STUDIA INFORMATICA 2010
Volume 31 Number 2A (89)

Michal KAWULOK
Politechnika Śląska, Instytut Informatyki
Bogdan SMOLKA
Politechnika Śląska, Instytut Automatyki

IMAGE COLORIZATION WITH COMPETITIVE PROPAGATION PATHS AND CHROMINANCE BLENDING

Summary. This paper presents a new method for image colorization based on manually added scribbles. First, we determine color propagation paths in the image by minimizing geodesic distance from the scribbles using Dijkstra algorithm. After that, blending distance is calculated along each path to determine final chrominance. The results are compared with those obtained with other existing methods.

Keywords: image colorization, chrominance blending

KOLORYZACJA OBRAZÓW ZA POMOCĄ RYWALIZUJĄCYCH ŚCIEŻEK PROPAGACJI ORAZ MIESZANIA CHROMINANCJI

Streszczenie. Artykuł przedstawia nową metodę koloryzacji obrazów na podstawie ręcznie naniesionych mazów. Pierwszym krokiem jest znalezienie ścieżek propagacji barwy w obrazie poprzez minimalizację odległości za pomocą algorytmu Dijkstry. Następnie obliczana jest odległość barwna wzdłuż każdej wyznaczonej ścieżki i na jej podstawie dokonywane jest mieszanie chrominancji. Wyniki działania algorytmu są porównane z otrzymanymi za pomocą innych istniejących metod.

Słowa kluczowe: koloryzacja obrazów, mieszanie chrominancji

1. Introduction

Image colorization is an automatic or user-assisted process of adding colors to a grayscale image. Usually, the purpose of this transformation is to enhance visual attractiveness of monochrome photographs or videos which color versions are not available, but can be also treated as a method of image segmentation and marking regions of interest, e.g. in case of medical images. Not only are image colorization algorithms used for adding colors to grayscale images, but also for color modifications known as *recolorization*, *color transfer* between images or extracting certain image regions (so called *matting*). Applications of image colorization include colorization of old movies (which is quite controversial for many artists, but desired in the mass culture world), enhancing old photographs with faded colors, segmentation of medical images, interior design or make-up simulators and many others. Noteworthy, image colorization techniques may enhance functionality of multimedia databases, e.g. to generate dynamically various color versions of the same image or to represent an image as a set of regions of unique chrominance.

Usually, the colorization is performed based on a set of manually-added color scribbles to a grayscale image as presented in Fig. 1. After that, the image is colorized automatically by spreading the scribbles in the luminance channel. Ideally, the image would be segmented and colored independently on how precisely the scribbles are placed, but in practice the result strongly depends on scribbles' initial positions due to poor color propagation algorithms. Therefore, color propagation and chrominance blending are challenging tasks addressed by computer vision community.

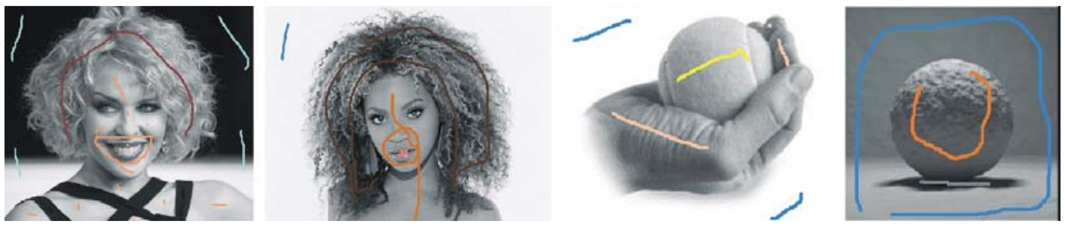
Fig. 1. Examples of images with manually added scribbles
Rys. 1. Przykłady obrazów z ręcznie naniesionymi mazami

1.1. Related work

Image colorization attracts considerable attention from the academia world and various methods have been already proposed. Overview of the existing techniques is presented in this section. There are also some commercial applications which support the colorization, but they require high user's interaction and experience.

The first method of adding colors to the image was proposed by Gonzalez and Wintz [2] in a form of *luminance keying*. It operates based on a function which maps every luminance level into color space. Obviously, the whole color space cannot be covered in this way without increasing manual input from the user. Welsh et al. proposed a method of *color transfer* [11] which colorizes a grayscale image based on a given reference color image. This method matches textural and luminance information and can be performed automatically, but gives better results with user assistance. Sykora et al. [10] proposed an unsupervised method for image colorization by example, which at first matches similar image feature points to predict their color. After that, the color is spread all over the image by probabilistic relaxation. Horiuchi [3] proposed an iterative probabilistic relaxation, in which a user defines colors for selected grayscale values, based on which the image is colorized.

There are also a number of methods which are focused on using the prior information delivered by a user in a form of color scribbles. Levin et al. [7] formulated an optimization problem based on an assumption that neighboring pixels of similar intensity should have similar color values under the limitation that the colors indicated in the scribbles remain the same. Yatziv and Sapiro [12] proposed a method for determining propagation paths in the image by minimizing geodesic distances from every scribble. Based on the distances from each scribble, pixel color is obtained by blending scribble chrominances. In other works, the color is also propagated from scribbles with probabilistic distance transform [9], using cellular automaton [6] or by random walks with restart [5].

1.2. Outline of the proposed method

Our research was focused on image colorization based on manually-added scribbles. The color propagation paths are calculated by minimizing local pixel distance integrated along the path using Dijkstra algorithm [1]. Usually in other well-established colorization techniques [9, 12] the distance between two neighboring pixels is proportional to a difference in their brightness. This performs correctly for colorization of vast plain areas, but often fails for textured surface (e.g. human hair). Therefore, we decided to adapt the distance to every scribble depending on the textural properties of the region where the scribble is placed. Details of the path optimization are described in section 2.

Usually the distances, based on which the paths are optimized, are used for determining weights for chrominance blending as well. However, we propose to separate these two tasks and determine the weights based on the path properties. Details of this algorithm are presented in section 3. Experimental results and comparisons with Levin [7] and Yatziv [12] methods are presented in section 4. The paper is concluded in section 5.

2. Color propagation paths

In order to colorize a monochromatic image Y based on a set of n initial scribbles $\{S_i\}$, first it is necessary to determine the propagation paths from each scribble to every pixel in the image. A path from pixel x to y is defined as a discrete function $p(t):[0,l] \to N^2$ which maps position in the path to pixel coordinate. The position is an integer ranging from 0 for the path beginning ($p(0) = x$) to 1 for its end ($p(l) = y$). Also, if $p(i) = a$ and $p(i+1) = b$, a and b are neighboring pixels. The paths should be determined in such a way that number of desired chrominance changes along the path is minimal.

2.1. Propagation paths optimization

In our approach the paths are determined by minimization of a *total path cost*:

$$C(p) = \sum_{i=0}^{l-1} dist\big(p(i), p(i+1)\big), \tag{1}$$

where *dist* is a *local distance* between two neighboring pixels, defined in section 2.2. The minimization is performed with Dijkstra algorithm [1, 4] in the following way:

1. A pixel queue Q is initialized with all pixels belonging to a scribble.
2. Distance array D which covers all pixels in the image is created. Every pixel $q \in Q$ is assigned with a zero distance (D(q)=0) and all remaining pixels are initialized with an infinite distance.
3. A single pixel q is popped from the queue and for each of its 8 neighbors Ni(q) the following actions are performed:
4. Local distance dist(q,s) between q and its neighbor s is calculated to find a total cost of ps.
5. If C(ps) is smaller than a current distance D(s), the distance is updated and the pixel s is added to the queue Q and it is associated with a new path ps.
6. If the queue is empty, the algorithm finishes. Otherwise, point (3) is repeated.

With the described algorithm optimal propagation paths and distances from the scribble to every pixel are obtained. Noteworthy, the optimal paths depend on how the distance between two neighboring pixels is calculated.

Yatziv and Sapiro [12] define the path by minimizing integrated luminance gradient in the direction of the path. Hence, the local distance is defined as intensity differences between subsequent pixels in the path. This is an interesting approach, appropriate to determine paths which are supposed to cross easily plain areas without strong edges and suitable for such images, in which luminance difference is proportional to probability of chrominance change. This approach is similar to a traveler who intends to cross an island with beaches along the coast and mountains in the centre. He would choose a longer way along the coast rather than a shorter one across the mountains. However, if he is placed in the centre, the effort of getting to the coast will be quite high. This is reasonable, but if a scribble is added to a rough area of the image (e.g. human hair), the distance will grow rapidly. Moreover, it often does occur better to join two points which belong to the same rough area by leaving that area with minimal cost, taking a longer way along a plain area and getting back to the rough area. It would be desired by a traveler, but it is not suitable for image colorization. We would prefer not to leave the rough area, because it is likely to have uniform chrominance, so the number of chrominance changes along the path would be two instead of zero.

When a scribble is placed in a rough area, it is better to follow high gradients without much cost, making it similar to an idea of *intelligent scissors* proposed by Mortensen [8] for interactive image segmentation. With an intelligent scissors tool a user performs object segmentation using a mouse. The algorithm finds the shortest path between the starting point and a mouse pointer in such a way, that the path is sticky to the strongest gradient. Local cost between two neighboring pixels depends on the laplacian zero-crossings, gradient magnitude and direction. Basically, the cost is lower if the path follows the gradient direction and the gradient magnitude of the path pixels is high.

2.2. Local distance between pixels

During our study we developed two ways of calculating the local distances, namely *plain distance* and *gradient distance*. The first one is similar to those used in other colorization methods, and its aim is to minimize intensity changes along the path. It is calculated as:

$$dist_p(x,y) = 1 - \exp\left(-\frac{|Y(x)-Y(y)|}{h_p}\right), \tag{2}$$

where h_p is the normalization factor which determines how sensitive the distance is, depending on local intensity changes. Its value was empirically set to 30 on the basis that it delivered the best results for the tested images. This kind of distance is suitable for determining paths in uniform regions which texture is not characterized by strong gradients.

However, for objects which texture is not smooth, the paths are not found correctly in this way and also the distance from a scribble grows rapidly when high gradients are crossed. Therefore, in such cases the distance should be inversely proportional to gradient strength, so that the path obtained by distance minimization is sticky to high gradients. In order to obtain it, the propagation direction should be taken into account, so that the distance is smaller if the path follows the edge. We define a *gradient distance* as:

$$dist_g(x,y) = 1 - \exp\left(-\frac{1}{h_g|\nabla Y(y)|(1+\cos\gamma)}\right), \tag{3}$$

where γ is an angle between the gradient vector in y and propagation direction from x to y. The normalization factor h_g has the same purpose as in (2). Here it was set to 0.5, taking into account that both distance metrics should be balanced with each other.

Propagation paths obtained by minimizing these two distances, as well as the distance used by Yatziv [12], are presented in Figs. 2 and 3. Fig. 2 shows propagation paths from a hair scribble to a grid of pixels rendered over a background image of gradient magnitude. Fig. 3 presents propagation paths to a selected pixel of human hair reached from two different scribbles added to hair and skin region. Total path cost and length are also given in the figure. The path leading from the hair scribble should not leave the hair region which is obtained only with the gradient distance (c). However, the path leading from the skin scribble is acceptable only for Yatziv (a) and plain (b) distance. In case of the gradient distance the path crosses an eye which is definitely incorrect.

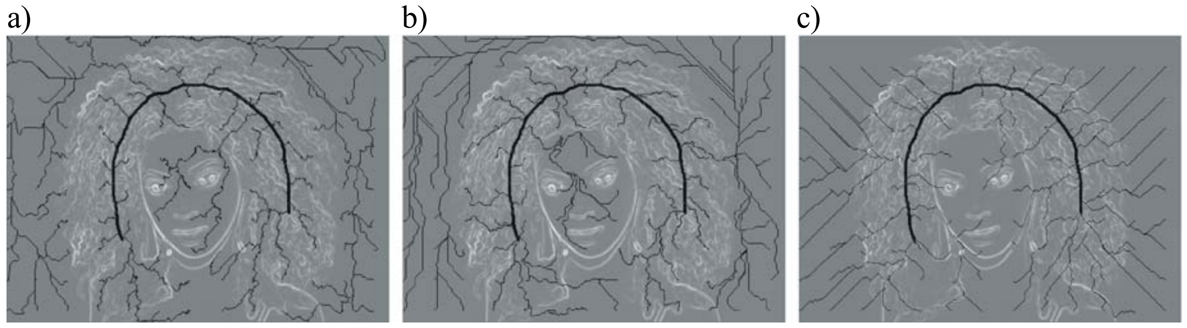

Fig. 2. Propagation paths determined based on: a) Yatziv's distance, b) plain distance, c) gradient distance

Rys. 2. Obraz ze ścieżkami wyznaczonymi na podstawie: a) odległości Yatziva, b) odległości gładkiej, c) odległości gradientowej

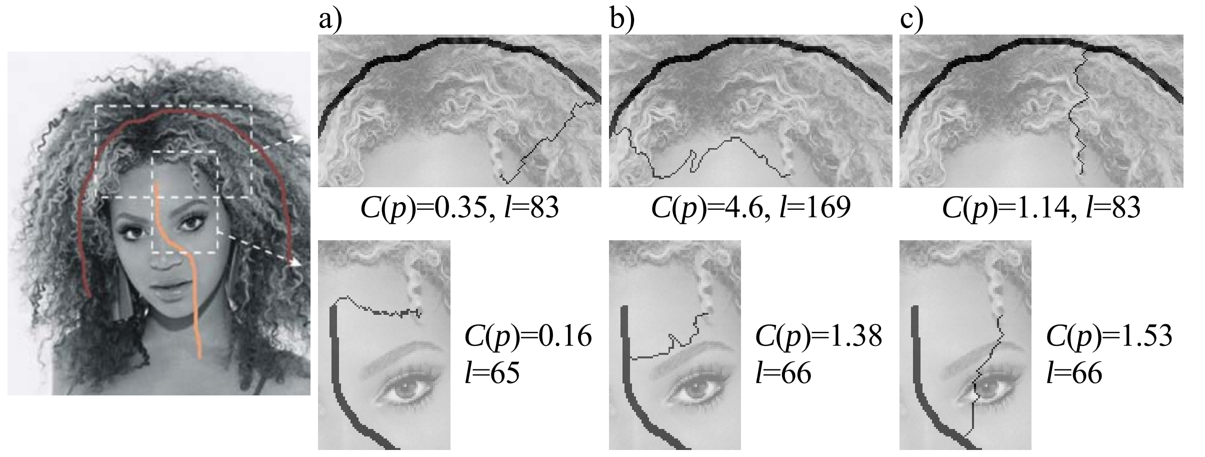

Fig. 3. A single point accessed by three different paths obtained with: distances: a) Yatziv's distance, b) plain distance, c) gradient distance

Rys. 3. Ścieżki wiodące do wybranego punktu otrzymane na podstawie: a) odległości Yatziva, b) odległości gładkiej, c) odległości gradientowej

This example shows clearly that the distance type used for determining a path should depend on a texture which is expected to be colorized. This choice may be done by a user who adds the scribbles and a potential colorization application may have various scribble brushes associated with certain distance metrics. However, in our method we provide automatic selection as well with a *competitive approach*. For every scribble we start the propagation algorithm with both types of paths and for each pixel we select that kind of path, with which the distance is smaller. Hence, for harsh surfaces the gradient paths usually prevail while on smooth areas the plain paths propagate better. This selection can be done either separately for every pixel or for a whole scribble.

We also investigated possibility of changing the path type dynamically during propagation. For every pixel the distance type was set to that one, for which the integrated value along the path was smaller. However, the experiments showed that it is better to preserve the same path type with the competitive approach rather than changing it dynamically.

3. Chrominance blending

Once the propagation paths are found, it is necessary to determine chrominance for every pixel in order to add colors to a grayscale image. Existing colorization methods which operate based on the distance transforms perform chrominance blending using the same distance with which the paths have been optimized. The final pixel chrominance is calculated as a weighted mean of scribbles' colors defined by a user and the weights are obtained as a function of the total path cost. Usually two or three strongest components are taken into account which provides a good visual effect of smooth color transitions.

This is a reasonable technique which usually gives satisfactory visual results. However, we found it better to separate the path optimization problem from chrominance blending, because proper propagation paths are a necessary, but not a sufficient condition of correct image colorization. We calculate the weights based on the image properties along the paths, but we do not use the same distances as for determining the paths. We calculate the final color value $v(x)$ of a pixel x in a similar way as in [12]:

$$v(x) = Y(x)\frac{\sum_i v_i w_i(x)}{\sum_i w_i(x)}, \tag{4}$$

where v_i is chrominance of i-th scribble and $w_i(x)$ is its weight in pixel x. We use YC_rC_b color space for the blending and we calculate color values separately for C_r and C_b channels. We obtain the weights with a formula:

$$w_i(x) = \left(C_i^b(x) + 1\right)^{-1}, \tag{5}$$

where $C_i^b(x)$ is a *blending distance* from i-th scribble to pixel x. The blending distance can be the same as those used for path optimization defined in Eq. (2) and (3), but we calculate it as:

$$C_i^b(x) = l/\sigma_i \sum_{j=0}^{l-1} dist\left(p_x(j), p_x(j+1)\right)^2 + \alpha l, \tag{6}$$

where σ_i is a *scribble strength* normalized from 0 to 1 and α equals 0.02. With this distance we add a *topological penalty* which causes shorter paths be preferred over the long ones.

Also we give a possibility of defining the scribble strength which is crucial for rough indication of region size that should be affected by the scribble. Colorization algorithms are low-level image operations and do not extract image features. In this case knowledge about the image content is user-exclusive and may be used for limiting the strength of scribbles added to small parts, e.g. human eyes.

Noteworthy, the same metric can be used for path optimization, but then the determined paths often make it impossible to colorize the image correctly. On the other hand, the distances used for the path optimization are often insufficient for correct image colorization. Therefore, by separating the local path distance from the blending distance, the method is more flexible and delivers better results.

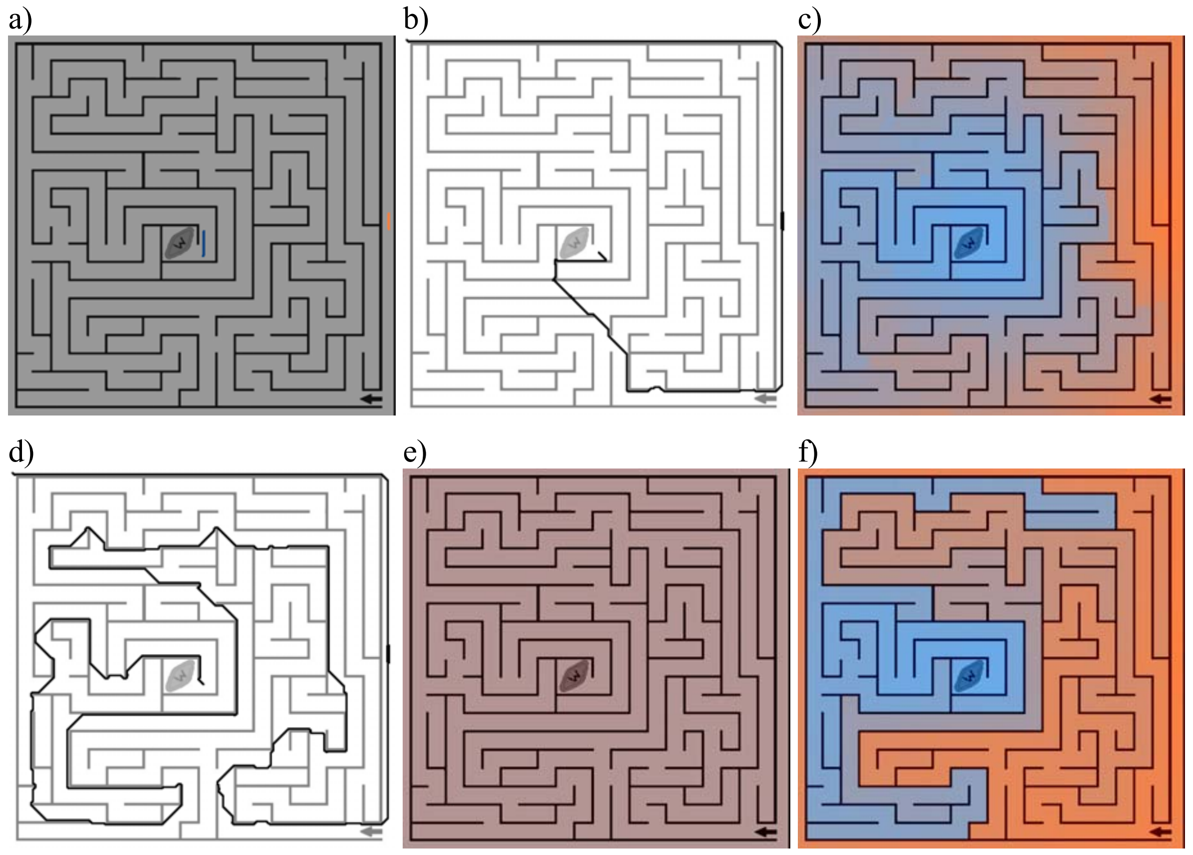

Fig. 4. Propagation paths and colorization results obtained for various local distances and blending distances

Rys. 4. Ścieżki propagacji oraz wynik koloryzacji uzyskany dla różnych odległości lokalnych oraz odległości używanych do mieszania barw

This is explained in Fig. 4, where an artificial maze image is colorized: (a) – original image with scribbles, (b), (c) – propagation paths to two selected pixels and colorization result obtained for local distance and blending distance calculated as in Eq. (6), (d) – paths obtained for local distance calculated with Eq. (1) and colorization result without (e) and with (f) topological penalty. The propagation paths are different in both cases and those determined with Eq. (6) do not allow to colorize the image correctly (c). On the other hand, if the blending distance is the same as the local distance, the blending weights are equal all over the image (e). Also the same effect was obtained with Levin and Yatziv methods. Only application of our blending technique gives the expected result.

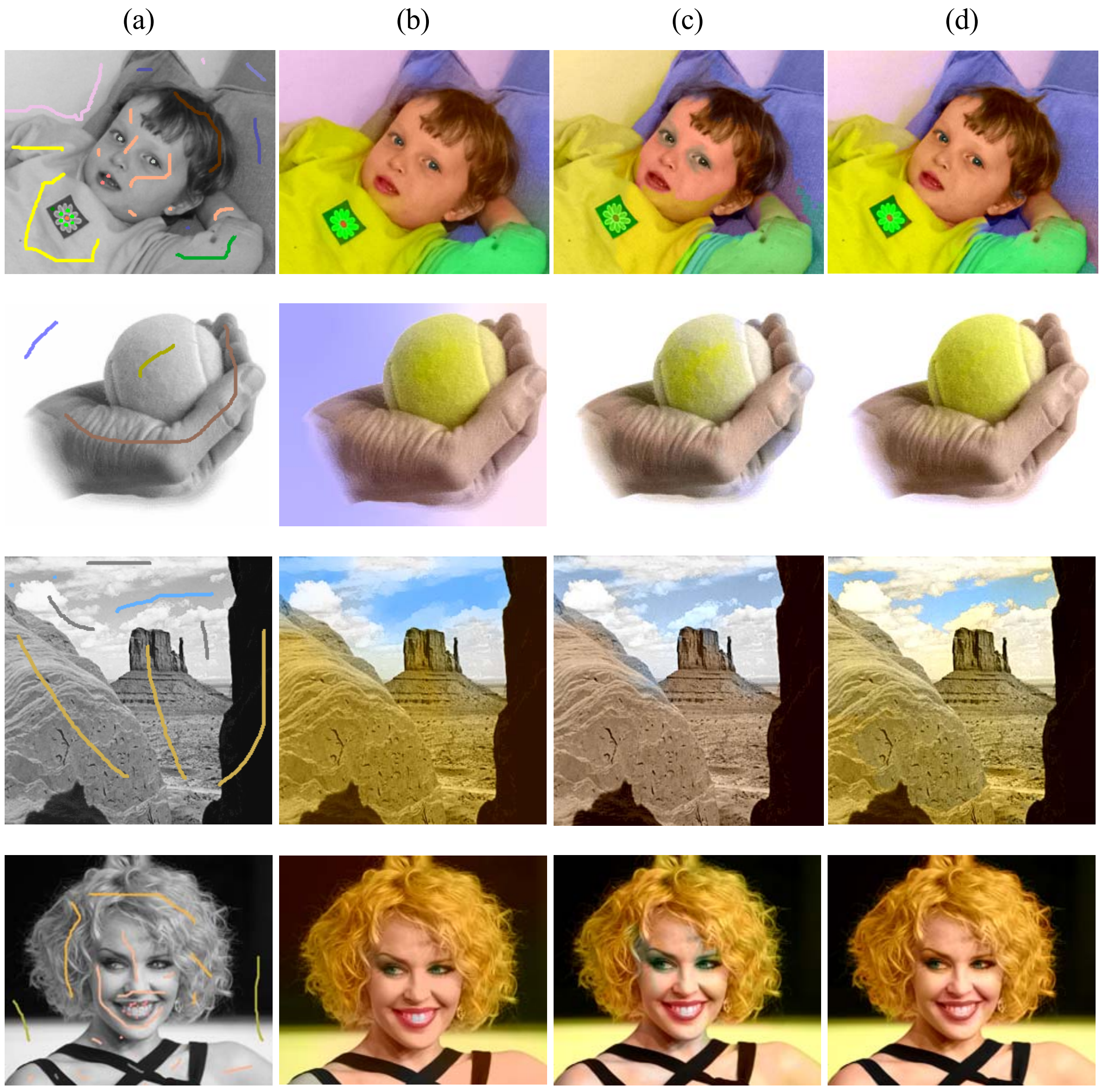

Fig. 5. Examples of colorization result: input image with: a) scribbles, b) results obtained with Levin’s, c), Yatziv’s, d) our algorithm (d)

Rys. 5. Przykłady koloryzacji obrazów: a) obraz wejściowy z mazami, b) wynik otrzymany metodą Levina, c) Yatziva, d) wg zaproponowanego algorytmu

4. Colorization results

The proposed method was compared with two well-established colorization techniques proposed by Levin [7] and Yatziv [12]. The first one is published in the form of MATLAB code and for the latter a Java applet is available to colorize a fixed set of images (for others we used our implementation of Yatziv's algorithm). We applied the competitive approach to choose between gradient and plain distance (section 2.2.). In majority of cases we set the scribble strength to 1, however we reduced it to 0.1 for eyes and lips. We evaluated the colorization only on the basis of the obtained visual result as this is commonly adapted practice in image colorization.

The colorization results are presented in Fig. 5 and 6. In Fig. 5 the subsequent columns show an original image with input scribbles (a) and colorization result obtained with the following algorithms: Levin's (b), Yatziv's (c) and ours (d). It may be observed that our method performs much better for areas with high gradients like human hair. The hair regions are almost perfectly segmented from the background which makes it possible to achieve very natural colorization result (the images of women in Fig. 5 and 6). Noteworthy, this cannot be achieved with two other methods, in which the hair scribble quickly loses with the background scribbles (Yatziv's result for the woman in Fig. 5 and 6 and Levin's result in Fig. 6). In case of the woman in Fig. 5 the hair scribble dominates the background with Levin's method (b) and hair also is incorrectly segmented. It may be also noticed that due to the topological penalty an image of the tennis ball (Fig. 5) is colorized better with our method than with the Yatziv's. Thanks to that improvement, the background scribble does not win in the hand area which is closer to the hand scribble. In case of Yatziv's method a very plain background implicates almost zero cost for scribble propagation.

In general, our method delivers the best visual results for the analyzed cases. Noteworthy, the result obtained with two other algorithms can be improved if more scribbles are added, but our assumption was to colorize images with as few scribbles as possible to achieve natural effect with minimal effort. The presented examples demonstrate that the colorization process can be definitely facilitated with the proposed method.

5. Conclusions and future work

In this paper we presented a new method for image colorization based on manually added scribbles. Our main contribution is utilization of two metrics of local distance for path optimization and competitive selection of the most appropriate one. Moreover, we proposed to make distinction between the path cost and blending weights which we consider to be two separate problems. The presented experimental results show that with our method image colorization is more intuitive and good visual results can be obtained by adding relatively few scribbles compared with other popular methods.

The main directions of possible improvement are concerned with more advanced methods for determining the blending weights and with colorization of image sequences. We are planning to investigate possibility of determining the blending weights based on statistical analysis of pixels along the path and also to utilize texture descriptors for that purpose. This may allow for creating decision rules used for the chrominance blending. Another challenge would be to colorize an image sequence based on a set of scribbles added to only one frame. Finally, the colorization techniques can be applied to multimedia databases, allowing them to store not only annotated images, but also their segmented parts which can be easily extracted and moved to other locations.

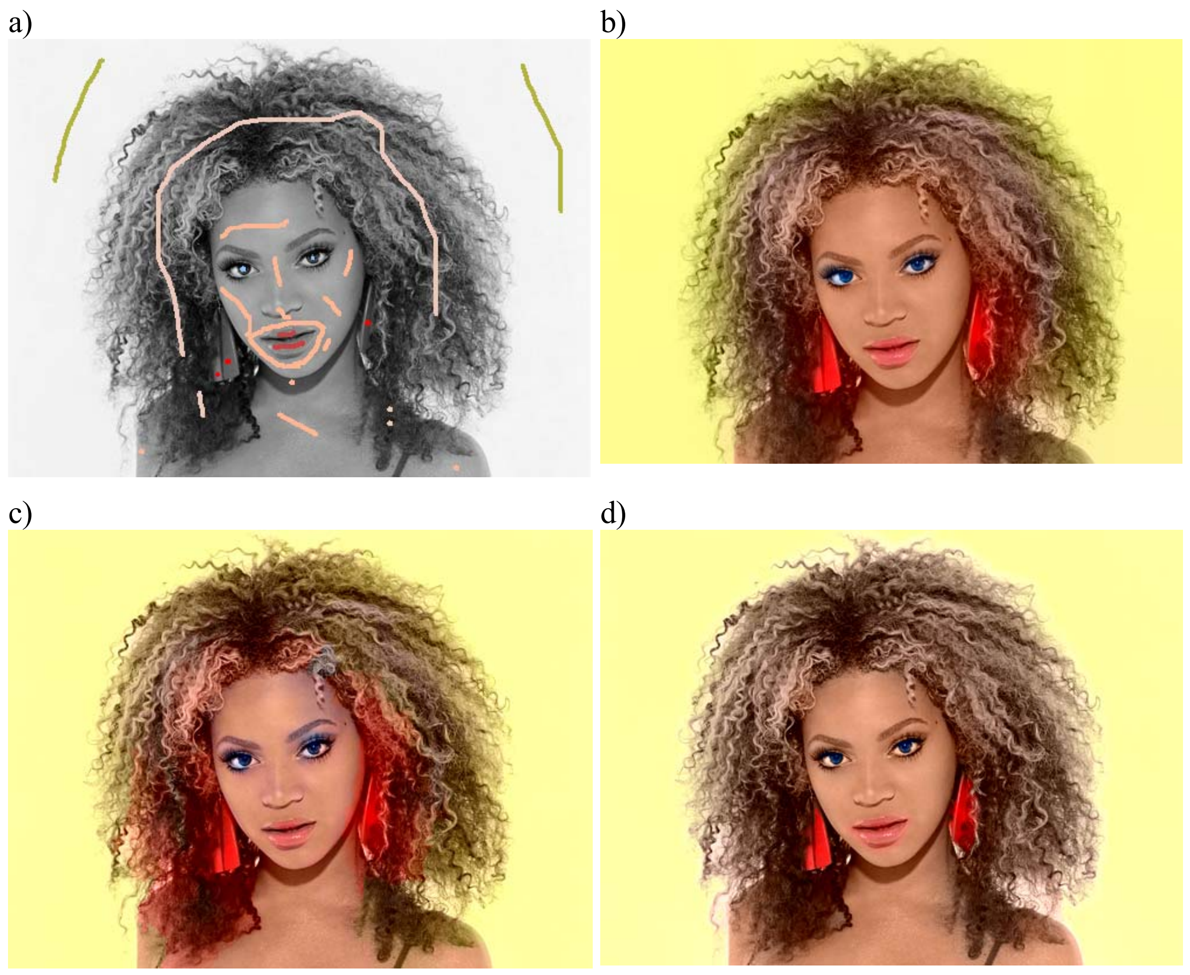

Fig. 6. Examples of colorization result with: b) Levin algorithm, c)Yatziv algorithm, d) our algorithm (d)

Rys. 6. Koloryzacja za pomocą metody: b) Levina, c) Yatziva, d) oraz zaproponowanej

BIBLIOGRAPHY

1. Dijkstra E.W.: A note on two problems in connexion with graphs. Numerische Mathematik, Vol. 1, 1959, pp. 269÷271.

2. Gonzalez R.C., Woods R.E.: Digital Image Processing, AddisonWesley Publishing, 1987.
3. Horiuchi T.: Colorization algorithm using probabilistic relaxation. Image Vision Comput., 2004, Vol. 22, No. 3, pp. 197÷202.
4. Ikonen L., Toivanen P.J.: Distance and nearest neighbor transforms on gray-level surfaces. Pattern Recognition Letters, Elsevier 2007, Vol. 28, No. 5, pp. 604÷612.
5. Kim T.H., Lee K.M., Lee S.U.: Edge-preserving colorization using data-driven random walks with restart, in: Proc IEEE Int. Conf. on Image Processing 2009, pp. 1661÷1664
6. Konushin V., Vezhnevets V.: Interactive Image Colorization and Recoloring based on Coupled Map Lattices. Proc. GraphiCon 2006, pp. 231÷234.
7. Levin A., Lischinski D., Weiss Y.: Colorization Using Optimization. Proceedings of ACM SIGGRAPH 2004, pp. 689÷694.
8. Mortensen E.N., Barrett W.A.: Interactive Segmentation with Intelligent Scissors. Graphical Models and Image Processing, 1998, Vol. 60, No. 5, pp. 349÷384.
9. Lagodzinski P., Smolka B.: Digital Image Colorization Based on Probabilistic Distance Transform, LNCS 5197, Springer-Verlag 2006, pp. 626÷634.
10. Sykora D., Burianek J., Zara J.: Unsupervised Colorization of Black-and-White Cartoons. Proceedings of ACM SIGGRAPH 2004, pp. 121÷127.
11. Welsh T., Ashikhmin M., Mueller K.: Transferring color to greyscale images. ACM Trans. Graph. (TOG), 2002, Vol. 21, No. 3, pp. 277÷280.
12. Yatziv L., Sapiro G.: Fast Image Video Colorization Using Chrominance Blending. IEEE Transactions on Image Processing, Vol. 15, No. 5, 2006, pp. 1120÷1129.

Recenzenci: Dr inż. Dariusz Mrozek,
Dr hab. inż. Ewa Piętka, prof. Pol. Śląskiej

Wpłynęło do Redakcji 31 stycznia 2010 r.

Omówienie

Koloryzacja obrazów jest procesem polegającym na dodawaniu barwy do obrazu monochromatycznego. Może być on realizowany automatycznie, aczkolwiek najczęściej jest wykonywany przy wsparciu człowieka. Głównym celem koloryzacji jest podniesienie atrybutów estetycznych obrazu, zdjęcia bądź sekwencji wideo, których barwne wersje nie są dostępne. Ponadto, algorytmy koloryzacji są również wykorzystywane w celu segmentacji

lub wycinania fragmentów obrazu (tzw. matting), co może znaleźć zastosowanie w rozszerzeniu funkcjonalności multimedialnych baz danych.

Koloryzacja dokonywana jest najczęściej przy wsparciu użytkownika, który nanosi na obraz mazy o pożądanym kolorze. Następnie mazy są propagowane w obrazie, a każdemu pikselowi przypisywana jest barwa na podstawie odpowiednio zdefiniowanej odległości od mazów oraz oryginalnej wartości w kanale luminancji.

W ramach opisanych badań została zaproponowana nowa metoda wyznaczania ścieżek propagacji barwy oraz mieszania barw na podstawie odległości od poszczególnych mazów. Ścieżki są wyznaczane z wykorzystaniem algorytmu Dijkstry na zasadzie minimalizacji sumy odległości lokalnych pomiędzy sąsiadującymi pikselami. Przeprowadzone badania pozwoliły na wyciągnięcie wniosku, że sposób liczenia odległości lokalnych powinien być uzależniony od rodzaju tekstury, w ramach której ma być dokonywana propagacja. W związku z tym zostały zaproponowane dwie metryki oraz podane kryterium doboru najbardziej odpowiedniej dla koloryzowanej tekstury. Ponadto, oryginalnym wkładem autorów jest oddzielenie odległości służącej do optymalizacji ścieżki od odległości wykorzystywanej do mieszania barw, co zdecydowanie poprawia rezultat końcowy koloryzacji.

W ramach badań eksperymentalnych dokonane zostało porównanie zaproponowanej metody z dwiema innymi wiodącymi metodami koloryzacji. Pozwoliło ono na analizę mocnych stron metody oraz na wyznaczenie kierunków dalszych prac nad udoskonaleniem algorytmu. Wśród nich należy wymienić analizę statystyczną pikseli wzdłuż każdej ze ścieżek, wykorzystanie deskryptorów tekstury do obliczania wag przy mieszaniu barw, a także koloryzację całych sekwencji obrazów.

Adresses

Michal KAWULOK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, mkawulok@polsl.pl .

Bogdan Smolka: Politechnika Śląska, Instytut Automatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, bsmolka@polsl.pl .