Robot Sensing Techniques Based on High-Dimensional Moment Invariants and Tensors

Vishal Markandey, *Member, IEEE*, and Rui J. P. deFigueiredo, *Fellow, IEEE*

Abstract— **A generalization of the concepts of moments and moment invariants to n-dimensions is presented. These concepts are used to develop techniques for object identification and attitude determination. Experimental results for these techniques are presented, and theoretical error analyses are developed.**

I. Introduction

THE concepts of moments and moment invariants have been extensively applied to the problems of object recognition and attitude determination [1]–[4], [6], [7]. Hu [1] uses 2-D invariants for estimating orthogonal transformations in the image plane, and Dudani *et al.* [2] apply 2-D invariants to object silhouettes. Cyganski and Orr [3] have used a tensor representation to derive the invariant formulation for 2-D image data of 3-D objects. Sadjadi and Hall [4] have developed an object recognition technique using 3-D moment invariants, while Lo and Don [11] have used 3-D moments for object identification and positioning. Faber and Stokley [12] compute the affine transformation between two 3-D medical images using a tensor formulation similar to that of Cyganski and Orr [3]. Bamieh and deFigueiredo [6] use invariants of 2-D moments with respect to 3-D scaling, translation, and rotation, all of which correspond to invariants with respect to 2-D affine transformations. We extend this formulation to the general n-dimensional case and discuss its applicability in multisensor fusion.

In this paper, we first present a general theoretical framework for the n-dimensional case, which provides a basis for unifying the earlier work in the field mentioned above, for the special cases of $n = 2$ and 3. An incentive for developing such a unified representation is the increasing use of multiple sensors and sensor fusion in robot vision, for example 2-D and 3-D data of the same scene could be respectively obtained from camera and laser range devices. One of the problems in sensor fusion today is the integration of such data from multiple sensors into a uniform representation, ideally without loss of information. Our technique uses high-dimensional moment invariants for the representation of features using data obtained from multiple-sensor or high-dimensional measurement spaces. The term "high dimensions," as used in this paper, refers to dimensions greater than three. Examples of high-dimensional data are multispectral satellite data for agricultural, surveillance, and other applications, and the combinations of visible, several infrared, and laser radar measurements used in military applications.

The rest of this paper is organized as follows: A formulation for n-dimensional moments and moment invariants is given in Section II. This includes a technique for invariant construction from raw data. Section III is a discussion of object representation and identification using moment invariants. It is shown in this section that while 2-D moment invariant representations are limited to polyhedral objects, 3-D representations can be extented to objects with curved surfaces, under the condition that the object surface can be subdivided into well defined patches. An attributed graph representation similar to that in [6] and [9] is used. Each node of the graph represents a face or patch of the object and has its moment invariants associated with it. Arcs between nodes represent connectivity between the corresponding faces or patches. The graph representation derived from the image is a subgraph of the corresponding object's graph representation. For object identification, graphs obtained from images are matched against object graphs in the system library, using a recursive graph matching algorithm. Experimental results for object recognition are presented. Section IV is a discussion of attitude determination using moment tensors. It turns out that if a face or patch of an object has any axis of reflection symmetry or will have it under any affine transformation, then its moment tensors will reduce to zero. As many geometric objects do have symmetric surfaces, this can be a serious constraint. A technique is developed for automatically checking the symmetry conditions of a polygon and if symmetry conditions exist, artificially deforming it so that the symmetry condition is destroyed. This deformed polygon is then used for attitude determination. Experimental results for attitude determination are presented. Section V is an error analysis that explores the sensitivity of moment invariants and attitude parameters to errors in input.

II. n-Dimensional Moment Invariants

n-dimensional moments of order $(p_1 + p_2 + \ldots + p_n)$ for a function $f(x_1, x_2, \ldots, x_n)$ are defined in terms of the Riemann integral as

$$\mu_{p_1 p_2 \ldots p_n} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} x_1^{p_1} x_2^{p_2} \cdots x_n^{p_n} \cdot f(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \cdots dx_n \quad (1)$$

Manuscript received July 27, 1988; revised August 14, 1991.

V. Markandey is with the Computer Science Laboratory, Central Research Laboratories, Texas Instruments, Dallas, TX 75265.

R. J. P. deFigueiredo is with the Department of Electrical and Computer Engineering and the Department of Mathematics, University of California, Irvine, CA 92717.

IEEE Log Number 9105404.

where $p_1, p_2, \ldots, p_n = 0, 1, 2, \ldots$. In this paper, we assume that f is sufficiently well behaved for the above moments to exist.

If $f(x_1, x_2, \ldots, x_n)$ is assumed to be a piecewise continuous (and therefore bounded) function and is nonzero only in a finite part of R^n, then moments of all order exist, and it can be proved that the central moments $\{m_{p_1 p_2 \ldots p_n}\}$ (see [4] and [5] for a definition) uniquely determine $f(x_1, x_2, \ldots, x_n)$ and are uniquely determined by it.

A. Moment Generating and Characteristic Functions

The moment generating and characteristic functions for the function $f(x_1, x_2, \ldots, x_n)$ are respectively defined as

$$M(u_1, u_2, \ldots, u_n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} e^{(u_1 x_1 + u_2 x_2 + \ldots + u_n x_n)} \cdot f(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \ldots dx_n \tag{2}$$

$$\phi(u_1, u_2, \ldots, u_n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} e^{(iu_1 x_1 + iu_2 x_2 + \ldots + iu_n x_n)} \cdot f(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \ldots dx_n \tag{3}$$

where $i = \sqrt{-1}$.

B. Central Moments

The central moments are defined as

$$m_{p_1 p_2 \ldots p_n} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} (x_1 - \bar{x}_1)^{p_1} (x_2 - \bar{x}_2)^{p_2} \ldots (x_n - \bar{x}_n)^{p_n} f(x_1, x_2, \ldots, x_n)\, dx_1 dx_2 \ldots dx_n \tag{4}$$

where

$$\bar{x}_1 = \frac{\mu_{10\ldots0}}{\mu_{00\ldots0}}, \quad \bar{x}_2 = \frac{\mu_{01\ldots0}}{\mu_{00\ldots0}}, \ldots, \quad \bar{x}_n = \frac{\mu_{00\ldots1}}{\mu_{00\ldots0}}. \tag{5}$$

From here on, all moments refered to are central moments.

C. Algebraic Forms and Invariants

A homogenous polynomial in n variables of order q is called an n-form of order q if and only if

$$\rho(x_1, x_2, \ldots, x_n) = \sum_{p_1, p_2, \ldots, p_n = 0}^{q} \frac{q!}{p_1!\, p_2! \ldots p_n!}\, a_{p_1 p_2 \ldots p_n} \cdot x_1^{p_1} x_2^{p_2} \ldots x_n^{p_n} \tag{6}$$

where $p_1 + p_2 + \ldots + p_n = q$ and $p_1, p_2, \ldots, p_n$ are all integers. A homogenous polynomial $I(a)$ of coefficients $a_{p_1 p_2 \ldots p_n}$ is an algebraic invariant of weight w if

$$I(a'_{p_1 p_2 \ldots p_n}) = \Delta^w I(a_{p_1 p_2 \ldots p_n}) \tag{7}$$

where $a'_{p_1 p_2 \ldots p_n}$ are the new coefficients obtained under a linear transformation of the original polynomial and Δ is the determinant of the transformation matrix. When $w = 0$, the invariants are called absolute invariants.

D. Moment Invariants

Expanding the exponential factor in the moment-generating function (2) into a series form, we have

$$M(u_1, u_2, \ldots, u_n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \sum_{q=0}^{\infty} \frac{1}{q!} (u_1 x_1 + u_2 x_2 + \ldots + u_n x_n)^q f(x_1, x_2, \ldots, x_n)\, dx_1\, dx_2 \ldots dx_n. \tag{8}$$

Interchanging the orders of summation and integration

$$M(u_1, u_2, \ldots, u_n) = \sum_{q=0}^{\infty} \frac{1}{q!} \sum_{p_1, p_2, \ldots, p_n = 0}^{q} \cdot m_{p_1 p_2 \ldots p_n} \frac{q!}{p_1! p_2! \ldots p_n!} u_1^{p_1} u_2^{p_2} \ldots u_n^{p_n}. \tag{9}$$

Let $(u_1, u_2, \ldots, u_n)$ and $(x_1, x_2, \ldots, x_n)$ undergo the following transformations:

$$\begin{bmatrix} u_1 \\ u_2 \\ \cdot \\ \cdot \\ u_n \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} & \cdot & \cdot & t_{1n} \\ t_{21} & & & & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ t_{n1} & \cdot & \cdot & \cdot & t_{nn} \end{bmatrix} \begin{bmatrix} u'_1 \\ u'_2 \\ \cdot \\ \cdot \\ u'_n \end{bmatrix} \tag{10}$$

and

$$\begin{bmatrix} x'_1 \\ x'_2 \\ \cdot \\ \cdot \\ x'_n \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} & \cdot & \cdot & t_{1n} \\ t_{21} & & & & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ t_{n1} & \cdot & \cdot & \cdot & t_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ x_n \end{bmatrix}. \tag{11}$$

Transformation (10) is refered to as a contragredient transform and (11) as a cogredient transform. The variables of these two transforms satisfy the condition

$$u_1 x_1 + u_2 x_2 + \ldots + u_n x_n = u'_1 x'_1 + u'_2 x'_2 + \ldots + u'_n x'_n. \tag{12}$$

By applying transforms (10) and (11) to (8) we obtain

$$M_1(u'_1, u'_2, \ldots, u'_n) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \sum_{q=0}^{\infty} \frac{1}{q!} (u'_1 x'_1 + u'_2 x'_2 + \ldots + u'_n x'_n)^q \cdot f'(x'_1, x'_2, \ldots, x'_n) \cdot \frac{1}{|J|} dx'_1 dx'_2 \ldots dx'_n \tag{13}$$

where $f'(x'_1, x'_2, \ldots, x'_n) = f(x_1, x_2, \ldots, x_n)$ and $|J|$ is the Jacobian of the transform (10). $M_1(u'_1, u'_2, \ldots, u'_n)$ is the

moment generating function after the transformation. If the moments after transformation are defined as

$$m'_{p_1p_2\cdots p_n} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}(x'_1)^{p_1}\,(x'_2)^{p_2}\cdots(x'_n)^{p_n} \cdot f'(x'_1,x'_2,\ldots,x'_n)\,dx'_1\,dx'_2\,\ldots\,dx'_n \tag{14}$$

then

$$M(u'_1,u'_2,\ldots,u'_n) = \frac{1}{|J|}\sum_{q=0}^{\infty}\frac{1}{q\,!}\sum_{p_1,p_2,\ldots,p_n\,=\,0}^{q}\frac{q\,!}{p_1!p_2!\ldots p_n!}\cdot m'_{p_1p_2\cdots p_n}\,u'^{p_1}_1\,u'^{p_2}_2\,\ldots\,u'^{p_n}_n. \tag{15}$$

It is a fundamental result in invariant theory [5] that the transformation for the "a" coefficients in (6) is the same as that for the monomial $x_1^{p_1}x_2^{p_2}\ldots x_n^{p_n}$ in the expansion of the expression

$$(u_1x_1 + u_2x_2 + \ldots + u_nx_n)^m. \tag{16}$$

From (8), (9), (13), and (15) it is obvious that the same relationship holds between the kth order moments and the monomials, except for division by the $|J|$ term. From this we have the following theorem:

Theorem 1—Fundamental Theorem of Moment Invariants: If an algebraic form of order q has algebraic invariants

$$I(a'_{p_1p_2\cdots p_n}) = \Delta^w\ I(a_{p_1p_2\cdots p_n}) \tag{17}$$

then the moments of order q have invariants

$$I(m'_{p_1p_2\cdots p_n}) = |J|\ \Delta^w\ I(m_{p_1p_2\cdots p_n}). \tag{18}$$

E. Moment Invariants and Linear Transformations

Theorem 2: When the underlying space is linearly transformed, moments of a given order get transformed as a tensor. A moment tensor of a function $f(x_1,x_2,\ldots,x_n)$ is

$$T^{ijk\cdots} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}x^ix^jx^{k\cdots} \cdot f(x_1,x_2,\ldots,x_n)\,dx_1\,dx_2\,\ldots\,dx_n \tag{19}$$

where $i,j,k,\ldots$ take the values $1,2,\ldots,n$. (The rest of this section and also Section IV assume a knowledge of tensors and tensor transformations. Space limitations preclude the possibility of providing some background material on tensors. The interested reader may refer to [10] for a detailed discussion of tensors and tensor transformations.)

Proof: Consider the transformation of space

$$\hat{x}^i = Q^i_j\,x^j \quad \text{or} \quad x^j = \tilde{Q}^j_i\,\hat{x}^i \tag{20}$$

where the Einstein summation convention is used to represent the transformation. The moments after transformation are

$$\hat{T}^{ijk\cdots} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}\hat{x}^i\hat{x}^j\hat{x}^{k\cdots}\,f(\tilde{Q}^1_{\alpha_1}\,\hat{x}_{\alpha_1}\,,\,\tilde{Q}^2_{\alpha_2}\cdot\hat{x}_{\alpha_2}\,,\ldots,\tilde{Q}^n_{\alpha_n}\,\hat{x}_{\alpha_n})\,\hat{dx}_1\ldots\hat{dx}_n$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}Q^i_{\alpha_1}x_{\alpha_1}Q^j_{\alpha_2}x_{\alpha_2}\cdots \cdot f(x_1,x_2,\ldots,x_n)\frac{1}{|J|}dx_1dx_2\ldots dx_n \tag{21}$$

or

$$\hat{T}^{ijk\cdots} = Q^i_{\alpha_1}Q^j_{\alpha_2}\ldots T^{ijk\cdots}\frac{1}{|J|} \tag{22}$$

where $|J|$ is the Jacobian $|Q^i_j|$. If we consider the normalized moments $m^{ijk\cdots} = T^{ijk\cdots}/\eta$, where η is the zeroth order moment and

$$\hat{\eta} = \frac{\eta}{|J|} \tag{23}$$

then

$$\hat{m}^{ijk\cdots} = Q^i_{\alpha_1}Q^j_{\alpha_2}\ldots m^{\alpha_1\alpha_2\cdots} \tag{24}$$

which shows that m is indeed a tensor.

F. Invariant Construction

If $T^{ijk\cdots}$ is the tensor representation of a moment $m_{p_1p_2\cdots p_n}$ and the tensor components are $t_1,t_2,\ldots,t_k$ then (18) reduces to

$$\Psi(t'_1,t'_2,\ldots,t'_k) = \Delta^w\ \Psi(t_1,t_2,\ldots,t_k) \tag{25}$$

where Ψ is the algebraic invariant for the tensor. From this representation of invariants for moment tensors, we have the following theorem for invariant construction:

Theorem 3: An algebraic invariant of weight w and order d of a tensor T is a linear combination of terms that have been constructed from A by $d-1$ tensor multiplications and $|w|$ total alternations.

This theorem provides a means for constructing invariants from moments. As an example, the construction of moment invariants for order 2 and weight 2 are considered here for the 2-D and 3-D cases. The invariant is constructed by two alternations of the product $m^{ij}m^{kp}$. For the 2-D case it is

$$\Psi_1 = 2(m^{11}\,m^{22} - (m^{12})^2) \tag{26}$$

and for the 3-D case it is

$$\Psi_1 = 2\,[m^{11}\,m^{22} + m^{11}\,m^{33} + m^{22}\,m^{33} - (m^{12})^2 - (m^{13})^2 - (m^{23})^2] + 4\,[m^{11}m^{23} + m^{12}m^{23} - m^{13}m^{23} + m^{12}m^{33} - m^{13}m^{21} - m^{13}m^{22}]. \tag{27}$$

It should be noted that these Ψ's are not absolute invariants. An absolute invariant I_i can be computed from

$$I_i = \Psi_i\,\zeta^w \tag{28}$$

where ζ is the determinant of the transformation matrix.

III. Object Representation and Identification

For object identification using moment invariants, an object is represented in terms of a graph, each node of which represents a face or patch of the object. Arcs between nodes represent connectivity between the corresponding faces or patches. Associated with each node is a feature vector containing the moment invariant values. An important issue here is the number of moment invariants required for the completeness of representation. While for a given order of moment tensor the number of invariants is finite (according to Hilbert), the order of moment tensors has no upper bound. But it is not practical to use moment invariants of an arbitrarily high order. Also, as shown in the error analysis section, the error sensitivity of moment invariants increases with the tensor order. It has been experimentally found that if tensors up to the fourth order are used, good representation accuracy is obtained while maintaining the errors within reasonable limits.

For object identification, graph representations of objects are stored in the system library. Given an image of an object, its graph representation is computed. The sequence of processing operations required to convert raw data to the graph representation is described in [8]. The graph representation derived from data will be a subgraph of the graph representation of the complete object. The graph-matching algorithm described below is used for matching the data-derived graph representation against object graph representations stored in the system library for object identification.

A. Graph Matching

Let G_i represent the image graph and G_o represent the object graph. We pick a node of G_i and try to match it against all nodes of G_o. If a match is found, then the next node of G_i is picked and a match is attempted against the remaining nodes of G_o. This procedure is implemented as a search algorithm as follows.

Let the number of nodes in G_i be n, and let the number of nodes in G_o be m. The nodes of G_i are numbered as G_i^q, and those of G_o as G_o^k. Arrange sets of nodes of G_o in n levels. The level number is denoted by q. Each level contains the nodes in G_o with respect to which the qth node of G_i needs to be matched. The number of nodes in any level is $m - q$. The nodes in the above sets are numbered G_{io}^{st} when they correspond to a match between G_i^s and G_o^t. Here, s and t are dummy indices. A cost of 1 is assigned to a match between two nodes if the Euclidean distance between their moment invariant vectors is above a certain threshold (no match) and a cost of 0 if this distance is below the threshold (match). If two matching nodes are connected in either the image or object graph and the corresponding nodes in the other graph are not connected, a cost of 1 is assigned to the match, otherwise the connectivity cost is 0.

1) Main Algorithm:

```
For q from 0 to n-1 do:
 For k from 0 to m-q-1 do:
   †Call function: zero-cost-match (G_i^q, G_o^k)
   If (function zero-cost-match returns
```

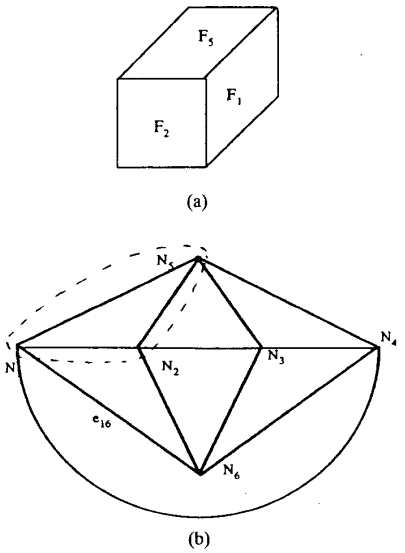

Fig. 1. Graph representation of a 3-D object. Subgraph of the portions seen by the camera: (a) object and (b) attributed graph.

```
    zero-cost)
     then (go to ⋆)
   else (next k)
 Call function back-track
 Next k
⋆Mark nodes G_i^q and G_o^k as an associated
  pair
Next q
Declare object identification
```

2) Function Zero-Cost-Match:

```
If (G_i^q and G_o^k are matched)
 then (If (G_i^q has marked neighbors)
   then (If (G_i^q and G_o^k have correspond-
     ing marked neighbors)
     then (return zero-cost)
  else (return unit-cost)
 else (return zero-cost)
else (return unit-cost)
```

3) Function Back-Track:

```
If q=0, declare object not identified,
 terminate.
 else
 (q=q-1
   For a from q to n-1 do:
   Check for next marked node at level q,
    let it be G_o^k
   For b from k to m-a-1 do:
   Do rest
 )
```

Note: The rest of the function back-track is essentially the same as from † of the main algorithm, and so is not repeated here. a and b are dummy indices used in function back-track to differentiate from the indices in the main algorithm.

```
Object Library
  1: Solarmax.OBJT
  2: Shuttle.OBJT
  3: Octbox.OBJT
  4: Octball.OBJT
  5: Cube.OBJT

  Choice? 5

  Enter roll, pitch, and yaw (space separated) : 45 45 45

Recognition Phase
  Attempting to match Solarmax.OBJT
  Sorry, no match

  Attempting to match Shuttle.OBJT
  Sorry, no match

  Attempting to match Octbox.OBJT
  Sorry, no match

  Attempting to match Octball.OBJT
  Sorry, no match

  Attempting to match Cube.OBJT
  OBJECT IDENTIFIED IS:
  Cube.OBJT

  Match correspondences:
  Wireframe face # --> Model face #
  0 --> a
  1 --> b
  2 --> e
```

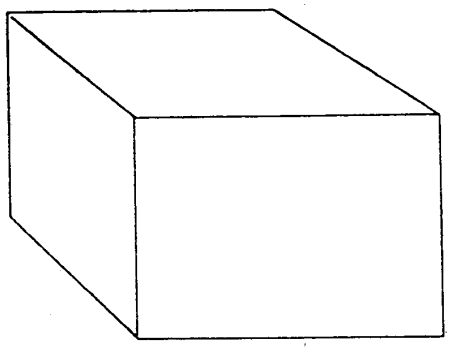

Fig. 2. Graph matching for object recognition.

B. Example

The graph representation of a 3-D object (a cube) is shown in Fig. 1, and simulation results of using the graph-matching algorithm described above for object recognition are shown in Fig. 2.

IV. Attitude Determination

Three-dimensional rigid body motion has two components — rotation and translation. If (x, y, z) are the coordinates of a point before motion and (x', y', z') are the coordinates after motion

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = R \begin{bmatrix} x \\ y \\ z \end{bmatrix} + T \tag{29}$$

where

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \tag{30}$$

is the rotation matrix and

$$T = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \tag{31}$$

is the translation vector.

The technique for attitude determination using camera data has been presented by Bamieh and deFigueiredo [6] and is briefly summarized here.

The translation components can be computed as

$$\Delta x = \frac{m_{10}}{m_{00}} \qquad \Delta y = \frac{m_{01}}{m_{00}}. \tag{32}$$

Δz cannot be computed by this method.

Consider the following unit rank tensors formed by contraction of second-, third-, and fourth-order moment tensors.

$$v^m = m^{ij}\ \epsilon_{ik}\ \epsilon_{jl}\ m_{klm}\ /\ \eta^4 \tag{33}$$

$$\lambda^m = m^{ij}\ m^{klm}\ \epsilon_{in}\ \epsilon_{jo}\ \epsilon_{kp}\ \epsilon_{lq}\ m^{nopq}\ /\ \eta^7 \tag{34}$$

where $\epsilon_{ik} = 0$ for $i = k, \epsilon_{12} = 1$, and $\epsilon_{21} = -1$. If v^m, λ^m denote the tensors for a surface patch after motion and Υ^m, Λ^m for before motion

$$\begin{bmatrix} v^1 \\ v^2 \end{bmatrix} = \begin{bmatrix} Q_1^1 & Q_2^1 \\ Q_1^2 & Q_2^2 \end{bmatrix} \begin{bmatrix} \Upsilon^1 \\ \Upsilon^2 \end{bmatrix} \tag{35}$$

$$\begin{bmatrix} \lambda^1 \\ \lambda^2 \end{bmatrix} = \begin{bmatrix} Q_1^1 & Q_2^1 \\ Q_1^2 & Q_2^2 \end{bmatrix} \begin{bmatrix} \Lambda^1 \\ \Lambda^2 \end{bmatrix}. \tag{36}$$

According to Cyganski and Orr [3], when an object undergoes a 3-D transformation, its tensors also undergo the same transformation. So after simplification, $Q_1^1 = r_1, Q_2^1 = r_2, Q_1^2 = r_4$, and $Q_2^2 = r_5$. The remaining parameters of the R matrix can be computed using the fact that the sum of squares of any row or column equals one.

For $n = 3$, we have

$$\Delta x = \frac{m_{100}}{m_{000}} \qquad \Delta y = \frac{m_{010}}{m_{000}} \qquad \Delta z = \frac{m_{001}}{m_{000}}. \tag{37}$$

Also, using an additional unit rank tensor obtained by contracting second-, fourth-, and fifth-order moment tensors

$$\xi^m = m^{ij}\ m^{klmn}\ \epsilon_{io}\ \epsilon_{jp}\ \epsilon_{kq}\ \epsilon_{lr}\ \epsilon_{ns}\ m^{opqrs}\ /\ \eta^8 \tag{38}$$

we have the following relations.

Let

$$Q = \begin{bmatrix} Q_1^1 & Q_2^1 & Q_3^1 \\ Q_1^2 & Q_2^2 & Q_3^2 \\ Q_1^3 & Q_2^3 & Q_3^3 \end{bmatrix}. \tag{39}$$

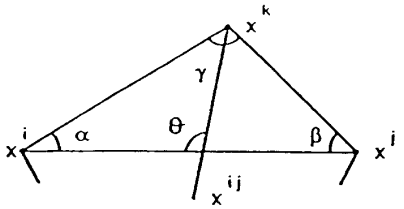

Fig. 3. Symmetry conditions for polygons.

Then

$$\begin{bmatrix} v^1 \\ v^2 \\ v^3 \end{bmatrix} = Q \begin{bmatrix} \Upsilon^1 \\ \Upsilon^2 \\ \Upsilon^3 \end{bmatrix}; \quad \begin{bmatrix} \lambda^1 \\ \lambda^2 \\ \lambda^3 \end{bmatrix} = Q \begin{bmatrix} \Lambda^1 \\ \Lambda^2 \\ \Lambda^3 \end{bmatrix}; \quad \begin{bmatrix} \xi^1 \\ \xi^2 \\ \xi^3 \end{bmatrix} Q \begin{bmatrix} \Xi^1 \\ \Xi^2 \\ \Xi^3 \end{bmatrix} \tag{40}$$

The Q parameters can be computed from the tensor values. The r parameters can then be computed using the following relations:

$$Q_1^1 = r_1 \quad Q_2^1 = r_2 \quad Q_3^1 = r_3 \quad Q_1^2 = r_4 \quad Q_2^2 = r_5$$
$$Q_3^2 = r_6 \quad Q_1^3 = r_7 \quad Q_2^3 = r_8 \quad Q_3^3 = r_9.$$

Thus the rotation paramters can be computed directly from the Q parameters.

If a face or patch has any axis of reflection symmetry, its tensors all go to zero [10]. To avoid this condition, given a face or patch, we first check it for symmetry. If it has symmetry or will have symmetry under an affine transformation, we distort it so that it will not have symmetry under an affine transform. While the symmetry conditions for a polygon are easy to check in terms of its vertices, the same is not true for curved faces. We overcome this difficulty by fitting a plane face to the vertices of a curved face and using this polygon for attitude determination. As the object is rigid, the polygon undergoes the same transformation as the curved face. So from here on we discuss symmetry conditions only for polygons.

A. Symmetry Conditions for Polygons

To check the symmetry conditions of a polygon, we use an approach where we check the polygon for symmetry on a vertex-by-vertex basis. The symmetry conditions for a polygon depend on whether it has odd or even number of vertices. If the number of vertices is odd, the axis of symmetry will pass through a vertex and the midpoints of two other vertices. If the number of vertices is even, the symmetry axis will pass through two vertices or two vertex midpoints.

Let the vertices of a polygon be designated $x^1, x^2, \ldots, x^N$ where $x^i = [x_i, y_i]^T$. Also, let the midpoint of vertices x^i and x^j be x^{ij}, where $x^{ij} = (x^i + x^j)/2$.

1) Odd Polygons:

Theorem 4: If a symmetry axis passes through x^k and x^{ij}, then

$$\langle x^i - x^j , x^k - x^{ij} \rangle = 0 \tag{41}$$

for $i \neq k; j \neq k$.

Proof: Consider the triangle $x^i x^j x^k$, which is a part of the polygon in Fig. 3.

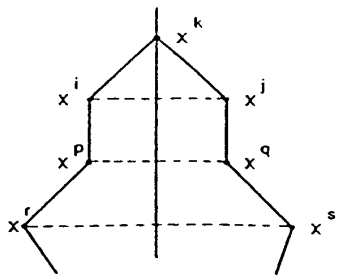

Fig. 4. Symmetry conditions for polygons.

Here, $x^{ij} - x^k$ is the axis of symmetry. For the triangle to be symmetric, it must be isosceles, i.e., $\alpha = \beta$. But

$$\alpha + \frac{\gamma}{2} + \theta = \pi \tag{42}$$

where

$$\gamma = \pi - (\alpha + \beta). \tag{43}$$

Therefore

$$\frac{(\alpha - \beta)}{2} + \theta = \frac{\pi}{2} \tag{44}$$

$$\Rightarrow \theta = \frac{\pi}{2}. \tag{45}$$

The proof is valid in general for all odd polygons because further pairs of polygons down to the maximally distant vertex pair from x^k would form lines parallel to $x^i - x^j$, as shown in Fig. 4. The maximally distant vertex with respect to another vertex is defined as the vertex reached by traversing the maximum number of edges along the polygon. For x^k, x^r and x^s are the maximally distant vertices in the above figure as both are three edges away. In general, for an odd n-gon, for vertex x^k, the maximally distant vertices are $x^{k+(n-1)/2}$ and $x^{k+(n+1)/2}$. In checking for a symmetry axis, one needs to check a vertex and its maximally distant vertex pair. A vertex need not be checked with any other vertex pair. So every vertex in the polygon is checked for a symmetry axis with respect to its maximally distant vertex pair and if no vertex is found to have an axis of symmetry through it, the polygon is declared nonsymmetric. For an n-gon the algorithm requires $O[n]$ processing time.

2) Even Polygons: For an even vertex polygon, two kinds of axes of symmetry can exist.

1) Axis through two vertices.
2) Axis through two midpoints of vertices.

Theorem 5—Axis Through Two Vertices: An axis of symmetry will pass through vertices x^a and x^b if there exists another pair of vertices x^i and x^j such that

$$\langle x^i - x^j , x^a - x^{ij} \rangle = 0 \tag{46}$$

and

$$\langle x^i - x^j , x^b - x^{ij} \rangle = 0 \tag{47}$$

for $i \neq a$ or $b, j \neq a$ or b. x^i and x^j will form a pair of symmetric points with respect to $x^a - x^b$.

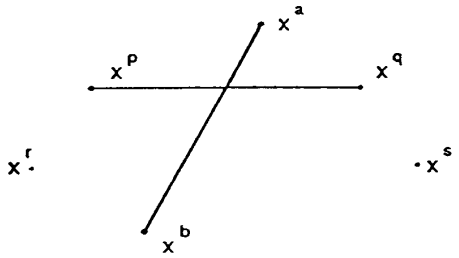

Fig. 5. Symmetry conditions for polygons.

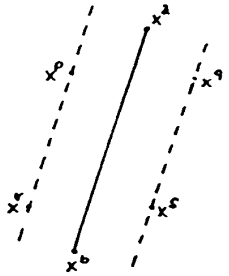

Fig. 6. Symmetry conditions for polygons.

The proof is analogous to the one for odd polygons given above and so is not repeated here. As for odd polygons, here also we check for existence of symmetry axis between a vertex x^k and its maximally distant vertex $x^{(k+n/2)}$ where n is the total number of vertices in the polygon. The search for existence of symmetry axis is conducted with respect to every vertex. The entire search would thus require $O[n]$ processing time. If x^p is the maximally distant vertex with respect to x^k then if (x^k, x^p) have been checked for symmetry (x^p, x^k) need not be checked. Thus by marking the vertices which have already been checked and not checking them again we reduce the processing time to $O[n/2]$.

Theorem 6—Axis Through Two Midpoints of Vertices: If $\langle x^p - x^q \rangle$ and $\langle x^r - x^s \rangle$ are parallel and $\langle x^p - x^r, x^r - x^s \rangle = \alpha$, $\langle x^q - x^s, x^r - x^s \rangle = \beta$, then an axis of symmetry will pass through x^{pq} and x^{rs} if $\alpha = \beta$ or $\alpha = \pi - \beta$.

Proof: In Fig. 5, let $(x_a - x_b)$ be the symmetry axis and x^p, x^q be the two points symmetric with respect to it. Two other points x^r, x^s can be symmetric with respect to $(x^a - x^b)$ such that $(x^p - x^r)$ and $(x^q - x^s)$ are parallel to $(x^a - x^b)$ or they are not parallel.

Case 1): $(x^p - x^r)$ and $(x^q - x^s)$ are parallel to $(x^a - x^b)$. In Fig. 6, as $(x^p - x^r)$ and $(x^q - x^s)$ are parallel , by definition they make the same angle with any line, specifically with $(x^r - x^s)$, so $\alpha = \beta$.

Case 2): As in Fig. 7, if $(x^p - x^r)$ and $(x^q - x^s)$ are not parallel, they will meet at some point x^t that will lie on $(x^a - x^b)$. So we have a triangle $x^r x^t x^s$ with an axis of symmetry through x^t and x^{rs}. By theorem 4

$$\langle x^t - x^r , x^r - x^s \rangle = \langle x^t - x^s , x^s - x^r \rangle \Rightarrow \alpha + \beta = \pi. \quad (48)$$

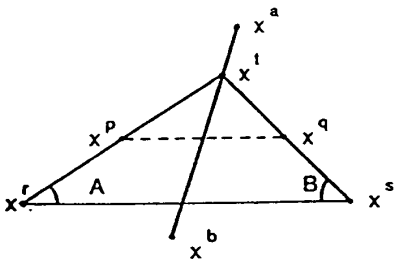

Fig. 7. Symmetry conditions for polygons.

As in the case of vertices, we define a maximally distant edge for a given edge. For an edge $(x^{k+1} - x^k)$, the maximally distant edge is $(x^{(k+1)+n/2} - x^{k+n/2})$ where n is the number of vertices in the polygon. In checking for a symmetry axis, an edge is checked only with its maximally distant edge. Also, if an edge has been checked as a maximally distant edge, it is marked and need not be checked again. The total processing time for an n-vertex polygon is $O[n/2]$.

From theorems 5 and 6 it is obvious that the total processing time for a symmetry check on an even polygon is $O[n]$.

From theorem 4 and the above results we have:

Lemma: A symmetry check for an n-vertex polygon has an $O[n]$ processing time.

If the face has an axis of symmetry as verified above, then it is subjected to distortion that removes the axes of symmetry. It has been found that while for any particular distortion there always exists an affine transformation that would yield an axis of symmetry, if the polygon is subjected to two separate distortions that are antisymmetric with respect to each other, there exists no affine transformation that yields axes of symmetry for both cases. If these two distortions are refered to as D_1 and D_2, then the procedure consists of subjecting the polygon to D_1 and checking it for whether it has any axis of symmetry. If it does, it is subjected to D_2 and by the above argument it will not have any axis of symmetry. Note that all deformations are conducted in the plane of the polygon so that the attitude is preserved under deformation.

B. Example

The technique developed above for attitude determination using moment tensors was tested on real image data. The test object was a model of the space shuttle shown in Fig. 8. Attitude determination results for various positions of the shuttle are given in Table I.

V. Error Analysis

An error analysis for moment invariants and attitude results is presented in the following subsections.

A. Moment Invariants

Theorem 7: Error sensitivity of moment invariants increases with their order.

Proof: Let the function $f(x, y)$ be expanded about a point (x_0, y_0), this point being chosen such that it lies within

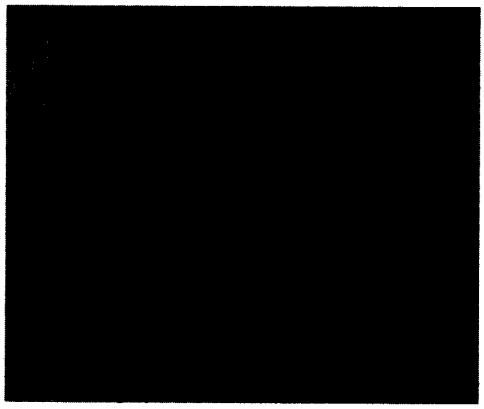

Fig. 8. Test object.

TABLE I
ATTITUDE DETERMINATION RESULTS

Actual Attitude			Computed Attitude			Error		
10.0	0.0	0.0	9.44	0.37	0.18	0.56	-0.37	-0.18
0.0	-30.0	0.0	0.35	-29.58	0.42	-0.35	-0.42	-0.42
0.0	0.0	20.0	-0.47	0.19	20.74	0.47	-0.19	-0.74
-10.0	0.0	-20.0	-9.61	0.76	-20.24	-0.39	-0.76	0.24
-15.0	10.0	0.0	-15.31	10.54	0.63	0.31	-0.54	-0.63
10.0	10.0	10.0	10.83	11.39	10.23	-0.83	-1.39	-0.23
10.0	-20.0	10.0	9.56	-20.65	10.18	0.44	0.65	-0.18
20.0	-10.0	-20.0	19.69	-11.53	-20.02	0.31	1.53	0.02
14.0	-16.0	11.0	14.58	-15.81	10.65	-0.58	-0.19	0.35
-10.0	11.0	-18.0	-9.67	11.39	-18.59	-0.33	-0.39	0.59

a rigid planar patch, i.e., the function's derivatives of all order exist and are continuous at (x_0, y_0).

$$f(x,y) = f(x_0,y_0) + [(x - x_0)(y - y_0)]\, f'(x,y)|_{x_0,y_0} + \frac{[x - x_0)(y - y_0)]^2}{2!} f''(x,y)|_{x_0,y_0} + \ldots. \tag{49}$$

Considering terms only up to first-order expansion and letting the error be

$$(x,y) \longrightarrow (x + \delta x, y + \delta y) \tag{50}$$

the error term in $f(x,y)$ is

$$E = \delta x\,(y - y_0) + \delta y\,(x - x_0). \tag{51}$$

Considering the Taylor series expansion of $x^p y^q f(x,y)$, and computing

$$\mu_{pq} = \int\int x^p y^q f(x,y) \tag{52}$$

the error term comes out as

$$\int\int x^p y^q [\,\delta x(y - y_0) + \delta y(x - x_0)]dxdy. \tag{53}$$

So as $(p + q)$ increases, the error term gets multiplied by a larger factor. This means that error sensitivity increases with $(p+q)$, i.e., with the order of moment invariants. In support of this argument, Fig. 9 shows numerical results for relationship between percent error in the wireframe vertex coordinates and percent error in the moment invariant values for the first-, second-, and fourth-order moment invariants. The third-order moment invariants are unstable and hence not used. This fact can be used to derive a representation of a polygon in terms of a finite number of moment invariants. Assuming that the algorithm has been designed to handle a 30% error in the moment invariants, using the three invariants in Fig. 9, a maximum of 20% error in input can be tolerated. If more invariants were used, the maximum input error that could be tolerated would decrease.

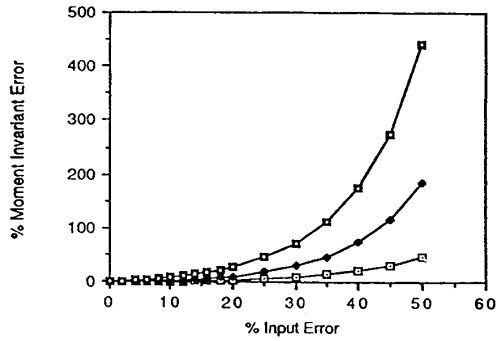

Fig. 9. Moment invariant error analysis.

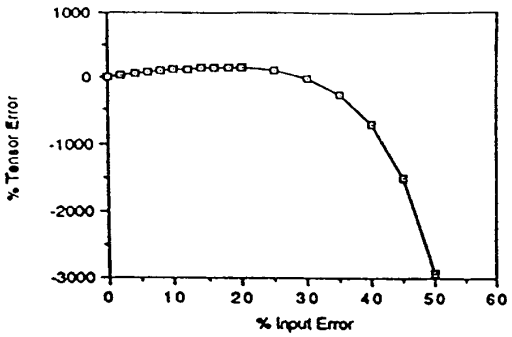

Fig. 10. Attitude determination error analysis.

B. Attitude

Figs. 10–13 show the relationship between percent error in input and the tensors $v_1, \lambda_1, v_2, \lambda_2$. The relationships are linear for input errors up to 10% and then become nonlinear.

Solving (35) and (36) for the Q parameters gives

$$Q_1^1 = \frac{v^1\Lambda^2 - \lambda^1\Upsilon^2}{\Upsilon^1\Lambda^2 - \Upsilon^2\Lambda^1}. \tag{54}$$

For the linear region

$$\tilde{Q}_1^1 = \frac{(v^1 + a_1\delta x_1)\Lambda^2 - (\lambda^1 + b_1\delta x_1)\Upsilon^2}{\Omega} \tag{55}$$

where $\Omega = \Upsilon^1\Lambda^2 - \Upsilon^2\Lambda^1$

$$\tilde{Q}_1^1 = \frac{v^1\Lambda^2 - \lambda^1\Upsilon^2 + \delta x_1(a_1\Lambda^2 - b_1\Upsilon^2)}{\Omega}.$$

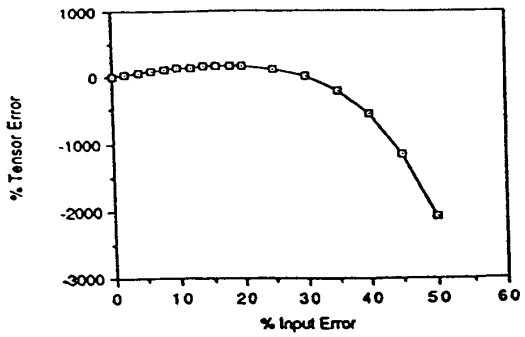

Fig. 11. Attitude determination error analysis.

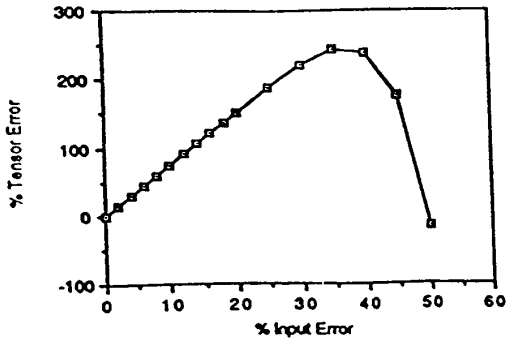

Fig. 12. Attitude determination error analysis.

Therefore

$$\tilde{Q}_1^1 = Q_1^1 + \delta x_1 K_1. \tag{57}$$

Similarly

$$\tilde{Q}_2^1 = Q_2^1 + \delta x_1 K_2 y \tag{58}$$

$$\tilde{Q}_1^2 = Q_1^2 + \delta x_1 K_3 \tag{59}$$

$$\tilde{Q}_2^2 = Q_2^2 + \delta x_1 K_4. \tag{60}$$

As four of the rotational parameters are linearly related to the Q's, their error also increases linearly with input. For the remaining five parameters also, only the linear terms remain on expansion.

VI. Conclusions

The main contributions of this paper are:

1) Extension of the concept of moments and moment invariants to n-dimensional polynomial functions.
2) Attitude determination using moments. While the idea was originally presented in [6], it had the limitation that it required the polyhedron under consideration to have nonsymmetric faces. A technique to overcome this limitation was presented in this paper.

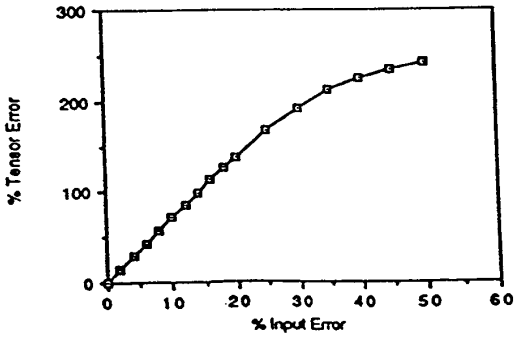

Fig. 13. Attitude determination error analysis.

3) Error analyses of moments and moment invariants, as related to object identification and attitude determination.
4) A common mathematical framework has been developed for the integration of information from multiple sensors in the same format. This is of importance in sensor fusion where information from different sensors has to be integrated together. In our representation scheme the underlying mathematical model is common—different sensor outputs correspond to different dimensions of the model.

References

[1] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 179–187, 1962.
[2] S. A. Dudani, K. F. Breeding, and R. B. McGee, "Aircraft identification by moment invariants," *IEEE Trans. Computers*, vol. C-26, no. 2, pp. 39–45 1977.
[3] D. Cyganski and J. A. Orr, "Object identification and orientation in 3-space with no point correspondence information," in *Proc. Int. Conf. ASSP*, 1984, pp. 23.8.1–23.8.4.
[4] F. A. Sadjadi and E. L. Hall, "Three-dimensional moment invariants," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-2, pp. 127–1136, 1980.
[5] G. Gurevich, *Theory of Algebraic Invariants*. London: Noordhoff, 1964..
[6] B. Bamieh and R. J. P. deFigueiredo, "A general moment invariants/attributed-graph method for three-dimensional object recognition from a single image," *IEEE J. Robotics Automat.*, vol. RA-2, no. 1, pp. 31–41, 1986.
[7] V. Markandey, H. Tagare, and R. J. P. deFigueiredo, "A technique for 3-D robot vision for space applications," in *Proc. JPL Workshop Space Telerobotics*, 1987, vol. II, pp. 111–123.
[8] R. J. P. deFigueiredo *et al.*, "An integrated software module for conversion of noisy camera data into symbolic form," Tech. Rep. EE8718, Dept. of Elec. and Comput. Eng., Rice Univ., Houston, TX, 1987.
[9] R. J. P. deFigueiredo and N. Kehtarnavaz, "Model-based orientation-independent 3-D machine vision techniques," *IEEE Trans. Aerospace Electron. Syst*, vol. 24, no. 5, pp. 597–607, 1988.
[10] B. Bamieh and R. J. P. deFigueiredo, "General moment invariants and their application to 3D object recognition from a single image," Tech. Rep. EE8513, Dept. of Elec. and Comput. Eng., Rice Univ., Houston, TX, 1985.
[11] F. A. Sadjadi and E. L. Hall: "3-D moment forms: Their construction and application to object identification and positioning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 11, no. 10, pp. 1053–1064, 1989.
[12] T. L. Faber and E. M. Stokley, "Affine transform determination for 3-D objects: A medical Imaging application," in *Proc. IEEE Conf. Comput. Vision Pattern Recognition*, 1986.

Vishal Markandey (S'86–M'90) received the bachelor's degree in electronics and communication engineering from Osmania University, India, in 1985 and the master's degree in electrical engineering from Rice University, Houston, TX, in 1988.

He has been Texas Instruments Incorporated, Dallas, TX, since 1988 where he is a Member of Technical Staff.

Rui J. P. deFigueiredo (S'54–M'59–SM'74–F'76) received the S.B. and S.M. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, and the Ph.D. degree in applied mathematics from Harvard University, Cambridge, MA

From 1965 to 1990, he was a member of the faculty of Rice University, Houston, TX. Since 1990, he has held the position of Professor of Electrical and Computer Engineering and of Mathematics and Director of the Laboratory for Intelligent Machines and Systems at the University of California, Irvine. The main thrust of his current research is in the area of artifical neural networks with applications to intelligent signal and image processing and intelligent control. He has 220 publications to his credit.

Prof. deFigueiredo has chaired or co-chaired eight international conferences, held six editorial positions of archival journals, and chaired or participated in a number of national and international committees, panels, and missions. For these contributions he has received several awards and certificates of recognition.