Cartoon filter via adaptive abstraction

Davide Guastella[a], Cesare Valenti[a,*]

[a] *Dipartimento di Matematica e Informatica, Università degli Studi di Palermo, Italy*

Abstract

Abstraction in computer graphics defines a procedure that discriminates the essential information that is worth keeping. Usually details, that correspond to higher frequency components, allow to distinguish otherwise similar images. Vice versa, low frequencies are related to the main information, which are larger structures. Contours themselves may also be identified by high frequencies and separate each pictured component. The underlying idea of the proposed algorithm consists in identifying these edges, by a redundant wavelet transform, and in blurring the inner areas of the components, by an adaptive circular median filter. In spite of its implementation simplicity, our unsupervised methodology provides results similar to those obtained by more complex techniques already described in the literature.

Keywords: Multiresolution abstraction, Cartoon filter, Redundant wavelets, Circular median filter

*Corresponding author at: Dipartimento di Matematica e Informatica, via Archirafi 34, 90123 Palermo, Italy. Tel: +39 091 23891076.

Email address: cesare.valenti@unipa.it (Cesare Valenti)

Preprint submitted to Journal of Visual Communication and Image Representation

1. Introduction

Abstraction algorithms return non-realistic scenes, still keeping their main information content. These techniques are widely used in computer graphics with applications in the videogame (e.g. "Prince of Persia", 2008) and in the film (e.g. "A scanner darkly", 2006) industries. In the former case, the so-called cel-shading technique [1] reduces in real time the amount of data that describe the three-dimensional scene with a high quality level. The graphic tool used to model the objects let's the user arrange position, texture and properties of the virtual environment, including lighting and the camera location. This allows to quickly simulate a cartoon-like effect without the actual need for a hard intervention by an artist. In the former case, the interpolated rotoscoping techniques [2] return frames which show a typical effect due to handmade drawings.

Most abstraction algorithms apply a variety of methods to reduce the amount of details within the areas that correspond to the objects in the image, while maintaining the salient information in proximity of their more highlighted contours. Interesting reviews of the main stylization methods can be found in [3, 4].

Segmentation-based techniques are usually avoided due to over segmentation and require a continuous supervision to mitigate eventual errors [3]. Anyhow, these segmentation-based approaches are unsuitable on video streams, because of their excessive computational time. Segmentation techniques based on clustering, like superpixel [5] or normalized cuts [6] were not taken into account because of their dependency on minimization functions that involve user-defined parameters (e.g. the cluster size or the affinity function) which depend on the given image and therefore are unsuitable on video streams.

An iterative technique based on the minimization of the gradient information is presented in [7] to globally maintain and possibly enhance the most prominent set of edges by increasing steepness of transition while not affecting the overall acutance. An approximate solution is achieved, because finding the global optimum is a NP-hard problem.

The bilateral filter [8] is based on blurring through appropriate weighted Gaussian functions that preserve as much as possible the areas close to the edges. Again, the main drawback is the computational complexity caused by the non-separability of the proposed functions [3] although recent implementations result quite fast [9]. A framework of different methods was described in [10] where the bilateral filter is applied to further smooth the interior of uniform regions. The subsequent difference of Gaussians enhances the contours and a particular quantization technique reduces the color space palette of the final image.

A pipeline for image abstraction that exploits color quantization, bilateral filtering and difference of Gaussians was introduced in [3, 11]. The bilateral filtering was modified conveniently to set the eccentricity of the smoothing window according to the local gradient. Also, a particular thresholding function was employed to perform color quantization.

The method developed by Litwinowicz [12] is widely used in the cinematographic field. It computes the optical flow between two consecutive frames to determine the main direction to orient a sequence of simple brush strokes. Input parameters like thickness, length, hardness and color of the brush require an accurate control by the artist and they greatly influence the quality of the abstract representation.

Implementations based on the Kuwahara filter [13, 14] divide the image into partially overlapped square blocks and assign values to the their central pixels according to the average brightness of the blocks, weighted by a Gaussian convolution [15]. These approaches are generally fast but return images that suffer due to block artifacts. The variant presented in [16] considers circular sectors instead of square blocks and arranges the eccentricity [13] according to an anisotropy function calculated through the eigenvalues of the Harris matrix [17]. This increased complexity makes it possible to get better results even with fixed size windows. A multi-scale approach which uses a pyramid to guide both the shape and the size of the windows was reported in [18] to reduce details with different dimensions.

This paper describes a simple yet effective abstraction methodology made by a pipeline that can be computed in a fast way (see Fig. 1). The following sections describe the main steps which are luminance channel extraction, edge detection by the 'à trous' wavelet, distance transform by mathematical morphology and adaptive smoothing filter. A comparison with state-of-the-art algorithms is reported together with conclusions and future works.

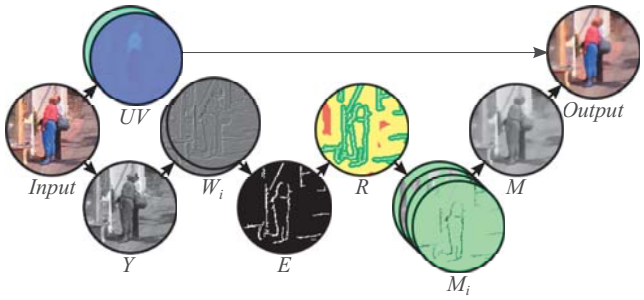

Figure 1: Sketch of the main steps of our methodology.

2. Luminance conversion

In order to obtain a valid representation of the luminance information, we compared various techniques as suggested in [19]. Indeed, we verified experimentally that all these methods are qualitatively equivalent to each other for the final abstraction (Fig. 8). Therefore, we decided to convert colour to grayscale images by extracting the standard luminance information. Given the RGB color representation of a pixel $\mathbf{p}$, its YUV counterpart is obtained by:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix}_{\mathbf{p}} = \begin{bmatrix} 0.30 & 0.59 & 0.11 \\ -0.15 & -0.29 & 0.44 \\ 0.62 & -0.52 & -0.10 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{\mathbf{p}} \tag{1}$$

where Y takes into account the non-uniform human perception of primary colors (Fig. 2).

Figure 2: Input image and its YUV channels (U and V emphasized for better display purposes).

3. Edges by wavelets

The wavelet analysis is a powerful mathematical tool for representing and processing data, to enhance or to suppress components with specific frequencies (i.e. size and shape). We applied the so-called *à trous* algorithm [20] because it is very fast and retains the maximum resolution (i.e. the output image does not undergo decimation unlike the usual multiresolution analysis [21]).

We perform a *filterbank* (i.e. succession of low-pass and high-pass filters) on the luminance Y channel:

$$I_0(\mathbf{p}) = Y(\mathbf{p}), \quad I_i(\mathbf{p}) = I_{i-1}(\mathbf{p}) \circledast \ell_i \tag{2}$$

where the non-zero elements of the low-pass filter ℓ_i are given by the isotropic kernel ℓ [22–24]:

$$\ell = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \quad \ell_i(2^{i-1}\mathbf{q}) = \ell(\mathbf{q}) \tag{3}$$

and the pixel $\mathbf{q}$ spans the 3×3 neighborhood centered in $\mathbf{p}$. The high-pass filter is defined as the difference between two consecutive spatial scales, which provides the wavelet coefficients:

$$W_i(\mathbf{p}) = I_{i-1}(\mathbf{p}) - I_i(\mathbf{p}). \tag{4}$$

This wavelet algorithm takes a constant time when computing a series of W_i due to the advantage that the number of non-zero elements in ℓ_i is always equal to nine (Fig. 3) and the convolution can be speeded up by considering the separability property of ℓ:

$$\ell = \frac{1}{16}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}. \tag{5}$$

Small objects are enhanced in the first planes W_1 and W_2. A threshold is applied to binarize these planes and the set of contours E is obtained by a subsequent pixel-wise logical disjunction of the binarized images:

$$E(\mathbf{p}) = \bigvee_{i=1,2} \{W_i(\mathbf{p}) > \mu_i - 1.5\times\sigma_i\} \tag{6}$$

where μ_i and σ_i are respectively the mean and standard deviation of the floating-point values of the plane W_i. Sometimes, the resulting edge includes very small components caused by noise or negligible details in the input image. The usual mathematical morphology opening φ with structuring element S defined by the discrete disk of radius equal to one pixel improves the final result (Fig. 4) [25]:

$$E^{\varphi}(\mathbf{p}) = \delta(\varepsilon(E(\mathbf{p}))) \tag{7}$$

where

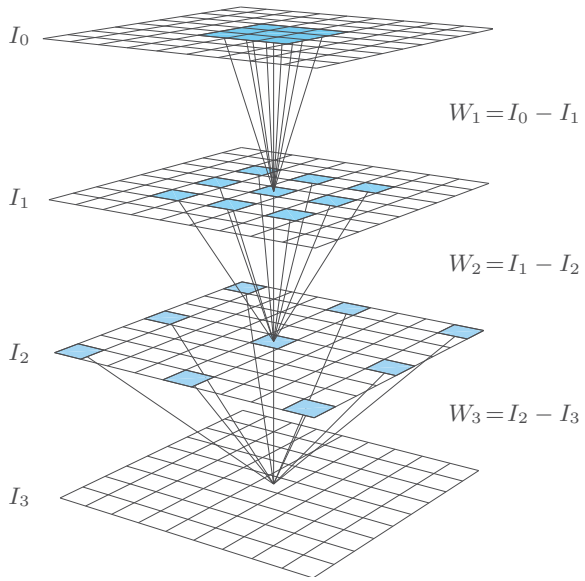

Figure 3: Bigger and bigger (light-blue) versions of ℓ.

$$\varepsilon(E(\mathbf{p})) = \min_{\mathbf{q}\in S} E(\mathbf{p}+\mathbf{q}), \quad \delta(E(\mathbf{p})) = \max_{\mathbf{q}\in S} E(\mathbf{p}+\mathbf{q}). \tag{8}$$

4. Distance transform

The median filter we use to smooth the image is quite expensive in terms of computational time with respect to the remaining part of our abstraction algorithm. To limit the size of its window we compute the minimum discrete Euclidean distance between each pixel and the nearest contour just obtained by the wavelet transform. In particular, we make use of the algorithm due to [26] and further investigated in [27].

$$D(\mathbf{p}) = \min\{\|\mathbf{p}-\mathbf{q}\| : E^{\varphi}(\mathbf{q}) = 1\}. \tag{9}$$

This map gives the size of the smoothing window, for each pixel of the input image: the closer to the edge, the smaller will be the window in order to keep details. Vice versa, zones far from the edges are smoothed much more. This corresponds in some way to the adaptive approaches proposed in [13, 16].

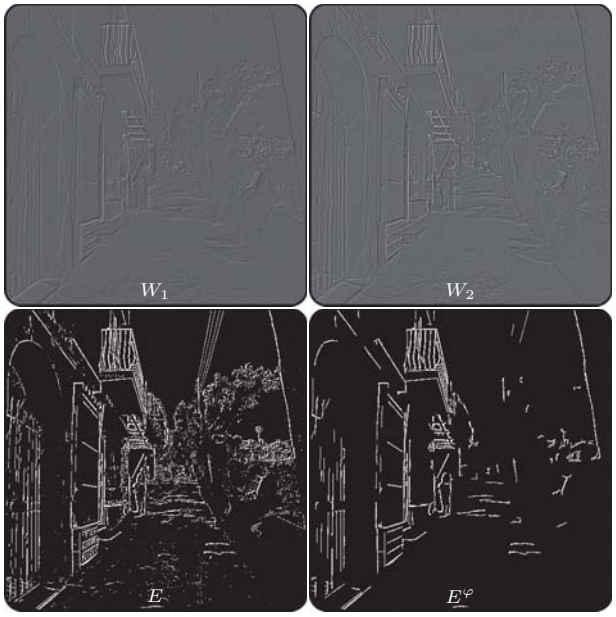

Figure 4: Image representation of the first two wavelet planes of Fig. 2b and their binary composition, before and after the opening.

To reduce the size of the median window, we applied various decay formulae on the distances D. The best results correspond to the rounded cube root of the distance transform thus to maintain pronounced edges and to keep most details of the structures, while blurring the overall input image (Fig. 5).

$$R(\mathbf{p}) = \left[\sqrt[3]{D(\mathbf{p})} \right]. \tag{10}$$

5. Adaptive median

To smooth the images, we realized a median filter which achieves effective results from an abstraction viewpoint. We preferred to use a circular window because it is isotropic although it requires more processing time with respect to particular implementations which rely on square windows [28].

Given the luminance image Y and the distance map R of the edges E^{φ}, we compute the median $M(\mathbf{p})$ among the values within a circular window centered in $\mathbf{p}$ with radius $R(\mathbf{p})$ (Fig. 6):

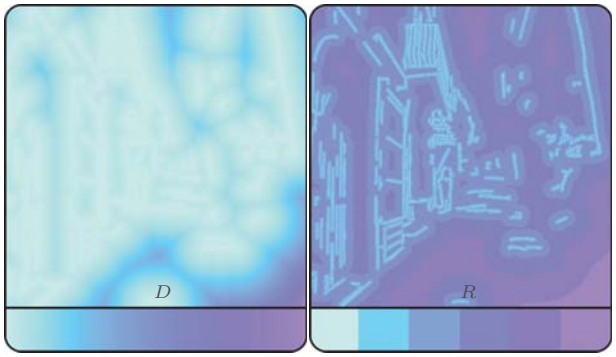

Figure 5: Distance transform map D of the binary edges E^{φ} of Fig. 4 and its rounded cube root R which greatly reduces the number of different distances.

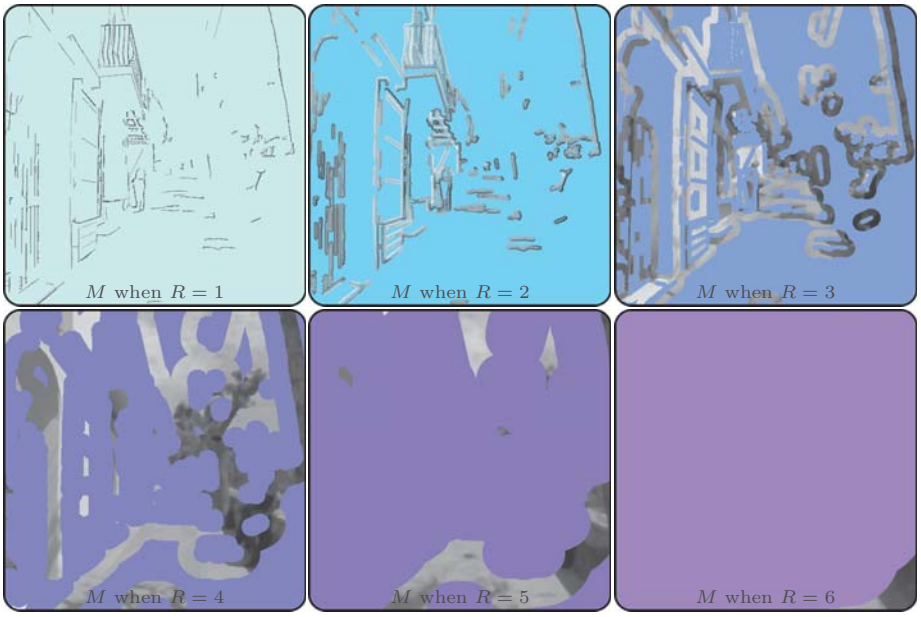

Figure 6: The map R of Fig. 5 provides the radii for the circular median filter in the various zones of the input image. The background color represents "don't care areas" and the radius.

Fast parallel implementations, based on 3-way quick select algorithms [29] can be applied: they rely on the fact that it is sufficient to delete and insert a small amount of values in an already ordered set, to be updated. Instead, we used a serial approach based on histogram evaluation which resembles a non-comparative integer sorting algorithm: to compute the median within the

circular window, it spans the histogram relative to that window while its cumulative sum is less then half of the area of the window itself (Fig. 7). In the worst case, computing and spanning the histogram require a number of direct memory accesses equal to the area and to the number of luminance levels, respectively. An analogous algorithm [28] was proved to have a constant time complexity.

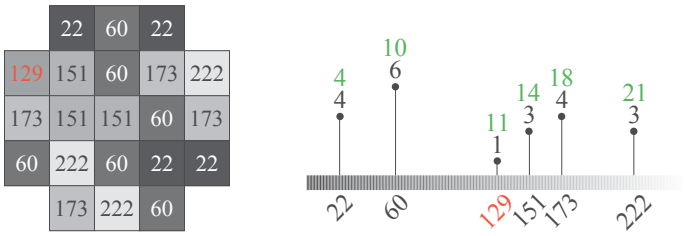

Figure 7: The median value (red) is the bin that halves the neighborhood area in the histogram cumulative sum (green). Here, it is 129 from 21 values because $R=2$.

The adaptive median images of Fig. 6 are composed to obtain the final abstraction image (Fig. 8) while the color abstraction image is obtained by assembling the grayscale abstraction together with the original chrominance channels (if any) of Fig. 2.

Figure 8: Abstraction images obtained from different luminance information approaches show very few differences.

6. Multi-scale adaptive median

This median filter remains the bottleneck of our abstraction methodology, therefore we developed also a multi-scale approach based on a pyramid built in

a fine-to-coarse manner by the Lanczos downsampling interpolation [30]. We verified that more common interpolations like nearest neighborhood and bilinear introduce artefacts in the resized image (Fig. 9).

Figure 9: Composition of the chrominance channels U and V of Fig. 2 with the luminance Y multi-scale abstraction result, considering three interpolation algorithms.

Just a few pixels require a small radius for the median window (Fig. 5), thus we limit the multi-scale algorithm in the case of radii greater than 3. Instead of applying directly on $\mathbf{p}$ the circular median filter with radius $R(\mathbf{p})$, we consider the radius 3 on the image proportionally reduced by the coefficient $(2\times 3+1)/(2\times R(\mathbf{p})+1)$ (Fig. 10). Moreover, it must be considered that this downsampling step is performed once and that it lets many adjacent pixels in the original image share the same circular window in the reduced image: this speeds the computation because we calculate once their median value.

7. Experimental results

First applications of the Kuwahara filter concerned noise reduction while preserving the edges of the components in medical images [14]. To measure the robustness of the considered methods, we studied their outputs in the case of different amounts of random Gaussian or salt&pepper noise. Fig. 11 shows the average response with respect to the original clean images of Figs. 13–16, by using the structural similarity measure SSIM [31, 32] and the standard pixel-wise PSNR. We experimentally observed that our algorithm together with 'anisotropic Kuwahara' and 'Papari et al.' is able to eliminate the majority of

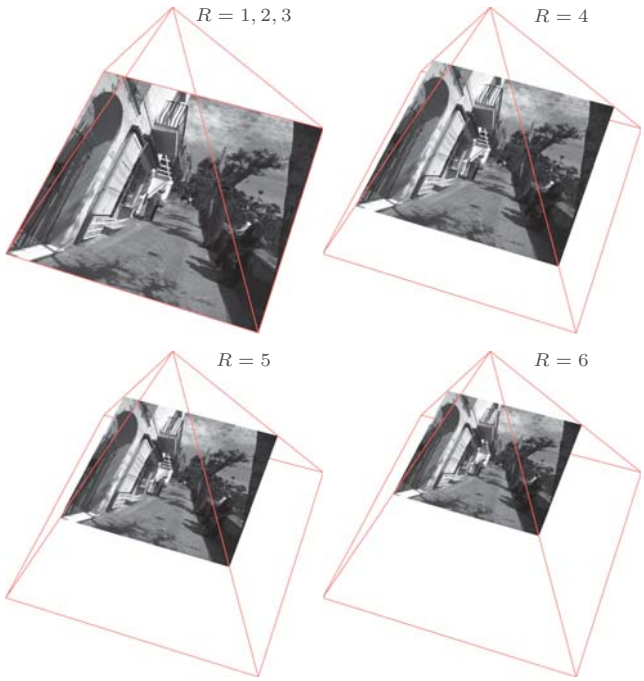

Figure 10: Regardless to the actual radius $R > 3$, the circular median filter is computed with radius equal to 3 on properly reduced versions of the image.

the introduced distortion without degrading the image. Both 'bilateral filter' and 'L_0 gradient minimization' return worse results for salt&pepper noise.

We carried out a comparison of the execution times by the same algorithms, summarized in Fig. 12. For 'Papari et al.' [16] and 'L_0 gradient minimization' [7] we used the original software written by the authors in the MatLab language which is widely used for numerical computations. This language is interpreted, although quite efficient due to just-in-time compilation, vector primitives and parallel extensions (eventually executed by built-in instructions in a user-trasparent way) for GPUs and multi-core CPUs. Regarding the 'bilateral filter' [8] we compiled the corresponding command of the open source graphic library OpenCV for digital image processing [33]. Our methodology was coded in pure C++ language using this library, too. These programs were run five times per image on an entry-level machine equipped with an Intel i3-2120, 8GB

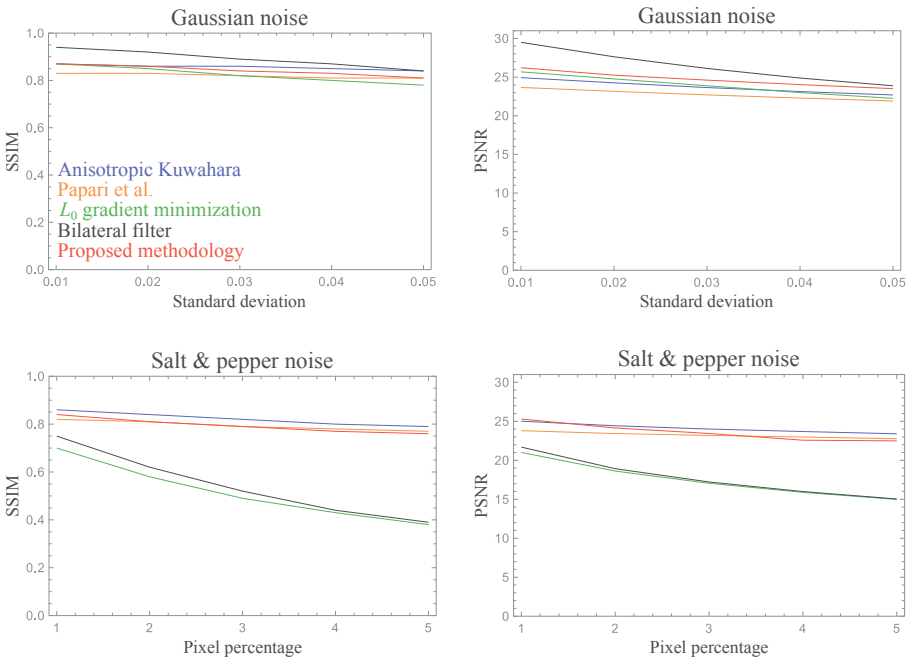

Figure 11: Quality measures versus amount of noise.

RAM and Linux. As of the 'anisotropic Kuwahara' [18] we report the elapsed time obtained by the authors through their optimized GPU implementation on a graphics card equipped with nVidia GTX580 (the rest of the hardware was not specified).

Approximately 0.20 seconds are required by our methodology to process a truecolor image with 512×512 pixels using a single core of the CPU. As future extensions, optimizations like parallelization through GPUs and multi-core CPUs to compute both the wavelet planes and the circular median filter can still be made, thus a significant speed-up can be expected.

8. Conclusions

The proposed algorithm does not require any intervention by the artist thus facilitating the development of videos. In fact, the wavelet edge detector we described maintains only the relevant parts of the contours, while reducing the presence of irrelevant edges which are usually due to noise. Our methodology

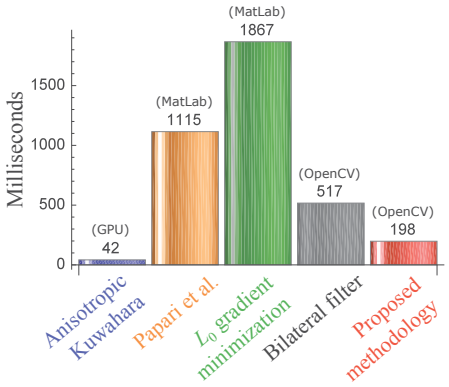

Figure 12: Average elapsed time in milliseconds for 512×512 truecolor images.

operates on the luminance channel of the frames and therefore the appearance of their colors does not change over time, instead of approaches that modify the color palette of each single frame [10]. As an optional refinement the user could overlap on our smoothed image the edges obtained by ad hoc techniques [34, 35].

Our methodology meets the following essential rules for artistic stylization algorithms, as recommended in [3]: it is not supervised (the artist does not have to provide any parameter value); it does not belong to any particular artistic movement (the methodology allows to process images of any kind and it does not simulate images of a particular school of art like impressionism or cubism); it is simple and innovative with respect to relevant literature.

In general, the quality of the abstraction methods is rather subjective and cannot be evaluated via metrics: to compare the proposed technique and the algorithms currently considered as state-of-the-art, we carried out experiments on the images kindly provided by the authors of those techniques [13] and by using freely distributed implementations with default values for the input parameters. Even if the considered approaches produce different outputs, it is anyway desirable to develop a set of methods to present the artist a variety of abstraction results. Figs. 13–16 show some examples available with demo videos processed by the considered approaches at http://math.unipa.it/cvalenti/adaptiveabstraction.

Figure 13: Qualitative comparison among our methodology and other abstraction techniques.

Figure 14: Qualitative comparison among our methodology and other abstraction techniques.

Figure 15: Qualitative comparison among our methodology and other abstraction techniques.

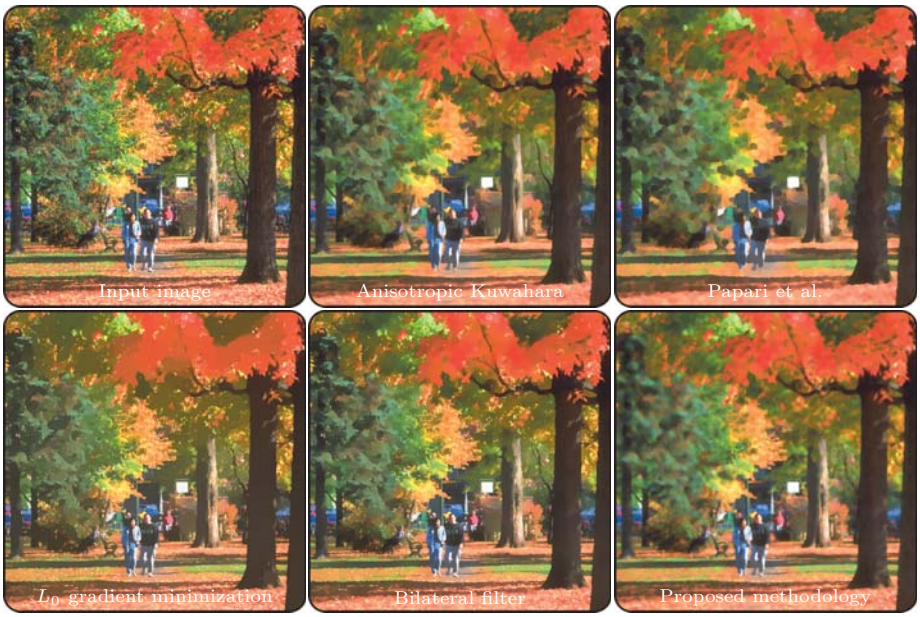

Figure 16: Qualitative comparison among our methodology and other abstraction techniques.

Acknowledgements

We are grateful to Professor Fabrizio Agnello, head of the laboratory of computer graphics of the Department of Architecture in the University of Palermo, for his useful suggestions during the development of the proposed methodology and for testing the correctness of its final results.

References

[1] A. Lake, C. Marshall, M. Harris, M. Blackstein, Stylized rendering techniques for scalable real-time 3D animation, in: International Symposium on Non Photorealistic Animation and Rendering, ACM, 2000, pp. 13–20.

[2] A. Agarwala, A. Hertzmann, D. Salesin, S. Seitz, Keyframe-based tracking for rotoscoping and animation, ACM Trans. Graph. 23 (3) (2004) 584–91.

[3] D. Mould, Image and Video-Based Artistic Stylisation. Computational Imaging and Vision, Vol. 42, Springer, 2013, Ch. Region-Based Abstraction, pp. 125–46.

[4] J. Kyprianidis, J. Collomosse, T. Wang, T. Isenberg, State of the "art": a taxonomy of artistic stylization techniques for images and video, IEEE Transactions on Visualization and Computer Graphics 19 (5) (2013) 866–85.

[5] X. Tian, L. Jiao, L. Yi, K. Guo, X. Zhang, The image segmentation based on optimized spatial feature of superpixel, Journal of Visual Communication and Image Representation 26 (2015) 146–60.

[6] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (8) (2000) 888–905.

[7] L. Xu, C. Lu, Y. Xu, J. Jia, Image smoothing via $L0$ gradient minimization, ACM Transactions on Graphics 30 (6) (2011), additional material at http://www.cse.cuhk.edu.hk/~leojia/projects/L0smoothing.

[8] C. Tomasi, R. Manduchi, Bilateral filtering for gray and color images, in: Proceedings of the IEEE International Conference on Computer Vision, 1998, pp. 839–46.

[9] K. Chaudhury, Acceleration of the shiftable O(1) algorithm for bilateral filtering and nonlocal means, IEEE Transactions on Image Processing 22 (4) (2013) 1291–300.

[10] H. Winnemöller, S. Olsen, B. Gooch, Real-time video abstraction, ACM Transactions on Graphics 25 (3) (2006) 1221–6.

[11] J. Kyprianidis, J. Döllner, Image abstraction by structure adaptive filtering, in: Proc. Eurographics UK Theory and Practice of Computer Graphics, 2008, pp. 51—-58.

[12] P. Litwinowicz, Processing images and video for an impressionist effect, in: Proceedings of the ACM SIGGRAPH Conference on Computer Graphics, ACM, 1997, pp. 407–14.

[13] J. Kyprianidis, H. Kang, J. Döllner, Image and video abstraction by anisotropic Kuwahara filtering, Computer Graphics Forum 28 (7) (2009) 1955–63, additional material at http://www.kyprianidis.com/p/pg2009/gallery1.

[14] M. Kuwahara, K. Hachimura, S. Eiho, M. Kinoshita, Digital Processing of Biomedical Images, Springer, 1976, Ch. Processing of RI-Angiocardiographic Images, pp. 187–202.

[15] J. Shen, Y. He, Depth-aware video abstraction, in: 4th Pacific-Rim Symposium on Image and Video Technology, 2010, pp. 475–80.

[16] G. Papari, N. Petkov, P. Campisi, Artistic edge and corner enhancing smoothing, IEEE Transactions on Image Processing 16 (10) (2007) 2449–62, additional material at http://www.cs.rug.nl/˜imaging/artisticsmoothing.

[17] F. Bellavia, D. Tegolo, C. Valenti, Improving Harris corner selection strategy, IET Computer Vision 5 (2) (2011) 87–96.

[18] J. Kyprianidis, Image and video abstraction by multi-scale anisotropic Kuwahara filtering, in: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering, ACM, 2011, pp. 55–63, additional material at https://code.google.com/p/gpuakf.

[19] C. Kanan, G. Cottrell, Color-to-grayscale: does the method matter in image recognition?, PLoS ONE 7 (1) (2012).

[20] M. Shensa, The discrete wavelet transform: wedding the à trous and Mallat algorithms, IEEE Transactions on Signal Processing 40 (10) (1992) 2464–82.

[21] M. González-Audícana, X. Otazu, O. Fors, A. Seco, Comparison between Mallat's and the 'à trous' discrete wavelet transform based algorithms for the fusion of multispectral and panchromatic images, Int J Remote Sens 26 (3) (2005) 595–614.

[22] R. Jain, R. Kasturi, B. Schunck, Machine Vision, McGraw-Hill, 1995.

[23] B. Ballarò, A. Florena, V. Franco, D. Tegolo, C. Tripodo, C. Valenti, An automated image analysis methodology for classifying megakaryocytes in chronic myeloproliferative disorders, Med Image Anal 12 (6) (2008) 703–12.

[24] F. Bellavia, A. Cacioppo, C. Lupaşcu, P. Messina, G. Scardina, D. Tegolo, C. Valenti, A non-parametric segmentation methodology for oral videocapillaroscopic images, Comput Meth Prog Bio 114 (3) (2014) 240–6.

[25] P. Dokládal, E. Dokládalová, Computationally efficient, one-pass algorithm for morphological filters, Journal of Visual Communication and Image Representation 22 (5) (2011) 411–20.

[26] C. Maurer, R. Qi, V. Raghavan, A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimen-

sions, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (2) (2003) 265–70.

[27] R. Fabbri, L. Da F. Costa, J. Torelli, O. Bruno, 2D Euclidean distance transform algorithms: a comparative survey, ACM Computing Surveys 40 (1) (2008) 1–44.

[28] S. Perreault, P. Hebert, Median filtering in constant time, IEEE Transactions on Image Processing 16 (9) (2007) 2389–94.

[29] F. Bellavia, M. Cipolla, D. Tegolo, C. Valenti, An evolution of the non-parameter Harris affine corner detector: a distributed approach, in: Parallel and Distributed Computing, Applications and Technologies, 2009, pp. 18–25.

[30] K. Turkowski, S. Gabriel, Graphics Gems, Academic Press, 1990, Ch. Filters for Common Resampling Tasks, pp. 147–65.

[31] Z. Wang, A. Bovik, H. Sheikh, E. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Transactions on Image Processing 13 (4) (2004) 600–12.

[32] W. Lin, J. Kuo, Perceptual visual quality metrics: a survey, Journal of Visual Communication and Image Representation 22 (4) (2011) 297–312.

[33] G. Bradski, The OpenCV library, Dr. Dobb's Journal of Software Tools 25 (11) (2000).

[34] H. Winnemöller, J. Kyprianidis, S. Olsen, Xdog: An extended difference-of-gaussians compendium including advanced image stylization, Computers and Graphics 36 (11) (2012) 740–53, additional material at https://code.google.com/p/xdog-demo.

[35] C. Topal, C. Akinlar, Edge drawing: a combined real-time edge and segment detector, Journal of Visual Communication and Image Representation 23 (6) (2012) 862–72.