as a set of n ordered points on the contour of the shape, $[\mathbf{z}_1^T \mathbf{z}_2^T \cdots \mathbf{z}_n^T]$, in a 2-D Euclidean space, where $\mathbf{z}_i = (x_i, y_i)$, and $\mathbf{z} \in \mathbb{R}^{2n}$. Our goal is to learn a probabilistic, generative model of $\mathbf{z}$ which is invariant to 2-D translation, scaling, and rotation. We begin by making the contour invariant with respect to position and scale, normalizing $\mathbf{z}$ so as to have unit length with centroid at the origin, that is,

$$\begin{aligned} \mathbf{z}' &= \{\mathbf{z}'_i = (x_i - \bar{x}, y_i - \bar{y})\} && (1) \\ \tau &= \frac{\mathbf{z}'}{|\mathbf{z}'|}, && (2) \end{aligned}$$

where τ is called the *pre-shape* of the contour $\mathbf{z}$. Since τ is a unit vector, the space of all possible pre-shapes of n points is the unit hyper-sphere, $\mathbb{S}_*^{2n-3}$, called *pre-shape space*[3].

Any pre-shape is a point on the hypersphere, and it can be shown that all 2-D rotations of the pre-shape lie on an orbit, $\mathcal{O}(\tau)$, of this hypersphere. (In fact, $\mathcal{O}(\tau)$ is a "great circle" orbit, or orbit of maximal length on this hypersphere.) In other words, rotating an object in a 2-D image corresponds to rotating its pre-shape along a great circle orbit of a hypersphere.

Since we can rotate any pre-shape through its orbit without changing the geometry of $\mathbf{z}$, we define the "shape" of $\mathbf{z}$ as an equivalence class of pre-shapes over rotations. In this way we arrive at a convenient vector-based description of shape which is fully invariant to translation, scaling, and rotation.

If we can define a distance metric between shapes, then we can infer a parametric distribution over the shape space. The spherical geometry of the pre-shape space requires a geodesic distance rather than Euclidean distance. The distance between τ_1 and τ_2 is defined as the smallest distance between their orbits,

$$\begin{aligned} d_p[\tau_1, \tau_2] &= \inf[d(\varphi, \psi) : \varphi \in \mathcal{O}(\tau_1), \psi \in \mathcal{O}(\tau_2)] && (3) \\ d(\varphi, \psi) &= \cos^{-1}(\varphi \cdot \psi). && (4) \end{aligned}$$

Kendall et al. (1999) defined d_p as the *Procrustean metric* where $d(\varphi, \psi)$ is the geodesic distance between φ and ψ, and φ and ψ are specific vectors on the orbits of τ_1 and τ_2. (Note that while great circle orbits in a standard 3-D sphere will always intersect, the extra dimensions in a hypersphere allow for great circles that do not intersect.)

Since the inverse cosine function is monotonically decreasing over its domain, it is sufficient to maximize $\varphi \cdot \psi$, which is equivalent to minimizing the sum of squared distances between corresponding points on φ and ψ (since φ and ψ are unit vectors). For every rotation of φ along $\mathcal{O}(\tau_1)$ there exists a rotation of ψ along $\mathcal{O}(\tau_2)$ which will find the global minimum geodesic distance. Thus, to find the minimum distance, we need only rotate one pre-shape while holding the other one fixed. We call the rotated ψ which achieves this optimum the *orthogonal Procrustes fit* of τ_2 onto τ_1, and the angle θ^* is called the *Procrustes fit angle*.

We can solve for the minimization of equation (3) in closed form by representing the points of τ_1 and τ_2 in complex coordinates $(x + yi)$, which naturally encode rotation in the plane by scalar complex multiplication. This gives d_p as

$$\begin{aligned} d_p[\tau_1, \tau_2] &= \cos^{-1} |\tau_2^H \tau_1| && (5) \\ \theta^* &= \arg(\tau_2^H \tau_1), && (6) \end{aligned}$$

where τ_2^H is the *Hermitian*, or complex conjugate transpose of the complex vector τ_2, and $\arg(\cdot)$ is the complex argument operator: i.e. $\arg(x + iy) = \tan^{-1}(y/x)$.

4. Learning Shape Models

In order to learn a probabilistic model of the geometry of different object classes, we compute a distribution for each object class from complete object contours extracted from training images. We will start by describing the classical tangent-space model learning approach of Dryden and Mardia (1998), which requires known point-to-point correspondences between all of the training shapes as input. We then describe how to compute correspondences between the points on two shape contours, so that by section 4.6 we can present a model learning algorithm which does not require correspondences to be known ahead of time.

In many applications, pre-shape data will be tightly localized around a mean shape. In such cases, the tangent space to the pre-shape hypersphere located at the mean shape will be a good approximation to the pre-shape space, as in figure 2. By linearizing the distribution in this manner, one can take advantage of standard multivariate statistical analysis techniques by representing the shape distribution as a Gaussian and reducing the dimensionality of the model with Principal Components Analysis (PCA) in order to prevent overfitting[4].

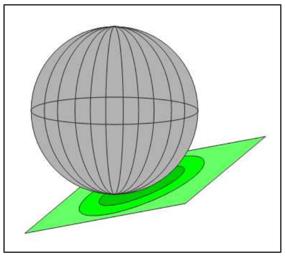
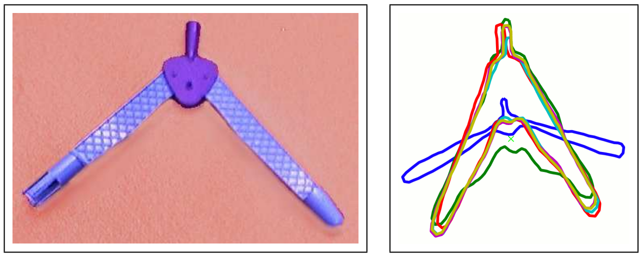


Fig. 2. Although the distribution of pre-shape geometries lies on the surface of a hypersphere, we approximate this distribution with a distribution over the plane tangent to the sphere.

[2] For a fuller treatment of this subject, we refer the reader to Dryden and Mardia (1998); Kendall et al. (1999); Small (1996).

[3] Following Small (1996), the star subscript is added to remind us that $\mathbb{S}_*^{2n-3}$ is embedded in $\mathbb{R}^{2n}$, not the usual $\mathbb{R}^{2n-2}$.

[4] In cases where the pre-shape data is more spread out, one can use a *complex Bingham distribution* (Dryden and Mardia 1998), one of several distributions which attempt to incorporate the non-linearity of shape space directly, with no approximation. The primary advantage to using the tangent space Gaussian model lies in its simplicity; more experimentation is needed to determine whether the gains in modelling accuracy by using other shape distributions would justify the additional complexity in our robotic grasping domain.



(a) Example Object

(b) Class Distribution

Fig. 3. (a) An example image of a chalk compass. The compass can deform by opening and closing. (b) Sample shapes from the learned distribution along different eigenvalues of the distribution.

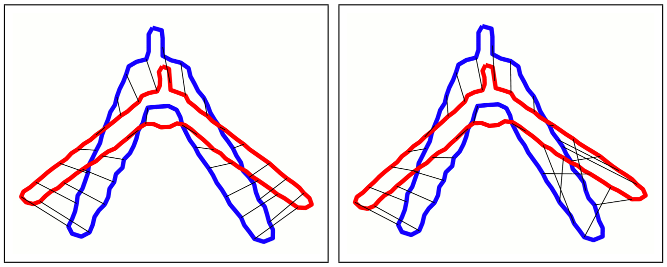


Fig. 4. Order-preserving matching (left) vs. Non-order-preserving matching (right). The thin black lines depict the correspondences between points in the red and blue contour. Notice the violation of the cyclic-ordering constraint between the right arms of the two contours in the right image.

In order to fit a tangent space Gaussian approximation to a set of shapes, it is sufficient to compute the mean and covariance of the training data. For each object class c, we compute a mean shape μ^* from a set of pre-shapes $\{\tau^{(1)}, \dots, \tau^{(n)}\}$ by minimizing the sum of Procrustean distances from each pre-shape to the mean,

$$\mu^* = \operatorname*{arginf}_{\mu} \sum_j [d_p(\tau^{(j)}, \mu)]^2, \tag{7}$$

subject to the constraint that $\|\mu\| = 1$. In 2-D, this minimization can be done in closed form; iterative algorithms exist for computing μ^* in higher dimensions (Berge 1977; Gower 1975).

In order to estimate the covariance of the shape distribution from the sample pre-shapes $\{\tau^{(1)}, \dots, \tau^{(n)}\}$, we rotate each $\tau^{(j)}$ to fit the mean shape μ (in the Procrustean sense), and then project the rotated pre-shapes into the tangent space of the pre-shape hypersphere at the mean shape. The *tangent space coordinates* for pre-shape $\tau^{(j)}$ with respect to mean shape μ are given by

$$\mathbf{v}^{(j)} = (I - \mu\mu^H)e^{i\theta^*}\tau^{(j)}, \tag{8}$$

where $i^2 = -1$ and θ^* is the optimal Procrustes-matching rotation angle of $\tau^{(j)}$ onto μ. (The $e^{i\theta^*}$ term rotates the pre-shape $\tau^{(j)}$ by θ^*, while $(I - \mu\mu^H)$ projects the rotated pre-shape into tangent space.)

We then use Principal Components Analysis (PCA) in the tangent space to model the principal axes of the Gaussian shape distribution of $\{\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(n)}\}$. Figure 3(a) shows one example out of a training set of images of a deformable object. Figure 3(b) shows sample objects drawn from the learned distribution. The red contour is the mean, and the green and blue samples are taken along the first two principal components of the distribution.

4.1. *Shape Classification*

Given K previously learned shape classes $C_1, \dots, C_K$ with shape means $\mu^{(1)}, \dots, \mu^{(K)}$ and covariance matrices $\Sigma^{(1)}, \dots, \Sigma^{(K)}$, and given a measurement $\mathbf{x}$ of an unknown object shape, we can now compute the likelihood of a shape class given a measured object, $P(C_k|\mathbf{x})$. The shape classification problem is then one of finding the maximum likelihood class, $\hat{C}$, which we can compute as

$$\hat{C} = \arg\max_{C_k} P(C_k|\mathbf{x}) \tag{9}$$

$$= \arg\max_{C_k} P(\mathbf{x}|C_k)P(C_k). \tag{10}$$

Given the mean and covariance of a shape class, we can compute the likelihood of a measured object given C_k by computing τ, the pre-shape of $\mathbf{x}$, and then projecting τ into C_k's tangent space with equation (8), yielding $p(\mathbf{x}|C_k) = \mathcal{N}(\mathbf{v}^{(k)}; \mathbf{0}, \Sigma^{(k)})$. Assuming a uniform prior on C_k, we can compute the maximum likelihood class as

$$\hat{C} = \operatorname*{argmax}_{C_k} \mathcal{N}(\mathbf{v}^{(k)}; \mathbf{0}, \Sigma^{(k)}). \tag{11}$$

Note that we place the origin of the tangent space k at $\mu^{(k)}$; as a result, the Gaussian distribution in the tangent space is zero mean but the projection onto the tangent space implicitly accounts for the distance from the mean.

4.2. *Data Association and Shape Correspondences*

Evaluating the likelihood given by equation (11) requires calculating the Procrustean distance d_p between the pre-shape of the observed contour $\mathbf{x}$ and the mean $\mu^{(k)}$. More generally, the Procrustean distance between any two contours $\mathbf{x}$ and $\mathbf{y}$ implicitly assumes that there is a known correspondence between point $\mathbf{x}_i$ in $\mathbf{x}$ and point $\mathbf{y}_i$ in $\mathbf{y}$, for all i. Therefore, before we can compute the probability of a contour or learn the mean and covariance of a set of pre-shapes, we must be able to compute the correspondences between contours, matching each point in $\mathbf{x}$ to a corresponding point on $\mathbf{y}$[5].

Our goal is to match the points of one contour, $\mathbf{x}_1, \dots, \mathbf{x}_n$ to the points on another, $\mathbf{y}_1, \dots, \mathbf{y}_m$. Let Φ denote a correspondence vector, where ϕ_i is the index of $\mathbf{y}$ to which $\mathbf{x}_i$ corresponds; that is: $\mathbf{x}_i \rightarrow \mathbf{y}_{\phi_i}$. We wish to find the most likely Φ given $\mathbf{x}$ and $\mathbf{y}$, that is, $\Phi^* = \text{argmax}_\Phi\, p(\Phi|\mathbf{x}, \mathbf{y})$. If we assume that the likelihood of individual points $\{\mathbf{x}_i\}$ and $\{\mathbf{y}_j\}$ are conditionally independent given Φ (that is, two measurements of the same object are independent given knowledge

[5] There is also an assumption that the number of points in $\mathbf{x}$ and $\mathbf{y}$ are the same. If $\mathbf{x}$ and $\mathbf{y}$ differ in their number of points, we must find a way to add points to one shape (or remove points from the other) in order to bring them into one-to-one correspondence. We address this issue in section 4.6.

of the object, a very standard assumption in robotics), then

$$\begin{aligned}\Phi^* &= \operatorname*{argmax}_{\Phi} \frac{1}{Z} p(\mathbf{x}, \mathbf{y}|\Phi) p(\Phi) \\ &= \operatorname*{argmax}_{\Phi} \frac{1}{Z} \prod_{i=1}^{n} p(\mathbf{x}_i, \mathbf{y}_{\phi_i}) p(\Phi) \qquad (12)\end{aligned}$$

where Z is a normalizing constant.

Solving for the most likely correspondences between sets of data is an open problem in a number of fields, including computer vision and robotic mapping. As object geometries vary due to projection distortions, sensor error, or even natural object dynamics, it is non-trivial to determine *which* part of an object image corresponds to *which* part of a previous image. However, we can take advantage of geometric properties of objects to prune the search space, generating solutions to the correspondence problem efficiently. These geometric properties constitute priors over the likelihood $p(\Phi)$ in equation (12), either reducing or setting to 0 the *a priori* likelihood of certain correspondences.

4.3. Priors over Correspondences

The first geometric property that we use is a hard constraint on correspondence orderings. By the nature of object contours, our specific shape correspondence problem contains a *cyclic order-preserving* constraint, that is, correspondences between the two contours cannot "cross" each other numerically (as opposed to geometrically—we are not suggesting that lines drawn between matched points cannot cross). For example, if $\mathbf{x}_3$ corresponds to $\mathbf{y}_3$ and $\mathbf{x}_5$ corresponds to $\mathbf{y}_5$, then $\mathbf{x}_4$ must correspond to $\mathbf{y}_4$ (or nothing); it is a violation of the cyclic ordering constraint for $\mathbf{x}_4$ to correspond to $\mathbf{y}_2$ or $\mathbf{y}_6$.

Scott and Nowak (2006) define the Cyclic Order-Preserving Assignment Problem (COPAP) as the problem of finding an optimal one-to-one matching such that the assignment of corresponding points preserves the cyclic ordering inherited from the contours. Figure 4 shows an example set of correspondences (the thin black lines) that preserve the cyclic order-preserving constraint on the left, whereas the correspondences on the right of figure 4 violate the constraint at the right of the shape (notice that the association lines cross). We therefore set $p(\Phi) = 0$ for any correspondence vector that would violate the order-preserving constraint. In the following sections, we show how the COPAP algorithm can be used to solve for these correspondences using an appropriate point-assignment cost function for matching the contours of deformable objects.

The second geometric property we use as a prior $p(\Phi)$ over correspondences is to prefer models which provide the greatest number of corresponding points between the two shapes. The correspondence model must allow for the possibility that due to variations in object geometry, some points $\{\mathbf{x}_i, \ldots, \mathbf{x}_j\}$ in sequence do not correspond to any points in $\mathbf{y}$. For example, if sensor noise has introduced spurious points along an object edge or if the shapes vary in some significant way, such as an animal contour with three legs where another has four, the most likely correspondence is that some points in $\mathbf{x}$ are simply not present in $\mathbf{y}$, or must be "skipped". However, we prefer models where as much of $\mathbf{x}$ is matched to $\mathbf{y}$ as possible. We therefore use a prior over correspondences, $p(\Phi)$, that is an exponential distribution over the number of skipped correspondences, subject to the cyclic ordering constraint.

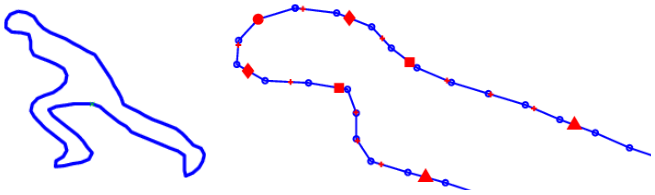


Fig. 5. Local shape neighborhoods. (a) The full contour of a running person. (b) Closeup of the top of the contour in (a), with local shape neighborhoods about the point ● of size $k = 1$ (◆), $k = 2$ (■), and $k = 3$ (▲), where the original contour points are shown as small blue circles (○) and the interpolated neighborhood points are shown as small red $+$'s. The neighborhoods are chosen so that the length of the largest neighborhood (▲) is 20% of the full contour length.

We "skip" individual correspondences in $\mathbf{x}$ by allowing $\phi_i = 0$. (Points $\mathbf{y}_j$ are skipped when $\nexists i \;\; s.t. \;\; \phi_i = j$). To minimize the number of such skipped assignments, we give diminishing likelihood to Φ as the number of skipped points increases. For Φ with k_Φ skipped assignments (in $\mathbf{x}$ and $\mathbf{y}$),

$$p(\Phi) = \begin{cases} \frac{1}{Z_\Phi} \exp\{-k_\Phi \cdot \lambda\} & \text{if } \Phi \text{ is cyclic ordered} \\ 0 & \text{otherwise,} \end{cases} \qquad (13)$$

where Z_Φ is a normalizing constant and λ is a likelihood penalty for skipped assignments. A high value of λ indicates that the algorithm should skip over as few points as possible, while a low value tells the algorithm to skip over any points that do not have a near-perfect match on the other shape boundary. Throughout this paper, we use a value of $.03$ for λ.

We add to this prior the cyclic-ordering constraint by allowing $p(\Phi) > 0$ if and only if

$$\exists\omega \;\; s.t. \;\; \phi_\omega < \phi_{\omega+1} < \cdots < \phi_n < \phi_1 < \cdots < \phi_{\omega-1}. \qquad (14)$$

We call ω the *wrapping point* of the assignment vector Φ. Each assignment vector, Φ, which obeys the cyclic-ordering constraint must have a unique wrapping point, ω.

4.4. Correspondence Likelihoods: PLSD

Given an expression for the correspondence prior, we also need an expression for the likelihood that two points $\mathbf{x}_i$ and $\mathbf{y}_{\phi_i}$ correspond to each other, $p(\mathbf{x}_i, \mathbf{y}_{\phi_i})$, which we model as the likelihood that the local geometry of the contours match. Section 3 described a probabilistic model for global geometric similarity using the Procrustes metric, and we modify this model for computing the likelihood of local geometries. We compute this likelihood by first forming a distance metric over local shape geometry, which we call the *Procrustean Local Shape Distance* (PLSD). Given such a distance, d_{PLS}, we compute the likelihood as the probability of d_{PLS} under a zero-mean Gaussian model with fixed variance, σ. Since σ is fixed for every local shape correspondence likelihood, we can simply write it as part of the normalization constant to ensure that the distribution $p(\mathbf{x}_i, \mathbf{y}_{\phi_i})$ sums to one. Thus,

$$p(\mathbf{x}_i, \mathbf{y}_{\phi_i}) = \frac{1}{Z_{PLS}} \exp\left\{-[d_{PLS}(x_i, y_{\phi_i})]^2\right\} \qquad (15)$$

where Z_{PLS} is a normalization constant.

In order to compute the Procrustean local shape distance, we first need a description of the local shape about $\mathbf{x}_i$. (When the local spacing of $\mathbf{x}$ and $\mathbf{y}$ is uneven, we sample points evenly spaced about $\mathbf{x}_i$ and $\mathbf{y}_{\phi_i}$, interpolating as necessary to ensure that there is the same number of evenly-spaced points in the local neighborhood on each shape.) We define the *local neighborhood* of size $k \in \mathbb{N}$ about $\mathbf{x}_i$ as:

$$\eta_k(x_i) = \langle \delta_x^i(-2^k\Delta), ..., \delta_x^i(0), ..., \delta_x^i(2^k\Delta)\rangle \qquad (16)$$

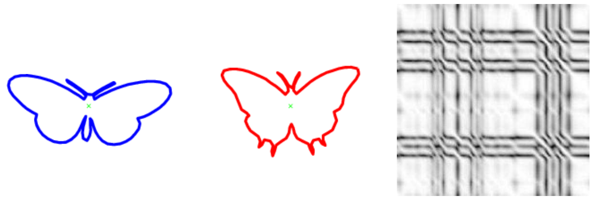
where $\delta_x^i(d)$ returns the point from $\mathbf{x}$'s contour a distance of d starting from $\mathbf{x}_i$ (clockwise for d positive or counter-clockwise for d negative). Also, $\delta_x^i(0) = \mathbf{x}_i$. The parameter Δ determines the step size between points, and thus the resolution of the local shape. We have found that setting Δ such that the largest neighborhood is between 10–30% of the total shape circumference (we use 20% throughout this paper) yields good results on most datasets (figure 5).

The Procrustean Local Shape Distance, d_{PLS}, between two points, x_i and y_j is the mean Procrustean shape distance over neighborhood sizes k:

$$d_{PLS}(x_i, y_j) = \sum_k \xi_k \cdot d_P[\eta_k(\mathbf{x}_i), \eta_k(\mathbf{y}_j)] \qquad (17)$$

with neighborhood size prior ξ. Experimentally, we found that setting ξ_k to be inversely proportional to the contour length of the local shape neighborhood of size k yielded the best results on our datasets[6]. Intuitively, this choice of neighborhood size prior expresses the common-sense principle that the points closest to $\mathbf{x}_i$ and $\mathbf{y}_j$ matter most in determining the quality of the local shape match between $\mathbf{x}_i$ and $\mathbf{y}_j$. The Procrustean Local Shape Distance can thus be thought of as a locally-weighted, multi-scale shape distance describing the "dissimilarity" between the local shapes around $\mathbf{x}_i$ and $\mathbf{y}_j$. Since it is a weighted combination of Procrustean distances, it is also invariant to changes in position, scale, and orientation, which is why we chose to use it in this work to match the points of deformable contours.

In figure 6(c), we see the matrix of squared Procrustean Local Shape Distances between all pairs of points on two butterfly contours. The figure shows that the squared-distance matrix has a very regular structure. The dark, high-distance rows and columns correspond to strong local features on each shape—for example, the tips of the wings, or the antennae; while the light, low-distance rows and columns correspond to flat, smooth portions of the two contours.



(a) contour $\mathbf{x}$ (b) contour $\mathbf{y}$ (c) PLSD matrix

Fig. 6. PLSD matrix for two butterfly contours (a-b). The intensity of pixel (i, j) represents the squared Procrustean Local Shape Distance between $\mathbf{x}_i$ and $\mathbf{y}_j$. Note that the squared-distance matrix has a very regular structure.

4.5. *Solving* COPAP

Although we assume independence between local features $\mathbf{x}_i$ and $\mathbf{y}_j$ in equation (12), the cyclic-ordering constraint leads to dependencies between the assignment variables ϕ_i in a non-trivial way. However, if we initially assign the wrapping point ω from equation (14), the cyclic constraint then becomes a linear one, which leads to a Markov chain. The standard approach to solving COPAP is thus to try setting the wrapping point, ω, to each possible value from 1 to n. Given $\omega = k$, the resulting chain can be solved by dynamic programming. (We refer the reader to Scott and Nowak (2006) for a full treatment of this dynamic programming algorithm.)

In this approach, the point-assignment likelihoods of equation (15) are converted into a cost function $C(i, \phi_i) = d_{PLS}(x_i, y_{\phi_i})$ by taking a log likelihood, and Φ is optimized using

$$\Phi^* = \underset{\Phi}{\text{argmax}} \ \log\left(\prod_i p(\mathbf{x}_i, \mathbf{y}_{\phi_i}) \cdot p(\Phi)\right) \qquad (18)$$

$$= \underset{\Phi}{\text{argmin}} \left(\sum_i C(i, \phi_i)\right) + \lambda \cdot k(\phi) \qquad (19)$$

$$\text{s.t. } \forall \phi_i \ p_{co}(\phi_i) > 0$$

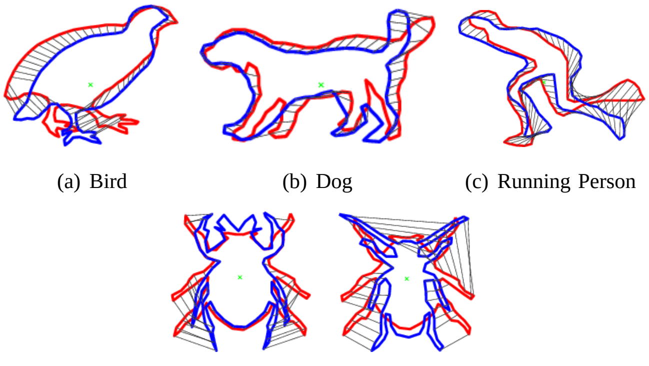
where $k(\Phi)$ is the number of points skipped in the assignment Φ. Solving for Φ using equation (19) takes $O(n^2m)$ running time; however a bisection strategy exists in the dynamic programming search graph (Maes 1990; Scott and Nowak 2006) which reduces the complexity to $O(nm \log n)$.

Figure 7 shows examples of the inference process and correspondences between pairs of contours. Figure 7(a) is interesting because the correspondence algorithm has correctly associated all of the single leg present in the blue contour with the right-most leg in the red contour, and skipped any associations of the left-leg in the red contour. Figure 7(e), the beetle model, shows a failed correspondence at the top-right leg of beetle; this is a challenging case because there are a number of similar structures for the correspondence to match.

4.6. *Model Learning With Unknown Correspondences*

In the beginning of this section, we showed how to learn a probabilistic shape model from a set of training shapes in one-to-one correspondence with each other. We can now extend this learning algorithm to the case when correspondences be-

[6] Throughout the experiments in this work, we use $k = \{1, 2, 3\}$ with $\xi_1 = 4/7$, $\xi_2 = 2/7$, and $\xi_3 = 1/7$ since the local neighborhood of size $k + 1$ is twice as long as the local neighborhood of size k.


(a) Bird (b) Dog (c) Running Person

(d) Beetle (e) Beetle

Fig. 7. Examples of shape correspondences found using COPAP with the Procrustean Local Shape Distance. Note that in (e) the top-right legs of the beetles are incorrectly matched due to the local nature of the point assignment.

tween training shapes are unknown. The entire model learning algorithm is shown in table I.

In order to build a shape model from a set of n training shapes, $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$, we first estimate the mean shape, $\hat{\mu}$, by sequentially adding in one training shape at a time and recomputing the model. Each time a new training shape, $\mathbf{x}^{(i)}$ is added to the model, we first correspond $\mathbf{x}^{(i)}$ to $\hat{\mu}$ using COPAP. We then update $\hat{\mu}$ by taking the average of τ_x and $\tau_{\hat{\mu}}$ (after rotating the pre-shape of $\mathbf{x}^{(i)}$, τ_x, to match $\hat{\mu}$'s pre-shape, $\tau_{\hat{\mu}}$), where the average is weighted by the number of training shapes used to compute $\hat{\mu}$ so far.

Once an initial estimate of mean shape is obtained, we use this estimate as a reference shape to bring all n training shapes into one-to-one correspondence with each other by again using COPAP to correspond each $\mathbf{x}^{(i)}$ to the mean shape estimate, $\hat{\mu}$. After the training shapes have been brought into one-to-one correspondence with $\hat{\mu}$ (and thus with each other), we can then use the fully-corresponded training set as input to the basic tangent space PCA learner from the beginning of section 4 to estimate mean shape and covariance.

A critical issue we have not yet addressed is ensuring that the training contours are all sampled with the same point density in the same regions and have the same overall length to allow an eventual one-to-one correspondence of training data to the model. We ensure this by incrementally growing the model to take the union of all contour points on all training instances. During learning, if $\mathbf{x}^{(i)}$ contains points not in the current model $\hat{\mu}$, we add the skipped points from $\mathbf{x}^{(i)}$ to $\hat{\mu}$, and vice-versa. Once the initial mean shape has been computed, we iteratively recompute the mean, adding points to $\mathbf{x}^{(i)}$ where $\hat{\mu}$'s points are skipped in the correspondence (but *not* vice-versa since all of $\mathbf{x}^{(i)}$ was added to the model in a previous iteration). Additionally, we have the option to *force* all of $\mathbf{x}^{(i)}$'s points to correspond to some point on $\hat{\mu}$, so that no training points are thrown away in the model learning process[7].

```
LEARN-SHAPE-MODEL($\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}$)

**Input:** A set of $n$ full shape contours, $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$.

**Output:** Shape model $\mathbf{S}$ consisting of mean shape $\mu$ and covariance $\Sigma$.

  1) Set $\hat{\mu} \leftarrow \mathbf{x}^{(1)}$.
  2) For $i = 2, \ldots, n$:
       a) $\mathbf{x} \leftarrow \mathbf{x}^{(i)}$.
       b) $\Phi \leftarrow$ COPAP($\mathbf{x}, \hat{\mu}$).
       c) $(\mathbf{x}', \hat{\mu}') \leftarrow$ ADD-SKIPPED-PTS($\mathbf{x}, \hat{\mu}, \Phi$).
       d) $\tau_x \leftarrow$ PRESHAPE($\mathbf{x}'$).
       e) $\tau_{\hat{\mu}} \leftarrow$ PRESHAPE($\hat{\mu}'$).
       f) Rotate $\tau_x$ to fit $\tau_{\hat{\mu}}$.
       g) $\hat{\mu} \leftarrow \frac{(i-1)\tau_{\hat{\mu}} + \tau_x}{i}$.
  3) For $i = 1, \ldots, n$:
       a) SET-SKIP-COST($\mathbf{x}_j^{(i)}, \infty$), $\forall j$
       b) $\Phi \leftarrow$ COPAP($\mathbf{x}^{(i)}, \mu$).
       c) $\mathbf{x}^{(i)} \leftarrow$ ADD-SKIPPED-PTS($\mathbf{x}^{(i)}, \mu, \Phi$).
  4) Let $\mathbf{X} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$.
  5) Return $(\mu, \Sigma) \leftarrow$ SHAPE-PCA($\mathbf{X}$).
```

TABLE I
THE SHAPE MODEL LEARNING ALGORITHM, WITH UNKNOWN CORRESPONDENCES. IN THE FIRST FOR-LOOP, AN INITIAL ESTIMATE OF THE MEAN SHAPE, $\hat{\mu}$, IS LEARNED. IN THE SECOND FOR-LOOP, ALL THE TRAINING SHAPES ARE BROUGHT INTO ALIGNMENT WITH $\hat{\mu}$, RESULTING IN A DATASET OF MATCHED SHAPES, $\mathbf{X}$ WHICH ARE FED INTO THE BASIC TANGENT SPACE PCA ALGORITHM FROM THE BEGINNING OF SECTION 4.

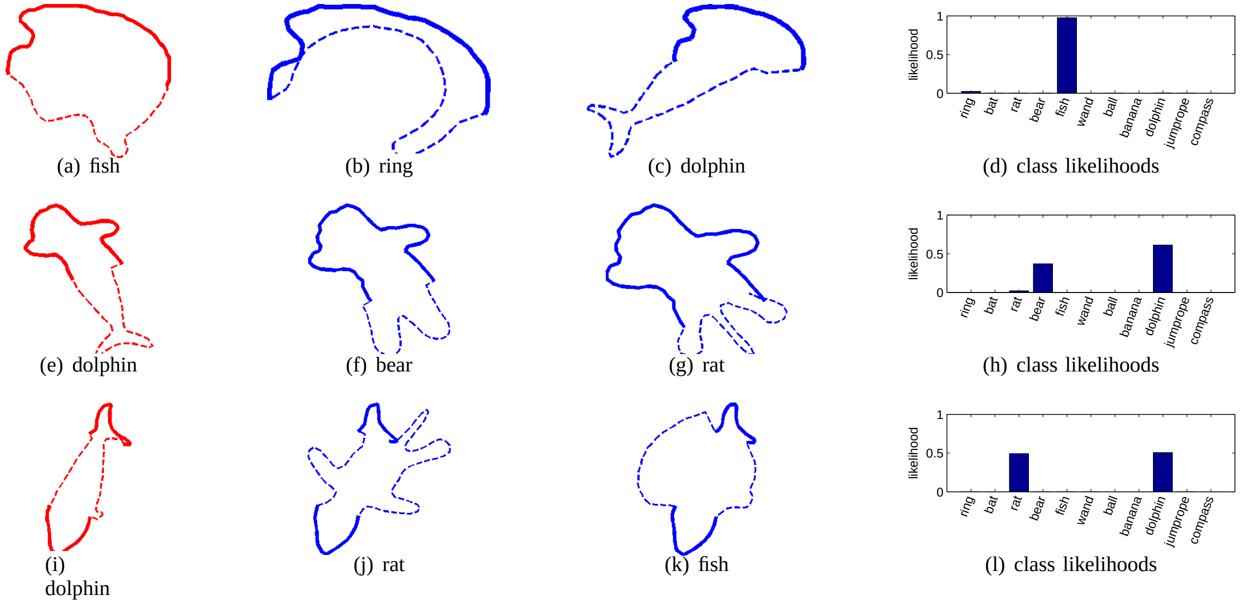
Although not strictly necessary, this option has been successful in increasing the accuracy of shape models in our datasets, so we include this step throughout the experiments described here.

We use a similar (but simpler) algorithm to compute the likelihood of a new shape measurement, $\mathbf{y}$ with respect to a shape model, $\mathbf{S} = \mathcal{N}(\mu, \Sigma)$, when correspondences between $\mathbf{y}$ and $\mathbf{S}$ are unknown. First, we bring $\mathbf{y}$'s points into one-to-one correspondence with the mean shape, μ using COPAP. As before, we prohibit COPAP from skipping any of $\mathbf{y}$'s points so that measurements are not thrown away. Then, we rotate $\mathbf{y}$'s pre-shape to match μ, project into tangent-space, and compute the Gaussian likelihood of the projected pre-shape with respect to Σ. The shape likelihood algorithm is shown in table II.

5. SHAPE COMPLETION: OSIRIS

We now turn to the second technical contribution in this paper, which is an algorithm for estimating the complete geometry of an object from an observation of part of its contour, with respect to a given shape model. (We will refer to this as the “shape completion” problem.) After presenting our shape completion algorithm, which we call OSIRIS (Occluded Shape Inference Routine for Identification of Silhouettes), we will then discuss how to perform classification from partial shape observations; finally, we will conclude this section with

[7] We implement this asymmetric correspondence option by changing the skip cost from λ to ∞ in COPAP for all points on $\mathbf{x}^{(i)}$. In the algorithm in Table I, the function SET-SKIP-COST on line (3a) sets the skip cost to ∞ for all of $\mathbf{x}^{(i)}$'s points.



(a) fish (b) ring (c) dolphin (d) class likelihoods

(e) dolphin (f) bear (g) rat (h) class likelihoods

(i) dolphin (j) rat (k) fish (l) class likelihoods

Fig. 8. Example shape completions. Each row contains a different partial contour. Each column shows the most-likely completion given each of the top three most likely classes, ranked in order from left to right. The first row has a single most likely class; the fish is completed correctly. The second and third row have multiple competing hypotheses. In row 2, the dolphin and bear are both reasonable completions, although the dolphin is slightly more likely. In row three, both the dolphin and rat lead to very reasonable completions and are equally likely.

```
SHAPE-LIKELIHOOD($\mathbf{y}$,$\mathbf{S}$)

Input: A full shape contour, $\mathbf{y}$, and a shape model $\mathbf{S}$ with mean $\mu$ and covariance $\Sigma$.

Output: Likelihood, $L$.
  • SET-SKIP-COST($\mathbf{y}_i, \infty$), $\forall i$
  • $\Phi \leftarrow$ COPAP($\mathbf{y}, \mu$).
  • $\mathbf{y}' \leftarrow$ ADD-SKIPPED-PTS($\mathbf{y}, \mu, \Phi$).
  • $\tau_y \leftarrow$ PRESHAPE($\mathbf{y}'$).
  • Rotate $\tau_y$ to fit $\mu$.
  • $\mathbf{v} \leftarrow$ PROJECT($\tau_y, \mu$).
  • Return $L \leftarrow \mathcal{N}(\mathbf{v}; \mathbf{0}, \Sigma)$.
```

TABLE II
THE SHAPE LIKELIHOOD ALGORITHM, WITH UNKNOWN CORRESPONDENCES.

some extensions to the basic OSIRIS algorithm.

5.1. The Shape Completion Algorithm

We phrase the shape completion problem as a maximum likelihood estimation problem, estimating the missing points of a shape with respect to a Gaussian tangent space shape distribution D as

$$\mathbf{z}^* = \arg\max_{\mathbf{z}} P_D(\mathbf{y}, \mathbf{z}), \tag{20}$$

where $\mathbf{y}$ and $\mathbf{z}$ represent the observed and hidden portions of the object boundary, respectively. A key challenge we face in finding the hidden points $\mathbf{z}$ is that in order to compute P_D as a Gaussian tangent-space likelihood, we must know which dimensions in the model distribution D correspond to the observed and hidden points $\mathbf{y}$ and $\mathbf{z}$. Our approach therefore to solving the shape completion optimization is to jointly optimize over both the correspondences, $\mathbf{\Phi}$, and the hidden points, $\mathbf{z}$, as in

$$(\mathbf{z}^*, \mathbf{\Phi}^*) = \arg\max_{\mathbf{z}, \mathbf{\Phi}} P_D(\mathbf{y}, \mathbf{z}, \mathbf{\Phi}). \tag{21}$$

There is no closed-form solution to this optimization, but we do know how to solve for either $\mathbf{z}^*$ given $\mathbf{\Phi}^*$ or $\mathbf{\Phi}^*$ given $\mathbf{z}^*$: given knowledge of the correspondences of the observed points $\mathbf{y}$ to the model, we will show in section 5.3 how to determine which model dimensions $\mathbf{z}$ are unobserved and infer maximum likelihood values for these points, completing the shape. Similarly, given knowledge of the values for both $\mathbf{y}$ and $\mathbf{z}$, the correspondence algorithm for complete contours given in section 4.2 can be applied (with slight modifications), which we describe next in section 5.2. We therefore alternately compute a local estimate $\hat{\mathbf{z}}$ given $\hat{\mathbf{\Phi}}$, then compute a local estimate $\hat{\mathbf{\Phi}}$ given $\hat{\mathbf{z}}$, which leads to the approximate, iterative procedure[8] given in table III. In practice, we have found this algorithm to converge after only a few iterations. (Note that to begin this process, we assume an initial assignment, $\hat{\mathbf{\Phi}}_0$; finding a good initial assignment is very important, which we discuss in the next section.)

In order to optimize the data associations $\mathbf{\Phi}$ given the current estimated complete shape, $\hat{\mathbf{x}} = \{\mathbf{y}, \hat{\mathbf{z}}\}$, we will use the COPAP correspondence algorithm, augmented to handle partial

[8] Note that this iterative procedure can be thought of as a "hard" expectation maximization (EM) algorithm.

observations. Recall that COPAP requires two specific shape contours as arguments, and outputs an assignment vector, $\mathbf{\Phi}$. Since our correspondence problems is from $\hat{\mathbf{x}}$ to the dimensions of the model distribution D, we use a representative shape from D within the COPAP algorithm. One possible choice is to compute correspondences from $\hat{\mathbf{x}}$ to the mean shape, μ. However, better results can be achieved on each iteration by corresponding $\hat{\mathbf{x}}$ to $\hat{\mathbf{x}}_{\perp}$—the projection of $\hat{\mathbf{x}}$ into D's eigenspace (i.e., the linear space spanned by the top-k principal components). This projected shape is often a closer match to $\hat{\mathbf{x}}$ than the mean shape is (since linear projection yields the closest shape in Euclidean distance to $\hat{\mathbf{x}}$ in D's eigenspace), resulting in more accurate point correspondences.

We note that we assume that each contour piece can be associated with an object correctly, and that all contours in $\mathbf{y}$ are (different) partial observations of a single object, O, with shape distribution D. Thus, our task is to connect the contour pieces $\mathbf{y}^{(\mathbf{1})}, \ldots, \mathbf{y}^{(\mathbf{m})}$ with hidden contour pieces, $\mathbf{z} = \{\mathbf{z}^{(\mathbf{1})}, \ldots, \mathbf{z}^{(\mathbf{m})}\}$, so that when they are connected in the right order, the points of $\mathbf{y}$ and $\mathbf{z}$ together form a single, continuous object boundary.

From the viewpoint of a computer vision algorithm which has just extracted a set of partial contours from an image, the assumption of correctly associating a contour with an object may present somewhat of a challenge. First, the grouping problem is a complex and well-studied problem in computer vision as well as in human vision. Mistakes in grouping partial observations are commonplace, and will have a substantially negative impact on the results of any shape completion resulting from such a grouping. For practical applications, a search over possible groupings may be necessary to avoid such mistakes.

5.2. Correspondences in Shape Completion

To begin the iterative optimization of the partial correspondences, we must first generate an initial correspondence vector, $\hat{\mathbf{\Phi}}_0$. In this case, we do not yet have an estimate $\hat{\mathbf{z}}$, but only a set of observed contour pieces, $\mathbf{y}^{(\mathbf{1})}, \ldots, \mathbf{y}^{(\mathbf{m})}$. Our task is to determine (1) their ordering, (2) the number of hidden points connecting each piece, and (3) the point correspondences from the ordered contour pieces (both observed and hidden) to the model, D.

We can constrain the ordering of the contours by noting that the interiors of all the observed object segments must remain on the interior of any completed shape. For most real-world cases, this topological constraint is enough to identify a unique connection ordering; in cases where the ordering of components is still ambiguous, a search process through the orderings can be used to identify the most likely correspondences.

Given a specific ordering of observed contour segments, we then add a set of hidden, or "wildcard" points connecting the partial contour segments. This forms a single, complete contour, $\mathbf{x}$, where some of the points are hidden and some are observed. We then correspond the points of $\mathbf{x}$ to the model mean shape, μ, by running a modified COPAP algorithm, where we modify the correspondence likelihood of equation (15) so that $p(\mathbf{x}_i, \mu_j)$ is uniform for all μ_j when $\mathbf{x}_i$ is unobserved; that is, all unobserved (wildcard) points required to complete the contour may be assigned to any of μ's points with zero (or minimal) cost. (We must still pay a penalty of λ for skipping hidden points, however.) We refer to this new algorithm as COPAP-PARTIAL.

```
COMPLETE-SEGMENTS($\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}\}$, $\mathbf{S}$, $R$)

Input: Set of $M$ partial shape contours (polylines) $\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}\}$, shape model $\mathbf{S}$ (with mean $\mu$), and size ratio $R$.

Output: Completed shape, $\mathbf{z}$.
  1) $(\mathbf{x}, \mathbf{h}) \leftarrow$ CONNECT$(\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}\}, R)$,
     where $\mathbf{h}_i = 1 \iff \mathbf{x}_i$ is hidden.
  2) $\Phi \leftarrow$ COPAP-PARTIAL$(\mathbf{x}, \mathbf{h}, \mu)$.
  3) Set $\mathbf{x}' \leftarrow$ ADD-SKIPPED-PTS$(\mathbf{x}, \mu, \Phi)$;
     update $\mathbf{h} \rightarrow \mathbf{h}'$.
  4) $\mathbf{z} \leftarrow$ COMPLETE-SHAPE$(\mathbf{x}', \mathbf{h}', \mathbf{S})$.
  5) $L \leftarrow$ SHAPE-LIKELIHOOD$(\mathbf{z}, \mathbf{S})$.
  6) $\mathbf{z}_{best} \leftarrow \mathbf{z}$.
  7) $L_{best} \leftarrow L$.
  8) While $L \geq L_{best}$:
       a) $\mathbf{x} \leftarrow \mathbf{z}$.
       b) $\tau_x \leftarrow$ PRESHAPE$(\mathbf{x})$.
       c) Rotate $\tau_x$ to fit $\mu$.
       d) $\mathbf{x}_{\perp} \leftarrow$ PROJECT$(\tau_x, \mu)$.
       e) SET-SKIP-COST$(\mathbf{x}_i, \infty), \forall i$ s.t. $\mathbf{h}_i = 0$
       f) $\Phi \leftarrow$ COPAP$(\mathbf{x}, \mathbf{x}_{\perp})$.
       g) $\mathbf{x}' \leftarrow$ ADD-SKIPPED-PTS$(\mathbf{x}, \mathbf{x}_{\perp}, \phi)$;
          update $\mathbf{h} \rightarrow \mathbf{h}'$.
       h) $\mathbf{z} \leftarrow$ COMPLETE-SHAPE$(\mathbf{x}', \mathbf{h}', \mathbf{S})$.
       i) $L \leftarrow$ SHAPE-LIKELIHOOD$(\mathbf{z}, \mathbf{S}, k)$.
       j) If $L > L_{best}$:
            • $\mathbf{z}_{best} \leftarrow \mathbf{z}$.
            • $L_{best} \leftarrow L$.
  9) Return $\mathbf{z} \leftarrow \mathbf{z}_{best}$.
```

TABLE III
THE PARTIAL SHAPE COMPLETION ALGORITHM, OSIRIS.

In order to identify how large the hidden portion of the contour is (and therefore, how many hidden points should be added to connect the observed contour segments), we use the insight that objects of the same type generally have a similar scale. We can therefore use the ratio of the observed object segment areas to the expected full shape area in order to (inversely) determine the ratio of hidden points to observed points. If no size priors are available, one may also perform multiple completions with varying hidden point ratios, and select the best completion using a generic prior such as the minimum description length (MDL) criterion.

In subsequent iterations of the optimization, equation (21) requires us to compute correspondences from $\hat{\mathbf{x}}$ to $\hat{\mathbf{x}}_\perp$ given a current estimate of the complete shape, $\hat{\mathbf{x}}$ (with hidden points vector $\hat{\mathbf{h}}$ indicating which of $\hat{\mathbf{x}}$'s points are hidden and which are observed). For this correspondence problem, we assume that $\hat{\mathbf{x}}$'s shape is roughly correct, and so we again disallow skipped assignments to $\hat{\mathbf{x}}$'s observed points, changing the skip cost from λ to ∞ for the observed points on $\hat{\mathbf{x}}$. (However, we use the standard correspondence likelihood equation (15) for both observed and hidden points.)

5.3. Shape Completion with Known Correspondences

One final challenge remains to complete our shape completion algorithm—namely, solving the shape completion optimization when correspondences are known:

$$\mathbf{z}^* = \arg\max_{\mathbf{z}} P_D(\mathbf{y}, \mathbf{z}|\mathbf{\Phi}). \tag{22}$$

As noted in section 5.1, this optimization problem is extremely non-linear, since transforming $\mathbf{x} = \{\mathbf{y}, \mathbf{z}\}$ into a pre-shape τ by normalizing scale and position requires knowledge of the position of the hidden points, $\mathbf{z}$, and rotating and projecting τ into D's tangent space requires knowledge of this pre-shape. Translation and projection are both linear operations, so the primary sources of non-linearity in equation (22) are:

1) scaling $\mathbf{x}$ onto the pre-shape sphere, and
2) rotating the resulting pre-shape, τ to match the model mean shape, μ.

D is modeled as a Gaussian distribution (in μ's tangent space), therefore any linear transformation of D will also be Gaussian. In other words, if $g(\mathbf{x})$ is a linear function of $\mathbf{x}$, then $P_D(g(\mathbf{x}))$ is a Gaussian likelihood function of $\mathbf{x}$, and therefore a maximum value $\mathbf{x}^*$ of $P_D(g(\mathbf{x}))$ can be found in closed form. Thus, given a fixed rotation factor, θ and scaling factor, α, equation (22) can be maximized in closed form.

For every pair (θ, α), there exists a corresponding $\mathbf{z}^*$ which achieves the maximum likelihood,

$$\zeta(\theta, \alpha) = \max_{\mathbf{z}} P_D(\mathbf{y}, \mathbf{z}|\mathbf{\Phi}, \theta, \alpha). \tag{23}$$

Thus, the shape completion with known correspondences optimization problem can be reduced to the 2-D optimization,

$$(\theta^*, \alpha^*) = \arg\max_{\theta,\alpha} \zeta(\theta, \alpha). \tag{24}$$

While this is still a non-linear optimization, we have reduced the dimensionality of the problem from $2(n-p)$ to 2, where p is the number of observed points, and n is the total number of points on the contour $\mathbf{x}$—a significant improvement.

Any number of non-linear optimization methods can be used to solve equation (24) for θ^* and α^*. Here we use a simple sampling technique to arrive at initial estimates, $\hat{\theta}^*$ and $\hat{\alpha}^*$. If necessary, an iterative method such as gradient descent or simulated annealing can be used to refine these estimates further.

To make this concrete, we assume that correspondences $\mathbf{\Phi}$ have already been applied to $\mathbf{x} = \{\mathbf{y}, \mathbf{z}\}$, and that $\mathbf{y}$ contains the first p points of contour $\mathbf{x}$, which are observed, and $\mathbf{z}$ contains the $n-p$ unknown points that complete the shape. (Note that the dimensions of the distribution mean and covariance can be permuted so that $\mathbf{y}$ and $\mathbf{z}$ correspond to the beginning p and final $n-p$ points of the model, respectively.) Then, we can write

$$\mathbf{x} = [\mathbf{y}\ \mathbf{z}]^T. \tag{25}$$

Given shape distribution D on n points with mean μ and covariance matrix Σ, and fixed orientation θ and scale α, we derive $\mathbf{z}$ in the following manner.

For a complete contour $\mathbf{x}$, we normalize for orientation and scale using

$$\mathbf{x}' = \frac{1}{\alpha} R_\theta \mathbf{x} \tag{26}$$

where R_θ is the rotation matrix of θ. To center $\mathbf{x}'$, we then subtract off the centroid:

$$\mathbf{w} = \mathbf{x}' - \frac{1}{n} C\mathbf{x}' \tag{27}$$

where C is the $2n \times 2n$ checkerboard matrix[9],

$$C = \begin{bmatrix} 1 & 0 & \cdots & 1 & 0 \\ 0 & 1 & \cdots & 0 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & \cdots & 1 & 0 \\ 0 & 1 & \cdots & 0 & 1 \end{bmatrix}. \tag{28}$$

Thus $\mathbf{w}$ is the centered pre-shape. Now let M be the matrix that projects into the tangent space defined by the Gaussian distribution (μ, Σ):

$$M = I - \mu\mu^T \tag{29}$$

The Mahalanobis distance with respect to D from $M\mathbf{w}$ to the origin in the tangent space is:

$$d_\Sigma = (M\mathbf{w})^T \Sigma^{-1} M\mathbf{w}. \tag{30}$$

Minimizing d_Σ is equivalent to maximizing $P_D(\cdot)$, so we continue by setting $\frac{\partial d_\Sigma}{\partial \mathbf{z}}$ equal to zero, and letting

$$W_y = M_y(I_y - \frac{1}{n}C_y)\frac{1}{\alpha}R_\theta^y \tag{31}$$

$$W_z = M_z(I_z - \frac{1}{n}C_z)\frac{1}{\alpha}R_\theta^z \tag{32}$$

where the subscripts "y" and "z" indicate the left and right sub-matrices of M, I, and C that match the dimensions of $\mathbf{y}$ and $\mathbf{z}$. This yields the following system of linear equations which can be solved for the missing data, $\mathbf{z}$:

$$(W_y\mathbf{y} + W_z\mathbf{z})^T \Sigma^{-1} W_z = 0. \tag{33}$$

Equation (33) holds for fixed orientation, θ and scale, α. To design a sampling method for θ and α, we match the partial shape, $\mathbf{y}$, to the partial mean shape, μ_y, by computing the pre-shapes of $\mathbf{y}$ and μ_y and finding the Procrustes fitting rotation, θ_y^*, from the pre-shape of $\mathbf{y}$ onto the pre-shape of

[9] Recall that we represent a shape $\mathbf{x}$ as a vector of n ordered 2-D points, $[\mathbf{x}_1^T \mathbf{x}_2^T \cdots \mathbf{x}_n^T]$, so that $\mathbf{x} \in \mathbb{R}^{2n}$. Thus, multiplying a shape vector $\mathbf{x}$ by the checkerboard matrix C simply adds up the x- and y- coordinates of $\mathbf{x}$'s points.

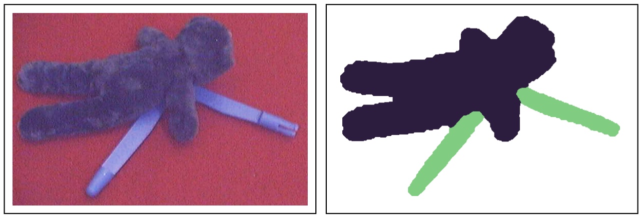


Fig. 9. An example of occluded objects, where the bear occludes the compass. (a) The original image and (b) the image segmented into (unknown) objects. The contour of each segment must be matched against a known model.

μ_y. This angle can then be used as a mean for a von Mises distribution (the circular analog of a Gaussian) from which to sample orientations. Similarly, we can sample scales from a Gaussian with mean α_y, the ratio of scales of the partial shapes $\mathbf{y}$ and μ_y, as in

$$\alpha_y = \frac{\|\mathbf{y} - \frac{1}{p}C_y\mathbf{y}\|}{\|\mu_y - \frac{1}{p}C_y\mu_y\|}. \tag{34}$$

Any sampling method for shape completion will have a *scale bias*—completed shapes with smaller scales project to a point closer to the origin in tangent space, and thus have higher likelihood (since our probability model for shapes is a zero-mean Gaussian in tangent space). One way to fix this problem is to solve for $\mathbf{z}$ by performing a constrained optimization on d_Σ where the scale of the centered, completed shape vector is constrained to have unit length:

$$\|\mathbf{x}' - \frac{1}{n}C\mathbf{x}'\| = 1. \tag{35}$$

However, this constraint yields a much more difficult non-linear optimization. Furthermore, in our experiments this scale bias has not appeared to provide any obvious errors in shape completion, although more testing and analysis are needed to determine the precise effect of the scale bias on the quality of shape completions.

5.4. *Partial Shape Classification*

The partial shape classification problem is

$$c^* = \arg\max_c P(C = c|\mathbf{y}) \tag{36}$$

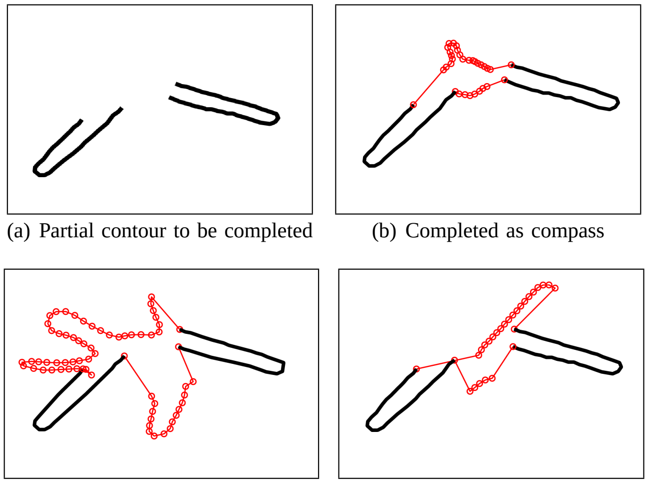
where

$$P(C|\mathbf{y}) = \frac{P(C, \mathbf{y})}{P(\mathbf{y})} \propto \int P(C, \mathbf{y}, \mathbf{z}) d\mathbf{z} \tag{37}$$

Marginalizing over the hidden data, $\mathbf{z}$, is computationally infeasible, so we approximate this marginal with the estimate $\hat{\mathbf{z}}$, the output of our shape completion algorithm, yielding:

$$P(C|\mathbf{y}) \approx \eta \cdot P(\mathbf{y}, \hat{\mathbf{z}}|C) \tag{38}$$
$$= \eta \cdot P_D(\mathbf{y}, \hat{\mathbf{z}}) \tag{39}$$

where η is a normalizing constant, D is the Gaussian tangent-space shape model of class C, and thus $P_D(\mathbf{y}, \hat{\mathbf{z}})$ is the complete shape class likelihood of the completed shape with respect to class C.



(a) Partial contour to be completed

(b) Completed as compass

(c) Completed as stuffed animal

(d) Completed as jump rope

Fig. 10. Shape completion of the partial contour of the compass in figure 9. Note that the correct completion (b) captures the knob in the top of the compass. The hypothesized completions in (c) and (d) lead to very unlikely shapes.

5.5. *Extensions*

As noted above, we have assumed throughout this discussion that a single contour representation is appropriate in modeling all object boundaries. This assumption can be relaxed, since multiple-contour representations can be handled with a search over partitions of the contour pieces $\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(m)}$. A second simplification was to hold the observations, $\mathbf{y}$, fixed in the shape completion algorithm—however, one could easily incorporate Gaussian observation likelihoods $P_{obs}(\cdot)$ into equation (20), solving for both $\mathbf{y}$ and $\mathbf{z}$, as in $(\mathbf{y}^*, \mathbf{z}^*) = \arg\max_{\mathbf{y},\mathbf{z}} P_D(\mathbf{y}, \mathbf{z}) P_{obs}(\mathbf{y}|\mathbf{y}_{obs})$.

In our presentation, we have focused on modeling the inherent shape variability in a class, however perspective transformations can also be included in our shape model, either during the training phase (by including multiple camera angles in the training data) or by including a perspective term in the search for optimal scales and orientations during shape completion (equation (23)).

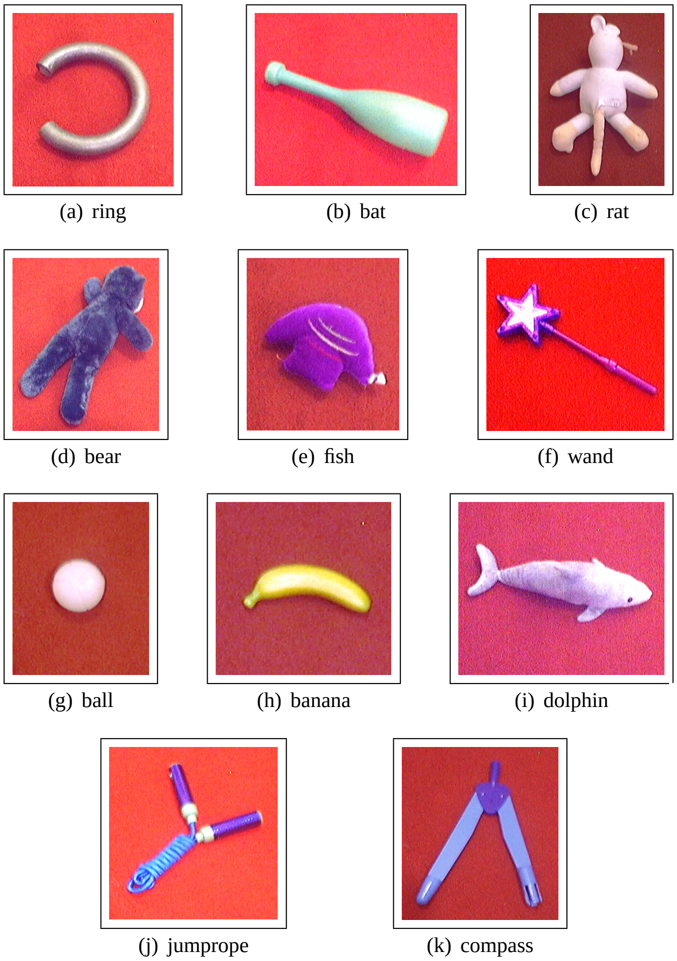
Finally, in some cases it may be possible to take “negative” information into account during the classification of partial contours. For example, in figure 9, it would be unlikely that the correct completion of the compass object would extend onto areas of the image labeled as “background” by the image processor (which is colored white in the segmented image)—thus, the completion as a “stuffed animal” in figure 10(c) should be given less likelihood than the other completions.

One can incorporate negative information by adding an image likelihood term to equation (39), as in $P(C|\mathbf{y}) \approx \eta \cdot P_D(\mathbf{y}, \hat{\mathbf{z}}) \cdot P_{image}(\hat{\mathbf{z}})$.

6. Results

6.1. *Toys Dataset*

For our grasping experiments, we generated a dataset of 11 toy shape classes. To learn shape models, we collected 10



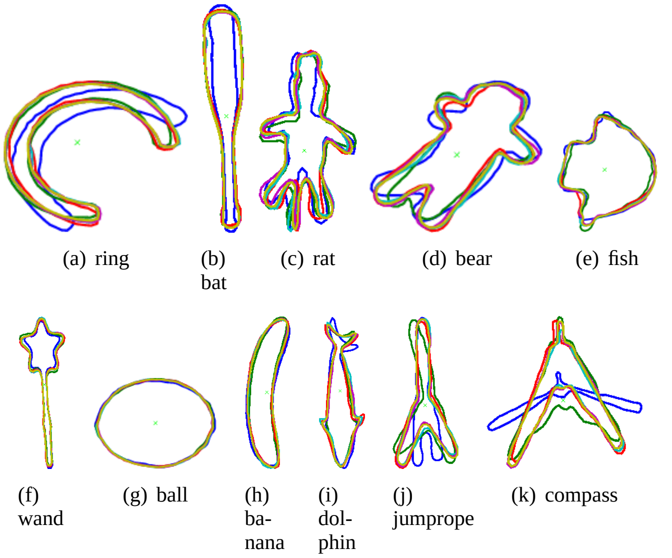
(a) ring (b) bat (c) rat (d) bear (e) fish (f) wand (g) ball (h) banana (i) dolphin (j) jumprope (k) compass

Fig. 11. Examples of one shape from each of the 11 classes of toys. Several of the classes contain objects which deform, either because of articulations (compass, jumprope) or because of the object's soft material (rat, bear, fish, dolphin). Two other classes (ring, bat) contain multiple instances of rigid objects.

images of each object type, segmented the object contours from the background using color thresholding (we learned simple 1- and 2-color models for each object which we calibrated at the start of each experiment), and used the shape distribution learning algorithm of section 4.6 to build probabilistic shape models for each class, using contours of 100 points each. One example object from each class is seen in figure 11.

We reduced the dimensionality of the covariance in each class using PCA. Reducing the covariance to three principal components led to 100% prediction accuracy of the training set, and 98% cross-validated ($k = 5$) prediction accuracy. In figure 12, we show the effects of the top 3 principal components on the mean shape for each class.

We then generated a test set of 880 simulated partial shape observations by occluding our training shapes with randomly-placed rectangles of varying sizes and orientations. Using cross-validation, we obtained estimates of the classification and detection rates on our partial shape dataset as a function of the percentage of occluded points on each shape contour (figure 13). The detection rate was nearly 90% even when 70% of the shape was occluded from view, and at 80% occlusion the classification rate was 47%, which is still much better than random guessing ($1/11 \approx 9\%$).



(a) ring (b) bat (c) rat (d) bear (e) fish (f) wand (g) ball (h) banana (i) dolphin (j) jumprope (k) compass

Fig. 12. Shape models learned for each of the object classes. The red contours are the mean shapes, and the others are sampled along each of the top three eigenvectors.

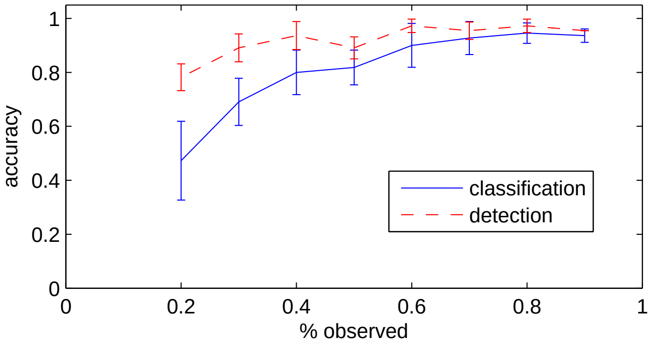


Fig. 13. Cross-validated classification and detection rates as a function of occlusion. The solid blue line shows the percentage of times a partial shape was correctly classified, while the red dotted line shows the percentage of trials in which the correct class was given at least a 5% probability. Note that the detection rate is nearly 90% even when 70% of the shape is occluded from view, and at 80% occlusion the classification rate is 47%, which is still much better than random guessing ($1/11 \approx 9\%$).

In table IV, we show classification and detection results from our manipulation experiments in section 7.

6.2. *MPEG-7 Results*

In addition to the experiments we performed on images of real objects, we also wished to explore the performance of our shape recognition algorithms on a more complex dataset of synthetic objects. For this purpose we chose a subset of 20 classes (out of 70) from the MPEG-7 shape dataset. Examples from each of the 20 classes are shown in figure 14. We used

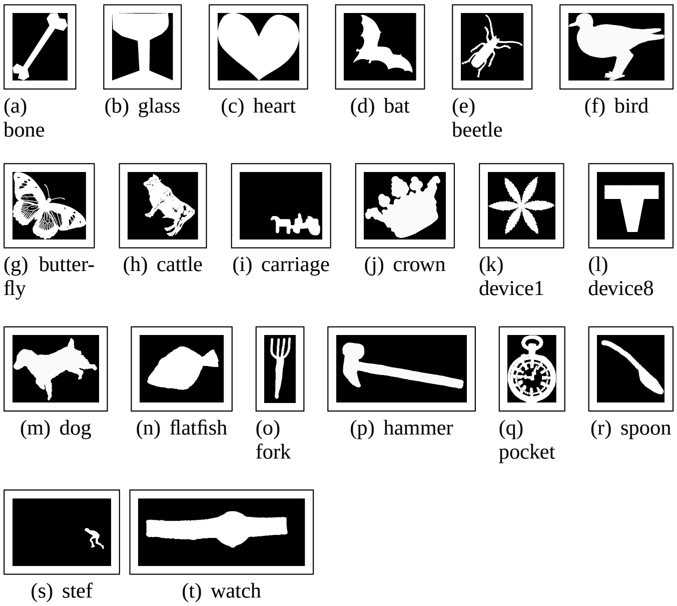

(a) bone (b) glass (c) heart (d) bat (e) beetle (f) bird (g) butterfly (h) cattle (i) carriage (j) crown (k) device1 (l) device8 (m) dog (n) flatfish (o) fork (p) hammer (q) pocket (r) spoon (s) stef (t) watch

Fig. 14. One shape from each of the 20 classes of our subset of the MPEG-7 dataset.

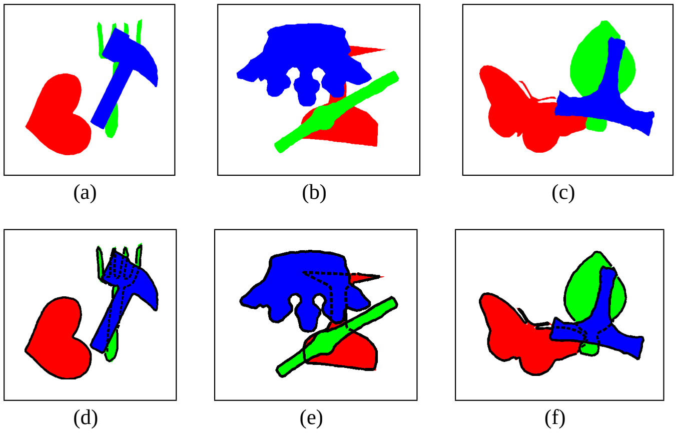

(a) (b) (c) (d) (e) (f)

Fig. 15. Top row: synthetic pile images generated from the MPEG-7 dataset. Bottom row: occluded shape boundaries completed with OSIRIS, with respect to the most likely object class. All of the partially occluded shapes in these three images were correctly classified.

16 (out of 20) randomly chosen images in each class to train probabilistic shape models, once again using contours of 100 points each. We used 6 principle components in the PCA for each class, due to the added complexity in the MPEG-7 dataset compared with the toys dataset. The other four images in each class were held out as a test set.

From this test set, we generated 100 synthetic images containing three randomly placed (and randomly selected) overlapping shapes, as in the top row of figure 15. In the bottom row, we show the completed contours of the partially occluded shapes in each scene, as computed by OSIRIS.

Object	Partial	Complete
ring	3/8	15/15
bat	7/10	8/10
rat	9/13	4/4
bear	7/7	7/7
fish	9/9	6/6
banana	-	1/2
dolphin	1/2	-
compass	1/3	5/5
totals	37/52	46/49
	71.15%	93.88%
detect $> 5\%$	42/52	48/49
	80.77%	97.96%

TABLE IV
CLASSIFICATION RATES ON TOYS TEST SET.

	Partial	Complete
correct class	99/134	160/166
	73.88%	96.39%
detect $> 5\%$	110/134	164/166
	82.09%	98.80%

TABLE V
CLASSIFICATION RATES ON MPEG-7 TEST SET.

In table V, we show the classification and detection results on the 300 objects from the synthetic MPEG-7 pile images. Complete (fully observed) shapes were classified correctly 96.39% of the time, while partially occluded shapes were classified correctly at a rate of 73.88%. The $> 5\%$ detection rate (i.e. the percentage of objects for which the algorithm gave at least 5% likelihood to the correct class) for complete shapes was 98.80%, while the detection rate for partial shapes was 82.09%. However, further inspection reveals that most of the incorrect classifications and detections came from a single class–"spoon". If results from this class are left out, the classification rates are vastly improved: 97.48% for full shapes and 80.49% for partial shapes. (The detection rates are similarly improved to 99.37% for full shapes and 89.43% for partial shapes.)

6.3. Discussion

In both the toys and MPEG-7 experiments, the full shape classification and detection rates were quite good (well above 90%), which validates our model learning algorithm from section 4.6. On shape completion with OSIRIS and partial shape recognition, our initial results are promising, yet there are some simple techniques that could be applied to improve upon the existing system. We have already discussed some of these techniques, such as using negative information about where the object is *not* (in addition to the positive observations of the un-occluded portions of the object boundary). Other techniques will require a bit more effort to integrate into the system, such as maintaining multiple correspondence hypotheses when there is no single optimal point matching between two shapes (OSIRIS currently is susceptible to getting stuck in local maxima on the completion likelihood manifold).

The utter failure to correctly classify partial shapes in the "spoon" class from the MPEG-7 dataset demonstrates a weakness in our partial shape classification formula in that it struggles to classify objects which have smooth, highly deformable contours. One reason is that by replacing the marginal density over the hidden points, $\mathbf{z}$ in equation (37), with the likelihood of the maximum likelihood completion, $\hat{\mathbf{z}}$ in equation (39), there is a bias towards less "peaky" shape distributions; that is, towards shape classes containing less variation. Since the "spoon" class contains both a great deal of shape variability, as well as a lack of distinctive boundary features to match (the contours are very smooth), our algorithm typically finds high-likelihood completions for partially-observed spoons with respect to several shape classes, and it chooses to classify the spoon as an instance of a class which contains less variability than the spoon class contains. In order to handle these smooth, highly deformable shape classes, a better approximation will need to be found to the marginal density in equation (37).

Finally, it should be noted that we selected the 20 class subset from the MPEG-7 dataset as a representative sample of the types of objects our algorithm is designed to model. Many of the classes which we left out either contained very little shape variation, or contained types of variation which our algorithm was not intended to handle, such as deep cuts into the contours (which we have found to be best handled by image scale space techniques), or vastly different views of complex, 3-D objects containing multiple articulations and self-occlusions.

7. Grasp Planning

Our manipulation strategy is a pipelined process: first, we estimate the complete geometric structure of the scene and then plan a grasp. But before we can decide how an individual object is grasped, we must first decide *which* object to grasp. The problem domains of primary interest—such as the "box-of-toys" world of figure 1—are domains with a single "desired" object or object type; for example, a teddy bear. Thus, our ultimate goal is to retrieve a specific object or type of object from the scene. Sometimes, the desired object will be at the top of the pile, fully in view. In this case, after analyzing the image and recognizing the object, we will be able to plan a grasp to retrieve the object, irrespective of the placement of other objects in the scene. However, if the desired object is occluded, before attempting to pick it up, we must determine the probability that the sensed object is actually the desired object, and the probability that a planned grasp on the accessible part of the object will be successful. If either of these probabilities are below a pre-determined threshold, we first remove one or more occluding objects and then re-analyze the scene before planning a grasp of the desired object. We implement the first test as a threshold on the class likelihood of the sensed object, $p(C_i|\mathbf{m})$; the second test is a function of our strategy for grasping a single object, described below. Our proposed manipulation process is given in algorithm 1.

Algorithm 1 The Manipulation Process.

```
Require: An image of a scene, and learned models of objects
Segment the image into object components
Extract contours of components
Determine maximum-likelihood correspondence between observed contours and known models
Infer complete geometry of each object from matched contours
Return planned grasp strategy based on inferred geometries
```

7.1. Grasping a Single Object

We have developed a grasp planning system for our mobile manipulator (shown in figure 16), a two-link arm on a mobile base with an in-house-designed gripper with two opposable fingers. Each finger is a structure capable of edge and surface contact with the object to be grasped.

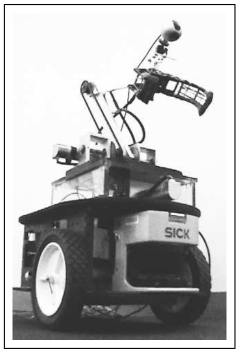
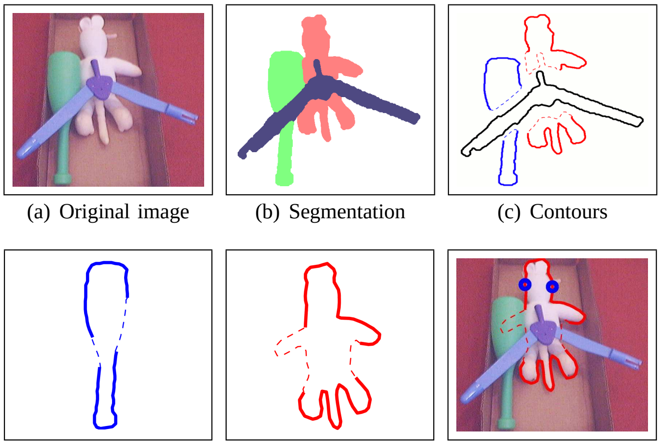


Fig. 16. Our mobile manipulator with a two link arm and gripper. We use a simple webcam mounted on the gripper to capture images of the objects in front of the robot.

The input to the grasp planning system is the object geometry with the partial contours completed as described in Section 5. The output of the system is two regions, one for each finger of the gripper, that can provide an equilibrium grasp for the object following the algorithms for stable grasping described by Nguyen (1989). Intuitively, the fingers are placed on opposing edges so that the forces exerted by the fingers can cancel each other out. Friction is modeled as Coulomb friction with empirically estimated parameters. The grasp planner is implemented as a search for a pair of grasping edges that yield maximal regions for the two grasping fingers using the geometric conditions derived by Nguyen (1989). If two edges can be paired such that their friction cones are overlapping, we then identify maximal regions for placing the fingers so that we can tolerate maximal uncertainty in the finger placement using Nguyen's criterion. If the desired object is fully observed, we can use the above grasping algorithm unchanged. If it is partially occluded, we must filter out finger placements which lie on hidden (inferred) portions of the object's boundary. If, after filtering out infeasible grasps, there is still an accessible grasp of sufficient quality according to Nguyen's criterion, we can attempt a grasp of the object.

In figures 17 and 18 we show the results of two manipulation experiments, where in each case we seek to retrieve a single type of object from a box of toys, and we must locate and grasp this object while using the minimum number of object grasps possible. In both cases, the object we wish to retrieve is occluded by other objects in the scene, and so a naïve grasping strategy would first remove the objects on top of the desired object until the full object geometry is observed, and only then would it attempt to retrieve the object. Using the inferred geometry of the occluded object boundaries to classify and plan a grasp for the desired object, we find in both cases that we are able to grasp the object immediately, reducing the number of grasps required from 3 to 1. In addition, we are able to successfully complete and classify the other objects in each scene, even when a substantial portion of their boundaries is occluded. The classification of this test set of 7 object contours (from 6 objects classes) was 100% (note the correct completions in figures 17 and 18 of the occluded objects).

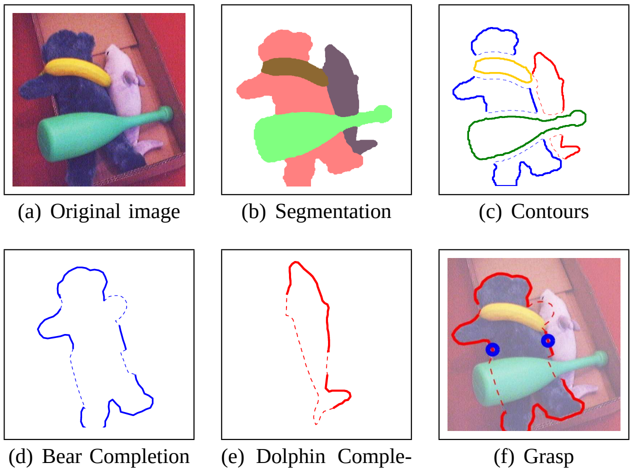

(a) Original image (b) Segmentation (c) Contours

(d) Bat Completion (e) Rat Completion (f) Grasp

Fig. 17. An example of a very simple planning problem involving three objects. The chalk compass is fully observed, but the stuffed rat and green bat are partially occluded by the compass. After segmentation (b), the image decomposes into five separate segments shown in (c). The learned models of the bat and the rat can be completed (d) and (e), and the complete contour of the stuffed rat is correctly positioned in the image (f). The two blue circles correspond to the planned grasp that results from the computed geometry.

For a more thorough evaluation, we repeated the same type of experiment on 20 different piles of toys. In each test, we again sought to retrieve a single type of object from the box of toys, and in some cases, the manipulation algorithm required several grasps in order to successfully retrieve an object, due to not being able to find the object right away or because the occluding objects were blocking access to a stable grasp of the desired object. Figures 19 and 20 show 2 of the 20 trials in our experiment. Both trials are examples in which it took the robot more than one grasp to retrieve the desired object.

In figure 19, the object to be retrieved is the purple fish, which is initially occluded by the green bat. After segmentation and contour completions, the algorithm is able to recognize the fish (figure 19(c)), but it realizes that the bat is in the way, and so it plans a grasp of the bat (figure 19(d)) and removes it. This time, the fish is again completed (figure 19(h)) and successfully classified as a fish, and a grasp is planned and executed (figure 19(i)). All contours all correctly classified throughout the experiment.


(a) Original image (b) Segmentation (c) Contours

(d) Bear Completion (e) Dolphin Completion (f) Grasp

Fig. 18. A more complex example involving four objects. The blue bat and the yellow banana are fully observed, but the stuffed bear and dolphin are significantly occluded. After segmentation (b), the image decomposes into seven separate segments shown in (c). The learned models of the bear and the dolphin can be completed (d) and (e), and the complete contour of the stuffed bear is correctly positioned in the image (f). The two blue circles correspond to the planned grasp given the geometry.

In figure 20, the object to be retrieved is the yellow ring, which is initially occluded by both the blue bear and the green bat. After segmentation and contour completions, the algorithm is able to recognize the ring (figure 20(d)), but it realizes that it must remove the bat before it can access the ring, so it plans a grasp of the bat (figure 20(g)) and removes it. This time, the ring is again correctly completed and identified (figure 20(k)), and a grasp is executed (figure 20(l)), but fails due to the weight of the bear lying on top of the ring. After another round of image analysis, the ring is successfully retrieved (figure 20(p)). Note that the rat was misclassified as a bear throughout this experiment; however this classification error had no effect on the retrieval of the ring.

In total, 52 partial and 49 complete contours were classified, 33/35 grasps were successfully executed (with 3 failures due to a hardware malfunction which were discounted). In table IV, we show classification rates for each class of object present in the images. Partially-observed shapes were correctly classified 71.15% of the time, while fully-observed shapes were correctly classified 93.88% of the time. Several of the errors were simply a result of ambiguity—when we examine the $> 5\%$ detection rates (i.e. the percentage of objects for which the algorithm gave at least 5% likelihood to the correct class), we see an improvement to 80.77% for partial shapes, and 97.96% for full shapes. While a few of the detection errors were from poor or noisy image segmentations, most were from failed correspondences from the observed contour to the correct shape model. The most common reason for these failed correspondences was a lack of local features for the COPAP algorithm to latch onto with the PLSD point assignment cost. These failures would seem to argue for a combination of local and global match likelihoods in the correspondence algorithm, which is a direction we hope to explore in future work.

8. Conclusions

In this paper, we used the Procrustean shape metric–which is invariant to position, scale, and orientation–to develop dense probabilistic models of object geometry, based on tangent space principle components analysis. In section 4.2, we presented a probabilistic model for shape correspondences based on the Cyclic Order-Preserving Assignment Problem (COPAP) framework, with a new likelihood model for local shape similarity based on the Procrustean Local Shape Distance (PLSD). We then used COPAP to train probabilistic shape models from datasets with unknown correspondences. In section 5, we presented the OSIRIS algorithm, which uses learned Procrustean shape models to infer the hidden parts of partially-occluded, deformable objects. Finally, in section 7, we applied our shape inference algorithms to the task of robotic grasping, where we demonstrated that our learned models allow us to efficiently recognize and retrieve complex objects from a pile of toys in the presence of sensor noise and occlusions. We also presented results of the OSIRIS algorithm on synthetic images generated from the toys dataset as well as the MPEG-7 dataset.

In order to extend our algorithms to process models of views of 3-D objects which contain multiple articulations and self-occlusions, it may be useful to combine a skeleton or parts-based models with our global parametric models in order to achieve robustness to these highly variable shapes. Another set of object classes which are currently problematic are those containing varying numbers of protrusions, such as the “beetle” class in the MPEG-7 dataset. (Some beetle contours have six legs while other have eight.) While the alignment penalty λ in COPAP encourages the smoothing out of assignments, the correct correspondence of two shapes with protrusions may be to skip over large portions of the contours. This is because such portions of contours contain a disproportionate number of points in comparison with the ratio of area of the protrusion to the area of the entire shape.

In future work, we hope to demonstrate improved performance on recognition tasks by incorporating additional priors into the correspondence and completion models, in order to bias the inference procedure towards smoother, more natural correspondences and completions. We would also like to investigate the use of other probability densities for modeling shapes in shape space. While tangent space PCA has many benefits–including simplicity–many of the shape classes we have encountered in our work would be better modeled by a multi-modal distribution, such as a Gaussian mixture model or a complex Bingham. Additionally, we expect that the sequential model learning algorithm of section 4.6 could be vastly improved by using a hierarchical model merging algorithm, which would merge similar shapes until merging no longer improves the model; such a technique would lend itself quite nicely to generating mixture models for shape densities.

9. Acknowledgments

Jared Glover and Nicholas Roy were supported by the National Science Foundation Division of Information and Intelligent Systems under grant # 0546467 and the Air Force Office of Scientific Research under STTR Contract FA9550-06-C-0088. Nicholas Roy and Daniela Rus were supported by the National Science Foundation Division of Computer and Network Systems under grant # 0707601. Daniela Rus was supported by the National Science Foundation under grants # 0426838 and 0735953.

References

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence* 24(24): 509–522.

Berge, J. M. F. T. (1977). Orthogonal procrustes rotation for two or more matrices. *Psychometrika* 42(2): 267–276.

Blake, A. and Isard, M. (1998). *Active Contours*. Springer-Verlag.

Bone, G. M. and Du, Y. (2001). Multi-metric comparison of optimal 2d grasp planning algorithms. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*.

Bookstein, F. (1984). A statistical method for biological shape comparisons. *Theoretical Biology* 107: 475–520.

Cootes, T., Taylor, C., Cooper, D., and Graham, J. (1995). Active shape models–their training and application. *Computer Vision and Image Understanding* 61(1): 38–59.

Cremers, D., Kohlberger, T., and Schnorr, C. (2003). Shape statistics in kernel space for variational image segmentation. *Pattern Recognition* 36(9): 1929–1943.

Dryden, I. and Mardia, K. (1998). *Statistical Shape Analysis*. John Wiley and Sons.

Elidan, G., Heitz, G., and Koller, D. (2006). Learning object shape: From drawings to images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*.

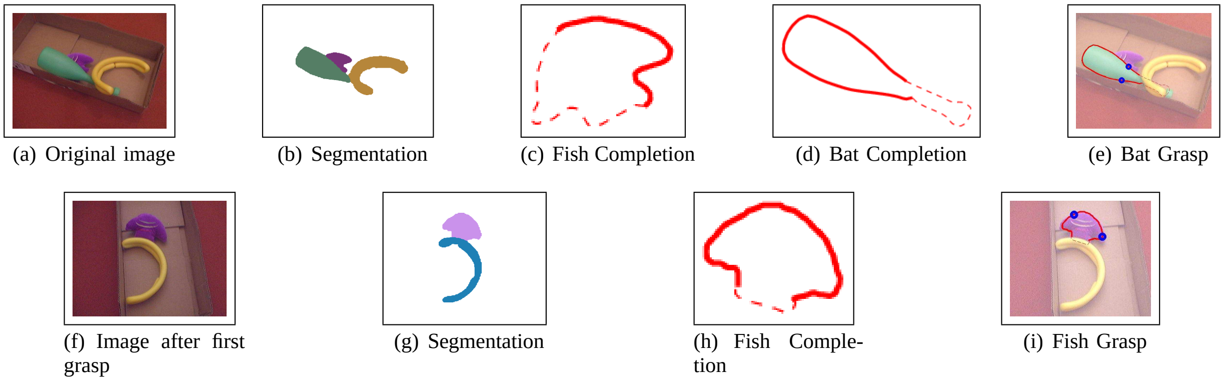
Felzenszwalb, P. (2005). Representation and detection of deformable shapes. *IEEE Trans. Pattern Analysis and Machine Intelligence* 27(2).

Glover, J., Rus, D., and Roy, N. (2006). Probabilistic models of object geometry for grasp planning. In *Proceedings of Robotics: Science and Systems (RSS 2008)*. Zurich, Switzerland.

Gottschalk, P. G., Turney, J. L., and Mudge, T. N. (1989). Efficient recognition of partially visible objects using a logarithmic complexity matching technique. *International Journal of Robotics Research* 8(6): 110–131.

Gower, J. C. (1975). Generalized procrustes analysis. *Psychometrika* 40(1): 33–51.

Grimson, W. E. L. and Lozano-Pérez, T. (1987). Localizing overlapping parts by searching the interpretation tree. *IEEE*



(a) Original image

(b) Segmentation

(c) Fish Completion

(d) Bat Completion

(e) Bat Grasp

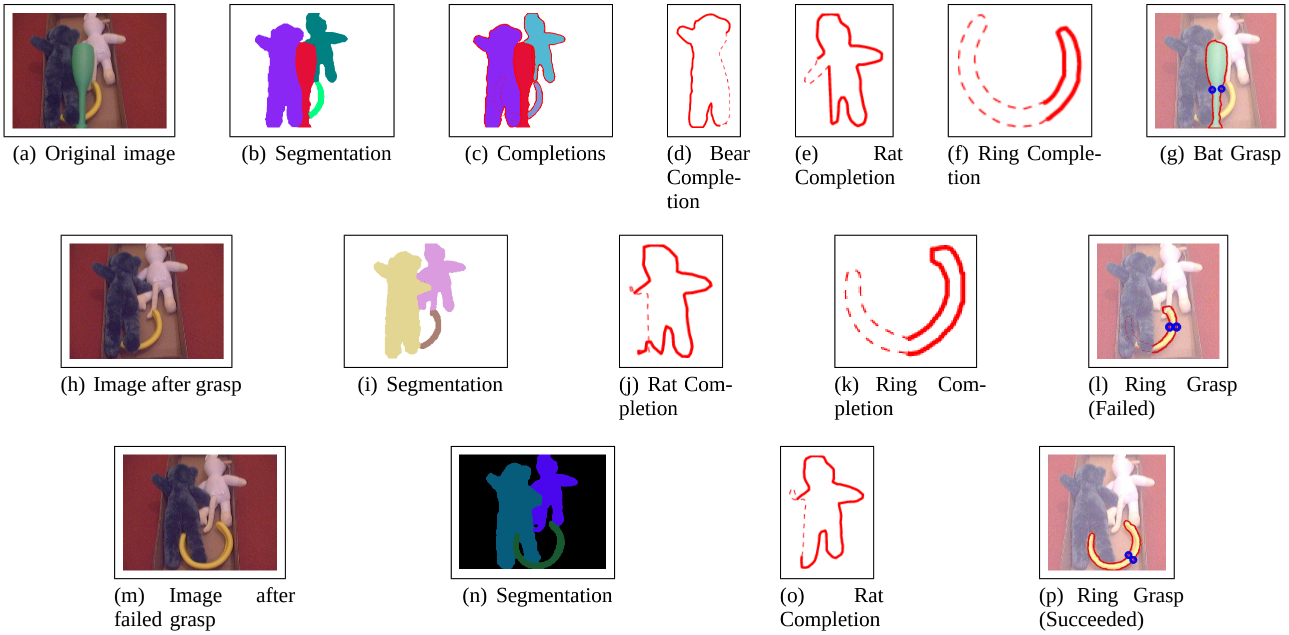
(f) Image after first grasp

(g) Segmentation

(h) Fish Completion

(i) Fish Grasp

Fig. 19. Example of an experiment where more than one grasp is required to retrieve the desired object (the purple fish). The robot is able to immediately recognize the fish, but it is required to first remove the bat before a good grasp of the fish is available. Note that the vehicle base moves in order to complete the first grasp, so the perspective of the container changes after the first grasp.



(a) Original image

(b) Segmentation

(c) Completions

(d) Bear Completion

(e) Rat Completion

(f) Ring Completion

(g) Bat Grasp

(h) Image after grasp

(i) Segmentation

(j) Rat Completion

(k) Ring Completion

(l) Ring Grasp (Failed)

(m) Image after failed grasp

(n) Segmentation

(o) Rat Completion

(p) Ring Grasp (Succeeded)

Fig. 20. In this experiment, three grasps were necessary to retrieve the desired object (the yellow ring). The robot is able to immediately recognize the yellow ring, but it is required to first remove the bat and must try to pick up the ring twice before a successful grasp is achieved.

Trans. Pattern Anal. Mach. Intell. 9(4): 469–482. ISSN 0162-8828.

Hu, M. K. (1962). Visual pattern recognition by moment invariants. In *IRE Transactions on Information Theory*, volume IT-8, 179187.

Johnson, A. and Hebert, M. (1999). Using spin images for efficient object recognition in cluttered 3-d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 21(5): 433 – 449.

Katz, D. and Brock, O. (2008). Manipulating articulated objects with interactive perception. In *In Proceedings of the IEEE International Conference on Robotics and Automation ICRA*.

Kendall, D. (1984). Shape manifolds, procrustean metrics, and complex projective spaces. *Bull. London Math Soc.* 16: 81–121.

Kendall, D., Barden, D., Carne, T., and Le, H. (1999). *Shape and Shape Theory*. John Wiley and Sons.

Koch, M. W. and Kashyap, R. L. (1987). Using polygons to recognize and locate partially occluded objects. *IEEE Trans. Pattern Anal. Mach. Intell.* 9(4): 483–494. ISSN 0162-8828.

Kristof, W. and Wingersky, B. (1971). Generalization of the orthogonal procrustes rotation procedure to more than two matrices. In *Proceedings, 79th Annual Convention, APA*, 89–90.

Latecki, L. J., Lakämper, R., and Eckhardt, U. (2000). Shape descriptors for non-rigid shapes with a single closed contour. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 424–429.

Lin, C. C. and Chellappa, R. (1987). Classification of partial

2-d shapes using fourier descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* 9(5): 686–690. ISSN 0162-8828.

Maes, M. (1990). On a cyclic string-to-string correction problem. *Inf. Process. Lett.* 35(2): 73–78.

McInerney, T. and Terzopoulos, D. (1996). Deformable models in medical images analysis: a survey. *Medical Image Analysis* 1(2): 91–108.

Mirtich, B. and Canny, J. (1994). Easily computable optimum grasps in 2-d and 3-d. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*.

Mokhtarian, F. and Mackworth, A. K. (1992). A theory of multiscale curvature-based shape representation for planar curves. In *IEEE Trans. Pattern Analysis and Machine Intelligence*, volume 14.

Nguyen, V.-D. (1989). Constructing stable grasps. *I. J. Robotic Res.* 8(1): 26–37.

Pollard, N. S. (1996). Synthesizing grasps from generalized prototypes. In *Proceedings of the International Conference on Robotics and Automation*.

Saxena, A., Driemeyer, J., Kearns, J., Osondu, C., and Ng, A. Y. (2006). Learning to grasp novel objects using vision. In *Proc. International Symposium on Experimental Robotics (ISER)*.

Scott, C. and Nowak, R. (2006). Robust contour matching via the order preserving assignment problem. *IEEE Transactions on Image Processing* 15(7): 1831–1838.

Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Trans. Pattern Analysis and Machine Intelligence* 22(8): 888–905.

Shimoga, K. B. (1996). Robot grasp synthesis algorithms: A survey. *International Journal of Robotics Research* 15(3): 230–266.

Small, C. G. (1996). *The statistical theory of shape*. Springer.