Texture Analysis using Informed Search in Graphs

Romulo L. Frutuoso, João Paulo P. Gomes, Emanuele M. dos Santos,
Joaquim B. Cavalcante Neto and Creto A. Vidal
Universidade Federal do Ceará
Departamento de Computação, Fortaleza, Ceará 60455–760
Email: frutuoso.romulo@gmail.com, jpaulo@lia.ufc.br, emanuele@lia.ufc.br,
joaquimb@lia.ufc.br, cvidal@lia.ufc.br

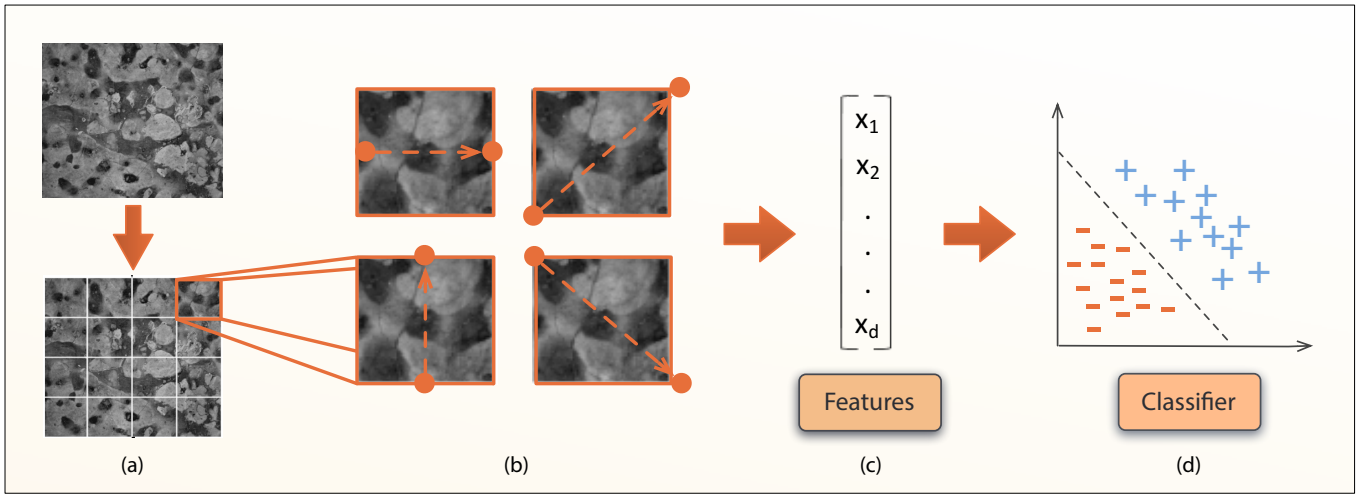

Fig. 1. A schematic overview of the TAISG (Texture Analysis using Informed Search in Graphs) method. For each texture image of a training set the following steps are performed: (a) Each original image is divided into sub-images of size $l_i \times l_i$. (b) For each sub-image, compute feature vectors that will be based on the paths determined by an informed search method in four directions. (c) All the feature vectors for all sub-images of different sizes are concatenated into a single feature vector. (d) The resulting feature comprise a training set that is used to train a LDA classifier. . The full procedure is repeated adjusting the control parameter in the informed search method until it achieves the best configuration for the image set.

***Abstract*—In this paper we propose a variant oisf the TASPG algorithm for texture recognition. TASPG (Texture Analysis based on Shortest Paths in Graphs) is a recently proposed texture recognition method that extracts features from paths along texture images. Although TASPG achieved promising results, its application may be limited by its high computational cost which stems from the extensive use of Dijkstra's algorithm. In this work, we propose a variant of TASPG, called TAISG, that uses an informed search algorithm to reduce the number of visited nodes in the search procedure. The proposed method was compared with TASPG and other texture classification methods and showed good results, both in recognition rate and in computational cost.**

***Keywords*-pattern recognition; texture analysis; search methods; informed search**

I. Introduction

Although humans can recognize different textures in a straightforward way, texture classification is one of the most challenging problems in automatic pattern recognition [1]. That difficulty stems from the lack, in the literature, of a formal definition capable of describing the concept of texture [2].

As a consequence, many strategies have been proposed for texture recognition. Classical methods are mostly based on second order statistics ([3], [4]) and spectral analysis ([5], [6]). More recently, approaches based on fractal analysis [7], graph theory [1], complex network theory [8] and gravitational models [9] have achieved promising results.

Texture analysis methods can be seen as feature extraction strategies, i.e., as techniques that extract meaningful information from texture images, resulting in a vector of features. After that, a classifier is used to assess its resulting recognition rate.

Among those recently proposed methods, the work of Mesquita Sa Junior *et al.* [1] showed superior results when compared with both classical and novel methods on public texture data sets. The method, referred to as TASPG (Texture Analysis based on Shortest Paths in Graphs), models each image with a graph and searches for the shortest paths within specific orientations. Feature vectors describing each image are obtained by calculating statistics over those paths. Shortest paths are found by executing the classical Dijkstra's algorithm[10].

Despite its remarkable performance, using TASPG may be impractical in applications with limited memory and pro-

cessing capability. It is known that Dijkstra's algorithm has exponential time and space complexity [11], which might lead to problems in large images.

In this paper, we propose a variant of TASPG that uses an informed search algorithm. The informed search method uses a heuristic term to guide the path search in order to reduce the number of computations. The proposed method is compared with the standard TASPG, and achieves similar results with a significant reduction in time and space complexity.

The remaining of this paper is organized as follows. in Section II, we review the related work. In Section III, the formulation for TASPG is showed. In Section IV, we present the proposed variant of TASPG. In Section V we show the experiments and the results with public texture datasets. Finally, in Section VI, the conclusions and directions for future work are presented.

II. Related Work

Many different methods for extracting image texture information have been proposed over the years. Classical approaches include methods based on second-order statistics such as Co-occurrence matrices [3] and spectral analysis such as Fourier descriptors [5], Gabor filters [6] and wavelets [12].

More recently, some authors proposed alternative ways of exploring texture images in order to fill the gaps left by other methods [13]. Some remarkable performances were obtained by approaches based on trajectories produced by deterministic walkers [14], gravitational models [9], [15] and graph theory [1].

In [9], the authors used concepts of fractal geometry to extract texture features. The main idea is to extract features from the estimation of fractal dimension under different scales. The proposed method extracts entropy features from the deviation of the Bouligand-Minkowski [16] descriptors computed over several sub images.

Gravitational models are also used in Mesquita Sa Junior *et al.* [15]. In that paper, the authors convert the image into a simplified dynamical system in gravitational collapse process whose collapsing states are described by using the lacunarity method. The lacunarity method consists in gliding a box of a specified size over the texture pattern and count the number of gaps that exist in the binary pattern.

The method proposed in [14] models the image as a surface and uses a deterministic tourist walk algorithm to identify attractor zones. Features are extracted from the path that each tourist walks before reaching an attractor zone.

The idea of extracting features from paths in images is also explored in Mesquita Sa Junior *et al.* [1]. The proposed method, referred to as TASPG (Texture Analysis based on Shortest Paths in Graphs), models each image as a graph and finds the shortest paths between several start and final pixels using Dijkstra's algorithm. Statistical descriptors are extracted from each shortest path.

Considering all the presented methods, we found that some of the most promising results were obtained by Mesquita Sa Junior *et al.* [1]. Despite its remarkable recognition rate, the use of Dijkstra's algorithm results in a computationally expensive method (exponential in time and space). In order to address this problem, we propose a variant of TASPG that reduces its computational cost significantly with a similar recognition rate.

III. Texture Analysis based on Shortest Paths in Graphs

The TASPG method aims at characterizing a texture by extracting features from the shortest paths between selected points of an image. Initially, an undirected graph is built such that each pixel is considered a vertex of the graph. An undirected edge connects two vertices only if the Chebyshev distance between them is less than 1. A weight w is associated with an edge $e = (v, v')$ according to Eq. 1:

$$w_e = |I(x, y) - I(x', y')| + \frac{I(x, y) - I(x', y')}{2} \quad (1)$$

where x and y are coordinates of a pixel and $I(x, y)$ and $I(x', y')$ are the gray-level intensities of the neighboring pixels associated with vertices v and v', respectively.

After obtaining the graph, Dijkstra's algorithm is used to find the shortest paths in four directions as shown in Fig. 2.

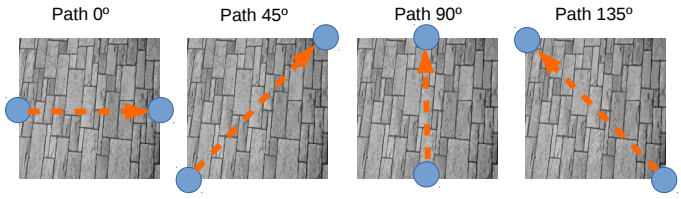

Fig. 2. The four different directions considered in the computation of the shortest paths by the TASPG method.

The same procedure is repeated for sub-images of the original image as a way to obtain information regarding local textures. Each sub-image is generated by dividing the original image into boxes of size $l \times l$, where l is a divisor of the original image size.

For each sub-image eight features are obtained from the shortest paths calculated for each of the four directions (0^o, 45^o, 90^o and 135^o). Four features comprise the mean values of the shortest paths in all directions and the other four are the standard deviations for the same paths. The 8-dimensional feature vectors for all sub-images generated for a given l are concatenated, resulting in the feature vector ψ_l.

Several values of l are chosen so that information about textures in different scales are obtained. The concatenation of the feature vectors related to the n values of l results in the final feature vector φ. The values of l are hyper-parameters and can be chosen by cross validation. The TASPG method is summarized in Fig. 3.

In a nutshell, the intuition behind TASPG is that similar textures could result in graphs that have similar shortest paths. The directions are included to provide rotation invariance and the sub-images are used for scale invariance. The behaviors

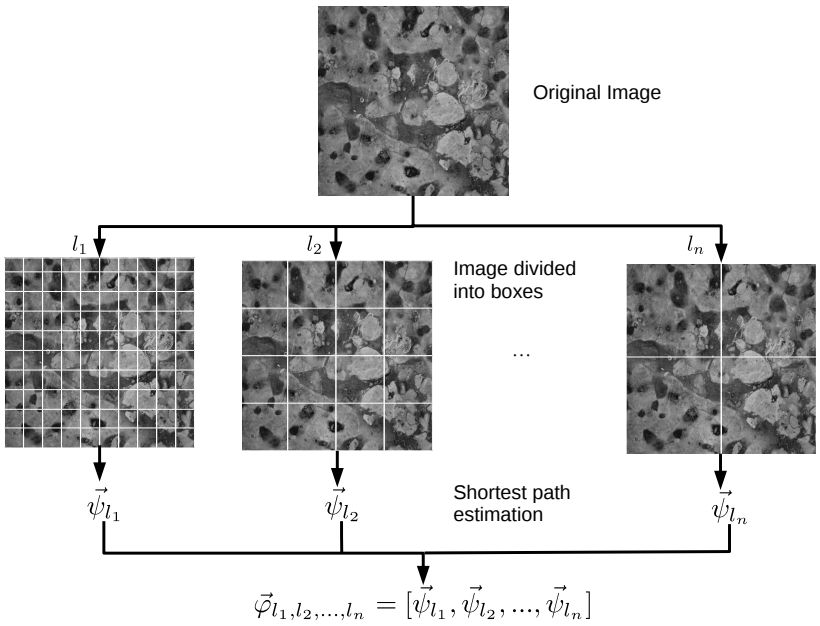

Fig. 3. Depiction of how the feature vector for a given image is generated using the TASPG method. Each original image is divided into sub-images of size $l_i \times l_i$, where l_i is a divisor of the original image size. For each sub-image, an 8-dimensional feature vector is computed using the mean values and the standard deviations of the shortest paths in all four directions. For all sub-images generated for a given l_i, these vectors are concatenated resulting in the feature vector $\vec{\psi_{l_i}}$. As n different sizes of l_i are chosen, finally, all the n feature vectors $\vec{\psi_{l_i}}$ are concatenated into the final feature vector $\vec{\varphi_{l_1,l_2,...,l_n}}$.

of the shortest path are encoded in the mean and standard deviation calculations.

IV. Informed Search for Texture Analysis

As presented in the last section, TASPG is based on the application of Dijkstra's algorithm in four directions on each image. Dijkstra's algorithm finds a path between two nodes in a weighted graph and it is known to be complete (i.e., it finds a solution if it exists) and optimal (i.e., finds the shortest path). The algorithm is part of a broad class of search algorithms, known as uninformed (blind) search methods that do not use any additional information to guide their search procedure [11].

It is known that Dijkstra's algorithm has exponential time and space complexity [11]. In general terms, Dijkstra's algorithm will visit an exponential number of nodes (vertices) in the search for the shortest path, and will store these visited nodes in memory. As a consequence, its application is unfeasible in many real problems.

A possible approach to reduce time and space complexity relies on the use of some available information that may indicate states that are most probably part of the minimum path. Methods that use such approach are classified as informed search methods.

In Dijkstra's algorithm, this feature may be included by adding an extra (heuristic) term on the cost function. The well known A* algorithm uses such strategy, designing its cost function as:

$$J(v) = g(v) + h(v) \tag{2}$$

where $J(v)$ is the cost to visit node v, $g(v)$ is the path cost from the start to the node v and $h(v)$ is an heuristic that estimates the cost from v to the goal. Choosing an admissible heuristic (i.e., the heuristic does not overestimate the true cost to the goal) leads to a complete and optimal algorithm. Notice that the design of good heuristics depends on the search problem.

Considering the problem of texture analysis, an heuristic function can be designed since, when a search is performed, the direction of the goal node is known. One possible option is to use the Euclidean distance in the space of coordinates as a measure of how many nodes should be visited from v to the goal node. In addition, we used a multiplicative term C that controls how directed the search shall be. The resulting heuristic term is given by:

$$h(v) = C \left[(x - x^*)^2 + (y - y^*)^2\right]^{\frac{1}{2}} \tag{3}$$

where x and y are the coordinates of v and x^* and y^* are the coordinates of the goal node.

Using high values of C leads to directed searches that visit only a small number of states, thus reducing time and space complexity. Fig. 4 shows the effect of varying C in the search algorithm. Each row of the picture shows a different value of C while the columns show the search procedure's progress.

In Fig. 4, the red pixels indicate that a node is visited during the search procedure. Considering that, one can notice that Dijkstra's algorithm ($C = 0$) check almost all pixels of this image as it searches for the shortest path. On the other hand, using $C = 350$ results in a more directed search and

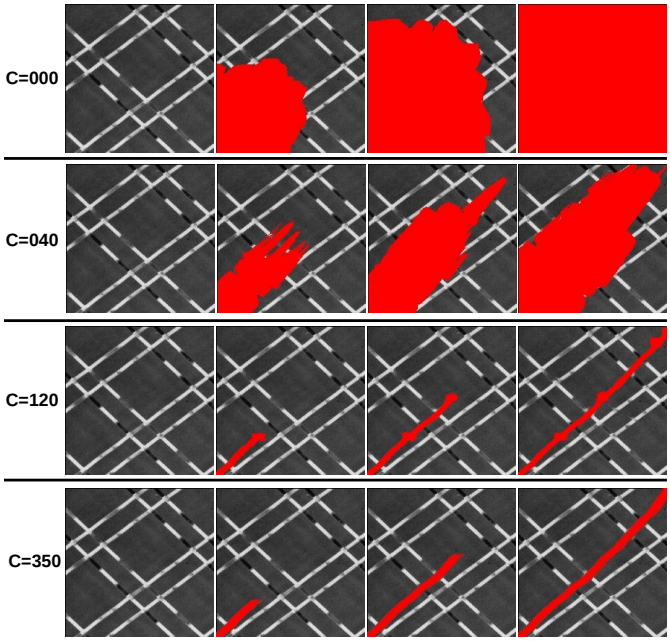

Fig. 4. The impact of the multiplicative term C in the search algorithm: larger values of C lead to more directed searches that visit smaller number of nodes (shown in red), thus reducing time and space complexity.

less visited nodes. Notice that high values of C may result in nonoptimal (do not find the shortest path) search methods since the heuristic may be inadmissible.

Even though TASPG is based on features extracted from the shortest paths, there are no theoretical guarantees that using such paths will lead to the best texture classification results. Considering that observation, it is possible that any choice of C may lead to a better recognition performance and also reduced time and space complexity.

In the proposed method, named Texture Analysis using Informed Search in Graphs (TAISG), C is chosen so that the recognition rate is not significantly affected by an increase of C. The classification task is performed using a Linear Discriminant Analysis (LDA) classifier. The significance of the difference between results of TAISG with different values of C is assessed by using the Kolmogorov-Smirnov hypothesis test (K-S test, [17]).

The proposed method is described by the following steps.

STEP 1 - Define the values $l_i, i = 1, \ldots, n$.

STEP 2 - Set $C = 0$.

STEP 3 - Divide the training set of texture images into ten subsets.

STEP 4 - For each l_i, calculate the feature vectors ψ_{l_i} and generate the final feature vector φ. The feature vector is obtained for all images that belong to nine of the ten subsets.

STEP 5 - Classify the images of the remaining subset using an LDA classifier.

STEP 6 - Repeat steps 4 and 5 changing the remaining subset and store the classification rates obtained when each of the ten subsets is used for testing.

STEP 7 - Repeat steps 2 to 6 for increasing values of C.

STEP 8 - Check whether the classification performance resulting fron each different choice of C is significantly distinct from the results obtained when $C = 0$. The statistical significance is given by the K-S test with $p - value = 0.05$. Choose the highest C that does not show a significant classification performance reduction.

V. Experiments and Results

We validate our technique through a series of experiments on two texture recognition public data sets. The first data set consists of a set of textures extracted from Brodatz's book [18]. That database is a well-established benchmark for research on texture analysis. For this work, we use 40 classes as in Mesquita Sa Junior *et al.* [1]. Each image is 200×200 pixels with 256 gray levels. Several examples of Brodatz's texture images are shown in Fig. 5.

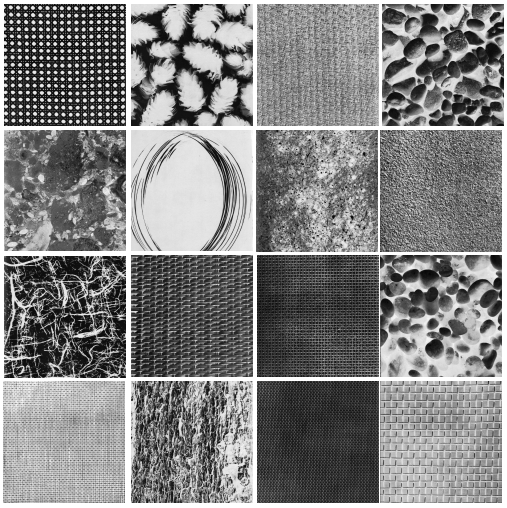

Fig. 5. A sample from Brodatz's texture image data set used in the experiments.

The second data set is a database of images obtained from different viewpoints, with perspective distortions and non-rigid transformations [19]. The UIUC database is a collection of 1000 grayscale texture images (25 classes with 40 samples each) of 640×480 pixels with 256 gray levels. Sample images from the UIUC data set are shown in Fig. 6.

An initial analysis is performed to verify the impact of C on the recognition rate of TAISG. We define the recognition rate as the percentage of correctly classified texture samples. For this experiment we set $l = 4, 5, 8, 10, 20, 25, 40, 50$, i.e. features were obtained from sub-images of sizes 4×4, 5×5, 8×8, 10×10, 20×20, 25×25, 40×40 and 50×50. The resulting feature vectors are generated by concatenating features for all values from l. The values in l were reported to have the best performance according to Mesquita Sa Junior

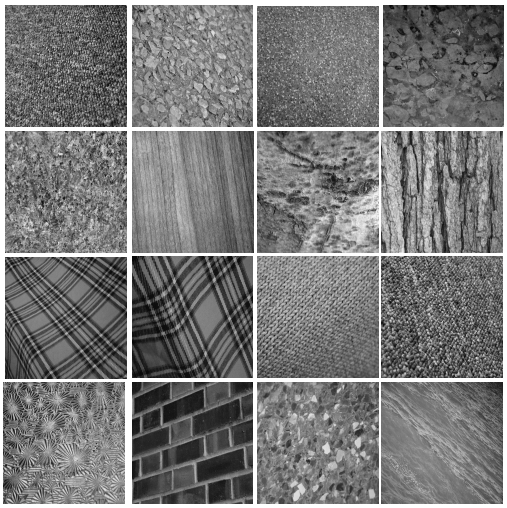

Fig. 6. A sample from the UIUC texture image dataset used in the experiments.

et al. [1]. In all experiments we split the datasets in training and testing subsets (40% for training and 60% for testing).

Figs. 7 and 8 show the recognition rates of TAISG as C is increased for both data sets. Additionally, we show the performance of TASPG. The dotted line shows when the classification rate of TAISG is significantly different from TASPG's rate.

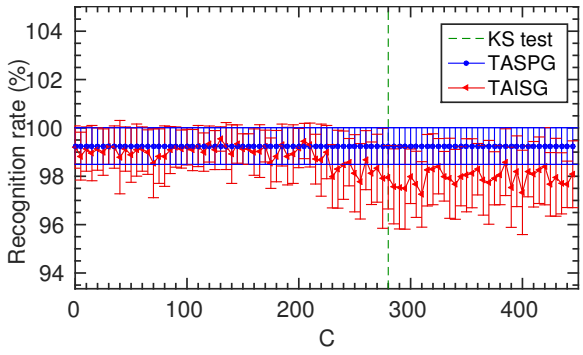

Fig. 7. Comparison of recognition rates of TAISG and TASPG for Brodatz's data set. The dotted line indicates when the classification rate for a given C is significantly different from TASPG's rate. In this case, the value of is $C = 280$.

Notice that TAISG recognition rate decreases as C increases and TAISG results are equivalent to TASPG up to $C = 280$ and $C = 120$ for Brodatz's and UIUC's data sets, respectively. The effect of C on reducing the number of visited nodes can be seen in Figs 9 and 10. Those figures show the reduction rate of visited nodes when compared with TASPG.

Considering the results presented in this section, we can state that TAISG and TASPG achieved a similar classification rate when $C = 280$ and $C = 120$ for Broadtz's and UIUC's data sets respectively. For the same values of C, TAISG reduced the number of visited nodes to 11.54% and 37.87% of the number of visited nodes in TASPG.

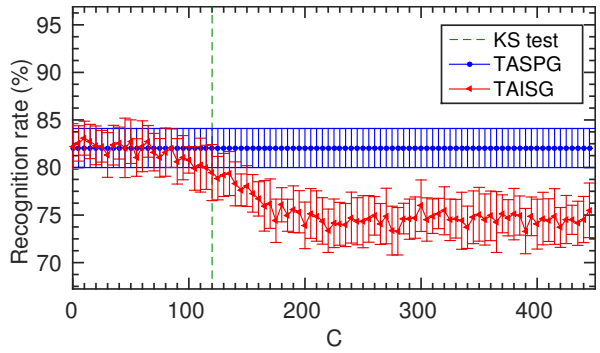

Fig. 8. Comparison of recognition rates of TAISG and TASPG for UIUC's data set. The dotted line indicates when the classification rate is significantly different from TASPG's rate. In this case, the value of is $C = 120$.

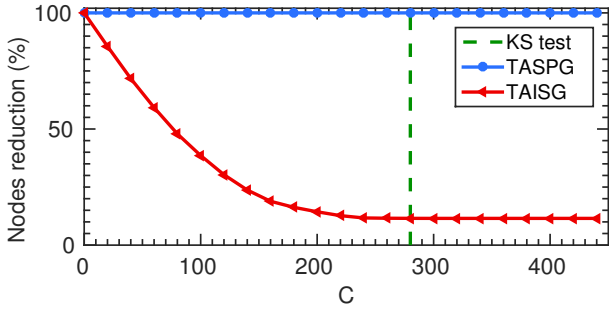

Fig. 9. Comparison of the number of visited nodes of TASPG and TAISG for Broadtz's data set as C is varied. The number of visited nodes for TASPG is defined as the baseline (100%) and for TAISG results are shown as a percentage of this baseline. The dashed vertical line indicates when the recognition rates of TAISG and TASPG are significantly different.

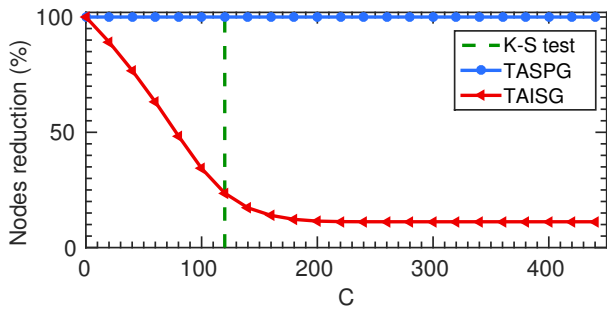

Fig. 10. Comparison of the number of visited nodes of TASPG and TAISG for UIUC's data set as C is varied. The number of visited nodes for TASPG is defined as the baseline (100%) and for TAISG results are shown as a percentage of this baseline. The dashed vertical line indicates when the recognition rates of TAISG and TASPG are significantly different.

The reduction on the number of visited nodes and the classification performance were also analyzed when l is changed. Tables I and II show the recognition rates and the visited nodes' reduction rates for several values of l. Experiments were repeated 10 times and tables show both average recognition rates and standard deviations.

Notice that, even though TAISG reduced the number of visited states for all values of l, this reduction is more significant when the number of sub-images is increased (the

TABLE I
RESULTS FOR BRODATZ'S DATA SET.

Sets of sub-images size $\{l_1, l_2, ..., l_n\}$	Recognition rate TASPG	Recognition rate TAISG	C	Nodes reduction
{4, 5}	0.9333 ± 0.0264	0.9388 ± 0.0172	445	0.7367
{4, 5, 8}	0.9562 ± 0.0162	0.9421 ± 0.0178	445	0.7789
{4, 5, 8, 10}	0.9558 ± 0.0130	0.9412 ± 0.0192	375	0.8073
{4, 5, 8, 10, 20}	0.9575 ± 0.0175	0.9437 ± 0.0150	445	0.8358
{4, 5, 8, 10, 20, 25}	0.9604 ± 0.0166	0.9483 ± 0.0172	445	0.8521
{4, 5, 8, 10, 20, 25, 40}	0.9537 ± 0.0214	0.9429 ± 0.0203	445	0.8705
{4, 5, 8, 10, 20, 25, 40, 50}	0.9925 ± 0.0076	0.9796 ± 0.0131	280	0.8846
{4, 5, 8, 10, 20, 25, 40, 50, 100}	0.9912 ± 0.0074	0.9796 ± 0.0131	445	0.8971

TABLE II
RESULTS FOR UIUC'S DATA SET.

Sets of sub-images size $\{l_1, l_2, ..., l_n\}$	Recognition rate TASPG	Recognition rate TAISG	C	Nodes reduction
{4, 5}	0.6760 ± 0.0225	0.6538 ± 0.0263	35	0.0775
{4, 5, 8}	0.6970 ± 0.0239	0.6683 ± 0.0262	75	0.1899
{4, 5, 8, 10}	0.6970 ± 0.0262	0.6775 ± 0.0216	45	0.1081
{4, 5, 8, 10, 20}	0.6993 ± 0.0258	0.6667 ± 0.0258	100	0.3026
{4, 5, 8, 10, 20, 25}	0.6918 ± 0.0252	0.6695 ± 0.0234	125	0.3997
{4, 5, 8, 10, 20, 25, 40}	0.7035 ± 0.0250	0.6850 ± 0.0295	105	0.3336
{4, 5, 8, 10, 20, 25, 40, 50}	0.8393 ± 0.0162	0.8193 ± 0.0160	120	0.6213
{4, 5, 8, 10, 20, 25, 40, 50, 100}	0.8385 ± 0.0155	0.8293 ± 0.0236	150	0.5631

dimension of l is increased). This is an expected result since increasing the dimension of l implies an increase in the number of executions of the search algorithm.

The classification rate of TAISG is also compared with several state-of-the-art texture analysis methods. All methods are described in the related work section.

For all methods we also used the LDA algorithm for classification. All hyper-parameters defined for each method were set as recommended by the method's original reference. Results for Broadtz's and UIUC's data sets are shown in Table III.

TABLE III
RECOGNITION RATE FOR SEVERAL TEXTURE ANALYSIS METHODS

Algorithm	Brodatz	UIUC
Fourier descriptors	87.75	35.10
Co-occurrence matrices	82.50	41.10
Gabor filters	97.00	56.10
Wavelet descriptors	87.50	38.80
Tourist walk	95.50	48.70
Gravitation (BouligandMinkowski)	98.75	57.80
Gravitation (gliding-box)	97.00	54.10
TAISG	97.96	81.93

Notice that TAISG outperformed almost all methods on Broadtz's data set. Only the Gravitation (Bouligand-Minkowski) method achieved a higher recognition rate with a difference of less than 1%. In UIUC's data set the performance gap between TAISG and the other approaches is noteworthy. The difference between TAISG and the second best performance is around 25%.

VI. CONCLUSION

In this paper, we proposed a variant of the TASPG method for texture recognition in images. The proposed approach, called TAISG, adds a heuristic term to the cost function used on Dijkstra's algorithm so that the search procedure can be performed in a more direct way. The use of that heuristic function reduces the number of visited nodes, thus improving the method's computational efficiency.

The effect of the heuristic term on the search method's result is controlled by a parameter C. In TAISG, C is chosen so that its performance is not significantly different from TASPG's performance. For the statistical significance analysis we used the K-S hypothesis test.

On the basis of our experiments, we can state that TAISG reached a significant reduction on the number of visited nodes (11.54% on Broadtz's data set and 37.87% on UIUC's data set) without significant reduction on the recognition rate when compared with TASPG. The recognition rate is also superior to other state-of-the-art texture analysis methods in most cases (except for Gravitation BouligandMinkowski on Broadtz's data set) with even more noticeable results on UIUC's data set.

Future work shall include the design of admissible heuristics for the search algorithm. This may avoid the cross validation procedure, reducing the time spent to find C, while preserving the optimality of Dijkstra's algorithm.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of CAPES/FUNCAP (Grant AE1-0052-00077.01.00/11) and CNPq (research fellowship 307941/2013-2)

REFERENCES

[1] J. J. de Mesquita Sa Junior, A. R. Backes, and P. C. Cortez, "Texture analysis and classification using shortest paths in graphs," *Pattern Recognition Letters*, vol. 34, no. 11, pp. 1314 – 1319, 2013.

[2] D. S. Ebert, K. F. Musgrave, D. Peachey, K. Perlin, and S. Worley, *Texturing and Modeling: A Procedural Approach*. Morgan Kaufmann, 2002.

[3] R. Haralick, "Statistical and structural approaches to texture," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 786–804, May 1979.

[4] G. Beliakov, S. James, and L. Troiano, "Texture recognition by using glcm and various aggregation functions," in *Fuzzy Systems, 2008. FUZZ-IEEE 2008. (IEEE World Congress on Computational Intelligence). IEEE International Conference on*, June 2008, pp. 1472–1476.

[5] R. Azencott, J.-P. Wang, and L. Younes, "Texture classification using windowed fourier filters," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 2, pp. 148–153, Feb 1997.

[6] B. Manjunath and W. Ma, "Texture features for browsing and retrieval of image data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 8, pp. 837–842, Aug 1996.

[7] A. F. Costa, G. Humpire-Mamani, and A. J. M. Traina, "An efficient algorithm for fractal analysis of textures," in *2012 25th SIBGRAPI Conference on Graphics, Patterns and Images*, Aug 2012, pp. 39–46.

[8] L. daF. Costa, F. A. Rodrigues, G. Travieso, and P. R. V. Boas, "Characterization of complex networks: A survey of measurements," *Advances in Physics*, vol. 56, no. 1, pp. 167–242, January 2005.

[9] J. B. Florindo and O. M. Bruno, "Texture analysis by multi-resolution fractal descriptors," *Expert Systems with Applications*, vol. 40, no. 10, pp. 4022 – 4028, 2013.

[10] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271.

[11] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Pearson Education, 2003.

[12] C. S. Lu, P. C. Chung, and C. F. Chen, "Unsupervised texture segmentation via wavelet transform," *Pattern Recognition*, vol. 30, no. 5, pp. 729 – 742, 1997.

[13] J. J. de Mesquita Sa Junior and A. R. Backes, "A simplified gravitational model to analyze texture roughness," *Pattern Recognition*, vol. 45, no. 2, pp. 732 – 741, 2012.

[14] A. R. Backes, W. N. Gonalves, A. S. Martinez, and O. M. Bruno, "Texture analysis and classification using deterministic tourist walk," *Pattern Recognition*, vol. 43, no. 3, pp. 685 – 694, 2010.

[15] J. J. de Mesquita Sá, A. R. Backes, and P. C. Cortez, "A simplified gravitational model for texture analysis," *Journal of Mathematical Imaging and Vision*, vol. 47, no. 1, pp. 70–78, 2013.

[16] M. R. Schroeder, *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*, 1991.

[17] F. J. Massey, *Journal of the American Statistical Association*, vol. 46, no. 253, 1951.

[18] P. Brodatz, *Textures : a photographic album for artists and designers*. New York: Dover Publications, 1966.

[19] S. Lazebnik, C. Schmid, and J. Ponce, "A sparse texture representation using local affine regions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1265–1278, Aug 2005.