Rank-Based Decompositions of Morphological Templates

Peter Sussner and Gerhard X. Ritter, *Senior Member, IEEE*

***Abstract*—Methods for matrix decomposition have found numerous applications in image processing, in particular for the problem of template decomposition. Since existing matrix decomposition techniques are mainly concerned with the linear domain, we consider it timely to investigate matrix decomposition techniques in the nonlinear domain with applications in image processing. The mathematical basis for these investigations is the new theory of rank within minimax algebra. Thus far, only minimax decompositions of rank 1 and rank 2 matrices into outer product expansions are known to the image processing community. In this paper we derive a heuristic algorithm for the decomposition of matrices having arbitrary rank.**

***Index Terms*—Image processing, matrix decomposition, matrix rank, minimax algebra, morphology, template.**

I. Introduction

CONVOLUTIONS are a fundamental tool in image processing. Classical examples of two-dimensional (2-D) linear convolutions include image correlation, the mean filter, the discrete Fourier transform, and a multitude of edge mask filters. Nonlinear convolutions are used in such operations as the median filter, the medial axis transform, and erosion and dilation as defined in mathematical morphology. For large convolution masks or structuring elements, the computation cost resulting from implementation can be prohibitive. However, in many instances, this cost can be significantly reduced by decomposing the templates representing the masks or structuring elements into a sequence of smaller templates. In addition, such decomposition can often be made architecture specific and, thus, resulting in optimal transform performance. In this paper, we provide methods for decomposing morphological templates which are analogous to decomposition methods used in the linear domain. Specifically, we define the notion of the rank of a morphological template and we present an algorithm for decomposing such templates based on the rank notion.

Linear convolutions using masks of templates and nonlinear morphological convolutions using structuring elements have the common characteristic that they require applying a template or structuring element to an image, pixel by pixel, in order to yield a new image. The notion of templates and structuring elements, when viewed as small images, are identical, only the operations of combining the weights with image pixels in order to obtain a new pixel value differ. In linear convolutions, the combining operation is a linear sum, while in the morphological convolution the nonlinear operation of maximum (or minimum) of the sum of pixel values and corresponding template weights is applied.

Intuitively, the problem of template decomposition is that given a template $\mathbf{t}$, find a sequence of smaller templates $\mathbf{t}_1, \cdots, \mathbf{t}_n$ such that applying $\mathbf{t}$ to an image is equivalent to applying $\mathbf{t}_1, \cdots, \mathbf{t}_n$ sequentially to the image. In other words, t can be algebraically expressed in terms of $\mathbf{t}_1, \cdots, \mathbf{t}_n$.

One purpose of template decomposition is to fit the support of the template (i.e., the convolution kernel) optimally into an existing machine constrained by its hardware configuration. For example ERIM's CytoComputer [21] cannot deal with templates of size larger than 3×3 on each pipeline stage. Thus, a large template, intended for image processing on a CytoComputer, has to be decomposed into a sequence of 3×3 or smaller templates.

A more important motivation for template decomposition is to speed up template operations. For large convolution masks, the computation cost resulting from implementation can be prohibitive. However, in many instances, this cost can be significantly reduced by decomposing the masks or templates into a sequence of smaller templates. For instance, the linear convolution of an image with a gray-valued $n \times n$ template requires n^2 multiplications and $n^2 - 1$ additions to compute a new image pixel value; while the same convolution computed with an $1 \times n$ row template followed by an $n \times 1$ column template takes only $2n$ multiplications and $2(n-1)$ additions for each new image pixel value. This cost saving may still hold for parallel architectures such as mesh connected array processors [11], where the cost is proportional to the size of the template.

The problem of decomposing *morphological templates* has been investigated by a host of researchers. Zhuang and Haralick [29] gave a heuristic algorithm based on tree search that can find an optimal two-point decomposition of a morphological template if such a decomposition exists. A two-point decomposition consists of a sequence of templates each consisting of at most two points. A two-point decomposition may be best suited for parallel architectures with a limited number of local connections since each two-point template can be applied to an entire image in a multiply-shift-accumulate cycle [11]. Xu [27] has developed an algorithm, using chain code information, for the decomposition of convex morphological templates for two-point system configurations. Again using chain-code information, Park and Chin [16] provide an optimal decomposition of

Manuscript received February 3, 1997; revised March 12, 2000. This work was supported in part by the U.S. Air Force under Contract F08635-89-C-0134. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Stephen E. Reichenbach.

P. Sussner is with the Institute of Mathematics, Statistics, and Scientific Computation, State University of Campinas, 13083 Campinas, S.P., Brazil (e-mail: sussner@ime.unicamp.br).

G. X. Ritter is with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: ritter@cise.ufl.edu).

Publisher Item Identifier S 1057-7149(00)06003-6.

convex morphological templates for 4-connected meshes. However, all the above decomposition methods work only on binary morphological templates and do not extend to gray-scale morphological templates.

A very successful general theory for the decomposition of templates, in both the linear and morphological domain, evolved from the theory of image algebra [5], [6], [17], [20] which provides an algebraic foundation for image processing and computer vision tasks. In this setting, Ritter and Gader [6], [18] presented efficient methods for decomposing discrete Fourier transform templates. Zhu and Ritter [28] employ the general matrix product to provide novel computational methods for computing the fast Fourier transform, the fast Walsh transform, the generalized fast Walsh transform, as well as a fast wavelet transform.

In image algebra, template decomposition problems, for both linear and morphological template operations, can be reformulated in terms of corresponding matrix or polynomial factorization. Manseur and Wilson [14] used matrix as well as polynomial factorization techniques to decompose two-dimensional linear templates of size $m \times n$ into sums and products of 3×3 templates. Li [12] was the first to investigate polynomial factorization methods for morphological templates. He provides a uniform representation of morphological templates in terms of polynomials, thus reducing the problem of decomposing a morphological template to the problem of factoring the corresponding polynomials. His approach provides for the decomposition of one-dimensional (1-D) morphological templates into factors of two-point templates. Crosby [2] extends Li's method to 2-D morphological templates.

Davidson [4] proved that any morphological template has a weak local decomposition for mesh-connected array processors. Davidson's existence theorem provides a theoretical foundation for morphological template decomposition, yet the algorithm conceived in its constructive proof is not very efficient. Takriti and Gader formulate the general problem of template decomposition as optimization problems [8], [26]. Sussner *et al.* [22] use a similar approach to solve the even more general problem of morphological template approximation. As proven in [23], these problems are NP-complete. Therefore, researchers usually try to exploit the special structure of certain morphological templates in order to find decomposition algorithms. For example, Li and Ritter [13] provide very simple matrix techniques for decomposing binary as well as gray-scale linear and morphological convex templates. A separable template is a template that can be expressed in terms of two 1-D templates consisting of a row and a column template. Gader [7] uses matrix methods for decomposing any gray-scale morphological template into a sum of a separable template and a totally nonseparable template. If the original template is separable, then Gader's decomposition yields a separable decomposition. If the original template is not separable, then his method yields the closest separable template to the original in the mean square sense.

The strong decomposition of a rank 1 template is an easy task both in the linear and in the nonlinear domain [13]. O'Leary [15] showed that any linear template of rank 1 can be factored exactly into a product of 3×3 linear templates. Templates of higher rank are usually not as efficiently decomposable. However, the LU factorization yields a proven method for determining a rank-based decomposition of a linear template of arbitrary, unknown rank [9], [17].

In an earlier paper, we introduced a polynomial time algorithm for the rank based decomposition of morphological templates of rank 2 [24]. This paper develops a heuristic algorithm for the rank-based decomposition of morphological templates of arbitrary rank.

This paper is organized as follows: First, we introduce the reader to the language of image algebra. This mathematical theory is suited to describe all image processing operands and operations in a translucent manner. Since we focus on morphological or nonlinear image processing, we proceed with a brief description of the algebraic structures defined in the nonlinear domain. Relating matrices to rectangular templates, we establish a new rank method for the morphological decomposition of matrices and rectangular templates in Section IV. We conclude with computational results and suggestions for further research.

II. Some Image Algebra Background

Image algebra is a *heterogeneous* or *many-valued* algebra in the sense of Birkhoff and Lipson [1], [17], with multiple sets of operands and operators. In a broad sense, image algebra is a mathematical theory concerned with the transformation and analysis of images. Although much emphasis is focused on the analysis and transformation of digital images, the main goal is the establishment of a comprehensive and unifying theory of image transformations, image analysis, and image understanding in the discrete as well as the continuous domain [17], [19], [20]. In this paper, we restrict our attention only to the notations and operations that are necessary for establishing the results mentioned in the introduction. Hence, our focus is on morphological image algebra operations.

Henceforth, let $\mathbf{X}$ be a subset of the digital plane $\mathbb{Z}^2 = \{(i, j): i, j \in \mathbb{Z}\}$, where $\mathbb{Z}$ denotes the set of integers. For any set $\mathbb{F}$, we denote the set of all functions from $\mathbf{X}$ into $\mathbb{F}$ by $\mathbb{F}^{\mathbf{X}}$. The set $\mathbb{F}$ of interest will be the real numbers with the symbol $-\infty$ appended. More precisely, $\mathbb{F} = \mathbb{R}_{-\infty} = \mathbb{R} \cup \{-\infty\}$, where $\mathbb{R}$ denotes the set of real numbers. The algebraic system associated with $\mathbb{R}_{-\infty}$ will be the lattice ordered semi-group $(\mathbb{R}_{-\infty}, \vee, +)$. We use the symbols $\vee$ and $\wedge$ to denote the binary operations of maximum and minimum, respectively. For any real number x, the number $\lfloor x \rfloor$ is defined as the largest integer n such that $n \leq x$.

Images and Templates: From the image algebra perspective, images are considered to be functions and templates are viewed as functions whose values are images. In particular, an $\mathbb{R}_{-\infty}$-valued image $\mathbf{a}$ over the point set $\mathbf{X}$ is a function $\mathbf{a}: \mathbf{X} \to \mathbb{R}_{-\infty}$ (i.e. $\mathbf{a} \in \mathbb{R}_{-\infty}^{\mathbf{X}}$), while an $\mathbb{R}_{-\infty}$-valued template $\mathbf{t}$ on $\mathbf{X}$ is a function $\mathbf{t}: \mathbf{X} \to \mathbb{R}_{-\infty}$ (i.e. $t \in (\mathbb{R}_{-\infty}^{X})^{X}$). For notational convenience, we define $\mathbf{t}_{\mathbf{y}}$ as $\mathbf{t}(\mathbf{y})$ for all $\mathbf{y} \in X$.

Our focus will be on *translation invariant* $\mathbb{R}_{-\infty}$-valued templates over $\mathbf{X}$ since gray-scale structuring elements can be realized by these templates. A template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ is called *translation invariant* if and only if

$$\mathbf{t}_{\mathbf{y}+\mathbf{z}}(\mathbf{x}+\mathbf{z}) = \mathbf{t}_{\mathbf{y}}(\mathbf{x}) \qquad \forall\, \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbf{X} = \mathbb{Z}^2 \qquad (1)$$

whenever $\mathbf{y}+\mathbf{z}$ and $\mathbf{x}+\mathbf{z}$ are elements of $\mathbf{X}$. The *support* of a template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ at a point $\mathbf{y}$ is denoted by $S(\mathbf{t_y})$ and defined as follows:

$$S(\mathbf{t_y}) = \{\mathbf{x} \in \mathbf{X}\colon \mathbf{t_y}(\mathbf{x}) \neq -\infty\}. \tag{2}$$

A translation invariant template $\mathbf{t}$ is called *rectangular*, if $S(\mathbf{t_y})$ forms a rectangular discrete array.

Example: Let $\mathbf{r} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ be the translation invariant template which is determined at each point $\mathbf{y} \in \mathbf{X}$ by the following function values of $\mathbf{x} = (x_1, x_2) \in \mathbf{X}$:

$$r_y(x) = \begin{cases} 2, & \text{if } \mathbf{x} = \mathbf{y} - (0,1) \\ 6, & \text{if } \mathbf{x} = \mathbf{y} \\ 3, & \text{if } \mathbf{x} = \mathbf{y} + (0,1) \\ \mathbf{r_y}(x + (1,0)) + 4, & \text{if } x_1 + 1 = x \\ \mathbf{r_y}(\mathbf{x} - (1,0)) + 1, & \text{if } x_1 - 1 = x \\ -\infty, & \text{else.} \end{cases} \tag{3}$$

If $\mathbf{y} = (x, y)$, we can visualize the rectangular template $\mathbf{r}$ as shown in Fig. 1.

Additive Maximum Operations: The basic operations of addition and maximum on $\mathbb{R}_{-\infty}$ induce pixelwise operations on $\mathbb{R}_{-\infty}$-valued images and templates [20], [19]. These operations can also be used to define lattice based convolution operators. In particular, forming the *additive maximum* ("$\boxed{\vee}$") of an image $\mathbf{a} \in \mathbb{R}_{-\infty}^{\mathbf{X}}$ and a template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ results in the image $\mathbf{a} \boxed{\vee}\, \mathbf{t} \in \mathbb{R}_{-\infty}^{\mathbf{X}}$, which is determined by the following function values:

$$(\mathbf{a} \boxed{\vee}\, \mathbf{t})(\mathbf{y}) = \bigvee_{\mathbf{x} \in \mathbf{X}} \mathbf{a}(\mathbf{x}) + \mathbf{t_y}(\mathbf{x}). \tag{4}$$

Clearly, each template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ defines a function

$$\begin{aligned} f_{\mathbf{t}} : \mathbb{R}_{-\infty}^{\mathbf{X}} &\to \mathbb{R}_{-\infty}^{\mathbf{X}} \\ \mathbf{a} &\mapsto \mathbf{a} \boxed{\vee}\, \mathbf{t}. \end{aligned} \tag{5}$$

The dual operation of *additive minimum* ("$\boxed{\wedge}$") between images and templates can be defined in a similar fashion by interchanging the operation $\vee$ with the operation $\wedge$. Using the terminology of mathematical morphology, the additive maximum operation expresses standard gray-scale dilation while the additive minimum operation expresses standard gray-scale erosion [10].

The notion of additive minimum for combining an image with a template can be extended for combining templates. The *additive maximum* of a template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ and a template $\mathbf{s} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ is defined as the template $\mathbf{r} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ which determines $f_{\mathbf{s}} \circ f_{\mathbf{t}}$, the composition of $f_{\mathbf{t}}$ followed by $f_{\mathbf{s}}$. Specifically,

$$(\mathbf{s} \boxed{\vee}\, \mathbf{t})_{\mathbf{y}}(\mathbf{z}) = \bigvee_{\mathbf{x} \in \mathbf{X}} (\mathbf{t_x}(\mathbf{z}) + \mathbf{s_y}(\mathbf{x})) \qquad \forall\, \mathbf{y}, \mathbf{z} \in \mathbf{X}. \tag{6}$$

These relationships induce associative properties for image and template operations which we provide after the following examples.

Example: Many image processing techniques such as the Rolling Ball Algorithm and algorithms for noise removal employ morphological image-template products of the form $\mathbf{a} \boxed{\vee}\, \mathbf{t}$ or $\mathbf{a} \boxed{\wedge}\, \mathbf{t}$ [19].

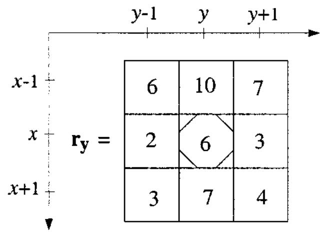


Fig. 1. Support of the template r at point y. The hashed cell indicates the location of the target point $y = (x, y)$.

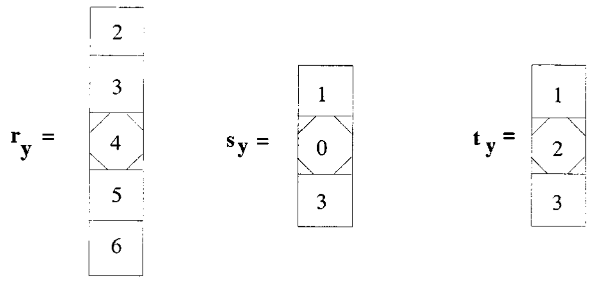


Fig. 2. Template $\mathbf{r}$ constitutes the additive maximum of the templates $\mathbf{s}$ and the template $\mathbf{t}$.

Example: The templates $\mathbf{r}, \mathbf{s}, \mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ in Fig. 2 satisfy $\mathbf{r} = \mathbf{s} \boxed{\vee}\, \mathbf{t}$.

Some Properties of Image and Template Operations: The following associative and distributive laws hold for an arbitrary image $\mathbf{a} \in \mathbb{R}_{-\infty}^{\mathbf{X}}$ and arbitrary templates $\mathbf{t} \in (R_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ and $\mathbf{s} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$:

$$\begin{aligned} \mathbf{a} \boxed{\vee} (\mathbf{s} \boxed{\vee}\, \mathbf{t}) &= (\mathbf{a} \boxed{\vee}\, \mathbf{s}) \boxed{\vee}\, \mathbf{t}, \\ \mathbf{a} \boxed{\vee} (\mathbf{s} \vee \mathbf{t}) &= (\mathbf{a} \boxed{\vee}\, \mathbf{s}) \vee (\mathbf{a} \boxed{\vee}\, \mathbf{t}). \end{aligned} \tag{7}$$

These results establish the importance of template decomposition.

Strong Decompositions of Templates: A sequence of templates $(\mathbf{t}^1, \cdots, \mathbf{t}^k)$ in $(\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ is called a *(strong) decomposition* (with respect to the operation "$\boxed{\vee}$") of a template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ if $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ can be written in the form

$$\mathbf{t} = \mathbf{t}^1 \boxed{\vee}\, \mathbf{t}^2 \boxed{\vee} \cdots \boxed{\vee}\, \mathbf{t}^k. \tag{8}$$

In the special case where $k = 2$, we speak of a *separable* template if the support of $\mathbf{t}^1$ is a 1-D vertical array and the support of $\mathbf{t}^2$ is a 1-D horizontal array.

Example: The template $\mathbf{r} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ given in Fig. 1 represents a separable template since this template decomposes into a vertical strip template $\mathbf{s} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ and a horizontal strip template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ as shown in Fig. 3.

Weak Decompositions of Templates: A sequence of templates $(\mathbf{t}^1, \cdots, \mathbf{t}^{k_n})$ in $(\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ together with a strictly increasing sequence of natural numbers $(k_1, \cdots, k_n)$ is called a *(weak) decomposition* (with respect to the operation "$\boxed{\vee}$") of a template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ if the template $\mathbf{t}$ can be represented as follows:

$$\mathbf{t} = (\mathbf{t}^1 \boxed{\vee} \cdots \boxed{\vee}\, \mathbf{t}^{k_1}) \vee \cdots \vee (\mathbf{t}^{k_{n-1}+1} \boxed{\vee} \cdots \boxed{\vee}\, \mathbf{t}^{k_n}). \tag{9}$$

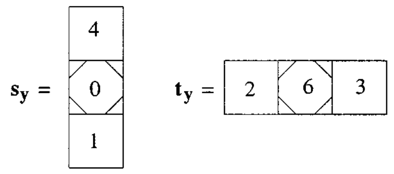


Fig. 3. Pictorial representation of a column template $\mathbf{s}$ and a row template $\mathbf{t}$.

We say $(\mathbf{s}^1, \cdots, \mathbf{s}^k)$ is a weak decomposition of a rectangular template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ into separable templates if each $\mathbf{s}^i$, where $i = 1, \cdots, k$, is separable and $\mathbf{t} = \mathbf{s}^1 \vee \cdots \vee \mathbf{s}^k$.

Correspondence Between Rectangular Templates and Matrices: Note that there is a natural bijection ϕ from the space of all $m \times n$ matrices over $\mathbb{R}_{-\infty}$ into the space of all rectangular $m \times n$ templates in $(\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$.

Let $\mathbf{y} = (x, y) \in \mathbf{X}$ be arbitrary and $\mathbf{x} = (x_1, x_2) \in \mathbf{X}$ be such that

$$x_1 = x - \left\lfloor \frac{m}{2} \right\rfloor, \quad x_2 = y - \left\lfloor \frac{n}{2} \right\rfloor. \tag{10}$$

The image of a matrix $\mathbf{A} \in (\mathbb{R}_{-\infty})^{m \times n}$ under ϕ is defined to be the template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ which satisfies

$$\begin{aligned} &\mathbf{t_y}(x_1 + i - 1, x_2 + j - 1) = a_{ij} \\ &\qquad \forall\, i = 1, \cdots, m, \;\forall\, j = 1, \cdots n, \\ &\mathbf{t_y}(y_1, y_2) = -\infty \\ &\qquad \forall\, y_1, y_2 \notin [x_1, x_1 + m - 1] \times [x_2, x_2 + n - 1]. \end{aligned} \tag{11}$$

Henceforth, we restrict our attention to rectangular templates whose target pixel is centered, i.e., rectangular templates of the above form.

The theory of minimax algebra [3] examines the algebraic structures arising from the lattice operations "maximum," "minimum," and "addition" including the space of all matrices over $\mathbb{R}_{-\infty}$ together with the operation "additive maximum." The natural correspondence between rectangular templates in $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ and matrices over $\mathbb{R}_{-\infty}$ allows us to use a minimax algebra approach in order to study the weak decomposability of rectangular templates into separable templates.

Example: Let $\mathbf{A} \in \mathbb{R}^{3 \times 3}$ be the matrix and $\mathbf{u}$, $\mathbf{v}$ the vectors

$$\mathbf{A} = \begin{pmatrix} 6 & 10 & 7 \\ 2 & 6 & 3 \\ 3 & 7 & 4 \end{pmatrix}, \quad \mathbf{u} = \begin{pmatrix} 4 \\ 0 \\ 1 \end{pmatrix}, \quad \mathbf{v} = (2, 6, 3). \tag{12}$$

The function ϕ maps $\mathbf{A}$ to the square template $\mathbf{r} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ in Fig. 1, and it maps the column vector $\mathbf{u}$ to the column template $\mathbf{s} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ and the row vector $\mathbf{v}$ to the row template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ in Fig. 3. The reader may want to verify these mappings using (10) and (11).

III. Ranks of Matrices in Minimax Algebra

In this section, we develop a new notion of matrix rank within the mathematical framework of minimax algebra. We relate this concept of matrix rank to the one given by Cuninghame-Green [3] and derive the notion of the rank of a morphological template.

Algebraic Structures and Operations in Minimax Algebra: The mathematical theory of minimax algebra deals with algebraic structures such as bands, belts and blogs. For example, $\mathbb{R}_{-\infty}$ together with the operations of maximum ("$\vee$") and addition forms a belt. Cuninghame-Green defines the matrix rank for matrices over certain subsets of the blog $\mathbb{R}_{\pm\infty}$. For our purposes it suffices to consider $\mathbb{R}$, the finite elements of $\mathbb{R}_{\pm\infty}$.

Operations such as the maximum ("$\vee$"), the minimum ("$\wedge$"), and the addition on $\mathbb{R}$ induce entrywise operations on $\mathbb{R}^{m \times n}$, the set of all $m \times n$ matrices over $\mathbb{R}$. Minimax algebra also defines compound operations such as "$\boxdot$"—pronounced "additive maximum"—from $\mathbb{R}^{m \times k} \times \mathbb{R}^{k \times n}$ into $\mathbb{R}^{m \times n}$, an operation similar to the regular matrix product known from linear algebra. (An obvious dual of this operation is provided by the "additive minimum" operation.) Given matrices $\mathbf{A} \in \mathbb{R}^{m \times k}$ and $\mathbf{B} \in \mathbb{R}^{k \times n}$, the additive maximum $\mathbf{C} = \mathbf{A} \boxdot \mathbf{B} \in \mathbb{R}^{m \times n}$ is determined by

$$c_{ij} = \bigvee_{l=1}^{k} (a_{ik} + b_{kj}) \qquad \forall\, i = 1, \cdots, m, \;\forall\, j = 1, \cdots, n. \tag{13}$$

If $\mathbf{A}$ is a matrix in $\mathbb{R}^{m \times n}$ and if $\mathbf{u}^i$ are column vectors in $\mathbb{R}^{m \times 1}$ and $\mathbf{v}^i$ are row vectors in $\mathbb{R}^{1 \times n}$ for $i = 1, \cdots, k$, then the following equivalence holds for the corresponding rectangular template $\phi(\mathbf{A})$, the vertical strip templates $\phi(\mathbf{u}^i)$, and the horizontal strip templates $\phi(\mathbf{v}^i)$:

$$\mathbf{A} = \bigvee_{i=1}^{k} (\mathbf{u}^i \boxdot \mathbf{v}^i) \Leftrightarrow \phi(\mathbf{A}) = \bigvee_{i=1}^{k} (\phi(\mathbf{u}^i) \boxdot \phi(\mathbf{v}^i)). \tag{14}$$

Linear Dependence of Vectors [3]: A vector $\mathbf{v} \in \mathbb{R}^n$ is said to be *linearly dependent* on the vectors $\mathbf{v}^1, \cdots, v^k \in \mathbb{R}^n$ if and only if there exists a (not necessarily unique) set of scalars $c_i \in \mathbb{R}$, $i = 1, \cdots, k$, such that

$$\mathbf{v} = \bigvee_{i=1}^{k} (c_i + \mathbf{v}^i). \tag{15}$$

Otherwise, the vector $\mathbf{v} \in \mathbb{R}^n$ is called *linearly independent* from the vectors $\mathbf{v}^1, \cdots, \mathbf{v}^k \in \mathbb{R}^n$. The vectors $\mathbf{v}^1, \cdots, \mathbf{v}^k \in \mathbb{R}^n$ are *linearly independent* if each one of them is linearly independent from the others.

Example: Consider the following elements of $\mathbb{R}^3$:

$$\mathbf{v} = \begin{pmatrix} 5 \\ -1 \\ 7 \end{pmatrix}, \quad \mathbf{v}^1 = \begin{pmatrix} 3 \\ -5 \\ 2 \end{pmatrix}, \quad \mathbf{v}^2 = \begin{pmatrix} -6 \\ 1 \\ 9 \end{pmatrix}. \tag{16}$$

Since $\mathbf{v} = [2 + \mathbf{v}^1] \vee [(-2) + \mathbf{v}^2]$, the vector $\mathbf{v}$ is linearly dependent on $\mathbf{v}^1$ and $\mathbf{v}^2$.

Rank of a Matrix: The *(separable) rank* of a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is denoted by rank$(\mathbf{A})$ and defined as the minimal number r of column vectors $\mathbf{u}^1, \cdots, \mathbf{u}^r \in \mathbb{R}^{m \times 1}$ and row vectors $\mathbf{v}^1, \cdots, \mathbf{v}^r \in \mathbb{R}^{1 \times n}$ which permit a representation of A in the following form:

$$\mathbf{A} = \bigvee_{i=1}^{r} (\mathbf{u}^i \boxdot \mathbf{v}^i). \tag{17}$$

A representation of this form is called a *rank decomposition* or *separable decomposition* of $\mathbf{A}$. We say $\mathbf{A}$ is a *separable matrix* (with respect to the operation $\boxtimes$) if $\operatorname{rank}(\mathbf{A}) = 1$.

Rank of a Rectangular Template: If $\mathbf{t} = \phi(\mathbf{A})$ for some real-valued matrix $\mathbf{A}$, then we define the *rank of the template* $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ as the rank of $\mathbf{A}$.

Our interest in matrix and template ranks in the nonlinear domain is motivated by the problem of morphological template decomposition since the rank of a morphological template $\mathbf{t} \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ represents the minimal number of separable templates whose maximum is $\mathbf{t}$ or, equivalently, the minimal number r of column templates $\mathbf{r}^i \in (\mathbb{R}_{-\infty}^{\mathbf{X}})^{\mathbf{X}}$ and row templates $\mathbf{s}^i \in (\mathbb{R}_{-\infty}^{X})^{X}$ such that $\mathbf{t} = \bigvee_{i=1}^{r}(\mathbf{r}^i \boxtimes \mathbf{s}^i)$. Cuninghame-Green provides a different definition of matrix rank in [3].

IV. Matrix Decompositions in Minimax Algebra

As an introduction to the general problem of rank-based matrix decomposition in minimax algebra, we would like to present a short review of some results which have been established before. The reader should bear in mind the consequences for the corresponding rectangular morphological templates.

Theorem 1: If a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ has a representation $\mathbf{A} = \bigvee_{l=1}^{k}(u^l \boxtimes v^l)$ in terms of column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$ and row vectors $\mathbf{v}^l \in \mathbb{R}^{1\times n}$, where $l = 1, \cdots, k$, then A can be expressed in the following form:

$$\mathbf{A} = \bigvee_{l=1}^{k}(\mathbf{w}^l \boxtimes \mathbf{v}^l) \tag{18}$$

where $w^l \in \mathbb{R}^{m\times 1}$ is given by

$$w_i^l = \bigwedge_{j=1}^{n}(a_{ij} - v_j^l) \quad \forall i = 1, \cdots, m. \tag{19}$$

Similarly, given $\mathbf{A}$ and the k column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$, we can compute row vectors $\mathbf{w}^l \in \mathbb{R}^{1\times n}$ for $l = 1, \cdots, k$ such that $A = \bigvee_{l=1}^{k}(u^l \boxtimes w^l)$

$$w_j^l = \bigwedge_{i=1}^{m}(a_{ij} - u_i^l) \quad \forall j = 1, \cdots, n. \tag{20}$$

Remark: Theorem 1 implies that, for any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ of separable rank k, it suffices to know the row vectors $\mathbf{v}^l \in \mathbb{R}^{1\times n}$, $l = 1, \cdots, k$ which permit a weak decomposition of $\mathbf{A}$ into k separable matrices in order to determine a representation of $\mathbf{A}$ in the form

$$\mathbf{A} = \bigvee_{l=1}^{k}(\mathbf{w}^l \boxtimes \mathbf{v}^l), \quad \text{where} \quad \mathbf{w}^l \in \mathbb{R}^{m\times 1} \quad \forall\, l = 1, \cdots, k. \tag{21}$$

From now on we use the notation $\mathbf{a}(i)$, $i = 1, \cdots, m$ to denote the ith row vector of an arbitrary matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, and we use the notation $\mathbf{a}[j]$, $j = 1, \cdots, n$, to denote the jth column vector of $\mathbf{A}$.

A theorem by Li and Ritter [13] allows for the following elegant reformulation.

Theorem 2: If $\mathbf{A} \in \mathbb{R}^{m\times n}$ is a separable matrix and $1 \leq i_0 \leq m$ is an arbitrary index then each row vector $\boldsymbol{a}(i)$ is linearly dependent on the i_0th row vector $\boldsymbol{a}(i_0)$ of $\boldsymbol{A}$.

Assuming that every row vector of a given matrix $\mathbf{A}$ is linearly dependent on $\mathbf{a}(1)$, the matrix $\mathbf{A}$ is separable and Theorem 2 yields a strong decomposition $\mathbf{A} = \mathbf{w}^1 \boxtimes \mathbf{a}(1)$, where $\mathbf{w}^1 \in \mathbb{R}^{m\times 1}$. We showed in an earlier paper [24] that Theorem 2 can be generalized in a natural way to include matrices of rank 2:

Theorem 3: A matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ has rank 2 if and only if there are two row vectors of $\mathbf{A}$ on which all other row vectors depend (linearly).

Theorem 3 provides for a straightforward algorithm to determine if a given matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ is of rank ≤ 2 in the minimax algebra sense. Consider each tuple of row vectors of $\mathbf{A}$ and test all row vector of $\mathbf{A}$ for linear dependence on this vector tuple. If the matrix rank is indeed less than 3, then this algorithm also computes a tuple of vector pairs into which the matrix can be decomposed.

A similar theorem does not hold for matrices of separable rank $k \geq 3$. This fact is expressed by the following theorem.

Theorem 4: For every natural number $k \geq 3$ there are matrices over $\mathbb{R}_{-\infty}$ which are weakly $\boxtimes$-decomposable into a product of k vector pairs, but not all of whose row vectors are linearly dependent on a single k-tuple of their row vectors.

We showed in a previous paper that the general problem of determining rank decompositions of matrices is NP-complete [23]. However, the NP-completeness of a certain problem does not preclude the existence of an efficient algorithm for solving arbitrary instances of this problem. For matrices having relatively small rank compared to their size, we suggest the following heuristic algorithm (Algorithm $\mathcal{H}$). This heuristic is based on the following observations:

Remark: Suppose that $\mathbf{A} \in \mathbb{R}^{m\times n}$, whose rank is an unknown integer r. Hence, $\mathbf{A}$ can be represented as the maximum of matrices $\mathbf{A}^l \in \mathbb{R}^{m\times n}$, where $l = 1, \cdots, r$ and $\mathbf{A}^l = \mathbf{x}^l \boxtimes \mathbf{y}^l$ for some real-valued column vectors $\mathbf{x}^l$ of length m and some real-valued row vectors $\mathbf{y}^l$ of length n. According to Theorem 2, the separability of the matrix $\mathbf{A}^l$ induces the equality of the differences $a_{ij_1}^l - a_{ij_2}^l$ for all $i = 1, \cdots, m$ and for arbitrary, but fixed $l \in \{1, \cdots, r\}$ and $j_1, j_2 \in \{1, \cdots, n\}$. On the other hand, if the differences $a_{ij_1} - a_{ij_2}$ are all equal for $i \in I \subseteq \{1, \cdots, m\}$, it seems reasonable to assume that there exists an index l such that $a_{ij_1} = a_{ij_1}^l = x_i^l + y_{j_1}^l$ and $a_{ij_2} = a_{ij_2}^l = x_i^l + y_{j_2}^l$ for most $i \in I$ provided that r is relatively small and that $|\{a_{ij}: i = 1, \cdots, m; j = 1, \cdots, n\}|$ is relatively large compared to r. This assumption and Lemmas 1 and 2 of the Appendix play an important role in Algorithm $\mathcal{H}$ which intends to determine r as well as column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$ and row vectors $\mathbf{v}^l \in \mathbb{R}^{1\times n}$ such that

$$\mathbf{A} = \bigvee_{l=1}^{r}(\mathbf{u}^l \boxtimes \mathbf{v}^l). \tag{22}$$

Remark: Assume that $\mathbf{A} = \bigvee_{l=1}^{k}(\mathbf{x}^l \boxtimes \mathbf{y}^l)$, where x^l and y^l are unknown real-valued vectors. Algorithm $\mathcal{H}$ constructs vectors $\mathbf{u}^l$ and $\mathbf{v}^l$ for $l = 1, \cdots, k$ such that $\mathbf{u}^l \boxtimes \mathbf{v}^l \leq \mathbf{A}$. The maximum $\bigvee_{l=1}^{k}(\mathbf{u}^l \boxtimes \mathbf{v}^l)$ reaches $\mathbf{A}$, i.e., $\bigvee_{l=1}^{k}(\mathbf{u}^l \boxtimes \mathbf{v}^l) = \mathbf{A}$, if there exist sets I^l and J^l for all $l = 1, \cdots, k$ such that the following conditions hold:

$$\begin{aligned} I^l \times J^l &\subseteq \{(i, j)\colon x_i^l + y_j^l = a_{ij}\} \\ I^l \times J^l &\subseteq \{(i, j)\colon x_i^l + y_j^l \le u_i^l + v_j^l\} \\ \bigcup_{l=1}^{k}(I^l \times J^l) &= \{1, \cdots, m\} \times \{1, \cdots, n\}. \end{aligned} \tag{23}$$

Lemmas 1 and 2 of the Appendix provide some further insight into these matters. Additionally, by combining Lemmas 1 and 2 we establish a theorem (Theorem 8) that should lead to an even better understanding of Algorithm $\mathcal{H}$.

Algorithm $\mathcal{H}$: Let $\mathbf{A} \in \mathbb{R}^{m\times n}$. This algorithm determines a parameter $k \ge \text{rank}(\mathbf{A})$ and progressively constructs column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$ and row vectors $\mathbf{v}^l \in \mathbb{R}^{1\times n}$ such that

$$\mathbf{A} = \bigvee_{l=1}^{k}(\mathbf{u}^l \boxtimes \mathbf{v}^l). \tag{24}$$

Furthermore, this algorithm generates a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ which has the following property:

$$\begin{aligned} &m_{ij} = l \Rightarrow u_i^l + v_j^l = a_{ij} \\ &\qquad \forall\, i = 1, \cdots, m;\, \forall\, j = 1, \cdots, n;\, \forall\, l = 1, \cdots, k. \end{aligned} \tag{25}$$

Informally speaking, each entry m_{ij} represents a label indicating which index $l = 1, \cdots, k$ maximizes the sum $u_i^l + v_j^l$.

Initialize the matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ to be the zero matrix $0_{m\times n}$ and let $l = 0$. As long as the matrix $\mathbf{M}$ has a zero entry, we increment the parameter l by 1, and perform the following steps. Otherwise $l = k$ and the algorithm stops.

Step 1) *(Find the most frequent tuple difference):* Determine a strictly increasing sequence of maximal length $(i_1, i_2, \cdots, i_p)$ such that there exist two different column indices j_1 and j_2 having the following property:

$$\begin{aligned} &a_{i_1 j_1} - a_{i_1 j_2} = a_{i_2 j_1} - a_{i_2 j_2} = \cdots = a_{i_p j_1} - a_{i_p j_2} \\ &\quad \text{and} \quad m_{i_t j_1} = 0 \text{ or } m_{i_t j_2} = 0\ \forall\, t = 1, \cdots, p. \end{aligned} \tag{26}$$

For any $t = 1, \cdots, p$ satisfying $m_{i_t j_1} = m_{i_t j_2} = 0$, we set $m_{i_t j_1} = m_{i_t j_2} = l$. If either one of the entries $m_{i_t j_1}$ or $m_{i_t j_2}$ differs from 0 for all $t = 1, \cdots, p$, then set $m_{i_t j_1} = m_{i_t j_2} = l$ for all $t = 1, \cdots, p$.

Step 2) *(Extend the markings of found tuples horizontally):* For all $i \in \{i_1, \cdots, i_p\}$ and all $j = 1, \cdots, n$, we set $m_{ij} = l$ if the following two conditions both hold:

a) $m_{ij_1} = l$.

b) $a_{ij} - a_{ij_1} = \bigwedge_{z=1}^{m}(a_{zj} - a_{zj_1})$.

Step 3) *(Create preliminary row vectors $\tilde{\mathbf{v}}^l \in \mathbb{R}_{-\infty}^{m\times 1}$):* Initialize the vector $\tilde{\mathbf{v}}^l \in \mathbb{R}_{-\infty}^{m\times 1}$ by setting $\tilde{v}_j^l = -\infty$ for all $j = 1, \cdots, n$. Let $s = \min\{i \in \{i_1, \cdots, i_p\}\colon m_{ij_1} = l\}$. For all $j = 1, \cdots, n$ such that there exists an index $i \in \{i_1, \cdots, i_p\}$ satisfying $m_{ij} = l$, change the value of $\tilde{v}_j^l$ to $a_{ij} + a_{sj_1} - a_{ij_1}$, where $i \in \{i_1, \cdots, i_p\}$. Note that the value of $\tilde{v}_j^l$ does not depend on the choice of an element i of the set $\{i_1, \cdots, i_p\} \cap \{i = 1, \cdots, m\colon m_{ij} = l\}$ since condition b) of STEP 2 holds for all such i.

Step 4) *(Extend horizontal markings vertically and create column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$):* Define the column vector $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$ as follows:

$$u_i^l = \bigwedge_{j=1}^{n}(a_{ij} - \tilde{v}_j^l) \tag{27}$$

Note that u_i^l adopts a finite value for all $i = 1, \cdots, m$, since $\mathbf{A}$ is a finite matrix and $\tilde{v}_{j_1}^l$ is finite. Increase the values of all zero entries of $\mathbf{M}$ to l whenever $a_{ij} - \tilde{v}_j^l = u_i^l$.

Step 5) *(Final horizontal extension of markings and creation of final vectors $v^l \in \mathbb{R}^{m\times 1}$):* For all $j = 1, \cdots, n$ such that $\tilde{v}_j^l$ is finite, we equate v_j^l with $\tilde{v}_j^l$. Otherwise, we define v_j^l as $\bigwedge_{i=1}^{m}(a_{ij} - u_i^l)$. Any remaining zero entry m_{ij} is set to l if $a_{ij} = u_{ij} + v_j^l$.

Example: Let us first illustrate how Algorithm $\mathcal{H}$ works with a simple example. Suppose $\mathbf{A} \in \mathbb{R}^{3\times 4}$ is the following matrix:

$$\mathbf{A} = \begin{pmatrix} 8 & 6 & 3 & 4 \\ 5 & 3 & 3 & 4 \\ 6 & 6 & 7 & 8 \end{pmatrix}. \tag{28}$$

A matrix $\mathbf{M} \in \mathbb{R}^{3\times 4}$ is initialized as $0_{3\times 4}$. Then we execute Step 1–Step 5 for the parameter $l = 1$.

Since $a_{i3} - a_{i4} = -1$ for all $i = 1, 2, 3$, we set $(i_1, \cdots, i_p) = (1, 2, 3)$, $j_1 = 3$, and $j_2 = 4$. The matrix $\mathbf{M} \in \mathbb{R}^{3\times 4}$ now has form

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}. \tag{29}$$

Considering $\bigwedge_{z=1}^{3}(a_{z1} - a_{z3})$ and $\bigwedge_{z=1}^{3}(a_{z2} - a_{z3})$, we recognize that

$$\begin{aligned} \bigwedge_{z=1}^{3}(a_{z1} - a_{z3}) &= (8-3) \wedge (5-3) \wedge (6-7) \\ &= -1 = a_{31} - a_{33} \end{aligned}$$

and

$$\begin{aligned} \bigwedge_{z=1}^{3}(a_{z2} - a_{z3}) &= (6-3) \wedge (3-3) \wedge (6-7) \\ &= -1 = a_{32} - a_{33}. \end{aligned} \tag{30}$$

Therefore

$$M = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}. \tag{31}$$

Initially, all entries of the vector $\tilde{v}^1$ are set to $-\infty$. Since $m_{13} = 1$, the parameter $s = 1$. When executing STEP 3, these entries adopt the following values:

$$\begin{aligned}\tilde{v}_1^1 &= a_{31} + a_{13} - a_{33} = 6 + 3 - 7 = 2.\\ \tilde{v}_2^1 &= a_{32} + a_{13} - a_{33} = 6 + 3 - 7 = 2.\\ \tilde{v}_3^1 &= a_{i3} + a_{13} - a_{i3} = a_{13} = 3 \qquad \forall\, i = 1, 2, 3.\\ \tilde{v}_4^1 &= a_{i4} + a_{13} - a_{i3} = (a_{i4} - a_{i3}) + a_{13} = 1 + 3 = 4\\ &\quad \forall\, i = 1, 2, 3. \end{aligned} \tag{32}$$

Step 4 computes a vector $\mathbf{u}^1 \in \mathbb{R}_{3\times 1}$ as follows:

$$\begin{aligned} u_1^1 &= \bigwedge_{j=1}^{4} (a_{1j} - \tilde{v}_j^1)\\ &= (8-2) \wedge (6-2) \wedge (3-3) \wedge (4-4) = 0,\\ u_2^1 &= \bigwedge_{j=1}^{4} (a_{2j} - \tilde{v}_j^1)\\ &= (5-3) \wedge (3-2) \wedge (3-3) \wedge (4-4) = 0,\\ u_3^1 &= \bigwedge_{j=1}^{4} (a_{3j} - \tilde{v}_j^1)\\ &= (6-2) \wedge (6-2) \wedge (7-3) \wedge (8-4) = 4. \end{aligned} \tag{33}$$

Since the vector $\tilde{\mathbf{v}}^1$ is finite, we have $\mathbf{v}^1 = \tilde{\mathbf{v}}^1$ in Step 5.

Now the parameter l changes to 2, since there are still some zero entries of M left. These zeros of M are eliminated in Step 1, since $a_{11} - a_{12} = a_{21} - a_{22} = 2$ and $m_{11} = m_{12} = m_{21} = m_{22} = 0$. Thus,

$$M = \begin{pmatrix} 2 & 2 & 1 & 1\\ 2 & 2 & 1 & 1\\ 1 & 1 & 1 & 1 \end{pmatrix}. \tag{34}$$

Note that Step 2–Step 5 effect no change in the matrix $\mathbf{M}$, since $\mathbf{M} > 0_{3\times 4}$. Step 3 and Step 4 create the following vectors $\tilde{\mathbf{v}}^2 \in \mathbb{R}^{1\times 4}$ and $\mathbf{u}^2 \in \mathbb{R}^{3\times 1}$:

$$\tilde{v}^2 = (\ 8 \quad 6 \quad -\infty \quad -\infty\ ), \quad \mathbf{u}^2 = \begin{pmatrix} 0\\ -3\\ -2 \end{pmatrix}. \tag{35}$$

Step 5 produces a row vector $\mathbf{v}^2 \in \mathbb{R}^{1\times 4}$, where $v_i^2 = \tilde{v}_i^2$ for $i = 1, 2$. The entries v_3^2 and v_4^2 are computed as follows:

$$\begin{aligned} v_3^2 &= \bigwedge_{i=1}^{3} (a_{i3} - u_i^2) = (3-0) \wedge (3-(-3)) \wedge (7-(-2)) = 3,\\ v_4^2 &= \bigwedge_{i=1}^{3} (a_{i4} - u_i^2) = (4-0) \wedge (4-(-3)) \wedge (8-(-2)) = 4. \end{aligned} \tag{36}$$

Since all entries of $\mathbf{M}$ are strictly greater than 0, Algorithm $\mathcal{H}$ finishes yielding the following result:

$$\begin{aligned} \mathbf{A} &= (\mathbf{u}^1 \boxtimes \mathbf{v}^1) \vee (\mathbf{u}^2 \boxtimes \mathbf{v}^2)\\ &= \left[\begin{pmatrix} 0\\ 0\\ 4 \end{pmatrix} \boxtimes (2 \quad 2 \quad 3 \quad 4) \right]\\ &\quad \vee \left[\begin{pmatrix} 0\\ -3\\ -2 \end{pmatrix} \boxtimes (8 \quad 6 \quad 3 \quad 4) \right]\\ &= \begin{pmatrix} 8 & 6 & 3 & 4\\ 5 & 3 & 3 & 4\\ 6 & 6 & 7 & 8 \end{pmatrix}. \end{aligned} \tag{37}$$

Theorem 5: Algorithm $\mathcal{H}$ is a polynomial time algorithm.

Proof: Let us analyze Step 1 through Step 5 for $\mathbf{A} \in \mathbb{R}^{m\times n}$ and for fixed $l \in \mathbb{N}$.

Since there are $(n(n-1)/2)$ different tuples of column indices j_1 and j_2, $m\cdot(n(n-1)/2)$ subtractions are needed in order to compute the differences $a_{ij_1} - a_{ij_2}$ for all $i = 1, \cdots, m$ and for all $j_1, j_2 = 1, \cdots, n$. For fixed j_1 and j_2, the comparison of the m real numbers $a_{ij_1} - a_{ij_2}$ takes $(m(m-1)/2)$ operations. Hence, these comparison operations are of computational complexity $O((1/4)\cdot n^2 m^2)$. All other operations of Step 1 and the operations of Step 2 through Step 5 have smaller computational complexity. Since the number of zero entries of $\mathbf{M} \in \mathbb{R}^{m\times n}$ decreases when performing Step 1 through Step 5, Algorithm $\mathcal{H}$ is a polynomial time algorithm.

Theorem 6: Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, Algorithm $\mathcal{H}$ produces an integer k, column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$ and row vectors $v^l \in \mathbb{R}^{1\times n}$ for $l = 1, \cdots, k$ such that

$$\mathbf{A} = \bigvee_{l=1}^{k} (\mathbf{u}^l \boxtimes \mathbf{v}^l). \tag{38}$$

Moreover, Algorithm $\mathcal{H}$ generates a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ such that the following relationship holds:

$$m_{ij} = l \Rightarrow u_i^l + v_j^l \geq a_{ij}. \tag{39}$$

Proof: Suppose that Algorithm $\mathcal{H}$ is applied to a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, yielding as an output an integer k, column vectors $\mathbf{u}^l \in \mathbb{R}^{m\times 1}$ and row vectors $v^l \in \mathbb{R}^{1\times n}$ for all $l = 1, \cdots, k$, as well as a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$.

Since the final matrix $\mathbf{M}$ belongs to the set $\{1, \cdots, k\}^{m\times n}$, it suffices to show that the following properties hold for all i, j, and l:

1) $u_i^l + v_j^l \leq a_{ij}$;
2) $m_{ij} = l \Rightarrow u_i^l + v_j^l \geq a_{ij}$.

The first property immediately follows from the fact that u_i^l equals $\bigwedge_{z=1}^{n} (a_{iz} - v_z^l)$, since $\bigwedge_{z=1}^{n} (a_{iz} - v_z^l) \leq a_{ij} - v_j^l$.

Suppose that $m_{ij} = l$ after completion of STEP 3. This assumption implies that $i \in \{i_1, \cdots, i_p\}$ and $v_j^l = \tilde{v}_j^l =$

$a_{ij}+a_{sj_1}-a_{ij_1}$. In this case, the following argumentation shows the validity of property 2:

$$\begin{aligned} u_i^l + v_j^l &= \bigwedge_{z=1}^{n} (a_{iz} - \tilde{v}_z^l) + v_j^l \\ &\geq \bigwedge_{z=1}^{n} (a_{iz} - (a_{iz} + a_{sj_1} - a_{ij_1})) + v_j^l \\ &= (a_{ij_1} - a_{sj_1}) + (a_{ij} + a_{sj_1} - a_{ij_1}) \\ &= a_{ij}. \end{aligned} \tag{40}$$

The inequality above holds for the following reason: Either $\tilde{v}_z^l = -\infty$ or $\tilde{v}_z^l = \bigwedge_{t=1}^{m}(a_{tz}-a_{tj_1})+a_{sj_1} \leq a_{iz}-a_{ij_1}+a_{sj_1}$ for all $i = 1, \cdots, n$.

Clearly $u_i^l + v_j^l = a_{ij}$, if the entry m_{ij} of the matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ has been set to l in Step 4 or in Step 5.

Theorem 7: Given a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ of rank 1, Algorithm $\mathcal{H}$ determines a column vector $\mathbf{u}^1 \in \mathbb{R}^{m\times 1}$ and a row vector $\mathbf{v}^1 \in \mathbb{R}^{1\times n}$, such that

$$\mathbf{A} = \mathbf{u}^1 \boxtimes \mathbf{v}^1. \tag{41}$$

Proof: By Theorem 2, each row vector $\mathbf{a}(i)$ of $\mathbf{A}$, where $i = 2, \cdots, n$, is linearly dependent on the first row vector $\mathbf{a}(1)$. Hence, there exist λ_i such that $\mathbf{a}(i) = \lambda_i + \mathbf{a}(1)$, which implies that the following equalities hold:

$$\begin{aligned} a_{11} - a_{12} &= -\lambda_i + a_{i1} - (-\lambda_i + a_{i2}) = a_{i1} - a_{i2} \\ &\forall i = 2, \cdots, n. \end{aligned} \tag{42}$$

Hence, Step 1 of Algorithm $\mathcal{H}$ results in a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ of the following form:

$$\mathbf{M} = \begin{pmatrix} 1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 0 & 0 & \cdots & 0 \end{pmatrix}. \tag{43}$$

Since the differences $a_{zj} - a_{z1}$, where z ranges from 1 to m, are all equal for arbitrary, but fixed $j \in \{1, \cdots, n\}$, all entries of $\mathbf{M}$ are set to 1 in Step 2. Step 3 yields a row vector $\tilde{\mathbf{v}}^1 = \mathbf{a}(1)$. In Step 4, we compute a column vector $\mathbf{u}^1$ as follows:

$$\begin{aligned} u_i^1 &= \bigwedge_{j=1}^{n} (a_{ij} - \tilde{v}_j^1) \\ &= \bigwedge_{j=1}^{n} (a_{ij} - a_{1j}) \\ &= a_{i1} - a_{11}. \end{aligned} \tag{44}$$

Hence, $\mathbf{u}^1 = \mathbf{a}[1] - a_{11}$. STEP 5 equates $\mathbf{v}^1$ with $\tilde{\mathbf{v}}^1$, and the algorithm finishes yielding $\mathbf{A} = \mathbf{u}^1 \boxtimes \mathbf{v}^1$.

Remark: An application of Gader's algorithm to a separable matrix will also yield a separable decomposition of the matrix [7]. In view of Theorem 2, the separability of a matrix is easy to recognize anyway. Thus, the intended use of Algorithm $\mathcal{H}$ is for matrices of rank ≥2. If the matrix is close to being separable and an approximation in terms of a separable matrix is desired, Gader's algorithm should be chosen. Algorithm $\mathcal{H}$ is best suited for finding exact outer product representations of matrices with small rank.

Theorem 8: Suppose $A \geq \mathbf{x} \boxtimes \mathbf{y}$ for a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, a column vector $\mathbf{x} \in \mathbb{R}^{m\times 1}$ and a row vector $\mathbf{y} \in \mathbb{R}^{1\times n}$. Let $\mathbf{u}$, $\tilde{\mathbf{v}}$, $\mathbf{v}$ denote the vectors $\mathbf{u}^l$, $\tilde{\mathbf{v}}^l$, $\mathbf{v}^l$ constructed from the matrix $\mathbf{A}$ in Algorithm $\mathcal{H}$ for an arbitrary, but fixed parameter l. *Let* j_1, j_2 be as in Algorithm $\mathcal{H}$. If $I \subseteq \{1, \cdots, m\}$ and $J \subseteq \{1, \cdots, n\}$ denote the sets

$$\begin{aligned} I = \{i : u_i + \tilde{v}_{j_1} &= x_i + y_{j_1} = a_{ij_1}, \\ u_i + \tilde{v}_{j_2} &= x_i + y_{j_2} = a_{ij_2}\} \end{aligned}$$

and

$$J = \left\{ j\colon a_{ij} - a_{ij_1} = \bigwedge_{z=1}^{m} (a_{z1} - a_{zj_1}) \text{ for some } i \in I \right\}, \tag{45}$$

then

$$\begin{aligned} a_{ij} \geq u_i + v_j &= u_i + \tilde{v}_j \geq x_i + y_j \\ &\forall i = 1, \cdots, m; \quad \forall j \in J. \end{aligned} \tag{46}$$

In particular, if $x_i + y_j = a_{ij}$ for some $i \in \{1, \cdots, m\}$ and some $j \in J$, then $u_i + v_j = a_{ij}$.

Remark: Suppose that $\mathbf{A}$ has representation $\mathbf{A} = \bigvee_{l=1}^{r}(\mathbf{x}^l \boxtimes \mathbf{y}^l)$, where $r = \operatorname{rank}(\mathbf{A})$. Theorem 8 provides conditions which guarantee that $x_i^l + y_j^l$ coincides with $u_i^l + v_i^l$ for certain i and j, where $\mathbf{u}^l$ and $\mathbf{v}^l$ are the vectors constructed in Algorithm $\mathcal{H}$. We obtain the desired result $\mathbf{A} = \bigvee_{l=1}^{r}(\mathbf{u}^l \boxtimes \mathbf{v}^l)$ if the following condition is satisfied: For all i and j, there exists an index $l \in \{1, \cdots, r\}$ depending on i and j such that

$$a_{ij} = x_i^l + y_j^l = u_i^l + v_j^l. \tag{47}$$

The interested reader may find the proof of Theorem 8 in the Appendix.

Example: The following example shows that although Algorithm $\mathcal{H}$ always determines a weak decomposition of a given matrix $\mathbf{A}$ into outer products of column vectors and row vectors, the number of these outer products is not always minimal. In other words, there are instances when the parameter r determined by Algorithm $\mathcal{H}$ exceeds the rank of the matrix $\mathbf{A}$. Let $\mathbf{A} \in \mathbb{R}^{4\times 5}$ be the following matrix.

$$\begin{aligned} \mathbf{A} &= \begin{pmatrix} -2 & 0 & 0 & -1 & 4 \\ 5 & 3 & 1 & 0 & 8 \\ 3 & 5 & 4 & 2 & 7 \\ 3 & 1 & 0 & -2 & 6 \end{pmatrix} \\ &= \begin{pmatrix} 0 \\ 1 \\ 3 \\ -5 \end{pmatrix} \boxtimes \begin{pmatrix} -5 & 0 & 0 & -1 & 4 \end{pmatrix} \\ &\vee \begin{pmatrix} -7 \\ 0 \\ -2 \\ -2 \end{pmatrix} \boxtimes \begin{pmatrix} 5 & 3 & 1 & 0 & 8 \end{pmatrix} \\ &\vee \begin{pmatrix} -1 \\ 0 \\ 4 \\ 0 \end{pmatrix} \boxtimes \begin{pmatrix} -3 & 1 & 0 & -2 & 0 \end{pmatrix}. \end{aligned} \tag{48}$$

Since $\mathbf{A}$ is weakly decomposable into three vector pairs, the rank of $\mathrm{rank}(\mathbf{A}) \leq 3$. Using Theorem 3, we can show that $\mathrm{rank}(\mathbf{A}) = 3$. However, Algorithm $\mathcal{H}$ yields $k = 4$ and computes the following column vectors $\mathbf{u}^1, \cdots, \mathbf{u}^4$ and row vectors $\mathbf{v}^1, \cdots, \mathbf{v}^4$ such that $\mathbf{A} = \bigvee_{l=1}^{4} \mathbf{u}^l \boxtimes \mathbf{v}^l$.

$$\mathbf{u}^1 = \begin{pmatrix} 0 \\ 3 \\ 5 \\ 1 \end{pmatrix}, \quad \mathbf{u}^2 = \begin{pmatrix} -7 \\ 0 \\ -2 \\ -2 \end{pmatrix},$$
$$\mathbf{u}^3 = \begin{pmatrix} 0 \\ 1 \\ 4 \\ 0 \end{pmatrix}, \quad \mathbf{u}^4 = \begin{pmatrix} 0 \\ 1 \\ 3 \\ -1 \end{pmatrix},$$
$$\mathbf{v}^1 = (-2 \quad 0 \quad -2 \quad -3 \quad 2), \quad \mathbf{v}^2 = (5 \quad 3 \quad 1 \quad 0 \quad 8),$$
$$\mathbf{v}^3 = (-2 \quad 0 \quad 0 \quad -2 \quad 3), \quad \mathbf{v}^4 = (-2 \quad 0 \quad 0 \quad -1 \quad 4). \tag{49}$$

V. Conclusion

Computational Results: We have coded Algorithm $\mathcal{H}$ in MATLAB [30]. We randomly generated 100 matrices of size 16×16 having integer values ranging from 0–255 and rank r for $r = 3, 4, 5$. We succeeded in finding a rank decomposition for the following percentages of the given matrices:

1) rank 3: 91%.
2) rank 4: 88%.
3) rank 5: 74%.

The average CPU time on a Sun SPARC 4 workstation and the average amount of flops required for the rank decomposition depend on the rank of the given matrix A:

1) rank 3: 13.45 s, 75 885 flops.
2) rank 4: 18.02 s, 110 950 flops.
3) rank 5: 24.03 s, 146 710 flops.

Concluding Remarks: Note that the intended use of Algorithm $\mathcal{H}$ is the problem of weakly decomposing a morphological $m \times n$ template into r outer products of column templates and row templates. Savings in computation time can only be achieved if $r < (m \cdot n)/(m + n)$.

As r approaches the limit of $16^2/(2 \cdot 16) = 8$ in this example, the success rate reveals a further decrease (about 43% for rank 6 matrices) while more and more operations are needed. Informally speaking, the decrease in the success rate of the algorithm as the matrix rank increases can be explained as follows:

The algorithm attempts to guess an outer product representation of the given matrix using a minimal number of vector pairs. Differences of corresponding entry tuples provide clues which can be used in order to construct such a rank decomposition. As the rank of the matrix to be decomposed gets higher, it becomes more likely that the same difference of corresponding entry tupels is accidental, leading to a wrong guess in the outer product representation.

Wrong guesses are also more frequent in the situation where the discrete matrix values are densely distributed over a small range than in the situation where the matrix values are sparsely distributed over a wide range. However, the following payoff exists: The probability that a wrong guess causes the failure of the algorithm appears much smaller in the first case than in the second case. Further research is needed in order to determine the criteria for optimal algorithm performance. Other suggestions for further research include the following.

Usefulness of algorithm for nonoptimal decomposition: The algorithm is guaranteed to determine an outer product representation. An application of the corresponding strip templates to an arbitrary image will still lead to savings in computation time as long as $(m \cdot n)/(m + n)$ exceeds the number of outer products determined.

Approximation capabilities: The execution of the algorithm may always be stopped at time r yielding an approximation of the given matrix, the given template respectively, in terms of r outer products.

Generalizations and variations: The algorithm may potentially be adapted to produce decompositions of templates other than rectangular templates and/or decompositions into templates other than strip templates.

Appendix

We state and prove the two lemmas mentioned in this manuscript and provide a proof of Theorem 8.

Lemma 1: Let $\mathbf{x}, \mathbf{u} \in \mathbb{R}^{m \times 1}$ *and* $\mathbf{y}, \mathbf{v} \in \mathbb{R}^{1 \times n}$. *Suppose that there exists an index* $j' \in \{1, \cdots, n\}$ *such that* $x_i + y_{j'} = u_i + v_{j'}$ *for all* $i \in I \subseteq \{1, \cdots, m\}$. *If for all* $j \in J \subseteq \{1, \cdots, n\}$ *there exists an index* $i_j \in I$ *such that* $u_{i_j} + v_j \geq x_{i_j} + y_j$ *then*

$$u_i + v_j \geq x_i + y_j \qquad \forall i \in I, \quad \forall j \in J. \tag{50}$$

Proof:

$$\begin{aligned} u_i + v_j &= (u_i - u_{i_j}) + (u_{i_j} + v_j) \\ &= [(u_i + v_{j'}) - (u_{i_j} + v_{j'})] + (u_{i_j} + v_j) \\ &\geq [(x_i + y_{j'}) - (x_{i_j} + y_{j'})] + (x_{i_j} + y_j) \\ &= [x_i - x_{i_j}] + (x_{i_j} + y_j) \\ &= x_i + y_j \qquad \forall i \in I, \quad \forall j \in J. \end{aligned} \tag{51}$$

Lemma 2: Suppose $\mathbf{u}, \mathbf{x} \in \mathbb{R}^{m \times 1}$, $\mathbf{v}, \mathbf{y} \in \mathbb{R}_{-\infty}^{1 \times n}$, *and* $j_1, j_2 \in \{1, \cdots, m\}$. *Furthermore, let a subset* I *of* $\{1, \cdots, m\}$ *and a subset* J *of* $\{1, \cdots, n\}$ *be given such that* v_j *and* x_j *are finite for all* $j \in J$. *If the following conditions are satisfied then* $u_i + v_j \geq x_i + y_j$ *for all* $i = 1, \cdots, m$ *and for all* $j \in J$.

$$\begin{aligned} u_i + v_{j_1} &= x_i + y_{j_1} = a_{ij_1} \qquad \forall i \in I, \\ u_i + v_{j_2} &= x_i + y_{j_2} = a_{ij_2} \qquad \forall i \in I, \\ u_{i'} + v_{j_1} &= x_{i'} + y_{j_1} \quad \text{or} \quad u_{i'} + v_{j_2} = x_{i'} + y_{j_2} \\ &\qquad \forall i' \in \{1, \cdots, m\} \backslash I, \\ u_i + v_j &\geq x_i + y_j \qquad \forall i \in I; \quad \forall j \in J. \end{aligned} \tag{52}$$

Proof: It suffices to show that for arbitrary $i' \in \{1, \cdots, m\} \backslash I$ and for arbitrary $j \in J$ the relation

$u_{i'} + v_j \geq x_{i'} + y_j$ holds. Suppose j' denotes j_1 if $u_{i'} + v_{j_1} = a_{ij_1}$ and j_2 otherwise. We obtain the following estimate for the difference $u_i - u_{i'}$, where i is an arbitrary element of I:

$$\begin{aligned} u_{i'} - u_i &= (u_{i'} + v_{j'}) - (u_i + v_{j'}) \\ &\geq (x_{i'} + y_{j'}) - (x_i + y_{j'}) \\ &= x_{i'} - x_i \end{aligned} \tag{53}$$

Using this estimate, the inequality $u_{i'} + v_j \geq x_{i'} + y_j$ now follows from the inequality $u_i + v_j \geq x_i + y_j$.

Proof of Theorem 8: Note that $\mathbf{u} \boxed{\vee} \mathbf{v} \leq \mathbf{A}$ by construction of the vectors $\mathbf{u}$ and $\mathbf{v}$. Furthermore, the set $\{i_1, \cdots, i_p\}$ generated in STEP 1 of Algorithm $\mathcal{H}$ contains the set I since $a_{ij_1} - a_{ij_2} = \tilde{v}_{j_1} - \tilde{v}_{j_2}$ for all $i \in I$. Let us first show that $a_{ij} \geq u_i + \tilde{v}_j \geq x_i + y_j$ for all $i \in I$ and for all $j \in J$. By Lemma 1, it suffices to show that for all $j \in J$ there exists an index i_j such that $u_{i_j} + \tilde{v}_j \geq x_{i_j} + y_j$.

By the definition of J, each index $j \in J$ entails an index $i_j \in I$ such that $m_{i_j j}$ is set to l in STEP 2 of Algorithm $\mathcal{H}$. If s denotes $\min\{i \in \{i_1, \cdots, i_p\}\colon m_{ij_1} = l\}$, then $\tilde{v}_j = a_{i_j j} + a_{sj_1} - a_{i_j j_1}$ for all $j \in J$. For an arbitrary, but fixed element $j \in J$ the inequalities $\tilde{v}_{j'} \leq a_{i_j j'} + a_{sj_1} - a_{i_j j_1}$ hold for all $j' = 1, \cdots, n$. Therefore, we are able to deduce the following equalities:

$$\begin{aligned} u_{i_j} + \tilde{v}_j &= \bigwedge_{j'=1}^{n} (a_{i_j j'} - \tilde{v}_{j'}) + \tilde{v}_j \\ &= (a_{i_j j} - \tilde{v}_j) + \tilde{v}_j \\ &= a_{i_j j} \qquad \forall\, j \in J. \end{aligned} \tag{54}$$

Since $\mathbf{A} \geq \mathbf{x} \boxed{\vee} \mathbf{y}$, the conditions of Lemma 1 are satisfied, yielding that $u_i + \tilde{v}_j \geq x_i + y_j$ for all $i \in I$ and for all $j \in J$.

Provided that $u_{i'} + \tilde{v}_{j_1} = x_{i'} + y_{j_1}$ or $u_{i'} + \tilde{v}_{j_2} = x_{i'} + y_{j_2}$ for all $i' \in \{1, \cdots, n\} \backslash I$ can be shown, an application of Lemma 2 concludes the proof of the theorem, since $\tilde{v}_j = v_j$ for all $j \in J$. Let us fix an arbitrary row index $i' \notin I$. By the definition of $u_{i'}$ as $\bigwedge_{j=1}^{n} (a_{i'j} - \tilde{v}_j)$, there exists a column index j' such that $u_{i'} = a_{i'j'} - \tilde{v}_{j'}$. Since $u_{i'}$ is finite, $\tilde{v}_{j'}$ is finite and equals $v_{j'}$. Therefore, j' either equals j_2 or is an element of the set J' which we define as the set of all column indices j such that $a_{ij} - a_{ij_1} = \bigwedge_{z=1}^{m} (a_{zj} - a_{zj_1})$ for some $i \in \{i_1, \cdots, i_p\}$. Note that J' includes j_1. The case $j' = j_2$ leaves nothing to show, since $u_{i'} = a_{i'j_2} - \tilde{v}_{j_2}$. Assuming that $j' \in J'$, we finish the proof of the theorem as follows. We use the facts that $\mathbf{A} \geq \mathbf{u} \boxed{\vee} \tilde{\mathbf{v}}$ and that $a_{ij_1} = u_i + \tilde{v}_{j_1}$ for all $i \in \{i_1, \cdots, i_p\}$.

$$\begin{aligned} a_{i'j'} - a_{i'j_1} &\geq \bigwedge_{z=1}^{m} (a_{zj'} - a_{zj_1}) = a_{ij'} - a_{ij_1} \\ &\quad \text{for some } i \in \{i_1, \cdots, i_p\} \\ &\Rightarrow u_{i'} + \tilde{v}_{j'} - a_{i'j_1} \geq (u_i + \tilde{v}_{j'}) - (u_i + \tilde{v}_{j_1}) \\ &\quad \text{for some } i \in \{i_1, \cdots, i_p\} \\ &\Rightarrow u_{i'} + \tilde{v}_{j_1} \geq a_{i'j_1} \\ &\Rightarrow u_{i'} + \tilde{v}_{j_1} = a_{i'j_1} \\ &\Rightarrow u_{i'} + \tilde{v}_{j_1} = x_{i'} + y_{j_1}. \end{aligned} \tag{55}$$

REFERENCES

[1] G. Birkhoff and J. Lipson, "Heterogeneous algebras," *J. Comb. Theory*, vol. 8, pp. 115–133, 1970.

[2] F. J. Crosby, "Maxpolynomials and morphological template decomposition," Ph.D. dissertation, Univ. Florida, Gainesville, 1995.

[3] R. Cuninghame-Green, *Minimax Algebra: Lecture Notes in Economics and Mathematical Systems 166*. New York: Springer-Verlag, 1979.

[4] J. L. Davidson, "Nonlinear matrix decompositions and an application to parallel processing," *J. Math. Imag. Vis.*, vol. 1, no. 2, pp. 169–192, 1992.

[5] ——, "Classification of lattice transformations in image processing," *Comput. Vis., Graph., Image Process.*, vol. 57, pp. 283–306, May 1993.

[6] P. D. Gader, "Necessary and sufficient conditions for the existence of local matrix decompositions," *SIAM J. Matrix Anal. Applicat.*, pp. 305–313, July 1988.

[7] ——, "Separable decompositions and approximations for gray-scale morphological templates," *CGVIP*, vol. 53, pp. 288–296, July 1991.

[8] P. D. Gader and S. Takriti, "Decomposition techniques for gray-scale morphological templates," *Proc. SPIE*, vol. 1350, pp. 431–442, July 1990.

[9] G. H. Golub and C. F. van Loan, *Matrix Computations*. Baltimore, MD: John Hopkins Univ. Press, 1983.

[10] H. J. A. M. Heijmans, "Mathematical morphology: A modern approach in image processing based on algebra and geometry," *SIAM Rev.*, vol. 37, no. 1, pp. 1–36, 1995.

[11] S.-Y. Lee and J. K. Aggarwal, "Parallel 2–D convolution on a mesh connected array processor," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-9, pp. 590–594, July 1987.

[12] D. Li, "Morphological template decomposition with max-polynomials," *J. Math. Imag. Vis.*, vol. 1, pp. 215–221, Sept. 1992.

[13] D. Li and G. X. Ritter, "Decomposition of separable and symmetric convex templates," in *Proc. Image Algebra Morphological Image Processing*, vol. 1350, San Diego, CA, July 1990, pp. 408–418.

[14] Z. Z. Manseur and D. C. Wilson, "Decomposition methods for convolution operators," *Comput. Vis., Graph., Image Process.*, vol. 53, no. 5, pp. 428–434, 1991.

[15] D. P. O'Leary, "Some algorithms for approximating convolutions," *Comput. Vis., Graph., Image Process.*, vol. 41, no. 3, pp. 333–345, 1988.

[16] H. Park and R. T. Chin, "Optimal decomposition of convex morphological structuring elements for 4-connected parallel array processors," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 16, no. 3, pp. 304–313, 1994.

[17] *Image Algebra*, available via anonymous ftp from ftp.cis.ufl.edu/pub/ia/documents, 1995.

[18] G. X. Ritter and P. D. Gader, "Image algebra: Techniques for parallel image processing," *J. Parallel Distrib. Comput.*, vol. 4, pp. 7–44, April 1987.

[19] G. X. Ritter and J. N. Wilson, *Handbook of Computer Vision Algorithms in Image Algebra*. Boca Raton, FL: CRC, 1996.

[20] G. X. Ritter, J. N. Wilson, and J. L. Davidson, "Image algebra: An overview," *Comput. Vis., Graph., Image Process.*, vol. 49, pp. 297–331, Mar. 1990.

[21] S. R. Sternberg, "Biomedical image processing," *Computer*, vol. 16, Jan. 1983.

[22] P. Sussner, P. M. Pardalos, and G. X. Ritter, "Global optimization problems in computer vision," in *State of the Art in Global Optimization*, C. A. Floudas and P. M. Pardalos, Eds. Norwell, MA: Kluwer, 1995, pp. 457–474.

[23] ——, "On integer programming approaches for morphological template decomposition problems in computer vision," *J. Combin. Optim.*, vol. 1, no. 2, pp. 165–178, 1997.

[24] P. Sussner and G. X. Ritter, "Decomposition of gray-scale morphological templates using the rank method," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 649–658, June 1997.

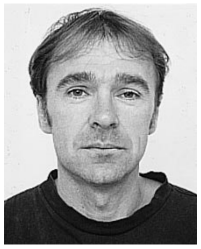
[25] S. Takriti and P. D. Gader, "Local decomposition of gray-scale morphological templates," *J. Math. Imag. Vis.*, vol. 2, pp. 39–50, 1992.

[26] ——, "Local decomposition of gray-scale morphological templates," *J. Math. Imag. Vis.*, vol. 2, pp. 39–50, 1992.

[27] J. Xu, "Decomposition of convex polygonal morphological structuring elements into neighborhood subsets," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, Feb. 1991.

[28] H. Zhu and G. X. Ritter, "The p–product and its applications in signal processing," *SIAM J. Matrix Anal. Applicat.*, vol. 16, pp. 579–601, Apr. 1995.

[29] X. Zhuang and R. M. Haralick, “Morphological structuring element decomposition,” *Comput. Vis., Graph., Image Process.*, vol. 35, pp. 370–382, 1986.
[30] P. Sussner, “Heuristic algorithm for computing outer product expansions of matrices in minimax algebra,” available via anonymous ftp from ftp://ftp.ime.inicamp.br.



Peter Sussner received the Diploma degree in mathematics from the University of Erlangen, Germany, in 1990, and the Ph.D. degree in mathematics from the University of Florida, Gainesville, in 1996.

In August 1991, he joined the University of Florida as a Fulbright Scholar. He was a Member of a research team with the Center for Computer Vision and Visualization. He is currently a Faculty Member of the Institute of Mathematics, Statistics, and Scientific Computation, State University of Campinas, Brazil. His research interests include neural networks, image algebra, image processing, and numerical optimization.

Gerhard X. Ritter (M’83–SM’85) received the B.A. and Ph.D. degrees from the University of Wisconsin, Madison, in 1966 and 1971, respectively.

He is currently Professor of computer science, the Chair of the Department of Computer and Information Science and Engineering, the Director of the Center for Computer Vision and Visualization, and Professor of mathematics with the University of Florida, Gainesville. In addition to his academic positions, he has been a Visiting Fellow at NASA Ames Research Labs and a Research Geophysicist, Texaco, Inc., Research Center, Bellaire, TX. He is the author of two books and over 80 referred publications in computer vision and mathematics. His current research interests are in mathematical imaging, image algebra, and artificial neural networks. He was the editor of the XI World Computer Congress. He is the Editor-in-Chief of the *Journal of Mathematical Imaging and Vision* and Program Chair of the International Society for Optical Engineering’s Program on Mathematical Imaging which encompasses five annual conferences on vision and imaging.

Dr. Ritter is the 1989 recipient of the International Federation of Information Processing Societies (IFIPS) Silver Core Award. He is a Fellow of the International Society for Optical Engineering (SPIE) and a member of the Board of Directors of SPIE. He is a member of the IEEE Computer Society, the IEEE Technical Committees on Pattern Analysis and Computational Medicine, the Association for Computing Machinery, the Society of Industrial and Applied Mathematics, the Mathematical Association of America, the American Society for Engineering Education, and the International Neural Network Society.