Volume xx (*200y*), *Number z, pp. 1–14*

Bayesian Scheme for Interactive Colorization, Recolorization and Image/Video Editing

Oscar Dalmau[1] and Mariano Rivera[1] and Teresa Alarcón[2]

[1] Centro de Investigación en Matemáticas, A.C. Jalisco S/N, Colonia Valenciana C.P. 36240,Guanajuato, Gto, México
[2]Centro Universitario de los Valles, Universidad de Guadalajara. Carretera Guadalajara Ameca Km. 45.5 C.P. 46600. Ameca, Jalisco, México

Abstract

We propose a general image and video editing method based on a Bayesian segmentation framework. In the first stage, classes are established from scribbles made by a user on the image. These scribbles can be considered as a multimap (multilabel map) that defines the boundary conditions of a probability measure field to be computed for each pixel. In the second stage, the global minima of a positive definite quadratic cost function with linear constraints, is calculated to find the probability measure field. The components of such a probability measure field express the degree of each pixel belonging to spatially smooth classes. Finally, the computed probabilities (memberships) are used for defining the weights of a linear combination of user provided colors or effects associated to each class. The proposed method allows the application of different operators, selected interactively by the user, over part or the whole image without needing to recompute the memberships. We present applications to colorization, recolorization, editing and photomontage tasks.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications—Image Colorizarion, Recolorizaction

1. Introduction

In this paper, we propose an interactive method for image/video colorization and editing. Our method is based on a probabilistic Bayesian framework for image segmentation. Our strategy consists of applying a particular image transformation to each segmented region. We demonstrate that our framework is very general, it can be applied to monochrome or color images, as well as to video. Our technique accepts different kind of transformations: colorization, recolorization, tonal transformations, artistic effects and geometric transformations.

Our proposal extends and generalizes our previous conference paper [DRM07], in which we proposed an interactive method for the particular task of colorization. The principal contributions of this paper are:

1. we improve the colorization operator proposed in our previous work [DRM07],
2. we extend the previous operator to other image and video editing tasks,
3. we introduce *the distance* as a feature for probabilistic segmentation methods,
4. we introduce an off-line entropy control for probabilistic segmentation methods.

Unlike reported approaches for image editing, we present an interactive framework for gray and color image/video editing tasks. The criterion we use for extracting similar regions is based on a similarity measure given by probability distributions. In the case of colorization (or recolorization), our method assumes that regions with similar intensity (color) distributions should have similar colors. In general, the same transformation is applied to regions with similar distribution.

Recent works for image editing are reported in Refs. [LLW04, WC05, LFUS06, YS06, DRM07, AP08]. Most of them tackle a particular image editing task. Some solutions to the matting problem have been proposed in Refs. [WC05, RKB04, WC05, LRAL08]. Lischinski et al. present an interactive technique for the local adjustment of tonal values [LFUS06]. An and Pellacini present an inter-

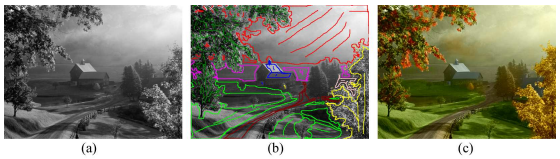

Figure 1: *Interactive colorization using multimaps for defined regions. (a) Original gray scale image (luminance channel), (b) multimap, (c) colored image in which the chrominance channels are provided by the user. The colorization was achieved with the method presented in this paper.*

active editing method for tonal adjustment, and for changing the appearance (low and high dynamic range) of color images [AP08]. Both methods use a propagation strategy based on a quadratic functional. Yatziv and Sapiro present an approach close related with our scheme, see Ref. [YS06]. They compute a layered map of geodesic distances from each pixel to the scribbles. Another particular editing task is image colorization. This technique consists of introducing color to grayscale, sepia or monochromatic images, for instance see Refs. [LLW04, SBv04, YS06, DRM07]. In spite of the fact that many colorization algorithms have been developed in recent years [RAGS01, WAM, Hor02, BR03, CWSM04, WH04, LLW04, SBv04, QG05, TJT05, KV06, YS06, QWCO*07, DRM07], only a few of them are based on a segmentation procedure [CWSM04, TJT05, KV06, DRM07]. The reason seems to be, for coloring purposes, that regions to be segmented come from different groups: faces, coats, hair, forest, landscapes and, up to now, there has not been a proficient method that automatically discerns among this huge range of features. In the absence of a general purpose automatic segmentation algorithm, other alternatives have appeared in recent years and techniques that use human interaction (semi-automatic algorithms) have been introduced as part of computer vision algorithms. In interactive colorization procedures one uses a gray scale image as the luminance channel and estimates the chrominance for each pixel. This is achieved by propagating the colors from user labeled pixels. The process is illustrated in Fig. 1. One can find two main groups in colorization methods. In the first group one can set those methods in which the color is transferred from a source color image to a greyscale one using some corresponding criteria [RAGS01, WAM, Hor02]. In the second group are the semiautomatic algorithms that propagate the color information provided by a user in some regions of the image [LLW04].

Among automatic methods we can mention the following works. Reinhard et al. stated the basis for transferring color between digital images [RAGS01]. Afterwards, Welsh et al. extended the previous method for colorizing greyscale images using a scan–line matching procedure [WAM]. In that technique, the chromaticity channels are transferred from the source image to the target image by finding regions that best match their local mean and variance of the luminance channel . In order to improve the last method, Blasi and Recupero proposed a sophisticated data structure for accelerating the matching process: *the antipole tree* [BR03]. Chen et al., in Ref. [CWSM04], proposed a combination of composition [PD84] and colorization [WAM] methods. First, they extract objects from the image to be colorized by applying a matting algorithm, then each object is colorized using the method proposed by Welsh et al., and in the last step, they make a composition of all objects to obtain the final colorized image. Tai et al. treat the problem of transferring color among regions of two natural images [TJT05].

Automatic color transfer methods will equally transfer color between regions with similar luminance features (gray level mean, standard deviation or higher–level pixel context features, as in [ICOL05]). However, this approach may fail in many cases. For instance, consider the simple case in which the gray mean is the used feature, then a grass field and a sky region may have similar gray scale values, but different colors. Therefore, in general, colorization requires high–level knowledge that can only be provided by the final user: a human. For this reason, semiautomatic methods have successfully been used in image colorization. These methods take advantage of user interaction for providing high level knowledge [LLW04, KV06, YS06, DRM07]. For instance, Levin et al. presented a novel interactive method for colorization based on an the minimization of a free-parameter cost functional [LLW04].

Video colorization is, in general, implemented as an hybrid technique: the user provides scribbles for some frames and afterward such information is propagated to the reminder frames. Previous solutions to this problem are reported in Refs. [LLW04, SBv04, YS06]. The method in Ref. [YS06] uses the idea of geodesic distance in volumetric data (*i.e.*with the time as the third dimension). In spite of the method presents good results it is limited to work with sequences with small displacements: videos with fixed background (without camera movements) and short movements of objects. Another approach to video colorization is presented in Ref. [LLW04]. This approach uses optical flow (Lukas and Kanade, [LK81]) information for redefining 3D pixels neighboring.

In this paper we propose an interactive image/video editing framework. Our method is general enough for implementing very different editing process for image and video. The paper is organized as follows. Section 2 presents a description of the general scheme. In order to simplify our presentation, we study the particular case of colorization in Section 3. Then the generalization and extensions to other editing effects are presented in Section 4. Section 5 presents an application for video colorization. That application illustrates how to combine intensity and spatial distance features

in our framework. Section 6 shows experimental results and, finally, we present our conclusions in Section 7.

2. Description of the Proposed Scheme

Our proposal uses a segmentation procedure based on the minimization of a quadratic cost function with well-known convergence and numerical stability properties [ROM07, RDT08]. The scheme can be summarized in the following steps:

1. Feature learning,
2. Image layering,
3. Layering process and composition operator.

In this scheme, the interactive process is concentrated in the feature learning stage, see details in Subsection 3.1. Basically, some scribbles are made by a user on an image, these scribbles should describe regions on which some transformation will be applied. We have then, a set of colors used to make the scribbles in some regions of the image. This color set is named the *label palette*. Moreover, we have an operator bank composed by filters (for example: directional blurring), enhancement operators, tone transfer functions [GW02], geometric transformations (rotation, scaling, etc), artistic effects [Hol88, McA04], a second set of colors named the *colorization palette*, in the case of colorization. How many colors (for the *label palette*) and similarly, how many operators (in the *operator bank*) to use is decided by the user. The *label palette* is actually formed by colors that are considered as labels, and will be used for the learning process. The feature learning process consists of achieving empirical distributions of certain features of the image, for example in our experiments we use color (intensity) empirical histograms on regions marked by the user, although other local characteristics could also be included.

The second step, image layering, conceptually consists of a soft segmentation of the image, or a sequence of images, in regions with an assumed similar color histogram, or in general it is assumed that regions to be transformed have similar distribution of certain feature: intensity, color, distance, etc. This approach requires achieving a robust segmentation method that is, in itself, a challenging problem. Afterwards, comes the Layering process. In this step, the bank of operators will be used. Then, the layering process depends on the selected operators or on the specific editing task. Finally, a composition operator is applied to combine the transformed layers.

3. Image Colorization

In this section, we present our proposal scheme for colorization task, thoroughly explaining all its details and benefits. In the next sections we extend this scheme to recolorization and other interactive tasks.

In the image colorization task the Layering process, and Composition operator of the general scheme presented in Section 2 are combined in one step that we call here *Transferring color operator*. First, some scribbles are made by a user on a gray scale image, these scribbles should describe regions that will have the same color. The operator bank is actually the *colorization palette* that will be used to colorize the image. Therefore, we have two color palettes with the same number of colors, the first one containing colors used to make the scribbles (*label palette*) and the second one containing colors to colorize the image (*colorization palette*), see Fig. 2 (c). In the second step, we apply a probabilistic image segmentation method and, finally, in the last step, we need to build an appropriate transferring color function and assign color to each detected region.

3.1. Feature learning

The introduced notation in this section will be preserved for rest of the article. We consider the segmentation case in which some pixels in the region of interest, Ω, are labeled by hand in an interactive process. Assuming $\mathcal{K}$ as the class label set, we define the pixels set (region) that belongs to the class k as $R_k = \{r : \mathcal{R}(r) = k\}$, and

$$\mathcal{R}(r) \in \{0\} \cup \mathcal{K},\ \forall r \in \Omega, \tag{1}$$

is the label field (class map or *multimap*) where $\mathcal{R}(r) = k > 0$ indicates that the pixel r is assigned to the class k and $\mathcal{R}(r) = 0$ if the pixel class is unknown and needs to be estimated. Let g be an intensity image such that $g(r) \in t$, where $t = \{t_1, t_2, \ldots, t_T\}$ are discrete intensity values. Let $h_k(t) : \mathbb{R} \to \mathbb{R}$ be the intensity empirical histogram on the marked pixels which belong to class k, where

$$h_k(t) = \frac{\sum_{r \in R_k} \delta(g(r) - t)}{|R_k|} \tag{2}$$

is the ratio between *the number of pixels in R_k whose intensity is t* and *the total number of pixels in the region R_k*. We smooth $h_k(t)$, see [DHS01, HTF09], and denote as $\hat{h}_k(t)$ the smoothed normalized histograms (i.e. $\sum_t \hat{h}_k(t) = 1$) of the intensity values, then the likelihood of the pixel r to a given class k is computed with:

$$v_k(r) = \frac{(\hat{h}_k \circ g)(r) + \varepsilon}{\sum_{j=1}^{K} \left[(\hat{h}_j \circ g)(r) + \varepsilon\right]},\ \forall k > 0; \tag{3}$$

where $\circ$ is the composition operator, i.e. $(\hat{h}_k \circ g)(r) = \hat{h}_k(g(r))$, ε is a small positive real value, we used $\varepsilon = 1 \times 10^{-4}$. Note that, although parametric models (such as Gaussian Mixture Models) can be used for defining the likelihood functions, we used smoothed normalized histograms because they are computationally more efficient, i.e. they are implemented as look up tables. In our experiments we are just using the pixel intensity as local feature, however other local statistics could be taked into account, as the local mean of the intensity values on a pixel neighborhood.

3.2. Image layering

Now the task is to compute the probability measure field α at each pixel such that $\alpha_k(r)$ is the probability of the pixel r to be assigned to the class k. Such a probability vector field α must satisfy:

$$\sum_{k=1}^{K} \alpha_k(r) = 1, \tag{4}$$

$$\alpha_k(r) \geq 0, \quad \forall k \in \mathcal{K}, \forall r \in \Omega, \tag{5}$$

$$\alpha(r) \approx \alpha(s), \quad \forall r \in \Omega, \forall s \in \mathcal{N}_r, \tag{6}$$

where $\mathcal{N}_r$ denotes the set of the first neighbors of r: $\mathcal{N}_r = \{s \in \Omega : |r-s| = 1\}$. Note that, the conditions in Eqs. (4)–(5) constrain α to be a probability measure field and the condition in Eq. (6) to be spatially smooth. The probability measure field α allows us to divide the image in layers in correspondence to the established classes.

Although our framework admits any probabilistic segmentation method [MVRN01, MAB03, ROM07] we select a Quadratic Markov Measure Field (QMMF) model [ROM07] because the solution conducts to a linear system. According to [ROM05, ROM07], the smooth image multiclass segmentation is formulated as the maximization of the posterior distribution, that takes the form $P(\alpha|R,g) \propto \exp[-U(\alpha,\theta)]$ and the maximum a posteriori (MAP) estimator is computed by minimizing the cost function:

$$U(\alpha) = \sum_r \Big\{ \alpha(r)^T \mathbf{D}(r)\alpha(r) + \frac{\lambda}{2} \sum_{s \in \mathcal{N}_r} w_{rs} ||\alpha(r) - \alpha(s)||_2^2 \Big\}, \tag{7}$$

subject to the constraints in Eqs. (4)–(5); where

$$\mathbf{D}(r) = -\text{diag}\{\log v_1(r), \log v_2(r), \ldots, \log v_K(r)\}$$

is a definite-positive diagonal matrix, i.e. $v_k(r) \in (0,1)\ \forall r \in \Omega, k \in \mathcal{K}$.

The soft constraint in Eq. (6) is enforced by introducing a Gibbsian prior distribution based on Markov Random Field (MRF) models [GG84, Li01, WG04] and the spatial smoothness is controlled by the positive parameter λ in the regularization potential (second term in Eq. (7)), where the weight $w_{rs} \approx 0$ if a class edge is probably allocated between the pixels r and s, otherwise $w_{rs} \approx 1$. As the weight function, see [DRM07], we use

$$w_{rs} = \frac{\beta}{\beta + |g(r) - g(s)|^2}, \tag{8}$$

where β is a small positive value, i.e. $\beta = 10^{-3}$.

The convex quadratic programming problem in Eq. (7) subject to the constraints in Eqs. (4)–(5) can efficiently be solved by using the Lagrange multiplier procedure for the equality constraint, see Eq. (4). Unlike the proposal in [ROM05], we are not penalizing the $\alpha(r)$'s entropy, and thus the energy functional in Eq. (7) is convex. Therefore, we guarantee convergence to the global minima see [ROM05, ROM07]. For our purposes, we have found that the mode (hard segmentation computed as the winner–take–all) of the solution should be correct. The entropy, that controls the smooth transition between classes, can be adjusted off-line at the composition stage, see Section 3.3.

3.3. Transferring color operator

Once the measure field, α, is computed, colors $C_k = [R_k, G_k, B_k]^T$ in *RGB* color space are selected by the user to form the *colorization palette*, thereafter every selected color is assigned to a color (or class) in the corresponding *label palette* by the user. Then, the color C_k is converted into a color space in which the intensity and the color information are independent. For example, the color spaces: $L\alpha\beta$ [RCC98, RAGS01], *YIQ* [GL00, WOZ02, YK03, GWE04], *YUV* [WOZ02, GWE04], $I_1I_2I_3$ [GL00], *HSV* [GWE04], CIE-*Lab* and CIE-*Luv* [WS82]. In general, we denote the transformed color space by LC_1C_2:

$$\begin{pmatrix} L_k \\ C_{1k} \\ C_{2k} \end{pmatrix} = \mathcal{T} \begin{pmatrix} R_k \\ G_k \\ B_k \end{pmatrix}, \tag{9}$$

where L_k is the luminance component, C_{1k}, C_{2k} are the chrominance components for the class k; $\mathcal{T}$ is the applied transformation. Such a transformation is linear for the *YIQ*, *YUV*, $I_1I_2I_3$ spaces. For the CIE, $l\alpha\beta$ and *HSV* spaces, the transformation $\mathcal{T}$ is non–linear. The reader can find details of the color transformations used in this paper in Refs. [RCC98, WS82, OKS80]. For computing the color component (to colorize) at each pixel, we obtain the components $l(r)$, $c_1(r)$ and $c_2(r)$ ($\forall r \in \Omega$) in the LC_1C_2 color space, see Eqs. (10)–(12). We based our colorization operator on the one proposed by Dalmau et al. in [DRM07]. Unlike that work, we propose to change the luminance component of the original image. The color components are obtained as a linear combination of the chrominance components C_{1k} and C_{2k}. Our colorization operator is defined as:

$$l(r) = (f \circ \hat{l})(r), \tag{10}$$

$$c_1(r) = \sum_{k=1}^{K} \hat{\alpha}_k(r) C_{1k}, \tag{11}$$

$$c_2(r) = \sum_{k=1}^{K} \hat{\alpha}_k(r) C_{2k}; \tag{12}$$

where $\hat{l}$ is the first component of $\mathcal{T}(g)$, i.e. $\hat{l}$ is the luminance component of the gray intensity image g, and f is a transformation function of the luminance component, i.e. an enhancement operator, a piecewise-linear transformation, a tone transfer function or an intensity transformation function [GW02]. The introduction of the function f is very important because now we can modify the luminance channel of the original image, and for instance, in colorization task,

it allows us to colorize dark region with lighter colors. Subsection 4.1 presents a procedure for estimating f for adapting the image intensity to be close to the one of the selected color.

The components $\hat{\alpha}_k(r)$ of $\hat{\alpha}(r)$ can be understood as the contribution (matting factor) of the class color C_k to the pixel r. The matting factors are computed with:

$$\hat{\alpha}_k(r) = \frac{\alpha_k^n(r)}{\sum_{j=1}^{K} \alpha_j^n(r)}, \tag{13}$$

where the n-factor allows us to control the color transition smoothness (off-line entropy control), for $n \to \infty$ the colorization is achieved with flat colors such as in a cartoon. According to our experiments the entropy control can easily and effectively be adjusted with the power n. Note that, because of Eq. (4), for $n = 1$, one has $\hat{\alpha}_k(r) = \alpha_k(r)$.

Finally, the colored image $\tilde{g}$ is transformed into the *RGB* color space by applying the corresponding inverse transformation:

$$\tilde{g}(r) = \mathcal{T}^{-1} \begin{pmatrix} l(r) \\ c_1(r) \\ c_2(r) \end{pmatrix}. \tag{14}$$

Observe that the step of assigning color to each region R_k is completely independent of the segmentation stage. It means that, once we have computed the vector measure field, α, for the whole image, we can reassign colors to one or more regions by just recomputing the color components with Eqs. (11) and (12), and transforming the image with the Eq. (14). Also it may be possible to assign the same color to different regions. This makes the proposed method very versatile.

In summary, the process of colorization begins by making a "hard" pixel labeling from the user marked regions (multimap): for every $r \in R_k$ with $k > 0$, we set $\alpha_k(r) = 1$ and $\alpha_l(r) = 0$ for $l \neq k$. Next, the remainder pixels are "soft" segmented by minimizing the functional in Eq. (7). Fig. 2 illustrates the colorization process. Panel (a) shows the original gray scale image, the scribbles for 6 classes are shown in Panel (b) and the colored image in Panel (d). The computed class memberships (probabilities or α_k layers) are shown in Fig. 3. Finally the color assignment to each region is done by applying Eqs. (10)–(12) in LC_1C_2 color space followed by the inverse transformation into RGB color space. Note that smooth inter–region probability transitions produce smooth color transitions (as in the face regions) and, conversely, sharp intensity edges produce sharp color transitions.

4. Extension and Generalization of the Colorization Scheme

In this section we extend our colorization scheme to image and video editing. The first subsection presents a procedure for adjusting the luminance component according to the selected *colorization palette*. The next two subsections present

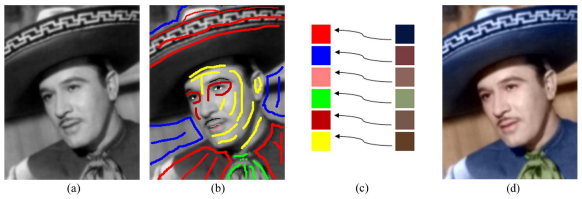

Figure 2: *Result of colorization using 6 classes. (a) Grayscale image, (b) scribbled image, (c) palettes link (d) colorized image.*

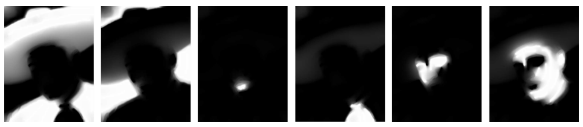

Figure 3: *Colorization using 6 classes. Each image represents a component of the vector measure of the probability of each class(α_k layer).*

our extension for image recolorization. The following subsection generalizes the *transferring color operator* to other operators. The final subsection presents a mechanism for including in our framework other image features, as for instance shortest path distance.

4.1. Luminance adjustment

One problem of the colorization methods in [DRM07, LLW04, YS06], is that they only use the chrominance components of the colors provided by the user. So, in many cases, the resulting colorized image has regions whose colors are very different from those given in the colorization palette. This is due to, the values of the luminance channel in these regions mismatched with the corresponding luminance channel of the user given color. In this subsection we study a particular transformation function of the luminance component $\hat{l}$ of the image. This function is of the form:

$$f_k(\hat{l}) = \begin{cases} \mathbf{0}, & a_k\hat{l} \leq \mathbf{0}; \\ a_k\hat{l}, & \mathbf{0} < a_k\hat{l} < \mathbf{M}; \\ \mathbf{M}, & a_k\hat{l} \geq \mathbf{M}; \end{cases} \tag{15}$$

where '**M**' is the maximum value of the luminance component in the LC_1C_2 space and a_k is a scalar value (luminance gain control) for each class. The aim is to obtain at each region to be colorized a color as similar as possible to that given by the user, in the *colorization palette*. Although a_k could be manually adjusted, we found that this gain control a_k can automatically be computed by solving the optimization problem:

$$a_k = \arg\min_a \sum_{r \in \Omega_k} \left(a\hat{l}(r) - L_k\right)^2, \text{ s.t: } 0 \leq a \leq b_k, \tag{16}$$

where Ω_k is the region of the image that corresponds to the class k, $b_k = \frac{\mathbf{M}}{\max_{r\in\Omega_k}\hat{I}(r)}$ (with $\hat{I}(r) > 0$) and L_k is the luminance component of the k-th color in the *colorization palette*, see Eq. (9). The solution to (16) is given by the closed formula:

$$a_k = \min\left\{b_k, \frac{L_k \sum_{r\in\Omega_k}\hat{I}(r)}{\sum_{r\in\Omega_k}\hat{I}^2(r)}\right\}.$$

4.2. Semi-Automatic Recolorization

The recolorization task consists of changing colors in the original image. Unlike colorization in which g is a gray level, now g represents a color image. A common approach to such a task is based on the definition of a context independent mapping function from the original colors to the new ones. This simple approach, although computationally efficient, fails with noisy images or when similar colors are mapped to different colors. Here, we propose two possible extensions of the above presented colorization scheme for recolorization task, that is robust to the above mentioned problems.

The Semi-Automatic or interactive Recolorization is a straightforward extension of colorization task. The difference is that now, the image g is given in the *RGB* color space, so $g(r) \in t$, where $t = \{t_1, t_2, \ldots, t_T\}$ are vectorial values that correspond to channels in the *RGB* color space. Therefore, the density distributions for each class are then empirically estimated in *RGB* color space by using a histogram technique, similar to the colorization case, but now in $\mathbb{R}^3$, *i.e.* $h_k(t) : \mathbb{R}^3 \to \mathbb{R}$, where $h_k(t) = \frac{\sum_{r\in R_k}\delta(g(r)-t)}{|R_k|}$ is the ratio between *the number of pixels in R_k whose color is equal to t* and *the number of pixel in the region R_k*, see Ref. [DHS01, HTF09]. Then, we can compute the likelihood of each pixel r belonging to class k. That is, let $\hat{h}_k : \mathbb{R}^3 \to \mathbb{R}$ be the smoothed normalized histograms ($\sum_t \hat{h}_k(t) = 1$) of vectorial values, then likelihood of the pixel r to a given class k is computed by using the Eq. (3).

4.3. Quasi-Automatic Recolorization

We present a variant for colorization that reduces the user interaction. Our method relies on the assumption that for a particular class of images the likelihood density functions have previously been learned. The instance we present is the case in which the classes correspond to linguistic colors: brown, green, orange, red, etc [BK69]. In Ref. [AM09] Alarcon and Marroquin proposed a segmentation method based on a combination between a color categorization method and Bayesian technique. The elaborated color categorization model describes each voxel in the color space L*u*v* as a vector of probabilities, whose components express the degree to which the voxel belongs to one of the eleven color categories established by Berlin and Kay [BK69] in 1969, as basic and universal. Alarcon and Marroquin obtained the color categorization model by an interactive technique. They performed a color naming experiment considering 336 color samples and 32 subjects. Each subject was instructed to select the most likely color basic category k for the observed color sample c. With this procedure the likelihood, $P(k|c)$, for each shown color sample is calculated, using the expression [AM09]:

$$P(k|c) = \frac{A_{kc}}{M}, k = 1, \ldots, K; \tag{17}$$

where A_{kc} denotes the number of assignments to the class k for the color sample c, and M is the total number of subjects; K is the number of the used color basic categories and it was considered $K = 11$. The number of color classes included in [AM09] is non arbitrary, but is based on relevant findings about the human color naming process, reported by Berlin and Kay [BK69]. Nevertheless, the color model proposed by Alarcon and Marroquin, as specified in [AM09], can include different categories, respect to those established by Berlin and Kay. More details related to the color model elaboration are in Ref. [AM09]. As a result of the color naming experiment done in Ref. [AM09] the likelihood, $P(k|c)$, for a limited number of color samples in the color space was calculated. In order to know the likelihood for the whole space, Alarcon and Marroquin [AM09] used an interpolation procedure in which each category is modeled as a linear combination of quadratic splines. After the interpolation process, the likelihood function $\tilde{P}(k,c)$ is obtained, *i.e.* the likelihood $\tilde{P}(k,c)$ is known for each category k and for all colors c in the color space. The computed solution is interpreted as a probabilistic measure field (a probabilistic dictionary), that is used in Ref. [AM09] as likelihood in a segmentation Bayesian technique. From the above, the obtained color model gives a quasi-automatic segmentation method, that could be applied in many image processing tasks. Color video restoring and video recolorization are examples of these tasks, in which the use of this technique leads to the considerably reduction of the human effort. The quasi-automatic recolorization approach, proposed in this manuscript, considers the color categorization model $\tilde{P}$, see [AM09], and the algorithm is the following:

Algorithm 1 Quasi-Automatic Recolorization

Require: I (color image) and $\tilde{P}$ (color categorization model)

1: Set the likelihood v using $\tilde{P}$, i.e. $v_k(r) \leftarrow \tilde{P}(k|I(r))$, for all k, r.
2: Probabilistic segmentation, i.e. Minimize the functional in Eq. (7) subject to the constraints in the Eqs. (4)–(5).
3: Apply the Colorization operator, see Eqs. (10)–(12).

Note that the segmentation (steps 1 and 2) is completely automatic. The semiautomatic part is hidden in the color naming experiment of the color categorization model $\tilde{P}$, that in our recolorization experiments we assume to be known, Ref. [AM09]. Due to the color categorization model, now the

label palette is not fixed by the user, but automatically by $\tilde{P}$, and it includes only the basic color categories set by Berlin and Kay [BK69]. The solution found in step 2 gives the segmentation of color image in terms of basic colors. However, as specified in [AM09], we can learn different color categories and different number of them, that is $K \neq 11$. This will depends on the color composition of the image, or set of images, for the specific application. In the step 3 the *label palette* and the *colorization palette* are considered for applying colorization operator (Subsection 3.3). Like in the colorization approach (Subsection 3), the color palette is again established interactively by the user.

4.4. Image editing effects

The scheme explained in Subsection 3 can be very useful for other interactive image processing tasks, if some slight change is made. Now we explain the more general scheme, in which colorization is a particular case. Based on the scheme presented in Section 2, we have the following interactive general scheme:

1. Feature learning,
2. Image layering,
3. Layering process and Composition operator.

The first two stages are similar to the Section 3, and only the third stage is changed. The transferring color function is replaced by: *Layering process* and a more general operator, named here *Composition Operator*. The *Layering process* includes tonal transferring functions, artistic/geometric transformations, image filters, etc. Instances of such processing are blur or sharpen, or any other effect or combinations. The *Composition Operator* allows us to combine the information (processed layers) obtained in the previous step. This operator set could be applied at the same time or sequentially.

In this modality, the user widens his interaction options. As explained in Section 2, the interactive process of stage one consists of scribbling, labeled pixels, made by a user in different regions of interest. Actually, these labeled pixels are the training data of the (supervised) learning stage. The second stage is the same as in the Subsection 3. In the third stage an *operator set* $\mathcal{F} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_K\}$ is selected by the user. The triad operator $\mathcal{F}_k = [\mathcal{F}_{l,k}, \mathcal{F}_{c_1,k}, \mathcal{F}_{c_2,k}]^T$ acts over the layer k and over each channel, i.e. L, C_1 and C_2. Hence, similarly to colorization, each operator in the operator set must be linked to a color in *label palette*, see Fig. 4. Finally, the operator set $\mathcal{F}$ is applied on each detected layer,

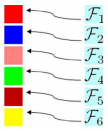

Figure 4: *To each label (left column) is assigned a composition operator (right column).*

and is combined using the following *composition operator*:

$$l(r) = \sum_{k=1}^{K} \hat{\alpha}_k(r) \mathcal{F}_{l,k}[\hat{l}](r), \tag{18}$$

$$c_1(r) = \sum_{k=1}^{K} \hat{\alpha}_k(r) \mathcal{F}_{c_1,k}[\hat{c}_1](r), \tag{19}$$

$$c_2(r) = \sum_{k=1}^{K} \hat{\alpha}_k(r) \mathcal{F}_{c_2,k}[\hat{c}_2](r). \tag{20}$$

We note again that $\mathcal{F}$ is a family of operators. That is, these operators can be functions that change the intensity or color of the image (tone or color transferring functions) similar to the ones used in Refs. [LFUS06, AP08], and also functions that change the geometry of the image (artistic or geometric transformations), see Ref. [Hol88]. Instances of such operators are:

1. The *identity operator*: $\mathcal{E}[f] = f$,
2. The *layer selection*: $\mathcal{S}_k[f] = \alpha_k f$,
3. The *multiple layer selection* : Suppose that the user select a set of layers $\mathcal{A} \subset \mathcal{K}$ then $\mathcal{S}_{\mathcal{A}}[f] = \sum_{k \in \mathcal{A}} \mathcal{S}_k[f]$,
4. The *colorization*, see Subsection 3,
5. The *recolorization*, see Subsection 4, and
6. The *matting*, see Refs. [LRAL08, RKB04, WC05],

among many others. We show some combinations of operators in the experiment section, Section 6, where characteristics of experiments are discussed, see Figs. 19, 20, 21. We remark that the layer operator $\mathcal{F}$ can be composed by a sequential application of basic operators, for instance a selection-colorization-bluring (see our experiments).

4.5. Shortest–Path Distances

Intensity (or color) distributions allow to segment images into classes where the pixels have similar intensity or color. Under this assumption, a class can be partitioned in different (disconnected) regions. However, there can be particular tasks where connectedness is an important feature to be taken into account. For instance the methods in [Gra06, YS06] segment images in regions with small geodesic distances. We can incorporate connectedness information into the QMMFs framework by defining the pixel likelihood that depends on some kind of distance between each pixel to the user scribbles.

In this work we incorporate the distance between pixels in form of a graph connectivity, *i.e.* the larger distance,

the smaller connectivity factor (and conversely) [YVW*05]. According to the QMMF model the connectivity factor is better expressed as a likelihood probability u. This can be computed, for instance, by means of a diffusion [Gra06], in particular, with the random walker formulation through the combinatorial Dirichlet problem:

$$X = \arg\min_{x} \sum_{\substack{<r,s>: \\ \mathcal{R}(r)=0, \\ s\in\mathcal{N}_r}} w_{rs}\|x_r - x_s\|^2, \tag{21}$$

with $X = [x_{rk}]_{r:\mathcal{R}(r)=0,\, k\in\mathcal{K}}$, $x_r \in \mathbb{R}^K$ and boundary conditions at the marked pixels. The probability of the class corresponding to marked pixels is fixed to 1 and for the other classes fixed to zero:

$$m_{rk} = \begin{cases} 1 & \mathcal{R}(r) = k, k > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Then the solution to the unmarked pixels is computed by solving the set of linear systems

$$LX = -B^T M$$

where $L = [w_{rs}]_{r,s:\mathcal{R}(r)=\mathcal{R}(s)=0}$, $B = [w_{rs}]_{r,s:\mathcal{R}(r)\neq 0,\, \mathcal{R}(s)=0}$ and $M = [m_{rk}]$, see [Gra06] for details. Then the connectedness likelihood is

$$u_k(r) = \begin{cases} m_{rk} & \mathcal{R}(r) > 0 \\ x_{rk} & \text{otherwise.} \end{cases} \tag{22}$$

Now we have 2 sources of information (likelihoods) u and v: u that encodes some kind of spatial distance and v the color (intensity) likelihood. Then, assuming independence between the information sources, the joint likelihood $\bar{v}$ is given by

$$\bar{v_k}(r) = u_k^a(r) v_k^{(1-a)}(r) \tag{23}$$

where the mix parameter (power) $0 \leq a \leq 1$ denotes our confidence, or reliability, on each source. Thus, the join likelihood can be computed in a preprocessing stage and being directly used in the QMMF framework by substituting $\mathbf{D}(r)$ by

$$\bar{\mathbf{D}}(r) = -\text{diag}\{\log \bar{v_1}(r), \log \bar{v_2}(r), \ldots, \log \bar{v_K}(r)\}$$

into the cost function (7).

5. Application: Video Colorization

As a demonstration of the subsection 4.5, we present an application for video colorization that incorporates two likelihood sources: shortest path distances and intensity similarity, *i.e.* we use the joint likelihood (23).

In the literature, one can find previous interactive works dealing with this problem [LLW04, SBv04, YS06]. The method presented in [YS06] is suitable for colorization of videos with fixed background (without camera movements) and small object displacements. The approach presented in [LLW04] uses implicitly optical flow what allows to colorize more complex videos: larger displacements and camera movements. This method does not propagate the scribbles, but calculates an overall video colorization by redefining neighboring pixels, establishing in this way a temporal coherency. That is, the pixels $r_0 = (x_0, y_0)$ and $r_1 = (x_1, y_1)$, in the frames t and $t+1$ respectively, are neighbors if:

$$\|(r_0 + d(r_0, t)) - r_1\| < T, \tag{24}$$

where T is a parameter, $d(r,t) = d(x,y,t) = (d_x(x,y,t), d_y(x,y,t))$ represents the optical flow computed by an standard method, in particular [LLW04] uses the Lucas and Kanade algorithm [LK81].

Figure 5: *Scribbles made on one frame of the video. This frame is available in a video at http://www.cs.huji.ac.il/~yweiss/Colorization/.*

Here we present a video colorization method that also incorporates an optical flow-based strategy. The basic idea consists of: given a frame $g(r,t)$ and its corresponding user scribbles $mask(r,t)$ (a *multimap*) then, we propagate the scribbles to the frame $t+1$ by warping the $mask(t)$ using the optical flow $d(r,t)$ to obtain the multimap at frame $t+1$. That is

$$mask(r, t+1) = mask(\lfloor r + d(r,t) + q \rfloor, t), \tag{25}$$

where $q = (0.5, 0.5)$, and $\lfloor \cdot \rfloor$ is the floor function, *i.e.* $\lfloor x \rfloor$ is the largest integer not greater than x. Then, we use the estimated (warped) *multimap* to colorize the $t+1$-frame. After a number of propagated *multimap* the user needs to make small rectifications to the multimap due to the error introduced by the optical flow propagation process or large scene changes. The algorithm continues up to a desired number of frames.

In Fig. 5 we show the multimap on the frame of the video used for the experiment, available at http://www.cs.huji.ac.il/~yweiss/Colorization/. Next panels show three mulitmaps estimated by our method based on the standard multigrid optical flow algorithm of the Horn and Schunck [HS80]. Fig. 6 depicts four colorized frames of the video. We note that our method relies strongly on the computed optical flow. It is implemented as a frame by frame colorization and does not include a temporal regularization. However, that temporal regularization can be introduced as in [LLW04].

Figure 6: *Four colorized frames representative of the full colorized video.*

6. Experiments and Results

In order to evaluate the performance of our proposal, several experiments were carried out. First, we describe the colorization process by using an experiment. Then, we compare the performance of the colorization method proposed by Levin et al. in [LLW04] and our method. Additionally, we show some other results of colorization and recolorization that demonstrate the method capabilities. And finally, we present some experiments of image editing.

Most of the images presented here were taken from Berkeley Image Database [MFTM01], available at http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/. First, they were converted into greyscale images and then the process of colorization was applied.

6.1. Colorization experiments

Figs. 7, 8, 9 and 10 compare our method with the one proposed by Levin et al. [LLW04] and Dalmau et al. [DRM07]. In the experiments of Figs. 7, 8 and 9 the *label palette* and *colorization palette* are the same, and the colors in both palettes are the ones used in the corresponding multimap. In general, the methods reported in [LLW04] and [DRM07] have good results in many situations. However Figs. 7 and 8 show that the Levin et al. method, code available at http://www.cs.huji.ac.il/~yweiss/Colorization/, obtains poor results when the image has disconnected regions that need to be colorized with the same color, or when the pixels to be colorized are far away from the user's scribbles. Both methods, the reported in [LLW04] and [DRM07], also work poorly when colorizing dark regions with light colors or when colorizing light regions with dark colors, see Fig. 9. In order to show the influence of the luminance component transformation in the recolorization context, in Fig. 10 we compare our proposal with the Dalmau et al. method. Based on the visual comparison between Fig. 10 (c) and Fig. 10(d) we conclude that the color RGB composition of the image obtained from the transformation luminance function, see Fig. 10 (d), is more similar to the colors fixed in the colorization palette than the image obtained without using the transformation luminance function, see Fig. 10 (c). In order to give a numerical argument of the above explanation, a test of the RGB composition of the red class, see Fig. 10 panels (a) and (b), was carried out. The RGB components used to recolorize this class were $[208, 54, 88]$, and the average RGB components, corresponding to the red class, obtained by Dalmau's and our method were $[153.4, 10.1, 32.7]$ and $[207.5, 62.2, 88.7]$ respectively.

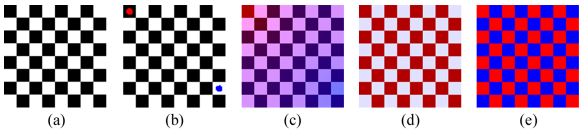

Figure 7: *Comparison between our method and the method reported in [LLW04] when colorizing pixels are located far away from the user's scribbles or when colorizing pixels are in disconnected regions. (a) Image, (b) scribbles, (c) colorization using the Levin et al. method, (d) colorization using the Dalmau et al. method, (e) colorization using our proposal.*

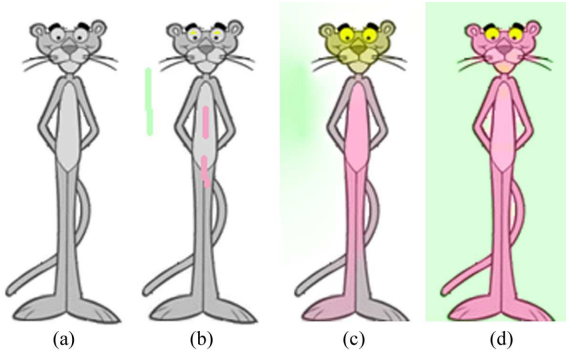

Figure 8: *Comparison between our method and the method reported in [LLW04]. (a) Original image, (b) multimap, (c) Levin et al. result, (d) our result.*

In Fig. 11 we show additional experiments that demonstrate the method capability. Moreover, once we have computed the probability measure field, α, we can reassign colors to some labels and colorize the image by just reapplying Eqs. (10)–(12). In Fig. 13 we present experimental results that illustrate the method flexibility by changing colors and keeping the memberships fixed.

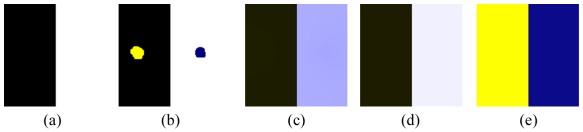

Figure 9: *Comparison between our method and the method reported in [LLW04] when colorizing dark regions with light colors, or when colorizing light regions with dark colors. (a) Image, (b) scribbles, (c) colorization using the Levin et al. method, (d) colorization using the Dalmau et al. method, (e) colorization using our proposal.*

Figure 10: *Comparison between our method and the method reported in [DRM07]. (a) label palette (left column) and colorization palette (right column), (b) multimap, (c) Dalmau et al. result, (d) our result.*

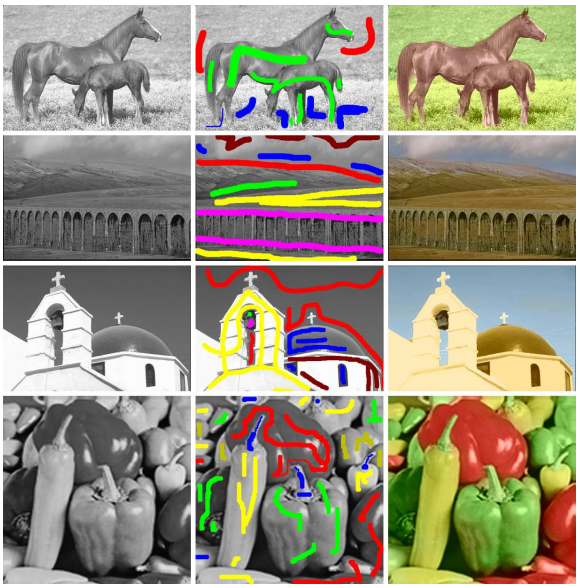

Figure 11: *Columns from left to right: Gray scale image, multimap and colored image.*

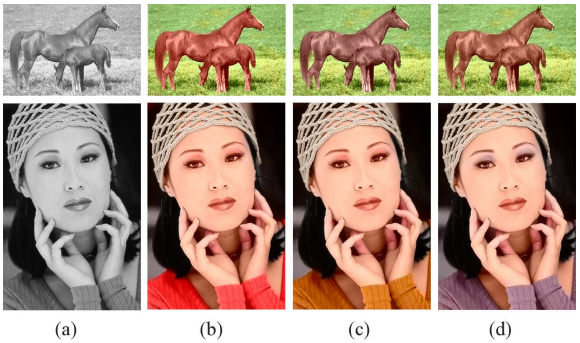

Figure 13: *Flexibility for assigning and reassigning colors to images. (a) Original image (b)-(d) colorizations.*

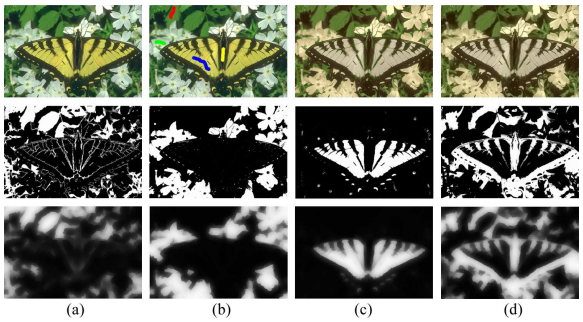

Figure 14: *Semi-automatic recolorization. The first row, from left to right, shows: (a) original image, (b) multimap, (c) recolorization by using likelihood, (d) recolorization by using segmentation. The second and third rows show the class memberships (α layers) corresponding the likelihood and the soft segmentation respectively.*

Experiments with different color models demonstrate that the final results do not depend on the chosen model but on the user ability for selecting the appropriate color palette, see Fig. 12.

6.2. Semi-automatic recolorization

Fig. 14 illustrates a semi-automatic recolorization. The first row, panels (a) and (b) show the original image and the multimap, respectively. Panels (c) and (d) show two recolorizations: panel (c) shows a recolorization that was reached using the likelihood (Eq. (3) and Section 4.2) and panel (d) illustrates a recolorization that was attained by using the QMMF based on the probabilistic segmentation, α. For comparisson purposes, in Fig. 15 we show details of the original image and the corresponding colorizations. Note that in likelihood based colorization some regions in the butterfly body are colorized in green because the color similarity of the butterfly body and the leaves. As we can see, from this experiment, the likelihood is not enough for obtaining good recolorizations and in general the regularization stage is needed.

Figure 12: *Colorization results using different color models: (a) Original image, (b) YUV, (c) LUV.*

The proposed scheme also admits a multimap with imprecise scribbles, see the left image in Fig. 16. This image shows a multimap with scribbles that overlap different regions (the horses and the grass). The second panel shows the recolorized image using this multimap. In this case, the vector field α is computed on the whole image, *i.e.* $\alpha(r)$ is also computed in the pixels marked by the user ($\mathcal{R}(r) \neq 0$), and

Figure 15: *Likelihood vs Segmentation. Zoom of sub-images taken from recolorization using, as a measure field, the likelihood and the segmentation respectively. (a) Subimage, (b) recolorization using likelihood, (c) recolorization using segmentation.*

Figure 16: *Recolorization with imprecise scribbles.*

the multimap is only used to compute the *likelihoods*. The third panel shows the authomaticaly refined multimap computed by keeping the resulted classification at those pixels whose entropy is small: $1-\sum_k \alpha_k^2(r) < 0.4$. The most right image in Fig. 16 depicts the recolorized image using the refined multimap.

In the next experiment we assume that color distributions do not change throughout the video sequence. Therefore, the method only needs one multimap, i.e. the user scribbles only on the first frame of the sequence. Left panel in Fig. 17 exhibits the scribbles for three classes provided by the user (multimap). The right panel shows the multimap refined similarly to the previous experiment. First row in Fig. 18 shows demostrative frames from the original video and the semiautomatic recolorization is shown in second row. We remark that about 300 frames were colorized using the distribution estimated from the first frame, the video duration is approximately of 10 seconds.

6.3. Quasi-automatic recolorization

The follow experiments illustrate the quasi-automatic recolorization presented in subsection 4.3. In this case, we

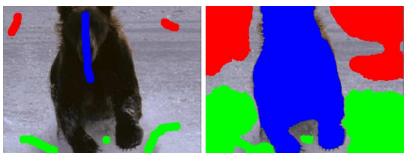

Figure 17: *Left Image, scribble on the first video frame of the video in first row of Fig. 18. Right image, automatic generated multimap from the user scribbles, see text.*

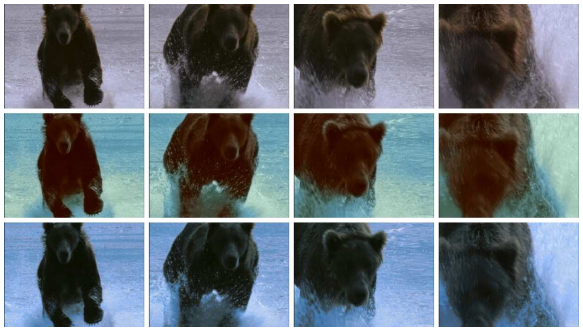

Figure 18: *First row show 4 frames of a video sequence of approximately 10 seconds (about 300 frames). Second row shows the semiautomatic recolorization (the required user scribbles are presented in Fig. 17). Third row shows the quasi-automatic recolorization. The difference in colorization corresponds to different selected colorization palettes.*

Figure 19: *(a) Original image, (b) mask obtained from vector measure field, (c) extracted colorized object (d) photomontage using the mask in panel (c).*

use fixed likelihood density functions that correspond to the color categorization model $\tilde{P}$ from Ref. [AM09], with $K = 11$. As the *label palette* is fixed, the only user interaction is the linkage between the *label palette* and the *colorization palette*. Except for this small interaction, the procedure is fully automatic.

Results of a video colorization experiment are shown in the third row of Fig. 18. The recolorized video frames agree with the original frames in first row. We note that the difference between recolorizations (second and third row) is due to different *colorization palettes* used. Important to remark it is that both strategies preserve transparency between the bear and the water.

6.4. Other image editing experiments

In this Subsection we illustrate the capability of our method by some editing examples. As our method allows to divide the image in layers, we can select some particular layers for segmenting and colorizing a particular object, see Fig. 19. More formally, let $\mathcal{A} \subset \mathcal{K}$ be a layer set selected by a user. Then, the *mask* in Fig. 19 (b) can be obtained by

$$mask(r) \;=\; \sum_{k\in\mathcal{A}} \hat{\alpha}_k(r). \qquad (26)$$

With the mask, Fig. 19 (b), we can now extract the colorized image Fig. 19 (c), or simply we can make a blending (image

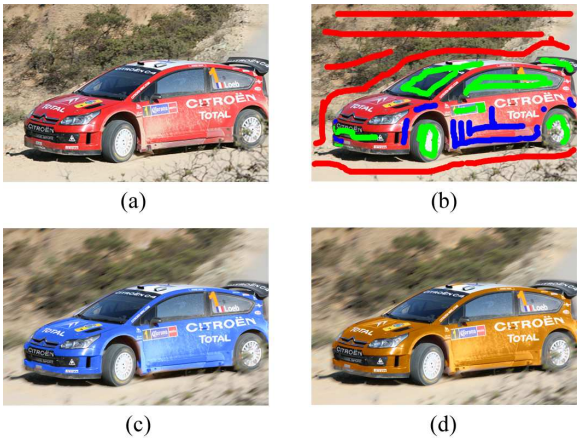

Figure 20: *Combining different operators: blur (red class), identity (green class) and recolorization (blue class). (a) Original color image, (b) multimap, (c)-(d) multioperator composition.*

Figure 21: *Artistic effects. Source images (first two images at the left) and artistic effects using a combination of the processed image sources (last two images at the right), see text.*

composition) with the colorized image and another image, see Fig. 19 (d).

It is also possible to combine several operators (Subsection 4.4). In Fig. 20 three operators were used: directional blur (for the background), recolorization (for some red parts in the car) and identity (for the rest of the car). The goal is to introduce motion effect and recolorize part of the car. We remark that the applied colorization method is not a simple mapping between colors. It can be seen that while some of the car's red regions have been changed, others remain unchanged, such as the flag on the window and the logo on the door, see Fig. 20 (c)-(d).

Morover, the interactive general scheme accepts more than one source image. Figs. 21 show two source images. Thus, given the multimap provided by interaction (illustrated on the left image) we can obtain the composed image in the third panel, from left to right, using a pond ripple effect to the source image 1 weighed by the layer that corresponds to the red class and the source image 2 weighed by the layer that corresponds to the green class (third image from the left). Moreover, the composed image at the right can be computed by using a circular ripple effect to the source image 2 weighed by the layer that corresponds to the green class and the source image 1 weighed by the layer that corresponds to the red class.

Regarding the computational cost, the *Image Layering* is the most time consuming step in our proposal. Our method computes the layering of an image by solving a convex quadratic programming problem. This minimization corresponds to solving a positive definite and symmetric linear system. The length of the linear system depends on the size of the image and the number of classes. We implement our approach using a Gauss-Seidel scheme in Matlab with .m and .mex files. Although it is not the most efficient implementation, a 480x640 color image can be segmented in about a second per class on an Intel 2.5Mhz iMac. Our algorithm can be implemented using Multigrid Gauss-Seidel. Moreover, it can be parallelized in blocks (for a multicore CPU) or in cellular automata (for a GPU based implementation, CUDA or OpenCL). This is beyond the scope of the current work but we have a CUDA based implementation for binary segmentation at a rate of 60 frames per second on an NVIDIA 8800 GT. Our implementation is based on the QMMF algorithm reported in [RD09].

7. Conclusions

We have presented a three stage interactive image and video editing procedure. The first step consists of scribbles made by a user over the image. The second step consists of computing a probabilistic segmentation and in the third step the color or effect properties are specified.

The proposed interactive editing method is based on the multiclass probabilistic image segmentation algorithm. The segmentation process consists of the minimization of a linearly constrained positive definite quadratic cost function and thus the convergence to the global minima is guaranteed. We associate an image transformation (colorization, recolorization, directional blur, edge enhancement, etc) to each class and then the pixel color components are a linear combination of a set of operators applied over regions defined by the classes. The color component values depend on the computed probability of each pixel belonging to the respective class.

We have demonstrated the flexibility of the presented scheme by implementing different image and video editing applications, as for instance: colorization, recolorization, blending, blur and combinations. Our method can successfully be used for applying particular transformations to isolated objects. It also accepts different source images. As result, complex artistic effects are obtained.

A quasi-automatic recolorization version is presented that could be very useful in video color restoration and video recolorization.

We have extended the QMMF models to accept mixture of

likelihoods. New likelihoods can effectively codify the segmentation of other methods. In particular, we investigate the pixel connectivity through the shortest–path distance. The shortest–path based likelihood was computed by the random walker segmentation method. We included a video colorization application that relies on this approach.

Acknowledges. This research was supported in part by CONACYT (grant 61367) and PROMEP (grant 103.5/08/2919). O. Dalmau was also supported in part by a PhD scholarship from CONACYT, Mexico. The authors thank the anonymous reviewers for their advice and comments that helped to improve the quality of the manuscript

References

[AM09] ALARCÓN T. E., MARRQUÍN J. L.: Linguistic color image segmentation using a hierarchical Bayesian approach. *Color Res Appl 34* (August 2009), 299–309.

[AP08] AN X., PELLACINI F.: Appprop: all-pairs appearance-space edit propagation. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers* (New York, NY, USA, 2008), ACM, pp. 1–9.

[BK69] BERLIN B., KAY P.: *Basic Color Terms: Their universality and Evolution*. Berkeley: University of California, 1969.

[BR03] BLASI D., RECUPERO R.: Fast Colorization of Gray Images. In *Eurographics Italian Chapter 2003* (2003).

[CWSM04] CHEN T., WANG Y., SCHILLINGS V., MEINEL C.: Grayscale Image Matting and Colorization. In *In Proceedings of Asian Conference on Computer Vision (ACCV 2004)* (Jan. 27–30, 2004), pp. 1164–1169.

[DHS01] DUDA R. O., HART P. E., STORK D. G.: *Pattern Classification*, second ed. John Wiley & Sons, Inc., New York, 2001, pp. 164–165.

[DRM07] DALMAU O., RIVERA M., MAYORGA P. P.: Computing the alpha-channel with Probabilistic Segmentation for Image Colorization. In *IEEE Proc. Workshop in Interactive Computer Vision (ICV'07)* (2007), pp. 1–7.

[GG84] GEMAN S., GEMAN D.: Stochastic relaxation, Gibbs distributions and Bayesian restoration of images. *IEEE PAMI 6* (1984), 721–741.

[GL00] GUO P., LYU M. R.: A Study on Color Space Selection for Determining Image Segmentation Region Number. In *Proc. of the 2000 International Conference on Artificial Intelligence (IC-AI'2000), Monte Carlo Resort, Las Vegas, Nevada, USA* (2000), vol. 3, pp. 1127–1132.

[Gra06] GRADY L.: Random walks for image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence 28*, 11 (Nov. 2006), 1768–1783.

[GW02] GONZALEZ R. C., WOODS R. E.: *Digital Image Processing*, second ed. Prentice Hall, Upper Saddle River, New Jersey, USA, 2002, pp. 76–107.

[GWE04] GONZALEZ R. C., WOODS R. E., EDDINS S. L.: *Digital Image Processing using Matlab*. Prentice Hall, Upper Saddle River, New Jersey, USA, 2004, pp. 204–207.

[Hol88] HOLZMANN G. J.: *Beyond Photography - The Digital Darkroom*. Prentice Hall, Englewood Cliffs, NJ, 07632, 1988, pp. 109–114.

[Hor02] HORIUCHI T.: Estimation of color for gray-level image by probabilistic relaxation. In *Proc. IEEE Int. Conf. Pattern Recognition* (2002), pp. 867Ű–870.

[HS80] HORN B. K., SCHUNCK B. G.: *Determining Optical Flow*. Tech. rep., Cambridge, MA, USA, 1980.

[HTF09] HASTIE T., TIBSHIRANI R., FRIEDMAN J.: *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, second ed. Springer Science+Business Media, 233 Spring Street, New York, NY 10013, USA, 2009, pp. 208–209.

[ICOL05] IRONY R., COHEN-OR D., LISCHINSKI D.: Colorization by Example. In *Eurographics Symposium on Rendering 2005 (EGSR'05)* (2005), pp. 201–210.

[KV06] KONUSHI V., VEZHNEVETS V.: Interactive Image Colorization and Recoloring based on Coupled Map Lattices. In *Graphicon'2006 conference proceedings, Novosibirsk Akademgorodok, Russia* (2006), pp. 231–234.

[LFUS06] LISCHINSKI D., FARBMAN Z., UYTTENDAELE M., SZELISKI R.: Interactive local adjustment of tonal values. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers* (New York, NY, USA, 2006), ACM, pp. 646–653.

[Li01] LI S. Z.: *Markov Random Field Modeling in Image Analysis*. Springer-Verlag, Tokyo, 2001, pp. 11–15.

[LK81] LUCAS B. D., KANADE T.: An iterative image registration technique with an application to stereo vision (ijcai). In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)* (April 1981), pp. 674–679.

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization using Optimization. *ACM Transactios on Graphics 23*, 3 (2004), 289–694.

[LRAL08] LEVIN A., RAV-ACHA A., LISCHINSKI D.: Spectral Matting. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*, 10 (2008), 1699–1712.

[MAB03] MARROQUIN J. L., ARCE E., BOTELLO S.: Hidden Markov measure field models for image segmentation. *IEEE PAMI 25* (2003), 1380–1387.

[McA04] MCANDREW A.: *Introduction to Digital Image Processing with Matlab*. Thomson Course Technology, 2004, pp. 87–139, 449–466.

[MFTM01] MARTIN D., FOWLKES C., TAL D., MALIK J.: A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In *Proc. 8th Int'l Conf. Computer Vision* (2001), vol. 2, pp. 416–423.

[MVRN01] MARROQUIN J. L., VELAZCO F., RIVERA M., NAKAMURA M.: Gauss-Markov Measure Field Models for Low-Level Vision. *IEEE PAMI 23* (2001).

[OKS80] OTHA Y., KANADE T., SAKAI T.: Color information for region segmentation. *Comput. Graphics Image Processing 13* (1980), 22–241.

[PD84] PORTER T., DUFF T.: Compositing Digital Images. *Computer Graphics 18*, 3 (1984), 253–259.

[QG05] QIU G., GUAN J.: Color by linear neighborhood embedding. In *IEEE International Conference on Image Processing (ICIP'05)* (Sept. 11–14, 2005), pp. III – 988–91.

[QWCO*07] QING L., WEN F., COHEN-OR D., LIANG L., XU Y. Q., SHUM H.: Natural Image Colorization. In *Rendering Techniques 2007 (Proceedings Eurographics Symposium on Rendering)* (2007), Eurographics.

[RAGS01] REINHARD E., ASHIKHMIN M., GOOCH B., SHIRLEY P.: Color transfer between images. *IEEE Computer Graphics and Applications 21*, 5 (2001), 34–41.

[RCC98] RUDERMAN D. L., CRONIN T. W., CHIAO C. C.: Statistics of cone responses to natural images: Implications for

visual coding. *J. Optical Soc. of America A 15*, 8 (1998), 2036–2045.

[RD09] RIVERA M., DALMAU O.: Quadratic Programming for Probabilistic Image Segmentation. *submitted to IEEE Transactions on Image Processing* (2009).

[RDT08] RIVERA M., DALMAU O., TAGO J.: Image segmentation by convex quadratic programming. In *ICPR* (2008), pp. 1–5.

[RKB04] ROTHER C., KOLMOGOROV V., BLAKE A.: "Grabcut": Interactive Foreground Extraction using Iterated Graph Cuts. 309–314.

[ROM05] RIVERA M., OCEGUEDA O., MARROQUIN J. L.: Entropy Controlled Gauss-Markov Random Measure Fields for Early Vision. In *VLSM, LNCS 3752* (2005), pp. 137–148.

[ROM07] RIVERA M., OCEGUEDA O., MARROQUÍN J. L.: Entropy-Controlled Quadratic Markov Measure Field Models for Efficient Image Segmentation. *IEEE Transactions on Image Processing 16*, 12 (2007), 3047–3057.

[SBv04] SÝKORA D., BURIÁNEK J., ŽÁRA J.: Unsupervised Colorization of Black and White Cartoons. In *NPAR '04: Proceedings of the 3rd international symposium on Non-photorealistic animation and rendering* (New York, NY, USA, 2004), ACM, pp. 121–127.

[TJT05] TAI Y. W., JIA J., TANG C. K.: Local Color Transfer via Probabilistic Segmentation by Expectation-Maximization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (2005), vol. 1, pp. 747–754.

[WAM] WELSH T., ASHIKHMIN M., MUELLER K.: Transferring color to greyscale images.

[WC05] WANG J., COHEN M.: An interactive optimization approach for unified image segmentation and matting. In *ICCV* (2005), vol. 2, pp. 936–943.

[WG04] WON C. S., GRAY R. M.: *Stochastic Image Processing*. Kluwer Academic/Plenum, New York, 2004, pp. 11–21.

[WH04] WANG C. M., HUANG Y. H.: A Novel Color Transfer Algorithm for Image Sequences. *Journal of Information Science and Engineering 20*, 6 (2004), 1039–1056.

[WOZ02] WANG Y., OSTERMANN J., ZHANG Y.-Q.: *Video Processing and Communications*. Prentice Hall, 2002, pp. 18–19.

[WS82] WYSZECKI G., STILES W.: *Color Science: Concepts and methods, quantitative data and formulae*, second ed. Wiley, 1982, pp. 165–168.

[YK03] YANG C. C., KWOK S. H.: Efficient gamut clipping for color image processing using LHS and YIQ. *Opt. Eng. 42* (March 2003), 701–711.

[YS06] YATZIV L., SAPIRO G.: Fast Image and Video Colorization Using Chrominance Blending. *Image Processing, IEEE Transactions on 15*, 5 (2006), 1120–1129.

[YVW*05] YEN L., VANVYVE D., WOUTERS F., FOUSS F., VERLEYSEN M., SAERENS M.: Clustering using a random walk based distance measure. In *ESANN 2005: European Symposium on Artificial Neural Networks* (2005), pp. 317–324.