LEARNING SIMPLE TEXTURE DISCRIMINATION FILTERS

Rui F. C. Guerreiro *Pedro M. Q. Aguiar*

Institute for Systems and Robotics, Instituto Superior Técnico
Av. Rovisco Pais, 1049-001 Lisboa, Portugal
rfcg@isr.ist.utl.pt, aguiar@isr.ist.utl.pt

ABSTRACT

Current texture analysis methods enable good discrimination but are computationally too expensive for applications which require high frame rates. This occurs because they use redundant calculations, failing in capturing the essence of the texture discrimination problem. In this paper we use a learning approach to obtain simple filters for this task. Although others have proposed learning-based methods, we are the first to simultaneously achieve discrimination rates comparable with state-of-the art methods at high frame rates. We particularize the general methodology to different filter structures, *e.g.*, rotationally discriminant filters and rotationally invariant ones. We use Genetic Algorithms for learning and test our method against state-of-the-art ones, using the Brodatz album.

***Index Terms*—Image texture analysis, genetic algorithms**

I. INTRODUCTION

Images of real scenes often contain texture patterns. The analysis of such patterns is used in texture discrimination, segmentation, synthesis, and Shape From Texture. This has multiple applications, *e.g.*, automated textile and paint inspection, medical image analysis, document processing.

Although texture discrimination has been studied for many years (for a review on texture analysis, see [1]) there are no highly discriminant methods that work at high frame rates. The most common state-of-the-art discrimination methods are Gray-Level Co-occurrence Matrices (GLCM) and Gabor Filter Banks (GFB).

GLCM [2] creates a $2^8 \times 2^8$ matrix (for 8-bit precision) for each texture patch. Its ij entry computes the co-occurrences of a pair of neighboring pixels with values i and j. From this matrix, a vector of so-called Haralick features (Energy, Contrast, Dissimilarity, ...) is computed and used in measuring Euclidean distances between patches.

GFB [3] compute, for every pixel, a filter bank of about ten to fifty filters. The large dimensional vectors per pixel are clustered with K-means and a histogram with K bins is computed. Each bin counts the pixels that belong to each cluster. The distance between texture patches is computed with histogram distance metrics such as the χ^2 test. Both GLCM and GFB are computationally very complex.

In this paper we use a learning approach to obtain a computationally simple discrimination method. Others have used learning approaches for this task. Reference [4] proposed a Genetic Programming scheme to evolve discriminators of pixel histograms of different textures. This method leads to simple but poor discrimination. A neural-network is used in [5] and reference [6] challenges filter-bank methods by using small pixel neighborhoods in a Markov Random Fields scheme. These methods, like the ones referred above, are highly discriminant but computationally too expensive for high frame rate implementations.

Our approach imposes simple filters and we particularize the general methodology to filter structures capable of high frame rates, which implement rotationally discriminant and rotationally invariant filters. We use a Genetic Algorithm for learning the filter parameters. We obtain an accuracy comparable to state-of-the-art methods in discriminating both known and new textures of the Brodatz album.

Paper organization Section II introduces our learning methodology for texture discrimination. Section III particularizes it for rotationally discriminant and invariant filters and Section IV shows how to learn them. Section V shows experimental results and we conclude on Section VI.

II. TEXTURE DISCRIMINATION FILTERS

Consider a training set of N texture patches from T textures classes, and a function $C:\{0,...,N-1\}\to\{0,...,T-1\}$ that maps each patch to the corresponding class. Patch n is denoted by $\boldsymbol{t}_n\in\mathbb{R}^{p\times q}$ and $\phi:\mathbb{R}^{p\times q}\to\mathbb{R}^v$ is the texture discrimination filter we wish to learn. Modeling $\phi(\boldsymbol{t}_n)$ as a random variable following a Gaussian distribution of average $\boldsymbol{\mu}_{C_n}$ and covariance $\boldsymbol{I}$, the maximum-likelihood class estimate for a new texture patch, $\hat{j}_{ML}$, is given by

$$\hat{j}_{ML} = \arg\max_j p(\phi(\boldsymbol{t})|j) = \arg\min_j \left\|\phi(\boldsymbol{t}) - \boldsymbol{\mu}_j\right\|_2^2, \quad (1)$$

$$\boldsymbol{\mu}_j = \frac{\sum_{\{n:C_n=j\}} \phi(\boldsymbol{t}_n)}{\sum_{\{n:C_n=j\}} 1}, \forall_{j\in\{0,...,T-1\}}. \quad (2)$$

Partially supported by FCT, under ISR/IST plurianual funding (POSC program, FEDER) and grants MODI-PTDC/EEA-ACR/72201/2006 and SFRH/BD/48602/2008.

Although it is possible to learn a discrimination filter $\phi(.)$ that handles the complete texture patch directly, this is not the typical approach in texture discrimination methods. Julesz [7] studied extensively the way humans perceive textures and proposed the *theory of textons*. Textons are the basic sub-elements of textures. Julesz considered that humans discriminate textures by first analyzing individual textons and then integrating that information with respect to their occurrence rates, locations, etc. Typical discrimination methods use this approach: they analyze smaller portions of the texture patch – the textons – and combine the overall data into a discriminating feature vector.

We incorporate this approach into our texture discrimination filter $\phi(.)$ aswell. Let $\boldsymbol{t}_n(x,y)$ denote a small subsection of $\boldsymbol{t}_n$ centered at point (x,y) – a texton. We define a *texton analysis filter*, $\psi(.)$, which extracts features from each texton, $\psi\left(\boldsymbol{t}_n(x,y)\right)$. These features are concentrated in an *aggregation matrix* , $\Psi(\boldsymbol{t}_n)$, which is processed with an *integration function*, $\Phi(.)$. This function combines the various texton features to form a global feature vector for the texture patch, $\phi(\boldsymbol{t}_n)$. Formally,

$$\Psi(\boldsymbol{t}_n) = \begin{bmatrix} \psi(\boldsymbol{t}_n(0,0)) & \cdots & \psi(\boldsymbol{t}_n(p-1,0)) \\ \vdots & \ddots & \vdots \\ \psi(\boldsymbol{t}_n(0,q-1)) & \cdots & \psi(\boldsymbol{t}_n(p-1,q-1)) \end{bmatrix},$$
$$\phi(\boldsymbol{t}_n) = \Phi(\Psi(\boldsymbol{t}_n)). \tag{3}$$

Therefore, learning texture filter $\phi(.)$ consists of learning the texton analysis filter $\psi(.)$ and integration function $\Phi(.)$.

III. EXAMPLES OF FILTER STRUCTURES

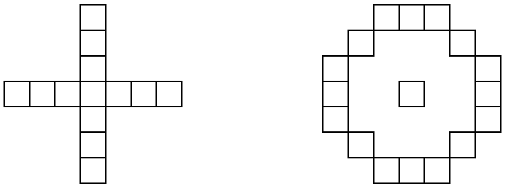
In this section we propose two filter structures for $\phi(.)$, using the texton approach of (3). They implement rotationally discriminant filters and rotationally invariant ones.

Texton analysis filter – The two filter structures we propose have similar texton analysis filters. We impose that each filter is a point-wise multiplication of a texton and a matrix. Consequently, the aggregation matrix $\Psi(\boldsymbol{t}_n)$ is simply the convolution of texture $\boldsymbol{t}_n$ and matrix $\boldsymbol{w}$,

$$\Psi(\boldsymbol{t}_n) = \boldsymbol{t}_n \otimes \boldsymbol{w}, \tag{4}$$

where $\otimes$ is the convolution operator. Using I texton analysis functions, the i-th aggregation matrix, $\Psi^i(\boldsymbol{t}_n)$, is the convolution of texture patch $\boldsymbol{t}_n$ and matrix $\boldsymbol{w}^i$, with $i \in \{1,\ldots,I\}$. I is a parameter that we can adjust. As we increase I, the overall discrimination ability improves, at the expense of a higher computational cost.

Matrices $\boldsymbol{w}^i$ are estimated in the training phase and make up different filter supports through the location of non-zero values. The rotationally discriminant filters use $\boldsymbol{w}^i$ with horizontal and vertical non-zero values (Figure 1.a) and the rotationally invariant ones use $\boldsymbol{w}^i$ with circular non-zero values (Figure 1.b).



(a) rotationally discriminant (b) rotationally invariant

Fig. 1. Location of non-zero values in texton analysis filters

Integration function – The convolutions of a texture patch with a set of filters, $\left\{\Psi^i(\boldsymbol{t}_n)\right\}$, aggregates data about textons. Many texture discrimination methods compute a histogram of texton data and use it as a discrimination feature. In this scenario, the distance between two texture patches is given by histogram distance metrics such as the χ^2 test. The integration function we propose, $\Phi(.)$, approximates the histogram of $\Psi^i(\boldsymbol{t}_n)$ by easy-to-compute high-order statistics. We compute the average, standard deviation and two standardized moments of $\Psi^i(\boldsymbol{t}_n)$,

$$\begin{aligned} \kappa_1 &= \mu'_1 \equiv \mu, \\ \kappa_2 &= \sqrt{\mu'_2 - {\mu'_1}^2} \equiv \sigma, \\ \kappa_3 &= \left(\mu'_3 - 3\mu'_2\mu'_1 + 2{\mu'_1}^3\right)/\kappa_2^3, \\ \kappa_4 &= \left(\mu'_4 - 4\mu'_3\mu'_1 + 6\mu'_2{\mu'_1}^2 - 3{\mu'_1}^4\right)/\kappa_2^4, \\ \mu'_l &= E\left[\Psi\left(\boldsymbol{t}_n\right)^l\right] = \frac{1}{pq}\sum_{x=0}^{q-1}\sum_{y=0}^{p-1}\left(\Psi\left(\boldsymbol{t}_n\right)_{x,y}\right)^l, \end{aligned} \tag{5}$$

where κ_j is the j-th statistic we compute and μ'_l is the l-th moment about the origin of $\Psi(\boldsymbol{t}_n)$. μ'_l can be computed efficiently by using a sliding window box filter for each l. The discrimination vector, $\phi(\boldsymbol{t}_n)$, is obtained by multiplying weights $\boldsymbol{\alpha}_j^i$ by the $\kappa_j\left(\Psi^i\left(\boldsymbol{t}_n\right)\right)$ statistics.

Method I - Rotationally discriminant filter – We define

$$\Psi_H^i(\boldsymbol{t}_n) = \boldsymbol{t}_n \otimes \left(\boldsymbol{w}^i\right)^T,$$
$$\Psi_V^i(\boldsymbol{t}_n) = \boldsymbol{t}_n \otimes \boldsymbol{w}^i,$$
$$\phi(\boldsymbol{t}_n) = \Phi(\Psi(\boldsymbol{t}_n)) = \begin{bmatrix} \boldsymbol{\alpha}_{H_1^1}\kappa_1\left(\Psi_H^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{H_2^1}\kappa_2\left(\Psi_H^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{H_3^1}\kappa_3\left(\Psi_H^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{H_4^1}\kappa_4\left(\Psi_H^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{V_1^1}\kappa_1\left(\Psi_V^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{V_2^1}\kappa_2\left(\Psi_V^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{V_3^1}\kappa_3\left(\Psi_V^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{V_4^1}\kappa_4\left(\Psi_V^1(\boldsymbol{t}_n)\right) \\ \vdots \end{bmatrix},$$

where $\left\{\Psi_H^i(\boldsymbol{t}_n), \Psi_V^i(\boldsymbol{t}_n)\right\}$ is the set of aggregation matrices given by the convolution of texture patch $\boldsymbol{t}_n$ with matrices $\left\{\left(\boldsymbol{w}^i\right)^T, \boldsymbol{w}^i\right\}$. The non-zero values of $\boldsymbol{w}^i$ define a filter with vertical support and $\left(\boldsymbol{w}^i\right)^T$, a filter with horizontal support. $\left\{\kappa_j\left(\Psi_H^i(\boldsymbol{t}_n)\right), \kappa_j\left(\Psi_V^i(\boldsymbol{t}_n)\right)\right\}$ are the j-th statistics of the aggregation matrices, which are multiplied by weights $\boldsymbol{\alpha}_j^i = \left\{\boldsymbol{\alpha}_{H_j^i}, \boldsymbol{\alpha}_{V_j^i}\right\}$ to make up a discrimination vector, $\phi(\boldsymbol{t}_n)$. Vector $\phi(\boldsymbol{t}_n)$ has size $(8I) \times 1$.

Method II - Rotationally invariant filter – We define

$$\Psi_C^i(\boldsymbol{t}_n) = \boldsymbol{t}_n \otimes \boldsymbol{w}_C^i,$$

$$\phi(\boldsymbol{t}_n) = \Phi\left(\Psi_C(\boldsymbol{t}_n)\right) = \begin{bmatrix} \boldsymbol{\alpha}_{C_1^1}\kappa_1\left(\Psi_C^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{C_2^1}\kappa_2\left(\Psi_C^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{C_3^1}\kappa_3\left(\Psi_C^1(\boldsymbol{t}_n)\right) \\ \boldsymbol{\alpha}_{C_4^1}\kappa_4\left(\Psi_C^1(\boldsymbol{t}_n)\right) \\ \vdots \end{bmatrix},$$

where the non-zero values of $\boldsymbol{w}_C^i$ create a filter with circular support, as illustrated in Figure 1.b. The aggregation matrix, $\Psi_C^i(\boldsymbol{t}_n)$, is the convolution of $\boldsymbol{t}_n$ and $\boldsymbol{w}_C^i$ and $\kappa_j\left(\Psi_C^i(\boldsymbol{t}_n)\right)$, the corresponding j-th statistic. The texture discrimination vector $\phi(\boldsymbol{t}_n)$ is created by multiplying each statistic with weight $\boldsymbol{\alpha}_{C_j^i}$ and has thus size $(4I) \times 1$.

IV. LEARNING TEXTURE FILTERS

In this section, we show how to learn the filter parameters, in particular for the structures presented in Section III.

Problem formulation – Assuming that all classes are equally likely, the sum of the probabilities of estimating the correct class, C_n, is given by

$$\begin{aligned}\sum_{n=0}^{N-1} p\left(C_n|\phi\left(\boldsymbol{t}_n\right)\right) &= \sum_{n=0}^{N-1} \frac{p\left(\phi\left(\boldsymbol{t}_n\right)|C_n\right)}{\sum_{i=0}^{T-1} p\left(\phi\left(\boldsymbol{t}_n\right)|i\right)} = \\ &= \sum_{n=0}^{N-1} \frac{\mathcal{N}(\phi(\boldsymbol{t}_n)|\boldsymbol{\mu}_{C_n}, \boldsymbol{I})}{\sum_{i=0}^{T-1} \mathcal{N}(\phi(\boldsymbol{t}_n)|\boldsymbol{\mu}_i, \boldsymbol{I})}, \qquad (6)\end{aligned}$$

where each $\boldsymbol{\mu}_j$ is given by (2). We learn the discrimination filter, $\phi(.)$, by maximizing (6) as a function of its internal parameters, $\boldsymbol{w}^i$ and $\boldsymbol{\alpha}^i$, for a chosen I.

Although we address the maximization of (6) in the sequel, this problem can be simplified by removing its denominator. Because the denominator favors solutions where texture classes have different averages, it is necessary to impose that all averages, $\boldsymbol{\mu}_i$, are at least ϵ apart. This leads to the simpler optimization problem,

$$\begin{aligned}\max_{\phi} \sum_{n=0}^{N-1} \mathcal{N}\left(\phi(\boldsymbol{t}_n)|\boldsymbol{\mu}_{C_n}, \boldsymbol{I}\right) &= \min_{\phi} \sum_{n=0}^{N-1} \left\|\phi(\boldsymbol{t}_n) - \boldsymbol{\mu}_{C_n}\right\|_2^2, \\ \text{subject to } \left\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\right\|_2^2 &\geq \epsilon^2, \forall_{i\neq j}. \qquad (7)\end{aligned}$$

Filter estimation – The maximization of (6) is an optimization problem in the domain of the internal parameters of $\phi(.)$, $\boldsymbol{\alpha}^i$ and $\boldsymbol{w}^i$. Because function (6) is nonconcave, this problem is complex. We use a *Genetic Algorithm* (GA) [8] for this purpose. Although GAs provide no guarantee of convergence to the global optimum, they are known to perform well in nonconcave problems, as the crossover step moves the population away from local optima that a traditional hill climbing algorithm might get stuck in. They are easy to implement but have the disadvantage of being slow, although the training phase occurs offline. Our algorithm consists of:

GENETIC ALGORITHM

1) Create a population where each member contains internal parameters of $\phi(.)$, initialized with random uniform values
2) Compute fitness of each member with *fitness function* (6)
3) *Elitism*: select and preserve the fittest members
4) Create a new generation from the fittest members by:
4.1) *Crossover*: copying values from two random members
4.2) *Mutation*: adding random uniform noise to a copy of a member. The magnitude of the added noise decreases with every generation, to enable hill-climbing
5) Iterate from 2 onwards until convergence.

V. EXPERIMENTS

Training – We use a training set consisting of $N = 1025$ texture patches from $T = 41$ texture classes. Each class corresponds to an image from the Brodatz album [9], from which 25 non-overlapping patches are extracted. The Brodatz album, illustrated in Figure 2, is the de facto standard for evaluating texture analysis methods, with hundreds of studies having been applied to its images.

We use patches of size $p \times q = 60 \times 60$ and texton subpatches of size 7×7. The genetic algorithm, described in Section IV, uses a population of 5000 members, 10% elitism rate, 50% crossover rate and 40% mutation rate. It is initialized with random uniform values of magnitude 1 and the mutation noise has magnitude 1 in the first generation and is multiplied by 0.8 in every generation. We obtain $\boldsymbol{w}^i$ and $\boldsymbol{\alpha}^i$ that maximize (6) for the texture discrimination filters, $\phi(.)$, we propose in Section III.

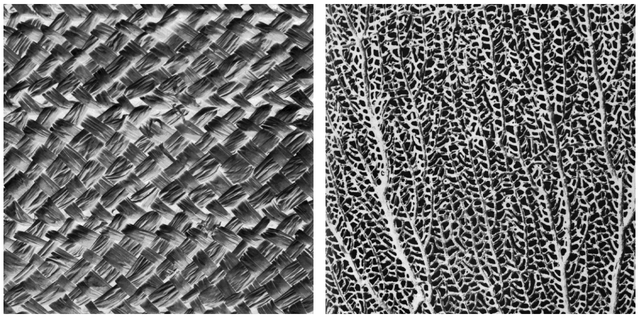


Fig. 2. D18 and D87 images from the Brodatz database [9]

Test – We evaluate the discrimination filters obtained in the training phase, $\phi(.)$, in two scenarios. In the first, we classify texture patches from the same classes used in the training phase (using $p \times q = 60 \times 60$). This illustrates applications where we know beforehand the set of textures we need to discriminate and we want a filter $\phi(.)$ that

Table I. Classification rate. † – uses database 1 of [10]

Method I			Method II			GLCM†	GFB†
I	training	DB1†	I	training	DB1†		
1	93.2%	94.6%	4	80.0%	72.7%	66.7%	92.2%
2	97.7%	97.7%					
3	98.7%	97.1%					

is highly discriminant and efficient in this scenario. In the second scenario, we use filter $\phi(.)$ on database 1 (DB1) of the large comparative study of popular texture discrimination methods, in [10]. By doing so we can: 1) directly compare the performance of our method against other popular methods and; 2) test the generalization ability of filter $\phi(.)$, since 45% of the classes in this database are not present in the training set. DB1 [10] consists of $N = 960$ non-overlapping texture patches of size $p \times q = 160 \times 160$. It contains $T = 60$ texture classes where each corresponds to an image of the Brodatz album.

Analysis – Table I shows the accuracy of our method and two state-of-the-art discrimination methods: GLCM [2] and GFB [3]. Method I has an accuracy comparable to state-of-the-art methods, for any I and for both test sets. In particular, a single horizontal and vertical convolution and respective high-order statistics, extremely fast to compute, achieves an accuracy of 93.2% and 94.6% in both test sets. Increasing I yields higher classification accuracy, except for method I and $I = 3$, in DB1 [10]. This occurs due to a natural overfitting of function $\phi(.)$ to the training set. Table I also shows that method II is able to successfully classify 80.0% and 72.7% of the patches in both test sets, with $I = 4$, regardless of their rotation angle.

The frequency response of filter $\boldsymbol{w}$, for method I and $I = 1$, is plotted in Figure 3.a. It shows that $\boldsymbol{w}$ is essentially a wide band-pass filter that attenuates the lowest and highest frequencies. This suggests that textures are identified simply by analyzing the statistics of the transition magnitudes. The low frequency attenuation indicates that the average value of each texton is not useful in texture discrimination. Because the Brodatz album is not very sharp, the high-frequency attenuation is probably meant to reduce the effect of noise. Figure 3.b plots the frequency response of filters $\boldsymbol{w}^i$, for method I and $I = 3$. It shows a finer (and redundant) distribution of the frequency spectrum by each filter, which justifies the increased accuracy as I increases.

VI. CONCLUSIONS

In this paper, we use a learning approach to obtain simple filters for texture discrimination. We particularized the general methodology for rotationally discriminant and rotationally invariant filter structures, obtaining an accuracy comparable to state-of-the-art methods, at high frame rates.

VII. REFERENCES

[1] M. Tüceryan and A. Jain, "Texture analysis," in *Handbook of Pattern Recognition & Computer Vision*, River Edge, NJ, USA, 1993, pp. 235–276, World Scientific Publishing Co., Inc.

[2] R. Haralick, I. Dinstein, and K. Shanmugam, "Textural features for image classification," *TSMC*, vol. SMC-3, pp. 610–621, 1973.

[3] J. Malik, S. Belongie, T. Leung, and J. Shi, "Contour and texture analysis for image segmentation," *IJCV*, vol. 43, no. 1, pp. 7–27, 2001.

[4] B. Lam, V. Ciesielski, and A. Song, "Visual texture classification and segmentation by genetic programming," in *Genetic and Evolutionary Computation for Image Proc. and Analysis*. 2008, Hindawi Pub. Co.

[5] F. Tivive and A. Bouzerdoum, "A nonlinear feature extractor for texture segmentation," *ICIP*, 2007.

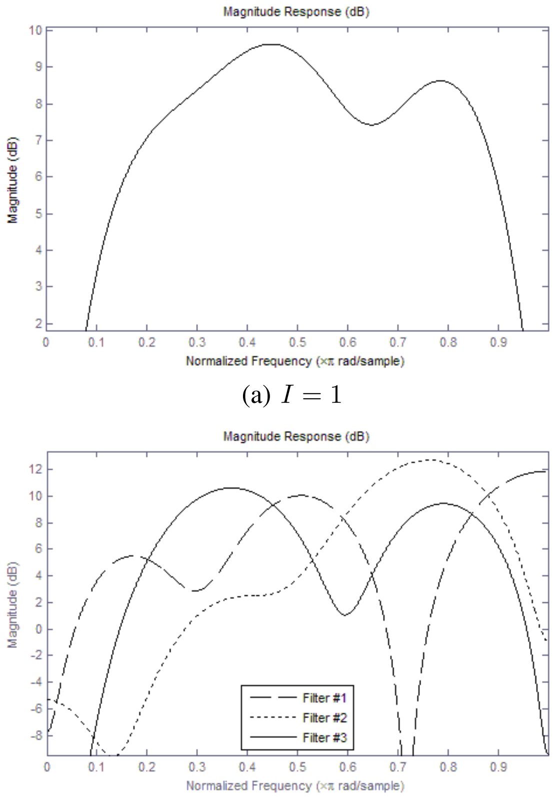
[6] M. Varma and A. Zisserman, "Texture classification: are filter banks necessary?," in *CVPR*, 2003.

[7] B. Julesz, "Textons, the elements of texture perception, and their interactions," *Nature*, vol. 290, 1981.

[8] M. Mitchell, *An Introduction To Genetic Algorithms*, MIT Press, Cambridge, MA, USA, 1996.

[9] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*, Dover Publications, NY, 1966.

[10] M. Partio, B. Cramariuc, and M. Gabbouj, "An ordinal co-occurrence matrix framework for texture retrieval," *Journal on Image and Video Proc.*, 2007.



(a) $I = 1$

(b) $I = 3$

Fig. 3. Frequency response of filters $\boldsymbol{w}^i$, for method I