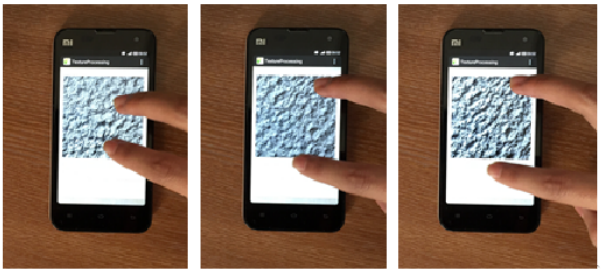
Toward a psychophysical-based procedural texture generation system for interactive design

Xiaoxu Cai, Jun Liu, Lin Qi, Junyu Dong
Department of Information Science and Engineering
Ocean University of China
Qingdao, China
Email: qilin@ouc.edu.cn

Ying Gao
Colleage of Information Science and Technology
Chengdu University of Technology
Chengdu , China

Hui Yu
School of Creative Technologies
University of Portsmouth
London, UK

***Abstract*—Procedural textures have been widely used as they can be easily generated from various mathematical models. However, the model parameters are not perceptually meaningful or uniform for non-expert users. In this paper, we proposed a system that can generate procedural textures interactively along certain perceptual dimensions. We built a procedural texture dataset and measured twelve perceptual properties of a small subset through psychophysical experiments. The perceived magnitude of the rest textures was estimated by Support Vector Machines using computational features from a cascaded PCA network. For a given texture displayed on a touch screen, the user makes finger gestures which were then transferred to magnitude changes in perceptual space. The texture in the database that matches the new perceptual scale and with nearest distance in computational feature space will be chosen and displayed. We reported our experiment results for two particular perceptual properties: surface roughness and directionality. Other properties can be manipulated similarly.**



a b c

Fig. 1: The user makes a "zoom" finger gesture on the touch screen. From a to c, as the distance between the two fingers increases, the perceived roughness of the sample texture is increased.

I. Introduction

Procedural texture models provide a convenient and economic solution to produce a variety of textures in fields like computer games, animation, geology, and digital design. Many procedural texture models have been proposed, among which Cellular texture [1], Perlin noise [2], Texton placement [3], Wavelet noise [4], Fractal texture [5], Cellular Automaton [6], Reaction-Diffusion texture [7] and Matrix transformation are the most popular ones. However, there is a gap between the demand of ordinary users and the actual output of the models: humans normally prefer to describe desired textures in perceptual dimensions, for example a texture that is "repetitive, directional and highly structured", whereas the procedural models are expressed in mathematical formulas with obscure parameters. As these models and their parameters are not meaningful or uniform in perceptual space, the challenging problem is how to pick up the proper procedural model and set its parameters to generate certain textures that satisfy user's requirement. Therefore, a system that hides the physical model details and can interactively generate textures along certain perceptual dimensions is urgently required.

In this paper, we proposed such a system that users can easily change the appearance of procedural textures along certain perceptual dimensions without being aware of the specific models. Fig. 1 shows a user using our system on a smart phone. We chose 22 procedural texture models and two perceptual properties to build a system prototype. Additional models and perceptual properties can be manipulated in a similar manner. We built an image pool by sampling the parametric space of the chosen models and measured 12 perceptual scales for a small subset. We used the state-of-the-art technique to extract computational features of all the images and trained Support Vector Machines to regress perceptual scales for the rest textures in the pool. The system was implemented on devices with a touch screen. For a given texture displayed on the screen, a user can make finger gestures on the screen if he or she wants to change the texture for some certain perceptual properties. The system reads the magnitude of the finger gestures and transfers to change in perceptual space. The texture in the pool that matches the new perceptual scale and with nearest distance in computational feature space will be chosen and displayed. Fig.2 illustrates the general framework of the system.

II. Related Work

A. Procedural texture generation models

Procedural models are widely used in research and industry fields as they can efficiently add rich visual details to synthetic images [8]. Different textures can be produced by varying a set of input parameters. Several techniques have been proposed for controlling parameters of procedural models in frequency domain [9] and spatial domain [10] [11] . Other solutions

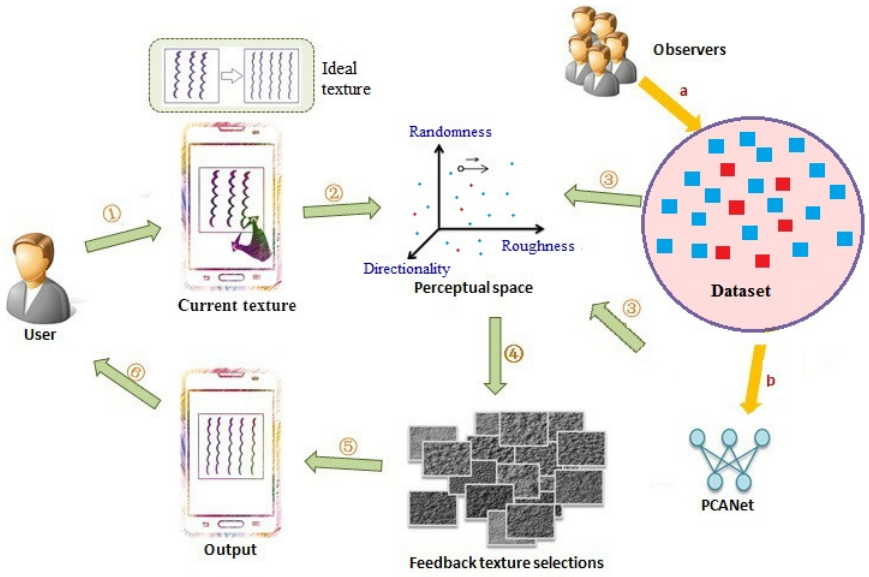


Fig. 2: Our framework. 1 - User opens application; 2 - User changes one perceptual feature; 3 - Collection of perceptual scales; 4 - Texture selection collated; 5 - Distance measure ; 6 - Displaying texture on the Screen. a - Observers given the perceptual scales of samples; b - Extracting the computational features using PCANet and then Predicting the perceptual scales by training models. The red part stands for the textures that used for psychophysical experiments, the blue part stands for the textures whose perceptual scales were estimated.

include extracting noise functions from an example image to automatically produce an image closely resembling the example [12]. The limitation of these methods is that they only work for a sub-class of procedural texture models. Textures generated by a single model often share similar appearance, whereas the texture user desired may comes from another model or fused output of several models.

We proposed to build a large pool of textures by collecting several popular procedural models and densely sampling model parameters. Then we studied the properties of these textures by computational and psychophysical approaches.

B. Computational and perceptual texture features

Human vision system is good at discriminating textures, which employs a powerful mechanism for feature extraction and works well for various types of textures. Much work has been made in addressing what texture features are important in visual texture perception. Several texture features were proposed for texture analysis and synthesis [13] [14] [15]. However, these methods are biased for specific texture types [16]. Some works were carried out on extracting texture features in accordance with human's visual performance for texture classification, retrieval and discrimination. Tamura et al [17] was the first attempt to extract perceptual texture features using computational representations. Since then, a number of similar works were carried out on studying perceptual texture features using computational approaches [18] [19]. However, these proposed perceptual features produced different results on different texture databases for texture similarity measurement. Gabor wavelets [20] and Local Binary Patterns(LBP) [21] are widely used as the most successful local appearance descriptors in texture classification. It has been shown that the performance can be significantly improved by fusing these two features [22]. Recently, features produced by Principal Component Analysis Network (PCANet) [23] sets new records in texture classification and object detection.

To build perceptual based procedural texture generation system, we have to measure perceptual scales of the procedural textures. But it is impossible to do so for all textures that the procedural models can produce. Therefore, we measured a small subset and employed computational features to regress for the rest. More details will be described in following sections.

For example, visually perceived roughness of random phase fractal noise surfaces are highly related with its parameters [24].

Although computational texture analysis has achieved fine results over recent decades, the approaches usually operate on the basis of a priori notions of texture, not necessarily tied to human experience [25]. Cognitive styles theories suggest that we divide into visual and verbal thinkers [26]. Humans are capable of describing textures by perceptual scales [27]. It is of tremendous advantage to learn - either from human-provided labels, or from investigation of the underlying biological mechanisms.

III. Psychophysical-based procedural texture generation

A. Image pool Generation

We collected 22 procedural texture models, which are Forest Fire Model(FFM), Surface Tension Model(STM), Ex-

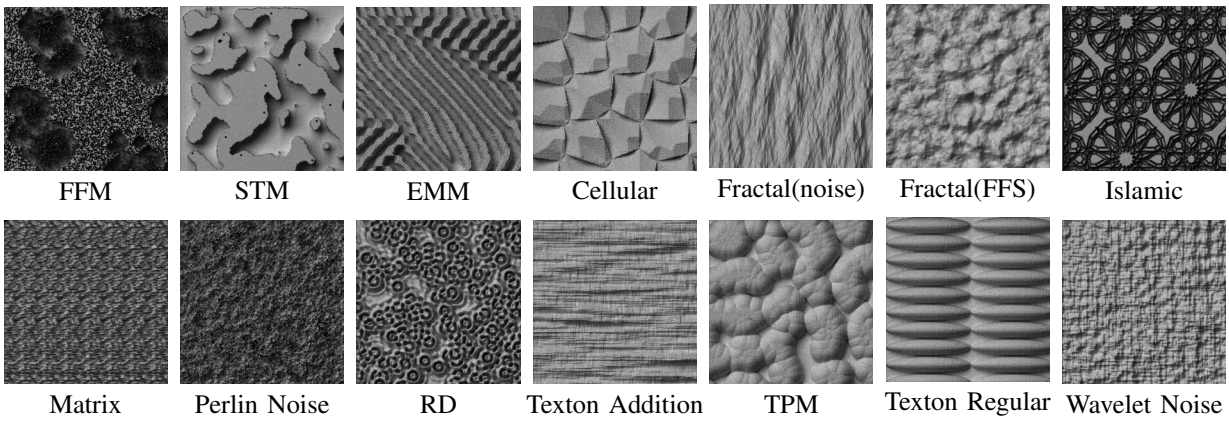


Fig. 3: Samples in our procedural texture pool.

citable Media Model(EMM), Cellular, Folding Texton, Folding Cellular, Folding Fractal, Folding Perlin, Fractal($1/f^{-\beta}$ noise, and Fourier Spectral Synthesis(FSS)), Islamic Patterns, Matrix Transformation(MT), Perlin Noise, Reaction Diffusion(RD), Texton Addition, Texton Probability Map(TPM), Texton Random Grid(TRG), Texton Random Walk(TRW), Texton Regular (TR) and Wavelet Noise(WN). For each model, we generated a large number of textures of size $256 * 256$ pixels by densely sampling model parameters so as to cover noticeable different appearances. The number of textures sampled from each model varied, which depended on the range of texture appearance the models can generate. These textures were used as surface height maps, which were rendered using Lambertian reflectance under the same area lighting conditions. We had totally 8800 textures. Fig. 3 shows some samples in the dataset.

B. Psychophysical Experiments

To produce psychophysical-based textures, we have to first measure perceptual scales of each texture in the pool. However, the number of textures is too huge as the workload in psychophysical experiments. We proposed a comprised method that we measured a small subset of textures psychophysically and estimated the perceptual properties of the rest textures by predictions using computational features and machine learning techniques.

1) Stimuli: Each texture was printed on a $4 * 4$ inch photographic paper with a resolution of 128 pixels per inch. The advantage of using printed photographs in the experiments is that subjects are able to look through more images that showing on a screen display and it is more favorable for subjects to make judgments for the particular experiment method. Totally, we chose 450 samples that comes from the 22 procedural models.

2) Procedure: We used the 12 perceptual texture dimensions proposed in [27], which are contrast, repetition, granularity, randomness, roughness, feature density, directionality, structural complexity, coarseness, regularity, locally orientation and uniformity. Fifty-eight graduate students from a variety of majors (25 female and 33 male, aged between 23 and 35) participated in the psychophysical experiments. All of the participants are naïve to the purpose of the experiment. The

Fig. 4: Psychophysical experiments.

450 samples were divided into nine groups. The order of the images in each group was randomized. The observers were asked to go through all the images in a group and rated the perceptual scales of each texture on a 9-point Likert scales. Each subject was assigned two or three groups of samples. Fig.4 shows observers participate in the experiment.

3) Results: The experiment results were normalized (min-max normalization) and averaged across observers. A sample-feature matrix R was constructed, where $R_{i,j}$ represented the jth perceptual features for sample i. Note that we only obtained perceptual scales of a small set of our texture pool (450 out of 8800). For the rest textures, we employ computational texture features and machine learning techniques to estimate their perceptual scales automatically. Tab. I is average perceptual scales of samples corresponding to the images in Fig. 3.

C. Computational Features Extraction

To estimate the perceptual scales of a new procedural texture as human rated, we turned to computational methods for solutions. Several methods for texture feature extraction have shown their success in classification task, for example, the LBP and Gabor features are the most popular ones [21] [22]. In recent years, features that automatically learned from examples other than hand-crafted are setting new records in texture classification and object detection, such as the PCANet [23]. We compared these methods in using our data and chose the one with best performance.

1) LBP features: The LBP operator forms labels for the image pixels by thresholding the 3×3 neighborhood of

TABLE I: Average perceptual scales of selected samples.

Number	contrast	repetitive	granular	random	rough	density	direction	complexity	coarse	regular	oriented	uniform
FFM	5.3	2.4	4.65	7	5.75	6.55	1.75	4.85	5.95	2.85	2.4	3.4
STM	6.7	3.1	2.15	7.05	5.6	3.7	2	6.1	4.85	2.75	2.6	3.4
EMM	5.75	4.5	1.85	4.65	5.95	5.3	4.75	4.05	5	4.3	6	4.75
Cellular	6.2	6.05	4.35	3.15	5.2	4.75	4.9	4.35	4.7	5.9	5.95	6
Fractal(noise)	4.95	3.95	1.55	4.75	4.65	3.4	7.1	2.75	4.9	4.4	6.85	5.05
Fractal(FFS)	4.75	3.45	4.05	6.35	6.8	5.95	2.8	4.25	5.5	3.1	2.7	5.25
Islamic	6.8	7	4.8	3	5.5	4.25	4.05	6.95	3.75	6.8	4.6	5.45
Matrix	3.45	7.5	4.2	2.95	4.7	7.75	6.85	4.2	5.4	7.2	7	7.5
Perlin Noise	3.4	3.1	3.45	5.95	6.75	7.05	1.8	3.6	5.7	2	1.7	5.15
RD	5.45	4.85	6.1	6.25	6.8	6.15	2.4	4.75	4.55	3.35	2.85	4.65
Texton Addition	4.85	4.85	2.3	4.2	5.65	5.3	7.45	3.3	4.4	5.1	7.3	5.25
TPM	4.95	4.9	4.6	5.3	5	4.55	3	5.3	4.25	3.4	2.95	4.05
Texton Regular	6	8.35	4.15	1.95	4.55	5.55	8.15	2.8	5.3	8.35	8.75	8.2
Wavelet Noise	4.7	3.9	3	5.25	5.5	6.35	5.25	5.1	5.4	3.85	4.85	5

each pixel with the center value. The histogram of labels is used as the texture descriptor. Extended LBP operator uses neighborhoods of different sizes. We select the $(8, 2)$ rotation-invariant LBP descriptor, which means 8 neighboring pixels on a circle of radius 2. The 36 bins histogram was used as the description of texture image.

2) Gabor features: Gabor features are Gaussians modulated by complex sinusoids. A 2-D Gabor function $g(x, y)$ and its Fourier transform $G(u, v)$ can be defined as Eq.(1) and Eq.(2):

$$g(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp[-\frac{1}{2}(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2})+2\pi jWx] \quad (1)$$

$$G(u,v) = \exp -\frac{1}{2}[\frac{(u-W)^2}{\sigma_\mu^2}+\frac{\nu^2}{\sigma_\upsilon^2}] \quad (2)$$

where $\sigma_\mu = 1/(2pi\sigma_x)$ and $\sigma_\nu = 1/(2pi\sigma_y)$ define the spatial extent and frequency bandwidth of the Gabor filter. $W = 2\pi\mu$, μ refers to the frequency and W denotes the width of the function. It has been pointed out that the design of Gabor filter should ensure that the adjacent half-peak magnitude contours of the filter responses in the frequency domain should touch each other [20]. In our experiments, 48 Gabor wavelets coefficients were constructed as Gabor feature vectors (four scales and six orientations).

3) PCANet Features: Other than the two hand-crafted features described above, we also tested a feature that automatically learned from samples – PCANet [23]. We gave N input training images of size $m \times n$, and assumed that the patch size (or 2D filter size) is $k_1 \times k_2$ at all stages. And only the PCA filters need to be learned from the input images. The inputs are first convoluted with PCA filters to produce a set of feature maps, they are then fed into the next stage as input. We converts the outputs generated in the second stage back into a single integer-valued image. For each of integer-valued images, we partition it into B blocks. We compute the histogram of the decimal values in each block, and concatenate all the B histograms into one vector. After this encoding process, the feature of the input image becomes the set of block-wise histograms. In our experiments, the size of our samples is 256×256. We set the parameters of PCANet to the filters $k_1 = k_2 = 7$, the number of filters $L_1 = L_2 = 8$, and 64×64 block size. Therefore, a texture can be described by a 32768-bin histogram . In order to speed up the calculation we hold 95% accumulation contribution rate by PCA, which resulted in a 1206-bin histogram.

D. Predicting perceptual scales using Computational features

Each texture used in the psychophysical experiment has two representations: computational feature vector and perceptual feature vector. Since the perceptual data are from 9 point Likert scales, we used these scales as class labels. Therefore, predicting perceived properties of those textures not shown in the psychophysical experiment becomes a problem of classification. The task now is to classify textures to 9 clusters for each of the 12 perceptual features according to their computational features. The SVM was employed, where the computational feature vectors are used as the training data and the Likert scales were used as the class labels. A total of 12 SVM classifiers were trained and each classifier was responsible for one perceptual feature prediction.

1) Predicting perceptual scales: We used the LIBSVM implementation of the Support Vector Machines (SVMs) [28][29]. The kernel function RBF was selected for its non-linear property and low number of parameters. We employed the PSO algorithm to search for the optimal SVM parameter values. Leave-one-out cross-validation (LOOCV) was used to estimate the accuracy of the predictive model. Fig. 5 shows the framework of our predicting work.

We compared the prediction accuracies of the four computational features described above. Tab. II summarizes the results. It is obvious that the PCANet features shows the best accuracy, following by fusion Gabor&LBP, Gabor and LBP. The results suggest that we will choose PCANet as the computational feature to estimate perceived properties.

For the textures other than those used in psychophysical experiment, we extracted their computational features using PCANet and estimated their perceptual scales by the 12 SVM classifiers separately. Therefore, we obtained perceptual scales for all 8800 samples in the texture pool. It should also be noted that when new textures are added into the texture pool, we can easily predict their perceptual scales using the proposed approach.

E. Design of the interactive system

As a prototype system, we show our implementation using two perceptual properties – roughness and directionality. Other properties can be manipulated similarly.

One issue needs to be pointed out is that the 12 perceptual features are not independent, which means that some features

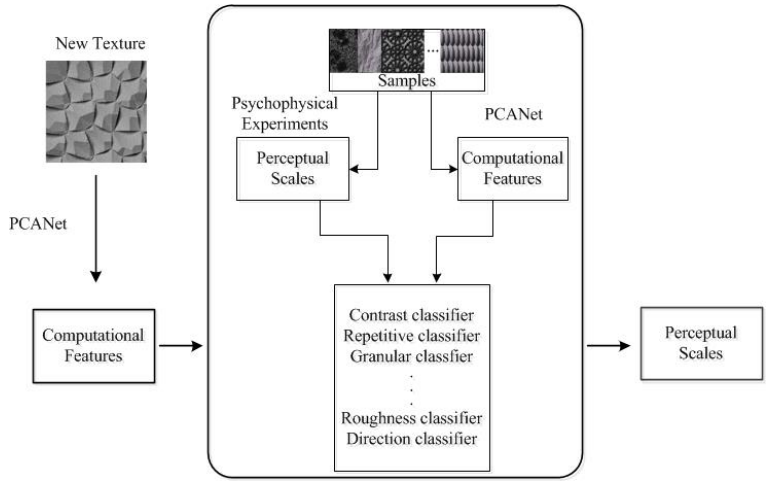


Fig. 5: The framework of predicting perceptual scales using computational features.

TABLE II: Comparisons different computational features for perceptual scales using average leave-one-out recognition rates(%).

	perceptual features											
Method	contrast	repetitive	granular	random	rough	density	direction	complexity	coarse	regular	oriented	uniform
LBP	87.99	85.70	82.11	87.58	87.09	87.01	87.66	84.86	88.07	85.29	86.93	86.52
Gabor	91.91	86.44	85.95	87.99	87.83	90.28	90.93	86.44	88.89	88.15	90.11	89.46
Gabor+LBP	92.72	89.38	87.99	90.77	91.42	92.16	93.14	89.95	90.93	91.42	92.97	91.09
PCANet	**99.78**	**99.67**	**99.22**	**99.22**	**99.78**	**99.56**	**99.78**	**99.44**	**99.44**	**99.67**	**99.78**	**99.22**

are correlated [27]. For example, our psychophysical experiment shows that the correlation coefficient between the consistency of directionality and local directionality is 0.97, and the correlation coefficient between roughness and randomness is -0.55. Therefore, when we change the scale of one perceptual feature, other features should not be held constant. We used different relaxations for different features according to their correlation coefficients with the feature being changed. This guarantees that there will be a small number of candidates.

The initial texture is displayed on the screen. Users can make finger gestures on the touch screen if they want to alter textures in terms of some dimensions. Take the roughness as an example. Users make finger gesture "zoom". The system reads the magnitude of the gesture and transfers it into corresponding changes for roughness. There will be several candidate textures, and the one that is nearest in the computational feature space will be shown. The directionality property can be similarly manipulated. The two-point sliding was used, where the sliding direction is to define the orientation of the directionality and the sliding distance defines the magnitude of the changes. Fig. 6 shows some testing results of our system.

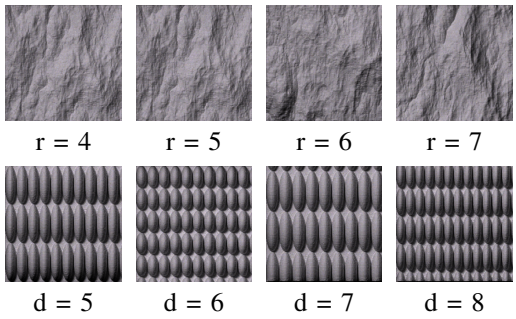


Fig. 6: The output of our system. Number under the image is the perceived roughness(r) / directionality(d) of the texture.

IV. Conclusion

We introduced a large dataset of 8800 textures generated by 22 procedural texture models and applied it to study the problem of generating images by perceptual features. Looking for the perceptual scales of textures , we developed 12 classification models to predict them and we regarded roughness and directionality as the reference to generate textures, other perceptual scales can be operated as the same. Only need one or more touches on the screen, the users will obtain an ideal texture.

Acknowledgment

This Work Was Supported By National Natural Science Foundation Of China (NSFC) (No.61271405, 61401413) , The PH.D. Program Foundation Of Ministry Of Education Of China (No.20120132110018), International Science & Technology Cooperation Program of China (ISTC)(NO.2014DFA1041) and China Postdoctoral Science Foundation (No.2014M551963).

References

[1] Steven Worley. A cellular texture basis function. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 291–294. ACM, 1996.

[2] Ken Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3):287–296, 1985.

[3] Yiming Liu, Jiaping Wang, Su Xue, Xin Tong, Sing Bing Kang, and Baining Guo. Texture splicing. In *Computer Graphics Forum*, volume 28, pages 1907–1915. Wiley Online Library, 2009.

[4] Robert L Cook and Tony DeRose. Wavelet noise. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 803–811. ACM, 2005.

[5] Heinz-Otto Peitgen, Dietmar Saupe, Michael Fielding Barnsley, Yuval Fisher, and Michael McGuire. *The science of fractal images*. Springer New York etc., 1988.

[6] Andrew Adamatzky, Andrew Wuensche, and Benjamin De Lacy Costello. Glider-based computing in reaction-diffusion hexagonal cellular automata. *Chaos, Solitons & Fractals*, 27(2):287–295, 2006.

[7] Lionel G Harrison. *Kinetic theory of living pattern*, volume 28. Cambridge University Press, 1993.

[8] Ares Lagae, Sylvain Lefebvre, Rob Cook, Tony DeRose, George Drettakis, David S Ebert, JP Lewis, Ken Perlin, and Matthias Zwicker. A survey of procedural noise functions. In *Computer Graphics Forum*, volume 29, pages 2579–2600. Wiley Online Library, 2010.

[9] Jarke J. van Wijk. Spot noise texture synthesis for data visualization. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '91, pages 309–318, New York, NY, USA, 1991. ACM.

[10] John P Lewis. Generalized stochastic subdivision. *ACM Transactions on Graphics (TOG)*, 6(3):167–190, 1987.

[11] Jong-Chul Yoon and In-Kwon Lee. Stable and controllable noise. *Graphical Models*, 70(5):105–115, 2008.

[12] Ares Lagae, Peter Vangorp, Toon Lenaerts, and Philip Dutré. Procedural isotropic stochastic textures by example. *Computers & Graphics*, 34(4):312–321, 2010.

[13] David J Heeger and James R Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238. ACM, 1995.

[14] Robert M Haralick. Statistical and structural approaches to texture. *Proceedings of the IEEE*, 67(5):786–804, 1979.

[15] Wei-Ying Ma and Bangalore S Manjunath. Texture features and learning similarity. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR'96, 1996 IEEE Computer Society Conference on*, pages 425–430. IEEE, 1996.

[16] Benjamin J Balas. Texture synthesis and perception: Using computational models to study texture representations in the human visual system. *Vision research*, 46(3):299–309, 2006.

[17] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *Systems, Man and Cybernetics, IEEE Transactions on*, 8(6):460–473, 1978.

[18] Noureddine Abbadeni. Computational perceptual features for texture representation and retrieval. *Image Processing, IEEE Transactions on*, 20(1):236–246, 2011.

[19] Kenji Fujii, Shinofu Sugi, and Yoichi Ando. Textural properties corresponding to visual perception based on the correlation mechanism in the visual system. *Psychological Research*, 67(3):197–208, 2003.

[20] B.S. Manjunath and W.Y. Ma. Texture features for browsing and retrieval of image data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(8):837–842, Aug 1996.

[21] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, Jul 2002.

[22] Quan-You Zhao, Ao-Chang Pan, Jan-Jia Pan, and Yuan-Yan Tang. Facial expression recognition based on fusion of gabor and lbp features. In *Wavelet Analysis and Pattern Recognition, 2008. ICWAPR'08. International Conference on*, volume 1, pages 362–367. IEEE, 2008.

[23] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification? *arXiv preprint arXiv:1404.3606*, 2014.

[24] Padilla. *Mathematical models for perceived roughness of three-dimensional surface textures*. PhD thesis, Heriot-Watt University, 2008.

[25] Tim Matthews, Mark S Nixon, and Mahesan Niranjan. Enriching texture analysis with semantic data. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1248–1255. IEEE, 2013.

[26] David A. Robb, Stefano Padilla, Britta Kalkreuter, and Mike J. Chantler. Crowdsourced feedback with imagery rather than text: Would designers use it? In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1355–1364, New York, NY, USA, 2015. ACM.

[27] A Ravishankar Rao and Gerald L Lohse. Towards a texture naming system: identifying relevant dimensions of texture. In *Visualization, 1993. Visualization'93, Proceedings., IEEE Conference on*, pages 220–227. IEEE, 1993.

[28] Y Li Faruto. Libsvm-farutoultimateversion-a toolbox with implements for support vector machines based on libsvm. *Software available at http://www. ilovematlab. cn*, 2009.

[29] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.