Z-images

Vincent Vigneron, Tahir Qasim Syed, Leonardo Tomazeli Duarte, Elmar Lang, Sadaf Iqbal Behlim, Ana Maria Tome

Z-Images

Vincent Vigneron[1(✉)], Tahir Q. Syed[3], Leonardo T. Duarte[2], Elmar Lang[5], Sadaf I. Behlim[3], and Ana-Maria Tomé[4]

[1] IBISC-lab EA-4526, Université d'Evry val d'Essonne, Évry, France
vincent.vigneron@ibisc.univ-evry.fr
[2] Laboratory of Signal Processing for Communications FCA/UNICAMP, Limeira, Brazil
leonardo.duarte@fca.unicamp.br
[3] Department of Computer Science, National University of Computer and Emerging Sciences, Karachi 75030, Pakistan
{tahir.syed,sadaf.iqbal}@nu.edu.pk
[4] Departamento de Electronica, Telecomunicacoes e Informática, Universidade de Aveiro, Aveiro, Portugal
ana@ua.pt
[5] Computational Intelligence and Machine Learning Group, Universität Regensburg, Regensburg, Germany
Elmar.Lang@biologie.uni-regensburg.de

Abstract. Local patterns and other patch based features have been an integral part of various computer vision applications as they encode local structural and statistical information. In this paper, we propose an image coding technique that utilizes Zeckendorf representation of pixel intensities and basic mathematical operators such as intersection, set difference, maximum, summation etc. for summarization of image regions. The algorithm produces a *Z-coded* image that tells about the homogeneity or the contrast in image regions with all codes in a range of 0 to 255.

Keywords: Image descriptor · Generative model · Zeckendorf theorem · LBP

1 Introduction

Summarizing local image content is considered a fundamental step in all classical computer vision tasks such as object detection and recognition [3,8], texture analysis [11], motion analysis [2,11], image restoration and reconstruction [6] etc. This step usually involves extraction of low level features such as finding

V. Vigneron—This research was supported by the program *Cátedras Franco-Brasileiras no Estado de São Paulo*, an initiative of the French consulate and the state of São Paulo (Brazil). We thank our colleagues Prof. João M.T. Romano, Dr. Kenji Nose and Dr. Michele Costa, who provided insights that greatly assisted this work.

out where the edges are, detecting interest points and their region of interests by applying local neighborhood operations. Since imaging data are noisy, locally-correlated and usually with too many data points per 'unit' of useful information, these *intermediate representations* lead to an understanding of the scene in an image without the noisy and meaningless influence of very many pixels that carry little inferable information. These representations usually encode either structural information that is extraction of texture as a set of repeated primitive textons or statistical information that how different pixel intensities are distributed in a local neighborhood.

The amount of information extracted from different regions of an image usually depends on the size of the neighborhood, the reading order of the neighbors and the mathematical function that is used to extract the relationship between two neighboring pixels. Most of the descriptors that encode local structures i.e. Local Binary Patterns (LBP) [4] and its variants [5] such as Census Transform (CT) [9] etc. depend on the *reading order* as they compute the feature value as the weighted sum of neighboring pixels $\{g_p | p = 0, \ldots, P-1\}$ w.r.t. their order in the neighborhood.

There exists many variants of LBP (see [5, Chap. 2, p. 26] for a summary of the variants) because basic LBP has many problems that need to be addressed. For instance, LBP and CT both generates $8-bit$ string for a 3×3 neighborhood by computing the Heaviside function $t(\cdot)$ of the difference between neighboring pixels and the central pixel *i.e.* $(g_i - g_c)$ which is shown in Fig. 1. The only difference between these two descriptors is the reading order of neighboring pixels and the sign of the difference which results in 2 different bit patterns. Given the $8-bit$ string, the LBP and CT code is calculated as:

$$L_{P,R}(\mathbf{r}_c) = \sum_{p=0}^{P-1} 2^p \cdot t(g_p - g_c), \text{ with } t(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise.} \end{cases}, \tag{1}$$

where P is the number of pixels in the neighborhood considering the distance R between central pixel and its neighbors.

Local Binary Pattern (LBP) is considered a computationally efficient structural descriptor and its applications have evolved into almost all fields of computer vision, because of its robustness to monotonic gray-scale changes, illumination invariance and its computational simplicity. Invariance w.r.t. any monotonic

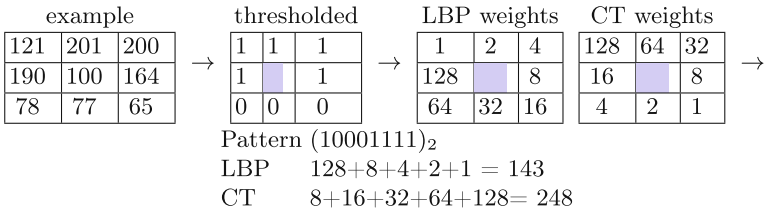

Fig. 1. Neighbors of center pixel g_c participating in LBP or CT code generation in the case of a 3×3 neighborhood ($P = 8$ and $R = 1$).

transformation of the gray scale is achieved by considering in Eq. (1) the signs of the differences $t(g_i - g_c), i = 0, \ldots, P-1$. But the *independence* of g_c and $\{|g_0 - g_c|, \ldots, |g_{P-1} - g_c|\}$ is not warranted in practice. Moreover, under certain circumstances the LBP misses the local structure as it does not consider the central pixel. The binary data produced by them are sensitive to noise mostly in uniform regions of an image.

To reduce this noise sensitivity, a 3-level operator has been proposed by Tan and Trigg [7] which describes a pixel relationship with its neighbor by a ternary code *i.e.* $\{-1, 0, 1\}$ rather than a binary code. The size of this code is reduced by splitting it into two LBP codes (Positive and Negative) which results into two $8 - bit$ strings thus needing 16 bit space for representation.

This paper discusses an encoding scheme that is *independent* of the size of neighborhood and the reading order of neighboring pixels in Sect. 2. An algorithm is also proposed for generating Z-codes of an image in Sect. 3, which could be utilized in contour detection, image segmentation and adaptive image quantization. In Sect. 4 we discuss the results and compare its characteristics with the classic LBP.

2 Zeckendorf Encoding

The LBP operator generates an integer value in the range of 0 to 255. We propose to represent each pixel intensity N as an ordered collection of positive integers whose sum is N.

2.1 Zeckendorf's Theorem

The Zeckendorf's theorem [10] states that every positive integer N can be represented uniquely as the sum of distinct Fibonacci numbers such that the sum *does not include any two non-consecutive Fibonacci numbers.*

Theorem 1 (Zeckendorf's theorem). *Every positive integer can be expressed as a sum of distinct Fibonacci numbers. If N is any positive integer, there exist positive integers $n_i \geq 2$, with $n_{i+1} \geq n_i + 1$, such that $N = \sum_{i=0}^{k} F_{n_i}$, where F_j is the $j-th$ Fibonacci number.*

The famous Fibonacci sequence $< 1, 1, 2, 3, 5, 8, \ldots >$ is a sequence of numbers, $x(n)$, that satisfies the difference equation

$$x(n) = x(n-1) + x(n-2) \quad \text{for} \quad n \geq 0 \tag{2}$$

that is $x(n)$ is the sum of the 2 previous values with initial conditions $x(0) = x(1) = 1$.

Proof. For any positive integer n, there is always a positive integer m such that $x(m) \leq n \leq x(m+1)$. If $n \neq x(m)$,

$$0 < n - x(m) < x(m+1) - x(m) = x(m-1). \tag{3}$$

Since $n - x(m)$ is positive, there exists a positive integer p such that

$$x(p) \leq n - x(m) < x(p+1). \tag{4}$$

Now $x(p) \leq n - x(m) < x(m-1)$ implies $p \leq m-2$, *i.e.* $x(p)$ and $x(m)$ are not consecutive Fibonacci numbers. If $n - x(m) \neq x(p)$, there exists a positive integer $q \leq p-2$ such that

$$x(q) \leq n - x(m) - x(p) < x(q+1) \tag{5}$$

and the process continues. Ultimately, we must reach the point where the partial sum equals a Fibonacci number – say $x(t)$ – and thereby obtain the desired representation

$$n = x(m) + x(p) + x(q) + \ldots + x(t). \tag{6}$$

2.2 Z-Representation

An 8-bit gray scale image has the intensity values in the range of [0,255]. The distinct Fibonacci numbers below 255 are $\{1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233\}$. Each pixel intensity of an image can be represented as a sum of distinct non-consecutive Fibonacci numbers. For instance the only Zeckendorf representation of pixel value 255 is $(233, 21, 1)_{\text{Zck}}$. Since there are 12 different possibilities to represent any 8-bit intensity value, therefore each number is represented using 12 bits no consecutive bits are ON i.e. 1 because of non-consecutive Fibonacci numbers constraint. Table 1 shows some Zeckendorf representations and their bit patterns.

Table 1. Zeckendorf's representation of some numbers.

N	Partition	Bit pattern	N	Partition	Bit pattern
1	1	1 0 0 0 0 0 0 0 0 0 0 0	50	34 + 13 + 3	0 0 1 0 0 1 0 1 0 0 0 0
2	2	0 1 0 0 0 0 0 0 0 0 0 0	51	34 + 13 + 3 + 1	1 0 1 0 0 1 0 1 0 0 0 0
3	3	0 0 1 0 0 0 0 0 0 0 0 0	131	89 + 34 + 8	0 0 0 0 1 0 0 1 0 1 0 0
4	3 + 1	1 0 1 0 0 0 0 0 0 0 0 0	132	89 + 34 + 8 + 1	1 0 0 0 1 0 0 1 0 1 0 0
5	5	0 0 0 1 0 0 0 0 0 0 0 0	154	144 + 8 + 2	0 1 0 0 1 0 0 0 0 0 1 0
6	5 + 1	1 0 0 1 0 0 0 0 0 0 0 0	159	144 + 13 + 2	0 1 0 0 0 1 0 0 0 0 1 0
7	5 + 2	0 1 0 1 0 0 0 0 0 0 0 0	174	144 + 21 + 8 + 1	1 0 0 0 1 0 1 0 0 0 1 0
8	8	0 0 0 0 1 0 0 0 0 0 0 0	190	143 + 34 + 8 + 3	0 0 1 0 1 0 0 1 0 0 1 0
9	8 + 1	1 0 0 0 1 0 0 0 0 0 0 0	222	144 + 55 + 21 + 2	0 1 0 0 0 0 1 0 1 0 1 0
10	8 + 2	0 1 0 0 1 0 0 0 0 0 0 0	254	233 + 21	0 0 0 0 0 0 1 0 0 0 0 1

3 Proposed Algorithm for Z-Coding

Based on the proposed partition property of integers, we design an algorithm that encodes pixel relationship with its local neighborhood of dimension $m \times n$ by an integer value ranging from 0 to 255 we named $Z - code$. We apply different basic mathematical functions and operators *i.e.* minimum, maximum, summation and some set operators i.e. intersection, set difference etc. The sequence in which operations and functions are applied results in images that could be directly used in computer vision pipeline for object *detection* and *recognition*.

Example 1 (Z-coding). Consider a pixel, of intensity 183, surrounded by the following gray-levels $\{210, 106, 231, 233, 79, 142, 209, 188\}$.

The Zeckendorf decomposition for the neighboring pixel is respectively $\{144 + 55 + 8 + 3, 89 + 13 + 3 + 1, 144 + 55 + 21 + 8 + 3, 233, 55 + 21 + 3, 89 + 34 + 13 + 5 + 1, 144 + 55 + 8 + 2, 144 + 34 + 8 + 2\}$ and for the central pixel is $\{144, 34, 5\}$. Applying the Algorithm 1, we find when considering the first neighboring pixel $\{144, 34, 5\} \cap \{144, 55, 8, 3\} = \{144\}$ Since dummy $\neq \emptyset$ therefore $s(1)$ will be equal to 144. Similarly for the second pixel $\{144, 34, 5\} \cap \{89, 13, 3, 1\} = \emptyset$. Since dummy $= \emptyset$ therefore the $s(2)$ will be equal to J_0 which is 183. After the vector s is populated with the values

Algorithm 1. Algorithmic principle behind the image Z-coding

Require: Image I of size $J \times K$: texel of size O;N is number of pixels arround center pixel J_0
Ensure: Z-coded image Z of size $J \times K$ of input image I

Initialization $Z \leftarrow \emptyset$; $j = 2, k = 2$
for $j = 2$ to $J - 1$ **do**
  for $k = 2$ to $K - 1$ **do**
    $J_0 = I(j, k)$; ▷ central pixel of the texel
    $texel \leftarrow$ Intensity Values of N neighbouring pixels arround J_0
    $\mathbf{s} \leftarrow \mathbf{0}$
    $\mathcal{S}^0 \leftarrow$ **zeckendorf** (J_0)
    for $i = 1$ to N **do**
      $\mathcal{S} \leftarrow$ **zeckendorf** (texel(i));
      dummy $\leftarrow \mathcal{S}^0$ *op* $\mathcal{S}$ ▷ *op* is intersection or set difference operator
      if (dummy $= \emptyset$)) **then**
        $s(i) \leftarrow J_0$ ▷ J_0 for quantization and 0 for contours
      else
        $s(i) \leftarrow \max(\text{dummy})$ ▷ *max* for quantization and *sum* for contours
      end if
    end for
    $Z(j, k) \leftarrow \max(\mathbf{s})$
  end for
end forreturn Z
Function zeckendorf(x)
Decomposes an integer x as a sequence of Fibonacci numbers
EndFunction

s = 144, 183, 144, 183, 183, 34, 144, 144, the maximum value is computed of this set which is treated as the Z-code for the given pixel i.e. 183 in this example.

4 Results and Discussion

The algorithm proposed in Sect. 2.1 results in two different kinds of images based on the initial operator which is applied i.e. either set difference or intersection. The *intersection* operator find the similarity among the pixel and its neighborhood Zeckendorf representation and place a value which is common among them thus results in an image that is quantized in terms of their representation as shown in Table 2 2nd row. The set *difference operator* extracts *ultrametric* contours of an image resulting in image segmentation [1] shown in 3rd and 4rth row of Table 2.

Compared to other local descriptors (see Sect. 1), the Z-coding

Table 2. Z-coded images using Zeckendorff representation – 1st row shows original images, 2nd row shows quantized images obtained by applying intersection operator, 3rd row contains ultrametric contours obtained by applying set-difference operator and last row shows complemented results of 3rd row

– can be extended to any neighborhood size or geometry – this is because in Table 1, the Z-value is computed from a set of differences of stack values (set difference operation). This stack contains the grayscale values of the pixels. It is arbitrarily extendable. In addition, 2^p weighting or any binary coding is disregarded,
– is *invariant* to any shift in grayscale as there is no ranking in the set of differences of grayscale values nor in the weights,
– is *order-invariant* (same argument as above)
– does not consider the central pixel effect as the central pixel value does not enter in the set difference operation but since the pick values in a finite and bounded set of integers, the algorithm guarantee that the z-value $z_i \in \{0, \ldots, 255\}$,
– is *nonlinear* because the set difference operation is nonlinear,
– follows an integer generating scheme as the stacked values are provided by the recurrence Eq. (2),
– is less sensitive to any noise influence as the Z-coding acts as a *quantizer*. The quantization error is considered an additive noise source, and it is assumed to be uniformly distributed over the range of values of the quantization error Δ,[1]
– comes with a *performance criterion*. The criterion to maximize is, for each pixel (i, j) of the image, the sum $S_{ij} = \sum_p (\frac{g_{\sigma(p)}}{g_{\sigma(p+1)}} - \frac{2}{1+\sqrt{5}})^2$ over the neighboring pixels ranked in increasing order.

5 Conclusion

In this paper we describe a new image coding algorithm *Z-coding* based on integer generating function and Zeckendorf theorem of integer decomposition. This image coding method is invariant to shift and rotation, could be extended to any neighborhood size and is independent of the reading order of neighboring pixels. It summarizes an image either by identifying the homogeneity in various irregular regions of image or by extracting soft contours using local contrast quantification which could be utilized in computer vision pipeline.

References

1. Arbelaez, P.: Boundary extraction in natural images using ultrametric contour maps. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR Workshops 2006, New York, NY, USA, 17–22 June 2006, p. 182 (2006)
2. Delicato, L.S., Serrano-Pedraza, I., Suero, M., Derrington, A.M.: Two-dimensional pattern motion analysis uses local features. Vis. Res. **62**, 84–92 (2012)
3. Heikkilä, M., Pietikäinen, M.: A texture-based method for modeling the background and detecting moving objects. Pattern Anal. Mach. Intell. **28**(4), 657–662 (2006)

[1] This assumption is generally valid for a small number of values and the noise power (variance) is $\frac{\Delta^2}{P-1}$, where $P-1$ is the number of surrounding pixels.

4. Ojala, T., Pietikäinen, M., Harwood, D.: A comparative study of texture measures with classification based on feature distributions. Pattern Recogn. **29**, 51–59 (1996)
5. Pietikinen, M., Hadid, A., Zhao, G., Ahonen, T.: Computer Vision Using Local Binary Patterns. Computer Imaging and Vision, vol. 40. Springer, London (2011)
6. Syed, T.Q., Behlim, S.I., Merchant, A.K., Thomas, A., Khan, F.M.: Leveraging mutual information in local descriptions: from local binary patterns to the image. In: Murino, V., Puppo, E. (eds.) ICIAP 2015. LNCS, vol. 9280, pp. 239–251. Springer, Cham (2015). doi:10.1007/978-3-319-23234-8˙23
7. Tan, X., Triggs, B.: Enhanced local texture feature sets for face recognition under difficult lighting conditions. In: Zhou, S.K., Zhao, W., Tang, X., Gong, S. (eds.) AMFG 2007. LNCS, vol. 4778, pp. 168–182. Springer, Heidelberg (2007). doi:10.1007/978-3-540-75690-3˙13
8. Uijlings, J.R.R., van de Sande, K.E.A., Gevers, T., Smeulders, A.W.M.: Selective search for object recognition. Int. J. Comput. Vis. **104**(2), 154–171 (2013)
9. Zabih, R., Woodfill, J.: Non-parametric local transforms for computing visual correspondence. In: Eklundh, J.-O. (ed.) ECCV 1994. LNCS, vol. 801, pp. 151–158. Springer, Heidelberg (1994). doi:10.1007/BFb0028345
10. Zeckendorf, E.: Représentation des nombres naturels par une somme de nombres de fibonacci ou de nombres de lucas. Bull. Soc. Roy. Sci. Liege **41**, 179–182 (1972)
11. Zhao, G., Pietikainen, M.: Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE Trans. Pattern Anal. Mach. Intell. **29**(6), 915–928 (2007)