Novel Representation of Multidimensional Datasets: The Framework *n*D-EVM/Kohonen

Ricardo Pérez-Aguila *Ricardo Ruiz-Rodríguez*

***Abstract* - In this paper we are going to describe the steps that conform a novel approach for representation of multidimensional datasets through the proposed Framework *n*D-EVM/Kohonen. In this sense two phases are going to be taken in account: 1) the application of 1-Dimensional Kohonen Self-Organizing Maps (1D-KSOMs) in order to achieve (*n*-1)D hypervoxelizations' segmentations, $n \geq 2$, taking in account their geometrical and topological properties to characterize the information contained in the datasets. 2) The segmented multidimensional datasets are specified as Orthogonal Polytopes whose *n*-th Dimension is associated to a 1D-KSOM classification. Subsequently, the *n*D representation is concisely expressed via the Extreme Vertices Model in the *n*-Dimensional Space (*n*D-EVM). There is presented a comparative analysis based on the use of False Color Maps in order to understand they way our considered 1D-KSOMs distribute appropriately their weights vectors along the classification space, even better than classifications based exclusively on color intensity. Additionally, we present some arguments to sustain that a 1D-KSOM is an adequate option in terms of temporal complexity and, on the other hand, that our representation is concise in terms of spatial complexity because of the *n*D-EVM properties.**

***Index Terms* - Representation and Manipulation of Hypervoxelizations, Polytopes Representation Schemes, Geometrical and Topological Interrogations, 1-Dimensional Kohonen Self-Organizing Maps, False Color Maps**

I. Introduction

The representation of a polytope through a scheme of Hyperspatial Occupancy Enumeration is a list of identical hyperspatial cells occupied by the polytope. Specific types of cells, called hypervoxels [10] are Hyper-Boxes (HBs) of a fixed size that lie in a fixed grid in the (*n*-1)-Dimensional space, $n \geq 2$. The collection of HBs can be codified as an (*n*-1)D array $C_{x_1,x_2,...,x_{n-1}}$. The array represents the coloration of each hypervoxel. If $C_{x_1,x_2,...,x_{n-1}} = 0$, the *black* hypervoxel $C_{x_1,x_2,...,x_{n-1}}$ represents an unoccupied region. If $C_{x_1,x_2,...,x_{n-1}} = k$, where $k > 0$ is in a given color scale (black & white, grayscale, RGB, etc.), then the occupied hypervoxel $C_{x_1,x_2,...,x_{n-1}}$ represents an used region from the (*n*-1)D space with intensity *k*. In fact, the set of occupied cells defines an orthogonal polytope *p* whose vertices coincide with some of the occupied cells' vertices.

Manuscript received December 17, 2014; revised April 10, 2015.

Ricardo Pérez-Aguila is member of the Group for Multidisciplinary Research Applied to Education and Engineering (GIMAEI) at the Universidad Tecnológica de la Mixteca (UTM), Carretera Huajuapan-Acatlima Km. 2.5, Huajuapan de León, Oaxaca 69004, México (e-mail: ricardo.perez.aguila@gmail.com).

Ricardo Ruiz-Rodríguez is member of the Group for Multidisciplinary Research Applied to Education and Engineering (GIMAEI) at the Universidad Tecnológica de la Mixteca (UTM), Carretera Huajuapan-Acatlima Km. 2.5, Huajuapan de León, Oaxaca 69004, México (e-mail: ricardo.ruizrodriguez@gmail.com).

In the representation through an array, the spatial complexity of a (*n*-1)D hypervoxelization is at least $\prod_{i=1}^{n-1} n_i$ where n_i is the length of the grid along X_i-axis. Furthermore, depending on the considered color scale hypervoxels would have additional storing requirement: if the color space is RGB then each one requires three bytes (four, if the alpha channel is considered) for codifying its corresponding intensity. Some capture devices represent natively datasets through hypervoxelizations. But sometimes their storing requirements make difficult their manipulation. Several efforts have been made in order to reduce the spatial complexity of datasets. In [11] is presented an algorithm for compression by means of quadtrees in order to encode slices of data. Such encodings are used for discovering similarities between consecutive slices. In [30], 3D medical datasets are compressed via a method sustained in the use of octrees. This pair of works share us evidence of the spatial conciseness provided by considering the use of Solid Representation Schemes.

There is also a critical point to be boarded which leads in turn to the proper steps to be followed by us: the consideration of an additional approach in such way the points in a dataset could be characterized by taking in account not only their color intensity. Due to the presence of scanning noise and artifacts, a classification based only in intensities is sometimes not enough. Let an intelligent approach, such as an Artificial Neural Network (ANN), be responsible for automatically identifying the classes of points present in a multidimensional dataset by taking in account their geometry, topology, neighborhood, etc. By considering more appropriate classification of points, a much better conciseness it is expected to be obtained.

This work is devoted to describe an alternative representation for datasets whose color scales are not binary (in [15] is presented a methodology designed specifically for datasets with black & white color scale). The main idea is to apply a 1-Dimensional Kohonen Self-Organizing Map (1D-KSOM) in order to segment datasets taking in account their geometry and topology. In this last sense, KSOMs-based segmentation has been a topic under attack by several researchers. In [32] is presented the Moving Average Self-Organizing Map (MA-SOM) which is applied for segmenting medical images obtained by means of X-Ray Computer Tomography, Breast Ultrasound Imaging, and Magnetic Resonance Imaging. In [2], Kohonen Networks are used together with a threshold technique in order to segment satellite images. The paper by Jiang et al [9] presents a survey of application of ANNs for medical support in processing images in tasks such as Image Segmentation, Edge Detection, and Computer Aided Diagnosis by means of application of the Multilayer Perceptrons, Hopfield Networks, and SOMs, among others. All these works give us evidence of how the KSOMs

approach has gained confidence among Image Processing researchers thanks to valuable properties such as robustness to noise, plasticity, parallelism, and computational efficiency. In the context of our proposed framework, we take an additional step: the specification of (n-1)D datasets, $n \geq 2$, as nD polytopes where the n-th Dimension is associated precisely to a 1D-KSOM's provided classification. Then, the nD representation is concisely expressed and manipulated through a polytopes' representation scheme: the Extreme Vertices Model in the n-Dimensional Space (nD-EVM).

This work is structured as follows: The **Section II** will describe the fundamentals behind the nD-EVM. **Section III** shows the basics behind the 1D-KSOMs. In **Section IV** it is described the way a 1D-KSOM assists us in the automatic non-supervised segmentation of a dataset. In the **Section V** it is mentioned how the classification provided by our 1D-KSOMs sustain the methodology for conversion of a (n-1)D hypervoxelization dataset to a nD-EVM. There will be presented examples of datasets finally expressed under the nD-EVM. By comparing the storing requirements between the original and the nD-EVM representations it is established the power of conciseness of our proposal. In **Section VI** we present a comparative analysis in order to sustain some claims related to our proposed 1D-KSOM based segmentation. Finally, **Section VII** presents conclusions and lines for future research.

II. Polytopes Modeling by Means of the Extreme Vertices Model in the n-Dimensional Space (nD-EVM)

This section is a summary of results originally presented in [1] & [14]. For the sake of brevity, only the required concepts for the purpose of this work are presented and some propositions are only enunciated. Formal details and proofs can be found in the two abovementioned references.

A. The n-Dimensional Orthogonal Pseudo-Polytopes

Definition 2.1: A nD Singular HB in the nD Euclidean Space is the continuous function

$$\begin{array}{rccc} I^n: & [0,1]^n & \rightarrow & [0,1]^n \\ & x & \mapsto & I^n(x) = x \end{array}$$

Definition 2.2: For i = 1, 2, ..., n the (n-1)D singular HBs $I^n_{(i,0)}$ and $I^n_{(i,1)}$ are stated as: If $x \in [0,1]^{n-1}$ then

- $I^n_{(i,0)}(x) = (x_1,...,x_{i-1},0,x_i,...,x_{n-1})$
- $I^n_{(i,1)}(x) = (x_1,...,x_{i-1},1,x_i,...,x_{n-1})$

Definition 2.3: In a nD singular general HB c the (i,α)-cell is defined as $c_{(i,\alpha)} = c \circ I^n_{(i,\alpha)}$.

Definition 2.4: Orientation of cell $c \circ I^n_{(i,\alpha)}$ is set by $(-1)^{\alpha+i}$.

Definition 2.5: An (n-1)D oriented cell is given by the scalar-function product $(-1)^{i+\alpha} \cdot c \circ I^n_{(i,\alpha)}$.

Definition 2.6: A formal linear combination of kD singular general HBs, $1 \leq k \leq n$, for a closed set A is called a k-chain.

Definition 2.7 [31]: Given a nD singular HB I^n the (n-1)-chain, called the boundary of I^n, is given by

$$\partial(I^n) = \sum_{i=1}^{n} \left(\sum_{\alpha=0,1} (-1)^{i+\alpha} \cdot I^n_{(i,\alpha)} \right)$$

Definition 2.8 [31]: Given a nD singular general HB c the (n-1)-chain, called boundary of c, is defined by

$$\partial(c) = \sum_{i=1}^{n} \left(\sum_{\alpha=0,1} (-1)^{i+\alpha} \cdot c \circ I^n_{(i,\alpha)} \right)$$

Definition 2.9 [31]: The boundary of n-chain $\sum c_i$, where each c_i is a nD singular general HB, is given by

$$\partial\left(\sum c_i\right) = \sum \partial(c_i)$$

Definition 2.10: A collection c_1, c_2, …, c_k, $1 \leq k \leq 2^n$, of nD singular general HBs is a combination of nD HBs iff

$$\left[\bigcap_{\alpha=1}^{k} c_\alpha([0,1]^n) = (\underbrace{0,...,0}_{n}) \right] \wedge$$

$$\left[(\forall i, j,\ i \neq j,\ 1 \leq i, j \leq k)\left(c_i([0,1]^n) \neq c_j([0,1]^n)\right) \right]$$

The first part of the conjunction says the intersection between all the nD singular general HBs is the origin, while the second part states there are not overlapping nD HBs.

Definition 2.11: An nD Orthogonal Pseudo-Polytope p, or just an nD-OPP p, is an n-chain composed by nD HBs arranged in such way that by selecting a vertex, in any of these HBs, we have that such vertex describes a combination of nD HBs composed up to 2^n HBs.

B. The nD-EVM: Foundations

Definition 2.12 [14]: Let c be a combination of HBs in the nD Euclidean Space. An **Odd Edge** is an edge with an odd number of incident HBs of c.

Definition 2.13 [14]: A **Brink**, or extended edge, is the maximal uninterrupted segment, built out of a sequence of collinear and contiguous odd edges of an nD-OPP.

Definition 2.14 [14]: The **Extreme Vertices** of an nD-OPP p are the ending vertices of all the brinks in p.

Definition 2.15 [14]: Let p be an nD-OPP. The **Extreme Vertices Model** of p, denoted by $\mathbf{EVM_n(p)}$, is defined as the model as only stores to all the extreme vertices of p.

C. Sections and Slices of nD-OPPs

Definition 2.16: Let p be an nD-OPP. A ***k*D couplet** of p, $1 < k < n$, is the maximal set of kD cells of p that lies in a kD space, such that a kD cell e_0 belongs to a kD extended hypervolume iff e_0 belongs to an (n-1)D cell present in $\partial(p)$.

Definition 2.17: The Projection Operator for (n-1)D cells, points, and set of points is respectively defined as follows:

- Let $c(I^n_{(i,\alpha)}(x)) = (x_1,..,x_n)$ be an (n-1)D cell embedded in the nD space. $\pi_j\left(c(I^n_{(i,\alpha)}(x))\right)$ will denote the projection of the cell $c(I^n_{(i,\alpha)}(x))$ onto an (n-1)D space embedded in nD space whose supporting hyperplane is perpendicular to X_j-axis: $\pi_j\left(c(I^n_{(i,\alpha)}(x))\right) = (x_1,...,\hat{x}_j,...,x_n)$.
- Let $v = (x_1,...,x_n)$ a point in the nD space. The projection of v in the (n-1)D space, denoted by $\pi_j(v)$, is given by $\pi_j(v) = (x_1,...,\hat{x}_j,...,x_n)$.
- Let Q be a set of points in nD space whose projection, denoted by $\pi_j(Q)$, is defined as the set in (n-1)D space such that $\pi_j(Q) = \left\{ p \in \Re^{n-1} : p = \pi_j(x),\ x \in Q \right\}$.

Where $\hat{x}_j$ is the coordinate corresponding to X_j-axis to be suppressed.

Definition 2.18: Consider an nD-OPP p:

- Let np_i be the number of distinct coordinates present in the vertices of p along X_i-axis, $1 \leq i \leq n$.
- Let $\Phi^i_k(p)$ be the k-th (n-1)D couplet of p which is perpendicular to X_i-axis, $1 \leq k \leq np_i$.

Theorem 2.1 [14]: The projection of the set of (n-1)D-couplets, $\pi_i\left(\Phi_k^i(p)\right)$, of an nD-OPP p, can be obtained by computing the regularized XOR ($\otimes$) between the projections of its previous $\pi_i\left(S_{k-1}^i(p)\right)$ and next $\pi_i\left(S_k^i(p)\right)$ sections, i.e.,

$$\pi_i\left(\Phi_k^i(p)\right) = \pi_i\left(S_{k-1}^i(p)\right) \otimes^* \pi_i\left(S_k^i(p)\right), \ \forall k \in [1, np_i]$$

Theorem 2.2 [14]: The projection of any section, $\pi_i\left(S_k^i(p)\right)$, of an nD-OPP p, can be obtained by computing the regularized XOR between the projection of its previous section, $\pi_i\left(S_{k-1}^i(p)\right)$, and the projection of its previous couplet $\pi_i\left(\Phi_k^i(p)\right)$.

D. The Regularized XOR operation on the nD-EVM

Theorem 2.3 [1]: Let p and q be two nD-OPPs having $EVM_n(p)$ and $EVM_n(q)$ as their respective EVMs in nD space, then $EVM_n(p \otimes^* q) = EVM_n(p) \otimes EVM_n(q)$

This result allows expressing a formula for computing nD-OPPs' sections from couplets and vice-versa, by means of their corresponding EVMs. These formulae are obtained by combining **Theorem 2.3** with **Theorem 2.1**; and **Theorem 2.3** with **Theorem 2.2**, respectively:

Corollary 2.1 [1]:
$$EVM_{n-1}\left(\pi_i(\Phi_k^i(p))\right) = EVM_{n-1}\left(\pi_i(S_{k-1}^i(p))\right) \otimes EVM_{n-1}\left(\pi_i(S_k^i(p))\right)$$

Corollary 2.2 [1]:
$$EVM_{n-1}\left(\pi_i(S_k^i(p))\right) = EVM_{n-1}\left(\pi_i(S_{k-1}^i(p))\right) \otimes EVM_{n-1}\left(\pi_i(\Phi_k^i(p))\right)$$

Corollary 2.3 [1]: Let p and q be two disjoint or quasi disjoint nD-OPPs having $EVM_n(p)$ and $EVM_n(q)$ as their respective Extreme Vertices Models, then

$$EVM_n(p \cup q) = EVM_n(p) \otimes EVM_n(q)$$

III. A Survey on 1-Dimensional Kohonen Self-Organizing Maps (1D-KSOMs)

A Kohonen Network with ψ inputs and O neurons may be used to classify points embedded in an ψ-Dimensional space into O categories ([5] & [23]). Input points have the form $(x_1, \ldots, x_i, \ldots, x_\psi)$. Each neuron j, $j = 1, 2, \ldots, O$, has associated an ψ-Dimensional Weights Vector which has a representation of its corresponding class κ_j. All these vectors have the form $W_j = (w_{j,1}, \ldots, w_{j,\psi})$, $j = 1, 2, \ldots, O$.

A set of training points are presented to the network T times. According to [8], all values of Weight Vectors should be randomly initialized. In the t-th presentation, $t = 1, 2, \ldots, T$, the neuron whose Weights Vector W_j, $1 \le j \le O$, is the most similar to the input point P is chosen as Winner Neuron. In the model proposed by Kohonen, such selection is based on the Squared Euclidean Distance. The selected neuron is that with the minimal distance between its Weights Vector and the input point P:

$$d_j = \sum_{i=1}^{\psi}\left(P_i - W_{j,i}(t)\right)^2 \qquad 1 \le j \le O$$

Once the ℓ-th Winner Neuron, $1 \le \ell \le O$, in the t-th presentation, has been identified each one of the network's Weights Vectors is updated according to:

$$W_{j,i}(t+1) = W_{j,i}(t) + \frac{1}{t+1}\varphi(j,\ell)\left[P_i - W_{j,i}(t)\right] \qquad \begin{matrix} i = 1,2,...,\psi \\ j = 1,2,...,O \end{matrix}$$

Where the term $1/(t + 1)$ is the learning coefficient and $\varphi(j, \ell)$ is a neighborhood function that denotes the distance between the Winner Neuron ℓ and the neuron j. For neurons close enough to the Winner Neuron, $\varphi(j, \ell)$ should be a value near to 1. On the other hand, $\varphi(j, \ell)$ is close to zero for those neurons characterized as distant to the Winner Neuron. When the T presentations have been achieved, the values of the Weights Vectors correspond to coordinates of the '*gravity centers*' of the clusters of the O categories.

IV. Multidimensional Hypervoxelizations' Segmentation Achieved by 1D-KSOMs

The automatic non-supervised segmentation based on the assistance of a 1D-KSOM shares to identify, during its training processes, the proper representations for a previously established number of classes. A dataset can be segmented in such way each type of region is appropriately identified. Many methods for description, object recognition or indexing are sustained on a preprocessing based on automatic segmentation [12], [34] & [35]. This **Section** describes our methodology, also presented in [22], which is inspired in some facts established originally in [16].

It is clear each hypervoxel has an intensity which, it is understood, captures, or is associated, to a particular property (a type of tissue, material, etc.); however, it is important to consider the hypervoxels around it. The neighborhood surrounding a given hypervoxel complements the information about the properties to be identified [13] & [20]. Let v be a hypervoxel in an (n-1)D dataset, $n \ge 2$. Given v it is possible to build a subdataset by taking those hypervoxels inside a hypercubical neighborhood of radius r and center at v. Hypervoxel v and its neighboring hypervoxels is called a (n-1)-Dimensional mask.

The network's training set is composed by all the (n-1)D masks of radius r that can be generated in a given (n-1)D dataset. The total number of masks of radius r extracted from a (n-1)D dataset with size $n_1 \times n_2 \times \ldots \times n_{n-1}$ is given by

$$(n_1 - 2r)\cdot(n_2 - 2r)\cdot \ldots \cdot(n_{n-1} - 2r)$$

Precisely this formula give us the size of our training sets.

As commented in **Section III** a 1D-KSOM expects as input a vector, or point, embedded in the ψ-Dimensional Space. A (n-1)D mask is formerly seen as a (n-1)D array, but it is clear by stacking its columns on top of one another a vector is obtained. This straightforward procedure *linearizes* a (n-1)D mask making it a suitable input for the network. The resulting vectors, composed by

$$\psi = (2r + 1)^{(n-1)}$$

components, preserve the properties associated to the original (n-1)D masks. Each component has the color intensity of the corresponding hypervoxel in its associated (n-1)D mask. By this way, we have specified the elements that conform our proposed approach for automatic segmentation of (n-1)D hypervoxelizations. We proceed now to describe some experiments and the obtained results.

Our study cases are based in 3D datasets described in **Table I**. Such datasets were taken from *The Volume Library* [25], and the University of Iowa's Department of Radiology [6]. Datasets' visualizations shown in **Table I** were obtained via a volume rendering software available at [33].

The implemented 1D-KSOMs have the following topologies and training conditions:

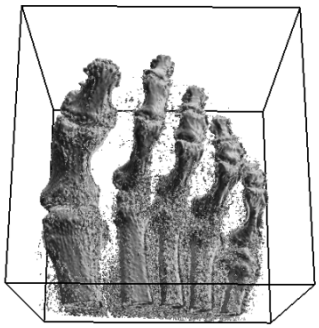
- 3D masks radius: $r = 3$
- Inputs: $\psi = 7 \times 7 \times 7 = 343$
- Neurons (classes): $O = 80$
- Presentations: $T = 10$

According to previous **Section** we must define a neighborhood function. We will use following rule:

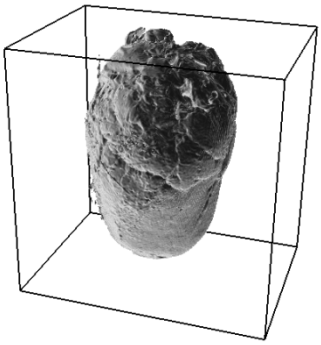
$$\varphi(j,\ell)=\begin{cases}1 & j=\ell\\ 0 & j\neq\ell\end{cases}$$

That is, once the Winner Neuron ℓ has been identified, only its weights are updated. In **Table II** we can see the corresponding cardinalities of the training sets associated to each considered voxelization.

TABLE I
VOLUME DATASETS TO BE CONSIDERED FOR AUTOMATIC SEGMENTATION VIA 1D-KSOMS.



Foot. Rotational C-arm x-ray scan of a human foot. Tissue and bone are present in the dataset. Voxelization size: $(256 \times 256 \times 256) \equiv 16{,}777{,}216$



VL-Sheep. MRI scan of a heart's sheep. Voxelization size: $(352 \times 352 \times 256) \equiv 31{,}719{,}424$

TABLE II
TRAINING SETS' CARDINALITIES FOR VOXELIZATIONS DESCRIBED IN TABLE I.

Dataset	Voxelization Size (Number of voxels)	Training Set's Cardinality
Foot	16,777,216	15,625,000
VL-Sheep	31,719,424	29,929,000

The **Table III** shows the obtained distribution of the training 3D masks in each one of the available 80 classes. According to this **Table**, for networks trained with 3D masks associated to datasets *Foot* and *VL-Sheep* only 72 and 58 classes are respectively used. A second observation arises: in the two proposed networks there is a class which has associated more than 65% of the elements in its corresponding training set. Seeing **Table III** it is possible to identify to such classes:

- Dataset *Foot*: class 32, 66.4553%
- Dataset *VL-Sheep*: class 54, 75.5389%.

As we will see with more detail in **Section V**, we can mention in advance these abovementioned classes grouped 3D masks corresponding to empty regions in the original datasets. Members in these classes are precisely part of the point of discussion relative to the fact hypervoxelizations' spatial complexities make sometimes difficult their manipulation and storage: under hypervoxelization representation we are also dealing with the processing and '*cargo space*' of empty regions. However, we will see in next **Section** by considering the EVM model, how these regions can be taken in account in the representation without impacting spatial conciseness.

TABLE III
CLASSIFICATION OF 3D MASKS ACCORDING TO 1D-KSOMS WITH 343 INPUTS, 80 OUTPUT NEURONS, AND 10 PRESENTATIONS.

Class	*Foot*	*VL-Sheep*	Class	*Foot*	*VL-Sheep*
1	3,112	34	**41**	14,770	0
2	10,667	0	**42**	6,348	5
3	312	0	**43**	26,460	38,782
4	23,667	0	**44**	1,404	675
5	58,416	389,056	**45**	87	4
6	44,806	48,680	**46**	48,289	702,629
7	138,048	548	**47**	14	0
8	3,013,498	59,225	**48**	6,215	8
9	1,749	0	**49**	17	89,659
10	10,956	141,278	**50**	35,728	1
11	808	0	**51**	12,825	0
12	280	152,233	**52**	31,285	18,724
13	872,443	430	**53**	63,299	1
14	53	21	**54**	10	22,608,065
15	8,376	29,684	**55**	277	8
16	2,801	24	**56**	898	0
17	2,122	3	**57**	3,037	0
18	456	1	**58**	60,161	0
19	129	1	**59**	8	13
20	73,284	1	**60**	14	0
21	8,177	0	**61**	2	47
22	111,767	0	**62**	102	0
23	56	530,516	**63**	1,306	0
24	21,907	1	**64**	638	222,877
25	5	133	**65**	659	0
26	555	1189	**66**	1	56,789
27	25	0	**67**	11	7
28	17	1,322,902	**68**	77	20,987
29	454	3	**69**	44	1,979
30	6	1	**70**	225	14
31	1,955	111,352	**71**	3,449	0
32	10,383,656	0	**72**	23,700	1
33	44	71,241	**73**	363	3
34	1,143	1	**74**	166,745	1,035,368
35	113,825	0	**75**	263	51
36	2,847	33,647	**76**	1,397	127
37	148,771	595	**77**	153	0
38	356	235	**78**	13,226	239,181
39	37,209	0	**79**	145	2
40	2,533	30	**80**	57	1,999,928

We conclude this **Section** by giving some words about time complexity associated to 1D-KSOMs' training process. As seen in **Section III** the training algorithm is by definition iterative and the number of executed steps depends on the following parameters: the number T of presentations, the number O of classes, the number ψ of input neurons, the chosen neighborhood function, and finally, the training set's size $s = (n_1 - 2r)\cdot(n_2 - 2r)\cdot \ldots \cdot(n_{n-1} - 2r)$. We suppose the Winner Neuron is identified by a linear search and because of our used neighborhood function only the Winner Neuron has its weights updated. Then, the total number of steps executed for training a 1D-KSOM in order to process one of our multidimensional datasets is given by:

$$Time(s, T, O, \psi) = T\cdot s\cdot O\cdot\psi$$

Factor ψ tightly bounds the time required for computing the Euclidean Distance between input and weights vectors and also the time for updating the Winner Neuron's weights. The number ψ of inputs for the 1D-KSOM depends on the radius r and the dimensionality of the source hypervoxelization. Hence, function $Time(s, T, O, \psi)$ is rewritten as:

$$Time(s, T, O, r, n) = T\cdot s\cdot O\cdot(2r+1)^{(n-1)}$$

Values shared by our time function correspond to the number of executed steps in order to train a 1D-KSOM. The topic related to 1D-KSOMs' time complexity is well known and it has been widely boarded in the literature. We make the mention in order to point out the required time, from an asymptotically point of view and under our current

application context, is in fact impacted mainly by the training sets' sizes and the considered radius r. Considering sufficiently large values for (n-1)D datasets' lengths $n_1, n_2, \ldots, n_{n-1}$ and by taking in account in the practice values for 1D-KSOMs parameters T and O are lower, and constant, then we have the required time for training a 1D-KSOM in order to classify a set of (n-1)D masks is given by:

$$Time(r, n_1, n_2, \ldots, n_{n-1}) = (n_1 - 2r)\cdot(n_2 - 2r)\cdot \ldots \cdot(n_{n-1} - 2r)\cdot(2r + 1)^{(n-1)}$$

In other words, the operations performed during a 1D-KSOM's training are in fact simple and efficient and it is the hypervoxelization's size which establish the final required time. Hence, these observations give us arguments to sustain the claim a 1D-KSOMs is an adequate choice, in terms of time complexity, for building, in an automatic scheme, a segmentation for our working datasets.

V. The Framework Based on 1D-KSOMs and the nD-EVM

The 1D-KSOMs, defined in the above **Section**, for processing multidimensional datasets have provided us with segmentations by establishing a number of classes and a size for the (n-1)D masks that describe the used training sets. As commented previously, each one of these (n-1)D masks describes in fact a subdataset by considering a hypervoxel of interest and its hypercubical neighborhood of radius r. Then, the masks are sent as input to a 1D-KSOM and through the corresponding training process there are determined their associated classes. Once the training has finished given a (n-1)D mask as input it is obtained as output the number of its corresponding class. Our hypervoxelizations were originally expressed with K color intensities, however, we have now through a 1D-KSOM these K intensities have been mapped onto a set of O elements: the number of classes established for training the 1D-KSOM. Therefore, the color intensity value for a hypervoxel can now be replaced by its corresponding number of associated class given as output by the network. The hypervoxelization's original size is not affected except on its boundaries because masks of radius r cannot be completed. For instance, to consider a hypervoxelization again as the appropriate representation for storing the segmentation achieved by means of a 1D-KSOM does not provides a reduction in terms of spatial complexity.

Now we introduce our process, also mentioned in [22] for expressing (n-1)D hypervoxelizations using the nD-EVM, the representation scheme described in **Section II**. The conversion of a (n-1)D hypervoxelization to a nD polytope, and therefore to a nD-EVM, is in fact a straight procedure. Some aspects of the methodology to be described in the next two paragraphs were originally presented, for the 4D case, in [18].

A nD hyperprism is a polytope generated by the parallel motion of an (n-1)D polytope; it is bounded by the (n-1)D polytope in its initial and final positions and by several (n-1)D hyperprisms [4] & [29]. Then, each hypervoxel in a (n-1)D dataset is extruded towards the n-th Dimension, that is, it is converted in a nD hyperprism whose bases are precisely the original hypervoxel and its height is given by its corresponding class number which was generated as output by the corresponding 1D-KSOM. The vertices' coordinates $X_1, X_2, \ldots, X_{n-1}$ in the hyperprism's bases correspond to the original hypervoxel's coordinates. The inferior base's points have their X_n coordinate equal to zero, while in the remaining vertices the X_n coordinate is equal to the 1D-KSOM class number value. Let us call xf to the set composed by the nD hyperprisms (the extruded hypervoxels) of the extruded (n-1)D dataset.

Let pr_i be a nD hyperprism in xf and npr the number of hyperprisms in that set: npr is in fact equal to the number of hypervoxels in the original dataset. Since all hyperprisms in xf are quasi disjoint nD-OPPs, the extreme vertices, of the whole nD extruded dataset, can be easily obtained by computing the regularized union of all the hyperprisms in xf. Hence, **Corollary 2.3** is applied in the following way:

$$EVM_n(F) = \bigotimes_{i=1}^{npr} EVM_n(pr_i \in xf)$$

Where F is the nD-OPP that represents the union of all the hyperprisms in xf. Therefore, it is obtained a representation for a (n-1)D Dataset through a nD-OPP and the EVM.

Our proposed **Framework *n*D-EVM/Kohonen** takes place when a dataset's segmentation achieved by a 1D-KSOM is represented by using the nD-EVM, considering the conversion methodology abovementioned taking in account the class numbers generated as output by the corresponding 1D-KSOM. From a 'nD *point of view*', the X_n-axis refers to the number of class which is associated a hypervoxel respect to the segmented dataset.

Summarizing, the following is the workflow, also presented in [22], for our **Framework *n*D-EVM/Kohonen**:

Input: A (n-1)D hypervoxelization under K color intensities, $n \geq 2$, $K \geq 3$.

Step 1: Given value r for radius of (n-1)D masks it is built a training set.

Step 2: Given values T and O for number of presentations and number of classes, respectively, it is trained a 1D-KSOM using as training set all the masks generated in the previous step.

Step 3: The training set is used again as input for the 1D-KSOM with adjusted weights. Each (n-1)D mask is sent as input to the 1D-KSOM and it is obtained the corresponding output: the number of its associated class. Coordinates, respect to the original hypervoxelization, of central hypervoxel in the input mask, are used together with class number for building a nD hyperprism.

Step 4: All generated hyperprisms are integrated in such way it is obtained the corresponding whole nD-OPP which in turn is expressed in the nD-EVM.

Output: The final nD-EVM.

The importance behind a dataset is the information can be obtained from it. If the datasets are represented through the nD-EVM, via the **Framework *n*D-EVM/Kohonen**, then the extraction of its couplets perpendicular to X_n-axis corresponds to the classification of the elements in the original model according to their output provided by the corresponding 1D-KSOM. Given a 1D-KSOM trained for characterization in O classes, then it is possible to obtain at most, from its corresponding nD-OPP, O+1 (n-1)D couplets perpendicular to X_n-axis. The '*extra*' couplet is the result of the union of all the inferior bases of the nD hyperprisms in xf: the n-th coordinate in the points of such bases is zero.

The **Tables IV** and **V** show some 3D couplets, perpendicular to X_4-axis, obtained from our 3D datasets, *Foot* and *VL-Sheep*, respectively, after the application of **Framework *n*D-EVM/Kohonen**. The 3D-EVM views from these **Tables** were achieved by means of visualization software developed in [26], [27] & [28]. The projection of 3D couplets under our 4D-EVM representation collect those

voxels that were grouped inside the same class by the corresponding 1D-KSOM. We can see, for example, in **Table IV**, how it is possible to appreciate couplets (specifically 14, 42, 79) where bone tissue is only present. In contrast, **Table IV**'s remaining 3D couplets, except couplet 33, describe particular groupings of skin, muscle, and nail tissue, among others.

In **Section IV** we made mention of the fact that our considered 1D-KSOMs were grouping more than 65% of the elements in their training sets in just one class. This can be visually appreciated in 3D couplets 33 and 41 from **Tables IV** and **V**, respectively. These 3D couplets present a block-like structure whose minimal bounding box has the same dimensions that the original voxelizations minus the volume removed by those 3D masks of radius 3 that could not be completed at the datasets' boundaries. These are precisely the regions that correspond to empty space in the original voxelizations. In **Section IV** we gave an argument regarding the observation that these 3D couplets are considered in our 4D-EVM representation. We also mentioned this empty space is precisely one of the disadvantages behind a voxelization because it only contributes to spatial complexity. Our argument serves for the purpose of touching a crucial point: the Spatial Complexity of the representations achieved by the **Framework *n*D-EVM/Kohonen**. The **Table VI** shows the cardinalities for the 4D-EVMs obtained for expressing our 2 voxelizations. Let p be a dataset expressed under a (n-1)D hypervoxelization with size ($x_1Size \times x_2Size \times ... \times x_{n-1}Size$) and with $EVM_n(p)$ as its corresponding EVM after the application of our framework. Consider the ratio

$$\frac{x_1Size \cdot x_2Size \cdot ... \cdot x_{n-1}Size}{Card(EVM_n(p))}$$

See model *Foot* (**Table IV**). Its source voxelization has size ($256 \times 256 \times 256$) which implies that it is required to store 16,777,216 voxels. The 4D-EVM associated to *Foot* has 2,460,266 extreme vertices. Hence, the proposed ratio gives the value 6.8192 which tell us the number of stored voxels that belong to the original representation of the object is precisely 6.8192 times greater than the number of obtained extreme vertices. The **Table VI** shows the ratio Number-of-voxels/Number-of-Extreme-Vertices for the two models described in **Table I**.

Our other study case comes from dataset *VL-Sheep*: the original voxelization is formed by 31,719,424 voxels, while the **Framework *n*D-EVM/Kohonen** shared a 4D-EVM with 4,870,178 extreme vertices: giving the ratio 6.5129. We should take in account our representations are also considering the storing of the regions corresponding to empty space. This give us an advantage by storing in the same representation both the occupied and the empty space but without affecting the memory needed to store the models. These results share us evidence of the conciseness' power achieved when the grouping properties of 1D-KSOMs are combined with *n*D-EVM's storing properties: spatial complexity is reduced and a more appropriate segmentation is achieved, impacting how datasets' elements are classified and for instance accessed.

We attack the issue related to Temporal Complexity of our proposed Framework. Previously we made mention training of a 1D-KSOM for our purposes is, from an asymptotical point of view, dominated by the original (n-1)D dataset's size and the chosen radius for the mask that in turn defines its training set. For this reason, adjustment of 1D-KSOMs for the purpose of automatic segmentation is a very good choice because their associated training and evaluation algorithms are time efficient. What about the time required for building the final *n*D-EVM representation? We recall pr_i is a *n*D hyperprism in xf: the set composed by the *n*D hyperprisms, formerly the extruded hypervoxels, of the extruded (n-1)D dataset once the corresponding 1D-KSOM has been queried. We remember the generation of the corresponding *n*D-EVM is sustained in an application of **Corollary 2.3** producing the formula:

$$EVM_n(F) = \bigotimes_{i=1}^{npr} EVM_n(pr_i \in xf)$$

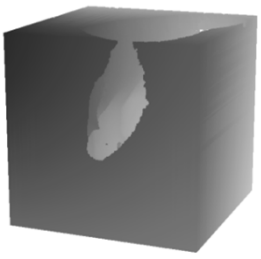
Which in turns depends of Regularized Boolean Operator XOR. In **Section II**, **Theorem 2.3** served as support for performing an XOR between two *n*D-EVMs. In [1], [14], and [26] this is implemented by keeping all vertices belonging to either $EVM_n(p)$ or $EVM_n(q)$ and by discarding any vertex that belongs to both $EVM_n(p)$ and $EVM_n(q)$. Moreover, since the EVM is a sorted model, the final procedure consists on a simple merging-like algorithm running on linear time [1], establishing temporal efficiency.

TABLE IV
SOME 3D COUPLETS, PERPENDICULAR TO X_4-AXIS, EXTRACTED FROM THE 4D-EVM ASSOCIATED TO DATASET *FOOT*.


3D Couplet 6


3D Couplet 9


3D Couplet 14


3D Couplet 33


3D Couplet 42


3D Couplet 54


3D Couplet 75

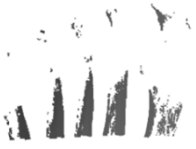
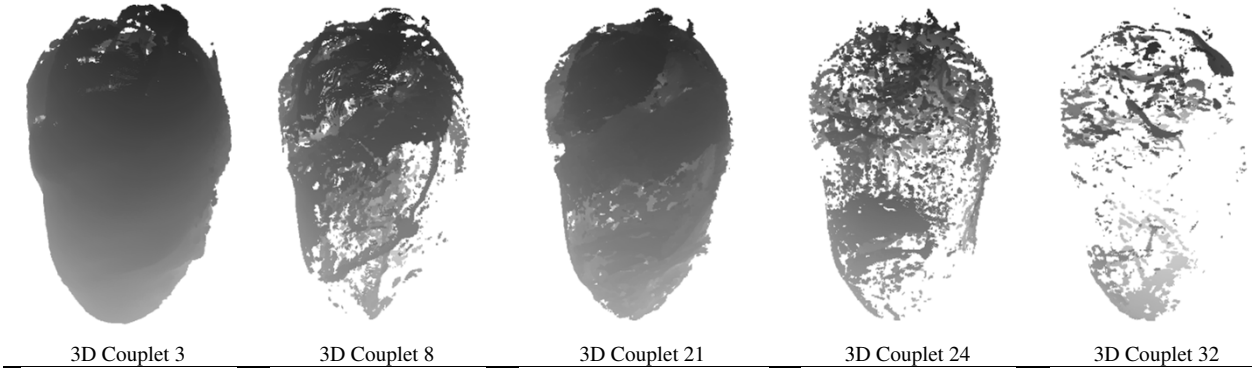
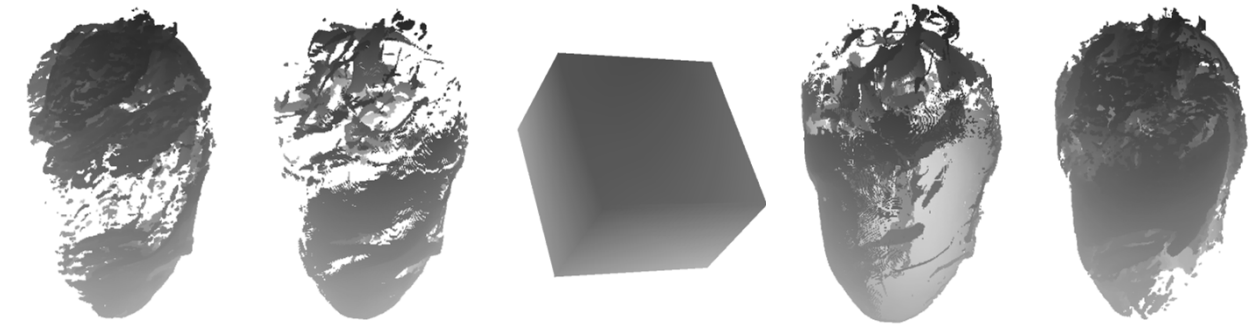

3D Couplet 79

TABLE V
SOME 3D COUPLETS, PERPENDICULAR TO X_4-AXIS, EXTRACTED FROM THE 4D-EVM ASSOCIATED TO DATASET *VL-SHEEP*.



3D Couplet 3 | 3D Couplet 8 | 3D Couplet 21 | 3D Couplet 24 | 3D Couplet 32



3D Couplet 35 | 3D Couplet 37 | 3D Couplet 41 | 3D Couplet 45 | 3D Couplet 58

TABLE VI
THE RATIO NUMBER-OF-VOXELS/NUMBER-OF-EXTREME-VERTICES FOR DATASETS SHOWN IN TABLE I AFTER PROCESSING VIA **FRAMEWORK *n*D-EVM/KOHONEN**.

Object p	**Voxelization Size (Number of voxels)**	**Card(EVM_4(p)) (Number of Extreme Vertices)**	$\frac{x_1 Size \cdot x_2 Size \cdot x_3 Size}{Card(EVM_4(p))}$
Foot	16,777,216	2,460,266	6.8192
VL-Sheep	31,719,424	4,870,178	6.5129

However, because of our specific application it is possible to reduce even more the XOR's temporal complexity by recurring to a *n*D-EVM implementation based on Trie Trees such as discussed in [14] (Trie Trees were originally introduced in the classic paper by [7]). The coordinates of an Extreme Vertex can be seen as a sequence of length *n*. Consider for example the set of Extreme Vertices in a 4D unit hypercube *h*. We have:

$EVM_4(h)$ = {(0,0,0,0), (0,0,0,1), (0,0,1,0), (0,0,1,1),
(0,1,0,0), (0,1,0,1), (0,1,1,0), (0,1,1,1), (1,0,0,0), (1,0,0,1),
(1,0,1,0), (1,0,1,1), (1,1,0,0), (1,1,0,1), (1,1,1,0), (1,1,1,1)}

Then, our sequences have length $n = 4$. Now we proceed to introduce these 16 points in a Trie Tree in such way each one of its nodes stores their corresponding X_i-coordinate. The structure has height given by $n = 4$ levels. See **Figure 1**.

The node 0 in the first level points to a subtree that represents to all the values of extreme vertices whose first coordinate is 0. In a similar way, node 1 in the same level points to the subtree that represents to all the values of extreme vertices whose first coordinate is 1. The first of these two referred subtrees contains the vertices embedded in the first couplet perpendicular to X_1-axis, i.e. $\Phi_1^1(h)$; while the second subtree contains the vertices embedded in the second couplet perpendicular to X_1-axis, i.e., $\Phi_2^1(h)$.

At this point makes sense Trie Trees are a native way for representing *n*D-EVMs. Searching, adding, and deleting a sequence can be performed in constant time provided the length of the sequences is always constant [3]. This fact give us support for an advantage: the length of our sequences, properly our stored Extreme Vertices, is constant, that is, the number *n* of dimensions. Then, by instance, searching, adding, and deleting an Extreme Vertex is performed in constant time. Other advantages were made clear in the above paragraphs: a Trie Tree provide us an immediate access to couplets. According to the operation to perform, a copy of an extracted subtrie could be not necessary and only a pointer to the root of the subtrie would be enough. Hence, EVM-based algorithms can be implemented taking in account this tree structure. Refer to [14] for more details about *n*D-EVM implemented via Trie Trees.

Returning to the question related to temporal complexity for building the final *n*D-EVM representation in our proposed Framework, we can see now that by taking in account the Trie Tree implementation all we have to do is to process each hyperprism pr_i in the set *xf*. All vertices of pr_i are Extreme Vertices and also the vertices in the partial *n*D-EVM *F*. Therefore, each vertex *v* is searched in the Trie Trie associated to $EVM_n(F)$. If *v* is already present in $EVM_n(F)$ then it is removed from $EVM_n(F)$ because we are performing an XOR. If *v* is not present in $EVM_n(F)$ then it is added to the structure. We see the whole process reduces to checking vertices against $EVM_n(F)$ which is assumed implemented in a Trie Tree. Due to, as previously mentioned, these operations are performed in the constant time given by *n*, then the time complexity is finally dominated, once again, by the number of hyperprisms to process, that is, the size of the original hypervoxelization.

VI. THE FRAMEWORK *n*D-EVM/KOHONEN VERSUS COLOR-INTENSITY BASED SEGMENTATION: A COMPARATIVE ANALYSIS

The 1D-KSOMs we implemented use as part of their processes of training and classification the Euclidean metric over the ψ-Dimensional Space. Because each one of the

representatives of the classes in the networks are themselves vectors then we can determine the Euclidean distance between any pair of representatives. We proceed to define a False Color Map (FCM) that represents the distribution of the Weights Vectors in the subspace $[0, 1]^{\psi}$. Upper and lower bounds for Euclidean distance between any two Weights Vectors are $d_{\max} = \sqrt{\psi}$ and $d_{min} = 0$, respectively. Every Euclidean distance d between two Weights Vectors is related with a grayscale intensity through $(d/d_{\max}) \cdot 255$. Hence, if $d = 0$ then it has associated the black color while if $d = d_{max}$ then it has associated the white color.

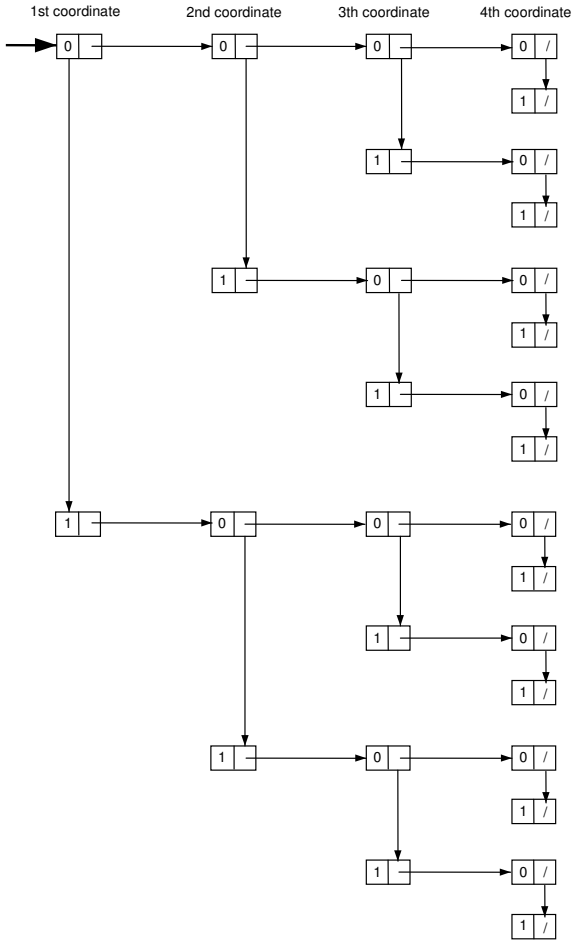


Fig. 1. The Trie Tree associated to the EVM of a 4D unit hypercube.

For purposes of comparative analysis, now we face the situation related to the fact that a segmentation exclusively based on intensities only considers the color intensity of the hypervoxel of interest and nothing more. This is the critical aspect that differentiates Color-Intensity Based Segmentation (CIBS) from our proposal because the fundament behind the use of (n-1)D masks is to take in account geometrical and topological properties such that a 1D-KSOM use them in order to achieve and to enhance the classification. On the other hand, because of simplicity in implementing CIBS the notion of a representative, a Weights Vector, for a class is absent: the hypervoxels included in a CIBS class share the same intensity. In order to also apply our proposed FCMs to classification achieved by CIBS we build a vector representative for elements in a same class. Consider a hypervoxel v with color intensity c+1, then, it is located in class κ_{c+1}. Now, we will consider the neighborhood of radius r for the given hypervoxel v respect to the original hypervoxelization. Then, we are defining a (n-1)D mask whose center is v. Given value for r, the corresponding (n-1)D mask also have $(2r + 1)^{(n-1)}$ elements. The mask is also *linearized* in such way we obtain the vector representation $\left[g_{1,i,c+1} \quad g_{2,i,c+1} \quad \cdots \quad g_{\psi,i,c+1} \right]^T$. Index i only refers to an arbitrary position assigned to the mask belonging class κ_{c+1}. One of the elements in the vector is precisely color intensity c+1 of central hypervoxel v while the rest are the color intensities of the hypervoxels inside its neighborhood. Given the above elements we build a vector representative of class κ_{c+1}, but under CIBS, as follows:

$$W_{c+1}^{CIBS} = \frac{1}{Card\left(\kappa_{c+1}\right)} \begin{bmatrix} \sum_{i=1}^{Card(\kappa_{c+1})} g_{1,i,c+1} \\ \sum_{i=1}^{Card(\kappa_{c+1})} g_{2,i,c+1} \\ \vdots \\ \sum_{i=1}^{Card(\kappa_{c+1})} g_{\psi,i,c+1} \end{bmatrix}$$

That is, given the members of class κ_{c+1}, under CIBS, its representative has as components the average color intensity for each one of the $(2r + 1)^{(n-1)}$ positions in each one of the (n-1)D masks defined for the class' members. We have now representatives in order to apply them over our FCMs.

One of the aims behind the use of 1D-KSOMs for classification of points is the proper distribution of the Weights Vectors along the subspace defined by the hypercube $[0, 1]^{\psi}$. We expect the Minimal Bounding Box for Weights Vectors (MBBWV) extends as much as possible over such hypercube. However, we are dealing, in our experiments, with $\psi = 343$ dimensions so the proposed FCMs provide us a way for analyzing, in a 2D context, how good are distributed the Weights Vectors along hypercube $[0, 1]^{\psi}$. If it is false our asseveration related to the fact geometry and topology described by a neighborhood enhances classification, in particular with our 1D-KSOMs, then intensities for FCMs associated to CIBS would tend towards whiter colors than those intensities in FCMs related to 1D-KSOMs classifications. In other words, CIBS's Weights Vectors distributions should be as good, or even better, than those Weights Vectors distribution reported by the use of our proposed 1D-KSOMs.

The **Table VII** shows FCMs for Weights Vectors distributions associated to 1D-KSOMs described in **Section IV**. In these cases the maps show distribution for 80 Weights Vectors. The Table also shows FCMs for Weights Vectors, as specified in the above paragraphs, associated to CIBS. In this situation we deal with 256 Weights Vectors because, as abovementioned, CIBS takes in account all the available intensities in the grayscale. The main diagonal for the hypercube $[0, 1]^{343}$, the one defined for the classification space in our experiments, has a length $d_{\max} = \sqrt{343} \approx 18.52$.

Let us comment in first place observations regarding dataset *VL-Sheep*. The maximum distance between any two Weights Vectors under 1D-KSOM segmentation is 13.2839 while under CIBS we have a maximum distance with value 7.9367. On the other hand, minimal distances reported for 1D-KSOM segmentation and CIBS are 1.1346 and 0.0039, respectively. These values lead us to infer CIBS' MBBWV can be easily embedded in 1D-KSOM segmentation's MBBWV. Moreover, because $d_{max} \approx 18.52$, we determine that Weights Vectors for 1D-KSOM segmentation are more

extended over hypercube $[0, 1]^{343}$ than Weights Vectors for CIBS. The average distance under 1D-KSOM segmentation is 7.7369 while it is 2.8018 under CIBS. This implies Weights Vectors for CIBS are *nested* in such way we cannot expect a clear differentiation of the regions occupied by the 256 classes. A simple inspection over the corresponding FCMs visually corroborates our claims by observing how intensities for FCM associated to 1D-KSOM tend towards whiter colors than those intensities in FCM related to CIBS.

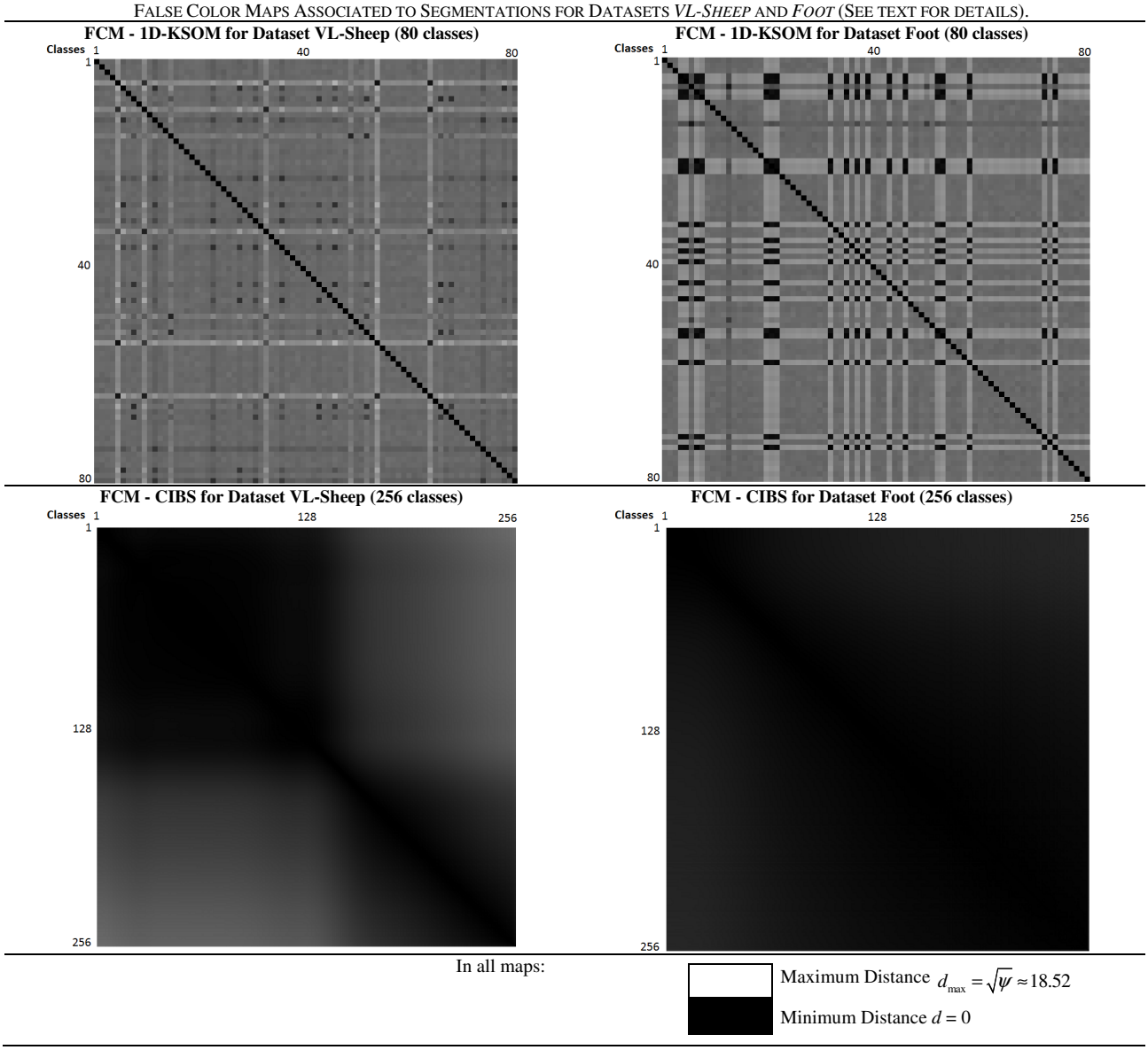
In the case related to dataset *Foot* we found maximum distances with values 11.0783 and 3.3166 for 1D-KSOM segmentation and CIBS, respectively. Minimal distance for 1D-KSOM segmentation has value 0.1139 while for CIBS is 0.0039. Finally, 1D-KSOM segmentation's average distance is 8.1409 while it is 1.0644 under CIBS. A set of similar conclusions, just as the ones in the above paragraph, can be obtained from this case leading to effectively conclude Weights Vectors for 1D-KSOM segmentation are much better distributed in $[0, 1]^{343}$ than Weights Vectors for CIBS.

Our analysis based on the use of FCMs has provided us elements for concluding, in terms of the presented experiments, the validity of our asseveration related to the fact geometry and topology described by a neighborhood enhances classification of elements in a dataset.

VII. Conclusions

In this work we have listed the elements that conform to our **Framework *n*D-EVM/Kohonen** for representing in a very concise way higher dimensional hypervoxelizations. The application of 1D-KSOMs shares us an intelligent classification of the elements in a dataset in such way it is obtained a segmentation which in turn is expressed with one additional dimension under the *n*D-EVM. This additional dimension together with the concepts accompanying the EVM provide a representation in such way elements belonging to the same class are stored in the same geometrical entity: a couplet. Because of the way the EVM represents an orthogonal polytope it is obtained an advantage respect to spatial complexity. Our pair of experiments serves as evidence of this aspect by showing ratios Number-of-Voxels/Number-of-Extreme-Vertices 6.51 and 6.81. Moreover, we have low time complexity required for training 1D-KSOMs and the generation of the final *n*D-EVM representation, because these processes are achieved by means of very efficient methods and structures, such as Trie Trees in the EVM case. Thus, the dominating time is imposed precisely by the sizes of the original hypervoxelizations.

Table VII
False Color Maps Associated to Segmentations for Datasets *VL-Sheep* and *Foot* (See text for details).



The **Framework *n*D-EVM/Kohonen** is also the object of study and analysis in [22]. In that work there are presented five study cases where voxelizations were manipulated by 1D-KSOMs with 40 output neurons and whose training sets were specified in terms of mask radius $r = 2$. In [22] we report ratios Number-of-Voxels/Number-of-Extreme-Vertices were located in the range from 5.64 to 32.43, showing once again the power of conciseness our proposal has. Moreover, in [22] there are presented some error functions whose objective is to measure the quality of the representatives obtained once the corresponding 1D-KSOMs have been trained. The results are encouraging in the sense the 1D-KSOMs' Weights Vectors are better positioned than those representatives built in terms of CIBS. For more details refer to [22].

It is possible to compute other geometrical and topological interrogations over an EVM. By this way it can be obtained more information and properties about the represented datasets. There are well specified procedures under the *n*D-EVM which allow performing Regularized Boolean Operations, Polytopes Splitting, Discrete Compactness Computation, Morphological Operations, Connected Components Labeling, Boundary Extraction, among others. In [1], [14], [17], [18], [19] & [24] there are described with enough detail algorithms based in the *n*D-EVM which are useful and efficient for performing these interrogations and/or manipulations.

Our immediate line of future research considers certain elements presented originally in [21]. The idea is to apply **Framework *n*D-EVM/Kohonen** in the representation and manipulation of Computer Tomography scans. We also aim to incorporate other metrics, besides the Euclidean Distance, for identifying the Winner Neuron in 1D-KSOMs' training in order to observe how it is impacted **Framework *n*D-EVM/Kohonen**'s conciseness and classification.

REFERENCES

[1] A. Aguilera, *Orthogonal Polyhedra: Study and Application*, PhD Thesis, Universitat Politècnica de Catalunya, 1998.

[2] M. Awad, "An Unsupervised Artificial Neural Network Method for Satellite Image Segmentation", *The International Arab Journal of Information Technology*, Vol. 7, No. 2, April 2010.

[3] F. Bodon. "Surprising results of trie-based FIM algorithms", *IEEE ICDM Workshop on Frequent Itemset Mining Implementations (FIMI'04)*, CEUR Workshop Proceedings, Vol. 90, Brighton, United Kingdom, November 2004.

[4] H.S.M. Coxeter, *Regular Polytopes*, Dover Publications, Inc., 1963.

[5] E. Davalo, P. Naïm, *Neural Networks*, Macmillan Press Ltd, 1992.

[6] Department of Radiology, University of Iowa. Web Site (February 2015): http://www.medicine.uiowa.edu/radiology/

[7] E. Friedkin, "Trie Memory", *Communications of the ACM*, Vol. 3, Number 9, pp. 490-499, 1960.

[8] J. Hilera, V. Martínez, *Redes Neuronales Artificiales* (Artificial Neural Networks), written in Spanish, Alfaomega, 2000, México.

[9] J. Jiang, P. Trundle, J. Ren, "Medical image analysis with artificial neural networks", *Computerized Medical Imaging and Graphics*, 34, pp. 617–631, 2010, Elsevier.

[10] A. Jonas, N. Kiryati, "Digital Representation Schemes for 3-D Curves", *Technical Report CC PUB* #114, The Technion - Israel Institute of Technology, Haifa, Israel, 1995.

[11] G. Klajnsek, B. Rupnik, D. Spelic, "An Improved Quadtree-based Algorithm for Lossless Compression of Volumetric Datasets", *6th WSEAS Int. Conference on Computational Intelligence, Man-Machine Systems and Cybernetics*, 1:264-270, Spain, December 2007.

[12] S. Marchand-Maillet, Y.M. Sharaiha, *Binary Digital Image Processing: A Discrete Approach*, Academic Press, 1999.

[13] R. Pérez Aguila, P. Gómez-Gil, A. Aguilera, "Non-Supervised Classification of 2D Color Images Using Kohonen Networks and a Novel Metric", *Lecture Notes in Computer Science*, Vol. 3773, pp. 271-284. Springer-Verlag Berlin Heidelberg, 2005.

[14] R. Pérez-Aguila, *Orthogonal Polytopes: Study and Application*, PhD Thesis, Universidad de las Américas - Puebla (UDLAP), 2006. http://catarina.udlap.mx/u_dl_a/tales/documentos/dsc/perez_a_r/

[15] R. Pérez-Aguila, "Modeling and Manipulating 3D Datasets through the Extreme Vertices Model in the n-Dimensional Space (nD-EVM)", *Research in Computer Science, Special Issue: Industrial Informatics*, México, 31:15-24, 2007.

[16] R. Pérez-Aguila, "Brain Tissue Characterization Via Non-Supervised One-Dimensional Kohonen Networks", *Proc. of the XIX International Conference on Electronics, Communications and Computers CONIELECOMP 2009*, pp. 197-201. IEEE Computer Society. February 26-28, 2009. Cholula, Puebla, México.

[17] R. Pérez-Aguila, "Computing the Discrete Compactness of Orthogonal Pseudo-Polytopes via Their nD-EVM Representation", *Mathematical Problems in Engineering*, vol. 2010, Article ID 598910, 28 pages, 2010. doi:10.1155/2010/598910

[18] R. Pérez-Aguila, "Towards a New Approach for Modeling Volume Datasets Based on Orthogonal Polytopes in Four-Dimensional Color Space", *Engineering Letters*, Vol. 18, Issue 4, pp. 326-340, ISSN: 1816-0948 (online), 1816-093X (print), IAENG, November 2010.

[19] R. Pérez-Aguila, "Efficient Boundary Extraction from Orthogonal Pseudo-Polytopes: An Approach Based on the nD-EVM", *Journal of Applied Mathematics*, vol. 2011, Article ID 937263, 29 pages, 2011, doi: 10.1155/2011/937263. ISSN: 1110-757X, e-ISSN: 1687-0042.

[20] R. Pérez-Aguila, *Una Introducción al Cómputo Neuronal Artificial* (An Introduction to Artificial Neural Computing), written in Spanish, El Cid Editor, Argentina, First Edition, September 2012, ISBN (Print): 978-1-4135-2424-6, ISBN (Digital): 978-1-4135-2434-5.

[21] R. Pérez-Aguila, "Enhancing Brain Tissue Segmentation and Image Classification via 1D Kohonen Networks and Discrete Compactness: An Experimental Study", *Engineering Letters*, Vol. 21, Issue 4, pp. 171-180, ISSN: 1816-0948 (online), 1816-093X (print), IAENG, Nov. 2013.

[22] R. Pérez-Aguila, R. Ruiz-Rodríguez, "Concise and Accessible Representations for Multidimensional Datasets: Introducing a Framework Based on the nD-EVM and Kohonen Networks", *Applied Computational Intelligence and Soft Computing*, Vol. 2015, Hindawi Publishing Corporation, 2015. Web Site: http://www.hindawi.com/journals/acisc/2015/676780

[23] H. Ritter, T. Martinetz, K. Schulten, *Neural Computation and Self-Organizing Maps, An introduction*, Addison-Wesley, 1992.

[24] J. Rodríguez, D. Ayala, "Erosion and Dilation on 2D and 3D Digital Images: A new size-independent approach", *Vision Modeling and Visualization 2001*, Germany, 1:143–150, 2001.

[25] S. Roettger, The Volume Library. Web Site (last visit in September, 2014): http://www9.informatik.uni-erlangen.de/External/vollib/

[26] R. Ruiz-Rodríguez, *Implementación del EVM (Extreme Vertices Model) en Java*, M.Sc Thesis, UDLAP, 2002. http://catarina.udlap.mx/u_dl_a/tales/documentos/msp/ruiz_r_r/

[27] R. Ruiz Rodríguez, "A 3D Editor for Orthogonal Polyhedra Based on the Extreme Vertices Model", *Décimo Congreso Internacional de Investigación en Ciencias Computacionales CIICC 03*, Oaxtepec, Morelos, México, Octubre 2003. ISBN: 968-5823-02-2.

[28] R. Ruiz Rodríguez, "Using the Ordered Union of Disjoint Boxes Model for the Visualization as Solid Objects of Orthogonal Pseudo Polyhedra Defined in the Extreme Vertices Model", *II Congreso Internacional de Informática y Computación de la ANIEI*, ANIEI 2003, Zacatecas, Zacatecas, México, Octubre 2003.

[29] D.M.Y. Sommerville, *An Introduction to the Geometry of N Dimensions,* Dover Publications Inc., 1958.

[30] W. Song, S. Hua, Z. ou, H. An, K. Song, "Octree Based Representation and Volume Rendering of Three-Dimensional Medical Data Sets", *International Conference on BioMedical Engineering and Informatics 2008*, 1:316-320, 2008.

[31] M. Spivak, *Calculus on Manifolds: A Modern Approach to Classical Theorems of Advanced Calculus*, HarperCollins Publishers, 1965.

[32] N. Torbati, A. Ayatollahi, A. Kermani, "An efficient neural network based method for medical image segmentation", *Computers in biology and medicine*, 44:76-87, 2014.

[33] University of Tübingen. The Official Volren and Volvis Homepage. Web Site (last visit in September, 2014): http://www.gris.uni-tuebingen.de/edu/areas/scivis/volren/software/software.html

[34] K. Yuan, F. Peng, S. Feng, W. Chen, "Pre-Processing of CT Brain Images for Content-Based Image Retrieval", *Proc. International Conference on BioMedical Engineering and Informatics 2008*, Vol. 2, pp. 208-212.

[35] J. Zerubia, S. Yu, Z. Kato, M. Berthod, "Bayesian Image Classification Using Markov Random Fields", *Image and Vision Computing*, 14:285-295, 1996.