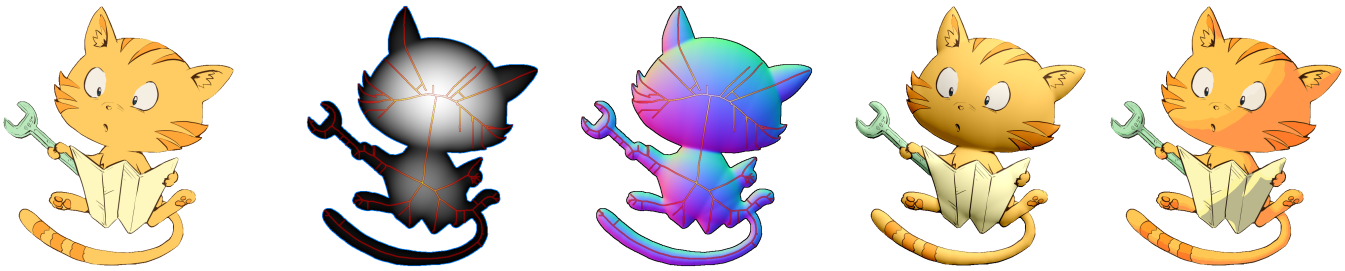
AUTOMATIC ILLUMINATION OF FLAT-COLORED DRAWINGS BY 3D AUGMENTATION OF 2D SILHOUETTES

David Tschumperlé, Christine Porquet and Amal Mahboubi

Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, F-14050 Caen, France
{David.Tschumperle, Christine.Porquet, Amal.Mahboubi}@unicaen.fr



(a) Flat-colored drawing [1] *(b)* Estimated 3D elevation map *(c)* Estimated normal map *(d)* Two examples of automatic illumination rendering

Fig. 1: Our method allows to automatically illuminate a flat-colored drawing *(a)*, by estimating a plausible 3D elevation map from its silhouette *(b)*, as well as a normal map *(c)*. A Phong illumination model is then applied, able to generate different kinds of rendering *(d)*.

ABSTRACT

In this paper, a new automatic method for the illumination of flat-colored drawings is proposed. First, we reconstruct a 3D augmentation of a 2D silhouette from the analysis of its skeleton. Then, we apply the Phong lighting model that relies on the estimated normal map to generate an illuminated drawing. This method compares favorably to recent state-of-the-art methods, *e.g.* those using convolutional neural networks.

Index Terms— Flat-colored drawings, Illumination and Shading, Silhouette, 2D Skeleton, 3D Augmentation, Bumpmap, Normalmap, Phong illumination, Tools for artist.

1. INTRODUCTION

In the field of digital illustration, the transformation of a sketch into a finished artwork requires a number of essential stages, among which the colorization of line art drawings, and the illumination and shading of flat-colored drawings. The automation of these tasks is a particularly difficult, open problem, because of the small amount of geometrically exploitable information that this type of images provides. This is even more true for the illumination problem, which basically requires an estimation, even a rough one, of the 3D relief of the drawing to be processed.

Nevertheless, some automatic or semi-automatic algorithmic techniques have been described in the literature, with the goal of accelerating these manually tedious tasks for the artist. This is the case for flat-solid colorization (*e.g.* [2]) and for the illumination of fixed or animated colorized drawings. For the latter, various formalisms can be found in the litterature: neural networks and *GAN*s [3, 4, 5], interactive multiview 2.5D approaches [6, 7], convolution on integral lines [8], tree-based region representation [9], analysis of silhouettes partitions [10], of their skeletons [11] or of hatching lines added by the user [12]. In these state-of-the-art methods, user interaction is sometimes allowed to guide the algorithms towards a more personalized illumination result (*e.g.*, to correct errors in the algorithmic estimation of the 3D scene geometry).

In this paper, a new method for the automatic illumination of flat-colored drawings is described. Compared to state-of-the-art approaches, our method has three significant advantages: 1. It is extremely simple to implement, parallelizable, and does not require much computational power nor storage memory. 2. Nonetheless, our illumination results largely compete with more time-consuming methods based, for example, on convolutional neural networks [3, 4, 5]. 3. It naturally allows interactivity, since the user can provide geometric guiding lines to refine the algorithm output.

Our process is based on the reconstruction of 2D silhouettes augmented in 3D, from the analysis of their skeletons (detailed in section 2), then by the application of a Phong illumination model [13] which exploits the normal map estimated at the first step (section 3). In section 4, our method is finally validated on a panel of various flat-colored drawings.

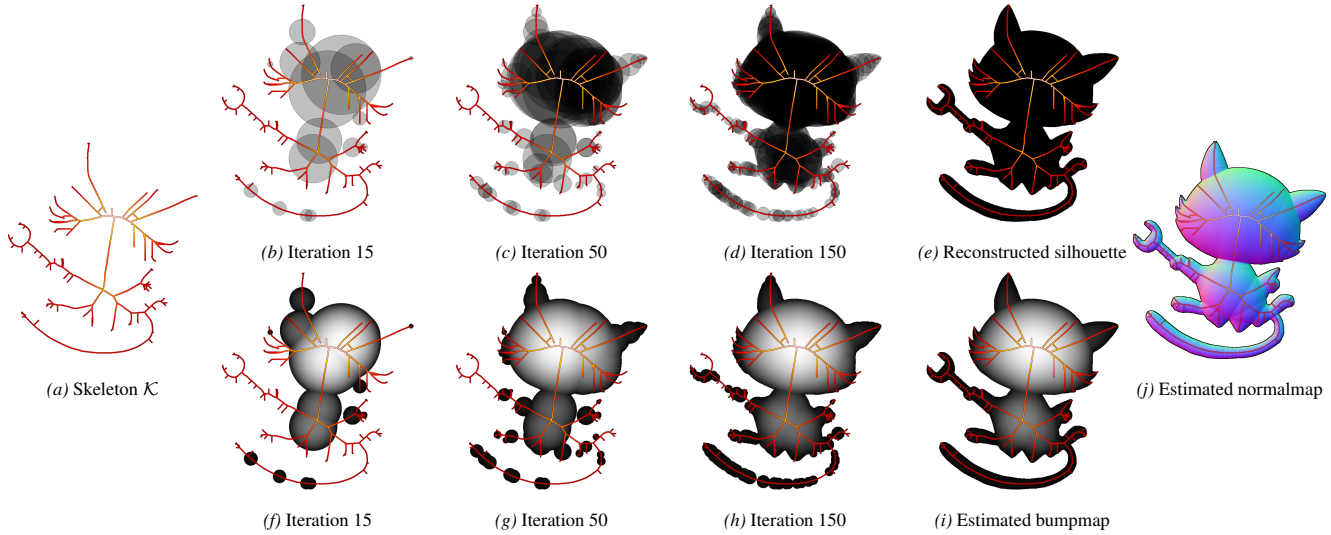


Fig. 2: Any 2D silhouette can be reconstructed by iteratively drawing a set of disks centered at the points of the skeleton $\mathcal{K}$ *(a-e)*. Similarly, we suggest to estimate the 3D elevation as the maximum blending of 3D half-spheres, centered at these same points *(a-i)*.

2. FROM A 2D SILHOUETTE TO A 3D ELEVATION

2.1. Notations

The flat-colored drawing to be illuminated is seen as a function $I : \Omega \to \mathbb{R}^4$ (*i.e.*, a $RGBA$ color image, with an alpha channel), defined on a 2D domain Ω. The binary silhouette of I is defined by the function $S : \Omega \to \{0, 1\}$, with $S(P) = 1$ only for the points $P \in \Omega$ where the alpha channel of $I(P)$ is non zero (opaque pixels). The distance function $D : \Omega \to \mathbb{R}$ is assumed to be known. It gives the minimal euclidean distance between each point such as $S(P) = 1$ and its nearest silhouette border (D can be computed with a linear-time algorithm such as [14]). We have then:

$$D(P) = \min_{Q \in \Omega} \|P - Q\| \quad | \quad S(Q) = 0$$

The skeleton $\mathcal{K}$ of S is estimated as the set of points of D that are locally maximal (*i.e.* maximal for a distance function, the median axes of D).

2.2. Reconstruction of a silhouette S from its skeleton $\mathcal{K}$

$\mathcal{K}$ contains the centers of the maximal 2D disks in S. Thus, by drawing the set of disks centered at the different points $Q \in \mathcal{K}$, of respective radii $D(Q)$, S can be reconstructed (as shown in Fig.2*a-e*). This can be also formalized as :

$$\forall P \in \Omega, \quad S(P) = \max_{Q \in \mathcal{K}} f_Q(P), \quad \text{where}$$

$$f_Q(P) = \begin{cases} 1 & \text{if } \|P - Q\| \leq D(Q) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

f_Q is the indicator function of the disk centered at Q, of radius $D(Q)$. $S(P)$ is non zero only if there exists at least one maximal disk which contains it.

2.3. Estimation of the 3D elevation map (*a.k.a.* bumpmap)

In order to estimate a plausible 3D elevation map of the silhouette S, we simply propose to replace the indicator function f_Q (binary value) by a real-valued function g_Q modeling the elevation of a 3D half-sphere, centered in $Q \in \mathcal{K}$, of radius $D(Q)$, whose curved part is turned towards the viewer. By analogy with equation (1), the bumpmap $B : \Omega \to \mathbb{R}$ can then be computed as: $B(P) = \max_{Q \in \mathcal{K}} \; g_Q(P)$,

$$\text{with} \quad g_Q(P) = \sqrt{\max(0, D(Q)^2 - \|P - Q\|^2)} \tag{2}$$

In practice, this estimation of bumpmap is done by iteratively drawing each function g_Q, for each point Q of skeleton $\mathcal{K}$,starting from an image B with zero initial values.
At a given iteration, the value of $B(P)$ is updated only if its corresponding $g_Q(P)$ is of greater value. These different iterations are illustrated in Fig.2*a,f,g,h,i*. Surprisingly, this is a straightforward way of estimating B that leads in practice to quite plausible 3D elevation maps, even for silhouettes having complex shapes (as shown by our results, detailed in section 4).

2.4. 3D normal map computation

Once the bumpmap B has been estimated with equation (2), the corresponding normalmap $\hat{N} : \Omega \to \mathrm{SO}(3)$ can be computed as:

$$N(P) = \left(\frac{\partial B}{\partial x}(P), \frac{\partial B}{\partial y}(P), -1 \right)^T \tag{3}$$

where $\hat{N} = N/\|N\|$ is the normalized version of vector N. Normalmaps are usually stored and displayed as normalized RGB images, as shown in Fig.2*j*.

Geometric properties: When S is a 2D disk, $\mathcal{K}$ is composed of a single point and the estimated bumpmap B then corresponds visually to that of a 3D sphere. More generally, our method tends to generate rounded 3D elevations for any kind of silhouettes (Fig.5*a-c*). This behavior is actually desirable for modeling the 3D appearance of cartoon-type drawings.

2.5. Regularization step

Equation (2) generates a continuous bumpmap function, although not C^1, which can lead to visible discontinuities when computing the associated normalmap (equation (3) being a derivative one). Applying a guided smoothing method (*e.g.* with the fast algorithm described in [15]), using the silhouette S as a guide image, solves this issue in practice, by efficiently regularizing B (and thus $\hat{N}$) (Fig.3*a-e*).

3. SYNTHETIZING THE ILLUMINATION LAYER

The estimated normalmap $\hat{N}$ is actually the only image required for the generation of the illumination layer J. Indeed, we rely on the Phong lighting model, which calculates the global illumination as the sum of an ambient light (weighted by K_a), a diffuse component (K_d) and a specular component (K_s), as follows:

$$J(P) = K_a + K_d\ (\hat{L}(P) \cdot \hat{N}(P)) + K_s\ \max(0, R(P) \cdot \hat{V}(P))^{\alpha}$$

where

$$\left\{ \begin{array}{lcl} L(P) & = & (L_x - P_x, L_y - P_y, L_z)^T \\ V(P) & = & (C_x - P_x, C_y - P_y, C_z)^T \\ R(P) & = & 2\ (\hat{N}(P) \cdot \hat{L}(P))\ \hat{N}(P) - \hat{L}(P) \end{array} \right. \quad (4)$$

Parameters of the model are (L_x, L_y, L_z), the 3D position of the light, (C_x, C_y, C_z), the camera position, K_a, K_d, K_s, the different illumination weights and α, the specularity coefficient of the illuminated object (Fig.3*f-h*). For the user, it is also convenient to offer additional parameters to be able to globally adjust the values of J (*e.g.*, to choose the $J_{\min}/J_{\max}$ normalization values or to apply gamma correction on J).

The final illumination result is obtained by merging the input image I with this illumination layer J, using an appropriate layer blending mode [16] (typically *Burn, Dodge, Grain Merge, Hard Light, Overlay, Screen* or *Soft Light*). Optionally, a quantization of layer J can emphasize the *cartoon* style of the resulting rendering (Fig.1*d*, Fig.4*e* and Fig.5*i*).

4. ADDING GUIDING LINES AND RESULTS

4.1. User guidance

Sometimes, drawings may include occluded elements whose geometric variations are not visible on the silhouettes alone (for instance, an arm above the body, as in Fig.5*e,j*). In that case, our method cannot recreate the corresponding variations in the estimated normalmap, which may lead to unrealistic illumination effects (often giving a "swollen" appearance to the illuminated drawing, as shown in Fig.4*a,b,c*).
Allowing the user to add a few guiding lines on the drawing brings a simple and efficient solution to this problem. These guiding lines are used to sculpt the silhouette S by constraining the points P that compose them to verify $S(P) = 0$ (and therefore $B(P) = 0$). Such a constraint tends to create sharp discontinuities within the estimated bumpmaps and normalmaps. It helps generating more realistic illuminations near the edges of the occluded regions (Fig.4*a,d,e* and Fig.5*g,k*).

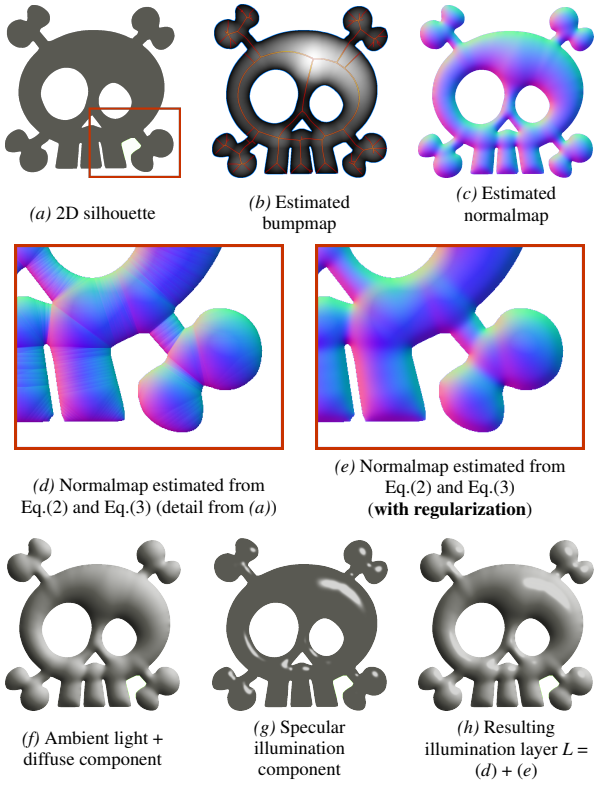
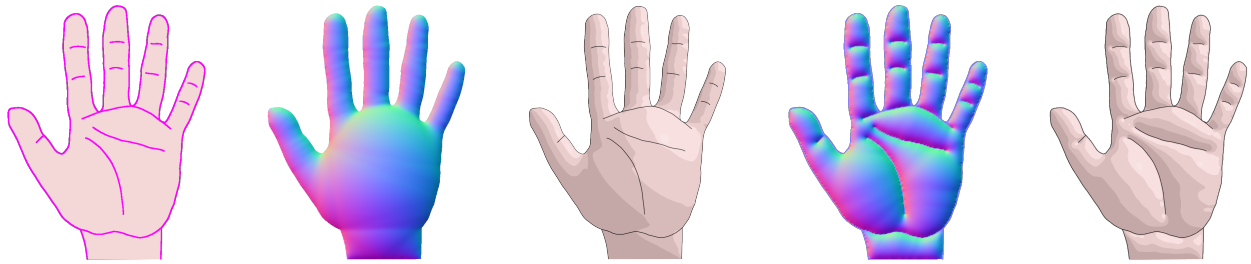


Fig. 3: Regularization effect on the normalmap *(a-e)* and illumination components calculated according to Phong model (*f-h*).

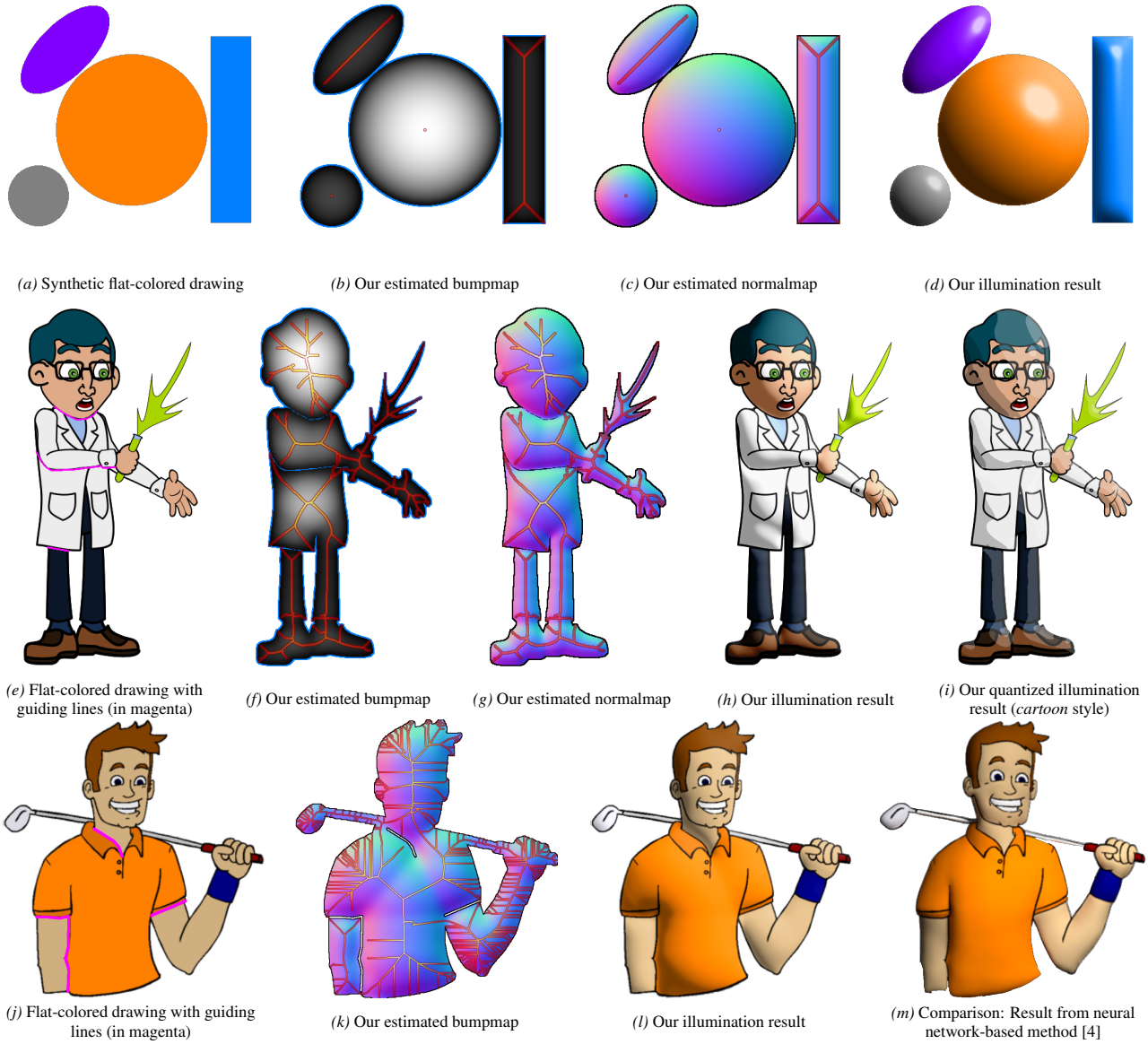
4.2. Results and conclusion

Our illumination algorithm generates visually credible results (Fig.1,4 and 5), similar in quality to far more ressouce-intensive recent approaches based for instance on convolutional neural networks [4] (Fig.5*l,m*). Moreover, our method is easily parallelizable, and thus allows the artist to save time on the tedious process of lighting/shading flat-colored drawings. To foster an open and reproducible research, our algorithm has been integrated into G'MIC [17], an open-source framework for image processing, providing plug-ins for digital image retouching tools such as GIMP, Krita, Photoshop, Affinity Photo, Paint.NET and other software.



(a) Flat-colored drawing with guiding lines (in magenta) | *(b)* Normalmap estimated from the silhouette only | *(c)* Illumination generated from (*b*) | *(d)* Normalmap taking into account guiding lines | *(e)* Illumination generated from (*d*)

Fig. 4: User-defined guiding lines (in magenta) can be taken into account for the estimation of locally more detailed bumpmaps/normalmaps, in the case of complex shapes to be illuminated.



(a) Synthetic flat-colored drawing | *(b)* Our estimated bumpmap | *(c)* Our estimated normalmap | *(d)* Our illumination result

(e) Flat-colored drawing with guiding lines (in magenta) | *(f)* Our estimated bumpmap | *(g)* Our estimated normalmap | *(h)* Our illumination result | *(i)* Our quantized illumination result (*cartoon* style)

(j) Flat-colored drawing with guiding lines (in magenta) | *(k)* Our estimated bumpmap | *(l)* Our illumination result | *(m)* Comparison: Result from neural network-based method [4]

Fig. 5: Results of our automatic illumination technique: From 2D flat-colored drawings *(a,e,j)*, our algorithm computes the skeletons of the silhouettes then estimates 3D elevation maps *(b,f)* and the related normals maps *(c,g,k)*. The latter are then used in a Phong lighting model [13] in order to generate different types of illuminated drawings (*d,h,i,l*). Guiding lines (in magenta) can be seen on *(e,j)*. For the sake of comparison, image *(m)* coming from [4] is shown, a recent method based on convolutional neural networks.

5. REFERENCES

[1] D. Revoy, "Pepper & Carrot webcomics," https://www.peppercarrot.com/, 2022.

[2] S. Fourey and D. Tschumperlé; D. Revoy, "A fast and efficient semi-guided algorithm for flat coloring linearts," in *Int. Symp. on Vision, Modeling & Visualization*, 2018.

[3] Z. Gao, T. Yonetsuji, T. Takamura, T. Matsuoka, and J. Naradowsky, "Automatic illumination effects for 2D characters," in *Proceedings of NIPS Workshop on Machine Learning for Creativity and Design*, 2018.

[4] M. Hudon, S. Lutz, R. Pagés, and A. Smolic, "Augmenting hand-drawn art with global illumination effects through surface inflation," in *European Conference on Visual Media Production*, 2019.

[5] Q. Zheng, Z. Li, and A. Bargteil, "Learning to shadow hand-drawn sketches," 2020.

[6] J.P. Gois, B. Marques, and H.C. Batagelo, "Interactive shading of 2.5 d models.," in *Graphics Interface*, 2015, pp. 89–96.

[7] D. Sỳkora, D. Sedlacek, S. Jinchao, J. Dingliana, and S. Collins, "Adding depth to cartoons using sparse depth (in)equalities," in *Comp. Graphics Forum*, 2010.

[8] Y. Wang, O. Gonen, and E. Akleman, "Global illumination for 2D artworks with vector field rendering," in *ACM SIGGRAPH 2014 Posters*. 2014.

[9] R. Nascimento, F. Queiroz, A. Rocha, T.I. Ren, V. Mello, and A. Peixoto, "Colorization and illumination of 2D animations based on a region-tree representation," in *24th SIBGRAPI Conference on Graphics, Patterns and Images*. IEEE, 2011.

[10] D. Bandeira and M. Walter, "Automatic sprite shading," in *2009 VIII Brazilian Symposium on Games and Digital Entertainment*. IEEE, 2009, pp. 27–31.

[11] H. Bezerra, B. Feijó, and L. Velho, "An image-based shading pipeline for 2D animation," in *Brazilian Symp. on Comp. Graphics and Image Processing*. IEEE, 2005.

[12] M.T. Bui, J. Kim, and Y. Lee, "3D-look shading from contours and hatching strokes," *Computers & Graphics*, vol. 51, 2015.

[13] B.T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, 1975.

[14] H. Breu, J. Gil, D. Kirkpatrick, and M. Werman, "Linear time euclidean distance transform algorithms," *IEEE Trans. on Pattern Anal. and Machine Intel.*, vol. 17, no. 5, 1995.

[15] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 6, pp. 1397–1409, 2012.

[16] Y. Ma, "The mathematic magic of photoshop blend modes for image processing," in *International Conference on Multimedia Technology*, 2011.

[17] D. Tschumperlé and S. Fourey, "G´MIC: A Full-Featured Open-Source Framework for Image Processing," https://gmic.eu/, 2008–2022.

[18] M. Hudon, M. Grogan, R. Pagés, J. Ondrej, and A. Smolic, "2DToonShade: A stroke based toon shading system," *Computers & Graphics*, vol. vol. 1, 2019.

[19] M. Hudon, R. Pagés, M. Grogan, J. Ondřej, and A. Smolić, "2D shading for cel animation," in *Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*, 2018.

[20] L. Petikam, K. Anjyo, and T. Rhee, "Shading rig: Dynamic art-directable stylised shading for 3D characters," *ACM Trans. Graph.*, vol. 40, no. 5, sep 2021.