Signal Processing 10 (1986) 385-393
North-Holland

POLYNOMIAL REPRESENTATION OF PICTURES

Murray EDEN and Michael UNSER (Member EURASIP)

Biomedical Engineering and Instrumentation Branch, Division of Research Services, National Institute of Health, Bethesda, MD 20205, U.S.A.

Riccardo LEONARDI (Member EURASIP)

Laboratorie de Traitement des Signaux, École Polytechnique Fédérale de Lausanne, 16, Chemin de Bellerive, CH-1007 Lausanne, Switzerland

Received 24 October 1984
Revised 22 August 1985 and 26 November 1985

Abstract. In many image processing applications, the discrete values of an image can be embedded in a continuous function. This type of representation can be useful for interpolation, geometrical transformations or special features extraction. Given a rectangular $M \times N$ discrete image (or sub-image), it is shown how to compute a continuous polynomial function that guarantees an exact fit at the considered pixel locations. The polynomials coefficients can be expressed as a linear one-to-one separable transform of the pixels. The transform matrices can be computed using a fast recursive algorithm which enables efficient inversion of a Vandermonde matrix. It is also shown that the least square polynomial approximation with $M' \times N'$ coefficients, in the separable formulation, involves the inversion of two $M' \times M'$ and $N' \times N'$ Hankel matrices.

Zusammenfassung. In mehreren Anwendungen der Bildverarbeitung können digitale Bildgrauwerten als Teile ununterbrochenen Funktionen behandelt werden. Diese Darstellung gilt auch für Interpolation, geometrische Transformationen oder Ermittlung von speziellen Formen. In diesem Artikel handelt es sich um die Verrechnung einer ununterbrochenen Polynom Funktion für ein gegebenes rechteckiges digitales Bild, die eine genaue Interpolation in bestimmten Punkten ermöglicht. Die Polynomial Koeffizienten können mit einer linearen trennbaren Punkttransformation ermittelt werden. Die Transformationmatrizen sind mit einem schnellen rekursiven Algorithmus erreichbar, mit dem eine Vandermondematrize einfach umgekehrt wird. Es wird auch gezeigt, dass die polynomiale Approximation der kleinsten Quadraten mit $M' \times N'$ Koeffizienten, im Fall eines rechtwinkligen Bild, zwei $M' \times M'$ and $N' \times N'$ Hankel Matrizen Umkehrungen ernötigt.

Résumé. Dans beaucoup d'applications du traitement des images, on peut considérer les valeurs discrètes d'une image numérique comme faisant partie d'une fonction continue. Ce type de représentation peut être utile pour l'interpolation, pour effectuer des transformations géométriques ou extraire des propriétés locales. Étant donnée une image (ou sous-image) rectangulaire de dimension $M \times N$ donnée sous forme numérique, on montre comment on peut trouver un polynôme garantissant une interpolation exacte des points de l'image. Les coefficients du polynôme sont obtenus au moyen d'une transformation linéaire bijective séparable des points de l'image. Les matrices de cette transformation peuvent être calculées en utilsant un algorithme récursif rapide qui permet l'inversion efficace des matrices de Vandermonde. Dans un deuxième temps, on expose comment trouver la meilleure approximation polynômiale au sens des moindres carrés avec $M' \times N'$ coefficients dans le cas d'une image rectangulaire, ce qui permet une formulation séparable du problème. Le résultat comprend notamment l'inversion de deux matrices carrées de Hankel de rang respectivement M' et N'.

Keywords. Interpolation, inversion, least square approximation, polynomial approximation, separability, Vandermonde matrix.

1. Introduction

In the great majority of picture processing applications nowadays, the original picture to be manipulated is represented by a set of discrete values with the sample points uniformly spaced. It is customary to represent the scene as a function $g(k, l)$ defined on the nodes of a square lattice.

This consensus in favor of 'digitizing' pictures has come about because of the power of the digital computer. The complicated algorithms it can execute are impractical or impossible with typical analog imaging devices, say a video chain. In recent years, a wide variety of filter procedure have been developed for performing numerical manipulations on pictures. Such procedures have been suggested for picture interpolation, smoothing, compression, enhancement, texture simulation, contour and feature detection. A picture filter may be regarded as a transformation from one picture to another. In the great majority of these transformations, the numerical procedure is the digital analog of an operation on a continuous function that the quantized sample points are intended to approximate.

It has been pointed out, already in 1958, by Schreiber et al. [5] and most recently by Kocher [4] or Haralick [2, 3] that the discrete values can be embedded in a continuous function $g(x, y)$ defined on the unit square surrounding every pixel. Such operations as interpolation, projection, and affine transformation take on a consistent interpretation and the numerical computations entailed by an operation may be simplified. A local continuous image representation may also be used to extract edges or local texture properties.

In general, very little can be said about the nature of the continuous function $g(x, y)$. If the original object is something like a photograph or X-ray image; that is derived from an optical system, we can only specify that the function is bounded and nonnegative. Nonetheless, because the picture and its transforms are intended to be looked at by a human observer, the picture can be considered as having been bandpass filtered, with no information loss experienced so far as the observer is concerned, as long as the observer does not approach the image too closely.

Thus, we explicitly assume that the information in the picture is bounded by certain psychological properties of the human observer. It should be recalled that visual discrimination falls off rapidly for linear dimensional frequencies above about ten cycles per degree of visual angle. Accoringly, it is safe to assume $g(x, y)$ is continuous elsewhere (within the unit square). We are guaranteed by Weierstrass' theorem to find a uniform approximation by polynomials. An even stronger condition can be assumed: that $g(x, y)$ is analytic in the interval of interest. In this case, we can approximate as closely as we please with a power series (see, for example, [1]).

We shall show that bidimensional polynomial coefficients allow an exact representation (Section 3) or an approximation (Section 5) to a rectangular array. We shall show that, by considering rectangular regions, it is possible to use separability of the polynomial representation to reduce computation in both interpolation and least square approximation.

A polynomial representation has, at the least, the following advantages:

(1) The correspondence between array values and the polynomial coefficients allows the discrete set of values to be described by an analytical continuous function. Further, this function has a very simple expression. Therefore, operations such as affine transformations can easily be defined and computation can be simplified. The polynomial form is also much better adapted to perform mathematical operations such as derivation, integration, or gradient evaluation than are more elaborate analytic forms, let alone computing discrete approximations to these intrinsically continuous mathematical constructs.

(2) Polynomials fit the geometrical forms of images harmoniously. Slowly varying surfaces in images are well represented by polynomials. The subjective quality of polynomials is pleasant to the human eye. More particularly, one avoids the oscillatory effects that are so frequently present in Fourier representations or any other periodic function transforms. In these instances, the investigator may be obliged to use a very large set of coefficients to eliminate the oscillatory artefact in the transformed picture.

2. One-dimensional interpolation theorem

The interpolation theorem in one dimension establishes that given N distinct points k_i ($i = 1, 2, \ldots, N$ and $k_i \neq k_j$), there is a unique polynomial

$$p(x) = \sum_{i=1}^{N} a_i x^{i-1} \tag{1}$$

satisfying the N equations

$$g(k_j) = \sum_{i=1}^{N} a_i k_j^{i-1} \tag{2}$$

with $j = 1, 2, \ldots, N$.

Let us define $\boldsymbol{g} = [g(k_1) \ldots g(k_N)]^T$ as the vector of the function values at the N points of interest and $\boldsymbol{a} = [a_1 \ldots a_N]^T$ the vector of the polynomial coefficient values. Using this formulation, the system of equations characterized by equation (2) can be rewritten as

$$\boldsymbol{g} = \boldsymbol{V}(k_1, \ldots, k_N) \cdot \boldsymbol{a}, \tag{3}$$

where $\boldsymbol{V}$ is the $N \times N$ matrix defined by

$$\boldsymbol{V}(k_1, \ldots, k_N) = \begin{bmatrix} 1 & k_1 & k_1^2 & \cdots & k_1^{N-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & k_N & k_N^2 & \cdots & k_N^{N-1} \end{bmatrix}. \tag{4}$$

We will refer to a matrix of this form as the Vandermonde matrix and designate it by $\boldsymbol{V}$. The polynomial coefficient vector $\boldsymbol{a}$ may be computed from $\boldsymbol{g}$ by inverting the matrix $\boldsymbol{V}$,

$$\boldsymbol{a} = \boldsymbol{V}(k_1, \ldots, k_N)^{-1} \cdot \boldsymbol{g}. \tag{5}$$

Equation (4) or (5) defines a one-to-one linear mapping between the space of function values and the space of polynomial coefficients. The evaluation of the polynomial coefficients can simply be performed by applying a linear transform defined by the matrix $\boldsymbol{V}^{-1}$ to the vector of function values $\boldsymbol{g}$.

3. Two-dimensional interpolation-conditions for separability

In two dimensions, it is in general not true that, given P distinct points, there will be a linear combination of P polynomial terms of the form $x^i y^j$ that can take on the P preassigned values $g(k_i, l_j)$. However, in picture processing, the number of points and their locations in the picture are almost always under the control of the experimentor. In particular, arbitrary values may be chosen for the highest power of x (say $M-1$) and the highest power of y (say $N-1$) and we define two sets of abscissas k_i and ordinates l_j, respectively:

$$\begin{aligned} &\text{horizontal:} \quad K = \{k_1, k_2, \ldots, k_M\}, \\ &\text{vertical:} \quad L = \{l_1, l_2, \ldots, l_N\}. \end{aligned} \tag{6}$$

As in the one-dimensional case, it will be shown that given $M \times N$ distinct points defined by $K * L$ ($k_i \neq k_j$ and $l_i \neq l_j$; the symbol "$*$" represents the cartesian product of two sets), there is a unique bidimensional polynomial

$$P(x, y) = \sum_{i=1}^{M} \sum_{j=1}^{N} a_{i,j} x^{i-1} y^{j-1}, \tag{7}$$

satisfying the $M \times N$ equations

$$g(k, l) = \sum_{i=1}^{M} \sum_{j=1}^{N} a_{i,j} k^{i-1} l^{j-1}$$

$$\text{with } (k, l) \in K * L. \tag{8}$$

Of particular interest is the fact that the transform matrix for the family is separable. Separability very substantially decreases the number of computations needed to solve for the polynomial coefficients. In order to prove separability, we introduce the following matrix notation:

$$\boldsymbol{G} = \begin{bmatrix} g(k_1, l_1) & g(k_1, l_2) & \cdots & g(k_1, l_N) \\ g(k_2, l_1) & g(k_2, l_2) & \cdots & g(k_2, l_N) \\ \vdots & \vdots & & \vdots \\ g(k_M, l_1) & g(k_M, l_2) & \cdots & g(k_M, l_N) \end{bmatrix} \tag{9}$$

is the $M \times N$ matrix of the grey values of the pixels defined by $K * L$ and

$$\boldsymbol{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \vdots & \vdots & & \vdots \\ a_{M,1} & a_{M,2} & \cdots & a_{M,N} \end{bmatrix} \tag{10}$$

is the $M \times N$ matrix of the bidimensional polynomial coefficients. To ensure separability, we need to show that the $M \times N$ coefficients are given by equation (8). It can be written as the following matrix equation:

$$\boldsymbol{G} = \boldsymbol{B}_k \cdot \boldsymbol{A} \cdot \boldsymbol{B}_l^{\mathrm{T}}. \tag{11}$$

The $M \times M$ transformation matrix $\boldsymbol{B}_k$ operates on the rows while the $N \times N$ transformation matrix $\boldsymbol{B}_l$ operates on the columns. The explicit form of the equation satisfying the required condition is

$$\begin{bmatrix} g(k_1,l_1) & g(k_1,l_2) & \cdots & g(k_1,l_N) \\ g(k_2,l_1) & g(k_2,l_2) & \cdots & g(k_2,l_N) \\ \vdots & \vdots & & \vdots \\ g(k_M,l_1) & g(k_M,l_2) & \cdots & g(k_M,l_N) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & k_1 & k_1^2 & \cdots & k_1^{M-1} \\ 1 & k_2 & k_2^2 & \cdots & k_2^{M-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & k_M & k_M^2 & \cdots & k_M^{M-1} \end{bmatrix}$$

$$\cdot \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \vdots & & & \vdots \\ a_{M,1} & a_{M,2} & \cdots & a_{M,N} \end{bmatrix}$$

$$\cdot \begin{bmatrix} 1 & 1 & \cdots & 1 \\ l_1 & l_2 & \cdots & l_N \\ \vdots & \vdots & & \vdots \\ l_1^{N-1} & l_2^{N-1} & \cdots & l_N^{N-1} \end{bmatrix}. \tag{12}$$

Using the notation introduced in the previous section, equation (12) can be rewritten as

$$\boldsymbol{G} = \boldsymbol{V}(k_1,\ldots,k_M) \cdot \boldsymbol{A} \cdot \boldsymbol{V}(l_1,\ldots,l_N)^{\mathrm{T}}, \tag{13}$$

so that

$$\boldsymbol{A} = \boldsymbol{V}(k_1,\ldots,k_M)^{-1} \cdot \boldsymbol{G} \cdot (\boldsymbol{V}(l_1,\ldots,l_N)^{\mathrm{T}})^{-1}. \tag{14}$$

The matrices $\boldsymbol{V}_k = \boldsymbol{V}(k_1,\ldots,k_M)$ and $\boldsymbol{V}_l = \boldsymbol{V}(l_1,\ldots,l_N)$ are in the Vandermonde form; that is, the determinants $|\boldsymbol{V}_k|$ and $|\boldsymbol{V}_l|$ are simply

$$|\boldsymbol{V}_k| = \prod_{j=1,i>j}^{M} (k_i - k_j), \tag{15}$$

$$|\boldsymbol{V}_l| = \prod_{j=1,i>j}^{N} (l_i - l_j). \tag{16}$$

Clearly, these quantities are nonzero when $k_i \neq k_j$ and $l_i \neq l_j$, $\forall i,j$ which is sufficient to prove that the inverse of each matrix $\boldsymbol{V}_k$ or $\boldsymbol{V}_l$ exists and that the solution for $\boldsymbol{A}$ is unique (interpolation theorem).

4. A fast algorithm for evaluating the transform matrix coefficients

The polynomial coefficients may be obtained from the picture array $\boldsymbol{G}$ by applying a separable linear transform defined by equation (14). The computation of the transform coefficients requires the inversion of the two Vandermonde matrices $\boldsymbol{V}_k$ and $\boldsymbol{V}_l$. Such an inversion can be obtained as shown in [6], using the Gaussian elimination method. It will be shown next that the inverse of these matrices can be computed recursively in a much faster way.

The proposed algorithm takes into account the particular structure of these matrices and will produce the inverse of all lower order Vandermonde matrices as a byproduct.

4.1. Order recursion for fast Vandermonde matrix inversion

We shall consider the following Vandermonde matrix $\boldsymbol{V}$:

$$\boldsymbol{V} = \boldsymbol{V}(x_1,\ldots,x_N) = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{N-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^{N-1} \end{bmatrix}. \tag{17}$$

The determinant of $\boldsymbol{V}$ is given by

$$|\boldsymbol{V}| = \prod_{j=1,i>j}^{N} (x_i - x_j). \tag{18}$$

Let us call $\boldsymbol{U}_n$ the inverse of the submatrix $\boldsymbol{V}_n = \boldsymbol{V}(x_1,\ldots,x_n)$ with $n \leq N$. We also define $a_{i,j}$ to be the elements of a matrix $\boldsymbol{A}$.

By using Lagrange's interpolation formula, one can find (see Appendix A) the expression for $\boldsymbol{U}_n$.

The elements $u_{i,j}^{(n)}$ are given by

$$u_{i,j}^{(n)} = \varphi_{j,i}^{(n)} / \alpha_i^{(n)}, \tag{19}$$

where $\varphi_{i,j}^{(n)}$ and $\alpha_i^{(n)}$ are expressed by the following recursive relations.

Recursion starts with

$$\begin{aligned} &\varphi_{1,1}^{(1)} = 1, \\ &\alpha_1^{(1)} = 1. \end{aligned} \tag{20}$$

$\alpha_i^{(n)}$ can be computed using the simple recursive relation

$$\alpha_i^{(n)} = \begin{cases} \alpha_i^{(n-1)}(x_i - x_n) & \text{if } n \neq i, \\ \alpha_i^{(n-1)} & \text{if } n = i. \end{cases} \tag{21}$$

To evaluate $\varphi_{i,j}^{(n)}$ we use the following recursive equations:

$$\begin{cases} \varphi_{i,1}^{(n)} = -x_n \varphi_{i,1}^{(n-1)} & (j=1), \\ \varphi_{i,j}^{(n)} = \varphi_{i,j-1}^{(n-1)} - x_n \varphi_{i,j}^{(n-1)} & \\ & (j=2,\ldots,n-1), \\ \varphi_{i,n}^{(n)} = \varphi_{i,n-1}^{(n-1)} = 1 & (j=n), \end{cases} \tag{22}$$

$$\begin{cases} \varphi_{n,1}^{(n)} = -x_1 \varphi_{1,1}^{(n-1)} & (j=1), \\ \varphi_{n,j}^{(n)} = \varphi_{1,j-1}^{(n-1)} - x_1 \varphi_{1,j}^{(n-1)} & \\ & (j=2,\ldots,n-1), \\ \varphi_{n,n}^{(n)} = 1 & (j=n). \end{cases} \tag{23}$$

The inverse of matrix V is obtained by evaluating expression (19) for $n = N$.

4.2. Convolutional formulation

We have shown in Section 3 that the polynomial coefficients can be computed from a separable linear tranform given by equation (14). The expression for a given coefficient a_{rs} as a function of the input picture array is

$$a_{rs} = \boldsymbol{v}_r^{\mathrm{T}} \cdot \boldsymbol{G} \cdot \boldsymbol{v}_s', \tag{24}$$

where the vector $\boldsymbol{v}_r$ is formed from the M elements of row r of matrix $\boldsymbol{V}_k^{-1}(k_1, k_2, \ldots, k_M)$ and $\boldsymbol{v}_s'$ is formed from the N elements of row s of matrix $\boldsymbol{V}_l^{-1}(l_1, l_2, \ldots, l_N)$. The separable mask corresponding to this coefficient can be formed from the direct (or Kronecker) product

$$\boldsymbol{V}_{rs} = \boldsymbol{v}_r \odot (\boldsymbol{v}_s')^{\mathrm{T}}$$

$$= \begin{bmatrix} v_{rs}(k_1, l_1) & v_{rs}(k_1, l_2) & \cdots & v_{rs}(k_1, l_N) \\ v_{rs}(k_2, l_1) & v_{rs}(k_2, l_2) & \cdots & v_{rs}(k_2, l_N) \\ \vdots & \vdots & & \vdots \\ v_{rs}(k_M, l_1) & v_{rs}(k_M, l_2) & \cdots & v_{rs}(k_M, l_N) \end{bmatrix}. \tag{25}$$

Using this notation, the coefficients a_{rs} are obtained from the summation of all the term by term products between the bidimensional mask $\boldsymbol{V}_{rs}$ and the array $\boldsymbol{G}$:

$$a_{rs} = \sum_{i=1}^{M} \sum_{j=1}^{N} g(k_i, l_j) v_{rs}(k_i, l_j). \tag{26}$$

5. Least square approximation

In the foregoing sections, we have derived polynomial expressions that fit exactly with some set of $M \times N$ grey level values assigned to $M \times N$ coordinate locations in a picture. More commonly, those involved in picture processing will be interested in determining a polynomial which is a best approximation in the least square sense to the complete set of picture point grey level values. We show below how the same set of bidimensional polynomial expressions can serve this purpose. As before, the separability of the dimensions simplifies computation.

5.1. Separable least square approximation

Let $K = \{k_1, k_2, \ldots, k_M\}$ and $L = \{l_1, l_2, \ldots, l_N\}$ define a set of coordinate points $K * L$ on which the grey level values are to be approximated by a $p \times q$ polynomial with $p = M' - 1 \leq M$ and $q = N' - 1 \leq N$. Let $\hat{\boldsymbol{G}}$ denote the picture array of the estimated values:

$$\hat{\boldsymbol{G}} = \boldsymbol{V}'(k_1, k_2, \ldots, k_M) \cdot \boldsymbol{A}' \cdot \boldsymbol{V}'(l_1, l_2, \ldots, l_N)^{\mathrm{T}}. \tag{27}$$

The matrix $V'(k_1, k_2, \ldots, k_M)$ (respectively $V'(l_1, l_2, \ldots, l_N)$) is of dimension $M \times M'$ (respectively $N \times N'$) and is obtained in taking the first M' (respectively N') columns of $V(k_1, k_2, \ldots, k_M)$ (respectively $V'(l_1, l_2, \ldots, l_N)$); A' is the $M' \times N'$ matrix of polynomial coefficients. The estimation error is given by

$$G - \hat{G} = G - V'(k_1, k_2, \ldots, k_M) \cdot A' \cdot V'(l_1, l_2, \ldots, l_N)^{\mathrm{T}}. \quad (28)$$

The optimal set of coefficients A' that minimizes the square mean error is given by (see Appendix B)

$$A' = (V_k'^{\mathrm{T}} \cdot V_k')^{-1} \cdot V_k'^{\mathrm{T}} \cdot G \cdot V_l' \cdot (V_l'^{\mathrm{T}} \cdot V_l')^{-1}. \quad (29)$$

The matrices that appear on the left- and right-hand sides can be seen as pseudo-inverses. If we define

$$V_k = (V_k'^{\mathrm{T}} \cdot V_k')^{-1} \cdot V_k'^{\mathrm{T}}$$

and

$$V_l = (V_l'^{\mathrm{T}} \cdot V_l')^{-1} \cdot V_l'^{\mathrm{T}},$$

then equation (29) can be rewritten in a similar form to equation (14) of Section 3:

$$A' = V_k \cdot G \cdot (V_l)^{\mathrm{T}}. \quad (30)$$

Remark: The separable approach to solve the least square approximation problems can be generalized to any linear transformation.

5.2. *Properties*

- If $M' = M$ and $N' = N$, the V_k' and V_l' are square matrices and equation (30) simply becomes the interpolation formula for separable expressions, i.e.,

$$A = V_k'^{-1} \cdot G \cdot (V_l'^{\mathrm{T}})^{-1}. \quad (31)$$

- In the case of least square approximation by polynomials, the matrices $V_k'^{\mathrm{T}} \cdot V_k'$ and $V_l'^{\mathrm{T}} \cdot V_l'$ are Hankel matrices of the general form

$$\begin{bmatrix} \nu_0 & \nu_1 & \cdots & \nu_i \\ \nu_1 & \nu_2 & \cdots & \nu_{i+1} \\ \vdots & \vdots & & \vdots \\ \nu_i & \nu_{i+1} & \cdots & \nu_{2i} \end{bmatrix}. \quad (32)$$

This matrix is completely specified by $2i+1$ independent terms. Let us define the two products of matrices respectively by H_k and H_l. In the case of H_k, its elements are equal to

$$\nu_i = \sum_{j=1}^{M} k_j^i. \quad (33)$$

If $p = M' - 1$ is the highest power of k in the polynomial expression, i is at most equal to $2p$. In the case of H_l, the elements are computed in the same way with k replaced by l, M by N, M' by N', and p by q.

Therefore, we see that when separability applies, the computation of the best least square approximation is simply obtained by the inversion of two Hankel matrices of respective sizes $M' \times M'$ and $N' \times N'$ and by four matrix multiplications ($M' = p+1$; $N' = q+1$). The five matrices involved in the multiplication are of sizes $M' \times M'$, $M' \times M$, $M \times N$, $N \times N'$, and $N' \times N'$. If separability is not used and if the same characteristic polynomial coefficients were selected to establish the approximation, the computation would require the inversion of one symmetric matrix of size $(M'N') \times (M'N')$ and two matrix-vector multiplications. The matrix-vector multiplications consists in a $(M'N') \times (MN)$ matrix with a MN-sized vector and in an $(M'N') \times (M'N')$ matrix with an $M'N'$ vector. As far as the approximation is elaborated on rectangular regions, separability provides a great gain in computation time.

6. Conclusion

A method to obtain an exact or simplified least square representation of an image by a set of polynomial coefficients has been presented. The coefficients are computed using a separable linear transformation. Fast algorithms have been derived for the computation of the transform matrices in the case of an exact fit or a least square approximation. This technique may be seen as a particular case of transform processing when using polynomial functions. Nevertheless, it has the advan-

tage to be directly related to a simple continuous image representation in terms of power series. This type of description may be useful in many image processing applications. It is, for example, especially suited to compute geometrical transformations.

We are continuing to study the potential of the polynomial representation in picture coding applications. We would also point out that local polynomial image representation appears to be particularly suited for feature extraction and may be adaptable to such operations as edge detection or the measurement and classification of texture properties.

Appendix A. Inverse Vandermonde matrix

We shall consider the following Vandermonde matrix V:

$$V = V(x_1, \ldots, x_N) = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{N-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_N & x_N^2 & \cdots & x_N^{N-1} \end{bmatrix}. \tag{A.1}$$

The determinant of V is given by

$$|V| = \prod_{j=1, i>j}^{N} (x_i - x_j). \tag{A.2}$$

Let us call U the inverse of the matrix V. We also define $a_{i,j}$ to be the elements of a matrix A.

Let us consider the interpolation polynomial $p_{N-1}(x)$ defined as

$$p_{N-1}(x) = \sum_{j=1}^{N} a_j x^{j-1}. \tag{A.3}$$

The elements of U, $u_{i,j}$, can easily be computed using Lagrange's interpolation formula:

$$p_{N-1}(x) = \sum_{i=1}^{N} g(x_i) s_i^{(N-1)}, \tag{A.4}$$

where $g(x_i)$ represent the values of the interpolated function at $x = x_i$, and $s_i^{(N-1)}(x)$ are polynomials of order $N-1$ of the form

$$s_i^{(N-1)} = \prod_{j=1, j\neq i}^{N} \frac{x - x_j}{x_i - x_j} = \sum_{j=1}^{N} c_{i,j}^{(N)} x^{j-1}. \tag{A.5}$$

We shall associate a matrix $C^{(N)}$ to the elements $c_{ij}^{(N)}$.

The polynomials $s_i^{(N-1)}(x)$ have the following property:

$$s_i^{(N-1)}(x_j) = \delta_{i,j} = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \tag{A.6}$$

where $\delta_{i,j}$ is the Kronecker's symbol.

Let a represent the vector of coefficients of the interpolating polynomial. Using equation (5), we have

$$a_j = \sum_{i=1}^{N} u_{ji} g(x_i). \tag{A.7}$$

Now introducing this expression in equation (A.3), and comparing it with relation (A.6), we have

$$u_{ji} = c_{i,j}^{(N)}. \tag{A.8}$$

Therefore, the elements of the inverse Vandermonde matrix are simply obtained by transposing the matrix $C^{(N)}$.

Let us now find the recursive relations which allow to calculate $c_{i,j}^{(N)}$. For this purpose, we define the following ratios:

$$c_{i,j}^{(N)} = \varphi_{i,j}^{(N)} / \alpha_i^{(N)}, \tag{A.9}$$

where

$$\alpha_i^{(N)} = \prod_{j=1, j\neq 1}^{N} (x_i - x_j). \tag{A.10}$$

This denominator is the same for each element of a given row of $C^{(N)}$. It can recursively be computed using the following expression:

$$\alpha_i^{(N)} = \begin{cases} \alpha_i^{(N-1)}(x_i - x_N) & \text{if } N \neq i, \\ \alpha_i^{(N-1)} & \text{if } N = i. \end{cases} \tag{A.11}$$

To find the expression of the numerator $\varphi_{i,j}^{(N)}$, one has to consider the two following cases.

Case 1 (first $N-1$ rows of $C^{(N)}$). Using equation (A.5), we have

$$\sum_{j=1}^{N} \varphi_{i,j}^{(N)} x^{j-1} = \prod_{j=1, j\neq i}^{N} \frac{(x - x_j)}{(x_i - x_j)} \alpha_i^{(N)}.$$

Using (A.11) we obtain

$$\begin{aligned}
&\sum_{j=1}^{N} \varphi_{i,j}^{(N)} x^{j-1} \\
&\quad = \prod_{j=1, j\neq i}^{N} \frac{(x-x_j)}{(x_i-x_j)}\, \alpha_i^{(N-1)}(x_i-x_N) \\
&\quad = \prod_{j=1, j\neq i}^{N-1} \frac{(x-x_j)}{(x_i-x_j)}\, \alpha_i^{(N-1)}(x-x_N) \\
&\quad = (x-x_N) \sum_{j=1}^{N-1} \varphi_{i,j}^{(N-1)} x^{j-1} \\
&\quad = \prod_{j=1, j\neq i}^{N-1} (x-x_j)(x-x_N) \\
&\quad = \sum_{j=1}^{N-1} \varphi_{i,j}^{(N-1)} x^{j-1}(x-x_N). \qquad \text{(A.12)}
\end{aligned}$$

By comparing the terms corresponding to the same powers of x, the following recursive relations are obtained:

$$\begin{cases}
\varphi_{i,1}^{(N)} = -x_N \varphi_{i,1}^{(N-1)} & (j=1), \\
\varphi_{i,j}^{(N)} = \varphi_{i,j-1}^{(N-1)} - x_N \varphi_{i,j}^{(N-1)} & \\
& (j=2,\ldots,N-1), \\
\varphi_{i,N}^{(N)} = \varphi_{i,N-1}^{(N-1)} = 1 & (j=N).
\end{cases} \qquad \text{(A.13)}$$

Case 2 (last line of $C^{(N)}$). Using equation (A.5), we find that

$$\begin{aligned}
&\sum_{j=1}^{N} \varphi_{N,j}^{(N)} x^{j-1} \\
&\quad = \prod_{j=1}^{N-1} (x-x_j) \\
&\quad = \prod_{j=1, j\neq i}^{N-1} (x-x_j)(x-x_i) \quad \forall i \neq N \\
&\quad = \sum_{j=1}^{N-1} \varphi_{i,j}^{(N-1)} x^{j-1}(x-x_i). \qquad \text{(A.14)}
\end{aligned}$$

By comparing the terms corresponding to the same powers of x, the following recursive relations are deduced $\forall i = 1, 2, \ldots, N-1$.

$$\begin{cases}
\varphi_{N,1}^{(N)} = -x_i \varphi_{i,1}^{(N-1)} & (j=1), \\
\varphi_{N,j}^{(N)} = \varphi_{i,j-1}^{(N-1)} - x_i \varphi_{i,j}^{(N-1)} & \\
& (j=2,3,\ldots,N-1), \\
\varphi_{N,N}^{(N)} = 1 & (j=N).
\end{cases} \qquad \text{(A.15)}$$

Appendix B. Bidimensional least square approximation

Let the approximation of G with $M' \times N'$ coefficients be given by

$$\hat{G} = V_k' \cdot A \cdot V_l'^{\mathrm{T}}, \qquad \text{(B.1)}$$

where V_k' and V_l' are $M' \times M$ and $N' \times N$ matrices respectively with $M' < M$ and $N' < N$. We want to find the matrix A^* that minimizes the square mean error of the approximation

$$\begin{aligned}
\varepsilon^2 &= |G - \hat{G}|^2 = \mathrm{tr}[(G-\hat{G})^{\mathrm{T}} \cdot (G-\hat{G})] \\
&= \sum_{i=1}^{M} \sum_{j=1}^{N} [(g_{i,j} - \hat{g}_{i,j})^2]. \qquad \text{(B.2)}
\end{aligned}$$

Minimization of this quantity is obtained when

$$\left.\frac{\partial \varepsilon^2}{\partial A}\right|_{A=A^*} = 0. \qquad \text{(B.3)}$$

To compute the optimum A^*, we establish here the following rules of gradient matrices:

(i) $\dfrac{\partial}{\partial A}[\mathrm{tr}(B \cdot A \cdot C)] = B^{\mathrm{T}} \cdot C^{\mathrm{T}}$,

(ii) $\dfrac{\partial}{\partial A}[\mathrm{tr}(B \cdot A^{\mathrm{T}} \cdot C \cdot A \cdot B^{\mathrm{T}})]$

$$= 2C \cdot A \cdot B^{\mathrm{T}} \cdot B.$$

The gradient of ε^2 with respect to A can be written as

$$\begin{aligned}
\frac{\partial \varepsilon^2}{\partial A} &= \frac{\partial}{\partial A}(\mathrm{tr}[(G - V_k' \cdot A \cdot V_l'^{\mathrm{T}})^{\mathrm{T}} \\
&\qquad \cdot (G - V_k' \cdot A \cdot V_l'^{\mathrm{T}})]) \\
&= \frac{\partial}{\partial A}(\mathrm{tr}[G^{\mathrm{T}} \cdot G - V_l' \cdot A^{\mathrm{T}} \cdot V_k'^{\mathrm{T}} \\
&\qquad \cdot G - G^{\mathrm{T}} \cdot V_k' \cdot A \cdot V_l'^{\mathrm{T}} \\
&\qquad + V_l' \cdot A^{\mathrm{T}} \cdot V_k'^{\mathrm{T}} \cdot V_k' \cdot A \cdot V_l'^{\mathrm{T}}]). \qquad \text{(B.4)}
\end{aligned}$$

Recalling that, for any matrix A and B,

(iii) $\mathrm{tr}[AB] = \mathrm{tr}\,[BA]$,

(iv) $\mathrm{tr}[B] = \mathrm{tr}[B^{\mathrm{T}}]$,

(v) $\mathrm{tr}[A+B] = \mathrm{tr}[A] + \mathrm{tr}[B]$,

expression (B.4) can be manipulated as follows:

$$\varepsilon^2 = \mathrm{tr}[\boldsymbol{G}^{\mathrm{T}} \cdot \boldsymbol{G}] - 2\,\mathrm{tr}[\boldsymbol{G}^{\mathrm{T}} \cdot \boldsymbol{V}'_k \cdot \boldsymbol{A} \cdot \boldsymbol{V}'^{\mathrm{T}}_l] + \mathrm{tr}[\boldsymbol{V}'_l \cdot \boldsymbol{A}^{\mathrm{T}} \cdot \boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{V}'_k \cdot \boldsymbol{A} \cdot \boldsymbol{V}'^{\mathrm{T}}_l]. \quad \text{(B.5)}$$

Applying (i) and (ii), it follows that

$$\frac{\partial \varepsilon^2}{\partial \boldsymbol{A}} = -2\boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{G} \cdot \boldsymbol{V}'_l + 2(\boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{V}'_k) \cdot \boldsymbol{A} \cdot (\boldsymbol{V}'^{\mathrm{T}}_l \cdot \boldsymbol{V}'_l). \quad \text{(B.6)}$$

The gradient will be zero for the particular value

$$\boldsymbol{A}^* = (\boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{V}'_k)^{-1} \cdot \boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{G} \cdot \boldsymbol{V}'_l \cdot (\boldsymbol{V}'^{\mathrm{T}}_l \cdot \boldsymbol{V}'_l)^{-1}, \quad \text{(B.7)}$$

as long as matrices $\boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{V}'_k$ and $\boldsymbol{V}'^{\mathrm{T}}_k \cdot \boldsymbol{V}'_l$ are non-singular. Notice that if $\boldsymbol{V}'_k$ and $\boldsymbol{V}'_l$ are square matrices, that is, when $M' = M$ and $N' = N$, this form reduces to the interpolation formula (14). In such a case, the error of approximation turns out to be zero (exact fit).

References

[1] P.J. Davis, *Interpolation and Approximation*, Blaisdell, New York, 1963.

[2] R.M. Haralick, "Digital step edges from zero crossing of second directional derivatives", *IEEE Trans. PAMI*, Vol. PAMI-6, No. 1, 1984, pp. 58-68.

[3] R.M. Haralick and L. Watson, "A Facet model for image data", *Computer Graphics Image Process.*, Vol. 15, 1981, pp. 113-129.

[4] M. Kocher, "Codage d'images à haute compression basé sur un modèle contour-texture", Ph.D. Thesis No. 476, Swiss Federal Institute of Technology, Lausanne, Switzerland, 1983.

[5] W.F. Schreiber, C.F. Knapp and N.D. Kay, "Synthetic highs: An experimental TV bandwidth reduction system", *J. Sci. Motion and TV Engrs.*, Vol. 68, August 1959, pp. 525-537.

[6] G. Strang, *Linear Algebra and Its Applications*, Academic Press, New York, 1976.