PIECEWISE CUBIC MAPPING FUNCTIONS FOR IMAGE REGISTRATION

ARDIEH GOSHTASBY
Department of Computer Science, University of Kentucky, Lexington, KY 40506-0027, U.S.A.

(Received 18 September 1986)

Abstract—In a recent paper, a registration technique was proposed that registered two images by triangulating the images and mapping corresponding triangular regions in the images using a linear function. This paper extends the previous technique to include piecewise nonlinear functions as mapping functions. A nonlinear mapping function in addition to providing continuous mapping, provides smooth transition from one local function to another. The obtained mapping function which is a piecewise combination of local mapping functions is therefore continuous and smooth all over. A technique based on the Clough–Tocher subdivision is described which determines piecewise cubic mapping functions for image registration.

1. INTRODUCTION

Image registration is the process of overlaying two images of the same scene. This process is required in many applications such as change detection, depth perception, and scene classification. The general approach to image registration is to locate unique features in the images, establish correspondence between the features, and use the feature correspondences to determine a mapping function that can register the images.

Some image registration techniques assume that the images have only linear geometric differences and register the images by the transformation of the Cartesian coordinate systems or the affine transformation. Some image registration techniques allow nonlinear geometric differences between the images but use global polynomial mapping functions to register the images. In these techniques, if the geometry of the images is locally different, the global mapping functions that are used would not be able to register the images well. This is mostly because the least-squares technique that is used to determine the parameters of the mapping functions, averages a local geometric difference over the whole image area independent of the position of the difference. There are some techniques that allow local geometric difference between the images. These techniques register the images point by point by establishing correspondence between the point features in the images. No formulated mapping function exists that would map one image as a whole into another.

In this paper, a technique is proposed that uses piecewise mapping functions to register the images. A mapping function is modeled as a composite surface which is made up of triangular cubic patches, each patch characterizing local geometric difference between the images. This paper is an extension of an earlier paper that used piecewise linear mapping functions to register images. In Ref. (1) knowing a number of feature points from each image, the images were divided into triangular areas. Then registration was carried out between corresponding triangular areas in the images using a linear mapping function. The overall effect was that the images were registered by piecewise linear mapping functions. The mapping function was a linear one and could provide continuity at the boundary between neighboring functions but it could not provide a smooth transition from one local function to another. In this paper, the piecewise mapping idea is extended to include nonlinear functions. Nonlinear mapping functions provide smooth transition from one local function to another.

Consider knowing n control points from each image. Let \( (x_i, y_i) \), \( i = 1, n \) be the coordinates of control points from one of the images which we will refer to as the reference image. Also let \( (X_i, Y_i) \), \( i = 1, n \) be the coordinates of control points in the other image, which we will refer to as the sensed image. Suppose that the correspondence between the control points in the two images is also known. The objective is to determine two mapping functions \( X = f(x, y) \) and \( Y = g(x, y) \) that can map the sensed image into the reference image. \( f \) and \( g \) represent the geometric difference between the images. Knowing the positions of
corresponding control points in the images the mapping functions parameters can be determined by solving the following system of equations.

\[ X_i = f(x_i, y) \quad (1.1) \]
\[ Y_i = g(x_i, y) \quad \text{for } i = 1, n. \quad (1.2) \]

In Ref. (1), functions \( f \) and \( g \) were assumed to be linear with three unknown parameters. Then the parameters of \( f \) and \( g \) were determined by the coordinates of three vertices of the triangle over which the functions were defined. \( f \) and \( g \) can be nonlinear functions also as long as they provide continuity and smooth transition from one local function to another.

The problem of determining functions \( f \) and \( g \) can be solved easily if we transform it into a surface fitting problem. Determining the \( X \)-component mapping function that satisfies (1.1) is equivalent to determining a smooth surface that passes through points \((x_i, y_i, X_i)\), \( i = 1, n \). The requirement that a mapping function characterize local geometric difference between the images can be interpreted as determining a surface that depends on local measurements (nearby control points). Composite surfaces which are obtained by joining many local surface patches with each patch characterizing measurements in a local neighborhood, provide this property.

Considerable research has been carried out in the area of Computer Aided Geometric Design to fit composite surfaces to scattered data. The obvious approach is to triangulate the data and fit a surface patch to each triangle in such a way that neighboring patches join smoothly. Powell and Sabin used quadratic patches,\(^{16}\) Percell used cubic and quartic patches,\(^{16}\) and Akima used quintic patches\(^{18}\) to construct composite surfaces. In the following, we show how piecewise cubic mapping functions can be obtained by fitting a composite cubic surface to scattered 3-D data. We will only determine the \( X \)-component mapping function. The \( Y \)-component mapping function is determined similarly.

To determine the \( X \)-component mapping function, or equivalently to determine a composite surface that passes through points \((x_i, y_i, X_i)\), \( i = 1, n \), we triangulate the data in the \( xy \)-plane (in the reference image), and define a cubic patch over each triangle in such a way that neighboring patches join smoothly. The following sections describe each step of the process in more detail. We assume that the geometric difference between the images is continuous and smooth. We will not consider registration of images that have occluded parts because then they will have discontinuous geometric differences. Also, images from 3-D scenes containing prismatic objects will not be considered because they will have discontinuous gradients in their geometric differences.

2. COMPOSITE SURFACES

In this section, given 3-D points \((x_i, y_i, X_i)\), \( i = 1, n \), we determine a composite surface that passes through the given points and provides first derivative continuity everywhere (this is a requirement for the surface to be smooth everywhere). The surface is obtained by triangulating the data in the \( xy \)-plane and constructing a triangular patch over each triangle in such a way that neighboring patches join smoothly.

2.1. Triangulation

Given points \((x_i, y_i)\), \( i = 1, n \), we would like to divide the convex hull of the points into triangular regions. There are different triangular configurations for any data set. We are interested in that configuration in which points inside a triangle are closer to the three vertices of the triangle than to any other points in the set. This is called the optimal triangulation and it avoids generation of triangles with sharp angles and long edges. Optimal triangulation ensures that nearby measurements are used to determine a surface patch and therefore the obtained composite surface would be

![Fig. 1. The circle test. (a) For any two neighboring triangles ABC and DBC, pass a circle through triangle ABC. If triangle DBC falls inside the circle then replace the triangles by BDA and CDA as shown in (b). Otherwise, leave the triangles as they are.](image)
Insensitive to the far away measurements (control points).

Various algorithms are available for determining the optimal triangulation. We briefly mention three of the well known ones here. It is possible to start with any triangulation and work toward an optimal one. To do so, for each pair of neighboring triangles, say ABC and DBC as in Fig. 1, pass a circle through one of the triangles, say ABC. If the other triangle falls inside the circle then replace the two triangles by triangles BDA and CDA. Otherwise leave the triangles as they are. This technique has computational complexity of $O(n^3)$.

Lee and Schachter introduce a recursive algorithm based on the divide and conquer idea that determines the optimal triangulation of a set of points with computational complexity $O(n \log n)$. In this method the data is divided into the left-half and the right-half using the $x$-value of the points. Then each subset is divided into halves themselves and the process is recursively continued until each subset contains only three or four points, which is easy to triangulate. Next the triangles from each subset are merged to obtain larger subsets and again the merging process is recursively continued until all the triangular subsets are merged into one overall triangulation.

Green and Sibson determined optimal triangulation of a set of points by the Dirichlet tessellation of the set. Dirichlet tessellation is obtained by creating a region for each set point such that points in that region are closer to the set point assigned to the region than to any other set points. Each region is obtained by the intersections of the open half-plane bounded by the normal bisectors of lines joining two set points. Once Dirichlet tessellation becomes known, triangulation is obtained by joining set points of neighboring regions. The computational complexity of this algorithm is $O(n)$.

Our preference has been with the divide and conquer algorithm because of its speed and ease of programming.

2.2. Cubic triangular patches

A cubic patch $f$ is a bivariate polynomial of degree three with ten parameters,

$$f(x, y) = a_1 + a_2 x + a_3 y + a_4 x^2 + a_5 y^2 + a_6 x^3 + a_7 y^2 + a_8 x^2 y + a_9 xy^2 + a_{10} y^3.$$

These parameters can be determined if we can obtain ten relations between them. Assuming A, B, and C are vertices of a triangle [see Fig. 2(a)], we can obtain three relations by substituting the coordinates of A, B, and C into the equation of the patch. From the partial derivatives of the patch with respect to $x$ and $y$ at the three vertices of the triangle we obtain six more relations. Since a patch should join smoothly with the neighboring patches, this means that the partial derivatives of the two patches that share the same triangle edge, in the direction normal to the edge, should be the same. This provides three more relations, bringing the total number of relations to twelve. Since the number of unknown parameters is ten, the problem is overconstrained and there is no solution to it.

To be able to obtain a solution, a triangle is divided into three subtriangles as has been suggested by Clough and Tocher; see Fig. 2(b). O can be any point inside the triangle. Now if we fit a cubic patch to each of the subtriangles OAB, OBC, and OAC we would have three patches producing in overall thirty unknown parameters. We need thirty relations to determine these parameters.

Suppose $(x_A, y_A, X_A), (x_B, y_B, X_B),$ and $(x_C, y_C, X_C)$ are coordinates of points corresponding to A, B, and C. Also suppose that the partial derivatives of the patch that fits the three points, with respect to $x$ and $y$, are known. Let these partial derivatives be

$$\frac{\partial f}{\partial x_A}, \frac{\partial f}{\partial y_A}, \frac{\partial f}{\partial x_B}, \frac{\partial f}{\partial y_B}, \frac{\partial f}{\partial x_C}, \frac{\partial f}{\partial y_C}.$$ We will show later how to estimate these derivatives if they are not given. Finally, we assume that partial derivatives of the patch, in the direction

![Fig. 2. Dividing a triangle into the Clough-Tocher subtriangles.](image-url)
normal to each edge of the triangle, at a point on each edge are given. In other words assume that \( \frac{\partial f}{\partial n_A} (x_D, y_D) \), and \( \frac{\partial f}{\partial n_B} (x_D, y_D) \) are given, see Fig. 2(a). Let the patches that fit to subtriangles OBC, OAB, and OAC be, correspondingly, represented by:

\[ f_1(x, y) = a_{11} + a_{12}x + a_{13}y + a_{14}x^2 + a_{15}xy + a_{16}y^2 + a_{17}x^3 + a_{18}xy^2 + a_{19}y^3 + a_{20}x^2y + a_{21}x^3y + a_{22}xy^3 + a_{23}y^4, \]

(2.1)\( f_2(x, y) = a_{21} + a_{22}x + a_{23}y + a_{24}x^2 + a_{25}xy + a_{26}y^2 + a_{27}x^3 + a_{28}xy^2 + a_{29}y^3 + a_{30}x^2y + a_{31}x^3y + a_{32}xy^3 + a_{33}y^4, \)

(2.2)\( f_3(x, y) = a_{31} + a_{32}x + a_{33}y + a_{34}x^2 + a_{35}xy + a_{36}y^2 + a_{37}x^3 + a_{38}xy^2 + a_{39}y^3 + a_{40}x^2y + a_{41}x^3y + a_{42}xy^3 + a_{43}y^4. \)

(2.3)\[ \]

Now we can write the following relations for patch \( f \):

1. Since patch \( f_1 \) passes through points \((x_B, y_B, X_B) \) and \((x_C, y_C, X_C) \) then,

\[ X_B = a_{11} + a_{12}x_B + a_{13}y_B + a_{14}x_B^2 + a_{15}x_By_B + a_{16}y_B^2 + a_{17}x_B^3 + a_{18}x_By_B^2 + a_{19}y_B^3 + a_{20}x_B^2y_B + a_{21}x_B^3y_B + a_{22}x_By_B^3 + a_{23}y_B^4 \]

(3.1)\[ X_C = a_{31} + a_{32}x_C + a_{33}y_C + a_{34}x_C^2 + a_{35}x_Cy_C + a_{36}y_C^2 + a_{37}x_C^3 + a_{38}x_Cy_C^2 + a_{39}y_C^3 + a_{40}x_C^2y_C + a_{41}x_C^3y_C + a_{42}x_Cy_C^3 + a_{43}y_C^4 \]

(3.2)\[ \]

2. From the partial derivatives of the patch with respect to \( x \) and \( y \) at point \((x_B, y_B, X_B) \) we have,

\[ \frac{\partial f}{\partial x} = a_{12} + 2a_{14}x + a_{15}y + 3a_{16}x^2 + 2a_{17}x^3 + 2a_{18}xy + a_{19}x^2y + a_{20}x^3y + a_{21}x^2y^2 + 2a_{22}x^3y^2 + 3a_{23}x^3y^3. \]

(3.3)\[ \]

3. From the partial derivatives at point \((x_C, y_C, X_C) \) we have,

\[ \frac{\partial f}{\partial x} = a_{32} + 2a_{34}x + a_{35}y + 3a_{36}x^2 + 2a_{37}x^3 + 2a_{38}xy + a_{39}x^2y + a_{40}x^3y + a_{41}x^2y^2 + 2a_{42}x^3y^2 + 3a_{43}x^3y^3. \]

(3.4)\[ \]

4. From the partial derivative at direction normal to BC at point \( P \) we have,

\[ \frac{\partial f}{\partial n_B} = \frac{\partial f}{\partial y}. \]

(3.7)\[ \]

where \( f'_1 \) is \( f_1 \) after being transformed so that the new \( x \)-axis is parallel to BC, and \( \frac{\partial f}{\partial y} \) is partial derivative of \( f_1 \) with respect to \( y \) (\( y' \) being \( y \) after the transformation). This gives the partial derivative of the patch in the direction normal to BC.

Similarly, we obtain seven relations for each of the patches \( f_1, f_2 \), and \( f_3 \) bringing the total number of equations to twenty one. Nine other relations can be obtained as follows.

1. Since point O is common between patches \( f_1, f_2 \), and \( f_3 \) then we have,

\[ f_1(x_O, y_O) = f_2(x_O, y_O) \]

(4.1)\[ f_1(x_O, y_O) = f_3(x_O, y_O) \]

(4.2)\[ \]

2. Since patches \( f_1, f_2 \), and \( f_3 \) should have the same partial derivatives at point \( O \) then we have,

\[ \frac{\partial f_1}{\partial x}(x_O, y_O) = \frac{\partial f_2}{\partial x}(x_O, y_O) \]

(4.3)\[ \]

\[ \frac{\partial f_1}{\partial y}(x_O, y_O) = \frac{\partial f_2}{\partial y}(x_O, y_O) \]

(4.4)\[ \]

\[ \frac{\partial f_1}{\partial x}(x_O, y_O) = \frac{\partial f_3}{\partial x}(x_O, y_O) \]

(4.5)\[ \]

\[ \frac{\partial f_1}{\partial y}(x_O, y_O) = \frac{\partial f_3}{\partial y}(x_O, y_O) \]

(4.6)\[ \]

3. Since patches \( f_1, f_2 \), and \( f_3 \) should join smoothly, then the two patches that share the same edge should have the same partial derivative in the direction normal to the edge. Referring to Fig. 2(b) we can therefore have,

\[ \frac{\partial f_2}{\partial n_1}(x_U, y_U) = \frac{\partial f_3}{\partial n_1}(x_U, y_U) \]

(4.7)\[ \]

\[ \frac{\partial f_1}{\partial n_2}(x_W, y_W) = \frac{\partial f_2}{\partial n_2}(x_W, y_W) \]

(4.8)\[ \]

\[ \frac{\partial f_1}{\partial n_3}(x_V, y_V) = \frac{\partial f_3}{\partial n_3}(x_V, y_V) \]

(4.9)\[ \]

where \( U, V, \) and \( W \) are points on OA, OC, and OB, respectively.

From the above thirty relations we can obtain the thirty parameters belonging to patches \( f_1, f_2 \), and \( f_3 \).

2.3. Estimation of the partial derivatives

Since usually only positional data about the control points is available, there is a need to estimate the partial derivatives at the points so that fitting of a composite surface to the points becomes possible.

Suppose we are interested in estimating the partial derivatives at point \( P = (x_P, y_P, X_P) \) of a surface being fitted to the data. An easy method would be to fit a bivariate polynomial of degree two with six parameters to point \( P \) and five of its nearest neighbors and then determine the partial derivatives of the obtained polynomial at point \( P \).

A more accurate estimation is obtained if we use more than six points. Lawson estimated the partial
Piecewise cubic mapping functions

529
derivatives by a weighted quadratic least-squares method using n nearest points of $P_n$, where $n \geq 6$. In this technique, parameters of $f(x, y)$, a polynomial of degree two with six parameters, were determined by minimizing,

$$E = \sum_{i=1}^{n} (X_i - f(x_i, y_i))^2 W_i,$$

where $W_i = [R - \sqrt{(x_0 - x)^2 + (y_0 - y)^2}]^2$ and $R$ is the longest distance between $P_0$ and the n points.

To estimate the partial derivatives at $P_0$ using point $P_i$ and $n$ of its nearest points, $P_0, P_1, \ldots, P_n$, Akima's determined vector products $(P_i - P_j) \times (P_j - P_k)$ with $P_i$ and $P_j$ being all possible combinations of the points. Then determined the average of the resultant vector products,

$$n = \frac{1}{T} \sum_{i=1}^{n} \sum_{j=1}^{n} (P_i - P_j) \times (P_j - P_k),$$

where $T = \sum_{i=1}^{n} \sum_{j=1}^{n} 1$. Assuming $n = (n_x, n_y, n_z)$, then partial derivatives of the surface normal to $n$ were taken to be the partial derivatives of the surface at $P_0$. Or in other words,

$$\frac{\partial f}{\partial x} = -\frac{n_y}{n_z}$$

$$\frac{\partial f}{\partial y} = -\frac{n_z}{n_x}$$

A procedure similar to that of Akima was used by Kluczewicz. Kluczewicz passed a plane through points $P_j, P_k, P_l$ for all possible combinations of $P_j$ and $P_k$. Let such a plane be represented by $A_{xj} x + B_{xj} y + C_{xj} x + D_{xj} = 0$. Then an average plane was defined as being the plane whose parameters were the average parameters of planes passing through $P_j, P_k, P_l$ for all variations of $i$ and $j$. Assuming the equations of the average plane is $A_{xj} x + B_{xj} y + C_{xj} x + D_{xj} = 0$, then

$$A_{xj} = \frac{1}{T} \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij}$$

$$B_{xj} = \frac{1}{T} \sum_{i=1}^{n} \sum_{j=1}^{n} B_{ij}$$

$$C_{xj} = \frac{1}{T} \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij}$$

$$D_{xj} = \frac{1}{T} \sum_{i=1}^{n} \sum_{j=1}^{n} D_{ij}$$

where again $T = \sum_{i=1}^{n} \sum_{j=1}^{n} 1$. Then partial derivatives of the average plane was again taken to be the partial derivatives of the surface at $P_0$. Or,

$$\frac{\partial f}{\partial x} = -\frac{A_{xj}}{C_{xj}}$$

$$\frac{\partial f}{\partial y} = -\frac{B_{xj}}{C_{xj}}$$

In this paper, the weighted quadratic least-squares has been used to estimate the partial derivatives. For an experimental comparison of different techniques for estimation of the partial derivatives, see the paper by Stead.

3. RESULTS

To show the practicality of the proposed technique on registration of images with local geometric differences, two rubber sheet images as shown in Fig. 3 were used. On a rubber sheet that had nonuniform thickness we wrote "Rubber Sheet Images". A picture was taken from the sheet as shown in Fig. 3(a). This is our reference image. Then the rubber sheet was stretched in different directions with different forces on the boundary of the sheet and another picture was

(a)

(b)

Fig. 3. Two rubber sheet images. (a) Image of the rubber sheet before being stretched. (b) Image of the rubber sheet after being stretched with different forces applied to the boundary of the rubber sheet in different directions. (a) and (b) are of size 256 x 256 and 290 x 280, respectively.
Fig. 4. (a) The positions of the 32 control points selected in the reference image. (b) Triangulation of the control points.
Piecewise cubic mapping functions

Fig. 5. (a) and (b) are, correspondingly, the X-component and the Y-component mapping functions. Ribbons of constant intensity in (a) and (b) show how a row and a column of the sensed image are mapped into the reference image, respectively.

(a) (b)

taken as shown in Fig. 3(b). This is our sensed image. The amount of expansion made to the rubber sheet at a given point depends on the thickness of the sheet and also on the forces applied to the boundary of the sheet. The thickness of the rubber sheet changes from point to point and is a local factor. Forces applied to the boundary of the sheet, although affecting the whole sheet but the effect is more on nearby points than on the far away ones. The obtained images, therefore, have local geometric differences.

Thirty-two feature points were selected among the line end points and line intersections in the images. Line end points and line intersections do not change under nonlinear geometric transformations. Figure 4(a) shows these control points in the reference image. A triangulation was made from the control points as shown in Fig. 4(b). Then to determine the X-component mapping function we used measurements \((x_i, y_i, X_i)\), where \((X, Y, x, y)\) are the coordinates of the \(i\)th corresponding control points, in the procedure of Section 2. The obtained X-component mapping function is shown in Fig. 5(a). Each ribbon of constant intensity shows how a row of the sensed image is mapped into the reference image. By looking at the mapping function in this form it is possible to tell the position and amount of geometric difference between the images vertically. Similarly, using measurements \((x_i, y_i, Y_i)\), from the corresponding control points in the images, the Y-component mapping function is obtained which shows the horizontal geometric difference between the images, see Fig. 5(b). From these images we can conclude that geometric difference between the images is more severe vertically than it is horizontally.

Using the obtained mapping functions, we re-sampled the sensed image by the nearest neighbor rule. This is shown in Fig. 6(a). Then overlaid the resampled sensed image with the reference image as shown in Fig. 6(b). There is no quantitative measure we can make to determine the goodness of the registration since this registration technique overlays corresponding control points exactly on top of each other. However, by observation from image of Fig. 6(b) we can see that a reasonably good registration is obtained. To detect any misregistrations, the resampled sensed image was subtracted from the reference image. The result is shown in Fig. 6(c) where hardly visibly, we find slight difference between the images due to sampling error and misregistration in the order of fractions of a pixel.

The computation time required to determine the mapping functions involves times for triangulating the control points, estimating the partial derivatives, and solving a system of thirty equations for each triangle. Assuming there are \(n\) control points in each image, time required to triangulate the control points is \(\text{log}_n\). Determination of the partial derivatives requires solution of two systems of six equations each. Assuming there are \(n\) control points on the boundary of the convex hull of the control points, we get \(N = 2(n - 1) = n_0\) triangles, and to fit a cubic patch to each triangle we need to solve a system of thirty equations. Our program on a VAX-11/750 required about eighteen minutes processing time to determine both component mapping functions.

4. Conclusion

Images that are taken from exactly the same viewpoint of a scene overlay exactly and need not be registered. Otherwise, the images will have nonlinear geometric differences and depending on the 3-D structure of the scene, nonlinearity of the sensor, viewpoints, and atmospheric turbulence, the geometric difference between some areas of the images could be quite severe. In such case, global polynomial mapping functions are inappropriate for image
rubber sheet images

Fig. 6. Overlaying of the reference and sensed images. (a) The sensed image was resampled to register with the reference image. (b) The resampled sensed image is overlaid with the reference image. (c) The resampled sensed image is subtracted from the reference image.

registration.

In this paper, a mechanism which uses piecewise cubic mapping functions to register images with local geometric differences was described. A mapping function was determined as a piecewise combination of many local mapping functions, each characterizing local geometric difference between the images. A local mapping function is determined by using only local information (nearby control points). The mapping functions obtained in this form are, therefore, only locally sensitive and do not influence registration of distant areas of the images.

The proposed technique was applied on registration of two rubber sheet images with local geometric differences and encouraging results are obtained. The mapping functions determined in this form allow comparison of geometric structure of the images and can show the amounts and positions of geometric difference between the images.

SUMMARY

This paper is an extension of piecewise linear mapping functions for image registration. Here, image registration has been approached as a surface fitting problem. Each component mapping function is modeled as a surface that passes through 3-D points obtained by control points in the reference image and the x- or y-component of corresponding control points in the sensed image. A composite surface with triangular cubic patches has been used to represent a mapping function. Each cubic patch is obtained using coordinates of corresponding control points in a local neighborhood in the images. The overall mapping function is, therefore, piecewise cubic and represents local geometric difference between the images.

REFERENCES


About the Author: Arunesh Goshtasby received the B.E. degree in Electronics Engineering from the University of Tokyo, Japan, in 1974, the M.S. degree in Computer Science from the University of Kentucky in 1975, and the Ph.D. degree also in Computer Science from Michigan State University in 1983.

Dr. Goshtasby joined the staff of the Computer Science Department, University of Kentucky in 1983 where he is presently an Assistant Professor of Computer Science. His main areas of research are Computer Vision, Image Processing, Computer Graphics, and Remote Sensing. He is the co-recipient of the William Fisher Memorial Award for the best contributed paper in the 17th International Symposium on Remote Sensing of Environment, 1983.

Dr. Goshtasby is a member of the Pattern Recognition Society, the IEEE Computer Society, and the Association of Computing Machinery.