Colorization by Multidimensional Projection

Wallace Casaca*, Erick Gomez-Nieto*, Cynthia O. L. Ferreira†, Geovan Tavares†
Paulo Pagliosa‡, Fernando Paulovich*, Luis Gustavo Nonato* and Afonso Paiva*

*ICMC, USP, São Carlos
† Dept. of Mathematics, PUC-Rio, Rio de Janeiro
‡FACOM, UFMS, Campo Grande

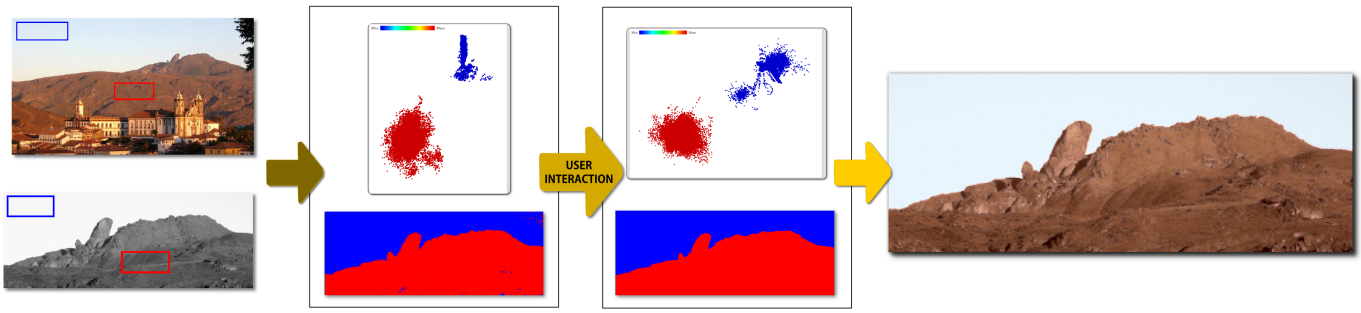


Fig. 1. Our approach takes as input a grayscale image and a set of colored pixels picked out from a source color image (left). Pixels in the grayscale image are mapped to a feature space producing high-dimensional data instances which are projected to a visual space using an interactive multidimensional projection method (middle left). User can then interact with projected data so as to untangle clusters (middle right) and thus improving colorization result (right).

***Abstract*—Most image colorization techniques assign colors to grayscale images by embedding image pixels into a high-dimensional feature space and applying a color pattern to each cluster of high-dimensional data. A main drawback of such approach is that, depending on texture patterns and image complexity, clusters of similar pixels can hardly be defined automatically, rendering existing methods prone to fail. In this work we present a novel approach to colorize grayscale images that allows for user intervention. Our methodology makes use of multidimensional projection to map high-dimensional data to a visual space. User can manipulate projected data in the visual space so as to further improve clusters and thus the colorization result. Different from other methods, our interactive tool is easy of use while still being flexible enough to enable local color modification. We show the effectiveness of our approach through a set of examples and comparisons against existing colorization methods.**

***Keywords*-Image colorization; image processing; multidimensional projection, high dimensional data.**

I. Introduction

Colorization is a computer-assisted process by which color is imparted to black-and-white films or to grayscale images. It has been widely used in photo processing and scientific illustration, to modernize black-and-white films and to restore damaged color films. Traditionally, colorization is tedious, time consuming and requires artistic skills to precisely add appropriate colors to a grayscale image.

Aiming at making the colorization process simpler and less laborious, several computational systems and tools have been proposed in the last decade, which can be roughly divided into two classes: example-based and scribble-based. Example-based methods accomplish the colorization process by matching the luminance of the grayscale image with the luminance of a source color image used to drive the colorization. In practice, user picks out regions (swatches) on the target grayscale image and matches those regions with color patterns in the source color image. A typical example-based colorization scheme was proposed by Welsh et al. [1], which combines image analogies [2] and color transfer [3] to colorize a grayscale image. Irony et al. [4] improve Welsh's method by using a segmented reference image and a texture matching scheme [5] to better colorize regions with similar textures. In a more recent work, Liu et al. [6] make use of multiple reference images retrieved from the Internet to guide the colorization process and mitigate problems related to environment illumination. Example-based methods end up being very sensitive to the source color image employed to drive the colorization process while still performing poorly in images with complex textured patterns.

In scribble-based methods the user drives the colorization by defining colored strokes onto the grayscale image. The classical work by Levin et al. [7] is a good representative of scribble-based approach. Levin's method aims at optimizing the color of all image pixels using the scribbles as constraints. Although it shows good results for various types of images, Levin's method tends to propagate colors beyond the texture boundaries, thus resulting in unpleasant colorizations. The technique proposed by Huang et al. [8] employs adaptive edge detection so as to prevent colors from going beyond region

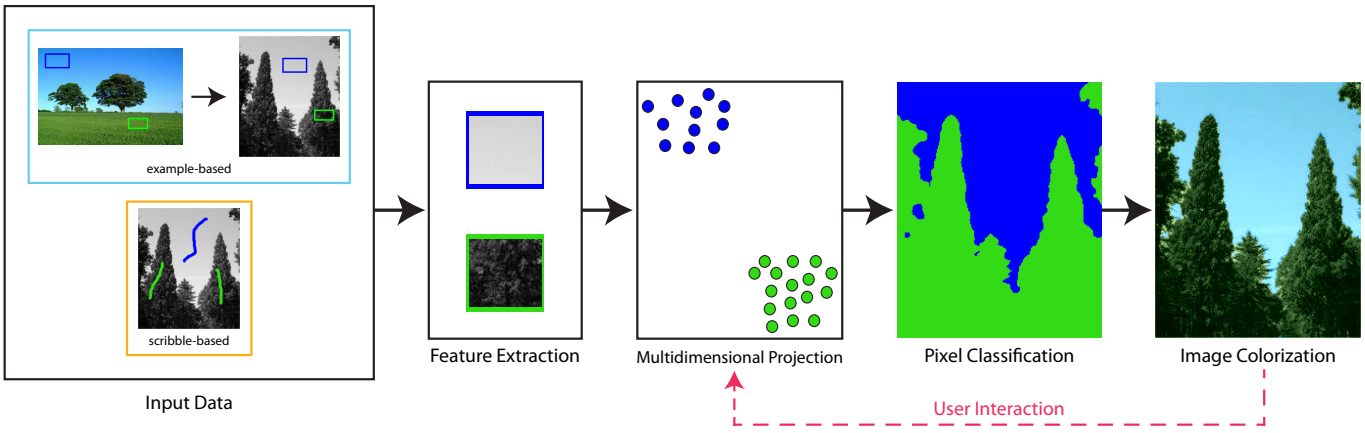


Fig. 2. Projcolor pipeline.

boundaries. Further improvements have been proposed by Yatziv and Sapiro [9], who present a faster scribble-based color optimization technique that relies on chrominance blending to perform the colorization, and by the authors of [10], [11], which employ texture continuity to colorize manga-cartoons and natural images, respectively. Despite the good results, all existing scribble-based approach require intensive user intervention, especially when the image contains complex structures or has different kinds of textures, which can demand lots of scribbles until acceptable outcomes are reached.

In this work we propose a new approach for colorizing grayscale images that relies on interactive multidimensional projection to replace the intensive user-driven scribble mechanism by a simple drag-and-drop manipulation of the badly colorized pixels. This novel interactive scheme, called *Projcolor*, operates from example-based as well as color scribbles, demanding just a few user interventions to correct imperfections in regions poorly colorized. Moreover, as the multidimensional projection allows for visualizing the neighborhood structure of the pixels under colorization, the user can control color propagation in an accurate and intuitive manner, a trait not present in other methods proposed in the literature. As we shall show, by interacting with the multidimensional projection the user can colorize complex textured regions quite easily, respecting region edges and avoiding introducing new scribbles. Since the user can accurately control the colorization process, our approach is less sensitive to the source color image.

Contributions In summary, the main contributions of this work are:

- the introduction of interactive multidimensional projection in the context of image colorization;
- a pixel neighborhood manipulation scheme to drive the colorization process;
- an efficient system that allows for colorizing grayscale images in an intuitive and accurate manner.

II. Multidimensional Projection-based Colorization

As illustrated in Fig. 2, the proposed colorization pipeline comprises four main steps, namely, input color data, feature extraction/selection, multidimensional projection, and image colorization. In our framework, the color data can be provided in two distinct ways: color swatches picked out from a RGB image, or color scribbles brushed by the user. Similar to Welsh et al. [1], the color data and the grayscale image are converted from RBG color and Grayscale space to the independent $l\alpha\beta$ space, where l is the luminance channel and α, β are two chromatic channels. During the feature extraction stage, each pixel of the grayscale image is mapped to a point in a high-dimensional space whose coordinates correspond to statistics measures and texture attributes computed from luminance and gray values in the neighborhood of the pixel. The high-dimensional data is project to a two-dimensional visual space so as to preserve neighborhood structures as much as possible. Colorization is then performed based on information computed from the projected data. Badly colored regions can be updated by interacting with the projected data in the visual space. In other words, neighborhood structures in the visual space can be modified by the user so as to generate better colorization. Details of each stage of the pipeline are presented in the following subsections.

A. Feature Extraction/Selection

Given a grayscale image H, the feature extraction step aims at mapping each pixel from H to a point in a high-dimensional feature space $\mathcal{X} \subset \mathbb{R}^N$. In more mathematical terms, we build a mapping $f : H \rightarrow \mathcal{X}$ that assigns to each pixel $p \in H$ a point $f(p) \in \mathbb{R}^N$ called *feature vector*. The coordinates of $f(p)$ are given by local feature descriptors computed in a neighborhood of p. In our implementation we are using *Gabor* [12], *first order moment* [13], *local binary patterns* and their variants [14], [15], *scale invariant feature transforms* [16], [17], and *ordinary statistical/morphological measures* [18].

In practice, features resulting from distinct extractors can be highly correlated and it is not necessary to keep all of them in the feature vector. Removing unnecessary data reduces the dimensionality of the feature space while still untangling clusters, thus impacting positively in the result of the colorization process. We handle unnecessary features by applying the well known learning-based feature selection tool called *WEKA* [19]. WEKA computes the correlation among features (see [20]) and ranks the more relevant ones. In our tests, about 20% of features are kept after filtering them out using WEKA.

B. Multidimensional Projection

The multidimensional projection step is responsible to map high-dimensional data generated during the feature extraction/classification step onto a two-dimensional visual space. In our implementation we are employing the multidimensional projection technique called *LAMP* (*Local Affine Multidimensional Projection*) [21]. Besides preserving neighborhood structures quite well, the LAMP technique enables a very flexible mechanism to interactively modify the projection according to user interventions, which is an important characteristic in the context of this work.

LAMP uses a set of control points to perform the mapping of a set of high-dimensional data $\mathcal{X}$ to the visual two-dimensional space. The set of control points is typically a small subset $\mathcal{X}_S \subset \mathcal{X}$ whose basic information in the visual space is known a priori ($\mathcal{X}_S$ can be mapped to the visual space using distance preserving optimization scheme such as [22]). The mapping of each instance x in $\mathcal{X}$ to visual space is carried out based on the linear mapping T_x that minimizes (see [21] for details):

$$\|AT_x - B\|_F \text{ subject to } TT^{\perp} = I, \tag{1}$$

where $\|\cdot\|_F$ is the Frobenious norm, $T^{\perp}$ is the transpose of T, I is the identity matrix, and A and B are matrices whose rows correspond to the coordinates of each control point in the high-dimensional and visual space, respectively. The constraint $TT^{\perp} = I$ is imposed to ensure that distances are preserved as much as possible during projection. When a user manipulates control points in the visual space s/he also changes entries in the rows of matrix B, thus tuning the mapping T_x to cope with the user intervention. More precisely, when a control point is moved in the visual space the data in its neighborhood is moved together, what allows for controlling neighborhood structures in the visual space.

In our colorization framework, the input control points are assigned through the pixels selected by the user-defined strokes or swatches. User can then interactively manipulate control points in the visual space based on the resulting colorization. The flexibility to change projection according to user manipulation of control points is exploited by the proposed colorization application as described below.

C. Image Colorization

The colorization of a grayscale image relies on pixel classification carried out in the visual space, that is, after mapping the high-dimensional data to the visual space using LAMP, a clustering mechanism is employed to group pixels with similar features (we apply the KNN classifier on the projected data).

Once "similar pixels" have been clustered, colors are associated to each cluster from the information initially provided by the user-defined strokes or swatches. More precisely, the user associates a color pattern to a subset of pixels and that color pattern is then propagated to all pixels in the cluster. The propagation mechanism is accomplished as follows: given a grayscale pixel p_g and a user provided set of colored pixels P_c, the $l\alpha\beta$ color of p_g is given by taking the α and β components from pixel $p_c \in P_c$ that minimizes the distance function d_l defined by:

$$d_l(l_c, l_g) = 0.5\left(|l_c - l_g| + |std_c - std_g|\right), \tag{2}$$

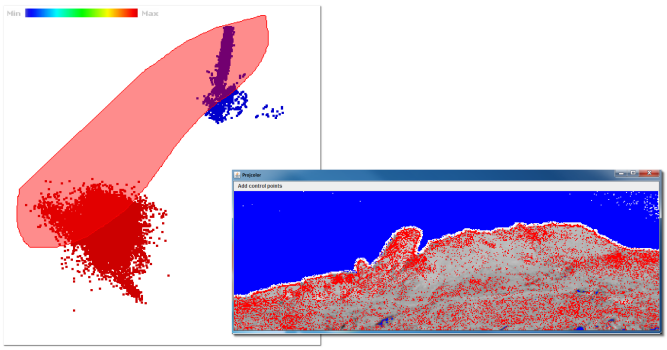
where l_c and l_g are the luminance values of p_c and p_g, respectively. The values std_c and std_g are the standard deviations of l_c and l_g, respectively. After processing the pixels in $l\alpha\beta$ color system, pixel conversion to RGB system is then performed in order to produce the final color result. In our experiments we use a pixel neighborhood of size 5×5 to compute standard deviations of the pixel luminance.

D. User Manipulation of Clusters

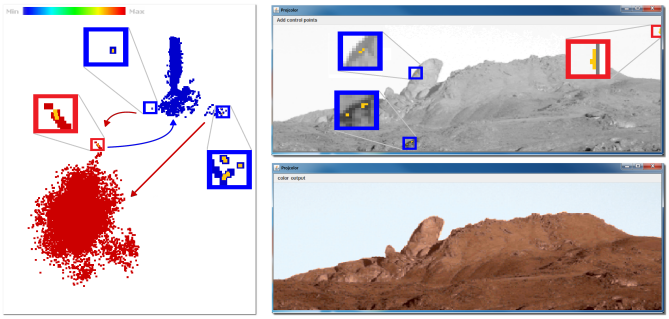
One of the main contributions of Projcolor is to exploit the flexibility provided by the multidimensional projection scheme towards interactively manipulating projected data so as to modifying clusters. When features extracted from distinct regions of an image are similar, pixels belonging to those regions cannot be clearly separated either in the high-dimensional space or in the projected space, since the multidimensional projection preserves neighborhood structures as much as possible. LAMP, however, enables to user with an interactive tool that allows for untangling data by manipulating control points. In fact, if the coloring result is not satisfactory, the user can select badly colored pixels, turning them into new control points that can be interactively moved in the visual space to better define clusters and thus to improve the result of the colorization. Fig. 3 illustrates the user intervention process where control points are defined by the user and then dragged to different clusters within visual space. Since LAMP also drags the neighborhood of the control points after user interaction it is necessary to define just a few control points to achieve satisfactory colorizations as shown in Fig. 3(b).

III. Results and Comparisons

Fig. 4 shows the result of colorizing the picture in Fig. 4(b) taking as input the set of color pixels within the two boxes (drawn by the user) in Fig. 4(a). The result of projecting the high-dimensional data onto the visual space is shown in Fig. 4(c). Notice that two clusters corresponding to the

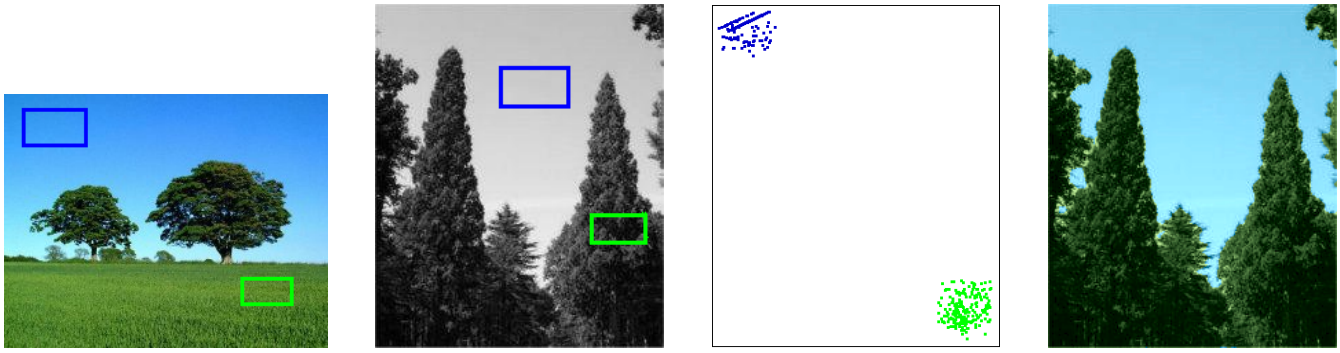


(a) Pixels corresponding to projected data (left) lying within a user highlighted regions are depicted in the grayscale image (right) and vice versa



(b) Badly clustered pixels defined as control points by the user (right top), drag-and-drop manipulation of the corresponding pixels within projected data (left) and the resulting colored image (right bottom)

Fig. 3. Illustration of user interaction with projected data and target image.



(a) Source image (b) Grayscale image (c) Multidimensional projection (d) Colorized image

Fig. 4. Multidimensional projection with example-based scheme.

sky and the trees are formed in the visual space. The clear separation of clusters enables an accurate colorization without user intervention as presented in Fig. 4(d).

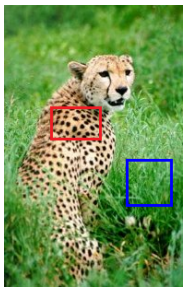
The colorization example presented in Fig. 5 illustrates the need of user intervention. Fig. 5(a) and (b) show the source color image (and the color examples took from it) and the grayscale image to be colored. Fig. 5(c) depicts the result of the multidimensional projection from which the color image in Fig. 5(d) is generated. Notice that pixel close the edge between the panther and the background grass are not colored properly. User can then select some of those badly colorized pixels (red circles in Fig. 5(d)), which are made control points for the multidimensional projection. Dragging the new control points to the correct cluster changes the projection layout (Fig. 5(e)) and improves the resulting colorization, as shown in Fig. 5(f).

Fig. 6 illustrates the capability of Projcolor to perform colorization based only on colored strokes, that is, the input is color scribbles drawn by the user on each region s/he wants to define a color. Color scribbles are simpler than example-based colorization, as the user does not need to provide a colored source image not a correspondence between them. Despite its simplicity and ease of use, the stroke-based scheme results in quite satisfactory colorizations.

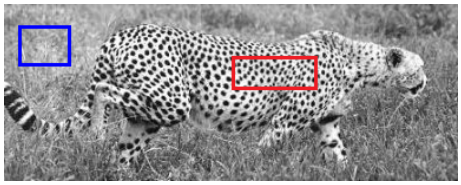
Comparisons In order to confirm the effectiveness of Projcolor we provide comparisons against Welsh's [1] (example-based) and Levin's [7] (scribble-based) methods.

Fig. 7 compares Projcolor with both techniques mentioned above. Colorization method proposed by Welsh et al. works properly in images that do not contain complex texture patterns and fails drastically when dealing with textures, as depicted in Fig. 7(b). Otherwise, Projcolor generated a clear and pleasant result as shown in Fig. 7(c). Fig. 7(d) shows the grayscale image and two user defined scribbles on it. Fig. 7(e) and (f) are the colorization produced by Levin's method and Projcolor respectively. Notice that our approach is much more robust when dealing with textures. It is important to say that the results presented in Fig. 7(e) and Fig. 8 were obtained using the MATLAB code available on authors website (`http://www.cs.huji.ac.il/~yweiss/Colorization/`).

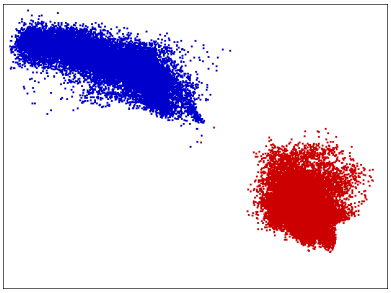
Fig. 8 brings another comparison between Projcolor and Levin's method. Fig. 8(b) and (c) are the resulting colorization obtained from Levin's method and Projcolor using only the two color strokes depicted in Fig. 8(a). Notice that Levin's method does not generate a good colorization due to the complexity of textured regions while Projcolor results in a pleasant colorization. As shown in Fig. 8(d) and (e), Levin's approach produces better results when several strokes are



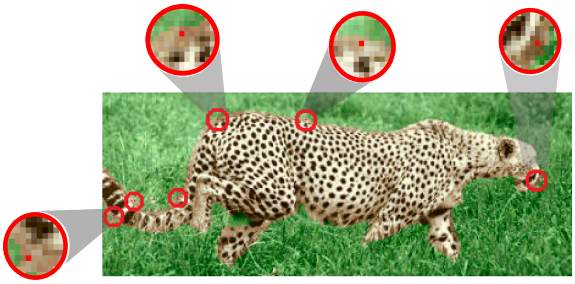
(a) Source image



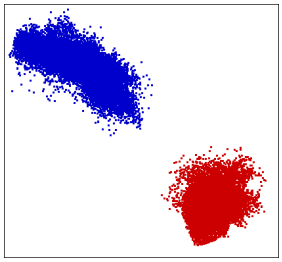
(b) Grayscale image



(c) Multidimensional projection



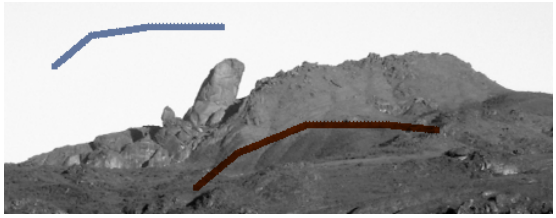
(d) Colored image with flaws



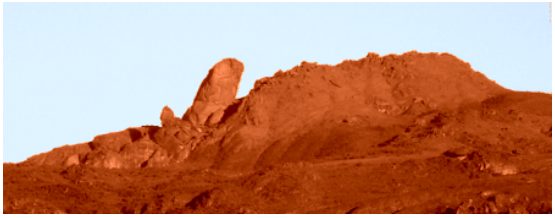
(e) User manipulated projection

(f) Final colorization

Fig. 5. Neighborhood manipulation to further improving colorization.



(a) Target image and its color strokes



(b) Projcolor result

Fig. 6. Colorization created from user-provided color strokes.

provided by the user. Projcolor, however, demanded just a couple of user intervention to update clusters and reach the nice result presented in Fig. 8(c), proving that Projcolor is more robust when dealing with complex textured images.

Regarding computational times, Projcolor is quite similar to the Welsh's and Levin's methods. The computational cost increases proportionally to the number of swatches or scribbles employed to color images, neighborhood sizes in feature extraction step and the image size. In our experiments almost all images were colorized in a few tens of seconds (our code was implemented in MATLAB with support to Java routines).

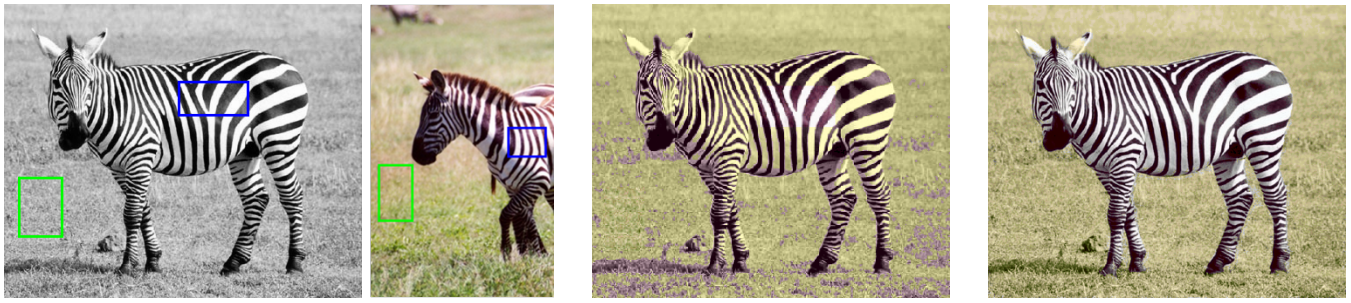
IV. Discussion and Limitations

The comparisons presented in Section III clearly show the effectiveness of the proposed coloring method, surpassing, in requisites such as accuracy and flexibility, existing methods. The local control of neighborhood structures enables the user with a sharp control of color propagation within pixel clusters, a characteristic not present in any other colorization method.

The good results when dealing with highly textured images such as the one in Fig. 5 show the robustness of the proposed approach. In fact, just a few interventions were needed to reach a pleasant colorization result. As mentioned above, regarding computational times, our method is comparable against two other approaches. In fact, our approach is a bit faster than the compared methods. Despite the good properties and results, Projcolor also has limitations. Although LAMP allows for controlling the extent neighborhoods are affected when manipulating control points, properly setting the number of control points that affects the projection is not a straightforward task, being this an issue to be further investigated.

V. Conclusion

In this work we proposed the use of multidimensional projection as basic tool for image colorization applications. The evaluation we provided shows that Projcolor outperforms existing techniques in terms of accuracy as well as flexibility. Besides enabling a local modification of badly colored regions, the proposed methodology turns out to be robust when dealing with complex textured images. In summary, flexibility and effectiveness render the proposed method one of the most attractive alternatives in the context of image colorization.



(a) Grayscale (left) and source color image (right)

(b) Welsh's colorization from (a)

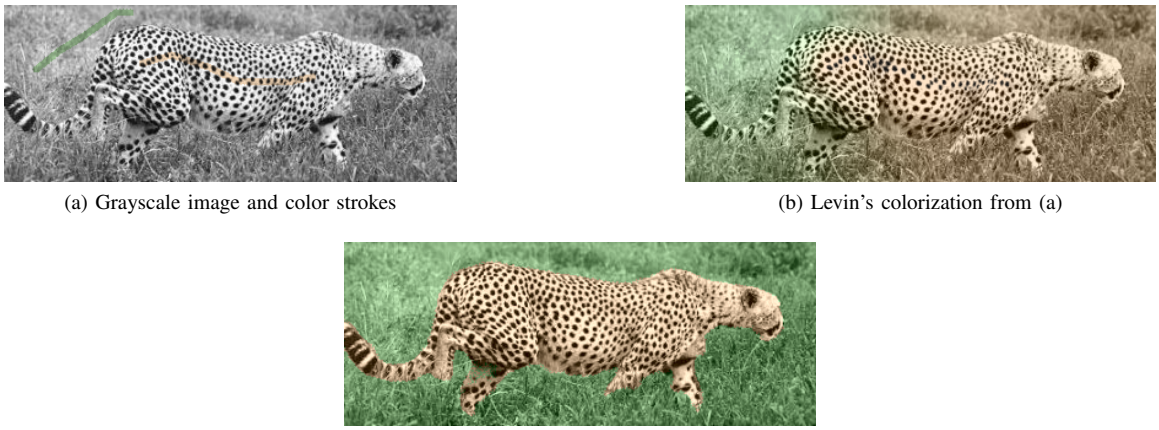
(c) Projcolor colorization from (a)

(d) Selected region by scribbling

(e) Levin's colorization from (d)

(f) Projcolor colorization from (d)

Fig. 7. Comparing Projcolor with Welsh's and Levin's methods.



(a) Grayscale image and color strokes

(b) Levin's colorization from (a)

(c) Projcolor colorization from (a)

(d) Adding new strokes

(e) Levin's colorization from (d)

Fig. 8. Comparing Projcolor and Levin's methods.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their useful comments to improve the quality of this paper. The authors are supported by PUC-Rio, CAPES, CNPq and FAPESP.

REFERENCES

[1] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 277–280, 2002.

[2] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. Salesin, "Image analogies," in *SIGGRAPH*, 2001, pp. 327–340.

[3] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color transfer between images," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 34–41, 2001.

[4] R. Irony, D. Cohen-Or, and D. Lischinski, "Colorization by example," in *Proc. of the Eurographics Symposium on Rendering*, 2005, pp. 201–210.

[5] Y. Schnitman, Y. Caspi, D. Cohen-Or, and D. Lischinski, "Inducing semantic segmentation from an example," in *Proc. of the 7th Asian Conf. on Computer Vision*, vol. 3852, 2006, pp. 373–384.

[6] X. Liu, L. Wan, Y. Qu, T.-T. Wong, S. Lin, C.-S. Leung, and P.-A. Heng, "Intrinsic colorization," *ACM Trans. Graph.*, vol. 27, no. 5, pp. 152:1–152:9, 2008.

[7] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 689–694, 2004.

[8] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu, "An adaptive edge detection based colorization algorithm and its applications," in *Proc. of the 13th ACM Intl. Conf. on Multimedia*, 2005, pp. 351–354.

[9] L. Yatziv and G. Sapiro, "Fast image and video colorization using chrominance blending," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1120–1129, 2006.

[10] Y. Qu, T.-T. Wong, and P.-A. Heng, "Manga colorization," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1214–1220, 2006.

[11] Q. Luan, F. Wen, D. Cohen-Or, L. Liang, Y.-Q. Xu, and H.-Y. Shum, "Natural image colorization," in *Proc. of the Eurographics Symposium on Rendering*, 2007, pp. 309–320.

[12] S. Arivazhagan, L. Ganesan, and S. P. Priyal, "Texture classification using gabor wavelets based rotation invariant features," *Pattern Recogn. Lett.*, vol. 27, no. 16, pp. 1976–1982, 2006.

[13] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, 2009.

[14] Y. Guo, G. Zhao, and M. Pietikinen, "Discriminative features for texture description." *Pattern Recognition (to appear)*, 2012.

[15] Y. G. G., , and M. Pietikinen, "Descriptor learning based on fisher separation criterion for texture classification." in *Proc. of the 10th Asian Conf. on Computer Vision*, 2010, pp. 1491–1500.

[16] D. Lowe, "Object recognition from local scale-invariant features," in *Proc. of the 7th IEEE Conf. on Computer Vision*, vol. 2, 1999, pp. 1150–1157.

[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[18] R. C. Gonzlez and R. E. Woods, *Digital Image Processing*. Prentice Hall, 2008.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, 2009.

[20] M. A. Hall, "Correlation-based feature selection for machine learning," The University of Waikato, Tech. Rep., 1998.

[21] P. Joia, D. Coimbra, J. Cuminato, F. Paulovich, and L. G. Nonato, "Local affine multidimensional projection," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, pp. 2563–2571, 2011.

[22] E. Tejada, R. Minghim, and L. G. Nonato, "On improved projection techniques to support visual exploration of multidimensional data sets," *Information Visualization*, vol. 2, no. 4, pp. 218–231, 2003.