Affine-Permutation Invariance of 2-D Shapes

Victor H. S. Ha, *Member, IEEE,* and José M. F. Moura, *Fellow, IEEE*

***Abstract*—Shapes provide a rich set of clues on the identity and topological properties of an object. In many imaging environments, however, the same object appears to have different shapes due to distortions such as translation, rotation, reflection, scaling, or skewing. Further, the order by which the object's feature points are scanned changes, i.e., the order of the pixels may be permuted. Relating two-dimensional shapes of the same object distorted by different *affine* and *permutation* transformations is a challenge. We introduce a shape invariant that we refer to as the *intrinsic shape* of an object and describe an algorithm, BLAISER, to recover it. The intrinsic shape is invariant to affine-permutation distortions. It is a uniquely defined representative of the equivalence class of all affine-permutation distortions of the same object. BLAISER computes the intrinsic shape from any arbitrarily affine-permutation distorted image of the object, without prior knowledge regarding the distortions or the undistorted shape of the object. The critical step of BLAISER is the determination of the shape orientation and we provide a detailed discussion on this topic. The operations of BLAISER are based on low-order moments of the input shape and, thus, robust to error and noise. Examples illustrate the performance of the algorithm.**

***Index Terms*—Affine-permutation invariance, blind algorithm for intrinsic shape recovery, fold number, intrinsic shape, orbits, orientation indicator index (OII), point-based reorientation algorithm (PRA), shape invariance, shape space.**

I. Introduction

IN IMAGE processing and computer vision, the shape of an object is described by the two-dimensional (2-D) projections of the three-dimensional (3-D) feature points of the object on the image plane. This perceived shape is highly dependent on the viewpoint under which the object is imaged and the order in which the feature points are scanned. This dependence has significant effects on many image processing and computer vision applications, for example, when detecting, classifying, or identifying an object from its 2-D image. A common way to address this problem is to restrict these shape variations to certain classes, for example, translations and rotations, and then designing a detector or a classifier that is "robust" to this variability, e.g., [1]–[5]. In this paper, we take a different approach to the problem and look for an invariant definition of the shape of an object. This invariant shape can then be used as template in detection or classification problems or in image database searches.

Invariant features of 2-D shapes under affine transformations have been studied in image processing and computer vision applications such as pattern recognition [23], [27], [28]. Good 2-D affine invariants are complete, easy to compute, stable under small distortions, and continuous. There are a number of different approaches to obtaining affine invariants. The normalization approach, in particular, has been well outlined in [11], [12], for example, and is the method chosen for our work in this paper. The author in [11] defines invariant features of planar curves under affine or projective transformations using the framework of group theory. Then, he shows an example of global normalization techniques to compute the affine invariants using low-order moments. The approach in this reference requires an edge detector prior to normalization and the orientation ambiguity (rotation and reflection) is not discussed in depth.

Shapes in general have also been studied in other contexts. For example, Kendall *et al.* [13], [14] have developed a theory of shape that makes the following assumptions. 1) *Labeled feature points*: The feature points that define a shape are labeled—they implicitly assume that the correspondence between features in different images of the same object has been established. 2)*Class of motions*: The class of motions that distort the shapes are restricted to translations, rotations, reflections, and uniform scaling. 3) *Shape space*: These references study the structure of the space of shapes after the allowed class of motions has been factored out. Grenander and his collaborators [16]–[20] describe the shapes of the objects by closed contours. The authors here map these contours onto nodes of object graphs and give a definition of shape in terms of invariance to a general class of transformation groups.

Our work is distinct from these previous works. We look for an invariant definition of the shapes *intrinsic shape* [6]–[8] that is invariant to the full class of affine-permutation distortions. We define the notion of intrinsic shape and present a *blind* algorithm that recovers it from any arbitrarily affine-permutation distorted image of the object. The algorithm is referred to as the blind algorithm for intrinsic shape recovery, or BLAISER for short. The algorithm is *blind* because no further information is needed other than the given input shape. We now list the important features of our work. 1) *Unlabeled feature points:* We work with unlabeled feature points. We represent the unknown labeling of the feature points by a permutation. We avoid the computation-intensive feature correspondence problem by doing so. 2) *Affine distortions:* Beyond the geometric distortions studied by Kendall *et al.*, i.e., translation, rotation, and uniform scaling, we consider a more general class of shape distortions—the affine distortions. This class of distortions is more

Manuscript received December 31, 2003; revised November 12, 2004. This work was supported in part by the Office of Naval Research under Grant N000 14-00-1-0593. V. H. S. Ha performed work for this paper while at Carnegie Mellon University. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Theirry Blu.

V. H. S. Ha is with Digital Media Solutions Laboratory, Samsung Information Systems America, Irvine, CA 92612 USA (e-mail: v.ha@samsung.com).

J. M. F. Moura is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213-3890 USA (e-mail: moura@ece.cmu.edu).

Digital Object Identifier 10.1109/TIP.2005.857271

general than Kendall's but more restrictive than Grenander's. 3) *Rigidity constraint:* Our approach exploits the rigidity constraint on the object's shape and its distortions. By doing so, we can recover the intrinsic shape using a simpler algebraic approach rather than a graph theoretic framework as in [19]. 4) *Low-order moment-based approach:* The affine-permutation invariant, i.e., intrinsic shape, is obtained by computing a set of low-order (up to third order) moments of the input shape. We avoid using higher order moments that are generally considered to be more sensitive to noise. 5) *Dense image representation:* We include all points in the object that are imaged by the sensor as the feature points, including the points that are inside the image contour, leading to a more detailed description of shapes, especially for those shapes with multiple complex structures (i.e., holes, disjoint parts, etc.). We require the shape to be segmented first, but unlike other previous works, feature extraction or edge detection is not needed. When this dense representation is combined with the moment-based approach in 4), more robust performance results are obtained. 6) *Intrinsic shape:* Most importantly, we are interested in recovering the canonical form of the object's shape, its intrinsic shape. The intrinsic shape is the affine-permutation invariant and is obtained via the normalization process that completely removes the full affine-permutation distortions. The study of intrinsic shapes leads to interesting applications in image processing and computer vision.

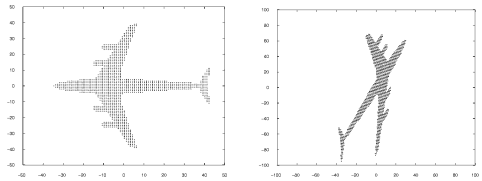
This paper is organized as follows. We start by introducing the concept of intrinsic shape in Section II. In Section III, we present the *blind* algorithm referred to as BLAISER. In Section IV, we focus on the shape orientation problem. In Section V, we discuss the robustness of BLAISER in the presence of random noise and erroneous pixels. Then, we apply BLAISER to a set of real data to verify the affine-permutation shape distortion model and illustrate the performance of BLAISER. Section VI summarizes and concludes the paper.

II. Intrinsic Shape

We introduce the notion of intrinsic shape. We first fix the notation for representing shapes and affine-permutation shape distortions. Next, we define what the intrinsic shape of an object is under the framework of group theory. Then, we formally state the problem of recovering intrinsic shape from affine-permutation distorted shapes. In this section, the concepts and algorithms are developed under an ideal environment where digitization error, background noise, occlusion, and other perturbations do not exist. The shapes are assumed to be segmented first. Then, all pixel points constituting the shape are identified as the feature points. These points have one-to-one correspondence with all of its affine-permutation distorted versions by a precise mathematical relationship (see Fig. 1 for typical input shapes used in this work). The nonideal environments are introduced later in Section V-A where we discuss the robustness of our approach to error, noise, and occlusion.

A. Definitions

Consider the shape of an object described by a configuration of N unlabeled points $\{p_k\}_{k=1...N}$ on a 2-D plane $\mathbb{R}^2$. Each point p_k is assigned a nonzero amplitude value and its location on the plane is specified by a pair of coordinates (x_k, y_k) with respect to a reference coordinate system with x and y axes. This shape is represented by a $2 \times N$ *configuration matrix* $\mathbf{X}$ that collects the x and y coordinates of the N feature points. That is



(a) (b)

Fig. 1. Typical input shapes. (a) Shape of airplane. (b) Distorted shape.

$$\mathbf{X} = \begin{bmatrix} x_1 & x_2 & \dots & x_N \\ y_1 & y_2 & \dots & y_N \end{bmatrix}. \tag{1}$$

The configuration matrix may be vectorized to a $2N$-dimensional vector $\mathbf{x}$, using the vec operation [22]. For example, in row–order notation, we stack the rows of $\mathbf{X}$ as

$$\mathbf{x} = \mathrm{vec}\mathbf{X} = [x_1\, x_2 \dots x_N\, y_1\, y_2 \dots y_N]^T. \tag{2}$$

We use interchangeably the terms shape, configuration matrix $\mathbf{X}$, or configuration vector $\mathbf{x}$.

We define the *configuration space* $\mathcal{X}$ as the collection of all configurations of N unlabeled points on a 2-D image plane. When we work with configuration matrices, $\mathcal{X}$ is the Euclidean space $\mathbb{R}^{2\times N}$ of $2 \times N$ matrices. When we work with vectors using the vec notation, $\mathcal{X}$ is the Euclidean space $\mathbb{R}^{2N}$ of $2N$-dimensional vectors. We exclude from these spaces lower order dimensional spaces that correspond to degenerate shapes where points at the same location are repeated.

B. Permutation

The reordering of the columns in a configuration matrix $\mathbf{X}$ is performed by multiplying the configuration matrix on the right by an $N \times N$ permutation matrix $\mathbf{P}$. The permutation matrix $\mathbf{P}$ is a highly sparse orthogonal matrix with only one nonzero entry, which is unity, per row, and per column. In practice, different permutation matrices describe different orders by which the feature points of the image are scanned by the input device. The unknown permutation matrix $\mathbf{P}$ indicates that the correspondence is not known between the pixels of different images of the same object.

C. Affine Distortion

In many imaging environments, the sensor and the imaged objects are arbitrarily oriented with respect to each other. Due to the unknown relative positions, the shape of the object captured in the image is geometrically distorted. These geometric distortions are well approximated by an affine transformation in many applications [23]. Affine distortions encompass a wide range of geometric distortions including translation, reflection, rotation, uniform (isotropic) and nonuniform (anisotropic) scaling, and

skewing. Two configuration matrices $\mathbf{X}$ and $\mathbf{X}^d$ are affine distorted from each other if they are related as

$$\mathbf{X}^d = \mathbf{A}\mathbf{X} + \mathbf{1}^T \otimes \boldsymbol{\delta} \tag{3}$$

where the linear distortion matrix $\mathbf{A}$ is a 2×2 invertible matrix of real numbers and the translation vector $\boldsymbol{\delta}$ is a 2×1 vector of real numbers. That is

$$\mathbf{A} = \begin{bmatrix} a_1^1 & a_2^1 \\ a_1^2 & a_2^2 \end{bmatrix}, \quad \delta = \begin{bmatrix} \delta_x \\ \delta_y \end{bmatrix}.$$

Note that $\otimes$ is the Kronecker product [22] and $\mathbf{1}$ is a $N \times 1$ vector of ones. The affine distortion parameters $(\mathbf{A}, \boldsymbol{\delta})$ completely describe the affine distortion of the configuration matrix $\mathbf{X}$ to $\mathbf{X}^d$.

D. Affine-Permutation Distortion

We now collect the two types of distortions, affine and permutation, in a single model. The two shapes of the object $\mathbf{X}^d$ and $\mathbf{X}$ are related by an affine-permutation distortion if

$$\mathbf{X}^d = (\mathbf{A}\ \mathbf{X} + \mathbf{1}^T \otimes \boldsymbol{\delta})\mathbf{P} = \mathbf{A}\ \mathbf{X}\ \mathbf{P} + \mathbf{1}^T \otimes \boldsymbol{\delta}. \tag{4}$$

In vec notation, the above equation is written as

$$\mathbf{x}^d = (\mathbf{P}^T \otimes \mathbf{A})\mathbf{x} + \mathbf{1} \otimes \boldsymbol{\delta} \tag{5}$$

where $\mathbf{x}^d = \mathrm{vec}\mathbf{X}^d$ and $\mathbf{x} = \mathrm{vec}\mathbf{X}$. By the properties of the Kronecker product [24], the matrix $\mathbf{P}^T \otimes \mathbf{A}$ is invertible.

E. Intrinsic Shape

The affine-permutation distortion in (5) is a special case of the general affine transform in $\mathbb{R}^{2N}$

$$\mathbf{x}^d = \mathrm{vec}\mathbf{X}^d = \mathbf{G}\,\mathrm{vec}\,\mathbf{X} + \mathbf{t} = \mathbf{G}\,\mathbf{x} + \mathbf{t} \tag{6}$$

where $\mathbf{G}$ is a $2N \times 2N$ invertible matrix of real numbers and $\mathbf{t} \in \mathbb{R}^{2N}$ is a generic translation vector. We consider the set, $\mathcal{G} = \{g = (\mathbf{G}, \mathbf{t})\}$, and define the binary operation $\circ : \mathcal{G} \times \mathcal{G} \to \mathcal{G}$ where

$$\begin{aligned} g_1 \circ g_2 &= (\mathbf{G}_1\,\mathbf{G}_2, \mathbf{t}_1 + \mathbf{G}_1\mathbf{t}_2), \\ \forall g_1 &= (\mathbf{G}_1, \mathbf{t}_1), g_2 = (\mathbf{G}_2, \mathbf{t}_2) \in \mathcal{G}. \end{aligned} \tag{7}$$

Clearly, $\mathcal{G}$ is a group, the affine group [25]. Within the generic affine group $\mathcal{G}$, we consider the subset

$$\mathcal{A} = \left\{a = \left(\mathbf{P}^{\mathbf{T}} \otimes \mathbf{A}, \mathbf{1} \otimes \boldsymbol{\delta}\right)\right\} \subset \mathcal{G} \tag{8}$$

with an invertible 2×2 matrix $\mathbf{A}$, a $N \times N$ permutation matrix $\mathbf{P}$, and a 2×1 real-valued vector $\boldsymbol{\delta}$. By the properties of the Kronecker product, this set is closed under the group operation $\circ$. The set $\mathcal{A}$ and its associated binary operation $\circ$ constitute a group in its own right, hence, a subgroup of $\mathcal{G}$. The group $\mathcal{A}$ is the important structure that models the affine-permutation distortions of shapes. We call $\mathcal{A}$ the affine-permutation group.

We now consider the action of a group on a set. The action ϕ of the group $\mathcal{A}$ on the configuration space $\mathcal{X}$ is defined by the mapping

$$\begin{aligned} \phi : \mathcal{A} \times \mathcal{X} &\longrightarrow \mathcal{X} \\ (a, \mathbf{X}) &\longmapsto a\mathbf{X} = \mathbf{A}\mathbf{X}\mathbf{P} + \mathbf{1}^T \otimes \boldsymbol{\delta} \\ (a, \mathbf{x}) &\longmapsto a\mathbf{x} = (\mathbf{P}^T \otimes \mathbf{A})\mathbf{x} + \mathbf{1} \otimes \boldsymbol{\delta}. \end{aligned}$$

The group action ϕ defines an equivalence relation among the configuration matrices $\mathbf{X} \in \mathcal{X}$. We represent this equivalence relation by the symbol $\equiv_{\mathcal{A}}$. It follows easily from their properties that equivalence relations partition the configuration space $\mathcal{X}$ into disjoint equivalence classes, also commonly referred to as orbits. The set $\mathcal{C}$ of orbits $\mathbf{C}$ is the factor space

$$\mathcal{C} = \mathbf{X}/_{\equiv_{\mathcal{A}}} = \{\mathbf{C} \subset \mathcal{X} : \forall \mathbf{X}_1, \mathbf{X}_2 \in \mathbf{C}, \mathbf{X}_1 \equiv_{\mathcal{A}} \mathbf{X}_2\}. \tag{9}$$

Often, to indicate explicitly the group, we use the notations $\mathcal{C}_{\mathcal{A}}$ and $\mathbf{C}_{\mathcal{A}}$. The action of the general affine group $\mathcal{G}$ on $\mathcal{X}$ is transitive, and so the corresponding $\mathcal{X}/_{\equiv_{\mathcal{G}}}$ has a single element. However, the action of the subgroup $\mathcal{A}$ on $\mathcal{X}$ is not transitive and there is more than one orbit in $\mathcal{X}$. This follows because $\mathcal{A}$ is strictly contained in $\mathcal{G}$, i.e., $\mathcal{A} \subset \mathcal{G}$.

Since the equivalence relation $\equiv_{\mathcal{A}}$ partitions the configuration space into disjoint sets, the orbits are either identical or nonintersecting. An orbit $\mathbf{C}_{\mathcal{A}}$ can be generated by any configuration $\mathbf{X}$ in the orbit as

$$\mathbf{C}_{\mathcal{A}} = \{\phi(a, \mathbf{X}) : \forall a \in \mathcal{A}\}. \tag{10}$$

Then, an orbit collects all possible configurations arising from affine-permutation distortions of the object's shapes. It is desirable to select from all these configurations a unique configuration $\mathbf{S}$ that we label the *intrinsic shape* of the object. Clearly, any configuration $\mathbf{X}$ of an object is class equivalent to its intrinsic shape $\mathbf{S}$. Our goal is to show that we can in fact identify for each orbit such a unique representative and to recover blindly the intrinsic shape $\mathbf{S}$ of the object from any of the configurations $\mathbf{X} \in \mathbf{C}_{\mathcal{A}}$.

The problem is summarized by the following input/output relationship:

$$\begin{aligned} \mathrm{Input} &: \mathbf{X} = \mathbf{A}\mathbf{S}\mathbf{P} + \mathbf{1}^T \otimes \boldsymbol{\delta} \in \mathbf{C}_{\mathcal{A}} \\ \mathrm{Output} &: \mathbf{S} \in \mathbf{C}_{\mathcal{A}} \quad \text{and} \quad (\mathbf{A}, \mathbf{P}, \boldsymbol{\delta}). \end{aligned}$$

Given an arbitrary configuration $\mathbf{X}$ of an unknown object, we find blindly its intrinsic shape $\mathbf{S}$, i.e., without knowledge of the group element $a = \{\mathbf{A}, \mathbf{P}, \boldsymbol{\delta}\}$ and its intrinsic shape $\mathbf{S}$. It is important to note that this statement raises two problems: 1) definition of the canonical representative of an orbit, i.e., the intrinsic shape; 2) design of the blind algorithm that recovers $\mathbf{S}$ corresponding to the given $\mathbf{X}$. We address these two issues in

the next section by introducing a new algorithm referred to as BLAISER.

III. BLAISER

In this section, we develop BLAISER, an algorithm that recovers blindly from any distorted configuration $\mathbf{X}$ in an orbit $\mathbf{C}_{\mathcal{A}}$ the corresponding intrinsic shape $\mathbf{S}$. Consider an orbit $\mathbf{C}_{\mathcal{A}}$, a member $\mathbf{X}$, and its unique intrinsic shape $\mathbf{S}$. The two configurations $\mathbf{X}$ and $\mathbf{S}$, each consisting of N points, are related as $\mathbf{X} = \mathbf{ASP} + \mathbf{1}^T \otimes \boldsymbol{\delta}$, where the parameterization $a = (\mathbf{A}, \mathbf{P}, \boldsymbol{\delta})$ of the affine-permutation distortion is unknown.

Definition 1 (Intrinsic Shape): The intrinsic shape $\mathbf{S}$ is defined for an orbit $\mathbf{C}_{\mathcal{A}}$ by the following four properties.

- The center of mass is located at the origin.
- The outer product is an identity matrix, i.e., $\mathbf{SS}^T = \mathbf{I}$.
- The *reorientation point* (to be defined in the discussion of shape reorientation) of $\mathbf{S}$ is aligned with the $\mathbf{x}$-coordinate axis.
- The columns in $\mathbf{S}$ are ordered in ascending y-coordinate values, then in ascending $\mathbf{x}$-coordinate values for the columns with the same y-coordinate values.

The algorithm that finds the intrinsic shape of any affine-permutation distorted shape is then summarized as below.

Algorithm 1 (BLAISER): BLAISER reduces the affine-permutation distorted shape $\mathbf{X} \in \mathbf{C}_{\mathcal{A}}$ to its intrinsic shape $\mathbf{S} \in \mathbf{C}_{\mathcal{A}}$ by the following four steps.

- Centering: $\mathbf{X} = \mathbf{ASP} + \mathbf{1}^T \otimes \delta \longrightarrow \mathbf{X}^c = \mathbf{ASP}$.
- Reshaping: $\mathbf{X}^c = \mathbf{ASP} \longrightarrow \mathbf{X}^s = \mathbf{USP}$.
- Reorientation: $\mathbf{X}^s = \mathbf{USP} \longrightarrow \mathbf{X}^u = \mathbf{SP}$.
- Sorting: $\mathbf{X}^u = \mathbf{SP} \longrightarrow \mathbf{X}^r = \mathbf{S}$.

Note that $\mathbf{U}$ is a 2×2 orthogonal matrix. The first two steps, centering and reshaping, are together referred to as "compacting" or "shape normalization" in the pattern recognition community. Readers are referred to [6], [11], and [27] for details. As shown in [11], the "compacting" process results in a 2-D affine invariant that is continuous, easy to compute, and robust to digitization errors. No distinguished points are needed to obtain this invariant. However, the orientation ambiguity is not resolved by these steps.

The critical part of BLAISER then lies in the reorientation step. We present a new efficient and robust algorithm for shape reorientation in this paper. The point-based reorientation algorithm (PRA) and the orientation indicator index (OII) given in Section IV remove the orientational ambiguity $\mathbf{U}$ of the normalized shape $\mathbf{X}^s = \mathbf{USP}$. The result is the reoriented shape $\mathbf{X}^u = \mathbf{SP}$. The details are discussed in Section IV.

The last step, sorting, can be implemented efficiently using any well-known sorting algorithm after affine distortions are first eliminated. This can be achieved in many different ways. We sort the reoriented shape $\mathbf{X}^u = \mathbf{X}\ \mathbf{P}$ to $\mathbf{X}^r$ by reordering the columns in ascending order of the y coordinate values, then in ascending order of the x coordinate values within the columns with the same value of y.

The computation of the intrinsic shape by BLAISER is based on the first-, second-, and third-order moments of the input shape. First, in the centering step, the first-order moment is set to zero. That is, $m_{10} = m_{01} = 0$ where the $(p+q)$th-order moment is defined as $m_{pq} = \sum_{(i,j)} x_i^p y_j^q$ for all points (x_i, y_j) in the shape. In the reshaping step, the second-order central moments are set as $m_{20} = m_{02} = 1$ and $m_{11} = 0$. Finally, in the reorientation step, the third-order central moments m_{30} and m_{03} are computed and used to generate the OII plot. The low-order moment-based approach of BLAISER provides robustness to error and noise as will be discussed in Section V-A

IV. Shape Reorientation

The centering, reshaping, and sorting steps of BLAISER are straight-forward and easy to understand. In this section, we discuss the remaining operation, reorientation. The normalized shape $\mathbf{X}^s$ obtained after the reshaping step in Algorithm 1 is still arbitrarily rotated or reflected. In the reorientation step, we remove this orientational ambiguity of the normalized shape $\mathbf{X}^s$.

Many research efforts have attempted to determine the orientation of an object and discover the fold number of rotationally symmetric shapes. 1) Principal axes [28], shape matrices [29], and mirror-symmetry axes [30] methods are unfortunately inapplicable to shapes that are rotational symmetric or nonmirror symmetric. 2) Modified Fourier descriptor [31], [32], generalized principal axis [33], fold principal axis [34], and fold-invariant shape-specific points [35] methods are not universal and fail on some tested shapes. 3) Techniques in [36] and [37] attempt to determine the fold number of rotationally symmetric shapes but again fail with some tested shapes. 4) Universal principal axes [38] and generalized complex (GC) moments [39] remove rotational ambiguity and determine the fold number, simultaneously. However, they use higher order moments that are generally considered to be more sensitive to error and they provide no guarantee of convergence within a finite number of iterations. None of these works mentioned so far deal with reflection. 5) The Shen-Ip symmetries detector [40] is concerned with the problem of detecting the reflectional and rotational symmetry axes of a shape. The algorithm requires the computation of all orders of the shape's GC moments, not just a finite number of them. Thus, it does not promise a correct determination of the symmetry axes within a finite number of GC moment computations. A practical implementation of their algorithm attempts to find at least three nonzero GC moments by computing up to 30th-order GC moments of the shape (depending on the complexity of the shape); this turns out to be inadequate for some of the shapes tested in their experiments.

In this paper, we develop the PRA that is a simple, efficient, and complete method to determine the fold number of the shape and to remove the full orientational ambiguity including both rotation *and* reflection. Its computational complexity is $O(N \log N)$ where N is the total number of feature points in the shape. To improve the robustness of PRA in a noisy environment, we introduce to PRA a new measure of orientation referred to as variable-size window OII (Δ-OII). The combined PRA-Δ-OII algorithm is, thus, efficient and stable [9]. First, we state a series of Lemmas about the orientational symmetry of a shape in Lemmas 1–9. These Lemmas together with the *reorientation point* p_o introduced in Lemma 10 become the basis of PRA presented in Theorem 1 in Section IV-E. Due to

lack of space, readers are referred to [6] for complete proofs of these Lemmas.

A. Shape Decomposition

PRA decomposes a shape into a set of subshapes where each subshape is composed of feature points located at the same distance from the center of the shape. To do that, we first rewrite the configuration matrix $\mathbf{X}^s$ of a normalized shape in polar coordinates

$$\mathbf{X}_\theta^\rho = \begin{bmatrix} \rho_1 & \rho_2 & \cdots & \rho_N \\ \theta_1 & \theta_2 & \ldots & \theta_N \end{bmatrix} \tag{11}$$

where

$$\rho_i = \sqrt{x_i^2 + y_i^2}, \quad \theta_i = \arctan\left(\frac{y_i}{x_i}\right).$$

We measure the angles θ_i from the x axis in the counterclockwise direction and use the 2-D version of the arctan function. Next, we sort the columns of the matrix x_θ^ρ by decreasing order of the ρ_i, then again by increasing order of the θ_i within the columns with the same value of ρ. We group the columns with the same value of ρ together as a subshape

$$\mathbf{O}_j = \begin{bmatrix} \rho_j & \rho_j & \cdots & \rho_j \\ \theta_1 & \theta_2 & \ldots & \theta_{N_j} \end{bmatrix}, \quad j = 1 \ldots R \tag{12}$$

where N_j is the number of points contained in the subshape $\mathbf{O}_j$ and R is the number of distinct values of ρ. Each subshape is the set of points located on a ring with radius ρ_i centered at the origin.

From the matrix $\mathbf{O}_j$, we compute and generate the list of angles $\mathbf{Z}_j$ where

$$\begin{aligned} \mathbf{Z}_j &= \begin{bmatrix} \theta_2 - \theta_1 & \theta_3 - \theta_2 & \ldots & 2\pi + \theta_1 - \theta_{N_j} \end{bmatrix} \\ &= \begin{bmatrix} \theta_{1,2} & \theta_{2,3} & \ldots & \theta_{N_j,1} \end{bmatrix}. \end{aligned} \tag{13}$$

Each angle $\theta_{i,j}$ is the angle between two neighboring points p_i and p_j, also denoted as $\measuredangle \vec{P}_i O \vec{P}_j$ where O is the origin and $\vec{P}_i$ is the vector from the origin to the point p_i. A circular shift of the list $\mathbf{Z}_j$, denoted as $\operatorname{cir} \mathbf{Z}_j$, represents a different starting point and does not change the description of the subshape $\mathbf{O}_j$. Only the relative order and the magnitudes of the angles in the list are important.

B. Rotational Symmetry

A shape has r-fold rotational symmetry if it is indistinguishable after being rotated about its center of mass by an angle $(2k\pi)/(r)$, $k = 1, \ldots, r$. A shape is considered nonrotational symmetric if it is one-fold rotational symmetric. The fold number of a shape is the largest integer r for which the shape is r-fold rotational symmetric.

Lemma 1: A shape containing a set of N points can only have r-fold rotational symmetry where r is an integer divisor of N.

Lemma 2: If a shape is r-fold rotational symmetric, it is also k-fold rotational symmetric where k is an integer divisor of r.

Lemma 3: A shape is nonrotational symmetric if any of the subshapes $\mathbf{O}_j, j = 1, \ldots, R$, is not rotational symmetric.

Lemma 4: If all of the subshapes $\mathbf{O}_j$ of the given shape $\mathbf{X}$ are r_j-fold rotational symmetric for $j = 1, \cdots R$, the shape is r-fold rotational symmetric where r is the greatest common denominator of the fold numbers $\{r_j\}_{j=1 \ldots R}$.

Lemma 5: If a subshape $\mathbf{O}_j$ containing N_j points is r_j-fold rotational symmetric, its list of angles $\mathbf{Z}_j$ has a circularly repeated pattern of length $(N_j)/(r_j)$. The sum of any $(N_j)/(r_j)$ consecutive angles in the list is equal to $(2\pi)/(r_j)$.

Lemma 6: For every shape, it is possible to identify a unique (up to a circular shift) list of angles $\mathbf{L}$ that is nonperiodic. Such a list $\mathbf{L}$ is defined as the *fundamental list of angles*.

Lemma 7: A rotation of the shape results in a cyclic shift of the fundamental list of angles. That is, for a shape $\mathbf{X}$ and its rotated version $\mathbf{X}^R$, we have the corresponding fundamental list of angles $\mathbf{L}$ and $\mathbf{L}^R$, respectively, where

$$\mathbf{L}^R = \operatorname{cir} \mathbf{L} \tag{14}$$

and "cir" is a circular shift of the lists.

C. Reflectional Symmetry

A general reflection is represented by an orthogonal matrix $\mathbf{F}$ of the form

$$\mathbf{F} = \begin{bmatrix} a & b \\ b & -a \end{bmatrix}, \quad a^2 + b^2 = 1 \quad \text{and } (a, b) \in \mathbb{R}. \tag{15}$$

Lemma 8: Any reflection matrix $\mathbf{F}$ can be written as a sequence of an appropriate rotation $\mathbf{R}$ and a reflection $\mathbf{F}_x$ about the x axis. That is

$$\mathbf{F} = \begin{bmatrix} a & b \\ b & -a \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} a & b \\ -b & a \end{bmatrix} = \mathbf{F}_x \mathbf{R} \tag{16}$$

where $a^2 + b^2 = 1$ and $(a, b) \in \mathbb{R}$.

In Lemma 8, the reflection $\mathbf{F}_x$ can be replaced by a reflection about an arbitrary axis through the origin, e.g., the reflection $\mathbf{F}_y$ about the y axis. In this paper, we decide to work with the x axis.

Lemma 9: If the configuration $\mathbf{X}^F$ is a reflected version of $\mathbf{X}$ with a general reflection $\mathbf{F}$ as in

$$\mathbf{X}^F = \mathbf{F}\mathbf{X} \tag{17}$$

then the list of angles $\mathbf{Z}^F$ of $\mathbf{X}^F$ is in the reverse order of the list $\mathbf{Z}$ of $\mathbf{X}$. That is, if

$$\mathbf{Z} = \operatorname{cir} \begin{bmatrix} \theta_{1,2} & \theta_{2,3} & \ldots & \theta_{N_j,1} \end{bmatrix} \tag{18}$$

then

$$\mathbf{Z}^F = \operatorname{cir} \begin{bmatrix} \theta_{N_j,1} & \ldots & \theta_{2,3} & \theta_{1,2} \end{bmatrix} \tag{19}$$

where "cir" is a circular shift of the lists.

According to Lemma 9, traversing the angles between any two rotationally consecutive points of the shape $\mathbf{X}$ in the counterclockwise direction is equivalent to traversing the angles of the reflected shape $\mathbf{X}^F$ in the clockwise direction.

D. Reorientation Point

The *reorientation point* is a specific feature/pixel point in the shape that is uniquely identifiable. PRA uses this reorientation

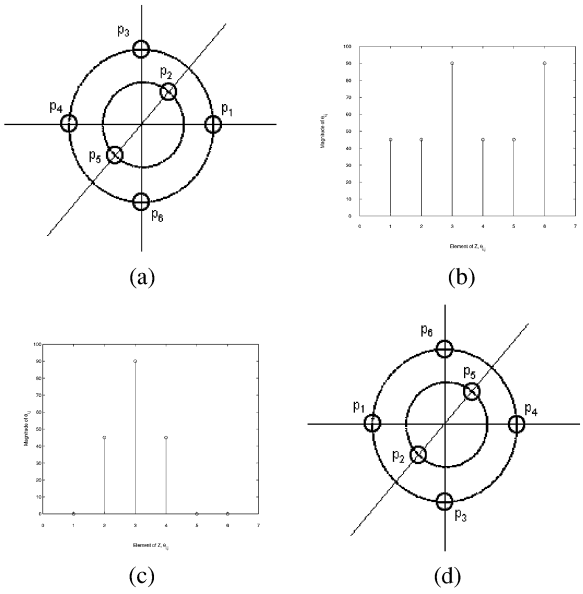


Fig. 2. Example: PRA. (a) Input shape $\mathbf{X}$. (b) List of angles $\mathbf{Z}$. (c) Fundamental list of angles $\mathbf{L}$. (d) Reoriented shape.

point to restore the correct orientation of the shape. In this section, we introduce the reorientation point and describe how to identify it from any given shape.

1) Concept: The reorientation point is critical to the shape reorientation process in PRA. PRA reorients the given shape by: 1) identifying the reorientation point of the shape; 2) rotating the shape until its reorientation point falls on the x axis; and 3) reflecting the shape about the x axis if the nearest neighboring point of its reorientation point in the clockwise direction is farther away (in the angular sense) from the reorientation point than the nearest neighboring point in the counterclockwise direction. To reorient shapes according to these steps, the reorientation point needs to be uniquely defined for any given distorted shape of the same object as stated in the following lemma.

Lemma 10: The reorientation point of a shape is uniquely determined from any given shape of the same object with unknown orientation.

2) Algorithm: We now explain how to determine the reorientation point from a given shape. There are three main steps: 1) determine the fold number of the shape; 2) construct the fundamental list of angles; and 3) identify the reorientation point of the shape.

Fold Number: First, we decompose the shape $\mathbf{X}$ into a set of subshapes $\mathbf{O}_j$ and compute the lists of angles $\mathbf{Z}_j$, $j = 1 \dots R$, associated with the subshapes. Then we compute the fold numbers r_j of the subshapes from $\mathbf{Z}_j$, $j = 1 \dots R$, using the result of Lemma 5. Finally, we determine the fold number r of the overall shape as the greatest common denominator of the fold numbers r_j, $j = 1 \dots R$, as stated in Lemma 4 [see, for example, Fig. 2(a)]. The shape in (a) consists of $N = 6$ points. The first subshape contains four points (p_1, p_3, p_4, p_6) and is four-fold symmetric. The second subshape contains two points (p_2, p_5) and is two-fold symmetric. The overall shape is, thus, two-fold symmetric.

Fundamental List of Angles: Once the fold number r of the shape is found, we construct the fundamental list of angles $\mathbf{L}$ introduced in Lemma 6 from one of r nonperiodic portions of the shape. That is, simply choose N/r consecutive elements from the list of angles with N elements. In our example, the list of angles is $\mathbf{Z} = [\theta_{1,2}, \theta_{2,3}, \cdots, \theta_{5,6}, \theta_{6,1}] = [\pi/4, \pi/4, \pi/2, \pi/4, \pi/4, \pi/2]$ [see Fig. 2(b)]. Since the fold number of the shape is 2, we take any $N/r = 3$ consecutive elements from $\mathbf{Z}$ as the fundamental list of angles $\mathbf{L}$. For example, $\mathbf{L} = [\theta_{2,3}, \theta_{3,4}, \theta_{4,5}] = [\pi/4, \pi/2, \pi/4]$ in Fig. 2(c).

Reorientation Point: We search for a unique element $\theta_{ij} = \measuredangle \vec{P}_i O \vec{P}_j$ of $\mathbf{L}$ by locating the element with the largest magnitude. If there is more than one element with the same maximum magnitude, we compare the magnitudes of the neighboring elements to break the tie. Denote all occurrences of the maximum magnitudes with the labels $\theta^*_{k=1\dots m}$ where m is the total number of such occurrences. We then compare the right-side neighbors of each of the elements $\theta^*_{k=1\dots m}$. We choose the one whose nearest right-side neighbor has the largest value. The chosen element is denoted as θ^*_R. Then, we repeat the same procedure by comparing the left-side neighbors of $\theta^*_{k=1\dots m}$ and obtain θ^*_L. In our example in Fig. 2(c), $\theta^*_R = \theta^*_L$. Finally, we identify the unique element θ^* in the list by picking one of the two angles $\{\theta^*_R, \theta^*_L\}$. We pick one of the two angles $\{\theta^*_R, \theta^*_L\}$ by comparing the right-side neighbors of θ^*_R with the corresponding left-side neighbors of θ^*_L, starting with the closest neighbors. We choose the unique element of the list $\theta^* = \theta^*_R$ if its kth right-side neighbor is larger than the kth left-side neighbor of θ^*_L for $k = 1 \dots N-1$. In the opposite case, we choose $\theta^* = \theta^*_L$. During the process, if all neighbors have the same magnitudes, the list $\mathbf{L}$ is symmetrical and we can choose any of the two. If the angle $\theta^*_R = \theta_{ij}$ is chosen as the unique element θ^* of the list, we set the point p_j to be the reorientation point p_o and align the vector $\vec{P}_o = \vec{P}_j$ with the x axis. If the angle $\theta^*_L = \theta_{ij}$ is chosen, we declare the point p_i to be p_o, align the vector $\vec{P}_o = \vec{P}_i$ with the x axis, and reflect the entire shape about the x axis. In our example, $\theta^* = \theta^*_R = \theta^*_L = \theta_{3,4}$. If we choose $\theta^* = \theta^*_R = \theta_{3,4}$, the point p_4 becomes the reorientation point p_o and the shape is rotated so that this point falls on the x axis. If we choose $\theta^* = \theta^*_L = \theta_{3,4}$, the point p_3 becomes the reorientation point p_o, the shape is rotated so that this point falls on the x axis, and the shape is reflected about the x axis. In both cases, the resulting shapes are identical [see Fig. 2(d)].

The reorientation point, computed using the fundamental list of angles, is invariant under rotation and reflection. This is because a rotation or a reflection shifts or reverses the fundamental list of angles cyclically (Lemma 7 and Lemma 8). The magnitudes and the relative locations of the elements are unaffected by rotation and reflection. The reorientation point, however, is not invariant under the full affine-permutation distortion. It can be computed only from the normalized (compact) shape X^s after the first two steps of BLAISER are performed.

3) Example: We give another simple example to illustrate how to determine the reorientation point from a given shape. According to Lemma 6, we construct a unique (up to a circular shift) nonperiodic list of angles $\mathbf{L}$ from the input shape. Consider, for example, the list $\mathbf{L}$ shown in Fig. 3 that consists of nine elements $[\theta_{1,2}\ \theta_{2,3}\ \theta_{3,4}\ \dots\ \theta_{9,1}]$. For simplicity, we represent these nine angles by $[\theta_a\ \theta_b\ \theta_c\ \dots\ \theta_i]$ with a single subscript. We define the *right-side (or left-side) neighbor* of an element in

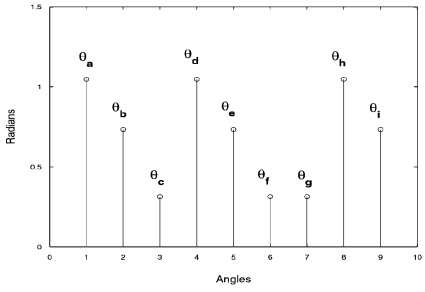


Fig. 3. Fundamental list of angles $\mathbf{L}$.

the list as the element that appears on the right (or left) in a cyclic manner. For example, in Fig. 3, the first right-side neighbor of the element θ_i is θ_a and the second left-side neighbor of the element θ_c is θ_a. To choose a unique element θ^* of the list, we start by picking the element that has the largest value. If there is a single element with the largest value, we simply denote it as θ^*. This element is uniquely identifiable in the list $\mathbf{L}$. If there are multiple elements sharing the same maximum value, we note all of their occurrences with the labels $\theta^*_{k=1\ldots m}$ where m is the total number of such occurrences. For the list in Fig. 3, there are three occurrences of the maximum value: the elements $\theta^*_1 = \theta_a$, $\theta^*_2 = \theta_d$, and $\theta^*_3 = \theta_h$. We need an algorithm that consistently chooses one out of these elements. We explain next how to determine θ^* when there are more than one angle with the maximum value.

Given the elements $\theta^*_{k=1\ldots m}$ with the same maximum value, we choose one by comparing the values of their neighbors. We first compare the right-side neighbors of each of the elements $\theta^*_{k=1\ldots m}$. We choose the one whose nearest right-side neighbor has the largest value. The chosen element is denoted as θ^*_R. In our example, the first right-side neighbors of $\theta^*_{k=1\ldots 3}$ are θ_b, θ_e, and θ_i. Since these elements have the same value, we move to the second right-side neighbors, θ_c, θ_f, and θ_a, and compare their values. We find that the magnitude of θ_a, the second right-side neighbor of θ^*_3, is the largest. Therefore, we assign $\theta^*_R = \theta^*_3 = \theta_h$. Because the list $\mathbf{L}$ is nonperiodic and finite, it is guaranteed that we locate a unique member θ^*_R in $\mathbf{L}$ by comparing the right-side neighbors. We repeat the same procedure by comparing the left-side neighbors of $\theta^*_{k=1\ldots m}$ and obtain θ^*_L. In our example, θ_i is the largest among the first left-side neighbors and we set $\theta^*_L = \theta_a$. Finally, we identify the unique element θ^* in the list by picking one of the two angles $\{\theta^*_R, \theta^*_L\}$. We pick one of the two angles $\{\theta^*_R, \theta^*_L\}$ by comparing the right-side neighbors of θ^*_R with the corresponding left-side neighbors of θ^*_L, starting with the closest neighbors. We choose the unique element of the list $\theta^* = \theta^*_R$ if its kth right-side neighbor is larger than the kth left-side neighbor of θ^*_L for $k = 1 \ldots N-1$. In the opposite case, we choose $\theta^* = \theta^*_L$.

We consider both the right-side and left-side neighbors to deal with the reflectional ambiguity of the shape. According to Lemma 10, the list $\mathbf{L}^F$ obtained from the reflected version of the shape contains the same set of elements as the list $\mathbf{L}$ from the nonreflected version. The difference is that the lists are in a reverse order of each other. Traversing the right-side neighbors of the list $\mathbf{L}$ corresponds to traversing the left-side neighbors of the reversed list $\mathbf{L}^F$. That is, the unique element obtained by comparing the right-side neighbors in the list corresponds to the unique element obtained by comparing the left-side neighbors in the list of the reflected shape. This is why we first determine both θ^*_R and θ^*_L from the list $\mathbf{L}$. When we finally choose one of these two angles, we are choosing an element that is unique for the shape even if the shape is a reflected version.

Next, we determine the reorientation point from the unique element θ^* of the list. The shape is rotated until this reorientation point falls on the x axis. If θ^*_L is chosen as θ^*, we reflect the shape about the x axis to compensate for the reflection term that is present in the shape. This is possible because any general reflection is decomposed into an appropriate rotation followed by a reflection about the x axis as stated in Lemma 9. We explain below how to determine the reorientation point from the unique element θ^* of the list.

If the angle $\theta^*_R = \theta_{ij}$ is chosen as the unique element θ^* of the list, we set the point p_j to be the reorientation point p_o and align the vector $\vec{P}_o = \vec{P}_j$ with the x axis. If the angle $\theta^*_L = \theta_{ij}$ is chosen, we declare the point p_i to be p_o, align the vector $\vec{P}_o = \vec{P}_i$ with the X axis, and reflect the entire shape about the $\mathbf{X}$ axis. In our example, the first two right-side neighbors of θ^*_R have values that are identical to the first two left-side neighbors of θ^*_L. However, the third right-side neighbor, θ_b, of θ^*_R is larger than the third left-side neighbor, θ_g, of θ^*_L. We, therefore, choose $\theta^*_R = \theta^*_3 = \theta_{8,9}$ in this example. Finally, the point p_9 is selected to be the reorientation point p_o. The shape is reoriented by rotating the shape until the vector $\vec{P}_o = \vec{P}_9$ is aligned with the X axis.

E. Point-Based Reorientation Algorithm (PRA)

Lemmas 1–10 provide the basis for the PRA that orients any given shape to its normalized orientation as stated in the following theorem.

Theorem 1 (PRA): The PRA achieves the reorientation of a shape by the following steps.

1) Convert the configuration matrix $\mathbf{X}$ of a normalized (compact) shape in rectangular coordinates to $\mathbf{X}^\rho_\theta$ in polar coordinates.
2) Sort the columns of the matrix $\mathbf{X}^\rho_\theta$ by decreasing order of ρ. Sort again by increasing order of θ within the columns with the same value of ρ.
3) Consider the subshapes $\mathbf{O}_j, j = 1, \ldots, R$, defined by the columns of $\mathbf{X}$ with the same value of ρ. Construct the list of angles $\mathbf{Z}_j$ given in (13) for each subshape.
4) For each subshape $\mathbf{O}_j$, determine the fold number, r_j, of its rotational symmetry. That is, given the number of points N_j of the subshape,
   (a) Find all integer divisors of N_j (Lemmas 1 and 2). Start with the largest integer divisor $k = N_j$.
   (b) Sum the first N_j/k consecutive angles in the list $\mathbf{Z}_j$. If the sum is equal to $2\pi/k$ and the list is periodic with the same pattern of length N_j/k, declare the subshape to be $r_j = k$-fold rotational symmetric (Lemma 5). Otherwise, set k to the next largest integer divisor of N_j and repeat.
5) Declare the shape to be r-fold rotational symmetric where r is the greatest common denominator of the fold numbers $\{r_j\}_{j=1\ldots R}$ (Lemmas 3 and 4).

6) Identify from the r-fold rotational symmetric shape its fundamental list of angles $\mathbf{L}$ that is nonperiodic (Lemma 6).
7) Pick the *reorientation point* p_o from the list $\mathbf{L}$.
8) Align the vector $\vec{P}_o$ with the x axis as described above (Lemma 10).

F. Variable-Size Window OII (Δ-OII)

In Section IV-E, we presented the PRA to remove the orientational ambiguity of 2-D shapes. The PRA first determines the fold number of the shape and removes both the rotational and reflectional ambiguities. The algorithm is efficient with computational complexity $\mathcal{O}(N \log N)$, where N is the number of feature points in the shape.

In practice, however, the exact locations of the feature points may be measured inaccurately due to the finite resolution of the input device and due to background noise. PRA may be sensitive to such disturbances. Moreover, the rectangular to polar coordinates conversion in the first step of PRA is bound to create serious problems if not carefully managed. To improve the robustness of PRA to such error and noise, we introduce a moment-based measure of the orientation, referred to as the Δ-OII. This measure is computed from the third-order central moments of the shape and improves the OII that we introduced in [5]. The coupled PRA-Δ-OII algorithm is robust to errors, such as round-off errors in pixel locations or missing/added feature points.

In this section, we define Δ-OII. We study and verify the properties of Δ-OII with examples: We test Δ-OII against the shapes that other methods have failed to work with. We also test Δ-OII against 200 symmetric and nonsymmetric shapes provided by a database [40]. We then propose an algorithm that combines Δ-OII and PRA for the robust determination of shape orientation.

1) Δ-OII: We now introduce Δ-OII of the centered shape $\mathbf{X}$. Let θ be the orientation angle of the given shape. Δ-OII(θ) is defined as

$$\Delta\text{–OII}_{\mathbf{X}}(\theta) = \sqrt{\mu_x^2(\theta) + \mu_y^2(\theta)} \tag{20}$$

where

$$\mu_x(\theta) = \sum_{k \in \mathtt{A}(\theta)} x_k^3 \quad \text{and} \quad \mu_y(\theta) = \sum_{k \in \mathtt{A}(\theta)} y_k^3,$$
$$\mathtt{A}(\theta) = \left\{ k : (x_k, y_k) \in \mathbf{X}, \forall x_k, y_k \in \frac{2\pi}{\Delta}\text{-Window} \right\} \tag{21}$$

and Δ is an integer multiple of 4. The $(2\pi/\Delta)$ window is the wedge on the first quadrant of the coordinate plane enclosed between the positive x axis and the straight line through the origin at angle $(2\pi/\Delta)$ rads with the x axis. This window contains all the pixels (x, y) where $x > 0$ and $y \leq x \tan(2\pi/\Delta)$. For example, the $(\pi/2)$ window is the first quadrant of the coordinate plane, enclosed between the positive x and the positive y axes. As the orientation θ of the shape changes, Δ-OII changes.

When the shape rotates by an angle θ, Δ-OII(θ) is computed from the fraction of the shape that falls within the $(2\pi/\Delta)$ window. We will compute in the sequel Δ-OII(θ) as we rotate the given shape $\mathbf{X}$ by θ.

The third-order central moments μ_x and μ_y computed by (21) are not zero for $\theta \in [0, 2\pi)$ unless the shape is defined by a single point at the origin. This is because the shape is centered at its center of mass and every $(2\pi/\Delta)$ window is located within the first quadrant of the coordinate plane where the coordinate values of the pixels (x_k, y_k) are nonnegative. Therefore, Δ-OII(θ) is nonzero.

2) Properties of Δ-OII: The OII plot introduced in [5] is a cosinusoidal function of the rotation angle θ for every shape whose third-order moments are non zero. This is not true for Δ-OII(θ). We list the properties of Δ-OII(θ) below, see [6] for complete proofs.

Property 1: If a shape is r-fold rotational symmetric, we can find a Δ such that Δ-OII(θ) is periodic in the rotation angle θ with the period $T = 2\pi/r$.

Property 2: If Δ-OII(θ) is periodic with $T = 2\pi/k$, the shape is r-fold rotational symmetric where $1 \leq r \leq k$.

Property 3: If a shape is a circle or a union of rings, its Δ-OII(θ) is flat and equal to a constant $K > 0$ for $\theta \in [0, 2\pi)$.

Property 4: The rotation of a shape circularly shifts its Δ-OII(θ).

Property 5: The reflection of a shape reverses its Δ-OII(θ).

In practice, we plot Δ-OII(θ) only for a discrete set of angles θ_i that we will assume to be uniformly spaced in the range $[0, 2\pi)$. This samples the original continuous plot. The resolution of this discrete plot has a direct effect on the performance of the reorientation operation using Δ-OII(θ). We adjust the resolution of the Δ-OII(θ) plot according to the complexity of the given shape by setting the number of discrete samples θ_i to be proportional to the number of pixels of the shape.

The next section illustrates the properties of Δ-OII(θ) by computing it for all the shapes in a database, [40], that contains about two hundred symmetrical and nonsymmetrical shapes. Of course, due to space limitations, we detail here the results for eight difficult shapes from the database.

3) Examples: We compute Δ-OII(θ) for the shapes, Shape 1–Shape 8, shown in Fig. 4. These are the shapes that cause difficulty and failure for many existing methods as discussed before. The plots shown next to the shapes are the corresponding Δ-OII(θ); in each plot, the horizontal axis is the rotation angle θ. We note that, in all eight plots, Δ-OII(θ) is 1) not flat and 2) exhibits the rotational symmetry of the shape through its periodicity, i.e., the fold number of the shape is equal to the periodicity of Δ-OII(θ).

We used the $(\pi/2)$ window for all the shapes in Fig. 4, except for Shape 1 in Fig. 4(a) for which $(\pi/4)$ window is used. As mentioned, we have also plotted Δ-OII(θ) for the 200 symmetric and nonsymmetric shapes in the database [40]. The plots of Δ-OII(θ) using the $(\pi/2)$ window (i.e., the first quadrant) exhibit the correct periodicity representing the fold number of the shape for all but nine shapes. The plots of Δ-OII(θ) using the $(\pi/4)$ window have resolved the problems of the $(\pi/2)$ window with these nine shapes. Correct results have been observed with the $(\pi/8)$ window as well.

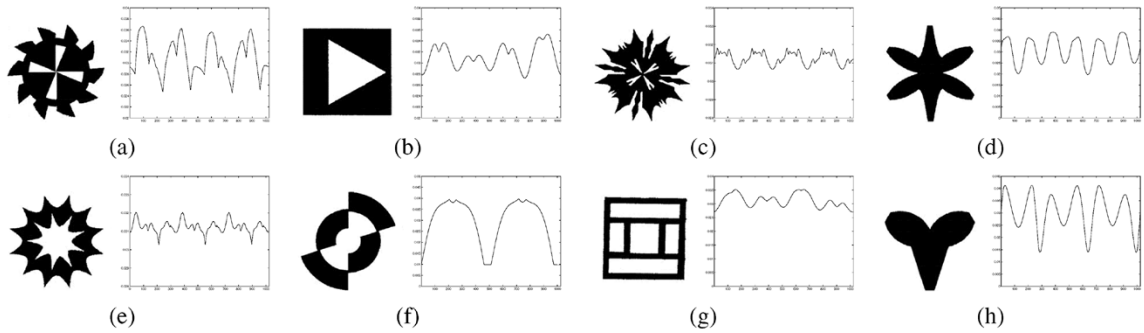


Fig. 4. Difficult shapes and their Δ-OII(θ) plots. (a) Shape 1. (b) Shape 2. (c) Shape 3. (d) Shape 4. (e) Shape 5. (f) Shape 6. (g) Shape 7. (h) Shape 8.

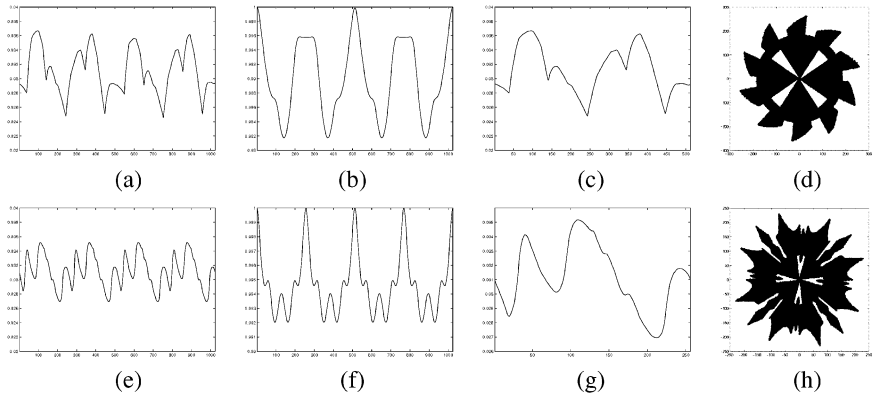


Fig. 5. Examples: Reorienting Shapes 1 and 3 in Fig. 4(a) and (c). (a) Δ-OII of Shape 1. (b) Circ. autocorrelation. (c) Δ-OII single period. (d) Reoriented Shape 1. (e) Δ-OII of Shape 3. (f) Circ. autocorrelation. (g) Δ-OII single period. (h) Reoriented Shape 3.

G. Reorientation of Shapes Using Δ-OII and PRA

We now develop an algorithm for shape orientation using Δ-OII and PRA. The basic idea is as follows. Since Δ-OII(θ) is a function of the rotation angle $\theta \in [0, 2\pi)$, if we locate a unique peak point of the plot and reorient the shape (i.e., rotate and reflect it) until this unique point is brought to the origin at $\theta = 0$, the shape can be brought to its unique normalized orientation. To achieve this, we first determine the fold number of the shape by plotting its Δ-OII(θ). Then, we pick out the nonperiodic portion of the plot, corresponding to its fundamental period, and denote this partial Δ-OII plot as the fundamental list of angles $\mathbf{L}$. We find the unique peak in $\mathbf{L}$ by comparing the right-side and left-side neighbors as has been done in PRA. When we locate the unique peak in $\mathbf{L}$, we perform an appropriate rotation and reflection as described in Section IV-D to bring the shape to the unique orientation. We make a set of statements that are necessary to develop the algorithm outlined above.

Lemma 11: The periodicity of Δ-OII(θ) is obtained by computing its circular autocorrelation and counting the number of peaks with magnitude 1 in the resulting autocorrelation.

A Δ-OII(θ) plot with periodicity d is periodic in the rotation angle θ with period $T = 2\pi/d$. That is, the plot is indistinguishable every time it is circularly shifted by the rotation $\theta = 2\pi/d$ rad. This happens d times over the circular shift of the plot in the computation of the circular autocorrelation, resulting in d peaks with magnitude 1 in the autocorrelation. For example, consider the plot of Δ-OII(θ) with periodicity $d = 2$ in Fig. 5(a). This is the Δ-OII(θ) plot for Shape 1 in Fig. 4(a) using a $(\pi/4)$ window. Its circular autocorrelation is shown in Fig. 5(b). We see that there are two peaks in the autocorrelation plot with magnitude 1 as expected.

We have experimentally verified with all the 200 symmetric and nonsymmetric shapes in the database [40] the following statement that we use below in the combined Δ-OII-PRA algorithm:

> "The fold number of a shape is given by the periodicity of Δ-OII(θ); this periodicity is unchanged across the window sizes $\Delta = 2\pi/4k$ and $2\pi/4(k+1)$ for an integer k."

The first part of the statement refines Property 2. According to this statement, the fold number of the shape is obtained by computing the periodicity of Δ-OII(θ) with the $2\pi/4k$-and $2\pi/4(k+1)$ windows until the two plots display the same periodicity. In our experiments, we have observed that the window size of $\Delta = \pi/4$ solves for the fold number for every shape tested.

Given Δ-OII(θ) with its periodicity, d, matched to the fold number of the shape, we designate the nonperiodic portion of the plot $\theta = [0, 2\pi/d]$, which corresponds to the fundamental period of the plot, as the partial plot $\mathbf{L}$. $\mathbf{L}$ contains a finite number of Δ-OII values that are nonperiodic. Then, we apply the same procedure described in PRA to locate a unique element of $\mathbf{L}$, and then to rotate and reflect the shape until this unique element is brought to the origin $\theta = 0$. The unique reorientation of the shape is achieved in this way by combining Δ-OII and PRA. In our example with Shape 1 in Fig. 4(a), we take the first half of the Δ-OII plot to be the nonperiodic list $\mathbf{L}$ as shown in Fig. 5(c). There is a single peak in the list $\mathbf{L}$ located at $\theta = 3\pi/16$. We

also observe that the left-side of this peak has larger values than the right-side of the peak. We, thus, first rotate the shape by an angle $\theta = -3\pi/16$ and then reflect it about the x axis to arrive at the unique orientation. The reoriented Shape 1 of Fig. 4(a) is shown in Fig. 5(d).

We provide another example using Shape 3 in Fig. 4(c). We obtain the Δ-OII plot of this shape using a $(\pi/4)$ window as shown in Fig. 5(e). The plot exhibits a periodicity of 4. The circular autocorrelation of the Δ-OII plot, as shown in Fig. 5(f), verifies the periodicity with four peaks with the magnitude of 1. We take the first quarter of the Δ-OII plot and denote it as the nonperiodic partial plot $\mathbf{L}$ as shown in Fig. 5(g). This $\mathbf{L}$ has a single peak at the location $\theta = 55\pi/256$. The right-side neighbors of this peak are larger than the left-side neighbors and there is no need for a final reflection. The final reoriented shape is obtained by rotating Shape 3 in Fig. 4(c) by an angle $\theta = -55\pi/256$ as in Fig. 5(h).

H. Combined Δ-OII-PRA Algorithm

The shape reorientation algorithm using Δ-OII and PRA is as follows. We start by setting the counter $k = 1$.

- **Step 1**: Given a shape as input, generate a pair of Δ-OII plots using $(2\pi/4k)$ and $(2\pi/4(k+1))$ windows.
- **Step 2**: Compute the periodicity d_1 and d_2 of the two plots.
- **Step 3**: If $d_1 = d_2$, set the nonperiodic portion $\theta = [0, 2\pi/d_1]$ of the plot generated by the $(2\pi/4k)$ window as $\mathbf{L}$. Otherwise, set $k = k+1$ and go to Step 1. This $\mathbf{L}$ plays the role of the fundamental list of angles in step 6) in PRA.
- **Step 4**: Locate and denote the unique peak in $\mathbf{L}$. This corresponds to finding the reorientation point in step 7) in PRA.
- **Step 5**: Rotate and reflect the shape until this unique element is brought to $\theta = 0$. This is step 8) in PRA.
- **Step 6**: End.

V. Experiments

In this section, we present some experimental results on BLAISER. First, in Section V-A, we discuss the robustness of BLAISER to error and noise. We study each step of BLAISER in this regard and illustrate the performance with examples. In Section V-B, we apply BLAISER to real-world images that are obtained by a digital camera at different viewing angles and distances. The results demonstrate the validity of the affine-permutation distortion model in the actual imaging process and the performance of BLAISER in it.

A. Robustness

In Section II, the concept of intrinsic shape has been developed under an ideal error-free environment. In practice, however, error and noise arise from many sources such as discretization of sampling grids, binary thresholding, rectangular to polar coordinates conversion, and background noise. In this section, we discuss the robustness of BLAISER to error and noise and show that the intrinsic shapes are good affine-permutation invariants that are stable under small perturbations.

Centering and Reshaping: The robustness of moment-based normalization in steps 1 and 2 of BLAISER has been discussed and demonstrated in [11]. The underlying idea is that the first and second-order moments of the shape computed in these steps are relatively insensitive to small data perturbations.

Orientation: In Section IV, Δ-OII is introduced to improve the robustness of PRA to error and noise. When Δ-OII is used, each element of the fundamental list of angles $\mathbf{L}$ is no longer computed from the exact location of each pixel as in PRA, but is computed from the third-order moments of the corresponding segment of the shape. This greatly reduces the sensitivity to small perturbations due to grid digitization, rectangular to polar conversion, and background noise. The same applies to the computation of the fold numbers and other similar computations performed in PRA.

Sorting: After removing the effect of affine distortion from the shape, we eliminate the permutation factor $\mathbf{P}$ by sorting the columns of $\mathbf{X}^u$ as specified in the sorting step of BLAISER. This provides a one-to-one pointwise correspondence when comparing copies of intrinsic shapes from the same object. Error and noise pixels disrupt this correspondence relationship. We can still do a good job in the presence of error and noise by performing an "approximate matching" of points. This is because the corresponding points will very likely be found near the expected locations after sorting the noisy shapes. So, rather than an exact pixel-to-pixel correspondence, we modify the algorithm to accomplish a small region-to-region correspondence. This is now robust to error and noise as we will see in the examples.

BLAISER: The performance of BLAISER with noisy images or images with erroneous pixels depends on several factors, including the amount and spatial distribution of the noise and error corrupting the images. We discuss briefly the robustness of BLAISER and illustrate it with examples. Consider the shape of the airplane in Fig. 6(a) and the plot of its Δ-OII(θ) in Fig. 6(b). The shape in Fig. 6(a) consists of 1208 pixels. We randomly add noise pixels to this shape and observe its effect on every step of BLAISER. Fig. 6(c), (e), and (g) shows the original image with 60, 120, and 241 noise pixels added, respectively, i.e., with 5%, 10%, and 20% of the total number of pixels added to the original shape. The results of BLAISER are shown in Table I. In the centering step, the center of mass (g_x, g_y) drifts slightly from the origin $(0,0)$. In the reshaping step, the second-order moments are set to $m_{20} = 1$, $m_{11} = 0$, and $m_{02} = 1$. Since the computation of these moments includes the noise/error pixels, the normalized shape $\mathbf{X}^n$ that results may be slightly distorted from the exact intrinsic shape. In the reorientation step, as more noise pixels are added to the shape, the peak of the Δ-OII plot shifts away from the expected location. This induces small errors in the rotation angles $\Delta\theta$ [see the Δ-OII plots shown in Fig. 6(d), (f), and (h)]. Even though the plots are distorted by the noise pixels, the location of the peak stays relatively unchanged. This stability helps the correct recovery of the shape orientation using Δ-OII.

Table I shows the performance of BLAISER for 6 different levels of noise/error, including the three levels considered in

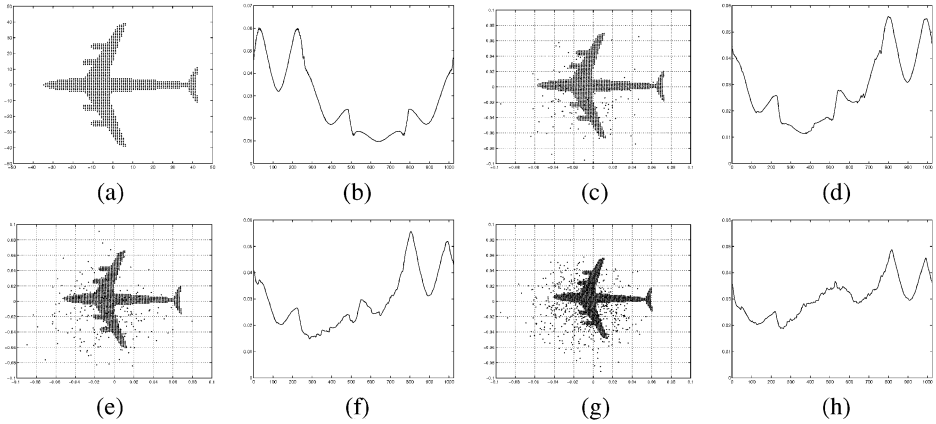


Fig. 6. Robustness to error. (a) Airplane shape. (b) Δ-OII of (a). (c) 10% added noise. (d) Δ-OII of (c). (e) 20% added noise. (f) Δ-OII of (e). (g) 50% added noise. (h) Δ-OII of (g).

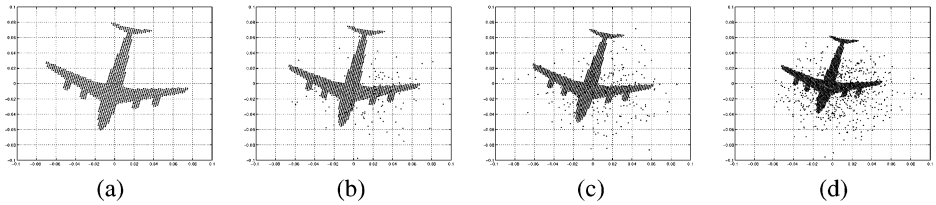


Fig. 7. Intrinsic shapes with added noise. (a) No noise. (b) 10% noise. (c) 20% noise. (d) 50% noise.

TABLE I
PERFORMANCE OF BLAISER UNDER NOISE

No. Noise Pixels	Centering (g_x, g_y)	Reorientation $\Delta\theta$
12 (1%)	(0, 0)	0°
24 (2%)	(0, 0)	0.7°
60 (5%)	(0, -1)	0.7°
120 (10%)	(-1, -1)	1.4°
241 (20%)	(-2, -2)	3.9°
604 (50%)	(-3, -3)	7.7°

Fig. 6. Observe that the drift in the center of mass is only 1 pixel in each x and y directions even with 120 noise pixels added. The rotation error is close to 1° in this case. The intrinsic shapes obtained in 4 of these cases are shown in Fig. 7. The intrinsic shapes appear similar to each other, with possibly some shrinking for the shape with the 50% error/noise. This is due to the noise pixels affecting the computation of the normalized shapes in the reshaping step as discussed above.

Occlusion: As we close the discussion on robustness, we comment briefly on the problem of occlusion in relation to the intrinsic shape computation. The shape normalization performed by BLAISER is a global operation applied to the entire shape. Therefore, it is not very effective with shapes that are occluded. Intrinsic shapes obtained from the occluded shapes will look different from one another. This is because the occluded shapes are no longer related by affine-permutation distortions, thus, do not belong to the same orbit.

B. Application to Real Images

Up to this point, the theory and algorithm for intrinsic shape recovery by BLAISER have been studied and illustrated using computer-generated synthetic shapes. In this section, we apply BLAISER to real world images.

To verify the affine-permutation shape distortion model and the performance of BLAISER in practice, we study 2-D planar shapes obtained from different viewing angles and distances. First, a grayscale image of an airplane is printed on a letter-size plain white paper. The physical dimensions of the shape on the printed page fit into a 4×5 in rectangle. Then, the printed image is placed on a flat floor and is photographed by a digital camera at different viewing angles and distances. The photos have been taken at about 4 to 5 feet away from the image with a 2X optical zoom. The results are shown in Fig. 8(a), (e), (i), and (m). Some blurring has been introduced to the images due to hand movement during the photo shots. The shapes appear in different sizes, orientations, and skews. The dimensions of the grayscale images are shown in Table II. Each pixel value in these images has an 8-bit accuracy and ranges from 0 to 255. In order to feed these shapes into BLAISER as input, we first apply to these grayscale images binary thresholding with the threshold value at 128 and obtain binary images. The dark pixels at value 0 after thresholding are denoted as feature points of the shapes.

Because of different viewing angles and different sizes of the images, the total number of feature points in each shape is different as shown in Table II. The first two steps of BLAISER, centering and reshaping, are applied to these binary images to obtain the normalized shapes in Fig. 8(b), (f), (j), and (n). Note that translation, scaling, and skewing distortions are removed in these shapes. The resulting shapes are still oriented at different angles. Then, in the reorientation step, the Δ-OII is computed and plotted for each shape [see Fig. 8(c), (g), (k), and (o)]. The Δ-OII plots of the normalized shapes are cyclically shifted versions of one another, as expected. The maximum points in the

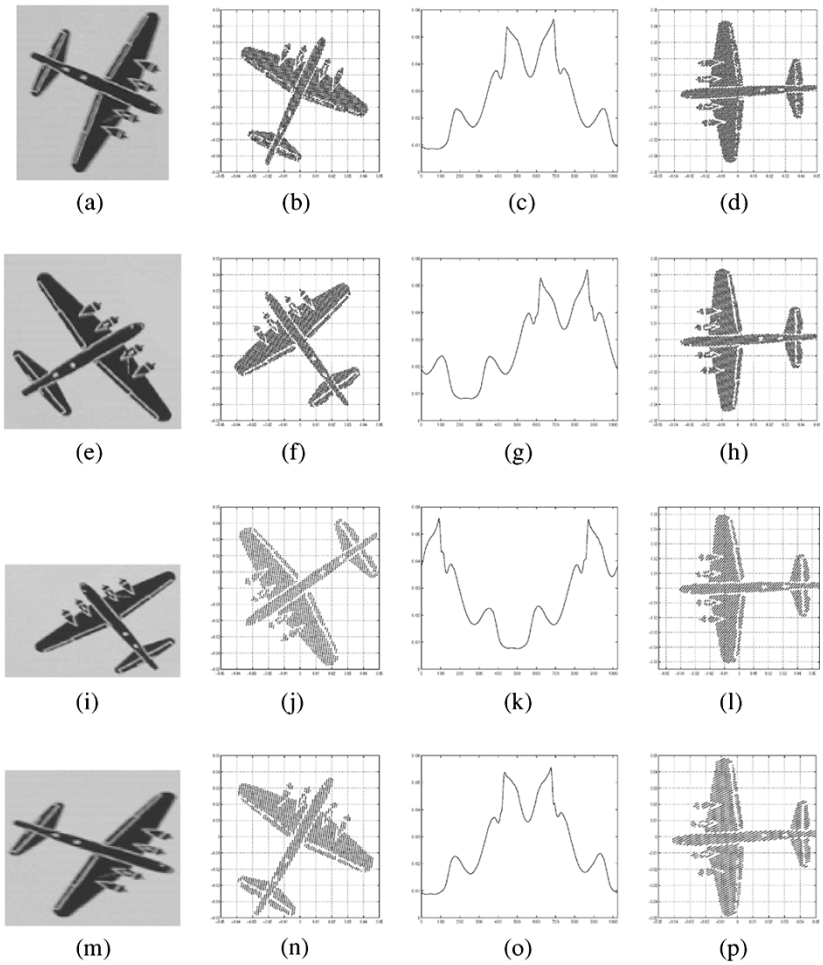


Fig. 8. Application of BLAISER to real data, "airplane." (a) Airplane a $\mathbf{X}_a$. (b) Normalized shape $\mathbf{X}_a^n$. (c) Δ-OII plot. (d) Intrinsic shape $\mathbf{S}_a$. (e) Airplane b $\mathbf{X}_b$. (f) Normalized shape $\mathbf{X}_b^n$, (g) Δ-OII plot. (h) Intrinsic shape $\mathbf{S}_b$. (i) Airplane c $\mathbf{X}_c$. (j) Normalized shape $\mathbf{X}_c^n$. (k) Δ-OII plot. (l) Intrinsic shape $\mathbf{S}_c$. (m) Airplane d $\mathbf{X}_d$. (n) Normalized shape $\mathbf{X}_d^n$. (o) Δ-OII plot. (p) Intrinsic shape $\mathbf{S}_d$.

TABLE II
EXPERIMENT WITH AIRPLANE IMAGES

Image	Dimension	# Feature Points	Rotation Angle θ
Airplane a	105×121	2879	242.23°
Airplane b	116×115	2765	304.10°
Airplane c	127×82	2105	31.99°
Airplane d	109×97	2291	347.66°

Δ-OII plots are searched and found at the locations shown as rotation angle θ in Table II for the normalized shapes $\mathbf{X}_a^n$, $\mathbf{X}_b^n$, $\mathbf{X}_c^n$, and $\mathbf{X}_d^n$, respectively. We rotate the normalized shapes by these angles in the clockwise direction to bring them to the same orientation. After simple rearranging of the columns of these shapes by the sorting step, we arrive at the intrinsic shapes $\mathbf{S}_a$, $\mathbf{S}_b$, $\mathbf{S}_c$, and $\mathbf{S}_d$, in Fig. 8(d), (h), (l), and (p), respectively. Observe that all four intrinsic shapes are very similar to one another, except that the intrinsic shapes $\mathbf{S}_c$ and $\mathbf{S}_d$ appear to be a little larger than the other two. This results from the fact that the shape distortions present in the shapes $\mathbf{X}_c$ and $\mathbf{X}_d$ are beyond those described by the affine-permutation distortion model, e.g., perspective and blurring. However, all of the intrinsic shapes obtained by BLAISER still look very much alike and are located at the same orientation. It is very clear that the intrinsic shapes are much more useful and effective than the distorted shapes themselves for the purpose of detection, identification, and classification of these shapes.

We carried out a second set of experiments with the shapes in Fig. 9, "Camel and Car" (see Table III). The intrinsic shapes obtained after applying BLAISER are shown in Fig. 9(d) and (h). These results again verify the good performance of BLAISER.

VI. CONCLUSION

In this paper, we introduced the notion of intrinsic shape as the unique representative of the orbit that contains all affine-permutation distorted shapes of the object. We presented BLAISER that recovers the intrinsic shape of the unknown object in a *blind* manner. We applied with success BLAISER to the about two hundred symmetrical and nonsymmetrical shapes in the database [40]. We also illustrated its robustness to error and noise. BLAISER can be applied to many interesting problems in image processing and computer vision. The concept of intrinsic shape is extended to 3-D shapes in [10].

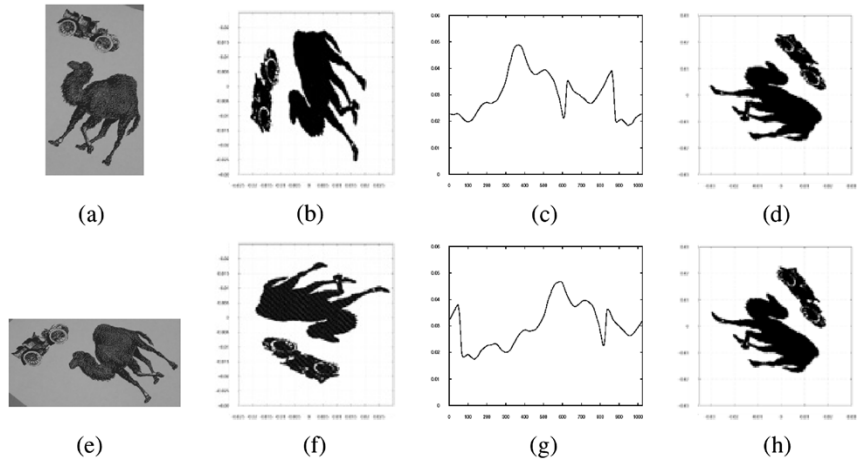


Fig. 9. Application of BLAISER to more real data, "Camel and Car." (a) Camel and Car a $\mathbf{X}_a$. (b) Normalized shape $\mathbf{X}_a^n$. (c) Δ-OII plot. (d) Intrinsic shape $\mathbf{S}_a$. (e) Camel and car b $\mathbf{X}_b$. (f) Normalized shape $\mathbf{X}_b^n$. (g) Δ-OII plot. (h) Intrinsic shape $\mathbf{S}_b$.

TABLE III
EXPERIMENT WITH CAMEL AND CAR IMAGES

Image	Dimension	# Feature Points	Rotation Angle θ
Camel and Car a	137×240	10876	$129.73°$
Camel and Car b	277×142	11289	$206.72°$

REFERENCES

[1] D. Casasent and G. Ravichandran, "Advanced distortion-invariant minimum average correlation energy (MACE) filters," *Appl. Opt.*, vol. 31, no. 8, pp. 1109–1116, Mar. 1992.

[2] B. V. K. V. Kumar, "Tutorial survey of composite filter designs for optical correlators," *Appl. Opt.*, vol. 31, no. 23, pp. 4773–4801, Aug. 1992.

[3] S. A. Benno and J. M. F. Moura, "Scaling functions robust to translations," *IEEE Trans. Signal Process.*, vol. 46, no. 12, pp. 3269–3281, Dec. 1998.

[4] C. He and J. M. F. Moura, "Robust detection with the gap metric," *IEEE Trans. Signal Process.*, vol. 45, no. 6, pp. 1591–1604, Jun. 1997.

[5] V. H. S. Ha and J. M. F. Moura, "Affine invariant wavelet transform," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, vol. 3, Salt Lake City, UT, May 2001, pp. 1937–1940.

[6] V. H. S. Ha, "Blind recovery of intrinsic shape," Ph.D. dissertation, Dept. Elect. Comput. Eng., Carnegie Mellon Univ., Pittsburgh, PA, Aug. 2002.

[7] V. H. S. Ha and J. M. F. Moura, "Intrinsic shape," in *Proc. 36th Asilomar Conf. Signals, Systems, and Computers*, vol. 1, Monterey, CA, Nov. 2002, pp. 139–143.

[8] D. Sepiashvili, J. M. F. Moura, and V. H. S. Ha, "Affine-permutation symmetry: Invariance and shape space," *Proc. 12th IEEE Signal Processing Workshop on Statistical Signal Processing*, pp. 293–296, 2003.

[9] V. H. S. Ha and J. M. F. Moura, "Efficient 2D shape orientation," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, Sep. 2003, pp. 225–228.

[10] ——, "Three-dimensional intrinsic shapes," presented at the IEEE Int. Conf. Image Processing, Oct. 2004.

[11] K. Åström, "Affine and projective normalization of planar curves and regions," in *Proc. Eur. Conf. Computer Vision*, vol. II, 1994, pp. 439–448.

[12] ——, "Affine invariants of planar sets," in *Proc. SCIA8*, Tromso, Norway, 1993, pp. 769–776.

[13] D. G. Kendall, "The diffusion of shape," *Adv. Appl. Prob.*, vol. 9, pp. 428–430, 1977.

[14] D. G. Kendall, D. Barden, T. K. Carne, and H. Le, *Shape and Shape Theory*. New York: Wiley, 1999.

[15] I. L. Dryden and K. V. Mardia, *Statistical Shape Analysis*. New York: Wiley, 1998.

[16] U. Grenander, *Lectures in Pattern Theory*. New York: Springer-Verlag, 1976.

[17] ——, *Abstract Inference*. New York: Springer-Verlag, 1981.

[18] ——, *Tutorial in Pattern Theory*. Providence, RI: Brown Univ., 1983.

[19] U. Grenander and D. M. Keenan, "On the shape of plane images," *SIAM J. Appl. Math.*, vol. 53, no. 4, pp. 1072–1094, Aug. 1993.

[20] U. Grenander and M. Miller, "Representations of knowledge in complex systems," *J. Roy. Stat. Soc. B*, vol. 56, no. 4, pp. 549–603, 1994.

[21] O. Schreier and E. Sperner, *Introduction to Modern Algebra and Matrix Theory*. New York: Chelsea, 1951.

[22] J. R. Magnus and H. Neudecker, *Matrix Differential Calculus*. New York: Wiley, 1988, ch. 2. Kronecker products, the vec operator and the Moore-Penrose inverse.

[23] J. Mundy and A. Zisserman, *Geometric Invariance in Computer Vision*. Cambridge, MA: MIT Press, 1992.

[24] A. Graham, *Kronecker Products and Matrix Calculus with Applications*. Chichester, U.K.: Ellis Horwood, 1981.

[25] S. MacLane and G. Birkoff, *Algebra*. London, U.K.: MacMillan, 1967.

[26] M. A. Armstrong, *Groups and Symmetry*. New York: Springer-Verlag, 1980.

[27] J. G. Leu, "Shape normalization through compacting," *Pattern Recognit. Lett.*, vol. 10, pp. 243–250, Oct. 1989.

[28] A. K. Jain, *Fundamentals of Digital Image Processing*. New York: Prentice-Hall, 1989.

[29] A. Taza and C. Y. Suen, "Discrimination of planar shapes using shape matrices," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, no. 5, pp. 1281–1289, Oct. 1989.

[30] G. Marola, "On the detection of the axes of symmetry of the symmetric and almost symmetric planar image," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 1, pp. 104–108, Jan. 1989.

[31] S. C. Pei and C. N. Lin, "Normalization of rotationally symmetric shapes for pattern recognition," *Pattern Recognit.*, vol. 25, no. 9, pp. 913–920, 1992.

[32] S. C. Pei and L. G. Liou, "Automatic symmetry determination and normalization for rotationally symmetric 2D shapes and 3D solid objects," *Pattern Recognit.*, vol. 27, no. 9, pp. 1193–1208, 1994.

[33] W. H. Tsai and S. L. Chou, "Detection of generalized principal axes in rotational symmetric shapes," *Pattern Recognit.*, vol. 24, pp. 95–104, 1991.

[34] S. L. Chou, J. C. Lin, and W. H. Tsai, "Fold principal axis—A new tool for defining the orientations of rotationally symmetric shapes," *Pattern Recognit. Lett.*, vol. 12, pp. 109–115, 1991.

[35] J. C. Lin, S. L. Chou, and W. H. Tsai, "Detection of rotationally symmetric shape orientation by fold-invariant shape-specific points," *Pattern Recognit.*, vol. 25, pp. 473–482, 1992.

[36] J. J. Leou and W. H. Tsai, "Automatic rotational symmetric determination for shape analysis," *Pattern Recognit.*, vol. 20, no. 6, pp. 571–582, 1987.

[37] J. C. Lin, W. H. Tsai, and J. A. Chen, "Detecting number of folds by a simple mathematical property," *Pattern Recognit.*, vol. 15, pp. 1081–1088, 1994.

[38] J. C. Lin, "Universal principal axes: An easy-to-construct tool useful in defining shape orientations for almost every kind of shape," *Pattern Recognit.*, vol. 26, no. 4, pp. 485–493, 1993.

[39] D. Shen and H. Ip, "General affine invariant image normalization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 5, pp. 431–440, May 1997.

[40] D. Shen and H. H. S. Ip, "Symmetry detection by generalized complex (GC) moments: A close-form solution," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 466–476, May 1999.

Victor H. S. Ha (S'00–M'02) received the B.A.Sc. degree in computer engineering from the University of Toronto, Toronto, ON, Canada, in 1995, the M.Eng. degree in electrical engineering from Cornell University, Ithaca, NY, in 1996, and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2002.

From 1996 to 1997, he was with LCCI, McLean, VA, as a Wireless/PCS Network Design Engineer. In 2001, he was with Apple Computer, Cupertino, CA, as an Intern on the QuickTime software development team. In 2002, he joined the Mobile Solution Laboratory, Digital Media R&D Center of Samsung Electronics, Suwon, Korea, where he worked on developing software and hardware implementations of MPEG-4 AVC/H.264-based video codec. He and the members of the Mobile Solution Laboratory designed the world's first implementation of the portable receivers for Digital Multimedia Broadcasting (DMB) in Korea. Since 2004, he has been working on video processing and image enhancement algorithms for HDTV in the Digital Media Solutions Laboratory, Samsung Information Systems America, Irvine, CA. His research interests include image and video compression, image enhancement, pattern recognition, and shape theory.

José M. F. Moura (S'71–M'75–SM'90–F'94) received the engenheiro electrotécnico degree from the Instituto Superior Técnico (IST), Lisbon, Portugal, in 1969, and the M.Sc., E.E., and the D.Sc. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology (MIT), Cambridge, in 1973 and 1975, respectively.

He has been a Professor of electrical and computer engineering at Carnegie Mellon University, Pittsburgh, PA, since 1986. From 1999 to 2000, he was a Visiting Professor of electrical engineering at MIT. Prior to this, he was on the faculty of IST from 1975 to 1984. He was Genrad Associate Professor of electrical engineering and computer science (Visiting) at MIT from 1984 to 1986 and a Visiting Research Scholar at the Department of Aerospace Engineering, University of Southern California, Los Angeles, during the summers from 1978 to 1981. He has published over 270 technical contributions, is the co-editor of two books, holds four U.S. patents on image and video processing and digital communications, and has given numerous invited seminars at U.S. and European Universities and Laboratories. His research interests include statistical signal processing and telecommunications, image processing, video representations.

Dr. Moura is a member of Sigma Xi, AMS, AAAS, IMS, and SIAM, and he is a corresponding member of the Academy of Sciences of Portugal (Section of Sciences). He has been elected President Elect for the period 2007 to 2008 for the IEEE Signal Processing Society and to serve as President in 2008 and 2009. He served as Vice-President for Publications for the IEEE Signal Processing Society (SPS) and was a member of the Board of Governors of the SPS from 2000 to 2002. He was also Vice-President for Publications for the IEEE Sensors Council from 2000 to 2002. He is on the Editorial Board of the PROCEEDINGS OF THE IEEE. He chaired the IEEE TAB Transactions Committee in 2002 and 2003. He was the Editor-in-Chief for the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1995 to 1999. He has been a member of several Technical Committees of the SPS. He was on the IEEE Press Board from 1991 to 1995. He received the IEEE Third Millennium Medal and the 2003 IEEE Signal Processing Society Meritorious Service Award.