TRCS 97-02

A New Framework for Image Invariants using Basis Expansion*

Yuan-Fang Wang
Department of Computer Science
University of California
Santa Barbara, CA 93106
E-mail: yfwang@cs.ucsb.edu
Web: http://www.cs.ucsb.edu/ yfwang

Abstract

We propose a general framework for computing invariant features from images. The proposed approach is based on a simple concept of basis expansion. It is widely applicable to many popular basis representations, such as wavelets [4, 5, 24, 25], short-time Fourier analysis [15, 30], and splines [2, 6, 33]. Exploiting formulations that use both global and local information about shape and color, the new approach is neither strictly global nor local. It has the advantage of tolerating a certain degree of occlusion (unlike global analysis) and does not require estimating high-order derivatives in computing invariants (unlike local analysis), whence is more robust. Furthermore, it enables a quasi-localized, hierarchical shape analysis which is not possible with other known invariant techniques. Unlike most current research on image invariants which concentrates on either geometry or illumination invariants, the proposed framework is very general and produces invariants which are insensitive to rigid motion, general affine transform, changes of parameterization and scene illumination, and perspective transform.

1 Introduction

Image features and shape descriptors that capture the essential traits of an object and are insensitive to environmental changes are ideal for recognition. The search for invariants (e.g., algebraic and projective invariants) is a classical problem in mathematics dating back to the 18th century [8, 23, 31]. The need for invariant image descriptors has long been recognized in computer vision [31, 42]. Invariant features form a compact, intrinsic description of an object, and can be used to design recognition algorithms that are potentially more efficient than, say, the aspect-based approaches [7, 9, 10]. Hence, it was even argued that object recognition is the search for invariants [42].

Invariant features can be designed based on many different methods and made invariant to rigid motion, general affine transform, scene illumination, occlusion, and projection. For examples, popular image invariants have been designed based on Fourier transform [1] and moments [20, 34, 38]. Many such invariant features are insensitive to changes in geometry induced by motion and change of viewpoint [1, 14, 20, 32, 34, 38, 41, 43], while others are immune to changes in illumination [13, 18, 19, 17, 22, 26, 27, 28, 36, 37, 40], and occlusion [3]. Invariants can be computed either

*Originally submitted on February 7th, revised on March 21st globally, such is the case in moment invariants, or based on local properties such as curvature and arc length. See [31, 42] for a comprehensive survey of the subject of invariants.

The proposed framework for image-derived invariants builds upon past research on image invariants and basis expansion. It is a unified framework which produces invariants insensitive to rigid motion, general affine transform, changes of parameterization and scene illumination, noise, partial occlusion, perspective transform, and view point change. Exploiting formulations that use both global and local information about shape and color, the new approach is neither strictly global nor local. Furthermore, it tolerates a certain degree of occlusion and is robust. We feel that the proposed framework has the following salient features:

1.) It ameliorates some difficulties encountered in computing global or local image invariants. Because the basis functions used (wavelets, short-time Fourier analysis, and splines) all have compact support, the proposed technique tolerates a certain degree of occlusion unlike the invariants based on global analysis. Furthermore, it does not require estimating any derivatives such as in the invariants using local analysis. Whence, the new method is more robust.

2.) To the best of our knowledge, it is the only invariant technique that enables a quasi-localized, hierarchical shape analysis. Because we employ basis functions of a compact support, invariant features thus computed capture local shape traits. Furthermore, the additional degree of freedom in designing a basis function—be it the scale in wavelet, the frequency in short-time Fourier analysis, or the polynomial degree in spline (Fig. 1)—gives the technique an additional descriptive power that is not found in other invariant techniques.

3.) Unlike most current research which concentrates exclusively on either geometry invariants or illumination invariants, the proposed framework is very general and produces invariants insensitive to rigid motion, affine transform, changes of parameterization and scene illumination, noise, partial occlusion, and perspective transform (see Sec. 2 and Sec. 3).

4.) The proposed framework is applicable to many basis functions. We propose to use the framework with wavelet, short-time Fourier analysis, and spline bases, which have been widely used in signal and speech processing, image analysis, computer vision, and computer graphics [4, 5, 6, 24, 25, 30, 33].

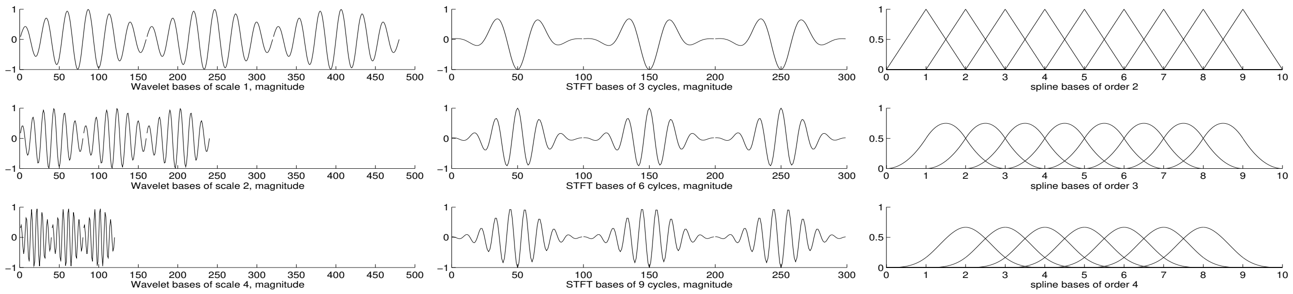
5.) We introduce the rational basis functions to facilitate the analysis of invariants under perspective transform. Though rational basis functions, such as NURBS, have been widely known in the computer graphics community [2, 6, 11, 29, 39], to the best of our knowledge they have never been used in computer vision before.

The remainder of this paper is organized as follows: Sec. 2 presents the framework of image-derived invariants. Sec. 3 presents some preliminary results, and finally, Sec. 4 contains concluding remarks.

2 Technical Rationale

We will illustrate the mathematical frameworks using a specific scenario where invariants for planar curves are sought. The particular basis functions we use in the illustration will be the wavelet bases and spline functions. Though the same framework can be easily extended to other bases such as the short-time Fourier analysis, and for 3D surfaces.

A word on the notational convention: matrices and vectors will be represented by bold-face



(a) (b) (c)

Figure 1: (a) Wavelet, (b) short-time Fourier analysis, and (c) b-spline bases showing two orthogonal dimensions.

characters, such as $\mathbf{M}$ and $\mathbf{V}$, while scalar quantities by plain-face characters such as S. 2D quantities will be in small letters while 3D quantities in capital letters. Hence, a 3D coordinate will be denoted as (X, Y, Z) while a 2D coordinate as (x, y). The symbol ˆ will be used to denote quantities in the Fourier domain. For example, $\mathbf{c}$ and $\hat{\mathbf{c}}$ denote coordinates (bold for vector quantities) of a 2D curve (small letter for 2D quantities) and its Fourier transform, respectively.

We will consider variation in an object's image induced by rigid motion, general affine transform, changes in parameterization and scene illumination, and perspective projection. In each case, we give examples of how invariants can be designed based on simple basis expansion.

Rigid Motion and Affine Transform Consider a 2D curve, where t denotes the arc length,

$$\mathbf{c}(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}$$

and its expansion onto the wavelet basis $\psi_{a,b} = \frac{1}{\sqrt{a}} g(\frac{t-b}{a})$ [5] as

$$\mathbf{u}_{a,b} = \int \mathbf{c}\psi_{a,b} dt \ ,$$

or

$$\mathbf{c}(t) = \int_a \int_b \mathbf{u}_{a,b}\psi_{a,b} da db \ .$$

If the curve is allowed a general affine transform with the transformed curve denoted by:

$$\mathbf{c}'(t) = \mathbf{mc} + \mathbf{t} = \mathbf{m} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} + \mathbf{t} \ ,$$

where $\mathbf{m}$ is any nonsingular 2×2 matrix and $\mathbf{t}$ represents the translational motion, then we have

$$\begin{aligned} \mathbf{u}'_{a,b} &= \int \mathbf{c}'\psi_{a,b} dt & & \\ &= \int (\mathbf{mc} + \mathbf{t})\psi_{a,b} dt &&= \int \mathbf{mc}\psi_{a,b} dt + \int \mathbf{t}\psi_{a,b} dt \\ &= \mathbf{m} \int \mathbf{c}\psi_{a,b} dt + \mathbf{t} \int \psi_{a,b} dt &&= \mathbf{mu}_{a,b} \ , \end{aligned} \tag{1}$$

or

$$\mathbf{c}'(t) = \int_a \int_b \mathbf{u}'_{a,b}\psi_{a,b} da db \ .$$

Note that we use the wavelet property $\int \psi_{a,b} dt = 0$ to simplify the second term in Eq. 1. Hence, the transformed curve can be generated using the transformed wavelet coefficients and the same wavelet bases, instead of transforming the curve point-by-point. This is an observation which is commonly made in the computer graphics community about curves generated by the spline functions and associated control vertices [2, 6, 33]. In that sense, $\mathbf{u}_{a,b}$'s function the same way as the control vertices in a spline curve.

If $\mathbf{m}$ represents a rotation (or the affine transform is a rigid motion of a translation plus a rotation), it is easily seen that invariant features can be derived using the ratio expression

$$\left| \frac{\mathbf{u}'_{a,b}}{\mathbf{u}'_{c,d}} \right| = \left| \frac{\mathbf{m}\mathbf{u}_{a,b}}{\mathbf{m}\mathbf{u}_{c,d}} \right| = \left| \frac{\mathbf{u}_{a,b}}{\mathbf{u}_{c,d}} \right| .$$

If the second term in Eq. 1 is not zero, but is a constant (e.g., for spline functions, the area under a spline basis integrates to a constant 1 for a uniformly spaced knot vector [33]), then invariant expressions can still be derived, albeit in a slightly more complicated form:

$$\left| \frac{\mathbf{u}'_{a,b} - \mathbf{u}'_{c,d}}{\mathbf{u}'_{e,f} - \mathbf{u}'_{g,h}} \right| = \left| \frac{(\mathbf{m}\mathbf{u}_{a,b} + \mathbf{v}) - (\mathbf{m}\mathbf{u}_{c,d} + \mathbf{v})}{(\mathbf{m}\mathbf{u}_{e,f} + \mathbf{v}) - (\mathbf{m}\mathbf{u}_{g,h} + \mathbf{v})} \right| = \left| \frac{\mathbf{m}(\mathbf{u}_{a,b} - \mathbf{u}_{c,d})}{\mathbf{m}(\mathbf{u}_{e,f} - \mathbf{u}_{g,h})} \right| = \left| \frac{(\mathbf{u}_{a,b} - \mathbf{u}_{c,d})}{(\mathbf{u}_{e,f} - \mathbf{u}_{g,h})} \right| ,$$

where $\mathbf{v}$ denotes the constant second term in Eq. 1.

For invariants under general affine transform, many forms using ratios, cross ratios, and ratios of ratios have already been derived [31, 42]. For example, it is known that the cross ratio of four collinear points are invariant under the affine transform, and the area of the triangle formed by any three $\mathbf{u}_{a,b}$ changes linearly in an affine transform (an invariant of weight 1 [31, 42]). So we have:

$$\begin{vmatrix} \mathbf{u'_{a,b}} & \mathbf{u'_{c,d}} & \mathbf{u'_{e,f}} \\ 1 & 1 & 1 \end{vmatrix} = \begin{vmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} \mathbf{u_{a,b}} & \mathbf{u_{c,d}} & \mathbf{u_{e,f}} \\ 1 & 1 & 1 \end{vmatrix} .$$

Hence, we have the following invariants:

$$\frac{\begin{vmatrix} \mathbf{u'_{a,b}} & \mathbf{u'_{c,d}} & \mathbf{u'_{e,f}} \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} \mathbf{u'_{g,h}} & \mathbf{u'_{i,j}} & \mathbf{u'_{k,l}} \\ 1 & 1 & 1 \end{vmatrix}} = \frac{\begin{vmatrix} \mathbf{u_{a,b}} & \mathbf{u_{c,d}} & \mathbf{u_{e,f}} \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} \mathbf{u_{g,h}} & \mathbf{u_{i,j}} & \mathbf{u_{k,l}} \\ 1 & 1 & 1 \end{vmatrix}} . \tag{2}$$

Affine Transform with a Change of Parameterization A more complicated scenario is the one where, in addition to an affine transform, a change of parameterization is also allowed, or

$$\mathbf{c}'(t) = \mathbf{m}\mathbf{c}(\pm t - t_0) + \mathbf{t} ,$$

where t_0 represents the shift of the origin in the parameterization, and $\pm$ represents the possibility of traversing the curve either counterclockwise or clockwise. Allowing an arbitrary change of parameterization and traversal direction in computing invariants implies that *no point correspondence is required.* In more details, we have

$$
\begin{array}{rclcl}
\mathbf{u}'_{a,b} & = & \int \mathbf{c}'\psi_{a,b}dt & & \\
 & = & \int(\mathbf{mc}(\pm t - t_0) + \mathbf{t})\psi_{a,b}dt & = & \int \mathbf{mc}(\pm t - t_0)\psi_{a,b}dt + \int \mathbf{t}\psi_{a,b}dt \\
 & = & \mathbf{m}\int \mathbf{c}(\pm t - t_0)\psi_{a,b}dt + \mathbf{t}\int \psi_{a,b}dt & = & \pm\mathbf{m}\int \mathbf{c}(\pm t - t_0)\psi(\pm t - t_0)_{a,b-t_0}d(\pm t - t_0) \\
 & = & \pm\mathbf{mu}_{a,b-t_0} \; . & &
\end{array} \tag{3}
$$

Whether the expression in Eq. 3 is positive or negative depends on whether the basis function is an odd (most wavelets are) or an even (most spline bases are) function. Now, the resulting wavelet coefficients $\mathbf{u}'_{a,b}$ and $\mathbf{u}_{a,b}$ are functions of two variables: the scale a and the displacement b. If we hold the sale a constant and perform a Fourier analysis on the displacement b, we have:

$$
\begin{array}{rclcl}
\hat{\mathbf{u}}_{a,\omega} & = & \int_b \mathbf{u}_{a,b}e^{-i\omega b}db & & \\
 & = & \int_b \int_t \mathbf{c}\psi_{a,b}dt \; e^{-i\omega b}db & = & \int_t \mathbf{c}\int_b \psi_{a,b}e^{-i\omega b}db \; dt \\
 & = & \int_t \mathbf{c}\sqrt{a}\hat{\psi}(-a\omega)e^{-i\omega t}dt & = & \sqrt{a}\hat{\psi}(-a\omega)\int_t \mathbf{c}e^{-i\omega t}dt \\
 & = & \sqrt{a}\hat{\psi}(-a\omega)\hat{\mathbf{c}} \; , & &
\end{array} \tag{4}
$$

similarly,

$$
\begin{array}{rclcl}
\hat{\mathbf{u}}'_{a,\omega} & = & \pm\int_b \mathbf{mu}_{a,b-t_0}e^{-i\omega b}db & & \\
 & = & \pm\int_b \int_t \mathbf{mc}\psi_{a,b-t_0}dt \; e^{-i\omega b}db & = & \pm\int_t \mathbf{mc}\int_b \psi_{a,b-t_0}e^{-i\omega b}db \; dt \\
 & = & \pm\int_t \mathbf{mc}\sqrt{a}\hat{\psi}(-a\omega)e^{-i\omega(t+t_0)}dt & = & \pm\sqrt{a}\hat{\psi}(-a\omega)\mathbf{m}\int_t \mathbf{c}e^{-i\omega(t+t_0)}dt \\
 & = & \pm\sqrt{a}\hat{\psi}(-a\omega)\mathbf{m}\hat{\mathbf{c}}e^{-i\omega t_0} \; , & &
\end{array} \tag{5}
$$

where $\hat{\mathbf{u}}$ ($\hat{\mathbf{u}}'$) denotes the Fourier transform of $\mathbf{u}$ ($\mathbf{u}'$) with respect to the displacement parameter b, and $\hat{\mathbf{c}}$ and $\hat{\psi}$ represent the Fourier transform of $\mathbf{c}$ and ψ, respectively. It can then be observed that

$$
|\hat{\mathbf{u}}'_{a,\omega}| = \mathbf{m}|\hat{\mathbf{u}}_{a,\omega}|
$$

which is an expression similar in spirit to Eq. 1. The physical meaning of the above equation should be clear: The $\mathbf{u}_{a,b}$ and $\mathbf{u}'_{a,b}$ sequences for a constant a are circular shifted (by t_0) and/or flipped ($\pm$) version of each other, after an affine transform. The shift or flip in space effects a constant phase modulation in the Fourier domain, which is ignored by examining only the transformed magnitude. Hence, for general affine motion with a change of parameterization, the following expression will be invariant (similar to Eq. 2)

$$
\frac{\begin{vmatrix} |\hat{\mathbf{u}}'_{\mathbf{a},\omega_1}| & |\hat{\mathbf{u}}'_{\mathbf{a},\omega_2}| & |\hat{\mathbf{u}}'_{\mathbf{a},\omega_3}| \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} |\hat{\mathbf{u}}'_{\mathbf{a},\omega_4}| & |\hat{\mathbf{u}}'_{\mathbf{a},\omega_5}| & |\hat{\mathbf{u}}'_{\mathbf{a},\omega_6}| \\ 1 & 1 & 1 \end{vmatrix}} = \frac{\begin{vmatrix} |\hat{\mathbf{u}}_{\mathbf{a},\omega_1}| & |\hat{\mathbf{u}}_{\mathbf{a},\omega_2}| & |\hat{\mathbf{u}}_{\mathbf{a},\omega_3}| \\ 1 & 1 & 1 \end{vmatrix}}{\begin{vmatrix} |\hat{\mathbf{u}}_{\mathbf{a},\omega_4}| & |\hat{\mathbf{u}}_{\mathbf{a},\omega_5}| & |\hat{\mathbf{u}}_{\mathbf{a},\omega_6}| \\ 1 & 1 & 1 \end{vmatrix}} \; . \tag{6}
$$

Variation in Lighting Condition So far, all the discussions deal with changes in the geometry, or the shape, of a 2D curve. Another possible variation in the appearance is due to different lighting: By this we mean that objects can be illuminated by light sources of different numbers and types. To simplify the notation, in the following derivation we will consider three spectral bands of red, green, and blue. Generalizing to an n-band illumination model is straightforward. Let $\mathbf{L}(t)$ denote the perceived image color distribution along a curve, we have:

$$\mathbf{L}(t) = \begin{bmatrix} r(t) \\ g(t) \\ b(t) \end{bmatrix} = \int \begin{bmatrix} f^r(\lambda) \\ f^g(\lambda) \\ f^b(\lambda) \end{bmatrix} s(\lambda, t) d\lambda \ ,$$

where λ denotes the wavelength, and $f^r(\lambda)$ the sensitivity of the red sensor (similar interpretations for the green and blue channels). Based on a Lambertian model [12], $s(\lambda, t)$ is

$$s(\lambda, t) = (\sum_{i=1}^{n} l_i(\lambda)\mathbf{N} \cdot \mathbf{N}_i)\rho(\lambda, t) + a(\lambda) \ ,$$

where n is the number of light sources used to illuminate the scene, $l_i(\lambda)$ the source luminance spectral distribution, N the surface normal, N_i the incident direction for source i, $\rho(\lambda, t)$ the surface reflectivity, and $a(\lambda)$ the ambient light luminance.

When the lighting condition changes, because lights are moved, turned on, or turned off, or the ambient light intensity changes, we have

$$\begin{aligned} s'(\lambda, t) &= (\textstyle\sum_j^m l'_j(\lambda)\mathbf{N} \cdot \mathbf{N}'_j)\rho(\lambda, t) + a'(\lambda) \ , \\ &= \frac{\sum_j^m l'_j(\lambda)\mathbf{N}\cdot\mathbf{N}'_j}{\sum_i^n l_i(\lambda)\mathbf{N}\cdot\mathbf{N}_i}[(\textstyle\sum_i^n l_i(\lambda)\mathbf{N} \cdot \mathbf{N}_i)\rho(\lambda, t) + a(\lambda)] + a'(\lambda) - \frac{\sum_j^m l'_j(\lambda)\mathbf{N}\cdot\mathbf{N}'_j}{\sum_i^n l_i(\lambda)\mathbf{N}\cdot\mathbf{N}_i} a(\lambda) \\ &= c_1(\lambda)s(\lambda, t) + c_2(\lambda) \ , \end{aligned}$$

where

$$\begin{aligned} c_1(\lambda) &= \frac{\sum_j^m l'_j(\lambda)\mathbf{N}\cdot\mathbf{N}'_j}{\sum_i^n l_i(\lambda)\mathbf{N}\cdot\mathbf{N}_i} \\ c_2(\lambda) &= a'(\lambda) - \frac{\sum_j^m l'_j(\lambda)\mathbf{N}\cdot\mathbf{N}'_j}{\sum_i^n l_i(\lambda)\mathbf{N}\cdot\mathbf{N}_i} a(\lambda) \end{aligned}$$

capture the changes in the gain and offset in the two different lighting conditions. Following a path similar to that adopted by several researchers [19, 26, 27, 36], we assume that surface reflectance functions are modeled as a linear combination of a small number of basis functions $s_k(\lambda)$, whence,

$$s(\lambda, t) = \sum_k \alpha_k(t) s_k(\lambda) \ ,$$

where $s_k(\lambda)$ denotes the k-th basis function for representing the surface reflectance properties, and $\alpha_k(t)$ is the space varying expansion coefficients. Then using an analysis which is similar to that employed in the affine case, we have

$$
\begin{aligned}
\mathbf{u}_{a,b} &= \int \mathbf{L}\psi_{a,b}dt \\
&= \int_t \int_\lambda \begin{bmatrix} f^r(\lambda) \\ f^g(\lambda) \\ f^b(\lambda) \end{bmatrix} s(\lambda,t)d\lambda\psi_{a,b}dt \\
&= \int_t \int_\lambda \begin{bmatrix} f^r(\lambda) \\ f^g(\lambda) \\ f^b(\lambda) \end{bmatrix} (\textstyle\sum_k \alpha_k(t)s_k(\lambda))d\lambda\psi_{a,b}dt \\
&= \textstyle\sum_k (\int_\lambda \begin{bmatrix} f^r(\lambda)s_k(\lambda) \\ f^g(\lambda)s_k(\lambda) \\ f^b(\lambda)s_k(\lambda) \end{bmatrix} d\lambda)(\int_t \alpha_k(t)\psi_{a,b}dt) \\
&= \textstyle\sum_k \begin{bmatrix} L_k^r \\ L_k^g \\ L_k^b \end{bmatrix} v_{a,b}^k \\
&= \begin{bmatrix} L_1^r & L_2^r & \cdots & L_k^r \\ L_1^g & L_2^g & \cdots & L_k^g \\ L_1^b & L_2^b & \cdots & L_k^b \end{bmatrix} \begin{bmatrix} v_{a,b}^1 \\ v_{a,b}^2 \\ \cdots \\ v_{a,b}^k \end{bmatrix} \\
&= \mathbf{L}_{rgb}\mathbf{v}_{a,b} \ ,
\end{aligned}
$$

where

$$
\begin{aligned}
\begin{bmatrix} L_k^r \\ L_k^g \\ L_k^b \end{bmatrix} &= \int_\lambda \begin{bmatrix} f^r(\lambda)s_k(\lambda) \\ f^g(\lambda)s_k(\lambda) \\ f^b(\lambda)s_k(\lambda) \end{bmatrix} d\lambda \\
v_{a,b}^k &= \int_t \alpha_k(t)\psi_{a,b}dt \ .
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
\mathbf{u}'_{a,b} &= \int \mathbf{L}'\psi_{a,b}dt \\
&= \int_t \int_\lambda \begin{bmatrix} f^r(\lambda) \\ f^g(\lambda) \\ f^b(\lambda) \end{bmatrix} [c_1(\lambda)s(\lambda,t) + c_2(\lambda)]\, d\lambda\psi_{a,b}dt \\
&= \int_t \int_\lambda \begin{bmatrix} f^r(\lambda) \\ f^g(\lambda) \\ f^b(\lambda) \end{bmatrix} [c_1(\lambda)(\textstyle\sum_k \alpha_k(t)s_k(\lambda)) + c_2(\lambda)]\, d\lambda\psi_{a,b}dt \\
&= \textstyle\sum_k \left\{ (\int_\lambda \begin{bmatrix} c_1(\lambda)f^r(\lambda)s_k(\lambda) \\ c_1(\lambda)f^g(\lambda)s_k(\lambda) \\ c_1(\lambda)f^b(\lambda)s_k(\lambda) \end{bmatrix} d\lambda)(\int_t \alpha_k(t)\psi_{a,b}dt) + \int_t \int_\lambda \begin{bmatrix} f^r(\lambda) \\ f^g(\lambda) \\ f^b(\lambda) \end{bmatrix} c_2(\lambda)d\lambda\psi_{a,b}dt \right\} \\
&= \textstyle\sum_k \begin{bmatrix} L_k^{r'} \\ L_k^{g'} \\ L_k^{b'} \end{bmatrix} v_{a,b}^k \\
&= \begin{bmatrix} L_1^{r'} & L_2^{r'} & \cdots & L_k^{r'} \\ L_1^{g'} & L_2^{g'} & \cdots & L_k^{g'} \\ L_1^{b'} & L_2^{b'} & \cdots & L_k^{b'} \end{bmatrix} \begin{bmatrix} v_{a,b}^1 \\ v_{a,b}^2 \\ \cdots \\ v_{a,b}^k \end{bmatrix} \\
&= \mathbf{L}'_{rgb}\mathbf{v}_{a,b} \ .
\end{aligned}
$$

Then it is easily shown that the following expression is invariant under different lighting conditions (similar to Eq. 2):

$$\left| \frac{\left[\mathbf{u}'_{a_1,b_1}\mathbf{u}'_{a_2,b_2}\cdots\mathbf{u}'_{a_k,b_k}\right]^T\left[\mathbf{u}'_{a_1,b_1}\mathbf{u}'_{a_2,b_2}\cdots\mathbf{u}'_{a_k,b_k}\right]}{\left[\mathbf{u}'_{c_1,d_1}\mathbf{u}'_{c_2,d_2}\cdots\mathbf{u}'_{c_k,d_k}\right]^T\left[\mathbf{u}'_{c_1,d_1}\mathbf{u}'_{c_2,d_2}\cdots\mathbf{u}'_{c_k,d_k}\right]} \right| = \left| \frac{\left[\mathbf{v}_{a_1,b_1}\mathbf{v}_{a_2,b_2}\cdots\mathbf{v}_{a_k,b_k}\right]^T\mathbf{L}'^T_{rgb}\mathbf{L}'_{rgb}\left[\mathbf{v}_{a_1,b_1}\mathbf{v}_{a_2,b_2}\cdots\mathbf{v}_{a_k,b_k}\right]}{\left[\mathbf{v}_{c_1,d_1}\mathbf{v}_{c_2,d_2}\cdots\mathbf{v}_{c_k,d_k}\right]^T\mathbf{L}'^T_{rgb}\mathbf{L}'_{rgb}\left[\mathbf{v}_{c_1,d_1}\mathbf{v}_{c_2,d_2}\cdots\mathbf{v}_{c_k,d_k}\right]} \right|$$

$$= \left| \frac{\left[\mathbf{v}_{a_1,b_1}\mathbf{v}_{a_2,b_2}\cdots\mathbf{v}_{a_k,b_k}\right]^T\mathbf{L}^T_{rgb}\mathbf{L}_{rgb}\left[\mathbf{v}_{a_1,b_1}\mathbf{v}_{a_2,b_2}\cdots\mathbf{v}_{a_k,b_k}\right]}{\left[\mathbf{v}_{c_1,d_1}\mathbf{v}_{c_2,d_2}\cdots\mathbf{v}_{c_k,d_k}\right]^T\mathbf{L}^T_{rgb}\mathbf{L}_{rgb}\left[\mathbf{v}_{c_1,d_1}\mathbf{v}_{c_2,d_2}\cdots\mathbf{v}_{c_k,d_k}\right]} \right|$$

$$= \left| \frac{\left[\mathbf{u}_{a_1,b_1}\mathbf{u}_{a_2,b_2}\cdots\mathbf{u}_{a_k,b_k}\right]^T\left[\mathbf{u}_{a_1,b_1}\mathbf{u}_{a_2,b_2}\cdots\mathbf{u}_{a_k,b_k}\right]}{\left[\mathbf{u}_{c_1,d_1}\mathbf{u}_{c_2,d_2}\cdots\mathbf{u}_{c_k,d_k}\right]^T\left[\mathbf{u}_{c_1,d_1}\mathbf{u}_{c_2,d_2}\cdots\mathbf{u}_{c_k,d_k}\right]} \right| .$$

This derivation is in spirit similar to that of [17, 19]. By using a ratio expression, we obtain a much simpler and computationally efficient form of invariants which do not required computing the color correlation matrix and the singular value decomposition of such a matrix [17, 19].

Perspective Transform Finally, we observe that all the above derivations assume a planar curve under a parallel projection. Perspective foreshortening distortion was not addressed. Allowing perspective transform makes the problem much harder as the transformation is a non-linear process involving a division in computing 2D coordinates. Simplification can be made by using a parallel or quasi-perspective model, where the size of the object is assumed to be much smaller than the average distance of the object to the camera. Then the projection can be linearized by replacing the division by varying Z coordinates with a multiplication of a constant magnification factor.

Fortunately, a more rigorous treatment of perspective invariants is possible. The projection process can be linearized using a tool which is well-established in computer graphics, but, to the best of our knowledge, has not been introduced into the computer vision community. The particular tool is the *rational* form of a basis function [2, 6, 11, 29, 39]. The most famous such expression is probably NURBS, stands for Non-Uniform Rational B-Spline, which was adopted as a standard for IGES (Initial Graphics Exchange Specification) [21]. NURBS are invariants under perspective transform. A 2D curve, resulting from the perspective projection of a 3D curve, can be generated by interpolating projected the 2D control vertices using the *rational* bases, instead of interpolating the 3D control vertices in space using the *non-rational* bases and then projecting the curve points one by one. The advantage of using the rational form of a basis function, as will be shown later, is that *the division is absorbed into the rational basis and the representation again becomes linear.*

We will use NURBS to illustrate this invariant property under the perspective transform. In a nutshell, a b-spline function is a polynomial of a finite support. Non-rational b-spline functions of order k (or a polynomial of degree $k-1$) are generated by the Cox-deBoor recursion formulas [33]:

$$N_{i,1}(t) = \begin{cases} 1 & if\ x_i \le t < x_{i+1} \\ 0 & otherwise \end{cases}$$

and

$$N_{i,k}(t) = \frac{(t-x_i)N_{i,k-1}(t)}{x_{i+k-1}-x_i} + \frac{(x_{i+k}-t)N_{i+1,k-1}(t)}{x_{i+k}-x_{i+1}} .$$

The values of x_i are elements of a knot vector satisfying the relations $x_i \le x_{i+1}$. To generate, say, a 2D curve, each spline basis function is modulated by a physical 2D point (usually called the control vertex) and interpolated over t:

$$\mathbf{p}(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \sum_i \mathbf{p}_i N_{i,k}(t) = \sum_i \begin{bmatrix} x_i \\ y_i \end{bmatrix} N_{i,k}(t) \ .$$

As can be easily seen, the same basis expansion idea is evident here. Similar to wavelet functions which are defined on two orthogonal dimensions of scale (a) and shift (b), there are two orthogonal dimensions in defining a b-spline basis function: its cardinality (i) and its order (k). The major difference is that the b-spline functions are positive and sum to 1 (for a uniformly-spaced knot vector), which implies these functions in fact are *not* mutually orthogonal.

The rational form of a b-spline function in 2D (3D) is the projection of a non-rational b-spline function in 3D (4D). Specifically, let

$$\mathbf{P}(t) = \begin{bmatrix} X(t) \\ Y(t) \\ Z(t) \end{bmatrix} = \sum_i \mathbf{P}_i N_{i,k}(t) = \sum_i \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} N_{i,k}(t) \tag{7}$$

represent a non-rational curve in 3D. Its projection in 2D will be:

$$\mathbf{p}(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = \begin{bmatrix} \frac{X(t)}{Z(t)} \\ \frac{Y(t)}{Z(t)} \end{bmatrix} = \sum_i \mathbf{p}_i R_{i,k}(t) = \sum_i \begin{bmatrix} x_i \\ y_i \end{bmatrix} R_{i,k}(t) = \sum_i \begin{bmatrix} \frac{X_i}{Z_i} \\ \frac{Y_i}{Z_i} \end{bmatrix} R_{i,k}(t) \ , \tag{8}$$

where

$$R_{i,k}(t) = \frac{Z_i N_{i,k}(t)}{\sum_i Z_i N_{i,k}(t)} \ .$$

To illustrate, Fig. 2 shows the rational b-spline bases of order four (third-degree polynomial) for an open knot vector $[X] = [000012222]$ [33]. By varying the Z coordinates of the associated control vertices, the shape of the bases adapts accordingly. When all the Z's are equal, the rational bases in fact reduce to the non-rational bases as expected. Fig. 3 shows sample b-spline curves using both periodical and open knot vectors [33]. Column (a) in Fig. 3 shows curves generated using non-rational spline bases in space. Columns (b) and (c) show the projections of the 3D curves using two different methods: One is generating points along the 3D curves in Fig. 3(a), then projecting them one by one. The other is projecting only the control vertices and then using the rational spline bases to interpolate them in 2D. The two methods produce identical results as shown in Fig. 3.

We can now formulate the problem of finding projective invariants as a curve fitting problem. Intuitively speaking, if a 2D curve results from the projection of a 3D curve, then it should be possible to interpolate the observed 2D curve using the projected control vertices and rational spline bases and obtain a good fit. If that is not the case, then the curve probably does not come from the projection of the particular 3D curve. Hence, the error in curve fitting is a measure of invariance (In the ideal case, the error should be zero).

In more details, if we let the "canonical" view of a 3D planar curve be represented as in Eq. 7, then for all other views we have:

$$\mathbf{P'}_i = \begin{bmatrix} X'_i \\ Y'_i \\ Z'_i \end{bmatrix} = \mathbf{M}\mathbf{P}_i + \mathbf{T} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \ , \tag{9}$$

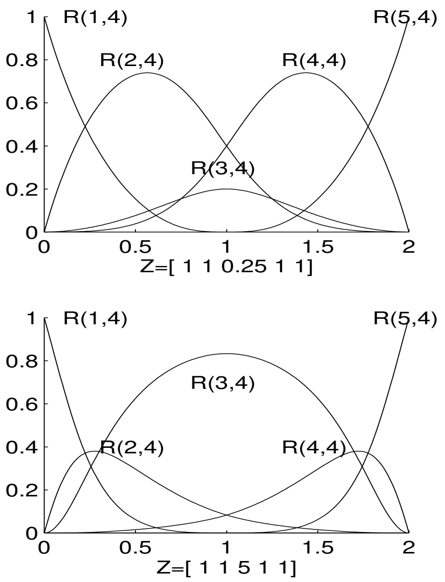


Figure 2: Rational b-spline basis of order 4 with an open knot vector $[X] = [000012222]$ and five control vertices.

where $\mathbf{M}$ and $\mathbf{T}$ represent the rotation and translation in between the two views. For the sake of convenience and without loss of generality, we will assume that the curve lies on the $X - Y$ plane $(Z = 0)$ in the canonical view, one can simplify Eq. 9 as

$$\begin{aligned} \mathbf{P'}_i &= \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ 0 \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \\ &= \begin{bmatrix} r_{11}X_i + r_{12}Y_i + T_x \\ r_{21}X_i + r_{22}Y_i + T_y \\ r_{31}X_i + r_{32}Y_i + T_z \end{bmatrix} . \end{aligned} \tag{10}$$

Hence

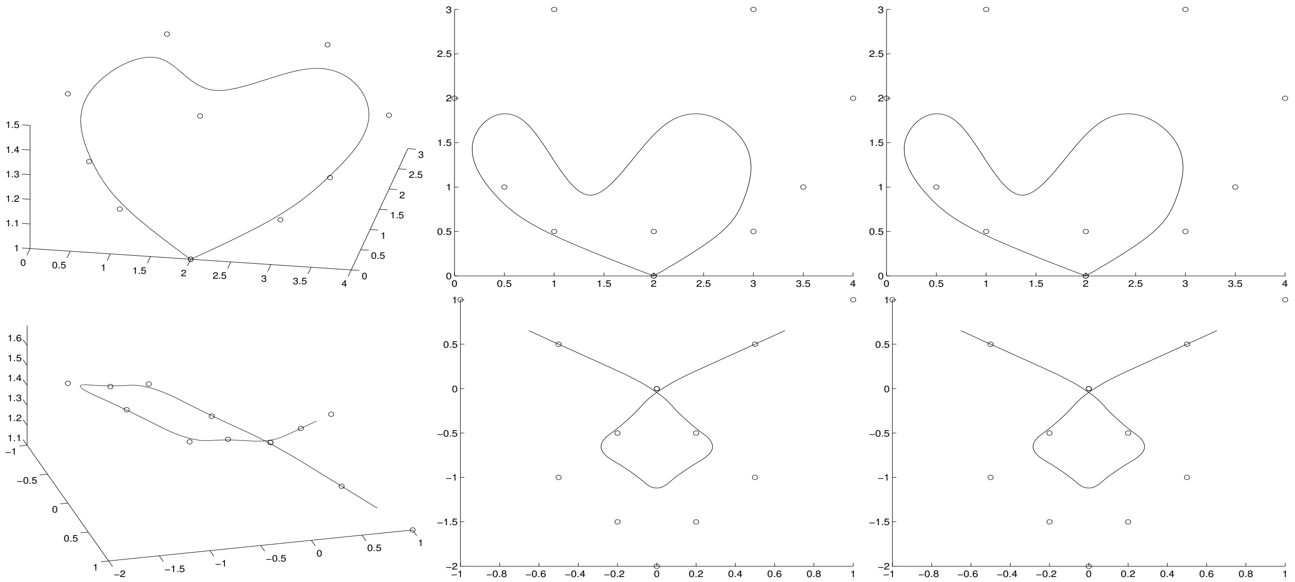
$$\mathbf{p}'_i = \begin{bmatrix} \frac{X'_i}{Z'_i} \\ \frac{Y'_i}{Z'_i} \end{bmatrix} = \begin{bmatrix} \frac{r_{11}X_i + r_{12}Y_i + T_x}{r_{31}X_i + r_{32}Y_i + T_z} \\ \frac{r_{21}X_i + r_{22}Y_i + T_y}{r_{31}X_i + r_{32}Y_i + T_z} \end{bmatrix} \tag{11}$$

$$R'_{i,k} = \frac{(r_{31}X_i + r_{32}Y_i + T_z)N_{i,k(t)}}{\sum_i (r_{31}X_i + r_{32}Y_i + T_z)N_{i,k(t)}} . \tag{12}$$

Hence, image invariant defined by the goodness of fitting is:

$$I = \int_t (\mathbf{D}(t) - \sum_i \mathbf{p}'_i R'_{i,k}(t))^2 , \tag{13}$$

where $\mathbf{D}(t)$ denotes the actual image data. There are a total of nine unknowns in Eqs. 11 and 12, six for rotation and three for translation. The number of rotation unknowns can be reduced to three which specify the direction of the rotation axis and the rotation angle around the axis. However, computing such an invariant can still be very expensive in a general scenario where the camera undergoes an arbitrary motion and a search in a six dimensional space is required.



(a) (b) (c)

Figure 3: (a) 3D b-spline curves generated using non-rational bases, (b) 2D b-spline curves generated by projecting points on the 3D curves one by one, and (c) 2D b-spline curves generated by projecting the control vertices and interpolating them with the rational bases in 2D. "o" marks a control vertex location.

We now introduce a mechanism which drastically reduces the search effort in computing the perspective invariant in Eq. 13. Observe that in Eq. 8 the shape of a 2D curve is determined by the *projected control vertices* and the *rational* spline bases, both of which are unknown. To estimate both the locations of the projected control vertices and the rational spline bases can be a highly nonlinear process. Instead, *Our approach updates them in two separate stages which maintain the linearity of the whole formulation.*

Initialization Specification of the rational basis functions requires knowledge of the depth of the control vertices Z_i, which is not known. As a first approximation we assume that all Z_i's are equal. This is equivalent to approximate the *rational* bases using the corresponding *non-rational* bases, which do *not* require knowledge of the object depth. Using the non-rational bases in place of the rational bases in Eq. 8, we estimate the 2D control vertex positions. These estimated control vertex positions are then used to constrain the unknown rotation and translation parameters using Eq. 11. A *linear* formulation for estimating the unknown motion parameters results using at least four 2D control vertex positions estimated from Eq. 8. (Even though there are nine motion parameters, one of them contributes to a scale change which cannot be determined uniquely.)

Iterative Update The estimated motion parameters can now be used to update the Z_i parameters, whence, a better approximation to the rational spline bases in Eq. 12 results. Furthermore, the control vertex positions can be updated using Eq. 11. We then predict how the curve should look like in the image based on the current estimates of the control vertex positions and the rational bases. Deviation of the predicted shape from the actual shape can then be used to compute a gradient direction along which the projected control vertices should be moved (while holding the rational basis constant). The updated control vertex positions can then be used to estimate the unknown motion parameters through Eq. 11. Hence a recursive process results to refine the positions of the 2D control vertices, the shapes of the rational spline functions, and the 3D motion parameters, until a convergence is achieved. This recursive procedure is summarized in the following pseudo-code program:

- **Step 0** Initialize
  - the iteration number to 0, or $n \longleftarrow 0$
  - $Z_i^{(0)} \longleftarrow 1$ and $R_{i,k}^{(0)} \longleftarrow N_{i,k}$ for all i
  - $r_{i,j}^{(0)} \longleftarrow 0$ and $T_k^{(0)} \longleftarrow 0$, for $i = 1, 2, 3,\ j = 1, 2$, and $k = x, y, z$.
  - Solve $\begin{bmatrix} x_i^{(0)} \\ y_i^{(0)} \end{bmatrix}$ using $\mathbf{p}(t)$ and the current estimate of $R_{i,k}^{(0)}$ based on Eq. 8.
- **Step 1** Repeat
  - **Step 1.a** Use $\begin{bmatrix} x_i^{(n)} \\ y_i^{(n)} \end{bmatrix}$ to estimate $r_{i,j}^{(n+1)}$ and $T_k^{(n+1)}$, $i = 1, 2, 3$, $j = 1, 2$, and $k = x, y, z$, based on Eq. 11.
  - **Step 1.b** Update $R_{i,k}^{(n+1)}$ using Eq. 12.
  - **Step 1.c** Use $r_{i,j}^{(n+1)}$ and $T_k^{(n+1)}$, $i = 1, 2, 3$, $j = 1, 2$, and $k = x, y, z$, and $R_{i,k}^{(n+1)}$ to compute $\mathbf{p}_i^{(n+1)}$ and $\mathbf{p}^{(n+1)}(t)$ based on Eqs. 8 and 11.
  - **Step 1.d** Compute the deviation between $\mathbf{p}(t)$ and $\mathbf{p}^{(n+1)}(t)$ to generate a gradient direction $\delta\mathbf{p}_i^{(n+1)}$ for updating 2D control vertex positions.
  - **Step 1.e** Update 2D control vertex positions $\mathbf{p}_i^{(n+1)}$ by $\mathbf{p}_i^{(n+1)'} = \mathbf{p}_i^{(n+1)} - \alpha\delta\mathbf{p}_i^{(n+1)}$ where α is the update rate.
  - **Step 1.f** If $|r_{i,j}^{(n+1)} - r_{i,j}^{(n)}|$ and $|T_k^{(n+1)} - T_k^{(n)}|$, $i = 1, 2, 3$, $j = 1, 2$, and $k = x, y, z$, are smaller than a preset threshold, then stop. Otherwise $n \longleftarrow n + 1$, go back to **Step 1**.

For **Step 0** above, assume that J points, $\mathbf{p}(t_j) = [x(t_j), y(t_j)]^T, j = 0 \cdots J-1$ are sampled from the image curve. Then it is readily shown that $\mathbf{p}_i$ can be estimated by:

$$\begin{bmatrix} R_{0,k}^{(0)}(t_0) & R_{1,k}^{(0)}(t_0) & \cdots & R_{I-1,k}^{(0)}(t_0) \\ R_{0,k}^{(0)}(t_1) & R_{1,k}^{(0)}(t_1) & \cdots & R_{I-1,k}^{(0)}(t_1) \\ \cdots & \cdots & \cdots & \cdots \\ R_{0,k}^{(0)}(t_{J-1}) & R_{1,k}^{(0)}(t_{J-1}) & \cdots & R_{I-1,k}^{(0)}(t_{J-1}) \end{bmatrix} \begin{bmatrix} x_0^{(0)} & y_0^{(0)} \\ x_1^{(0)} & y_1^{(0)} \\ \cdots & \cdots \\ x_{I-1}^{(0)} & y_{I-1}^{(0)} \end{bmatrix} = \begin{bmatrix} x(t_0) & y(t_0) \\ x(t_1) & y(t_1) \\ \cdots & \cdots \\ x(t_{J-1}) & y(t_{J-1}) \end{bmatrix}, \quad (14)$$

where t_j is approximated by the normalized polygonal arc length in between the sampling points:

$$t_j = \frac{\sum_{k=1}^{j} |p(t_k) - p(t_{k-1})|}{\sum_{k=1}^{J-1} |p(t_k) - p(t_{k-1})|}.$$

For **Step 1.a**, again, one can easily show that

$$\begin{bmatrix} X_0 & Y_0 & 0 & 0 & -X_0x_0^{(n)} & -Y_0x_0^{(n)} & 1 & 0 \\ 0 & 0 & X_0 & Y_0 & -X_0y_0^{(n)} & -Y_0y_0^{(n)} & 0 & 1 \\ X_1 & Y_1 & 0 & 0 & -X_1x_1^{(n)} & -Y_1x_1^{(n)} & 1 & 0 \\ 0 & 0 & X_1 & Y_1 & -X_1y_1^{(n)} & -Y_1y_1^{(n)} & 0 & 1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & & \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & & \\ X_{I-1} & Y_{I-1} & 0 & 0 & -X_{I-1}x_{I-1}^{(n)} & -Y_{I-1}x_{I-1}^{(n)} & 1 & 0 \\ 0 & 0 & X_{I-1} & Y_{I-1} & -X_{I-1}y_{I-1}^{(n)} & -Y_{I-1}y_{I-1}^{(n)} & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} \\ r_{12} \\ r_{21} \\ r_{22} \\ r_{31} \\ r_{32} \\ T_x \\ T_y \end{bmatrix} = \begin{bmatrix} x_0^{(n)} \\ y_0^{(n)} \\ x_1^{(n)} \\ y_1^{(n)} \\ \cdots \\ \cdots \\ x_{I-1}^{(n)} \\ y_{I-1}^{(n)} \end{bmatrix}, \quad (15)$$

by normalizing T_z to 1. The above equations are linear and can be easily solved.

For **Step 1.d** and **Step 1.e** above, define

$$E = \sum_j (\sum_i \mathbf{p}_i R_{i,k}(t_j) - p(t_j))^2 .$$

By holding $R_{i,k}$ constant, we have:

$$\delta\mathbf{p} = \begin{bmatrix} \delta\mathbf{p}_0 \\ \delta\mathbf{p}_1 \\ \cdots \\ \delta\mathbf{p}_{I-1} \end{bmatrix} = \begin{bmatrix} \frac{\partial E}{\partial \mathbf{p}_0} \\ \frac{\partial E}{\partial \mathbf{p}_1} \\ \cdots \\ \frac{\partial E}{\partial \mathbf{p}_{I-1}} \end{bmatrix} = \begin{bmatrix} -\sum_j(\sum_i \mathbf{p}_i R_{i,k}(t_j) - p(t_j))R_{1,k}(t_j) \\ -\sum_j(\sum_i \mathbf{p}_i R_{i,k}(t_j) - p(t_j))R_{2,k}(t_j) \\ \cdots \\ -\sum_j(\sum_i \mathbf{p}_i R_{i,k}(t_j) - p(t_j))R_{I-1,k}(t_j) \end{bmatrix}$$

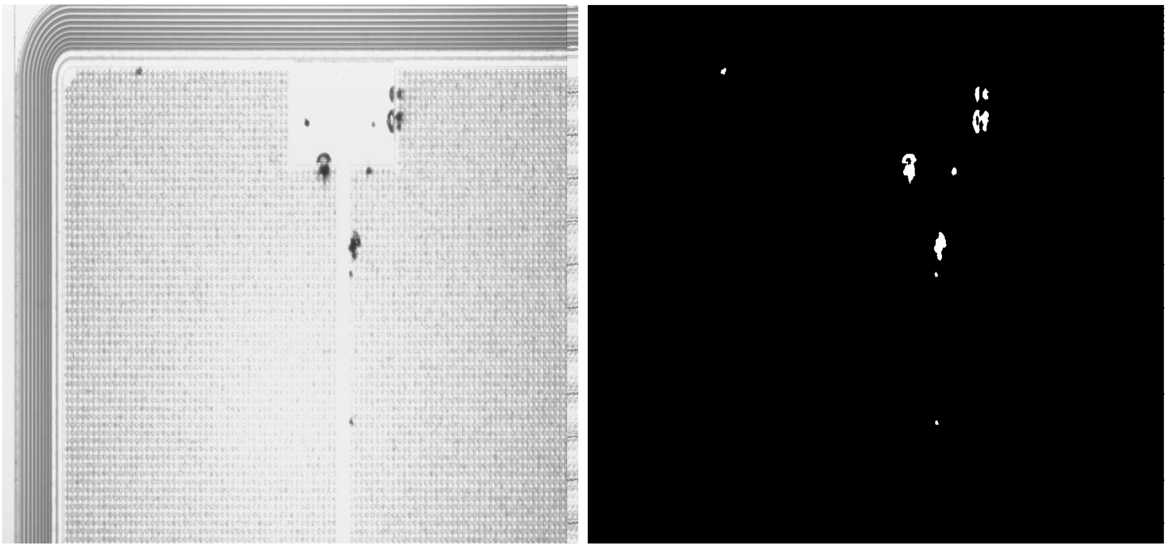
which can be readily computed from the current estimate of $\mathbf{p}_i$ and $R_{i,k}$.

Our experience with using non-rational spline bases to approximate rational spline bases indicates that such an approximation is very good under small perspective distortion and is quite reasonable even under large perspective distortion. Hence, a good initial search point is established and the convergence is achieved very efficiently. Furthermore, by using rational spline functions and partitioning the search in a number of stages as outlined in the above pseudo-code program, we maintain the linearity of the whole formulation. In fact, our experiments (next section) show that convergence is generally achieved in less than three or four iterations which is extremely efficient.

3 Experimental Results

Rigid Motion and Noise Effects To illustrate, Fig. 4 shows a sample wafer image and the associated binary defect map where white pixels mark the defect locations. (In this case, defects are due to arc-burns.) A typical defect region is extracted from Fig. 4 and depicted in solid lines in Fig. 5(a). A transformed shape is generated by rotating the original shape 31^o counterclockwise, scaling it by a factor of 3 in x and a factor of 2 in y, and finally displacing it by $[20, 4]^T$. A random amount of white noise, which can displace a contour point by up to two pixels in any direction, is added to each and every point in the transformed contour. The resulting shape is shown in dashed lines in Fig. 5(a).

We use the second-order b-spline function of a uniform knot vector [33] in the basis expansion step. The spline basis is approximated by a vector of length three: $[0.1, 1.0, 0.1]^T$. The projection coefficients as a function of t, the arc length, for the original and transformed shapes are depicted



(a) (b)

Figure 4: (a) A sample wafer image and (b) the associated binary defect map.

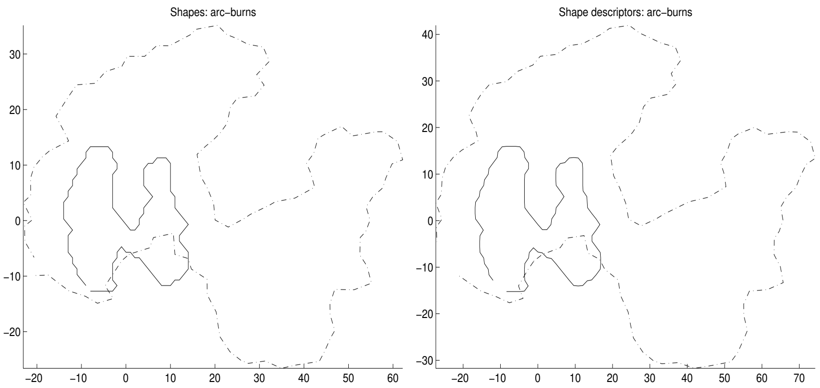
Table 1: Sample invariant features computed based on Eq. 2 for Fig. 5

Index position						Original	Transformed	Percentage
b	d	f	h	j	l	measure	measure	difference
2	6	9	12	15	18	-2.3095	-2.1172	9.08%
73	11	60	66	73	54	-1.6076	-1.6721	4.01%
104	90	94	56	82	108	0.1012	0.1125	10.01%
31	76	58	102	26	45	-0.0840	-0.0802	4.83%
36	7	62	53	22	13	0.7679	0.7657	0.28%

graphically in Fig. 5(b). As can be expected the coefficient plot is very similar to the shape plot because the spline basis performs a weighted average in a local neighborhood.

Tbl. 1 tabulates sample features computed using Eq. 2 for Fig. 5. Since we use only one scale (or a single spline basis of a particular order), the first indices, a, c, e, g, i, k, in Eq. 2 are the same. And we show only the second indices, b, d, f, h, j, l, and the feature values computed for the original and transformed shapes and the percentage difference. As can be observed from the table, the computed results are quite consistent (average about 5%) even in the presence of a significant amount of noise.

General Affine Transform and Occlusion Effects Complicated shapes that undergo general affine transform with possible occlusion can also be handled. Fig. 6 shows three such examples. For star: $\mathbf{m} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$, $\mathbf{t} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$, for heart: $\mathbf{m} = \begin{bmatrix} 0.1 & -0.3 \\ 0.4 & -0.2 \end{bmatrix}$, $\mathbf{t} = \begin{bmatrix} 0.2 \\ 0 \end{bmatrix}$, and for auto: $\mathbf{m} = \begin{bmatrix} cos(15^o) & sin(15^o) \\ -sin(15^o) & cos(15^o) \end{bmatrix}$, $\mathbf{t} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$. Here, we use the Harr wavelet: $[-1, 1]^T$, and, as a result, the coefficient plot shown in the second row in Fig. 6 no longer resembles the shape plot in the first row. However, a similar transform pattern for the shape and coefficient plots should



(a) (b)

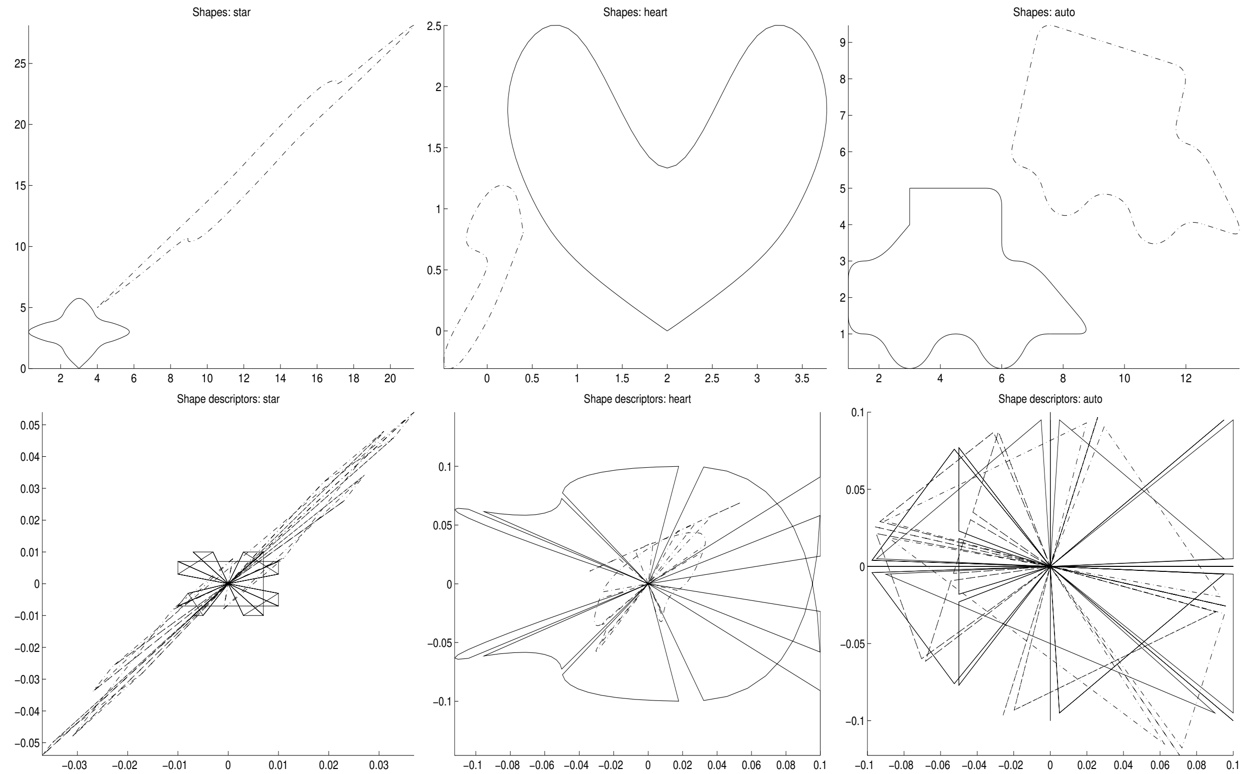
Figure 5: (a) Original and transformed shapes with noise added and (b) original and transformed shape descriptors using spline bases. Solid lines for original shapes and shape descriptors and dashed lines for transformed and noise-corrupted shapes and shape descriptors.

be evident. Sample invariant features for Fig. 6 are summarized in Tbl. 2. In Fig. 6(a) and (b), though the shapes deform significantly, the computed features from the original and transformed shapes are consistent. In Fig. 6(c), as long as no index in the range affected by the change in shape (< 50) is used, consistent results are obtained. These results demonstrate the tolerance to large deformation and occlusion.

General Affine Transform and Change of Parameterization Fig. 7 illustrates the invariants formulation under general affine transform with different parameterization (Eq. 6). Fig. 7(a) shows four distinct shapes, where one is a deformed version of another (the heart and the deformed heart shapes in Fig. 7(a)). Furthermore, the two heart shapes have different parameterization in that the starting point and the direction of traversing the two contours are different. Hence, a direct comparison of invariant features along the arc length is not possible. Fig. 7(b) shows the computed invariants as functions of the Fourier frequency of the heart and deformed heart shapes, based on the Fourier analysis presented in Eq. 6. As can been seen that the results are consistent. Fig. 7(c) shows the invariant descriptors of all four shapes. The similarity between the descriptors for the heart and the deformed heart shape is quite apparent.

Change of Illumination To illustrate invariance under illumination changes, we placed different color filters in front of the light sources used to illuminate the scene. Two examples are shown below: Fig. 8 shows the same mouse pad under white and blue illumination. We randomly placed three circular curves (the red, green, and blue curves in Fig. 8) and computed the invariant signatures along these three curves for both the images under white and blue illumination. Fig. 9(a), (c), and (e) show the r-g-b profiles along the red, green, and blue circles in Fig. 8 under white (solid) and blue (dashed) illumination. Fig. 9(b), (d), and (f) show the invariant profiles computed from the white (solid) and blue (dashed) illumination. As can be seen from the figure, they are quite consistent.

The degree of consistency of the illumination invariant features is best appreciated by comparing the above results with the following: Fig. 10(a)–(c) show the invariant signatures, now computed on *different* circular curves in Fig. 8 (red and green curves for Fig. 10(a), green and blue curves for Fig. 10(b), and blue and red curves for Fig. 10(c). As can be seen that the results are much



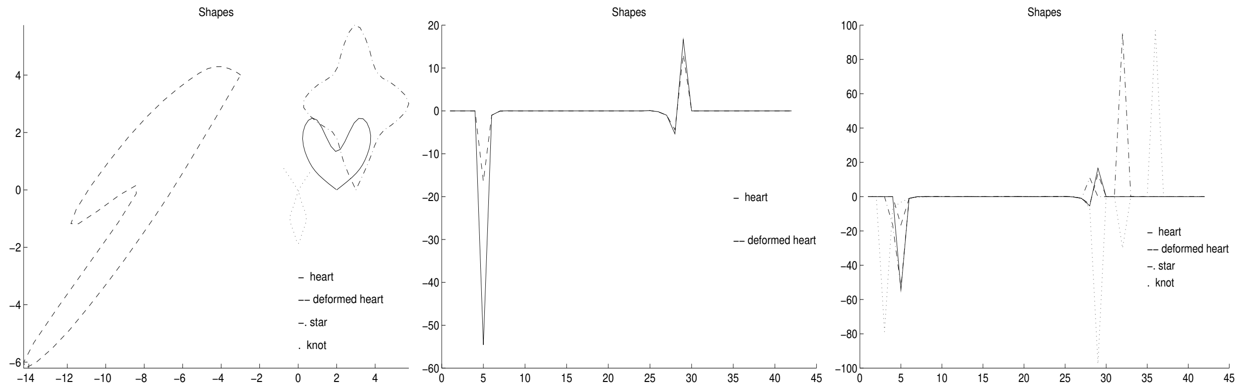
(a) (b) (c)

Figure 6: Original and transformed shapes and shape descriptors using wavelet bases for (a) star, (b) heart, and (c) auto. Solid lines for original shapes and shape descriptors and dashed lines for transformed shapes and shape descriptors.

less consistent. In fact, the RMS deviation of the invariant features computed from different curves is 150% to 260% those computed from the same curves. Finally, Fig. 11 shows a cookbook cover under white and red illumination and Fig. 12 shows the r-g-b color profiles and invariant signatures extracted from the red and green circles in Fig. 11. Again, the two curves of invariant signatures confirm closely.

Hierarchical Invariant Analysis The additional degree of freedom in designing the basis function enables a hierarchical shape analysis. To illustrate, Fig. 13 shows the original (solid lines) and deformed (dashed lines) shapes with different amounts of noise added. The noise level in Fig. 13(b) is 15 times higher than that in Fig. 13(a). As can be seen from Fig. 14 that our approach, which analyzes the shape at different scales locally, will eventually discover the similarity, even though the similarity may manifest at different levels of details. This is the case in Fig. 14, where shape similarity is apparent from scale 3 onward for small noise case and from scale 7 onward for large noise case. Traditional analysis relying on a single scale or requiring high-order derivatives of the contour function will have difficulty handling both cases.

Perspective Transform and View Point Change This is probably the hardest invariants formulation of all as perspective transform is non-linear. Our formulation, even though it is recursive in nature, is nonetheless linear and achieves excellent results in that the number of iterations to



(a) (b) (c)

Figure 7: (a) Four distinct shapes, (b) invariant shape features for the original and deformed heart shapes in the frequency domain, (c) invariant shape features for all four shapes.

(a) (b)

Figure 8: The same mouse pad under (a) white and (b) blue illumination.

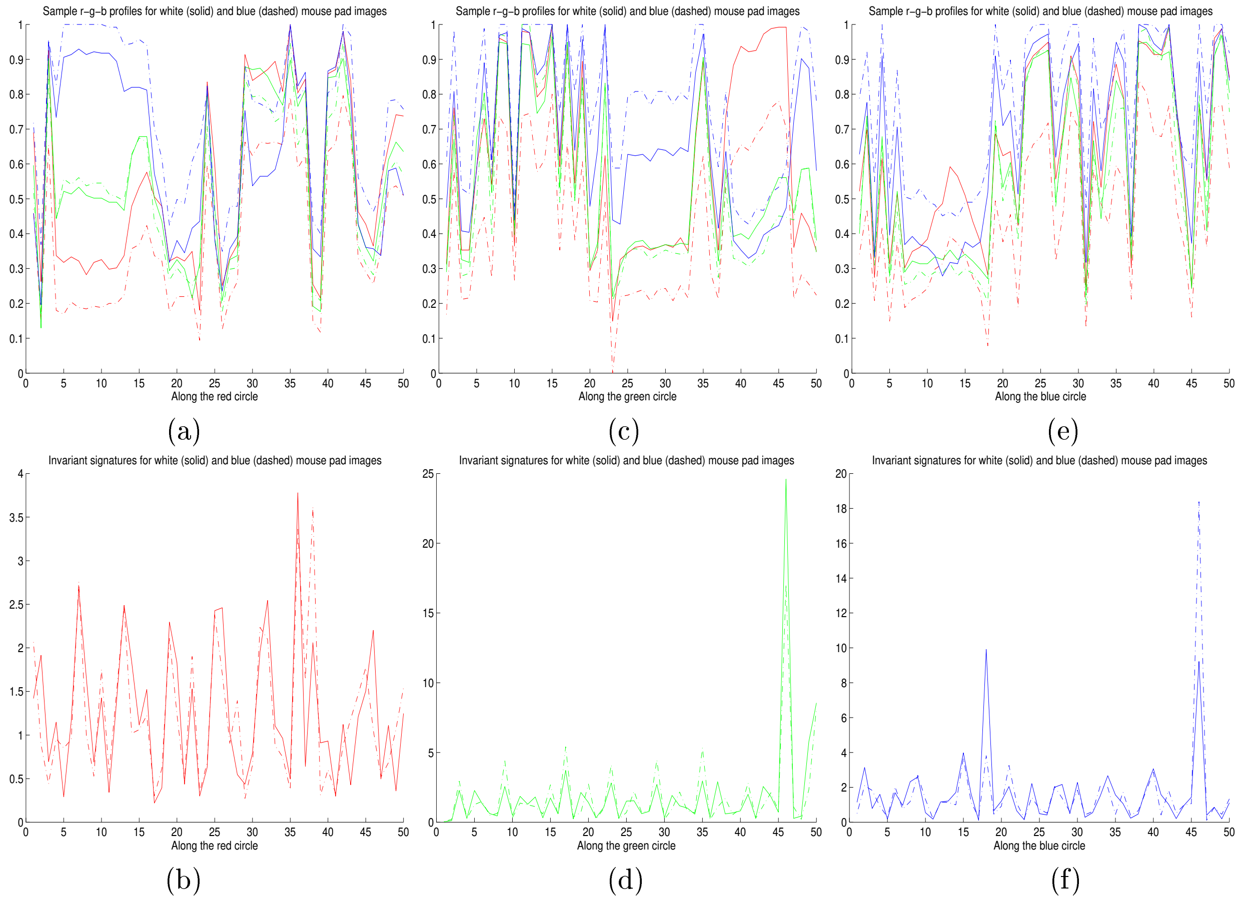


Figure 9: R-g-b profiles and invariant signatures computed for the mouse pad under white (solid) and blue (dashed) illumination. Profiles and invariant signatures were computed on the red (a) and (b), green (c) and (d), and blue (e) and (f) circles in Fig. 8.

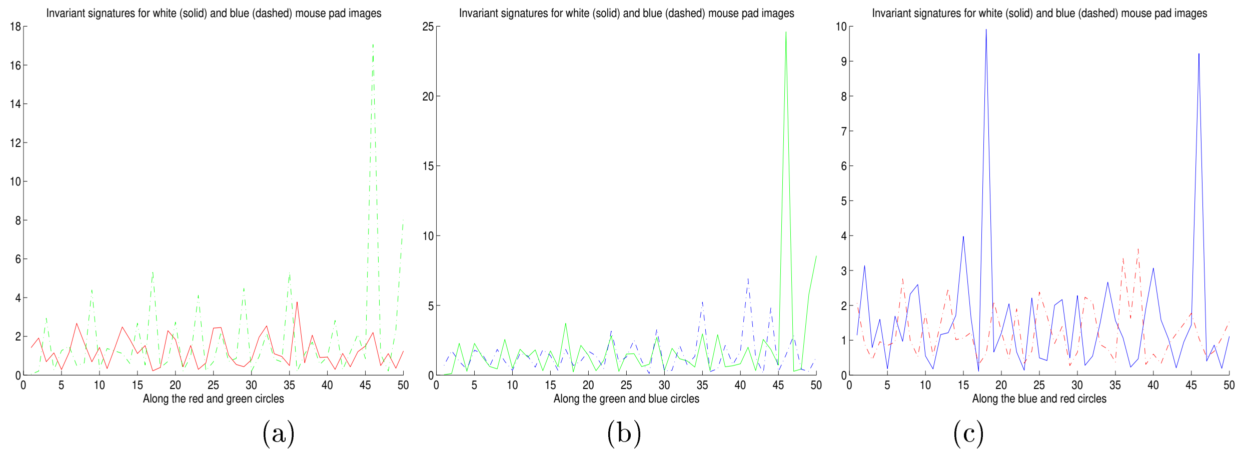
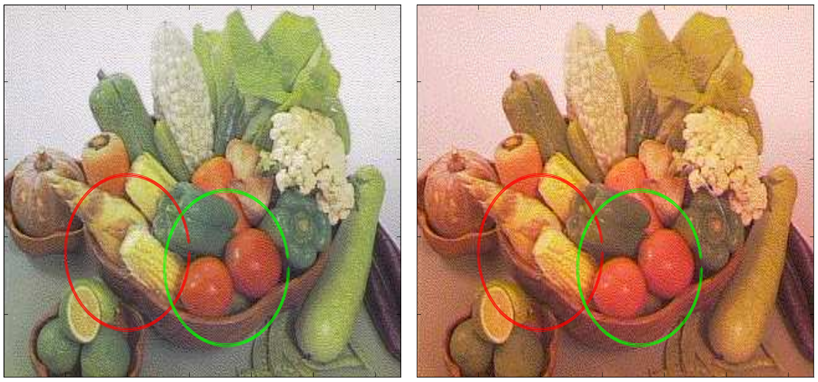
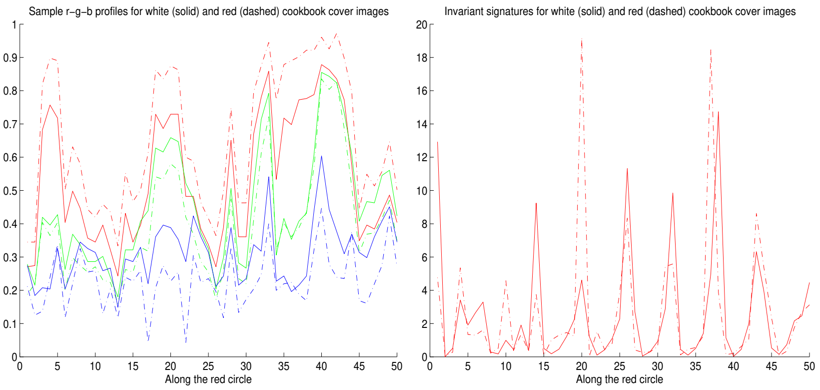


Figure 10: R-g-b profiles and invariant signatures computed for the mouse pad under white (solid) and blue (dashed) illumination. Profiles and invariant signatures were computed on the red and green circles (a), green and blue circles (b), and blue and red circles (c) in Fig. 8.

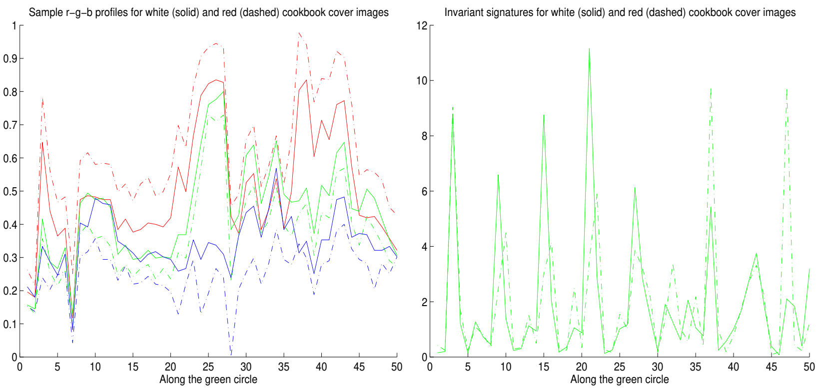


(a) (b)

Figure 11: The same cookbook cover under (a) white and (b) red illumination.



(a) (b)



(c) (d)

Figure 12: R-g-b profiles and invariant signatures computed for the cookbook cover under white (solid) and red (dashed) illumination. Profiles and invariant signatures were both computed on the red circle in Fig. 11 for (a) and (b), and on the green circle in Fig. 11 for (c) and (d)
.

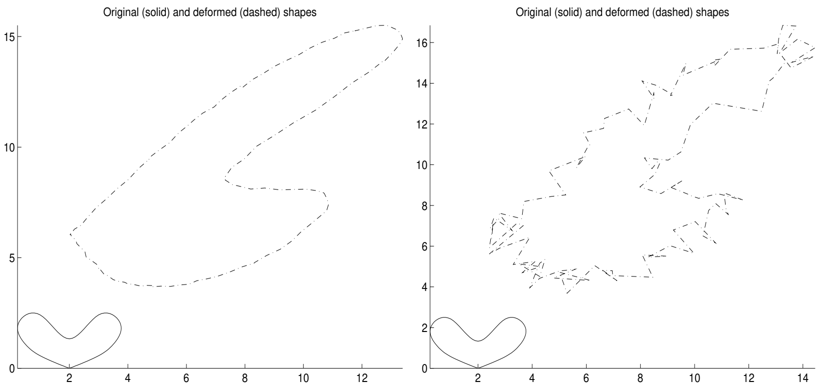
Table 2: Sample invariant features computed based on Eq. 2 for Fig. 6. The two entries in Auto showing a large deviation are due to occlusion. Features computed from non-occluded regions (the remaining three entries in Auto) are consistent. See text for details.

Index position						Original	Transformed	Percentage
b	d	f	h	j	l	measure	measure	difference
Star								
127	162	121	51	74	140	7.3931	7.3931	0
99	97	152	161	142	166	-12.7614	-12.7614	0
167	159	111	58	41	145	-0.1154	-0.1154	0
22	117	10	28	11	98	-2.1121	-2.1121	0
46	70	119	144	58	136	0.7984	0.7984	0
Heart								
96	35	3	1	84	4	-1.2043	-1.2043	0
26	97	76	92	87	1	17.9388	17.9388	0
66	68	37	30	29	106	4.3232	4.3232	0
106	43	2	30	18	72	1.0804	1.0804	0
109	68	106	70	55	79	-0.1856	-0.1856	0
Auto								
292	255	68	101	172	286	-2.0959	-2.0959	0
178	136	57	236	289	265	8.6400	8.6400	0
185	207	230	85	112	284	1.1077	1.1077	0
80	48	84	275	108	172	2.0356	0.5333	73.8%
50	108	105	235	152	74	-0.5534	-0.2913	90.0%

verify the invariance is very small (around three or four) even for large perspective distortion.

Figs. 15 and 16 show two such results. Both star and knot have a "canonical" view which is the head-on view. A 2D perspective image was generated by applying an arbitrary rotation and translation to the shape before perspective projection. For star, the rotation angle was -45^o around an axis $[3,2,1]^T$ and the translation vector was $[6,5,2]^T$. For knot, the rotation angle was -55^o around an axis $[3,0,1]^T$ and the translation vector was $[2,0,2]^T$. In Figs. 15 and 16, the first row shows the projected 2D shapes and the depth profiles of the control vertices from the particular view point in solid line. The search for invariance started by assuming there was no rotation nor translation and the depth was a constant 1. Under this assumption, the predicted 2D shapes (the head-on view) and the depth profiles (a constant 1) are shown in dashed line in the first row in Figs. 15 and 16. As can be seen by comparing the two curves in the first row of Figs. 15 and 16, perspective distortion was quite severe in both cases.

We then used the non-rational spline bases to approximate the rational spline bases and estimated the 2D control vertex positions. A good initial search point was established for the projected control vertex positions this way. An iteration of estimating the motion parameters, updating the control vertex positions, and revising the shapes of the rational spline bases (using Eqs. 11 and 12) ensued. The predicted 2D shapes and depth profiles, using now updated control vertices and rational bases, are displayed in the second row of Figs. 15 and 16. The discrepancy in the predicted and the actual shapes was then used for correcting the 2D control vertex positions, and another iteration of updating followed. Convergence in both cases was fast and most significant changes occurred in the first three iterations. The depth profiles converged only qualitatively as a scaling factor was not recovered.



(a) (b)

Figure 13: Original and transformed shapes with noise added. The noise level in (b) is 15 times that in (a). Solid lines for original shapes and shape descriptors and dashed lines for transformed and noise-corrupted shapes and shape descriptors.

4 Conclusion

In this paper we present a new framework for computing image invariants. The framework utilizes many desirable properties of wavelet and basis expansion techniques and is neither strictly global nor purely local. Furthermore, it is quite simple and straightforward to implement. Preliminary results on both real and synthetic images are very promising. These results demonstrate the tolerance to noise, occlusion, and perspective distortion, and the ability for multi-scale analysis.

Acknowledgment: We thank Dr. N. Nandhakumar for his careful proofreading of and constructive comments on this manuscript. We thank Mr. Ronald Alfreze for helping with the experiments. This research was supported in part by NSF/NASA/ARPA through the Alexandria Digital Library Project and in part by a grant from Electrglas Inc.

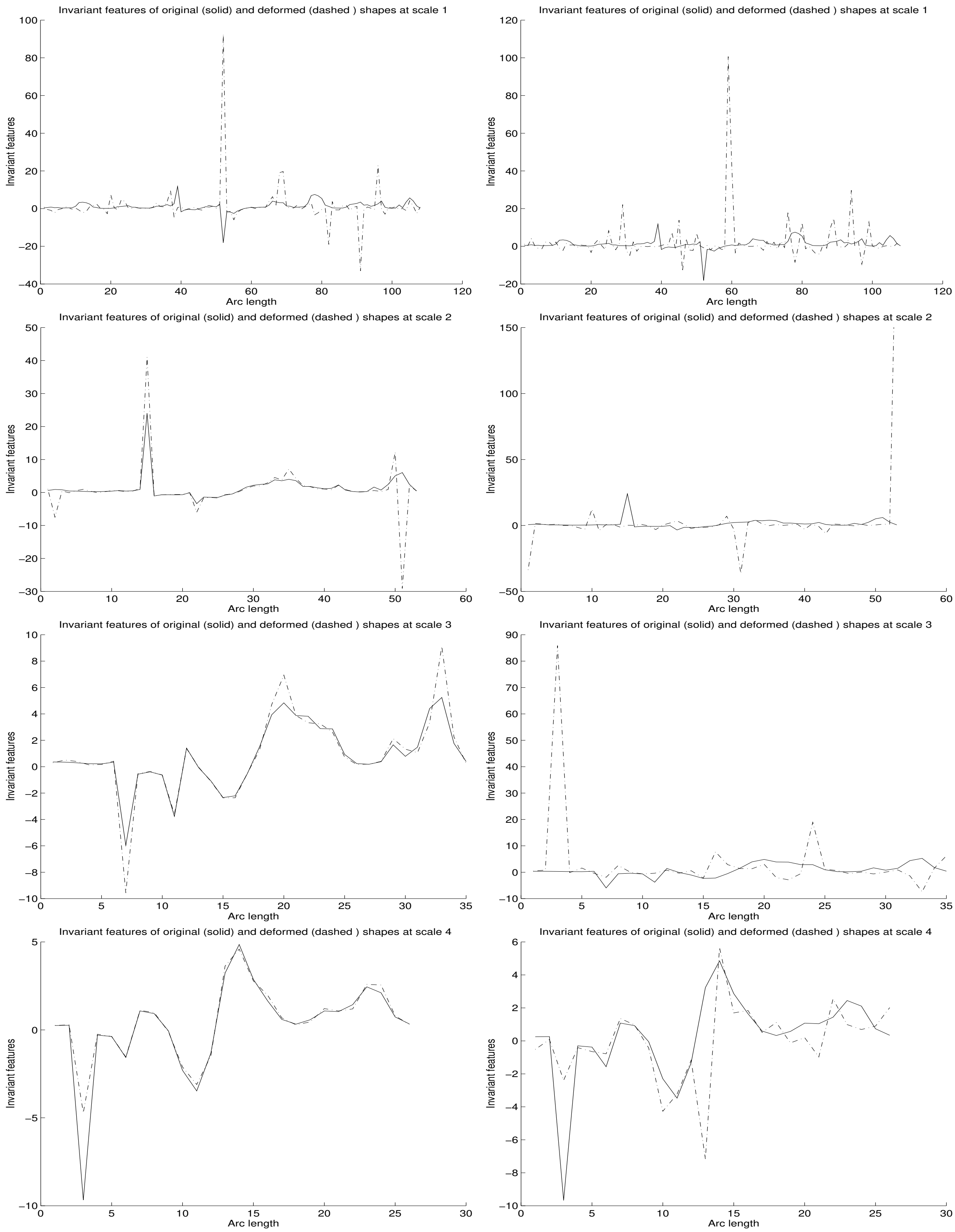


Figure 14: Invariant shape descriptors for the original and deformed shapes with noise added at different scales. Solid lines for original shapes and shape descriptors and dashed lines for transformed and noise-corrupted shapes and shape descriptors (continuing on the next page).

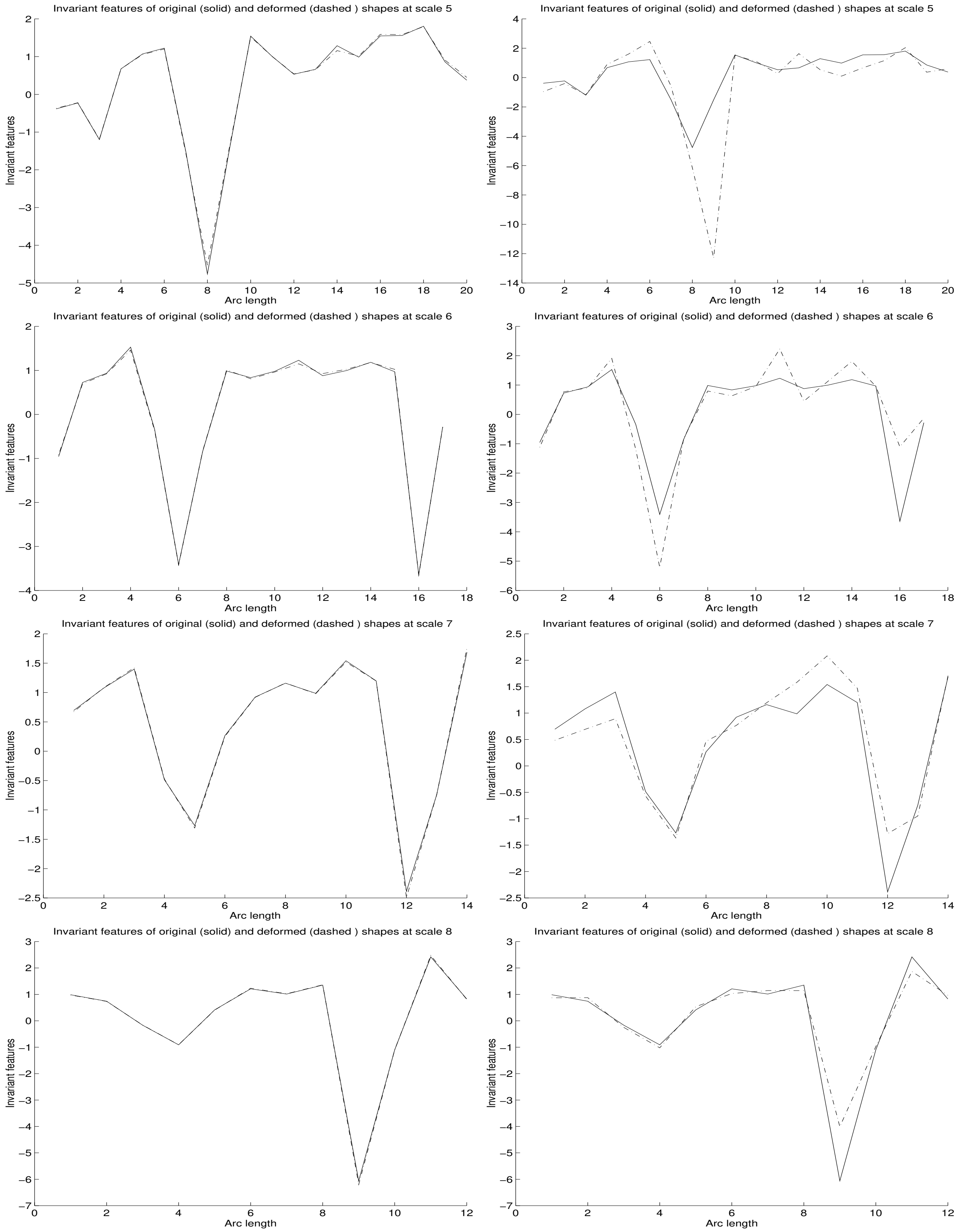


Figure: Invariant shape descriptors for the original and deformed shapes with noise added at different scales. Solid lines for original shapes and shape descriptors and dashed lines for transformed and noise-corrupted shapes and shape descriptors.

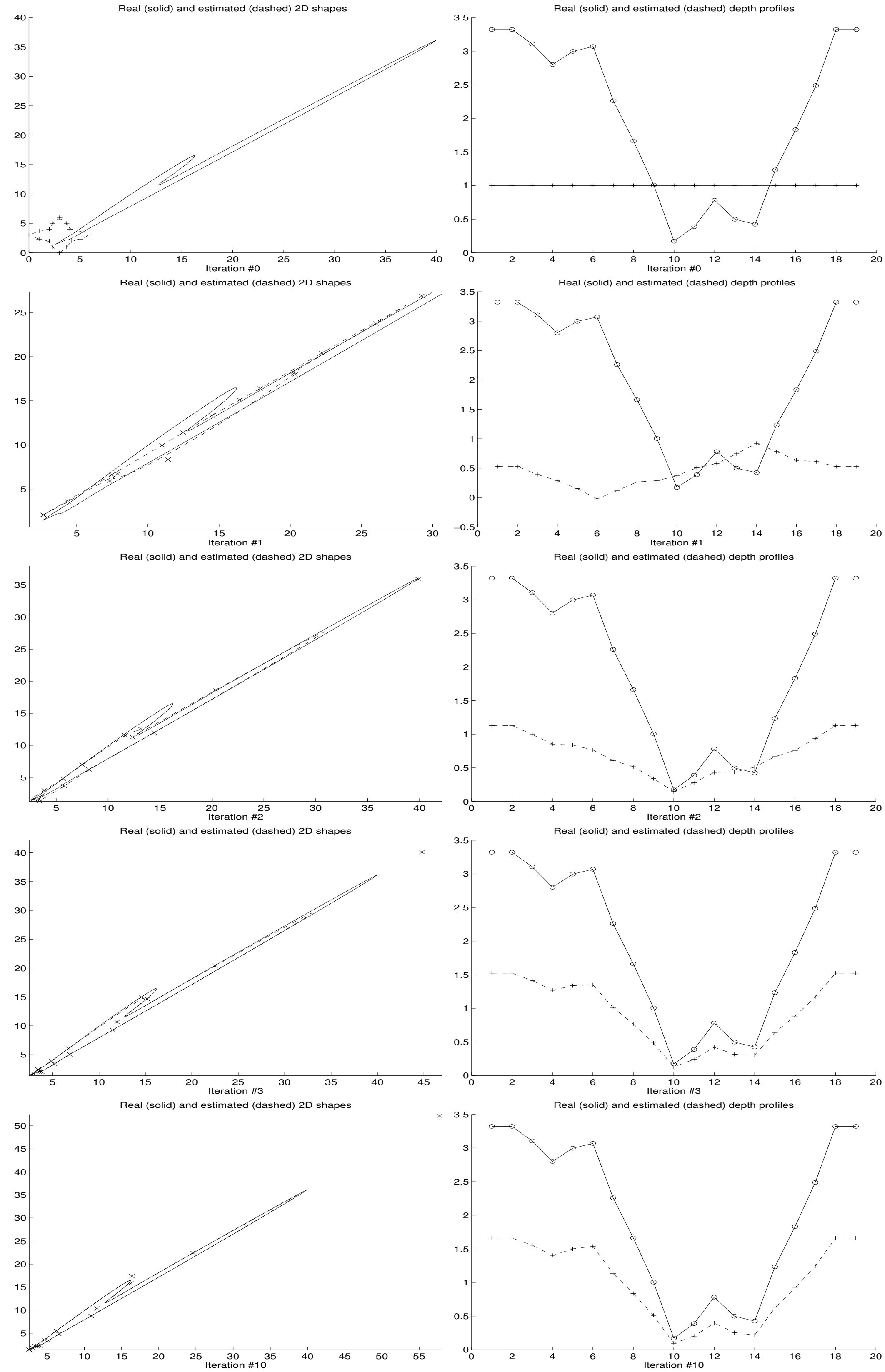


Figure 15: Real (solid) and estimated (dashed) shape and depth profiles for star after 0th, 1st, 2nd, 3rd, and 10th iterations. "o" and "+" denote the projected control vertex positions.

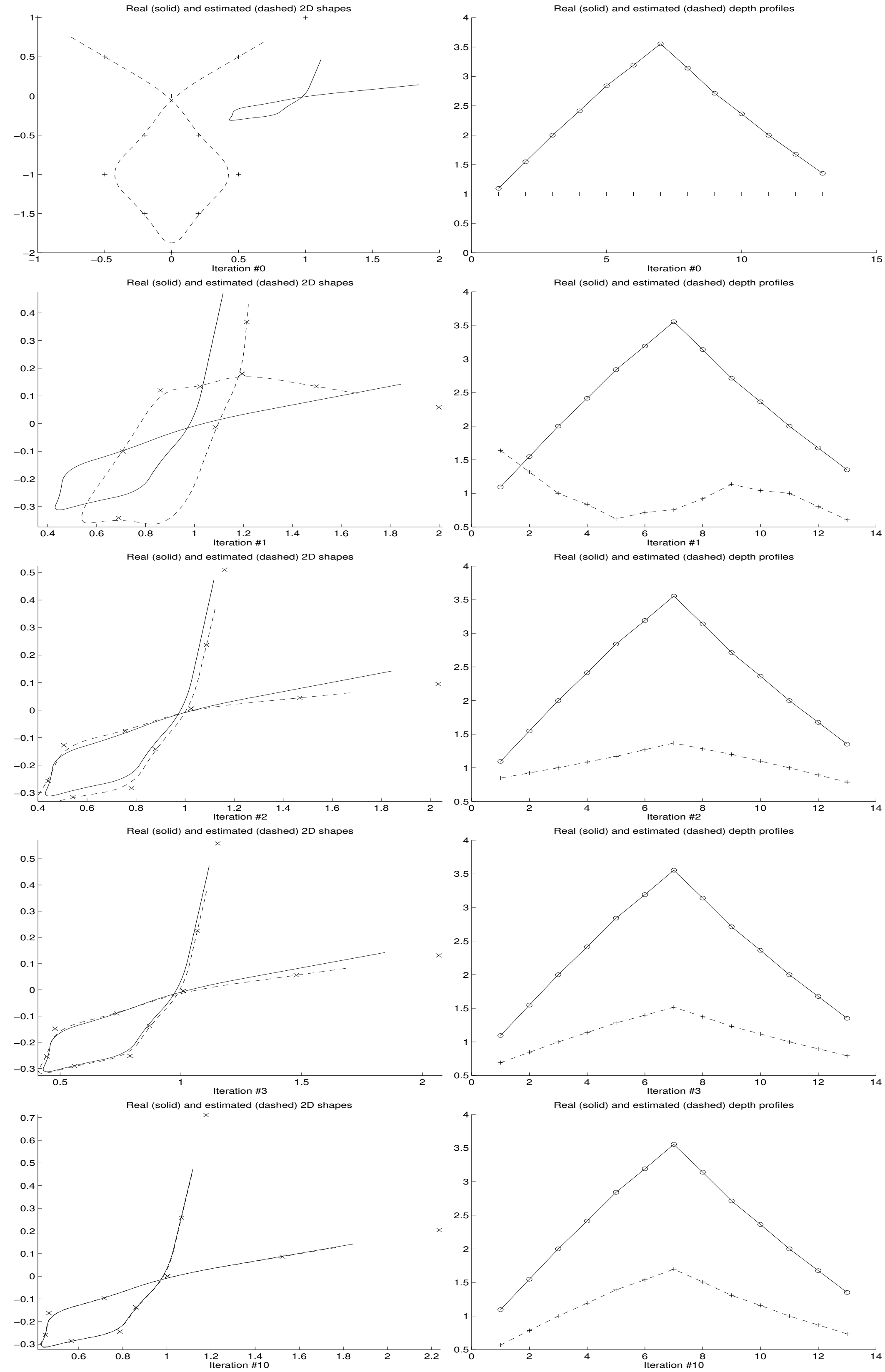


Figure 16: Real (solid) and estimated (dashed) shape and depth profiles for knot after 0th, 1st, 2nd, 3rd, and 10th iterations. "o" and "+" denote the projected control vertex positions.

References

[1] K. Arbter, W. E. Snyder, H. Burkhardt, and G. Hirzinger. Application of Affine-Invariant Fourier Descriptors to Recognition of 3-D Objects. *IEEE Trans. Pattern Analy. Machine Intell.*, 12:640–647, 1990.

[2] W. Bohm, G. Farin, and J. Kahmann. A Survey of Curve and Surface Methods in CAGD. *Comput. Aided Geometric Des.*, pages 1–60, 1984.

[3] A. M. Bruckstein, R. J. Holt, A. N. Netravali, and T. J. Richardson. Invariant Signatures for Planar Shape Recognition Under Partial Occlusion. In *Proc. Int. Conf. Pattern Recognit.*, pages 108–112, The Hague, Netherlands, 1992.

[4] J. M. Combes, A. Grossman, and Ph. Tchamitchian (Eds.). *Wavelets: Time-Freqency Methods and Phase Space, 2nd ed.* Springer-Verlag, Berlin, 1990.

[5] I. Daubechies. Orthonormal Bases of Compactly Supported Wavelets. *Commun. Pure Appl. Math.*, 41:909–960, 1988.

[6] C. deBoor. *A Practical Guide to Spline.* Springer-Verlag, New York/Berlin, 1978.

[7] S. J. Dickinson, A. Pentland, and A. Rosenfeld. 3-D Shape Recovery Using Distributed Aspect Matching. *IEEE Trans. Pattern Analy. Machine Intell.*, 14(2):174–198, 1992.

[8] L. E. Dickson. *Algebraic Invariants.* John-Wiley & Sons, 1914.

[9] D. Eggert and K. Bowyer. Computing the Perspective Projection Aspect Graph of Solids of Revolution. *IEEE Trans. Pattern Analy. Machine Intell.*, 15(2):109–128, 1993.

[10] D. Eggert, K. Bowyer, C. R. Dyer, and H. I. Christensen. The Scale Space Aspect Graph. *IEEE Trans. Pattern Analy. Machine Intell.*, 15(11):1114–1130, 1993.

[11] G. Farin. Algorithms for Rational Bézier Curves. *Comput. Aided Des.*, 15(2):73–77, Mar. 1983.

[12] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice, 2nd ed.* Addison-Wesley, Reading, MA, 1990.

[13] D. Forsyth. A Novel Algorithm for Color Constancy. *Int. J. Comput. Vision*, 5:5–36, 1990.

[14] D. Forsyth, J. L. Mundy, A. Ziserman, A. Heller C. Coelho, and C. Rothwell. Invariant Descriptors for 3-D Object Recognition and Pose. *IEEE Trans. Pattern Analy. Machine Intell.*, 13(10):971–991, Oct. 1991.

[15] D. Gabor. Theory of Communication. *J. Inst. Elec. Eng.*, 93:429–457, 1946.

[16] R. Haralick and L. Shapiro. *Computer and Robot Vision, Vol. I.* Addison-Wesley, Reading, MA, 1992.

[17] G. Healey and A. Jain. Using Physics-Based Invariant Representations for the Recognition of Regions in Multispectral Images. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognit.*, pages 750–755, San Francisco, CA, Jun. 96.

[18] G. Healey and D. Slater. Global Color Constancy: Recognition of Objects by Use of Illumniation-Invariant Properties of Color Distributions. *Opt. Soc. Am. A*, 11(11):3003–3010, Nov. 1994.

[19] G. Healey and L. Wang. Illumination-Invariant Recognition of Texture in Color Images. *Opt. Soc. Am. A*, 12(9):1877–1883, Sep. 1995.

[20] M. K. Hu. Pattern Recognition by Moment Invariants. *Proc. IRE*, 49:1428, 1961.

[21] IGES. *Initial Graphics Exchange Specifications, Ver. 3.0.* Nat. Bur. of Stds., Gaithersburg, MD, 1986.

[22] R. Kondepudy and G. Healey. Use of Invariants for Recognition of Three-Dimensional Color Textures. *Opt. Soc. Am. A*, 11(11):3037–3049, Nov. 1994.

[23] E. P. Lane. *Projective Differential Geometry of Curves and Surfaces.* Univ. of Chicago Press, Chicago, IL, 1932.

[24] S. G. Mallat. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. *IEEE Trans. Pattern Analy. Machine Intell.*, 11(7):674–693, 1989.

[25] S. G. Mallat. Multifrequency Channel Decompositions of Images and Wavelet Models. *IEEE Trans. Acoust. Speech Signal Processing*, 37:2091–2110, 1989.

[26] L. Maloney. Evaluation of Linear Models of Surface Spectral Refelctance with Small Number of Parameters. *Opt. Soc. Am. A*, 3:1673–1683, 1986.

[27] L. Maloney and B. Wandell. Color Constancy: A Method for Recovering Surface Spectral Reflectance. *Opt. Soc. Am. A*, 3:29–33, 1986.

[28] J. Michel, N. Nandhakumar, and V. Velten. Thermophysical Algebraic Invariants from Infrared Imagery for Object Recognition. *IEEE Trans. Pattern Analy. Machine Intell.*, 19:41–51, 1987.

[29] L. Peigl and W. Tiller. Curve and Surface Contructions using Rational B-Splines. *Comput. Aided Des.*, 19(9):485–498, 1987.

[30] L. R. Rabiner and D. W. Schafer. *Digital Processing of Speech Signals.* Prentice-Hall, Englewood Cliffs, NJ, 1978.

[31] T. H. Reiss. *Recognizing Planar Objects Using Invariant Image Features.* Springer-Verlag, Berlin, 1993.

[32] E. Rivlin and I. Weiss. Local Invariants for Recognition. *IEEE Trans. Pattern Analy. Machine Intell.*, 17(3):226–238, Mar. 1995.

[33] D. F. Rogers and J. A. Adams. *Mathematical Elements for Computer Graphics, 2nd Ed.* McGraw-Hill, New York, NY, 1990.

[34] F. A. Sadjadi and E. L. Hall. Three-Dimensional Moment Invariants. *IEEE Trans. Pattern Analy. Machine Intell.*, 2:127–136, Mar. 1980.

[35] R. Sedgewick. *Algorithms, 2nd ed.* Addison-Wesley, Reading, MA, 1989.

[36] D. Slater and G. Healey. The Illumination-Invariant Recognition of 3D Objects Using Local Color Invariants. *IEEE Trans. Pattern Analy. Machine Intell.*, 18(2):206–210, Feb. 1996.

[37] D. Slater and G. Healey. Using a Spectral Reflectance Model for the Illumination-Invariant Recognition of Local Image Structure. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognit.*, pages 770–775, San Francisco, CA, Jun. 1996.

[38] C. H. Teh and R. Chin. On Image Analysis by the Methods of Moments. *IEEE Trans. Pattern Analy. Machine Intell.*, 10(4):496–513, Jul. 1988.

[39] W. Tiller. Rational B-Splines for Curve and Surface Representation. *IEEE Comput. Graphics & App.*, 3(6):61–69, 1987.

[40] L. Wang and G. Healey. Illumination and Geometry Invariant Recognition of Texture in Color Images. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognit.*, pages 419–424, San Francisco, CA, Jun. 1996.

[41] D. Weinshall. Model-Based Invariants for 3-D Vision. *Int. J. Comput. Vision*, 10(1):27–42, 1993.

[42] I. Weiss. Geometric Invariants and Object Recognition. *Int. J. Comput. Vision*, 10(3):207–231, 1993.

[43] I. Weiss. Noise-Resistant Invariants of Curves. *IEEE Trans. Pattern Analy. Machine Intell.*, 15(9):943–948, Sep. 1993.