

# Łukasiewicz Transform Based Algorithm for Image Processing

Antonio Di Nola and Ciro Russo

**Abstract**—We define the Łukasiewicz Transform by mean of the partition of unity defined in [4] and semimodules over the semiring reducts of an MV-algebra. Then we describe the “Łukasiewicz Transform Based” (LTB) algorithm for image processing, showing some results of its application.

## I. INTRODUCTION

In the literature of image processing and data compression, very often the concept of transform appears (see, for instance, [10], [16]). Indeed a suitable discrete representation of a problem seems to be the best way - in terms of computability and accuracy of results - to approach many different tasks.

On the other hand, the theory of fuzzy relations is widely used in many applications (see, for instance, [5]) and particularly in the field of image processing ([7] – [9], [11]–[15]). As a matter of fact, fuzzy relations fit the problem of processing the representation of an image as a matrix whose elements’ range has been previously normalized into  $[0, 1]$ .

The aim of this paper is to provide a mathematical support for some techniques of image processing based on the theory of fuzzy sets, and to present an effective application of it.

Since these techniques depend on the algebraic structures defined on the real interval  $[0, 1]$ , it could be useful to study their properties and to find a sort of “linearity” with respect to operations. Hence an interest concerning semimodule structures related to Łukasiewicz semirings arises.

Making use of the theory of semimodules over semirings ([6]), we see that the structures of semimodule defined in a natural way on the (finite) cartesian power of an MV-chain has exactly one basis, and this fact leads to a natural definition of dimension for this class of semimodules; furthermore, over a fixed MV-chain, there exists only one of these semimodules of a given dimension, up to isomorphisms. These properties of uniqueness mean that, given an MV-chain  $A$  and a natural number  $n$ , the cartesian power  $A^n$  - with operations defined pointwise - is a general example of  $n$ -dimensional Łukasiewicz semimodule over the semiring reducts of  $A$ , and they also allow a very general and simple definition of a transform having the additional properties of being a semimodules’ homomorphism and a residuated map (Theorems 21 and 22).

More precisely, *Łukasiewicz Transform* is defined by mean of the partition of unity in MV-algebras defined in [4]. It turns out that it is a residuated map with respect to the natural lattice-order of MV-algebras. Furthermore the Łukasiewicz

Transform and its residual map are semimodules’ homomorphisms, as we already said, and one of their compositions yields the identity.

Such algebraic tools allow us to define a new algorithm for image processing - called Łukasiewicz Transform Based - that yields interesting results and can be an object of further studies and developments.

## II. PRELIMINARY NOTES

Since the next results deal with the theories of semimodules over semirings and MV-algebras, according to [6] and [1], we will now recall some basic definitions. We also refer to the above mentioned texts for the proves we will omit in this section.

### A. Semirings and semimodules

*Definition 1:* An algebraic structure  $(S, +, \cdot, 0, 1)$ , with two internal binary operations,  $+$  and  $\cdot$ , and two constants  $0, 1 \in S$ , is called a *semiring* if the following properties hold

- (S1)  $(S, +, 0)$  is a commutative monoid;
- (S2)  $(S, \cdot, 1)$  is a monoid;
- (S3)  $x \cdot (y + z) = xy + xz$  and  $(x + y) \cdot z = xz + yz$  for all  $x, y, z \in S$ ;
- (S4)  $0x = x0 = 0$  for all  $x \in S$ .

A semiring is said to be *commutative* iff the commutative property holds for the multiplication too.

We will consider only non-trivial semirings, i.e. semirings such that  $0 \neq 1$ .

*Definition 2:* Let  $S$  be a semiring. A *left  $S$ -semimodule* is a commutative monoid  $(M, +_M, 0_M)$  with an external operation, called *scalar multiplication*, with coefficients in  $S$

$$\bullet : (s, m) \in S \times M \mapsto s \bullet m \in M,$$

satisfying, for all  $s, s' \in S$  and  $m, m' \in M$ , the following conditions:

- (M1)  $(ss') \bullet m = s \bullet (s'm)$ ;
- (M2)  $s \bullet (m +_M m') = s \bullet m +_M s \bullet m'$ ;
- (M3)  $(s + s') \bullet m = s \bullet m +_M s' \bullet m$ ;
- (M4)  $1 \bullet m = m$ ;
- (M5)  $s \bullet 0_M = 0_M = 0 \bullet m$ .

The definition of *right  $S$ -semimodule* is analogous.

If  $S$  and  $S'$  are semirings and  $M$  is both a left  $S$ -semimodule and a right  $S'$ -semimodule,  $M$  will be called an  *$(S, S')$ -bisemimodule* iff it satisfies the following additional condition:

- (M6)  $(s \bullet_S m) \bullet_{S'} s' = s \bullet_S (m \bullet_{S'} s')$ , for all  $s \in S, s' \in S'$  and  $m \in M$ ,

where  $\bullet_S$  and  $\bullet_{S'}$  mean the external products with scalars in  $S$  and in  $S'$  respectively. In particular, if  $S$  is commutative,

Antonio Di Nola is with the Dipartimento di Matematica ed Informatica, Università di Salerno, Italy (phone: +39-089963363; fax: +39-089963303; email: adinola@unisa.it).

Ciro Russo is with the Dipartimento di Matematica ed Informatica, Università di Salerno, Italy (phone: +39-089963329; fax: +39-089963303; email: cirusso@unisa.it).

any left or right  $S$ -semimodule is an  $(S, S)$ -bisemimodule and we will call it, shortly, an  $S$ -bisemimodule.

Note that we will always omit the “•” symbol when there will not be any danger of confusion.

Throughout this section, we will denote the semiring  $(S, +, \cdot, 0, 1)$  simply by  $S$ , and  $M = (M, +_M, 0_M)$  will be a left semimodule over  $S$ . Obviously all the following definitions and results hold both for right and left  $S$ -semimodules.

*Definition 3:* Let  $X$  be a subset of  $M$  and let us give the following definitions.

- (i) A *linear combination* of elements of  $X$  is a sum  $\sum_{x \in X} f(x)x$ , where  $f : X \rightarrow S$  is a map such that the set  $\{x \in X \mid f(x) \neq 0\}$  (the so called “*support*” of  $f$ ) is finite. By the notation  $\sum_{x \in X} f(x)x$  we will always mean linear combinations, i.e. sums where all but a finite number of summands are equal to zero.
- (ii) Substructures generated by a subset are defined as usual: the *subsemimodule generated* by  $X$ , denoted by  $\langle X \rangle$ , is the intersection of all subsemimodules of  $M$  containing  $X$ , and it coincides with the set of all the linear combinations of elements of  $X$ .
- (iii) If  $M$  is finitely generated, the *rank* of  $M$  is the minimum number  $n \in \mathbb{N}$  such that there exists a set of generators  $X$  of  $M$  with  $|X| = n$ .
- (iv)  $X$  is *linearly independent* iff, given two linear combinations  $\sum_{x \in X} f(x)x$  and  $\sum_{x \in X} g(x)x$ , from  $\sum_{x \in X} f(x)x = \sum_{x \in X} g(x)x$  it follows that  $f = g$ .
- (v)  $X$  will be called *linearly dependent* if it isn't linearly independent, i.e.  $X$  is linearly dependent iff there exist two functions  $f \neq g : X \rightarrow S$  such that  $\sum_{x \in X} f(x)x = \sum_{x \in X} g(x)x$  and the supports of  $f$  and  $g$  are both finite.
- (vi) A *basis* for  $M$  over  $S$  is a linearly independent set of generators for  $M$  over  $S$ .
- (vii) An  $S$ -semimodule is called *free* iff it has a basis over  $S$ .

*Definition 4:* Assume  $M$  to be a free left-semimodule over  $S$ . We say that  $M$  has *dimension*  $n$ , or that it is  *$n$ -dimensional*, with  $n \in \mathbb{N}$ , iff for each basis  $B$  of  $M$  over  $S$ ,  $|B| = n$ . In this case we will write  $\dim M = n$ .

*Lemma 5:* Let  $M$  be a free left  $S$ -semimodule, with basis  $X$ . Then, for all  $y \in M \setminus X$ ,  $X \cup \{y\}$  is linearly dependent.

Moreover, if  $Y$  is a set of generators with  $Y \subseteq X$ , then  $X = Y$ .

*Theorem 6:* Let  $M = (M, +, 0)$  be a free left semimodule over  $S$ , and suppose that  $X$  is an infinite basis for  $M$ . Then, if  $Y$  is a basis for  $M$  over  $S$ ,  $X$  and  $Y$  are equipotent.

*Proof:* Since  $X$  is a set of generators, for all  $y \in Y$ , there exists a finite subset  $X_y$  of  $X$  such that  $y \in \langle X_y \rangle$ ; hence  $Y \subseteq \langle \bigcup_{y \in Y} X_y \rangle$ , so  $M = \langle Y \rangle = \langle \bigcup_{y \in Y} X_y \rangle$ .

It follows that  $\bigcup_{y \in Y} X_y (\subseteq X)$  is a set of generators for  $M$  and, by Lemma 5,  $X = \bigcup_{y \in Y} X_y$ ; so we have  $|X| =$

$\left| \bigcup_{y \in Y} X_y \right| \leq |Y|$ , by a classical set-theoretical result.

The proof for the inverse inequality is completely analogous. ■

*Corollary 7:* Let  $M$  be a free left  $S$ -semimodule having a finite basis  $B$  over  $S$ . Then all the bases of  $M$  over  $S$  are finite.

## B. MV-algebras

*Definition 8:* An algebraic structure  $(A, \oplus, *, 0)$  with an internal binary operation  $\oplus$ , an internal unary operation  $*$  and a constant 0 is called an *MV-algebra* if the following hold

- (A1)  $x \oplus (y \oplus z) = (x \oplus y) \oplus z$ ;
- (A2)  $x \oplus y = y \oplus x$ ;
- (A3)  $x \oplus 0 = x$ ;
- (A4)  $(x^*)^* = x$ ;
- (A5)  $x \oplus 0^* = 0^*$ ;
- (A6)  $(x^* \oplus y)^* \oplus y = (y^* \oplus x)^* \oplus x$ .

On every MV-algebra it is possible to define another constant 1 and two further operation as follows:

$$\begin{aligned} 1 &= 0^*, \\ x \odot y &= (x^* \oplus y^*)^*, \\ x \ominus y &= x \odot y^*. \end{aligned}$$

The following properties follow directly from the above definitions:

- (A7)  $1^* = 0$ ,
- (A8)  $x \oplus y = (x^* \odot y^*)^*$ ,
- (A5')  $x \oplus 1 = 1$  (rewriting (A5)),
- (A6')  $(x \ominus y) \oplus y = (y \ominus x) \oplus x$  (rewriting (A6)),
- (A9)  $x \oplus x^* = 1$ .

In what follows we will often denote an MV-algebra by  $(A, \oplus, \odot, *, 0, 1)$ .

*Lemma 9:* Let  $A$  be an MV-algebra and  $x, y \in A$ . Then following conditions are equivalent:

- (i)  $x^* \oplus y = 1$ ;
- (ii)  $x \odot y^* = 0$ ;
- (iii)  $y = x \oplus (y \ominus x)$ ;
- (iv) there exists an element  $z \in A$  such that  $x \oplus z = y$ .

Let  $A$  be an MV-algebra. For any two elements  $x$  and  $y$  of  $A$  let us agree to write

$$x \leq y$$

iff  $x$  and  $y$  satisfy the above equivalent conditions (i)–(iv). It follows that  $\leq$  is a partial order, called the *natural order* of  $A$ . This relation also determines a lattice structure, with 0 and 1 respectively infimum and supremum elements, and  $\vee$  and  $\wedge$  defined as follows:

$$\begin{aligned} x \vee y &= (x \odot y^*) \oplus y = (x \ominus y) \oplus y, \\ x \wedge y &= (x^* \vee y^*)^* = x \odot (x^* \oplus y). \end{aligned}$$

We will say that the MV-algebra  $A$  is *complete* if it is complete as a lattice, and that it is an *MV-chain* if it is totally ordered by its natural order.

An important example of MV-algebra is  $([0, 1], \oplus, *, 0, 1)$ , where  $[0, 1]$  is the real unit interval and, for all  $x, y \in [0, 1]$ ,

$$\begin{aligned}x \oplus y &= \min\{1, x + y\}, \\x^* &= 1 - x.\end{aligned}$$

Moreover  $\leq$  is the usual order relation between real numbers, lattice operations are – again – the usual lattice operations between real numbers, and

$$\begin{aligned}x \odot y &= \max\{0, x + y - 1\}, \\x \ominus y &= \max\{0, x - y\}.\end{aligned}$$

In what follows we will often use this structure.

### III. MV-SEMIMODULES AND ŁUKASIEWICZ SEMIMODULES

In this section we will treat semiring and semimodule structures defined on complete MV-algebras by mean of their operations and their lattice structure.

*Definition 10:* Let  $A = (A, \oplus, \odot, *, 0, 1)$  be a complete MV-algebra, and  $X$  be a non-empty set. Then

$$A^X = (A^X, \oplus, \odot, *, \mathbf{0}, \mathbf{1})$$

is a complete MV-algebra with operations defined pointwise. By [3], for any MV-algebra  $A$  it is possible to define two different semirings:  $\mathfrak{L}^\wedge = (A, \wedge, \oplus, 1, 0)$  and  $\mathfrak{L}^\vee = (A, \vee, \odot, 0, 1)$ , called the *semiring reducts* of  $A$ .

Moreover the monoid  $\mathfrak{M} = (A^X, \vee, \mathbf{0})$  is a bisemimodule over both  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$ , with scalar multiplications defined respectively as  $af = a^* \odot f$  and  $af = a \odot f$ , for all  $a \in A$  and  $f \in A^X$ . Analogously  $(A^n, \vee, \mathbf{0})$  is a bisemimodule over  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$ , for any  $n \in \mathbb{N}$ . We will call  $\mathfrak{M}$  a *MV- $\mathfrak{L}^\wedge$ -semimodule* (respectively: *MV- $\mathfrak{L}^\vee$ -semimodule*) over  $A$ .

If  $A$  is an MV-chain, the MV-semimodules over it will be called *Łukasiewicz  $\mathfrak{L}^\wedge$ -semimodule* and *Łukasiewicz  $\mathfrak{L}^\vee$ -semimodule* respectively.

*Lemma 11:* Let  $A$  be a non-trivial MV-algebra, and let  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$  be the semiring reducts of  $A$ . Then  $\mathfrak{M} = (A, \vee, 0)$  is a 1-dimensional free bisemimodule over  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$ . Moreover there exists exactly one basis for  $\mathfrak{M}$ , in both cases.

*Proposition 12:* Let  $A = (A, \oplus, \odot, *, 0, 1)$  be an MV-algebra, and  $X$  be a non-empty set with cardinality  $\kappa$ . Then the monoids  $(A^X, \vee, \mathbf{0})$  and  $(A^\kappa, \vee, \mathbf{0})$  - where  $\vee$  is defined pointwise, as usual - are isomorphic. Moreover they are isomorphic as bisemimodules over  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$  too.

*Proof:* It is trivial.  $\blacksquare$

*Theorem 13:* Let  $A$  be a non-trivial MV-chain and let  $n \in \mathbb{N}$ . Then  $\mathfrak{M} = (A^n, \vee, \mathbf{0})$  is an  $n$ -dimensional free bisemimodule over  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$ . Moreover there exists exactly one basis for  $\mathfrak{M}$ , in both cases.

*Theorem 14:* Let  $A = (A, \oplus, \odot, *, 0, 1)$  be an MV-chain and  $n \in \mathbb{N}$ . Then, up to isomorphisms,  $(A^n, \vee, 0)$  is the unique Łukasiewicz  $n$ -dimensional free bisemimodule over  $\mathfrak{L}^\wedge$  (respectively: over  $\mathfrak{L}^\vee$ ). Moreover it has exactly one basis over both  $\mathfrak{L}^\wedge$  and  $\mathfrak{L}^\vee$ .

*Proof:* It follows from Lemma 11, Proposition 12 and Theorems 6 and 13.  $\blacksquare$

### IV. RESIDUATED MAPS AND ŁUKASIEWICZ TRANSFORM

In [1] the authors define a functor  $\Gamma$  between the category of lattice ordered groups with strong unity and the one of MV-algebras.

*Definition 15:* Let  $A = \Gamma(G, +, 1)$  be an MV-algebra. A finite sequence of elements of  $A$ ,  $(a_0, \dots, a_{n-1})$  is a partition of unity if  $a_0 + \dots + a_{n-1} = 1$ .

Let us describe a partition of unity in the MV-algebra  $([0, 1]^{[0,1]}, \oplus, \odot, 0, 1)$ , as in [4].

Let  $n \in \mathbb{N}$ ,  $n > 1$ , and  $k \in \mathbb{Z}$  and set:

$$\begin{aligned}\text{if } k < 0, \text{ then } \pi_k^{n-1}(x) &= x \oplus x^*; \\ \text{if } k \geq n-1, \text{ then } \pi_k^{n-1}(x) &= x \odot x^*; \\ \text{if } 0 \leq k \leq n-2, \text{ then:}\end{aligned}$$

$$\begin{aligned}\pi_0^{n-1}(x) &= (n-1)x, \\ \pi_1^{n-1}(x) &= \bigoplus_{i=0}^{n-2} F_{0i}(x), \\ &\dots \\ \pi_k^{n-1}(x) &= \bigoplus_{i=k}^{n-2} F_{0,1,\dots,k-1,i}(x), \\ &\dots \\ \pi_{n-2}^{n-1}(x) &= F_{0,1,\dots,n-3,n-2}(x),\end{aligned}$$

where  $F_{0,1,\dots,k-1,i}(x)$  are defined as follows:

$$\begin{aligned}\text{for every integer } i > 0, F_{0,i}(x) &= x \odot ix, \\ \text{for every integer } i > 1, F_{0,1,i}(x) &= (F_{0,1}(x) \oplus \dots \oplus \\ &F_{0,i-1}(x)) \odot F_{0,i}(x),\end{aligned}$$

and, by induction,

for every integer  $i$  such that  $i > b$ ,

$$\begin{aligned}F_{0,1,\dots,k,i}(x) &= (F_{0,1,\dots,k-1,k}(x) \oplus \dots \\ &\oplus F_{0,1,\dots,k-1,i-1}(x)) \odot F_{0,1,\dots,k-1,i}(x).\end{aligned}$$

Set  $p_k(x) = \pi_{k-1}^{n-1}(x) \wedge (\pi_k^{n-1}(x))^*$ ,  $k = 0, \dots, n-1$ .

By (P1)-(P6) and Lemma 14 of [4], it follows that the  $n$ -tuple  $(p_0(x), \dots, p_{n-1}(x))$  is a partition of unity of the MV-algebra  $([0, 1]^{[0,1]}, \oplus, \odot, \mathbf{0}, \mathbf{1})$ , and it also bears a partition of  $[0, 1]$  by the nodes  $\{x_0, x_1, \dots, x_{n-1}\}$ , where  $x_k = \frac{k}{n-1}$ ,  $k = 0, \dots, n-1$ .

In analytical form we have

$$p_0(x) = \begin{cases} -(n-1)x + 1 & \text{if } 0 \leq x \leq \frac{1}{n-1}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

$$p_{n-1}(x) = \begin{cases} (n-1)x - (n-2) & \text{if } \frac{n-2}{n-1} \leq x \leq 1, \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

For  $k = 1, \dots, n-2$ ,

$$p_k(x) = \begin{cases} (n-1)x - (k-1) & \text{if } \frac{k-1}{n-1} \leq x \leq \frac{k}{n-1}, \\ -(n-1)x + k + 1 & \text{if } \frac{k}{n-1} \leq x \leq \frac{k+1}{n-1}, \\ 0 & \text{otherwise.} \end{cases}$$

*Remark 16:* If  $x = x_k = \frac{k-1}{n}$ ,  $k = 1, 2, \dots, n+1$ , then  $p_k(x) = 1$  and  $p_h(x) = 0$ , for  $h \neq k$ .

If  $k > 0$  and  $\frac{k-1}{n-1} < x < \frac{k}{n-1}$ , then  $p_{k-1}^*(x) = p_k(x) \neq 0, 1$  and  $p_h(x) = 0$ , for  $h \neq k-1, k, k+1$ .

If  $\frac{k}{n-1} < x < \frac{k+1}{n-1}$ , then  $p_k^*(x) = p_{k+1}(x) \neq 0, 1$  and  $p_h(x) = 0$ , for  $h \neq k-1, k, k+1$ .

*Proposition 17:*  $\{p_0, p_1, \dots, p_{n-1}\}$  is a partition of unity in the MV-algebra  $[0, 1]^{[0,1]}$ , having the property  $p_k \odot p_h = 0$  for  $k \neq h$ .

*Proof:* See the above remark. ■

According to [2], we set the following

*Definition 18:* Let  $(X, \leq)$  and  $(Y, \leq)$  be two ordered sets. A map  $h : X \rightarrow Y$  is said to be *residuated* if it is isotone and, for all  $y \in Y$ , the set  $\{x \in X : h(x) \leq y\}$  admits the greatest element, denoted by  $h^\sharp(y)$ . The map  $h^\sharp : Y \rightarrow X$  is called the *residual*, or the *residual map*, of  $h$ , and the pair  $(h, h^\sharp)$  is said to be *adjoint*.

Generally (see subsection 2.2 of [2]), for a residuated map  $h : X \rightarrow Y$  and its residual  $h^\sharp : Y \rightarrow X$ , we have that the residual application  $h^\sharp$  is the unique isotone map that satisfies both conditions

$$h \circ h^\sharp \leq I_Y \quad (3)$$

and

$$h^\sharp \circ h \geq I_X \quad (4)$$

where  $I_X, I_Y$  are the identity maps.

*Definition 19:* Let  $S$  be a semiring and let  $M = (M, +, 0)$  be an  $S$ -semimodule; furthermore, let  $X$  be a non-empty set and  $n \in \mathbb{N}$ . We know that both  $(M^X, +, 0)$  and  $(M^n, +, 0)$  are  $S$ -semimodules with operations defined pointwise.

We call *semimodules' transform* of order  $n$  a homomorphism  $h_n : M^X \rightarrow M^n$  such that there exists a homomorphism  $\lambda_n : M^n \rightarrow M^X$  having the following properties:

$$(T1) \quad h_n \circ \lambda_n \circ h_n = h_n,$$

$$(T2) \quad \lambda_n \circ h_n \circ \lambda_n = \lambda_n.$$

We recall that, if  $A = (A, \oplus, \odot, *, 0, 1)$  is a complete MV-algebra and  $X$  is a non-empty set,  $(A^X, \vee, 0)$  has two bisemimodules' structures (over  $\mathcal{L}^\wedge$  and  $\mathcal{L}^\vee$ ) shown in Definition 10.

*Definition 20:* Let us consider the MV-algebra

$$A = ([0, 1], \oplus, \odot, *, 0, 1),$$

and set  $X = [0, 1]$ . Thus  $A^X$  is the set of all functions from  $[0, 1]$  to  $[0, 1]$ ,  $A^n$  is the set of all the  $n$ -vectors valued on  $[0, 1]$  and  $\mathcal{L}^\wedge$  is the semiring reduct  $([0, 1], \wedge, \oplus, 1, 0)$ . We call *Lukasiewicz Transform* the operator

$$H_n : [0, 1]^{[0,1]} \longrightarrow [0, 1]^n$$

defined by:

$$H_n(f) = \left( \bigvee_{x \in [0,1]} f(x) \odot p_0(x), \dots, \bigvee_{x \in [0,1]} f(x) \odot p_{n-1}(x) \right), \quad (5)$$

where  $p_0, \dots, p_{n-1}$  is the partition of unity defined by (1) and (2).

*Theorem 21:*  $H_n$  is a Łukasiewicz  $\mathcal{L}^\wedge$ -semimodules' homomorphism from  $([0, 1]^{[0,1]}, \vee, 0)$  to  $([0, 1]^n, \vee, 0)$ .

*Proof:* It is trivial. ■

Since, by [2], for any adjoint pair  $(h, h^\sharp)$  hold both

$$h \circ h^\sharp \circ h = h$$

and

$$h^\sharp \circ h \circ h^\sharp = h^\sharp,$$

from Theorem 22 it will follow that  $H_n$  is also a semimodules' transform.

Let now define the *Lukasiewicz antitransform*  $\Lambda_n$  as the map:

$$(v_0, \dots, v_{n-1}) \mapsto \left( \bigvee_{k=0}^{n-1} v_k p_k \right)^*, \quad (6)$$

where the product  $v_k p_k$  means the external product  $v_k^* \odot p_k$ . We will see in Theorem 22 that  $\Lambda_n$  is the residual map of  $H_n$ , i.e.  $\Lambda_n = H_n^\sharp$ , and it has one further important property.

In a certain sense, we can say that the vector  $H_n(f)$  represents the components of  $f$  with respect of the "frame"  $p_0(x), \dots, p_{n-1}(x)$ .

On the other hand  $\Lambda_n$  tell us the way to linearly perform the elements of the basis  $p_0(x), \dots, p_{n-1}(x)$  by the  $n$ -tuple of scalars  $v_0, \dots, v_{n-1}$ .

Indeed consider  $H_n(p_k(x))$ . It is equal to the vector  $e^k$ , having all the components equal to 0, except the  $k$ -th, which is 1. Hence  $\Lambda_n(e^k) = p_k$ .

We will now see that  $H_n$  is a residuated map and that its residual  $H_n^\sharp$  coincides with  $\Lambda_n$ . Moreover, as we will see, in this case we get more. The proof of the following Theorem is not trivial and it involves several calculuses, so we must omit it for space constraints.

*Theorem 22:* The map  $H_n$  is residuated and  $\Lambda_n$  is its residual map. Moreover we have

$$H_n \circ \Lambda_n = I_{[0,1]^n}.$$

## V. THE ŁUKASIEWICZ TRANSFORM BASED (ŁTB) ALGORITHM FOR IMAGE PROCESSING

The Łukasiewicz Transform has been defined for functions  $f : [0, 1] \rightarrow [0, 1]$ , and this fact implies that the first step of its application to image processing consists in "adapting" the image to the domain of our operator. In other words, each image (i.e. each fuzzy matrix) must be seen as a  $[0, 1]$ -valued function defined on (a subset of)  $[0, 1]$ .

Let us consider a  $m \times n$  fuzzy matrix  $X = (x_{ij})$ , where  $i = 0, \dots, m-1$ , and  $j = 0, \dots, n-1$ . It is easy to see that we can rewrite  $X$  as a  $m \cdot n$  vector  $(x'_0, \dots, x'_{mn-1})$  by setting - for all  $k = 0, \dots, mn-1$  -  $x'_k = x_{q(k,n)r(k,n)}$ , where  $q(k, n)$  and  $r(k, n)$  are, respectively, quotient and remainder of the euclidean division  $k/n$ .

Then we set

$$D_X = \left\{ \frac{k}{mn-1} \mid k = 0, \dots, mn-1 \right\} \subset [0, 1],$$

so we can apply the Łukasiewicz Transform to the matrix  $X$  rewritten as

$$f_X : \frac{k}{mn-1} \in D_X \mapsto x'_k \in [0, 1]. \quad (7)$$

Every grey (respectively: RGB colour) image we processed has been treated as a fuzzy matrix (resp.: three fuzzy matrices, one for each colour), and each matrix has been divided in blocks. After these preliminary operations, we applied the Łukasiewicz Transform to each block separately.

## VI. APPLYING ŁTB ALGORITHM TO GREY AND RGB COLOUR IMAGES

In order to test the above method, from [17] we have extracted and processed several images. Here we show visual and numerical results for Testpat.1k of Fig. 1 and Mandrill of Fig. 5 plus the numerical results of three further well known images: Bridge (grey), Lena and Peppers (RGB).

We tested three processes of compression/decompression; in these processes we have divided the fuzzy matrix (or matrices, in case of RGB images) associated to the images in square blocks of sizes  $m_b \times n_b = 2 \times 2$ ,  $8 \times 8$  and  $4 \times 4$ , respectively compressed to blocks of sizes  $k_b \times h_b = 2 \times 1$ ,  $5 \times 5$  and  $2 \times 2$  by mean of formulas (7) and (5).

The respective compression rates are obviously  $\rho = (2 \times 2)/(2 \times 1) = 0.5$ ,  $\rho = (8 \times 8)/(5 \times 5) \approx 0.39$  and  $\rho = (4 \times 4)/(2 \times 2) = 0.25$ . The blocks  $2 \times 1$ ,  $5 \times 5$  and  $2 \times 2$  have been afterwards decompressed to blocks of the respective original sizes, with the formula (6), hence recomposed by giving the images shown in the following Figs. 2, 3 and 4, 6, 7, 8.

In Tables I, II and III we show the values of ŁTB algorithm execution time, compared with JPEG execution time for the same compression ratios.

TABLE I

NUMERICAL VALUES FOR GREY AND RGB COLOUR IMAGES ( $\rho = 0.5$ )

Image	JPEG RMSE	ŁTB RMSE	JPEG PSNR	ŁTB PSNR
Bridge	2.4985	58.4469	40.1773	12.7956
Testpat.1k	0.0833	61.1921	69.7131	12.3969
Lena	2.2606	53.1803	41.0464	13.6158
Mandrill	6.4739	55.8967	31.9075	13.1831
Peppers	2.0813	56.7046	41.7641	13.0584

TABLE II

NUMERICAL VALUES FOR GREY AND RGB COLOUR IMAGES ( $\rho = 0.39$ )

Image	JPEG RMSE	ŁTB RMSE	JPEG PSNR	ŁTB PSNR
Bridge	3.6168	56.3420	36.9643	13.1141
Testpat.1k	0.0833	66.2213	69.7131	11.7108
Lena	2.2606	51.6195	41.0464	13.8745
Mandrill	6.5917	53.5997	31.7509	13.5476
Peppers	2.0813	55.8171	41.7641	13.1954

TABLE III

NUMERICAL VALUES FOR GREY AND RGB COLOUR IMAGES ( $\rho = 0.25$ )

Image	JPEG RMSE	ŁTB RMSE	JPEG PSNR	ŁTB PSNR
Bridge	6.1120	68.3310	32.4071	11.4384
Testpat.1k	0.0833	74.2191	69.7131	10.7205
Lena	2.4819	61.6912	40.2351	12.3263
Mandrill	7.1230	65.7072	31.0775	11.7785
Peppers	2.1147	65.4880	41.6257	11.8076

## VII. COMPARING JPEG AND ŁTB ALGORITHMS

### A. Computability

The coding/decoding algorithms are usually compared by mean of their execution times and the values of some parameters (PSNR, RMSE, MSE).

The comparison between the ŁTB algorithm and JPEG is heavily conditioned by their underlying implementation. Indeed in the compression/reconstruction process the computational time, for each block of sizes  $8 \times 8$ , is characterized by the execution of the inverse DCT/DCT. The standard implementation of DCT determines an asymptotic computational time, of the DCT on one block, that is  $O(m \cdot n)$ , where  $m$  and  $n$  are, respectively, the number of rows and the one of columns of the block.

If we process  $8 \times 8$  blocks, the asymptotic time is not relevant anymore and it is more convenient to look at the number of operations executed. The standard implementation requires in general 1024 products and 896 sums for computing the DCT on a block of these sizes, but there exist several optimized DCT implementations (FastDCT et al.) that reduce significantly these numbers. For instance, the FastDCT developed by Feig requires only 54 products, 464 sums and 6 arithmetical shifts, giving the same result.

Furthermore we should add the time and operations required for other components of the application: quantization, downsampling and entropic encoding.

If we set  $x = m_b \cdot n_b \cdot h_b \cdot k_b$  and  $y$  as the number of colour channels of the image (one for grey images, three for RGB images), the ŁTB algorithm computes, for the compression of one block,  $x \cdot y$  products,  $x \cdot y$  comparisons and, at most,  $x \cdot y$  assignments. If, for instance, we set  $m_b = n_b = 3$ ,  $h_b = k_b = 2$  and  $y = 3$ , then the whole compression algorithm requires - for each block - 108 products, 108 sums, 108 comparisons and at most 108 assignments. All this values can be still reduced by mean of an underlying implementation endowed with a suitable advanced implementation.

From these considerations it follows that the ŁTB algorithm requires an execution time much shorter with respect to JPEG. Nevertheless the values in Tabs. IV–VI show that the JPEG application used for our tests (FreeImage Library 3.8, in [18]) is faster for some images. This fact depends on the sampling scheme in blocks of sizes  $8 \times 8$ , that supports an optimization of JPEG's implementations for several CPU architectures. In particular, for all CPU supporting MMX, SSE, SSE2, SSE3 and the ones with SIMD (Single Instructions Multiple Data) architecture, using a 64-bit sampling.

On the other hand, the source code of our CoDec has been realized with simple C-like optimizations, since our purpose was just showing the feasibility of this approach and the possible results. So this comparison should be read also considering the possibility of improving the application overworking SIMD architectures' optimizations.

### B. Numerical indexes

With regards to the numerical comparison between JPEG and ŁTB we can say that, even if there is still a consistent

TABLE IV  
RUNNING TIMES (IN *ms*) WITH  $\rho = 0.5$

Image	JPEG	ŁTB
Bridge	80.98	41.85
Testpat.1k	150.47	123.30
Lena	184.40	98.68
Mandrill	187.29	105.62
Peppers	150.61	85.96

TABLE V  
RUNNING TIMES (IN *ms*) WITH  $\rho = 0.39$

Image	JPEG	ŁTB
Bridge	51.68	290.81
Testpat.1k	82.13	4888.75
Lena	173.47	1835.67
Mandrill	98.65	6287.89
Peppers	92.95	1551.73

gap, the ŁTB algorithm has got an interesting property: unlike JPEG, an iterative application of the algorithm on the same image is lossy just for the first process and lossless for the following ones. In other words, once we have compressed and reconstructed an image, we can apply the same process again, on the reconstructed image, obtaining exactly the same compressed and the same reconstructed images (with fixed sizes for the blocks).

### VIII. CONCLUSIONS

In this work we propose a theoretical approach, in the framework of semimodules' theory and MV-algebras, to the task of image compression. Making use of several tools, both algebraic and logical, we have defined an operator having good mathematical properties and an algorithmic process with interesting, though still improvable, concrete results.

Nevertheless the Łukasiewicz Transform seems to be "stable" enough for being applied to other tasks in the fields of data compression and data mining. In other words, for the Łukasiewicz Transform, the properties of being a homomorphism and of yielding - together with its antitransform - an adjoint pair, make it a reliable and versatile tool.

Possible extensions in the range of applications of the above results, as well as some optimization criteria for the applications of Łukasiewicz Transform, can be guessed. They will be motivations for future works.

### REFERENCES

- [1] Cignoli R.L.O., D'Ottaviano I.M.L., Mundici D., *Algebraic Foundations of Many-valued Reasoning*, Trends in Studia Logica, Kluwer, Dordrecht, 2000.
- [2] Cohen G., Gaubert S., Quadrat J.-P., *Hahn-Banach Separation Theorem for Max-Plus Semimodules*, pp. 325–334 in: J.-L. Menaldi, E. Rofman, A. Sulem (Eds.), *Optimal Control and Partial Differential Equations*, Proceedings of Conference in honour of Professor Alain Bensoussan's 60th birthday, Dec. 4-5 2000, Paris, published by IOS Press, Amsterdam, 2000.
- [3] Di Nola A., Gerla B., *Algebras of Łukasiewicz's logic and their semiring reducts*, Idempotent mathematics and mathematical physics, 131–144, Con-temp. Math., 377, Amer. Math. Soc., Providence, RI, 2005.

TABLE VI  
RUNNING TIMES (IN *ms*) WITH  $\rho = 0.25$

Image	JPEG	ŁTB
Bridge	43.64	59.19
Testpat.1k	4211.14	181.17
Lena	141.04	178.40
Mandrill	113.55	4250.62
Peppers	98.28	157.85

- [4] Di Nola A., Lettieri A., *On Normal Forms in Łukasiewicz Logic*, Arch. Math. Logic, **43**, no. 6, 795–823, 2004.
- [5] Di Nola A., Sessa S., Pedrycz, W., Sanchez, E., *Fuzzy relation equations and their applications to knowledge engineering*, Kluwer, Dordrecht, 1989.
- [6] Golan J. S., *The theory of semirings with applications in mathematics and theoretical computer science*, Longman Scientific and Technical, Harlow and Wiley, New York, 1992.
- [7] Hirota K., Kawamoto K., Nobuhara H., Yoshida S.I., *On a lossy image compression/reconstruction method based on fuzzy relational equations*, Iran. J. Fuzzy Syst., **1**, 33–42, 2004.
- [8] Hirota K., Nobuhara H., Pedrycz W., *Relational image compression: optimizations through the design of fuzzy coders and YUV color space*, Soft Computing, **9**, 471–479, 2005.
- [9] Hirota K., Pedrycz W., *Fuzzy relational compression*, IEEE Trans. Syst. Man Cyber.–Part B, **29** (3), 407–415, 1999.
- [10] Kwak K.-C., Pedrycz W., *Face Recognition Using Fuzzy Integral and Wavelet Decomposition Method*, IEEE Trans. Syst. Man Cyber.–Part B, **34** (4), 1666–1675, 2004.
- [11] Loia V., Sessa S., *Fuzzy relation equations for coding/decoding processes of images and videos*, Inform. Sci., **171**, no. 1-3, 145–172, 2005.
- [12] Nachttegael M., Van der Weken D., Van De Ville D., Kerre E.E., Philips W., Lemahieu I., *An overview of fuzzy and fuzzy-classical filters for noise reduction*, Proceedings of the 10th FUZZ-IEEE Conference, 3–6, 2001.
- [13] Nobuhara H., Pedrycz W., Hirota K., *A digital watermarking algorithm using image compression method based on fuzzy relational equations*, IEEE Inter. Conference Fuzzy Syst., CD-ROM In: Proc Hawaii, 2002.
- [14] Nobuhara H., Pedrycz W., Hirota K., *Fast solving method of fuzzy relational equations and its application to image compression / reconstruction*, IEEE Trans. Fuzzy Syst., **8** (3), 325–334, 2000.
- [15] Nobuhara H., Takama Y., Hirota K., *Image compression/reconstruction based on various types of fuzzy relational equations*, Trans. Inst. Electrical Eng. Japan, **121-C**(6), 1102–1113, 2001.
- [16] Perfilieva I., *Fuzzy Transforms and Their Applications to Data Compression*, Proceedings of The 2005 IEEE International Conference on Fuzzy Systems, 294–299, 2005.
- [17] <http://sampl.eng.ohio-state.edu/~sampl/database.htm>
- [18] <http://freeimage.sourceforge.net>

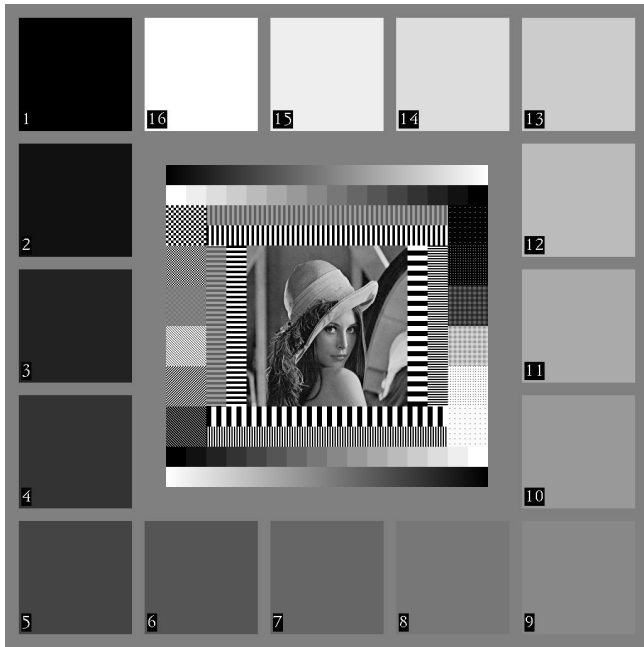


Fig. 1. Testpat.1k

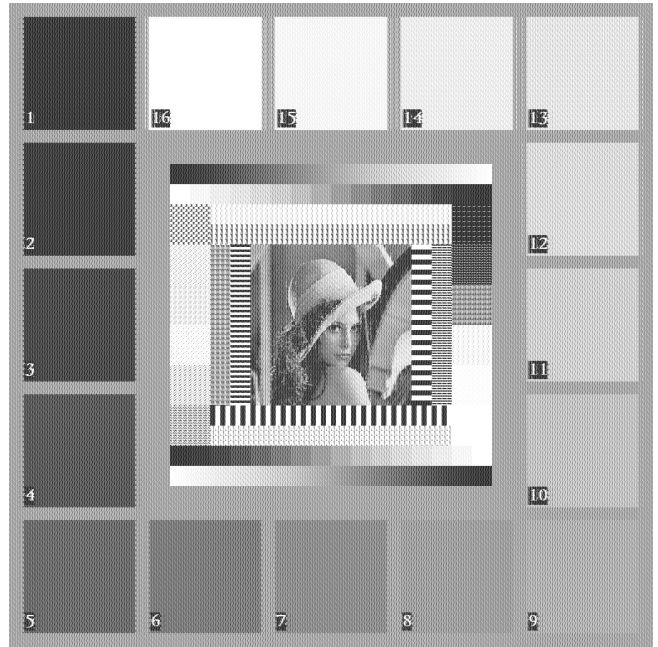


Fig. 3. Reconstructed Testpat.1k from blocks of sizes  $5 \times 5$  to blocks of sizes  $8 \times 8$  ( $\rho = 0.39$ )

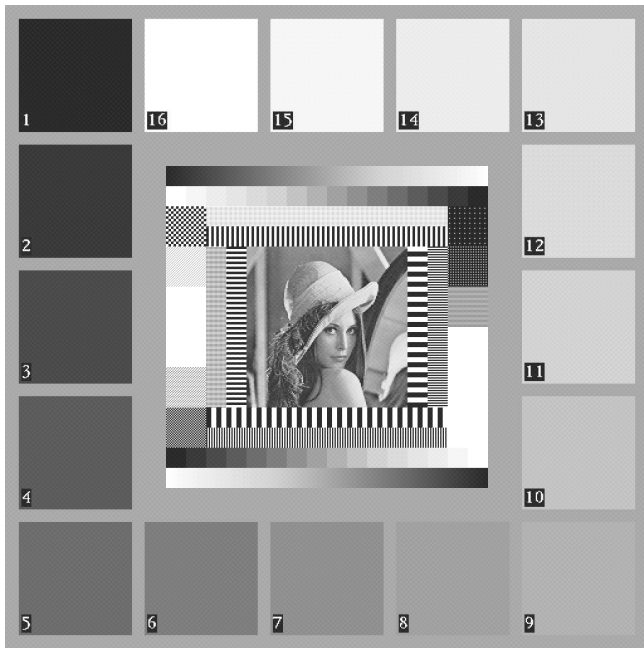


Fig. 2. Reconstructed Testpat.1k from blocks of sizes  $2 \times 1$  to blocks of sizes  $2 \times 2$  ( $\rho = 0.5$ )

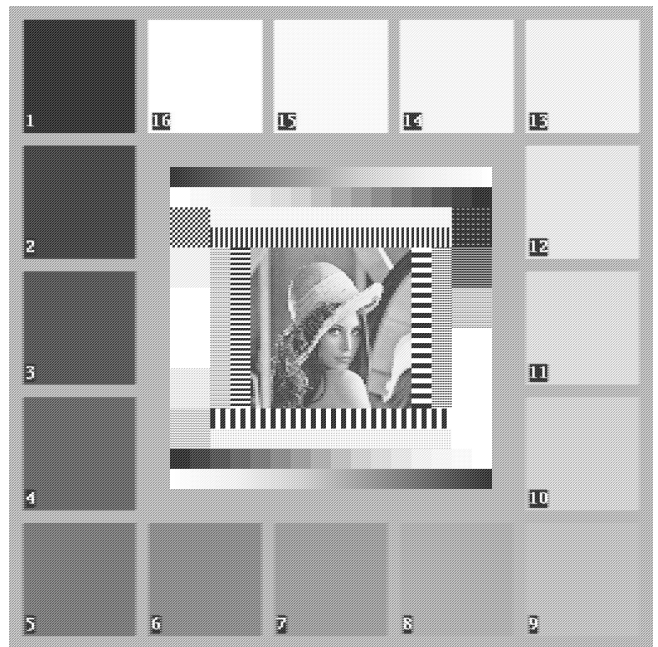


Fig. 4. Reconstructed Testpat.1k from blocks of sizes  $2 \times 2$  to blocks of sizes  $4 \times 4$  ( $\rho = 0.25$ )

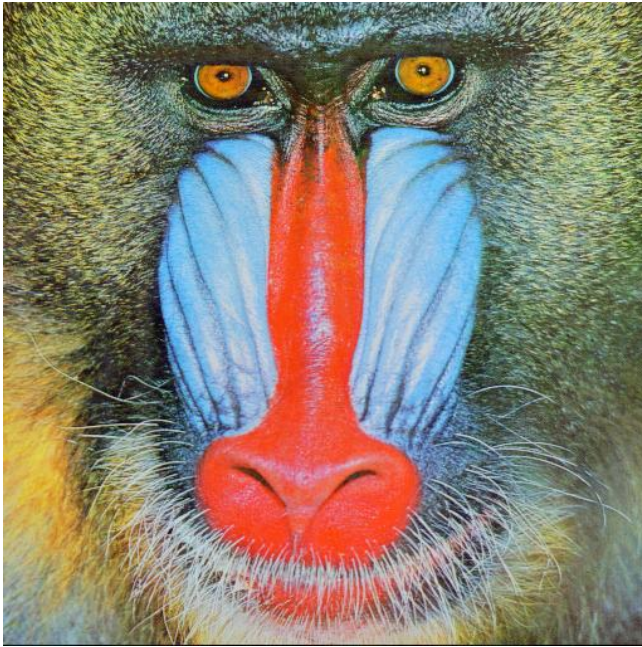


Fig. 5. Mandrill

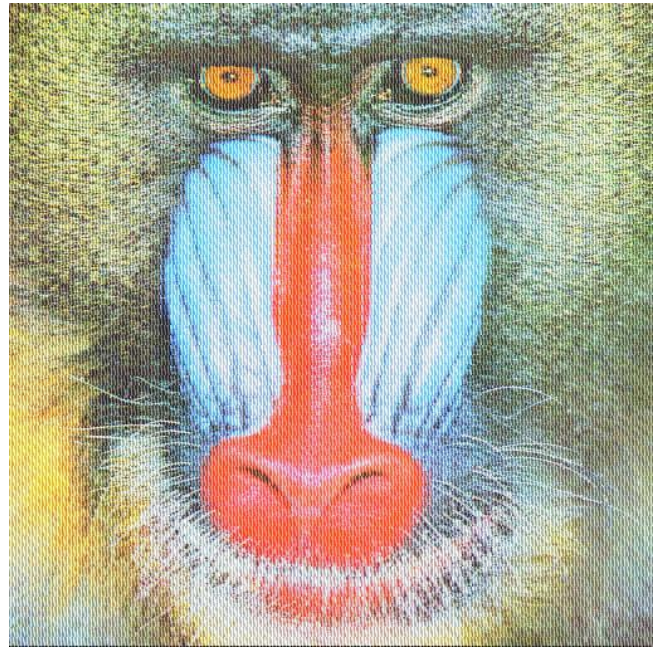


Fig. 7. Reconstructed Mandrill from blocks of sizes  $5 \times 5$  to blocks of sizes  $8 \times 8$  ( $\rho = 0.39$ )

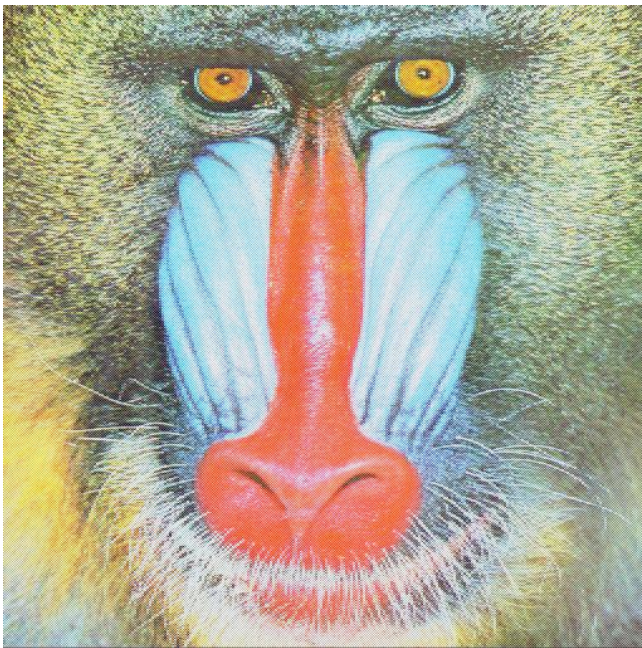


Fig. 6. Reconstructed Mandrill from blocks of sizes  $2 \times 1$  to blocks of sizes  $2 \times 2$  ( $\rho = 0.5$ )

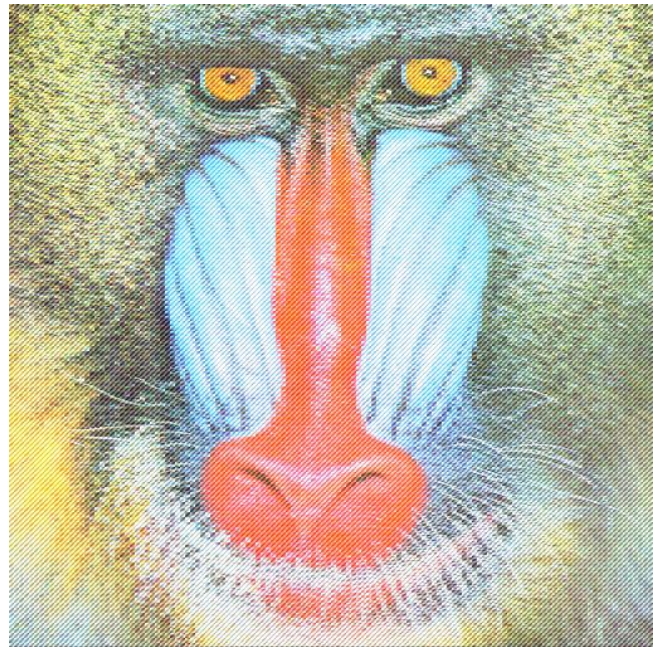


Fig. 8. Reconstructed Mandrill from blocks of sizes  $2 \times 2$  to blocks of sizes  $4 \times 4$  ( $\rho = 0.25$ )