LARS Information Note 103174

NASA CR-144397

SPLINE FUNCTION APPROXIMATION TECHNIQUES FOR IMAGE GEOMETRIC DISTORTION REPRESENTATION

BY

PAUL E. ANUTA

The Laboratory for Applications of Remote Sensing

Purdue University, West Lafayette, Indiana

1974

LARS Information Note: 103174

Spline Function Approximation Techniques for Image Geometric Distortion Representation

by

Paul E. Anuta
Laboratory for Applications of Remote Sensing
Purdue University
West Lafayette, Indiana

Introduction

A considerable amount of interest has recently developed in the correction of spatial image distortions for registration of multitemporal remote sensor imagery [1,2,3]. The problem arises in the case where a scene is imaged at two or more times under varying sensor states. It is desired that the multiple images be geometrically registered so that when converted to digital form they can be analyzed as a multidimensional image vector using computer techniques.

One technique for registering two images is to find corresponding points in the two images and use these point pairs to distort one image to match the other. This problem has two characteristics which make specification of the correction function difficult. The first is the fact that the distortion of one image with respect to another is highly sensor dependent. Certain sensors introduce a great deal of random spatial distortion while others are highly stable. An example of a distortion producing sensor is the multispectral line scanner carried by a low altitude aircraft. Pitch, yaw, and translational movements of the aircraft

---

The work described in this report was sponsored by the National Aeronautics and Space Administration (NASA) under Grant Number NGL-15-005-112 and Contract No. NAS9-14016.

introduce corresponding distortions in the imagery. A highly stable sensor example is a multispectral scanner or camera carried in satellite orbit about the earth. Only slight distortions are introduced in instruments of this type.

The second key problem is that identification of matching points or checkpoints, as they will be called, is highly data or scene dependent. Matching points cannot be found at regular intervals either visually or by correlation by using scene context. Road intersections or correlated scene features tend to be found at random over an area. In images gathered at widely separated epochs the scene may have undergone such drastic change that very few matching points can be found. This problem does not exist if tick marks or reseau grids, as they are properly called, are imaged in coincidence with the scene. This is possible for image forming sensors such as cameras or vidicons but not for scanners.

The problem addressed in this paper is that of determining the optimum two dimensional approximation function for image distortions when the data or checkpoints are unequally spaced. Although the distortion function is a two dimensional function of two dimensions, it can be separated into two independent one dimensional functions of two dimensions.

The general statement of the problem is;

Find:

$$\Delta_x(x,y) = F_x(c,p,x,y)$$
$$\Delta_y(x,y) = F_y(c,p,x,y) \qquad (1)$$

such that the distortion functions Δ_x, Δ_y are as "close" as possible to the true distortions of the subject image. Definition of the functional form and a meaning of "close" are two problems of equal importance. In the above:

$\Delta_{x,y}(x,y)$ are the estimated distortion values in two dimensions

$F_{x,y}$ are the approximating functions

c checkpoints

p parameters defining F

A formal statement of the problem requires assumption of some functional form for the true distortion over the two dimensional space f(x,y) considered and some form for the error ρ. Then the problem can be stated:

> Let f(x,y) be a real valued continuous function of two variables on a set R, and let F(A,x,y) be a real valued approximating function depending continuously on $x,y \in R$ and on parameter A. Given the error function ρ, determine the parameters $A^* \in Q$ such that
>
> $$\rho[F(A^*,x,y),\ f(x,y)] \leq \rho[F(A,x),\ f(x)] \qquad (2)$$
>
> for all $A \in Q$, where Q is the set of all possible parameter sets.

The choice of an approximating function is difficult since no explicit method exists for making such a choice. Only the general statement that the more complex the variations in the function are the more complex the approximating function must be can be made with certainty. The choice of error function is also

equally undefined. The choice of error measure is often based on generally favorable characteristics of certain well known functions. The error or distance function is commonly called a norm and a common class of norms which will be considered here is called the L_p norm. The L_p norm of the function $f(x)$ is denoted by $L_p(f)$ and is defined by:

$$L_p(f) = \left| \int_a^b |f(x)|^p dx \right|^{1/p} \qquad p \geq 1 \tag{3}$$

Then the best approximation to $f(x)$ is obtained when the L_p distance function is minimized:

$$\min_A \int_a^b |F(A,x,y) - f(x,y)|^p \, dx \tag{4}$$

The solution A^* for $p = p_1$ will in general be different for $p = p_2$ and the nature of the approximation varies sharply as p is varied. Some well known cases are for $p = 1$ which is the minimum sum of the absolute value norm and $p = 2$ the least squares norm. A third widely used norm is called the Tchebycheff norm which is simply the maximum error. An optimum approximation in the Tchebycheff norm minimizes the maximum error. The L_2 or least squares norm is generally preferred above all others because of its desirable heavier weighting of large errors more than small errors and because of its differentiability and its relationship to series of orthogonal functions. The L_2 norm will

be used in the work described here.

The form of the approximating function remains as the key problem in the study. The functions investigated in this study are from two classes: 1. Polynomials, and 2. Spline Functions. A large body of information exists on polynomial approximations and the work reported here is based largely on Ralston [4] and Rice [5]. Spline functions or piecewise continuous polynomials have received relatively little attention until the early 1960's and the work here is based primarily on Rice [6] and deBoor [7]. Polynomials are studied first and are used to define image distortion over the entire two-dimensional image space. As distortions became more severe the order of the approximating polynomial becomes high and solution problems become severe. Spline functions are investigated secondly to determine if lower order polynomials fitted together can approximate a higher order distortion with less computational difficulty. Theory for the one dimensional case is first developed and then the two-dimensional theory is developed. Algorithms for generating approximating functions are described and application of the techniques to description of image distortion in aircraft multispectral scanner imagery is described. Comparison of results and recommendations are presented in conclusion.

One-Dimensional Polynomial Approximation

Approximation of a function defined by a discrete set of points $\{f_i | i=1,\ldots,n\}$ defined at points x_i using a set of polynomials of largest order m $\{\phi_j(x) | j=0,\ldots,m\}$ is expressed as:

$$F(A,x) = \sum_{j=0}^{m} a_j \phi_j(x) \qquad (5)$$

The least squares approximation to $\{f_i\}$ is defined by the set of coefficients A* such that

$$\varepsilon_A = \sum_{i=1}^{n} (f_i - F(A,x))^2 \quad \text{is a minimum.} \qquad (6)$$

The error can be minimized in the L_2 norm using differentiation since the L_2 norm has the desireable property of being analytically differentiable. Taking the derivative of ε_a with respect to each coefficient and setting the result to zero gives:

$$\frac{\partial \varepsilon_A}{\partial a_k} = -2 \sum_{i=1}^{n} (f_i - \sum_{j=0}^{m} a_j \phi_j(x_i)) \phi_k(x_i) = 0 \qquad (7)$$

A unique solution A* is guaranteed since the L_2 norm is a strictly convex function of A. This expression creates what is known as the normal equations for the least squares approximation. Another advantage of the L_2 norm is that the resulting normal equations are linear in the parameter space A whereas higher order norms result in quadratic, cubic, etc. equations in A. The equations can be expressed as:

$$\sum_{j=0}^{m} g_{jk}a_j = P_k \qquad k=0,\ldots m \tag{8}$$

where:
$$g_{jk} = \sum_{i=1}^{n} \phi_j(x_i)\phi_k(x_i) \qquad j,k=0,\ldots,m \tag{8a}$$

and
$$P_k = \sum_{i=1}^{n} f_i \phi_k(x_i) \tag{8b}$$

In matrix form notation this formulation is expressed as:

$$G=\{g'_{ij}\}=\{\phi_j(x_i)\} \tag{9}$$

(g'_{ij} denotes the elements of G not g_{jk} in equation 8a) Then the summation for g_{jk} in terms of the matrix G is G^TG [8]. The summation for P_k is expressed as G^Tf, where f is a column vector $\{f_i\}$. Then the expression for the a_j becomes:

$$G^TGA=G^Tf \tag{10}$$

Where A is a column vector of the desired coefficients $\{a_j\}$. The solution for A is obtained by solving the above matrix equation for A

$$A=(G^TG)^{-1}G^Tf \tag{11}$$

The elements of G are the basis functions $\phi_j(x_i)$ evaluated at the ordinates $\{x_i\}$ and are independent of f. The elements of G are, however, highly dependent on the form of ϕ. If $\phi_j(x_i)$ is chosen to be x_i^j then the exponent of the abscissa grows as the square of the order of the approximation. The matrix G^TG becomes highly ill-conditioned for values of m larger than 5 or 6 and this makes inversion difficult due to round-off error in the computational process. To illustrate this, assume all values of x_i lie in the interval 0-1. The terms in G^TG are for $\phi_j=x^j$:

$$g_{jk} = \sum_{i=1}^{n} x_i^{j+k} \tag{12}$$

which is approximately n times a Riemann sum

$$g_{jk} = n\int_0^1 x^{j+k}dx = \frac{n}{j+k+1} \quad j,k=0,\ldots,m \tag{13}$$

Which in matrix form is: nH where:

$$H = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{m+1} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & & \\ \vdots & & & & \\ \frac{1}{m+1} & & & & \frac{1}{2m+1} \end{bmatrix}$$

The matrix H is the principal minor of the infinite Hilbert matrix and is a classical example of an ill-conditioned matrix. An ill-conditoned matrix is one which when its largest value is 1 has an inverse with very large values. Thus use of the basis functions $\phi_j = x^j$ should be restricted to approximations of low order.

Note that the elements of G^TG are cross or inner products of the space of basis functions. If the ϕ_j are chosen such that

$$\sum_{i=1}^{n} \phi_j(x_i)\phi_k(x_i) = \begin{cases} 0 & \text{for } j \neq k \text{ and} \\ \neq 0 & \text{for } j=k \end{cases} \tag{14}$$

then the off diagonal elements of G^TG are zero and the matrix is easily invertible. Polynomials with this property are called orthogonal and are of extreme importance in the theory of approximation. Furthermore if

$$\sum_{i=1}^{n} \phi_j(x_i)\phi_k(x_i) = 1 \quad \text{for } j=k \tag{15}$$

The ϕ_j are called orthonormal and the matrix G^TG becomes the identity matrix and the inversion problem vanishes. The approximation problem becomes

$$IA = G^T f \tag{16}$$

which is a simple matrix multiplication. The elements of A are

$$a_j = \sum_{i=1}^{n} f_i \phi_j(x_i)$$

A problem arises in that for finite point sets the orthonormal polynomials depend on the number and spacing of the points x_i. For cases where the points are equally spaced the polynomials can be defined parametrically in the number of points n and the order of the polynomial m. These polynomials are known as the Gram polynomials and can be found tabulated and their derivation is unnecessary. If, however, the points are unequally spaced as is the case for the checkpoints defining the image distortion, the Gram polynomials cannot be used. In this case, a recursive formula must be used to generate the polynomials. It can be shown by induction (see Ref 4, Page 241) that the following relationship generates the orthogonal polynomials:

$$\phi_{j+1}(x) = (x - C_{j+1})\phi_j(x) - d_j\phi_{j-1}(x) \tag{17}$$

Where:

$$P_o(x)=1$$

$$P_{-1}(x)=0$$

and C_j, d_j are to be determined. The coefficients are

$$d_k = \sum_{i=1}^{n} [\phi_k(x_i)]^2 / \sum_{i=1}^{n} [\phi_{k-1}(x_i)]^2$$

and

$$C_{k+1} = \sum_{i=1}^{n} x_i [\phi_k(x_i)]^2 / \sum_{i=1}^{n} [P_k(x_i)]^2$$

These polynomials are orthogonal but not normal, thus a normalizing factor is required to compute the coefficients for the approximating function.

$$F(A,x) = \sum_{j=0}^{m} a_j \phi_j(x) \qquad (18)$$

$$a_j = \frac{1}{\gamma_i} \sum_{i=1}^{n} f_i \phi_j(x_i)$$

And the normalizing factor γ_i is

$$\gamma_i = \sum_{i=1}^{n} (\phi_j(x_i))^2$$

These are in fact the diagonal elements of the matrix G^TG. If the order of the approximating polynomial is low (4 or 5) the non-orthogonal normal equation method is probably preferable since considerable labor is involved in getting the ϕ_j. For higher order functions the orthogonal polynomial method will be necessary. The two-dimensional polynomial case will be discussed next and then spline functions will be introduced.

Two Dimensional Polynomial Approximation

Two dimensional approximation is a direct extension of the one dimensional case for the L_2 norm. The chief problem lies in the size of the problem in that the number of terms increases as the square of the order of the approximation.

The two dimensional image distortions Δx and Δy must be estimated for every element in the picture from a limited number of unequally spaced points at which the distortion is known. The functional approximation problem is solved here by computing a least squares polynomial approximation to the given points using the normal equations. Given are a set of displacement values distributed over the two dimensional image space. Let $f_x(x_i,y_i)$, $f_y(x_i,y_i)$, $i=1,\ldots n$ be the true image displacements at the point x_i,y_i. The displacement over the entire image is to be approximated by a polynomial function based on the n measured values:

$$\Delta x(x,y) = \sum_{j=0}^{m} \sum_{k=0}^{m} a_{jk}\phi_{jk}(x,y)$$

$$\Delta_y(x,y) = \sum_{j=0}^{m} \sum_{k=0}^{m} b_{jk}\phi_{jk}(x,y) \qquad (19)$$

For approximations of low order the nonorthogonal monomial basis function x^j can be used and the normal

equations solved for the desired coefficients. The approximating functions in this case become:

$$\Delta x(x,y) = a_{00} + a_{10}x + a_{01}y + a_{20}x^2 \ldots$$

$$= \sum_{j=0}^{m} \sum_{k=0}^{m} a_{jk} x^j y^k \qquad (20)$$

$$\Delta y(x,y) = b_{00} + b_{10}x + b_{01}y + b_{20}x^2$$

$$= \sum_{j=0}^{m} \sum_{k=0}^{m} b_{jk} x^j y^k$$

where: $\Delta x,y$ are the approximated values

$\{a,b\}$ are parameters to be determined

The elements of the equation matrix G become:

$$g_{i\ell} = x_i^j y_i^k \qquad i=1,\ldots n$$

$$\ell=1,\ldots,(m+1)^2$$

$$j=0,\ldots,m$$

$$k=0,\ldots,m$$

And the normal equations are again generated by

$$G^T GA = G^T f_x$$

$$G^T GB = G^T f_y \qquad (21)$$

where: A,B are column vectors of the desired coefficients a_{jk}, b_{jk}

f_x, f_y are column vectors of the image distortions in the x and y directions.

The solution for the coefficients of the least squares functions are

$$A = (G^T G)^{-1} G^T f_x \qquad (22)$$

$$B = (G^T G)^{-1} G^T f_y$$

The ordering of the powers of x and y in the polynomial and the elimination of certain terms are two items of variation in procedure. The coefficient vector is actually a matrix of terms

$$A^T = \begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0m} \\ a_{10} & a_{11} & & \\ & & & \\ a_{m0} & \cdots & & a_{mm} \end{bmatrix}$$

The subscript notation is chosen so that the first subscript is the power of the x term and the second is the power of y. And the set of polynomials is also a matrix which in the case of $\phi_j(x) = x^j$ is

$$\Phi = \begin{bmatrix} 1 & x & x^2 & \cdots & x^m \\ y & xy & x^2y^2 & \cdots & x^m y^2 \\ y^2 & & & & \\ y^m & \cdots & & & x^m y^m \end{bmatrix}$$

and the approximating function is expressed as

$$\Delta x(x,y) = \Phi(x,y) \otimes A$$

$$\Delta y(x,y) = \Phi(x,y) \otimes B \qquad (23)$$

Where the $\otimes$ indicates the matrix dot product:

$$A \otimes B = \sum_{i=1}^{m} \sum_{j=1}^{m} a_{ij} \cdot b_{ij}.$$ The normal equations are solved in the same way as for the single dimensional case except two coefficient sets are obtained, one for Δx and the other for Δy.

Two Dimensional Orthogonal Polynomial Approximations

The generation of two dimensional orthogonal polynomials is again a straight-forward extension of the development for the one dimensional case. For the case of unequally spaced points, a recursive relationship can be used to generate the polynomials. Polynomial ϕ_j can be obtained as a function of lower order polynomials:

$$\phi_{j+1}(x,y) = (x+y-d_{j+1})\phi_j(x,y) - B_j\phi_{j-1}(x,y) \qquad (24)$$

where $\phi_1(x,y)=0$ and $\phi_0(x,y)=1$

we require that

$$\sum_{i=1}^{n} \phi_j(x_i,y_i)\phi_{k+1}(x_i,y_i)=0 \text{ for } j=0,1,\ldots,k \qquad (25)$$

Substituting the recursive expression into this requirement for j=k gives:

$$\sum_{i=1}^{n} \phi_j(x_i,y_i)x_i\phi_k(x_i,y_i) - \sum_{i=1}^{n} \phi_j(x_i,y_i)y\,\phi_k(x_i,y_i)$$
$$-\alpha_{k+1}\sum_{i=1}^{n} \phi_j(x_i,y_i)\phi_k(x_i,y_i)-\beta_k \sum_{i=1}^{n} \phi_j(x_i,y_i)\phi_{k-1}(x_i,y_i)=0$$
$$\text{for } j=0,1,\ldots k \qquad (26)$$

A double summation over (x_i,y_j) is not used because each coordinate pair is unique due to the random distribution of x_i and y_i thus the single summation takes into account all possible points in the two dimensional space.

For j=0,1,...k-2 in equation 26 the rightmost two terms are zero because the orthogonality condition holds for polynomials up to the one being generated.

The first two summations with x_i and y_i in the summands are polynomials of degree not greater than k-1 and can be expressed as a linear combination of the $\phi_j(x,y)$ and thus these terms will be zero also. For the case j=k-1 the third term is still zero but the fourth is non-zero. The first two terms are non-zero because the order of $x_i\phi_{k-1}(x_i,y_i)$ is k and product summation with $\phi_k(x,y)$ will be non-zero. Thus a condition for β is created:

$$\beta_k = \frac{\sum_{i=1}^{n} \phi_{k-1}(x_i,y_i)x_i\phi_k(x_i,y_i)+\sum_{i=1}^{n}\phi_{k-1}(x_i,y_i)y_i\phi_k(x_i,y_i)}{\sum_{i=1}^{n}[\phi_{k-1}(x_i,y_i)]^2} \qquad (27)$$

For j=k the fourth term is zero and the third term is non-zero. This gives an expression for α:

$$\alpha_{k+1} = \frac{\sum_{i=1}^{n} x_i [\phi_k(x_i,y_i)]^2 + \sum_{i=1}^{n} y_i [\phi_k(x_i,y_i)]^2}{\sum_{i=1}^{n} [\phi_k(x_i,y_i)]^2} \qquad (28)$$

Thus α, can be found which generates a sequence of orthogonal polynomials. Given that $\phi_1 = 0$, $\phi_0 = 1$

$$\phi_1(x,y) = (x+y-\alpha_1).$$

It is required that

$$\sum_{i=1}^{n} \phi_0(x,y)\phi_1(x,y) = 0$$

$$\text{or} \quad \sum_{i=1}^{n} (x_i+y_i-\alpha_1) = 0$$

$$\text{Thus} \quad \alpha_1 = \frac{1}{n} \sum_{i=1}^{n} (x_i+y_i)$$

From this ϕ_0 and ϕ_1 are orthogonal and ϕ_2 is generated from these two polynomials known to be orthogonal thus the assumptions used in deriving α and β will hold. The generation of the approximating function from the ϕ_j is identical to the case for one dimension.

$$\Delta x(x,y) = \sum_{j=0}^{m} a_j \phi_j(x,y)$$

$$\Delta Y(x,y) = \sum_{j=0}^{m} b_j \phi_j(x,y) \qquad (29)$$

where: $a_j = \sum_{i=1}^{n} x_i \phi_j(x_i,y_i) \Big/ \sum_{i=1}^{n} \phi_j(x_i,y_i)$, similarly for b_j

Use of Tensor Products for Two Dimensional Approximations

For a coordinate system in which the independent coordinates of the given data form a Cartesian product set $X \otimes Y$ the following tensor product of functions is defined:

The tensor product of two sets of functions $\{\phi_i | i = 1,2,\ldots,p\}$ and $\{\psi_i | i = 1,2,\ldots q\}$ is the set $\{\phi_i \psi_j | 1 \leq i \leq p,\ 1 \leq j \leq q\}$. This product is stated as $\{\phi_i\} \otimes \{\psi_j\}$.

The linear approximating function formed by such a product is:

$$F(A,x,y) = \sum_{i=1}^{m} \sum_{j=1}^{m} a_i b_j \phi_i(x) \psi_j(y) \qquad (30)$$

Where: A represents the coefficients a,b

m is the order of the function

It can be shown that if the functions $\{\phi_i | i=1,\ldots,m\}$ form an orthonormal set then the tensor product is an orthonormal set. Then the best L_2 approximation (least squares) can be found by solving the normal equations using the tensor product form of function.

The tensor product approach eliminates the need for constructing the two dimensional orthogonal polynomials which becomes an extremely cumbersome task. The tensor approach is used in the spline function method to be discussed next. The spline functions contain as a subset the polynomial function approach discussed above; thus, the following sections cover both cases and represent the major thrust of this report.

Spline Function Approximation

The one and two-dimensional approximating functions discussed up to this point are assumed to be valid over the entire region of interest with adequate accuracy. For cases in which the order of the distortion is low and the area covered is limited, the single function approach is adequate. When the image distortions become severe and the area to be represented increases the order of the functions required becomes impractically high. Use of two-dimensional functions of fourth, fifth or higher order is undesirable for computational reasons and because the number of control points required becomes excessive. The bi-quadratic function is of the form:

$$\Delta x(x,y) = a_0 + a_1 x + a_2 x + a_3 xy + a_4 x^2 + a_5 y^2 + a_6 x^2 y + a_7 xy^2 + a_8 x^2 y^2 \qquad (31)$$

and requires nine coefficients. The bi-cubic function requires 16 coefficients. Representation of higher order curves or surfaces can be achieved through use of lower order polynomials

joined together in a piecewise continuous manner. Such approximations are called spline functions [6] and constitute a class of extremely useful and successful functions which were first considered from a mathematical viewpoint by Schoenberg in 1946 [9] and research on these functions has increased steadily since. The great value of the piecewise polynomial approach is that complex disjointed functions arising from real, physical situations can be approximated rather conveniently. The random variations in multispectral scanner image geometry are generally unrelated from one place to another. Whereas for polynomials and most other mathematical functions their behavior in a small region determines their behavior everywhere. Piecewise continuous (spline) functions do not have this problem and for cubic or higher order polynomial splines the splines are smooth curves in the physical world.

A one-dimensional spline function is defined by a set of points called knots ξ_i:

$$a \leq \xi_0 < \xi_1 < \ldots < \xi_k < \xi_{k+1} = b$$

over the interval [a,b] and a set of polynomials of degree n valid between the knots and having n-1 continuous derivatives at the knots. One representation as presented by Rice [6] of splines is of the form:

$$S(A,\Xi,x) = \sum_{i=1}^{k} a_i (x-\xi_i)_+^n + \Pi(x) \qquad (32)$$

Where:

$$(x-\xi)_+^n = \begin{cases} (x-\xi)^n & x \geq \xi \\ 0 & x < \xi \end{cases}$$

$\Pi(x)$ = Polynomial of degree n with coefficients $a_i, i=k+1, \ldots k+n+1$

Λ = Parameter Vector $(a_1, a_2, \ldots a_{k+n+1})$

Ξ = Set of knots $(\xi_0, \xi_1, \ldots \xi_{k+1})$

Many other forms of representation exist for splines; however, the resulting functions are the same. Splines of greater than third order are generally not used, the advantage of the spline approach being that high order polynomials can be avoided. A first order spline would be a sequence of linear or ramp functions joined together at the knots forming the familiar piecewise linear functions. A second order spline would be a set of quadratic polynomials connected at the knots and having continuous first derivatives at the knots. Similarly, a third order spline would have cubic polynomials joined at the knots and having continuous first and second derivatives.

A more common representation for splines is:

$$S(\Lambda,\Xi,x) = \sum_{j=0}^{n} C_{ij} (x-\xi_{i-1})^j \qquad \xi_{i-1} \leq x \leq \xi_i \tag{33}$$

which illustrates the piecewise polynomial nature more explicitly. Also, when n is odd the splines can be uniquely represented in terms of the values and derivatives at the knots. Thus, for linear and cubic splines, the approximation can be completely specified by the values:

$$V_{ij} = \left.\frac{d^j S(A,\Xi,x)}{dx^j}\right|_{x=\xi_i} \qquad i=1, \ldots, k+1, j=0,1, \ldots \frac{n-1}{2}$$

The 1st and 3rd (odd) order splines turn out to be extremely convenient choices since for even functions the number of derivatives needed to be specified at the left and right knot is not equal and the spline cannot be uniquely specified by knot point values and derivatives. Furthermore, since the linear splines have n-1=0 continuous derivatives at the knots the spline function has jumps in slope at the knots and do not form "smooth" curves. Thus, the cubic spline becomes the natural or preferred order for spline approximating functions. The cubic spline in one dimension is of the form:

$$S(A,\Xi,x) = C_{i1}+C_{i2}(x-\xi_{i-1})+C_{i3}(x-\xi_{i-1})^2+C_{i4}(x-\xi_{i-1})^3$$
$$\text{for } \xi_{i-1} \leq x \leq \xi_i \qquad (34)$$

Note that four coefficients are required for each polynomial

which can thus be specified by four knot point values:

$$V_{i-1\ 0} = S(A,\Xi,\xi_{i-1}) \qquad V_{i,0} = S(A,\Xi,\xi_i)$$

$$V_{i-1,1} = \left.\frac{dS(A,\Xi,x)}{dx}\right|_{x=\xi_{i-1}} \qquad V_{i,1} = \left.\frac{dS(A,\Xi,x)}{dx}\right|_{x=\xi_i}$$

It can also be shown [10] that the entire cubic spline function is uniquely specified by only the function values at the knots and derivatives only at the end point knots.

Two Dimensional Spline Functions

The discussion of splines thus far has been in the context of one-dimensional functions. Second and higher dimensional splines are more complex analogs of the one dimensional case. The two-dimensional case will be discussed in terms of cubic, or more precisely, bi-cubic polynomials since they have the same advantages that the one-dimensional cubic splines have. It can also be proven that specification of 16 corner values and derivatives uniquely specifies the spline function polynomials. The two dimensional splines are defined on a rectangular grid in a two-dimensional plane. Let the divisions between the rectangles or knots on the X axis be defined by the set $\{\xi_i\}$ and the knots on the Y axis by the set $[\mu_i]$. Then the 2-dimen-

sional spline polynomials are of the form:

$$\Delta X(x,y) = \sum_{m=0}^{3} \sum_{n=0}^{3} a_{mn}^{ij} (x-\xi_{i-1})^m (y-\mu_{j-1})^n \qquad (35)$$

For $\xi_{i-1} \leq x \leq \xi_i$, $\mu_{j-1} \leq y \leq \mu_j$ $i=1,\ldots,K$, $j=1,\ldots,L$

For each polynomial there are 16 defining coefficients. A spline grid is depicted in Figure 1 for K=3 and L=3. The 16 corner conditions on each rectangle that specify the cubic polynomial follow: Let the spline function be represented as $S(x,y)$; then the corner values for rectangle (ξ_{i-1},μ_{j-1}), (ξ_i,μ_{j-1}), (ξ_{i-1},μ_j), (ξ_i,μ_j) are:

$S(\xi_{i-1},\mu_{j-1})$	$S(\xi_i,\mu_{i-1})$	$S(\xi_{i-1},\mu_j)$	$S(\xi_i,\mu_j)$
$\left.\frac{\partial S(x,y)}{\partial x}\right\|_{\substack{x=\xi_{i-1}\\y=\mu_{j-1}}}$	$\left.\frac{\partial S(x,y)}{\partial x}\right\|_{\substack{x=\xi_i\\y=\mu_{j-1}}}$	$\left.\frac{\partial S(x,y)}{\partial x}\right\|_{\substack{x=\xi_{i-1}\\y=\mu_j}}$	$\left.\frac{\partial S(x,y)}{\partial x}\right\|_{\substack{x=\xi_i\\y=\mu_j}}$
$\left.\frac{\partial S(x,y)}{\partial y}\right\|_{\substack{x=\xi_{i-1}\\y=\mu_{m-1}}}$	$\left.\frac{\partial S(x,y)}{\partial y}\right\|_{\substack{x=\xi_i\\y=\mu_{j-1}}}$	$\left.\frac{\partial S(x,y)}{\partial y}\right\|_{\substack{x=\xi_{i-1}\\y=\mu_j}}$	$\left.\frac{\partial S(x,y)}{\partial y}\right\|_{\substack{x=\xi_i\\y=\mu_j}}$
$\left.\frac{\partial S(x,y)}{\partial x\partial y}\right\|_{\substack{x=\xi_{i-1}\\y=\mu_{j-1}}}$	$\left.\frac{\partial S(x,y)}{\partial x\partial y}\right\|_{\substack{x=\xi_i\\y=\mu_{j-1}}}$	$\left.\frac{\partial S(x,y)}{\partial x\partial y}\right\|_{\substack{x=\xi_i\\y=\mu_j}}$	$\left.\frac{\partial S(x,y)}{\partial x\partial y}\right\|_{\substack{x=\xi_i\\y=\mu_j}}$

Fortunately the two dimensional analog of a previously stated result holds for the bi-cubic splines. That is that the polynomials are uniquely determined if only the value of the function is specified at each of the mesh points and the derivatives are specified only at the outside edges of the grid. Specifically, the piecewise bi-cubic function is uniquely specified by only the following values:

$$S(\xi_i,\mu_j) \qquad i=0,\ldots,L, \quad j=0,\ldots,M \quad ((L+1)\cdot(M+1) \text{ values})$$

$$\left.\frac{\partial S(x,y)}{\partial x}\right|_{\substack{x=\xi_i \\ y=\mu_j}} \qquad i=0,L, \quad j=0,1,\ldots,M \quad (2M \text{ values})$$

$$\left.\frac{\partial S(x,y)}{\partial y}\right|_{\substack{x=\xi_i \\ y=\mu_j}} \qquad i=0,1,\ldots L, \quad j=0,M \quad (2L \text{ values})$$

$$\left.\frac{\partial S(x,y)}{\partial y}\right|_{\substack{x=\xi_i \\ y=\mu_j}} \qquad i=0,L, \quad j=0,M \quad (4 \text{ values})$$

This elegant result will not be proven here, but the proof can be found in the cited report by de Boor [10]. For the nine rectangle mesh depicted in Figure 1 specification of 16 mesh function values, 16 edge derivatives and four cross corner derivatives or 36 total quantities completely specify

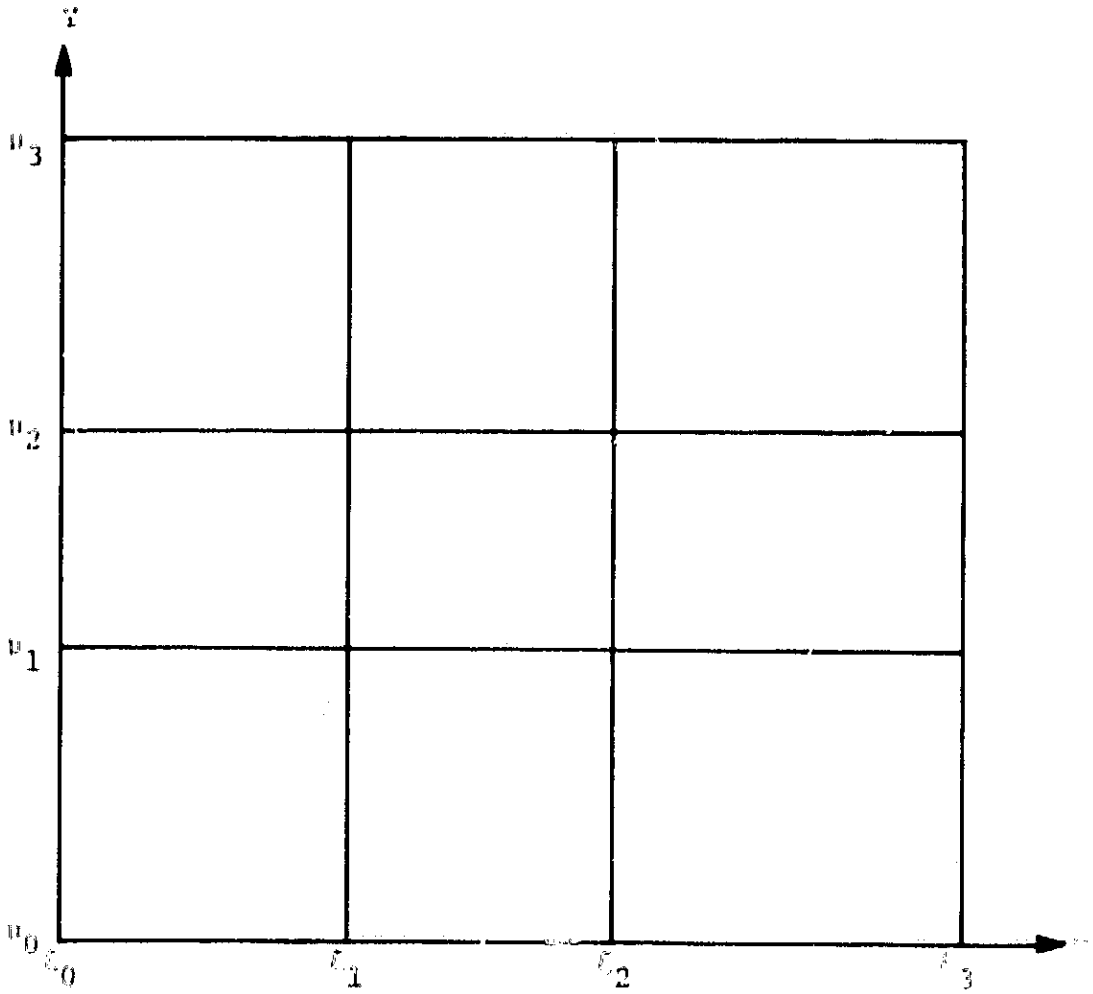

Figure 1. Example 4 by 4 mesh for spline function description.

the nine bi-cubic polynomials forming the spline function. The representation without the simplification would require the 16 values, 16 X derivatives, 16 Y derivatives, and 16 cross derivatives or 64 values.

Computation of the bi-cubic polynomial coefficients from the specified mesh point values can be accomplished through simultaneous equation solution also described in [10]. The method discussed assumes knowledge of the exact values and derivatives of the function at the grid nodes. In practice, these values are not known exactly and generally are a set of approximately known values and these are often unevenly spaced over the domain of the function. Thus, the real problem is computing an approximate two dimensional spline function based on this data. The least squares spline problem is thus the problem to be solved.

Least Squares Two-Dimensional Spline Approximation

The least squares solution for the two-dimensional spline approximation function is most advantageously computed using a set of orthogonal spline basis functions. Using this approach, once an orthogonal spline basis is obtained the approximation is easily computed using inner products. If $\{\psi_i\}$ is a complete orthonormal spline basis for the set of all spline functions

over the domain of interest, then

$$\langle \psi_i, \psi_j \rangle = \delta_{ij} \qquad i,j=0,\ldots N \tag{36}$$

Where: $\delta_{ij}=1$ if $i=j$ and zero otherwise.

The approximating spline function S is computed as

$$S = \sum_{i=0}^{n} \langle f, \psi_i \rangle \psi_i \tag{37}$$

If a non-orthogonal basis $\{\phi_i\}$ is known the orthonormal basis $\{\psi_i\}$ can be computed using a procedure similar to the one described for the one dimensional case. The Gram-Schmidt orthonormalization procedure is usually used for this purpose. This procedure is described by the two step iterative formula:

$$\begin{aligned} \psi_0 &= \phi_0 / \|\phi_0\| \\ \psi_i' &= \phi_i - \sum_{j=0}^{i-1} \langle \phi_i, \psi_j \rangle \psi_j \qquad i=0,\ldots N \\ \psi_i &= \psi_i' / \|\psi_i'\| \end{aligned} \tag{38}$$

By this process, the non-orthonormal basis $\{\phi_i\}$ is converted to the orthonormal spline basis function set $\{\psi_i\}$ which can be used directly for evaluating the approximating function coefficients. It is pointed out in [7] that in forming the initial basis $\{\phi_i\}$ it is advantageous to construct each ϕ_i so as to have one more extremum than ϕ_{i-1}. This tends to generate a "nearly" orthogonal initial basis which improves the accuracy of the resulting orthogonal basis.

The procedure for solving for the least squares bi-cubic approximating function consists of solving for the orthonormal spline basis functions for a given set of knots and then evaluating the approximating function and the mean squared (L_2) error. The knots can then be moved or increased in number to attempt to reduce the error. An algorithm for carrying out this process if presented in [7] for approximation in one independent variable. Development of a two-dimensional cubic analog for this process was carried out as part of this project. The algorithm facilitates geometric correction and registration of aircraft scanner data and similar data having almost any degree of distortion.

Two Dimensional Spline Function Approximation Algorithm

The algorithm developed is a generalization to two dimensions of the algorithm FXDKNT described in [7] by de Boor and Rice. The tensor product approach is used in generation of the spline basis functions rather than attempt to compute bi-variate basis functions. The algorithm was originally written to accomodate 100 data values to be approximated and up to 26 knots in addition to the left and right boundary knots, or a total of 28. In the two dimensional version the number of points was kept at 100 and the number of Y axis knots made the same as for X or 28. This greatly expands the size of the program but keeps the same capa-

bility in the new program on each axis should it be needed. In practice the number of knots on each axis for scanner imagery probably will not exceed five or six; however, in other applications the full power of the algorithm may prove useful.

The tensor product form of basis function generation results in two sets of orthonormal spline basis functions $\{\psi_i(x)\}$ and $\{\mu_j(y)\}$ from which the coefficients are obtained for the orthogonal projection of the function to be approximated onto these basis functions. The form of the approximating function is thus:

$$\Delta X(x,y) = \sum_{i=1}^{IX} \sum_{j=1}^{IY} a_{ij} \psi_i(x) \mu_j(y) \qquad (39)$$

Where ψ, μ are the spline basis functions

a_{ij} the coefficients of the approximation function in the basis $\psi_i(x)\mu_i(y)$

IX,IY are the number of X and Y dimension basis functions in the solution.

The a_{ij} are computed as the inner product of the function to be approximated and the basis functions:

$$a_{ij} = \langle f_x(x,y)\ \psi_i(x)\mu_j(y) \rangle \qquad (40)$$

Where

$\langle\ \rangle$ denotes the two dimensional inner product

The algorithm proceeds by first computing a single cubic polynomial approximating function over the entire set of points to be approximated. Then additional knots are introduced one by one and additional basis functions are computed. The coordinate of the new basis function is computed and the contribution of the new term is subtracted from the remaining error in the approximation.

The polynomial spline functions are computed from the basis functions and their coordinates by evaluating the value, derivatives in the x,y and cross directions at the corners of each of the rectangular segments of the domain being covered. These sixteen values are then used to define a cubic polynomial for each rectangular region. The sixteen values are then transformed into the 16 polynomial coefficients. The economizing procedure discussed previously is not used in the current algorithm. The resulting approximation is represented by the sixteen coefficients for each spline region for as many regions as were specified by the knot set. Thus, a function having four knots in the x direction and six knots in the y direction, including boundary knots, would have (4-1)x(6-1)x16 = 240 quantities specifying the approximating function plus the ten knot values.

The algorithm can be re-executed to add or delete knots to adjust the overall RMS error in the approximation to a desired level. An artifice was used in the computation of the two dimensional inner product to handle the case of randomly located

data points. The data points are constrained to lie on a quasi-rectangular grid and the means of the resulting groups are used as the x and y abscissa values in the inner product. A nearest neighbor rule is used to assign function values to the points at the nodes of the artificial uniform grid. This enables a simple trapezoidal integration inner product to be computed but causes error in the approximation. An iterative technique is then employed to correct for this error.

Example of Spline Function Approximation

The multispectral aircraft scanner system flown by the ERIM* organization produces imagery in long strips of nominally two miles in width at 5,000 feet altitude. This data is often subject to severe distortions due to pitch and yaw variations in the aircraft attitude and lateral motions due to cross winds since the scanner is fixed to the frame of the aircraft. The scanner is roll stabilized so that only pitch and yaw angular distortions are experienced. Thus, this type of imagery can be affected by five platform variables: pitch, yaw, and translational variations in three dimensions. An example of aircraft motion distortion in the MSS imagery is shown in Figure 2. More sophisticated scanners are stabilized on the pitch, roll, and yaw axes; however, this requires costly gimballing mounts and costly support control systems. In most scanner imagery cases, some degree of random distortion will be present and the spline function techniques are expected to be useful in a wide range of cases. The extreme flexibility of the spline approximating functions allows the case of using only one function for the entire image for simple distortion up to the case of many spline function regions covering the image to be handled with the same algorithm semi-automatically.

---

* Environmental Research Institute of Michigan

Figure 2. Aircraft Multispectral Scanner imagery (.58-.65μ band) showing severe geometric distortion due to crosswinds. LARS Run No. 71054100. Area is immediately west of Crawfordsville, Indiana. Date: August 17, 1971. Altitude: 5,000 ft.

Figure 3. Aerial photo of area in Figure 2 showing the desired image geometry.

The multispectral scanner imagery shown in Figure 2 is a typical example of ERIM low altitude flight data. The aerial photograph segment shown in Figure 3 covers the area imaged by the scanner and represents the desired geometric shape of image in Figure 2. The function required to transform the MSS imagery into the geometric form of the photo is specified by defining checkpoints or matching points in the image and map. These points can be obtained by a variety of manual or automatic methods and for this example they were obtained by measuring the coordinates in inches on the MSS image and photo using a coordinate digitizing table. The scale of the image is approximately 1:56300 and the scale of the photo is approximately the same. The coordinates of the checkpoints digitized from the imagery and photo are listed in Table 1.

The values from Table 1 were input to the two dimensional cubic spline algorithm first for the case of only one block. This results in a cubic polynomial fit over the entire region which in this example was for $.75 \leq x \leq 2.594$ and $.593 \leq y \leq 10.25$. The results of the approximation are listed in Table 2 which includes the values to be approximated (the x and y position of each conjugate point in the aircraft scanner image), the approximations, the error and three error statistics. The root of the mean squared error for the approximation is .063 for the x coordinate and .11 for y. The maximum error was .132 for x and .37 for y.

Table 1. Coordinates of Matching Points for Aircraft Scanner Data Correction Example. Purdue Flight Line 212. Scanner data obtained August 17, 1971. Aerial photograph made from Color IR photograph taken at 60,000 feet by NASA RB-57 in 1971.

Point No.	Photograph		Scanner Image	
	X	Y	X	Y
1	1.53	.594	1.06	.75
2	.781	.813	.063	.875
3	2.125	.593	1.875	.813
4	2.438	1.188	2.125	1.375
5	1.563	1.188	1.0	1.031
6	.750	1.188	.031	1.188
7	.969	2.188	.188	2.250
8	1.563	2.250	.875	2.250
9	2.250	2.656	1.875	2.625
10	1.031	4.093	.438	3.938
11	1.562	4.813	1.0	4.063
12	2.125	4.25	1.781	4.094
13	.969	5.375	.25	5.188
14	1.531	5.375	1.031	5.188
15	2.281	5.344	1.969	5.250
16	.968	6.50	.188	6.313
17	1.565	6.50	.938	6.313
18	2.438	6.50	2.0	6.375
19	.938	7.656	.125	7.469
20	1.531	7.688	.813	7.50
21	2.50	7.656	2.063	7.50
22	1.094	10.25	.156	10.031
23	1.625	9.969	.75	9.75
24	2.375	9.938	2.125	9.75
25	2.031	.562	1.75	.781
26	2.0	1.125	1.70	1.312
27	1.875	2.5	1.375	2.47
28	2.125	4.75	1.81	4.65
29	1.781	5.375	1.437	5.218
30	2.125	6.5	1.70	6.375
31	2.125	7.656	1.65	7.5
32	1.875	9.94	1.187	9.75

Table 2a Error data for x dimension approximation using one cubic spline block.

ROOT MEAN SQUARE ERROR = 0.625779E-01
AVERAGE ERROR = 0.524878E-01
MAXIMUM ERROR = 0.132832E 00 AT 1.780999 5.374999

APPROXIMATION AND SCALED ERROR CURVE

	DATA POINT		APPROXIMATION	DEVIATION X 10E+2
1	0.74999970	1.18799973	0.03031921	-0.068076
2	0.78099966	0.81299967	-0.06647825	-12.947815
3	0.93799961	7.65599918	0.13774538	1.274549
4	0.96799958	6.49999905	0.22627860	3.827875
5	0.96899956	2.18799973	0.26086825	7.286840
6	0.96899956	5.37499905	0.27540171	2.540183
7	1.03099918	4.09299946	0.33013874	-10.786104
8	1.09399986	10.24999905	0.09022141	-6.577849
9	1.52999973	0.59399968	0.94783473	-11.216474
10	1.53099918	5.37499905	0.91989422	-11.115497
11	1.53099918	7.68799877	0.86078262	4.778296
12	1.56199932	4.81299877	0.96592617	-3.407383
13	1.56299973	1.18799973	0.97424936	-2.575064
14	1.56299973	2.24999905	0.95527613	8.027655
15	1.56499958	6.49999905	0.95872116	2.072155
16	1.62499905	9.96899891	0.74205399	-6.794572
17	1.78099918	5.37499905	1.30416775	-13.283157
18	1.87499905	2.49999905	1.45851803	8.351898
19	1.87499905	9.93999958	1.14768982	-3.930950
20	2.00000000	1.12499905	1.64196968	-5.802917
21	2.03099918	0.56199974	1.68718338	-6.281567
22	2.12499905	4.74999905	1.77307796	-3.692150
23	2.12499905	6.49999905	1.73779392	3.779507
24	2.12499905	7.65599918	1.69654846	4.654884
25	2.12499905	0.59299970	1.81568146	-5.931759
26	2.12499905	4.24999905	1.77913666	-0.186253
27	2.24999905	2.65599918	1.90832806	3.332901
28	2.28099918	5.34399891	1.90369225	-6.530666
29	2.37499905	9.93799877	1.91258240	-2.541637
30	2.43799973	1.18799973	2.11465549	-1.034355
31	2.43799973	6.49999905	2.01200867	1.200867
32	2.49999905	7.65599918	2.09133911	2.833939

Table 2b Error data for y dimension approximation using one cubic spline block.

ROOT MEAN SQUARE ERROR = 0.110631E 00
AVERAGE ERROR = 0.764261E-01
MAXIMUM ERROR = 0.368560E 00 AT 1.561999 4.[illegible]

APPROXIMATION AND SCALED ERROR CURVE

	DATA POINT		APPROXIMATION	DEVIATION X 10E+2
1	0.74999970	1.18799973	1.32238770	13.438777
2	0.78099966	0.81299967	0.65502316	-21.997635
3	0.93799961	7.65599918	7.44279766	-2.620125
4	0.96799958	6.49999905	6.3228960[illegible]	[illegible]
5	0.96899956	2.18799973	2.1826248[illegible]	-6.737[illegible]
6	0.96899956	5.37499905	5.25064754	6.264762
7	1.03099918	4.09299946	3.88046741	-5.253136
8	1.09399886	10.24999905	10.0640220[illegible]	3.3[illegible]288
9	1.52999973	0.59399966	0.807[illegible]7961	5.[illegible]7971
10	1.53099918	5.37499905	5.00218201	-18.581772
11	1.53099918	7.68799877	7.49032593	-0.[illegible]67312
12	1.56199932	4.81299877	4.43155956	38.[illegible]5972
13	1.56299973	1.18799973	1.19262123	10.162231
14	1.56299973	2.24999905	2.00444984	-24.554916
15	1.56499958	6.49999905	6.22315973	-8.[illegible]3994
16	1.62499905	9.96899891	9.76933289	1.[illegible]3384
17	1.78099918	5.37499905	5.16595459	-5.[illegible]4487
18	1.87499905	2.49999905	2.48196507	1.196575
19	1.87499905	9.93999958	9.74389648	-6.[illegible]256
20	2.00000000	1.12499905	1.29202843	-1.[illegible]089
21	2.03099918	0.56199974	0.79676783	1.576817
22	2.12499905	4.74999905	4.67579651	2.[illegible]689
23	2.12499905	6.49999905	6.37898254	0.[illegible]350
24	2.12499905	7.65599918	7.51420593	1.420689
25	2.12499905	0.59299976	0.86055422	4.[illegible]5[illegible]
26	2.12499905	4.24999905	4.19601443	10.[illegible]1454
27	2.24999905	2.65599918	2.61439323	-1.[illegible]0581
28	2.28099918	5.34399891	5.16276550	-8.723354
29	2.37499905	9.93799877	9.7344360[illegible]	-1.[illegible]631
30	2.43799973	1.18799973	1.35630035	-1.[illegible]9869
31	2.43799973	6.49999905	6.25756836	-11.[illegible]059
32	2.49999905	7.65599918	7.57215881	7.[illegible]977

Next the y dimension was divided at the approximate midpoint by an additional knot at y = 5.0 and the spline approximation was computed for the two regions. Two cubic spline functions were thus computed which join with continuity in value and first and second derivative at the line y = 5.0. The fit was improved to an r.m.s. error of .04 for x but the y error remained about the same at .105. The maximum error was .088 for x and .287 for y. The two section spline improved results considerably and produced a smooth curve with no discontinuity at the knot line. The results are tabulated in Table 3 listing the same information as Table 2. This is a simple illustrative example and no attempt will be made here to optimize the fit to the aircraft data by varying the position of the knot or adding more knots. An algorithm which optimizes the positions of the knots is being developed as a continuation of this work. A great deal of flexibility is available in the spline approach and the error could be further reduced by appropriate manipulation of the knots.

Summary and Conclusions

This report presents a discussion of least squares approximation techniques with two dimensional spline function approximation being the main topic. A one dimensional algorithm due to de Boor and Rice was described and its extension to two dimensions is the subject of the work reported here. The algorithm is operational; however, certain problems with the two dimensional inner product remain to be solved. A technique was used in the current program in which a nearest neighbor

Table 3a Error data for x dimension approximation using two cubic spline blocks with new y knot at 5.0.

ROOT MEAN SQUARE ERROR = 0.404584E-01
AVERAGE ERROR = 0.346789E-01
MAXIMUM ERROR = 0.888879E-01 AT 1.03699[illegible] 4.69[illegible]

APPROXIMATION AND SCALED ERROR CURVE

	DATA POINT		APPROXIMATION	DEVIATION X 10L+3
1	0.74999970	1.18799973	0.03919234	8.192368
2	0.78099966	0.81299967	0.01602793	-46.971954
3	0.93799961	7.65599918	0.09274906	-32.250839
4	0.96799958	6.49999905	0.24594599	57.946136
5	0.96899956	2.18799973	0.21012408	22.124222
6	0.96899956	5.37499905	0.33351320	83.513396
7	1.03099918	4.09299946	0.34911191	-88.887863
8	1.09399986	10.24999905	0.11023420	-45.765686
9	1.52999973	0.59399968	1.09113693	31.137466
10	1.53099918	5.37499905	0.97816896	-52.830215
11	1.53099918	7.68799877	0.80351156	-9.488196
12	1.56199932	4.81299877	1.02039337	20.393372
13	1.56299973	1.18799973	0.98512670	-14.873743
14	1.56299973	2.24999905	0.87803519	3.035624
15	1.56499958	6.49999905	0.97232604	34.326431
16	1.62499905	9.96899891	0.77036572	20.366013
17	1.78099918	5.37499905	1.37683487	-60.164444
18	1.87499905	2.49999905	1.37457943	-0.419617
19	1.87499905	9.93999958	1.16143703	-25.562286
20	2.00000000	1.12499905	1.65709400	-42.904846
21	2.03099918	0.56199974	1.82848072	78.481674
22	2.12499905	4.74999905	1.82524395	15.244484
23	2.12499905	6.49999905	1.74788952	47.890656
24	2.12499905	7.65599918	1.63912487	-10.874748
25	2.12499905	0.59299970	1.91993427	44.935226
26	2.12499905	4.24999905	1.80994034	28.941147
27	2.24999905	2.65599918	1.87827873	3.279686
28	2.28099918	5.34399891	1.92720985	-41.789047
29	2.37499905	9.93799877	1.91138649	-26.612274
30	2.43799973	1.18799973	2.09398079	-31.818259
31	2.43799973	6.49999905	1.97709656	-22.903442
32	2.49999905	7.65599918	2.08380699	20.807266

Table 3b Error data for y dimension approximation using two cubic spline blocks with new y knot at 5.0.

ROOT MEAN SQUARE ERROR = 0.105688E 00
AVERAGE ERROR = 0.779330E-01
MAXIMUM ERROR = 0.287311E 00 AT 1.530999 5.374999

APPROXIMATION AND SCALED ERROR CURVE

	DATA POINT		APPROXIMATION	DEVIATION X 10E+2
1	0.74999970	1.18799973	1.25460529	6.660557
2	0.78099966	0.81299967	0.73697913	-13.802045
3	0.93799961	7.65599918	7.45837879	-1.762012
4	0.96799958	6.49999905	6.31870937	0.570965
5	0.96899956	2.18799973	2.23249722	-1.750183
6	0.96899956	5.37499905	5.23783875	4.983902
7	1.03099918	4.09299946	3.89429283	-4.370594
8	1.09399986	10.24999905	10.03150272	0.050354
9	1.52999973	0.59399968	0.57386208	-17.613754
10	1.53099918	5.37499905	4.90068913	-28.731049
11	1.53099918	7.68799877	7.58004379	8.004475
12	1.56199932	4.81299877	4.34725285	28.425308
13	1.56299973	1.18799973	1.18217087	15.117168
14	1.56299973	2.24999905	2.12380791	-12.619114
15	1.56499958	6.49999905	6.19944477	-11.355495
16	1.62499905	9.96899891	9.73657131	-1.342773
17	1.78099918	5.37499905	5.13450527	-8.349419
18	1.87499905	2.49999905	2.47062492	0.062561
19	1.87499905	9.93999958	9.74665833	-0.334072
20	2.00000000	1.12499905	1.28401184	-2.798748
21	2.03099918	0.56199974	0.84765500	6.665533
22	2.12499905	4.74999905	4.70245361	5.245399
23	2.12499905	6.49999905	6.39712429	2.212524
24	2.12499905	7.65599918	7.50129795	0.129890
25	2.12499905	0.59299970	0.91552597	10.252630
26	2.12499905	4.24999905	4.20598793	11.198807
27	2.24999905	2.65599918	2.59386349	-3.113556
28	2.28099918	5.34399891	5.18209648	-6.790257
29	2.37499905	9.93799877	9.73391724	-1.608181
30	2.43799973	1.18799973	1.34694672	-2.805233
31	2.43799973	6.49999905	6.21458817	-16.041077
32	2.49999905	7.65599918	7.57492161	7.492256

rule is used to define the values of randomly spaced data points at the nodes of a uniform grid. This makes computing the integral for the inner product simple but results in error in the approximation. A rule must be employed when using this program in selecting data points over the two dimensional surface so that a quasi-uniform grid is maintained. The points are then grouped by the program and the mean of the group on the x or y axes is taken as the respective abscissa. Further work needs to be done on this and other problems relating to randomly spaced points in two dimensional approximation problems. Subsequent reports will document the algorithm in detail and address certain of these problems.

It must be pointed out that the spline function approach to approximation is only one of a large number of methods each with their own advantages and disadvantages. For the problem of multidimensional approximation of functions the Weighting Function Technique of Jancaitis and Junkens [11] bears particular note and future work will evaluate this and other methods relative to the spline function approach.

Finally, it should be noted that the multidimensional approximation techniques have application in many earth resources data processing areas in addition to image geometric distortion representation. Any case in which randomly located measurements are made of physical, electromagnetic, socio-economic processes is a candidate for this type of approximation technique. Specifically, the conversion of digitized topographic contours,

airborne radiometer, and magnetometer and other geophysical data to a uniform grid format for computer overlay and image processing and analysis is the next application goal of the present work. Progress in these areas will be reported in subsequent documents.

REFERENCES

1. Anuta, P. E., "Spatial Registration of Multispectral and Multitemporal Imagery Using Fast Fourier Transform Techniques", IEEE Transactions on Geoscience Electronics, Vol. GE-8, October 1970.

2. Barnea, D. I., Silverman, H. F., "A Class of Algorithms for Fast Digital Image Registration", IEEE Transactions on Computers, Vol. C-21, pp 179-186, February 1972.

3. Lillestrand, R. L., "Techniques for Change Detection", IEEE Transactions on Computers, Vol. C-21, July 1972.

4. Ralston, Anthony, "A First Course in Numerical Analysis", McGraw Hill, New York, 1965.

5. Rice, John R., "The Approximation of Functions", Volume 1, Addison-Wesley, Reading, Massachusetts, 1964.

6. ibid, Volume 2

7. de Boor, Carl; Rice, John R., "Least Squares Cubic Spline Approximation I - Fixed Knots", Computer Science Department Report CSD TR20, Purdue University, Lafayette, Indiana, April 1968.

8. Lanczos, Cornelius, "Applied Analysis", Prentice Hall, Englewood Cliffs, New Jersey, May 1961.

9. Schoenberg, I. J., "Contributions to the Problem of Approximation of Equidistant Data by Analysis Functions", Quarterly of Applied Mathematics, Vol. 4, Part A, pp 45-99, Part B, pp 112-141, 1946.

10. de Boor, Carl, "Bicubic Spline Interpolation", Journal of Mathematics and Physics, Vol. 41, pp 212-218, 1962.

11. Jancaitis, James R.; Junkens, John L., "Weighting Function Techniques for Storage and Analysis of Mass Remote Sensing Data", Proceedings of Conference on Machine Processing of Remotely Sensed Data, Purdue University, Oct. 16-18, 1973, pp 2B-25, 2B-37.